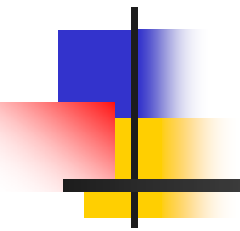
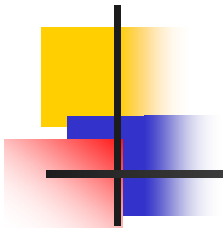
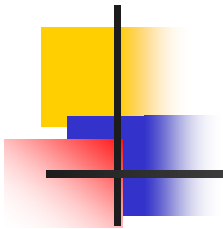


CPU Scheduling



Objectives

- Basic Concepts +
- Scheduling Criteria +
- Scheduling Algorithms +
- Scheduling Algorithm Evaluation +
- Summary +

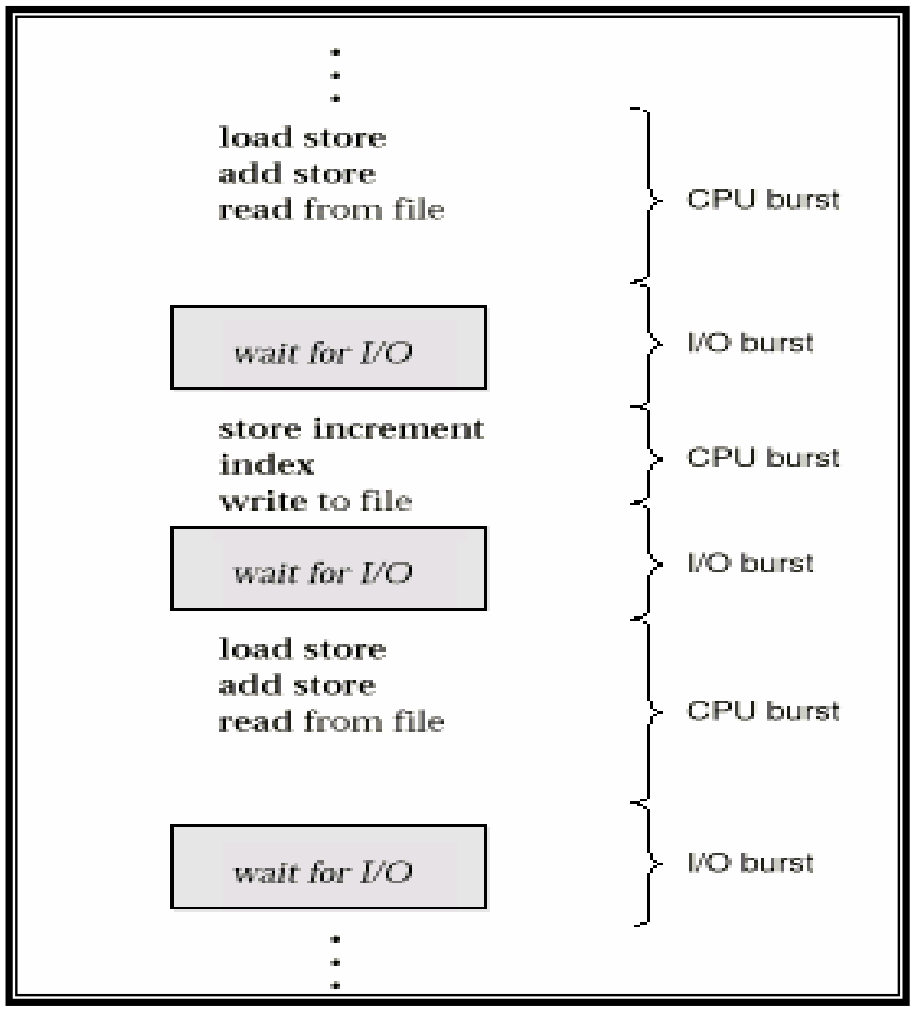


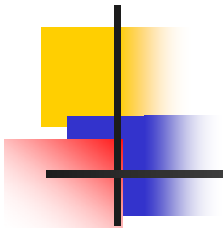
- Basic Concepts

- CPU-I/O Burst Cycle +
- Scheduling Schemes +
- Dispatcher +

-- CPU-I/O Burst Cycle

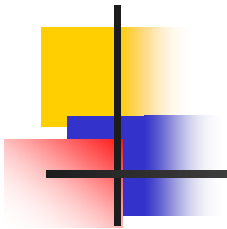
- Processes execution consists of a cycle of:
  - CPU execution
  - I/O wait
- Processes alternate between these two states.
- The frequency and duration of the CPU and I/O bursts affects the CPU scheduling algorithm





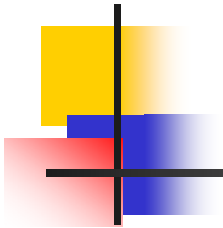
-- Scheduling Schemes

- CPU scheduling decisions may take under the following four circumstances:
  1. From running to waiting
  2. From running to ready
  3. From waiting to ready
  4. When a process terminates
- Nonpreemptive scheme
  - Takes place only under circumstances 1 and 4
  - Process keeps CPU until it:
    - Terminates
    - switches to waiting state
- Preemptive Scheme
  - All the above 4 circumstances are performed
  - Needs coordinated access to shared data, thus costly.



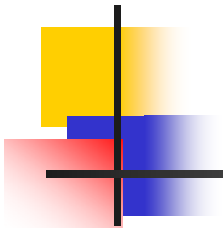
-- Dispatcher

- Gives control of the CPU to the process selected by the short-term scheduler.
- Its function involves:
  - Switching context
  - Switching to user mode
  - Jumping to the proper location in the user program to restart that program.
- It must be very fast for it is invoked during every process switch.
- In other words, the dispatch latency, which is the time the dispatcher takes to stop one process and start another must be very small.



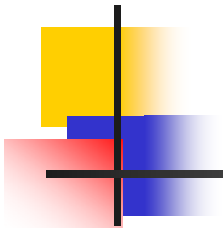
- Scheduling Criteria ...

- CPU Utilization: **keep the CPU as busy as possible**. 40% for lightly to 90% for heavily used system.
- Throughput: The number of processes completed per unit time.
- Turnaround Time: The interval of time a process took from the time it was submitted to the time it was completed.
- Waiting Time: The sum of periods that a process spent waiting in the ready queue.
- Response Time: The time from the submission of a request until the first response was produced. (for time-sharing environment)



... - Scheduling Criteria

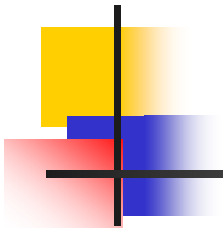
- Optimization Criteria
  - Max CPU utilization
  - Max throughput
  - Min turnaround time
  - Min waiting time
  - Min response time



- Scheduling Algorithms

- Decide which processes in the ready queue is to be allocated the CPU.

- Some of the existing algorithms are:
  - First Come First Served (FCFS) +
  - Shortest Job First (SJF) +
  - Priority Scheduling +
  - Round-Robin (RR) +
  - Multilevel Queue +
  - Multilevel Feedback queue +



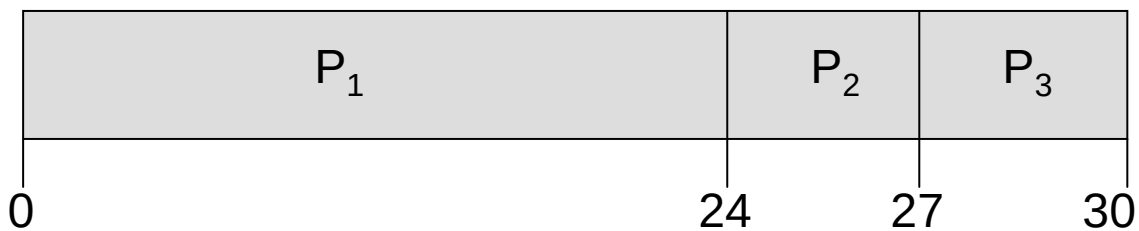
-- First Come First Serve (FCFS) Scheduling ...

- The process that requested the CPU first is allocated the CPU first.
- Advantage:
  - Simple to write and understand.
- Drawback
  - Can result in lower CPU and device utilization.
    - Example: In a dynamic environment where we have one CPU bound and many I/O bound processes.
  - Average waiting time is often long.

--- Example1: 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:

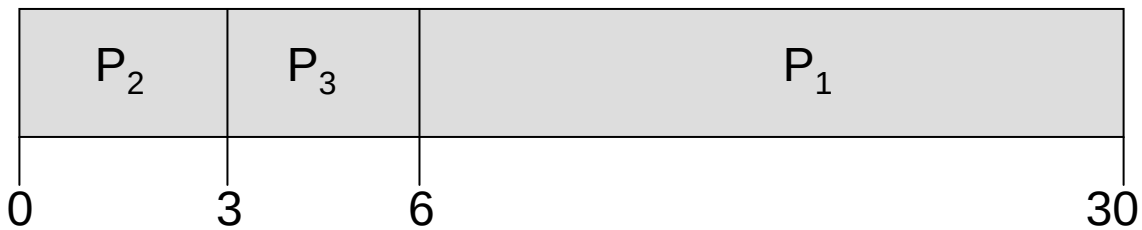


- Waiting time for P_1 = 0; P_2 = 24; P_3 = 27
- Average waiting time: (0 + 24 + 27)/3 = 17

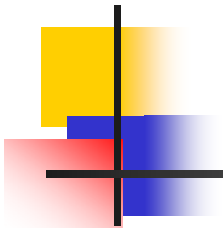
Example 2: FCFS Scheduling

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

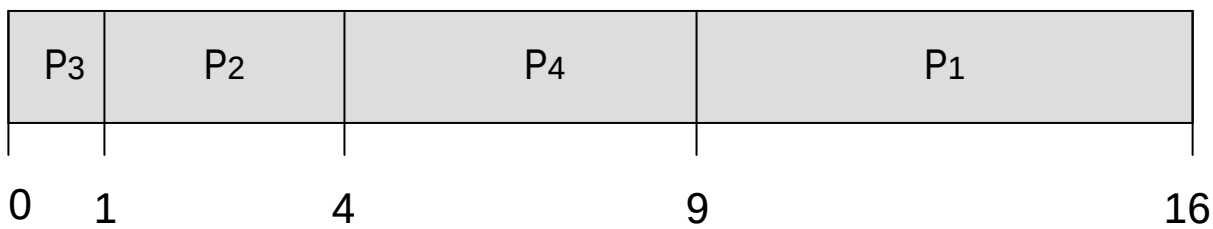


-- Shortest-Job-First (SJF) Scheduling

- Associate with each process the length of its next CPU burst. Use these lengths to schedule the process with the shortest time.
- If two process have the same length next CPU burst, FCFS is used.
- The problem is, it is difficult to tell how long the next CPU burst will be.
- Used frequently for long-term scheduling.
- Two schemes:
  - nonpreemptive – 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 gives the minimum average waiting time for a given set of processes.

--- Example 1: Non-Preemptive SJF

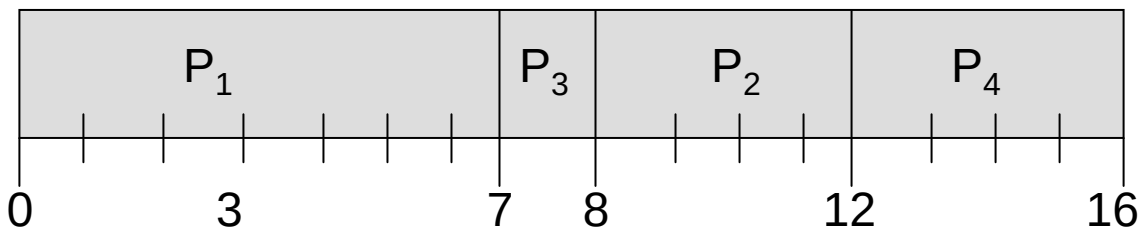
Process	Arrival Time	Burst Time
P_1	0.0	7
P_2	0.0	3
P_3	0.0	1
P_4	0.0	5



- Average waiting time = (9 + 1 + 0 + 4)/4 = 3.5

--- Example 2: Non-Preemptive SJF

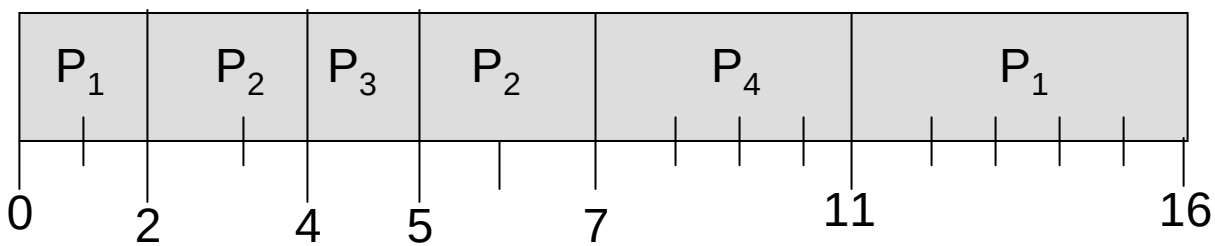
Process	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4



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

--- Example: Preemptive SJF

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

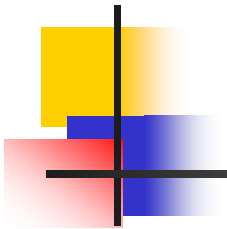


- Average waiting time = (9 + 1 + 0 +2)/4 = 3

--- Determining Length of Next CPU Burst

- Can only estimate the length.
- Can be done by using the length of previous CPU bursts, using exponential averaging.

1. t_n = actual lenght of n^{th} CPU burst
2. τ_{n+1} = predicted value for the next CPU burst
3. $\alpha, 0 \le \alpha \le 1$
4. Define : $\tau_{n=1} = \alpha\, t_n + (1-\alpha)\tau_n.$

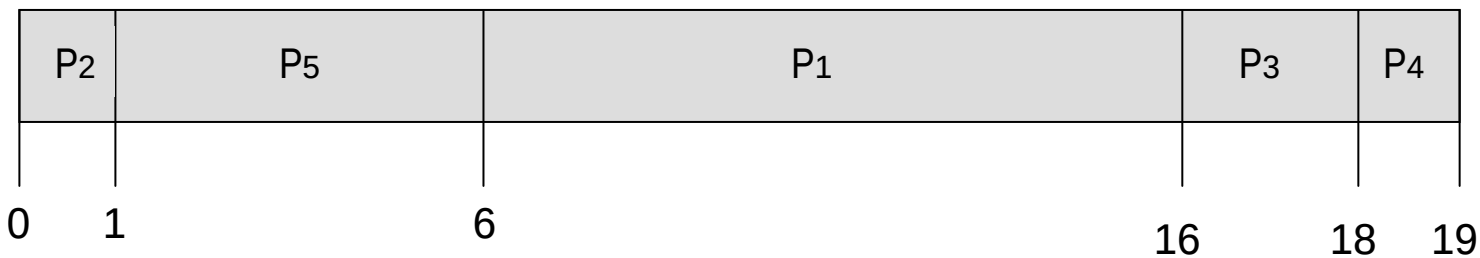


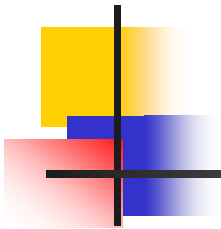
-- Priority Scheduling

- 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
  - Nonpreemptive
- SJF and FCFS are special kind of priority scheduling algorithms. 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.

--- Example of Nonpreemptive Priority Scheduling

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





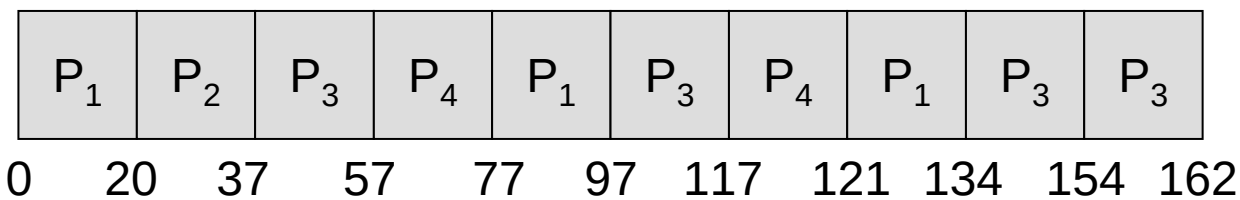
-- Round Robin (RR)

- 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 $\Rightarrow$ FIFO
  - q small $\Rightarrow$ q must be large with respect to context switch, otherwise overhead is too high.
  - Dispatch latency should be a factor in choosing the size of a time slice. This is because small time slice increases context switching.
- Average waiting time under RR is long.
- Turnaround time is also affected by the size of a time slice.

--- Example of RR with Time Quantum = 20

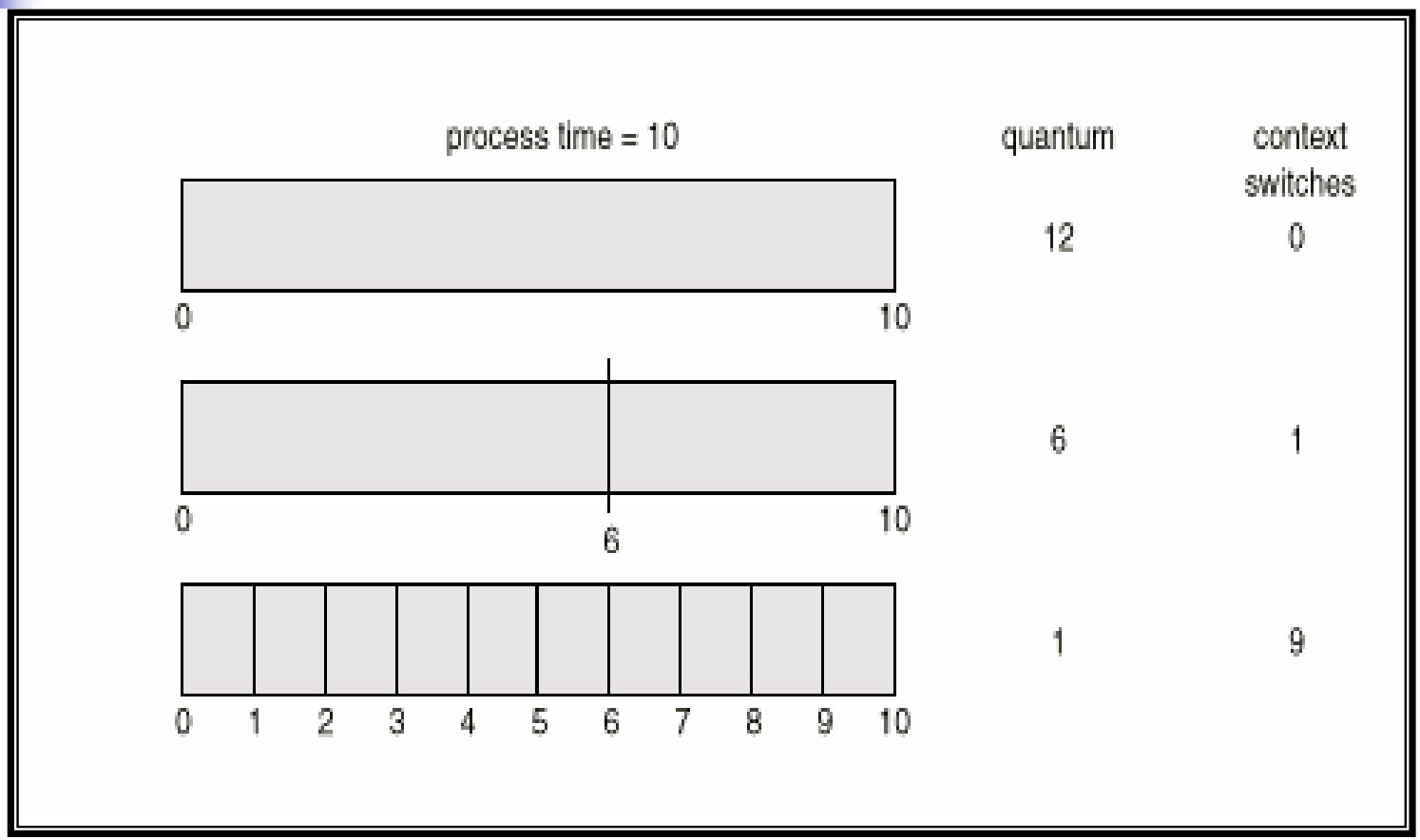
Process	Burst Time
P_1	53
P_2	17
P_3	68
P_4	24

- The Gantt chart is:

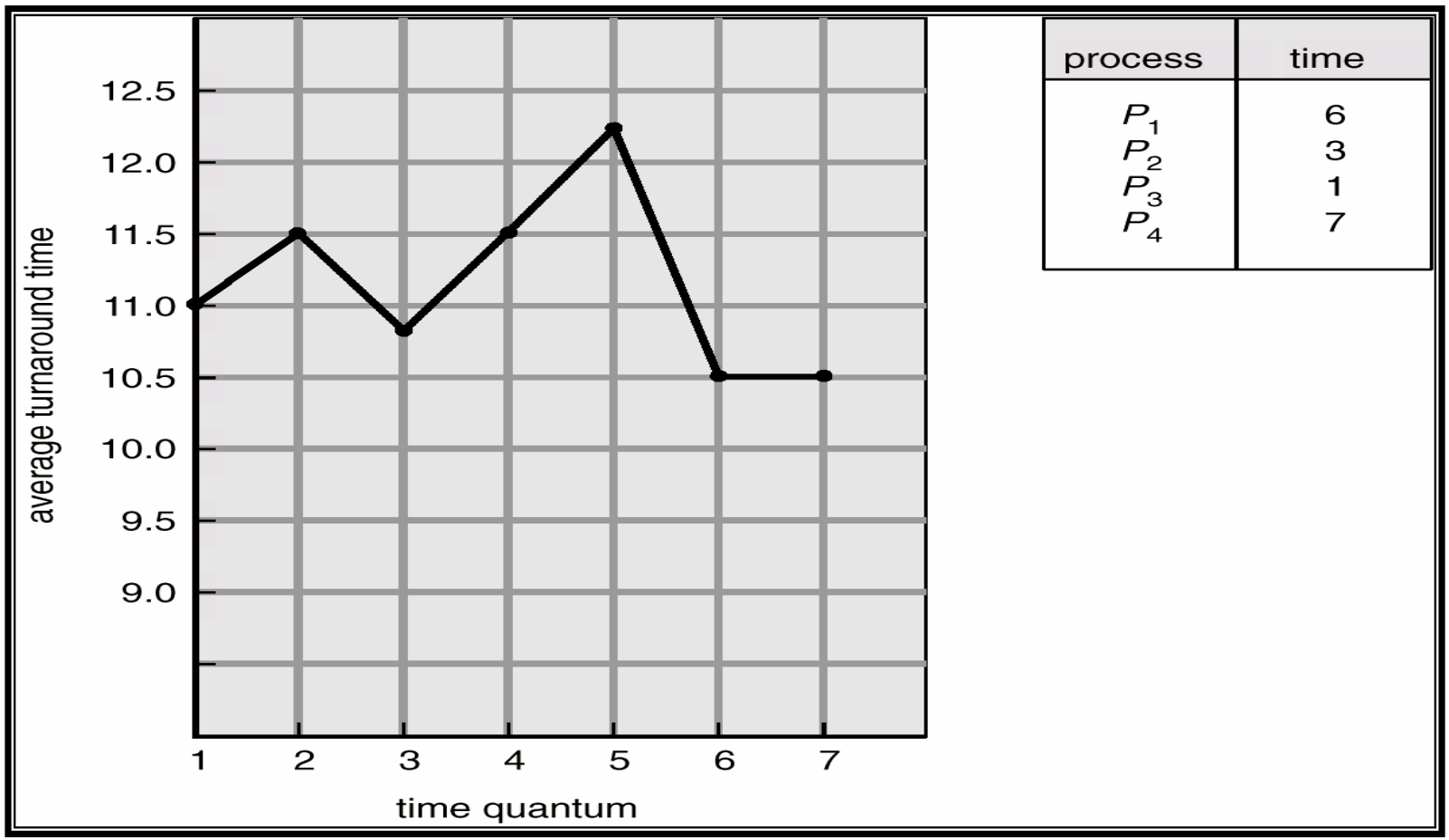


- Typically, higher average turnaround than SJF, but better *response*.

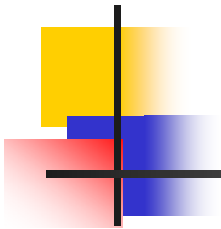
--- Time Quantum and Context Switch Time



Turnaround Time Varies With The Time Quantum



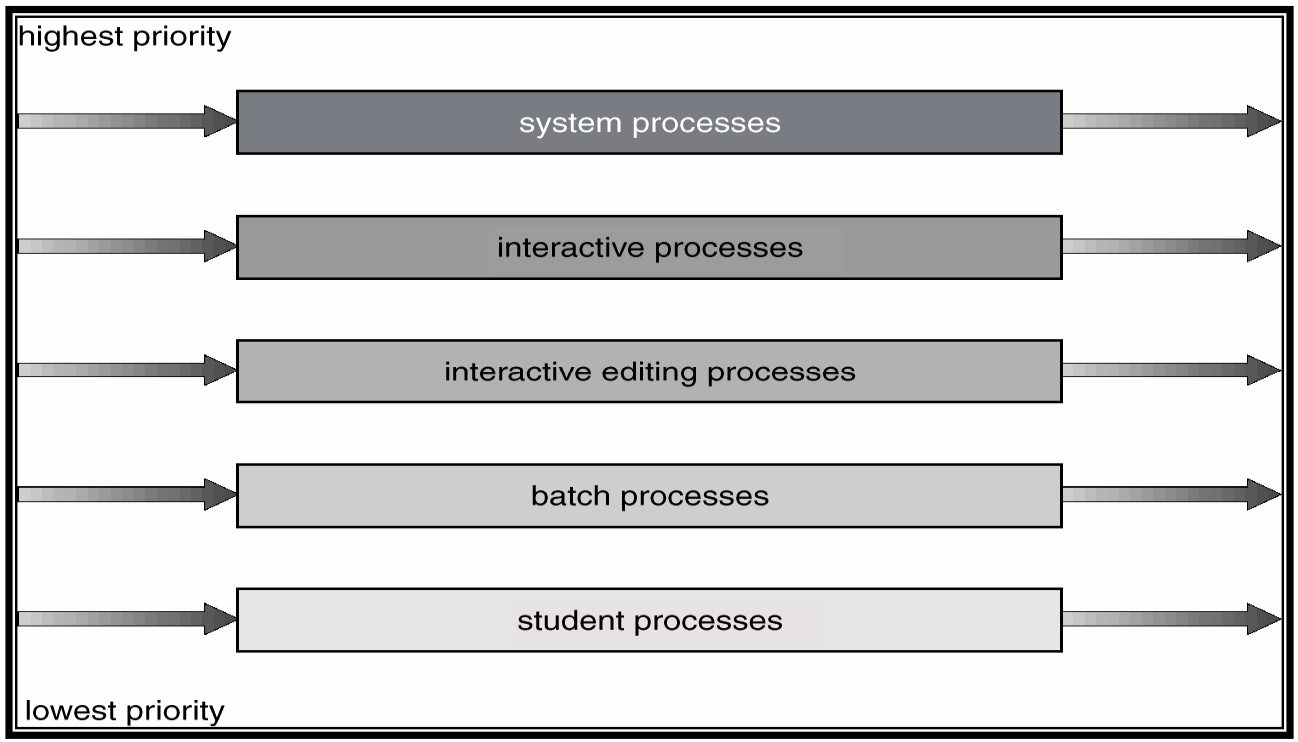
process	time
P_1	6
P_2	3
P_3	1
P_4	7

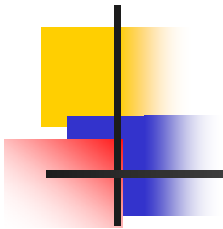


-- Multilevel Queue

- Ready queue is partitioned into separate queues. Example:
  - interactive
  - batch
- Each queue has its own scheduling algorithm. Example:
  - interactive – RR
  - batch – 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; Example:
    - 80% to foreground in RR
    - 20% to background in FCFS
- Disadvantage: Each process is permanently assigned to one queue.

--- Multilevel Queue Scheduling

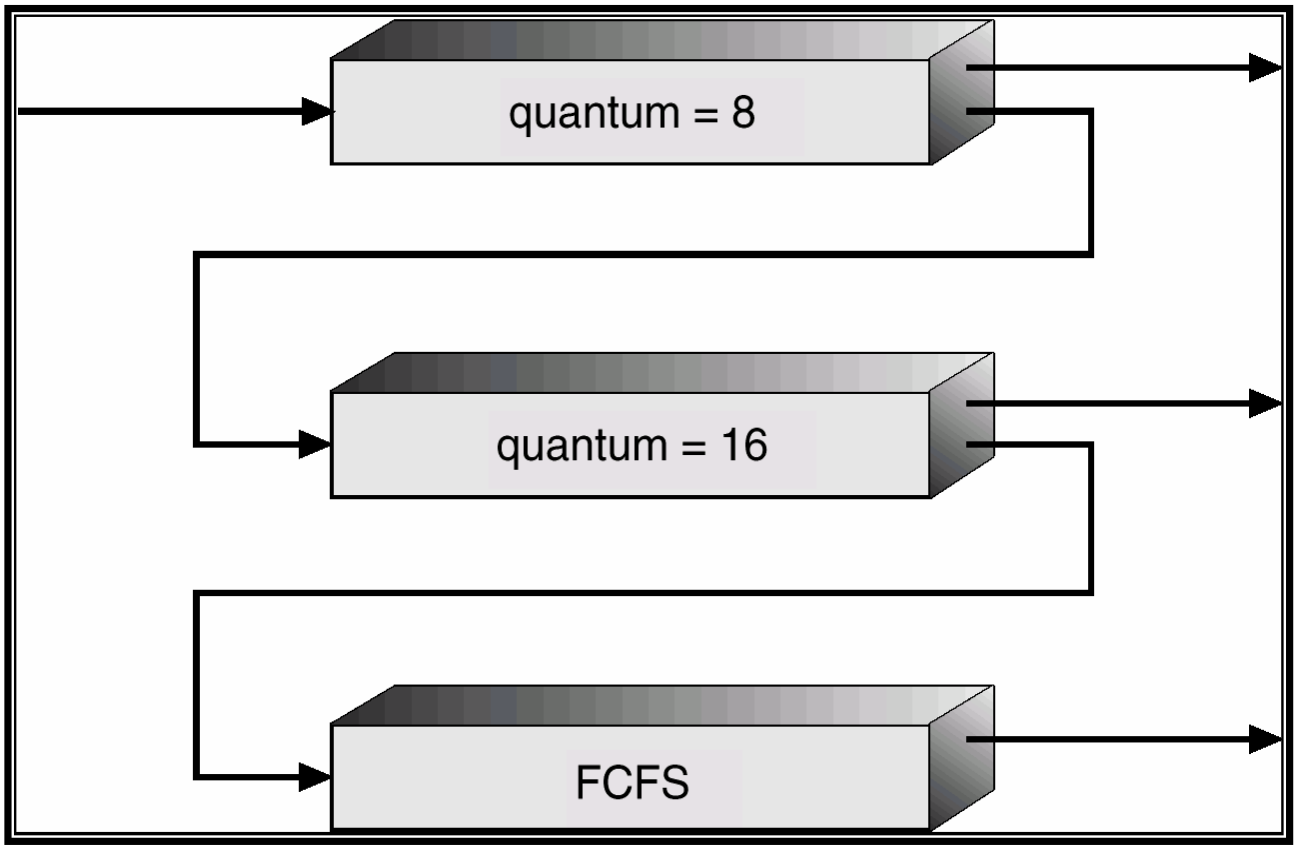


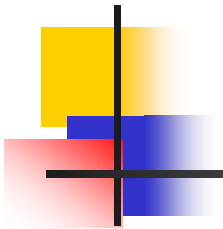


-- Multilevel Feedback Queue Scheduling ...

- Is like multilevel queue scheduling, however allows a process to move between queues.
- Process in a lower priority queue are only executed when the higher priority queues are empty.
- A process which arrives in a higher priority queue preempts a running process of lower priority queue.
- A process entering a ready queue is allocated the highest priority queue.
- Processes which consume a lot of CPU time are gradually moved to lower priority queues. This gives priority to interactive and I/O bound processes.
- Processes that wait too long in the lower priority queue are gradually moved to higher priority queues. (to prevent starvation)

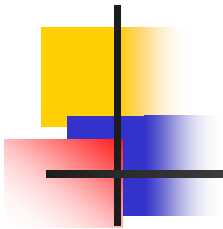
... --- Example of Multilevel Feedback Queue





... -- Multilevel Feedback Queue

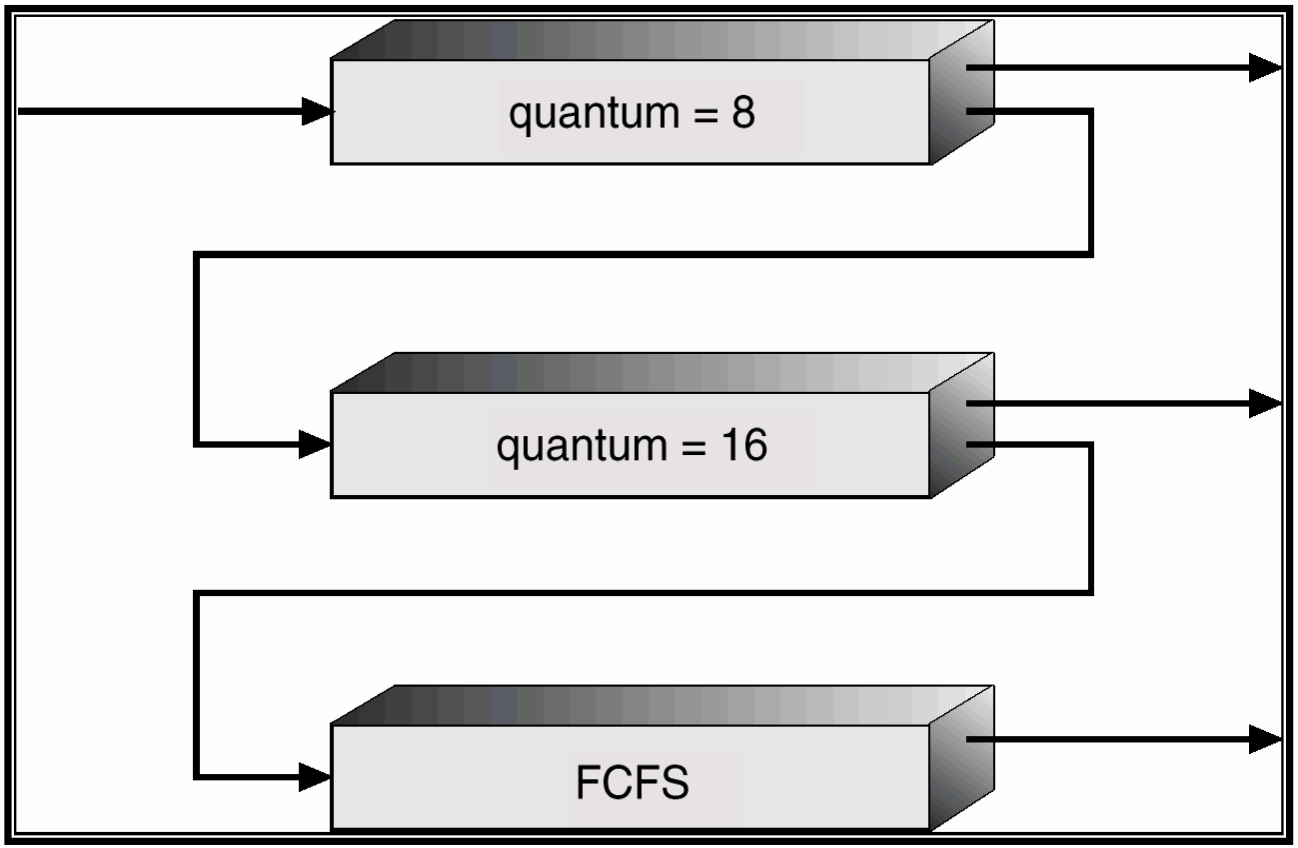
- Multilevel-feedback-queue scheduler defined by the following parameters:
  - number of queues
  - scheduling algorithms for each queue
  - method used to determine when to upgrade a process
  - method used to determine when to demote a process
  - method used to determine which queue a process will enter when that process needs service

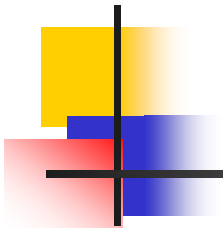


--- Example of Multilevel Feedback Queue ...

- Three queues:
  - Q_0 – time quantum 8 milliseconds
  - Q_1 – 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.

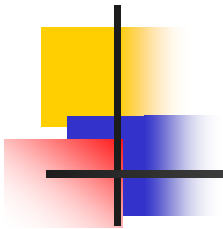
... --- Example of Multilevel Feedback Queue





- Scheduling Algorithm Evaluation

- First define the criteria to be used in selecting an algorithm. E.g. CPU utilization, response time, or throughput. Then choose one of the following methods to evaluate the various scheduling algorithms under consideration:
  - Deterministic Modeling
  - Queuing Models
  - Simulations
  - Implementation



-- Deterministic Modeling...

- Assumes a predetermined workload and defines the performance of each algorithm for that workload.
- Advantage:
  - It is simple and fast.
- Disadvantage:
  - Too specific.
  - Requires too much exact knowledge.

... -- Deterministic Modeling

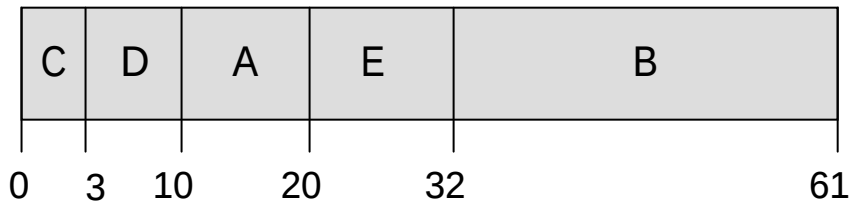
- Consider The following workload shown and SJF and RR (quantum=10) scheduling algorithms. Assume all processes arrived at time 0.

process	Burst Time
A	10
B	29
C	3
D	7
E	12

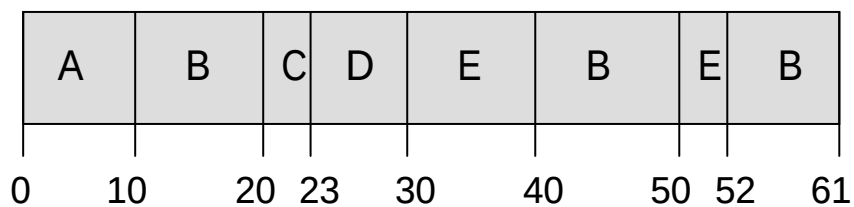
Avg. Waiting time:

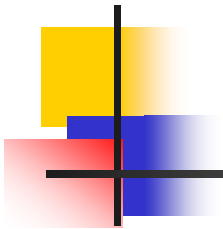
- SJF = 13
- RR = 23

SJF



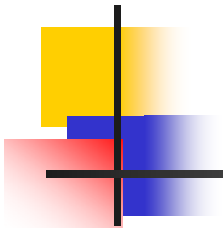
RR





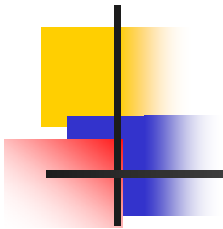
-- Queuing Models

- Mathematical formulas based on queuing network analysis, are used to compare scheduling algorithms.
- Some of the parameters these formulas use are:
  - the rate of processes entering a queue (λ).
  - The average waiting time in a queue (W).
- Example is Little's formula: $n = \lambda \times W$
- λ and W may be measured and then approximated or simply estimated.
- Disadvantage:
  - Limited due to algorithm and mathematical complexity
  - Necessary to make a lot of assumptions which may not be accurate.



-- Simulation

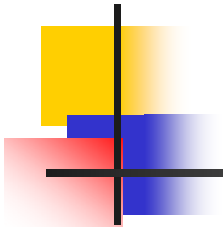
- Involves programming a model of a computer system.
  - Data structures to represent major components of the system.
  - Parameters to represent clock and so on.
- The input for the simulator can come from:
  - Random-number generators.
  - Mathematically defined distribution generated by monitoring the real system.
- Drawback:
  - Needs a lot of computation time.
  - Requires huge storage space.
  - Developing the simulator is also a major task.
  - Of limited accuracy. (after all, it is simulation)



-- Implementation

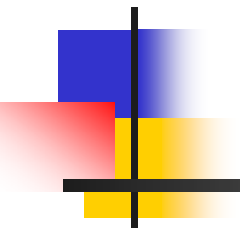
- Puts the actual algorithm in the real system.

- Drawbacks:
  - Developing the different scheduling algorithms.
  - Users may be frustrated by constantly changing environment.
  - The environment might change due:
    - New applications
    - The behavior of user. Example: when SJF is used users will break large programs into a set of smaller processes.



- Summary

- CPU & I/O burst cycle
- Scheduling Schemes: Preemptive & non-preemptive
- Short-term scheduler, Dispatcher, and Dispatch latency
- Scheduling criteria
  - CPU utilization
  - throughput
  - turnaround time
  - waiting time
  - response time
- Scheduling Algorithms
  - First Come First Served (FCFS)
  - Shortest Job First (SJF)
  - Priority Scheduling
  - Round-Robin (RR)
  - Multilevel Queue
  - Multilevel Feedback queue
- Evaluation of Scheduling algorithms
  - Deterministic model
  - Queuing model
  - Simulation
  - Implenetation



End

Chapter 5