OPERATING SYSTEM

OS process definition and management

Student name	ID
Mahmoud Essam Fathy	20221460231
Abdelrahman Ashraf Ragab	20221374041
Abdullah Hussein Ibrahim	20221427861
Zyad Ashraf Hafez	20221374025
Marwan Ali Abd-Elsatar	20221460240

Supervised by

Dr. Yasser Fouad

Contents

FreeBSD

Process? Processor? What are those things?

Let's make it easy, can a human do any action without his brain being awake to send any action to his body Or even sense? That's our **CPU Processor** work, The CPU works here as if it's the brain of the PC to know each Action when happens and what to do.
So, you can say **a program does nothing unless its instructions are executed by a CPU.**

What are the needs of a simple process?

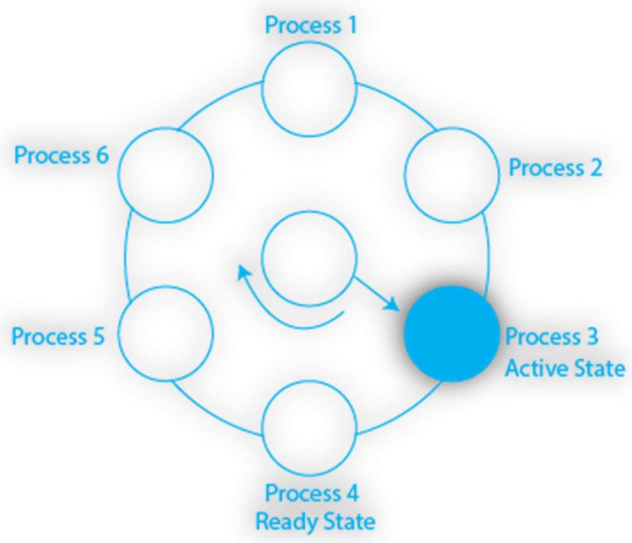
A process needs certain resources:

- **CPU Time:** for conducting which one should be first
- **Memory:** you know, the place where we can store this process in her queue.
- **Files**
- **I/O Devices**

And all those for completing the needed task.

Does that mean only 1 process only for 1 task?

Look, the advantage of using **"multiprocessing"** as we talked about is **performing a lot of tasks at the same time** without the need to wait for something till the first one is done. how was something like that performed?



How time-sharing works?

sharing of a computing resource among many tasks or users. It enables multi-tasking by a single user or enables multiple user sessions.

Is there a simple example about it?

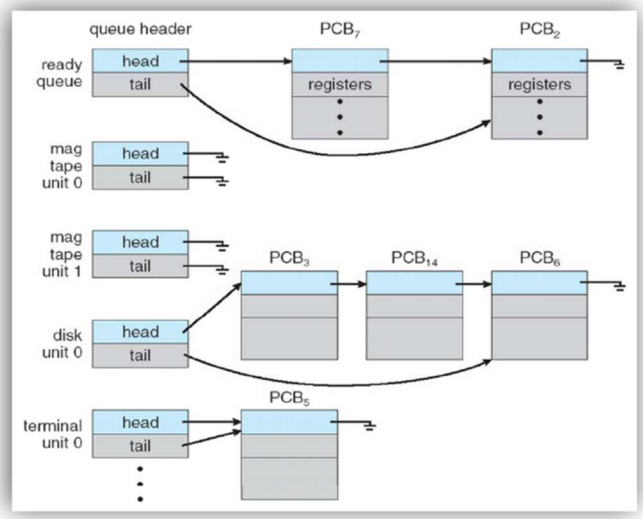
The user may have started a video editing program and instructed it to convert a one-hour video to a certain format (something that can take hours) and then gone off to surf the Web. Meanwhile, a background process that wakes up periodically to check for incoming emails may have started running. Thus, we have (at least) three active processes: the video editor, the Web browser, and the email receiver. Periodically, the operating system decides to stop running one process and start running another, perhaps because the first one has used up more than its share of CPU time in the past second or two

After getting suspended temporarily, it must later be restarted in exactly the same state it had when it was stopped. This means that all information about the process must be explicitly saved somewhere during the suspension

How it managed well?

The operating system is responsible for the following activities in connection with Process Management:

1. Scheduling processes and threads on the CPUs.
2. Create a child's process identical to the parents.
3. Terminate a process
4. Wait for a child process to terminate
5. Change the priority of the process
6. Block the process
7. Ready the process
8. Dispatch a process
9. Suspend a process
10. Resume a process
11. Delay a process
12. Fork a process

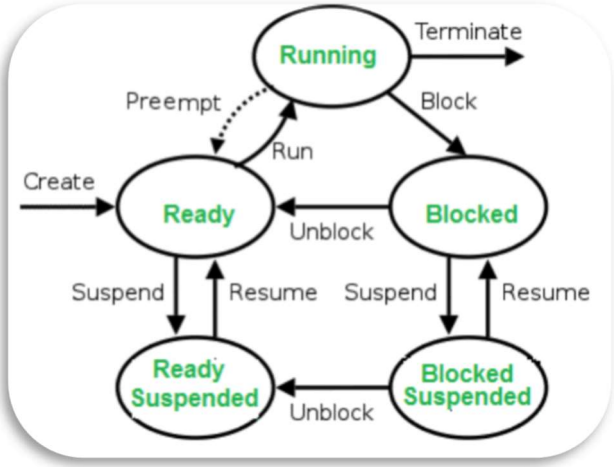


Some resources may need to be executed by one process at one time to maintain the consistency otherwise the system can become inconsistent and deadlock may occur.

States of Process:

A process is in one of the following states:

1. New: Newly Created Process (or) being-created process.
2. Ready: After the creation process moves to the Ready state, i.e. the process is ready for execution.
3. Run: Currently running process in CPU (only one process at a time can be under execution in a single processor)
4. Wait (or Block): When a process requests I/O access.
5. Complete (or Terminated): The process completed its execution.
6. Suspended Ready: When the ready queue becomes full, some processes are moved to a suspended ready state
7. Suspended Block: When the waiting queue becomes full.



Context Switching:

The process of saving the context of one process and loading the context of another process is known as Context Switching. In simple terms, it is like loading and unloading the process from the running state to the ready state.

When Does Context Switching Happen?

1. When a high-priority process comes to a ready state (i.e. with higher priority than the running process)
2. An Interrupt occurs
3. User and kernel-mode switch (It is not necessary though)
4. Preemptive CPU scheduling is used.

OS Scheduling System

What is CPU Scheduling?

- It is the procedure that involves choosing which process will use the CPU to execute while another is put on hold.
- CPU scheduling's primary responsibility is to ensure that, if the CPU is idle, The operating system at least chooses one of the tasks that are waiting to be executed from the ready queue.
- The CPU scheduler will handle the selection procedure, It chooses a process from among those that are available for execution in memory.
- It ensures that CPU utilization is maximized so that the computer is more productive.

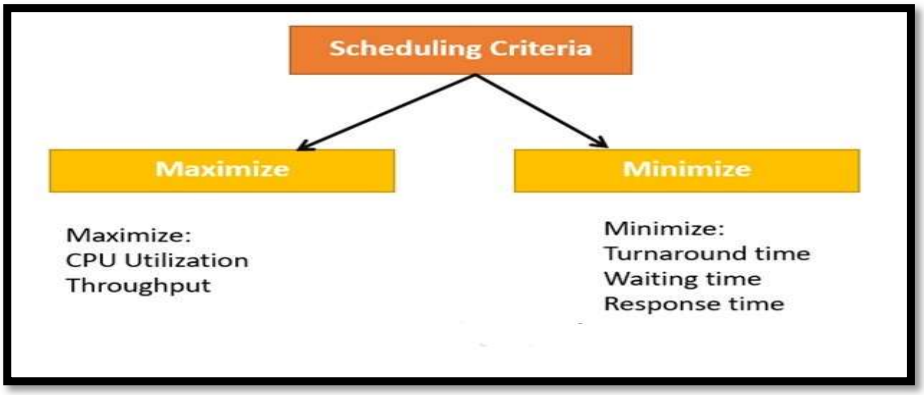
Process Scheduling: This is the procedure by which the process management chooses a different process based on a predetermined strategy and removes an active process from the CPU.

❋ Process scheduling is an essential part of a Multiprogramming operating system ❋

What is the need for a CPU scheduling algorithm?

- The process of choosing which process will use the CPU while another is suspended is known as CPU scheduling
- The primary purpose of CPU scheduling is to guarantee that the operating system has at least chosen a process from the ready-to-use line whenever the CPU is idle.

If most operating systems change their status from performance to waiting then there may always be a chance of failure in the system. So, in order to minimize this excess, the OS needs to schedule tasks in order to make full use of the CPU and avoid the possibility of deadlock.



Maximize

CPU utilization: The primary function for which the operating system must ensure that the CPU is kept as busy as feasible is CPU usage. It can have a value between 0% and 100%. On the other hand, it might vary from 40 percent for low-level systems to 90 percent for high-level systems in the RTOS.

Throughput: is the number of processes that complete their execution in a given amount of time. Thus, work is done when the CPU is occupied with carrying out the operation; the amount of work finished in a certain amount of time is referred to as throughput.

Minimize

Turnaround Time: Turnaround time is the length of time needed to complete a particular task. It is the computation of the total amount of time spent in line, waiting to enter memory, and using the CPU to execute. The turnaround time is the amount of time that passes between the process submission time and the completion time.

Waiting time: The waiting time is the amount of time a specific process needs to wait in the ready queue.

Response time: It is the amount of time in which the request was submitted until the first response is produced.

What are the different terminologies to take care of in any CPU Scheduling algorithm?

Arrival Time: Time at which the process arrives in the ready queue.

Completion Time: Time at which process completes its execution.

Burst Time: Time required by a process for CPU execution.

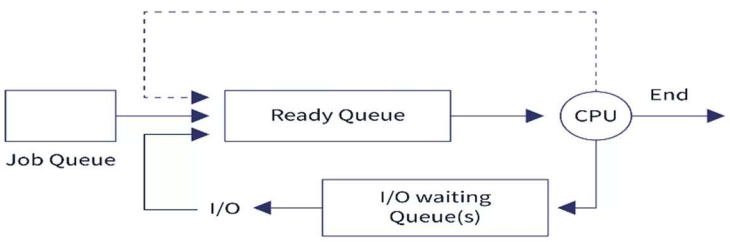
Turn Around Time: Time Difference between completion time and arrival time.

Turn Around Time = Completion Time – Arrival Time

Waiting Time = Turn Around Time – Burst Time

Things to take care of while designing a CPU Scheduling algorithm?

The structures of various CPU scheduling algorithms vary, and a number of factors influence the decision when selecting an algorithm. There are numerous requirements to compare CPU scheduling algorithms.



Categories in Scheduling

- **Non-preemptive:** A process's resource cannot be taken before the process has finished running,
  When a running process finishes and transitions to a waiting state, resources are switched.
- **Preemptive:** OS assigns resources to a process for a predetermined period of time, The process switches from running state to ready state or from waiting for state to ready state during resource allocation.

Types of Scheduling?

❋There are 3 main types of scheduling: -

1. Long Term (Job Scheduler)

- Brings the new process to the "Ready State".
- Controls the Degree of Multi-programming.

Multi-programming: describes the maximum number of processes that a single-processor system can accommodate efficiently.

- If Long Term scheduling is used, you should be careful while selecting I/O and CPU-bound processes.
- Job Scheduler Increases efficiency by maintaining a balance between both I/O and CPU-bound processes.

2. Short-Term (CPU Scheduler)

- Responsible for selecting one process from the ready state for scheduling it on the running state.
- Only selects the process to schedule it doesn't load the process on running.
- Responsible for ensuring no starvation due to high burst time processes.
- The dispatcher is responsible for loading the process selected by the Short-term scheduler on the CPU.

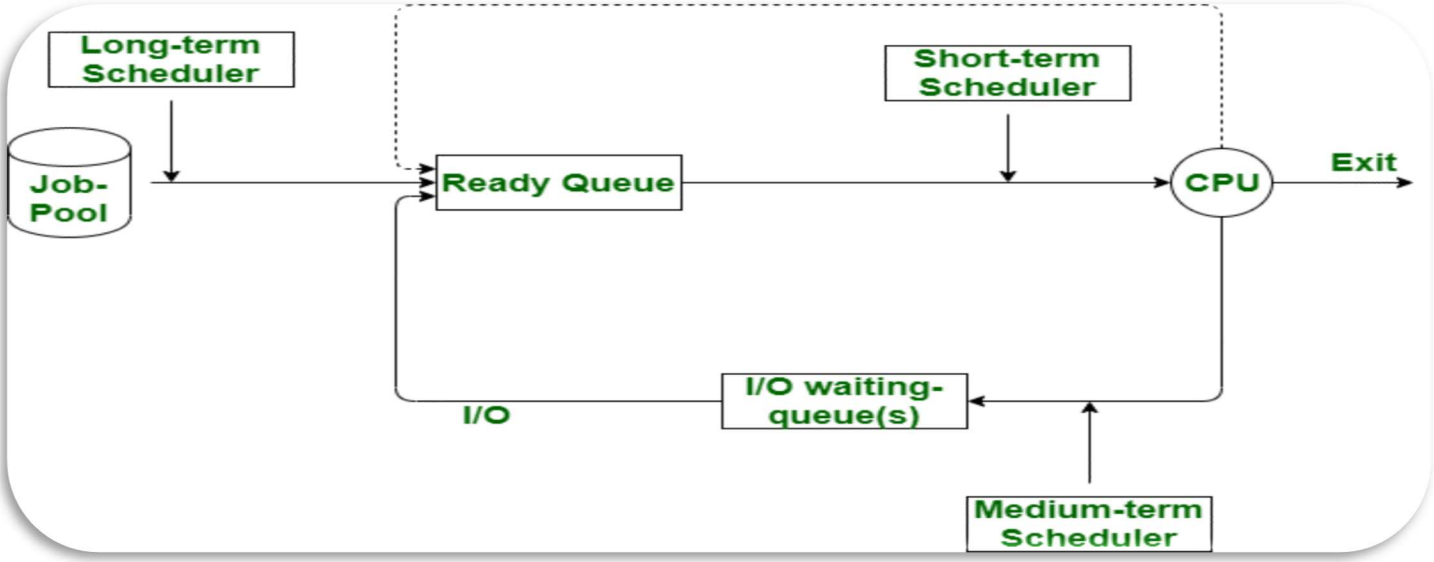
The dispatcher: Module that provides the control of the CPU to that process that gets selected by the short term-scheduler.

❖ **A Dispatcher Does Following:**

1. Switching context.
2. Switching to user mode.
3. Jumping to the proper location in the newly loaded program.

3. Medium-Term Scheduler

- Responsible for suspending and resuming the process.
- It mainly does swap (moving processes from main memory to disk and vice versa).
- Swapping may be necessary to improve the process mix or because a change in memory requirements has overcommitted available memory.
- It is helpful in maintaining a perfect balance between the I/O bound and the CPU bound.
- It reduces the degree of multiprogramming.



Most Popular Type of Schedular Algorithms?

There are many different algorithms for scheduling computer programs, and we are going to talk about some of them:

1. First Come, First Served (FCFS): Whichever program is added to the queue first is run until it finishes. Then the next program in the queue is run, and so on, yep it's fair but so bad at the performance (Non-preemptive)
2. Shortest Job First (SJF): It picks the program that will take the shortest amount of time as the one to run next. (Non preemptive)

3. Shortest Remaining Time(SRT): This is similar to Shortest Job Next, except that if a new program is started, the OS compares the time it needs with the time the currently running program has left. If the new program finishes sooner, then the currently running program is switched out and the CPU starts processing the new program. (preemptive)
4. Round Robin (RR): Time on the CPU is divided into equal parts called "time slices". Time slices are allocated to each program equally and cyclically. This means that if we had a list of three programs running, would repeat this order until one of the programs finished.

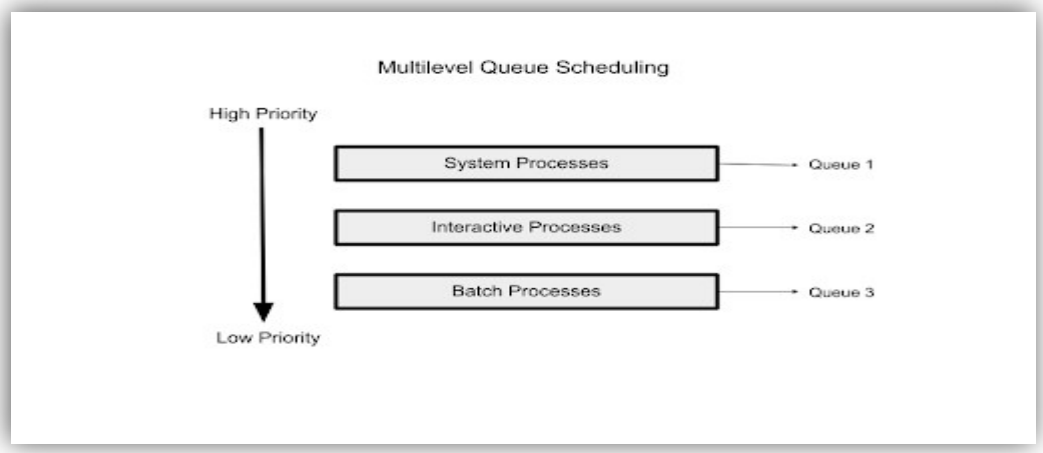
FreeBSD and Used Algorithm

Let's first discuss What's Multilevel Queue in detail then discuss why FreeBSD uses it.

A multilevel feedback queue (MLFQ) is a CPU scheduling algorithm that uses multiple queues with varying priorities to manage process execution. It dynamically adjusts priorities based on process behavior, promoting or demoting processes between queues.

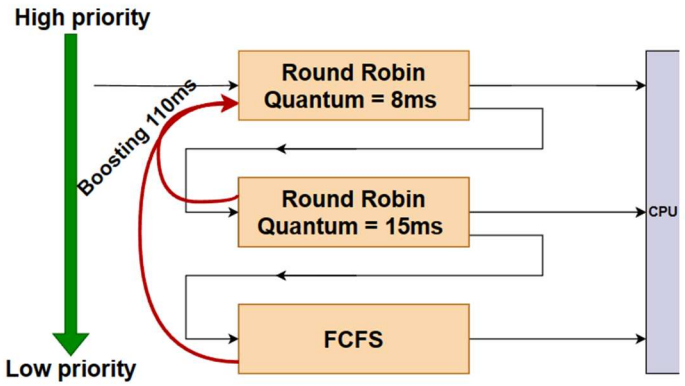
In General, programs are split into different queues by type — for example, system programs, or interactive programs. The programs of each type form a "queue".

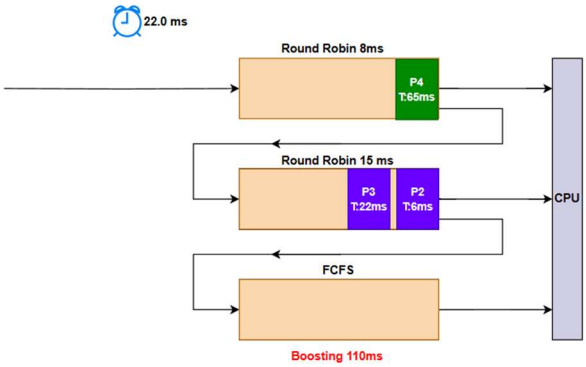
For each queue, a different queue scheduling algorithm will decide how the CPU time is split within that queue. For example, one queue may use round-robin scheduling, while another uses priority scheduling.



So how does FreeBSD Work with it?

After searching about general MLFQ we found that the most popular one was the mixture between RR and FCFS, which makes FreeBSD (Preemptive), by making Round-Robin works by priority, but with quantum time, by using it make great separation inside the process, by that will make the usual apps which system open always open as fast as it can to run, while non-usual apps like antivirus may take long time, so we put it inside FCFS queue and leaving it take its time, as it will by any chance will done as all queues are done.





It's in general Flexible scheduling and facilitates the movement of processes across multiple queues, but really too complex, and the Transition between different queues results in increased CPU overheads, to choose the optimal scheduler, alternative methods are necessary to determine the values.

References

1. Abraham Silberschatz-Operating System Concepts (9th,2012_12) book
2. Andrew Tanenbaum-Modern Operating Systems book
3. https://en.wikipedia.org/wiki/Time-sharing
4. https://www.javatpoint.com/os-process-schedulers
5. https://www.geeksforgeeks.org/introduction-of-process-management/
6. https://www.geeksforgeeks.org/cpu-scheduling-in-operating-systems/
7. https://www.turing.com/kb/different-types-of-non-preemptive-cpu-scheduling-algorithms
8. https://www.guru99.com/cpu-scheduling-algorithms.html
9. Difference between dispatcher and scheduler
10. Degree of multiprogramming
11. https://www.futurelearn.com/info/courses/computer-systems/0/steps/53513
12. https://www.cse.scu.edu/~m1wang/projects/Schedule_multiFeedbackQueue_10s.pdf
13. https://www.educative.io/answers/what-is-multilevel-feedback-queue-scheduling
14. https://queue.acm.org/detail.cfm?id=1035622