Principios de Sistemas Operativos

Procesos

Armando Arce, arce@itcr.ac.cr

Tecnológico de Costa Rica

Conceptos de procesos

El Proceso

El concepto fundamental en cualquier sistema operativos es el concepto de proceso que consiste en una abstracción de lo que es un programa en ejecución.

- En un sistema de multiprogramación la CPU alterna de proceso en proceso, ejecutando cada uno durante decenas o centenas de milisegundos, esto da la apariencia de paralelismo.
- Los varios procesos pueden compartir un solo procesador mediante el uso de un algoritmo de planificación para establecer cuando detener un proceso e iniciar otro.
- Un mismo programa puede estar siendo ejecutado en un instante determinado por varios procesos a la vez.

Concepto de proceso

La diferencia entre un programa y un proceso es sutil, pero crucial.

- Un programa es un algoritmo expresado en alguna notación apropiada.
- Un proceso es una actividad de algún tipo: tiene programa, entrada, salida y un estado.
- El estado incluye los valores actuales del contador de programa, registros y variables.
- Generalmente, un proceso incluye también la pila del proceso, que contiene datos temporales (como los parámetros de las funciones, las direcciones de retorno y las variables locales), y una sección de datos, que contiene las variables globales.
- El proceso puede incluir, asimismo, un cúmulo de memoria, que es la memoria que asigna dinámicamente el proceso en el tiempo de ejecución.

Concepto de proceso

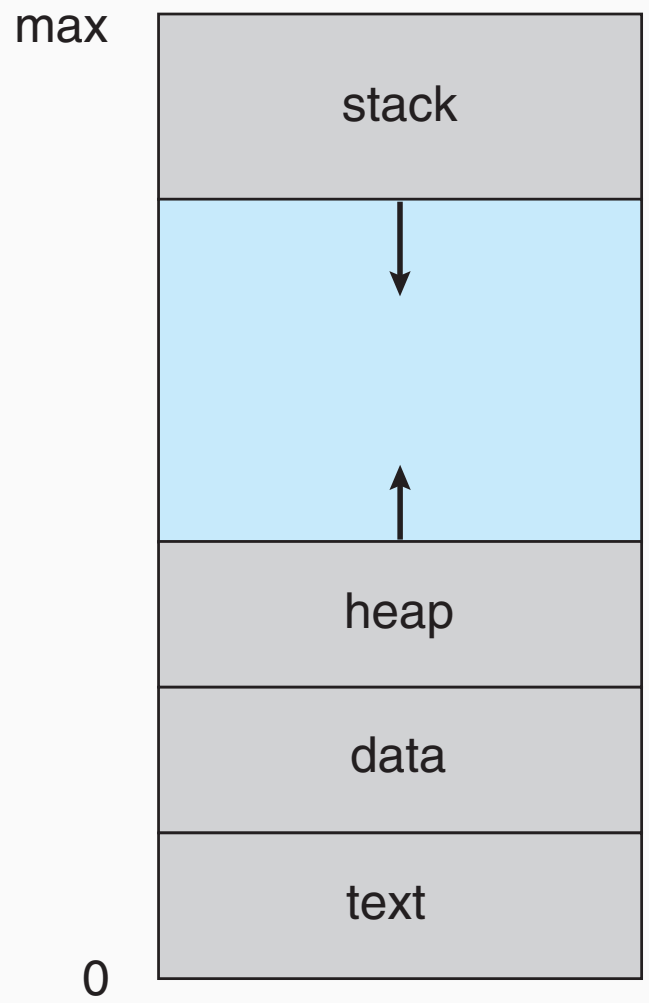

Figure 3.1 Process in memory.

Concepto de proceso

De esta forma, un programa por sí mismo no es un proceso; un programa es una entidad pasiva, un archivo que contiene una lista de instrucciones almacenadas en disco (a menudo denominada archivo ejecutable), mientras que un proceso es una entidad activa, con un contador de programa que especifica la siguiente instrucción que hay que ejecutar y un conjunto de recursos asociados.

- Un programa se convierte en un proceso cuando se cargue en memoria un archivo ejecutable.

Estado del proceso

A medida que se ejecuta un proceso, éste va cambiando estado. El estado de un proceso se define con base en actividad actual de dicho del mismo. Cada proceso puede estar en uno de los siguientes estados:

- Nuevo (N): El proceso está siendo creado
- Ejecutándose (0): Usando realmente la CPU en ese instante
- Preparado, Listo (R): El proceso está a la espera que le asignen el procesador
- Bloqueado, En espera (S): No puede ejecutarse en tanto no ocurra algún evento externo
- Terminado (): Ha terminado la ejecución del proceso

Estado del proceso

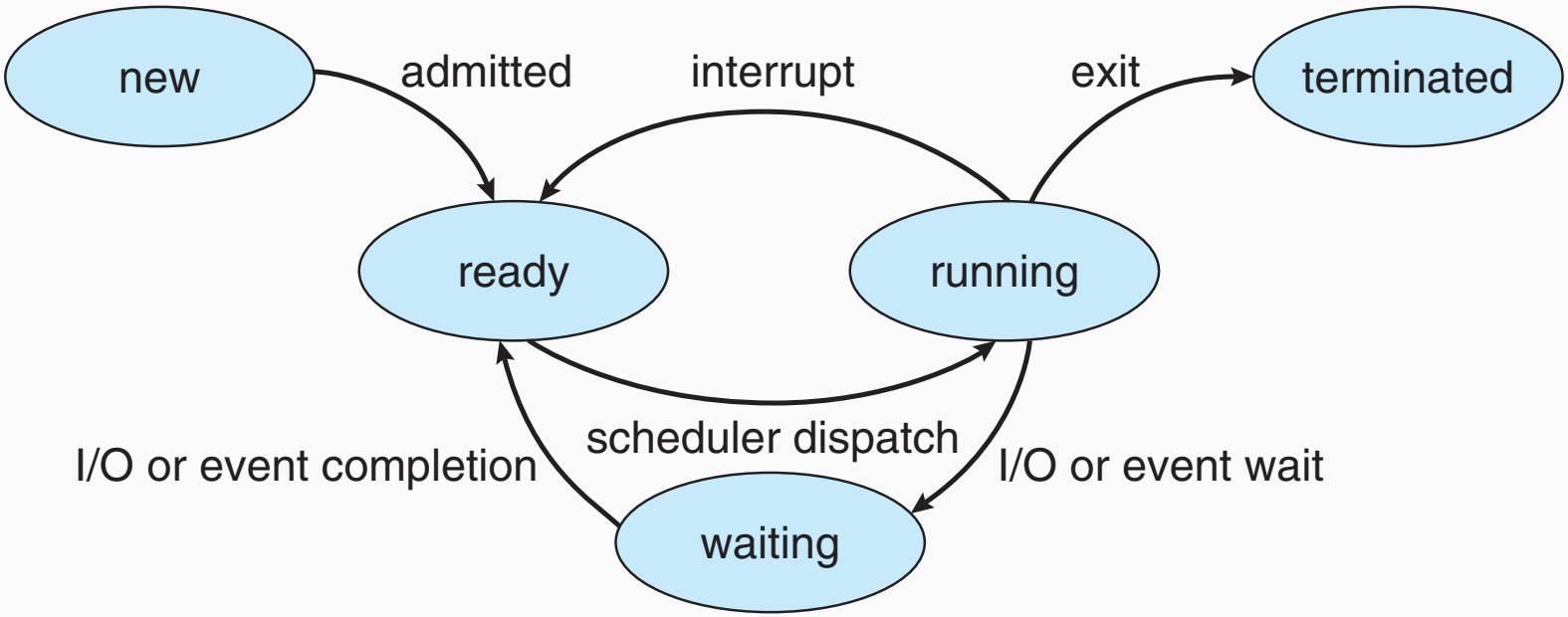

Figure 3.2 Diagram of process state.

Figure 2:

Estado del proceso

Todos los sistemas operativos cuentan con este tipo de estados aunque pueden cambiar su nombre.

- Es importante darse cuenta de que sólo puede haber un proceso ejecutándose en cualquier procesador en cada instante concreto.
- Sin embargo, pueden haber muchos procesos preparados y en espera.

Bloque de control de proceso (PCB)

Todo proceso tiene asociadas una entrada en la tabla de procesos que es mantenida por el sistema operativo.

- Esta entrada, llamada *bloque de control de procesos* (PCB), contiene toda la información de un proceso que se debe guardar cuando éste se conmuta del estado ejecutándose al estado preparado, a fin de poder reiniciarlo después como si nunca se hubiera detenido.

Bloque de control de proceso (PCB)

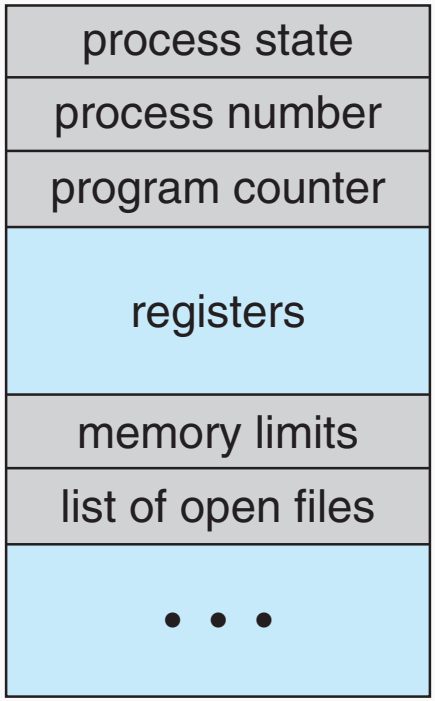

Figure 3.3 Process control block (PCB).

Figure 3:

Bloque de control de proceso (PCB)

Un bloque de control de proceso contiene muchos elementos de información asociados con un proceso específico, entre los que se incluyen:

- Estado del proceso: el estado puede ser nuevo, preparado, ejecución, en espera, detenido, etcétera.
- Contador del programa: el contador indica la dirección de la siguiente instrucción que va ejecutar dicho proceso
- Registros de la CPU: Incluyen los acumuladores, registros de índice, punteros de pila y registros de propósito general, además de toda la información de los indicadores de estado.

Bloque de control de proceso (PCB)

- Información de planificación de las CPU: esta información incluye la prioridad del proceso, los punteros a la cola de planificación y cualquiera otro parámetro de planificación requeridos.
- Información de administración de memoria: información acerca del valor de los registros base y límite, y las tablas de páginas o las tablas de segmentos.
- Información de estado de E/S: esta información incluye la lista de los dispositivos de E/S asignados al proceso, y una lista de archivos abiertos.

Bloque de control de proceso (PCB)

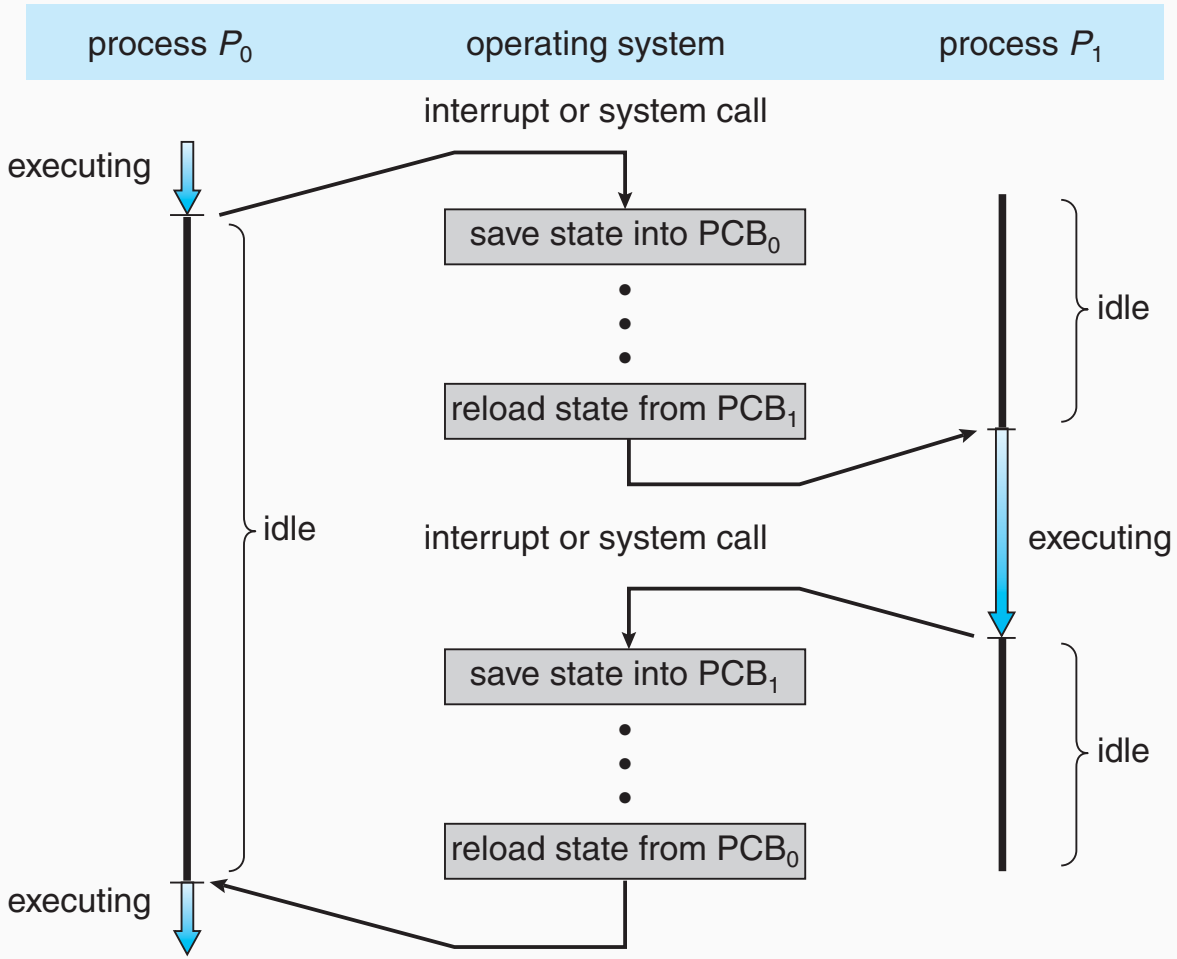

Figure 3.4 Diagram showing CPU switch from process to process.

Figure 4:

Planificación de procesos

Planificación de procesos

El objetivo de la multiprogramación es tener en ejecución varias procesos al mismo tiempo con el fin de maximizar la utilización de la CPU.

- El objetivo de los sistemas de tiempo compartido es conmutar la CPU entre los distintos procesos con tanta frecuencia que los usuarios puedan interactuar con cada programa mientras éste se ejecuta.
- Para conseguir estos objetivos, el planificador de procesos seleccionan un proceso disponible para ejecutar el programa en la CPU.
- En los sistemas de un solo procesador, nunca habrá más de un proceso en ejecución.

Colas de planificación

A medida que los procesos entran en el sistema, se coloca en una cola de trabajos que contiene todos los procesos del sistema.

- Los procesos que residen en la memoria principal y están preparados y en espera de ejecutarse se mantiene en una lista denominada cola de procesos preparados.
- Generalmente, esta cola se almacena en forma de lista enlazada. La cabecera de la cola de procesos preparados contiene punteros al primero y último bloques de control de procesos (PCB) de la lista.
- Cada PCB incluye un campo de puntero que apunta al siguiente PCB de la cola de procesos preparados.

Colas de planificación

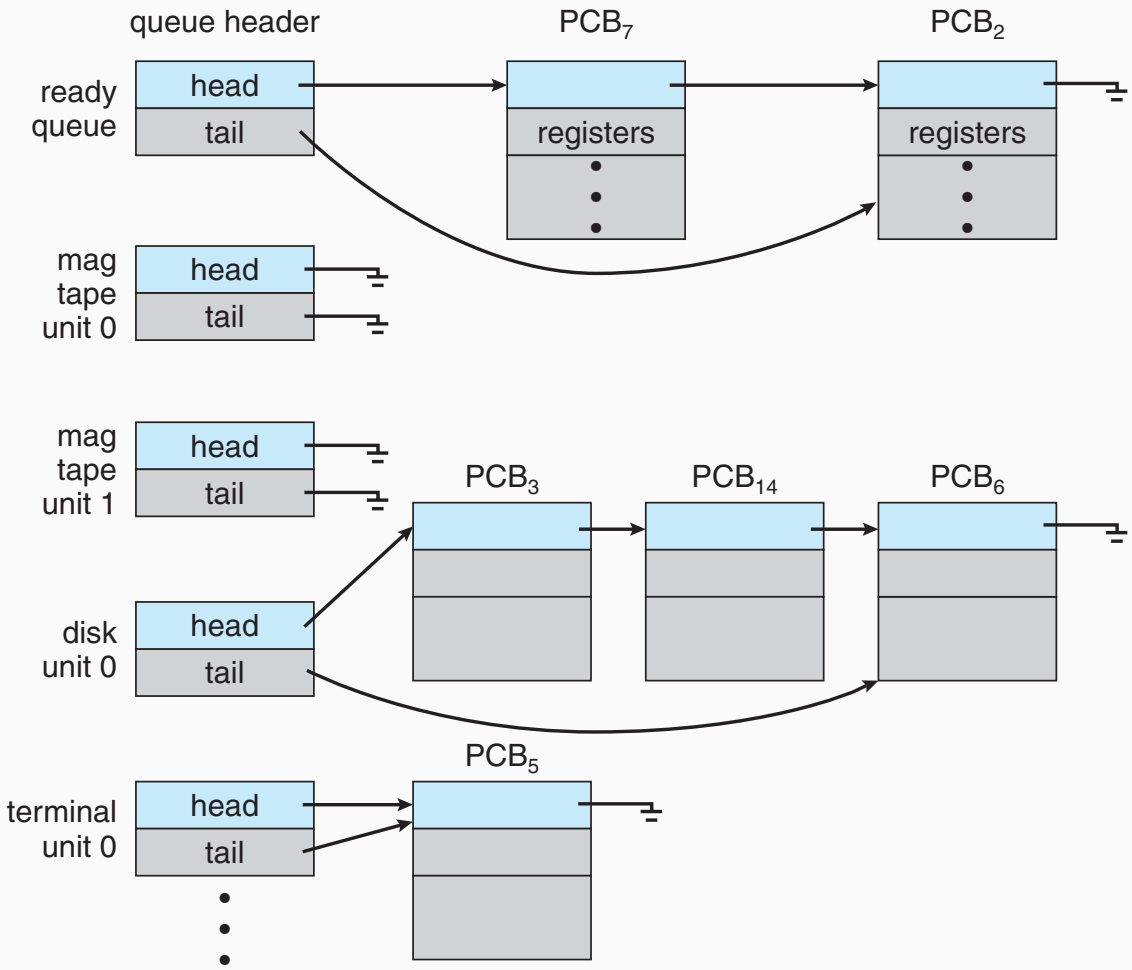

Figure 3.5 The ready queue and various I/O device queues.

Figure 5:

Colas de planificación

El sistema también incluye otras colas. Cuando se asigna la CPU a un proceso, éste se ejecuta durante un rato y finalmente termina, es interrumpido o espera a que se produzca un determinado suceso, como la terminación de una solicitud de E/S a un disco.

- Dado que hay muchos procesos en el sistema, el disco puede estar ocupado con la solicitud de E/S de algún otro proceso.
- Por tanto, el proceso que realice la nueva solicitud de E/S puede tener que esperar para poder acceder al disco.
- La lista de procesos en espera de un determinado dispositivo de entrada salida se denomina cola dispositivo.
- Cada dispositivo tiene su propia cola.

Colas de planificación

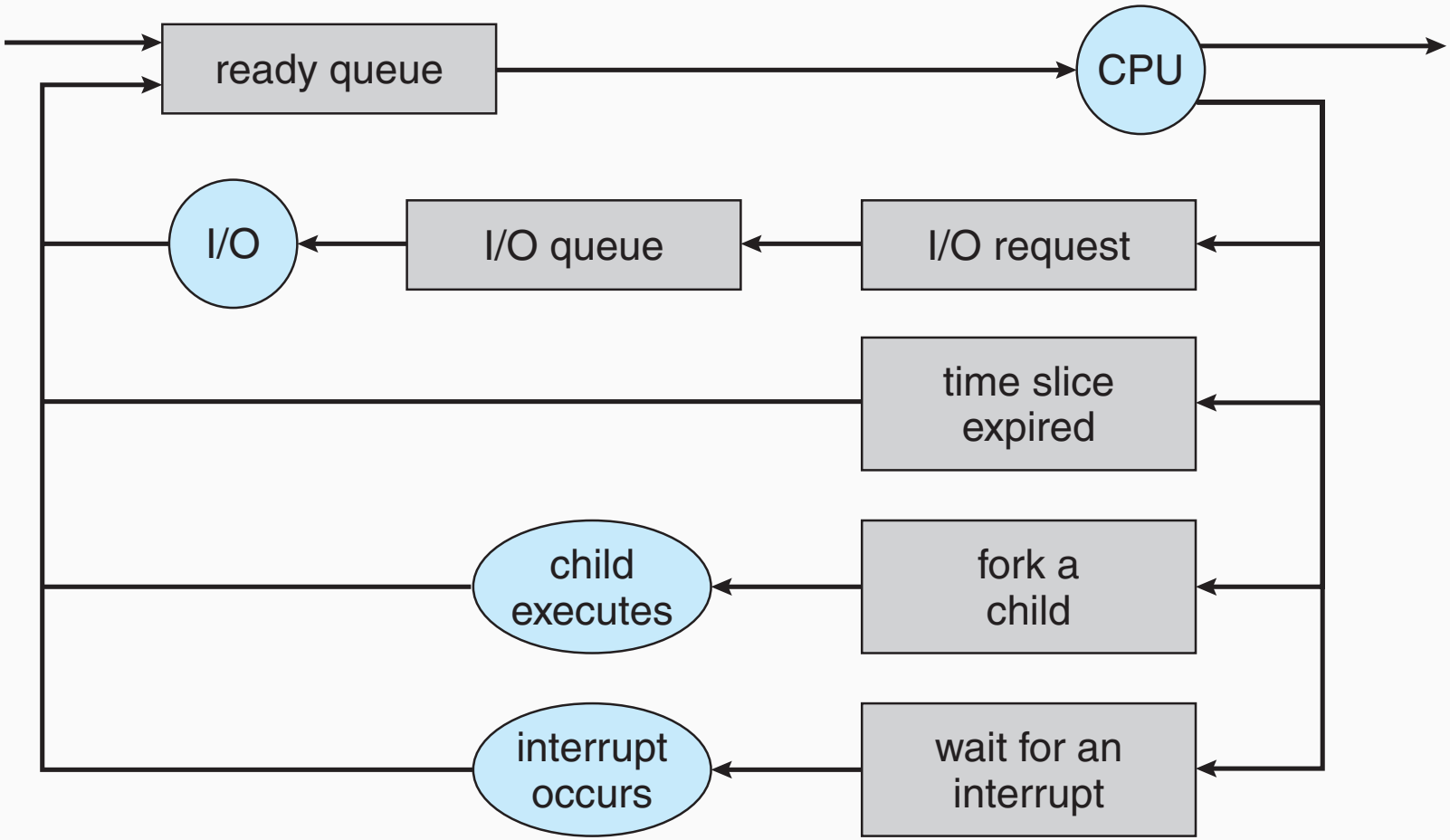

Figure 3.6 Queueing-diagram representation of process scheduling.

Figure 6:

Planificadores

Durante un tiempo de vida, los procesos se mueven entre las diversas colas de planificación.

- El sistema operativo, como parte de la tarea de planificación, debe seleccionar de alguna manera los podemos que se encuentran en estas colas.
- El proceso de selección se realiza mediante un planificador apropiado.

Planificadores

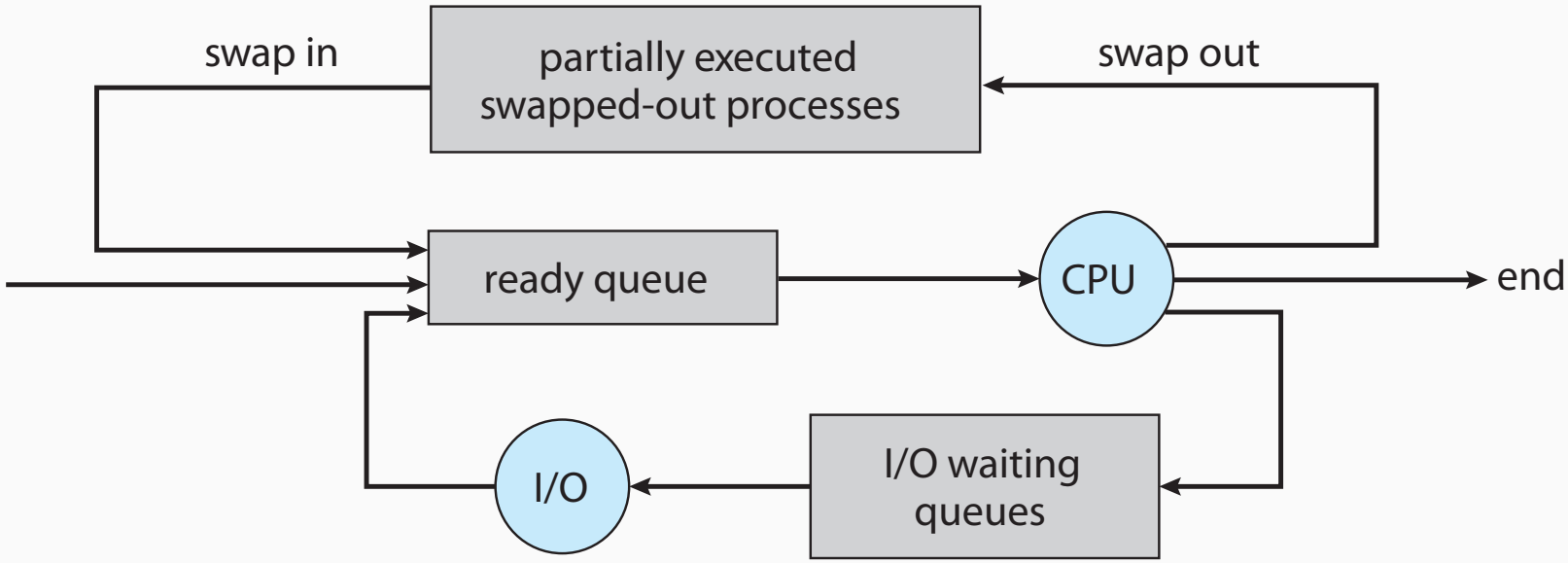

Figure 3.7 Addition of medium-term scheduling to the queueing diagram.

Figure 7:

Planificadores

A menudo, en un sistema de procesamiento por lotes, se envía más procesos de los que pueden ser ejecutados de forma inmediata.

- Estos procesos se guardan en la cola de un dispositivo de almacenamiento masivo (normalmente, un disco), donde se mantienen para su posterior ejecución.
- El planificador a largo plazo o planificar de trabajos selecciona procesos de esta cola y los carga en memoria para su ejecución.
- El planificador a corto plazo o planificador de la CPU selecciona dentro de los procesos que ya están preparados para ser ejecutados y asigna la CPU a uno de ellos.

Planificadores

Es importante que el planificador a largo plazo haga una elección cuidadosa. En general, la mayoría de los procesos pueden describirse como limitados por E/S o limitados por la CPU.

- Un proceso limitado por E/S es aquel que invierte la mayor parte de su tiempo en operaciones de E/S en lugar de en realizar cálculos.
- Por el contrario, un proceso limitado por la CPU genera solicitudes de E/S con poca frecuencia, usando la mayor parte de su tiempo en realizar cálculos.
- Es importante que el planificador a largo plazo seleccione una adecuada mezcla de procesos equilibrando los procesos limitados por el E/S y los progresos limitados por CPU, de forma que ambos tipos de dispositivos pasen ocupados la mayor cantidad del tiempo.

Planificadores

Algunos sistemas operativos, como los sistemas de tiempo compartido, pueden introducir un nivel intermedio adicional de planificación.

- La idea clave subyacente a un planificador a mediano plazo es que, en ocasiones, puede ser ventajoso eliminar procesos de la memoria (con lo que dejan de contender por la CPU) y reducir así el grado de multiprogramación.
- Después, el proceso puede volver a cargarse memoria, continuando su ejecución en el punto en que se interrumpió. Este esquema se denomina intercambio.
- El intercambio puede ser necesario para mejorar la mezcla de procesos o porque un cambio en los requisitos de memoria haya hecho que se sobrepase la memoria disponible, requiriendo que se libere memoria.

Cambio de contexto

Las interrupciones hacen que el sistema operativo obligue a la CPU a abandonar su tarea actual, para ejecutar una rutina del kernel. Estos sucesos se produce con frecuencia en sistemas de propósito general.

- Cuando se produce la interrupción, el sistema tiene que guardar el contexto actual del proceso que está ejecutando en la CPU, de modo que pueda restaurar dicho contexto cuando procesamiento concluya, suspendiendo el proceso y reanudando después.
- El contexto se almacena en el PCB del proceso que incluye el valor de los registros de la CPU, el estado del proceso y la información de gestión de memoria.

Cambio de contexto

La conmutación de la CPU a otro proceso requiere una salvaguarda del estado del proceso actual y la restauración del estado de otro proceso diferente.

- Esta tarea se conoce como cambio de contexto.
- Cuando se produce un cambio de contexto, el kernel guarda el contexto del proceso antiguo en su PCB y carga el contexto almacenado del nuevo por eso que se ha decidido ejecutar.

Cambio de contexto

El tiempo dedicado al cambio de contexto es tiempo desperdiciado, dado que el sistema no realiza ningún trabajo útil durante la computación.

- La velocidad del cambio contexto varía de una máquina otra, dependiendo de la velocidad de memoria, del número de registros que tenga que copiarse y de la existencia de instrucciones especiales (por ejemplo, una instrucción para cargar almacenar todos los registros).
- Las velocidades típicas son del orden de unos pocos milisegundos.

Operaciones sobre procesos

Identificador de procesos

La mayoría de los sistemas operativos identifican los procesos mediante un identificador de proceso unívoco o PID.

- El PID normalmente es un número entero que se asigna en forma secuencial conforme se crean nuevos procesos.
- Cada proceso cuenta con un proceso padre que es quien lo creó. El identificador del padre generalmente de conoce como PPID.
- El proceso también pertenece a algún usuario, que se identifica mediante un UID.

Operaciones sobre procesos

En la mayoría de sistemas, los procesos pueden ejecutarse en forma concurrente y pueden crearse y eliminarse dinámicamente.

- Por tanto, estos sistemas deben proporcionar un mecanismo para la creación y terminación de presos.

Creación de un proceso

Un proceso puede crear otros varios procesos nuevos mientras se ejecuta; para ello se utiliza una llamada al sistema específica para la creación de procesos.

- El proceso creador se denomina proceso padre y los nuevos procesos son los hijos de dicho proceso.
- Cada uno de estos procesos nuevos puede a su vez su vez crear otros procesos, dando lugar a un árbol de procesos.

Creación de un proceso

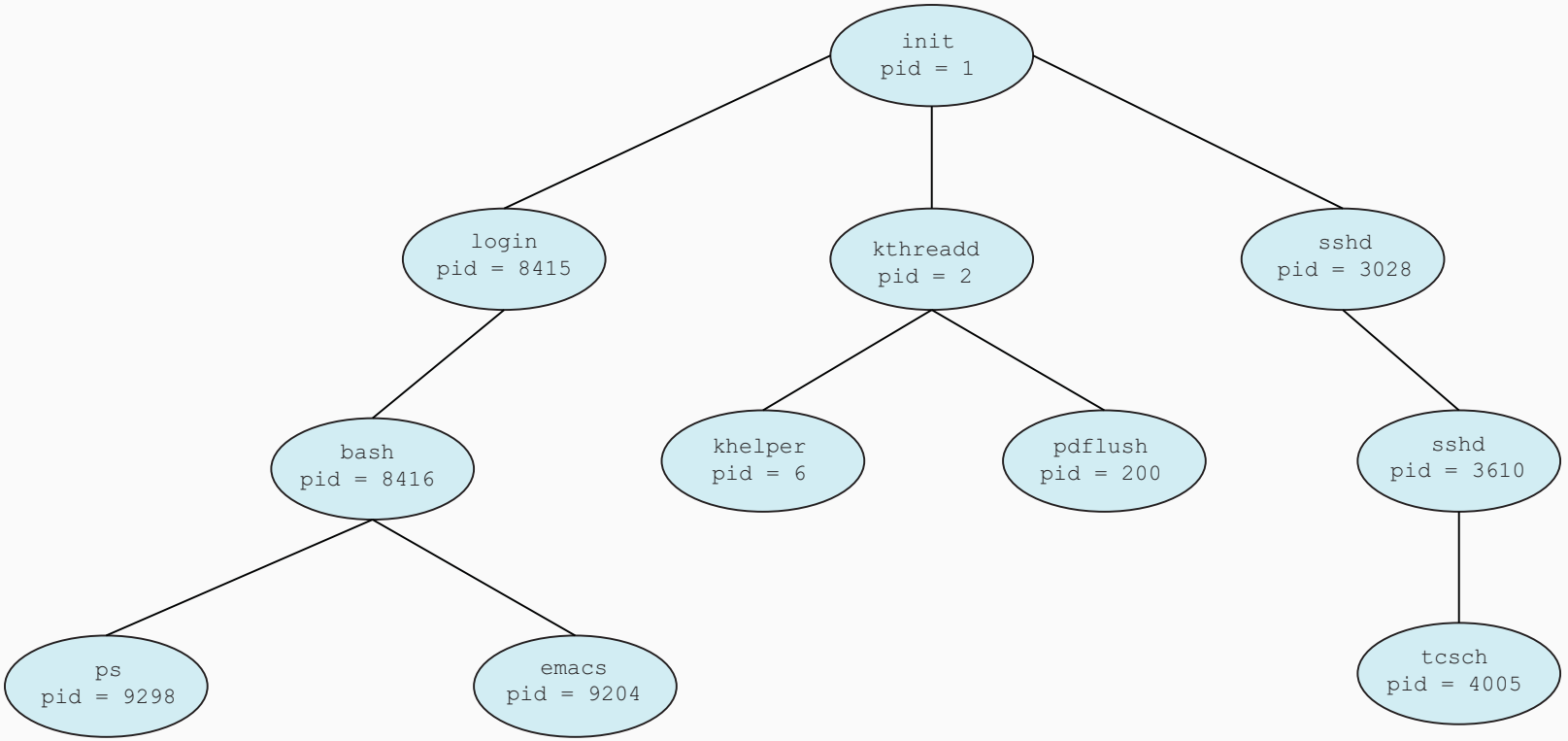

Figure 3.8 A tree of processes on a typical Linux system.

Figure 8:

Creación de un proceso

En general, un proceso necesitará ciertos recursos (tiempo de CPU, memoria, archivos, dispositivos de E/S) para llegar llevar a cabo sus tareas.

- Cuando un proceso creo su proceso, dicho subproceso puede obtener sus recursos directamente el sistema operativo o puede estar restringido a un subconjunto de los recursos del proceso padre.
- El padre pueda tener que repartir sus recursos entre los hijos, o puede compartir algunos recursos (como la memoria o los archivos) con alguno de sus hijos.
- Restringir un proceso hijo a un subconjunto de los recursos del padre evita que un proceso pueda sobrecargar el sistema creando demasiados subprocesos.

Creación de un proceso

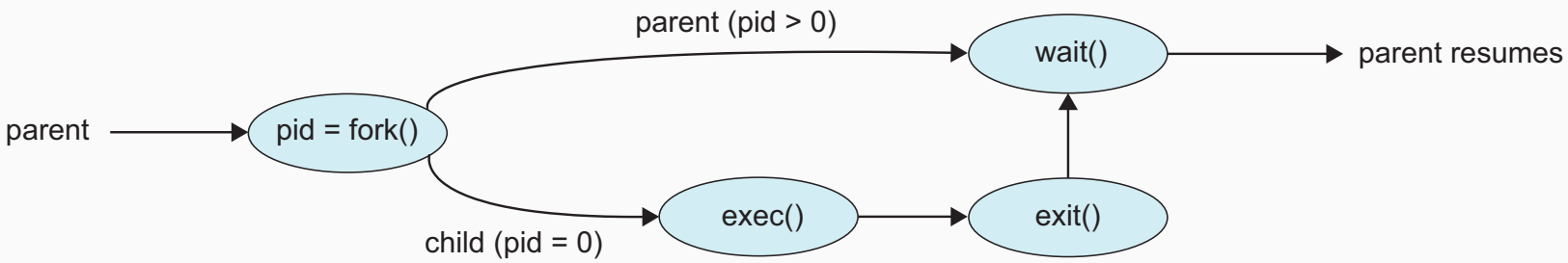

Figure 3.10 Process creation using the `fork()` system call.

Figure 9:

Creación de un proceso

Existen diferentes formas de realizar la creación de procesos en un sistema de cómputo.

- Cuando el sistema operativo arranca se crean múltiples procesos, algunos en primer plano (interactivos) y otros en segundo plano (no interactivos).
- Los procesos de segundo plano que realizan alguna actividad específica (correo,web,etc) se les llama demonios.
- Los procesos mismos también pueden crear otros procesos mediante llamadas al sistema.
- Esto da lugar, a veces, a varios procesos independientes pero que interactúan entre sí para resolver una tarea común.
- También, los usuarios pueden iniciar procesos cuando lo requieran. Esto se puede hacer mediante un ambiente gráfico o directamente mediante comandos.

Jerarquías de procesos

Los sistemas necesitan de algún mecanismo para crear y destruir procesos según sea necesario durante la operación.

- En UNIX, los procesos se crean con la llamada al sistema FORK (bifurcar), que crea una copia idéntica del proceso invocador.
- Cada proceso tiene su padre, y cero, uno, dos o más hijos.
- Aquí todos los hijos y demás descendientes forman juntos un grupo de procesos.

Terminación de procesos

Un proceso termina cuando ejecuta su última instrucción y pide al sistema operativo que lo elimine usando la llamada al sistema *exit()*.

- En este momento, el proceso puede devolver un valor de estado (normalmente, un entero) a su proceso padre (a través de la llamada al sistema *wait()*).
- El sistema operativo libera la asignación de todos los recursos del proceso, incluyendo las memorias física y virtual, los archivos abiertos y los búferes de E/S.

Terminación de procesos

Otra causa para que un proceso termine es la aparición de un error causado por el proceso, a menudo debido a un error de programación.

- La terminación puede producirse también en otras circunstancias. Un proceso puede causar la terminación de otro proceso a través de la adecuada llamada al sistema (por ejemplo, *kill()*).
- Normalmente, dicha llamada al sistema sólo puede ser invocada por el padre del proceso que va a terminar.
- Se debe hacer nota que un padre necesita conocer las identidades libres. Por tanto, cuando un proceso crea un proceso nuevo, se pasa al padre la identidad del proceso hijo que se acaba de crear.

Terminación de procesos

Un padre puede terminar la ejecución de uno de sus hijos por diversas razones, como por ejemplo, la siguientes:

- El proceso hijo a excedido el uso de alguno de los recursos que se han asignado. Para determinar si tal cosa ha ocurrido, el padre debe disponer de un mecanismo para inspeccionar el estado de sus hijos.
- La tarea asignada pues digo ya no es necesaria.
- El padre abandona el sistema, y el sistema operativo no permite que un proceso hijo continúe si su padre ya terminó.

Terminación de procesos

Algunos sistemas no permiten que un hijo siga existiendo si su proceso padre se ha completado.

- En tales sistemas, si un proceso termina (sea normal o anormalmente), entonces todos sus hijos también deben terminarse.
- Éste fenómeno, conocido como terminación en cascada, normalmente lo inicia el sistema operativo.

Referencia bibliográfica

A.SILBERSCHATZ, P. GALVIN, y G. GAGNE, Operating Systems Concepts, Cap. 3, 9a Edición, John Wiley, 2013.