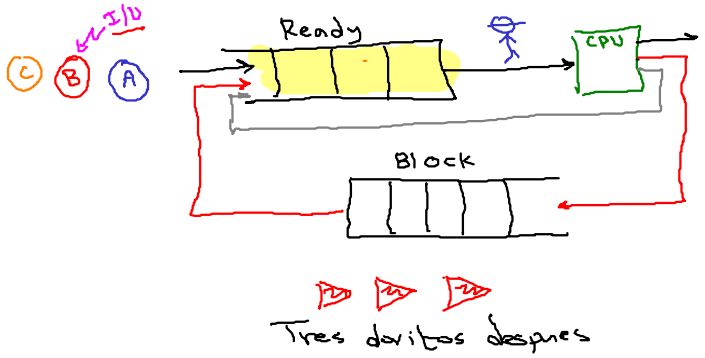
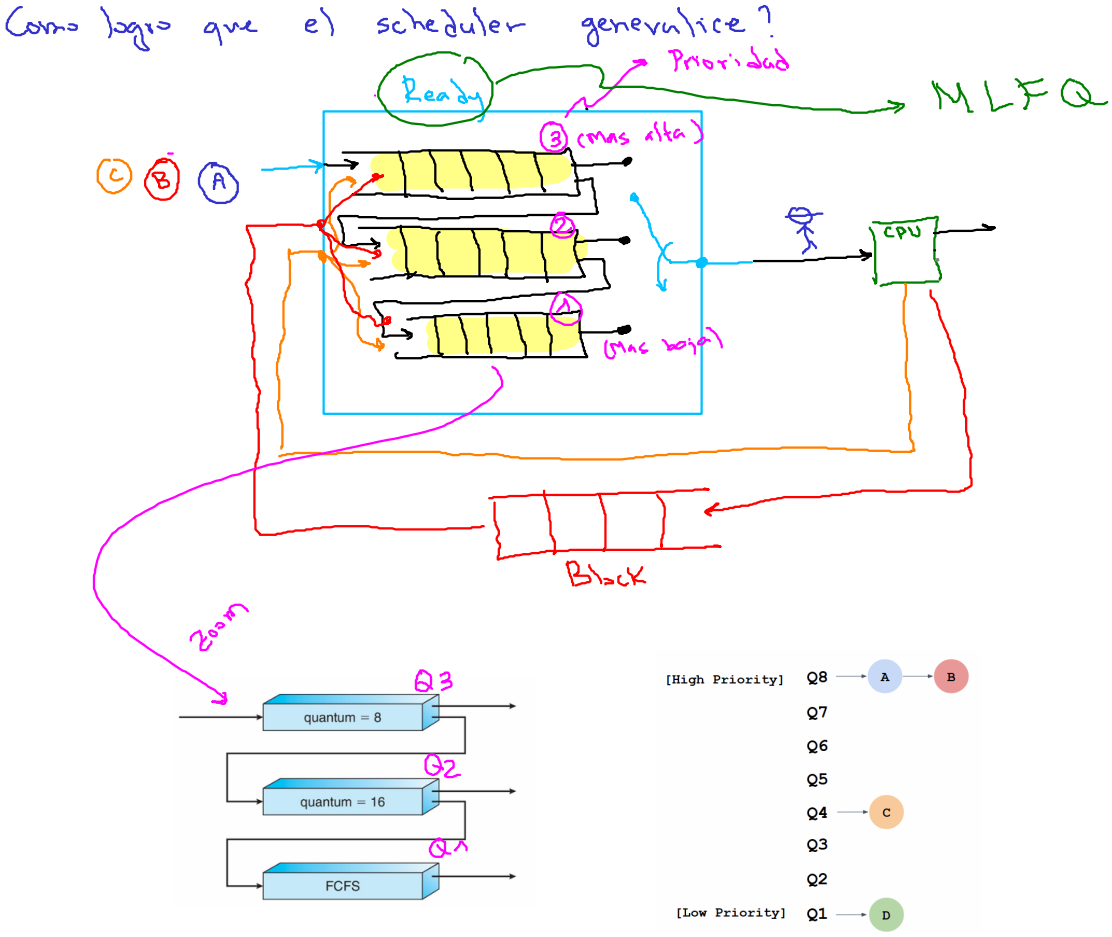
03/04/2025 - Sistemas Operativos (Ude@)

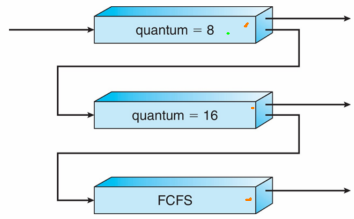
1. MLFQ (Multilevel feedback Queue)

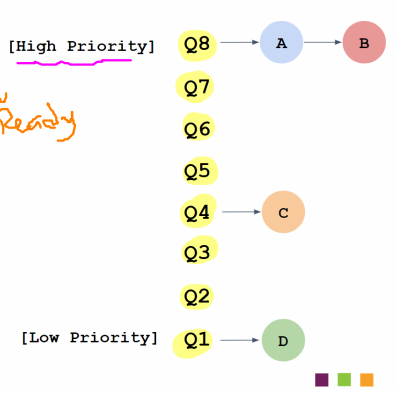




Conceptos importantes:

- Varias colas de procesos cada una con un nivel de prioridad diferente.
- Un proceso que este listo para su ejecución se ingresa en una de las colas.
- Cada proceso tiene una prioridad durante cierto tiempo pero esta puede cambiar.





1. Decir cual va a usar la CPU
2. Mover los procesos entre las colas

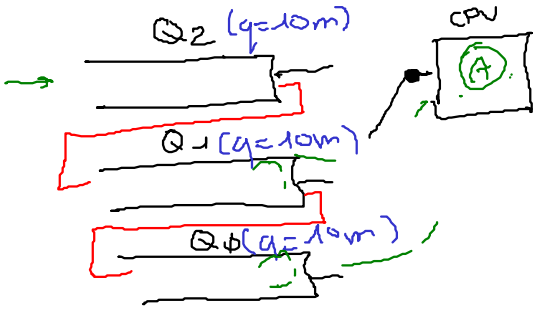
Reglas

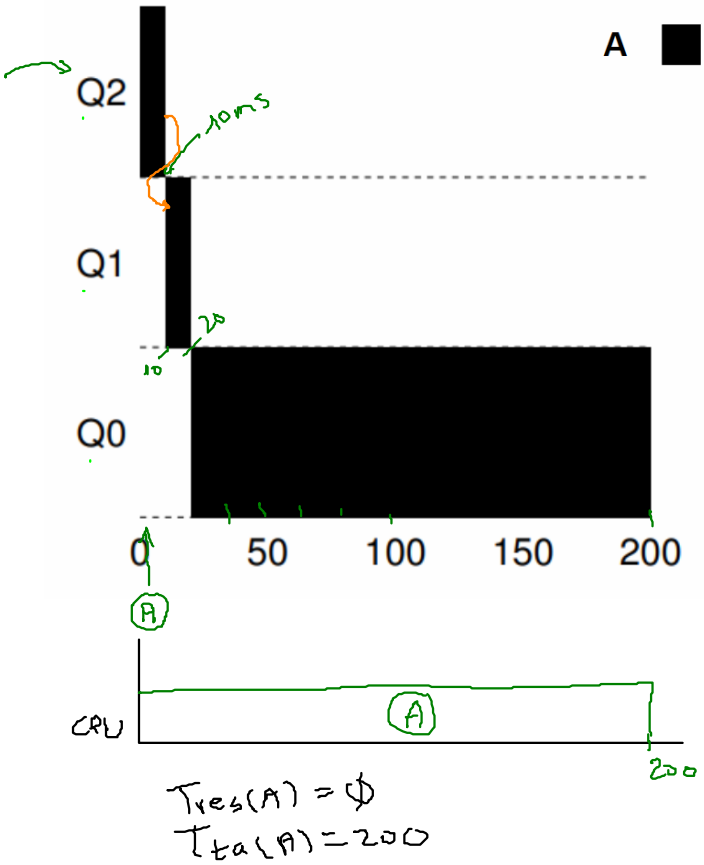
MLFQ: Reglas básicas

- **Regla 1**: Si **prioridad(A) > prioridad(B)**; **A** se ejecuta.
- **Regla 2**: Si **prioridad(A) = prioridad(B)**; **RR** para **A** y **B**.
- **Regla 3**: Cuando un trabajo llega al sistema es ubicado en la cola con la prioridad más alta.
- **Regla 4a**: Si el trabajo usa completamente el quantum de tiempo, se reduce su prioridad.
- **Regla 4b**: Si el trabajo entrega la CPU antes de finalizar su quantum de tiempo, mantiene el mismo nivel de prioridad.
- **Regla 5**: Después de un tiempo **S**, mueva todos los trabajos al mayor nivel de prioridad

Ejemplo 1 - Proceso CPU bound

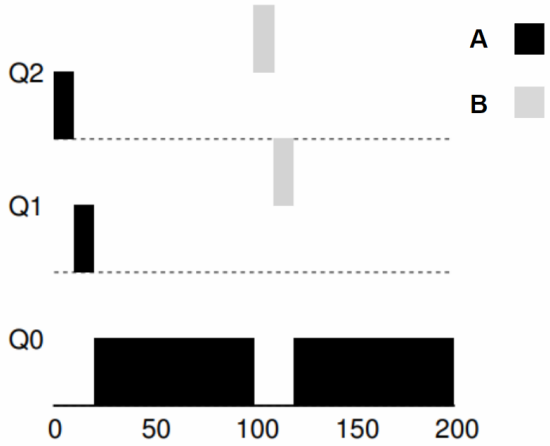
- MLFQ de 3 colas (Q2, Q1, Q0).
- Quantum de 10 ms para cada cola.
- Solo un proceso: A.

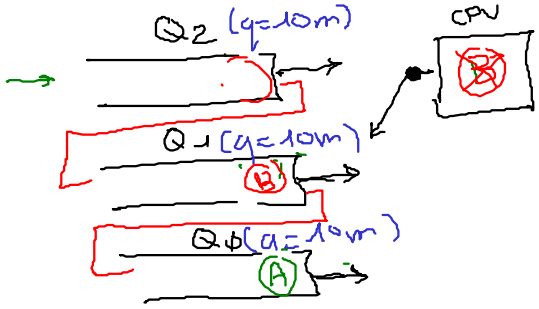


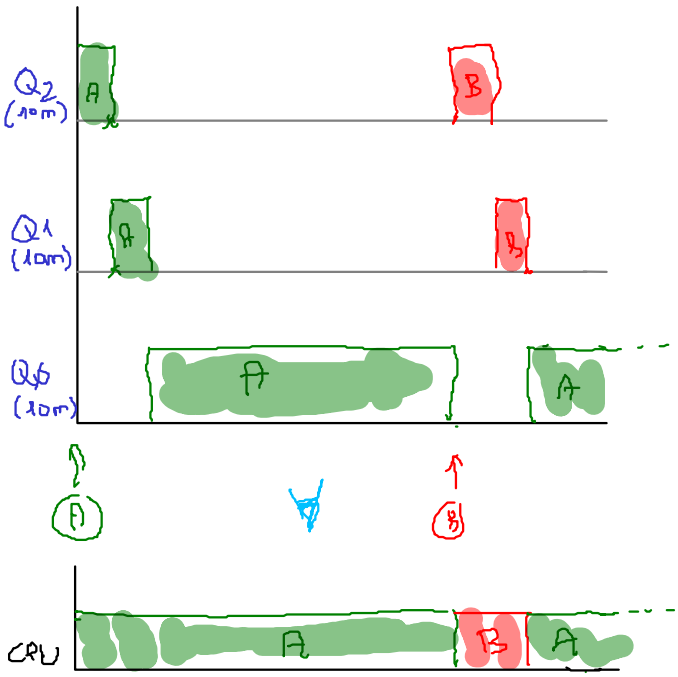


Ejemplo 2 - Llega un proceso corto

- MLFQ de 3 colas (Q2, Q1, Q0).
- Quantum de 10 ms para cada cola.
- Procesos:
  - **A**:
    - Proceso con uso intensivo de la CPU (CPU-bound).
  - **B**:
    - Proceso interactivo.
    - Tiempo de ejecución de 20 ms.
    - Llega en t = 100

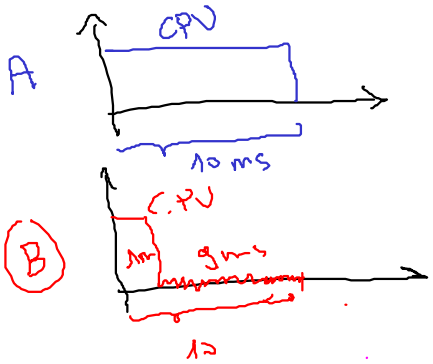


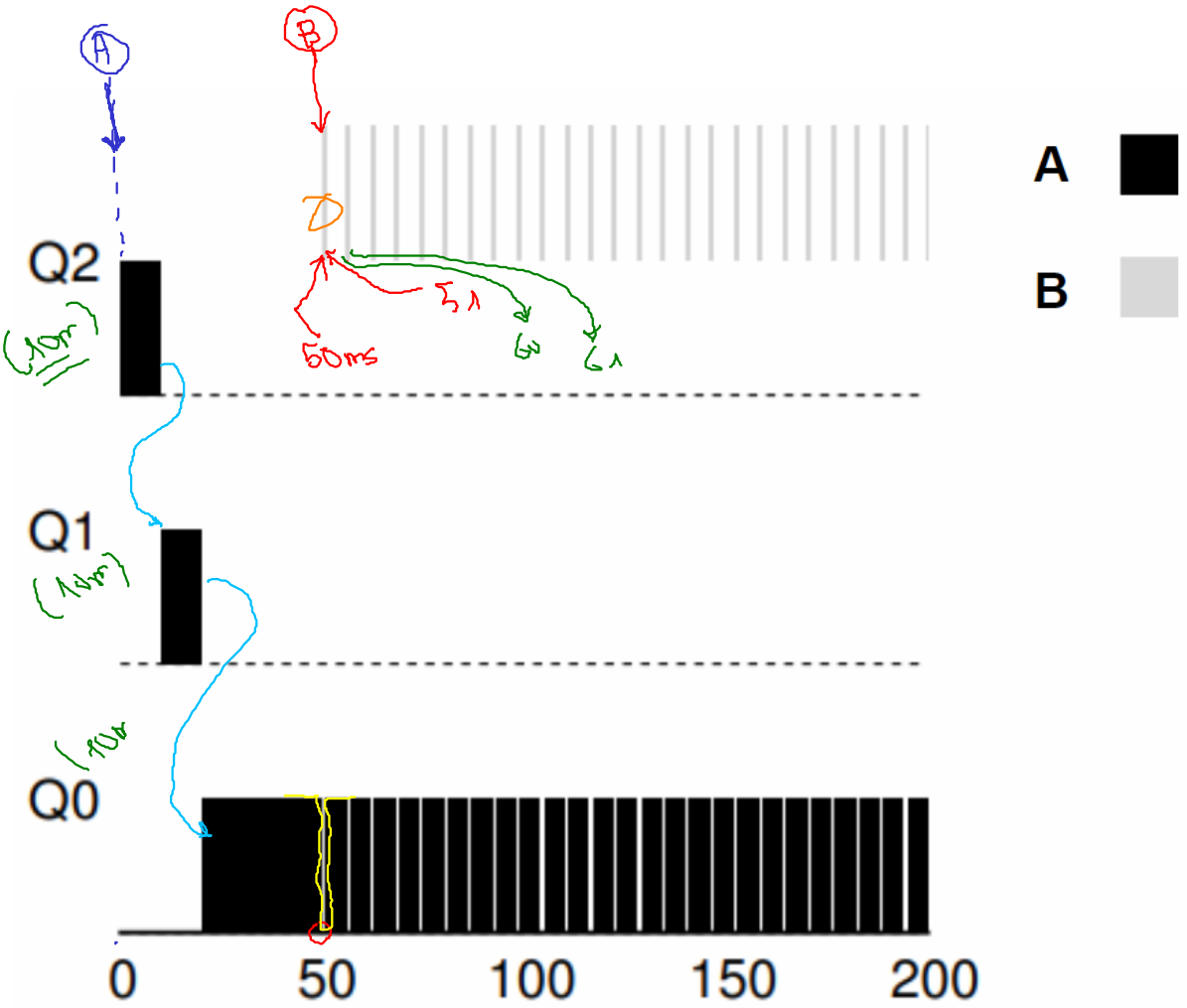




Ejemplo 3 - Proceso I/O bound

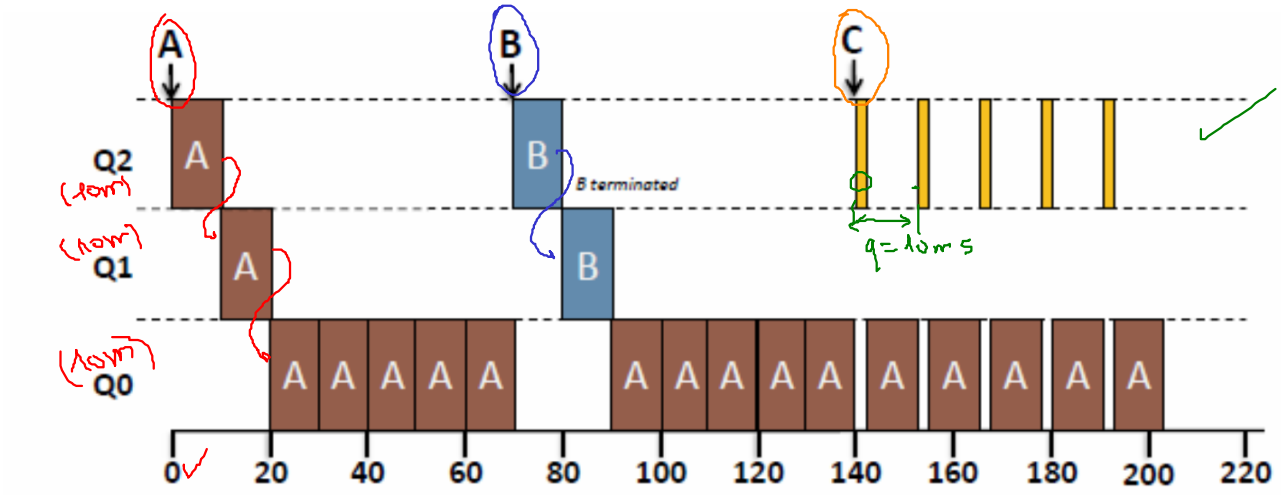
- MLFQ de 3 colas (Q2, Q1, Q0).
- Quantum de 10 ms para cada cola.
- Procesos:
  - **A**:
    - Proceso con uso intensivo de la CPU (CPU-bound).
  - **B**:
    - Proceso I/O bound.
    - Usa la CPU 1 ms y realiza una operación I/O.
    - Llega en t = 50 ms.





A Mixed I/O-intensive and CPU-intensive Workload

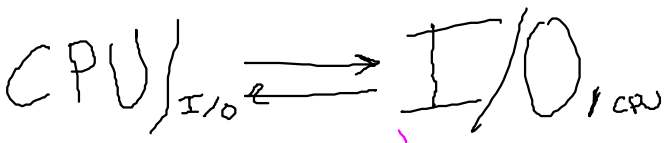
Resumen y comparación



Hecha la ley hecha la trampa...

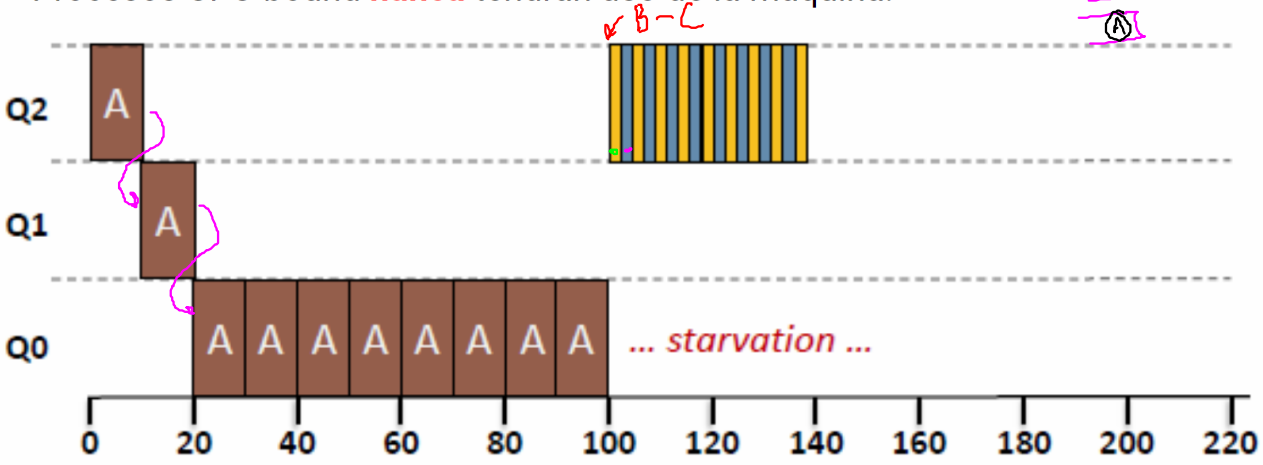
Problemas:

1. Cambio del comportamiento en el tiempo de un proceso



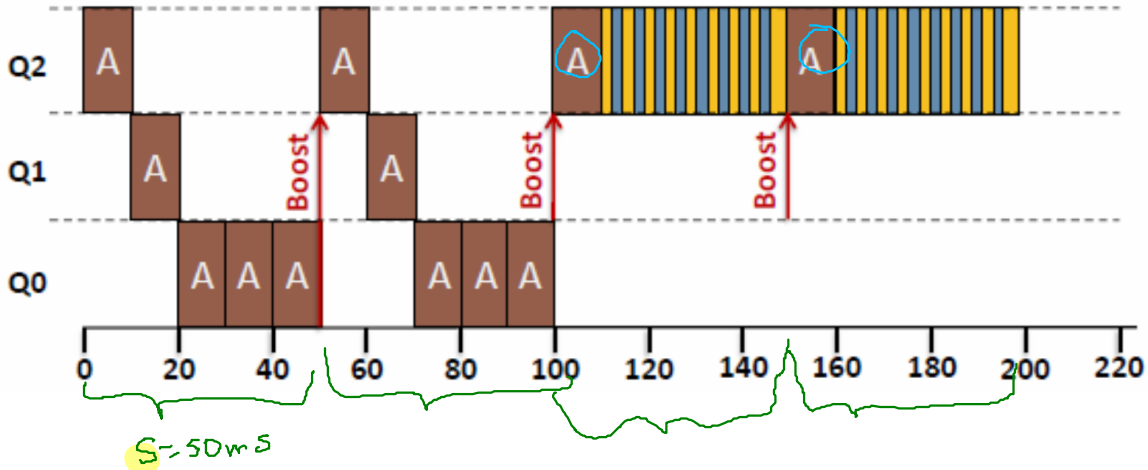
Problema: Inanición (starvation)

- Muchos procesos interactivos.
- Procesos CPU-bound **nunca** tendrán uso de la máquina.



Solución: Aging - Mejora de prioridad

Regla 5: Después de un tiempo **S**, mueva todos los trabajos al mayor nivel de prioridad



Hasta el momento retomando las reglas tenemos:

MLFQ - Clásico: Reglas basicas

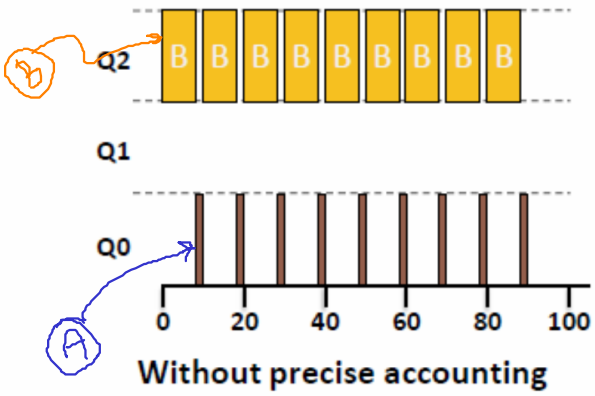
- **Regla 1**: Si **prioridad(A) > prioridad(B)**; **A** se ejecuta.
- **Regla 2**: Si **prioridad(A) = prioridad(B)**; **RR** para **A** y **B**.
- **Regla 3**: Cuando un trabajo llega al sistema es ubicado en la cola con la prioridad más alta.
- **Regla 4a**: Si el trabajo usa completamente el quantum de tiempo, se reduce su prioridad.
- **Regla 4b**: Si el trabajo entrega la CPU antes de finalizar su quantum de tiempo, mantiene el mismo nivel de prioridad.
- **Regla 5**: Después de un tiempo **S**, mueva todos los trabajos al mayor nivel de prioridad

2. Engañar al planificador

Problema: Engañar al planificador (Game the scheduler).

- Un proceso malicioso puede engañar al planificador liberando la CPU justo antes de que expire el quantum (intervalo de tiempo).
  - Proceso se ejecuta el **99% del quantum** y luego realiza una operación I/O.
  - Obtiene mayor uso de la máquina.

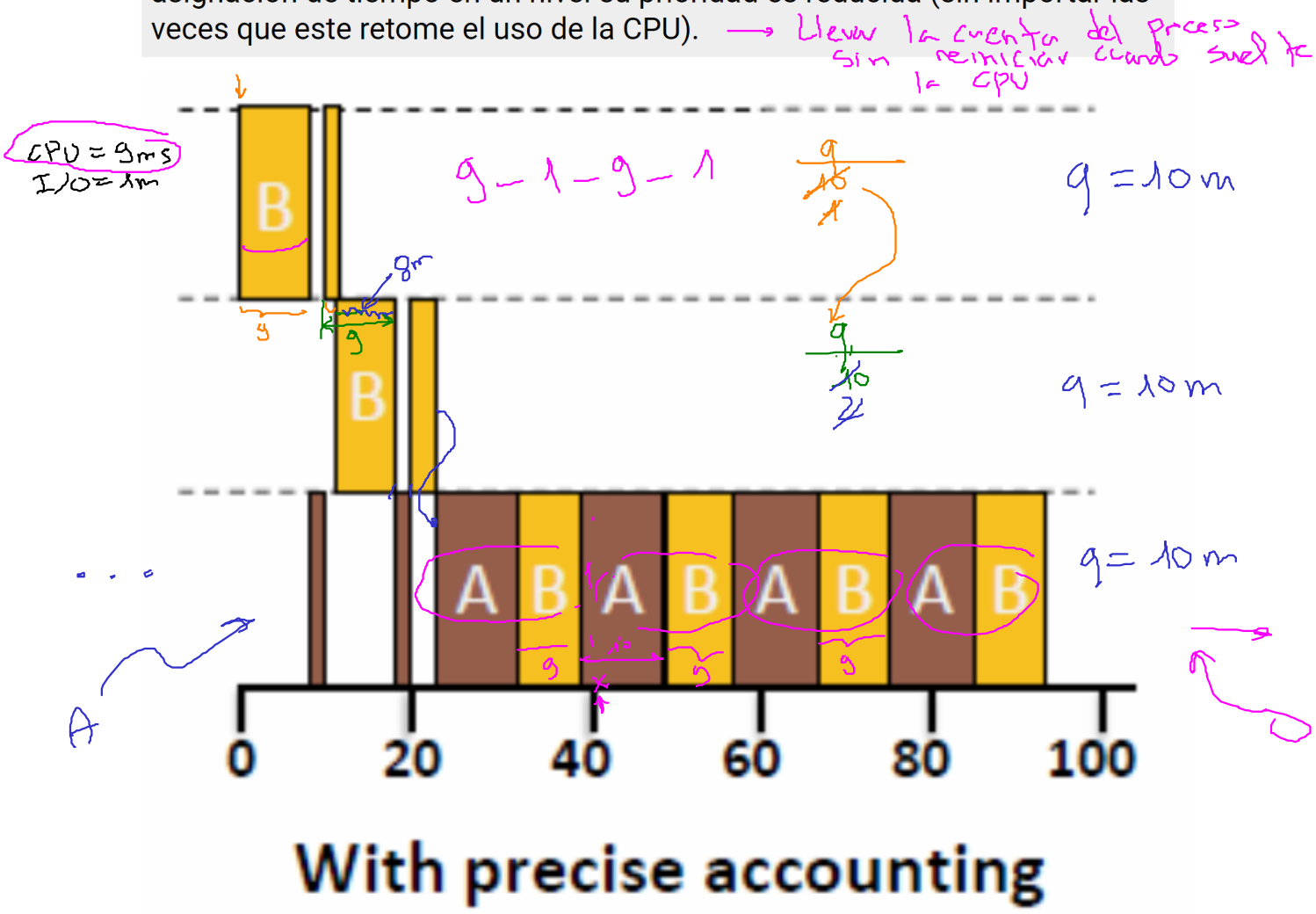
100% = 10ms
99% = 9.9ms



Without precise accounting

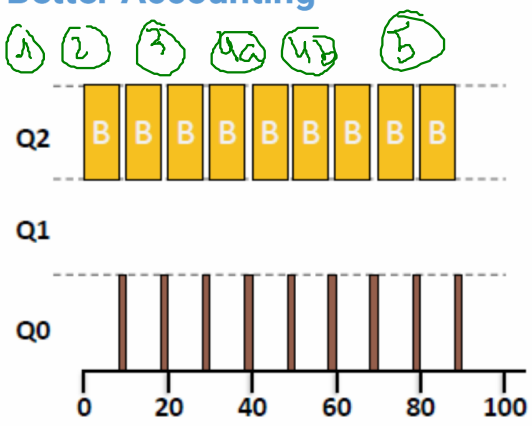
Solución: Reescribir la Regla 4a y 4b por una nueva regla 4

Nueva regla 4 (reemplaza 4a y 4b): Cuando un trabajo consume su asignación de tiempo en un nivel su prioridad es reducida (sin importar las veces que este retome el uso de la CPU).

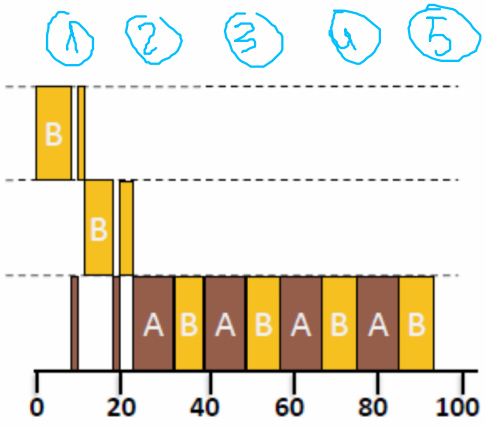


With precise accounting

Better Accounting



Without precise accounting



With precise accounting

Mejoras MLFQ

Mejoras al MLFQ

- Menor nivel de prioridad → mayor tiempo de quantum
  - **Alta prioridad**: quantum corto (~10 ms)
  - **Baja prioridad**: quantum largo (~100 ms)