"An Improved Round Robin Scheduling Algorithm for CPU scheduling"

Rakesh Kumar Yadav[1], Abhishek K Mishra [2], Navin Prakash[3] and Himanshu Sharma[4]

rakeshoso@yahoo.co.in
abhimishra2@gmail.com
naveenshran@gmail.com
cs_himanshu@rediffmail.com

College of Engineering and Technology, IFTM Campus,
Lodhipur Rajput, Moradabad, UP, INDIA

Abstract
There are many functions which are provided by operating system like process management, memory management, file management, input/output management, networking, protection system and command interpreter system. In these functions, the process management is most important function because operating system is a system program that means at the runtime process interact with hardware. Therefore, we can say that for improving the efficiency of a CPU we need to manage all process. For managing the process we use various types scheduling algorithm. There are many algorithm are available for CPU scheduling. But all algorithms have its own deficiency and limitations. In this paper, I proposed a new approach for round robin scheduling algorithm which helps to improve the efficiency of CPU.

Keywords: CPU scheduling, RR scheduling algorithm, Turnaround time, Waiting Time, Gantt chart

1. Introduction: CPU is a primary computer resource. So, its scheduling is central to operating system design. To improve both utilization and speed of CPU we need to keep several processes in memory at a time that means we use the sharing and multiprogramming concepts.

According to Seltzer, M P. Chen and J Outerhout 1990 [3], the last thirty years have seen an enormous amount of research in the area of disk scheduling algorithm. The core objective has been developed scheduling algorithms suited for certain goals sometimes with provable properties.

According to Silberchatz, Galvin and Gagne [2], in case of multi-programmed operation system CPU scheduling plays a fundamental role by switching the CPU among various processes. the intention of an Operating system should allow process many as possible running at all times in order to maximize the CPU utilization. In a multi-programmed operating. System a process is executed until it must wait for the completion of some I/O request. In this case the time has been used proficiently. A number of processes are kept in memory simultaneously and while one process occupy the CPU selected by the Operating.

According to Sabrina, F.C.D, Nguyen, S.Jha, D. Platt and F. Safaei [1] Scheduling is a fundamental operating system function. Almost all computer resources are sheduleed before use. The CPU is of course one of the primary resources. Thus its scheduling is central to Operating system design. CPU scheduling determines which process run when there are multiple runable processes CPU scheduling is important because it can have a big effect on resources utilization and overall performance of the system.

2. Contemporary RR scheduling Algorithm:
We can understand contemporary RR scheduling algorithm [5] by given below steps:

1. The scheduler maintains a queue of ready processes and a list of blocked and swapped out processes.
2. The PCB of newly created process is added to the end of the ready queue. The PCB of terminating process is removed from the scheduling data structures.
3. The scheduler always selects the PCB at the head of the ready queue.
4. When a running process finishes its slice, it is moved to the end of ready queue.
5. the event handler perform the following action
   a) When a process makes an I/O request or swapped out, its PCB is removed from the ready queue to blocked/swapped out list.

b) When I/O operation awaited by a process finishes or process is swapped in its PCB is removed from blocked/swapped list to the end of the ready queue.

Silberchatz, Galvin and Gagne [2] and DM Dhamdhere [5], Operating Systems Sibsankar Haldar 2009 [6] says that same things about RR algorithm. So we can conclude easily RR algorithm is extension of FCFS algorithm. Only difference is we added the preemptive and time slice concepts.

3. Scheduling Criteria.

Different CPU scheduling algorithms have different properties, and the choice of a particular algorithm may favor one class of processes over another. In choosing which algorithm to use in a particular situation, we must consider the properties of the various algorithms. Many criteria have been suggested for comparing CPU scheduling algorithms. Which characteristics are used for comparison can make a substantial difference in which algorithm is judged to be best. The criteria include the following:

- CPU Utilization. We want to keep the CPU as busy as possible.
- Throughput. If the CPU is busy executing processes, then work is being done. One measure of work is the number of processes that are completed per time unit, called throughput. For long processes, this rate may be one process per hour; for short transactions, it may be 10 processes per second.
- Turnaround time. From the point of view of a particular process, the important criterion is how long it takes to execute that process. The interval from the time of submission of a process to the time of completion is the turnaround time. Turnaround time is the sum of the periods spent waiting to get into memory, waiting in the ready queue, executing on the CPU, and doing I/O.
- Waiting time. The CPU scheduling algorithm does not affect the amount of the time during which a process executes or does I/O; it affects only the amount of time that a process spends waiting in the ready queue. Waiting time is the sum of periods spend waiting in the ready queue.
- Response time. In an interactive system, turnaround time may not be the best criterion. Often, a process can produce some output fairly early and can continue computing new results while previous results are being output to the user. Thus, another measure is the time from the submission of a request until the first response is produced. This measure, called response time, is the time it takes to start responding, not the time it takes to output the response. The turnaround time is generally limited by the speed of the output device.

It is desirable to maximize CPU utilization and throughput and to minimize turnaround time, waiting time, and response time. In most cases, we optimize the average measure. However, under some circumstances, it is desirable to optimize the minimum or maximum values rather than the average. For example, to guarantee that all users get good service, we may want to minimize the maximum response time. Investigators have suggested that, for interactive systems, it is more important to minimize the variance in the response time than to minimize the average response time. A system with reasonable and predictable response time may be considered more desirable than a system that is faster on the average but is highly variable. However, little work has been done on CPU-scheduling algorithms that minimize variance [7].

So we can conclude a good scheduling algorithm is open that is able to optimize the above performance measures. The optimization performance measures are:

- Maximize CPU utilization
- Maximize throughput
- Minimize turnaround time
- Minimize waiting time
- Minimize response time
- Maximize scheduler efficiency

4. Proposed algorithm

1. Allocate all processes to the CPU only one time as like present Round Robin scheduling algorithm.
2. After first time we select shortest job from the waiting queue and it shortest job assign to the CPU.
3. After that we select next shortest job and do step 2
4. Till the complete execution of all processes we repeat steps 2 and 3 that means while all the processes has not been finished(executed).

5. EXPERIMENT DISCUSSION

In our Proposed RR scheduling algorithm we have combined the working of SJF (shortest job first) scheduling algorithm along with contemporary RR scheduling algorithm

For doing this we have done many experiments but here I will discuss only two experiment because we assured results analysis is remain unchanged. Here I have two types of problems one is with arrival time another is without arrival time.

(I)
Suppose sate of processes is according to given below table and also let round robin quantum=2

Process	Arrival time	Burst time in ms
A	0.000	3
B	1.001	6
C	4.001	4
D	6.001	2

Then according to contemporary algorithm

Gantt chart =

A	B	A	B	C	D	B	C

Average turn around time is =8.25 ms
Total Waiting Time=18 ms

According to suggested algorithm:

Gantt chart

A	B	A	B	B	D	C	C

Total waiting Time =14 ms
Average turn around time=7.5 ms

(II)
Suppose problem is give below

Process	Burst Time in ms
A	24
B	20
C	8
D	10
E	3

Suppose RR quantum =5
According to contemporary algorithm:
Gantt chart

A	B	C	D	E	A	B	C	D	A	B	A

Waiting Time =146 ms
Average turnaround=42.2 ms

According to suggested algorithm.
Gantt chart

A	B	C	D	E	C	D	B	B	B	A	A	A	A

Total Waiting time=131 ms
Average turnaround time= 39.2 ms

We can see from the above experiment total waiting time and average turn around time both are reduced by using our proposed algorithm.
The reduction of total waiting time and turn around time shows maximum CPU utilization that and minimum response time. We can say that proposed algorithm much more efficient compare than contemporary algorithm.

6. Conclusion and future work

It we can see that our proposed algorithm is superior compare than present RR algorithm. Since we know that RR scheduling algorithm is designed especially for time sharing system. So in near future we can improve time sharing system by using this algorithm.

6. References:

1. Sabrina, F.C.D, Nguyen, S.Jha, D. Platt and F. Safaei, 2005. Processing resources scheduling in programmable networks. Computer commun, 28:676-687.
2. Silberchatz, Galvin and Gagne ,2003 .operating systems concepts,(6th edn, John Wiley and Sons)
3. Seltzer, M P. Chen and J outerhout, 1990.Disk scheduling revisited in USENIX. Winter technical conference.
4. Shamim H M 1998. Operating system, DCSA-2302. School of sciences and Technology. Bangladesh open university Gazipur-1705
5. D. M. Dhamdhere Operating Systems A Concept Based Approach**,** Second edition**,** Tata McGraw-Hill, 2006
6. Operating Systems Sibsankar Haldar 2009 , Pearson Education, India
7. http://en.wikipedia.org/wiki/Scheduling_(computing)