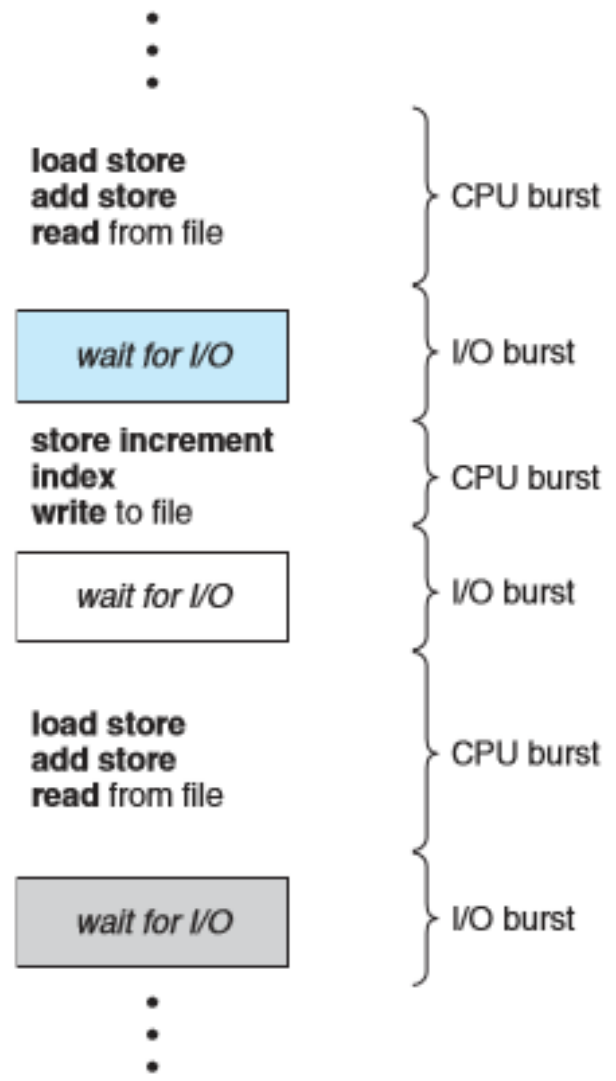
Sistemas Operativos 1

PROCESS MANAGEMENT
SCHEDULING

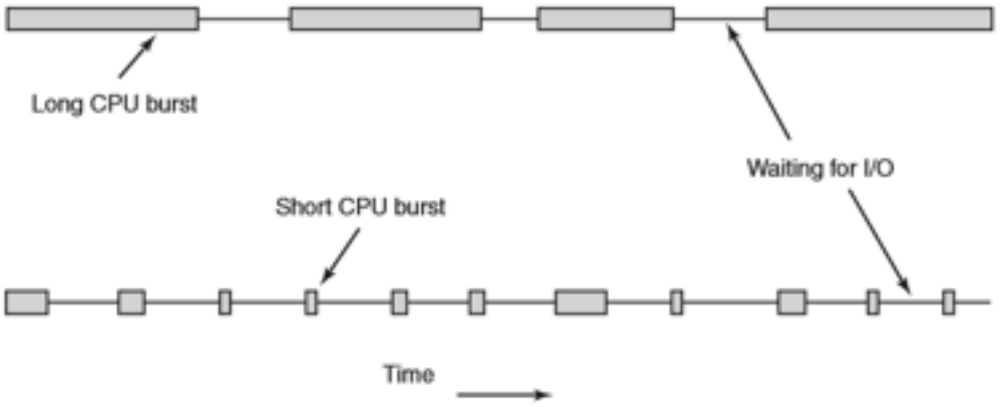


- multiprogramming
  Time-Sharing
  - MAXIMIZAR uso de CPU
  - SIEMPRE UN proceso corriendo

- CPU scheduler or **Short-term scheduler**

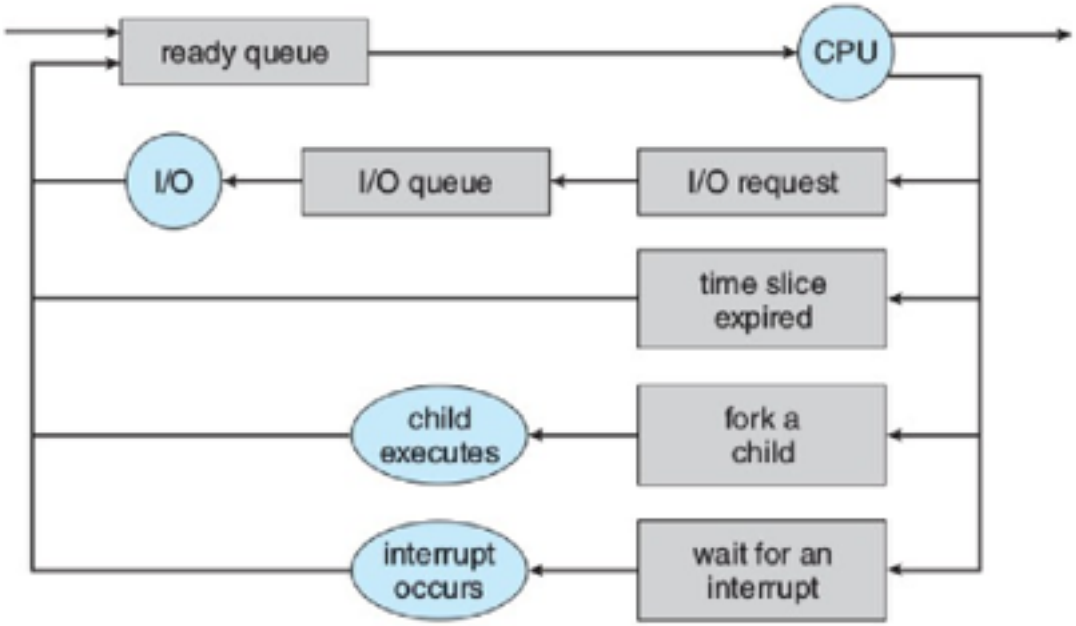
- **dispatcher**
  da el control de la CPU al proceso seleccionado por el scheduler
  1. Switching context
  2. Switching to user mode
  3. Jumping to the proper location in the user program to restart that program

  - *dispatch latency*



Preemptive Scheduling

CPU scheduling deals with the problem of deciding which of the processes in the ready queue is to be allocated the CPU



- CPU-scheduling decisions may take place under the following four circumstances:

nonpreemptive

1. When a process switches from the **running state** *TO* the **waiting state**

preemptive

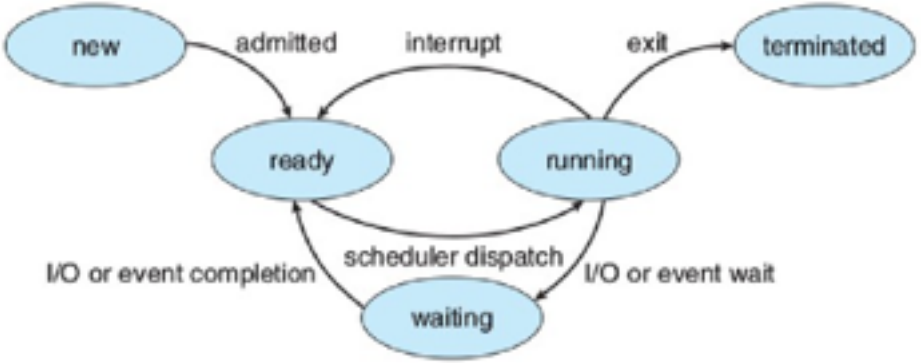
issue: race condition/ kernel structure changes

2. When a process switches from the **running state** TO the **ready state**

3. When a process switches from the **waiting state** TO the **ready state**

nonpreemptive

4. When a **process terminates**



Scheduling CRITERIA

CPU Utilization

Mantener la CPU lo más ocupada posible (p. ej. 40-90%)

Throughput

Rendimiento – # de procesos que terminan por unidad de tiempo

Turnaround time (submission to completion time)

Tiempo de retorno – tiempo insumido en ejecutar un proceso en particular

Tiempo de **espera** para **ingresar en memoria** + Tiempo de **espera** en la **cola de listos** + Tiempo de **ejecución** de **CPU** + Tiempo de **E/S**

Waiting time

Tiempo de espera en la cola de listos

Response time

Tiempo de respuesta

Cantidad de tiempo desde que se hace el pedido hasta la primer salida (sistemas interactivos o de tiempo compartido)

Scheduling Algorithms

CPU scheduling deals with the problem of deciding which of the processes in the ready queue is to be allocated

- First-Come, First-Served Scheduling (FCFS)
- Shortest-Job-First Scheduling (SJF)
- Priority Scheduling
- Round-Robin Scheduling (RR)
- Multilevel Queue Scheduling
- Multilevel Feedback Queue Scheduling

Scheduling Algorithms

First-Come, First-Served Scheduling (**FCFS**)

SIMPLE FIFO QUEUE

WAITING TIME

Process	Burst Time
P_1	24
P_2	3
P_3	3

convoy effect

average waiting time = (0 + 24 + 27)/3 = 17 milliseconds

Gantt chart

average waiting time = (0 + 3 + 6)/3 = 3 milliseconds

Scheduling Algorithms

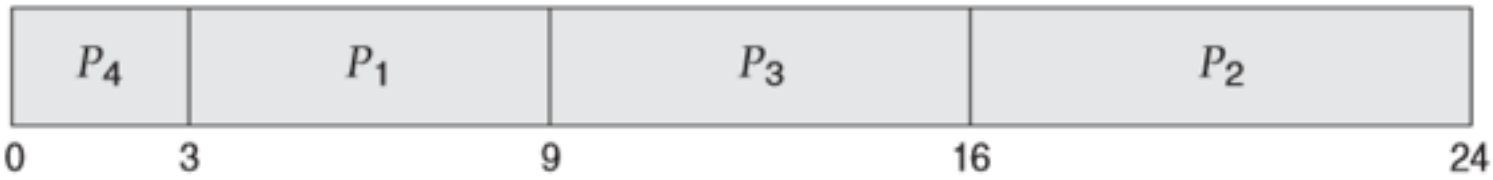
Shortest-Job-First Scheduling (**SJF**)

shortest-next- CPU-burst

- **nonpreemptive**
- **preemptive**: llega un proceso con una ráfaga de CPU menor que el remanente del proceso en ejecución, se expropia. (Shortest-Remaining-Time-First (SRTF))

- Process + next CPU burst
- CPU se asigna con el proceso de ***MENOR*** CPU burst
- DESEMPATA FCFS

Process	Burst Time
P_1	6
P_2	8
P_3	7
P_4	3



average waiting time is $(3 + 16 + 9 + 0)/4 = 7$ milliseconds (FCFS 10.25 milliseconds)

OPTIMO
asigna minimum average waiting time

BATCH

LA CLAVE
conocer longitud de la siguiente CPU burst

TIME SHARING

- **estimar** la ráfaga de cpu
- se utiliza las longitudes de las **ráfagas previas** usando un **promedio exponencial**

- **nonpreemptive**
- **preemptive**: llega un proceso con una prioridad mayor que la prioridad del proceso en ejecución, se expropia.

Scheduling Algorithms
Priority Scheduling

- Process + PRIORITY
- CPU se asigna con el proceso de ***MAYOR*** priority
- DESEMPATA FCFS
- PRIORITIES fixed range of numbers

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

average waiting time = 8.2 milliseconds

starvation
indefinite blocking

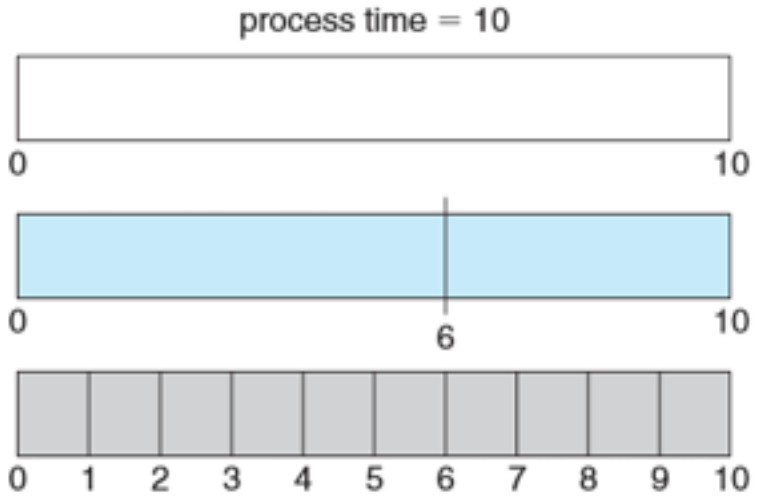
aging

Scheduling Algorithms
Round-Robin Scheduling (**RR**)

Process	Burst Time
P_1	24
P_2	3
P_3	3

- diseñado para Time Sharing
- FIFO queue
- time quantum / time slice (10-100 milliseconds)
- CPU asignada por 1 time quantum
- Timer

average waiting time = 17/3 = 5.66 milliseconds

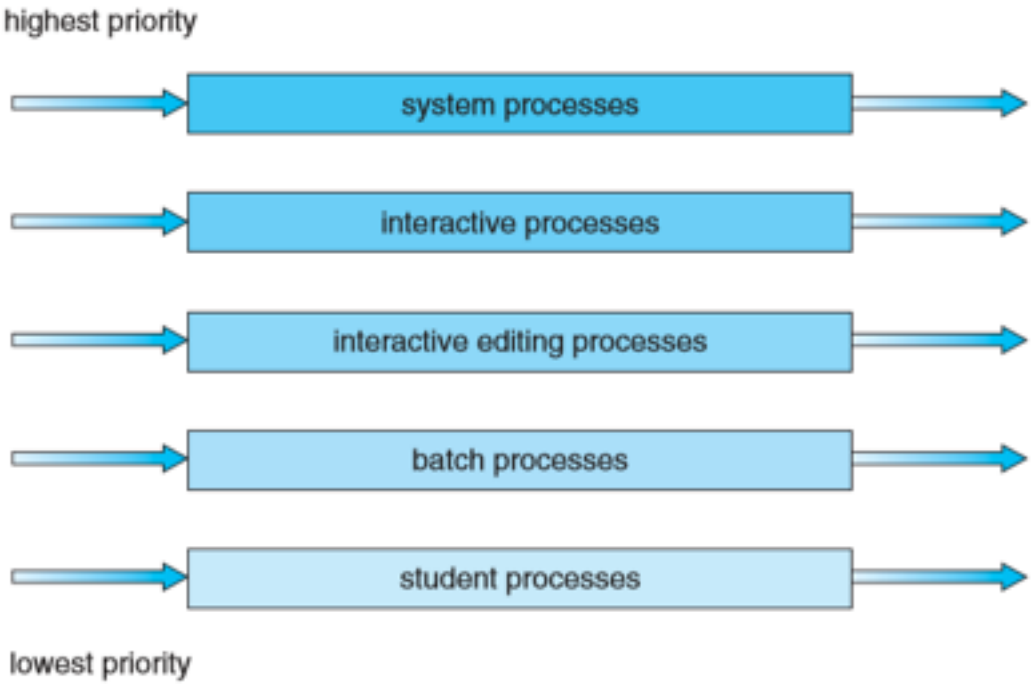


Si hay **N procesos** listos y un **quantum de Q**, entonces cada proceso obtiene **1/N** del **tiempo de CPU** en porciones de **a lo sumo Q unidades de tiempo**.
Ningún proceso espera más de **(N-1) Q** unidades de tiempo.

Q grande ⇒ FIFO

Q chico ⇒ Q debe ser grande respecto al cambio de contexto, de otra forma el overhead es alto

Scheduling Algorithms
Multilevel Queue Scheduling

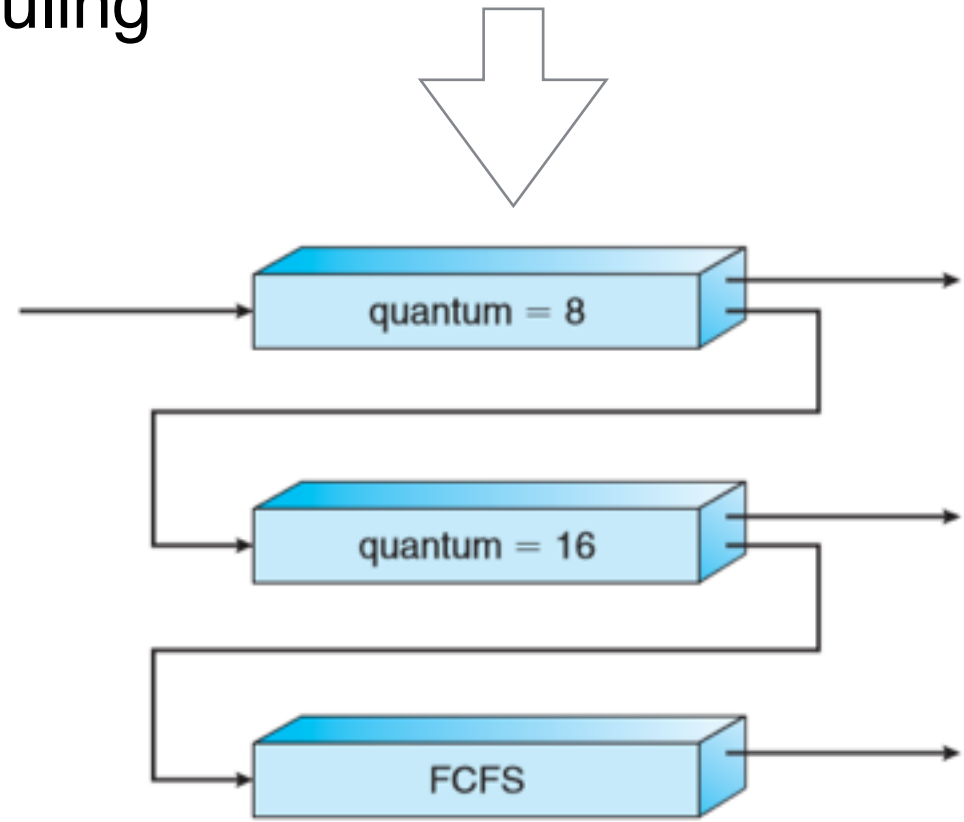


- Ready queue se divide en **foreground** (interactive) processes queue + **background** (batch) processes queue
- Cada cola define su propio algoritmo de planificación: foreground RR, background FCFS
- Panificación entre colas:
  - **prioridad fija**: primero procesos foreground y luego background. Posible inanición.
  - **partición de tiempo de CPU**: 80% foreground RR, 20% background FCFS

inflexible: los procesos no cambias de naturaleza y por ende de cola

Scheduling Algorithms
Multilevel Feedback Queue Scheduling

- permite que los procesos se muevan entre las colas
- agrupa los procesos según las características de CPU bursts
- mucho cpu time ⇒ mover a lower-priority queue
- **LA CLAVE**: mantener I/O-bound o interactive process en higher-priority queues
- **aging** prevents **starvation** ⇒ si espera mucho en lower-priority queue entonces pasar a higher-priority queue



Que define un este planificador ?

- Número de colas
- Algoritmo de planificación de cada cola
- Método usado para promover un proceso
- Método para des-promover un procesos
- Método usado para determinar en que cola entra un nuevo proceso