Sistemas Operativos

Práctica 2: *Scheduling*

Notas preliminares

- Los ejercicios marcados con el símbolo ★ constituyen un subconjunto mínimo de ejercitación. Sin embargo, aconsejamos fuertemente hacer todos los ejercicios.

Ejercicio 1

La siguiente secuencia describe la forma en la que un proceso utiliza el procesador.

Tiempo	Evento
0	*load store*
1	*add store*
2	*read* de archivo
3	espera E/S
..	..
10	espera E/S
11	*store increment*
12	inc
13	*write* en archivo
14	espera E/S
..	...
20	espera E/S
21	*load store*
22	*add store*

a) Identificar las ráfagas de CPU y las ráfagas de E/S.
b) ¿Qué duración tiene cada ráfaga?

Ejercicio 2 ★

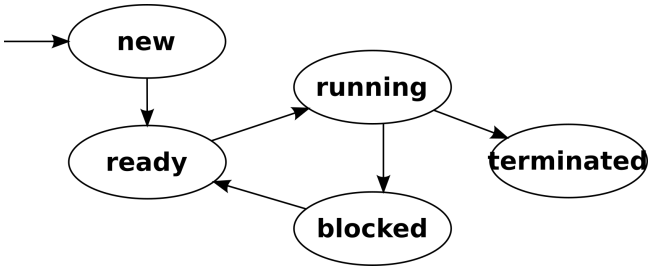
Sean P_0, P_1 y P_2 tales que

- P_0 tiene ráfagas cortas de E/S a ciertos dispositivos.
- P_1 frecuentemente se bloquea leyendo de la red.
- P_2 tiene ráfagas prolongadas de alto consumo de CPU y luego de escritura a disco.

a) Para planificar estos procesos, ¿convendría usar un algoritmo de Round Robin? ¿convendría usar uno de prioridades? Justifique su respuesta.

Ejercicio 3

¿A qué tipo de *scheduler* corresponde el siguiente diagrama de transición de estados de un proceso?



Ejercicio 4 ★

¿Cuáles de los siguientes algoritmos de *scheduling* pueden resultar en *starvation* (inanición) y en qué condiciones?

a) *Round-robin*.
b) Por prioridad.
c) SJF.
d) SRTF.
e) FIFO.
f) Colas de multinivel.
g) Colas de multinivel con feedback (aging).

Ejercicio 5

Considere una modificación a *round-robin* en la que un mismo proceso puede estar encolado varias veces en la lista de procesos *ready*. Por ejemplo, en un RR normal se tendrían en la cola ready a P1,P2,P3,P4, con esta modificación se podría tener P1,P1,P2,P1,P3,P1,P4.

a) ¿Qué impacto tendría esta modificación?
b) Dar ventajas y desventajas de este esquema. Piense en el efecto logrado, no en la forma de implementarlo.
c) ¿Se le ocurre alguna otra modificación para mantener las ventajas sin tener que duplicar las entradas en la lista de procesos *ready*?

Ejercicio 6

Considerar el siguiente conjunto de procesos:

Proceso	Ráfaga de CPU	Prioridad
P_1	10	3
P_2	1	1
P_3	2	3
P_4	1	4
P_5	5	2

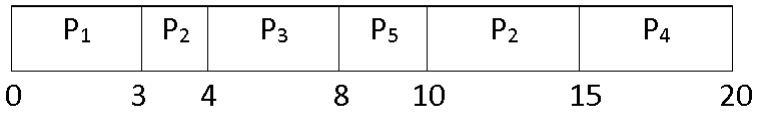
Se supone que los procesos llegan en el orden P_1, P_2, P_3, P_4, P_5 en el instante 0.

a) Dibujar los diagramas de Gantt para ilustrar la ejecución de estos procesos usando los algoritmos de scheduling FCFS, SJF, con prioridades sin desalojo (a menor el número, mayor la prioridad), round-robin (*quantum* de 1 unidad de tiempo, ordenados por el número de proceso).
b) ¿Cuál es el waiting time promedio y de turnaround promedio para cada algoritmo?
c) ¿Cuál de los algoritmos obtiene el menor waiting time promedio, y el menor turnaround?

Ejercicio 7

El siguiente diagrama de Gantt corresponde a la ejecución tres procesos en un sistema monoprocesador.

Proceso	Ráfaga de CPU	Instante de llegada (ms)
P_1	3	0
P_2	6	2
P_3	4	4
P_4	5	6
P_5	2	8



a) Calcular el *waiting time* y el *turnaround* promedios.
b) Indicar de qué tipo de *scheduler* se trata, justificando claramente esa conclusión.

Ejercicio 8

Suponiendo que los siguientes procesos llegan en los tiempos indicados.

Proceso	Ráfaga de CPU	Instante de llegada
P_1	8	0.0
P_2	4	0.4
P_3	1	1.0

a) ¿Cuál es el tiempo de *turnaround* promedio para estos procesos usando FCFS?
b) ¿Cuál es usando SJF?
c) SJF se supone que mejora la perfomance, pero al elegir ejecutar P_1 inicialmente no había forma de saber que iban a llegar dos cortos luego. Volver a calcular el tiempo de *turnaround* promedio pero dejando el procesador *idle* por una unidad de tiempo y luego usar SJF.

Ejercicio 9 ★

Para los procesos presentados en la siguiente tabla, realizar un gráfico de Gantt para cada uno de los algoritmos de scheduling indicados:

- FCFS.
- RR (*quantum*=10).
- SJF.

Proceso	Ráfaga de CPU	Instante de llegada
P_1	1	5
P_2	10	6
P_3	1	7
P_4	10	8

Calcular el *waiting time* y el *turnaround* promedios para cada una de los algoritmos.

Ejercicio 10

Considere los siguientes procesos:

Proceso	Ráfaga de CPU	Instante de llegada
P_1	8	0
P_2	8	5
P_3	6	14
P_4	5	15

a) Realizar un diagrama de Gantt para un algoritmo de scheduling *round-robin* con un *quantum* de 5 unidades de tiempo.
b) Realizar un diagrama de Gantt para un algoritmo tipo *shortest remaining time first*.
c) Calcular el tiempo de *turnaround* promedio en ambos casos.
d) A pesar de que uno de los dos casos tiene un tiempo de *turnaround* promedio mucho menor, explicar por qué en algunos contextos podría tener sentido utilizar la otra política. Para esto considere distintos tipos de procesos: real time, interactivos, batch, etc.

Ejercicio 11

Considere los siguientes procesos:

Proceso	Ráfaga de CPU	Instante de llegada	Cola asignada
P_1	4	0	1
P_2	3	0	1
P_3	8	0	2
P_4	5	10	2

a) Realizar un diagrama de Gantt para un algoritmo de scheduling *Multilevel feedback queue* con dos colas: una cola 1 con *quantum* de 1 unidad de tiempo, y una cola 2 con *FCFS*. La cola 1 tiene más prioridad que la 2. Usa política con desalojo. Para cada proceso se indica qué cola se le asigna en el momento de su llegada. La politica de *feedback* utilizada es que cuando un proceso termina su quantum, se reduce su prioridad.
b) Calcular el tiempo de *turnaround* promedio y el *waiting time* promedio.

Ejercicio 12

Considere un algoritmo de scheduling que favorece a aquellos procesos que han usado la menor cantidad de tiempo de procesador en el pasado reciente. ¿Explique por qué favorecería a los procesos que realizan muchas E/S, pero a la vez no dejaría a los intensivos en CPU en *starvation*?

Ejercicio 13

Explicar cómo los siguientes algoritmos favorecen (o desfavorecen) a los trabajos más cortos:

a) FIFO.
b) *Round-robin*.
c) *Multilevel feedback queue*.

Ejercicio 14 ★

Un sistema que atiende tareas *interactivas* de varias sucursales bancarias está conectado en forma directa a la central policial y, frente a un caso de robo, genera un proceso que activa una alarma en la central.

a) Diseñar un algoritmo que permita que, una vez generado ese proceso de alarma, tenga prioridad sobre el resto de las tareas (recordar que pueden generarse distintas alarmas desde distintas sucursales).

Nota: Especificar claramente la forma de administración de las colas.

Ejercicio 15 ★

Se tiene un sistema donde hay trabajos interactivos y de procesamiento de datos. Los de procesamiento de datos leen archivos inmensos, hacen pequeñas cuentas y los vuelven a grabar.

Se desea que los usuarios interactivos tengan la sensación de buen tiempo de respuesta, pero sin perjudicar excesivamente el *throughput* del sistema.

El *scheduler* puede funcionar con *round-robin* o con FCFS. ¿Qué política utilizaría y por qué? Justificar especialmente por qué la política elegida permite cumplir con ambos objetivos del sistema.

Ejercicio 16 ★

Una seriografía es una técnica para el estudio de los órganos en movimiento. Se realiza utilizando un aparato llamado seriógrafo, que ejecuta varias radiografías por segundo y muestra en una pantalla una serialización digital de estas imágenes, dando como resultado una especie de video.

Existen seriógrafos que permiten editar algunas características de las imágenes a medida que se van generando, mientras se está llevando a cabo el estudio médico. Entre otras cosas, permiten ajustar el brillo y el contraste de las imágenes, y hacer zoom-in y zoom-out. Así, se permite una edición "en vivo" del video.

Se tienen entonces los siguientes procesos:

- uno que genera las imágenes digitales a partir de los valores resultantes al irradiar al paciente
- uno que responde a los botones de ajuste de brillo y contraste
- uno que responde a los botones de ajuste de zoom

¿Qué política de scheduling permite esta toma y edición de imágenes "en vivo" de manera eficiente? Justificar.

Ejercicio 17 ★

Se tiene un sistema de vigilancia que utiliza cámaras y alarmas. Estos dispositivos se comunican con un servidor que atiende distintos tipos de procesos con una política de scheduling que deberá desarrollarse para este escenario particular.

Los módulos de procesamiento de video son procesos que leen el stream de video que llega desde una cámara y luego corren un algoritmo de detección de objetos. A su vez, estos procesos persisten las secuencias de video en discos del servidor. Para este tipo de procesos se quiere evitar situaciones de scheduling poco "justas".

En caso de detectar patrones considerados riesgosos, el sistema debe alertar a los operadores para que actúen de inmediato. Para este fin, se cuenta con un módulo de alarma que se lanza en un proceso y que gestiona su activación. Es crítico poder garantizar que cualquier alarma se active dentro de un deadline estricto que no puede perderse.

Por otro lado, el servidor cuenta con suficiente espacio para almacenar temporalmente muchas horas de frames en un formato "crudo" de video de todas las cámaras. Sin embargo, periódicamente se lanzan procesos que levantan grandes volúmenes de video grabados durante el día y le aplican un algoritmo de compresión que permite luego reemplazar las secuencias originales por otras de mucho menor tamaño. Estos procesos son lanzados durante la noche, cuando las áreas se encuentran mucho menos transitadas, por lo que las cámaras se configuran para trasmitir sólo en caso de detección de movimiento, así que la carga de procesos de procesamiento activos de video es muy baja y en forma de ráfagas de corta duracion.