Main Memory

Chapter 8

8.1 Background

La memoria consiste de listas de bytes con direcciones.
El CPU jala las instrucciones de memoria según el program counter.
Un ciclo basico de ejecucion toma las instrucciones de la memoria, la decodifica, y manda a traer los comandos a memoria.
La memoria solo ve direcciones.

8.1.1 Basic Hardware

La memoria principal y los registros están construidos dentro del procesador, y solo el almacenamiento de propósito general puede accesarlos.
Si la data no está en memoria se necesita mover antes de que el CPU trabaje con ella. Poder ingresar a memoria puede tomar varios ciclos del reloj del CPU.
En tal caso el procesador espera porque no tiene la data necesaria para completar la instrucción que le piden.

8.1.2 Address Binding

Un programa está en el disco como un ejecutable. Para que el proceso se ejecute se necesita que el programa se traiga a la memoria principal.
Dependiendo de la cantidad que necesite, la data se estará moviendo entre disco y memoria durante su ejecución.
Normalmente el binding de las instrucciones y data a direcciones de memoria se puede hacer en cualquier paso: tiempo de compilación, load time o execution time.

8.1.4 Dynamic Loading

Para usar mejor la memoria se usa el dynamic loading. Lo que hace él es no cargar totalmente la rutina a memoria hasta que se llame.
La ventaja de dynamic loading es que la rutina es cargada solo cuando se necesita, esto es muy provechoso para situaciones con una gran cantidad de data.

8.1.5 Dynamic Linking and Shared Libraries

Librerías dynamically linked son sistemas de librerías que le ayudan al usuario a correr applicaciones.

Dynamic linking es similar al dynamic loading, pues es pospuesto hasta que se le llame, facilitando las llamadas por el mismo recurso usado constantemente.

8.2 Swapping

Un proceso puede ser intercambiado temporalmente fuera de memoria y traído de regreso, el intercambio o swapping hace posible que las memorias físicas no se llenen de mala manera.

8.2.1 Standard Swapping

Swapping estándar: El intercambio estándar implica mover procesos entre la memoria principal y un espacio de respaldo.

- Debe ser lo suficientemente grande para acomodar copias de todas los registros de memoria para todos los usuarios, y debe dar acceso directo a estas imágenes de memoria.
- El sistema mantiene una cola lista de todos los procesos que están usando memoria y están listas para ejecutarse.
- Swapping se limita por factores como que el proceso tiene que estar totalmente dormido (idle).

8.3 Contiguous Memory Allocation

En la asignación de memoria contigua, cada proceso está contenido en una sola sección de la memoria que es contigua a la sección que contiene el siguiente proceso.

8.3.1 Memory Protection

Podemos evitar que un proceso acceda a la memoria que no posee mediante la combinación de dos ideas, el límite de registros y la reubicación de registros.
El límite de registros tiene especificado dónde empiezan y termina el área del proceso, si se pasa no se puede acceder a ella por riesgo de seguridad.

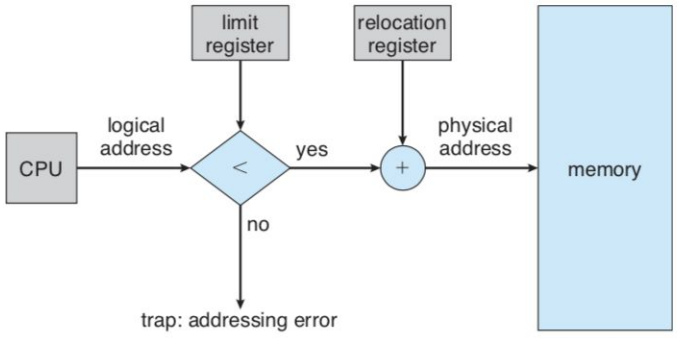

Figure 8.6 Hardware support for relocation and limit registers.

8.3.2 Memory Allocation

Al asignar memoria es más fácil dividir la memoria en varias particiones de tamaño fijo.

- Cada partición puede contener exactamente un proceso.
- El grado de multiprogramación está limitado por el número de particiones.

En este método de partición múltiple, cuando una partición está libre, se selecciona un proceso de la cola de entrada y se carga en la partición libre. Cuando el proceso termina, la partición queda libre para otro proceso.

El *problema de asignación de almacenamiento dinámico* se refiere a cómo satisfacer una solicitud de tamaño n de una lista de agujeros libres.

Algunas soluciones para resolver este problema son:

Primer ajuste Se asigna el primer *agujero* que sea lo suficientemente grande. La búsqueda puede comenzar ya sea al principio o en la ubicación donde terminó la búsqueda anterior. Se dejar de buscar cuando se encuentra un lugar disponible.
Mejor ajuste. Se asigna el agujero más pequeño que sea lo suficientemente grande. Se busca por tamaño dentro de la lista y se basa en eso.
El peor ajuste. Se asigna el agujero más grande.

8.3.3 Fragmentation

La fragmentación externa existe cuando hay suficiente espacio total en la memoria para satisfacer una solicitud, pero los espacios disponibles no son contiguos: el almacenamiento se fragmenta en una gran cantidad de agujeros pequeños. (copiado del libro)
Solución: la compactación. El objetivo es mezclar los contenidos de la memoria para colocar toda la memoria libre en un bloque grande.

8.4 Segmentation

8.4.1 Basic Method

Cada segmento tiene un nombre y una longitud.
Las direcciones especifican tanto el nombre del segmento como el desplazamiento dentro del segmento.

8.4.2 Segmentation Hardware

Una dirección lógica consta de dos partes: un número de segmento, s, y un desplazamiento en ese segmento, d. El número de segmento se usa como un índice para la tabla de segmentos.

El desplazamiento d de la dirección lógica debe estar entre el inicio y el límite del segmento.Si no ocurre un error.

8.5 Paging

La segmentación hace que el espacio de direcciones físicas de un proceso no sea contiguo.
La paginación es un esquema de administración de memoria que ofrece esta ventaja.
Pero, la paginación evita la fragmentación externa y la necesidad de compactación, mientras que la segmentación no lo hace.
También resuelve el problema de colocar trozos de memoria de diferentes tamaños en la tienda de respaldo.

8.5.1 Basic Method

Cada dirección generada por el CPU se divide en dos partes: un número de página (p) y un desplazamiento de página (d).
El número de página se utiliza como un índice en una tabla de páginas.
Cuando usamos un esquema de paginación, no tenemos una fragmentación externa (cualquier espacio libre puede asignarse a un proceso que lo necesite)
Pero si puede caer en fragmentación interna.
Si los requisitos de memoria de un proceso no coinciden con los límites de la página, es posible que el último cuadro asignado no esté completamente lleno.

8.5.2 Hardware Support

La implementación del hardware de la tabla en paginación.

La tabla de páginas se implementa como un conjunto de registros dedicados.
Cada acceso a la memoria debe pasar por el mapa de paginación, por lo que la eficiencia es importante.
El despachador de la CPU vuelve a cargar estos registros, al igual que vuelve a cargar los otros registros.

8.5.3 Protection

La protección de la memoria en un entorno paginado se logra mediante los bits de protección asociados con cada trama.
Cuando el bit se establece como no válido, la página no se encuentra en el espacio de direcciones lógicas del proceso.

8.6 Structure of the Page Table

8.6.1 Hierarchical Paging

Una dirección lógica se divide en un número de página de 20 bits y un desplazamiento de página de 12 bits.
La página de la página de la página, el número de la página se divide más.
La traducción de direcciones funciona desde la tabla de páginas externa hacia adentro (páginas asignada hacia adelante)

8.6.2 Hashed Page Tables

Para manejar espacios de direcciones de más de 32 bits se usa una tabla de páginas con hash, con el valor de hash como el número de página virtual.
Cada entrada en la tabla hash contiene una lista enlazada de elementos (linked list) que tienen hash en la misma parte (para manejar las colisiones).
Cada elemento consta de tres campos:

1. el número de página virtual
2. el valor del marco de página asignado
3. un puntero al siguiente elemento en la lista enlazada