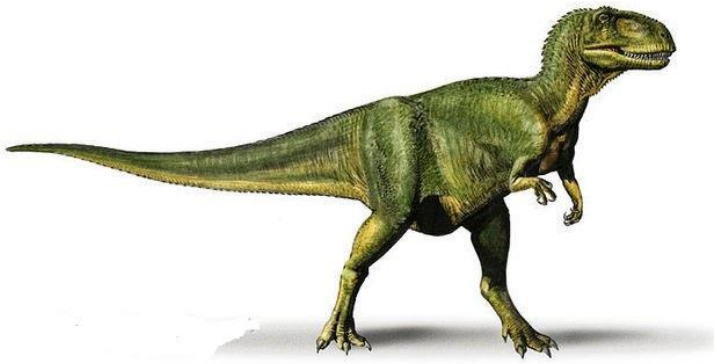


SISTEMAS OPERATIVOS Y REDES

Planificación de la CPU

Contenido

Planificación de la CPU

Prologo

Bienvenidos al resumen adaptado del libro Fundamentos de Sistemas Operativos (Silberchtaz-Galvin-Gagne) que acompaña la cátedra Sistemas Operativos y Redes de la carrera de Informática. Este documento tiene como objetivo proporcionar un acceso conciso y claro a los conceptos fundamentales presentados en el libro de texto principal, adaptado específicamente para satisfacer las necesidades y los objetivos del curso.

Los sistemas operativos son la columna vertebral de la informática moderna, y comprender su funcionamiento es esencial para cualquier estudiante de ciencias de la computación. Este resumen condensa los conceptos clave, los principios fundamentales y las técnicas avanzadas discutidas en el libro de texto original, proporcionando una guía clara y accesible para el estudio y la comprensión de los sistemas operativos.

A lo largo de este resumen, encontrarás una serie de capítulos que abordan temas importantes como la gestión de procesos, la gestión de memoria, la planificación de la CPU, entre otros. Cada capítulo se ha diseñado cuidadosamente para ofrecer una visión completa y bien estructurada de los conceptos presentados en el libro de texto, acompañado de ejemplos prácticos y explicaciones claras para facilitar el aprendizaje.

Espero que este resumen sea una herramienta valiosa para complementar tu estudio de sistemas operativos y te ayude a alcanzar tus objetivos te deseamos mucho éxito en tu cursada.

Lic. Mariano Vargas

Objetivos de este resumen

- Presentar los mecanismos de planificación de la CPU, que constituyen los cimientos de los sistemas operativos multiprogramados.
- Describir los distintos algoritmos para la planificación de la CPU.
- Exponer los criterios de evaluación utilizados para seleccionar un algoritmo de planificación de la CPU para un determinado sistema.

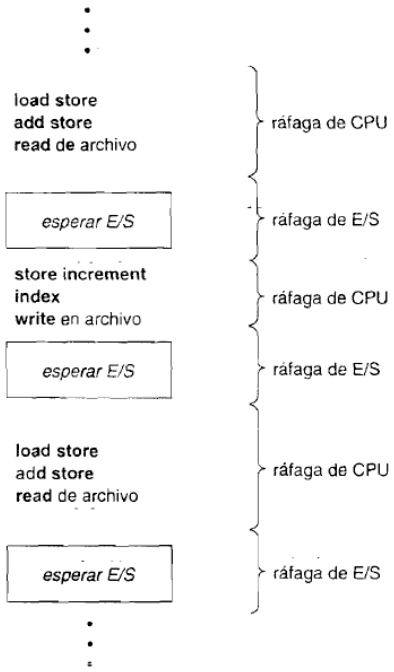
Conceptos básicos

En un sistema de un único procesador, sólo puede ejecutarse un proceso cada vez; cualquier otro proceso tendrá que esperar hasta que la CPU quede libre y pueda volver a planificarse. El objetivo de la multiprogramación es tener continuamente varios procesos en ejecución, con el fin de maximizar el uso de la CPU. La idea es bastante simple: un proceso se ejecuta hasta que tenga que esperar, normalmente porque es necesario completar alguna solicitud de E/S. En un sistema informático simple, la CPU permanece entonces inactiva y todo el tiempo de espera se desperdicia; no se realiza ningún trabajo útil. Con la multiprogramación, se intenta usar ese tiempo de forma productiva. En este caso, se mantienen varios procesos en memoria a la vez. Cuando un proceso tiene que esperar, el sistema operativo retira el uso de la CPU a ese proceso y se lo cede a otro proceso. Este patrón se repite continuamente y cada vez que un proceso tiene que esperar, otro proceso puede hacer uso de la CPU.

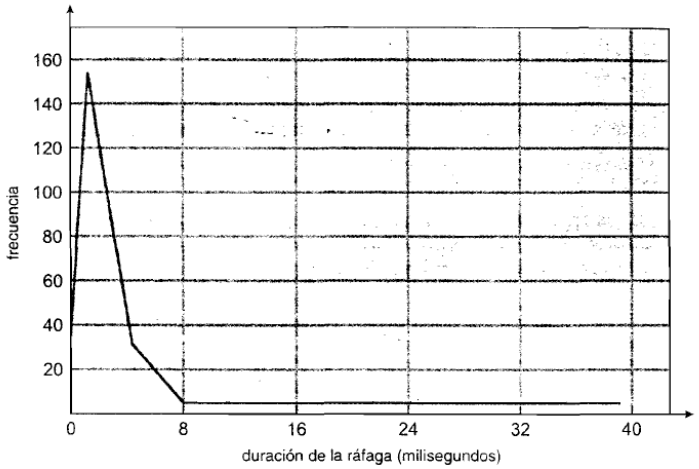
Este tipo de planificación es una función fundamental del sistema operativo; casi todos los recursos de la computadora se planifican antes de usarlos. Por supuesto, la CPU es uno de los principales recursos de la computadora, así que su correcta planificación resulta crucial en el diseño del sistema operativo.

Ciclo de ráfagas de CPU y de E/S

La adecuada planificación de la CPU depende de una propiedad observada de los procesos: la ejecución de un proceso consta de un **ciclo** de ejecución en la CPU, seguido de una espera de E/S; los procesos alternan entre estos dos estados. La ejecución del proceso comienza con una ráfaga de CPU. Ésta va seguida de una ráfaga de E/S, a la cual sigue otra ráfaga de CPU, luego otra ráfaga de E/S, etc. Finalmente, la ráfaga final de CPU concluye con una solicitud al sistema para terminar la ejecución (ver figura abajo).



La duración de las ráfagas de CPU se ha medido exhaustivamente en la práctica. Aunque varían enormemente de un proceso a otro y de una computadora a otra, tienden a presentar una curva de frecuencia similar a la mostrada en la Figura Generalmente, la curva es de tipo exponencial o hiperexponencial, con un gran número de ráfagas de CPU cortas y un número menor de ráfagas de CPU largas. Normalmente, un programa limitado por E /S presenta muchas ráfagas de CPU cortas. Esta distribución puede ser importante en la selección de un algoritmo apropiado para la planificación de la CPU.



Planificador de la CPU

Cuando la CPU queda inactiva, el sistema operativo debe seleccionar uno de los procesos que se encuentran en la cola de procesos listos para ejecución. El planificador a corto plazo (o planificador de la CPU) lleva a cabo esa selección del proceso. El planificador elige uno de los procesos que están en memoria listos para ejecutarse y asigna la CPU a dicho proceso.

Observa que la cola de procesos listos no necesariamente tiene que ser una cola FIFO (first-in, first-out). Como veremos al considerar los distintos algoritmos de planificación, una cola de procesos listos puede implementarse como una cola FIFO, una cola prioritaria, un árbol o simplemente una lista enlazada no ordenada. Sin embargo, conceptualmente, todos los procesos que se encuentran en la cola de procesos listos se ponen en fila esperando la oportunidad de ejecutarse en la CPU. Los registros que se almacenan en las colas son, generalmente, bloques de control de proceso (PCB) que describen los procesos en cuestión.

Planificación apropiativa

Puede ser necesario tomar decisiones sobre planificación de la CPU en las siguientes cuatro circunstancias:

1. Cuando un proceso cambia del estado de ejecución al estado de espera (por ejemplo, como resultado de una solicitud de E/S o de una invocación de wait para esperar a que termine uno de los procesos hijo).
2. Cuando un proceso cambia del estado de ejecución al estado preparado (por ejemplo, cuando se produce una interrupción).
3. Cuando un proceso cambia del estado de espera al estado preparado (por ejemplo, al completarse una operación de E/S).
4. Cuando un proceso termina.

En las situaciones 1 y 4, no hay más que una opción en términos de planificación: debe seleccionarse un nuevo proceso para su ejecución (si hay algún proceso en la cola de procesos listos). Sin embargo, en las situaciones 2 y 3 sí que existe la opción de planificar un nuevo proceso o no.

Cuando las decisiones de planificación sólo tienen lugar en las circunstancias 1 y 4, decimos que el esquema de planificación es sin desalojo o cooperativo; en caso contrario, se trata de un esquema apropiativo. En la planificación sin desalojo, una vez que se ha asignado la CPU a un proceso, el proceso se mantiene en la CPU hasta que ésta es liberada bien por la terminación del proceso o bien por la conmutación al estado de espera. Este método de planificación era el utilizado por Microsoft Windows 3.x; Windows 95 introdujo la planificación apropiativa y todas las versiones siguientes de los sistemas operativos Windows han usado dicho tipo de planificación. El sistema operativo Mac OS X para Macintosh utiliza la planificación apropiativa; las versiones anteriores del sistema operativo de Macintosh se basaban en la planificación cooperativa. La planificación cooperativa es el único método que se puede emplear en determinadas plataformas hardware, dado que no requiere el hardware especial (por ejemplo, un temporizador) necesario para la planificación apropiativa. Lamentablemente, la planificación apropiativa tiene un coste asociado con el acceso a los datos compartidos. Consideremos el caso de dos procesos que compartan datos y supongamos que, mientras que uno está actualizando los datos, resulta desalojado con el fin de que el segundo proceso pueda ejecutarse. El segundo proceso podría intentar entonces leer los datos, que se encuentran en un estado incoherente. En tales situaciones, son necesarios nuevos mecanismos para coordinar el acceso a los datos compartidos; veremos este tema en otro resumen.

La técnica de desalojo también afecta al diseño del kemel del sistema operativo. Durante el procesamiento de una llamada al sistema, el kernel puede estar ocupado realizando una actividad en nombre de un proceso. Tales actividades pueden implicar cambios importantes en los datos del kernel (por ejemplo, en las colas de E /S). ¿Qué ocurre si el proceso se desaloja en mitad de estos cambios y el kemel (o el controlador del dispositivo) necesita leer o modificar la misma estructura? El resultado será un auténtico caos. Ciertos sistemas operativos, incluyendo la mayor parte de las versiones de UNIX, resuelven este problema esperando a que se complete la llamada al sistema o a que se transfiera un bloque de E /S antes de hacer un cambio de contexto. Esta solución permite obtener una estructura del kemel simple, ya que el kernel no desalojará ningún proceso, mientras que las estructuras de datos del kemel se encuentren en un estado incoherente. Lamentablemente, este modelo de ejecución del kernel no resulta adecuado para permitir la realización de cálculos en tiempo real y el multiprocesamiento. Estos problemas y sus soluciones se describen en otros resúmenes.

Dado que, por definición, las interrupciones pueden producirse en cualquier momento, y puesto que no siempre pueden ser ignoradas por el kernel, las secciones de código afectadas por las interrupciones deben ser resguardadas de un posible uso simultáneo. Sin embargo, el sistema operativo tiene que aceptar interrupciones casi continuamente, ya que de otra manera podrían perderse valores de entrada o sobreescribirse los valores de salida; por esto, para que no puedan acceder de forma concurrente varios procesos a estas secciones de código, lo que se hace es desactivar las interrupciones al principio de cada sección y volver a activarlas al final. Es importante observar que no son muy numerosas las secciones de código que desactivan las interrupciones y que, normalmente, esas secciones contienen pocas instrucciones.

Concurrencia:

La concurrencia se refiere a la capacidad del sistema para ejecutar múltiples procesos o hilos de manera simultánea y eficiente. Esto implica que varias tareas pueden ejecutarse en paralelo, ya sea en un único procesador (mediante técnicas como la multitarea o la multiprogramación) o en varios procesadores (mediante multiprocesamiento). La concurrencia permite que los recursos del sistema, como la CPU, la memoria y los dispositivos de E/S, se utilicen de manera más efectiva, lo que mejora el rendimiento, la capacidad de respuesta y la eficiencia del sistema.

Despachador

Otro componente implicado en la función de planificación de la CPU es el despachador. El despachador es el módulo que proporciona el control de la CPU a los procesos seleccionados por el planificador a corto plazo. Esta función implica lo siguiente:

- Cambio de contexto.
- Cambio al modo usuario.
- Salto a la posición correcta dentro del programa de usuario para reiniciar dicho programa.

El despachador debe ser lo más rápido posible, ya que se invoca en cada conmutación de proceso. El tiempo que tarda el despachador en detener un proceso e iniciar la ejecución de otro se conoce como **latencia de despacho**.

Criterios de planificación

Los diferentes algoritmos de planificación de la CPU tienen distintas propiedades, y la elección de un algoritmo en particular puede favorecer una clase de procesos sobre otros. A la hora de decidir qué algoritmo utilizar en una situación particular, debemos considerar las propiedades de los diversos algoritmos.

Se han sugerido muchos criterios para comparar los distintos algoritmos de planificación. Las características que se usen para realizar la comparación pueden afectar enormemente a la determinación de cuál es el mejor algoritmo. Los criterios son los siguientes:

- **Utilización de la CPU**. Deseamos mantener la CPU tan ocupada como sea posible. Conceptualmente, la utilización de la CPU se define en el rango comprendido entre el 0 y el 100 por cien. En un sistema real, debe variar entre el 40 por ciento (para un sistema ligeramente cargado) y el 90 por ciento (para un sistema intensamente utilizado).
- **Tasa de procesamiento**. Si la CPU está ocupada ejecutando procesos, entonces se estará llevando a cabo algún tipo de trabajo. Una medida de esa cantidad de trabajo es el número de procesos que se completan por unidad de tiempo, y dicha medida se denomina tasa de procesamiento. Para procesos de larga duración, este valor puede ser de un proceso por hora; para transacciones cortas, puede ser de 10 procesos por segundo.
- **Tiempo de ejecución**. Desde el punto de vista de un proceso individual, el criterio importante es cuánto tarda en ejecutarse dicho proceso. El intervalo que va desde el instante en que se ordena la ejecución de un proceso hasta el instante en que se completa es el tiempo de ejecución. Ese tiempo de ejecución es la suma de los

períodos que el proceso invierte en esperar para cargarse en memoria, esperar en la cola de procesos listos, ejecutarse en la CPU y realizar las operaciones de E/S.

- **Tiempo de espera**. El algoritmo de planificación de la CPU no afecta a la cantidad de tiempo durante la que un proceso se ejecuta o hace una operación de E/S ; afecta sólo al período de tiempo que un proceso invierte en esperar en la cola de procesos listos El tiempo de espera es la suma de los períodos invertidos en esperar en la cola de procesos listos.
- **Tiempo de respuesta**. En un sistema interactivo, el tiempo de ejecución puede no ser el mejor criterio. A menudo, un proceso puede generar parte de la salida con relativa rapidez y puede continuar calculando nuevos resultados mientras que los resultados previos se envían a la salida para ponerlos a disposición del usuario. Por tanto, otra medida es el tiempo transcurrido desde que se envía una solicitud hasta que se produce la primera respuesta. Esta medida, denominada tiempo de respuesta, es el tiempo que el proceso tarda en empezar a responder, no el tiempo que tarda en enviar a la salida toda la información de respuesta. Generalmente, el tiempo de respuesta está limitado por la velocidad del dispositivo de salida.

El objetivo consiste en maximizar la utilización de la CPU y la tasa de procesamiento, y minimizar el tiempo de ejecución, el tiempo de espera y el tiempo de respuesta. En la mayoría de los casos, lo que se hace es optimizar algún tipo de valor promedio. Sin embargo, en determinadas circunstancias, resulta deseable optimizar los valores máximo y mínimo en lugar del promedio. Por ejemplo, para garantizar que todos los usuarios tengan un buen servicio, podemos tratar de minimizar el tiempo de respuesta máximo.

Algoritmos de Planificación

Se explica en clase