MEMORIA VIRTUAL

UNIDAD 9

UNIDAD 9. Memoria virtual

- 9.1. Introducción.
- 9.2. Paginación por demanda.
- 9.3. Reemplazo de páginas.
- 9.4. Algoritmos de reemplazo de páginas.
- 9.5. Asignación de marcos.
- 9.6. Hiperpaginación (Thrashing).
- 9.7. Segmentación por demanda.
- 9.8. Segmentación con paginación virtual

Memoria Virtual

- Permite separar la memoria lógica del usuario de la memoria física.
- Un proceso en ejecución no tiene porque encontrarse totalmente en memoria principal (sólo una parte)
- Un proceso puede ser mayor que la memoria física.
- Permite transferencia de información entre MP y MS
- Usa un dispositivo de almacenamiento secundario (disco) como dispositivo de intercambio
- Puede implementarse sobre Paginación o Segmentación o Segmentación Paginada
- La transferencia suele ser bajo demanda.

Memoria Virtual: Paginación por Demanda

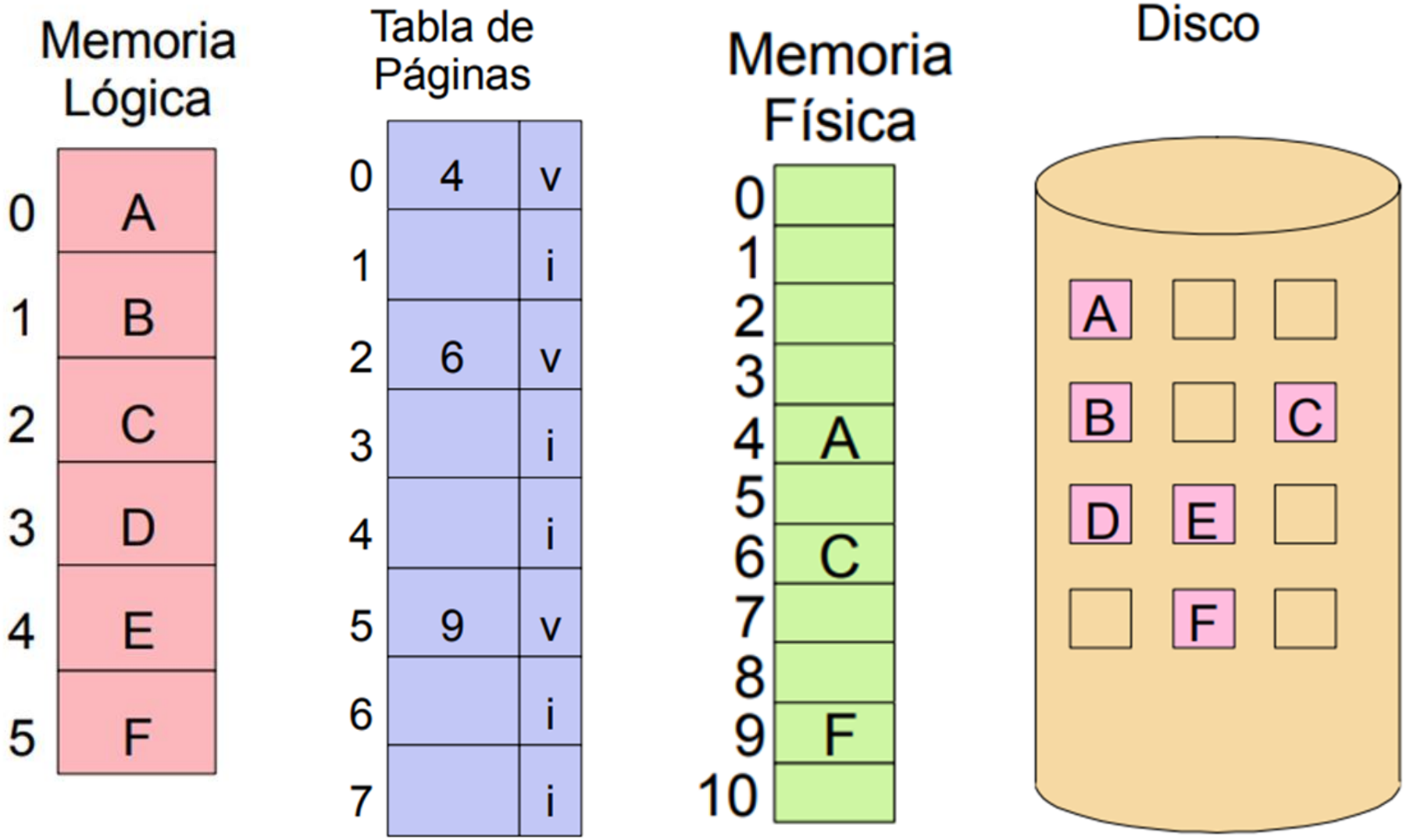
- Las paginas son cargadas por demanda.
- No se llevan páginas del almacenamiento secundario al primario hasta que son referenciadas explícitamente por el proceso en ejecución.
- Las razones del atractivo de esta estrategia son:
  - Los resultados de computabilidad, en especial el “problema de parada”, indican que el camino que tomará la ejecución de un programa no se puede predecir con exactitud.
  - Garantiza que solo las páginas que necesita el proceso sean traídas al almacenamiento principal.
  - La sobrecarga de proceso para decidir qué página traer al almacenamiento principal es mínima.
- El principal inconveniente está en los procesos que requieren acumular sus páginas una por una:
  - Los tiempos de espera de páginas son considerables.
  - Es creciente la cantidad de almacenamiento primario afectada al proceso que espera páginas, por lo que el “producto espacio - tiempo” se incrementa.
  - El “producto espacio - tiempo” indica la cantidad de almacenamiento que usa un proceso y la cantidad de tiempo que lo usa.

Memoria Virtual: Paginación Anticipada

- El S. O. intenta predecir las páginas que un proceso va a necesitar y a continuación **precarga** estas páginas cuando hay espacio disponible

- Mientras el proceso ejecuta sus páginas actuales, el sistema carga páginas nuevas que estarán disponibles cuando el proceso las necesite, debido a ello, el tiempo de ejecución de un proceso se puede reducir.

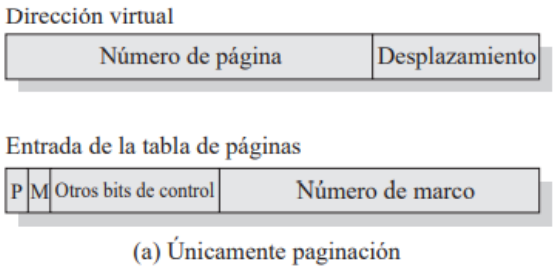
Memoria Virtual: Paginación



Memoria Virtual: Tabla de Páginas

- Cada proceso dispone de su propia tabla de páginas.
- Cada entrada en la tabla de páginas es más compleja que en la Paginación Simple, pues solo algunas de las páginas de un proceso se encuentran en MP, lo cual debe indicarse y si estuviera presente, especificar el número de marco correspondiente.
- La entrada de la tabla de páginas incluye un bit de modificado (M) que indica si los contenidos de la misma han sido alterados desde su carga. Si no hay cambios no se requiere escribir la página al momento del reemplazo.
- Pueden existir otros bits de control, como Protección y Compartición

Memoria Virtual: Hardware de Traducción



(a) Únicamente paginación

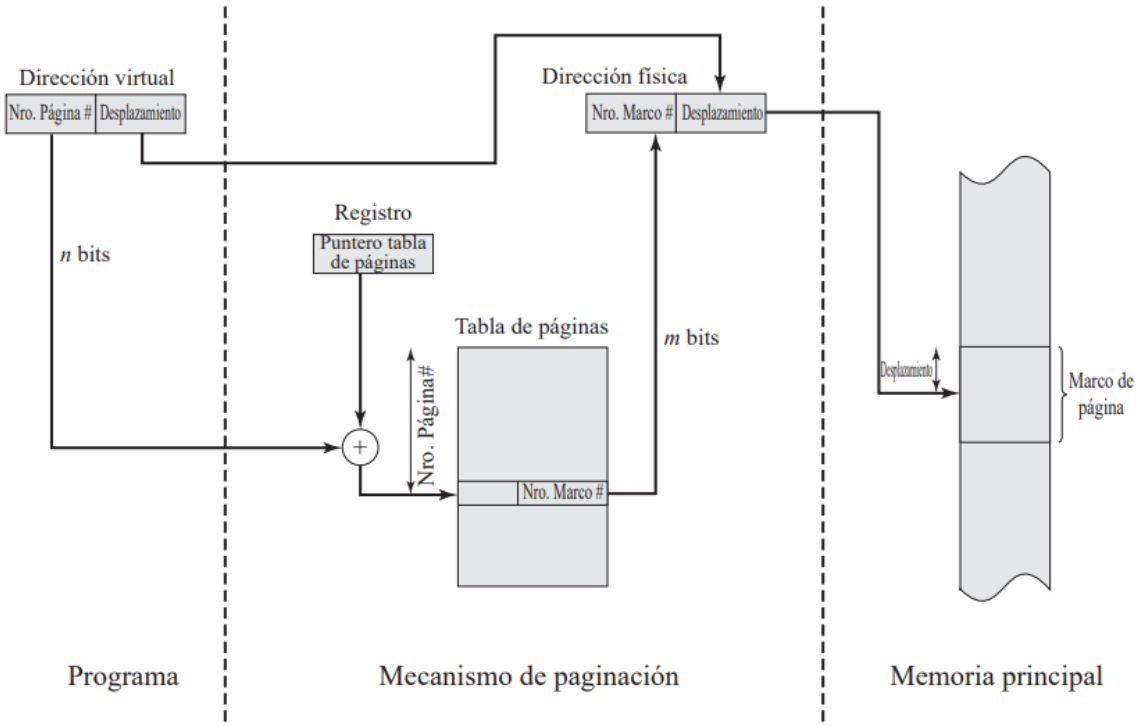


Figura 8.3. Traducción de direcciones en un sistema con paginación.

Paginación Jerárquica

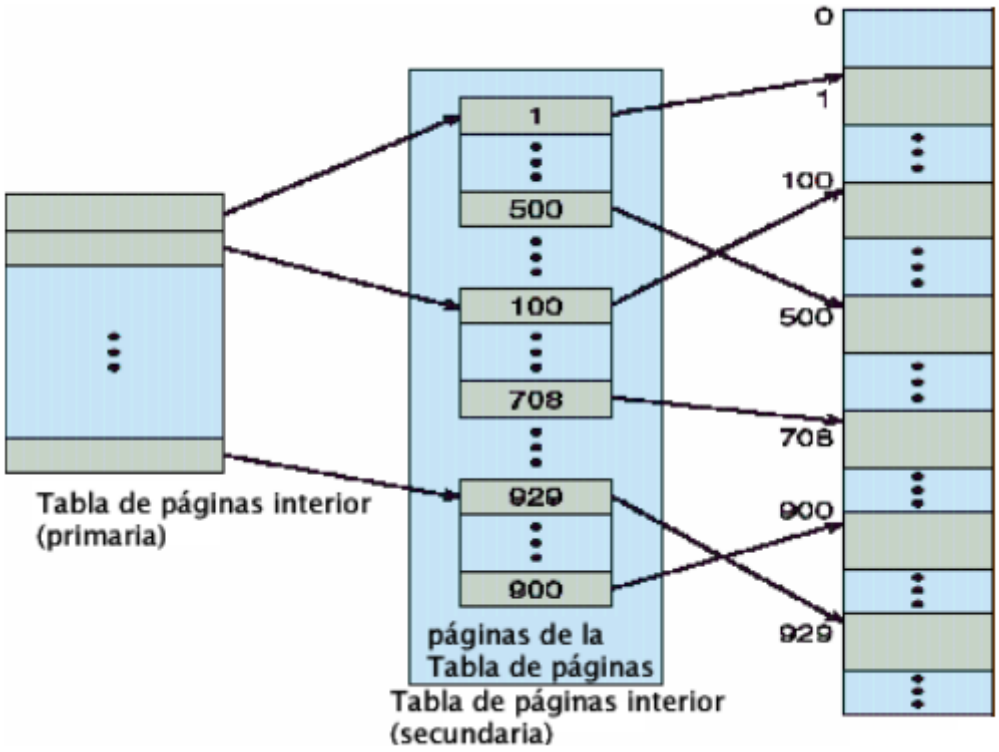
- Para evitar el problema de tener **tablas de páginas inmensas** en la memoria todo el tiempo, es posible **dividir la tabla de páginas** en fragmentos más pequeños, lo que se puede llevar a cabo de varias formas distintas.

- Una de esas formas consiste en utilizar un **algoritmo de paginación de dos niveles**, en el que la propia **tabla de páginas** está también **paginada**.

- Se forma una **jerarquía de niveles de paginación**.

- Al tener distintas tablas de páginas se seleccionan las que necesitan estar en memoria principal en un momento determinado.

Paginación Jerárquica



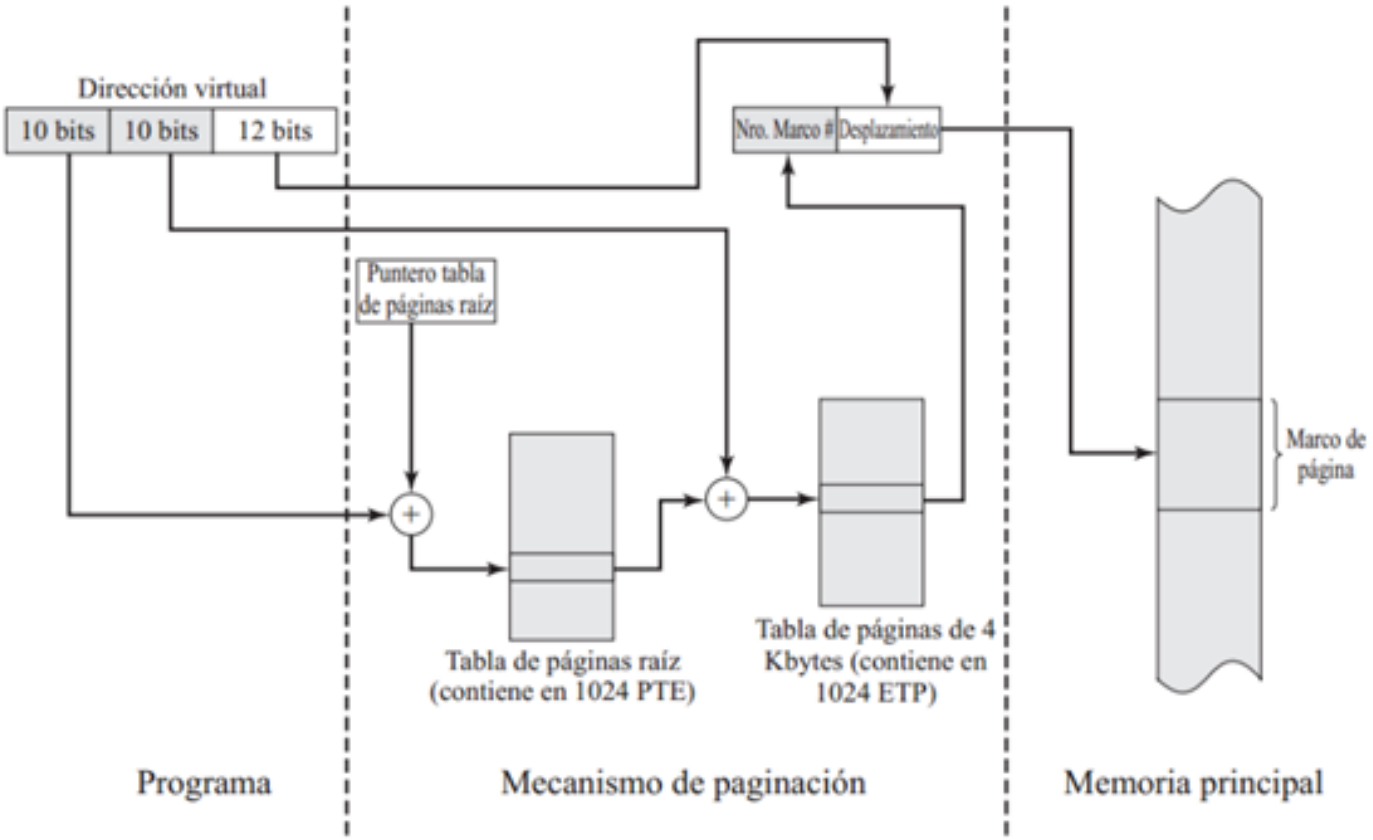
Paginación Jerárquica

- La dirección virtual constará de una tupla:

(tp1, tp2,..., tpn, despl)

- donde
  - ***tp1*** se utiliza como índice en la Tabla de Páginas de primer nivel, y localiza la tabla de páginas de segundo nivel
  - ***tp2*** se utiliza como índice en la tabla de páginas de segundo nivel, y localiza la tabla de páginas de tercer nivel
  - ....
  - ***tpn*** se utiliza como índice en la tabla de páginas de nivel n, y localiza el número de marco de la página indicada por ***tpn***.

Paginación Jerárquica: 2 niveles



Traducción de direcciones en un sistema de paginación de dos niveles.

Tabla De Paginas Invertida

- Se mantiene una única tabla de páginas ordenada por marcos.
- Se aplica cuando el tamaño de la memoria virtual es grande y no se podría representar mediante tablas de página comunes.
- La búsqueda se realiza secuencialmente por **id-proceso** y luego por **página**

Tabla De Paginas Invertida

Entradas a la tabla de páginas: (indizada por nro de marco)

Id-proc	Nro-pag	Referencia	Modificado

Dirección de carga de la tabla de páginas invertida:

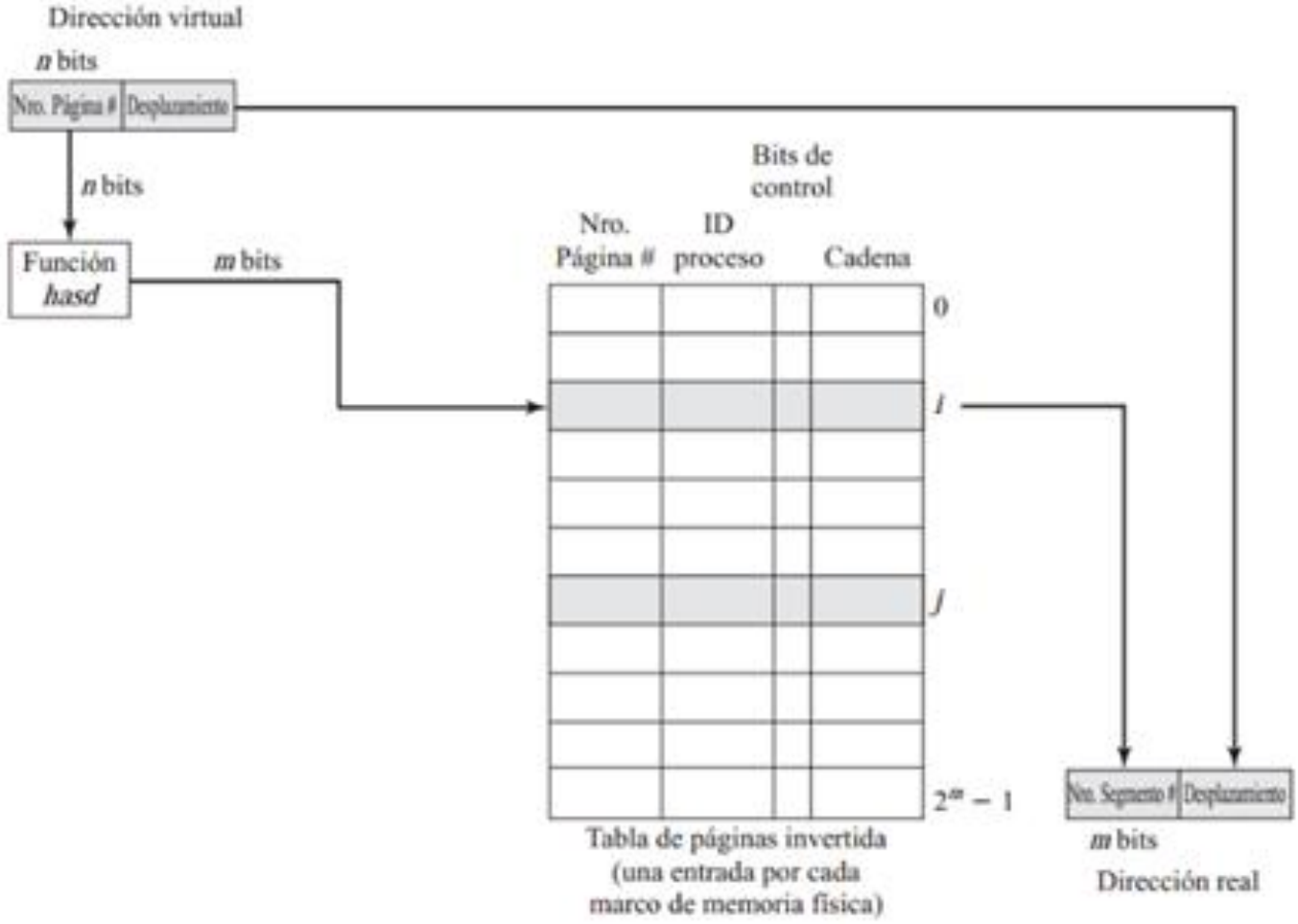
BCS

Estructura de la dirección virtual:

Id-proc	Pag	Despl

Dirección física: **Marco * Tamaño + Despl**

TABLA DE PÁGINAS INVERTIDA



Estructura de tabla de páginas invertida.

TABLA DE PAGINAS INVERTIDA

Sea la siguiente tabla:

	Id-Proc	Pag	Referencia	Modificado
0	1	1	1	0
1	0	0	1	0
2	2	0	0	0
3	0	2	1	1
4	1	2	0	0
5	1	7	1	0
6	2	1	1	1
7	2	15	1	1

Proceso 0: 4 Páginas
Pag 0 → Marco 1
…
Pag 2 → Marco 3
…

Proceso 1: 8 Páginas
Pag 0 → Marco 0
…
Pag 2 → Marco 4
…
Pag 7 → Marco 5

Proceso 2: 16 Páginas
Pag 0 → Marco 2
Pag 1 → Marco 6
…
Pag 15 → Marco 7

Las páginas tienen un tamaño de 512 Kb.

Sea la Dir. Lógica: (1,2,300) ⟶ **Dir. Física:** 4 * 512 + 300 = 2348

Buffer de Traducción Anticipada

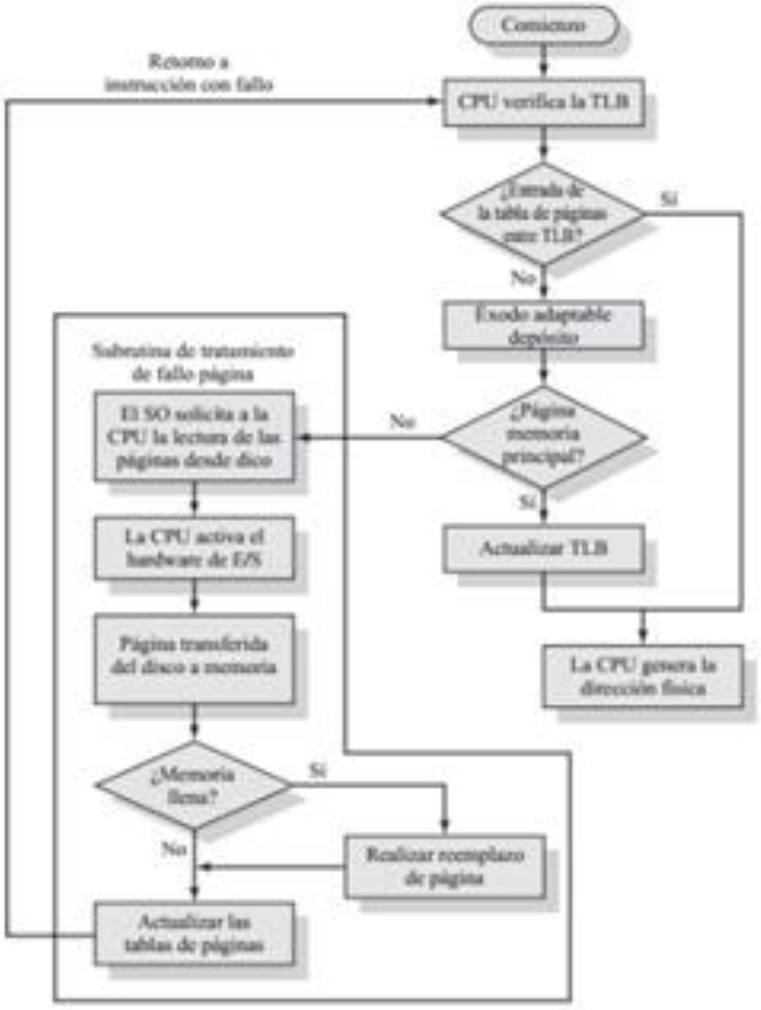
- Se utiliza un pequeño y rápido caché especial de memoria asociativa, llamada *Tranlation Look-side Buffer* (TLB).

- El TLB forma parte de la MMU y contiene los pares (página, marco) de las páginas mas recientemente usadas.

Buffer De Traducción Anticipada

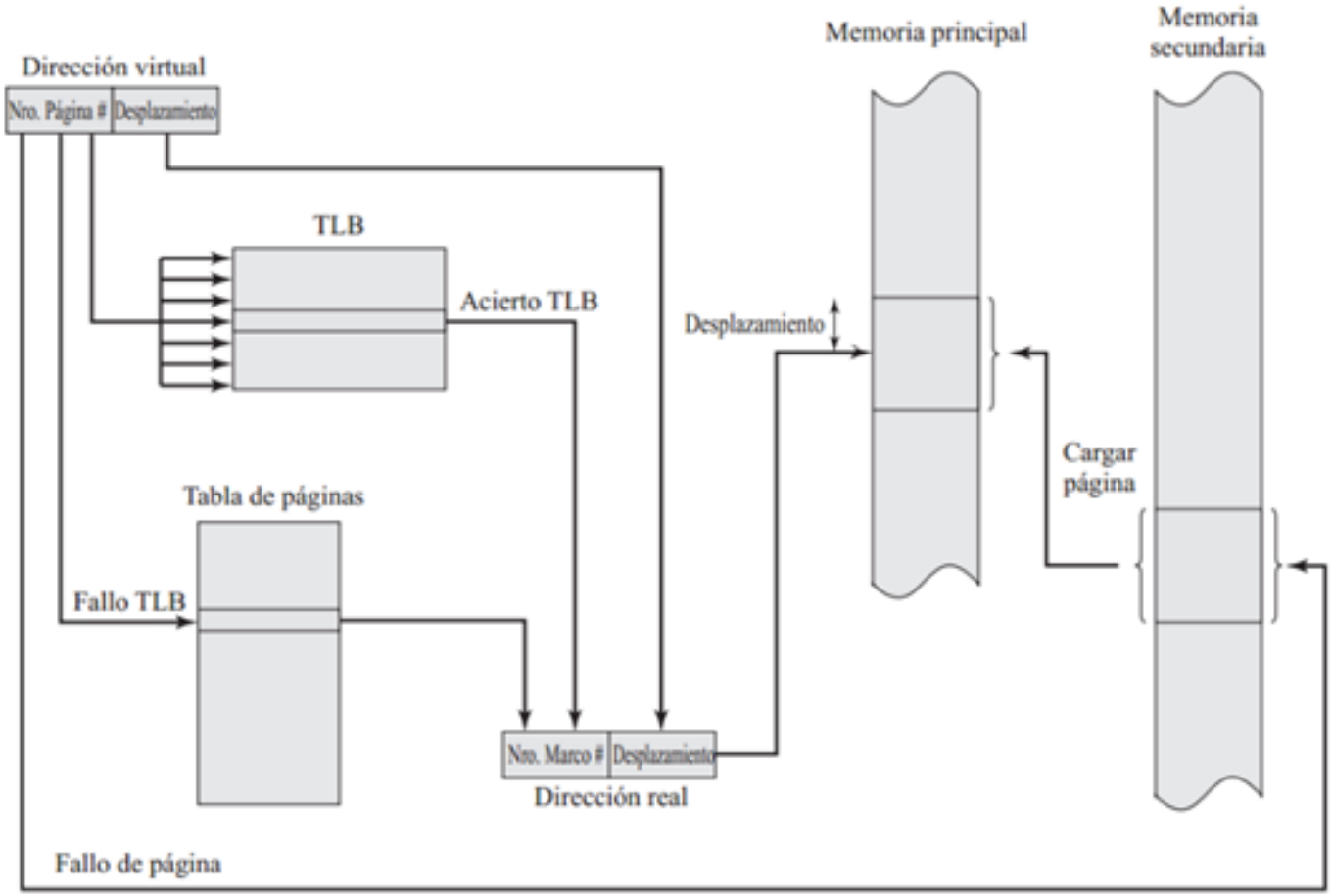
- Cuando la CPU genera una dirección lógica a la página p, la MMU busca una entrada (**p,m**) en la TLB.
- Si está, se usa el marco m sin acudir a RAM.
- Si no hay una entrada para p, la MMU debe acceder a la tabla de páginas en RAM, e incorporar una entrada para p en el TLB, posiblemente eliminando a otra.

BUFFER DE TRADUCCIÓN ANTICIPADA



Operación de paginación y TLB [FURH87].

BUFFER DE TRADUCCIÓN ANTICIPADA



Uso de la TLB.

Buffer De Traducción Anticipada

- Por pequeño que sea el TLB, la probabilidad de que la página esté en él es alta porque los programas suelen hacer muchas referencias a pocas páginas.

Estrategias Para La Administración De Memoria Virtual

- ***"Estrategias de búsqueda"***
  - Tratan de los casos en que una página o segmento deben ser traídos del almacenamiento secundario al primario.
  - Las estrategias de *"búsqueda por demanda"* esperan a que se haga referencia a una página o segmento por un proceso antes de traerlos al almacenamiento primario.
  - Los esquemas de *"búsqueda anticipada"* intentan determinar por adelantado a qué páginas o segmentos hará referencia un proceso para traerlos al almacenamiento primario antes de ser explícitamente referenciados.

- ***"Estrategias de colocación"***
  - Tratan del lugar del almacenamiento primario donde se colocará una nueva página o segmento.
  - Los sistemas toman las decisiones de colocación de una forma trivial ya que una nueva página puede ser colocada dentro de cualquier marco de página disponible.

- ***"Estrategias de reposición"***
  - Tratan de la decisión de cuál página o segmento desplazar para hacer sitio a una nueva página o segmento cuando el almacenamiento primario está completamente comprometido.

ALGORITMOS DE REEMPLAZO DE PÁGINAS

Estrategias de Reposición (Reemplazo)

Las principales son:

1. Reposición de páginas al azar.
2. Primero en entrar - primero en salir. (FIFO)
3. Algoritmo del Reloj (FIFO Optimizado)
4. Optimo
5. Menos recientemente usada. (LRU)
6. No usada recientemente. (NRU)
7. Menos frecuentemente usada. (NFU)

Estrategias de Reposición: FIFO

- Se registra el momento en que cada página ingresa al almacenamiento primario.
- Para reemplazar una página, se selecciona aquella que ha estado más tiempo almacenada.
- Se presenta el inconveniente de que se pueden reemplazar páginas muy usadas, que serán llamadas de nuevo al almacenamiento primario casi de inmediato.
- Se puede presentar la llamada *"anomalía FIFO":*

  Belady, Nelson y Shedler descubrieron que con la reposición FIFO, ciertos patrones de referencias de páginas causan más fallos de páginas cuando se aumenta el número de marcos (celdas) de páginas asignados a un proceso: en esto consiste la *"anomalía FIFO".*

Estrategias de Reposición: FIFO

Con 3 marcos: 9 fallos de página

A	B	C	D	A	B	E	A	B	C	D	E
A	**B**	**C**	**D**	**A**	**B**	**E**	**E**	E	**C**	**D**	D
	A	B	C	D	A	B	B	**B**	E	C	C
		A	B	C	D	A	A	A	B	E	**E**
*	*	*	*	*	*	*	✓	✓	*	*	✓

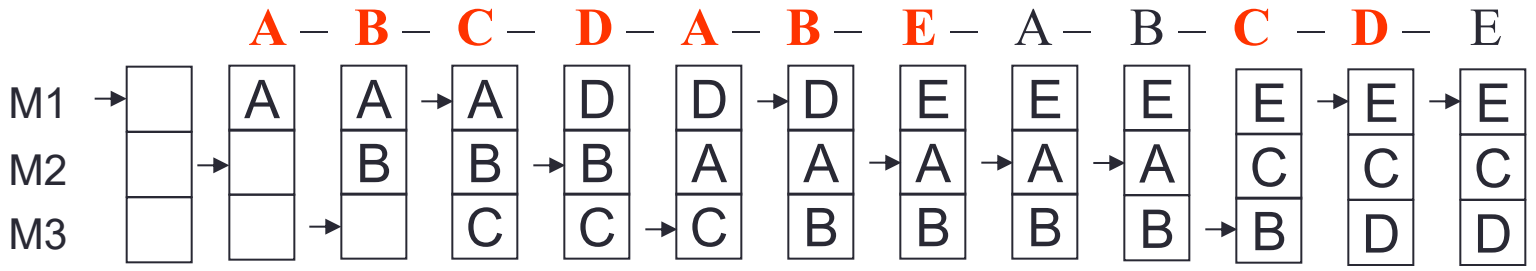
Con 4 marcos: 10 fallos de página

A	B	C	D	A	B	E	A	B	C	D	E
A	**B**	**C**	**D**	D	D	**E**	**A**	**B**	**C**	**D**	**E**
	A	B	C	C	C	D	E	A	B	C	D
		A	B	B	**B**	C	D	E	A	B	C
			A	**A**	A	B	C	D	E	A	B
*	*	*	*	✓	✓	*	*	*	*	*	*

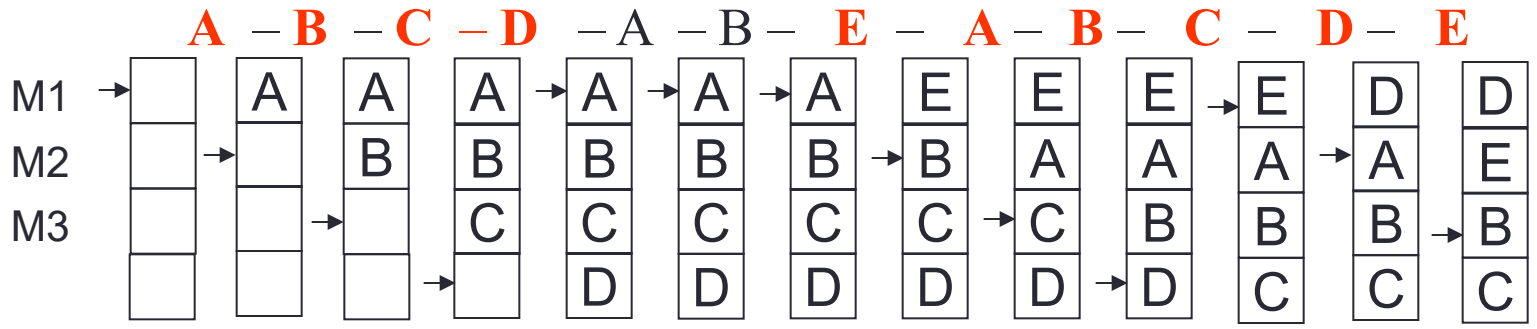
Estrategias de Reposición: FIFO

Referencias: A – B – C – D – A – B – E – A – B – C – D – E

Con 3 marcos: **9 fallos de página**



Con 4 marcos: **10 fallos de página**



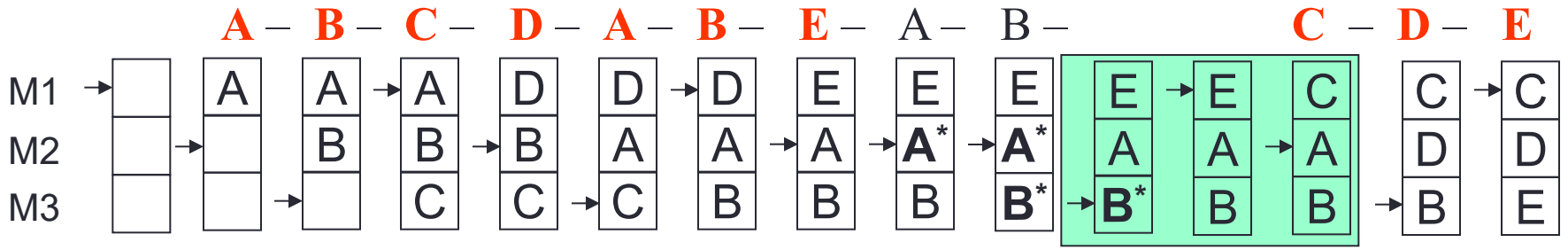
Estrategias de Reposición: Reloj

- Es una modificación sobre el funcionamiento del algoritmo anterior (FIFO).
- Registrado el tiempo de ingreso al almacenamiento primario, si la página es referenciada nuevamente, su bit de referencia se coloca en 1.
- Un puntero indica cual es la próxima página a desalojar (tiempo de carga mas antiguo).
- Al ocurrir un fallo de página, si la página apuntada por el puntero tiene el bit R a 0, la página se reemplaza, si R es 1, se lo pone a 0 y el puntero pasa a la siguiente página más antigua. El proceso continúa hasta encontrar una página con R = 0.
- Mejor de los casos: La página apuntada inicialmente tiene su bit R = 0
- Peor de los casos: Todas las páginas tienen el bit R = 1, recorriéndolas y volviendo a la página inicialmente apuntada cuyo bit ya se hizo 0.

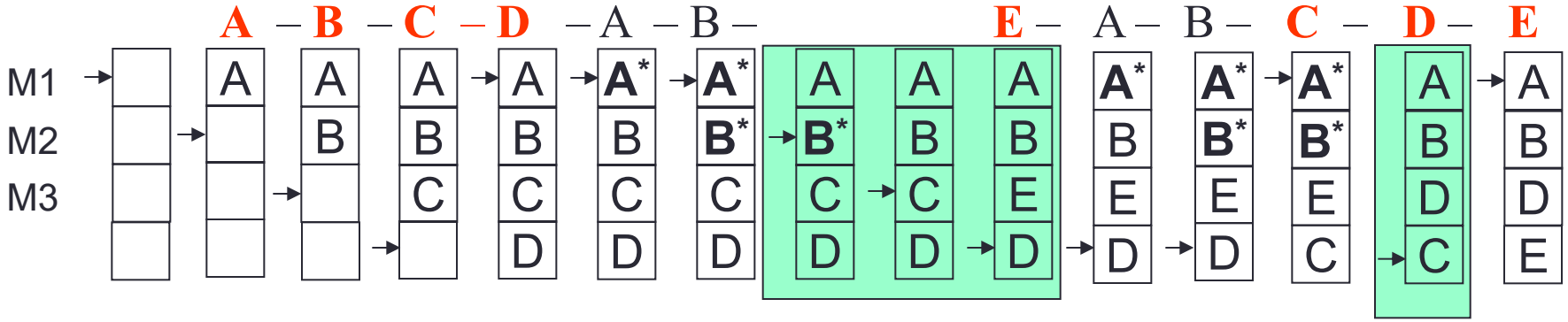
Estrategias de Reposición: Reloj

Referencias: A – B – C – D – A – B – E – A – B – C – D – E

Con 3 marcos: **10 fallos de página**



Con 4 marcos: **8 fallos de página**



Estrategia de Reposición: Optimo

- Ante un fallo de página se reemplaza la página que no se usará en el mayor período de tiempo
- Su desventaja reside en que solo puede aplicarse en secuencias conocidas de solicitudes de páginas, por lo que resulta imposible de implantar.
- Es utilizado en ambientes experimentales o de simulación.

Estrategia de Reposición: Optimo

Con 3 marcos: 7 fallos de página

A	B	C	D	A	B	E	A	B	C	D	E
A	**B**	**C**	**D**	D	D	**E**	E	E	E	E	**E**
	A	B	B	B	**B**	B	B	**B**	**C**	**D**	B
		A	A	**A**	A	A	**A**	A	A	A	A
*	*	*	*	✓	✓	*	✓	✓	*	*	✓

Con 4 marcos: 6 fallos de página

A	B	C	D	A	B	E	A	B	C	D	E
A	**B**	**C**	**D**	D	D	**E**	E	E	E	E	**E**
	A	B	C	C	C	C	C	C	**C**	**D**	D
		A	B	B	**B**	B	B	**B**	B	B	C
			A	**A**	A	A	**A**	A	A	A	B
*	*	*	*	✓	✓	*	✓	✓	✓	*	✓

Estrategia de Reposición: LRU

- Dado un fallo de página se reemplazará aquella página que no haya sido utilizada en las últimas instancias.
- La definición de este algoritmo reside en la idea de que las páginas que no son muy usadas son las más adecuadas para ser reemplazadas.
- Para su implementación se puede utilizar contadores, pilas o matrices de hardware.

Estrategia de Reposición: LRU

La implementación con matrices consiste en:

- Definir una matriz de orden NxN, siendo N la cantidad de marcos
- Cuando una página se ubica en el marco i ($0 \leq i \leq N-1$) de memoria, se coloca 1 en la fila i y 0 en la columna i.
- Ante un fallo de página, se busca en la matriz aquella fila j ($0 \geq j \leq N-1$) con menor valor binario, correspondiente al marco j que contiene la página a desalojar y recibirá a la nueva página.

Estrategia de Reposición: NRU

- Se utiliza el estado de los bits R y M.
- Inicialmente estos bits están en 0. Si la página ha sido modificada su bit M se hace 1. Cada vez que la página es referenciada el bit R se hace 1.
- Cada x instantes de reloj, se blanquean los bits R de cada página cargada. Los bits M no se limpian, porque esta información es necesaria para saber si hay de que volver o no a escribir la página en el disco.
- Cuando ocurre un fallo de página se reemplaza la página cuyo par de bits (R,M) tiene el menor valor binario.

Estrategia de Reposición: NRU

- El S.O. reconoce 4 categorías según los valores de los bits R y M
  - Clase 0: sin referencia, sin modificación (0,0)
  - Clase 1: sin referencia, con modificación (0,1)
  - Clase 2: con referencia, sin modificación (1,0)
  - Clase 3: con referencia, con modificación (1,1)

- Ventajas:
  - Fácil de comprender
  - Implantación eficiente
  - Rendimiento muy adecuado (aunque no óptimo)

Estrategia de Reposición: NRU

- Sea la siguiente tabla de páginas del proceso A

	Presencia	Marco	Referencia	Modificado
0	0	-	-	-
1	1	2	1	1
2	0	-	-	-
3	1	1	1	1
4	0	-	-	-
5	1	0	1	0
6	0	-	-	-
7	1	3	0	1

El proceso hace referencia a la página 4, que no está cargada en memoria, se produce un fallo de página y el SO busca el menor valor binario compuesto por el par (R,M)

La página a reemplazar será la 7, la cual deberá devolverse a disco, ya que ha sido modificada

Estrategia de Reposición: NRU

- La tabla de páginas del proceso A, quedará con la siguiente información

	Presencia	Marco	Referencia	Modificado
0	0	-	-	-
1	1	2	1	1
2	0	-	-	-
3	1	1	1	1
4	**1**	**3**	**0**	**0**
5	1	0	1	0
6	0	-	-	-
7	**0**	-	-	-

Estrategia de Reposición: NFU

- El algoritmo hace uso del bit **R** y de **N** registros de 8 bits (1 por marco).
- Cada registro tiene un valor inicial 00000000
- Cuando termina una instancia de tiempo, se corren todos los bits de estos registros una posición a la derecha perdiéndose el último bit de la derecha, en el primer bits de la izquierda se coloca el valor del bit R asociado a cada una de las páginas cargadas en dichos marcos.
- Cuando ocurre un fallo de página se reemplaza la página del marco cuyo registro tiene el menor valor binario.

Estrategia de Reposición: NFU

- Sean 4 marcos cargados, inicialmente sus registros tienen los siguientes valores:

RM0	0	0	0	0	0	0	0	0
RM1	0	0	0	0	0	0	0	0
RM2	0	0	0	0	0	0	0	0
RM3	0	0	0	0	0	0	0	0

- Se realizan las siguientes referencias a las páginas cargadas en dichos marcos

A – B – C – D – A – B – E – A – B – C – D – E

Estrategia de Reposición: NFU

A – B – C – D – A – B – E – A – B – C – D – E

RM0	0	0	0	0	0	0	0	0
RM1	0	0	0	0	0	0	0	0
RM2	0	0	0	0	0	0	0	0
RM3	0	0	0	0	0	0	0	0

RM0	**1**	0	0	0	0	0	0	0
RM1	**0**	0	0	0	0	0	0	0
RM2	**0**	0	0	0	0	0	0	0
RM3	**0**	0	0	0	0	0	0	0

(A/0)

Estrategia de Reposición: NFU

t0 – (A/0)

RM0	1	0	0	0	0	0	0	0
RM1	0	0	0	0	0	0	0	0
RM2	0	0	0	0	0	0	0	0
RM3	0	0	0	0	0	0	0	0

t1-(B/1)

RM0	0	1	0	0	0	0	0	0
RM1	1	0	0	0	0	0	0	0
RM2	0	0	0	0	0	0	0	0
RM3	0	0	0	0	0	0	0	0

t2-(C/2)

RM0	0	0	1	0	0	0	0	0
RM1	0	1	0	0	0	0	0	0
RM2	1	0	0	0	0	0	0	0
RM3	0	0	0	0	0	0	0	0

t3-(D/3)

RM0	0	0	0	1	0	0	0	0
RM1	0	0	1	0	0	0	0	0
RM2	0	1	0	0	0	0	0	0
RM3	1	0	0	0	0	0	0	0

A – B – C – D – A – B – E – A – B – C – D – E

Estrategia de Reposición: NFU

t4-(A/0)

RM0	1	0	0	0	1	0	0	0
RM1	0	0	0	1	0	0	0	0
RM2	0	0	1	0	0	0	0	0
RM3	0	1	0	0	0	0	0	0

t5-(B/1)

RM0	0	1	0	0	0	1	0	0
RM1	1	0	0	0	1	0	0	0
RM2	0	0	0	1	0	0	0	0
RM3	0	0	1	0	0	0	0	0

t6-(E/2)

RM0	0	0	1	0	0	0	1	0
RM1	0	1	0	0	0	1	0	0
RM2	1	0	0	0	0	0	0	0
RM3	0	0	0	1	0	0	0	0

t7-(A/0)

RM0	1	0	0	1	0	0	0	1
RM1	0	0	1	0	0	0	1	0
RM2	0	1	0	0	0	0	0	0
RM3	0	0	0	0	1	0	0	0

A – B – C – D – A – B – **E** – A – B – C – D – E

Estrategia de Reposición: NFU

t8-(B/1)

RM0	0	1	0	0	1	0	0	0
RM1	1	0	0	1	0	0	0	1
RM2	0	0	1	0	0	0	0	0
RM3	0	0	0	0	0	1	0	0

t9-(C/3)

RM0	0	0	1	0	0	1	0	0
RM1	0	1	0	0	1	0	0	0
RM2	0	0	0	1	0	0	0	0
RM3	1	0	0	0	0	0	0	0

t10-(D/2)

RM0	0	0	0	1	0	0	1	0
RM1	0	0	1	0	0	1	0	0
RM2	1	0	0	0	0	0	0	0
RM3	0	1	0	0	1	0	0	0

t11-(E/0)

RM0	1	0	0	0	0	0	0	0
RM1	0	0	0	1	0	0	1	0
RM2	0	1	0	0	0	0	0	0
RM3	0	0	1	0	0	1	0	0

A – B – C – D – A – B – **E** – A – B – C – **D** – **E**

MODELACIÓN DE ALGORITMOS DE PAGINACIÓN

Modelación de algoritmos de paginación

Anomalía de Belady

- Caso donde agregar marcos a la memoria de un proceso aumenta la cantidad de fallos de página, en lugar de disminuirlos.
- Ej: sean 2 sistemas con reemplazo de páginas por FIFO con 3 y 4 marcos respectivamente, con los accesos a páginas: 0, 1, 2, 3, 0, 1, 4, 0, 1, 2, 3, 4.

Secuencia de acceso	0	1	2	3	0	1	4	0	1	2	3	4
Mas reciente	0	1	2	3	0	1	4	4	4	2	3	3
		0	1	2	3	0	1	1	1	4	2	2
Mas antigua			0	1	2	3	0	0	0	1	4	4
	P	P	P	P	P	P	P			P	P	

9 Page Faults

Secuencia de acceso	0	1	2	3	0	1	4	0	1	2	3	4
Mas reciente	0	1	2	3	3	3	4	0	1	2	3	4
		0	1	2	2	2	3	4	0	1	1	3
			0	1	1	1	2	3	4	0	1	2
Mas antigua				0	0	0	1	2	3	4	0	1
	P	P	P	P	P	P	P			P	P	**P**

10 Page Faults

Modelación de algoritmos de paginación

Algoritmos de pila

- Se denomina **cadena de referencias** a la secuencia de páginas que solicita un proceso durante su ejecución.

- Se llaman **algoritmos de pila** aquellos cuya cadena de referencia de páginas que están en memoria para un numero ***m*** de marcos de memoria, es subconjunto de **m + 1**, es decir, *usando m + 1 marcos tenemos las misma páginas que con m más un adicional,* considerando un arreglo M con tantas posiciones como páginas tiene el proceso.

Cadena de ref.		0	2	1	3	5	4	3	1
Paginas		0	2	1	3	5	4	3	1
En			0	2	1	3	5	4	3
Memoria				0	2	1	3	5	4
Páginas					0	2	1	1	5
En						0	2	2	2
Swap							0	0	0
		P	P	P	P	P	P		P

M

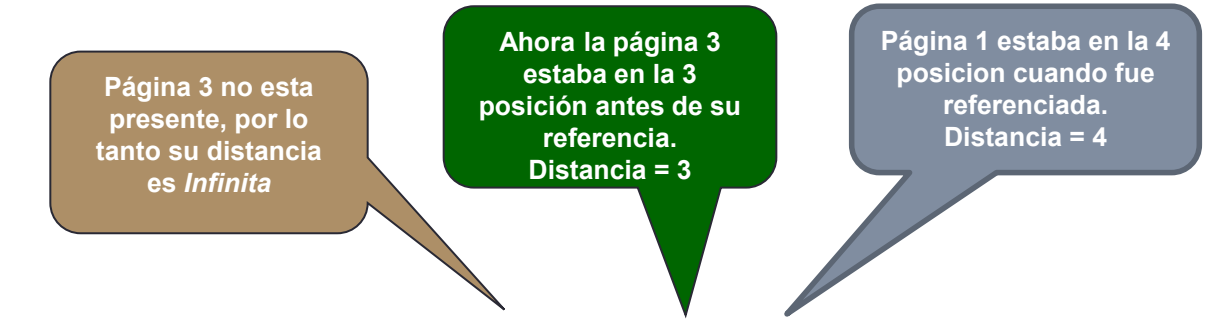
Cadena de ref.		0	2	1	3	5	4	3	1
Paginas		0	2	1	3	5	4	3	1
En			0	2	1	3	5	4	3
Memoria				0	2	1	3	5	4
					0	2	1	1	5
Páginas En						0	2	2	2
Swap							0	0	0
		P	P	P	P	P	P		

Modelación de algoritmos de paginación

Distancias de cadena

- En el caso de los algoritmos de pila, la referencia a una página puede abstraerse como la *distancia desde la parte superior de la pila* donde se colocó la página.

- La cadena de distancias lleva el conteo de la cantidad de veces que se ha accedido a páginas en cada una de las posibles distancias, considerando que



Cadena de ref.		0	2	1	3	5	4	3	1
Paginas		0	2	1	3	5	4	3	1
En			0	2	1	3	5	4	3
Memoria				0	2	1	3	5	4
Páginas					0	2	1	1	5
En						0	2	2	2
Swap							0	0	0
Cadena de distancias		∞	∞	∞	∞	∞	∞	3	4

Cadena de Distancias

$C_1 = 0$
$C_2 = 0$
$C_3 = 1$
$C_4 = 1$
$C_5 = 0$
$C_6 = 0$
$C_\infty = 6$

Modelación de algoritmos de paginación

Predicción de la tasa de fallos de página

La distancia de cadena permite apreciar claramente la cantidad de fallos de página que habrán con una determinada cadena de referencias usando una cantidad de marcos de memoria. Utilizando la siguiente fórmula se puede obtener el vector F, que contiene la cantidad de fallos de página con las condiciones dadas:

$$F_m = \sum_{k=m+1}^{n} C_k + C_\infty$$

m = marcos de página
n = número de páginas del proceso

Ej:

Sean las Distancias de Cadena

$C_1 = 4$
$C_2 = 4$
$C_3 = 2$
$C_4 = 3$
$C_5 = 0$
$C_6 = 2$
$C_\infty = 6$

$C_2 + C_3 + C_4 + C_5 + C_6 + C_\infty$	$F_1 = 17$
$C_3 + C_4 + C_5 + C_6 + C_\infty$	$F_2 = 13$
$C_4 + C_5 + C_6 + C_\infty$	$F_3 = 11$
$C_5 + C_6 + C_\infty$	$F_4 = 8$
$C_6 + C_\infty$	$F_5 = 8$
C_∞	$F_6 = 6$
	$F_\infty = 6$

Fallos de página con 2 marcos

En este caso aumentar un marco no aporta a la disminución de fallos de página

CONDICIONES DE DISEÑO PARA SISTEMAS PAGINADOS

El modelo del conjunto de trabajo

Políticas de asignación locales vs. Globales

Tamaño de página

Modelo de Conjunto de Trabajo (Working-Set)

- **Conjunto de Trabajo:** conjunto de páginas que se está utilizando en un momento dado. Este no es fijo, y cambia periódicamente a medida que el proceso se ejecuta.

- El algoritmo **Working Set** se basa en lograr mantener en memoria las páginas que se consideran dentro del conjunto de trabajo para así reducir al máximo la cantidad de Fallos de Página.

Modelo de Conjunto de Trabajo (Working-Set)

- Para establecer el **conjunto de trabajo** se puede usar uno de los siguientes métodos:
  - Páginas referenciadas en los k últimos accesos a memoria, aquellas que no están en el conjunto pueden ser desalojadas
  - k últimas páginas referenciadas, idem caso anterior
  - Usar el ***tiempo virtual de ejecución*** (tiempo real que ha ocupado el proceso en CPU) para determinar la edad de la página. Aquellas que sobrepasan una edad máxima **T** se consideran fuera del conjunto y pueden ser desalojadas. (RECOMENDADO)
- Se asume se tiene hardware que enciende los bits R y M adecuadamente.
- El bit R se apaga en cada interrupción de reloj, actualizando el tiempo de acceso a aquellas páginas cuyo bit R es 1.

Modelo de Conjunto de Trabajo (Working-Set)

- Usando un set de trabajo con T = 50

tVirtual = 80

Pág	Ult Uso	R
0	50	0
1	50	0
2	60	0
3	15	0
4	60	0

WS = {0, 1, 2, 4}

Accesos a páginas 1,2,4 durante quantum

tVirtual = 110

Pág	Ult Uso	R
0	50	0
1	50	1
2	60	1
3	15	0
4	60	1

WS = {0, 1, 2, 4}

tVirtual = 110

Pág	Ult Uso	R
0	50	0
1	**110**	**0**
2	**110**	**0**
3	15	0
4	**110**	**0**

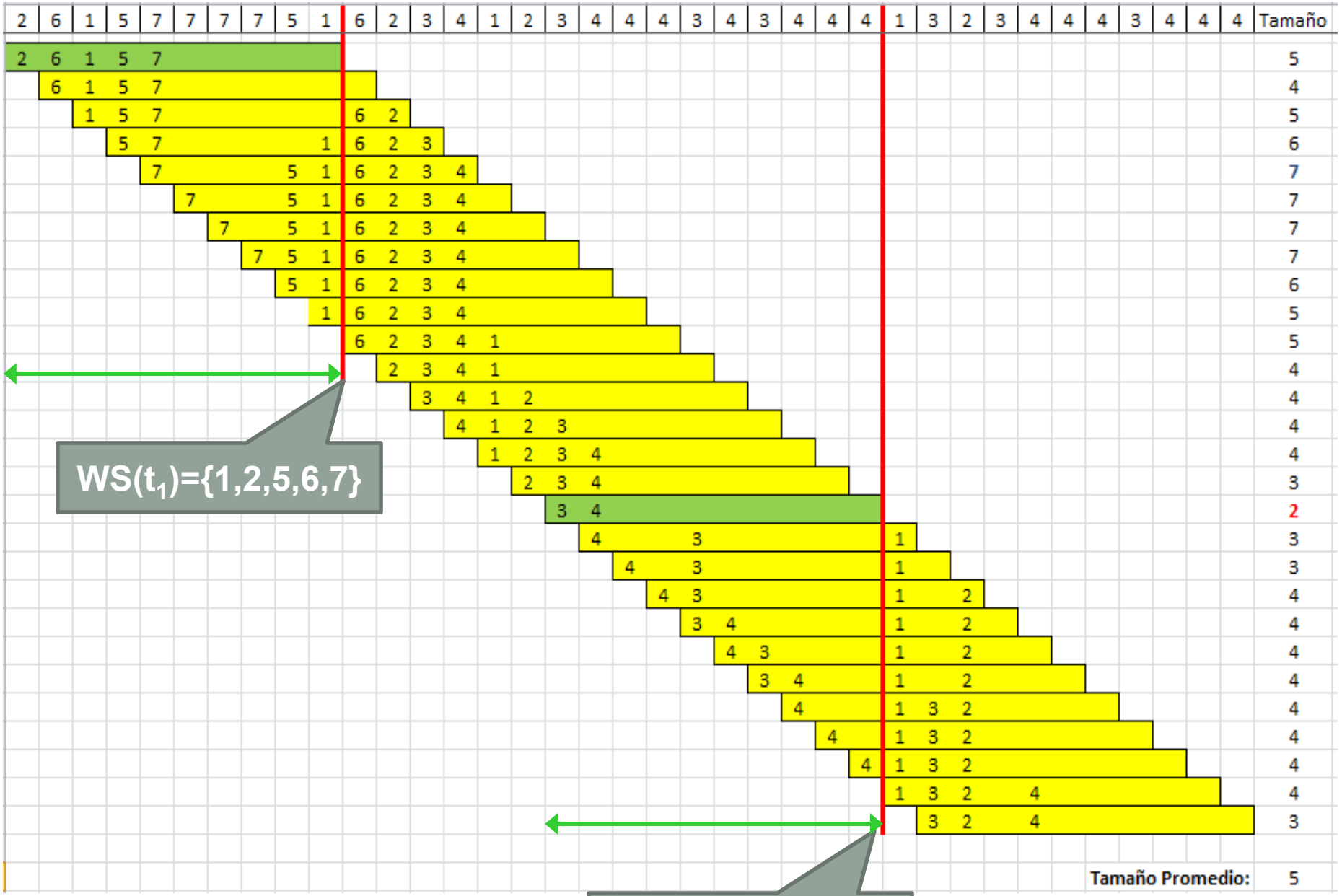
WS = {1, 2, 4}

Páginas que pueden ser desalojadas

Modelo de Conjunto de Trabajo (Working-Set)

$\Delta \equiv$ ventana working-set $\equiv$ un número fijo de referencias de páginas

- Ejemplo:
  - WSS_i (working set del proceso P_i) = número total de páginas referenciadas en el mas reciente Δ (varía en el tiempo)
    - si Δ es demasiado chico no acompaña la localidad.
    - si Δ es demasiado grande acompaña varias localidades.
    - si $\Delta = \infty \Rightarrow$ acompañará al programa entero.
  - $D = \sum WSSi \equiv$ demanda total de marcos
  - si $D > m \Rightarrow$ Hiperpaginación
  - Política: si D > m, entonces suspende uno de los procesos.



- Usando un Δ = 10 referencias

Condiciones de Diseños para Sistemas Paginados

- **Asignación local v/s global**

  - Cuando existen múltiples procesos ejecutándose, entonces nace el problema sobre si aplicar el algoritmo de remplazo de páginas localmente sobre el conjunto de marcos asignados al proceso que genera el fallo de página, o globalmente sobre el total de los marcos de memoria.

  - El método local no permite que el conjunto de trabajo crezca, pudiendo fácilmente producir hiperpaginación.

  - El método global permite ajustar el tamaño del conjunto de trabajo. Introduce el uso de un nuevo algoritmo de remplazo de páginas: PPF: Page Fault Frequency.

  - El algoritmo de Frecuencia de Fallos de Página analiza la cantidad de fallos de página por proceso, entregando más marcos a aquellos con altas tasas de Page Fault, y quitando a aquellos con bajo número de fallos.

Condiciones de Diseños para Sistemas Paginados

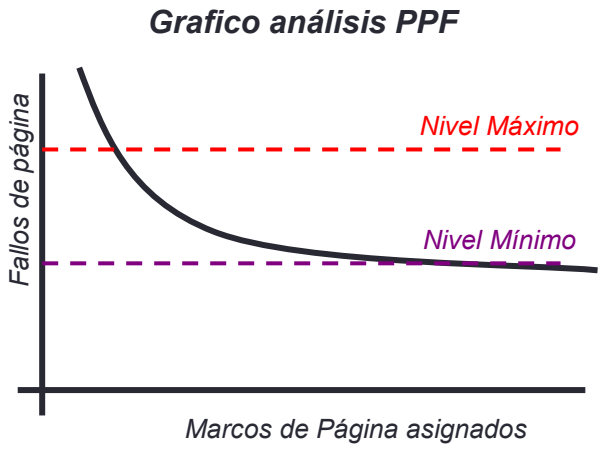
Asignación local v/s global

Ej: ***A*** *requiere un marco adicional*

Seleccionada usando política local

Seleccionada usando política global

Pág	Ult Uso
A0	50
A1	50
A2	40
B0	15
B1	60
B2	30
B3	70
B4	60
C0	25
C1	25



Condiciones de Diseños para Sistemas Paginados

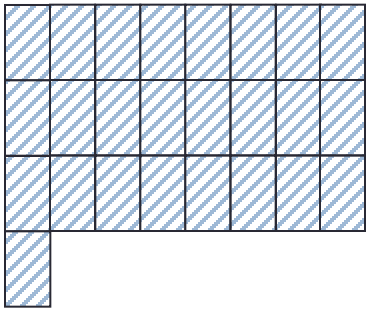
Tamaño de la página

- Es elegido por el sistema operativo, aun cuando el hardware pueda estar diseñado para trabajar con otro tamaño.
  Ej: SO con páginas de 1Kb, HW con páginas de 512b, el SO trabajará siempre con 2 páginas de HW por cada de software.

- Reducir el tamaño de la página, reduce la fragmentación interna pues se desperdicia menos memoria en la última página; sin embargo esto aumenta el tamaño de la tabla de páginas haciendo más engorroso el trabajo con esta.

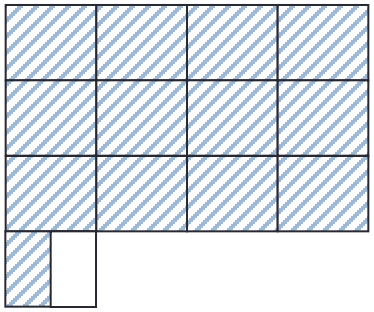
Ej: Proceso de 25kb

25 Páginas de 1Kb



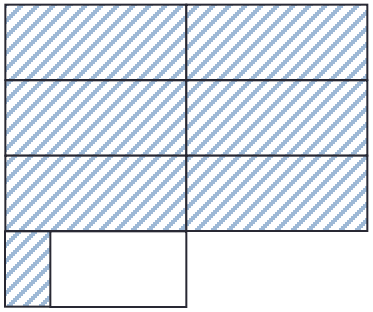
Desperdicio: 0kb

13 Páginas de 2Kb



Desperdicio: 1kb

7 Páginas de 4Kb



Desperdicio: 3kb

Condiciones de Diseños para Sistemas Paginados

Tamaño de la página

- La memoria en juego corresponde al tamaño de la tabla de páginas y al desperdiciado en la última página del proceso.

- Asumiendo que en promedio se pierde la mitad de la última página, se puede modelar esta cantidad de memoria a través de la siguiente fórmula:

$$\frac{s \cdot e}{p} + \frac{p}{2}$$

Siendo:

s = el tamaño promedio de los procesos
e = el tamaño de 1 entrada de la tabla de páginas
p = el tamaño de 1 página

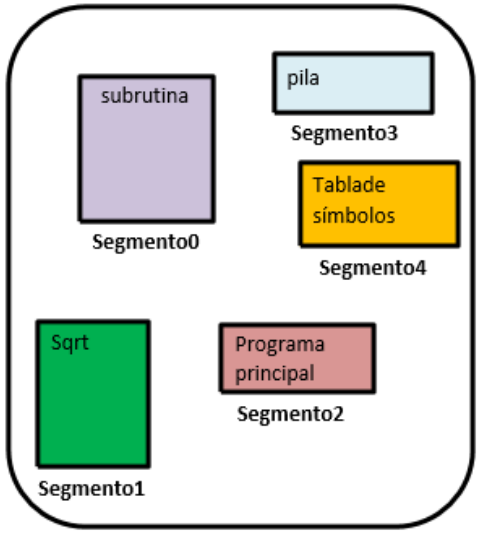
Podemos minimizar entonces la cantidad de memoria en juego con el tamaño de página como parámetro derivando e igualando a cero, dado que la segunda derivada es negativa para cualquier valor de *p* entero positivo.

$$-\frac{s \cdot e}{p^2} + \frac{1}{2} = 0 \quad \Rightarrow \quad p = \sqrt{2s \cdot e}$$

ASPECTOS DE IMPLANTACIÓN

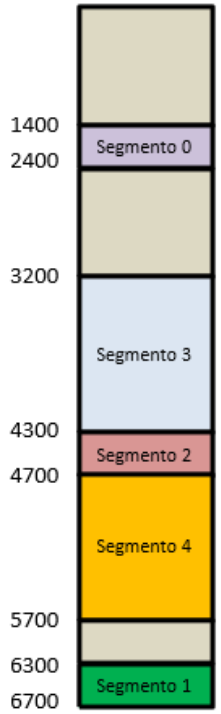
Memoria Virtual: Segmentación

- La Segmentación es un esquema de administración de la memoria que soporta la visión que el usuario tiene de la misma.
- Un espacio de direcciones lógicas es una colección de segmentos.
- Cada segmento tiene un nombre y una longitud.
- Las direcciones especifican tanto el nombre del segmento como el desplazamiento dentro del segmento. Por lo tanto, el usuario especifica cada dirección mediante dos cantidades : un nombre de segmento y un desplazamiento.



	Límite	Base
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

Tabla de segmentos



Memoria Virtual: Segmentación

- Al igual que en la Segmentación Simple, se tiene una tabla de segmentos por cada proceso.

- Las entradas en la tabla de segmentos son un poco mas complejas. Se requiere un bit en cada entrada para indicar si el segmento correspondiente se encuentra presente en la memoria principal o no. Si indica que el segmento está en memoria, la entrada también debe incluir la dirección de comienzo y la longitud del mismo.

- Otro bit de control es el bit de modificado, que indica si los contenidos del segmento se han modificado desde que se cargó por última vez en la memoria principal. Si no hay ningún cambio, no es necesario escribir el segmento cuando se reemplace en la memoria principal.

- Si se gestiona la protección y compartición, se necesitarán los bits correspondientes a estos fines.

Memoria Virtual: Segmentación

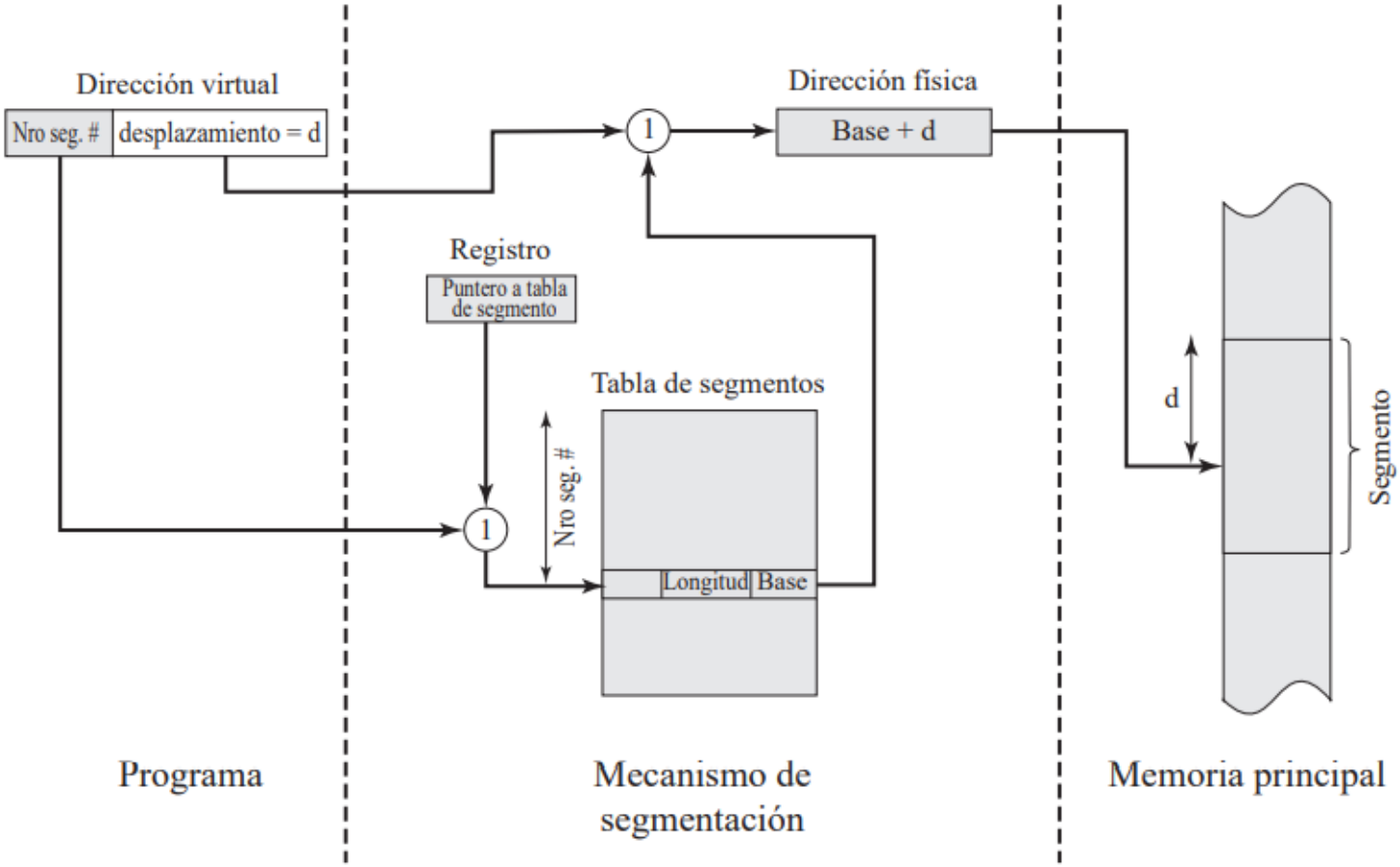


Figura 8.12. Traducción de direcciones en un sistema con segmentación.

MV: Ventajas de la Segmentación

1. Generalmente, el programador no conoce realmente el tamaño que utilizará una estructura de datos, luego si ésta tiene asignado un segmento podrá ser expandido o reducido según sus necesidades.

2. Los programas pueden ser compilados de manera independiente aunque sean módulos de otros programas.

3. Varios procesos pueden compartir segmentos, por si desean compartir datos.

4. Se pueden proteger los programas o datos asignando privilegios de acceso a los segmentos.

MV: Combinación entre Paginación y Segmentación

Las combinaciones entre paginación y segmentación son las siguientes:

1. **Memoria no segmentada y no paginación:** La dirección virtual es la misma que la dirección física.

2. **Memoria paginada no segmentada:** La memoria es un espacio de direcciones paginado.

3. **Memoria segmentada no paginada:** La memoria es un conjunto de direcciones lógicas.

4. **Memoria segmentada paginada:** La segmentación define particiones lógicas de memoria y la paginación gestiona la asignación de memoria dentro de las particiones.

Protección y Compartición

- La segmentación se presta a la implementación de políticas de protección y compartición.
  - Cada entrada de la tabla de segmentos incluye la longitud y la dirección base, un programa no podrá acceder por descuido a una posición de memoria principal mas allá de los límites de un segmento.
  - Para conseguir la compartición, un segmento se referenciará en las tablas de segmento de mas de un proceso.

- Otro mecanismo
  - Utilizar una estructura de anillo de protección, cuyos principios básicos son:
    - Un programa puede acceder solo a datos en el mismo anillo o en un anillo de menor privilegio
    - Un programa puede hacer llamadas a servicios que residan en el mismo anillo o en anillos más privilegiados

Protección y Compartición

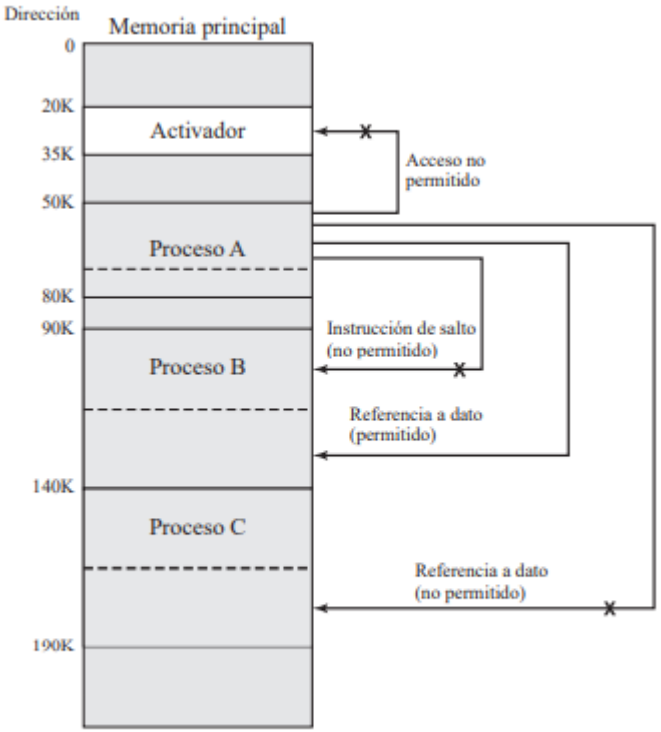
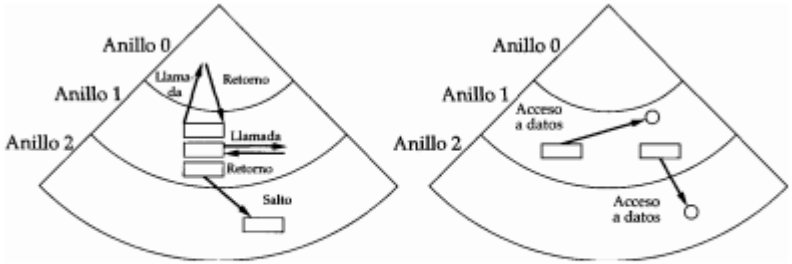


Figura 8.14. Relaciones de protección entre segmentos.



(a) Transferencia de control entre programas (b) Accesos a datos

FIGURA 7.13 Convenios de protección por anillos [FURH87]

FIN