Sistemas Operativos

Carmen Azorín Martí

Dirección relativa vs lógica:

	DIRECCIÓN LÓGICA	DIRECCIÓN FÍSICA
Qué es?	Es la dirección virtual generada por la CPU	La dirección física es una ubicación en una unidad de memoria.
Espacio de dirección	El conjunto de todas las direcciones lógicas generadas por la CPU en referencia a un programa se denomina Espacio de direcciones lógicas.	El conjunto de todas las direcciones físicas asignadas a las direcciones lógicas correspondientes se denomina Dirección física.
Visibilidad	El usuario puede ver la dirección lógica de un programa.	El usuario nunca puede ver la dirección física del programa
Acceso	El usuario usa la dirección lógica para acceder a la dirección física.	El usuario no puede acceder directamente a la dirección física.
Generacion	La dirección lógica es generada por la CPU	La dirección física es calculada por MMU

*La MMU se explica después

Mapa de memoria de un proceso:
Imagen del proceso: mapa + PCB (registro especial donde el sistema operativo agrupa toda la información que necesita conocer respecto a un proceso particular).

El mapa esta formado por: código, datos con valor inicial, datos sin valor inicial y la pila (argumentos del programa).

Formas del sistema operativo de organizar y gestionar memoria física:

- Organización:

    - Contigua: la asignación de almacenamiento para un programa se hace en un único bloque de posiciones continuas en memoria (particiones fijas y particiones variables).
    - No contigua: permiten dividir el programa en bloques colocados en zonas no necesariamente continuas en memoria principal (paginación, segmentación y segmentación paginada).
- Gestión: estrategias para obtener un rendimiento óptimo.
    - Estrategias de asignación
    - Estrategias de sustitución
    - Estrategias de búsqueda
- Protección
    - El SO se protege de los procesos de usuario
    - Los procesos de usuario se protegen entre ellos

***Swapping*:**
Es un mecanismo para mover programas entre memoria principal y auxiliar, normalmente disco (dispositivo de swap, con espacio para albergar las imágenes de memoria de los procesos).
El factor principal en el tiempo de intercambio es el tiempo de transferencia.
El intercambiador (o swapper) tiene responsabilidades:

- Seleccionar procesos para retirarlos de MP
- Seleccionar procesos para incorporarlos a MP
- Gestionar y asignar el espacio de intercambio

Memoria Virtual y utilidades:
La memoria virtual, es una técnica de administración de la memoria real que permite al sistema operativo brindarle al software de usuario y a sí mismo un espacio de direcciones mayor que la memoria real o física.
Porque el tamaño del programa, los datos y la pila pueden exceder la cantidad de memoria física disponible.
Se usa un almacenamiento a dos niveles:

- Memoria principal: partes del proceso necesarias en un momento dado
- Memoria secundaria (auxiliar): espacio de direcciones completo del proceso

Es necesario:

- Saber qué se encuentra en memoria principal
- Una política de movimiento entre MP y MS (auxiliar)

Además, la Memoria Virtual:

- Resuelve el problema del crecimiento dinámico de los procesos
- Permite aumentar el grado de multiprogramación

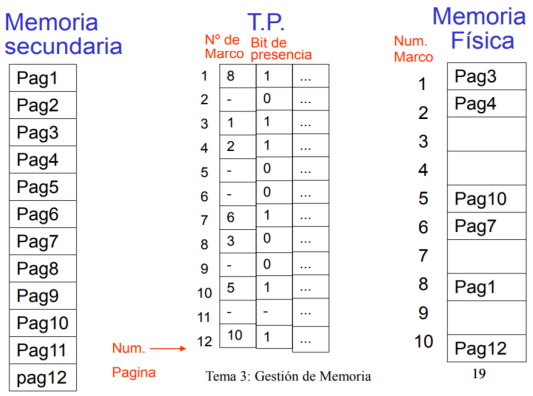
Unidad de gestión de Memoria:
La MMU es un dispositivo hardware gestionado por el SO que traduce direcciones lógicas a direcciones físicas.

En el esquema MMU mas simple, el valor del registro base se añade a cada dirección generada por el proceso al mismo tiempo que es enviado a memoria.
Además de la traducción, el MMU:

- Detecta si la dirección aludida se encuentra o no en MP
- Genera una excepción si no se encuentra en MP

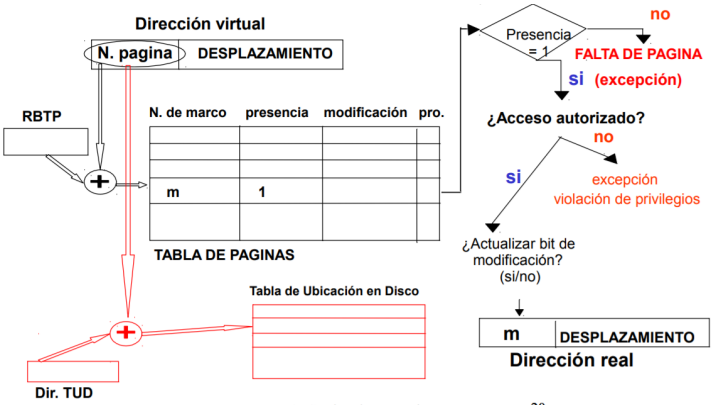
Mecanismos de paginación, segmentación y segmentación paginada:
Paginación:

- La memoria física se divide en bloques de tamaño fijos (marcos de página). El tamaño es potencia de dos, de 0.5 a 8 Kb.
- El espacio lógico de un proceso se divide en bloques del mismo tamaño, denominados páginas.
- Los marcos de páginas contienen páginas de los procesos.
- Las direcciones lógicas (generadas por la CPU) se dividen en número de páginas (p) y desplazamiento dentro de la página (d).
- Las direcciones físicas se dividen en número de marco (m, dirección base del marco donde está almacenada la página) y desplazamiento (d).
- Cuando la CPU genere una dirección lógica será necesario traducirla a la dirección física correspondiente (con el MMU).
- La tabla de páginas (una por proceso): mantiene información necesaria para realizar dicha traducción.
    a. Una entrada por cada página del proceso:
        - Número de marco (dirección base) en el que está almacenada la página si está en MP
        - Bit de presencia (en MP, si no está -> excepción)
        - Bit de modificación
        - Modo de acceso autorizado a la página (bits de protección) (si no tiene acceso -> excepción por violación de privilegios)



- Tabla de ubicación en disco (una por proceso): ubicación de cada página en el almacenamiento secundario (auxiliar).
- Tabla de marcos de página: usada por el SO y contiene información sobre cada marco de página.

- Esquema de traducción:
    a. Se mira el número de página en la dirección virtual
    b. Se busca el número de marco en la Tabla de Páginas
        - Se mira si tiene presencia
            - Si no -> excepción por falta de página
        - Se mira si tiene acceso autorizado
            - Si no -> excepción por violación de privilegios

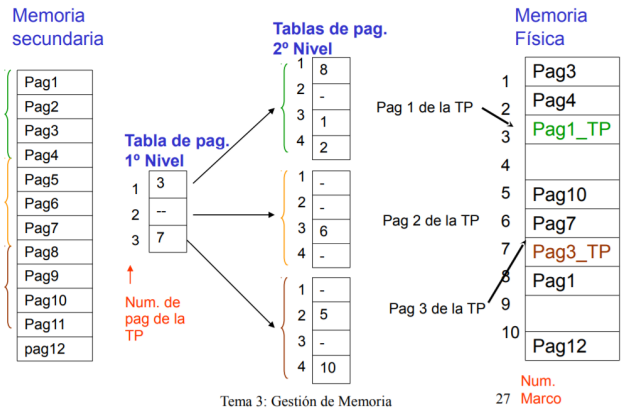


        - Se pregunta si se quiere actualizar el bit de modificación
            - esto cambia la dirección real

- Si ha habido falta de página
    a. Bloquear el proceso
    b. Encontrar la ubicación en disco de la página solicitada (tabla de ubicación en disco)
    c. Encontrar un marco libre. Si no hay, se opta por desplazar una página de MP
    d. Cargar la página desde disco al marco de MP
    e. Actualizar tablas (bit de presencia =1)
    f. Desbloquear proceso
    g. Reiniciar la instrucción que provocó la falta de página
- Implementación Tabla de Páginas:
    a. La Tabla de Páginas se mantiene en MP
    b. El registro base de la tabla de páginas (RBTP) apunta a la tabla de páginas (suele almacenarse en el PCB del proceso)
    c. Cada acceso a una instrucción o dato requiere dos accesos a memoria: uno a la tabla de páginas y otro a memoria (resuelto con TLB -> búfer de traducción anticipada)
    d. Un problema adicional viene determinado por el tamaño de la tabla de páginas
        - Por ejemplo:
            - Dirección virtual: 32 bits
            - Tamaño de página = 4kB = 2^(12) bits

        - Tamaño del campo de desplazamiento = 12 bits
        - Tamaño número de página virtual = 20 bits
        - Número de páginas virtuales = 2^20 bits
    - Solución: reducir el tamaño de la tabla de páginas:
        - Paginación multinivel: paginar las tablas de páginas.
- Paginación multinivel:
    a. La partición de la tabla de páginas permite al SO dejar particiones no usadas sin cargar hasta que el proceso las necesita. Aquellas porciones del espacio de direcciones que no se usan no necesitan tener una tabla.
    b. Lo que hacemos es dividir la tabla de páginas en partes del tamaño de una página.
    c. La dirección lógica se divide en:
        - Número de página (n bits):
            - Un número de página p1 (=k)
            - Desplazamiento de página p2 (=n-k)
        - Desplazamiento de página d (m bits)
    d. Así una dirección lógica es de la forma:

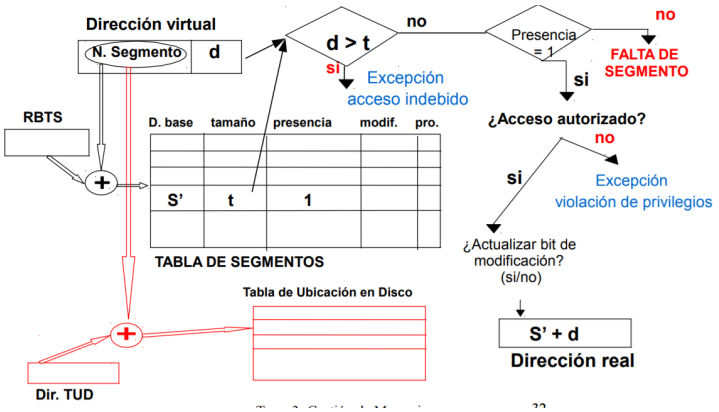
p1	p2	d



<u>Segmentación</u>:

- Un programa es una colección de unidades lógicas -segmentos- (procedimientos, funciones, pila, tabla de símbolos…)
- Tabla de segmentos:
    - Una dirección lógica es una dupla: <numero_de_segmento, desplazamiento>
    - Aplica direcciones bidimensionales definidas por el usuario en direcciones físicas de una dimensión
    - Cada entrada de la tabla tiene los siguientes elementos:
        - Base: dirección física donde reside el inicio del segmento en memoria
        - Tamaño: longitud del segmento

- Implementación tabla de segmentos:
  - Se mantiene en MP
  - El registro base de la tabla de segmentos (RBTS) apunta a la tabla de segmentos (se almacena en el PCB del proceso)
  - El registro longitud de la tabla de segmentos (STLR) indica el número de segmentos del proceso
  - El número de segmentos (s), generado en una dirección lógica, es legal si s<STLR (se almacena en el PCB del proceso)
- Esquema de traducción:
  - La dirección virtual nos da el número de segmento
  - Con la RBTS sabemos qué tabla de segmentos es
  - Buscamos la base donde empieza el segmento en memoria
  - Miramos el tamaño
    - Si el desplazamiento asociado a la dirección lógica es mayor que el tamaño -> Excepción por acceso indebido
  - Miramos el bit de presencia
    - Si está a 0 -> Excepción por Falta de Segmento
  - Miramos el bit de acceso
    - Si está a 0 -> Excepción por violación de privilegios
  - Se pregunta si se quiere actualizar el bit de modificación
    - Se modifica en la dirección real (Dirección base + desplazamiento)



<u>Segmentación Paginada</u>:

- La variabilidad del tamaño de los segmentos y el requisito de memoria contigua dentro de un segmento, complica la gestión de MP y MS
- Por otro lado, la paginación simplifica la gestión pero complica más los temas de compartición y protección.
- Si combinamos ambos enfoques, obtenemos la mayoría de las ventajas de la segmentación y eliminamos los problemas de memoria compleja.
- Esquema de traducción:
  - En la dirección virtual aparecen: <s', d (<p, d'>)> <numero_segmento, desp_segment (<numero_página, desp_pagina>)

    - Con el RBTS sabemos cuál es la tabla de segmentos a acceder
    - Miramos el tamaño (t) asociado al número de segmento
    - Si d ≤ t, entonces sabemos la tabla de página del segmento s, y en la dirección virtual nos dan p
    - Miramos el bit de presencia porque ya tenemos m (nºmarco)
    - Ya tenemos la dirección física <m, d'>

Memoria Virtual: Gestión

Gestión de Memoria Virtual con paginación. Criterios de clasificación:

- Políticas de asignación:
    - Fija
    - Variable
- Políticas de búsqueda:
    - Paginación por demanda
    - Paginación anticipada (NO es prepaginación)
- Políticas de sustitución:
    - Sustitución local
    - Sustitución global

Además, deben cumplirse ciertos criterios fuera de las políticas de sustitución:

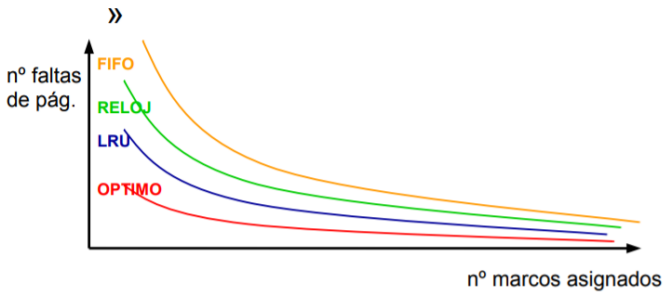
- Páginas limpias frente a sucias
    - Minimizar el coste de transferencia
- Páginas compartidas
    - Reducir el número de faltas de página
- Páginas especiales
    - Algunos marcos pueden estar bloqueados

Algoritmos de sustitución:

- Podemos tener las siguientes combinaciones:
    - A.fija y S.local
    - A.variable y S.local
    - A.variable y S.global
- Distintos algoritmos
    - Óptimo
        - Sustituye las páginas que no se van a referenciar o que se referenciarán más tarde
    - FIFO
        - Sustituye la página más antigua
    - LRU
        - Sustituye la página que fue objeto de la referencia más antigua
    - Algoritmo de reloj
        - Cada página tiene asociado un bit de referencia R (=1 por hardware)

- Los marcos de página se representan por una lista circular y un puntero a la página visitada hace más tiempo
- Selección de una página
    - 1. Consultar marco actual
    - 2. ¿R=0?
        - No, pongo R=0 -> ir al siguiente marco y volver al paso 1
        - Sí, seleccionar para sustituir e incrementar posición

Conclusión: Influye más la cantidad de MP disponible que el algoritmo de sustitución usado



Influencia del tamaño de página:

- Cuanto más pequeñas las páginas
    - Más grandes las tablas de páginas (se necesitan más páginas)
    - Mayor número de transferencia MP->Disco
    - Reducen la fragmentación interna
- Cuanto más grandes las páginas
    - Grandes cantidades de información que no será usada ocupando la MP
    - Aumentan la fragmentación interna
- Se busca un equilibrio

Propiedad de localidad y relación con un programa en ejecución:
Teoría del conjunto de trabajo y problema de la hiperpaginación:
Cómo gestiona Linux la memoria de un proceso:

->Interfaces para la **asignación** de memoria en páginas:

- struct page *alloc_pages (gfp_t gfp_mask, unsigned int order):
    - Asigna: 2^order páginas físicas contiguas
    - Devuelve: puntero a la struct page de la primera página, y si falla devuelve NULL
- unsigned long __get_free_pages(gfp_t gfp_mask, unsigned int order):
    - Asigna: 2^order páginas físicas contiguas
    - Devuelve: dirección lógica de la primera página.

->Interfaces para la **liberación** de memoria en páginas:

- void __free_pages(struct page *page, unsigned int order)
- void free_pages(unsigned long addr, unsigned int order):
    - Ambas liberan 2^order páginas a partir de la estructura página/de la página que coincide con la dirección lógica.

->Interfaces para la **asignación**/**liberación** de memoria en bytes:

- void * kmalloc(size_t size, gfp_t flags)
- void kfree(const void *ptr):
    - Similares a las de C malloc() y free()