Políticas de planificación de la CPU

- Objetivos:
  - » Buen rendimiento (productividad)
  - » Buen servicio
- Para saber si un proceso obtiene un buen servicio, definiremos un conjunto de medidas: dado un proceso *P*, que necesita un tiempo de servicio (ráfaga) *t*
  - *Tiempo de respuesta (T)*-- tiempo transcurrido desde que se remite una solicitud (entra en la cola de preparados) hasta que se produce la primera respuesta (no se considera el tiempo que tarda en dar la salida)
  - *Tiempo de espera (M)*-- tiempo que un proceso ha estado esperando en la cola de preparados: T - t
  - *Penalización (P)* -- T / t
  - *Indice de respuesta (R)* -- t / T : fracción de tiempo que P está recibiendo servicio.

Políticas de planificación de la CPU (y II)

- Otras medidas interesantes son:

- *Tiempo del núcleo* -- tiempo perdido por el SO tomando decisiones que afectan a la planificación de procesos y haciendo los cambios de contexto necesarios. En un sistema eficiente debe representar entre el 10% y el 30% del total del tiempo del procesador
- *Tiempo de inactividad* -- tiempo en el que la cola de ejecutables está vacía y no se realiza ningún trabajo productivo
- *Tiempo de retorno* – cantidad de tiempo necesario para ejecutar un proceso completo

Políticas de planificación de CPU (y III)

- Las políticas de planificación se comportan de distinta manera dependiendo de la clase de procesos
  - » Ninguna política de planificación es completamente satisfactoria, cualquier mejora en una clase de procesos es a expensas de perder eficiencia en los procesos de otra clase.
- Podemos clasificarlas en:
  - » **No apropiativas (sin desplazamiento)**: una vez que se le asigna el procesador a un proceso, no se le puede retirar hasta que éste voluntariamente lo deje (finalice o se bloquee)
  - » **Apropiativas (con desplazamiento)**: al contrario, el SO puede apropiarse del procesador cuando lo decida

Políticas de planificación de la CPU (y IV)

- FCFS
- El más corto primero:
  - » no apropiativo
  - » apropiativo
- Planificación por prioridades
- Por turnos (*Round-Robin*)
- Colas múltiples
- Colas múltiples con realimentación.

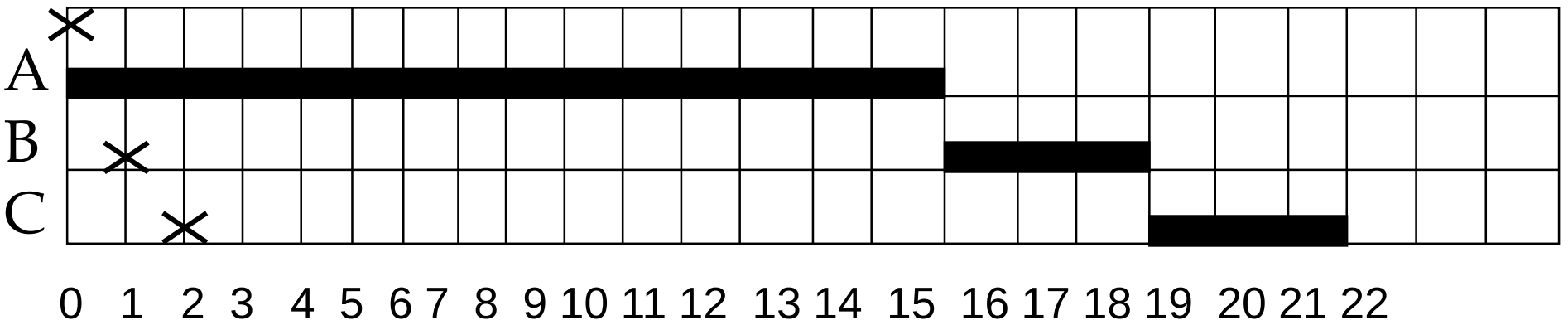
FCFS (*First Come First Served)*

- Los procesos son servidos según el orden de llegada a la cola de ejecutables.
- Es *no apropiativo*, cada proceso se ejecutará hasta que finalice o se bloquee.
- Fácil de implementar pero pobre en cuanto a prestaciones.
- Todos los procesos pierden la misma cantidad de tiempo esperando en la cola de ejecutables independientemente de sus necesidades.
- Procesos cortos muy penalizados.
- Procesos largos poco penalizados.

FCFS (y II)

Procesos	Tº llegada	Ráfaga	T(tº respuesta)	M(tº espera)
A	0	15	15-0=15	0
B	1	3	18-1=17	14
C	2	3	21-2=19	16

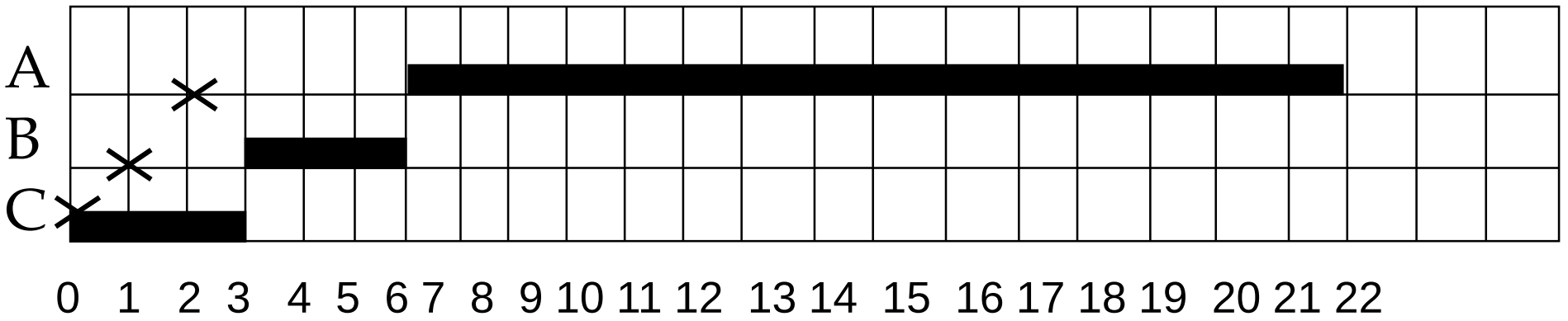
Diagrama de ocupación de la CPU



FCFS (y III)

Procesos	Tº llegada	Ráfaga	T (tº respuesta)	M (tº espera)
A	2	15	21-2 = 19	4
B	1	3	6-1 = 5	2
C	0	3	3-0 = 3	0

Diagrama de ocupación de la CPU



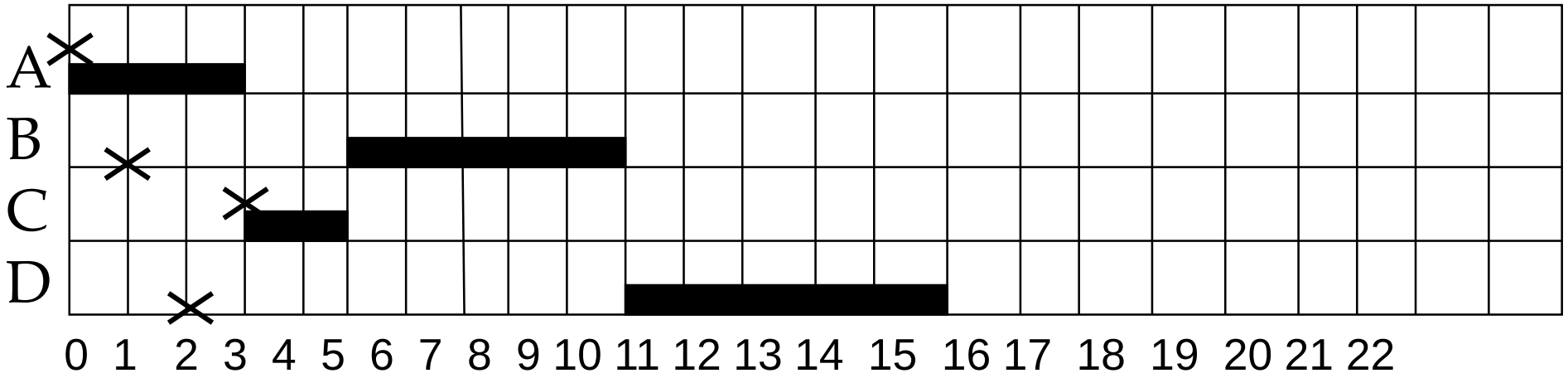
El más corto primero (SJF)

- Es *no apropiativo*.
- Cuando el procesador queda libre, selecciona el proceso que requiera un tiempo de servicio menor.
- Si existen dos o más procesos en igualdad de condiciones, se sigue FCFS.
- Necesita conocer explícitamente el tiempo estimado de ejecución (tº servicio) ¿Cómo?.
- Disminuye el tiempo de respuesta para los procesos cortos y discrimina a los largos.
- Tiempo medio de espera bajo.

El más corto primero (y II)

Procesos	Tº llegada	Ráfaga real	Ráfaga estimada
A	0	3	3
B	1	5	5
C	3	2	2
D	2	5	5

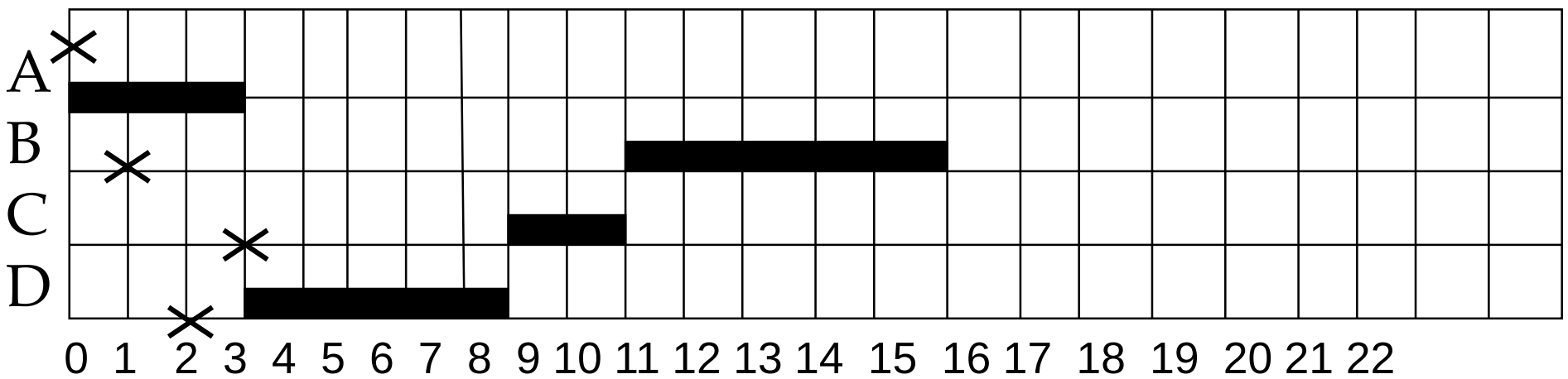
Diagrama de ocupación de la CPU



El más corto primero (y III)

Procesos	Tº llegada	Ráfaga real	Ráfaga estimada
A	0	3	3
B	1	5	6
C	3	2	5
D	2	5	5

Diagrama de ocupación de la CPU



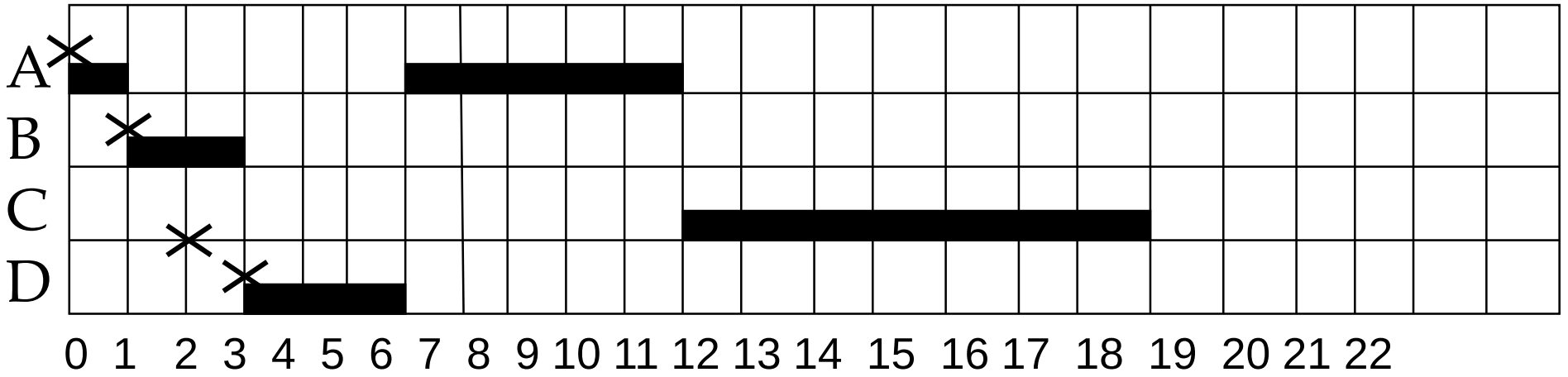
El más corto primero apropiativo (SRTF)

- Cada vez que entra un proceso a la cola de ejecutables se comprueba si su tiempo de servicio es menor que el tiempo de servicio que le queda al proceso que está ejecutándose. Casos:
  - » **Si es menor**: se realiza un cambio de contexto y el proceso con menor tiempo de servicio es el que se ejecuta.
  - » **No es menor**: continúa el proceso que estaba ejecutándose.
- El tiempo de respuesta es menor excepto para procesos muy largos.
- Se obtiene la menor penalización en promedio (mantiene la cola de ejecutables con la mínima ocupación posible).

El más corto primero apropiativo (y II)

Procesos	Tº llegada	Ráfaga real	Ráfaga estimada
A	0	6	6
B	1	2	2
C	2	7	7
D	3	3	3

Diagrama de ocupación de la CPU



¿Cómo podemos conocer la duración de la siguiente ráfaga de CPU?

- Sólo podemos estimar su duración.
- Podemos estimar el valor de la siguiente ráfaga basándonos en las ráfagas previas de CPU y utilizando una media exponencial, por ejemplo, para la n+1 ráfaga:

  T_n = duración actual de la n-ésima ráfaga

  Y_n = valor estimado de la n-ésima ráfaga

  $0 <= W <= 1$

  Definimos: $Y_{n+1} = W * T_n + (1-W)\ Y_n$

  Y_0 = valor inicial (constante o promedio global del sistema)

Ejemplos

- W = 0

  $Y_{n+1} = Y_n$ -> La historia reciente no influye.
- W = 1

  $Y_{n+1} = T_n$ -> Sólo cuenta la ráfaga actual.
- Si desarrollamos la formula:

  $$Y_{n+1} = W * T_n + (1-W) * W * T_{n-1} + (1-W)^2 * W * T_{n-2} + \ldots + (1-W)^q * W * T_{n-q}$$

  Si W = ½ -> cada término sucesivo tiene menos efecto.

Planificación por prioridades

- Asociamos a cada proceso un número de prioridad (entero).
- Se asigna la CPU al proceso con mayor prioridad (enteros menores = mayor prioridad)
  - » Apropiativa
  - » No apropiativa
- **Problema**: Inanición -- los procesos de baja prioridad pueden no ejecutarse nunca.
- **Solución**: Envejecimiento -- con el paso del tiempo se incrementa la prioridad de los procesos.

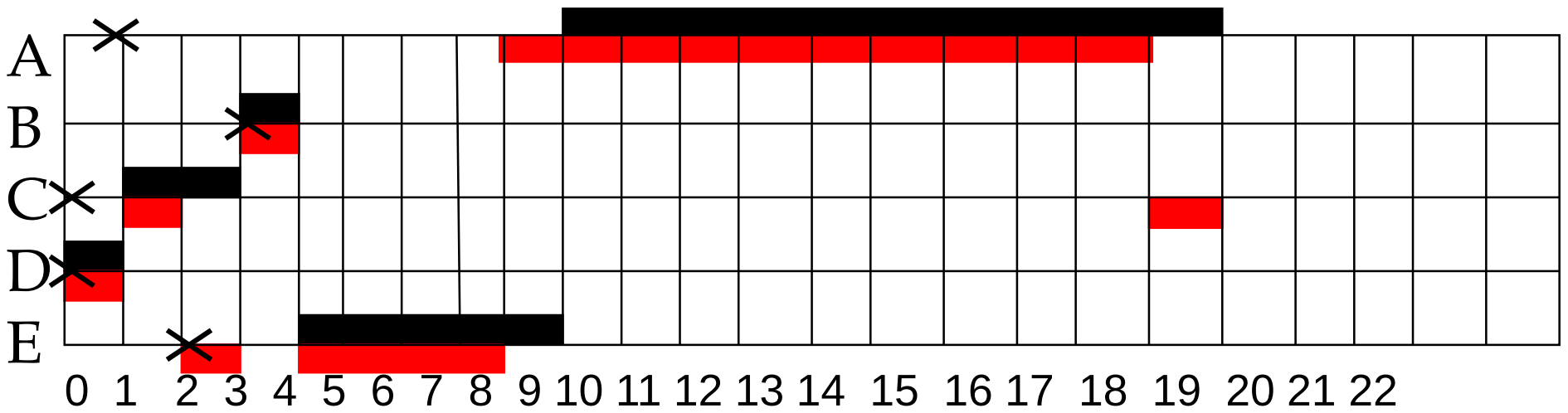
Planificación por prioridades (y II)

Procesos	Tº llegada	Ráfaga	Prioridad
A	1	10	3
B	3	1	1
C	0	2	3
D	0	1	2
E	2	5	2

No apropiativo

Apropiativo

Diagrama de ocupación de la CPU



Por Turnos (Round-Robin)

- La CPU se asigna a los procesos en intervalos de tiempo (quatum).
- Procedimiento:
  - » Si el proceso finaliza o se bloquea antes de agotar el quantum, libera la CPU. Se toma el siguiente proceso de la cola de ejecutables (la cola es FIFO) y se le asigna un quantum completo.
  - » Si el proceso no termina durante ese quantum, se interrumpe y se coloca al final de la cola de ejecutables.
- Es apropiativo.

***Nota**: En los ejemplos* supondremos que si un proceso *A* llega a la cola de ejecutables al mismo tiempo que otro *B* agota su quantum, la llegada de *A* a la cola de ejecutables ocurre antes de que *B* se incorpore a ella.

Por Turnos (y II)

Procesos	Tº llegada	Ráfaga
A	0	3
B	1	9
C	3	2
D	9	5

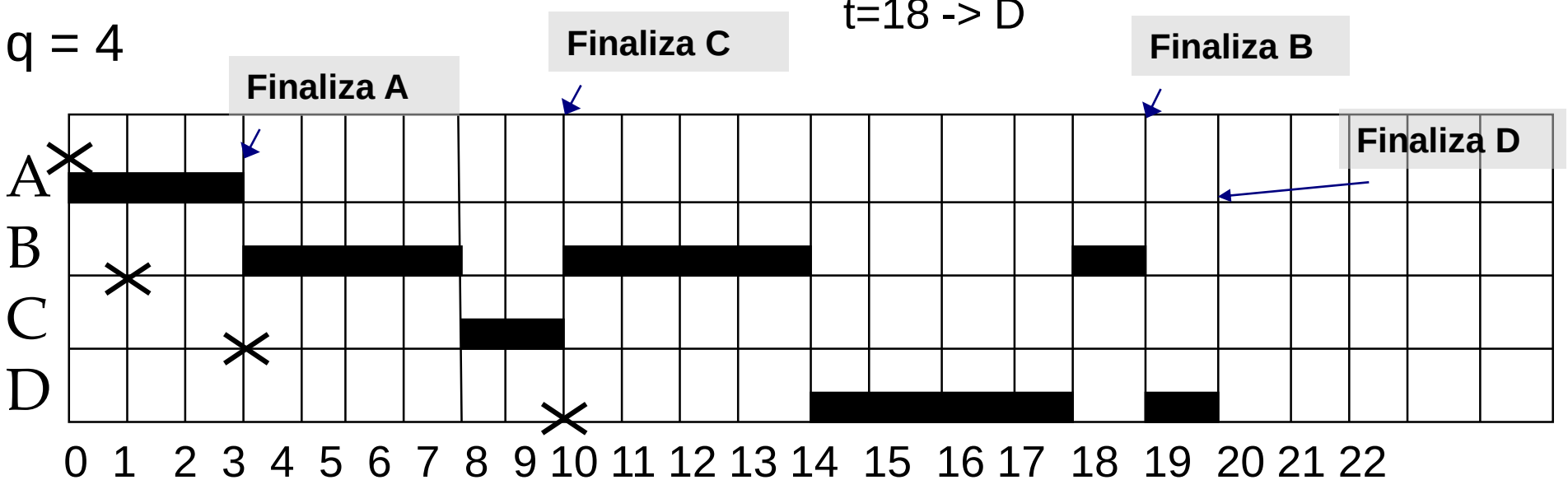
Estado de la cola de ejecutables

t=0 -> A; t=3 -> BC; t=7 -> CB

t=9 -> BD; t=13 -> DB; t=17 -> BD

t=18 -> D

q = 4



Por Turnos (y III)

- Los valores típicos del quantum están entre 1/60sg y 1sg.
- Penaliza a todos los procesos en la misma cantidad, sin importar si son cortos o largos.
- Las ráfagas muy cortas están más penalizadas de lo deseable.
- ¿valor del quantum?

  - muy grande (excede del tº de servicio de todos los procesos) -> se convierte en FCFS

  - muy pequeño -> el sistema monopoliza la CPU haciendo cambios de contexto (tº del núcleo muy alto)

Colas múltiples

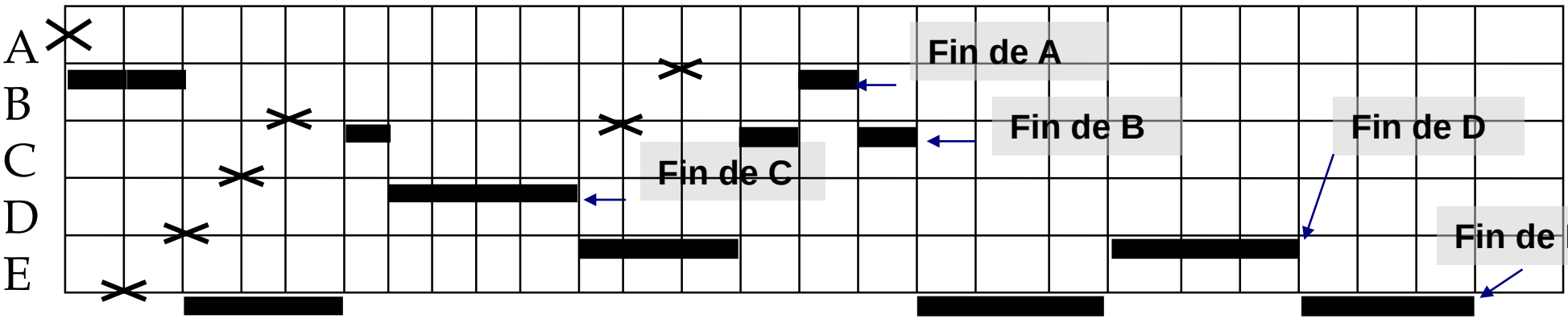
- La cola de preparados se divide en varias colas y cada proceso es asignado permanentemente a una cola concreta P. ej. interactivos y batch
- Cada cola puede tener su propio algoritmo de planificación P. ej. interactivos con RR y batch con FCFS
- Requiere una planificación entre colas
  - » Planificación con prioridades fijas. P. ej. primero servimos a los interactivos luego a los batch
  - » Tiempo compartido -- cada cola obtiene cierto tiempo de CPU que debe repartir entre sus procesos. P. ej. 80% interactivos en RR y 20% a los batch con FCFS

Colas múltiples (y II)

P	TºLlegada	Ráfaga	Bloqueo	Ráfaga	Cola
A	0	2	10	1	1
B	4	1	5	2	1
C	3	4	-	-	2
D	2	6	-	-	3
E	1	9	-	-	3

El algoritmo de cada cola es **RR** con **q=1, 2 y 3** mlsg. respectivamente.

El algoritmo entre colas es **prioridades no apropiativo**



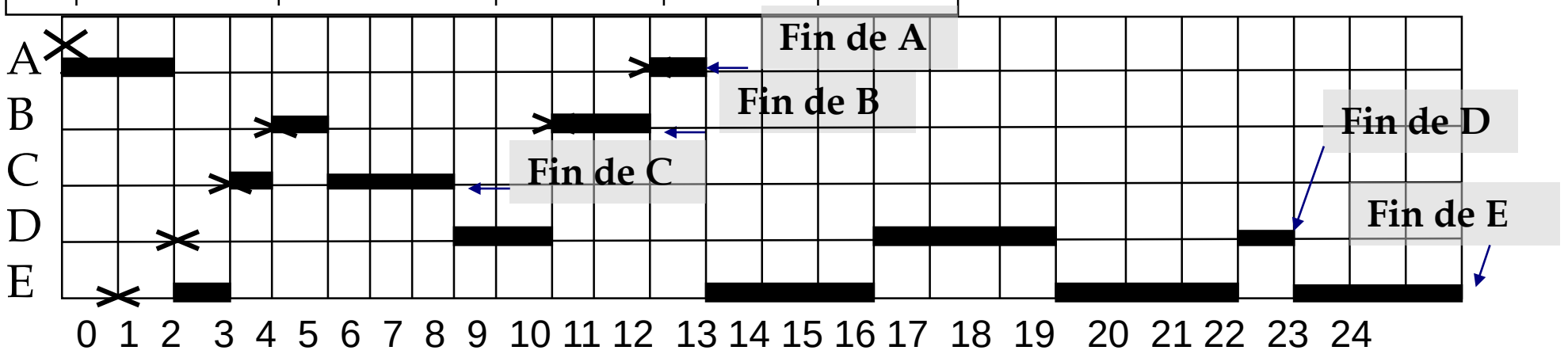
t = 0	1	2	5	6	8	10	13	14	15	16	19	22
Cola 1	A	A		B			BA	AB	B			
Cola 2				C	C	C						
Cola 3		E	ED	DE	DE	DE	DE	ED	ED	ED	ED	DE	E

Colas múltiples (y III)

P	TºLlegada	Ráfaga	Bloqueo	Ráfaga	Cola
A	0	2	10	1	1
B	4	1	5	2	1
C	3	4	-	-	2
D	2	6	-	-	3
E	1	9	-	-	3

El algoritmo de cada cola es **RR** con **q=1, 2 y 3** mlsg. respectivamente.

El algoritmo entre colas es **prioridades *apropiativo.***



t =	0	1	2	3	4	5	7	8	10	11	12	13	16	19	22	23
Cola 1	A	A			B				B	B	A					
Cola 2				C	C	C	C									
Cola 3		E	ED	DE	DE	DE	DE	DE	ED	ED	ED	ED	DE	ED	DE	E

Colas múltiples con realimentación

- Un proceso se puede mover entre varias colas
- Requiere definir los siguientes parámetros:
  - » Número de colas
  - » Algoritmo de planificación para cada cola
  - » Método utilizado para determinar cuando trasladar a un proceso a otra cola
  - » Método utilizado para determinar en qué cola se introducirá un proceso cuando necesite un servicio
  - » Algoritmo de planificación entre colas
- Mide en tiempo de ejecución el comportamiento real de los procesos
- Disciplina de planificación más general (Unix, Windows NT)

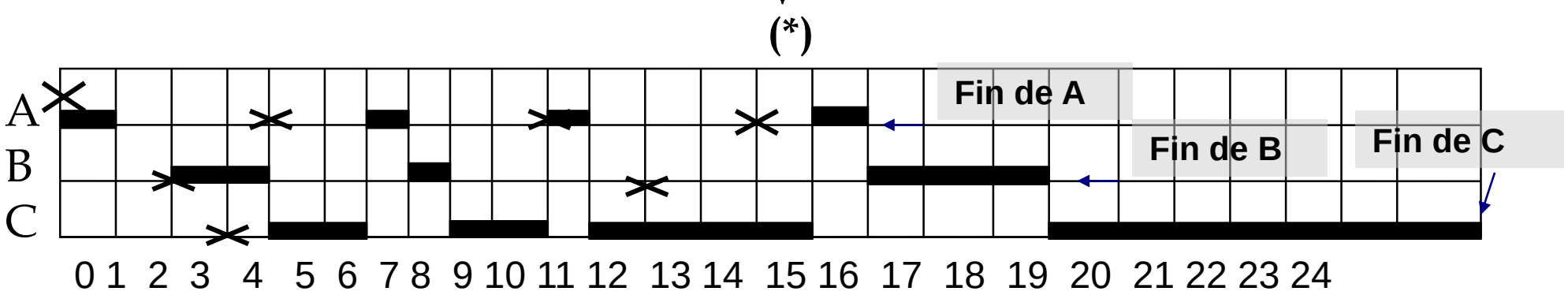
Ejemplo de colas múltiples con realimentación

- Tres colas gestionadas mediante **Round Robin**:
  - » Cola 1 con quantum = 2 milisegundos
  - » Cola 2 con quantum = 4 milisegundos
  - » Cola 3 con quantum = 8 milisegundos
- Algoritmo entre colas: **prioridades no apropiativo**, cola 1 mayor prioridad, cola 3 menor prioridad.
- Los procesos entran inicialmente en la cola 1
- Cuando un proceso se bloquea, al regresar a la cola de ejecutables sigue en la misma cola si no hay traspaso
- Un proceso se traspasa a una cola de menor prioridad cuando agota dos quantum de tiempo seguidos, o bien, agota uno y se bloquea en el siguiente

Ejemplo de colas múltiples con realimentación (y II)

P	TºLlegada	TºServicio total	Ráfaga	Bloqueo
A	0	4	1	3
B	2	6	3	4
C	3	23	-	-

(*) Los procesos tienen un comportamiento **cíclico**



t = 0	1	2	4	6	7	8	10	11	15	16	19	23
Cola 1	A		B	CAB	ABC	BC	C	A		A		
Cola 2								C	C	BC	BC	C
Cola 3												C