Entrega 5 - Sistemas Operativos II

Delfina Martín - Ignacio Litmanovich - Agustín Diaz

Abril 2021

Ejercicio 1

Entre los objetivos de la planificación de procesos que menciona en libro se encuentra el de **favorecer el uso esperado del sistema**, es decir, maximizar la prioridad de los procesos que sirvan a solicitudes de usuarixs interactivxs. Esto entra en conflicto con el modelo teórico del CFS (completely fair scheduler) ya que al priorizar ciertos procesos (en particular **los interactivos**) no es posible repartir el tiempo de uso de la CPU equitativamente entre las tareas listas.

De forma análoga a lo que ocurre con los procesos interactivos, priorizar a los **procesos que puedan causar bloqueos** o favorecer a los **procesos con comportamiento deseable** también entra en conflicto con el modelo teórico CFS.

El modelo CFS tampoco cumple con la propiedad de **ser predecible**: el tiempo que se ejecuta un proceso con ciertas características depende fuertemente de la carga actual del sistema (CFS asigna tiempo de CPU equitativamente) y este puede no coincidir con el tiempo que se ejecuta un proceso con características similares en un instante en el que la carga del sistema es diferente.

Por último, como los procesos no son penalizados por su consumo de recursos en CFS, podemos ver que el modelo tampoco **equilibra el uso de los recursos**.

Ejercicio 2

El CFS beneficia a los procesos interactivos como se explica a continuación.

El algoritmo base (sin tener en cuentra prioridad de procesos) toma como base un "fair lock" (poder de cómputo asignado a una tarea en una CPU multitarea, ideal y precisa). Además, cada tarea tiene un wait_runtime asociado: tiempo (del fair lock) que estuvo esperando a la CPU. Los cambios de contextos se realizan cuando alguna de las tareas en espera tiene mayor wait_runtime que la tarea en ejecución.

Este algoritmo se implementa usando un red black tree y la cuenta fair lock - wait_runtime para ordenarlo. La tarea que más necesidad de tiempo tiene (la siguiente en ejecutarse) será el nodo más a la izquierda en el árbol.

Para poder tener en cuenta las prioridades de los procesos, se asocia un peso al wait_runtime (que luego será usado para calcular la diferencia). De esta forma, el wait_runtime de las tareas de baja prioridad expira más rápido que el de las de alta prioridad, o lo que es lo mismo decir: el tiempo para las tareas de baja prioridad pasa más rápido. Pero este mecanismo no alcanza para distinguir los procesos interactivos de los no interactivos.

Para suplir esta falta, CFS comienza a implementar un mecanismo que permite agrupar tareas por usuario, según la jerarquía de procesos o haciendo una combinación de ambos. Los procesos según [1], se organizan en clases y en particular los interactivos se agrupan en una misma clase lo cual permite que tengan misma prioridad para diferentes usuarixs.

Ejercicio 3

Algunas aclaraciones generales:

- El color del segmento indica el proceso que se está ejecutando.
- Los números de la fila "Ejecución" indican en qué cola está el proceso que se está ejecutando al inicio del quantum. En las dos primeras tablas ese número se corresponderá con las prioridades asignadas inicialmente a los procesos.

[1] `http://lxr.linux.no/#linux+v2.6.28.4/Documentation/scheduler/sched-design-CFS.txt`

FB (q = 2)

En este algoritmo la prioridad de un proceso no cambia a lo largo del tiempo y cada cola de prioridad tiene asignado un quantum q = 2.

- En t = 0 hay un solo proceso por lo tanto está en la cola de mayor prioridad
- En t = 2 A está en la cola de mayor prioridad y B está en la cola de menor prioridad
- En t = 4 El proceso C se agrega al final de la cola de prioridad 1 por enunciado y se ejecuta D por ser el único proceso en la cola de mayor prioridad
- En t = 10 no hay procesos en la cola de prioridad 0, A y C están en la cola de prioridad 1 (en ese orden) y B en la de prioridad 2

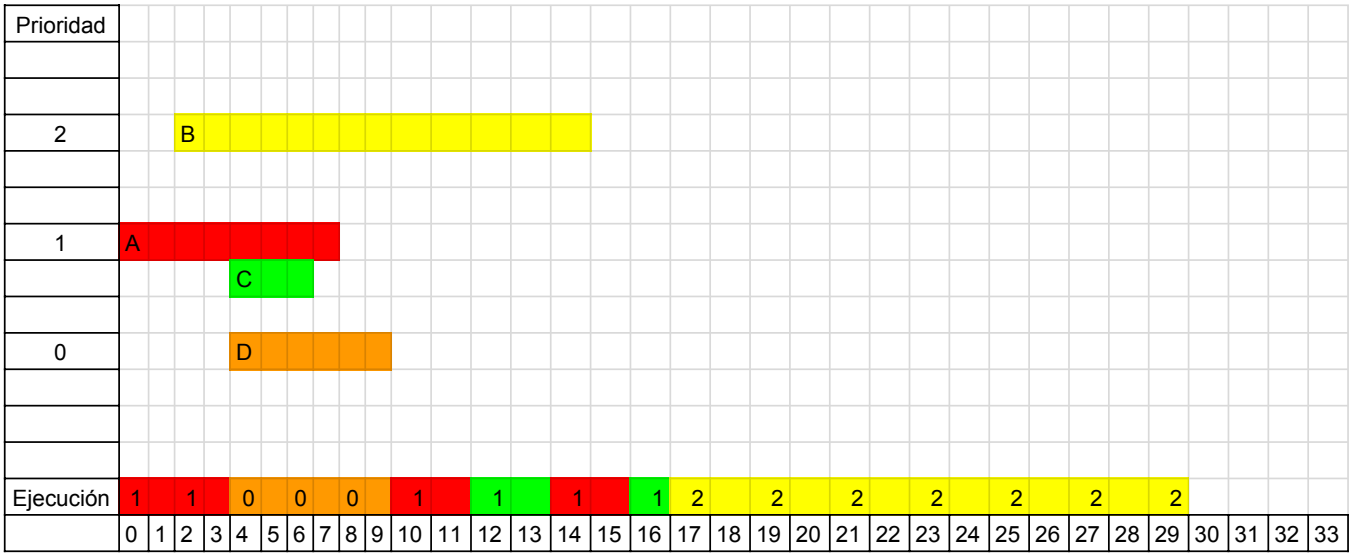

Figura 1: FB

El análisis para FB se puede ver en la tabla (1).

Proceso	Tiempo de llegada	t	Inicio	Fin	T	E	P	R
A	0	8	0	16	16	8	2	0.5
B	2	13	17	30	28	15	2.153846154	0.4642857143
C	4	3	12	17	13	10	4.333333333	0.2307692308
D	4	6	4	10	6	0	1	1
Promedio		7.5			15.75	8.25	2.371794872	0.5487637363

Tabla 1: Análisis algoritmo FB

FB - diferente quantum por cola

En este algoritmo la prioridad de un proceso no cambia a lo largo del tiempo y cada cola de prioridad tiene asignado un quantum $2^{(i+1)}$ donde i es el número de cola.

- En t = 4 El proceso C se agrega al final de la cola de prioridad 1 por enunciado

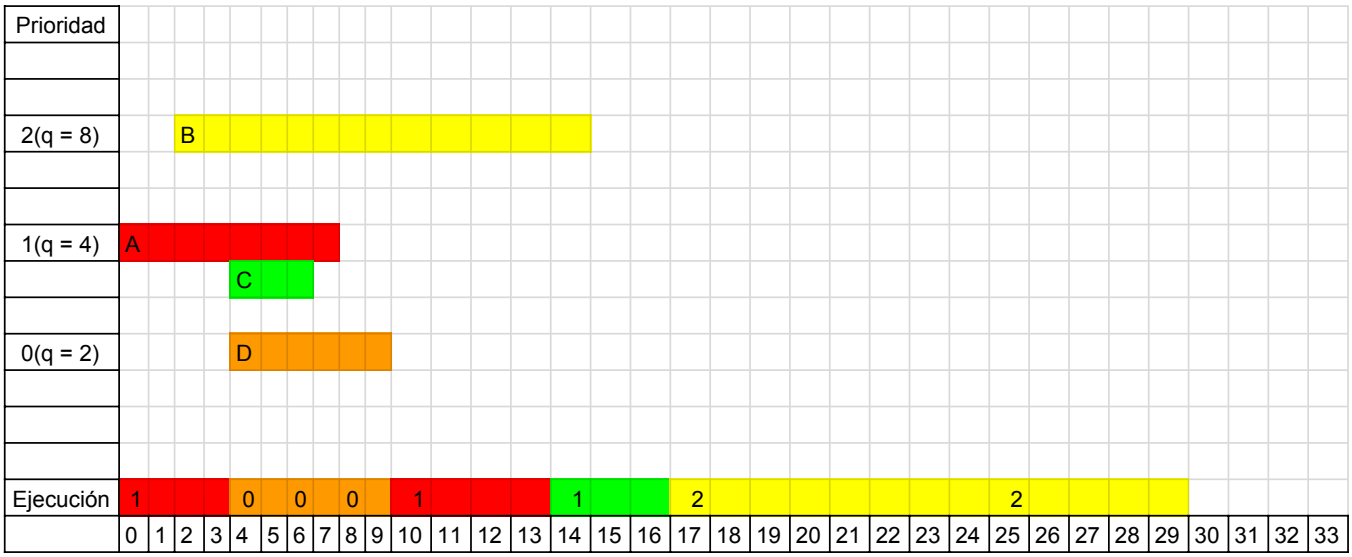

Figura 2: FB con diferente quantum por cola

El análisis para FB con diferentes quantums se puede ver en la tabla (2).

Proceso	Tiempo de llegada	t	Inicio	Fin	T	E	P	R
A	0	8	0	14	14	6	1.75	0.5714285714
B	2	13	17	30	28	15	2.153846154	0.4642857143
C	4	3	14	17	13	10	4.333333333	0.2307692308
D	4	6	4	10	6	0	1	1
Promedio		7.5			15.25	7.75	2.309294872	0.5666208791

Tabla 2: Análisis algoritmo FB con diferente quantum por cola

FB con retroalimentación

En este algoritmo la prioridad de un proceso cambia a lo largo del tiempo y cada cola de prioridad tiene asignado un quantum q = 2.

- En t = 4 A consumió 2 veces su quantum por lo tanto se degrada a la cola de prioridad 2 (y se coloca al final de la misma). Es decir, luego de t = 4 D tiene prioridad 0, C prioridad 1 y B y A prioridad 2
- En t = 4 El proceso C se agrega al final de la cola de prioridad 1 por enunciado
- En t = 8 D consumió 2 veces su quantum por lo tanto se degrada a la cola de prioridad 2 (y se coloca al final de la misma). Es decir, luego de t = 8 ningún proceso tiene prioridad 0, C y D tienen prioridad 1 y B y A prioridad 2
- En t = 17 B consumió 2 veces su quantum por lo tanto se degrada a la cola de prioridad 3 (y se coloca al final de la misma). Es decir, luego de t = 17 ningún proceso tiene prioridad 0 ni 1, A tiene prioridad 2 y B tiene prioridad 3

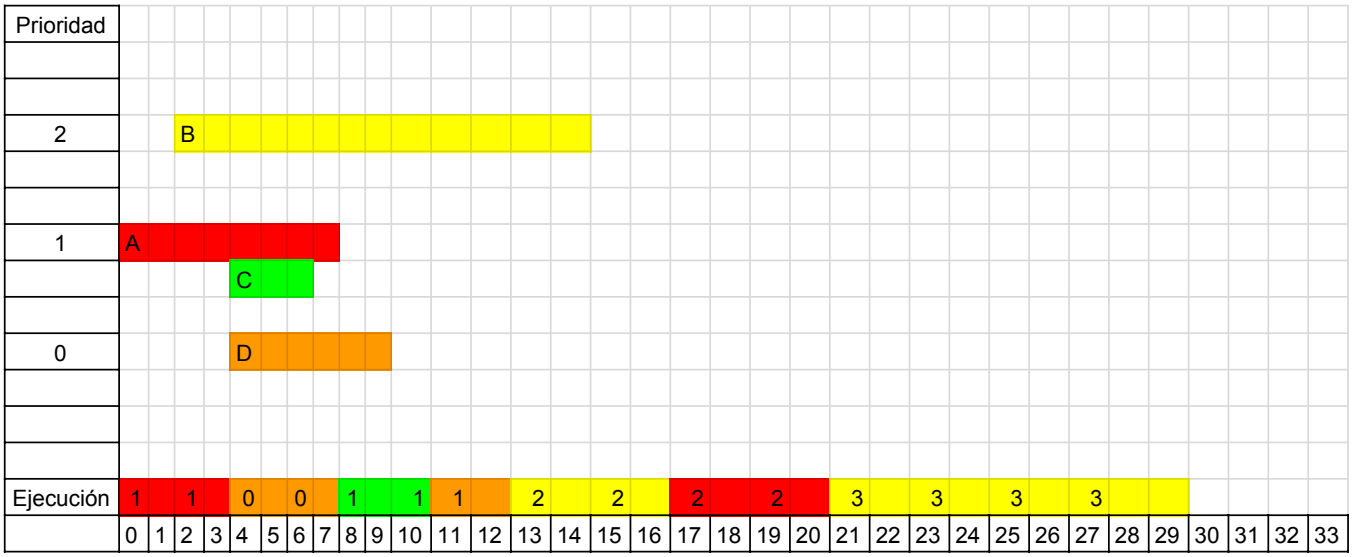

Figura 3: FB con retroalimentación

Inlcuimos también el estado de las colas en ciertos momentos relevantes en (3)

t =	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
cola 3																		B				B
cola 2			B		B>A				B>A			B>A		B>A				A				
cola 1	A		A		C				C>D			D										
cola 0					D																	

Tabla 3: Estado de las colas en FB con retroalimentación

El análisis para FB con retroalimentación se puede ver en la tabla (4).

Proceso	Tiempo de llegada	t	Inicio	Fin	T	E	P	R
A	0	8	0	21	21	13	2.625	0.380952381
B	2	13	13	30	28	15	2.153846154	0.4642857143
C	4	3	8	11	7	4	2.333333333	0.4285714286
D	4	6	4	13	9	3	1.5	0.6666666667
Promedio		7.5			16.25	8.75	2.153044872	0.4851190476

Tabla 4: Análisis algoritmo FB

Finalmente incluimos una vista comparativa de los resultados en (5)

Algoritmo	t	T	E	P	R
FB	7.5	15.75	8.25	2.371794872	0.5487637363
FB diferente quantum	7.5	15.75	8.25	2.371794872	0.5487637363
FB con retroalimentacion	7.5	16.25	8.75	2.153044872	0.4851190476

Tabla 5: Vista comparativa análisis de algoritmos

Conclusiones

En el ejemplo particular del ejercicio resulta dificil sacar conclusiones precisas dada la variación de los datos. No obstante incluimos algunos puntos a considerar.

El algoritmo FB con retroalimentación resultó tener mayor tiempo de respuesta, mayor tiempo de espera y menor proporción de respuesta frente a los otros dos algoritmos que tuvieron el mismo desempeño. Sin embargo la proporción de penalización en FB con retroalimentación resultó ser la menor con un valor promedio de 2.15s.