Study And Analysis of Basic CPU Scheduling Algorithms

Mrs. Shital Vivek Ghate[1]
[1, 2] RTMNU Nagpur, India

Abstract*- The process scheduling is the activity 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. CPU scheduling is the basis of multi-programmed operating systems. Scheduling is a fundamental operating system function, since almost all computer resources are scheduled before use. By switching the CPU among processes, the operating system can make the computer more productive .In this paper we study several CPU-scheduling algorithms. Objective of this paper is to describe various CPU-scheduling algorithms, and find out the best algorithm is for the particular situation.*

Keywords- execution time, scheduling, operating system(os), Scheduling Algorithm, turnaround time, waiting time.

I. INTRODUCTION

Basic Concepts

In a single-processor system, only one process can run at a time, any others must wait until the CPU is free and can be rescheduled. The objective of multiprogramming is to have some process running at all times, to maximize CPU utilization. A process is executed until it must wait, typically for the completion of some I/O request. In a simple computer system, the CPU then just sits idle. All this waiting time is wasted; no useful work is accomplished. With multiprogramming, we try to use this time productively. Several processes are kept in memory at one time. When one process has to wait, the operating system takes the CPU away from that process and gives the CPU to another process. This pattern continues. Every time one process has to wait, another process can take over use of the CPU. Scheduling of this kind is a fundamental operating-system function. Almost all computer resources are scheduled before use.

Process Scheduling Queues

The OS maintains all PCBs in Process Scheduling Queues. The OS maintains a separate queue for each of the process states and PCBs of all processes in the same execution state are placed in the same queue. When the state of a process is changed, its PCB is unlinked from its current queue and moved to its new state queue.

The Operating System maintains the following important process scheduling queues:

Job queue - This queue keeps all the processes in the system.

Ready queue - This queue keeps a set of all processes residing in main memory, ready and waiting to execute. A new process is always put in this queue.

Device queues - The processes which are blocked due to unavailability of an I/O device constitute this queue.

The OS can use different policies to manage each queue (FIFO, Round Robin, Priority, etc.). The OS scheduler determines how to move processes between the ready and run queues which can only have one entry per processor core on the system; in the given figure diagram. it has been merged with the CPU.

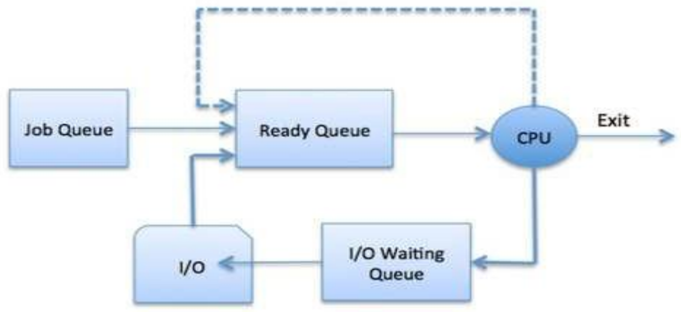


Figure 1.

When a new process is created, it enters into the system as in the running state. Processes that are not running are kept in queue, waiting for their turn to execute. Each entry in the queue is a pointer to a particular process. Queue is implemented by using linked list. Use of dispatcher is as follows. When a process is interrupted, that process is transferred in the waiting queue. If the process has completed or aborted, the process is discarded. In either case, the dispatcher then selects a process from the queue to execute.

Schedulers

are special system software which handle process scheduling in various ways. Their main task is to select the jobs to be submitted into the system and to decide which process to run. Schedulers are of three types:

- Long-Term Scheduler
- Short-Term Scheduler
- Medium-Term Scheduler

Long-Term Scheduler

It is also called a job scheduler. A long-term scheduler determines which programs are admitted to the system for processing. It selects processes from the job queue and loads them into memory for execution. Process loads into the memory for CPU scheduling. The primary objective of the job scheduler is to provide a balanced mix of jobs, such as I/O bound and processor bound. It also controls the degree of multiprogramming. On some systems, the long-term scheduler may not be available or minimal. Time-sharing operating systems have no long term scheduler. When a process changes the state from new to ready, then there is use of long-term scheduler.

Short-Term Scheduler

It is also called as CPU scheduler. Its main objective is to increase 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 a process among the processes that are ready to execute and allocates CPU to one of them. Short-term schedulers, also known as dispatchers, make the decision of which process to execute next. Short-term schedulers are faster than long-term schedulers.

Medium-Term Scheduler

Medium-term scheduling is a part of swapping. It removes the processes from the memory. It reduces the degree of multiprogramming. A running process may become suspended if it makes an I/O request. A suspended processes cannot make any progress towards completion. In this condition, to remove the process from memory and make space for other processes, the suspended process is moved to the secondary storage. This process is called swapping, and the process is said to be swapped out or rolled out. Swapping may be necessary to improve the process mix.

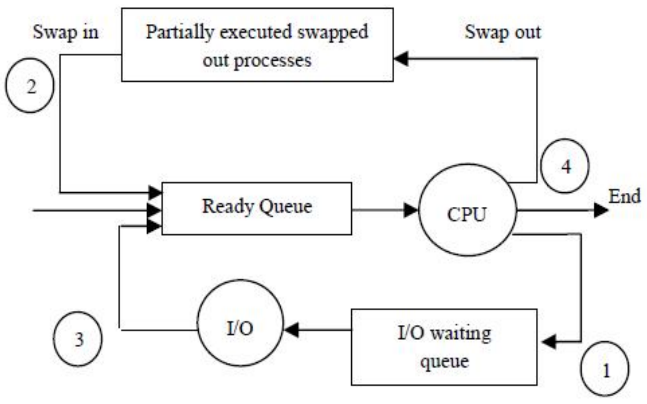


Figure 2. Process of Schedulers

II. SCHEDULING CRITERIA

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. 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**. The interval from the time of submission of a process to the time of completion is theturnaround 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.** Waiting time is the sum of the periods spent waiting in the ready queue.

- **Response time:** 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.

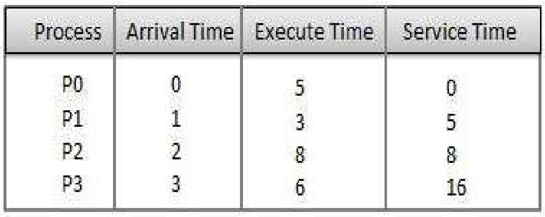
The turnaround time is generally limited by the speed of the output device.It is desirable to maximize CPU utilization and throughput and to minimize turnaround time, waiting time, and response time. In most cases, we optimize the average measure.

III. SCHEDULING ALGORITHM

First Come, First Served (FCFS)

It is the simplest CPU-scheduling algorithm. With this scheme, the process that requests the CPU first is allocated the CPU first. The implementation of the FCFS policy is easily managed with a FIFO queue. When a process enters the ready queue, its PCB is linked onto the tail of the queue. When the CPU is free, it is allocated to the process at the head of the queue. The running process is then removed from the queue. The code for FCFS scheduling is simple to write and understand. The average waiting time under the FCFS policy, however, is often quite long.

- Jobs are executed on first come, first served basis.
- It is a non-preemptive scheduling algorithm.
- Easy to understand and implement.
- Its implementation is based on FIFO queue.
- Poor in performance, as average wait time is high.



Process	Arrival Time	Execute Time	Service Time
P0	0	5	0
P1	1	3	5
P2	2	8	8
P3	3	6	16

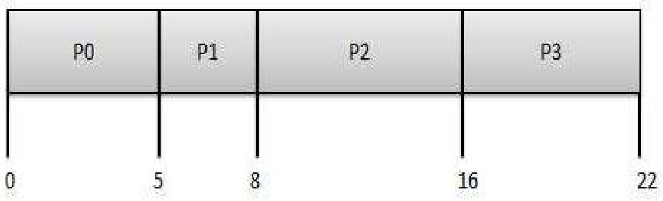
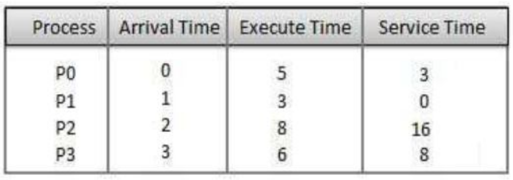


Figure 3.

Shortest Job Next (SJN)

This algorithm associates with each process the length of th eprocess's next CPU burst. When the CPU is available, it is assigned to the process that has the smallest next CPU burst. If the next CPU bursts of two processes are the same, FCFS scheduling is used to break the tie. Note that a more appropriate term for this scheduling method would be the shortest-next-CPU-burst algorithm, because scheduling depends on the length of the next CPU burst of a process, rather than its total length.

- This is also known as shortest job first, or SJF.
- This is a non-preemptive scheduling algorithm.
- Best approach to minimize waiting time.
- Easy to implement in Batch systems where required CPU time is known in advance.
- Impossible to implement in interactive systems where the required CPU time is not known.
- The processer should know in advance how much time a process will take.



Process	Arrival Time	Execute Time	Service Time
P0	0	5	3
P1	1	3	0
P2	2	8	16
P3	3	6	8

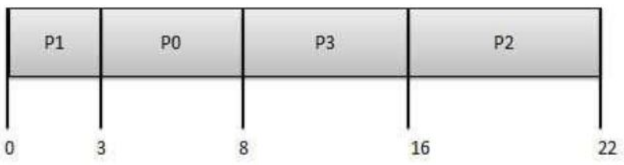
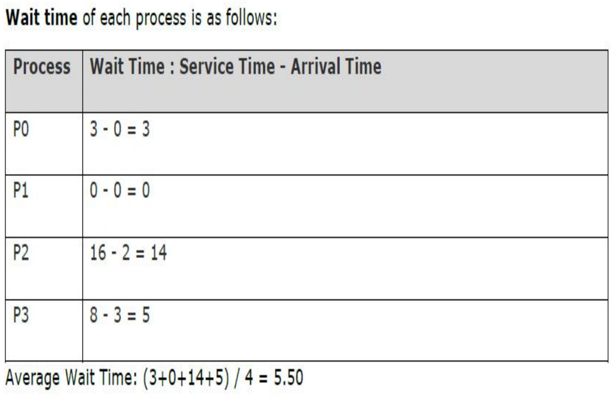


Figure 4.



Wait time of each process is as follows:

Process	Wait Time : Service Time - Arrival Time
P0	3 - 0 = 3
P1	0 - 0 = 0
P2	16 - 2 = 14
P3	8 - 3 = 5

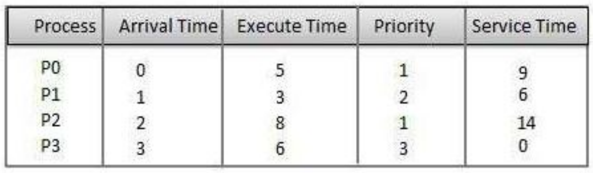
Average Wait Time: (3+0+14+5) / 4 = 5.50

Figure 5.

Priority Based Scheduling

The SJF algorithm is a special case of the general priority scheduling algorithm. A priority is associated with each process, and the CPU is allocated to the process with the highest priority. Equal-priority processes are scheduled in FCFS order. An SJF algorithm is simply a priority algorithm where the priority (p) is the inverse of the (predicted) next CPU burst. The larger the CPU burst, the lower the priority, and vice versa.

- Priority scheduling is a non-preemptive algorithm and one of the most common scheduling algorithms in batch systems.
- Each process is assigned a priority. Process with highest priority is to be executed first and so on.
- Processes with same priority are executed on first come first served basis.
- Priority can be decided based on memory requirements, time requirements or any other resource requirement.



Process	Arrival Time	Execute Time	Priority	Service Time
P0	0	5	1	9
P1	1	3	2	6
P2	2	8	1	14
P3	3	6	3	0

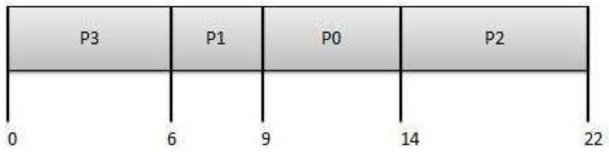
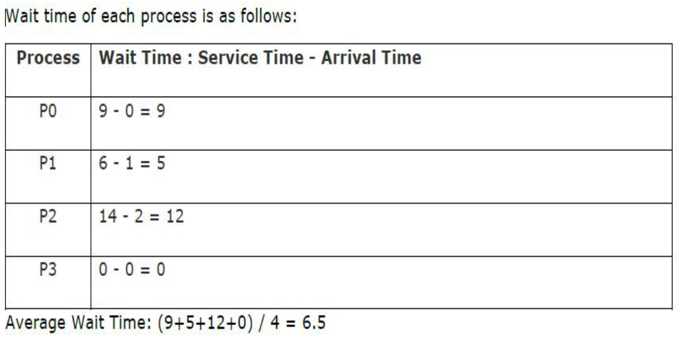


Figure 6.



Wait time of each process is as follows:

Process	Wait Time : Service Time - Arrival Time
P0	9 - 0 = 9
P1	6 - 1 = 5
P2	14 - 2 = 12
P3	0 - 0 = 0

Average Wait Time: (9+5+12+0) / 4 = 6.5

Figure 7.

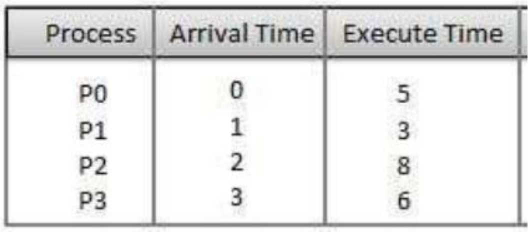
Shortest Remaining Time

- Shortest remaining time (SRT) is the preemptive version of the SJN algorithm.
- The processor is allocated to the job closest to completion but it can be preempted by a newer ready job with shorter time to completion.
- Impossible to implement in interactive systems where required CPU time is not known.
- It is often used in batch environments where short jobs need to be given preference.

Round Robin Scheduling

The round-robin (RR) scheduling algorithm is designed especially for timesharing systems. It is similar to FCFS scheduling, but preemption is added to switch between processes. A small unit of time, called a time quantum or time slice, is defined. A time quantum is generally from 10 to 100 milliseconds. The ready queue is treated as a circular queue. The CPU scheduler goes around the ready queue, allocating the CPU to each process for a time interval of up to 1 time quantum. To implement RR scheduling, we keep the ready queue as a FIFO queue of processes. New processes are added to the tail of the ready queue. The CPU scheduler picks the first process from the ready queue, sets a timer to interrupt after 1 time quantum, and dispatches the process. One of two things will then happen. The process may have a CPU burst of less than 1 time quantum. In this case, the process itself will release the CPU voluntarily. The scheduler will then proceed to the next process in the ready queue. Otherwise, if the CPU burst of the currently running process is longer than 1 time quantum, the timer will go off and will cause an interrupt to the operating system. A context switch will be executed, and 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. The average waiting time under the RR policy is often long.

- Round Robin is a preemptive process scheduling algorithm.
- Each process is provided a fix time to execute; it is called a quantum.
- Once a process is executed for a given time period, it is preempted and other process executes for a given time period.
- Context switching is used to save states of preempted processes.



Process	Arrival Time	Execute Time
P0	0	5
P1	1	3
P2	2	8
P3	3	6

Figure 8.

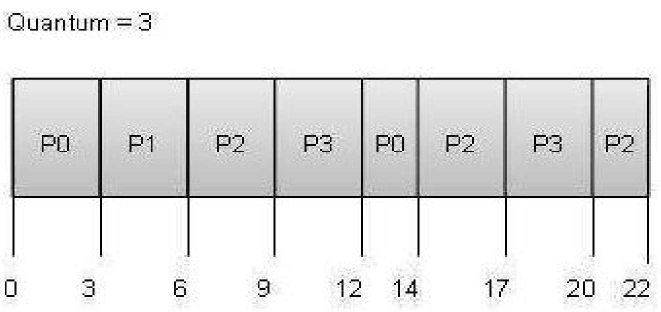
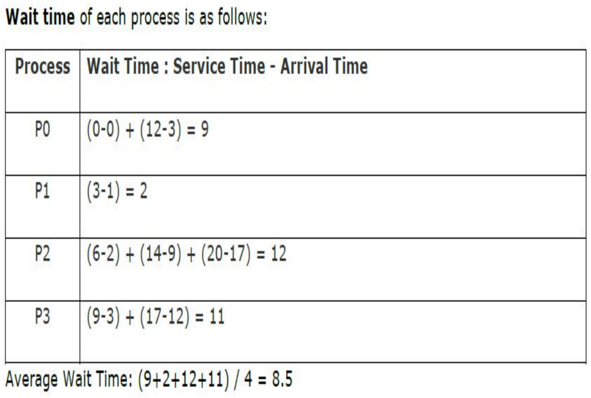


Figure 9.



Wait time of each process is as follows:

Process	Wait Time : Service Time - Arrival Time
P0	(0-0) + (12-3) = 9
P1	(3-1) = 2
P2	(6-2) + (14-9) + (20-17) = 12
P3	(9-3) + (17-12) = 11

Average Wait Time: (9+2+12+11) / 4 = 8.5

Figure 10.

From above analysis and discussion, we can say that the FCFS is simple to understand and suitable only for batch system where waiting time is largeThe FCFS scheduling algorithm is non-preemptive. Once the CPU has been allocated to a process, that process 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. The SJF scheduling algorithm is provably optimal, in that it gives the minimum average waiting time for a given set of processes. Moving a short process before a long one decreases the waiting time of the short process more than it increases the waiting time of the long process. Consequently, the average waiting time decreases. The priority scheduling algorithm is based on the priority in which the highest priority job can run first and the lowest priority job need to wait though it will create a problem of starvation. The round robin scheduling algorithm is preemptive which is based on round robin policy one of the scheduling algorithm which follows the interactive system and the round robin scheduling algorithm is deal with the time sharing system.

IV. CONCLUSIONS

The SJF scheduling algorithm is provably optimal, in that it gives the minimum average waiting time for a given set of processes. Moving a short process before long one decrease the waiting time of the short process more than it increases the waiting time of the long process. Consequently, the average waiting time decreases.

To get more accurate evaluation of scheduling algorithms, simulations are often used. But a simulation has limited accuracy. The only completely accurate way of evaluating a scheduling algorithm is to code and put it in the operating system to see how it works. This approach puts the actual algorithm in the real system for evaluation under real operating conditions.

REFERENCES

[1] Analysis and Comparison of CPU Scheduling Algorithms, Pushpraj Singh1, Vinod Singh2, Anjani Pandey, “Analysis and Comparison of CPU Scheduling Algorithms,” International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, ISO 9001:2008 Certified Journal, Volume 4, Issue 1, January 2014)

[2] Abraham Silberschatz, Peter Baer Galvin, Greg Gangne;” Operating System Concepts”, edition-7, ©2009, John Wiley and Sons, INC.

[3] ” Operating System, fundamental os concepts”, tutorials Pont,www.tutorialspoint.com

[4] James L. Peterson, Abraham Silerschatz, university of Texas at Austin ”Operating System concepts”©1983 by Addision-Wesley publishing Company,Inc.