Sistemas Operativos

Grado en Ingeniería Informática del Software

Tema 4: Gestión de memoria

Miguel Riesco Albizu

Tabla de Contenidos

1. Introducción al tema

¿Qué tareas forman parte de la gestión de memoria?

En los sistemas operativos modernos la gestión de memoria resuelve aspectos como:

1. La **carga** de programas y su ubicación.
2. La **traducción** de direcciones lógicas del programa a direcciones físicas de la memoria (por hardware)
3. Gestionar la posibilidad de presencia simultánea de **más de un programa** en memoria y su protección.
4. La posibilidad de **cargar rutinas en tiempo de ejecución** (rutinas de enlace dinámico).
5. La **compartición** de espacios de memoria por varios programas.
6. La ejecución de **programas que no caben** completos en memoria.
7. La **gestión eficiente** del espacio de memoria libre.

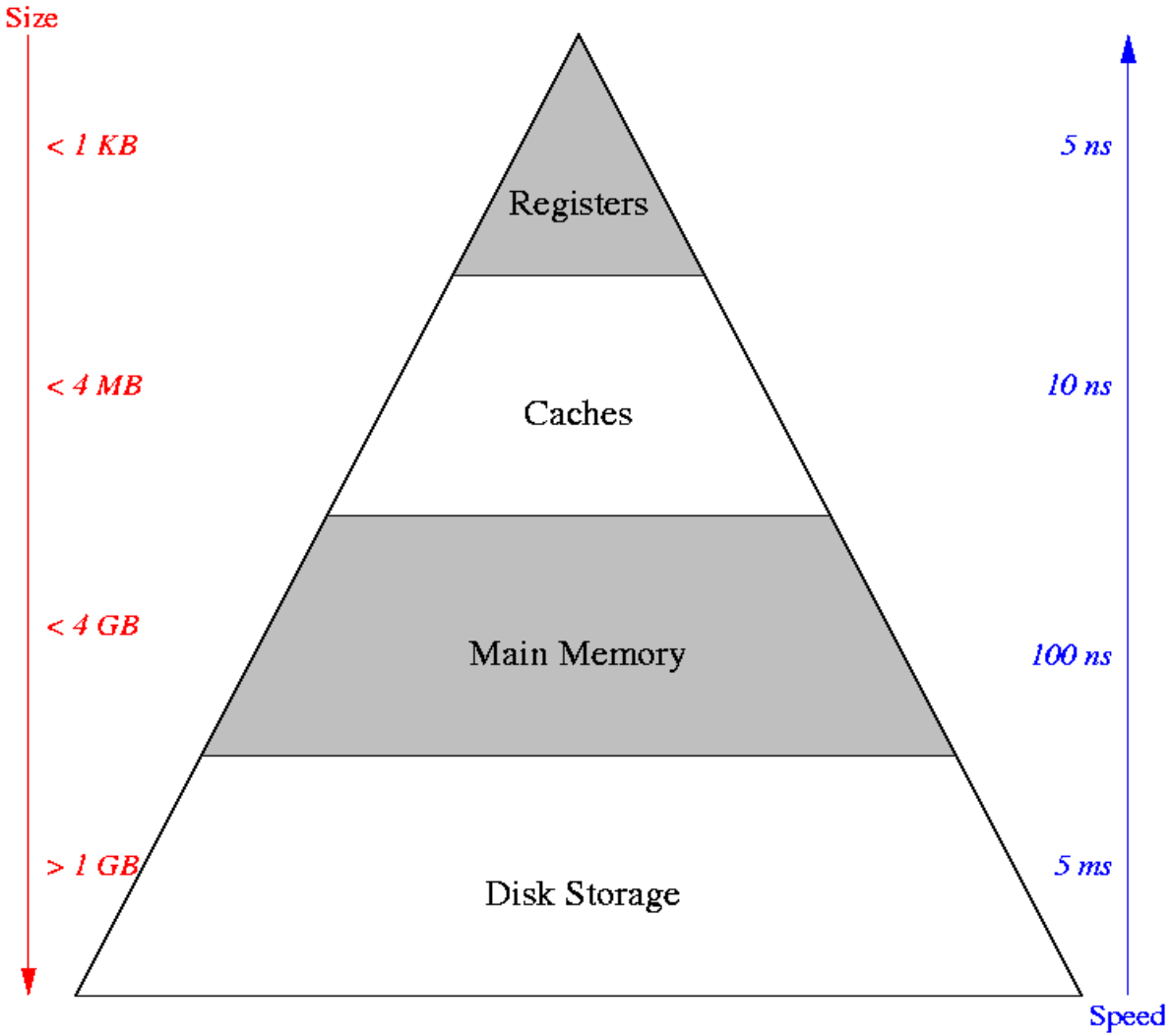
Tabla de Contenidos

2. Conceptos básicos

- Jerarquía de memorias
- Dirección física
- Dirección lógica
- Traducción de direcciones (correspondencia o reubicación)
  - Tipos de traducción (momentos)
  - Ejemplo de traducción básica (registro base)
  - ¿Quién realiza la traducción?
- Protección de memoria
  - ¿Quién controla la protección?
  - ¿Quién gestiona los fallos de protección?
  - Ejemplo de protección básica (registro límite)

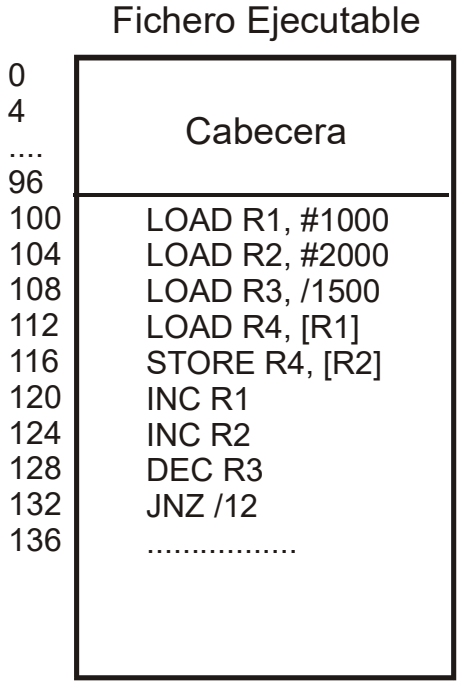
Define cada uno de estos conceptos

2.- Conceptos básicos
Jerarquía de memoria



2.- Conceptos básicos
Espacios lógicos independientes

- El código de un programa hace referencia a **direcciones lógicas del programa**, ***entre 0 y N***



- En ejecución el programa debe acceder a **direcciones físicas de memoria principal.**

2.- Conceptos básicos
Espacios lógicos independientes

- En sistemas monoprogramación no hay ningún problema. El proceso se carga a partir de la dirección de memoria física 0.
- En este caso, dirección física=dirección lógica.

- En sistemas multiprogramación hay más de un proceso en memoria a la vez, cada uno cargado a partir de su *dirección base.*
- Aquí, la dirección física no coincide con la lógica.
- Por lo tanto, es necesaria una traducción de direcciones lógicas a físicas. Este proceso se denomina ***traducción***, ***reubicación*** o ***correspondencia***.

2.- Conceptos básicos
Espacios lógicos independientes

- **Reubicación o correspondencia:** ***Traducción*** direcciones lógicas a físicas
- **Direcciones lógicas**: direcciones de memoria generadas por el programa
- **Direcciones físicas:** direcciones de memoria principal asignadas al proceso

Función de traducción BÁSICA

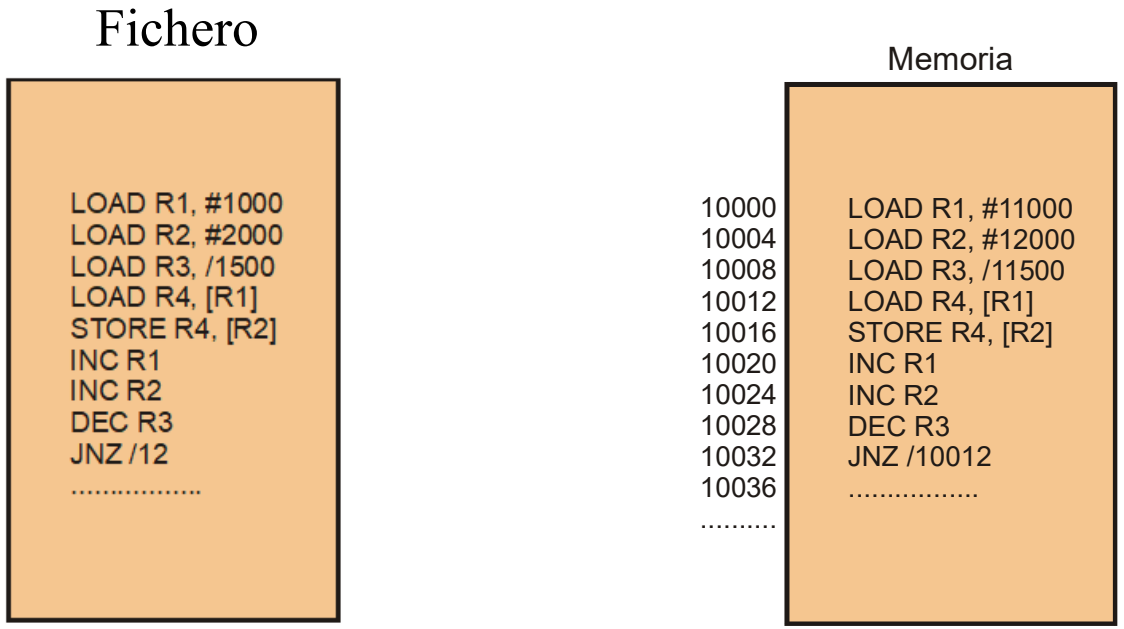
- Registro base: Primera posición del espacio **físico** del proceso.
- Dir. Física = Dir. Lógica + registro base.

- Dos alternativas:
  - Reubicación software
  - Reubicación hardware

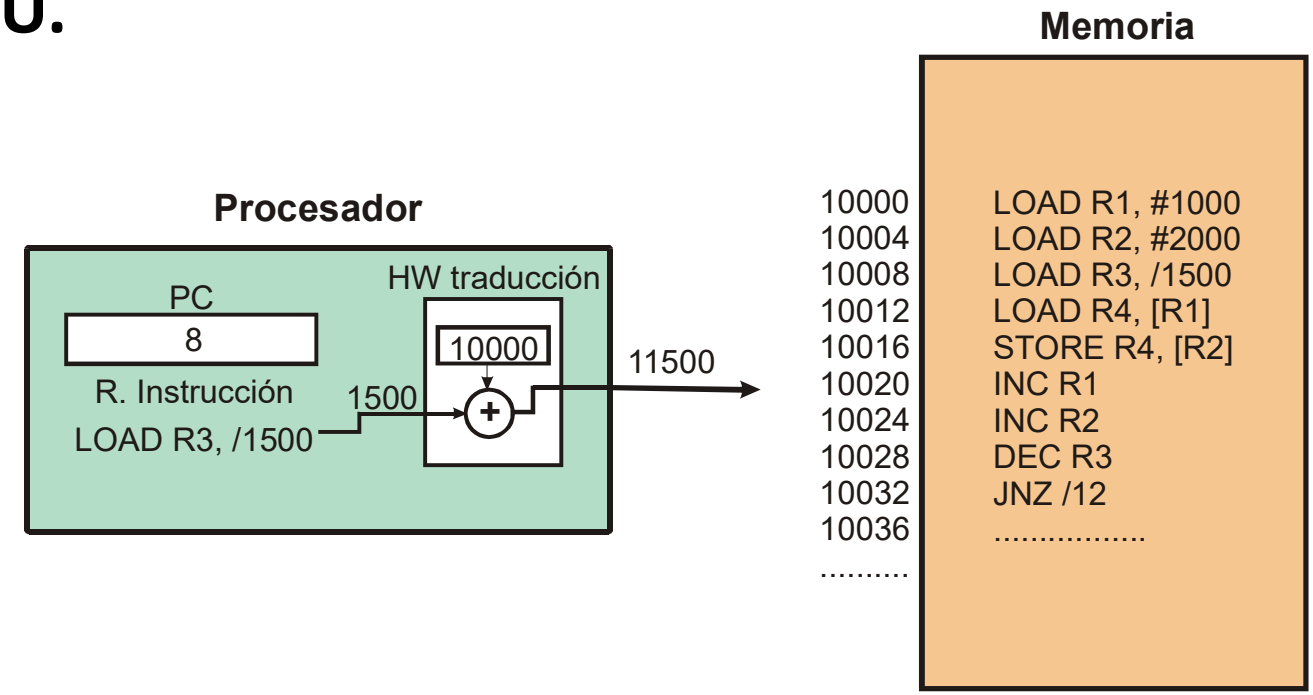
2.- Conceptos básicos

Espacios lógicos independientes: traducción por software

- Traducción durante la carga del programa.
- Programa en memoria distinto del ejecutable.
- Ventaja:
  - no requiere de hardware especial.
- Desventajas:
  - No permite mover programa en tiempo de ejecución fácilmente.
  - No incluye protección de memoria.



- Es el Hardware (la MMU) el que realiza la traducción
- S.O. se encarga de:
  - Transferir datos al Hw para realizar la traducción: establecer el valor del registro base de la MMU al cambiar de proceso.
- La traducción se realiza **en tiempo de ejecución por la MMU.**



2.- Conceptos básicos
Espacios lógicos independientes: protección

- **Monoprogramación**: Sólo se necesita proteger el SO
- **Multiprogramación**: Además, hay que proteger a los procesos entre sí.
- Es necesario validar **todas las direcciones** que genera el programa.
- No es posible realizarla por software. La protección debe realizarla el hardware del procesador.

  Mecanismo básico (implementado por la MMU)

```
Si dirección > tamaño del proceso
      entonces generarExcepcion,
sino traducir
```

- Si se genera la excepción, el SO debe actuar, normalmente abortando el proceso.

Tabla de Contenidos

3.- Esquemas de organización de memoria

Memoria Real

- Se reparte entre los procesos la memoria que realmente existe.
- Todo el proceso se encuentra cargado en memoria mientras existe en el sistema.

Swapping

- Para aumentar el número de procesos en el sistema, se pueden intercambiar procesos enteros hacia una zona de disco.

Memoria virtual

- Permite repartir entre los procesos más memoria de la que realmente existe.
- No necesita que esté cargado todo el proceso en memoria.

3.- Esquemas de organización de memoria

Memoria Real

- Asignación contigua
  - Con particiones fijas
  - Con particiones variables
- Asignación no contigua
  - Paginación simple
  - Segmentación simple
  - Segmentación + paginación simple

Memoria virtual

- Paginación
- Segmentación
- Segmentación + paginación

Esquema más utilizado: Memoria virtual con paginación

3.- Esquemas de organización de memoria
Parámetros cuantitativos

- **Fragmentación:** % de memoria desaprovechado (que no se puede utilizar)**.** Puede ser:
  - **Fragmentación interna:** espacio asignado a un proceso, pero que no está en uso.
  - **Fragmentación externa:** espacio no asignado a ningún proceso, pero que no es aprovechable.
  - **Fragmentación de tabla:** espacio utilizado por el SO para las estructuras de datos necesarias para gestionar la memoria.

Tabla de Contenidos

4.- Memoria real
Particiones fijas o estáticas

- La memoria se divide en un número **fijo** de particiones de tamaño **fijo**.
- El administrador puede configurar el número de particiones y el tamaño de cada una, pero eso exige reiniciar la máquina.
- A cada proceso se le asignará una partición donde quepa y que esté libre, en función de la *política de asignación.*
- Las estructuras de datos necesarias para su control son simples (poca ***fragmentación de tabla***).
- No genera ***fragmentación externa***, al ser las particiones estáticas.
- Genera ***fragmentación interna***, dado que el espacio desaprovechado en una partición no puede ser utilizado por otro proceso.

Ventajas

- Sencillez de implementación (tanto de Hw. como del Software).
- Traducción de memoria simple y eficiente.
- Los procesos tienen espacios lógicos independientes.
- Garantiza la protección de los procesos

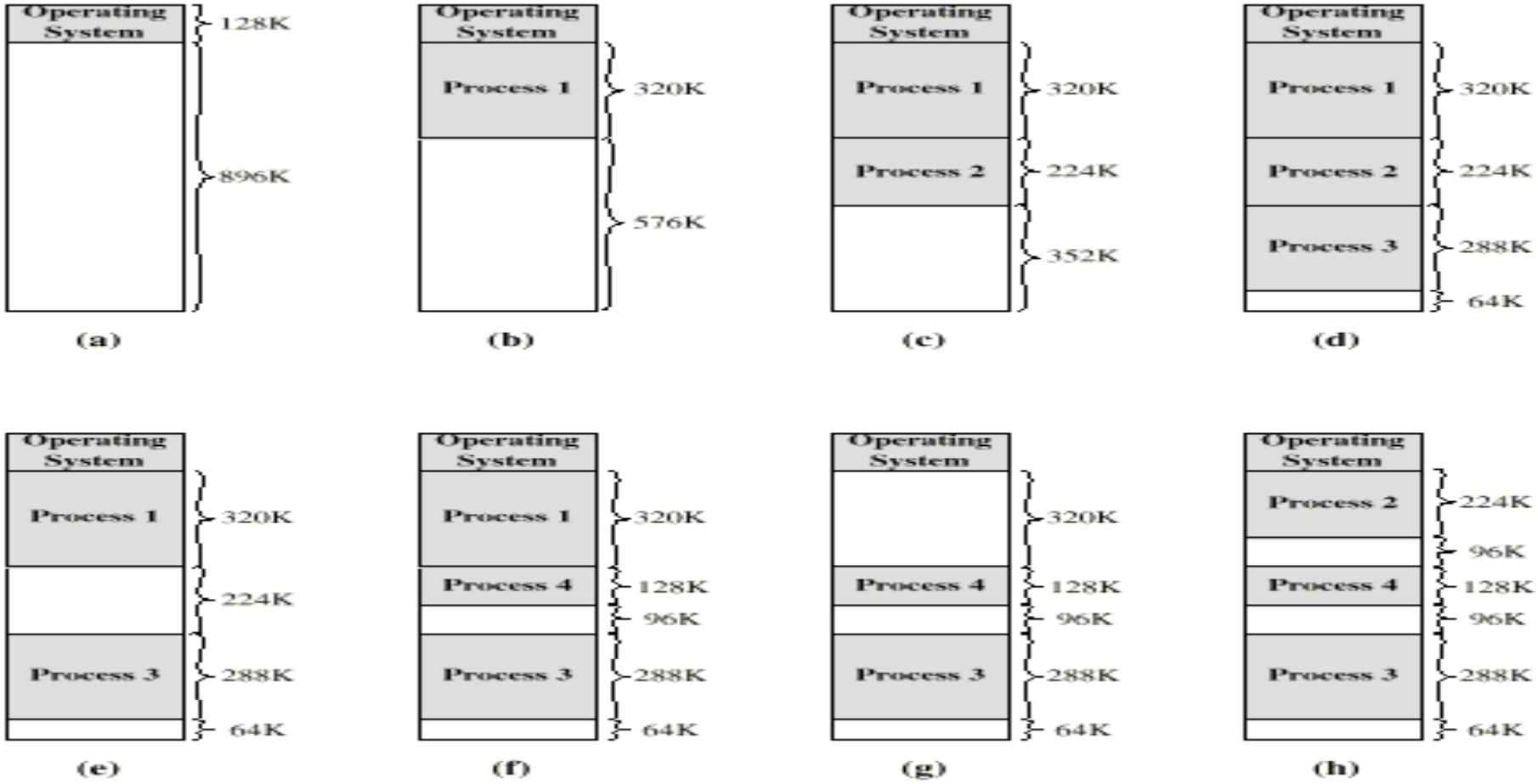
Inconvenientes

- Imposibilita la compartición de memoria.
- No contempla la gestión de regiones.
- Limita el grado de multiprogramación.
- Limita el tamaño del mapa de memoria de los procesos.

4.- Memoria real
Particiones variables o dinámicas

- La memoria al inicio de la ejecución del sistema la memoria aparece como un único hueco donde alojar procesos.
- Cuando se crea un proceso, se le asigna memoria dentro de un hueco donde quepa. El resto del hueco queda libre para otro proceso.
- La partición que se le asigne dependerá de la ***política de asignación*** que se use: *Mejor ajuste, Peor Ajuste, Primer Ajuste, Buddy, …*
- Las estructuras de datos necesarias para su control son simples (poca *fragmentación de tabla*), aunque más complejas que en particiones fijas.
- Genera *fragmentación externa*, que deberá ser resuelta por el sistema ***(compactación, condensación de huecos).***

4.- Memoria real
Particiones variables o dinámicas



Ventajas

- Relativa sencillez de implementación (tanto de Hw. como del Software). Algo más complicada que en particiones fijas.
- Traducción de memoria simple y eficiente.
- Los procesos tienen espacios lógicos independientes.
- Garantiza la protección de los procesos

Inconvenientes

- Imposibilita la compartición de memoria.
- No contempla la gestión de regiones.
- Limita el grado de multiprogramación pero menos que en particiones fijas.
- Limita el tamaño del mapa de memoria de los procesos, pero menos que en particiones fijas.

Tabla de Contenidos

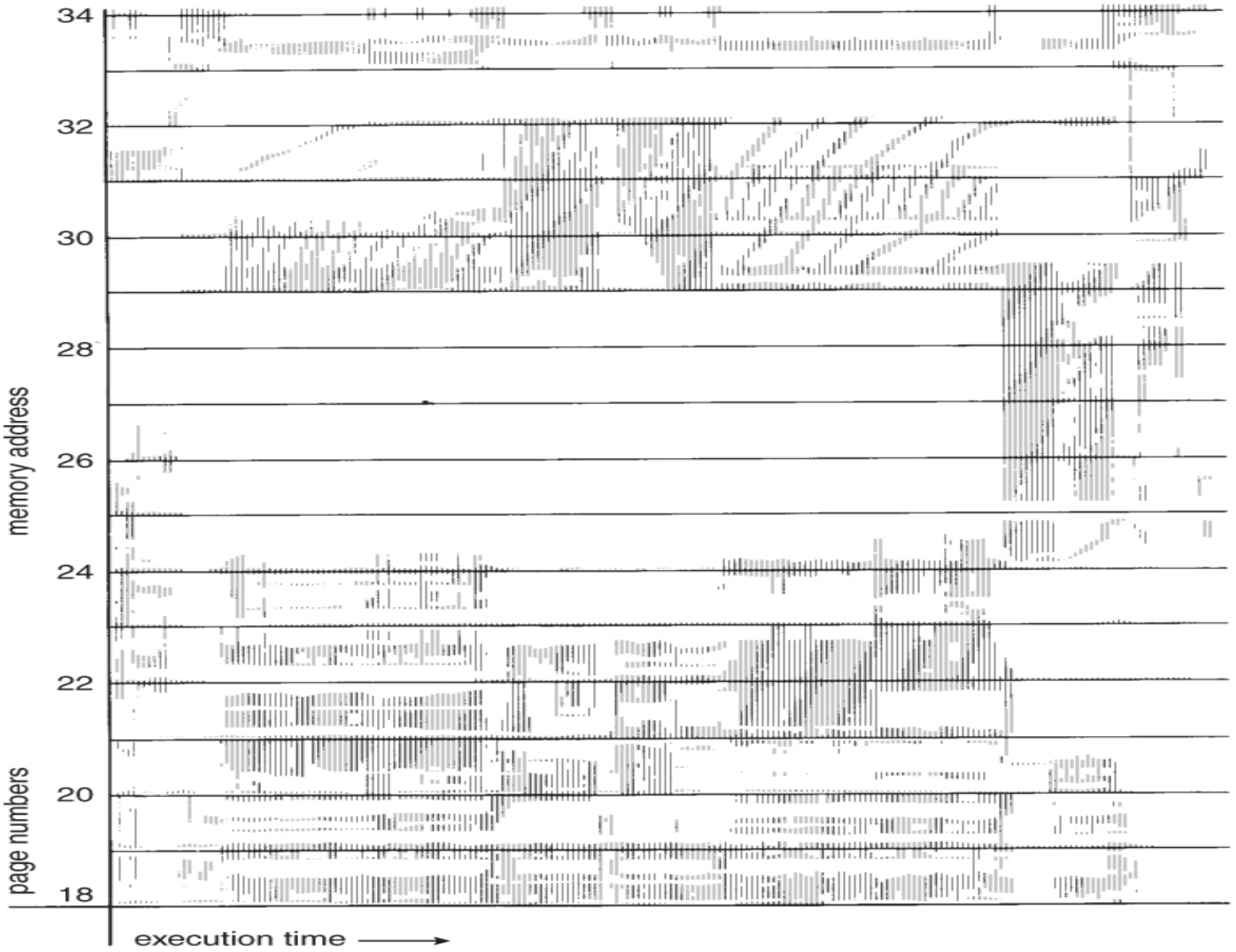
5.- Memoria virtual

- Técnica introducida ideada por Fritz-Rudolf Güntsch en 1956 e implementada John Fotheringham en 1961 para un sistema Atlas.
- “Amplía” la memoria principal utilizando disco para almacenamiento temporal de algunas zonas de procesos.
- No todo el programa tiene por qué estar cargado en memoria principal para su ejecución.
- El programa se divide en fragmentos, algunos de los cuales estarán en MP y otros en MS. Los fragmentos que se cargan y descargan cuando se necesite.
- Se utiliza **asignación no contigua**: los fragmentos no tienen por qué estar almacenados en posiciones contiguas de memoria.

5- Memoria virtual
Localidad de referencias

- Basa su funcionamiento en el ***principio de Localidad de referencias***, descrito en 1967 por Peter Denning:
- *Cuando se hace referencia a una dirección de memoria es muy probable que se haga referencia* ***a ella*** *o* ***a otras cercanas*** *en un tiempo corto.*
- Esto implica que las referencias a memoria tienden a agruparse en zonas relativamente pequeñas de memoria durante un determinado periodo de tiempo.

5.- Memoria virtual
Localidad de referencias



5.- Memoria virtual

Ventajas

1. **El tamaño del proceso puede ser mayor que el de la propia memoria**. El espacio de direcciones lógicas podría llegar hasta 2 elevado al número de bits del bus de dirección, independientemente de la memoria instalada.

2^{32}	=	4,294,967,296

2^{64}	=	18,446,744,073,709,551,616

2. Aumenta el grado de multiprogramación.
   - Se asigna una pequeña porción de memoria para cada proceso.
   - **Caben más procesos**

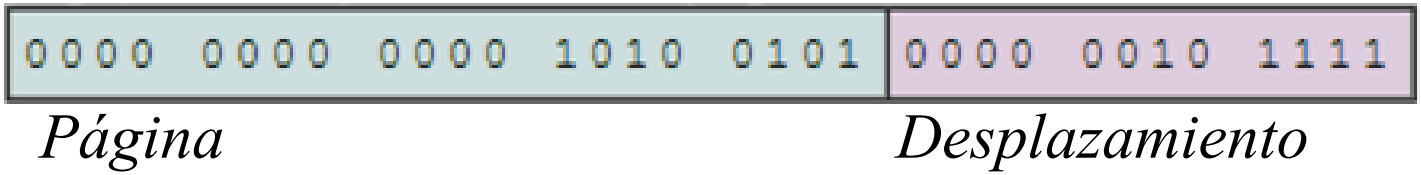
3. Transparente al programador.

   La **gestión corre a cargo del sistema operativo** en su totalidad

5.- Memoria virtual
Paginación

- Se divide el espacio de direcciones del proceso en fragmentos de igual tamaño, denominados ***páginas***.
- Se divide la memoria fragmentos de igual tamaño (e igual al tamaño de la página), denominados **marcos** de página.
- Las páginas de los procesos pueden cargarse en cualquier marco.
- Las direcciones lógicas se hacen corresponder con pares (página, desplazamiento).
- Las direcciones físicas se hacen corresponder con pares (marco, desplazamiento).

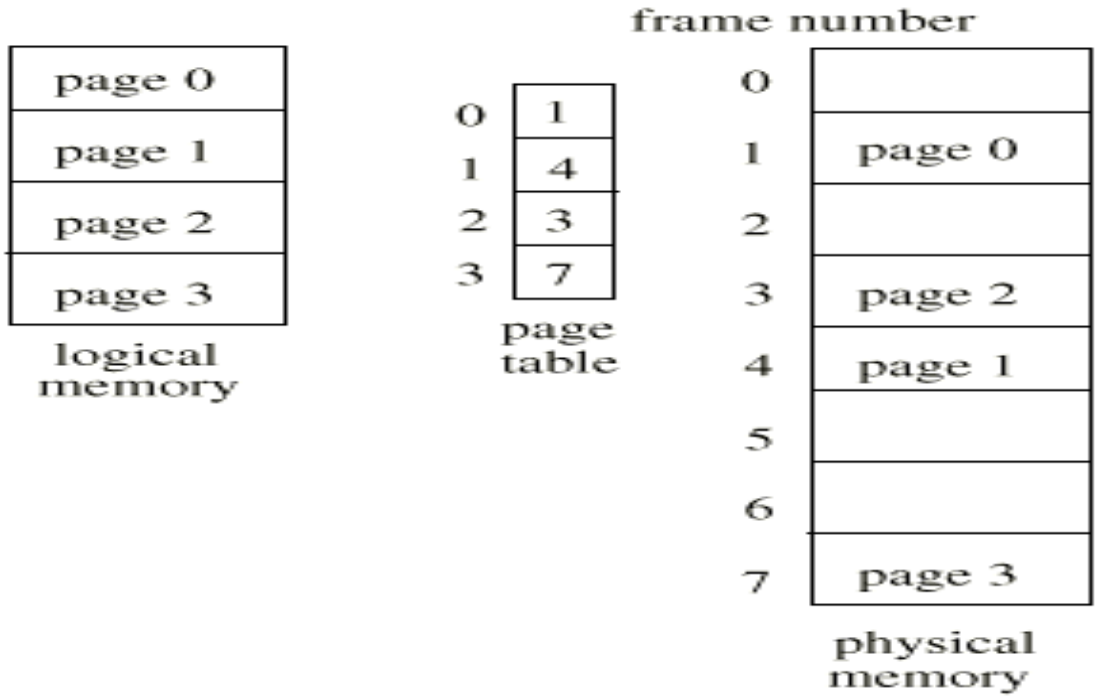
- Para traducir una dirección lógica en una física basta con sustituir el número de página por el número de marco donde la página está cargada.
- Si la página no está cargada en memoria se produce un ***fallo de página*** (*excepción*).
- Para facilitar la implementación, el tamaño de página es potencia de 2.
  - La transformación de una dirección "lineal" a un par (página, desplazamiento) se limita a dividir la dirección lineal en dos partes:



- Si tengo una dirección de 32 bits y un tamaño de marco de 4K
  - ¿Cuál es el tamaño máximo del proceso que puedo crear?
  - ¿Cuántos bits se usan para el desplazamiento?
  - ¿Cuántas páginas se pueden referenciar?

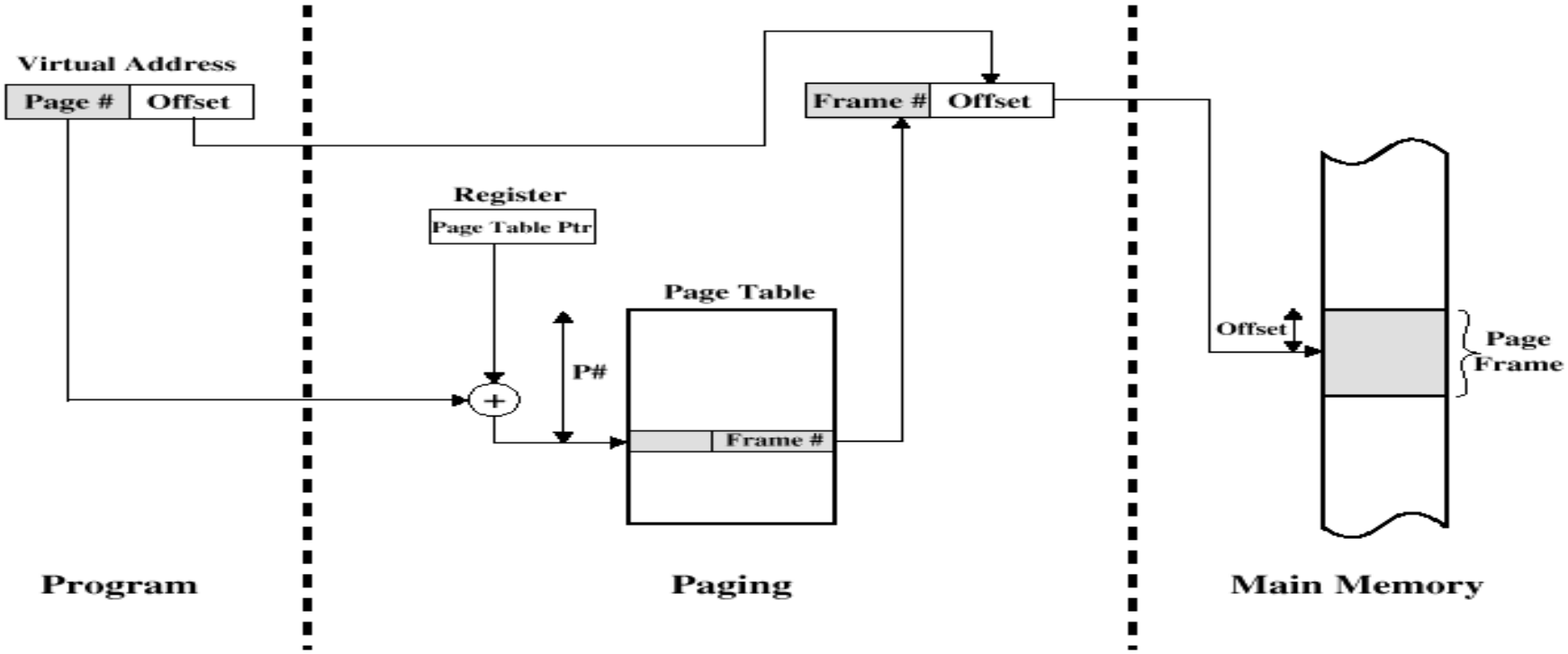
- Para gestionar el espacio de direcciones del proceso el sistema mantiene una tabla de páginas para cada proceso. Tiene una entrada para cada página del proceso.
- Para cada página, tendrá, al menos, la siguiente información:
  - El **número de marco** donde está cargada la página (en su caso).
  - **Bit de presencia.** Indica si la página está en memoria principal
  - **Bit de modificación.** Indica si la página ha sido modificada desde que se ha cargado de disco.
  - **Bits de protección** (r,w,x)
  - Otra información, en función de las políticas de administración de la memoria virtual.
- Además, el sistema necesitará una tabla de marcos, para saber cuáles están libres y cuales no.

Ejemplo de tabla de páginas para un proceso



Traducción de direcciones lógicas a físicas

Como se ha indicado, basta con sustituir el número de página por el número de marco donde la página está cargada, accediendo a la tabla de páginas para conocer este dato.



Problemas relacionados con la tabla de páginas

1. Rendimiento: hacen falta dos accesos a memoria para cada referencia a una dirección de memoria.

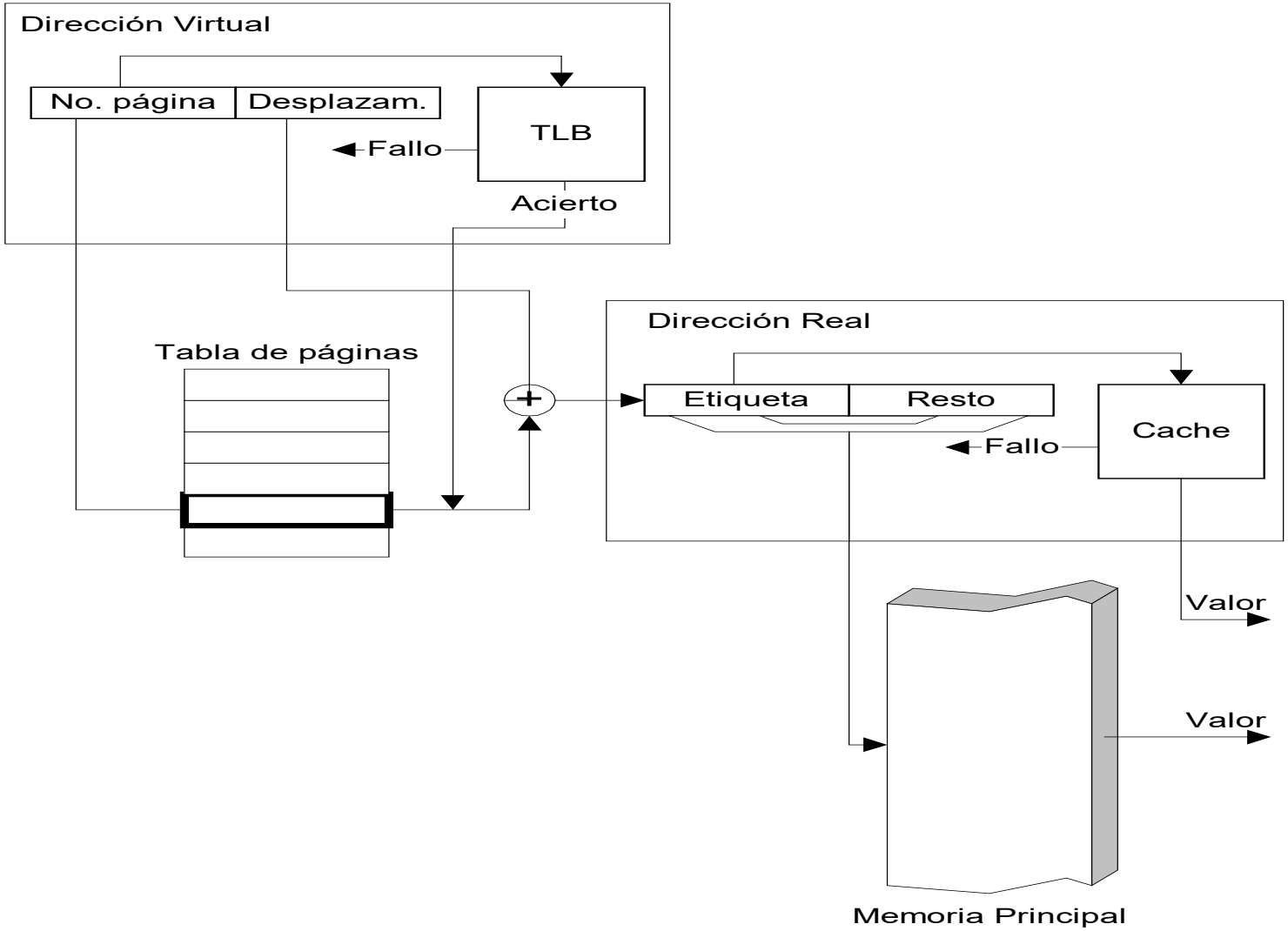
Uso adicional de un Hardware especial (TLB)

- *Memoria asociativa accesible por clave y muy rápida*
  - *Clave -> nº página. Contenido -> nº marco*
- *Almacena parte de la tabla de páginas*

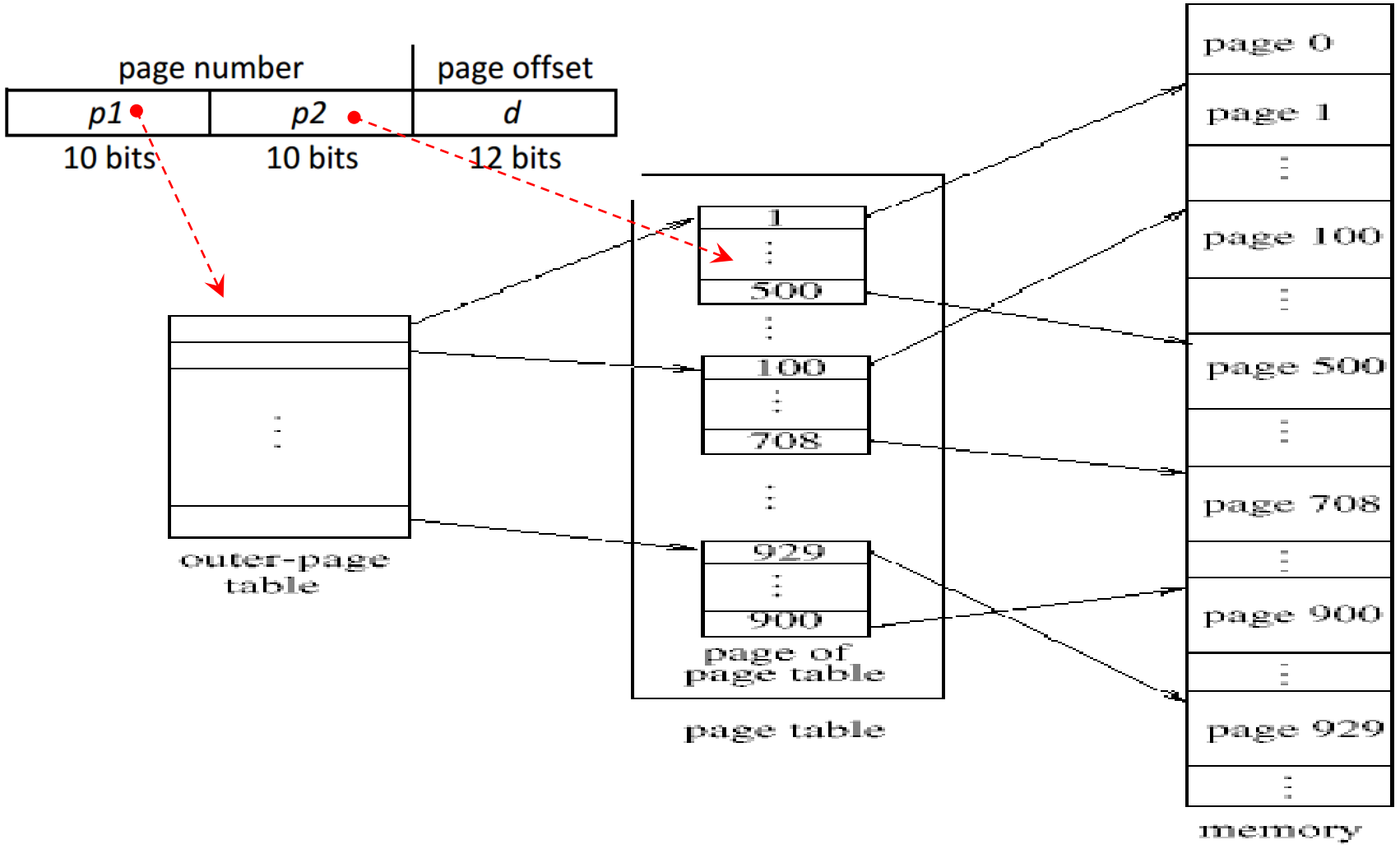
2. Almacenamiento de las tablas de páginas. Pueden ser muy grandes.

- *Tablas de páginas multinivel.*
- *Tablas hash de páginas.*
- *Tablas de páginas invertidas.*

5.- Memoria virtual
Paginación



Tablas de páginas multinivel



¿Cuál es el papel del Hardware?

- Traducción de direcciones.
- Protección (registro límite, bits de protección).

¿Cuál es el papel del Sistema Operativo?

- Asignar marcos a un nuevo proceso (¿cuántos?).
- Crear tabla de páginas y asociarla al PCB.
- Cargar páginas en marcos asignados.
- Gestionar fallos de página.
- Dar valor inicial y modificar dinámicamente la tabla de páginas.
- Decidir descargar páginas para cargar otras cuando sea necesario.
- Cargar parte de Tabla de Páginas en TLB cuando se pase el proceso a ejecución.
- …

Concepto

- Aplicación del concepto de región a la gestión de memoria a nivel hardware.
- Un proceso está compuesto de un conjunto de segmentos, cada uno de ellos con un espacio de direcciones propio.
- Cada segmento almacena una región del proceso.
- Cada segmento se almacena en posiciones contiguas (la gestión presenta problemas similares a las particiones variables).
- Se almacenará una tabla de segmentos por proceso, con el tamaño, bit de carga, bit de modificación, etc.
- Necesitará una tabla de particiones para gestionar las particiones libres/ocupadas en el sistema.

Ventajas:

- Permite aumentar el grado de multiprogramación (numero de procesos a la vez en memoria para ejecutarse).
- Garantiza la protección de los procesos
- **Posibilita la compartición de memoria.**
- **Contempla la gestión de regiones**.

Inconvenientes:

- **Problemas de gestión del almacenamiento real** (similar a particiones variables).
- Traducción de memoria no es simple y para ser eficiente se complica todavía más.

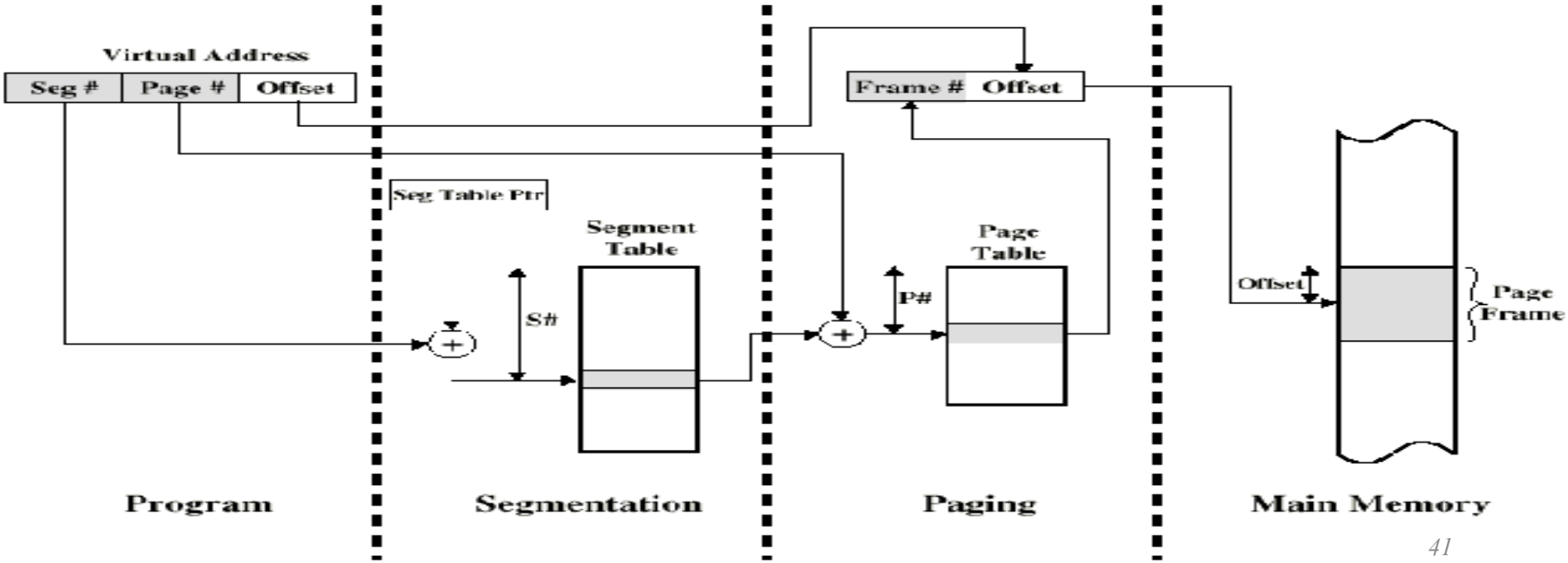
Concepto

- Se pretende juntar las ventajas de la segmentación para la gestión del espacio lógico con las de la paginación para la gestión del espacio físico.
- Un proceso está compuesto de un conjunto de segmentos, cada uno de ellos con un espacio de direcciones propio.
- Cada segmento se divide en un conjunto de páginas de igual tamaño.
- La memoria se divide en marcos de página de igual tamaño.
- Cada proceso tendrá una tabla de segmentos y cada segmento una tabla de páginas.

Estructuras de datos utilizadas

- Cada proceso
  - Una tabla de segmentos
  - Varias tablas de páginas

Traducción de direcciones



Ventajas:

- Permite aumentar el grado de multiprogramación.
- Aumenta el tamaño del mapa de memoria de los procesos.
- Garantiza la protección de los procesos
- Posibilita la compartición de memoria.
- Contempla la gestión de regiones.

Inconvenientes:

- Traducción de memoria no es simple y para ser eficiente se complica todavía más (algo más complicada que paginación pero menos que segmentación).

Tabla de Contenidos

2 tipos de decisiones de diseño

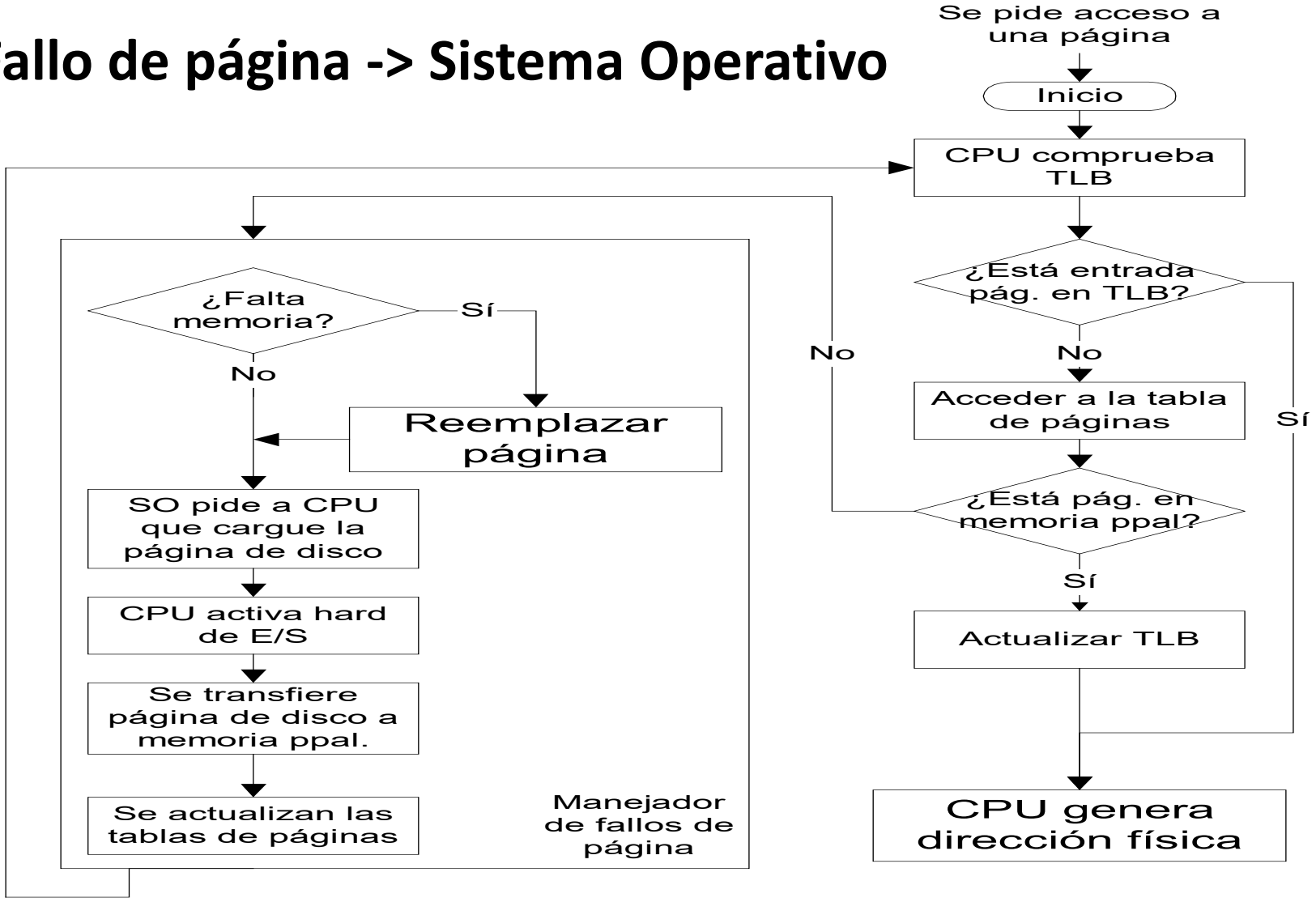
1. Dependientes del Hardware
   - Paginación, Segmentación, Segmentación con Paginación

2. Dependientes del Software
   - Qué hacer cuando se produce un ***Fallo de Página***
     - Políticas de gestión para la administración del almacenamiento:
       - Políticas de lectura
       - Políticas de reemplazo
       - Políticas de asignación

6.- Administración del almacenamiento virtual

Actuación del SO

- **Fallo de página -> Sistema Operativo**



6.- Administración del almacenamiento virtual

Política de lectura

Determina **cuándo se debe cargar una página** en memoria

Paginación por demanda

- Se traen a memoria cuando se produce el fallo.

Prepaginación

- Se traen a memoria varias páginas contiguas
- Mayor eficiencia de carga desde dispositivo secundario

6.- Administración del almacenamiento virtual

Política de reemplazo

Determina **qué página se reemplaza en memoria principal** cuando se debe cargar otra y no hay espacio para ella.

Alcance de reemplazo global/local

- Global
  - Se reemplaza una página de cualquier proceso que esté en memoria
- Local
  - Se reemplaza una página del proceso que la necesita

6.- Administración del almacenamiento virtual

Política de asignación

Determina **cuántos marcos se asignan a un proceso**

Buscar el número ideal (Equilibrio)

- Pocos para que quepan más procesos en memoria
- Suficientes para que no se produzca hiperpaginación

Estrategias de Asignación

- Fija
  - Con reemplazo local
  - No se adapta a las necesidades de memoria del proceso
- Dinámica
  - Con reemplazo local o global

HIPERPAGINACIÓN

Excesivo número de fallos de página

Implica

- ***Continua carga y descarga de páginas***
- ***Pérdida de rendimiento***

Algoritmos básicos

(los veremos con asignación fija y alcance local pero pueden aplicarse también con asignación dinámica y/o alcance global)

- Óptimo
- LRU, Menos recientemente usada
- FIFO, Primera en entrar, primera en salir
- Reloj

Algoritmo óptimo

Debe generar el **mínimo número de fallos de página**

- Reemplaza la página que tardará más tiempo en volverse a usar
- Genera el menor número de fallos de página
- **Imposible de implementar**, puesto que no se sabe a priori cuál va a ser.

- Lo que se hace realmente
  - Aproximaciones a este algoritmo mediante predicciones.

Algoritmo FIFO, primera en entrar, primera en salir

- Reemplaza la página que ha estado más tiempo en memoria, sin tener en cuenta si ha sido últimamente referenciada o no.
- Técnica sencilla de implementar. Sólo se necestia un lista FIFO de las páginas que están en memoria.
- Con asignación fija sufre la anomalía de Belady: puede ocurrir que teniendo asignado más marcos se generen más fallos.
- Rendimiento pobre: no tiene en cuenta ningún tipo de predicción

Calcula cuántos fallos de página se producen con la siguiente secuencia de referencias, usando FIFO con 3 y con 4 marcos:

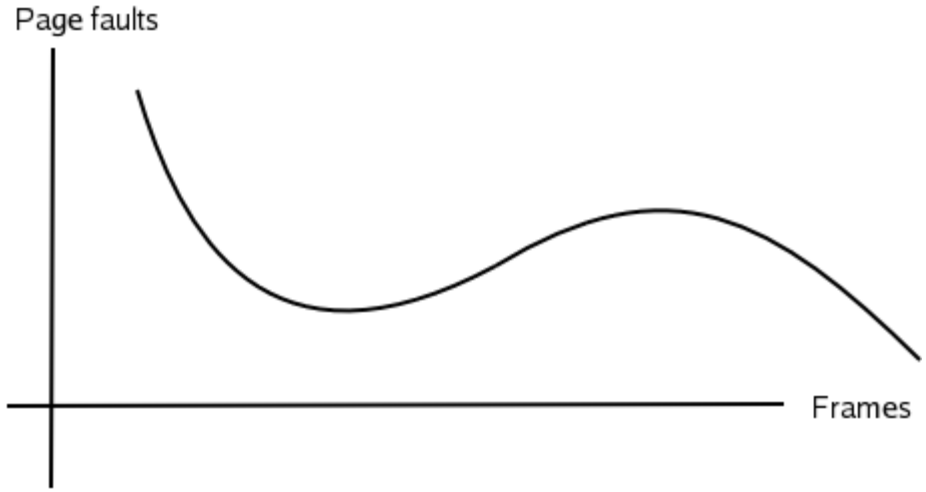
1 2 3 4 1 2 5 1 2 3 4 5

6.- Administración del almacenamiento virtual

Políticas de reemplazo

Tasa de fallos esperada

Tasa de fallos con la anomalia de Belady

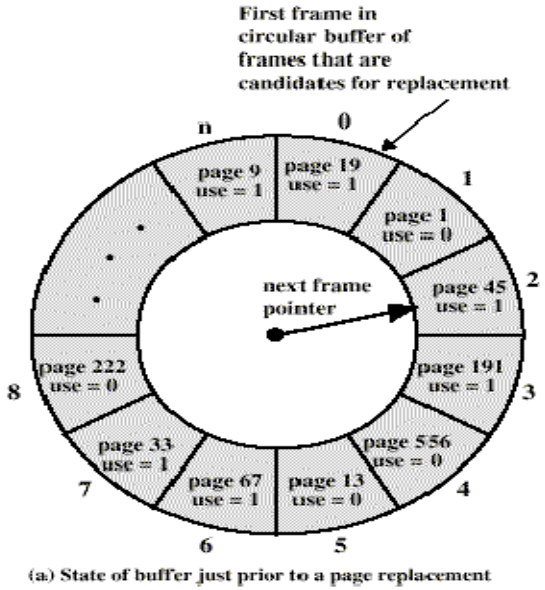


Algoritmo de la página menos recientemente utilizada (LRU)

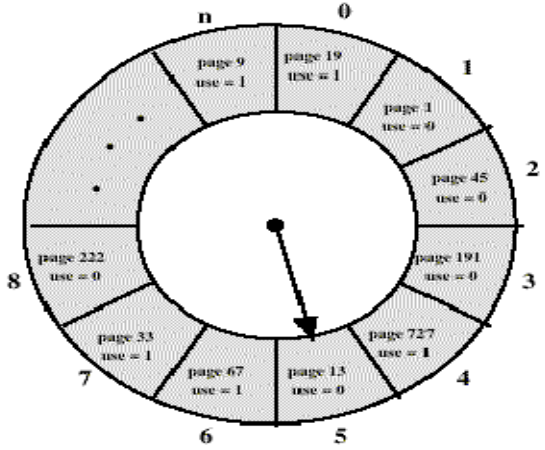
- Reemplaza la página que más tiempo hace que ha sido referenciada.
- Por principio de localidad es la página con menor probabilidad de ser referenciada.
- Política muy afinada. Es una buena predicción. No sufre la anomalía de Belady, pero **lleva una gran sobrecarga de ejecución** (para saber cuál es la menos recientemente usada)

Implementaciones de algoritmo basados en LRU

- Contadores: campo de tiempo de uso
- Pila
- Bits de referencia adicionales



(a) State of buffer just prior to a page replacement



(b) State of buffer just after the next page replacement

Algoritmo del reloj

- Aproximación al algoritmo LRU.
- Se expulsa primero las páginas que llevan mucho tiempo sin usarse (no necesariamente la que lleva más)
- Requiere asociar un “bit de referencia” a cada marco
- Cada vez que se haga referencia a una dirección de la página el Hw. pone este bit a 1.
- Si le toca expulsión, se mira el bit de referencia
  - Si está a 0 se expulsa
  - Si está a 1
    - no se expulsa y se pone a 0 (segunda oportunidad)
    - Se repite el proceso con la siguiente página del buffer

Estrategias de Asignación

- Fija
  - Reemplazo local
  - No se adapta a las necesidades de memoria del proceso
- Dinámica
  - Reemplazo local o global

Políticas de asignación dinámica

1. Estrategia del conjunto de trabajo
2. Estrategia basada en la frecuencia de fallos

Estrategia del conjunto de trabajo

Conjunto de trabajo (Peter Denning 1968) de un proceso en un instante virtual t, con parámetro Δ (W(t, Δ))

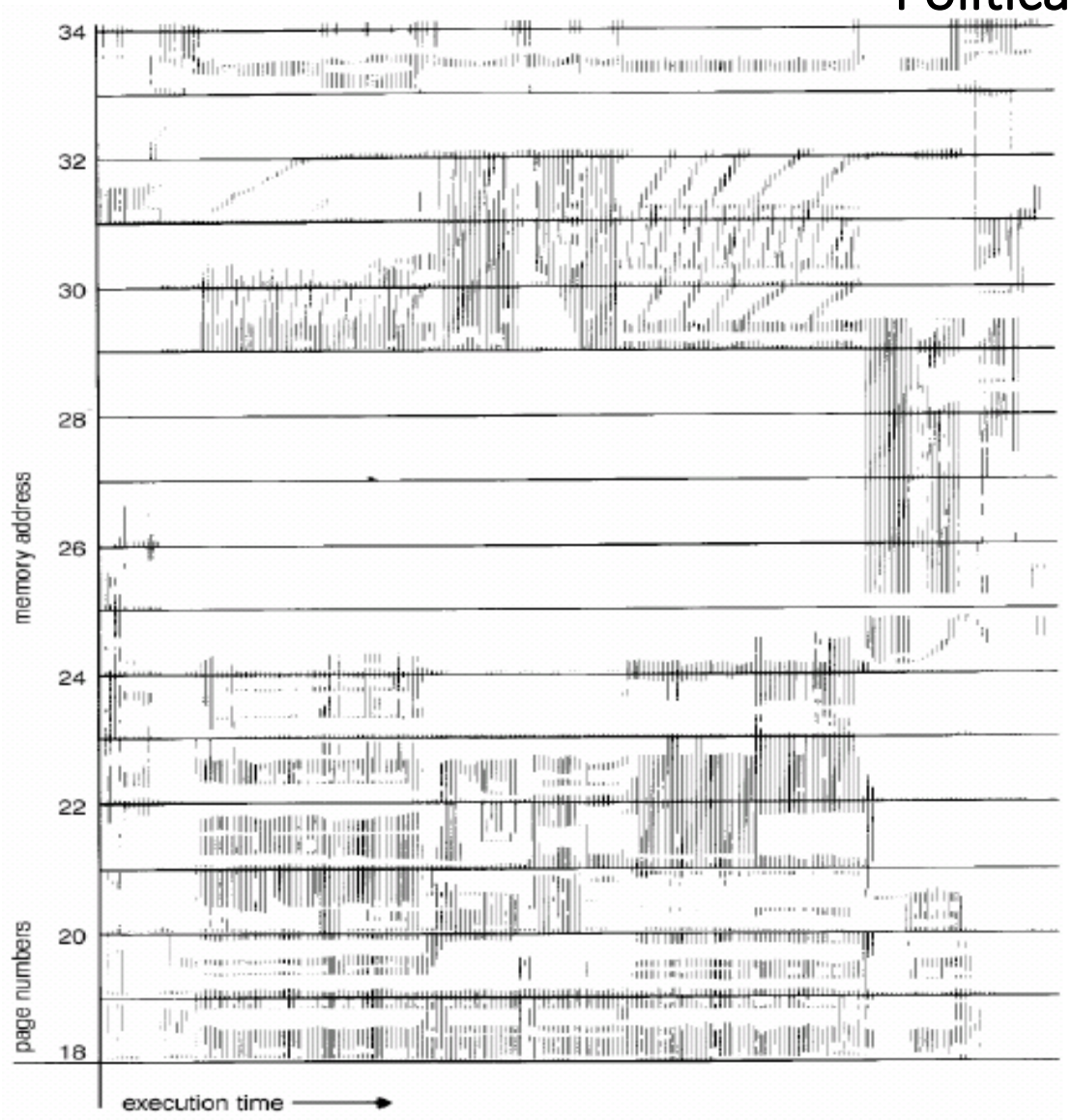
- Es el conjunto de páginas a las que el proceso ha hecho referencia en las últimas Δ unidades de tiempo (intervalo)
- A medida que se ejecuta el programa, tanto el tamaño de su conjunto de trabajo como la identidad de las páginas que lo forman varía con el tiempo.

Se elige un tamaño de la ventana (n últimas páginas referenciadas)

- Suficientemente grande para que no haya hiperpaginación Suficientemente pequeña para aprovechar la Memoria principal.

6.- Administración del almacenamiento virtual

Políticas de asignación dinámica



- Evolución del conjunto de trabajo
- Páginas accedidas por un proceso durante su ciclo de vida.

El principio del conjunto de trabajo **establece** que:

Cuando un proceso tiene en memoria el conjunto de trabajo se produce una baja tasa de fallos de página.

1. Un programa debería ejecutarse si y sólo si su conjunto de trabajo se encuentra en memoria.
2. Una página no debe ser retirada de memoria si es miembro del conjunto de trabajo de un programa en ejecución.

Consecuencias

- Ayuda a evitar la ***hiperpaginación*** manteniendo a la vez un elevado grado de multiprogramación.

Estrategia

- Si el conjunto de trabajo decrece, se liberan marcos asociados a las páginas que ya no están en el conjunto de trabajo
- Si el conjunto de trabajo crece, se asignan nuevos marcos. Si no hay marcos libres en memoria habrá que suspender algún proceso

Implementación

- Requiere la intervención del sistema operativo después de cada referencia a memoria. Demasiada sobrecarga.
- Se necesita una MMU específica para controlar las páginas accedidas en las últimas n referencias.

2. Estrategia de reemplazo por Frecuencia de Fallos de Página

Supone una aproximación del conjunto de trabajo.

La frecuencia de fallos de página resulta una medida de la eficiencia de ejecución de un proceso.

- Cuando se produce un fallo de página, si el proceso tiene una frecuencia de fallos grande, se incluye esa página en su conjunto de trabajo, y por tanto se aumenta el número de marcos
- Necesita saber el tiempo en que se produjo el último fallo de página
- Se establecen valores límite apropiados para la tasa de fallos
- Sólo consume tiempo después de cada fallo, no en cada referencia a memoria. Cuando hay un fallo de página, calcula el tiempo desde el último fallo de página. De acuerdo con este valor, pueda incrementar/decrementar el número de marcos.

Buffering de páginas

- Se mantiene un conjunto de marcos libres en memoria de reserva
- Existe
  - Una lista de páginas libres
  - Una lista de páginas libres - modificadas
    - Se deben guardar a disco cada cierto tiempo y pasan a libres
- Si se reduce el conjunto de trabajo
  - Se libera un marco (pasa a libres o a libres-modificadas si su contenido ha sido modificado respecto a lo que hay en disco)
- Si se aumenta el conjunto de trabajo
  - Se coge un marco de lista de libres
- Si quedan pocas libres se guardan las libres-modificadas y pasan a libres