Information Technology Institute

Operating System Fundamentals

Chapter Five

CPU SCHEDULING

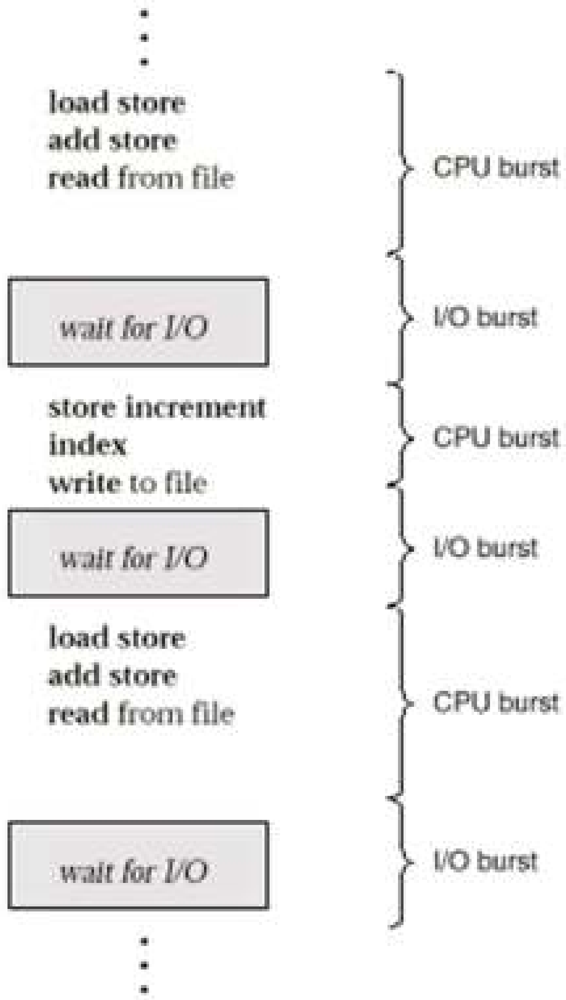
Table of Content

BASIC CONCEPTS

Basic Concepts

- Maximum CPU utilization obtained with multitasking
- CPU–I/O Burst Cycle
  - Process execution consists of a cycle of CPU execution and I/O wait.

Alternating Sequence of CPU And I/O Bursts



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.

CPU Scheduler Cont'd

- Preemptive
  - Process release the CPU before it finish execution
  - Example: Modern OS: Unix, Linux, Windows7
- Non-preemptive
  - Process release CPU when:
    - Running → Waiting
    - Running → Terminated
  - Example: MS Windows 3.1

Dispatcher

- Gives control of the CPU to the process selected by the short-term scheduler:
  - switching context
  - switching to suitable mode (User or Monitor)
  - jumping to the proper location in the user program to restart that program
- Dispatch latency
  - time taken by dispatcher to stop one process and start another running.

SCHEDULING CRITERIA

Scheduling Criteria

- CPU utilization
  - Keep the CPU as busy as possible
- Throughput
  - Number 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)

Optimization Criteria

- Maximize
  - CPU Utilization
  - Throughput
- Minimize
  - Turnaround time
  - Waiting time
  - Response time
- Considerations
  - Minimize maximum response time
  - Minimize the variance of response times

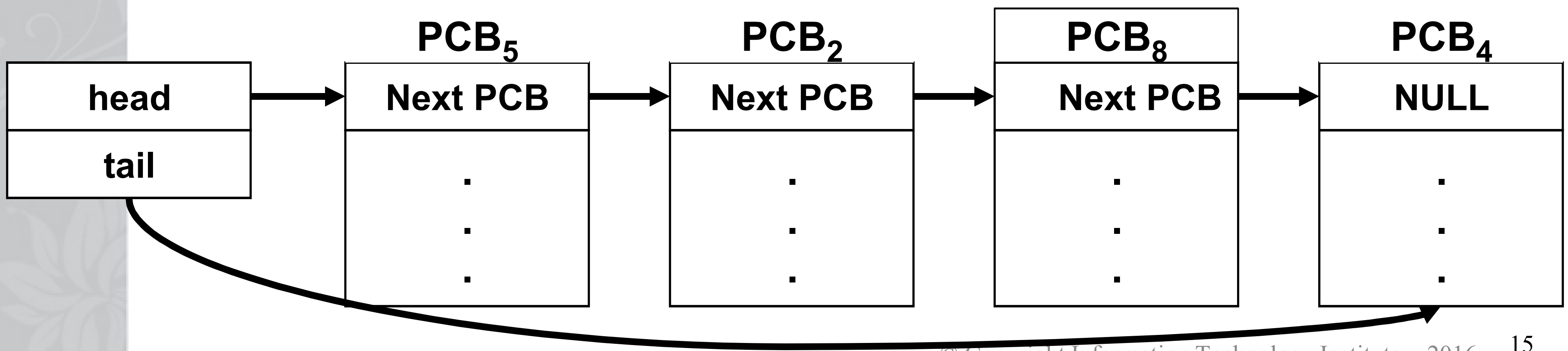
SCHEDULING ALGORITHMS

Scheduling Algorithms

- First-Come First Served
- Shortest-Job First
- Priority
- Round-Robin

First-Come, First-Served (FCFS) Scheduling

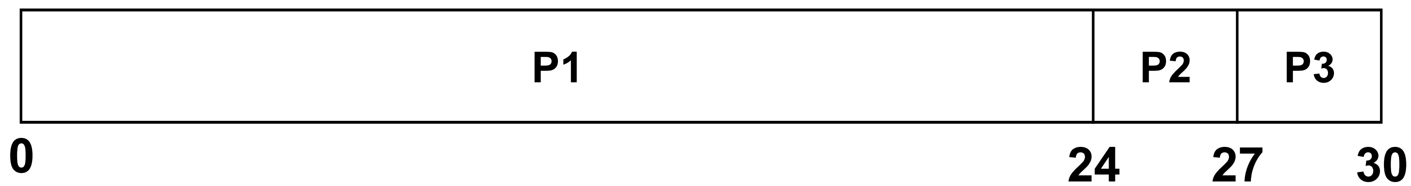
- Easily implemented
- Ready queue is FIFO
- P_n ready → P_n PCB is linked to tail of queue
- Process at head of ready queue → CPU
- Average waiting time is long!



Example 1

Process	Burst Time
P1	24
P2	3
P3	3

- Suppose that the processes arrive in the order: *P1* , *P2* , *P3* The Gantt Chart for the schedule is:

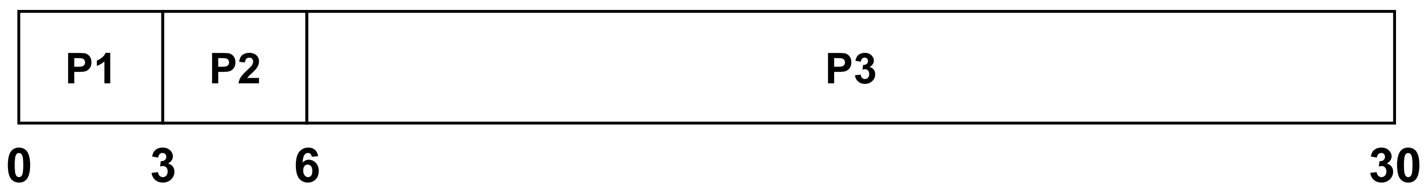


- Waiting time for *P1* = 0; *P2* = 24; *P3* = 27
- Average waiting time: (0 + 24 + 27)/3 = 17

Example 2

Process	Burst Time
P1	3
P2	3
P3	24

- Suppose that the processes arrive in the order: *P1* , *P2* , *P3* The Gantt Chart for the schedule is:



- Waiting time for $P1 = 0$; $P2 = 3$; $P3 = 6$
- Average waiting time: $(0 + 3 + 6)/3 = 3$

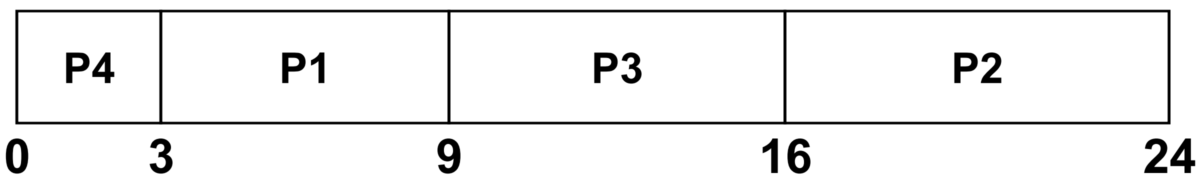
Shortest-Job-First (SJF) Scheduling

- 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.

Example 1

Process	Burst Time
P1	6
P2	8
P3	7
P4	3

- Suppose that all processes arrive at the same time: The Gantt Chart for the schedule is:

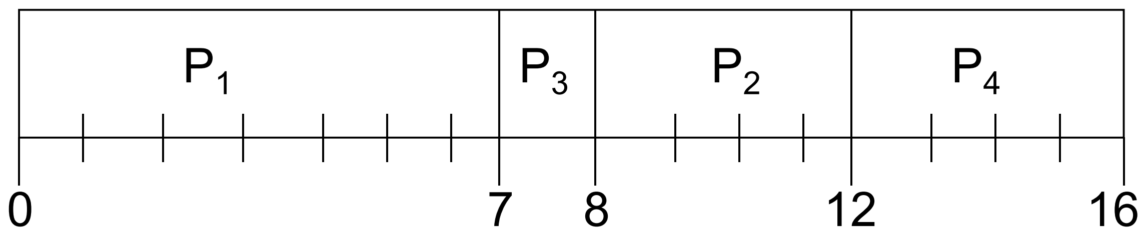


- Waiting time for *P1* = 3; *P2* =16; *P3* =9; P4=0
- Average waiting time: (3 + 16 + 9 + 0)/4 = 7

Example of Non-Preemptive SJF

Process	Arrival Time	Burst Time
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

- SJF (non-preemptive)

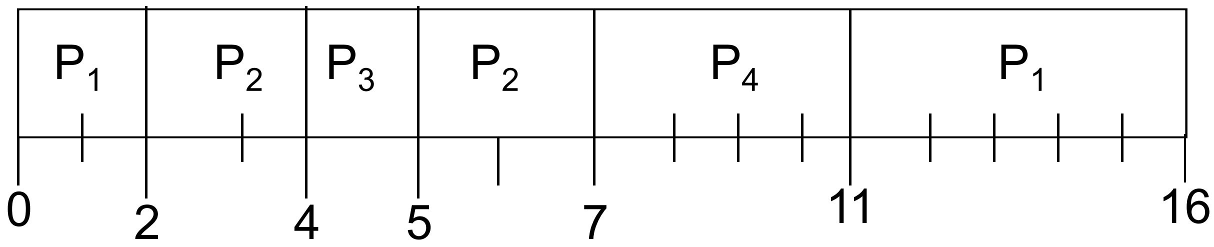


- Average waiting time = (0 + 6 + 3 + 7)/4 = 4

Example of Preemptive SJF

Process	Arrival Time	Burst Time
P1	0.0	7
P2	2.0	4
P3	4.0	1
P4	5.0	4

- SJF (preemptive)



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

Priority Scheduling

- 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 a priority scheduling where priority is the predicted next CPU burst time.
  - Problem ≡ Starvation – low priority processes may never execute.
  - Solution ≡ Aging – as time progresses increase the priority of the process.

Round Robin (RR)

- Each process gets a small unit of CPU time (*time quantum*), 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.
- Performance
  - q large FIFO
  - q small q must be large with respect to context switch, otherwise overhead is too high.

Example of RR, Time Quantum = 20

Process	Burst Time
P1	53
P2	17
P3	68
P4	24

- The Gantt chart is:

P_1	P_2	P_3	P_4	P_1	P_3	P_4	P_1	P_3	P_3

0 20 37 57 77 97 117 121 134 154 162

*Note: higher average turnaround than SJF, but better response.


Thank you