UNIVERSIDAD DE CÓRDOBA

ESCUELA POLITÉCNICA SUPERIOR

DEPARTAMENTO DE INFORMÁTICA Y ANÁLISIS NUMÉRICO

ASIGNATURA ***SISTEMAS OPERATIVOS***

2º DE GRADO EN INGENIERÍA INFORMÁTICA

PRÁCTICA 4

Algoritmos de planificación de procesos

Profesores:

Enrique García Salcines

Javier Pérez Rodríguez

Indice de la práctica

1 Objetivo de la práctica 3
2 Recomendaciones 3
3 Conceptos teóricos 3
3.1 Algoritmos de planificación 5
3.1.1 FCFS (First Come-First Served) 6
3.1.2 SJF (Shortest Job First) 8
3.1.3 Round Robin (RR) 9
4 Ejercicios Prácticos 10

1 Objetivo de la práctica

La presente práctica persigue familiarizar al alumnado con el diseño e implementación de algoritmos de planificación de procesos en sistemas multitareas. En concreto estudiaremos cómo el planificador del sistemas operativo toma la decisión de ejecutar en un momento determinado un proceso u otro.

En una primera parte se dará una introducción teórica sobre Algoritmos de Planificación, siendo en la segunda parte de la misma cuando se practicarán los conceptos aprendidos mediante programación en C, utilizando las herramientas que proporciona el estándar.

2 Recomendaciones

El lector debe completar las nociones dadas en las siguientes secciones con consultas bibliográficas, tanto en la Web como en la biblioteca de la Universidad, ya que unos de los objetivos de las prácticas es potenciar su capacidad autodidacta y su capacidad de análisis de un problema. Es recomendable que, aparte de los ejercicios prácticos que se proponen, pruebe y modifique otros que se encuentren en la Web (se dispone de una gran cantidad de problemas resueltos en C sobre esta temática), ya que al final de curso deberá acometer un examen práctico en ordenador como parte de la evaluación de la asignatura.

Al igual que se le instruyó en las asignaturas de Metodología de la Programación, es recomendable que siga unas normas y estilo de programación claro y consistente. No olvide tampoco comentar los aspectos más importantes de sus programas, así como añadir información de cabecera a sus funciones (nombre, parámetros de entrada, parámetros de salida, objetivo, etc). Estos son algunos de los aspectos que se también se valorarán y se tendrán en cuenta en el examen práctico de la asignatura.

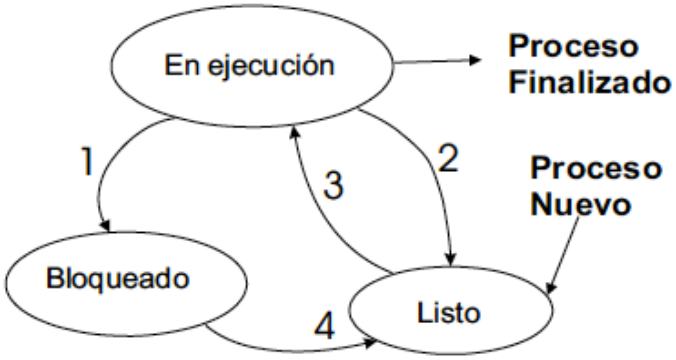
3 Conceptos teóricos

La Planificación de procesos en Sistemas Operativos es el conjunto de políticas y mecanismos incorporados al sistema operativo, a través de un módulo denominado planificador (scheduler), que debe decidir cuál de los procesos en condiciones de ser ejecutado conviene ser despachado primero y qué orden de ejecución debe seguirse. Esto debe realizarse sin perder de vista su principal objetivo que consiste en el máximo aprovechamiento del sistema, lo que implica proveer un buen servicio a los procesos existentes en un momento dado.

Como vimos en la primera práctica, un proceso es un programa que está en ejecución. Este proceso puede estar en 3 estados distintos “Listo” “Bloqueado” y “En Ejecución”.

- **En ejecución:** Está usando el procesador.
- **Bloqueado:** No puede hacer nada porque está espera un evento externo (esperando la conclusión de E/S).
- **Listo:** Está en memoria esperando turno para ejecutarse en la CPU (espera asignación del procesador).

- Diagrama de transición de estados:
  - 1.- Pasa a esperar un suceso (E/S) y se bloquea.
  - 2.- Expulsión de proceso de la CPU
  - 3.- El planificador elige otro proceso.
  - 4.- El suceso (E/S) que esperaba el proceso acaba.



- En un instante: sólo un proceso en ejecución, los demás estarán listos o en espera.

En la ejecución de un proceso se alternan la ejecución en CPU y la espera de E/S (entrada/salida):

- Ráfaga CPU: Tiempop de ejecución en CPU entre dos E/S.
- Ráfaga E/S: Tiempo entre solicitud y terminación de E/S

La Planificación de procesos tiene como principales objetivos la equidad, la eficacia, el tiempo de respuesta, el tiempo de regreso y el rendimiento:

- **Equidad:** Todos los procesos deben ser atendidos.
- **Eficacia:** El procesador debe estar ocupado el 100% del tiempo.
- **Tiempo de respuesta:** El tiempo empleado en dar respuesta a las solicitudes del usuario debe ser el menor posible.
- **Tiempo de retorno:** Reducir al mínimo el tiempo de espera de los resultados esperados por los usuarios por lotes.
- **Rendimiento:** Maximizar el número de tareas que se procesan por unidad de tiempo.

Para escoger qué proceso debe ejecutarse en cada momento se utilizan los algoritmos de planificación. Una característica de los algoritmos de planificación es la **expropiación**. Se puede decir que un algoritmo como expropiativo si podemos retirar un proceso que se está ejecutando para introducir uno nuevo.

Para estudiar la bondad de un algoritmo de planificación se suelen estudiar algunos parámetros que a continuación se detallan:

- **Tiempo de ejecución:** Tiempo que está ejecutándose el proceso en la CPU.
- **Tiempo de llegada:** Instante de tiempo en el que se ejecuta un proceso.
- **Tiempo de comienzo:** Instante de tiempo en el que el proceso entra en la CPU. Normalmente es igual al tiempo de finalización del proceso anterior.
- **Tiempo de finalización:** es el tiempo de comienzo más el tiempo de ejecución del proceso.
- **Tiempo de retorno:** Tiempo que transcurre desde que el proceso se lanza hasta que finaliza su ejecución. Se puede calcular como la suma del tiempo de finalización menos el tiempo de llegada del proceso.
- **Tiempo de espera:** Tiempo que el proceso está listo en espera de que pueda ejecutarse. Puede calcularse como el tiempo de retorno menos el tiempo de ejecución.
- **Productividad:** Número de trabajos realizados por unidad de tiempo.
- **Uso de la CPU:** Porcentaje de tiempo que el procesador pasa ejecutando procesos.

3.1 Algoritmos de planificación

A continuación se detallan los principales algoritmos de Planificación.

1) Primero en llegar primero en ser servido

Conocido como **FCFS** (First Come, First Served). Este algoritmo emplea una cola de procesos, asignando un lugar a cada proceso por el orden de llegada. Cuando el proceso llega es puesto en su lugar en la cola después del que llegó antes que él y se pone en estado de listo. Cuando un proceso comienza a ejecutarse no se interrumpe su ejecución hasta que termina de hacerlo.

2) Prioridad al más corto

Su nombre es **SJF** (Shortest Job First). El proceso que se encuentra en ejecución cambiará de estado voluntariamente, o sea, no tendrá un tiempo de ejecución determinado para el proceso. A cada proceso se le asigna el tiempo que usará cuando vuelva a estar en ejecución, y se irá ejecutando el que tenga un menor tiempo asignado. Si se da el caso de que dos procesos

tengan igual valor en ese aspecto emplea el algoritmo FCFS.

3) Round Robin (RR)

A cada proceso se le asigna un tiempo determinado para su ejecución, el mismo tiempo para todos. En caso de que un proceso no pueda ser ejecutado completamente en ese tiempo se continuará su ejecución después de que todos los procesos restantes sean ejecutados durante el tiempo establecido. Este es un algoritmo basado en FCFS que trata la cola de procesos que se encuentran en estado de listos como una cola circular.

4) Planificación por prioridad

En este tipo de planificación a cada proceso se le asigna una prioridad siguiendo un criterio determinado, y de acuerdo con esa prioridad será el orden en que se atienda cada proceso.

5) Planificación garantizada

Para realizar esta planificación el sistema tiene en cuenta el número de usuarios que deben ser atendidos. Para un número "n" de usuarios se asignará a cada uno un tiempo de ejecución igual a 1/n.

6) Planificación de Colas Múltiples

El nombre se deriva de MQS (Multilevel Queue Schedulling). En este algoritmo la cola de procesos que se encuentran en estado de listos es dividida en un número determinado de colas más pequeñas. Los procesos son clasificados mediante un criterio para determinar en qué cola será colocado cada uno cuando quede en estado de listo. Cada cola puede manejar un algoritmo de planificación diferente a las demás.

Debido al nivel de complejidad que podrían tener algunos de estos algoritmos, en la presente práctica vamos a analizar solamente los tres primeros.

3.1.1 FCFS (First Come-First Served)

En esta política de planificación, el procesador ejecuta cada proceso hasta que termina, por tanto, los procesos que en cola de procesos preparados permanecerán encolados en el orden en que lleguen hasta que les toque su ejecución. Este método se conoce también como FIFO (fist input, first output,Primero en llegar primero en salir).

Se trata de una política muy simple y sencilla de llevar a la práctica, pero muy pobre en cuanto a su comportamiento.

La cantidad de tiempo de espera de cada proceso depende del número de procesos que se encuentren en la cola en el momento de su petición de ejecución y del tiempo que cada uno de ellos tenga en uso al procesador, y es independiente de las necesidades del propio proceso.

Sus características son:

- No apropiativa.
- Es justa, aunque los procesos largos hacen esperar mucho a los cortos.
- Predecible.
- El tiempo medio de servicio es muy variable en función del número de procesos y su duración.

Ejemplo :

Proceso A → Tiempo ejecución → Tiempo llegada → Tiempo finaliza → Tiempo retorno → Tiempo espera .

Proceso	Tiempo de ejecución	Tiempo de llegada	Tiempo de comienzo	Tiempo de finalización	Tiempo de retorno	Tiempo de espera
A	8	0	0	8	8	0
B	4	1	8	12	12-1 = 11	11 – 4 = 7
C	9	2	12	21	21-2 = 19	19 - 9 =10
D	5	3	21	26	26-3 = 23	23 – 5 = 18
E	2	4	26	28	28-4 = 24	24 -2 = 22

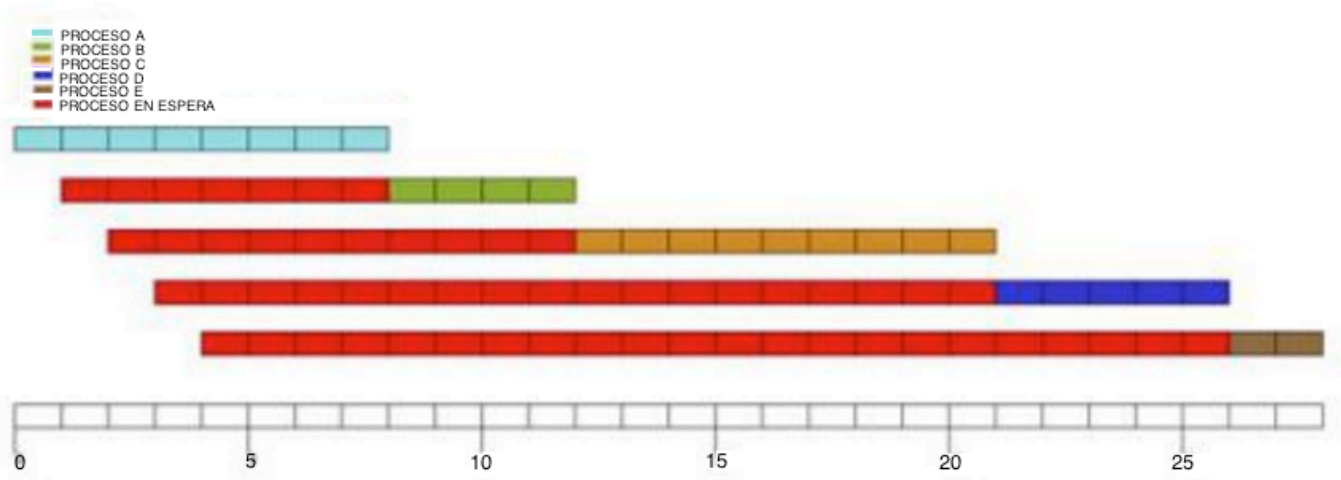


Tabla 1. Tiempos de simulación del algoritmo FCFS

En el caso de que los procesos de mayor tiempo de duración llegasen los primeros, el tiempo medio de espera sería mucho mayor. Podemos llegar a la conclusión de que este no es un algoritmo eficiente.

Conclusión

Este algoritmo está bien lo único que que los procesos largos hacen esperar mucho a los cortos, por tanto el tiempo de espara puede ser bastante largo.
El tiempo medio de servicio es muy variable en función del número de procesos y su duración.

3.1.2 SJF (Shortest Job First)

En este algoritmo da bastante prioridad a los procesos más cortos a la hora de ejecución y los coloca en la cola.

Ejemplo:
Una cola de personas en Mercadona delante de la caja, la persona que menos compra lleva esa pasa primero.

Proceso	Tiempo de ejecución	Tiempo de llegada	Tiempo de comienzo	Tiempo de finalización	Tiempo de retorno	Tiempo de espera
A	8	0	0	8	8	0
B	4	1	10	14	14 - 1= 13	13 – 4 = 9
C	9	2	19	28	28 -2 = 26	26 – 9 = 17
D	5	3	14	19	19 – 3 = 16	16 – 5 = 11
E	2	4	8	10	10 – 4= 6	6 – 2 = 4

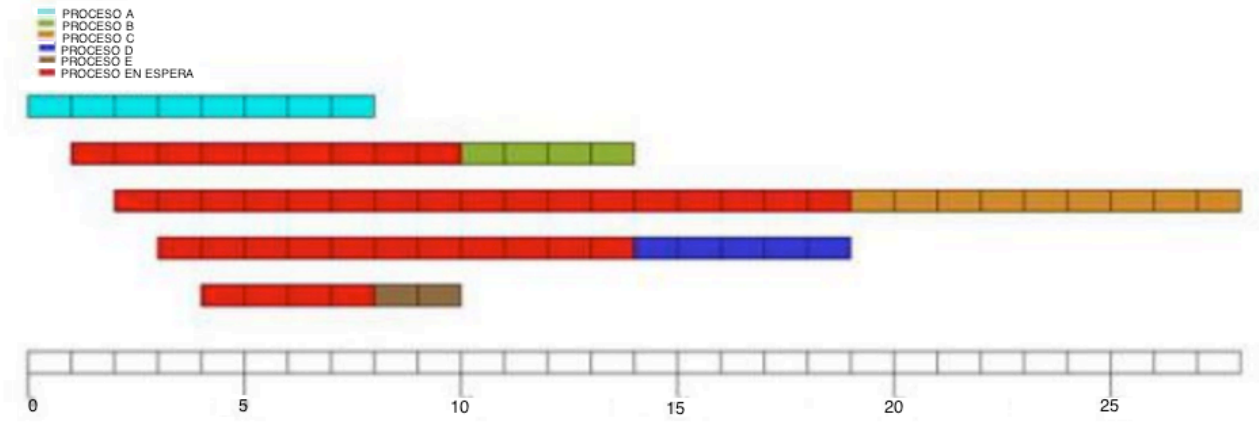


Tabla 2. Tiempos de simulación del algoritmo SJF

Conclusiones

En resumen, este algoritmo selecciona al proceso con el próximo tiempo ejecución más corto. El proceso corto saltará a la cabeza de la cola. Ejecución de un proceso consiste en ciclos de ejecución de CPU y ciclos de espera por E/S. El algoritmo selecciona aquel proceso cuyo próximo ciclo de ejecución de CPU sea menor. El problema está en conocer dichos valores, pero podemos predecirlos usando la información de los ciclos anteriores ejecutados.

- Se minimiza el tiempo de espera medio
- Los procesos de larga duración sufren riesgo de inanición.

3.1.3 Round Robin (RR)

Es un método para seleccionar todos los elementos en un grupo de manera equitativa y en un orden racional, normalmente comenzando por el primer elemento de la lista hasta llegar al último y empezando de nuevo desde el primer elemento.

Round Robin es uno de los algoritmos de planificación de procesos más complejos y difíciles, dentro de un sistema operativo asigna a cada proceso una porción de tiempo equitativa y ordenada, tratando a todos los procesos con la misma prioridad.

Se define un intervalo de tiempo denominado cuanto, cuya duración varía según el sistema. La cola de procesos se estructura como una cola circular. El planificado la recorre asignando un cuanto de tiempo a cada proceso. La organización de la cola es FIFO.

Proceso	Tiempo de ejecución	Tiempo de llegada	Tiempo de comienzo	Tiempo de finalización	Tiempo de retorno	Tiempo de espera
A	8	0	0-14 -23	3 -17 -25	25	25 -8 = 17
B	4	1	3 -17	6 -18	18 - 1= 17	17 -4 = 13
C	9	2	6 – 18 -25	9 -21 -28	28 -2 = 26	26 – 9 = 17
D	5	3	9 – 21	12 -23	23 – 3 = 20	20 – 5 = 15
E	2	4	12	14	14 – 4= 10	10 – 2 = 8

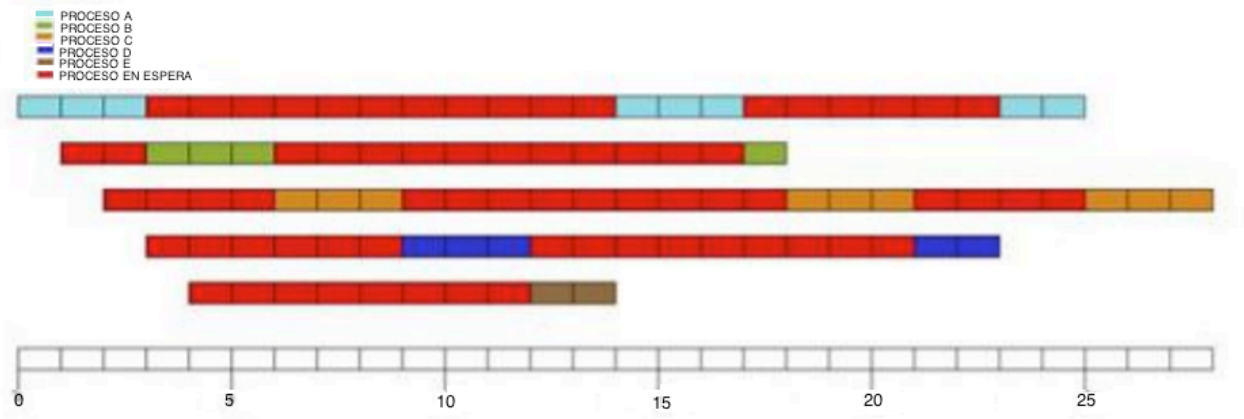


Tabla 3. Tiempos de simulación del algoritmo Round Robin

Conclusiones

- Es adecuado para implementar tiempo compartido
- Se comporta como FCFS pero cada proceso dispone de un quantum o cuanto de tiempo máximo
- Si el cuanto es muy grande (más grande que el mayor tiempo de CPU de los procesos), RR se convierte en FCFS ya que los procesos terminan sus ráfagas de CPU antes de que termine el cuanto.
- Si el cuanto es muy pequeño se provocarían constantemente cambios de contexto, disminuyendo el rendimiento.

4 Ejercicios Prácticos

1) Implementar un programa en lenguaje C, que **simule al algoritmo FCFS**. Para comprobar los resultados obtenidos introduzca los datos de procesos que aparecen reflejados en la **Tabla 1** del presente documento.

2) Implementar un programa en lenguaje C, que **simule al algoritmo SJF**. Para comprobar los resultados obtenidos introduzca los datos de procesos que aparecen reflejados en la **Tabla 2** del presente documento.

3) Implementar un programa en lenguaje C, que **simule al algoritmo Round Robin**. Para comprobar los resultados obtenidos introduzca los datos de procesos que aparecen reflejados en la **Tabla 3** del presente documento. Utilice *quantum = 3*.