Gestión de Memoria

Vicente Matellán

Área de Arquitectura y Tecnología de Computadores
Escuela de Ingenierías Industrial e Informática
Universidad de León

2^{do} Cuatrimestre - Segundo Curso Grado Ing. Informática

Memoria

- Almacén de información
- a repartir entre el SO y los procesos de usuario

Niveles de memoria

Nivel de procesos : Reparto de memoria entre los procesos
Gestionado por el SO

Nivel de regiones : Reparto de la memoria asignada a un proceso
Gestionado por el SO

Nivel de zonas : Reparto dentro de una región
Gestionado por el Lenguaje

Memoria principal

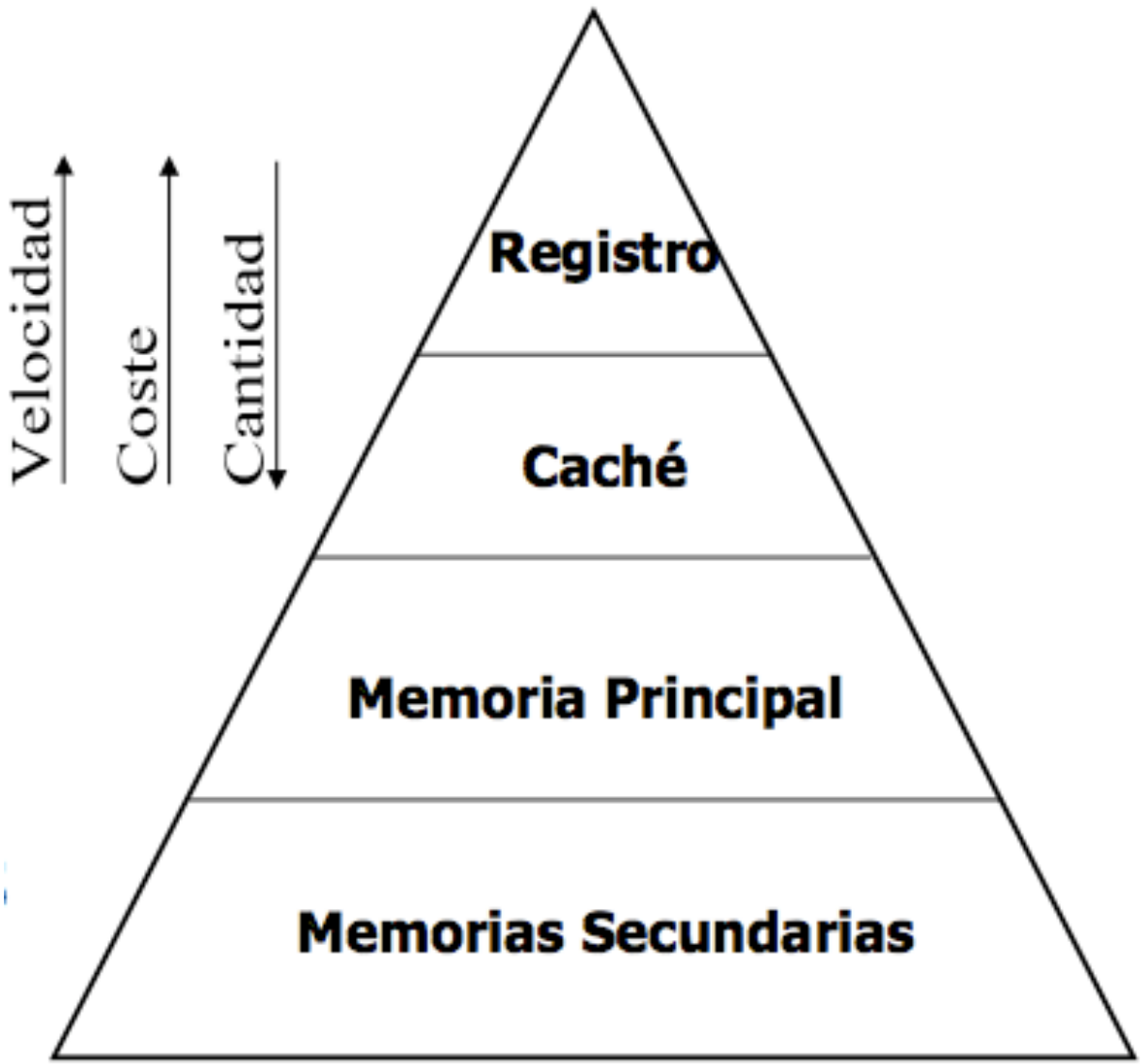
- Formada por celdas
- Cada celda puede almacenar datos o instrucciones
- La unidad básica es la palabra
- Cada palabra tiene una dirección para accederla
- Cada palabra es un byte o múltiplo de este
- Número de direcciones: 2^n (n sería el ancho del bus)
- Mapa de direcciones: conjunto de direcciones posibles

Direcciones Físicas y Lógicas

Lógicas : Usadas en un programa. Relativa a ese programa

Física : Una posición real de la memoria principal

Jerarquía de memoria



Necesidades de los procesos

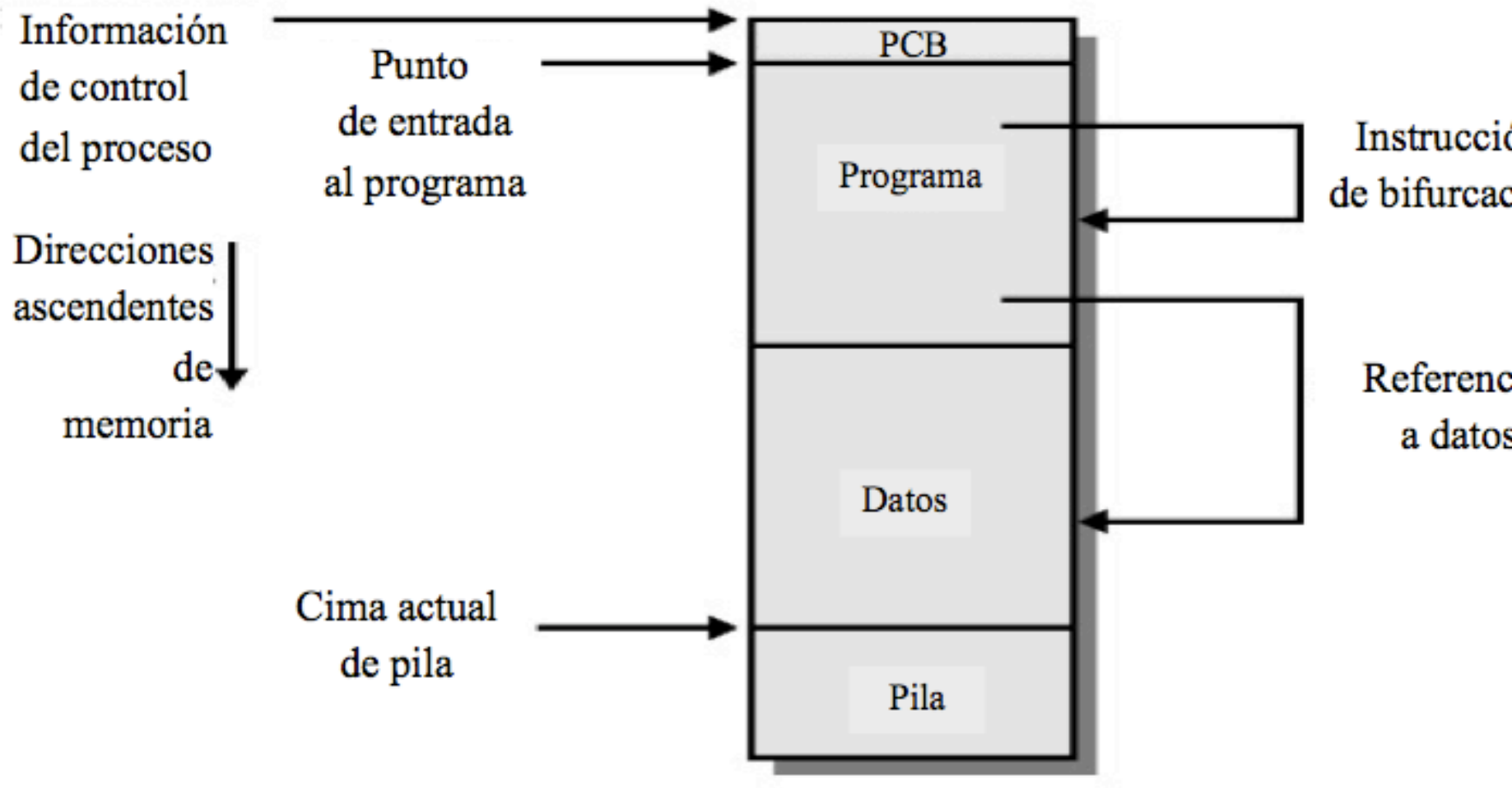
- Espacio lógico propio e independiente
- Espacio protegido
- Compartir memoria con otros
- Soporte para sus regiones
- Soporte de depuración
- Soporte de mapas muy grandes: memoria virtual
- Soporte tipos de datos: constantes, dinámicos, etc
- Persistencia (automática)
- Gestión de módulos: reubicación y carga dinámica

Necesidades del sistema operativo

- Tener espacio lógico propio
- Estar protegido de los programas
- Compartición de memoria: contar referencias
- Soporte regiones del S.O.: tabla procesos, ficheros abiertos...
- Carga dinámica de módulos del S.O.
- Gestión eficiente:
  - Evitar la fragmentación (interna y externa)
  - Evitar las *goteras* (*memory leaks*)
  - Usar memoria virtual
  - Soportar multi-procesadores

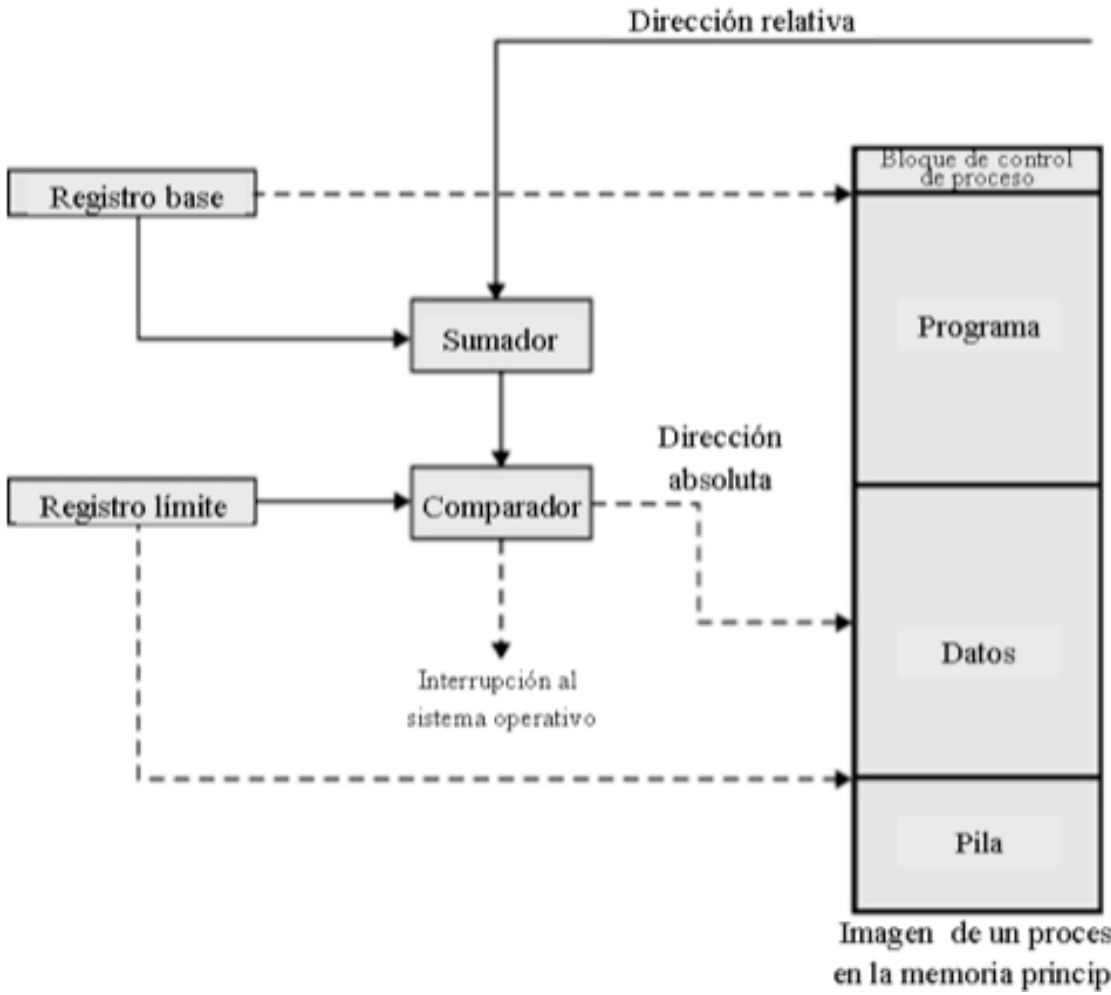
Funciones del sistema de Gestión de Memoria

- Gestión de la compartición
- Organización lógica: regiones
- Organización física: traducción (MMU)
- Protección (hardware)
- Reubicación: estática (t. compilación) y dinámica (ejecución)


Información
de control
del proceso
Punto
de entrada
al programa
Direcciones
ascendentes
de
memoria
PCB
Programa
Datos
Pila
Cima actual
de pila
Instrucci
de bifurcac
Referenc
a datos

Traducción de direcciones lógicas a físicas

- Hw para traducción (ETC)
- Permite reubicación
- El registro base es parte de los registros del procesador
- El registro límite permite aplicar compartición y protección



Datos estáticos

- Existen durante toda la vida
- Tienen posición fija en el mapa de memoria
- Datos de distintos tipos:
  - Globales (programa, módulo) o locales (función) significativo para el compilador
  - Constantes o variables significativo para compilador y SO
  - Con o sin valor inicial
- Lo lógico sería usar direccionamiento absoluto.
- Sin embargo, se suele usar relativo (un registro)...
- ...para tener código independiente de posición PIC (reubicable)

Datos asociados a una función

- Su vida está asociada a la ejecución de una función
- Son las variables locales y parámetros de función
- Se crean al invocar la función y se destruyen al terminar
- Habitualmente se guardan en la pila del proceso (registro de activación)
- El registro de activación guarda también la dirección de retorno
- La dirección en memoria se determina en tiempo de ejecución
- Las llamadas recursivas hacen que puedan existir varios *ejemplares*
- Se usa el registro de pila (para acceder al último registro) luego es PIC

Pila y *heap*

Pila:

- Organizada internamente en zonas independientes
- Cada zona corresponde a un registro de activación
- Gestión sencilla: no se generan huecos (LIFO)

Heap (montón):

- Almacena datos dinámicos
- El espacio se libera cuando ya no es necesario:
  1. A mano: `free`
  2. Recolector de Basura
- La dirección sólo se conoce en ejecución
- Direccionamiento indirecto
- En general lo gestiona el lenguaje de programación
- Sólo se hacen llamadas al SO para hacerlo crecer

Ciclo de vida de los programas

Compilación : procesado de los ficheros fuente y generación de regiones

- Código y constantes
- Variables con valor inicial
- Variables sin valor inicial
- Información de reubicación (para el montador)
- Tabla de símbolos (externos y exportados)
- Depuración

Montaje (enlazado) : agrupación de los ficheros fuente

- Reubicación de módulos
- Generación del fichero ejecutable

Carga y ejecución : carga en memoria y ejecución

No hablamos de bibliotecas: dinámicas, compartidas...

Mecanismos de gestión de memoria

Con carga completa del proceso en memoria:

Asignación Contigua (obsoleto)

- Particiones fijas
- Particiones dinámicas

Asignación No Contigua

- Segmentación (obsoleto)
- Paginación
- Sistemas Combinados

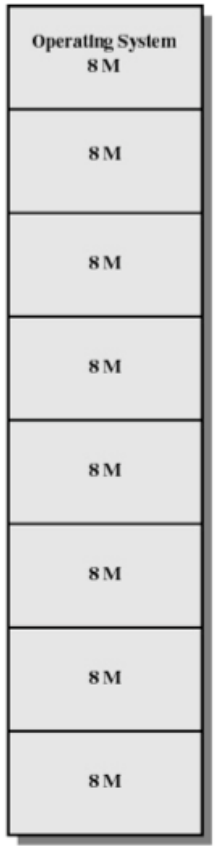
Sin requisito de carga completa:

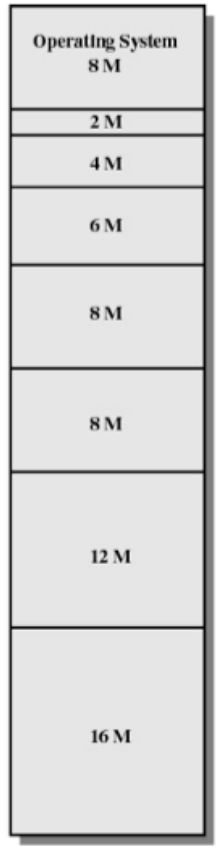
Asignación No Contigua

- Memoria virtual

Asignación Contigua: Particiones fijas

- Asigna a cada proceso una zona fija
- Fragmentación interna
- Dos variantes:
  1. Mismo tamaño
     - grandes → desperdicio
     - pequeñas → no caben
  2. Distintos tamaños:
     - Cola única: el proceso se coloca en la más pequeña de las libres
     - Colas múltiples: partición más pequeña donde quepa





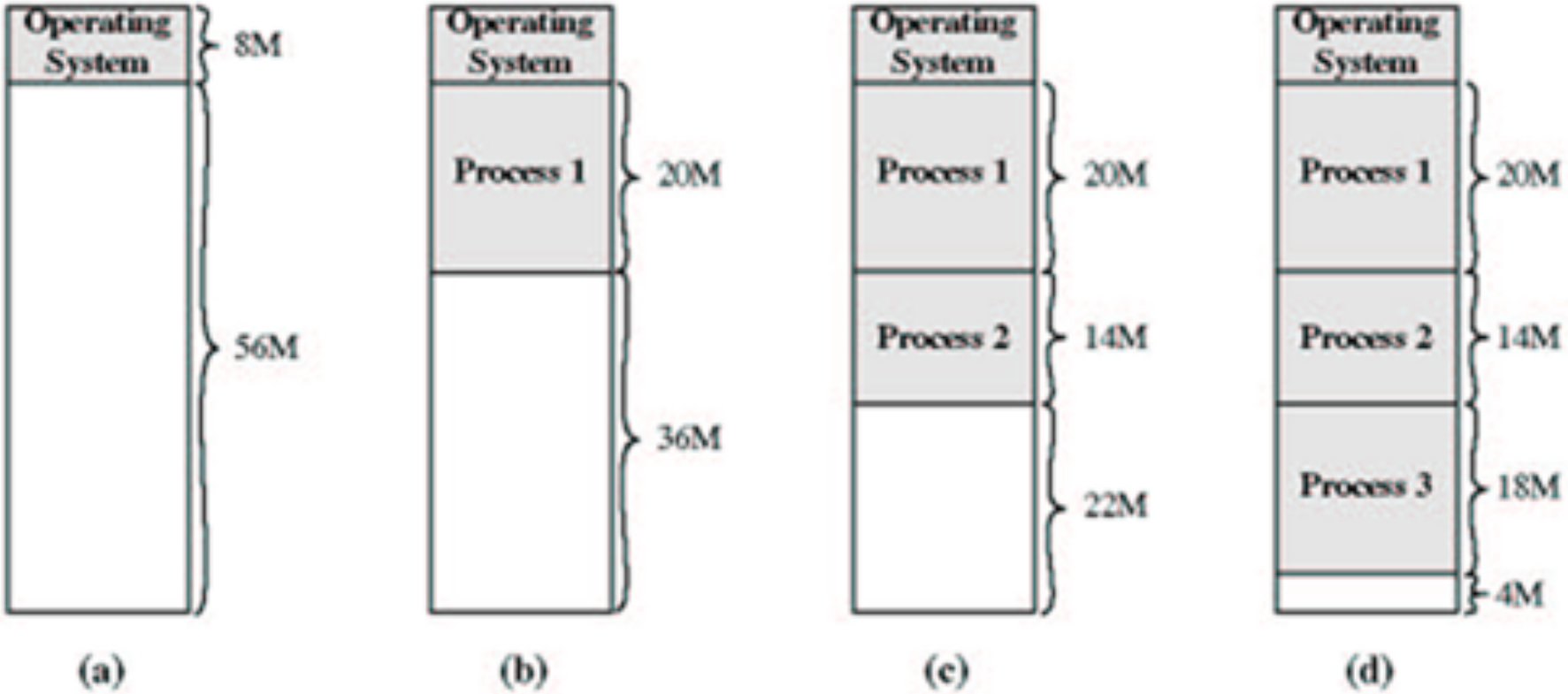
Asignación Contigua: Particiones fijas

- Grado de multiprogramación $\rightarrow$ número particiones
- Tabla de descripción de particiones:

Nº DE PARTICIÓN	BASE	TAMAÑO	ESTADO
0	0 K	100 K	ASIGNADA
1	100 K	300 K	LIBRE
2	400 K	100 K	ASIGNADA
3	500 K	250 K	ASIGNADA
4	750 K	150 K	ASIGNADA
5	900 K	100 K	LIBRE

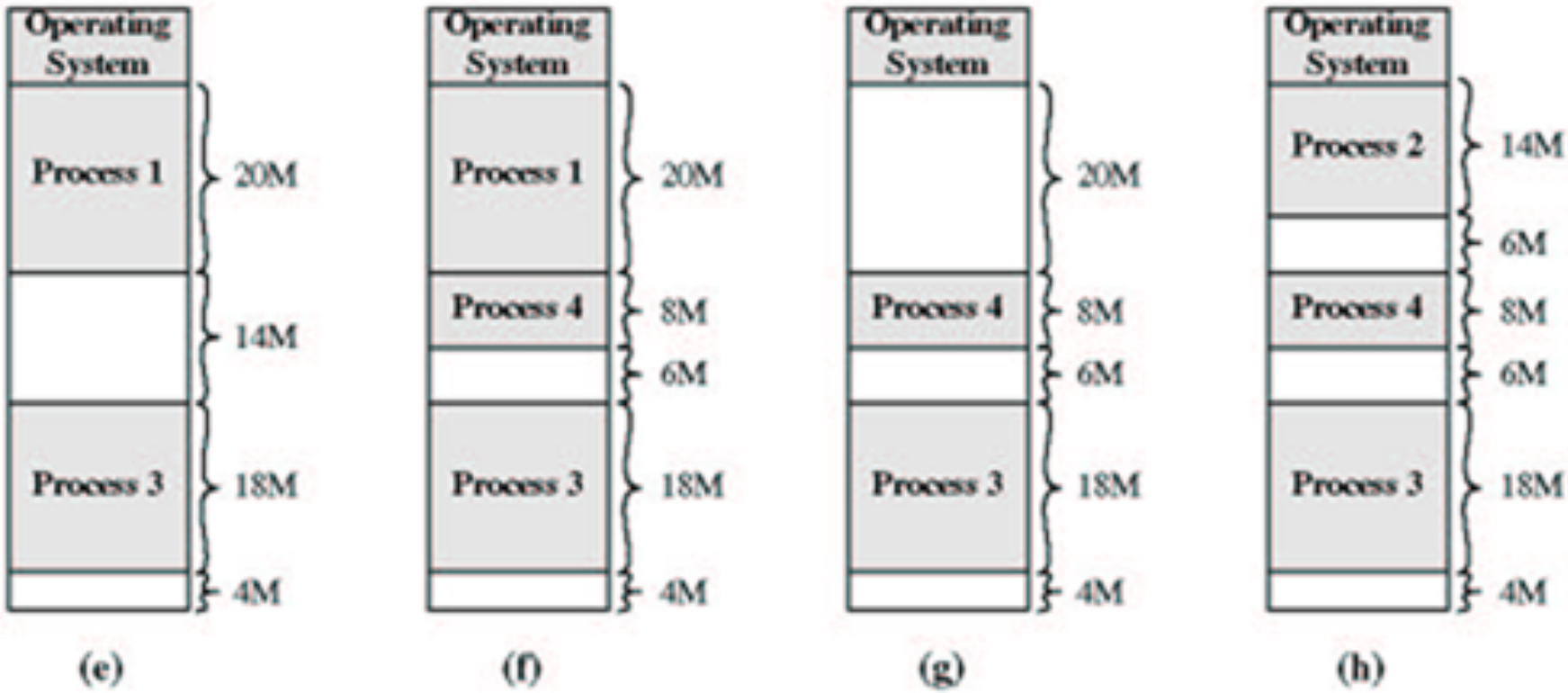
Asignación Contigua: Particiones dinámicas

- La partición se realiza al ejecutar el proceso:



Asignación Contigua: Particiones dinámicas - Fragmentación

- Genera fragmentación externa:



Asignación Contigua: Particiones dinámicas - Solución Fragmentación

- Usar compactación
- La compactación exige tiempo de CPU
- La compactación exige capacidad de reubicación
- Diferentes algoritmos de ubicación:

  Mejor ajuste : el hueco más pequeño que satisfaga la petición (pesada: comprobar todos u ordenar y genera huecos pequeños)

  Peor ajuste : el hueco más grande (mala)

  Primero que ajuste : suele ser la mejor: rápida

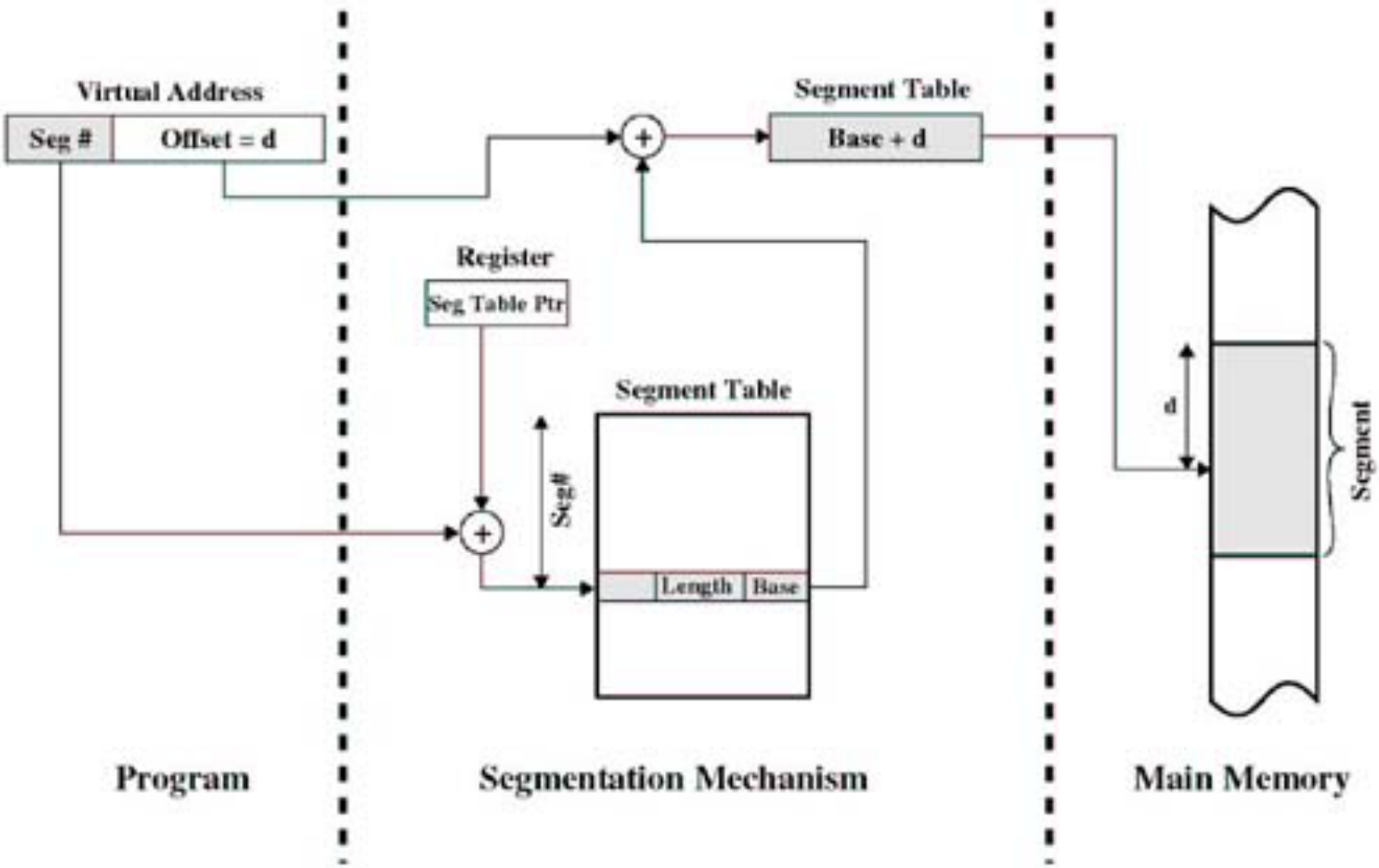
  Siguiente ajuste : variante de la anterior

Segmentación

- El proceso se divide en fragmentos del distinto tamaño llamados segmentos
- La memoria se divide dinámicamente en fragmentos del tamaño adecuado para cada segmento
- Un proceso puede ocupar varios segmentos
- Algoritmos de ubicación para decidir dónde se sitúa un segmento en memoria
- Tabla de Segmentos: indica la posición del segmento y el tamaño

Asignación no contigua: Segmentación - Traducción

- Direcciones lógicas: segmento y posición dentro del segmento

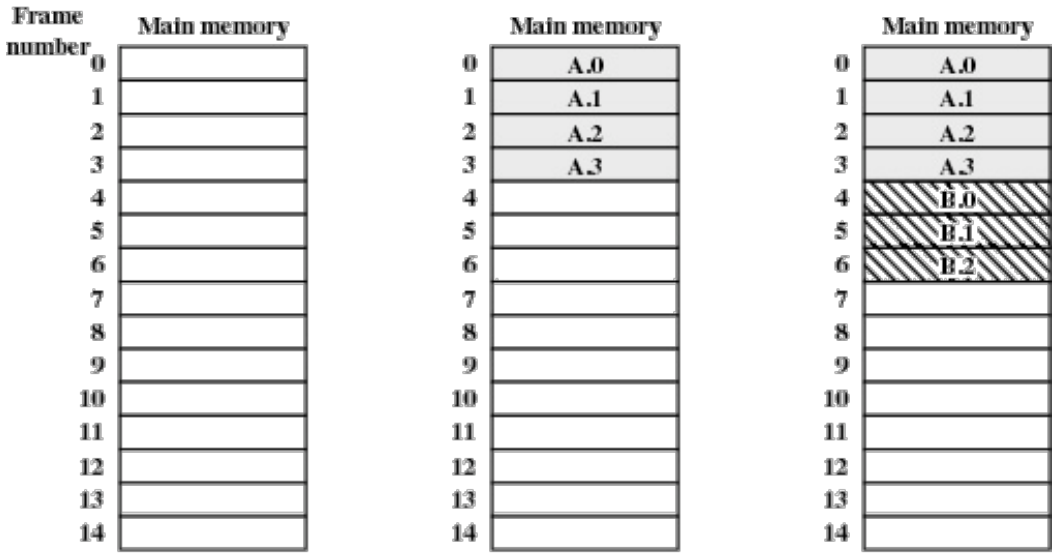


Paginación

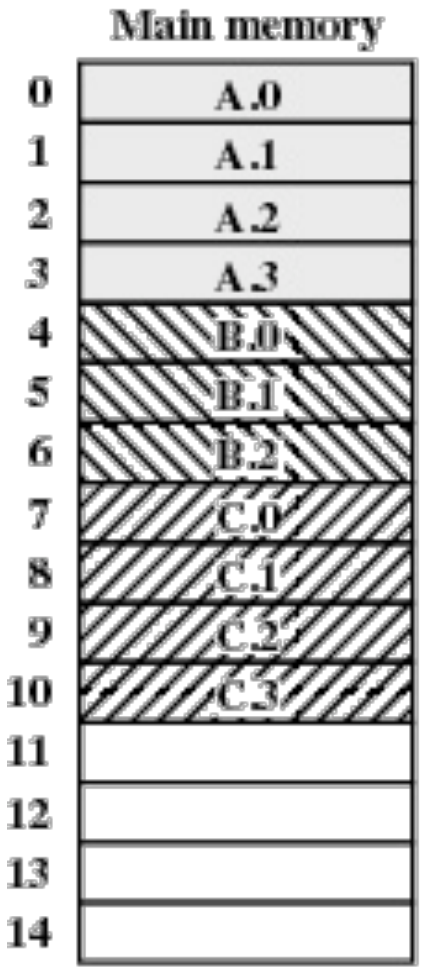
- La memoria se divide en trozos del mismo tamaño: Marcos de página
- El tamaño de los marcos es hardware: típico 4Kb
- Los procesos se dividen en porciones de ese mismo tamaño: páginas
- No presenta fragmentación externa
- Sólo presenta fragmentación interna en el último marco asigando a un proceso

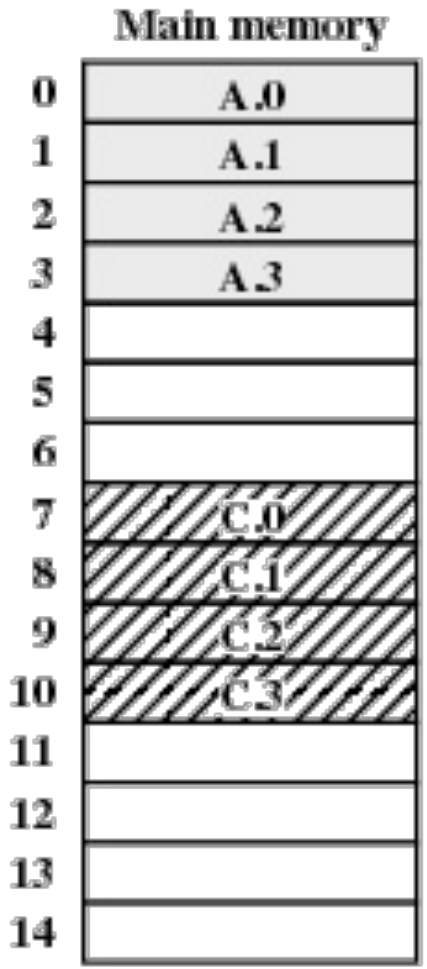
Paginación - Diferencias con particiones fijas

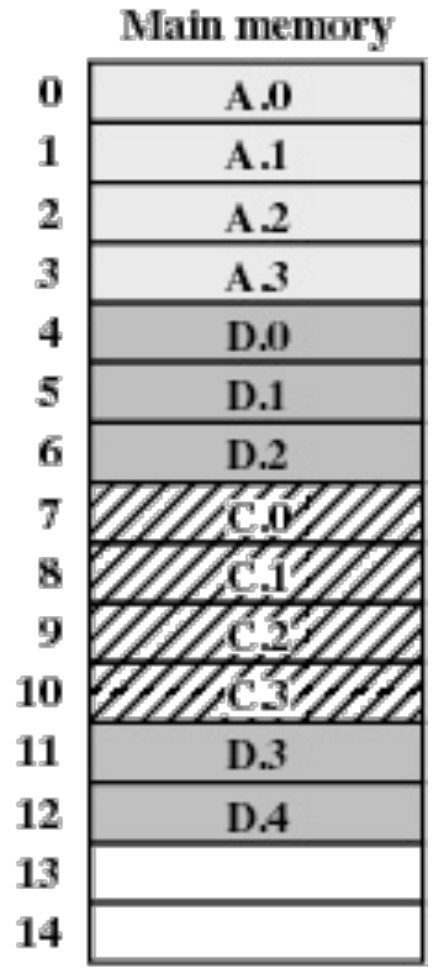
- Los marcos son de menor tamaño que las particiones
- En una partición se carga un programa completo
- En un marco solo una porción de programa (página)
- Un programa solo puede ocupar una partición, en paginación puede ocupar varios marcos
- Las instrucciones ocupan posiciones consecutivas en particiones, en paginación quizás



Paginación - Ejemplo



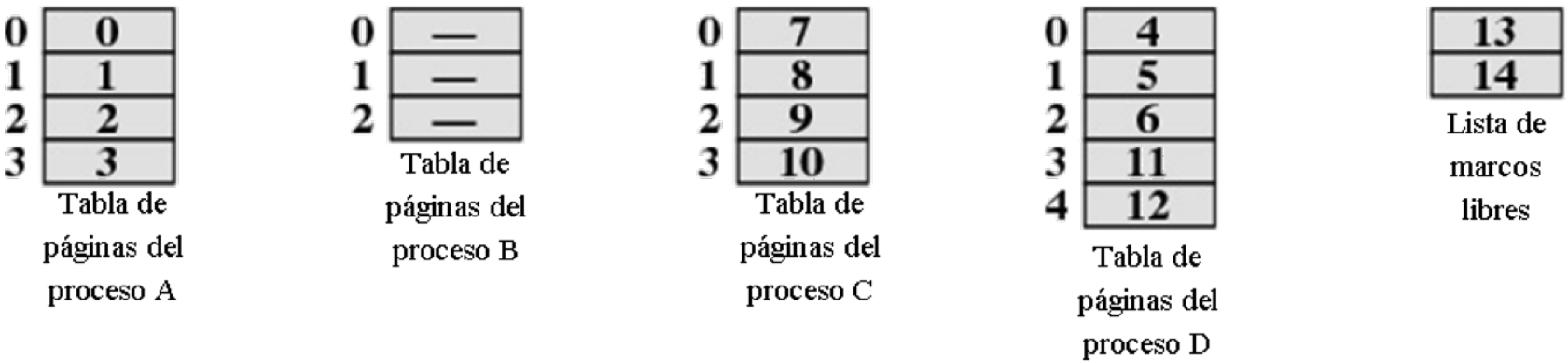




Paginación - Ejemplo

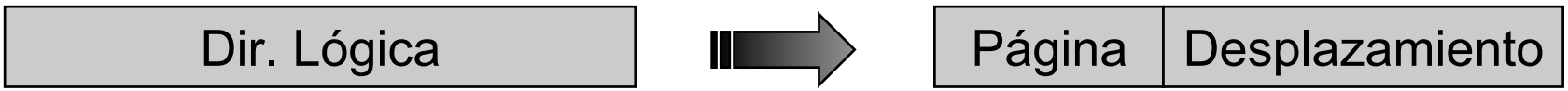
Estructuras de control

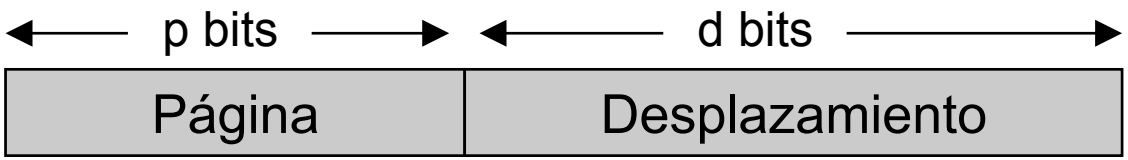
- Una TDP por proceso
- Una tabla de marcos libres



Traducción direcciones

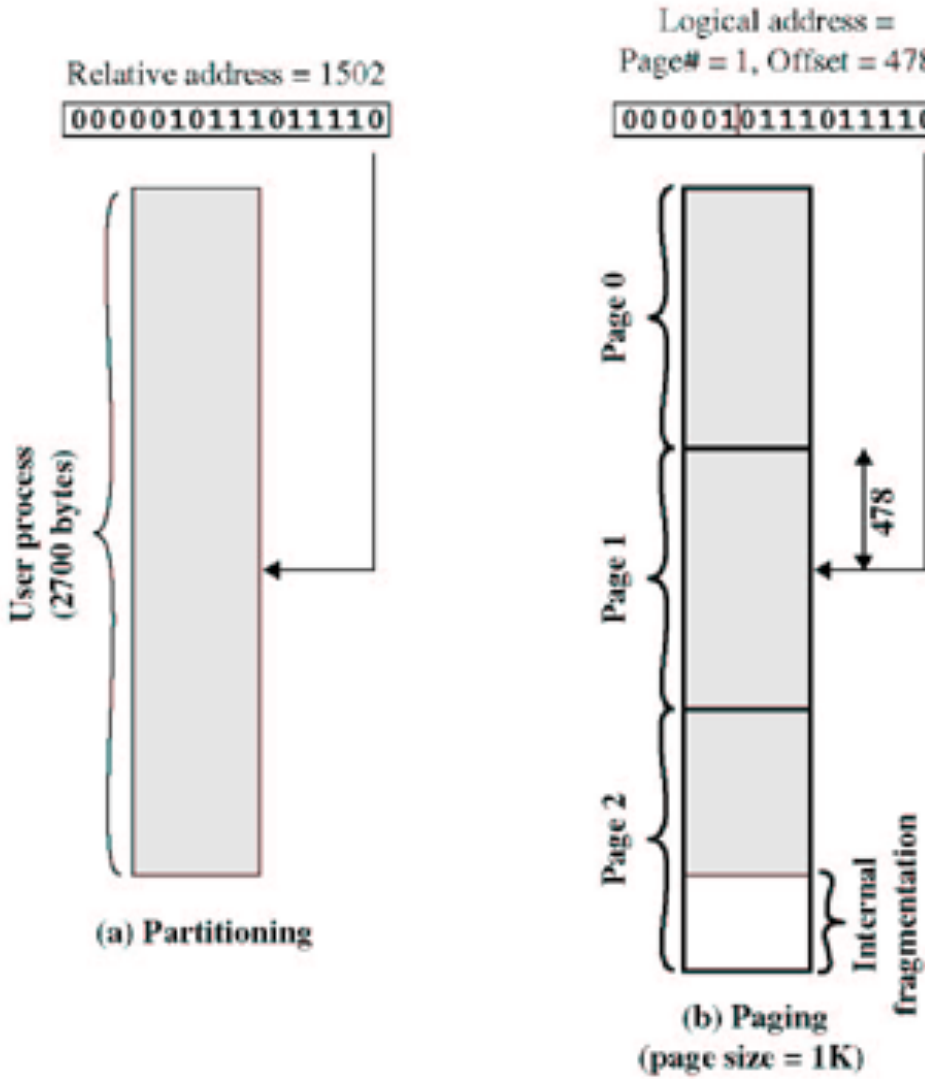
- Dirección lógica $\rightarrow$ página y desplazamiento
- Número máximo de páginas $= 2^p$
- Tamaño de la página $= 2^d$



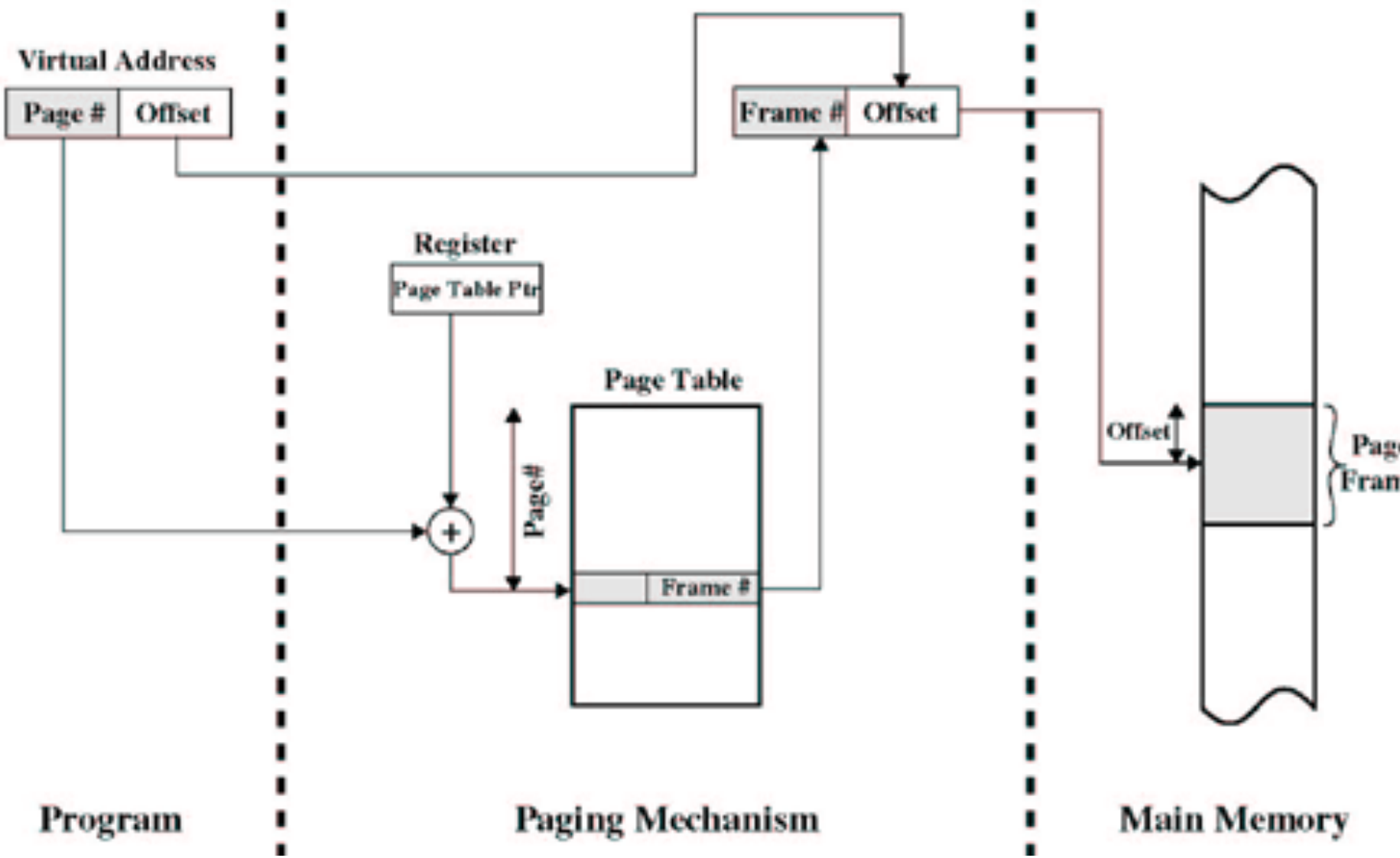


Paginación - Traducción de direcciones - Ejemplo

- Tamaño página: 1 Kb
- Dirección relativa: 1502
- Dirección lógica: (1,478)
- El marco se consulta en la TDP
- Dirección física:
  (marco * tamaño) + desplazamiento



Paginación - Dir Física



Paginación - Problemas

Ventajas

- Permite mapas de memoria muy grandes para los procesos
- No presenta fragmentación

Inconvenientes

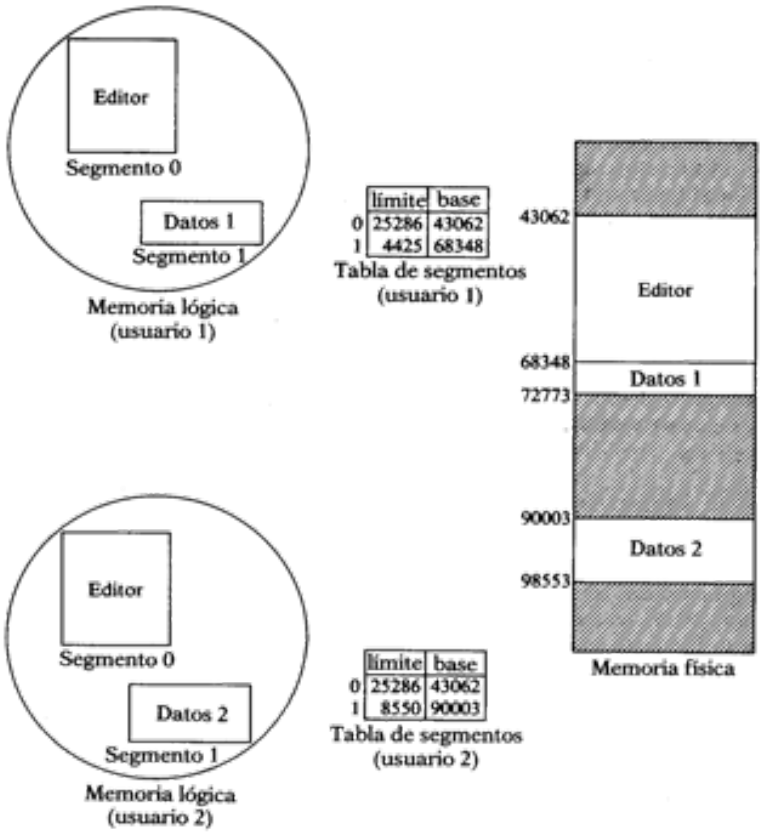
- Gasto de memoria: TDP muy grandes
- Ejemplo: procesador con 32 bits, tamaño página 4Kb
  Si cada proceso puede tener todo el espacio lógico.
  Mapa memoria: 2^{32}
  Tamaño TDP: $2^{20} * 4bytes = 4Mb$
- Además estarían vacías
- Ejemplo: anterior con un proceso con 16Mb
  Sólo se usarían: 4096 entradas
  Varios millones vacias
- Mucho peor con 64 bits
- Solución: tablas multinivel (super-páginas)
- Eficiencia: 2 accesos a memoria
- Solución: TLB

Paginación - TLB

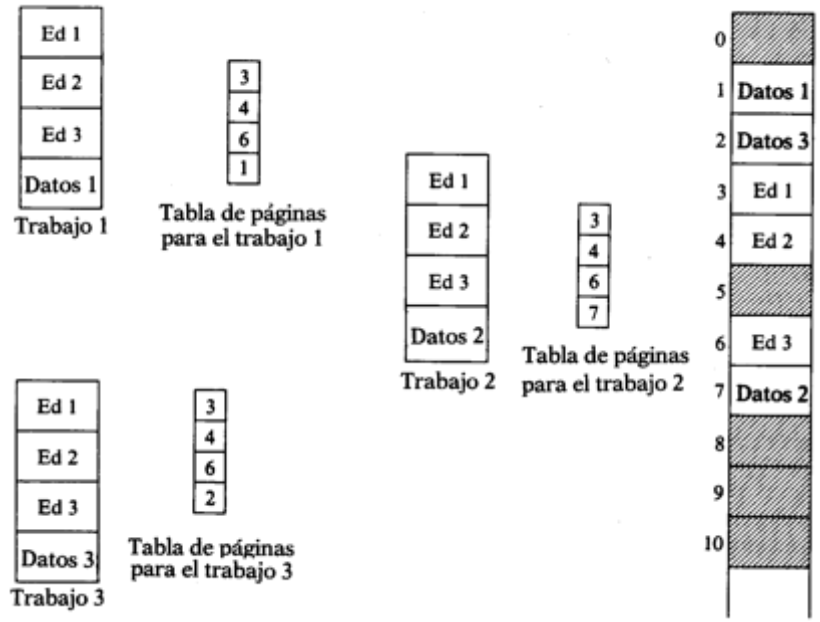
- La MMU incluye una memoria asociativa (tipo *cache*): TLB
- TLB: *Translation Lookaside Buffer (TLB)*
- Tipos:
  - Con identificación de procesos: cambiar tablas en el cambio de contexto
  - Sin identificación de procesos
- Alternativa a TLB: RBTP (Registro base de tablas de página)

Compartición en Segmentación y Paginación

Segmentación:

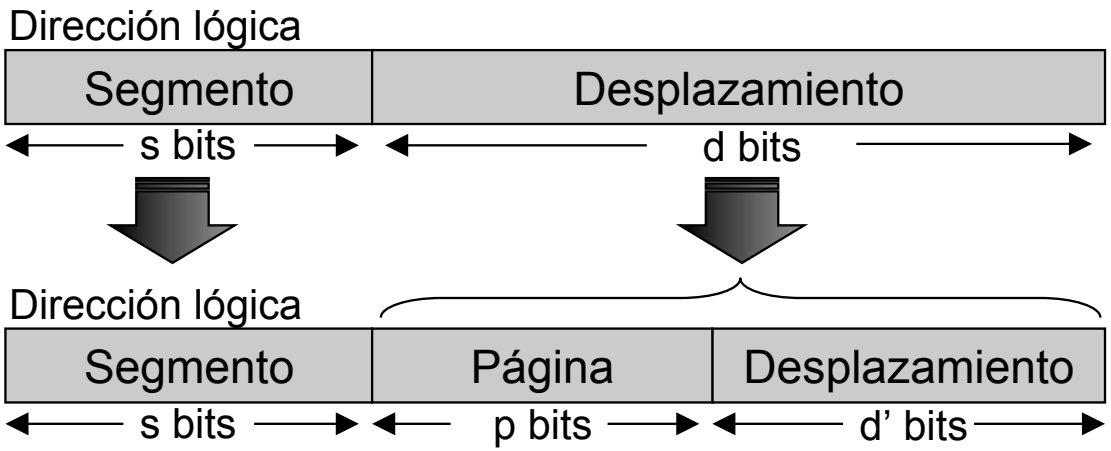


Paginación:



Segmentación Paginada

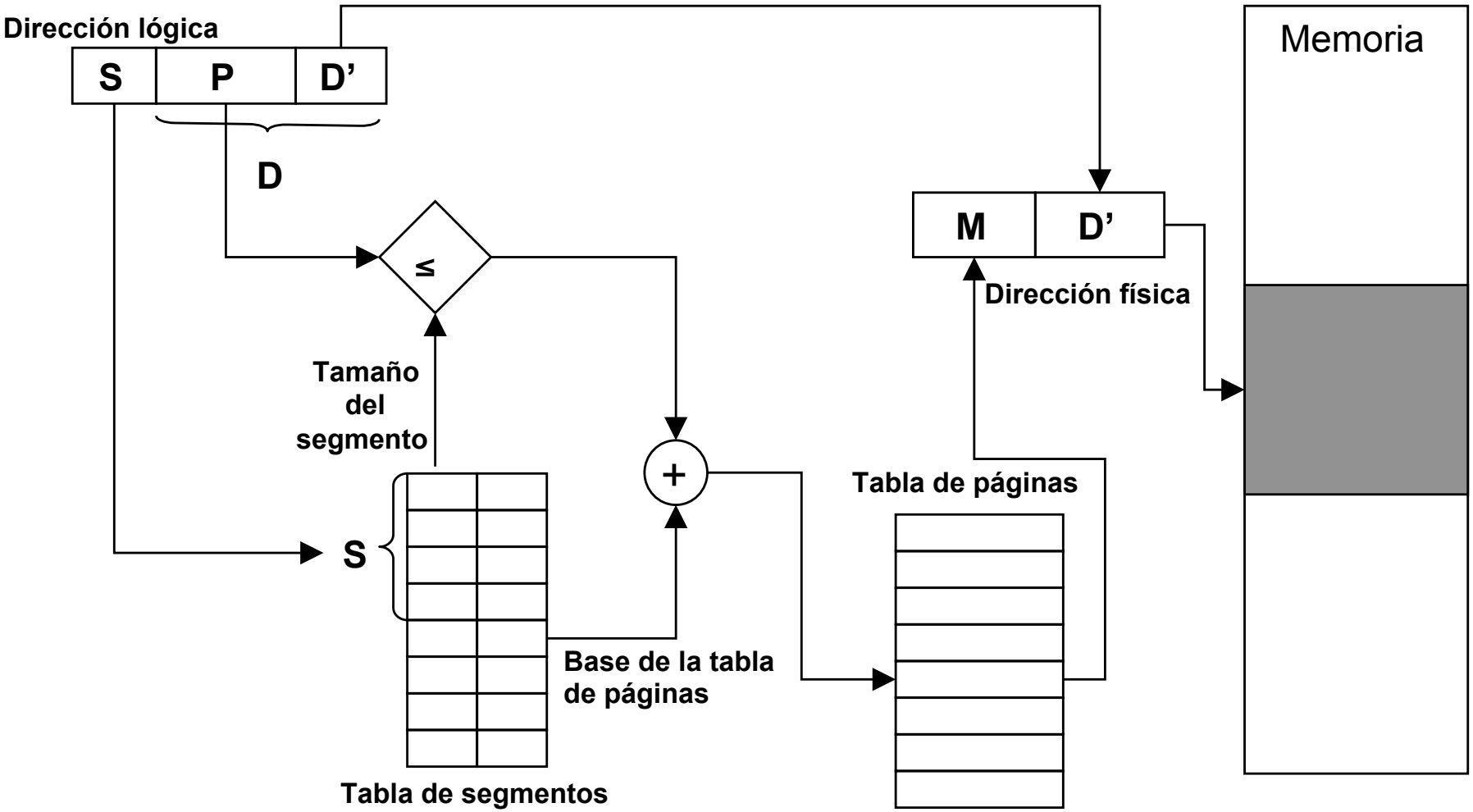
- Intenta eliminar la fragmentación externa de la segmentación
- La memoria principal se divide en marcos de página
- Para el programador se utilizan segmentos, para el sistema páginas
- Los segmentos se dividen en páginas del tamaño de los marcos
- Las páginas de un segmento no tienen que estar contiguas



Segmentación Paginada - Mecanismos

- Se usa una tabla de descripción de páginas (TDP) por cada segmento de un proceso, con la referencia del marco en el que se encuentra una página
- Una tabla de descripción de segmentos (TDS) por proceso, con la ubicación de la tabla de páginas de cada segmento y su número de páginas
- En principio una traducción necesitará de tres accesos a memoria en lugar de dos

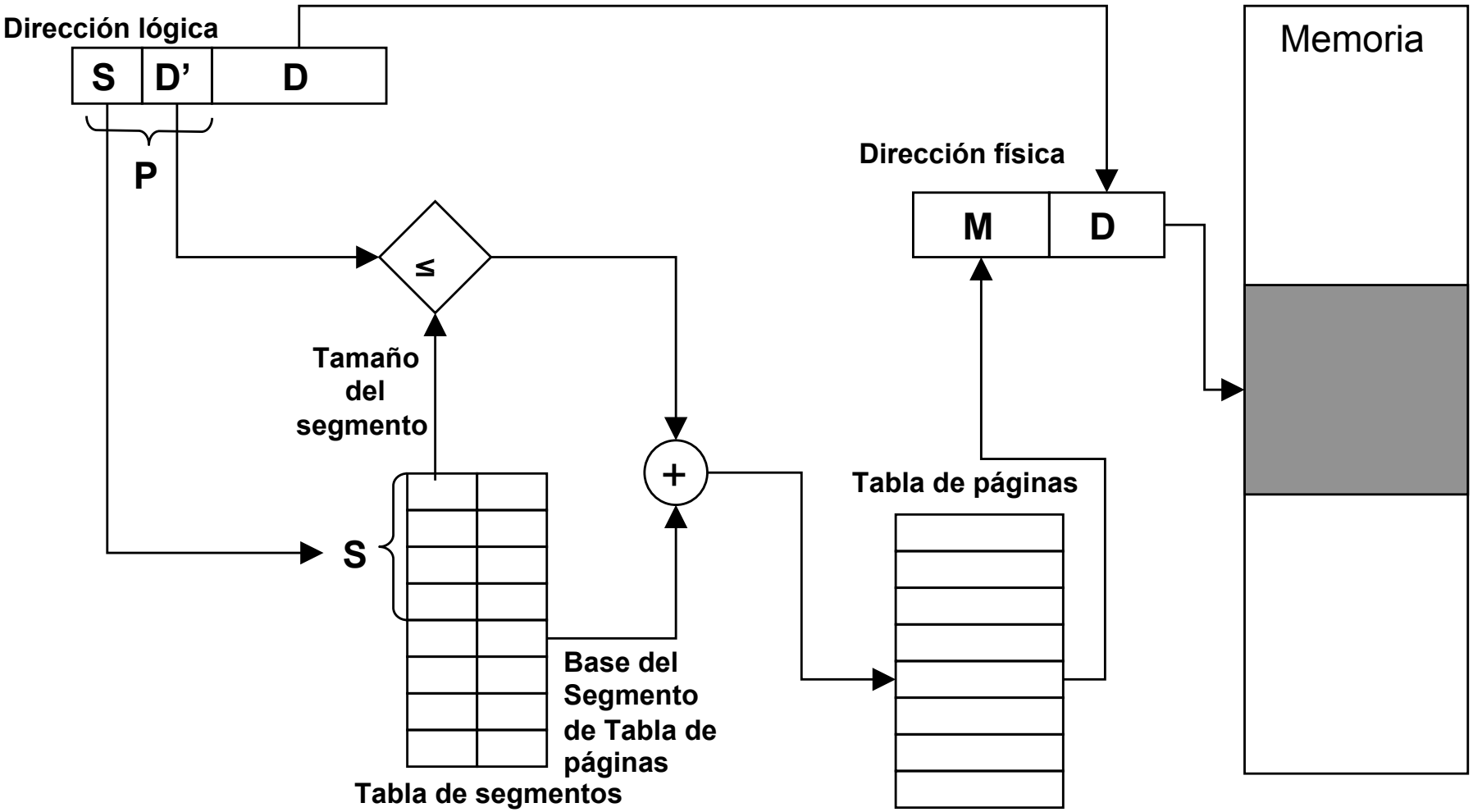
Segmentación Paginada - Traducción



Paginación segmentada

- Sistemas con tablas de páginas muy grandes
- Divide la tabla de páginas en segmentos y las almacena en posiciones no contiguas
- Los procesos generan direcciones relativas
- El S.O. las convierte en direcciones lógicas (página, desplazamiento)
- El número de página representa un número de segmento de la tabla de páginas y la página dentro de ese segmento

Paginación Segmentada - Traducción



Memoria Virtual

- No es necesario que todos los fragmentos estén en memoria
- Hablaremos de sólo de páginas no de fragmentos
- Las páginas pueden tener código o datos
- Cuando se quiere acceder a una página que no está en memoria: Fallo de página:
  - El SO tiene que traer la página a memoria desde el disco (E/S)
  - El proceso queda suspendido
  - Se puede pasar a ejecutar a otro que esté listo
- Ventajas
  - Aumenta el número de procesos en memoria
  - Permite procesos más grandes que la memoria

Tamaño de las páginas

	Ventaja	Inconveniente
Página Grande	Menos páginas por proceso Tablas más pequeñas	Más tiempo de transferencia entre disco y memoria
Página Pequeña	Más Multi-programación Menos fragmentación interna	Mayor número de transferencias entre disco y memoria

Bloques del mismo tamaño → Memoria virtual paginada
Bloques de distinto tamaño → Memoria virtual segmentada

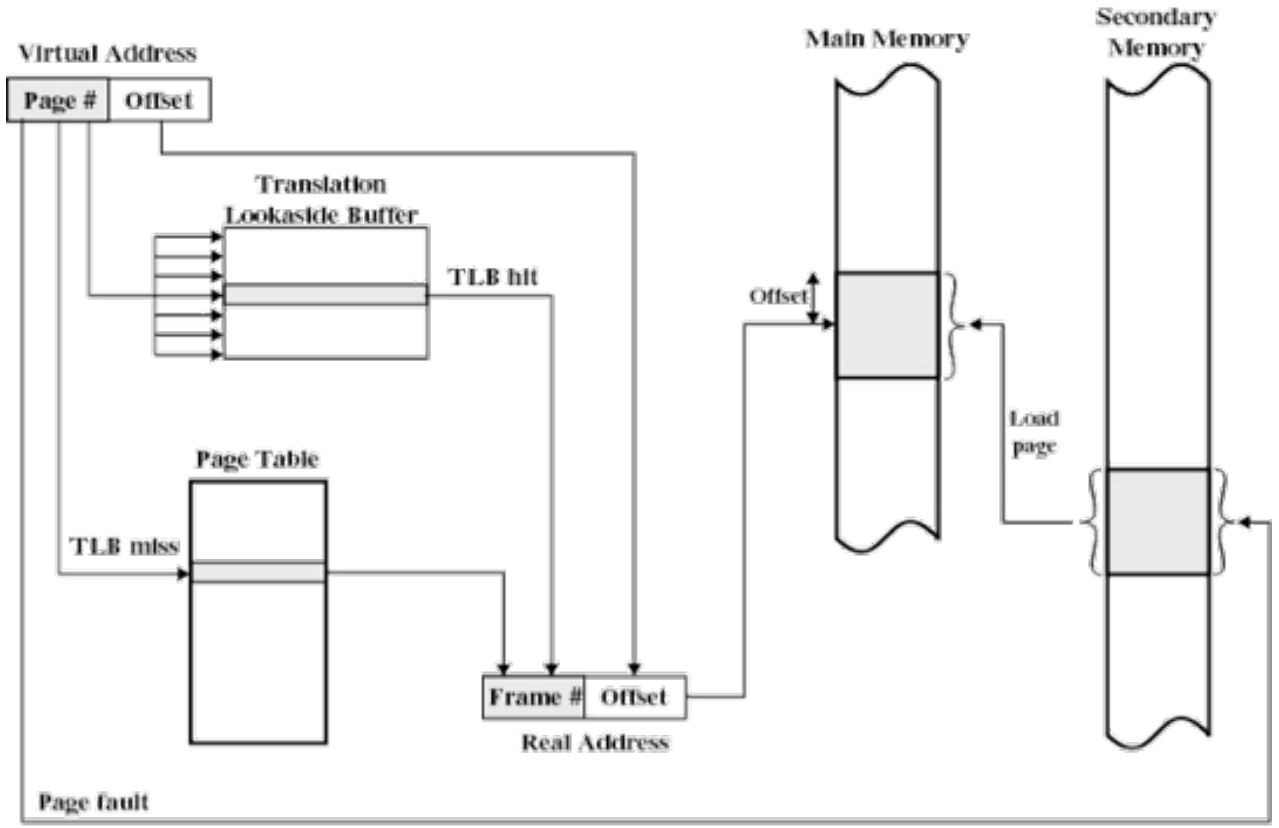
- Conjunto Residente: Porción del proceso en memoria
- Principio de Cercanía: Las referencias tienden a agruparse →
- durante cierto tiempo no habrá fallos de página
- Hiperpaginación: el sistema pasa mucho tiempo resolviendo fallos de página
- Sólución a la hiperpaginación: *adivinar* las páginas que va a necesitar → TLB

Memoria Virtual - TLB

- Usar una TLB
- Permite acceder rápidamente a las últimas entradas utilizadas de la tabla de páginas o segmentos
- Utiliza registros asociativos (rápidos y caros)
- Permiten buscar un valor en todos ellos a la vez

Memoria Virtual - Traducción

La traducción comienza buscando en la TLB, después busca en la tabla de páginas y por último produce un fallo de página y carga la página del disco



Paginación - Implementación

Paginación

- A cada entrada de la TDP se añaden dos bits (P y M):
- El bit de presencia (P): la página está en memoria principal
- El bit de modificación (M): si la página se ha modificado no necesita escribirse en disco cuando se expulsa de memoria

Segmentación

- Cada entrada de la tabla de segmentosincluye también P y M

Segmentación paginada

- Los bits P y M pertenecen a cada página de un segmento luego se encontrarán en la TDP de cada segmento

Memoria Virtual - Política de lecturas

Deciden cuándo debe entrar una página en memoria:

Paginación por demanda :

- La página se carga cuando el proceso lo solicita

Paginación previa :

- Carga un conjunto de páginas del mismo bloque
- Lectura en disco $\rightarrow$ leer un bloque (varias páginas)

Normalmente se combinan ambos :

- Previa al principio de la ejecución o cuando hay muchos fallos de página
- Por demanda cuando ya no hay muchos fallos de página

Memoria Virtual - Política de ubicación

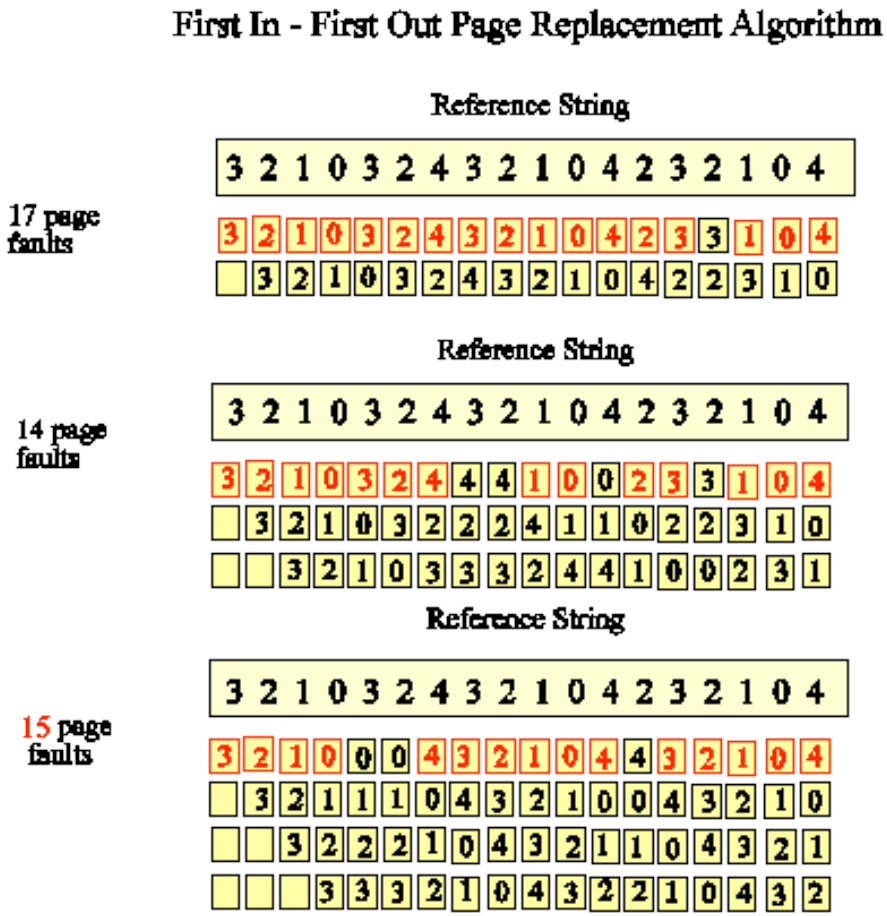
- Estudian en qué porción libre se colocará una página o segmento
- Segmentación:
  - Primer ajuste, siguiente ajuste, mejor ajuste y peor ajuste
  - Influye en el índice de fragmentación externa
- Paginación o segmentación paginada:
  - No es relevante
  - Dará igual que se cargue en un marco libre u otro (son todos del mismo tamaño)
  - No influye en la fragmentación interna

Memoria Virtual - Política de reemplazo

- Deciden qué página sale de memoria cuando se necesita espacio para cargar otra página
- Deben tener en cuenta que algunas páginas están bloqueadas y no pueden salir de memoria
- Dos tipos de algoritmos:
  1. Alcance local: la página a reemplazar pertenece al proceso que recibe la nueva página cargada.Normalmente los procesos tienen siempre el mismo número de páginas
  2. Alcance global: la página a reemplazar puede pertenecer a cualquier proceso. Los procesos pueden tener más o menos páginas en cada instante

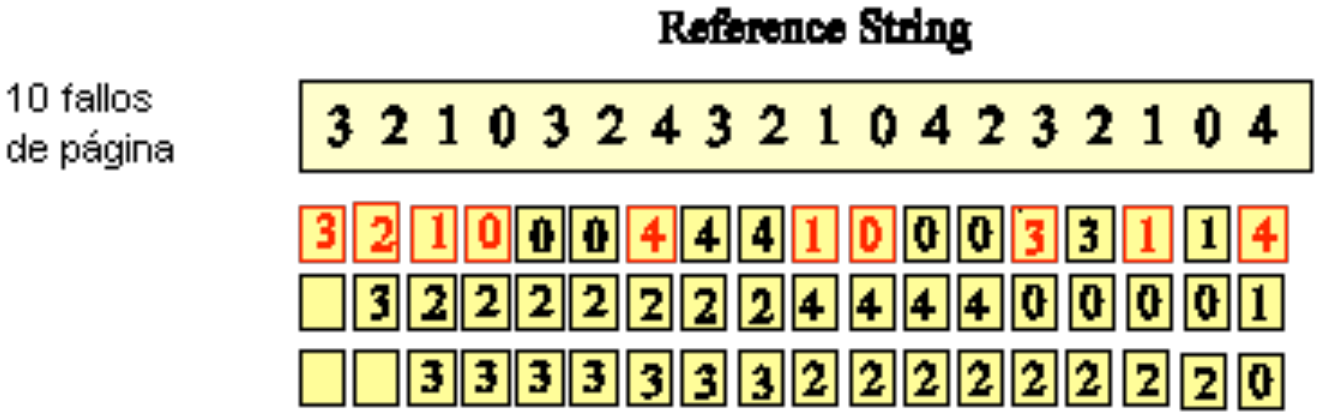
Política de reemplazo - FIFO

- Reemplaza la página que lleva más tiempo en memoria
- Anomalía de Belady: al utilizar más marcos se generan más fallos de página



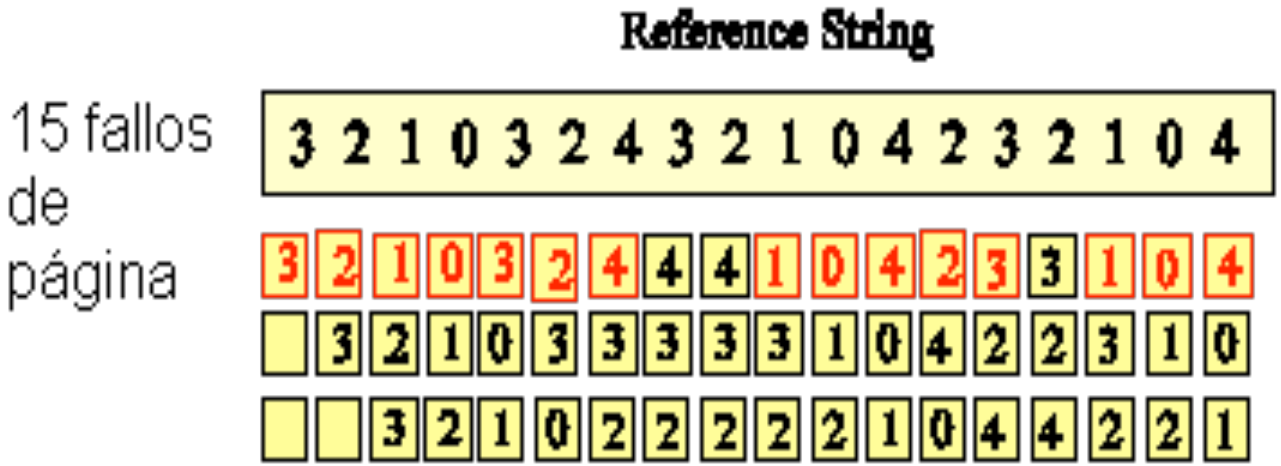
Política de reemplazo - Algoritmo óptimo

- Reemplaza la página que tardará más tiempo en usarse
- Implica conocer información acerca del futuro
- Sólo es útil con carácter teórico



Política de reemplazo - LRU

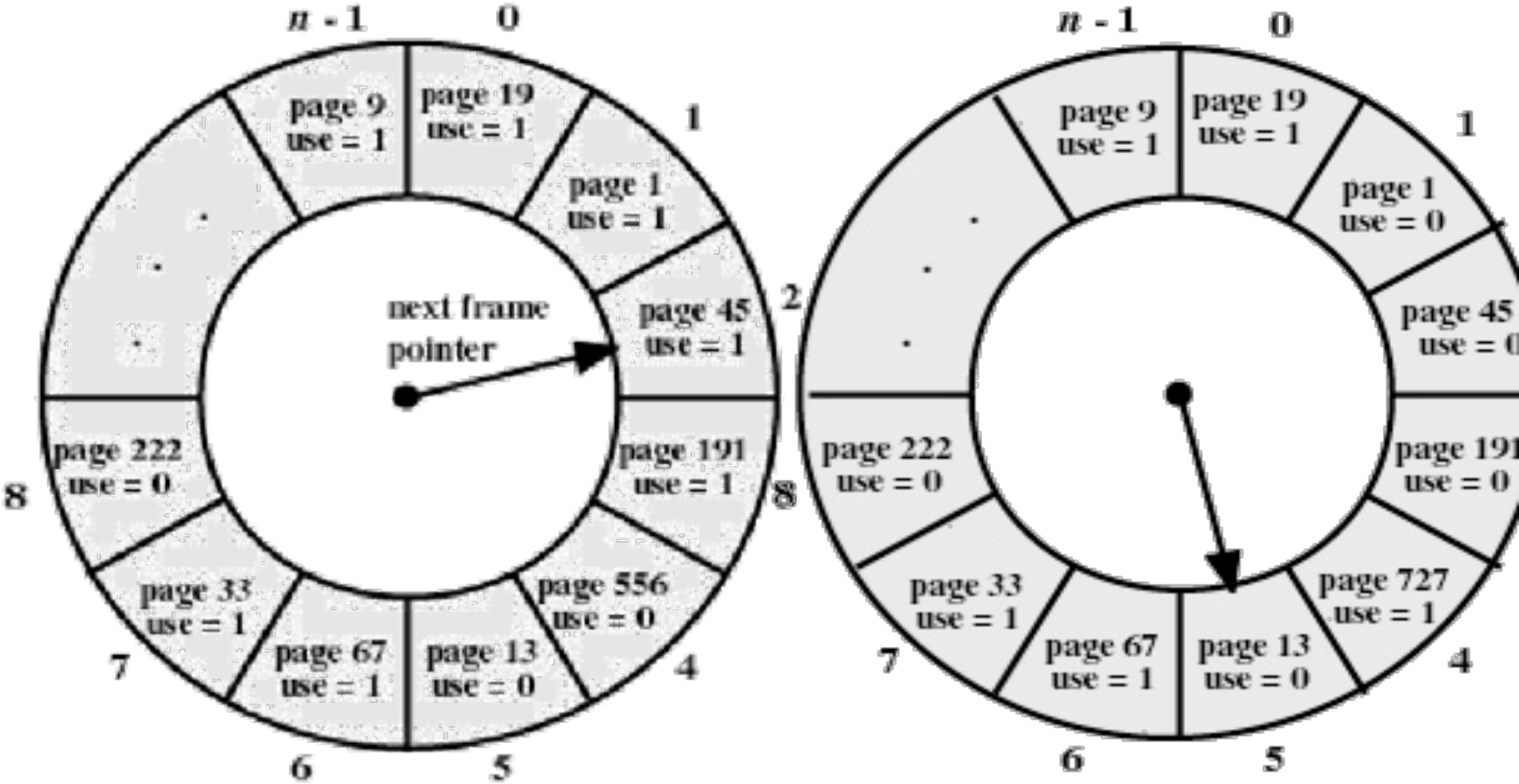
- Las usadas en el pasado cercano serán las usadas en el futuro cercano (principio de cercanía)
- Costoso de llevar a la práctica porque necesita anotar el instante de todas las páginas



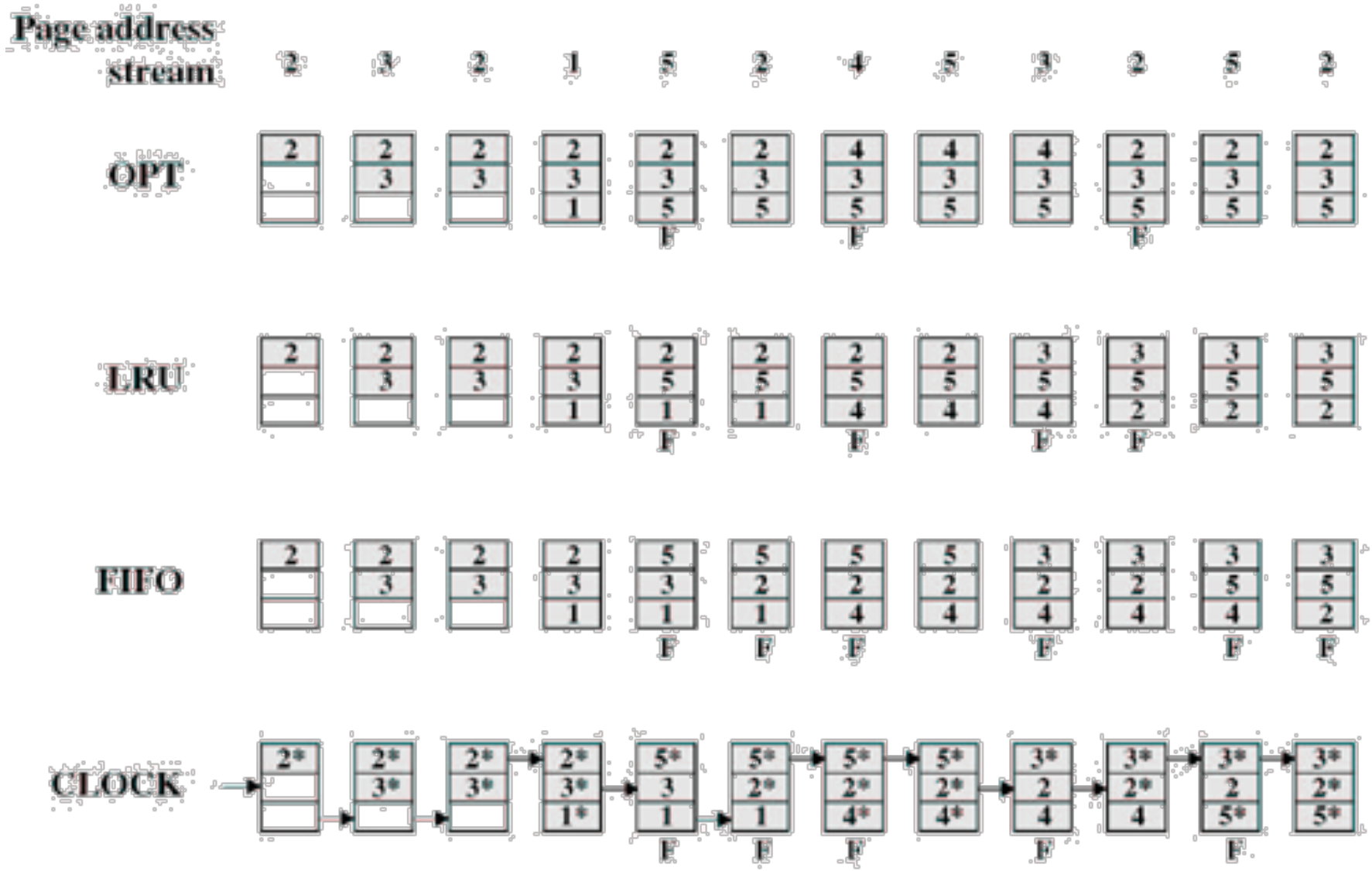
Política de reemplazo - Algoritmo del reloj

- Un bit de uso indica si una página se ha usado
- Al cargarse una página, su bit de uso se pone a 0
- El bit de uso se pone a 1 cuando se referencia una página que ya está en memoria
- Para el reemplazo, se recorre la lista de páginas de forma circular a partir de la última posición de reemplazo
- Al pasar por una página con bit de uso a 1, se pone a 0
- Si se encuentra una página con el bit de uso a 0, se reemplaza esa página
- El puntero queda señalando la siguiente página

Política de reemplazo - Algoritmo del reloj



Política de reemplazo - Comparación



Política de reemplazo - Almacenamiento intermedio

- Crea dos listas de páginas seleccionadas para ser suspendidas:
  1. Lista de páginas libres: páginas no modificadas a suspender
  2. Lista de páginas modificadas: páginas modificadas a suspender
- Cuando se produce un fallo de página
  1. La página a suspender se coloca en una de estas dos listas
  2. Se consulta si la página a cargar en memoria se encuentra en una de estas listas
     - Si está en alguna lista ya estaba en memoria
     - Si no está se tomará un marco de la lista de páginas libres. Si no hay marcos (páginas) en la lista de páginas libres, se descarga al disco una de la lista de páginas modificadas y se utiliza el marco
- Las páginas modificadas seleccionadas para ser suspendidas se descargan en el disco en momentos de poca carga o cuando se requiere un marco libre

Gestión del conjunto residente

Definición

- Mantener en memoria las páginas que un proceso necesita
- Asignación variable de marcos a un proceso (alcance local).

Conjunto de Trabajo

- W(t, δ): conjunto de páginas que el proceso ha usado en las últimas δ unidades de "tiempo virtual" a partir del instante t
- Tiempo virtual: cada instrucción ejecutada equivale a una unidad de tiempo virtual
- δ es la **ventana de trabajo**
- El tamaño **óptimo** de δ es imposible de calcular
- Solución: Algoritmos PFF o VSWS

Algoritmo de frecuencia de fallos de página (PFF)

- Se emplea un bit de uso y un umbral de tiempo F
- Se controla el tiempo entre fallos de página
- Si el tiempo entre dos fallos en menor que F:
  - La página se añade al conjunto residente del proceso que la necesita.
  - Se pone a uno su bit de uso.
  - Esto **aumenta** el tamaño del conjunto residente
- Si el tiempo entre dos fallos de página es mayor o igual que F:
  - Se suspenden todas las páginas con el bit de uso a cero
  - se pone a cero el bit de uso del resto.
  - Se asigna un marco para la nueva página.
  - Normalmente esto **reduce** el tamaño del conjunto residente
- PFF no funciona bien cuando el proceso cambia de grupo de páginas

Algoritmo de Conjunto de trabajo muestreado en intervalos variables (VSWS)

- Reemplaza las páginas no usadas en intervalos de tiempo variables
- Emplea el bit de uso y tres parámetros:
  - M: duración mínima del intervalo
  - L: duración máxima del intervalo
  - Q: número de fallos de página permitidos en el intervalo
- Si el tiempo virtual del intervalo llega a L, el intervalo termina
- Si se producen Q fallos de página en el intervalo:
  - Si el tiempo virtual del intervalo es menor que M, continuar con el intervalo
  - Si el tiempo virtual del intervalo es mayor que M, el intervalo termina
- Durante el intervalo, las páginas siempre se añaden al conjunto residente. Eso hace que aumente de tamaño
- El conjunto residente sólo puede disminuir al final del intervalo
- Cuando hay muchos fallos de página, el intervalo se reduce y el conjunto residente se vacía con mayor frecuencia

Políticas de Vaciado

- Deciden cuándo se lleva a memoria secundaria una página que está en memoria principal
- Vaciado por demanda:
  - La página se descarga cuando se reemplaza
- Vaciado previo:
  - Descarga un conjunto de páginas del mismo bloque de disco
- El almacenamiento intermedio de páginas combina ambos:
  - Con las páginas de la lista de páginas modificadas se utiliza vaciado previo y pasan a la lista de páginas libres
  - Cuando se necesita una página modificada solamente, se emplea vaciado por demanda

Suspensión de procesos

- Proceso con la prioridad más baja
- Proceso con más fallos de página
- Último proceso en entrar al sistema
- Proceso con menos páginas en memoria
- Proceso con más páginas en memoria
- Proceso con mayor tiempo de ejecución restante