INTERNATIONAL JOURNAL OF RESEARCH IN COMPUTER APPLICATIONS AND ROBOTICS
ISSN 2320-7345

A SURVEY ON CPU SCHEDULING

Meenakshi Saini and Navdeep Kumar

Research scholar, HEC Jagadhri, Haryana, India
H.O.D, CSE Deptt. , HEC Jagadhri, Haryana, India

Abstract: - Scheduling is the fundamental function of operating system. For scheduling, resources of system shared among processes which are going to be executed. CPU scheduling is a technique by which processes are allocating to the CPU for a specific time quantum. In this paper the review of different scheduling algorithms are perform with different parameters, such as running time, burst time and waiting times etc. The reviews algorithms are first come first serve, Shortest Job First, Round Robin, and Priority scheduling algorithm.

Keywords – Burst Time, CPU Scheduling, Operating System, Round Robin, Scheduling Algorithms.

INTRODUCTION

An operating system [1] [3] is a program that manages the hardware and software resources of a computer. It is the first thing that is loaded into memory when we turn on the computer. Without the operating system, each programmer would have to create a way to send data to a printer, tell it how to read a disk file, and how to deal with other programs.

In order for a computer to be able to handle multiple applications simultaneously, there must be an effective way of using the CPU. Several processes may be running at the same time, so there has to be some kind of order to allow each process to get its share of CPU time.

Over the years, scheduling has been the focus of intensive research, and many different algorithms have been implemented. Today, the emphasis in scheduling research is on exploiting multiprocessor systems, particularly for multithreaded applications, and real time scheduling.

CPU Scheduling

CPU scheduling [1] [6] is the basis of multi-programmed operating system. By switching the CPU among processes, the operating system can make the computer more productive. A multiprogramming operating system allows more than one processes to be loaded into the executable memory at a time and for the loaded processes to share the CPU using time-multiplexing. Part of the reason for using multiprogramming is that the operating system itself is implemented as one or more processes, so there must be a way for the operating system and application processes to share the CPU. Another main reason is the need for process to perform I/O operations in the normal course of computation. Since I/O operations ordinarily require orders of magnitude more time to complete than do CPU

instructions, multiprogramming systems allocate the CPU to another process whenever a process invokes an I/O operation. Scheduling refers to the way processes are assigned to run on the available CPUs, since there are many more processes running than there are available CPU.

CPU Scheduler

Whenever, the CPU becomes idle, the operating system must select one of the processes in the ready-queue to be executed. The selection process [1] is carried out the short-term scheduler or CPU scheduler. The CPU scheduler [2] selects a process from the processes in memory that are ready to execute and allocates the CPU to that process. The ready queue is not necessarily a first-in, first-out (FIFO) queue. It can be implemented as a FIFO queue a priority queue. A tree or simply an unordered linked list. Conceptually, however, all the processes in the ready queue are lined up waiting for a chance to run on the CPU. The records in the queues are generally process control blocks (PCB) of the processes.

Dispatcher

Another component involved in the CPU- scheduling [1] [7] function is the dispatcher. The dispatcher [1] is the module that gives control of the CPU to the process selected by the short-term scheduler; this involves:

- Switching context
- Switching to user mode
- Jumping to the proper location in the user program to restart that program

The dispatcher should be as fast as possible, since it is invoked during every process switch. The time it takes for the dispatcher to stop one process and start another running is known as the dispatch latency.

Scheduling Criteria

Different CPU scheduling algorithms have different [4] [8] , and the choice of a particular algorithm may favour 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.

The criteria [1] include the following:

- **CPU utilization**: We want to keep the CPU as busy as possible. Conceptually, CPU utilization can range from 0 to 100 percent. In a real system, it should range from 40 percent (for a lightly loaded system) to 90 percent (for a heavily used system).
- **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 time during which a process executes or does I/O; it affects only the amount of time that process spends waiting in the ready queue. Waiting time is the sum of the periods spent 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 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.

- **Context Switch:** A context switch is computing process of storing and restoring state of a CPU so that execution can be resumed from same point at a later time. Context switch are usually computationally intensive, lead to wastage of time, memory, scheduler overhead so much of the design of operating system is to optimize these switches.

It is desirable to maximize CPU utilization and throughput and minimize turnaround time, waiting time, and response time.

Scheduling Algorithms

CPU scheduling deals with the problem of deciding which of the processes in the ready queue is to to be allocated the CPU. There are many different CPU scheduling algorithms [1];
In this section we describe several of them.

1. First-Come, First- Served Scheduling

In this algorithm [2], the process to be selected is *the process which requests the processor first*. This is the process whose PCB is at the head of the ready queue. Contrary to its simplicity, its performance may often be poor compared to other algorithms. FCFS [3] may cause processes with short processor bursts to wait for a long time. If one process with a long processor burst gets the processor, all the others will wait for it to release it and the ready queue will be filled very much. This is called the *convoy effect*.

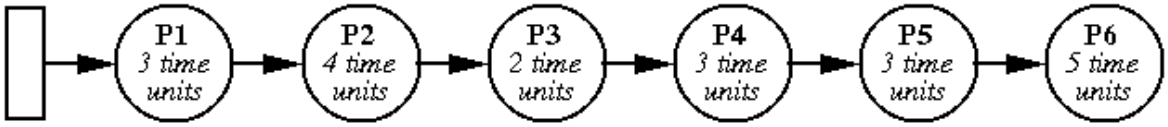

Figure 1.1 Diagram representing FCFS scheduling

The FCFS scheduling algorithm [1] is non-pre-emptive. Once the CPU has been allocated to a process, it keeps the CPU until it releases the CPU, either by terminating or by requesting I/O. The FCFS algorithm is thus particularly troublesome for time-sharing systems, where it is important that each user get a share of the CPU at regular intervals. It would be disastrous to allow one process to keep the CPU for an extended period.

2. Shortest-Job-First Scheduling

In this strategy the scheduler arranges processes [2] [3] with the Burst times in the ready queue, so that the process with low burst time is scheduled first. If two processes having same burst time and arrival time, then FCFS procedure is followed.

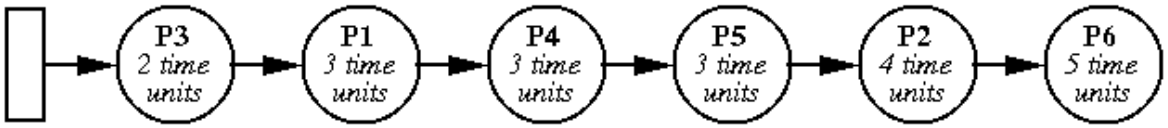

Figure 1.2 Diagram representing SJF scheduling

Shortest Remaining Time First (SRTF):

This is same as the SJF [1] with pre-emption, which small modification. For scheduling the jobs system [2] need to consider the remaining burst time of the job which is presently executed by the CPU also along with the burst time of the jobs present in the ready queue

.

3. Priority Scheduling

It provides the priority to each process and selects the highest priority process from the ready queue. A priority scheduling algorithm [4] can leave some low-priority processes in the ready queue indefinitely. If the system is heavily loaded, it is a great probability that there is a higher priority process to grab the processor. This is called the starvation problem. One solution for the starvation problem [1] might be to gradually increase the priority of processes that stay in the system for a long time.

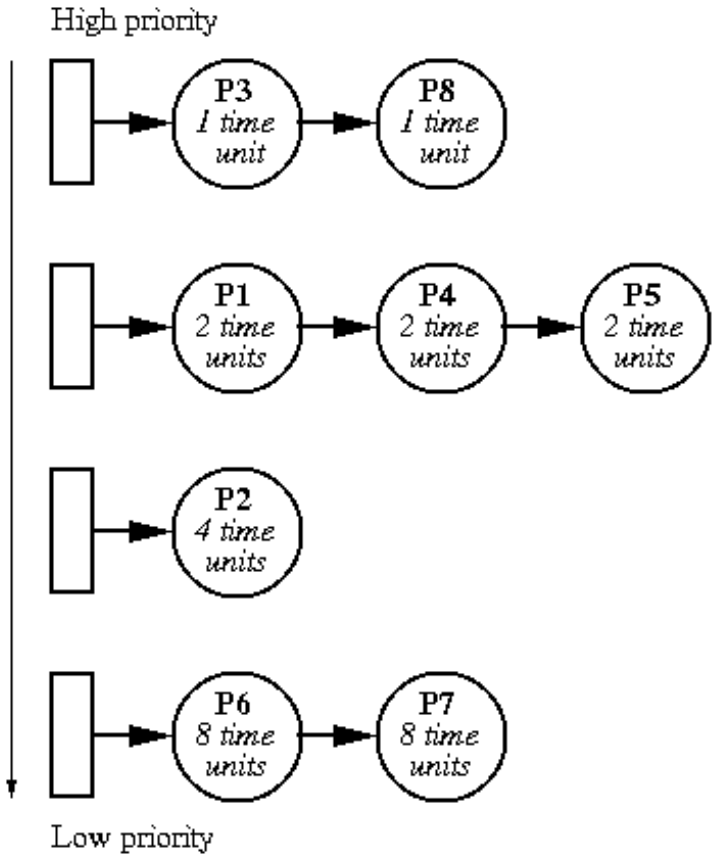

Figure 1.3 Diagram representing Priority scheduling

Priority scheduling [3] can be either pre-emptive or non-pre-emptive .A major problem with priority scheduling algorithms is indefinite blocking, or starvation. A process that is ready to run but waiting for the CPU can be considered blocked. A priority scheduling algorithm can leave some low- priority processes waiting indefinitely.
A solution to the problem of indefinite blockage of low-priority processes is aging**.** Aging is a technique of gradually increasing the priority of processes that wait in the system for along time.

4. Round-Robin Scheduling

Round Robin (RR) is one of the oldest, simplest, and fairest and most widely used scheduling algorithms, [1] designed especially for time-sharing systems. Here every process has equal priority and is given a time quantum after which the process is preempted. The OS using RRS, takes the first process from the ready queue, sets a timer to interrupt after one time quantum and gives the processor to that process. If the process has a processor burst time smaller than the time quantum, then it releases the processor voluntarily, either by terminating or by issuing an I/O request. The OS then proceed with the next process in the ready queue. On the other hand, if the process has a processor burst time greater than the time quantum, then the timer will go off after one time quantum expires, and it interrupts (preempts) the current process and puts its PCB to the end of the ready queue.

Table 1.1 Inputs for Round Robin Scheduling

Process	Burst Time
P1	24
P2	3
P3	3

If we use a time quantum of 4 milliseconds, then process *Pi* gets the first 4 milliseconds. Since it requires another 20 milliseconds, it is pre-empted after the first time quantum, and the **CPU** is given to the next process in the queue, process *P2.* Since process *P2* does not need 4 milliseconds, it quits before its time quantum expires. The CPU is then given to the next process, process *P3.* Once each process has received 1 time quantum, the CPU is returned to process *P1* for an additional time quantum. The resulting RR schedule is

P1	P2	P3	P1	P1	P1	P1	P1
0	4	7	10	14	18	22	26	30

The average waiting time is 17/3 = 5.66 milliseconds

In the RR scheduling algorithm, [2] no process is allocated the CPU for more than 1 time quantum in a row. If a process's CPU burst exceeds 1 time quantum, that process is pre-empted and is put back in the ready queue. The RR scheduling algorithm [4] is thus pre-emptive.

CONCLUSION

Different scheduling algorithms are reviews in this paper are according to their CPU overhead, throughput, turnaround time and response time. The FCFS have low throughput, low turnaround time, high turnaround time and low response time. SJF have the Medium CPU overhead, high throughput, medium turnaround and medium response time. RR has the high CPU overhead, medium throughput, medium turnaround time and high response

time. Priority Scheduling have the medium CPU overhead, low throughput, high turnaround time and high response time.

REFERENCES

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

[2] Sukanya Suranauwarat, "A CPU Scheduling Algorithm Simulator", 37th ASEE/IEEE Frontiers in Education Conference October 10 – 13, 2007.

[3] Tarek Helmy, Abdelkader Dekdouk, "Burst Round Robin: As a Proportional-Share Scheduling Algorithm", IEEE Proceedings of the fourth IEEE-GCC Conference on towards Techno-Industrial Innovations, pp. 424-428, 11- 14 November, 2007.

[4] Ishwari Singh Rajput" A priority based Round-Robin CPU scheduling algorithm for real Time system" (IJIET) International Journal of innovation in engineering & technology. Vol 1,issue 3 ,2012

[5] 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.

[6] A. Noon, A. Kalakech, and S. Kadry, "A new Round Robin Based scheduling algorithm for Operating Systems: Dynamic Quantum time Mean Average", International Journal of Computer Science Issues. 8(3), 224-229, 2011.

[7] M.H. Zahedi, M. Ghazizadeh¸ and M. Naghibzadeh, "Fuzzy Round Robin CPU Scheduling (FRRCS) Algorithm", Advances in Computer and Information Sciences and Engineering. 348-353, 2008.

[8] P.Surendra Varma " A finest time quantum for improving shortest remaining burst Round-Robin algorithm" Journal of global research in computer science. Vol 4, no.3, 2013.

[9] Jason Nieh "Virtual-Time Round-Robin: An O (1) Proportional Share Scheduler" Proceedings Of the 2001 USENIX Annual Technical Conference, Boston , Massachusetts , USA 2001.