UT 4: Gestión de recursos de un S.O.

6. PLANIFICACION DE CPU

Prof.: Baldomero Sánchez Pérez

6. **Tipos de planificadores**

- *Planificador a largo plazo (planificador de trabajos):*
  - Selección de trabajos a cargar en memoria principal.
  - Invocado muy infrecuentemente (segundos o minutos).
  - Puede ser (más) lento.
  - Controla el grado de multiprogramación en el sistema.
- *Planificador a medio plazo:*
  - Traslado de un proceso en memoria principal a disco (intercambio o“swapping”).
  - Posteriormente volverá a memoria principal.
  - Reduce la contienda por el uso de la CPU.
  - En ocasiones necesario ante los requisitos de memoria principal.
- *Planificador a corto plazo (planificador de la CPU):*
  - Selección del proceso listo que será ejecutado a continuación.
  - Invocado muy frecuentemente (milisegundos).sheduler
  - Debe ser rápido.
  - Son atendidos Dispatcher.

6.1. Evaluación de algoritmos de planificación

- **Parámetros de evaluación de un algoritmo de planificación:**
  - *Utilización de la CPU: Porcentaje de tiempo que el procesador está* ocupado.
  - *Productividad de la CPU: Número de trabajos por unidad de tiempo que* finalizan.
  - *Tiempo de retorno: Tiempo que tarda en ejecutarse un proceso.*
  - *Tiempo de espera: Tiempo de un proceso en lista de procesos listos.*
  - *Tiempo de respuesta: Tiempo de un proceso en dar la primera respuesta.*
- **Criterios de optimización en planificación:**
  - Maximizar utilización y productividad de la CPU.
  - Minimizar tiempo de retorno, de espera y de respuesta.

6.2. Algoritmos de planificación

- **Algoritmos de planificación monoprocesador:**
  - Planificación por “turno de llegada” (FCFS)
  - Planificación por “primero el trabajo más corto” (SJF y SRTF)
  - Planificación basada en “prioridades”
  - Planificación por “turno rotatorio” (RR)
  - Planificación basada en “colas multinivel”
    - Colas no realimentadas
    - Colas realimentadas

6.3. Algoritmo FCFS (“First-Come First-Served”)

- Los procesos pasan a CPU en orden de llegada a cola de procesos listos.
- Si el proceso en ejecución necesita E/S, se inserta al final de la cola de procesos listos al regresar a ésta.
- Algoritmo no expulsivo.
- Fácil implementación con cola FIFO.
- Poco eficiente.

6.3. Algoritmo FCFS (2)

- Ejemplo:

Proceso	*Ráfaga CPU*
P1	24 ut.
P2	3 ut.
P3	3 ut.

- Orden de llegada (en instante 0) a cola de procesos listos: P1, P2, P3.
  - Diagrama de Gant para la planificación:

  - Tiempo de espera: T
  - Tiempo medio de espera: (0+24+27)/3=17 ut.
  - *Efecto convoy: Procesos en espera debido a procesos limitados por CPU.*

6.3. Algoritmo FCFS (4)

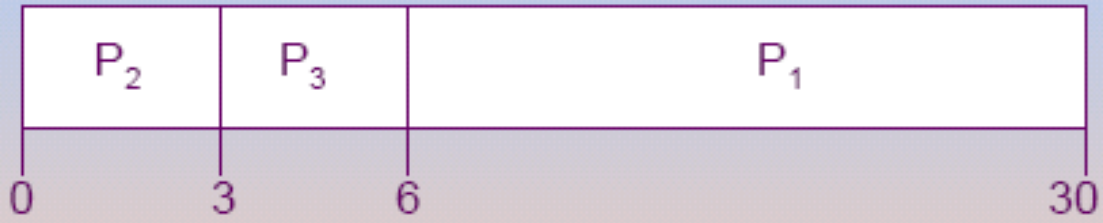
- Este algoritmo emplea los procesos en la cola de ‘listos’ Algoritmo no apropiativo.
  - Una petición no puede ser desplazada por la llegada de una petición con prioridad mas alta.
  - No hay reordenamiento de la cola de peticiones pendientes.
  - Se ignoran las relaciones posiciónales entre las peticiones pendientes.
  - Ofrece una varianza pequeña aunque perjudica a las peticiones situadas al final de la cola.

Planificación FCFS (Cont.)

Suponga que los procesos llegan en el orden;

P_2 , P_3 , P_1 .

- La gráfica de Gantt para la planificación es:



- Tiempo de espera: $P_1 = 6; P_2 = 0, P_3 = 3$
- Tiempo de espera promedio: (6 + 0 + 3)/3 = 3
- Mucho mejor que el caso previo **pero impredecible** ya que solo tiene una oportunidad en seis de ejecutar los trabajos en la secuencia mas ventajosa.
- *Existe un **efecto Convoy** cuando los procesos cortos esperan que un proceso grande suelte la CPU*
- *El algoritmo FCFS es **no-apropiativo***

6.4. Algoritmo SJF ("Shortest Job First")

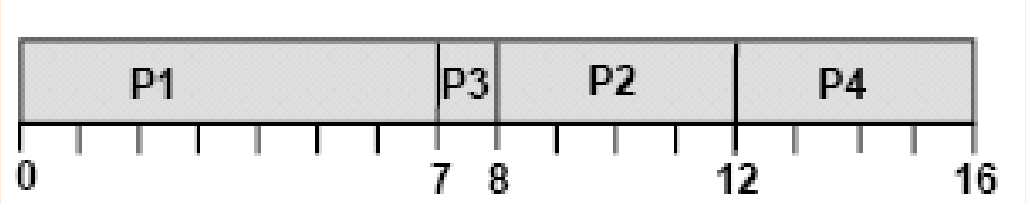
- Asociar a cada proceso el tiempo de ráfaga de CPU. (BCP)
- Seleccionar el proceso con menor ráfaga de CPU.
- En caso de empate, aplicar FCFS.
- Dos esquemas:
  - **no-preemptivo** – una vez que es atribuida la CPU a un proceso, este no puede ser preempcionado antes de terminar en la cpu.
  - **preemptivo** – cuando llega un nuevo proceso a la cola, con un timepo menor que el tiempo restante de ejecución del proceso en ejecución en la cpu, este es expulsado y se ejecuta el nuevo, Este esquema é conocido por Shortest-Remaining-Time-First (SRTF).
- Este algoritmo es no apropiativo.
- Su funcionamiento consiste en seleccionar el proceso más corto; Cuando termina se vuelve a elegir el más corto, de los procesos restantes. Y así hasta el final.

6.4. Algoritmo SJF (2)

- Ejemplo:

Proceso	*Tiempo llegada*	*Ráfaga CPU*
P1	0	7
P2	2	4
P3	4	1
P4	5	4

  - Diagrama de Gant para la planificación:



  - Tiempo de espera: TEP1=0 ut. TEP2= 8-2 =6 ut.
  - TEP3= 7-4 =3 ut. TEP4= 12-5 =7 ut.
  - Tiempo medio de espera: (0+6+3+7)/4=4 ut.
  - ***Proceso Tiempo llegada Ráfaga CPU***

6.4. Algoritmo SRTF ("Shortest Remaining Time First")

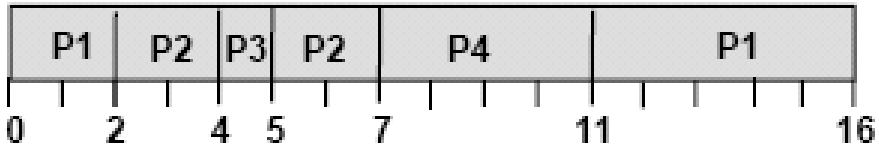
- Es un SJF apropiativo.
- Este algoritmo siempre ejecuta primero aquellos procesos a los que les queda menos tiempo para terminar .
- Este algoritmo también es conocido como 'optimo', pues con el se obtienen los mejores resultados.
- Óptimo al minimizar el tiempo medio de espera.

6.4. Algoritmo SRTF (2)

- Ejemplo:

Proceso	*Tiempo llegada*	*Ráfaga CPU*
P1	0	7
P2	2	4
P3	4	1
P4	5	4

  - Diagrama de Gant para la planificación:



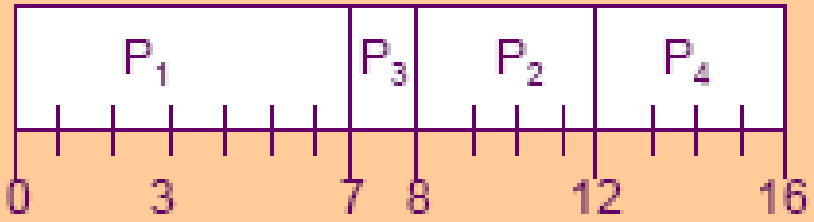
  - Tiempo de espera: TEP1= 0+9 = 9 ut. TEP2= (2-2) + 1 =1 ut.
  - TEP3= 4-4 =0 ut. TEP4= 7-5 =2 ut.
  - Tiempo medio de espera: (9+1+0+2)/4=3 ut.
  - ***Proceso Tiempo llegada Ráfaga CPU***

6.4.1 Ejemplo comparativo

SJF no-preemptivo

Processo	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (não-preemptivo)



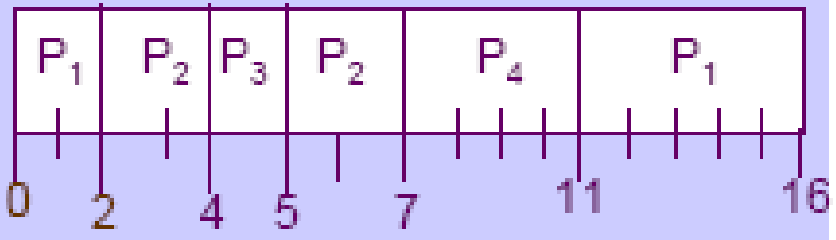
- Tempo médio de espera

  (0 + 6 + 3 + 7)/4 = 4

SJF preemptivo

Processo	Arrival Time	Burst Time
P_1	0.0	7
P_2	2.0	4
P_3	4.0	1
P_4	5.0	4

- SJF (preemptivo)



- Tempo médio de espera

  (9 + 1 + 0 +2)/4 = 3

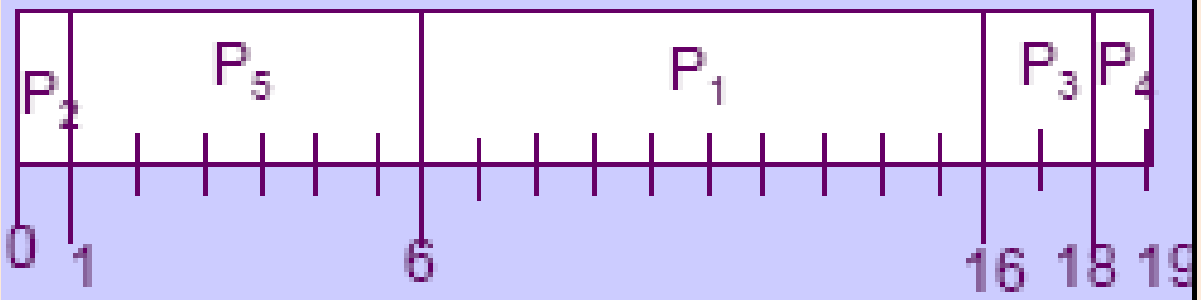
6.5. Algoritmo de prioridades

- Selecciona aquellos procesos que se encuentran en la cola de ‘listos’.
- En este algoritmo, los criterios de rendimiento, no son los más necesarios.
- El criterio principal es hacer que los procesos ‘en espera’ sean los primeros en ejecutarse
- El principal inconveniente, es que puede producir ‘inanición’, es decir si tenemos un proceso de prioridad baja, y muchos de alta, puede ocurrir que el primero no se ejecute nunca.
- Se puede llevar a cabo un proceso de envejecimiento, el cual hace ganar prioridad al primer proceso, permitiendo que se ejecute.

6.5. Algoritmo de prioridades (2)

Processo	Prioridade	Burst Time
P_1	3	10
P_2	1	1
P_3	3	2
P_4	4	1
P_5	2	5

- não-preemptivo



- Tempo médio de espera = 8.2

6.6. Algoritmo de Turno Rotatorio (Round Robin, RR)

- La desventaja principal es que cambia los procesos en ejecución con demasiada frecuencia.
- Lo que supone una pequeña perdida de tiempo.
- El tiempo perdido depende, del ‘tiempo de ejecución dado al proceso’ (Quanto).
- Para valores de ‘Quanto’ pequeños, el resultado es malo. Para valores grandes el algoritmo equivale al ‘FCFS’.

6.6.1. Ejemplo :RR

time quantum = 20 ms

Process	Burst Time
P_1	53
P_2	17
P_3	68
P_4	24

- A Carta Gantt é:

- Tipicamente, turnaround médio é mais elevado do que SJF, mas tem melhor *resposta*.