

Trabajo Práctico 1: Scheduling

Nombre: Maria Valentina Sancho.

LU: 1214/23.

1) Se generaron dos problemas con datos aleatorios deshabilitando E/S y limitando el tiempo máximo de los trabajos.

Para el primer problema, la duración máxima de cada trabajo fue configurada en 22 ms y se generaron 3 trabajos. Todos llegan en el instante 0.

Nombraré como P0 al trabajo que fue colocado primero a la cola de mayor prioridad, P1 al segundo y P2 al último.

El P0 tiene una duración de 18 ms, el P1 de 9 ms y P2 de 11 ms.

Todos los trabajos comienzan en la cola de máxima prioridad, y al ser varios en la misma cola, son elegidos por el criterio Round Robin. Es decir, se va a seguir el orden en el que entraron.

El quantum es de 10 ms, y el allotment (tiempo que pasa un proceso en una cola antes de pasar a otra de menor prioridad) es de un quantum.

Diagrama de Gantt:



El color azul simboliza que el proceso está ejecutándose en la cola de prioridad 2 y el verde que se ejecuta en la cola de prioridad 1.

P0 es ejecutado primero durante un quantum, y como al finalizar ese quantum sigue necesitando CPU y ya alcanzó su time allotment, es movido a la cola de prioridad 1 (baja a una cola de menor prioridad).

El trabajo P1 finaliza con 1 ms restante de su quantum, y sale de la cola. No será elegido nuevamente por el scheduler.

El trabajo P2 es ejecutado por un quantum, y por lo mismo que P0, es movido a la cola de prioridad 1.

Como P0 llegó antes que P2 a la cola de prioridad 1, es elegido antes por el scheduler, y finaliza en su segundo quantum.

Luego, el único proceso no finalizado en el scheduler es P2. Es elegido para ejecutar y finaliza en su segundo quantum, en la cola de prioridad 1.

Estadísticas finales:

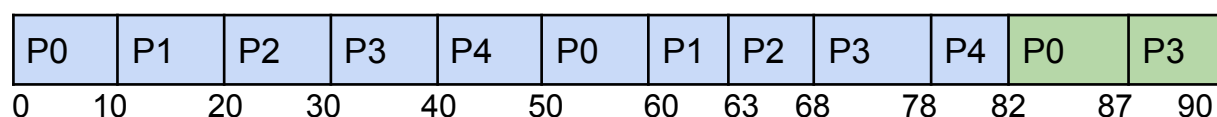
En el siguiente cuadro se puede observar el tiempo que cada proceso pasó en cada estado del scheduler: Running, Ready y Waiting.

Procesos	Running	Ready	Waiting	Turnaround
P0	18 ms	19 ms	0 ms	37 ms
P1	9 ms	10 ms	0 ms	19 ms
P2	11 ms	27 ms	0 ms	38 ms

Para el segundo problema, decidí configurar que la duración máxima del trabajo sea de 30 ms, que se generen 5 trabajos y que el allotment por cola sea de 2 quantums.

P0 tiene una duración de 25 ms, P1 de 13 ms, P2 de 15 ms, el P3 de 23ms y P4 de 14ms.

Diagrama de Gantt:



A diferencia del problema anterior, los procesos tienen 2 quantums en cada cola de cierta prioridad antes de ser cambiados a otra.

Al finalizar el segundo quantum de P0 (en $t = 60$ ms), es movido a la cola de prioridad 1.

Como sigue habiendo procesos en la cola de mayor prioridad (P1, P2, P3 y P4), se continúa haciendo Round Robin sobre ellos.

P1 finaliza en su segundo quantum, restando 7 ms para que termine, en $t = 63$ ms.

P2 también finaliza en su segundo quantum, en $t = 68$ ms.

P3 obtiene un segundo quantum y pasa a la cola de prioridad 0 junto a P0.

P4 termina en su segundo quantum tras 4 ms de ejecución.

Tras terminar P4, ya no hay más procesos de mayor prioridad o más prioritarios que los que están en la cola de prioridad 1. Luego, se pasa a ser Round Robin sobre P0 y P3 en la cola de prioridad 1. Como P0 es el primero que llegó a esa cola, es el primero en ser elegido por el scheduler.

P0 finaliza después de tener CPU durante 5 ms.

P3 es el único proceso no finalizado a esta altura. Es elegido por el scheduler y finaliza tras 3 ms corriendo.

Estadísticas finales

Proceso	Running	Ready	Waiting	Turnaround
P0	25 ms	62 ms	0 ms	87 ms
P1	13 ms	50 ms	0 ms	63 ms
P2	15 ms	53 ms	0 ms	68 ms
P3	23 ms	67 ms	0 ms	90 ms
P4	14 ms	68 ms	0 ms	82 ms

El turnaround promedio es de 78 ms.

2)

El Throughput es el criterio que mide la cantidad de trabajos terminados por unidad de tiempo. Lo que hice fue, al incrementar el tiempo actual, chequear si ese número es par (porque queremos medir el throughput cada dos unidades de tiempo) y si lo es, imprimir el número de trabajos finalizados hasta el momento sobre el tiempo actual.

El Waiting Time es el tiempo que un proceso está en el scheduler en estado Ready, es decir, el tiempo que que pasa siendo elegible por el scheduler pero no está siendo ejecutado.

Para esta implementación, agregué dos claves al diccionario asociado a cada trabajo en el diccionario job: waitingTime y finished, que se inicializan con los valores 0 y False respectivamente.

Agregué, luego de incrementar el tiempo actual, un ciclo que recorre todos los trabajos del diccionario job, y se fija si están en estado ready, es decir, si:

- El trabajo no está ejecutándose actualmente (en ese caso, estaría en estado RUNNING)
- El trabajo no está haciendo IO (en ese caso, estaría en estado WAITING)
- El trabajo todavía no finalizó

Si está en ese estado, se incrementa el valor de waitingTime.

Una vez que finaliza el trabajo, se cambia el valor a True en la clave "finished" de su diccionario.

Una vez que todos los trabajos terminan y se realiza el cálculo del turnaround y responseTime, se imprime el waiting time individual de cada proceso (simplemente se accede al valor con la clave "waitingTime" en su diccionario) y se va sumando cada uno. Al finalizar, se imprime la suma de waiting time de todos los trabajos sobre la cantidad de trabajos, es decir, el waiting time promedio.

3) Para acaparar todos los recursos del CPU con un trabajo que tenga E/S, pensé que el caso ideal para esto era un trabajo que dure poco tiempo, y su frecuencia E/S sea alta.

Así, como el proceso interactivo será el prioritario, el resto de trabajos solo podrán ejecutar cuando ese proceso pase a estado waiting, que será por muy poco tiempo.

Decidí también configurar que el tiempo E/S máximo sea de 2ms.

Esta es la configuración que utilicé: --jlist 0,30,5:0,30,0 -S -I -i 2 -c

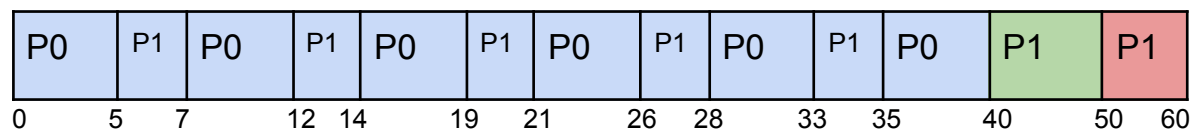
Lista dos trabajos que ingresan a la cola de mayor prioridad en el instante 0 y tienen una duración de 30 ms. El primer trabajo es interactivo y tiene una frecuencia de E/S de 5 ms. Configuré que los trabajos con E/S no bajen de prioridad, que los trabajos que finalicen E/S pasen al frente de la cola y que la duración fija de una E/S sea de 2 ms.

Cada vez que el P0 hace E/S, el P1 se ejecuta 2ms. Al final, la mayor parte de su ejecución ocurre cuando el P0 finaliza, porque al este tener prioridad

fija y cada vez que hace entrada y salida se agrega al principio de la cola, no es elegido por el scheduler a menos que el P0 esté haciendo E/S (en estado waiting).

Después de que finaliza P0, P1 es ejecutado por un quantum en la cola de prioridad 1 y como tiene tiempo de ejecución restante, baja a la cola de menor prioridad porque superó su allotment. Luego de un quantum, finaliza.

Diagrama de Gantt:



El siguiente cuadro detalla los tiempos que cada proceso pasa por cada estado durante la ejecución:

Trabajos	Turnaround	Running	Waiting	Ready
P0	40 ms	30 ms	$5 * 2\text{ms} = 10\text{ ms}$	0 ms
P1	60 ms	30 ms	0 ms	30 ms

P0 (el proceso que hace E/S) está en estado waiting durante 10 ms porque como la frecuencia de E/S es de 5 ms, se realizan 5 E/S mientras se ejecuta el problema y la duración de cada E/S es de 2 ms. Durante ese tiempo, P0 no es elegible por el scheduler para otorgarle tiempo de CPU, pero una vez que termina la E/S, este siempre es elegido primero. Así, el proceso fluctúa entre dos estados: waiting o running. En ningún momento es elegible por el scheduler y no es elegido.

P1 pasa 0 ms en estado waiting porque no es un proceso interactivo. Pasa 30 ms en estado ready porque al ser P0 ejecutado continuamente, este proceso solo es elegible por el scheduler cuando P0 pasa a estar en estado waiting.

La mayor parte de la ejecución del proceso P1 ocurre cuando P0 termina.

4) Para beneficiar el trabajo interactivo, configuré que los trabajos con E/S no cambian de prioridad, que luego de finalizar un E/S se colocan al principio de la cola y que el quantum sea de 5 ms.

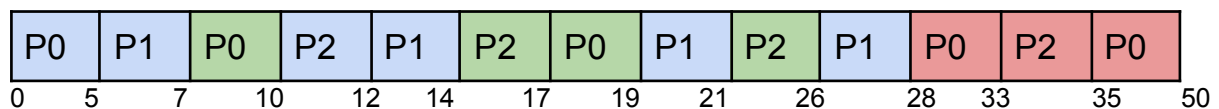
Decidí no modificar la duración máxima del IO.

Durante la ejecución, el trabajo interactivo permanece todo el tiempo en la cola de mayor prioridad. Luego de un quantum de 5 ms, P0 y el P2 pasan a una cola de menor prioridad.

El P0 y el P2 son elegidos por el scheduler solo cuando el P1 pasa a estado waiting, es decir, a hacer E/S.

El P1 es el primero en finalizar. Además, siempre que no está haciendo E/S es elegido por el scheduler.

Diagrama de Gantt:



Cuadro con detalle del tiempo que pasa cada proceso en cada estado:

Aclaración: El waitingTime, como aclaré en el anterior punto, es la suma de tiempos que pasa un proceso en estado ready sin ejecutarse. No tiene nada que ver con el estado waiting de un proceso. Un proceso está en estado waiting cuando espera por E/S, y no es elegible por el scheduler hasta que se complete esa operación.

Procesos	TurnAround	WaitingTime	Running	Waiting
P0	50ms	20ms	30ms	0ms
P1	28ms	5ms	8ms	20 ms
P2	25ms	23ms	12ms	0ms

Los procesos no interactivos son desfavorecidos porque fueron los que terminaron últimos, su turnAround es el mayor (también considerando que P2 entró 10 ms después que P0 y P1) y son los que pasan más tiempo en estado ready.

En particular, notar que el proceso P2 pasa más tiempo esperando por CPU que ejecutándose.

Para el caso en el que se benefician los trabajos no interactivos sobre los interactivos, se me ocurrió que, seteando que la duración máxima de E/S sea de un quantum, sería beneficioso para los otros trabajos, aparte de permitir

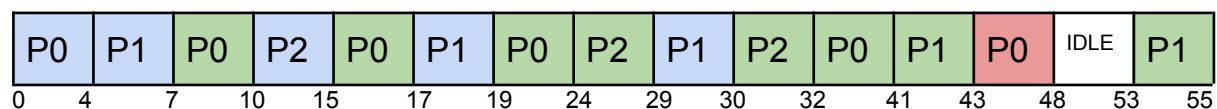
que los procesos que hacen E/S bajen de prioridad y de no configurar que pasen al inicio de la cola cuando finalizan E/S.

Configuré que el quantum en la cola 1 sea de 5 y en el resto de 10, y que el time Allotment de la cola 1 sea de 1 y en el resto de 2.

Esta fue la configuración utilizada --jlist 0,30,0:0,8,2:10,12,0 -i 10 -Q 5,10,10 -A 1,2,2 -c

Tras 5 ms de ejecución, todos los procesos bajan a la cola de prioridad 1.

Diagrama de Gantt:



Entran los tres procesos P0, P1 y P2 a la cola de máxima prioridad, y son elegidos con el criterio Round Robin.

P0 es el primero en ser elegido, se ejecuta durante un quantum de 5 ms y baja a la cola de menor prioridad.

Le sigue el trabajo interactivo, que ejecuta durante 2 ms y empieza su E/S. Pasa a estado waiting y es desalojado. P2 pasa a estado ready y es colocado en la cola de máxima prioridad, comienza a ejecutar y finaliza su quantum. Pasa a la cola de prioridad 1.

La cola de prioridad 1 tiene a P0 y P2.

El P1 sigue en la cola de máxima prioridad, finaliza E/ S y se ejecuta durante dos segundos antes de pasar nuevamente a estado waiting.

Le resta 1 ms de quantum luego de terminar la E/S.

Mientras tanto, se ejecutan en Round Robin P0 y P2, en la cola de prioridad 1.

Finaliza la E/S de P1 y se ejecuta durante 1 ms en la cola de máxima prioridad.

Pasa a la cola de menor prioridad, pero como P0 y P2 llegaron antes, por como funciona round robin, debe esperar dos quantums de 10 ms para ejecutar nuevamente.

El trabajo P2 finaliza.

P0 es elegido, y luego de un quantum, pasa a una cola de menor prioridad.

Como P1 a esa altura, sigue en la cola de prioridad 1, es prioritario. Tras 2 ms de ejecución es desalojado para realizar E/S.

Mientras P1 está en estado waiting, el único proceso en estado ready es P0, que se ejecuta hasta finalizar.

Mientras que la E/S del P1 se realiza, no hay otras tareas para ejecutar, entonces se ejecuta la IDLE.

Termina la E/S , y finalmente, P1 termina.

El beneficio se ve principalmente en el turnaround: el P1 pasa 55 ms desde que entra en el scheduler hasta que finaliza, mientras que los procesos no interactivos aprovechan sus quantums, y siempre están en estado ready cuando P1 pasa a estado waiting.

Aparte, como ejecutan más tiempo, bajan antes de prioridad, pero eso en este caso termina siendo un beneficio porque están primeros en la cola de prioridad 1, y para cuando P1 baja a esa cola de menor prioridad, tiene que esperar a que P0 y P2 sean elegidos.

Procesos	TurnAround	WaitingTime	Running	Waiting
P0	48 ms	18 ms	30 ms	18 ms
P1	55 ms	17 ms	8 ms	17 ms
P2	22 ms	20 ms	12 ms	20 ms

Lo que evidencia el desfavorecimiento de P1 sobre los otros procesos es que mientras su duración es tan solo de 8 ms, pasa 55 ms desde que entra hasta que finaliza. En particular, pasa más tiempo en estado waiting y ready que ejecutándose.

En conclusión, al no cambiar las prioridades de los procesos interactivos (no hacer que no puedan bajar de prioridad, o que se pongan al principio de la cola al finalizar una operación E/S), las constantes interrupciones durante su ejecución por cada operación de E/S, sumado a el tiempo que tardan esas operaciones, hacen que los trabajos no interactivos cuya ejecución solo es interrumpida para ejecutar otro proceso elegido por el scheduler, terminen en menor tiempo.

