ISSN:0975-9646

A Priority Based Round Robin CPU Scheduling Algorithm

Monika Belwal
M.Tech Scholar, UTU, Dehradun
Uttarakhand Technical University, Dehradun

Sanjay Kumar
Assistant Professor - CSE
Uttarakhand Technical University, Dehradun

***Abstract*— Operating system is an essential part of any computer system. Scheduling is the basic paradigm of an Operating System. Process scheduling is the technique of arrangement of processes in order to execute in a defined fashion. The aim of scheduling is to make the system efficient and fast. The basic scheduling algorithms are: First Come First Serve (FCFS), Round Robin, Priority Based Scheduling, Shortest Job First (SJF) etc. Our main focus is on Round Robin Scheduling algorithm. There are various issues related to Round Robin Scheduling. One of the limitations of Round Robin Scheduling is the length of the Time Quantum. If the Time Quantum is too large, the scheduling will be similar to FCFS otherwise a smaller Time Quantum results in increased Context Switches. Our main objective is to overcome this limitation of traditional Round Robin scheduling algorithm and make maximum utilization of the CPU and make the system more efficient. In this thesis, we proposed an algorithm that categorizes the processes as High priority processes and low priority processes. The proposed scheme reduces the average waiting time of high priority process irrespective of the low priority process. The overall average waiting time will change according to the set of processes considered. Based on the average waiting time, it is justified that the proposed scheme provides reduced average waiting time of the process set than previously proposed schemes.**

***Keywords*- CPU Scheduling, Round Robin Scheduling, Priority Scheduling, Waiting Time, Turnaround Time, Time Quantum.**

I. INTRODUCTION

The practice of executing a process to acquire the CPU control while the execution of another process is suspended (in waiting state) due to unavailability of resources (such as I/O), thus making full use of CPU is known as CPU scheduling [11]. The aim of CPU scheduling is to make the system effective, reckless, just & to maximize the utilization of CPU. The process scheduling is the action of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy. Process scheduling is an indispensable part of Multiprogramming operating systems [12]. The operating systems allow multiple processes to be loaded into the executable memory at a time and the loaded process shares the CPU using time multiplexing.

There are two types of CPU scheduling algorithms, preemptive and non-preemptive. In the Pre-emptive category of scheduling algorithms, a process which is allocated to the processor can be stopped and the running state of the corresponding process is changes to waiting state. [15]. The policy of temporarily suspending the processes that are logically runnable is called Preemptive Scheduling. The resources are allocated to a process for a partial time. Process can be interrupted in between. If a process with a high priority frequently arrives in the ready queue, the process with low priority may starve. The limitation of Preemptive scheduling is overheads of scheduling the processes. In the non-pre-emptive category of scheduling, if a process has been allotted to the CPU; the CPU cannot be taken away from that process until the execution of the process is completed [16]. A process holding the processor releases only after its completion not before that.

Although there are a number of CPU scheduling algorithms but some of the common are; First in first out (FIFO), Shortest job first (SJF), Priority scheduling and Round robin CPU scheduling algorithm.

II. LITERATURE SURVEY

In the FCFS scheduling, Jobs are implemented on first come, first serve basis [1]. It can be a non-preemptive as well as pre-emptive scheduling algorithm based on the necessities. It is simple to comprehend and implement. The implementation of FCFS is based on FIFO queue. The limitation of FCFS scheduling is its poor presentation as average wait time is high.

This is also known as shortest job first, or SJF [3]. This algorithm is both pre-emptive as well as non-preemptive by nature. It is considered as the best methodology to decrease waiting time. It is significant to implement SJF in Batch systems because the essential CPU time is known in advance. The implementation of SJF is not possible in interactive systems as the required CPU time is unknown for such systems. The processer should know in advance the amount of time the process will take.

Priority scheduling is a non-preemptive algorithm. Basically it is one of the most common scheduling algorithms in batch systems [5]. Each process is allotted a priority. The process with maximum priority is to be executed first and so on. Processes with identical priority are executed on first come first served basis. The priority of the processes is allocated based on the memory requirements, time requirements or any other resource requirement.

Round Robin is the preemptive process scheduling algorithm. Each process is delivered a fix time to execute, it is called a quantum [8]. In this type of scheduling, a process is executed for a particular time period called Time Quantum. When this Time Quantum is reduced to zero, it is preempted and other process start its execution for a given time period. Context switching is needed to store status of preempted processes.

Multiple-level queues are manual scheduling algorithm [15]. This algorithm uses other existing algorithms to group and organize jobs with common features. Numerous queues are retained for processes with mutual characteristics. Each queue can have its peculiar scheduling algorithms [8]. Priorities are allotted to each

queue. For example, OS-bound jobs can be arranged in one queue and all I/O-bound jobs in another queue. The Process Scheduler then in turn selects jobs from each queue and allots them to the CPU based on the algorithm allotted to the queue. Multi-level queue scheduling was generated for circumstances in which processes are certainly categorized into different groups.

III. SHORTCOMINGS OF EXISTING ALGORITHM

We have considered the traditional round robin algorithm as the existing algorithm. The RR algorithm can be considered as an efficient algorithm since it provides equal chance of execution to all the processes in the process set. Research have shown that our system consist of critical processes with high priority along with the normal (low priority) processes. The RR algorithm does not consider the priorities of the processes which can be considered as the major drawback. So we proposed a methodology in order to overcome this limitation of RR algorithm

Consider the following set of processes with time quantum 4 -

Table 1 Process in Ready Queue for Existing Methodology

Process Name	Priority	Burst Time
P0	0	5
P1	1	3
P2	1	12
P3	0	9
P4	0	8

We know that round robin scheduling provides equal opportunity to execute all the processes in the process set. Hence, the Gantt chart and waiting time for the above set of processes are shown in figure 1.

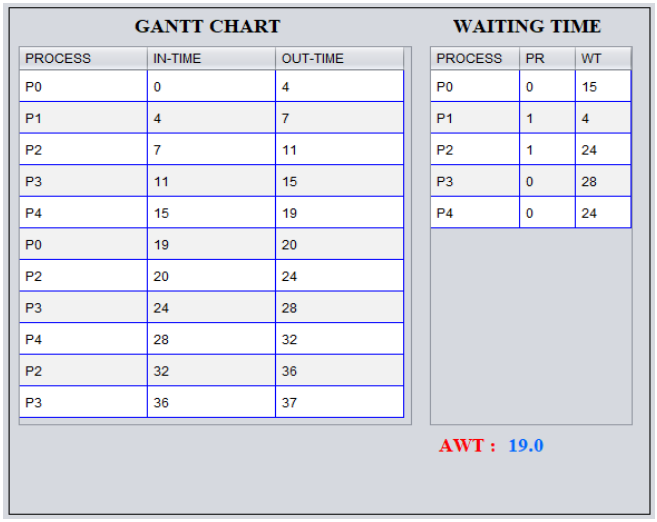


Figure 1 Gantt chart of Existing Methodology

The average waiting time (AWT) of low and high priority processes is shown below in figure 2.

AWT	Existing
AWT of LPQ	22.333334
AWT of HPQ	14.0
Overall AWT	19.0

Figure 2 Waiting Time Analysis of Existing Methodology

IV. PROPOSED METHOD

The Round Robin algorithm considers the job with equal priority. The processes are executed for a particular time slice called Time Quantum (TQ) at a time. So, a process can be executed until its time quantum (TQ) terminates or the process terminates by its own after conclusion of its CPU burst time. The processes contained in a system are of different priorities i.e. high priority process and low priority process. The high priority process or critical process may require the CPU on urgent basis (i.e. program to shut down computer because of increased temperature, alert on unauthorized access, etc.). The other types of processes are with normal priority.

.

V. PROPOSED ALGORITHM

Our proposed algorithm is given below-

Step 1: Enter process name, priority and burst time.
Step 2: Store the above details in a queue called READYQ
Step 3: Create two separate queues, first HIGHPQ for high priority processes and second LOWPQ for normal priority processes.
Step 4: Do step 5 to step 11 until remaining CPU burst time of processes of both the queues (HIGHPQ and LOWPQ) become zero.
Step 5: Select next process from either HIGHPQ or LOWPQ on alternate basis. First, a process form HIGHPQ must be selected as it should get priority over normal priority processes.
Step 6: If the remaining CPU burst time of selected process is greater than or equal to time quantum then do step 7, otherwise do step 8.
Step 7: Execute that process for duration of time quantum.
Step 8: Execute the selected process until its remaining burst time become zero.
Step 9: Update the remaining CPU burst time of the respective process in respective queue.
Step 10: Store the IN-TIME and OUT-TIME of the process into a table GANTTCHART.
Step 11: If above process has selected from HIGHPQ then swap the next turn to LOWPQ and vice versa.

The high priority processes should get priority to execute. In this research, I have proposed a methodology, which provides alternate chance to high and low priority processes. A process from high priority queue is selected first then next process is selected from low priority queue. The steps for the methodology are given below-

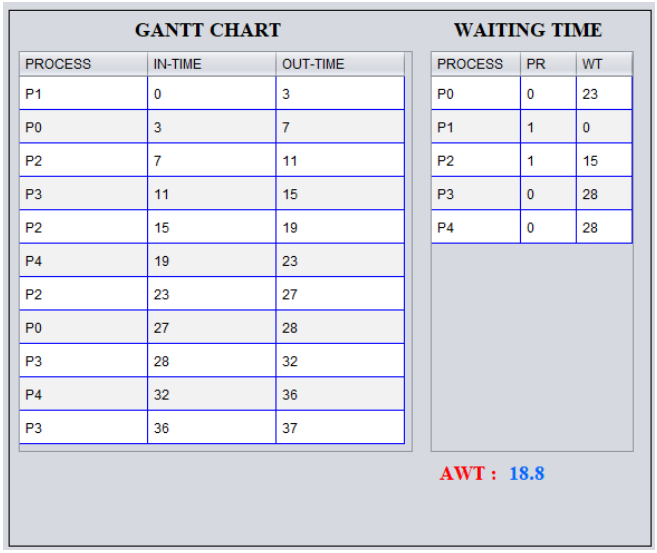
HIGHPQ: This queue contains the processes of high priority.

Process Name	Priority	Burst Time
P1	1	3
P2	1	12

LOWQPQ: This queue contains the processes of low priority.

Process Name	Priority	Burst Time
P0	0	5
P3	0	9
P4	0	8

The Gantt chart and waiting time for the processes of table 1 with time quantum 4 is shown below in figure 3.



GANTT CHART

PROCESS	IN-TIME	OUT-TIME
P1	0	3
P0	3	7
P2	7	11
P3	11	15
P2	15	19
P4	19	23
P2	23	27
P0	27	28
P3	28	32
P4	32	36
P3	36	37

WAITING TIME

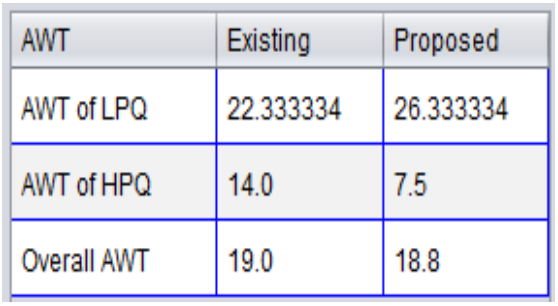
PROCESS	PR	WT
P0	0	23
P1	1	0
P2	1	15
P3	0	28
P4	0	28

AWT : 18.8

Figure 3 Working of Proposed Methodology

VI. Results And Analysis

In the figure shown below, using proposed algorithm the average waiting time of high priority process is 7.5 which is approximately half of average waiting time using existing algorithm. The overall waiting time of process set is also reduced using the proposed algorithm.



AWT	Existing	Proposed
AWT of LPQ	22.333334	26.333334
AWT of HPQ	14.0	7.5
Overall AWT	19.0	18.8

Figure 4 Result Analysis of Existing Vs Proposed Methodology

The same result can be analyzed using bar chart shown in figure 5.

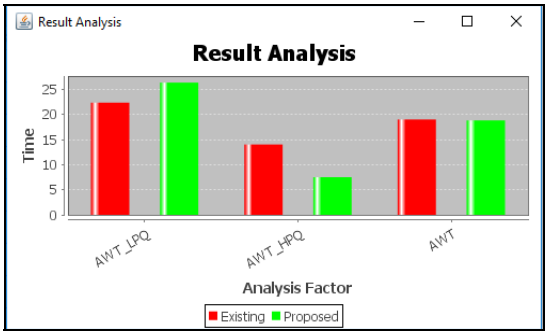


Figure 5 Result Analysis of Existing Vs Proposed Methodology Using Bar chart

VII. Conclusion

In this research, I have preserved the motivation of traditional round robin that all process should get equal chance to execute for a particular time quantum. The only improvement is that if high priority process are stored at rear side in the ready queue, then those processes will not bounded to execute too late due to late arrival. The proposed methodology will definitely reduce average waiting time of high priority processes however; it may increase the average waiting time of normal priority processes. The overall average waiting time of all the processes stored in ready queue may or may not improve depending on the set of processes. Although the proposed algorithm shows better result for high priority processes, still there is always a need and motivation for better results. In future, the result can be improved using variable time quantum. The execution of algorithm can also be improved by using efficient data structures.

VIII. References

[1] Sanjay Kumar Panda and Saurav Kumar Bhoi, "An Effective Round Robin Algorithm using Min-Max Dispersion Measure", International Journal on Computer Science and Engineering, 4(1), pp. 45-53, January 2012.

[2] Pallab Banerjee, Probal Banerjee, Shweta Sonali Dhal, "Comparative Performance Analysis of Average Max Round Robin Scheduling Algorithm (AMRR) using Dynamic Time Quantum with Round Robin Scheduling Algorithm usingStatic Time Quanmtum", International Journal of Innovative Technology and Exploring Engineering, 1(3), pp. 56-62,August 2012.

[3] P.Surendra Varma, "A Finest Time Quantum for Improving Shortest Remaining Burst Round Robin (SRBRR) Algorithm", Journal of Global Research in Computer Science, 4 (3), pp. 10-15, March 2013.

[4] Raman, Dr.Pradeep Kumar Mittal, "An Efficient Dynamic Round Robin CPU Scheduling Algorithm (EDRR)", International Journal of Advanced Research in Computer Science and Software Engineering, 4(5), pp. 907-910, May 2014.

[5] Silberschatz, A., P.B. Galvin and G. Gagne, Operating Systems Concepts. 7th Edn., John Wiley and Sons, USA., ISBN:13: 978-0471694663, pp. 944.

[6] Rakesh Mohanty, H. S. Behera, Khusbu Patwari, Monisha Dash, "Design and Performance Evaluation of a New Proposed Shortest Remaining Burst Round Robin (SRBRR) Scheduling Algorithm", Proc. of International Symposium on Computer Engineering & Technology 2010, Vol 17, pp. 126-137, 2010 .

[7] R. J. Matarneh, "Seif-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the Now Running Proceses", American Journal of Applied Sciences, 6(10), pp. 1831-1837, 2009.

[8] H. S. Behera, R. Mohanty, and D. Nayak, "A New Proposed Dynamic Quantum with Re-Adjusted Round Robin Scheduling Algorithm and Its Performance Analysis", International Journal of Computer Applications, 5(5), pp. 10-15, August 2010.

[9] Abraham Silberschatz, Peter Baer Galvin, Greg Gagne, "Operating System Concepts", Sixth Edition.

[10] E.O. Oyetunji, A. E. Oluleye," Performance Assessment of Some CPU Scheduling Algorithms", Research Journal of Information Technology,1(1): pp 22-26, 2009

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

[12] Rami J. Matarneh."Self-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of Now Running Processes", American J. of Applied Sciences 6(10):1831-1837, 2009.

[13] Sourav Kumar Bhoi, Sanjaya Kumar Panda, Debashee Tarai, "Enhancing cpu performance using subcontrary mean dynamic round robin (smdrr) scheduling algorithm" ,JGRCS, Volume 2, No. 12, December 2011, pp.17-21

[14] Rakesh Mohanty, H. S. Behera, Khusbu Patwari, Monisha Dash, "Design and Performance Evaluation of a New Proposed Shortest Remaining Burst Round Robin (SRBRR) Scheduling Algorithm", International Symposium on Computer Engineering & Technology (ISCET), Vol 17, 2010

[15] P.Surendra Varma , "A FINEST TIME QUANTUM FOR IMPROVING SHORTEST REMAINING BURST ROUND ROBIN (SRBRR) ALGORITHM" Journal of Global Research in Computer Science, 4 (3), March 2013, 10-15

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

[17] Ishwari Singh Rajput," A Priority based Round Robin CPU Scheduling Algorithm for Real Time Systems", (IJIET)International Journal of Innovations in Engineering and Technology Vol. 1 Issue 3 Oct 2012

[18] Manish Kumar Mishra & Abdul Kadir Khan, (2012) "An Improved Round Robin CPU Scheduling Algorithm", Journal of Global Research in Computer Science, Vol. 3, No. 6, pp 64-69.

[19] Abdulrazak Abdulrahim, Salisu Aliyu, Ahmad M Mustapha & Saleh E. Abdullahi, (2014) "An Additional Improvement in Round Robin (AAIRR) CPU Scheduling Algorithm", International Journal of Advanced Research in Computer Science and Software Engineering, Vol. 4, Issue 2, pp 601-610.

[20] Abdulrazak Abdulrahim, Saleh E. Abdullahi & Junaidu B. Sahalu, (2014) "A New Improved Round Robin (NIRR) CPU Scheduling Algorithm", International Journal of Computer Applications, Vol. 90, No. 4, pp 27-33.

[21] An Effective Round Robin Algorithm using Min-Max Dispersion Measure. Panda, Sanjaya Kumar; Bhoi, Sourav Kumar // International Journal on Computer Science & Engineering;Jan2012, Vol. 4 Issue 1, p45

[22] Designing Various CPU Scheduling Techniques using SCILAB. Saini, Mona // International Journal of Computer Science & Information Technolo;2014, Vol. 5 Issue 3, p2918

[23] Self-Adjustment Time Quantum in Round Robin Algorithm Depending on Burst Time of the Now Running Processes. Matarneh, Rami J. // American Journal of Applied Sciences;2009, Vol. 6 Issue 10, p1831

[24] Two Queue based Round Robin Scheduling Algorithm for CPU Scheduling. Jindal, Srishty; Grover, Priyanka // International Journal of Computer Applications;Nov2014, Vol. 105 Issue 1-18, p21

[25] A 2LFQ Scheduling with Dynamic Time Quantum using Mean Average. Lenka, Rakesh K.; Ranjan, Prabhat // International Journal of Computer Applications;6/1/2012, Vol. 47, p15

[26] Improvised Round Robin (CPU) Scheduling Algorithm. Sirohi, Abhishek; Pratap, Aseem; Aggarwal, Mayank // International Journal of Computer Applications;Aug2014, Vol. 99, p40