International Journal of Engineering Science Invention (IJESI)
ISSN (Online): 2319 – 6734, ISSN (Print): 2319 – 6726
www.ijesi.org ||Volume 7 Issue 3 Ver. 4 || March 2018 || PP72-75

An Optimized CPU Scheduling Algorithm

Annu
Assistant Professor in G.B.N.K.M.V. Anjanthali (Karnal)

Abstract: *CPU Scheduling is the basis of multi programmed operating systems .CPU scheduling is one of the most crucial operations performed by operating system. Different algorithms are available for CPU scheduling. Among those algorithms round robin scheduling and priority scheduling are the centre of concern of this paper. We combine these two algorithms to improve the various scheduling parameters which include CPU utilization, average turnaround time, average waiting time, response time and throughput of the processes. In this paper we enhance the performance of priority scheduling algorithm by combining it with the round robin (RR) scheduling algorithm.*

Keywords*: CPU Scheduling,* Round Robin (RR), Priority scheduling, average turnaround time, average waiting time.

Date of Submission:10-03-2018 Date of acceptance 20-03-2018

I. Introduction:

The CPU (Central Processing Unit) is the main important resource of the computer system; so it must be efficiently utilized. Therefore, the scheduling of CPU is central to the operating system design. CPU Scheduling is mainly the task of selecting a waiting process from the ready queue and allocating the CPU to it. Basically CPU Scheduling is defined as the art of determining which processes run on the CPU when there are the multiple processes waiting in the ready queue to get allocated to the CPU. The processes are scheduled according to the given burst time, arrival time and the priority of the process. The numbers of resource are used by the processes during execution like Memory, CPU etc [1]. The CPU is allocated to the selected process by what is known as dispatcher. The dispatcher is the module that gives the control of the CPU to the process selected by the short-term scheduler [2] or the CPU Scheduler. In other words, we can say it is made by the part of the operating system called the scheduler, using a CPU scheduling algorithm [3]. CPU Scheduling is the basis of multi-programmed operating systems (OS). CPU scheduling is the major task of a system which is performed by the OS [4]. CPU Scheduling becomes more complex when there are multiple CPUs available. By switching the CPU among various processes, the OS can make the computer more productive. In a single processor system, only one process can run at a time and any other processes will have to wait until the CPU is free and can be rescheduled. In a single processor system, whenever a process is busy in doing some I/O activity, then the CPU just sits idle. Therefore, all the waiting time is wasted and no useful work is accomplished. But with the multi-programming system, we try to use this time productively. The main goal of multi-programming is to have some processes running at all times in order to maximize CPU Utilization. Several processes are kept in the main memory at the same time. When one process has to wait, the OS takes the CPU away from that process and gives the CPU to another process. This pattern is repeated again and again. Every time, one process has to wait but another process can take the control of CPU, hence increases the CPU utilization.

II. Scheduling Criteria:

The various CPU scheduling algorithms exists which have different properties, and the choice of a particular algorithm may favor one class of processes over the other. The need for the scheduling algorithm is realized from the requirement of fast computer systems to execute more than one process at a time as well as to switch between those processes [5]. If we want to select a particular algorithm for a particular situation, we must consider the properties of various algorithms. The CPU scheduling criteria includes the various parameters which are described below [6]:

1. **CPU Utilization:** The main objective of the CPU Scheduling is to keep the CPU as busy as possible so that we can utilize CPU efficiently. The CPU should not sit idle. At one time, any of the processes waiting in the ready queue should be executed on the CPU.
2. **Throughput:** Throughput is defined as the number of processes that are completed per unit time. Short processes are completed faster in comparison to longer processes, thereby increasing the throughput of the system.

3. **Turnaround time:** It is the main important criteria in CPU Scheduling. It is defined as the time interval from the time of submission of a process to the time of its completion. Basically, it 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. The main objective is to minimize the turnaround time.
4. **Waiting Time:** Waiting time is defined as the sum of the periods spent waiting in the ready queue. Basically it is the difference in the start time and the ready time. Generally, the objective is to minimize the waiting time.
5. **Response Time:** Response time is defined as the time from the submission of a request until the first response is produced. Therefore, it is the time whenever the CPU takes to start responding, not the time it takes to output the response. Generally, the main goal is to minimize the response time.

So, from the above discussion we can conclude that a good scheduling algorithm for real time and time sharing system must possess the following properties [7]:

- Maximum CPU utilization.
- Maximum throughput.
- Minimum turnaround time.
- Minimum waiting time.
- Minimum response time.

On the basis of the criteria defined above the performance of the various CPU Scheduling algorithms are evaluated.

III. 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 described below:

a) First-Come, first-Served (FCFS) Scheduling: This is the simplest CPU scheduling algorithm. In this algorithm the process which requests the CPU first is allocated the CPU first. This algorithm is easily managed with a FIFO queue [8]. New process enters the queue through the tail of the queue and leaves through the head of the queue (when the process is allocated to the CPU). The processes are allocated to the CPU on the basis of their arrival at the queue. Once a process is allocated to the CPU, it is removed from the queue. A process does not free the CPU until it either terminates or performs I/O. When the CPU is free, it is allocated to the process at the head of the queue. The main disadvantage with the FCFS is that it can cause short processes to wait for very long processes, thereby, increasing the average waiting time which leads to bad performance. FCFS algorithm is non-preemptive in nature.

b) Shortest-Job-First (SJF) Scheduling: The SJF algorithm associates the length of the next CPU burst with each processes such that that the process that have the smallest next CPU burst is allocated to the CPU. The SJF uses the FCFS to break tie (a situation where two processes have the same length next CPU burst). That's why; this algorithm is also called as the shortest-next-CPU-burst algorithm. The SJF algorithm may be implemented as either a preemptive or non-preemptive algorithms. When the execution of a process that is currently running is interrupted in order to give the CPU to a new process with a shorter next CPU burst, it is called a preemptive SJF. On the other hand, the non-preemptive SJF will allow the currently running process to finish its CPU burst before a new process is allocated to the CPU. SJF algorithm is optimal because it provides the shortest average waiting time for a given set of processes [9]. But this algorithm may cause the problem of starvation.

c) Priority Scheduling: The PS algorithm associates with each process a priority and the CPU is allocated to the process based on their priorities. Usually, lower numbers are used to represent higher priorities. The process with the highest priority is allocated first. If there are multiple processes with same priority, typically the FCFS is used to break the tie. The priority can be allocated to a process either internally or externally. Internal priority uses some factors that are available to calculate the priority of a process. External priorities are set by criteria outside the boundary of the operating system [10]. Priority algorithm can also be implemented in a preemptive as well as in non-preemptive manner. This algorithm may also cause the problem of starvation or indefinite blocking because a process with lower priority can't be executed till the ready queue receives processes with higher priority.

d) Round Robin (RR) Scheduling: The RR algorithm is designed especially for time-sharing systems and is similar to the FCFS algorithm. Here, a small unit of time (called time quantum or time slice) is defined. A time quantum is generally from 10-100 milliseconds. So, the RR algorithm will allow the first process in the queue to run until it expires its quantum (i.e. runs for as long as the time quantum), then run the next process in the queue for the duration of the same time quantum. The RR keeps the ready processes as a FIFO queue. So, new

processes are added to the tail of the queue. Depending on the time quantum and the CPU burst requirement of each process, a process may need less than or more than a time quantum to execute on the CPU. In a situation where the process need more than a time quantum, the process runs for the full length of the time quantum and then it is preempted. The CPU is preempted, if a process does not complete before its CPU-time expires and given to the next process waiting in a queue [11]. The preempted process is then added to the tail of the queue again but with its CPU burst now a time quantum less than its previous CPU burst. This continues until the execution of the process is completed. The RR algorithm is naturally preemptive. If the time-quantum of the process is very large then this algorithm is turned into the FCFS algorithm [12].

4. Proposed Work: In our proposed work we combine the two main scheduling algorithms i.e. priority scheduling and round robin scheduling in order to improve the scheduling parameters. By using the combination of these two we can greatly reduce the average waiting time as well as the average turnaround time. In our example, we take five processes P1, P2, P3, P4 and P5 in ready queue assuming that all processes arriving at time 0 with burst time 10, 1, 2, 1 and 5 respectively as well as assuming their priorities are 3, 1, 4, 5 and 2 respectively. Firstly, we calculate the average waiting time and average turnaround time using the priority scheduling and after that calculate these two parameters by using our proposed algorithm as shown below:

Calculating average waiting time using the priority scheduling as follows:

Process	Burst Time in ms	Priority
P1	10	3
P2	1	1
P3	2	4
P4	1	5
P5	5	2

Table1. List of processes with their burst time and priority

Using priority scheduling, we can schedule these processes according to the following Gantt chart:

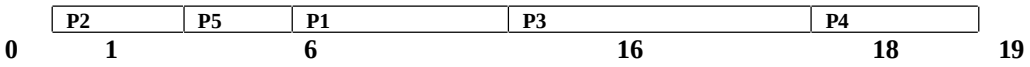

Fig1. Gantt chart for the scheduling of the processes

Waiting time for the above processes is shown below:
P1=6 ms, P2=0 ms, P3=16 ms, P4=18 ms and P5=1 ms

The average waiting time for the above scheduled processes is 8.2 milliseconds and the average turnaround time is 12 milliseconds.

But when we combine the priority scheduling with the Round Robin scheduling and scheduled processes according to this hybrid scheme, it will results in to the performance improvement of the scheduling parameters. Again we are taking the processes which are shown in the table 1 but we scheduled them by a combination of priority and round robin scheduling. The time quantum for the round robin scheduling is 2 milliseconds. We scheduled the processes according to the priority shown in the table but just for a time quantum of 2 ms in order to give a fair share of time to each process but the highest priority process is processed first after that lowered priority processes are processed. Gantt chart for the processes shown in table 1 are scheduled according to the proposed work is shown below:

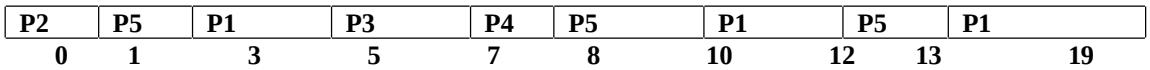

Fig2. Gantt chart for the processes shown in Table 1 using the proposed work

After applying the above hybrid scheme waiting time for the above processes is shown below:
P1=3+5+1=9 ms, P2=0 ms, P3=5 ms, P4=7 ms, P5=1+5+2=8 ms

Therefore, the average waiting time for the above scheduled processes is 5.8 ms and the average turnaround time is 9.6 milliseconds which are very less as compared to the average waiting time and the average turnaround time of the processes when they are scheduled according to the priority scheduling alone.

IV. Conclusion:

As we all know that RR is a very useful algorithm for time sharing system and priority scheduling algorithm simply allocates the CPU to the process with the highest priority. But the main problem with the priority scheduling is the starvation because a short process may wait for a longer process. In our proposed

work, we combine priority scheduling with the round robin scheduling in order to prevent the starvation and thereby improves the performance of the CPU.

References:

[1] Seltzer, M P. Chen and J outerhout, 1990.Disk scheduling revisited in USENIX. Winter technical conference Shamim H M 1998. Operating system, DCSA-2302.

[2] Abbas Noon, Ali Kalakech, 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.

[3] Suri, P.K and Sumit, M (2012): Design of Stochastic Simulator for Analyzing the Impact of Scalability on CPU Scheduling Algorithms, International Journal of Computer Applications (0975 – 8887) Volume 49– No.17, pp 4-9.

[4] .Neetu Goel, Dr. R. B. Garg, "A Comparative Study of CPU Scheduling Algorithms", International Journal of Graphics & Image Processing, Vol 2, issue 4, November 2012.

[5] William Stallings, Operating Systems Internal and Design Principles, 5th Edition 2006.

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

[7] 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.

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

[9] Oyetunji, E.O and Oluleye, A. E (2009): Performance Assessment of Some CPU Scheduling Algorithms, Journal of Information Technology 1(1): 22-26, ISSN: 2041-3114, pp 22-26.

[10] Pushpraj Singh, Vinod Singh, Anjani Pandey, "Analysis and Comparison of CPU Scheduling Algorithms", International Journal of Emerging Technology and Advanced Engineering, Volume 4, Issue 1, ISSN 22502459, ISO 9001:2008, January 2014.

[11] Sukumar Babu Bandarupalli, Neelima Priyanka Nutulapati, Prof. Dr. P.Suresh Varma, "A Novel CPU Scheduling Algorithm– Preemptive & Non-Preemptive", International Journal of Modern Engineering Research (IJMER), Vol.2, Issue.6, pp-4484-4490 ISSN: 2249-6645, Nov-Dec. 2012.

[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 10th December 2012.

Annu "An Optimized CPU Scheduling Algorithm "International Journal of Engineering Science Invention (IJESI), vol. 07, no. 03, 2018, pp72-75