

Sistemas Operativos

Dr. Luis Gerardo de la Fraga

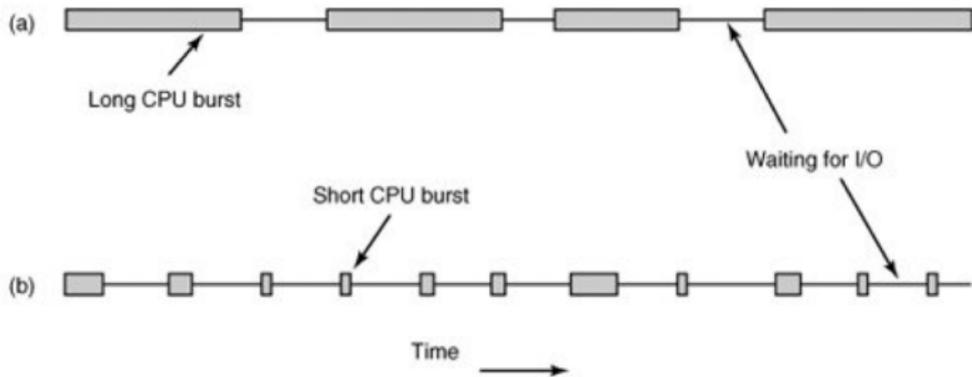
E-mail: fraga@cs.cinvestav.mx
<http://cs.cinvestav.mx/~fraga>

Departamento de Computación
Cinvestav

12 de junio de 2015

Despacho

- ▶ Cuando un SO tiene multiprogramación, existen múltiples procesos compitiendo al mismo tiempo por el CPU.
- ▶ Cuando más que un proceso están en el estado de listo y solo hay un CPU disponible, el CPU debe decidir que proceso corre primero.
- ▶ La parte del SO que decide esto se llama **el despachador**;
- ▶ el algoritmo que lo usa se llama **el algoritmo de despacho**.



Tipos de procesos: (a) limitados por el CPU, (b) limitados por E/S.

¿Cuándo se despacha?

- ▶ Se requiere absolutamente en dos ocasiones:
 1. Cuando un proceso termina.
 2. Cuando un proceso se bloquea debido a E/S o un semáforo.
- ▶ Aquí otro proceso debe escogerse para ejecutarse.

- ▶ Existen otras tres ocasiones donde puede usarse el despachador:
 1. Cuando se crea un proceso nuevo.
 2. Cuando ocurre una E/S
 3. Cuando ocurre una interrupción

Tipos de despacho de acuerdo con la interrupción de reloj

- ▶ Un algoritmo de despacho sin desplazamiento toma un proceso para ejecutar y lo deja correr hasta que se bloquea (por E/S a esperando a otro proceso) o hasta que voluntariamente deja el CPU.
- ▶ Un algoritmo de despacho con desplazamiento toma un proceso y lo deja ejecutar por un tiempo fijo máximo. Si se sigue ejecutando al final del intervalo de tiempo, el despachador lo suspende y ejecuta otro proceso.
- ▶ Para realizar el despacho con desplazamiento se necesita contar con una señal de reloj

Categorías de algoritmos de despacho (1/2)

- ▶ En ambientes diferentes se necesitan algoritmos de despacho diferente, debido a que en distintas áreas de aplicación se tienen distintas metas.
- ▶ Se pueden distinguir tres ambientes:
 1. En lotes
 2. Interactivo
 3. De tiempo real

En los ambientes [1] y posiblemente en [3] no se necesita el desplazamiento de procesos.

Metas de un algoritmo de despacho

▶ Para todos los sistemas

- ▶ Equidad en dar a cada proceso una cantidad justa de CPU
- ▶ Reforzamiento de la política para ver que la política actual se está llevando a cabo
- ▶ Balance para mantener ocupadas todas las partes del sistema.

▶ Para sistemas en lotes

- ▶ Rendimiento para maximizar los trabajos por hora
- ▶ Tiempo de vuelta para minimizar el tiempo entre envío y terminación
- ▶ Utilización del CPU para mantener ocupado el CPU todo el tiempo

▶ Para sistemas interactivos

- ▶ Tiempo de respuesta para responder rápidamente
- ▶ Proporcionalidad para satisfacer las expectativas de los usuarios

Categorías de algoritmos de despacho (2/2)

▶ **Sistemas de tiempo real**

- ▶ Satisfacción de los tiempos de terminación para prevenir pérdida de datos
 - ▶ Predictibilidad para evitar degradación de la calidad en sistemas multimedia
-
- ▶ De todas, la **equidad** es la más importante. Procesos comparables deberían tener un servicio comparable. No es justo dar a un proceso mucho más CPU que a otro proceso equivalente.
 - ▶ Otra meta general es mantener ocupadas todas las partes del sistema cuanto sea posible.

- ▶ Métricas usadas en centros de cómputo corporativos para ejecutar trabajos en lotes:
 1. Rendimiento (*throughput*), es el número de trabajos por segundo.
 2. Tiempo de vuelta (*turnaround time*), es el tiempo promedio desde el momento que se envía un trabajo hasta el momento que se completa. Este mide cuanto tiene que esperar un usuario promedio para obtener la salida.
 3. Utilización del CPU

- ▶ Para sistemas interactivos lo más importante es minimizar el tiempo de respuesta, esto es, el tiempo entre que se manda un comando y se obtiene la respuesta.
- ▶ En un sistema de tiempo real los procesos tienen líneas de finalización que deben o deberían cumplir.
- ▶ Por ejemplo, si una computadora controla un dispositivo que produce datos a intervalos regulares, si falla el proceso que recoge los datos en el tiempo puede resultar en pérdida de datos.
- ▶ En algunos sistemas de tiempo real, como los que involucran multimedia, la predictibilidad es importante.

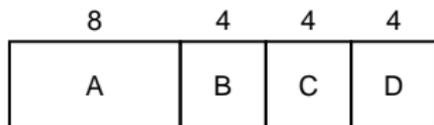
Despacho en sistemas en lotes

- ▶ Estos son los algoritmos usados en el despachador:
- ▶ El primer proceso que viene es el primero en ser servido.
- ▶ El trabajo más corto primero
- ▶ El tiempo restante más corto primero
- ▶ Despachador en tres niveles

El primer proceso que viene es el primero en ser servido.

- ▶ Es el más fácil de realizar
- ▶ Solo se necesita una lista simplemente ligada
- ▶ Tan pronto como llega un trabajo se pone al final de la cola de espera
- ▶ Cuando el proceso que corre se bloquea, el primer proceso en la cola es el siguiente en correr.
- ▶ Cuando un proceso bloqueado se vuelve listo, como cuando arriba un nuevo trabajo, se pone al final de la cola.

El trabajo más corto primero



(a)

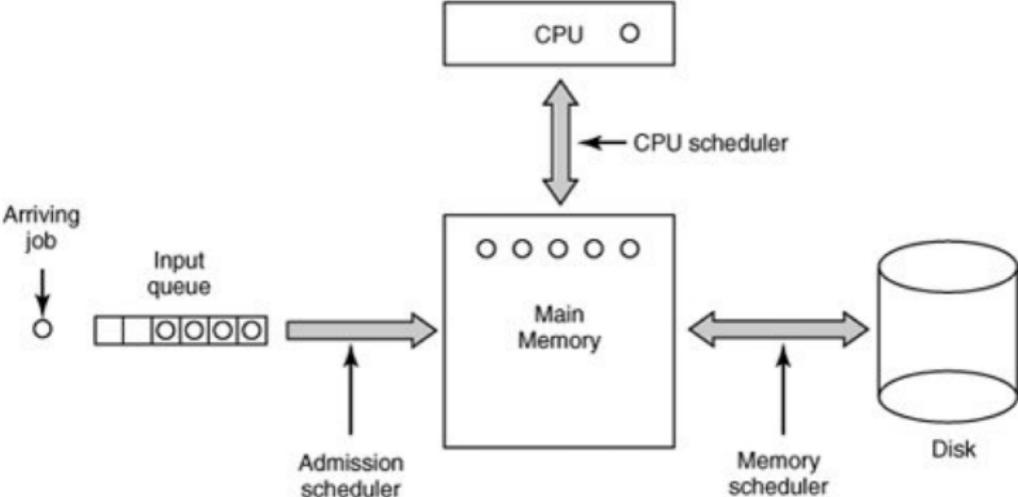


(b)

- ▶ Este es posiblemente óptimo según el tiempo de vuelta
- ▶ Solo es óptimo si se conocen de antemano todos los trabajos.

- ▶ Tanto el algoritmo de despacho del trabajo más corto primero como el del tiempo restante más corto primero necesitan conocer de antemano los tiempo de ejecución.

Despachador en tres niveles



Despacho en sistemas interactivos

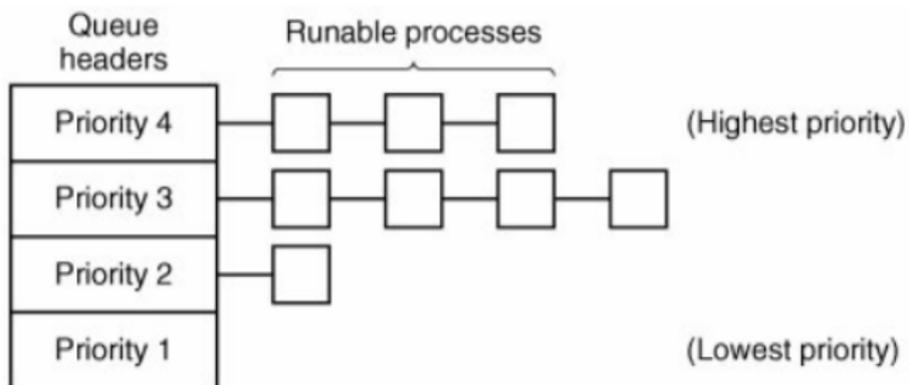
- ▶ Despacho todos contra todos (*round-robin*)
- ▶ Despacho por prioridades
- ▶ Colas múltiples
- ▶ El proceso más corto siguiente
- ▶ Despacho garantizado
- ▶ Despacho por lotería
- ▶ Despacho de compartición justa

Despacho todos contra todos

- ▶ Este es el algoritmo más viejo, más simple, más junto y más ampliamente usado.
- ▶ A cada proceso se le asigna una rebanada de tiempo que se le llama *quantum*
- ▶ Si el proceso llega al final de su quantum, se desplaza por otro proceso.
- ▶ Todo lo que se necesita mantener es una lista de los procesos en ejecución.
- ▶ ¿Cuánto debe ser el tamaño del quantum?
- ▶ A quantum de alrededor de 2050 mseg es un compromiso razonable.

Despacho por prioridades

- ▶ A cada proceso se le asigna una prioridad.
- ▶ Corre el proceso ejecutable con la mayor prioridad.
- ▶ Un algoritmo simple para dar un buen servicio a procesos limitados por E/S es poner su prioridad a $1/f$, donde f es la fracción del último quantum que ha usado el proceso.



El proceso más corto siguiente

- ▶ Trata de aplicar en sistemas interactivos este esquema que es óptimo en sistemas por lotes.

Despacho garantizado

- ▶ Realiza promesas realistas:
- ▶ Si hay n usuarios en una máquina, les da a cada uno $1/n$ del CPU.
- ▶ Si hay un solo usuario pero varios procesos, a cada uno de los procesos les da $1/n$ de los ciclos del CPU.
- ▶ ¿Cómo realizarlo?
- ▶ Tiene que estimarse la cantidad de CPU a la que se tiene derecho.
- ▶ Puede ser el tiempo de creación dividido entre n
- ▶ Un valor de 0.5 significa que el procesos ha tenido solo la mitad a lo que tiene derecho, un valor de 2.0 significa que el proceso ha tenido el doble a lo que tiene derecho.
- ▶ El algoritmo corre el proceso con el valor más bajo hasta éste que cambia arriba del valor de su competidor más cercano.

Despacho por lotería

- ▶ A cada proceso se le pueden dar tickets
- ▶ Los tickets pueden ser para varios recursos, incluido CPU
- ▶ El despachador escoge al azar un proceso y le da una rebanada del CPU por el ticket
- ▶ Aquí el número de tickets tiene el significado del porcentaje del recurso que obtendrá.

Despacho de compartición justa

- ▶ Les da a los usuarios la cantidad justa de CPU
- ▶ Si hay dos usuarios en una máquina y el primer usuario inicia 9 procesos y el usuario 2 solo un proceso, bajo el despacho de todos contra todos o prioridades iguales, el usuario 1 obtendrá el 90 % del CPU y el usuario 2 el 10 %.
- ▶ Este esquema despacha los procesos para garantizar que cada usuario obtendrá el 50 % del CPU.

Despacho en sistemas en tiempo real

- ▶ En este escenario típicamente existen uno o más dispositivos externos a la computadora
- ▶ La computadora debe reaccionar a estos estímulos de forma apropiada
- ▶ Existen sistemas de **tiempo real duros**, donde deben cumplirse todos los procesos, y
- ▶ sistemas de **tiempo real blandos**, donde se se pierde algún procesos no es crítico.
- ▶ Los eventos pueden ser **periódicos** o **aperiódicos**
- ▶ Si todos los tiempos de los procesos pueden cumplirse, entonces se dice que el sistema de tiempo real es despachable.
- ▶ Los sistema también pueden ser estáticos o dinámicos según el comportamiento del despachador.

Política versus mecanismo

- ▶ En el escenario donde un proceso genera varios hijos,
- ▶ ninguno de los algoritmos revisados maneja esta situación
- ▶ La solución es separar la política de despacho, esto es, como un proceso padre despacha a sus hijos, del mecanismo de despacho, esto es, el algoritmo general.
- ▶ El proceso padre podría poner y cambiar la prioridad de sus hijos.

Despacho de hilos

- ▶ Los algoritmos generalmente realizados son el de todos contra todos y por prioridad.

Despacho en linux

- ▶ Usa el algoritmo completamente justo (CFS, *completely fair scheduler*)
- ▶ Si introdujo en la versión 2.6.23
- ▶ Su complejidad en $O(1)$
- ▶ Su realización no está basada en colas de ejecución.
- ▶ Se realiza con un árbol rojo y negro que presenta un línea de tiempo de la ejecución futura de procesos.
- ▶ Adicionalmente, el despachador usa una granularidad de nanosegundos en la toma de tiempos.

Propiedades del despachador:

- ▶ El nodo más a la izquierda se escoge (porque tendrá el más bajo gasto en tiempo de ejecución) y se envía a ejecución.
- ▶ Si el proceso completa su ejecución, se quita del sistema y del árbol de despacho.
- ▶ Si el proceso alcanza su tiempo de ejecución máximo o se bloquea (voluntariamente o debido a una interrupción) se reinserta en el árbol de despacho con base en su nuevo gasto de tiempo de ejecución.
- ▶ Se selecciona el nodo más a la izquierda y se repite la iteración.

