CHAPTER 3 : RESOURCE MANAGEMENT

Processor Management

Introduction

- **Scheduling** is a key concept in
  - → computer multitasking
  - → multiprocessing operating system
  - → real-time operating system design.

- Computer multitasking - **multitasking** is a method by which multiple tasks, also known as processes, share common processing resources such as a CPU.
- Multiprocessing - **Multiprocessing** is the use of two or more central processing units (CPUs) within a single computer system.
- A **Real-Time Operating System** (**RTOS**) is a multitasking operating system intended for real-time applications.

- In typical designs, a task has three states: 1) running, 2) ready, 3) blocked.
- Most tasks are blocked, most of the time. Only one task per CPU is running.
- In simpler systems, the ready list is usually short, two or three tasks at most.

- In modern operating systems, there are typically many more processes running than there are CPUs available to run them.
- **Scheduling** refers to the way processes are assigned to run on the available CPUs.
- This assignment is carried out by software known as a **scheduler**.

- The **scheduler** is concerned mainly with:

  → CPU utilization - to keep the CPU as busy as possible.

  → Throughput - number of process that complete their execution per time unit.

  → Turnaround - total time between submission of a process and its completion.

  → Waiting time - amount of time a process has been waiting in the ready queue.

  → Response time - amount of time it takes from when a request was submitted until the first response is produced.

  → Fairness - Equal CPU time to each thread.

Type Of Scheduling Process

- Long-term scheduling
- Medium-term scheduling
- Short-term scheduling

Long-term Scheduling

- The long-term, or admission, scheduler decides which jobs or processes are to be admitted to the ready queue.
- When an attempt is made to execute a program, 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

Medium-term Scheduling

- The mid-term scheduler may decide to swap out a process which :

  → has not been active for some time, or a process which has a low priority,

  → or a process which is page faulting frequently,

  → or a process which is taking up a large amount of memory in order to free up main memory for other processes, swapping the process back in later when more memory is available,

  → or when the process has been unblocked and is no longer waiting for a resource.

Short-term Scheduling

- Also known as the dispatcher.
- Decides which of the ready, in-memory processes are to be executed (allocated a CPU) next following a clock interrupt, an IO interrupt, an operating system call or another form of signal.
- Makes scheduling decisions much more frequently than the long-term or mid-term schedulers

Scheduling Algorithms

- In computer science, a **scheduling algorithm** is the method by which threads, processes or data flows are given access to system resources.
- Example: processor time, communications bandwidth.
- This is usually done to load balance a system effectively or achieve a target quality of service.

- Scheduling algorithms:
  - → first in first out (FIFO)
  - → round robin scheduling
  - → shortest remaining time
  - → shortest remaining time
  - → priority
  - → multilevel queue

First In First Out (FIFO)

- An abstraction in ways of organizing and manipulation of data relative to time and prioritization.
- This expression describes the principle of a queue processing technique or servicing conflicting demands by ordering process by first-come, first-served (FCFS) behaviour: what comes in first is handled first,

"what comes in next waits until the first is finished, etc."

- Gives every process CPU time in the order they come.
- The way data stored in a queue is processed.
- Each item in the queue is stored in a queue (*simpliciter*) data structure.
- The first data to be added to the queue will be the first data to be removed, then processing proceeds sequentially in the same order.

Round Robin Scheduling

- **Round-robin** (RR) is one of the simplest scheduling algorithms for processes in an operating system.
- Which assigns time slices to each process in equal portions and in circular order, handling all processes without priority.
- Round-robin scheduling is both simple and easy to implement, and starvation-free.
- Round-robin scheduling can also be applied to other scheduling problems, such as data packet scheduling in computer networks.

- **RR process**

  - Round-robin job scheduling may not be desirable if the size of the jobs or tasks are strongly varying.

  - A process that produces large jobs would be favoured over other processes.

  - This problem may be solved by time-sharing

  - Example, by giving each job a time slot or *quantum* (its allowance of CPU time), and interrupt the job if it is not completed by then.

  - The job is resumed next time a time slot is assigned to that process.

Example 1 :

The time slot could be 100 milliseconds. If a *job1* takes a total time of 250ms to complete, the round-robin scheduler will suspend the job after 100ms and give other jobs their time on the CPU. Once the other jobs have had their equal share (100ms each), job1 will get another allocation of CPU time and the cycle will repeat. This process continues until the job finishes and needs no more time on the CPU.

- **Job1 = Total time to complete 250ms (quantum 100ms)**.
  1. First allocation = 100ms.
  2. Second allocation = 100ms.
  3. Third allocation = 100ms but job1 self-terminates after 50ms.
  4. Total CPU time of job1 = 250ms.

Shortest Job First

- Also known as ***Shortest Job Next*** (**SJN**), ***Shortest Process Next*** (**SPN).**
- Is a scheduling policy that selects the waiting process with the smallest execution time to execute next.
- Shortest job next is advantageous because of its simplicity and because it maximizes process throughput.

- (in terms of the number of processes run to completion in a given amount of time).
- It also minimizes the average amount of time each process has to wait until its execution is complete.
- However, it has the potential for process starvation for processes which will require a long time to complete if short processes are continually added.

- Shortest job next scheduling is rarely used outside of specialized environments because it requires accurate estimations of the runtime of all processes that are waiting to execute.

Shortest Remaining Time

- **Shortest remaining time** is a method of CPU scheduling that is a preemptive [defensive] version of shortest job next scheduling.
- In this scheduling algorithm, the process with the smallest amount of time remaining until completion is selected to execute.

- *Since the currently executing process is the one with the shortest amount of time remaining, and since that time should only reduce as execution progresses, processes will always run until they complete or a new process is added that requires a smaller amount of time.*

- Shortest remaining time is advantageous because short processes are handled very quickly.
- However, it has the potential for process starvation for processes which will require a long time to complete if short processes are continually added, though this threat can be minimal when process times follow a heavy-tailed distribution.

- Shortest remaining time scheduling is rarely used outside of specialized environments because it requires accurate estimations of the runtime of all processes that are waiting to execute.

Deadlock

- **Deadlock** refers to a specific condition when two or more processes are each waiting for the other to release a resource, or more than two processes are waiting for resources in a circular chain

- Deadlocks are particularly troubling because there is no *general* solution to avoid (soft) deadlocks
- common problem in multiprocessing where many processes share a specific type of mutually exclusive resource

“ When two trains approach each other at a crossing, both shall come to a full stop and neither shall start up again until the other has gone. ”

Illogical statute passed by the Kansas Legislature[1]