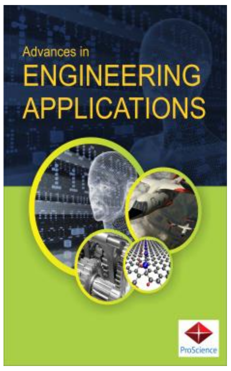
Advances in Engineering Applications

Journal Home Page: www.proscience-journals.com



An Augmented Dynamic Round Robin CPU Scheduling Algorithm

A.S.V. Balakrishna[1] N Srinivasu[1]*,

[1]Department of Computer Science and Engineering, K L University, Guntur, India

ARTICLE INFO: Received 4 July 2018, Accepted 11 July 2018

ABSTRACT

The Round Robin (RR) CPU scheduling algorithm is a fair scheduling algorithm that gives equal time quantum to all processes. The choice of the time quantum is critical as it affects the algorithm's performance. This paper proposes a new algorithm that enhanced on the Improved Round Robin CPU (IRR) scheduling algorithm. The proposed algorithm was implemented and benchmarked against basic round robin algorithms available in the literature. The proposed algorithm compared with the basic round robin algorithms, produces minimal average waiting time (AWT), average turnaround time (ATAT), and number of context switches (NCS). From the obtained results we observed that proposed algorithm met better scheduling criterion than the basic round robin scheduling algorithm.

Keywords: Scheduling Criteria, Scheduling algorithms, Round Robin, Time quantum, waiting time, context switches.

1. INTRODUCTION

Multiprogramming is one of the most important aspects of operating systems. Multiprogramming became possible when disks were introduced to the computing world. The concept of multiprogramming relies on the capability of a computer to store instructions for long-term use. The goal is to reduce CPU idle time by allowing new jobs to take over the CPU whenever the currently running job needed to wait.

*Corresponding author: Professor N. Srinivasu
Email srinivasu28@kluniversity.in

Process scheduling is an essential part of a Multiprogramming operating system.

CPU scheduling algorithms are used for better utilization of CPU. Main objective is increasing system performance in accordance with the chosen set of criteria. It is the change of ready state to running state of the process. CPU scheduler selects from among the processes that are ready to execute and allocates the CPU to one of them. CPU scheduling is the basis of multiprogramming systems. Maximum CPU utilization obtained with multiprogramming.

CPU scheduling more complex when multiple CPUs are available.

CPU scheduling algorithms CPU scheduling algorithms are used to allocate the CPU to the processes waiting in the ready queue. Some of the popular CPU scheduling algorithms are First Come First Serve (FCFS), Shortest Job First (SJF), Priority Scheduling (PS) and Round Robin (RR).The FCFS is the most intuitive and simplest technique is to allow the first process submitted to run first. In effect, processes are inserted into the tail (rear) of a queue when they are submitted. The next process is taken from the head (front) of the queue when each finishes running. The average waiting time in FCFS is quite long. Shortest-Job-First (SJF) is a non-preemptive discipline in which waiting job with the smallest estimated run-time-to-completion is run next. In other words, when CPU is available, it is assigned to the process that has smallest next CPU burst. The SJF scheduling is especially appropriate for batch jobs for which the run times are known in advance. Since the SJF scheduling algorithm gives the minimum average time for a given set of processes, it is probably optimal.

The SJF algorithm favors short jobs at the expense of longer ones. The obvious problem with SJF scheme is that it requires precise knowledge of how long a job or process will run, and this information is not usually available. The basic idea of Priority Scheduling is straightforward: each process is assigned a priority, and priority is allowed to run. Equal-Priority processes are scheduled in FCFS order. The shortest-Job-First (SJF) algorithm is a special case of general priority scheduling algorithm. Priority scheduling can be either pre-emptive or non preemptive. Round robin scheduling is a preemptive version of first-come, first-served scheduling. Each process gets a small unit of CPU time usually 10-100 milliseconds. After this time has elapsed, the process is preempted and added to the end of the ready queue. This time interval is known as a time-slice or quantum. If a process does not complete or get blocked because of an I/O operation within the time slice, the time slice expires and the process is preempted. This preempted process is placed at the back of the run queue where it must wait for all the processes that were already in the queue to cycle through the CPU. if a process gets blocked due to an I/O operation before its time slice expires, it is, of course, enters a blocked because of that I/O operation. Once that operation completes, it is placed on the end of the run queue and waits its turn.

A big advantage of round robin scheduling over non-pre-emptive schedulers is that it dramatically improves average response times and starvation free. By limiting each task to a certain amount of time, the operating system can ensure that it can cycle through all ready tasks, giving each one a chance to run. With round robin scheduling, interactive performance depends on the length of the quantum and the number of processes in the run queue. Round robin techniques is one of the most popular scheduling algorithms but the problem with that is more context switches, more waiting time. Round Robin scheduling is an older method of CPU time sharing, proposed ADDR algorithm meet better scheduling criterion than the basic round robin scheduling algorithm.

Scheduling Criteria There are many different CPU scheduling algorithms. The choice of a particular algorithm may favor one class of processes over another. There are several different criteria to consider when trying to select the "best" scheduling algorithm for a particular situation and environment, including: Context Switch: This is the process of storing and restoring context (state) of a preempted process, so that execution can be resumed from same point at a later time.

Throughput: This is the number of processes completed per unit time. May range from 10 / second to 1 / hour depending on the specific processes. Context switching and throughput are inversely proportional to each other.

CPU Utilization: This is a measure of how much busy the CPU is. CPU Utilization my range from 0 to 100 percent. The aim is to maximize the CPU utilization. Turnaround Time: Time required for a particular process to complete, from submission time to completion. The turnaround Time generally limited by the speed of the output device.

Waiting Time: How much time processes spend in the ready queue waiting their turn to get on the CPU. The CPU scheduling algorithm does not affect the amount of time during which a process executes or does input-output but it affects the amount of time that a process spends waiting in ready queue. Response Time: It is approximately the time of submission of a process until its first access to the CPU. The response time should be low for best scheduling. Fairness - Equal CPU time to each thread.

So, a good scheduling algorithm should possess the following characteristics: Context switches should be minimum. CPU utilization should be maximum, Throughput should be maximum. Turnaround time should be minimum, waiting time should be minimum. Response time should be minimum, Fairness in resource allocation is retained.

There are too many disadvantages in various CPU scheduling algorithms have so they are rarely used in timesharing and real time operating systems except for RR scheduling which is considered the most widely used CPU scheduling algorithms, but there are also some disadvantages of round robin CPU scheduling algorithm for operating system which are as follows: Larger waiting time and response time: it is a big drawback in round robin scheduling algorithm as it leads to degradation of system performance. Context Switches and Low throughput: Context switch leads to the wastage of time, memory and leads to scheduler overhead. If the number of context switches is low then the throughput will be high and some of the other disadvantages are also there.

With these observations it is found that the existing simple round robin architecture is not suitable for real time systems. So, its drawbacks are laminated in An Augmented Dynamic Round Robin CPU Scheduling Algorithm, it is modified and advanced version of round robin described in the next section.

The performance of RR scheduling is sensitive to time quantum selection, because if time quantum is very large then RR will be the same as the FCFS scheduling. If the time quantum is extremely too small then RR will be the same as Processor Sharing algorithm and number of context switches will be very high. Each value of time quantum will lead to a specific performance and will affect the algorithm's efficiency by affecting the processes waiting time, turnaround time, response time and number of context switches.

Various modifications to Round Robin CPU scheduling algorithm have been proposed by several authors. These modifications can be classified as follows:

Manish Kumar Mishra [22] this paper presents an improved Round Robin CPU scheduling algorithm coined enhancing CPU performance using the features of Shortest Job First and Round Robin scheduling with varying time quantum. The proposed algorithm is experimentally proven better than conventional RR. The simulation results show that the waiting time and turnaround time have been reduced in the proposed algorithm compared to traditional RR.

Ali Jbaeer Dawood [10], in this paper, the TQ studied to improve the efficiency of RR and performs the degrades with respect to Context Switching (CS), Average Wait Time (AWT) and Average Turned around Time (ATAT) that an overhead on the system. Thus, the new approach was proposed to calculate the TQ, known as Ascending Quantum and Minimum Maximum Round Robin (AQMMRR). The processes were ascending with shortest remaining burst time and calculate the TQ from multiply the summation of minimum and maximum BT by (80) percentage. The experimental result shows that AQMMRR performs better than RR and comparing with other two related works.

Manish Kumar Mishra,Abdul kadir [14] this algorithm is simple to implement, but it generally does not provide the fastest service. Round Robin being the most popular choice in time shared system, but it may not be suitable for real time systems because of larger waiting time, turnaround time and more number of context switches. This paper describes an improvement in RR. A simulator program has been designed and tested the Improved Round Robin (IRR). After improvement in RR it has been found that the waiting time and turnaround time have been reduced drastically.

Ishwari and Deepa [8] .The proposed algorithm improves all the drawbacks of round robin C P U scheduling algorithm. The paper also presents the comparative analysis of proposed algorithm with existing round robin scheduling algorithm on the basis of varying time quantum, average waiting time, average turnaround time and number of context switches.

Manish and AbdulKadir [9] In Round Robin CPU scheduling, performance of the system depends on the choice of the optimal time quantum. This paper presents an improved Round Robin CPU scheduling algorithm coined enhancing CPU performance using the features of Shortest Job First and Round Robin scheduling with varying time quantum. The proposed algorithm is experimentally proven better than conventional RR. The simulation results show that the waiting time and turnaround time have been reduced in the proposed algorithm compared to traditional RR.

Soraj and Roy [12] this paper gives the better result comparison to Simple Round Robin Scheduling Algorithm and solves the problem of shortest job first scheduling algorithm which is starvation. This approach also solves the problem higher average waiting time of first come first serve scheduling algorithm. The result performance I had shown in the performance chart above. The performance in the reference of average waiting time, turnaround time and context switches. For the future perspective the research should be useful with the knowing of arrival time with burst time and you can analysis this research for improvement. Debashree Nayak, Sanjeev Kumar Malla [13]

In this paper, we have proposed a new variant of RR scheduling algorithm known as Improved Round Robin (IRR) Scheduling algorithm, by arranging the processes according to their shortest burst time and assigning each of them with an optimal time quantum which is able to reduce all the above said disadvantages. Experimentally we have shown that our proposed algorithm performs better than the RR algorithm, by reducing context switching, average waiting and average turnaround time. In this paper, an algorithm is proposed that determines the time quantum dynamically, by taking the available burst time of processes in the system. This algorithm together with RR are implemented and their results were compared based on average waiting time, average turnaround time, average response time and number of context switches. Results of the analyses show that the proposed algorithm is promising as it outperforms other algorithms with respect to the average waiting time, average turnaround time and number of context switches scheduling criteria

2. Methods

2.1 THE PROPOSED ALGORITHM

RELATED WORK In the recent years, a number of CPU scheduling mechanisms have been developed for predictable allocation of processor. An Augmented Dynamic Round Robin Scheduling Algorithm for CPU Scheduling allocates the time quantum to all the process

2.2. ADRR CPU SCHEDULING ALGORITHM

An Augmented Dynamic Round Robin (ADRR) CPU scheduling algorithm works similar to Round Robin (RR) with an improvement. ADRR picks the first process from the ready queue and allocate the CPU to it for a time interval of up to 1 time quantum. After completion of process's time quantum, it checks the remaining CPU burst time of the currently running process. If the remaining CPU burst time of the currently running process is less than equal half of the time quantum, the CPU again allocated to the currently running process for remaining CPU burst time. In this case this process will finish execution and it will be removed from the ready queue. The scheduler then proceeds to the next process in the ready queue. Otherwise, if the remaining CPU burst time of the currently running process is longer than 1/2 time quantum, the process will be put at the tail of the ready queue. The CPU scheduler will then select the next process in the ready queue.

Following is the proposed ADRR CPU scheduling algorithm

```
Step 1. START
Step 2. Make a ready queue of the Processes say REQUEST.
Step 3. Do steps 4, 5 and 6 WHILE queue REQUEST becomes empty.
Step 4. Pick the first process from the ready queue and allocate the CPU to it for a time
        interval of up to 1 time quantum.
Step 5. If the remaining CPU burst time of the currently running process is less than equal to
        the one time quantum then allocate CPU again to the currently running process for
        remaining CPU burst time. After completion of execution, removed it from the ready
        queue and go to step 3.
Step 6. Remove the currently running process from the ready queue REQUEST and put it at
        the tail of the ready queue.
Step 7. END
```

2.3 Simulation

RR, ADRR algorithm were simulated and their performance on four performance criteria: AWT, ATAT, ART and NCS were observed. The simulations were carried out in a single processor environment with only CPU bound and no I/O bound processes. The system was assumed to have no context switching cost.

2.4 Experiments Performed

For performance evaluation of our proposed ADRR algorithm, we have taken two different cases. In first case arrival time has been considered zero and CPU burst time has been taken in increasing, decreasing and random orders. In second case arrival time has been considered non zero and CPU burst time has been taken in increasing, decreasing and random orders.

3 Results and Discussion

3.1 Illustrative Example

CASE 1 - Zero Arrival Time: In this case arrival time has been considered zero and CPU burst time has been taken in increasing, decreasing and random orders. Time quantum is 10 milliseconds.

CPU Burst Time in Increasing Order: We consider the ready queue with five processes P1, P2, P3, P4 and P5 arriving at time 0 with burst time 5, 12, 20, 26 and 34 respectively. The comparison result of RR and proposed ADRR are shown in Gantt chart representation (Table 1, 2, 3, 4 of RR and ADRR respectively.

Table 1 Gantt chart representation of ADRR with random tq=10 ms.

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

Table 2 Gantt chart representation of ADRR with tq= Avg. of all process Burst time

P1	P2	P3	P3	P4	P4	P5
0	5	17	36	37	56	63 97

Table 3 Gantt chart representation of ADRR with tq= (min. Burst time + max. Burst time)/2

P1	P2	P3	P4	P4	P5
0	5	17	37	57	63 97

Table 4 Comparison of RR and ADRR

Algorithm	Average Waiting Time(ms)	Average Turnaround Time(ms)	No.of Context Switch
RR	38.4	57.8	11
ADRR	27.2	42.6	6
ADRR with TQ=Avg. Bt.	24.4	43.8	4
ADRR with TQ= (min. BT. + max. BT.)/2	24.4	43.8	4

CPU Burst Time in Decreasing Order*:* We consider the ready queue with five processes P1, P2, P3, P4 and P5 arriving at time 0 with burst time 34, 26, 20, 12 and 5 respectively. The comparison result of RR and proposed ADRR are shown in the Gantt chart representation (Table5, 6, 7) of RR and ADRR (Table 8) respectively

Table 5. Gantt chart representation of ADRR with random tq=10 ms.

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

Table 6. Gantt chart representation of ADRR with tq= Avg. of all process Burst time

P1	P1	P2	P3	P4	P5
0	19	34	60	80	92 97

Table 7. Gantt chart representation of ADRR with tq= (min. Burst time + max. Burst time)/2

P1	P1	P2	P2	P3	P4	P5
0	20	34	54	60	80	92 97

Table 8. Comparison of RR and ADRR

Algorithm	**Average Waiting Time(ms)**	**Average Turnaround Time(ms)**	**No.of Context Switch**
RR	58	77.4	11
ADRR	46.4	59.8	7
ADRR with TQ=Avg. Bt.	53.2	72.6	4
ADRR with TQ= (min. BT. + max. BT.)/2	53.2	72.6	4

CPU Burst Time in Random Order: We consider the ready queue with five processes P1, P2, P3, P4 and P5 arriving at time 0 with burst time 20, 34, 5, 12 and 26 respectively. The comparison result of RR and proposed ADRR are shown in the Gantt chart (Table 9, 10, 11) representation of RR and ADRR (Table 12.) respectively

Table 9. Gantt chart representation of ADRR with random tq=10 ms.

P1	P1	P2	P3	P4	P4	P5	P2	P5	P5	P2
0	10	20	30	35	45	47	57	67	77	83 97

Table 10. Gantt chart representation of ADRR with tq= Avg. of all process Burst time

P1	P1	P2	P2	P3	P4	P5
0	19	20	39	54	59	77 97

Table 11.Gantt chart representation of ADRR with tq= (min. Burst time + max. Burst time)/2

P1	P2	P2	P3	P4	P5
0	20	40	54	59	71 97

Table 12. Comparison of RR and ADRR

Algorithm	Average Waiting Time(ms)	Average turnaround Time(ms)	No. of Context Switch
RR	48	67.4	11
ADRR	37	50.4	7
ADRR with TQ=Avg. Bt.	40.8	60.2	4
ADRR with TQ= (min. BT. + max. BT.)/2	40.8	60.2	4

3.2 CASE 2 – Non-Zero Arrival Time:

In this case arrival time has been considered non-zero and CPU burst time has been taken in increasing, decreasing and random orders. Time quantum is 10 milliseconds.

CPU Burst Time in Increasing Order: We consider the ready queue with five processes P1, P2, P3, P4 and P5 arriving at time 0, 4, 10, 15 and 17 with burst time 5, 12, 20, 26 and 34 respectively. The comparison result of RR and proposed ADRR are shown in the Gantt chart (Table 13) representation of RR and ADRR respectively.

Table 13. Comparison of RR and ADRR

Algorithm	Average Waiting Time(ms)	Average Turnaround Time(ms)	No.of Context Switch
RR	48.8	68.2	11
ADRR	17.2	32.6	6
ADRR with TQ=Avg. Bt.	15.2	34.6	4
ADRR with TQ= (min. BT. + max. BT.)/2	15.2	34.6	4

CPU Burst Time in Decreasing Order: We consider the ready queue with five processes P1, P2, P3, P4 and P5 arriving at time 0, 4, 10, 15 and 17 with burst time 5, 12, 20, 26 and 34 respectively. The comparison result of RR and proposed ADRR are shown in the Gantt chart representation of RR and ADRR (Table 14) respectively.

Table 14 Comparison of RR and ADRR

Algorithm	Average Waiting Time(ms)	Average Turnaround Time(ms)	No.of Context Switch
RR	29.2	48.6	11
ADRR	37.2	50.6	7
ADRR with TQ=Avg. Bt.	44	64.4	4
ADRR with TQ= (min. BT. + max. BT.)/2	53.2	72.6	4

CPU Burst Time in Random Order: We consider the ready queue with five processes P1, P2, P3, P4 and P5 arriving at time 0, 4, 10, 15 and 17 with burst time 5, 12, 20, 26 and 34 respectively. The comparison result of RR and proposed ADRR are shown in the Gantt chart representation of RR and ADRR (Table15) respectively.

Table 15. Comparison of RR and ADRR

Algorithm	Average Waiting Time(ms)	Avg. Turnaround Time(ms)	Number of Context Switch
RR	38.8	58.2	11
ADRR	27.8	41.2	7
ADRR with TQ=Avg. Bt.	31	51	4
ADRR with TQ= (min. BT. + max. BT.)/2	31	51	4

Table 16. Comparison with various Dynamic quantum values (if reaming Burst Time <= 1 Time Quantum)

Algorithm Type	Without Arrival time									With Arrival time								
	AVG .TAT			AVG. WT			NCS			AVG. TAT			AVG. WT			NCS		
	I	D	R	I	D	R	I	D	R	I	D	R	I	D	R	I	D	R
Basic RR	57.8	77.4	67.4	38.4	58	48	11	11	11	48.6	68.2	58.2	29.2	48.8	38.8	11	11	11
ADRR(TQ=10)	42.6	59.8	50.4	27.2	46.4	37	6	7	7	32.6	50.6	41.2	17.2	37.2	27.8	6	7	7
ADRR with TQ=AVG	43.8	72.6	60.2	24.4	53.8	40.8	4	4	4	34.6	64.4	51	15.2	44	31	4	4	4
ADRR with TQ=(min bt + max bt)/2	43.8	72.6	60.2	24.4	53.2	40.8	4	4	4	34.6	64.4	51	15.2	44	31	4	4	4
ADRR with TQ=Median	43.8	72.6	60.2	24.4	53.2	40.8	4	4	4	34.6	64.4	51	15.2	44	31	4	4	4

(I= Burst Time in Increasing order, D= Burst Time in Decreasing Order, R= Burst Time in Random Order, NCS= Number of context switches, TAT=Turnaround time, WT=Waiting time)

4. CONCLUSION

In terms of computing power, the CPU is the most important element of a computer system. CPU scheduling involves a careful examination of pending processes to determine an improved way to service the requests. Many CPU scheduling algorithms have been presented having some advantages and disadvantages. In this paper An Augmented Dynamic round robin CPU scheduling algorithm is proposed. ADRR with varying time quantum proposed in this paper giving better performance than conventional RR algorithm. Simulation results shows that the proposed ADRR CPU scheduling algorithm is more efficient than conventional RR algorithm also prove the correctness of the theoretical results. After improvement in RR it has been found that the waiting time and turnaround time have been reduced drastically. This algorithm can be implemented to improve the performance in the systems in which RR is a preferable choice.

REFERENCES

[1] 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.

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

[3] Behera, H.S, Mohanty, R and Debashree, N (2010): A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysis, International Journal of Computer Applications (0975 – 8887) Volume 5, No.5, pp 10-15.

[4] H.S. Behera, R. Mohanty, and Debashree Nayak, "A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysis", International Journal of Computer Applications, Vol. 5, No. 5, August 2010, pp. 10-15.

[5] Sunita Mohan, "Mixed Scheduling (A New Scheduling Policy)", Proceedings of Insight'09, 25-26 November 2009. [6] Helmy, T. and A. Dekdouk, "Burst Round Robin as a Proportional-share Scheduling Algorithm", IEEEGCC, http://eprints.kfupm.edu.sa/1462/, 2007.

[6] Operating Systems_ CPU Scheduling, http://www.cs.uic.edu/~jbell/Course Notes/OperatingSystems/5_CPU_Scheduling.html,accessed 8th October 2013.

[7] Ajit, S, Priyanka, G and Sahil, B (2010): An Optimized Round Robin Scheduling Algorithm for CPU Scheduling, International Journal on Computer Science and Engineering (IJCSE), Vol. 02, No. 07, 2383-2385, pp 2382-2385.

[8] Ishwari, S. R and Deepa, G (2012): 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, pp 1-11.

[9] Manish K. M. and Abdul Kadir K. (2012): An Improved Round Robin CPU Scheduling Algorithm, Journal of Global Research in Computer Science, ISSN: 2229-371X, Volume 3, No. 6, pp 64-69.

[10] Ali Jbaeer Dawood: Improving Efficiency of Round Robin Scheduling Using Ascending Quantum and Minumim-Maxumum Burst, J. of university of anbar for pure science: Vol.6:NO.2: 2012 ISSN: 1991-8941.

[11] Lalit, K, Rajendra, S and Praveen, S (2011): Optimized Scheduling Algorithm, International Journal of Computer Applications, pp 106-109.

[12] Soraj, H and Roy, K.C: Adaptive Round Robin scheduling using shortest burst approach, based on smart time slice", International Journal of Data Engineering (IJDE), Volume 2, Issue 3, www.cscjournals.org/csc/manuscript/Journals/IJDE/.../IJDE-57.pdf ,accessed 01th December 2012.

[13] Debashree Nayak, Sanjeev Kumar Malla, and Debashree Debadarshini, "Improved Round Robin Scheduling using Dynamic Time Quantum", International Journal of Computer Applications, Vol. 38, No. 5, January 2012, pp. 34-38.

[14]Manish Kumar Mishra, Abdul kadir khan "An improved round robin CPU scheduling algorithm" Journal of global research in

computer science (ISSN-2229-371X), volume 3, No.6, June 2012.

[15].http://www1bpt.bridgeport.edu/sed/projects/cs503/Spring_2001/kode/os/scheduling.htm

[16]. Milan Milenkovic, "Operating System Concepts and Design", Second Edition McGraw
Hill International, 1992.

[17]. Leland L. Beck, "System Software", 3rd Ed., Addison Wesley, 1997.

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

[19]. Ritika Gakher, Saman Rasool, "A New Approach for Dynamic Time Quantum Allocation in Round Robin Process Scheduling Algorithm", International Journal of Computer Applications (0975 – 8887) Volume 102– No.14, September 2014

[20]. Samih M. Mostafa, S. Z. Rida & Safwat H. Hamad, "Finding time quantum of round robin CPU scheduling algorithm in general computing system using integer programming" ,IJRRAS 5 (1) October 2010

[21]. 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), ISSN: 2319 – 1058,Vol. 1 Issue 3 Oct 2012.

[22]. Manish Kumar Mishra, Faizur Rashid, "An Improved Round Robin CPU Scheduling Algorithm With Varying Time Quantum", International Journal of Computer Science, Engineering and Applications (IJCSEA) Vol.4, No.4, August 2014.

[23] Abdullahi, I., and Junaidu, S. B (2013): Empirical Framework to Migrate Problems in Longer Job First Scheduling Algorithm (LJF+CBT), International Journal of Computer Applications (0975 – 8887) Volume 75– No.14, pp 9-14.