Chapter 6: CPU Scheduling

COURSE INSTRUCTORS:
DR. MUHAMMAD NASEEM, ENGR. FARHEEN QAZI, MS. FALAK SALEEM

COURSE : OPERATING SYSTEMS (SWE-204)
BATCH : 2019 FALL : 2020

DEPARTMENT OF SOFTWARE ENGINEERING
SIR SYED UNIVERSITY OF ENGINEERING & TECHNOLOGY

1

Today's Agenda

- Basic Concepts
- Scheduling Criteria
- Scheduling Algorithms
- Multilevel Queue
- Multilevel Feedback Queue
- Algorithm Evaluation

2

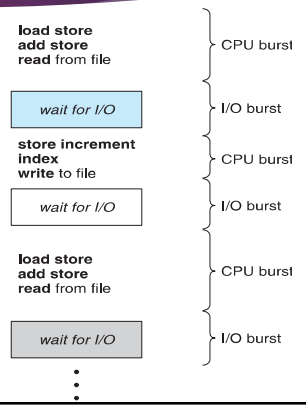
Objectives

- To introduce CPU scheduling, which is the basis for multiprogrammed operating systems
- To describe various CPU-scheduling algorithms
- To discuss evaluation criteria for selecting a CPU-scheduling algorithm for a particular system

3

Basic Concepts

- Maximum CPU utilization obtained with multiprogramming
- CPU–I/O Burst Cycle – Process execution consists of a **cycle** of CPU execution and I/O wait
- **CPU burst** followed by **I/O burst**
- CPU burst distribution is of main concern



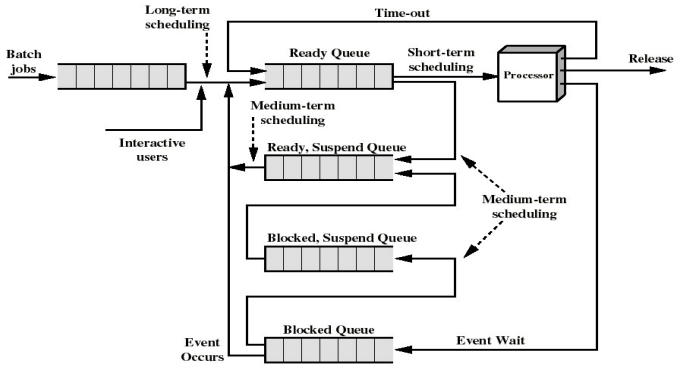
4

CPU Scheduler

- Selects from among the processes in memory that are ready to execute, and allocates the CPU to one of them.
- CPU scheduling decisions may take place when a process:
  1. Switches from running to waiting state
  2. Switches from running to ready state
  3. Switches from waiting to ready
  4. Terminates
- Scheduling under 1 and 4 is ***non-preemptive***
- All other scheduling is ***preemptive***

5

Queuing Diagram for Scheduling



6

Dispatcher

- Dispatcher module gives control of the CPU to the process selected by the short-term scheduler; this involves:
  - switching context
  - switching to user mode
  - jumping to the proper location in the user program to restart that program
- **Dispatch latency** – time it takes for the dispatcher to stop one process and start another running

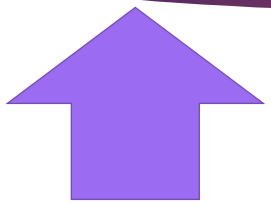
7

Scheduling Criteria

- **CPU utilization** – keep the CPU as busy as possible
- **Throughput** – # of processes that complete their execution per time unit
- **Turnaround time** – amount of time to execute a particular process
- **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, not output (for time-sharing environment)

8

Scheduling Algorithm Optimization Criteria



CPU utilization, Throughput

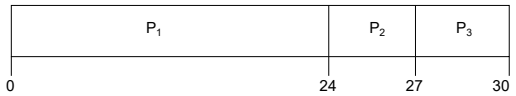
Turnaround Time, Waiting Time, Response Time

9

First-Come, First-Served (FCFS) Scheduling

Process	Burst Time
P_1	24
P_2	3
P_3	3

- Suppose that the processes arrive in the order: P_1 , P_2 , P_3
  The Gantt Chart for the schedule is:



- Waiting time for P_1 = 0; P_2 = 24; P_3 = 27
- Average waiting time: (0 + 24 + 27)/3 = 17

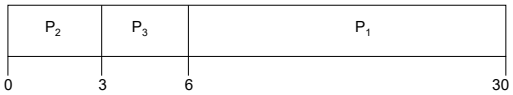
10

FCFS Scheduling (Cont.)

Suppose that the processes arrive in the order:

P_2 , P_3 , P_1

- The Gantt chart for the schedule is:



- Waiting time for P_1 = 6; P_2 = 0, P_3 = 3
- Average waiting time: (6 + 0 + 3)/3 = 3
- Much better than previous case
- **Convoy effect** - short process behind long process
  - Consider one CPU-bound and many I/O-bound processes

11

Shortest-Job-First (SJF) Scheduling

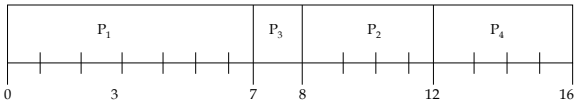
- Associate with each process the length of its next CPU burst
  - Use these lengths to schedule the process with the shortest time
- Two schemes:
  - **Non-preemptive** – once CPU given to the process it cannot be preempted until completes its CPU burst
  - **Preemptive** – if a new process arrives with CPU burst length less than remaining time of current executing process, preempt. This scheme is know as the Shortest-Remaining-Time-First (SRTF)
- SJF is optimal – gives minimum average waiting time for a given set of processes
  - The difficulty is knowing the length of the next CPU request
  - Could ask the user

12

Example of SJF (non-preemptive)

Process	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (non-preemptive)



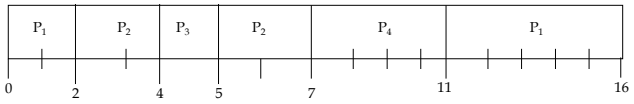
- Average waiting time = (0 + 6 + 3 + 7)/4 = 4

13

Example of SJF (preemptive)

Process	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (preemptive)



- Average waiting time = (9 + 1 + 0 +2)/4 = 3

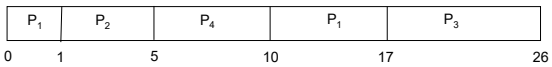
14

Example of Shortest-remaining-time-first

- Now we add the concepts of varying arrival times and preemption to the analysis

Process	*Arrival* Time	Burst Time
P_1	0	8
P_2	1	4
P_3	2	9
P_4	3	5

- *Preemptive* SJF Gantt Chart
- Average waiting time = (9 + 0 + 15 +2)/4 = 6.5 milliseconds.



- Average waiting time = [(10-1)+(1-1)+(17-2)+5-3)]/4 = 26/4 = 6.5 msec

15

Priority Scheduling

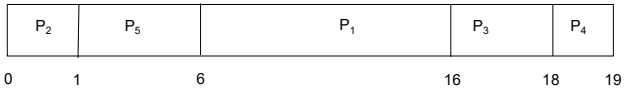
- A priority number (integer) is associated with each process
- The CPU is allocated to the process with the highest priority (smallest integer ≡ highest priority)
  - Preemptive
  - Non-preemptive
- SJF is priority scheduling where priority is the inverse of predicted next CPU burst time
- Problem ≡ **Starvation** – low priority processes may never execute
- Solution ≡ **Aging** – as time progresses increase the priority of the process

16

Example of Priority Scheduling (non-preemptive)

Process	Burst Time	Priority
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

- Priority scheduling Gantt Chart



- Average waiting time = 8.2 msec

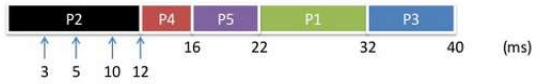
17

Example#02: Priority (non-preemptive)

- Example (Non-Preemptive)

Tasks	Arrival Times (ms)	Burst Time (ms)	Priority
P2	0.0	12	2
P3	3.0	8	5
P4	5.0	4	1
P1	10.0	10	4
P5	12.0	6	3

- Gantt chart



18

Contd....

- Time Calculation (Non-Preemptive)

P1 : 22 – 10 = 12 ms
P2 : 0 ms
P3 : 32 – 3 = 29 ms
P4 : 12 – 5 = 7 ms
P5 : 16 – 12 = 4 ms

Total waiting time : P1 + P2 + P3 + P4 + P5 = 52 ms
Average waiting time : 52 ms / 5 = 10.4 ms

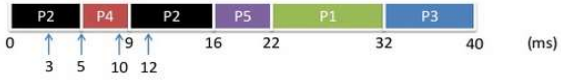
19

Example of Priority Scheduling (preemptive)

- Available in Preemptive and Non-Preemptive mode
- Example (Preemptive)

Tasks	Arrival Times (ms)	Burst Time (ms)	Priority
P2	0.0	12	2
P3	3.0	8	5
P4	5.0	4	1
P1	10.0	10	4
P5	12.0	6	3

- Gantt chart



20

Contd....

- Time Calculation (Non-Preemptive)

P1 : 22 – 10 = 12 ms
P2 : 0 ms
P3 : 32 – 3 = 29 ms
P4 : 12 – 5 = 7 ms
P5 : 16 – 12 = 4 ms

Total waiting time : P1 + P2 + P3 + P4 + P5 = 52 ms
Average waiting time : 52 ms / 5 = 10.4 ms

21

Round Robin (RR)

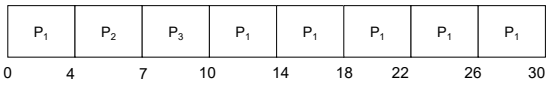
- Each process gets a small unit of CPU time (**time quantum** *q*), usually 10-100 milliseconds. After this time has elapsed, the process is preempted and added to the end of the ready queue.
- If there are *n* processes in the ready queue and the time quantum is *q*, then each process gets 1/*n* of the CPU time in chunks of at most *q* time units at once. No process waits more than (*n*-1)*q* time units.
- Timer interrupts every quantum to schedule next process
- Performance
  - *q* large ⇒ FIFO
  - *q* small ⇒ *q* must be large with respect to context switch, otherwise overhead is too high

22

Example of RR with Time Quantum = 4

Process	Burst Time
P_1	24
P_2	3
P_3	3

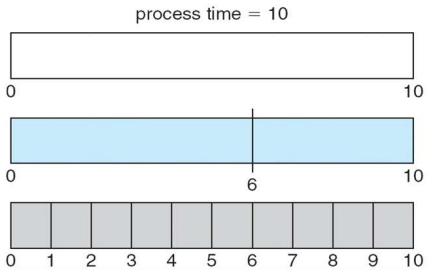
- The Gantt chart is:



- Typically, higher average turnaround than SJF, but better ***response***
- q should be large compared to context switch time
- q usually 10ms to 100ms, context switch < 10 usec

23

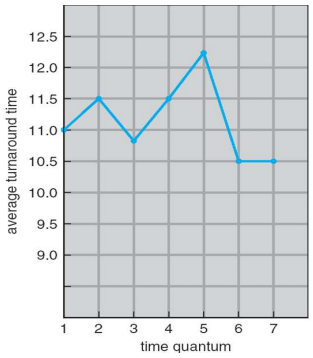
Time Quantum and Context Switch Time



quantum	context switches
12	0
6	1
1	9

24

Turnaround Time Varies With The Time Quantum



process	time
P_1	6
P_2	3
P_3	1
P_4	7

80% of CPU bursts should be shorter than q

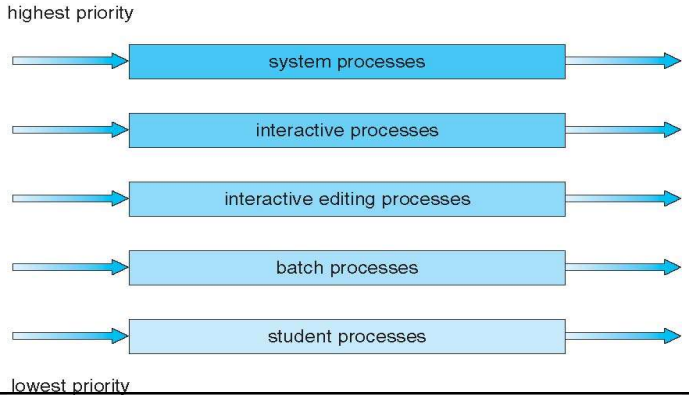
25

Multilevel Queue

- Ready queue is partitioned into separate queues, eg:
  - **foreground** (interactive)
  - **background** (batch)
- Process permanently in a given queue
- Each queue has its own scheduling algorithm:
  - foreground – RR
  - background – FCFS
- Scheduling must be done between the queues:
  - Fixed priority scheduling; (i.e., serve all from foreground then from background). Possibility of starvation.
  - Time slice – each queue gets a certain amount of CPU time which it can schedule amongst its processes; i.e., 80% to foreground in RR
  - 20% to background in FCFS

26

Multilevel Queue Scheduling



27

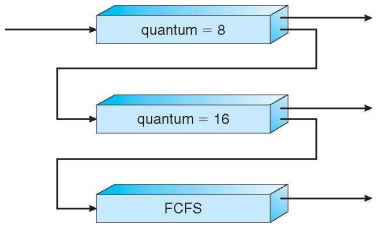
Multilevel Feedback Queue

- A process can move between the various queues; aging can be implemented this way
- Multilevel-feedback-queue scheduler defined by the following parameters:
  - number of queues
  - scheduling algorithms for each queue
  - method used to determine when to upgrade a process
  - method used to determine when to demote a process
  - method used to determine which queue a process will enter when that process needs service

28

Example of Multilevel Feedback Queue

- Three queues:
  - Q_0 – RR with time quantum 8 milliseconds
  - Q_1 – RR time quantum 16 milliseconds
  - Q_2 – FCFS
- Scheduling
  - A new job enters queue Q_0 which is served FCFS
    - When it gains CPU, job receives 8 milliseconds
    - If it does not finish in 8 milliseconds, job is moved to queue Q_1
  - At Q_1 job is again served FCFS and receives 16 additional milliseconds
    - If it still does not complete, it is preempted and moved to queue Q_2



29

Algorithm Evaluation

- How to select CPU-scheduling algorithm for an OS?
- Determine criteria, then evaluate algorithms
- Deterministic modeling
  - Type of **analytic evaluation**
  - Takes a particular predetermined workload and defines the performance of each algorithm for that workload

30

Queuing Models

- Describes the arrival of processes, and CPU and I/O bursts probabilistically
  - Commonly exponential, and described by mean
  - Computes average throughput, utilization, waiting time, etc
- Computer system described as network of servers, each with queue of waiting processes
  - Knowing arrival rates and service rates
  - Computes utilization, average queue length, average wait time, etc

31

Class Activity # 01

Draw the Gantt Chart and also calculate average waiting time and average turnaround time of the following algorithms using given table:

1. FCFS
2. SJF(Non-Preemptive)
3. SJF(Preemptive)
4. Round Robin (Quantum = 5).

Process	*Arrival Time*	*Burst Time*
P1	0.0	16
P2	0.3	5
P3	1.0	3

32

Class Activity # 02

Draw the Gantt Chart and also calculate average waiting time and average turnaround time of the following algorithms using given table:

1. FCFS
2. SJF(Non-Preemptive)
3. SJF(Preemptive)
4. Priority(Non-Preemptive)
5. Priority(Preemptive)
6. Round Robin (Quantum = 4).

Process	Arrival Time	Burst Time	Priority
P1	2	6	4
P2	5	3	1
P3	1	8	2
P4	0	3	1.5
P5	4	4	3.5

33

Class Activity # 03

Draw the Gantt Chart and also calculate average waiting time and average turnaround time of the following algorithms using given table:

1. FCFS
2. SJF(Non-Preemptive)
3. SJF(Preemptive)
4. Priority(Non-Preemptive)
5. Priority(Preemptive)
6. Round Robin (Quantum = 4).

Process	Arrival Time	Burst Time	Priority
P1	1	5	1
P2	7	3	2
P3	9	4	7
P4	12	2	4
P5	16	10	1.5

34

End of Chapter 6

Thank you

35