MEMORIA VIRTUAL

Karim Guevara Puente de la Vega

2017

Introducción

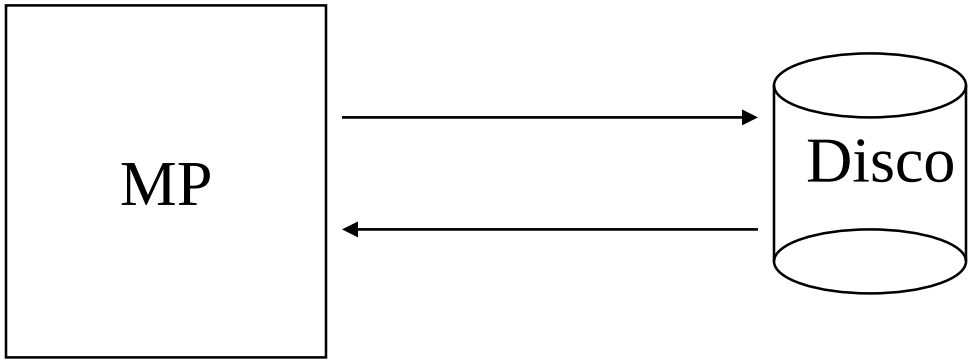
- Los programas pueden ser más grandes que la memoria
- Los programas suelen tener código que maneje condiciones de error.
- Tablas, listas o matrices se declaran con mucho espacio.
- Ciertos programas ofrecen características que nunca o raramente se utilizan.
- Por tanto…. NO NECESITAN TODO AL MISMO TIEMPO

Introducción

- Solución adoptada:
  - Dividir el programa en partes llamadas overlays
    - La ejecución se empieza en el primer overlay
    - Cuando termina se llama al siguiente y así
    - Se pueden tener en memoria varios overlays
  - El SO se encarga del intercambio
  - Pero de la partición del programa el programador… solución: **memoria virtual**

Memoria Virtual [Fotheringham, 1961]

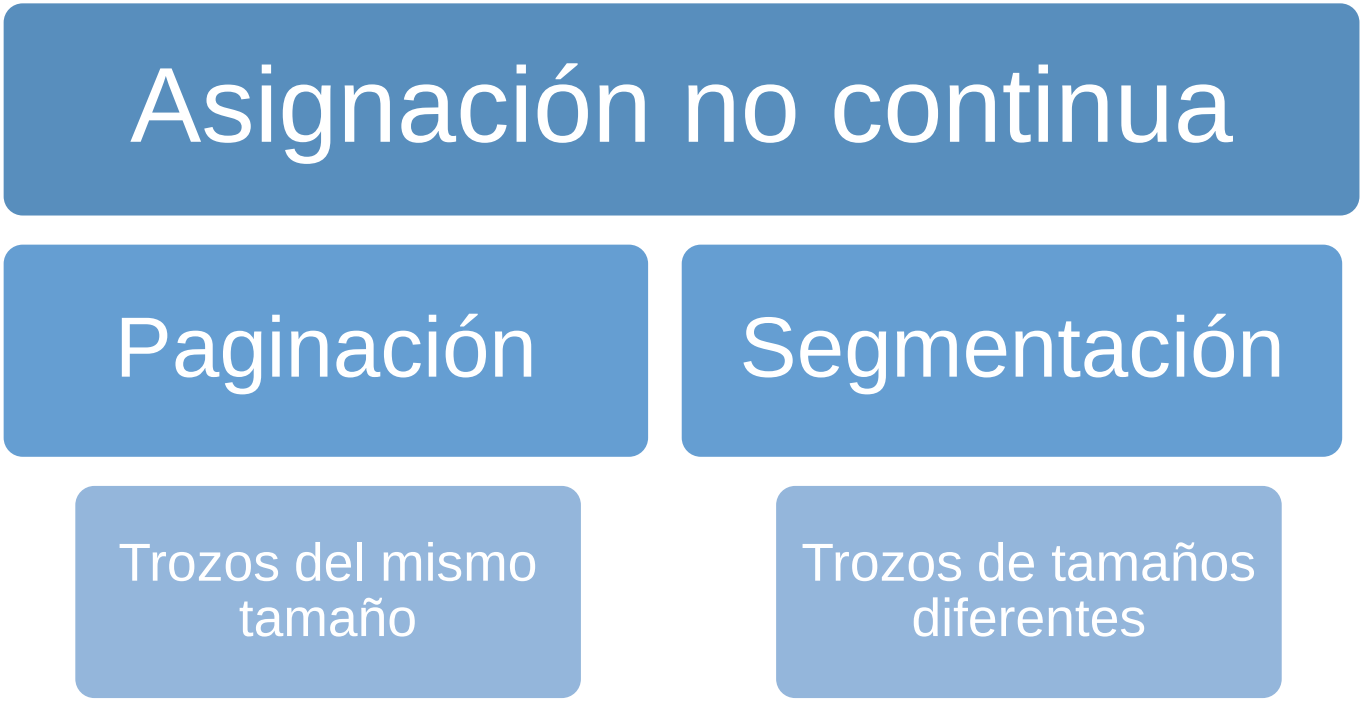
- Técnica que permite que un proceso sea mayor que la cantidad de memoria disponible
- En memoria están las partes del programa que se están utilizando, el resto en disco



Memoria Virtual

- Tamaño de los programas no limitado por la memoria principal.
- Se pueden cargan más programas en memoria para ejecutarse al mismo tiempo.
- Se necesitan menos E/S en la carga o expulsión a disco de un proceso y por tanto la ejecución global es más rápida.

Memoria Virtual



Memoria virtual

- En multiprogramación:
  - En memoria sólo una parte del programa
    - Se pueden tener más programa
    - P.e. memoria de 2M:
      - 2 programas de 1M completamente en memoria
      - 8 programas con 1/4 de M cada uno en memoria

- Es usual utilizar **paginación o segmentación**

Paginación

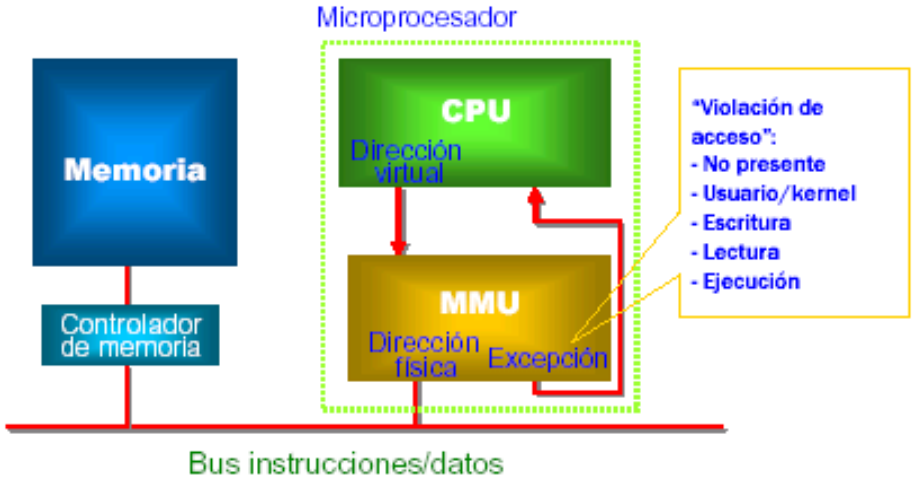
- Cada programa tiene su propio espacio de direcciones dividido en páginas.
- Cada página es un rango contiguo de direcciones.
- Las páginas se asocian a la memoria física
- No todas tienen que estar en la memoria física para poder ejecutar el programa.

Espacio lógico y físico

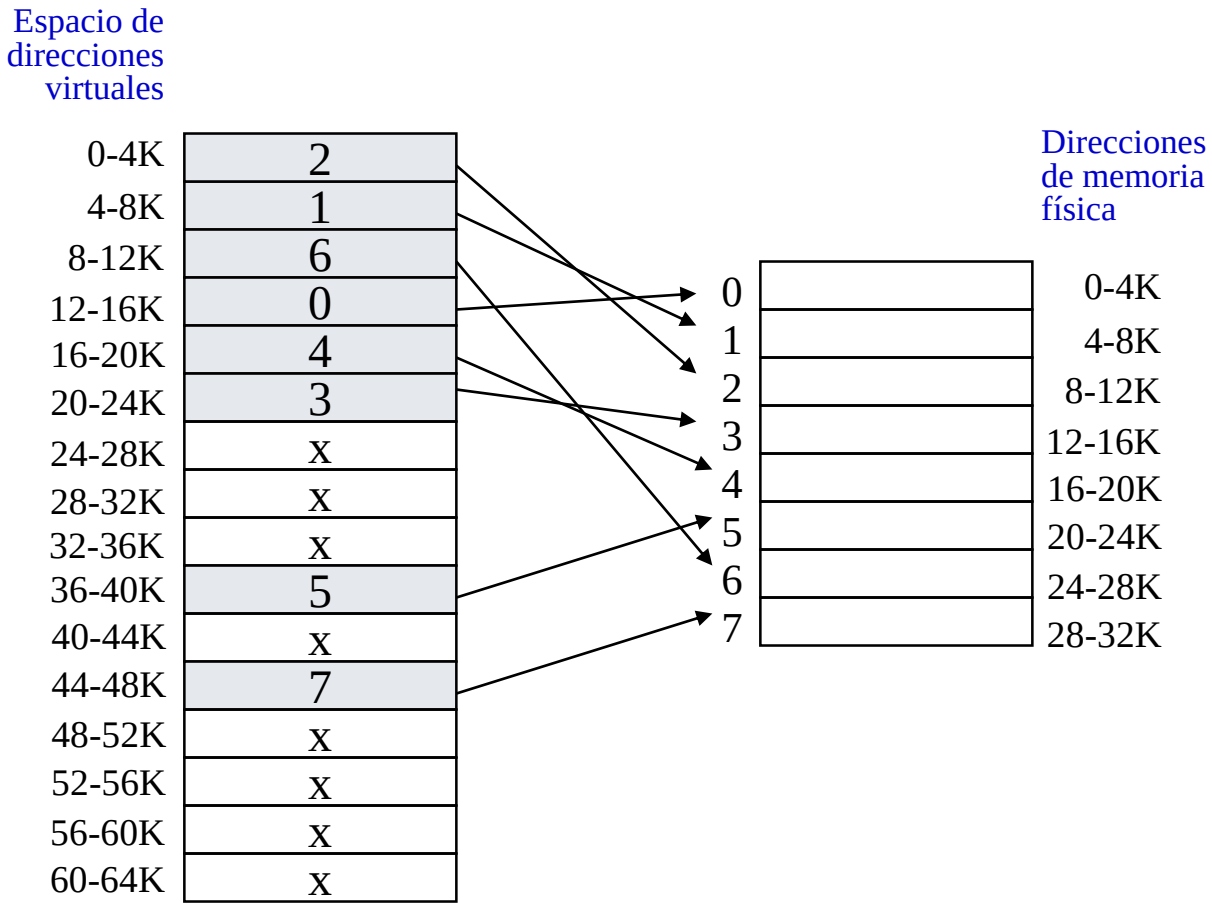
- **Dirección virtual:** dirección generada por un programa.
- Espacio direcciones físicas - direcciones físicas que se pasan al controlador de memoria
- Espacio de direcciones lógicas o virtuales - generada por la CPU (programa)
  - Se divide en unidades llamadas páginas
  - Las unidades correspondientes en memoria física son marcos de página
  - Las páginas y los marcos tienen idéntico tamaño.
  - La transferencia entre memoria y disco se realizan siempre en unidades de páginas.

Unidad de gestión de memoria (MMU)

- **Memory Management Unit** – dispositivo hardware que traduce direcciones virtuales en direcciones físicas.
- También implementa protección.
- El hardware determina la forma en la que el SO gestiona la MMU.
- En el esquema MMU más simple, el valor del registro de reubicación se añade a cada dirección generada por el proceso de usuario al mismo tiempo que es enviado a memoria.



Paginación



1) `move reg, 8192`
   `move reg, 24576`

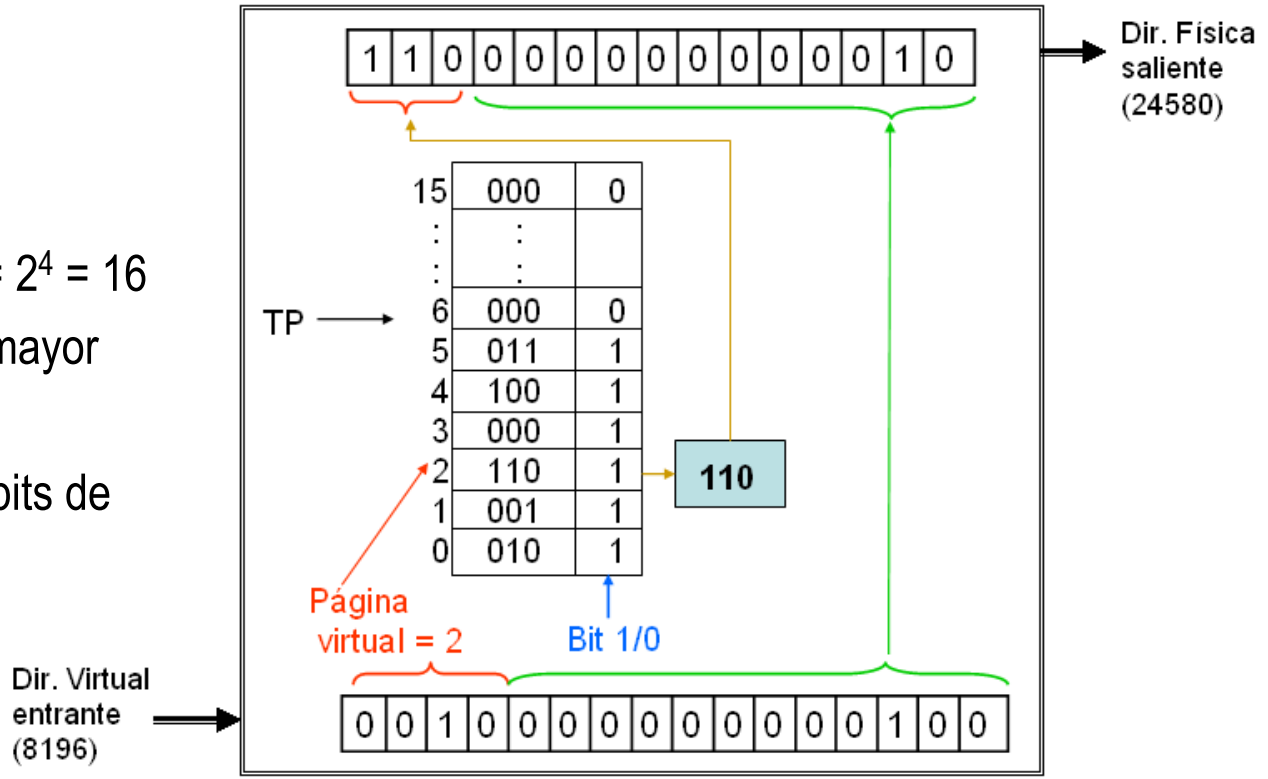
2) `move reg, 20500`
   `move reg, 12308`

3) `move reg, 32780`
   Fallo de página

¿Cómo funciona la MMU?

- Dirección virtual: 16 bits.
- Espacio de direccionamiento virtual: 64 K (2^{16})
- Tamaño de página :

  4 Kb (2^{12} bytes).
  - Cant. pág.: $2^{16} / 2^{12} = 2^4 = 16$
  - Nº página: 4 bits de mayor peso
  - Desplazamiento: 12 bits de menor peso

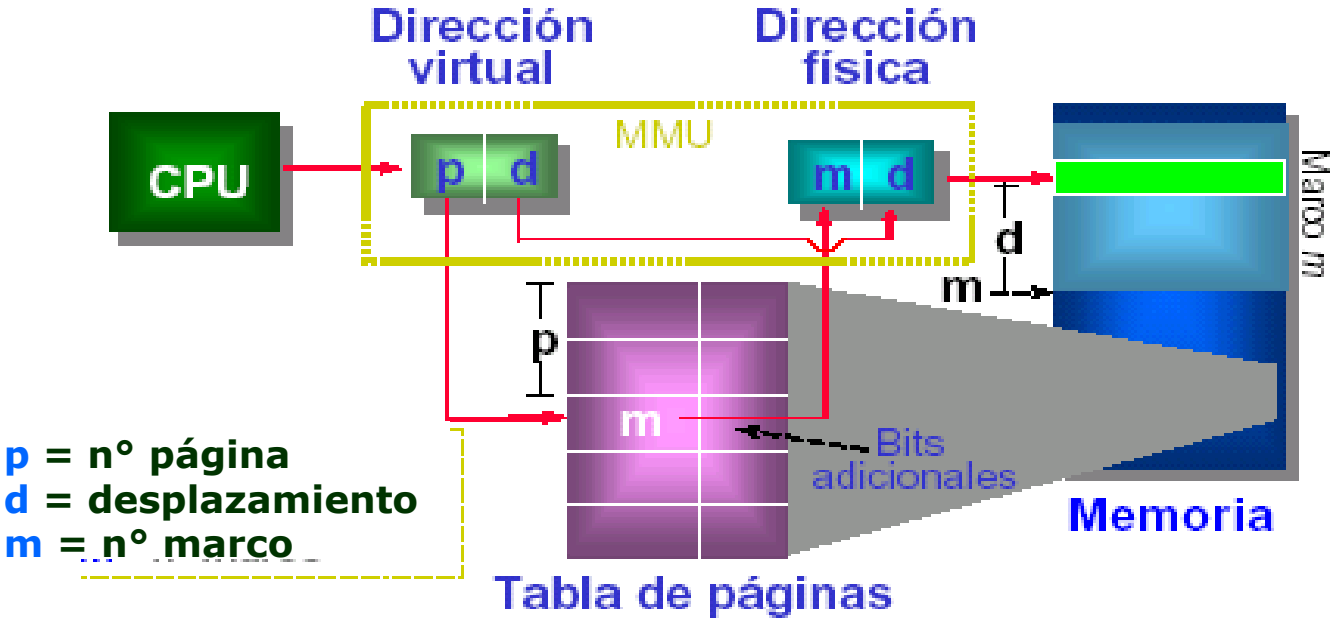


Fallo de página

- Trap generado por la MMU que captura el SO
- El SO:
  - Selecciona un marco de página
  - Copia su contenido en disco
  - Carga la página que se necesita en ese marco
  - Cambia la tabla de traducción de la MMU
  - Comienza de nuevo esa instrucción interrumpida

Tabla de páginas (TP)

- Tiene un elemento, entrada (PTE), por cada página del proceso. El contenido de esta entrada es la dirección del marco donde está cargada.



Estructura de una PTE

- Marco de página – número de marco donde esta alojada una página.
- Bit de presente / ausente - indica si la página tiene o no asociado un marco en memoria principal.
- Bits de protección - índica que tipo de acceso esta permitido por el proceso.
- Bit modificado (sucio) - indica si el contenido de la página fue modificado desde que se trajo de memoria secundaria.
- Bit referencia (solicitada) - se enciende cuando página es referenciada para leer o escribir

Aspectos de la traducción

- En todo sistema con paginación se debe tener en cuenta:
  1. La asociación/traducción debe ser rápida
     - Se debe hacer la traducción para cada referencia a memoria
     - En una instrucción puede haber varias referencias
  2. La tabla de páginas puede ser muy grande
     - Cada proceso cuenta con su propia tabla de páginas
     - Si dirección virtual es de 32 bits (2^{32}) y páginas de 4K (2^{12})
       - 1’000,000 de páginas (2^{32} /21^{2} = 2^{20})

Distintos diseños de la tabla de páginas

- Vector de registros rápidos en hardware
- Tabla de páginas en memoria
- Memoria asociativa
- Tablas de páginas multinivel

Vector de registros rápidos en hardware

- Una entrada por página virtual
- Al iniciar el proceso se carga la tabla de páginas del proceso en estos registros
  - Una copia de la TP en memoria principal
- Durante la asociación no hay que acceder a memoria
- Costosa en recursos si la tabla de páginas es grande (1 registro por página)
- Costosa en tiempo
  - En el cambio de contexto habría que cargar la tabla de páginas

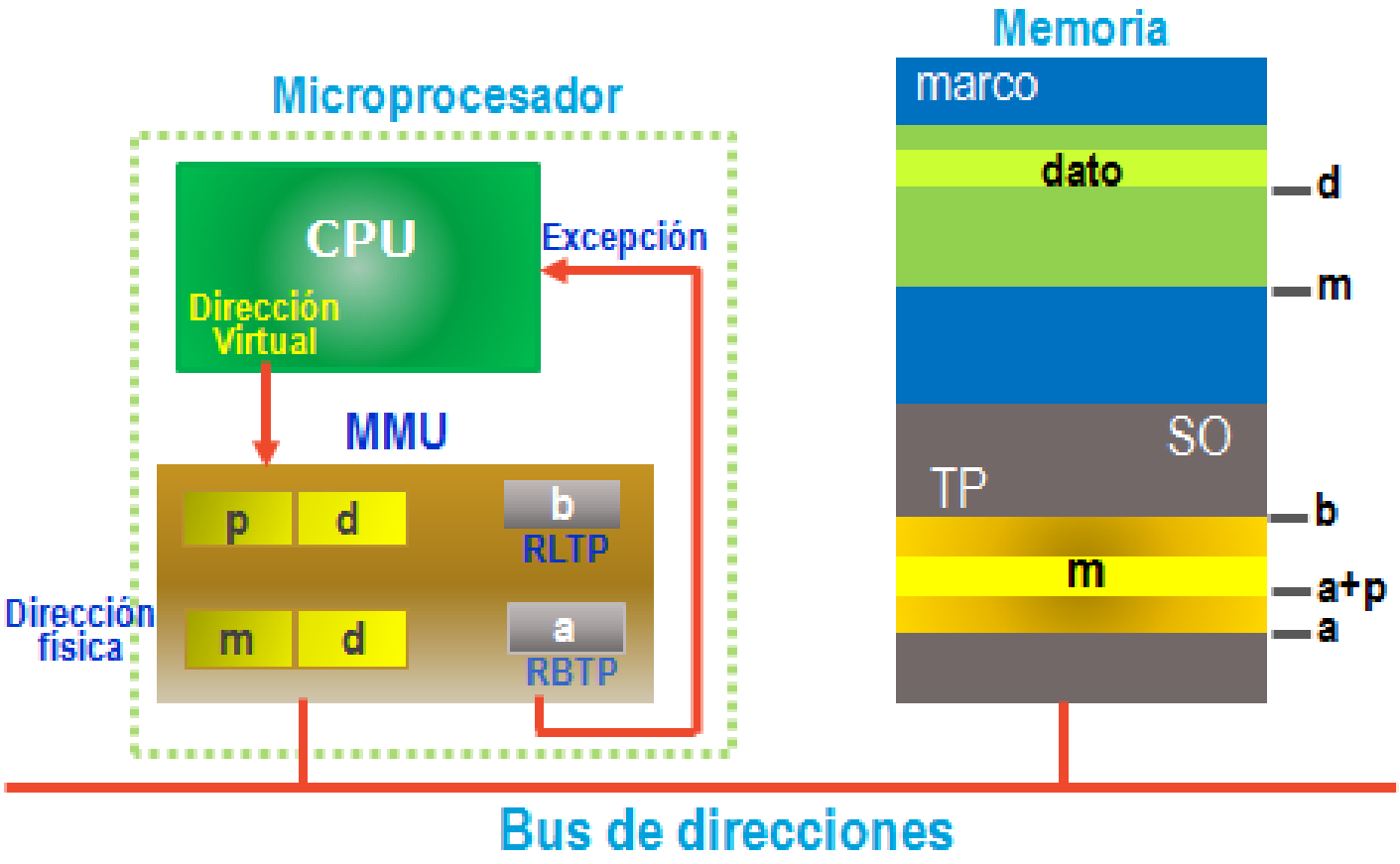
Tabla de páginas en memoria principal

- Para localizar la TP de un proceso, es necesario que la MMU disponga de:
  - **Registro Base de la Tabla de Páginas (RBTP)**: apunta a dirección base de la TP activa.
  - **Registro longitud de la tabla de páginas (RLTP)**: indica el tamaño de la TP activa

- Cambio de contexto
  - se modifican dos registros

- Hay que hacer una o más referencias a memoria por cada instrucción para acceder a la tabla de páginas

Tabla de páginas en memoria principal

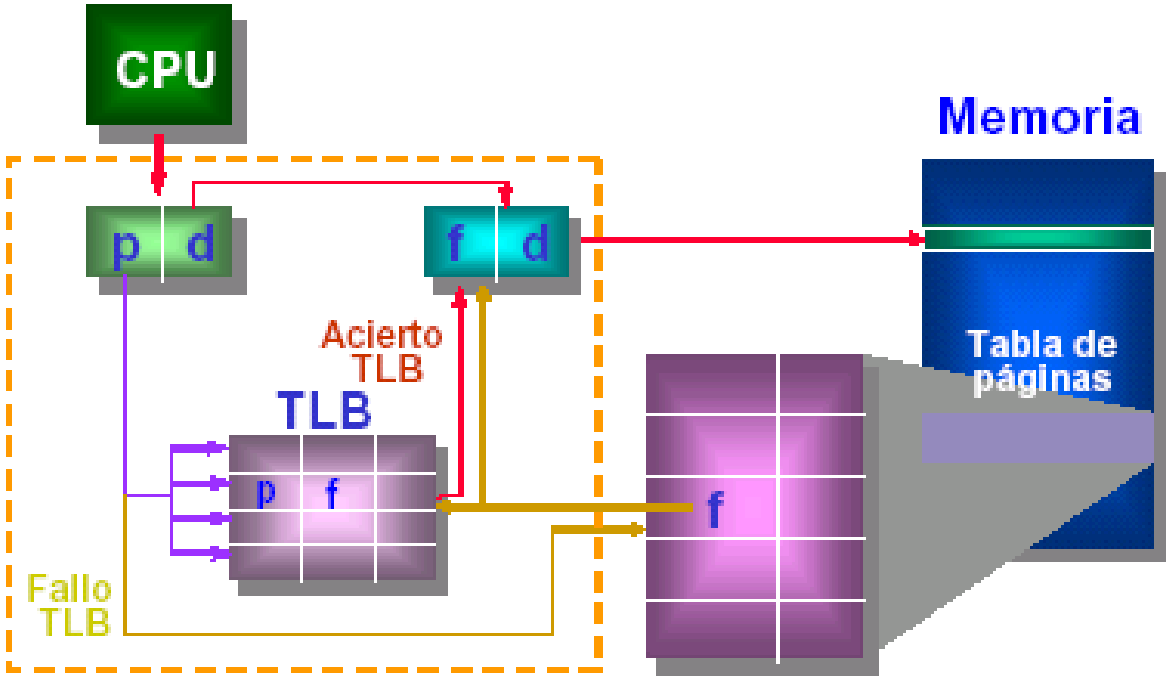


Memoria asociativa
Translation Lookaside Buffer – TLB

- Cada acceso a una instrucción/dato requiere dos accesos a memoria: uno a la TP y otro a la instrucción/dato.
- Se basa en la observación de que la mayoría de los programas hacen un gran número de referencias a un pequeño número de páginas
- Se equipa a la máquina con un dispositivo hardware (TLB) que permite asociar direcciones virtuales con físicas sin pasar por la TP
  - Parte de la MMU
  - Número pequeño de entradas
    - Cada entrada tiene la misma información de la TP, más la página virtual
  - Búsqueda en todas las entradas en paralelo

Memoria asociativa
Translation Lookaside Buffer – TLB

- Para la traducción
  - Se comprueba si la página esta en el TLB
  - Si está se hace la traducción
  - En caso contrario, se accede al la TP



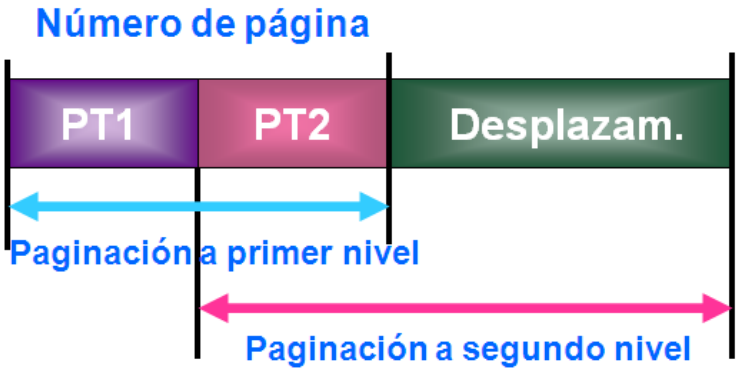
Tablas de páginas multinivel

- No todo el espacio de direcciones virtuales es utilizado por todos los procesos
  - Con 32 bits:
    - se pueden direccionar 4GBytes (2^{32} bytes)
    - con páginas de 4K: 2^{20} páginas ($2^{32}/2^{12}$)
    - tabla de páginas con 1 Millón de entradas
    - sólo se utilizan si el proceso ocupa 4 GBytes

Paginación multinivel

“Paginamos las tablas de páginas”

- No se necesita tener todas las TP´s en memoria todo el tiempo.
- La dirección lógica se divide:



PT1 = indice a TP nivel superior.

PT2 = indice a TP de segundo nivel

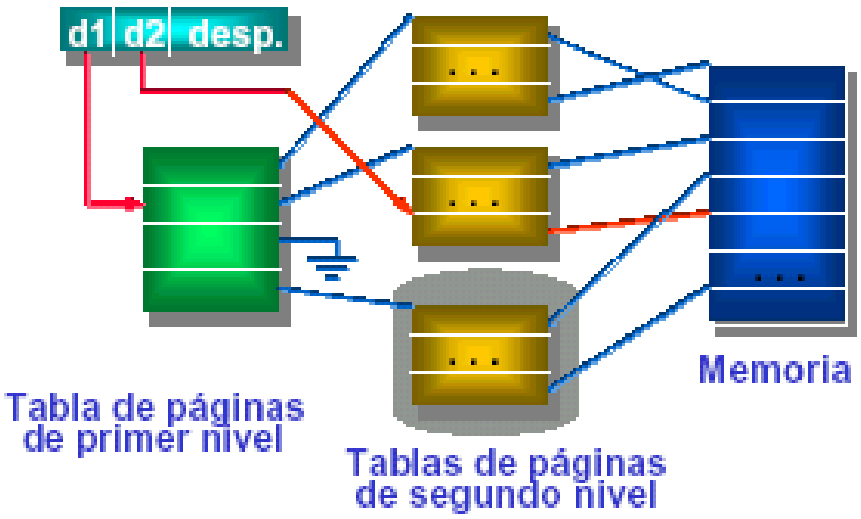


Tabla de páginas invertidas

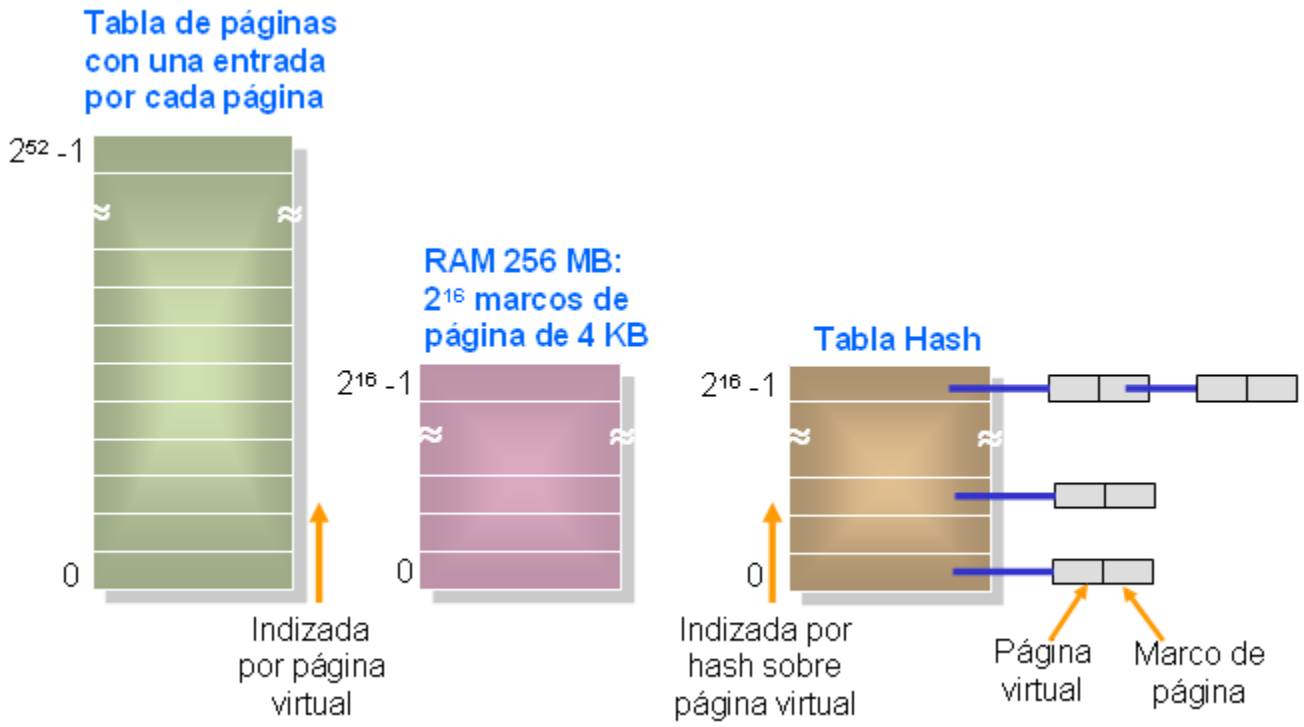
- ¿ Si la dirección virtual es de 64 bits?
  - Espacio de direcciones: 2^{64} bytes
  - Páginas: 4 Kb (2^{12} bytes)
  - Se requiere una tabla de páginas: 2^{52} entradas
  - Cada entrada requiere: 8 bytes

    + 30 millones de Gb

Tabla de páginas invertidas

- Hay una entrada por cada **marco** de página.
  - Si RAM=1 GB, páginas de 4 KB y dirección virtual de 64 bits, se requiere 262,144 entradas
- Cada entrada indica el proceso (***n***) y la página virtual (***p***) que esta en el marco correspondiente.
- La traducción de direcciones virtuales a físicas es más complicada.
  - Buscar una entrada (***n,p***) en toda la tabla….. **lento**
  - Solución: usar el TLB
    - Si hay fallo de TLB se debe de examinar por sw la tabla invertida
    - **Lento**, entonces utilizar tabla hash sobre la dirección virtual.
      - Todas las páginas virtuales que estén en memoria y tengan el mismo valor hash están enlazadas.

Tabla de páginas invertidas



ALGORITMOS DE REEMPLAZO DE PÁGINA

Paginación por demanda

- Cuando se produce un fallo de página: el SO debe decidir qué página que está en memoria debe pasar a disco para traer a memoria la página requerida.
- Si la página que sale ha sido modificada: se reescribe en disco (*bit sucio)*.
- Se puede elegir aleatoriamente, pero es más eficiente sacar una que no se vaya a utilizar pronto.
  - El algoritmo de sustitución debería producir el mínimo número de fallas de páginas.
- Este concepto es ampliable al reemplazo en los caches, pero a menor tiempo.

Algoritmos de reemplazo / sustitución

- La eficacia de los algoritmos se evalúa sobre una serie concreta de referencias a memoria y contabilizando el número de fallas de página.
- Se denomina *cadena* o *serie de referencia*, a la secuencia de números de páginas referenciadas por un proceso durante su ejecución:

$$R = r_1, r_2, r_3, ..., r_i, ...$$

  - P.e.: páginas de 100 bytes y las direcciones referenciadas: 0100, 0432, 0101, 0612, 0102, 0103, 0104, 0101, 0611, 0102, 0103, 0104, 0101, 0610, 0102, 0103, 0104, 0101, 0609, 0102, 015

$$R = 1, 4, 1, 6, 1, 6, 1, 6, 1, 6, 1, 0$$

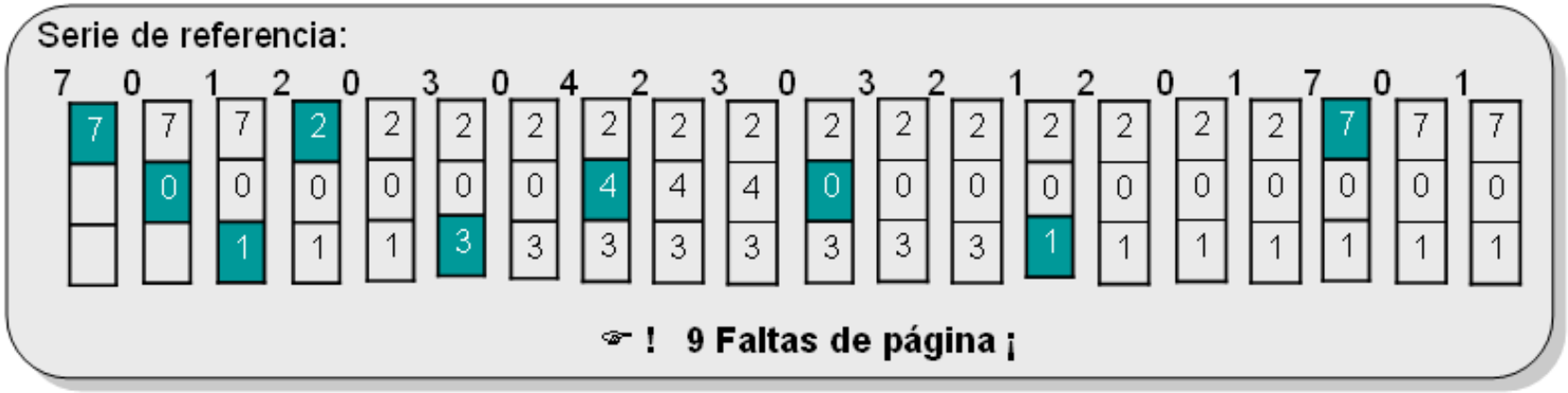
Algoritmos de reemplazo / sustitución

- Asignación local o global
  - Algoritmos estáticos: *local*
    - Proceso utiliza un número fijo de marcos
    - No ajustan la asignación a las necesidades del proceso
  - Algoritmos dinámicos: *global*
    - Número de marcos varía con la ejecución del proceso
    - Son sensibles a los requerimientos de los procesos

Óptimo

"Reemplazar la página que más tiempo vaya a tarda en ser referenciada"

- Difícil de implementar
  - Requiere conocer a priori la serie de referencia, el SO no tiene forma de saber esta información
- Irrealizable
  - Sirve para hacer estudios comparativos de otros algoritmos.



Página no usada recientemente - NRU
Not Recently Used

“Reemplazar al azar la página de la clase de número más bajo que no esté vacía”

- 2 bits asociados a cada página
  - bit R (bit de referencia): se pone a 1 cada vez que se lee o se escribe
  - bit M (bit de modificación): se pone a 1 cuando se escribe
- Cuando se carga en memoria: ambos a 0
- Con cada interrrupción: se pone el bit R a 0
- Con el fallo de página:

Clase 0:	No solicitada, no modificada
Clase 1:	No solicitada, modificada
Clase 2:	Solicitada, no modificada
Clase 3:	Solicitada, modificada

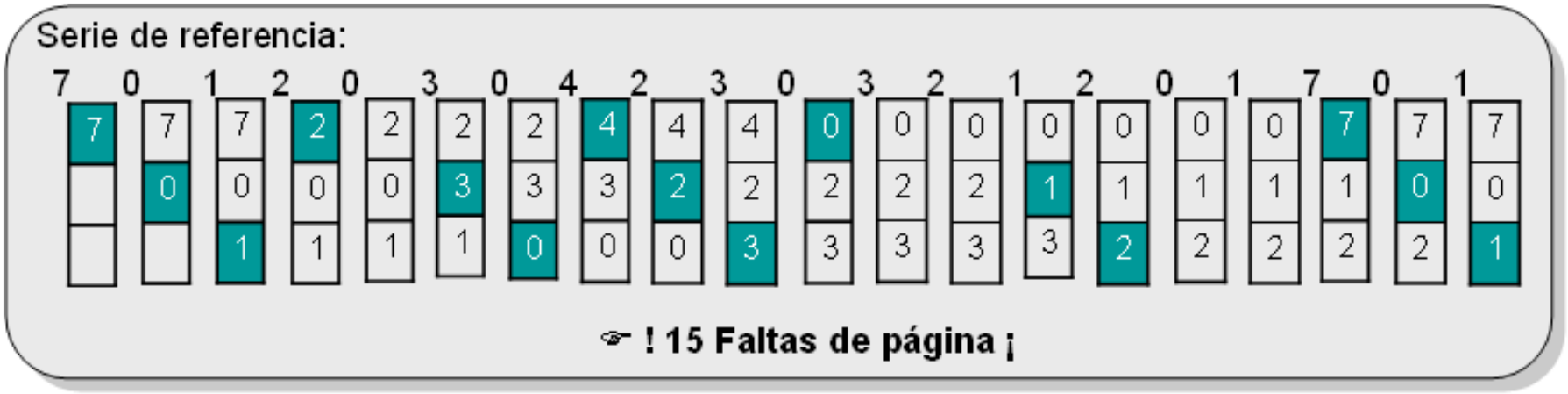
+ Implementación más o menos eficiente, desempeño aceptable.

Primero en entrar, primera en salir – FIFO

First-In, First-Out

“Reemplazar la página que lleva más tiempo en memoria”

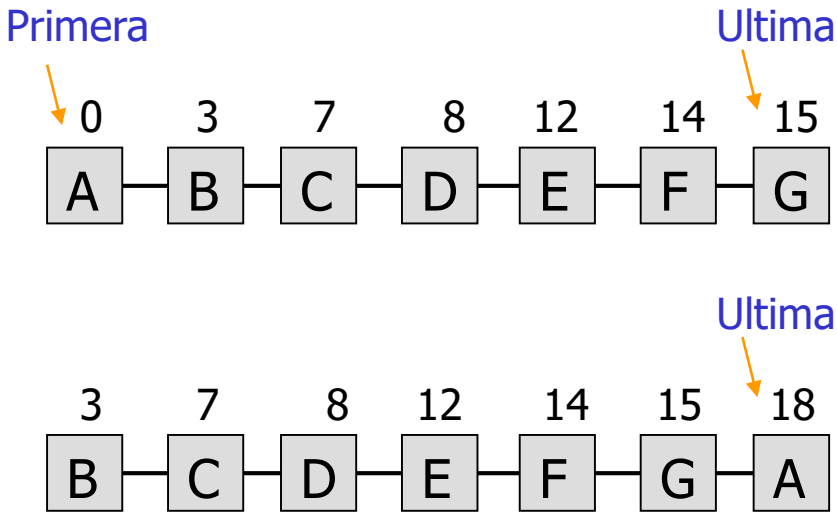
- Lista con todas las páginas en memoria
  - Una página que entra se añade al final
  - En el fallo de página se saca la página que está a la cabeza de la lista
- Puede producir muchas fallas de página



De Segunda oportunidad

“Reemplazar la página más antigua que no haya sido referenciada últimamente”

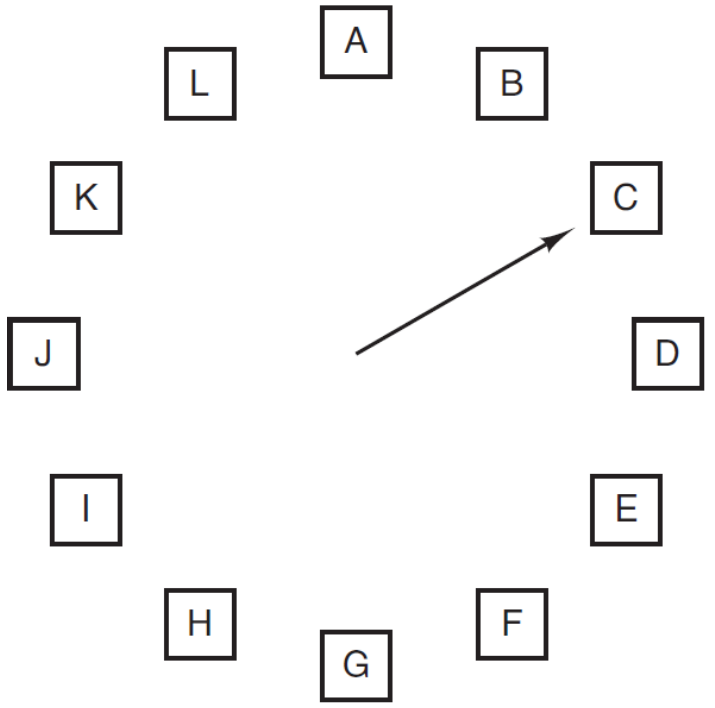
- Utiliza el bit de referencia R y una lista de las páginas cargadas.
  - Almacena el tiempo de referencia
- Recorre la lista,
  - Si R=0, intercambia la página.
  - Sino, la inserta al final de la lista y pone R=0



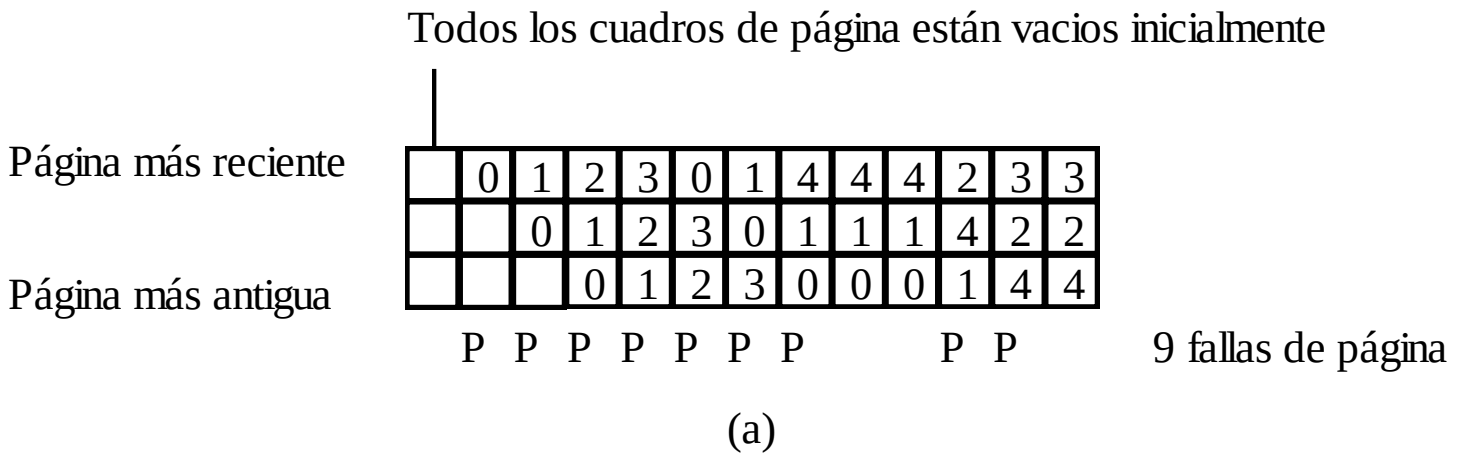
Reloj

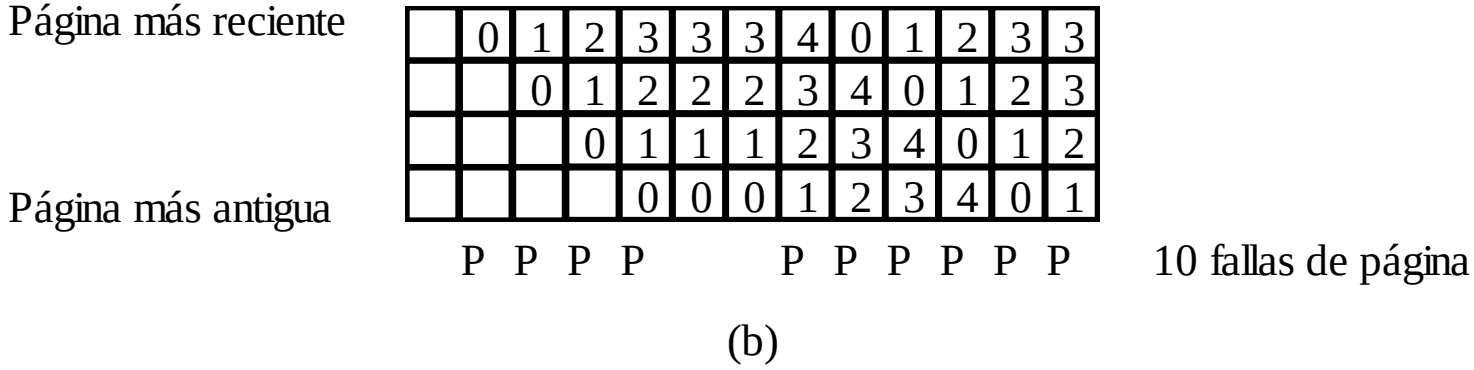
“Reemplazar la página más antigua que no haya sido referenciada últimamente”

- Más eficiente que “Segunda Oportunidad”
  - Los marcos de páginas se almacenan en una *lista circular*
- Cuando ocurre un fallo de página, si la página referenciada por la manecilla tiene
  - R=0, intercambia página
  - Sino, pone R=0



Anomalía de Belady





Página menos usada recientemente – LRU
Least Recently Used

“Reemplazar la página que tiene más tiempo sin usarse”

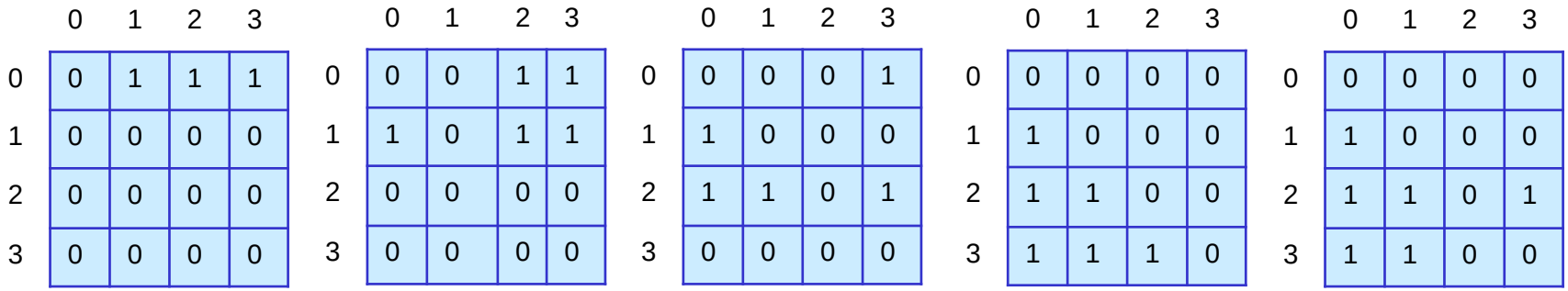
- Implementación: Usa lista de todas las páginas en memoria
  - Al principio: la usada más recientemente
  - Al final: la que lleve más tiempo sin usarse
  - Se actualiza con cada referencia a memoria. Se busca la página y se lleva al principio
  - Costoso

LRU: otra forma (I)

- Contador de 64 bits
  - Se incrementa después de cada instrucción
- Asociado a cada página:
  - un campo donde poder guardar el contador
  - cada vez que se accede a la página se actualiza al contador
- Con el fallo de página
  - Se examinan los contadores
  - Se sustituye la página con un contador menor

LRU: otra forma (II)

- Se implementa con ayuda de hardware:
  - N marcos de página
  - Matriz de NxN a 0
    - Cada vez que se accede al marco K
      - fila K a 1; columna k a 0
  - Lleva más tiempo sin utilizarse la página correspondiente a la fila de menor valor
  - P.e., si n=4 y la cadena de referencia es 0 1 2 3 2 1 0 3 2 3



Página no usada frecuentemente (NFU)

- Contador software asociado a cada página
  - con valor inicial 0
  - Con cada interrupción de reloj se suma al contador el bit R
  - lleva la cuenta del nº de pulsos en los que se ha hecho referencia
- Se sustituye la que tenga el contador más bajo
- Problema: nunca olvida
  - Puede ser que al principio de un programa se utilicen algunas páginas muy frecuentemente y luego ya no se utilicen.

Algoritmo de envejecimiento

- Simulación software de LRU
  - Antes de sumar R, los contadores se desplazan a la derecha
  - El bit R se suma al bit más a la izquierda
  - Se sustituye la página con contador más pequeño

Paginación por demanda

- Estrategia de paginación por petición
  - Se arrancan los procesos sin ninguna página en memoria
  - Cuando se inicia, se produce un fallo de página
  - Transcurrido cierto tiempo, el proceso tiene en memoria todas las páginas que le hace falta

Algoritmo del Conjunto de Trabajo

- Método del conjunto trabajo
  - **Localidad de referencias:**
    - en una fase de la ejecución de un programa, sólo se hace referencia a un conjunto pequeño de sus páginas
  - **Conjunto de trabajo**
    - El conjunto de páginas que utiliza un proceso en un momento dado
  - **Hiperpaginación (thrasing):**
    - Si se produce un fallo de página cada pocas instrucciones
  - **Prepaginación:** carga de páginas antes de empezar
    - El S.O. se encarga de que el conjunto de trabajo quepa en memoria reduciendo el índice de multiprogramación si hace falta
    - El S.O. lleva la cuenta de los conjuntos de trabajos

Algoritmo Conjunto de Trabajo

- Utiliza las necesidades actuales de memoria para determinar el número de marcos a asignar.
- Reduce la cantidad de fallos de página
  - Desalojar una página que no este en el conjunto de trabajo
- Objetivo
  - Mantener en memoria las páginas que forman el conjunto de trabajo
- **Implementación?.....**

Políticas de asignación global frente a local

- Local
  - Se busca la página a ser sustituida entre las del proceso
  - Se asigna una cantidad fija de memoria
  - Si el conjunto de trabajo disminuye, se desperdicia memoria
- Global
  - Se busca la página a ser sustituida entre todas
  - Se debe decidir cuanto asignar a cada proceso
    - Si el Conjunto de trabajo crece, incrementar el número de marcos
    - Si decrece, disminuir la cantidad de marcos.

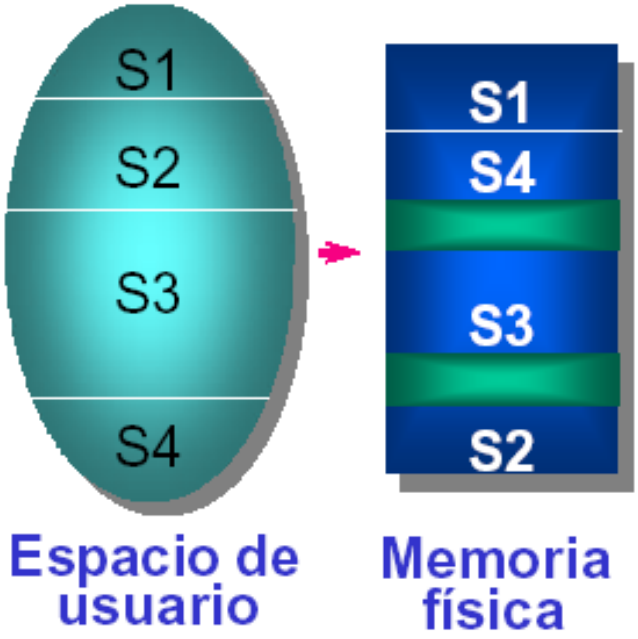
Tamaño de página

- Si grandes
  - Se desperdicia en promedio la mitad de una página por proceso, por fragmentación interna
  - Más espacio se desperdicia
  - Más porción de programa en memoria sin utilizar
  - Tablas de páginas más pequeñas
- Si pequeñas
  - Tablas de páginas más grandes
  - Menor fragmentación interna

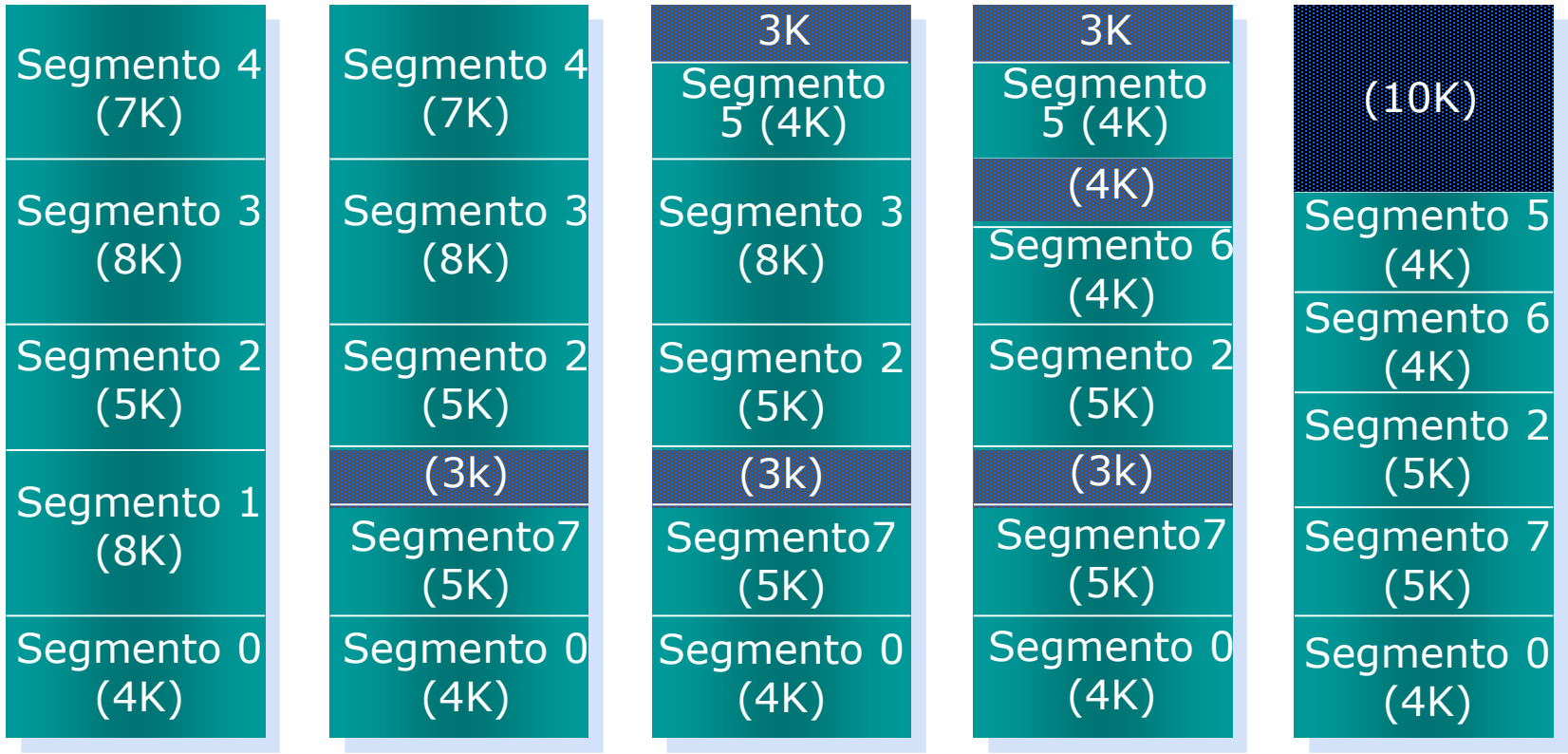
SEGMENTACIÓN

Segmentación

- Los procesos se dividen en bloques de tamaño variable
  - Segmentos
  - Entidad lógica
    - Se corresponden con la visión lógica del usuario
- La memoria física se reparte entre los segmentos de manera similar a lo visto para las particiones variables
- En memoria, algunos segmentos



Segmentación



Segmentación

El Programador

No se preocupa de referenciar segmentos sino sus objetos: módulos, rutinas, variables, etc.

El Compilador

Se ocupa de agrupar convenientemente los objetos del programador en segmentos, y de referenciarlos como **<segmento, desplazamiento>**

El Cargador

Se encarga de asignar un número a cada uno de los segmentos de que consta el programa a ejecutar

Estructura de datos

- Tabla de segmentos por proceso
  - Direcciones base y límite
  - Bits de protección
  - Bit de modificación
  - Bit de presencia
  - Bit de referencia
- Tabla del mapa del archivo
  - Direcciones en almacenamiento secundario de los segmentos del proceso
- Lista de huecos libres en memoria
  - Fragmentación externa

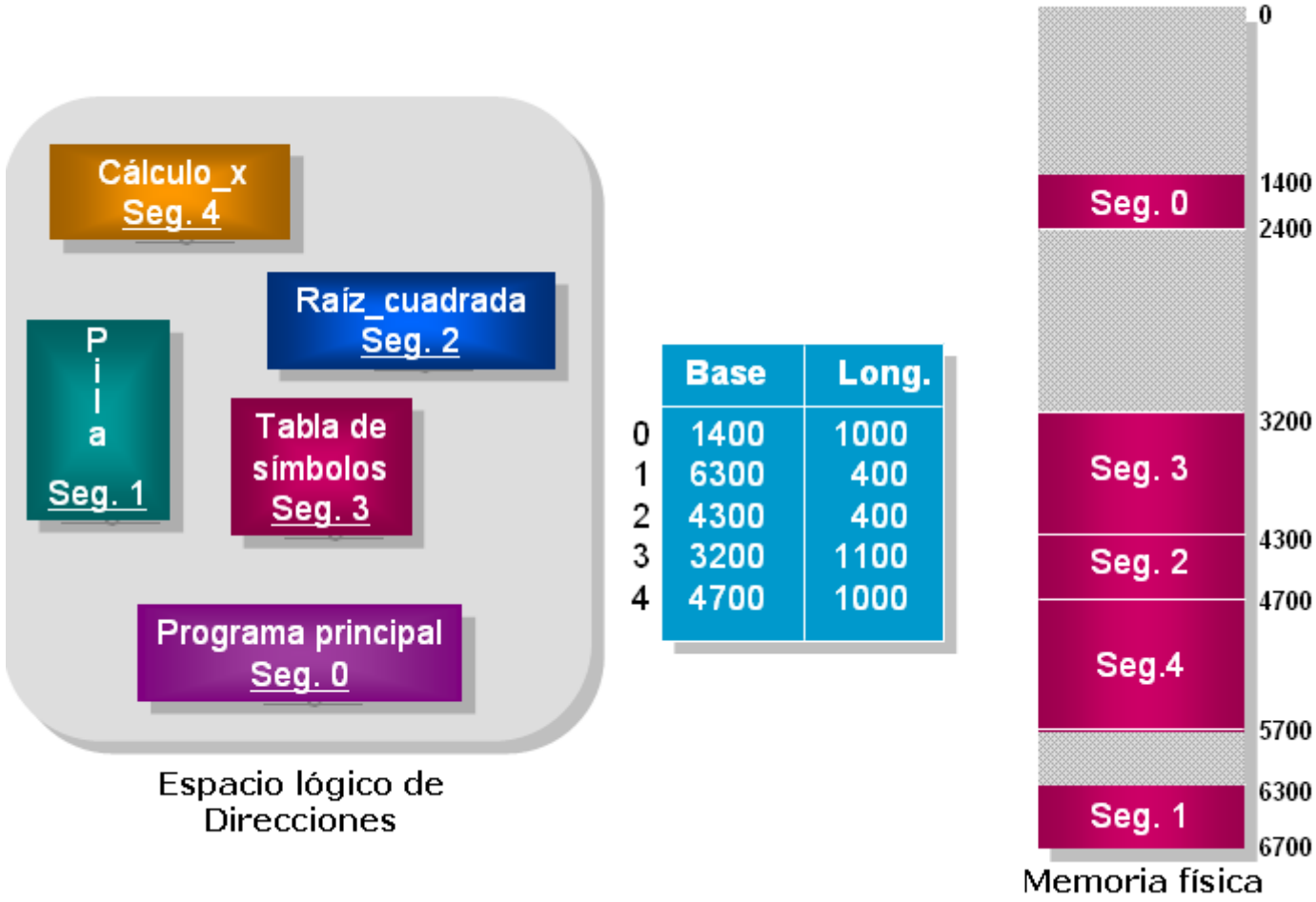
Traducción de direcciones

- Referencias a memoria:

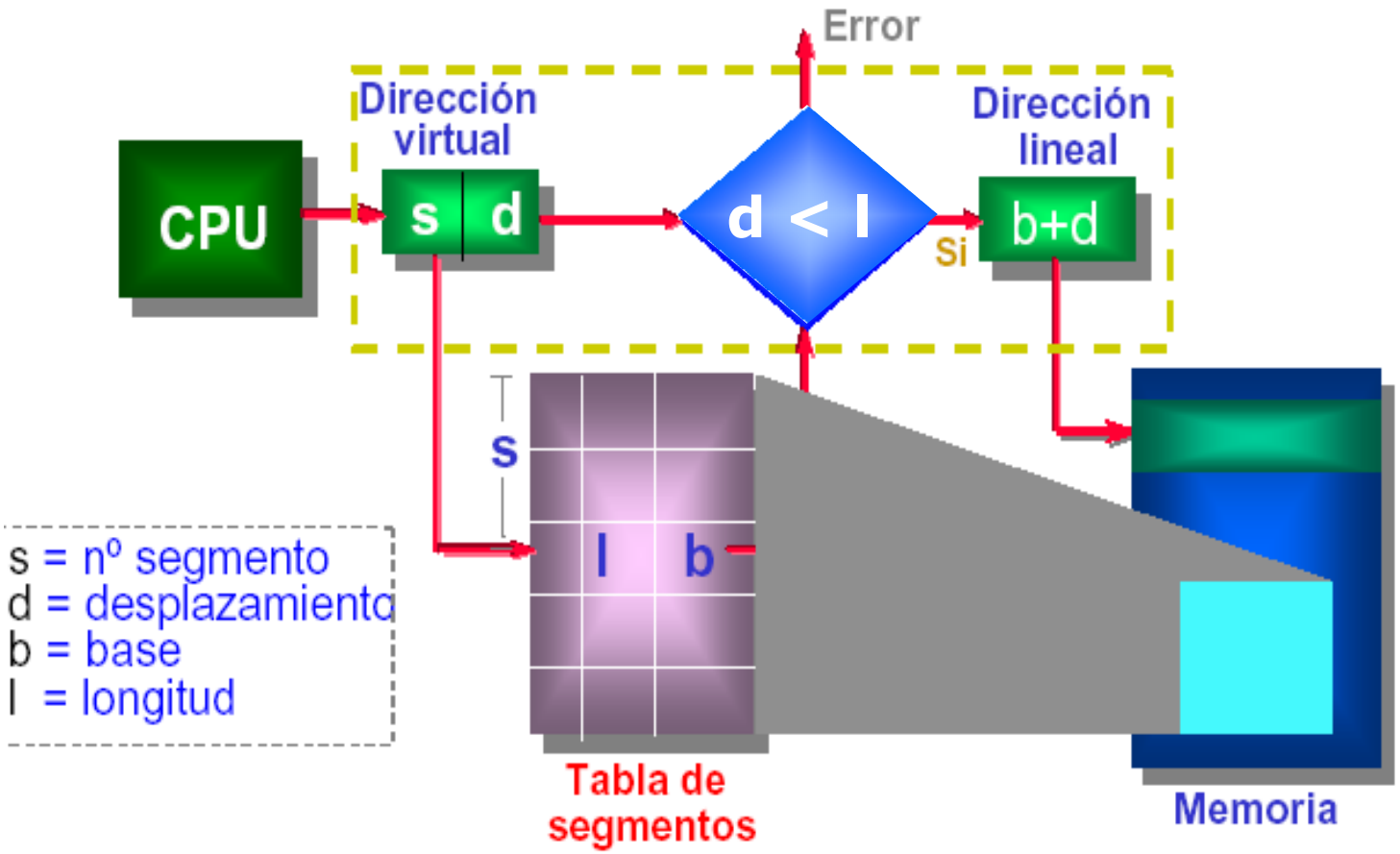
  Nº Segmento + Desplazamiento

- **Tabla de segmentos**: aplica direcciones bidimensionales definidas por el usuario en direcciones físicas de una dimensión
- Cada entrada de la tabla
  - **Base**: dirección física donde reside el inicio del segmento en memoria
  - Límite: longitud del segmento

Traducción de direcciones

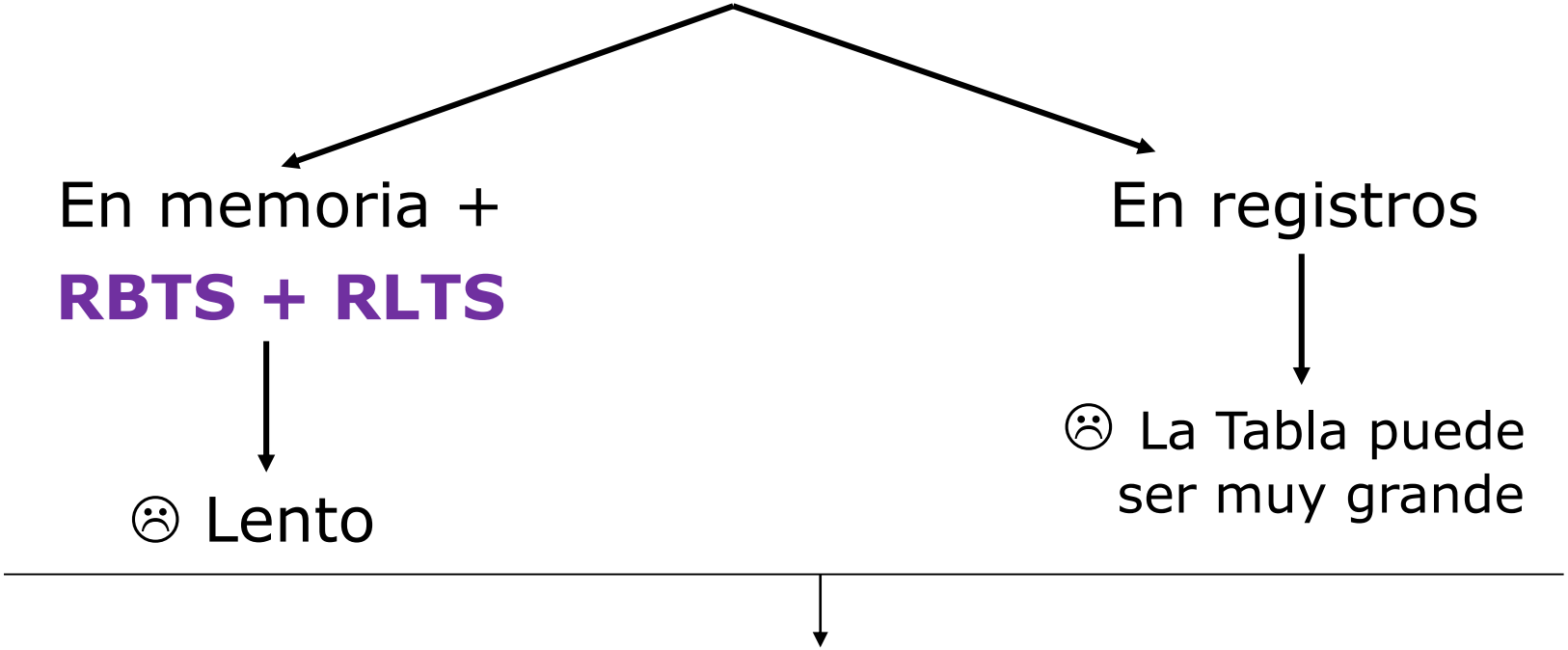


Traducción de direcciones



Implementación de la TS

- ¿Dónde poner la tabla de segmentos?



Implementación de la TS

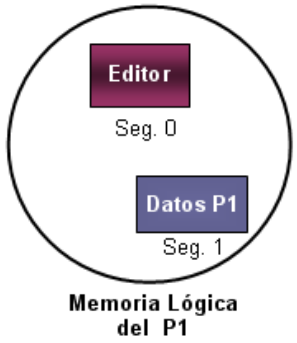
- Para acceder a la TS de un proceso se requiere
  - Registro Base de la Tabla de Segmentos (RBTS)
    - Direcciona a la posición inicial en memoria de la tabla segmentos
  - Registro Longitud de la Tabla de Segmentos (RLTS)
    - Indica el número de segmentos utilizados por el programa.
    - El número de segmento s es válido si s < RLTS

Ventajas vs. desventajas

+ Protección

Seg.	Base	Limite	Permisos
0	43062	25286	x x x
1	90003	8550	r - -

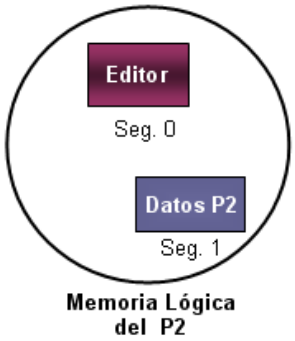
+ Compartición



Memoria Lógica del P1

Límite	Base
25286	43062
4425	68349

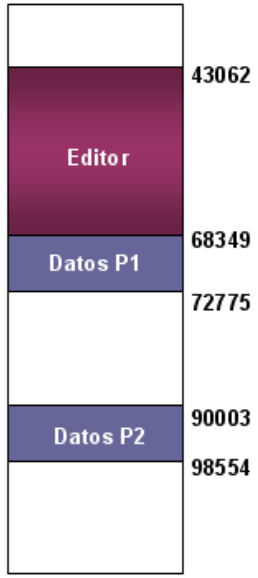
Tabla de segmentos de P1



Memoria Lógica del P2

Límite	Base
25286	43062
8550	90003

Tabla de segmentos de P2



Ventajas y desventajas

- Para cargar y ejecutar un programa se le debe asignar memoria libre para sus segmentos de memoria
  - Segmentos de memoria de tamaño variables
    - FRAGMENTACION EXTERNA

- ¿ Si tamaño de segmento > al tamaño de la memoria física ?

Segmentación paginada

- Solución de la fragmentación externa
- Un segmento es un espacio lineal de direcciones que puede ser paginado
  - Los procesos se dividen en segmentos y éstos en páginas
  - La memoria física se divide en marcos de página
- Estructuras de datos
  - Tabla de segmentos por proceso
  - Tabla de páginas por segmento
  - Mapa del archivo
  - Tabla de marcos de página

Dirección virtual

Nº Seg. + Nº Pag. en Seg. + desplazamiento en pág.

Esquema de traducción en Segmentación