IPASJ International Journal of Computer Science (IIJCS)
Web Site: http://www.ipasj.org/IIJCS/IIJCS.htm
A Publisher for Research Motivation ..... Email: editoriijcs@ipasj.org
Volume 2, Issue 11, November 2014 ISSN 2321-5992

Comparison of the Various CPU Scheduling Algorithms

Nikhil Selvaraj , Yeshwanth Raja , R. Ajay Adithya , Arjun Narendran

Undergraduate Student, Computer Science Department, Sri Venkateswara College of Engineering

ABSTRACT

Scheduling is a basic operating system function which decides the time and order in which the designated processes have to be executed. The CPU is the most important component of a computer which does the scheduling. CPU scheduling is done by switching the CPU among various processes. The main function of the operating system is to let as many processes as possible execute so that the best possible utilization of the CPU is achieved. In this paper, we review the basic scheduling algorithms and show which algorithm is best for a particular situation. We also modify a particular scheduling algorithm so that the performance in scheduling the processes is improved.

Keywords:- Starvation time, Gantt Chart, average waiting time.

1. INTRODUCTION

Scheduling is a fundamental operating-system function.Almost all computer resources are scheduled before use.The CPU is one of the primary computer resources. Thus,its scheduling is central of operating system design. Whenmore than one process is runnable, the OS must decidewhich one is to run first. That part of the OS concernedwith this decision is called scheduler and the algorithm ituses is called scheduling algorithm. A CPU scheduler is thepart of an operating system and responsible for arbitrating access to the CPU . A scheduler is an OS module thatselects the next job to be admitted into the system and thenext process to run.An operating system has three types of schedulers-longterm scheduler (also known as Job scheduler), mediumtermscheduler and short-term scheduler (also known asdispatcher and CPU scheduler).

1.1Types of schedulers

1.1.a) Long-term scheduler

The long-term scheduler (also known as admissionscheduler) decides which jobs or processes are to beadmitted to the ready queue (in the Main Memory); that is,when an attempt is made to execute a program, itsadmission to the set of currently executing processes iseither authorized or delayed by the long-term scheduler.Thus, this scheduler dictates what processes are to runon a system, and the degree of concurrency to be supportedat any one time - i.e.: whether a high or low amount ofprocesses are to be executed concurrently, and how thesplit between input output intensive and CPU intensiveprocesses is to be handled. So long term scheduler isresponsible for controlling the degree ofmultiprogramming. In modern operating systems, this isused to make sure that real time processes get enough CPUtime to finish their tasks. Without proper real timescheduling, modern GUIs would seem sluggish. The longterm queue exists in the Hard Disk or the "VirtualMemory"

1.1.b) Medium-term scheduler

The medium-term scheduler (also known as mid-termscheduler) may decide to swap out a process which has notbeen active for some time, or a process which has a lowpriority, or a process which is taking up a large amount ofmemory in order to free up main memory for otherprocesses, swapping the process back in later when morememory is available, or when the process has beenunblocked and is no longer waiting for a resource.

1.1.c) Short-term scheduler

The short-term scheduler (also known as CPUscheduler) selects from among the pool of process resident in memory that are ready to execute, and allocates the CPUto one of them. Thus the short-term scheduler makesscheduling decisions much more frequent than the longtermor mid-term schedulers. This scheduler can bepreemptive, implying that it is capable of forcibly removingprocesses from a CPU when it decides to allocate that CPUto another process, or non pre-emptive (also known as"voluntary" or "co-operative"), in that case the scheduler isunable to force processes off the CPU .

1.2 Basic terminologies used in CPU scheduling

CPU Time:

CPU time (or **process time**) is the amount of time for which the CPU was used for processing instructions of a computer program or operating system, as opposed to waiting for I/O operations or entering low-power (idle) mode.

The CPU time is measured in clock ticks or seconds. It is useful to measure CPU time as a percentage of the CPU's capacity, which is called the CPU usage. CPU time has two main uses. The first use is to quantify the overall busyness of the system. When the CPU usage is above 70%, the user may experience lag. The second use is to quantify how the processor is shared between computer programs. High CPU usage by a single program may indicate that it is highly demanding of processing power or that it may malfunction.
The CPU time can be divided into the following:

1. User time is the amount of time the CPU was busy executing code in user space.
2. System time is the amount of time the CPU was busy executing code in kernel space.
3. Idle time is the amount of time the CPU was not busy.
4. Steal time is the amount of time the operating system wanted to execute, but was not allowed to by the hypervisor.

Burst Time:

Burst time is the amount of time that a process uses the CPU for its execution. It can also be defined as the amount that a process requires between I/O waits.

Arrival Time:
Arrival time is the time at which the process is ready to be scheduled by the CPU for execution.
Waiting time:
Waiting time is the amount of time that a process has been waiting in the ready queue**.**
Throughput:
Throughput is the number of processes that complete their execution per unit time.

2. REVIEW OF EXISTING ALGORITHMS

2.1Types of scheduling algorithms
The various types of scheduling algorithms are:
1. First Come First Serve Scheduling
2. Shortest Job First Scheduling
3. Priority Scheduling

First Come First Serve Scheduling:
The most intuitive and simplest technique is to allow the first process submitted to run first. This approach is called as first-come, first-served (FCFS) scheduling. In effect, processes are inserted into the tail of a queue when they are submitted. The next process is taken from the head of the queue when each finishes running. The main advantage of this type of scheduling is that it is easy to understand and implement this algorithm. The main disadvantage is that its performance is poor as the average waiting time is high.
Shortest Job First Scheduling:
The process is allocated to the CPU which has least burst time. A scheduler arranges the processes with the least burst time in head of the queue and longest burst time in tail of the queue. This requires advanced knowledge or estimations about the time required for a process to complete. This algorithm is designed for maximum throughput in most scenarios. The main advantage of this type of scheduling is that the waiting time can be reduced. The disadvantages are that the processor should know in advance how much time each processwill take. Also the implementation is difficult.
Priority Scheduling:
The OS assigns a fixed priority rank to every process, and the scheduler arranges the processes in the ready queue in order of their priority. Lower priority processes get interrupted by incoming higher priority processes. Its main advantage is that it provides a good mechanism where the relative importance of each process may be well defined. Its disadvantage is that if the process with the higher priority takes up a large amount of time, then the lower priority processes' execution will be delayed.
2.2 Computation of Gantt chart, Waiting Time and Turnaround Time
Consider the following set of processes, with the length of the CPU-burst time in milliseconds. The Gantt charts for the FCFS and SJF algorithms below are based on table I. Table II gives the Gantt chart for priority scheduling algorithm

IPASJ International Journal of Computer Science (IIJCS)
Web Site: http://www.ipasj.org/IIJCS/IIJCS.htm
A Publisher for Research Motivation ..... Email: editoriijcs@ipasj.org
Volume 2, Issue 11, November 2014 ISSN 2321-5992

TABLE 1 PROCESS WITH ITS ID AND BURST TIME

Process ID	Burst Time (ms)
P_1	10
P_2	2
P_3	8
P_4	6

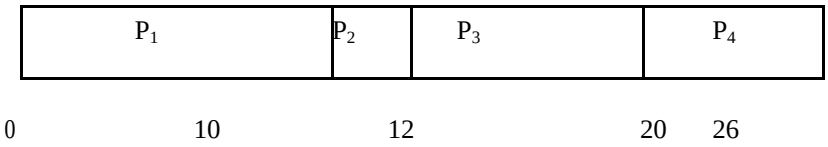
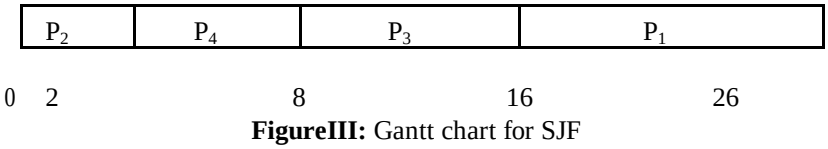


Figure II: Gantt chart for FCFS



FigureIII: Gantt chart for SJF

TABLE 2. PROCESS WITH ITS ID, BURST TIME AND PRIORITY

Process ID	Burst Time (ms)	Priority
P_1	10	3
P_2	2	1
P_3	8	4
P_4	6	2

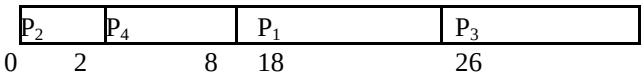


Figure V: Gantt chart for PS

For example, turnaround time for the process is calculated as time of submission of a process to the time of completion of the process is obtained through Gantt chart for SJF scheduling. Turnaround time for process P_1, P_2, P_3 and P_4 is observed as 26, 2, 16 & 8 and average turnaround time is (26+2+16+8)/4=13 ms. The waiting time for the process is calculated as time taken by the process to wait in the ready queue is observed from Gantt chart for SJF scheduling. Waiting time for process P_1, P_2, P_3 and P_4 is obtained as 16, 0, 8 & 2 respectively and average waiting time is (16+0+8+2)/4 = 6.5 ms. Similarly the turnaround time and waiting time is calculated for all other algorithms and summarized in Table III and Table IV. From the above discussion it is clear that First Come First Serve (FCFS) & Shortest Job First (SJF) is generally suitable for batch operating systems & Priority Scheduling (PS) is suitable for time sharing systems. SJF algorithm is optimum for all type of scheduling algorithm. Hence, it is an algorithm with an optimum criteria and suitable for all scenarios.

IPASJ International Journal of Computer Science (IIJCS)
Web Site: http://www.ipasj.org/IIJCS/IIJCS.htm
A Publisher for Research Motivation ..... Email: editoriijcs@ipasj.org
Volume 2, Issue 11, November 2014 ISSN 2321-5992

3.PROPOSED ALGORITHM

Improvement to Shortest Job First Algorithm:

In the normal shortest job first algorithm, the processes are executed in the increasing order of their burst times. If this happens, the process with the longest burst time will have to wait till all the other processes with shorter burst times get executed. Therefore, the starvation time will be more and the CPU utilization is not efficient. In order to avoid this, a significant improvement that can be made is to execute a long process after 60% of all the other shorter processes have finished execution. This will guarantee that the longest process gets executed at some time rather than the process's execution being delayed till the last. This in turn ensures effective CPU utilization.

Process ID	Burst Time (ms)
P_1	10
P_2	2
P_3	8
P_4	6

Figure III: Gantt Chart for SJf

From the above Gantt chart, the waiting times for processes P1, P2, P3 and P4 are 8, 0,18 and 2 respectively. The average waiting time is (0+2+8+18)/4=7. It is also clear that the finishing time reduces since here the finishing time is 24 ms while in the normal algorithm it is 26 ms.The starvation time reduces in this case since the waiting time of process P1 is only 8 ms while in the normal algorithm it is 16 ms.

4. RESULTS

The fundamental scheduling algorithm has been simulated and justified with C++ code. Comparison of the fundamental algorithm is shown in Figure VI and Figure VII.

i. Waiting Time

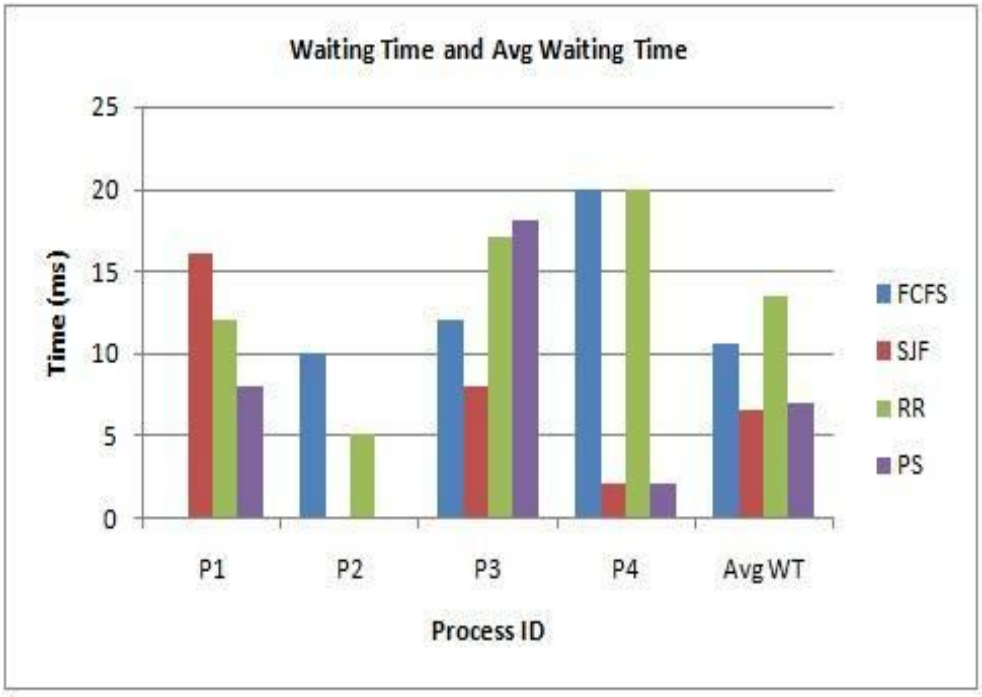


Figure VI: Comparison of Fundamental scheduling Algorithm-Waiting Time

IPASJ International Journal of Computer Science (IIJCS)
Web Site: http://www.ipasj.org/IIJCS/IIJCS.htm
A Publisher for Research Motivation ..... Email: editoriijcs@ipasj.org
Volume 2, Issue 11, November 2014 ISSN 2321-5992

ii. Turnaround Time

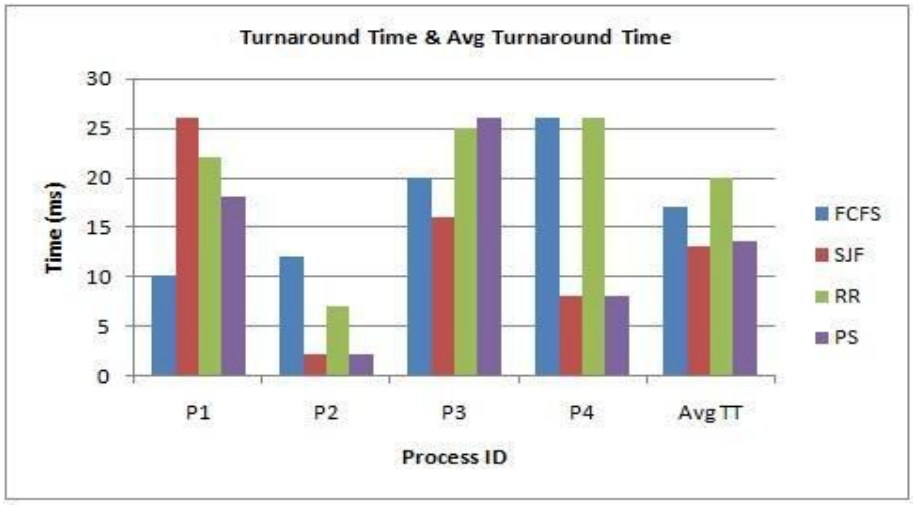


Figure VII: Comparison of Fundamental scheduling Algorithm-Turnaround Time

It is clearly observed that turnaround time, waiting time and response time of the processes are optimum for SJF scheduling algorithm compared to all other fundamental algorithms from Figure II, Figure. III, Fig. IV, Fig. V, (Figure II to Figure V are Gantt Chart) Figure VI and Figure VII. It can also be observed that throughput and CPU utilization rate are optimum. 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 large. SJF scheduling algorithm gives minimum average waiting time and average turnaround time. 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.

5. CONCLUSIONS

The SJF scheduling algorithm is to serve all types of job with optimum scheduling criteria. The treatment of shortest process in SJF scheduling tends to result in increased waiting time for long processes. And the long process will never get served, though it produces minimum average waiting time and average turnaround time. The shortest job first scheduling algorithm deals with different approach, in this algorithm the major benefit is it gives the minimum average waiting time. The drawbacks of the shortest job first scheduling algorithm can be overcome by the technique proposed above in which after 60% of all the processes have executed, the longest process gets executed. The only way to evaluate a scheduling algorithm to code it and has to put it in the operating system, only then a proper working capability of the algorithm can be measured in real time systems.

REFERENCES

[1] Abraham Silberschatz, Peter Baer Galvin, Greg Gangne;" Operating System Concepts", edition-6, 2002, John Wiley and Sons, INC.
[2] Mr. Sindhu M, Mr. Rajkamal R and Mr. Vigneshwaran P, "An Optimum Multilevel CPU Scheduling Algorithm". 978-0-7695-4058-0/10 $26.00 © 2010 IEEE
[3] Mr. Umar Saleem and Dr. Muhammad YounusJaved, "Simulation of CPU Scheduling Algorithm", 0-7803-6355-8/00/ $10.00 @ 2000 IEEE.
[4] Sun Huajin', GaoDeyuan, Zhang Shengbing, Wang Danghui; "Design Fast Round Robin Scheduler in FPGA", 0-7803-7547- 5/021/$17.00 @ 2002 IEEE.
[5] Md. Mamunur Rashid and Md. NasimAdhtar; "A New Multilevel CPU Scheduling Algorithm", Journals of Applied Sciences 6 (9): 2036-2039, 2009.
[6] Mr. Sindhu M, Mr. Rajkamal R and Mr. Vigneshwaran P, "An Optimum Multilevel CPU Scheduling Algorithm". 978-0-7695- 4058-0/10 $26.00 © 2010 IEEE
[7] Black, D.L., 1985. Scheduling support for concurrency and parallelism in the mach operating system. Computer, 23:123-126.
[8] AnkurBhardwas, Rachhpal Singh, Gaurav, "Comparative Study of Scheduling Algorithm in Operating System" International Journal of Computer and Distributed Systems, Vol. No. 3, Issue I, April-May