Review And Analysis of CPU Scheduling Algorithms

Amandeep Singh
Assistant professor,
DCA , CBSA,
Landran, Mohali

Abstract: Developing CPU scheduling algorithms and understanding their impact in practice can be difficult and time consuming due to the need to modify and test operating system kernel code and measure the resulting performance on a consistent workload of real applications. As processor is the important resource, CPU scheduling becomes very important in accomplishing the operating system (OS) design goals. The intention should be allowed as many as possible running processes at all time in order to make best use of CPU.

Keywords: Scheduler, State Diagrams, CPU-Scheduling, Performance

Introduction

Scheduling is one of the basic functions of any operating system, because scheduling of all the computer resources is done before their use. The CPU the most essential computer resource. Therefore its scheduling algorithm is a very important part of the OS design. When multiple processes are runnable, the OS has the onus of responsibility to decide which one is to run first. The part of the OS that takes this decision is called scheduler and the algorithm it works on is called scheduling algorithm. A CPU scheduler is a part of an operating system and is responsible for mediating access to the CPU. OS may have up to three types of schedulers: a long term scheduler (also known as an admission scheduler or high level scheduler), a medium-term scheduler and a short-term scheduler (also known as a dispatcher or CPU scheduler).

Long-Term Scheduler

The long-term or admission scheduler decides which jobs or processes are to be admitted to the ready queue; that is, when an attempt is made to execute a process its admission to the set of currently executing processes is either authorized or delayed by the long-term scheduler. Thus, this scheduler dictates what processes are to run on a system, and the degree of concurrency to be supported at any one time.

Mid-Term Scheduler

The mid-term scheduler temporarily removes process from main memory and place them on secondary memory (such as a disk drive) or vice versa. This is commonly referred to as —swapping of processes out" or "swapping in" (also incorrectly as "paging out" or "paging in").\

Short-Term Scheduler

The short-term scheduler (also known as the CPU scheduler) decides which of processes in the ready queue, in memory are to be executed (allocated a CPU) next following a clock interrupt, an Input-Output (IO) interrupt and an OS call or another form of signal. Therefore the short-term

scheduler makes scheduling decisions much more frequently than the long-term or midterm schedulers. This scheduler can be preemptive, meaning that it can forcibly remove processes from a CPU i.e. it can allocate the CPU (allocated to current process) to another process, or non-preemptive (also known as "voluntary" or "co-operative"), in that case the scheduler is unable to force processes off the CPU. The success of a CPU scheduler depends highly on the design of good quality scheduling algorithm. Good-quality CPU scheduling algorithms depends mainly on criteria such as response time, throughput, CPU utilization rate, waiting time, turnaround time and. Thus, the main focus of this proposed work is to develop a generalized optimum good quality scheduling algorithm suited for all types of jobs.

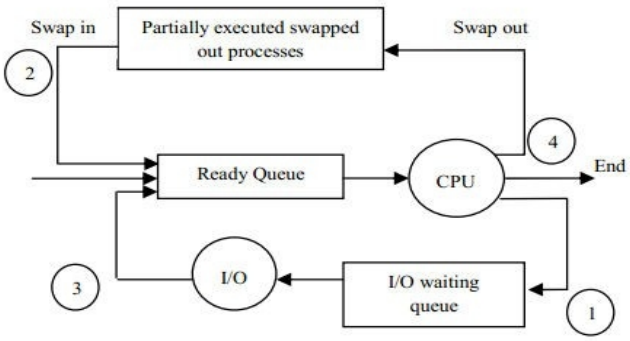


Fig. 1. Process of Schedulers

Fig. 1. Shows the following states have been executed in the

CPU Scheduler.
1. When a process switches from the running state to the waiting state.
2. When a process switches from the running state to the ready state.
3. When a process switches from the waiting state to the ready state.
4. When a process terminates.
The success of a CPU scheduler depends highly on the design of good quality scheduling algorithm. Good-quality CPU scheduling algorithms depends mainly on criteria such as response time, throughput, CPU utilization rate, waiting time,
turnaround time and. Thus, the main focus of this proposed work is to develop a generalized optimum good quality scheduling algorithm suited for all types of jobs.

Scheduling Parameters
a) **CPU Utilization**: It is the average fraction of time, during which the processor is busy.

b) **Throughput**: It refers to the amount of work completed in a unit of time. The number of processes the system can execute in a period of time. The higher the number, the more work is done by the system.

c) **Waiting Time:** The average period of time a process spends waiting. Waiting time may be expressed as turnaround time less the actual execution time.

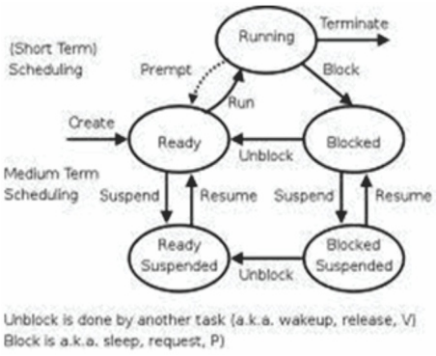
d) **Turnaround time**: The interval from the time of submission of a process to the time of completion is the turnaround time.

e) **Response time:** Response time is the time from submission of a request until the first response is produced.
f) **Priority:** give preferential treatment to processes with
higher priorities.
g) **Fairness:** Avoid the process from starvation. All the processes must be given equal opportunity to execute.

Overview of Existing CPU Scheduling Algorithms
First Come First Served (FCFS) Scheduling
It is the simplest CPU Scheduling algorithm. The criteria of this algorithm are the process that requests first, hold the CPU first "or which process enter the ready queue first is served first. The workload is processed in the order of arrival time, with no preemption. Once a process has been submitted to the CPU, it runs into completion without being interrupted. Such a technique is fair in the case of smaller processes but is quite unfair for long an unimportant job. Since FCFS does not involve context switching therefore it has minimal overhead. It has low throughput since long processes can keep processor occupied for a long time making small processes suffer. As a result waiting time, turnaround time and response time can be low.

Shortest Job First (SJF) Scheduling
The criteria of this algorithm are which process having the smallest CPU burst, CPU is assigned to that process next. If two process having the same CPU burst time FCFS is used to break up the tie . SJF can be worked as preemptive and non– preemptive in nature based on the arrival time and burst time of the processes. SJF reduces average waiting time of the processes as compared to FCFS. SJF favors shorter processes over longer ones which is an overhead as compared to FCFS. It selects the job with the smallest burst time ensuing CPU availability for other processes as soon as the current process reaches its completion. This prevents smaller processes from suffering behind larger processes in the ready queue for a longer time.

Priority Based Scheduling
In this algorithm, priority is associated with each process and on the basis of that priority CPU is allocated to the processes. Higher priority processes are executed first and lower priority processes are executed at the end. If multiple Processes having the same priorities are ready to execute, control of CPU is assigned to these processes on the basis of FCFS . Priority Scheduling can be preemptive and non-preemptive in nature.
Round Robin (RR) Scheduling
It is a preemptive scheduling algorithm. It is designed especially for time sharing systems. In this algorithm, a small unit of time called time quantum or time slice is assigned to each process . When the time quantum expired, the CPU is switched to another process. Performance of Round Robin totally depends on the size of the time quantum.



Multilevel Queue Scheduling

Our first composite algorithm: partition processes into different queues, each with its own scheduling algorithm (as appropriate for that queue) Canonical example:
interactive processes use RR in one queue; batch processes use FCFS in another. Now of course we have to schedule among queues:

Priority Scheduling

Queues have preset priorities RR scheduling; each queue is given some quantum during which its processes do work

Multilevel Feedback-Queue Scheduling

Multilevel queues plus the ability for a process to move to another queue Lots of parameters to play with: number of queues, scheduling algorithms per queue, queue selection rules. For example, track CPU burst times and move CPU bound processes to a lower -priority queue; vice versa for I/O-bound processes.

Multiple-Processor Scheduling

Focus on homogeneous multiple processors: allows any available processor to run any available process. Two approaches most modern OS's do SMP:

Conclusion

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. It is recommended that any kind of simulation for any CPU scheduling algorithm has limited accuracy.

Future Scope

In this paper we analyze that the process of scheduling and the different scheduling algorithms are reviewed. 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] Md. Mamunur Rashid and Md. Nasim Adhtar, — A New Multilevel CPU Scheduling Algorithm, Journals of Applied Sciences 6 (9): 2036-2039, 2009

[2] Silberschatz, A. P.B. Galvin and G. Gagne (2012), Operating System concepts, 8th edition, Wiley India,

[3] Sabrian, F., C.D. Nguyen, S. Jha, D. Platt and F.Safaei, (2005). Processing resource scheduling inprogrammable networks.Paper ID: J2013200 133 of 134 ISSN (Online): 2347-3878 Volume 2 Issue 3, March 2014

[4] Umar Saleem and Muhammad Younus Javed, Simulation of CPU Scheduling Alogrithm, 0-7803-6355-8/00/IEEE

[5] Sun Huajin', Gao Deyuan, Zhang Shengbing, Wang Danghui, "Design fast Round Robin Scheduler in FPGA‖, 0-7803-7547-5/021/2002 IEEE