CSC-203 Operating Systems[3]

Lecture: 12

Instructor: Sanjog Sigdel

sigdelsanjog.com.np

Date: June 5, 2019

Lecture 12

Process Management

Process Scheduling:

Goals, Batch System Scheduling(FCFS, SJF, SRTN)
Interactive System Scheduling(Round Robin, Priority Scheduling, Multiple Queues

Process Management

Process Scheduling:

- 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.

- It's an essential part of multiprogramming operating systems which allow more than one process to be loaded in the executable memory at a time and the loaded process shares the CPU using time multiplexing

- If only one CPU is available, a choice has to be made which process to run next. The part of the **operating system that makes the choice is called the scheduler**, and the **algorithm it uses is called the scheduling algorithm**

Process Management

Process Scheduling Goals:

- **Be fair** – give each process a fair share of the CPU, allow each process to run in a reasonable amount of time.
- **Be efficient** – keep the CPU busy all the time.
- **Maximize throughput** – serve the largest possible number of jobs in a given amount of time; minimize the amount of time users must wait for their results.
- **Minimize response time** – interactive users should see good performance.
- **Be predictable** – a given job should take about the same amount of time to run when run multiple times.
- **Minimize overhead** – don't waste too many resources. Keep scheduling time and context switch time at a minimum.

Process Management

Process Scheduling Criteria:

CPU utilization - keep the CPU as busy as possible
Throughput - number of process that complete their execution per time unit
Waiting time - amount of time a process has been waiting in the ready queue
Turnaround 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 submtited until the first response is produced,

Process Scheduling:

Turnaround time

- the elapsed time between the time the job arrives and the time that it terminates. This includes the delay of waiting for the scheduler to start the job because some other process is still running and others may be queued ahead.

Start time

- Also known as **release time**, the start time is the time when the task is scheduled to run and actually gets to start running on the CPU.

Process Scheduling:

Response time

This is the delay between submitting a process and it being scheduled to run (its start time). Again, if we look a process as a series of CPU bursts the response time applies to each CPU burst. It is the delay between a task being *ready to run* and actually running.

Completion time:

This is the time when the process terminates.

Throughput

Refers to the number of processes that complete over some unit of time.

Process Management

Categories of Scheduling Algorithms:

Batch: FCFS, SFJ, SRTN

INteractive: Round Robin, Priority, Multiple Queues

Realtime:

Preemptive scheduling: Tasks are usually **assigned with priorities.** At times it is necessary to run a certain task that has a higher priority before another task although it is running. Therefore, the running task is interrupted for some time and resumed later when the priority task has finished its execution. **Round Robin, SRTN**

Non-preemptive scheduling, a running task is executed till completion. It cannot be interrupted. **FCFS, SJF**

Process Management

First Come First Serve: provides an efficient, simple and error-free process scheduling algorithm that saves valuable CPU resources.

Characteristics

- Process are schedules in the order they are received
- Once the process has the CPU, it runs to completion
- Easily implemented, by managing a simple queue of by storing time the process was received
- Fair to all process

Problems:

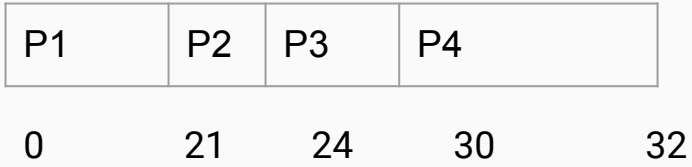
- No guarantee of good response time
- large average waiting time

Process Management

FCFS: Example

The average Waiting time=
(0+21+24+30)/4 = 18.75 ms

P.Id	B.T
P1	21
P2	3
P3	6
P4	2



Process Management

FCFS Example:

TurnAroundTime(TAT)=B.T +WT

Waiting Time(WT)=TAT-BT

AWT= ??

P. id	A.T	B.T	TAT = E.T - AT	WT = TAT-B.T
A	3	4	7-3=4	4-4=0
B	5	3	13-5=8	8-3=5
C	0	2	2-0=2	2-2=0
D	5	1	14-5=9	9-1=8
E	4	3	10-4=6	6-3=3

Shortest Job First(SJF) or Shortest Job Next

- Scheduling policy that selects the waiting process with the smallest execution time to execute next
- Has the advantage of having minimum average waiting time among all scheduling algorithms
- Greedy algorithm, may cause starvation if shorted process keep coming.

Characteristics

- The processing time are known in advances
- SFJ selects the process with shortest expected processing time. In case of tie FCFS scheduling is used
- Favors short jobs at the cost of long jobs

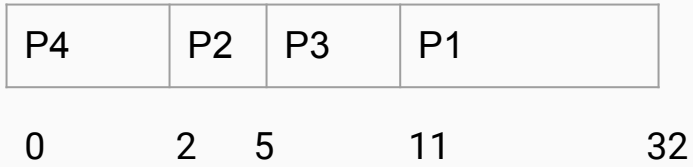
Process Management

Shortest Job First(SJF) or Shortest Job Next

Example:

Average Waiting time = (0+ 2 + 5 + 11)/4=4.5ms

P.Id	B.T
P1	21
P2	3
P3	6
P4	2



Shortest Remaining Time Next(SRTN)

- It's a preemptive version of shortest job next scheduling.
- The smallest amount of time remaining until completion is selected to execute.
- Advantageous because short processes are handled very quickly
- Low average waiting time than SJF
- Useful in timesharing
- Favors short jobs, long jobs can be victim of starvation

Process Management

SRTN Example:

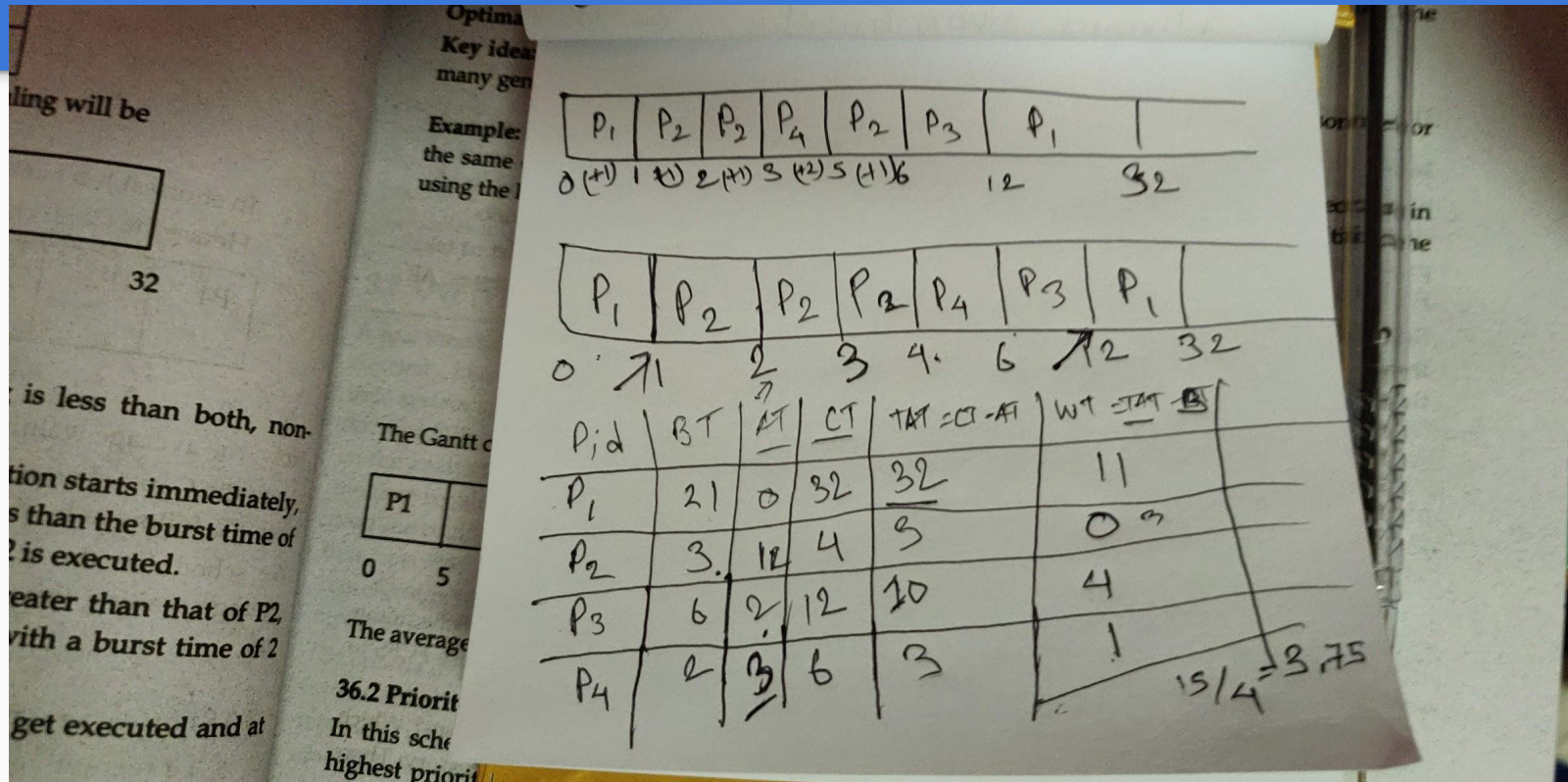
Average Waiting Time =

P.Id	B.T	AT	CT	TAT=CT-AT	WT=TAT-BT
P1	21	0			
P2	3	1			
P3	6	2			
P4	2	3			

P	P	P	P	P	P	P

0

AWT=
16/4= 4ms


P1 P2 P2 P4 P2 P3 P1
0 1 2 3 5 6 12 32
P1 P2 P2 P2 P4 P3 P1
0 1 2 3 4 6 12 32
Pid BT AT CT TAT=CT-AT WT=TAT-BT
P1 21 0 32 32 11
P2 3 1 4 3 0
P3 6 2 12 10 4
P4 2 3 6 3 1
15/4=3.75

Process Management

Categories of Scheduling Algorithms:

Batch: FCFS, SFJ, SRTN

INteractive: Round Robin, Priority, Multiple Queues

Realtime

June 6, 2019

Interactive System Scheduling

Round Robin Scheduling

In this algorithm the process is allocated with the CPU for the specific time period called time slice called quantum, which is normally of 10 to 100 milliseconds.

If the process completes its execution within this time slice, then it is removed from the queus otherwise it has to wait for another time slice. Preempted process is placed at the back of the ready list

Interactive System Scheduling

Round Robin Scheduling

Advantages

- Fail allocation of CPU across the process
- Used in timesharing system
- Low average waiting time when process lengths vary widely
- Poor average waiting time when process lengths are identical

Interactive System Scheduling

Round Robin Scheduling

Quantum size: If the quantum is very large, each process is given as much time as it needs for completion, RR degenerates to FCFS policy.

If quantum is very small, system busy at just switching from one process to another, the overhead of context-switching causes the system efficiency degrading

Process Management

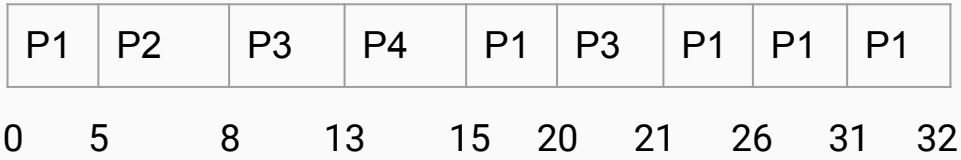
Interactive System Scheduling

Round Robin Scheduling

Example
Tie Quantum=5

Average Waiting Time=
(11+5+15+13)/4=11ms

P.Id	B.T	WT
P1	21	10+1=11
P2	3	5
P3	6	8+7=15
P4	2	13



Process Management

Interactive System Scheduling

Round Robin Scheduling

Example

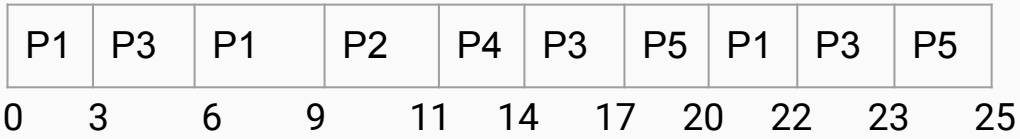
Time Quantum=3

Ready Queue

P1	P3	P1	P2	P4	P3	P5	P1	P3	P5

P.Id	AT	B.T	CT	TAT=CT-AT	WT= TAT-BT	RT
P1	0	8	22	22	14	0
P2	5	2	11	6	4	9-5
P3	1	7	23	22	15	3-1
P4	6	3	14	8	5	11-6
P5	8	5	25	17	12	17-8

Gantt Chart



Interactive System Scheduling
Priority Scheduling

- Priority is assigned to all the process and process with highest priority runs first.
- Priority assignment of processes is done on the basis of internal factor such as CPU and memory requirements or external factor such as user's choice

Problem: Starvation - low priority processes may never execute

SOlution: Aging - as time progresses increase the priority of the process

Process Management

Interactive System Scheduling

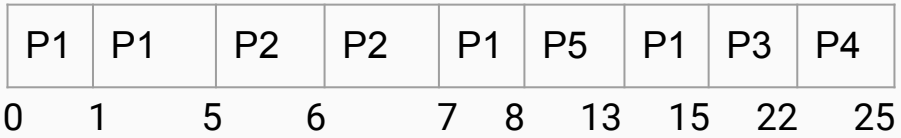
Priority Scheduling

Example

Priority **1 -> Highest, 5 ->Lowest**

P.Id	AT	B.T	Priority	CT	TAT=CT-AT	WT= TAT-BT
P1	0	8	3	15	15	7
P2	5	2	2	7	2	0
P3	1	7	4	22	21	14
P4	6	3	5	25	19	16
P5	8	5	1	13	5	0

Gantt Chart

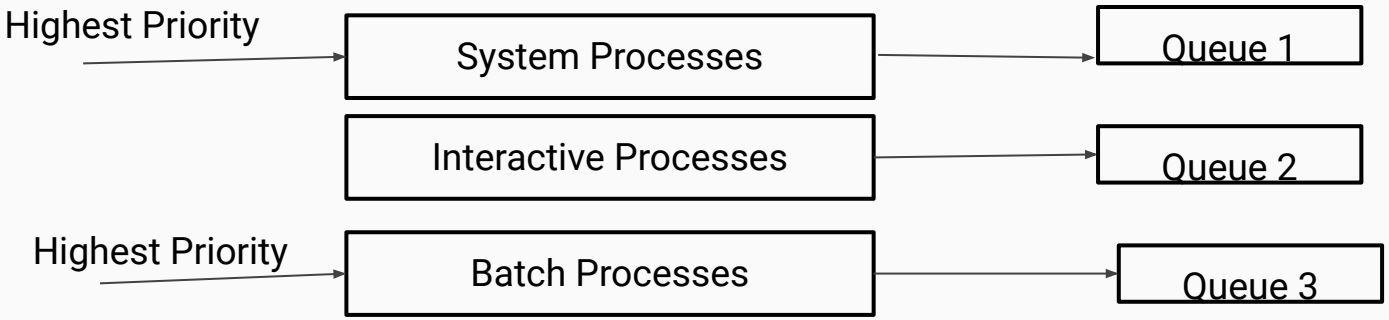


Multilevel Queue Scheduling

- Process in the ready queue can be divided into different classes where each class has its own scheduling needs.
- **Example:**
  A common division is a foreground(interactive) process and background(batch) processes.
- **These two classes have different scheduling needs, Multilevel Queue Scheduling assist in providing the resources**

Process Management

Multilevel Queue Scheduling



Multilevel Queue Scheduling

All three different type of processes have their own queue. Each queue has its own scheduling algorithm. For example, queue 1 and 2 uses ROund RObin while queue 3 uses SJF to schedule their processes.

Scheduling among the queues: **What will happen if all the queues have some processes? Which process should get the CPU?**

To determine this Scheduling among the queues is necessary. FOllowing two ways helps to do so:

1. **Fixed Priority**
   each process queues are assigned with priority Highest-Lowest. Lowest prioratized process can't run unless highest completes
2. **Time Slicing**
   Each queue is provided with certain CPU time to schedule their own processes

Process Management

Realtime System Scheduling

- Time plays an essential role in Real-Time Systems
- Typically, one or more physical devices external to the computer generate stimuli, and the computer must react appropriately to them within a fixed amount of time.

Examples:

the computer in a compact disc player gets the bits as they come off the drive and must convert them into music within a very tight time interval.

Patient monitoring in a hospital intensive-care unit**, the autopilot in an aircraft,** and **robot control** in an automated factory**. In all these cases, having the right answer but having it too late is often just as bad as not having it at all.**

Realtime System Scheduling

generally categorized as **hard real time**, meaning there are **absolute deadlines that must be met**—or else!— and **soft real time**, meaning that **missing an occasional deadline is undesirable, but nevertheless tolerable**. In both cases, real-time behavior is achieved by **dividing the program into a number of processes**, each of whose **behavior is predictable and known in advance**. These processes are generally **short lived a**nd can run to completion in **well under a second.**

Process Management

Realtime System Scheduling

The events that a real-time system may have to respond to can be further categorized as **periodic (meaning they occur at regular intervals)** or **aperiodic (meaning they occur unpredictably).**

Depending on how much time each event requires for processing, handling all of them may not even be possible. For example, if there are **m periodic events** and **event i occurs** with **period Pi** and requires **Ci sec** of CPU time to handle each event, then the **load can be handled only if**

$$\sum_{i=1}^{m} \frac{C_i}{P_i} \leq 1$$