La memoria virtual

aspectos fundamentales

Roberto Gómez C.

If it's there and you can see it

- *it's real*

If it's not there and you can see it

- *it's virtual*

If it's there and you can't see it

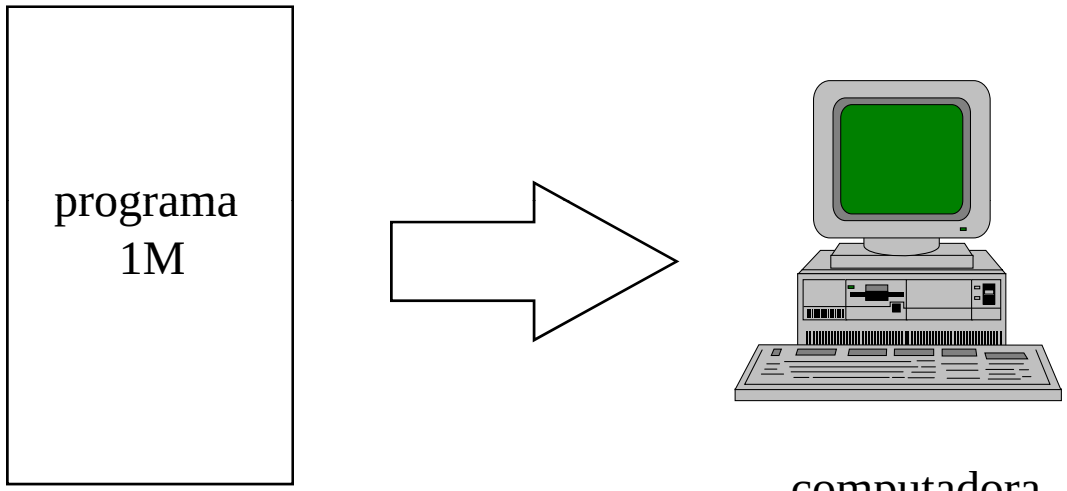
- *it's transparent*

If it's not there and you can´t see it

- *you erased it!*

IBM poster explaining virtual memory, circa 1978

El problema



Programas muy grandes no caben en memoria disponible

La solución

- Dividir programa en partes (capas)

capa 0: se ejecuta primero
termina
se llama la capa 1
se ejecuta la capa1
termina
se llama la capa2
se ejecuta la capa2
:
:

- Capas se mantienen en disco (zona de swap) y se intercambian con la memoria por medio del sistema operativo

Roberto Gómez C.

Aspectos solución a considerar

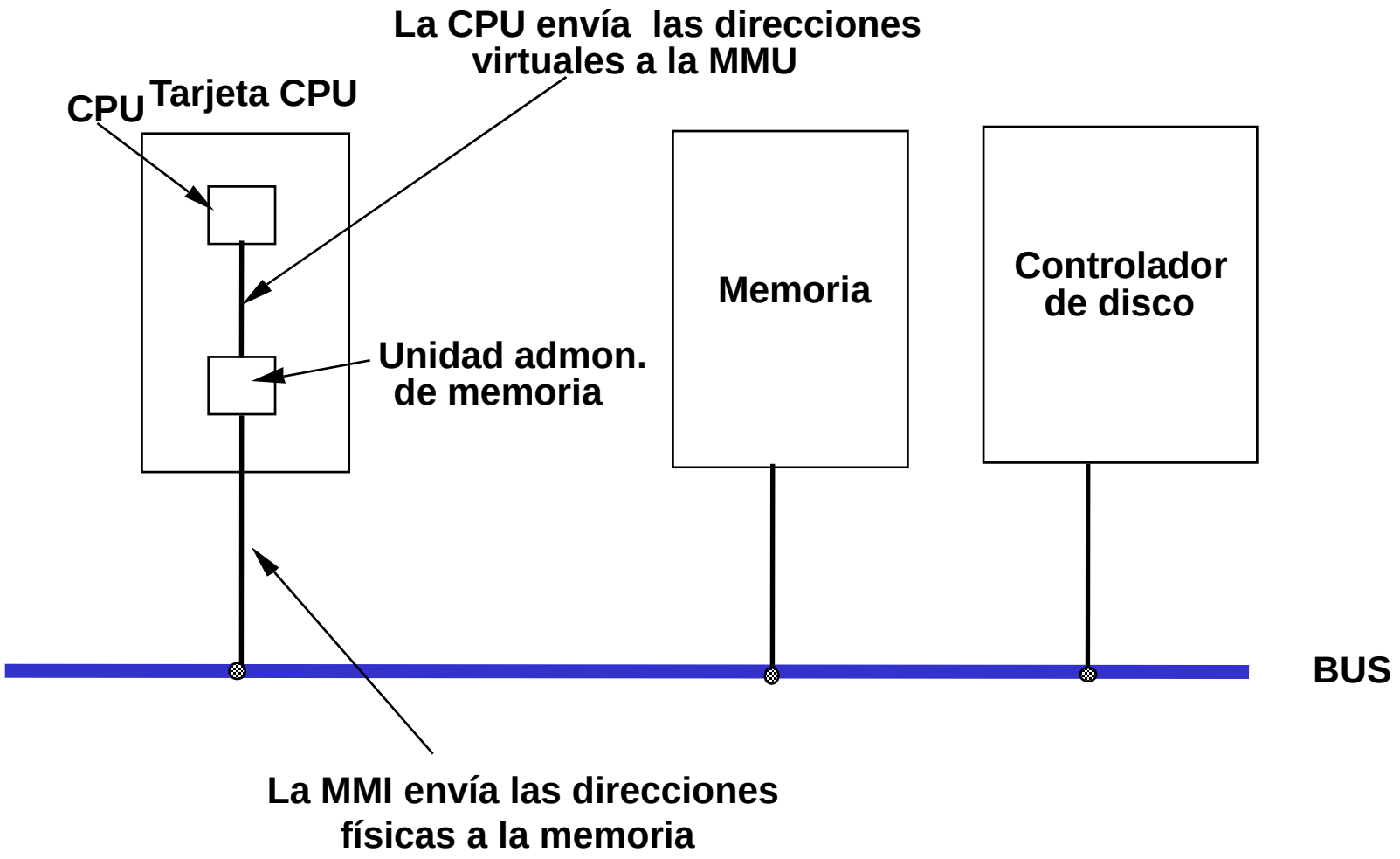
- Responsabilidades:
  - Intercambio capas disco/memoria: sist. operativo
  - Dividir programa: programador
- Originalidad
  - división programa responsabilidad del sistema
  - método diseñado por Fotheringham (1961)
  - método conocido como memoria virtual
- Principio
  - Sistema operativo mantiene en memoria aquellas partes del programa que se usen (referencien) y el resto permanece en disco
  - direcciones generadas dentro de un programa mediante índices, registros base, de segmento y otros, se denominan *espacio direcciones virtuales*
  - las direcciones de la memoria disponible (memoria física) se conocen con el nombre de *direcciones físicas*

Roberto Gómez C.

Más aspectos

  - si no hay memoria virtual:
    - la dirección virtual es la dirección física
  - con memoria virtual
    - es necesario saber que dirección física le corresponde a la dirección virtual
- Métodos
  - paginación
  - segmentación

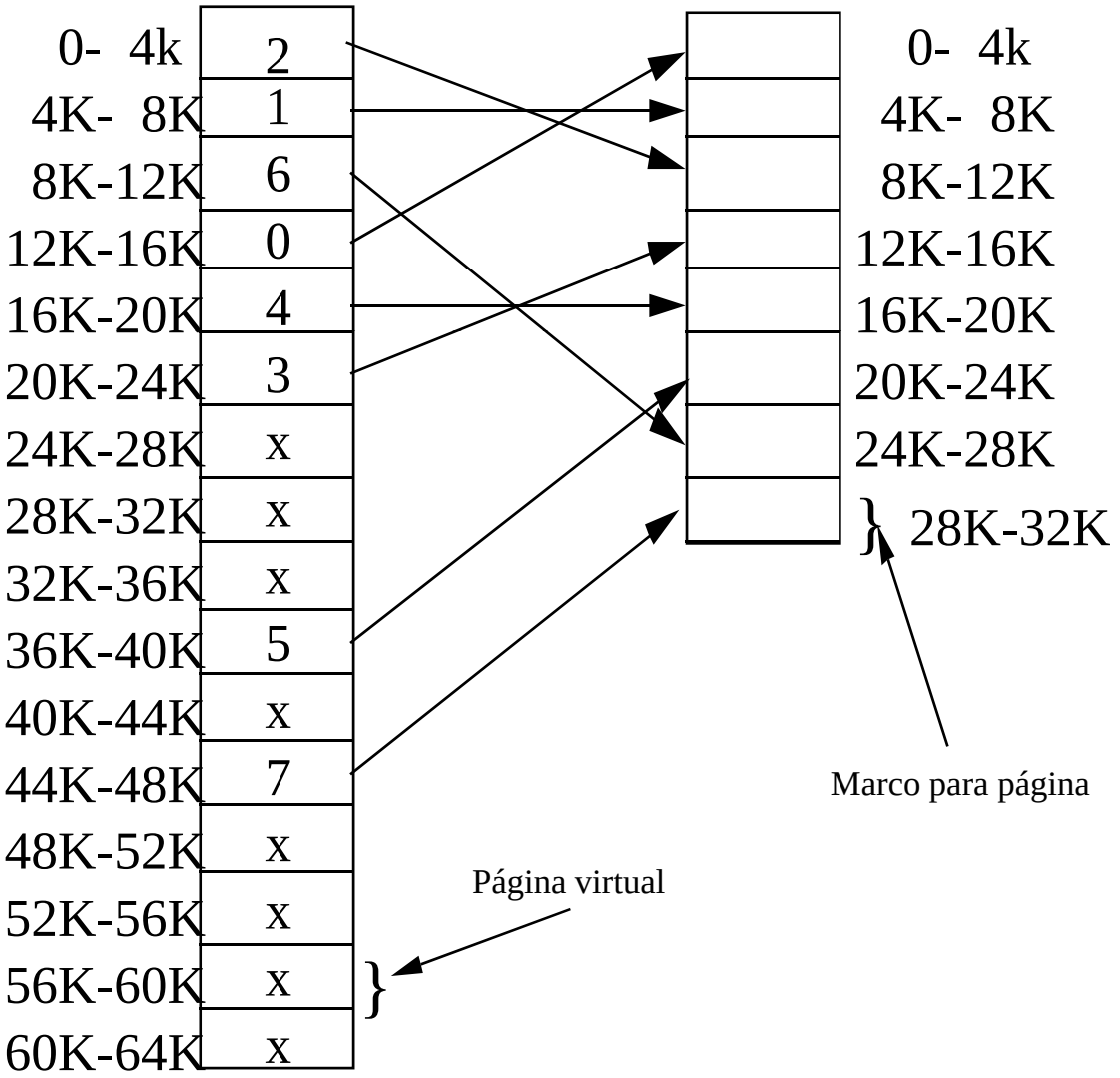
La MMU


La CPU envía las direcciones virtuales a la MMU
CPU
Tarjeta CPU
Memoria
Controlador de disco
Unidad admon. de memoria
BUS
La MMI envía las direcciones físicas a la memoria

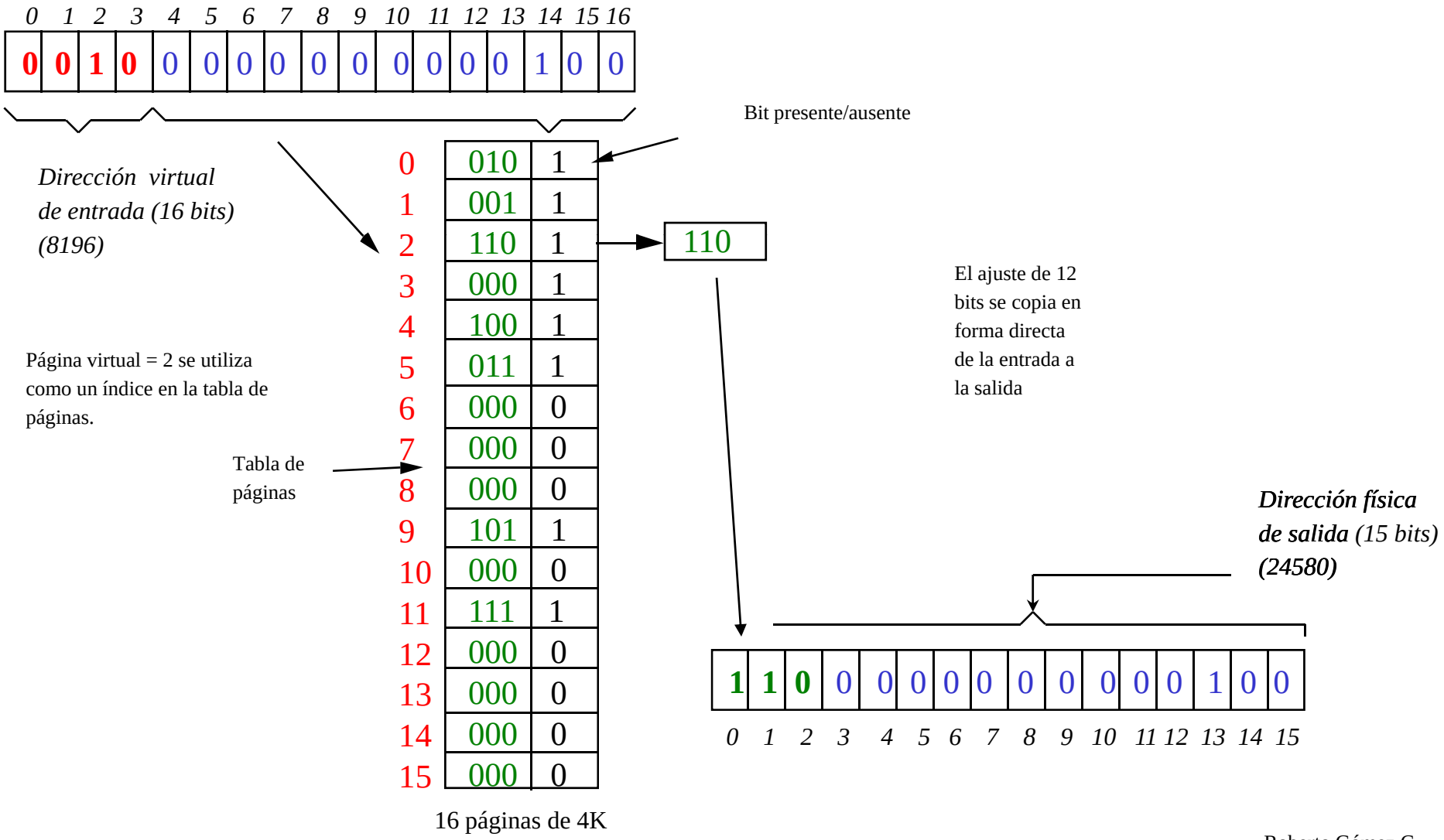
La paginación

- Espacio direcciones virtuales se divide en unidades llamadas *páginas*.
- Las unidades correspondientes en memoria física se llaman *marcos para páginas*
- Las páginas y los marcos tienen siempre el mismo tamaño.
- *Fallo de página*:
  - pagina no asociada con algún marco

Ejemplo paginación



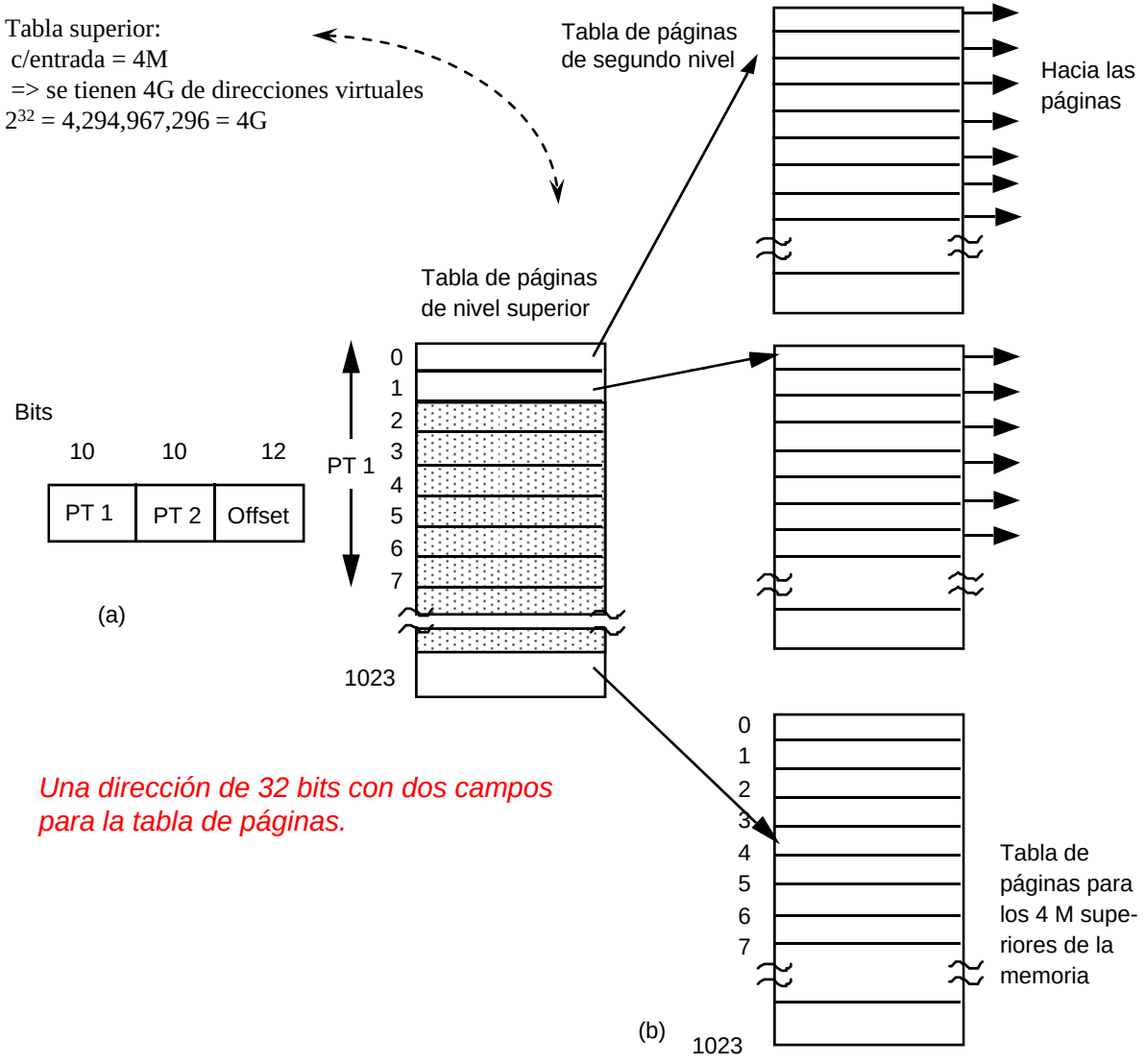
Direccionamiento en paginación



16 páginas de 4K

Roberto Gómez C.

Tablas de páginas multinivel

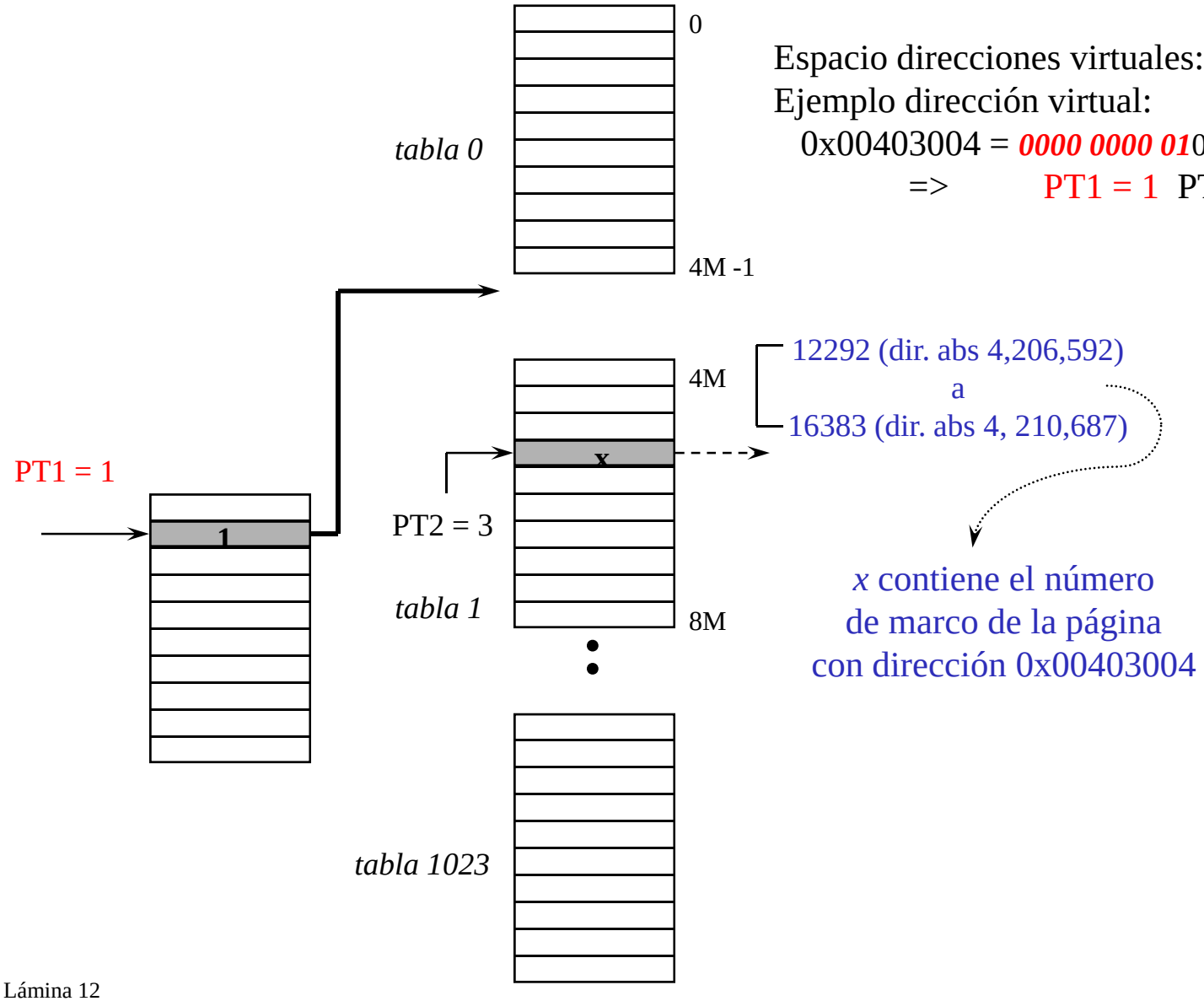


(a) Una dirección de 32 bits con dos campos para la tabla de páginas.

(b) Tablas de páginas de dos niveles

Roberto Gómez C.

Ejemplo tablas multinivel



Espacio direcciones virtuales: 2^{32} = 4,294,967,296 = 4G

Ejemplo dirección virtual:

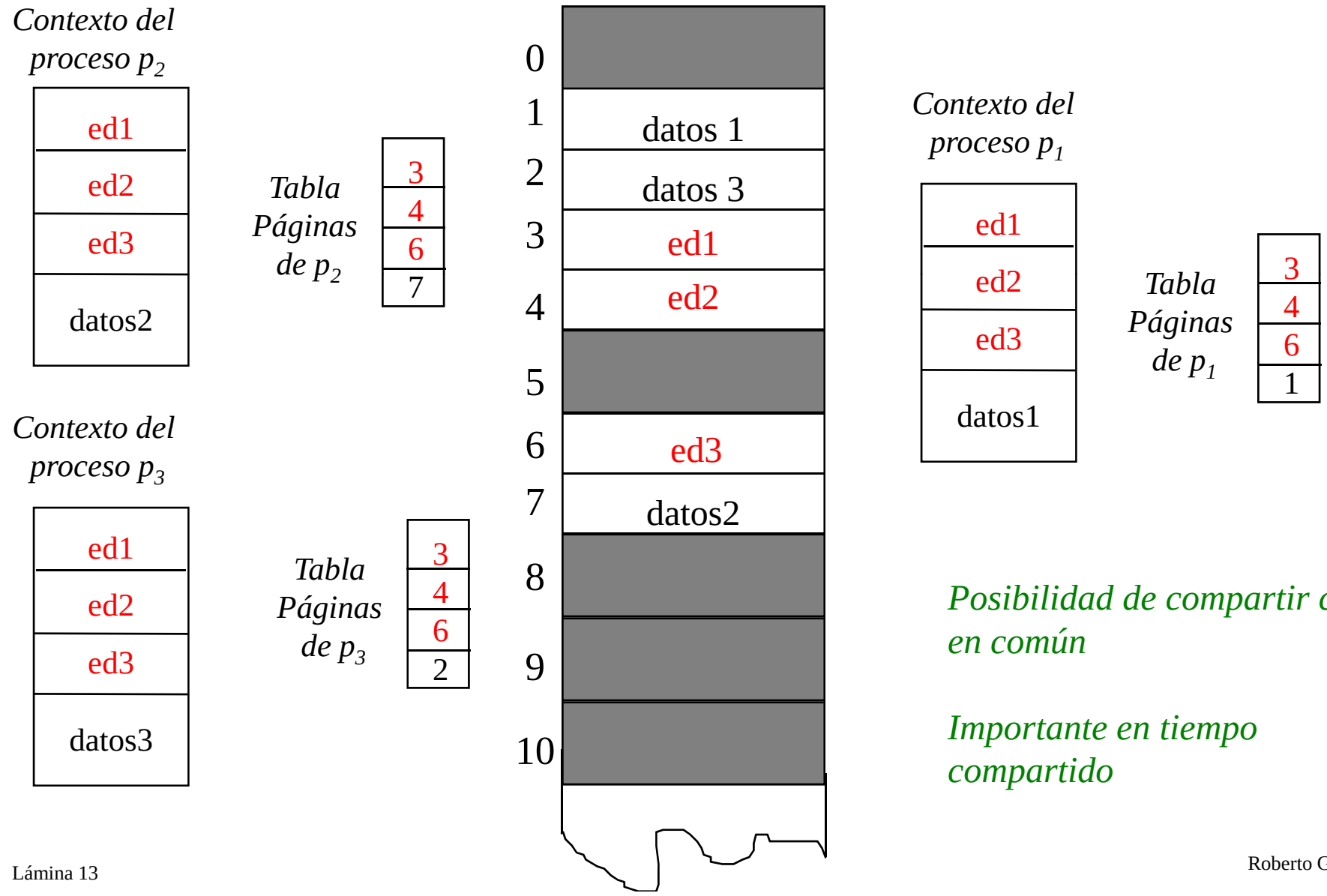
0x00403004 = ***0000 0000 01***00 0000 0011 ***0000 0000 0100***

=> PT1 = 1 PT2 = 3 Offset = 4

x contiene el número de marco de la página con dirección 0x00403004

Roberto Gómez C.

Ejemplo páginas compartidas

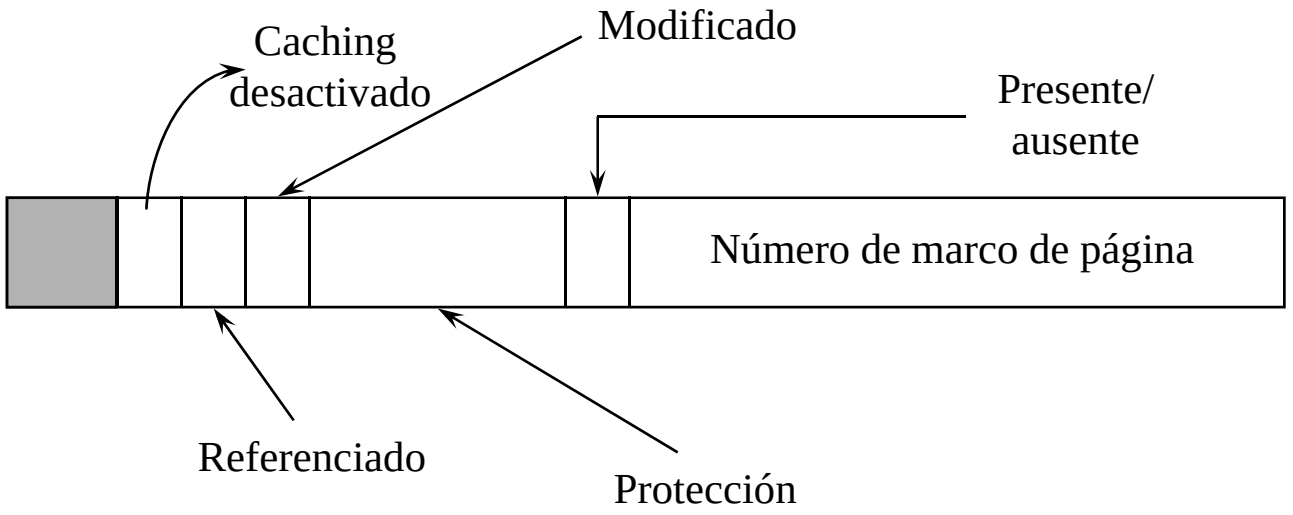


Posibilidad de compartir código en común

Importante en tiempo compartido

Roberto Gómez C.

Estructura tabla páginas



caching: páginas asociadas a registros de dispositivos
referenciado: 1 cuando se hace una referencia a una página para leer o escribir
modificado: recuperación marco, si página no ha sido modificada no se escribe a disco
protección: tipo acceso permitido, (0 lec/esc, 1 solo lectura)
presente/ausente: 1: entrada valida y puede ser utilizada
0: página no cargada en memoria

Roberto Gómez C.

Algoritmos reemplazo páginas y de asignación de marcos

- Dos aspectos a cuidar en la implementación de la paginación por demanda:
  - desarrollar un algoritmo de asignación de marco
  - desarrollar un algoritmo de reemplazo de páginas
- En el primero, si se tienen varios procesos en memoria, hay que decidir cuántos marcos se asignarán a cada uno
- En el último, si hay que reemplazar páginas hay que seleccionar cuales se reemplazarán

Roberto Gómez C.

Algunos algoritmos de reemplazo

- Algoritmo de uso no tan reciente (NRU)
- Algoritmo FIFO
- Algoritmo de la segunda oportunidad
- Algoritmo reemplazo del reloj
- Algoritmo de la menor uso reciente (LRU)
  - implementación contador 64 bits
  - implementación con matriz
  - implementación con pila
- Algoritmo maduración

Algoritmo de uso no tan reciente

- Computadoras tiene dos bits de estado asociados a cada página:
  - R: se activa si se hace referencia a la página
  - M: se activa cuando se escribe en la página
- Bits actualizados en cada referencia a memoria.
- Si bits no existen en hardware se pueden simular en software.
- Principio Algoritmo:
  - Al iniciar un proceso el S.O. asigna 0 a R y M de todas las páginas
  - En forma periódica se limpia el bit R, (para distinguir páginas que no tengan referencia de las que sí).

Categorías páginas en NRU

- Fallo de página: S.O. inspecciona todas las páginas y las divide en cuatro categorías, (según valores R y M):
  - caso 0: no referenciada, no modificada R=0 M=0
  - caso 1: no referenciada, pero ha sido modificada R=0 M=1
  - caso 2: se ha hecho referencia, pero no modificada R=1 M=0
  - caso 3: se ha hecho referencia y ha sido modificada R=1 M=1

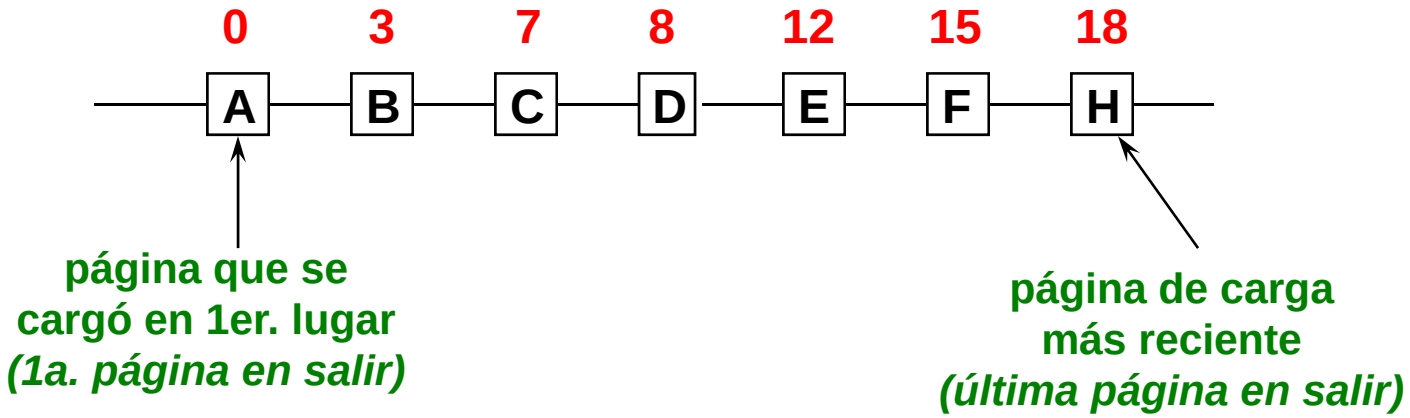
- Algoritmo NRU elimina página de la primera clase no vacía con el número más pequeño, (número de clase).
- Si todas tienen el mismo nivel se elimina el que llego primero

Comentarios algoritmo

- Hipótesis implícita:
  - es mejor eliminar una página modificada sin referencia en al menos un intervalo de reloj, que una página en blanco de uso frecuente.
- Ventajas
  - Fácil de comprender
  - Implantación eficiente
  - Rendimiento que, aún sin ser el óptimo, sí es adecuado con mucha frecuencia.

Algoritmo FIFO

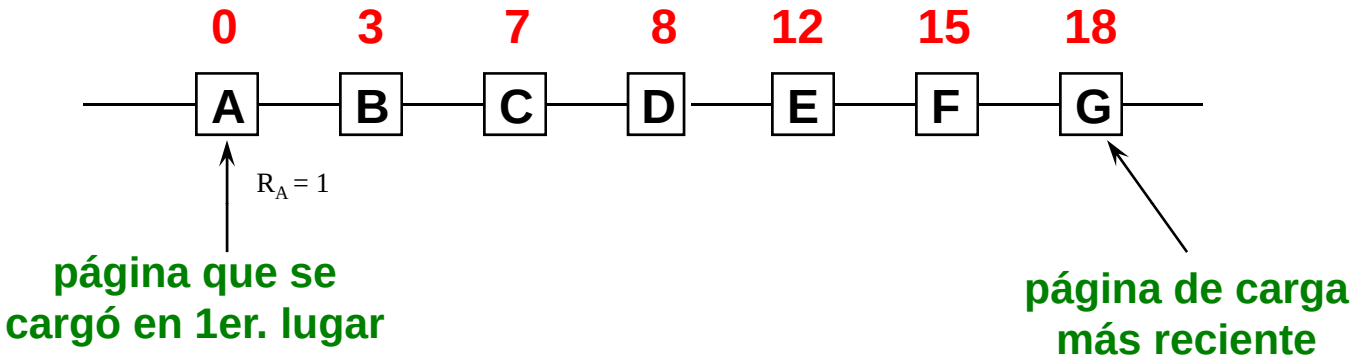
- Principio los primeros en entrar, son los primeros en salir.
- S.O. tiene una lista de todas las páginas en memoria, siendo la primera página la más antigua y la última la más reciente.
- En un fallo de página se elimina la primera página y se añade la nueva al final de la lista.
- FIFO es muy pocas veces usada en su forma más pura.

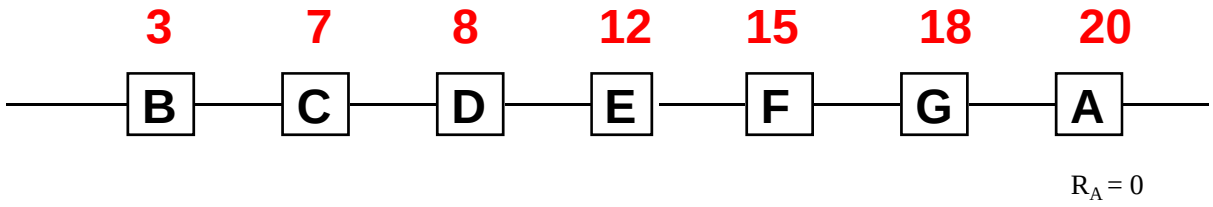


Algoritmo de la segunda oportunidad

- Modificación simple de FIFO
- Evita deshacerse de una página de uso frecuente inspeccionando el bit R de la página más antigua.
  - si (R = 0) => página antigua y no utilizada se reemplaza en forma inmediata
  - si (R=1) => el bit se limpia la página se coloca al final de la lista su tiempo de carga se actualiza
- Una variante del algoritmo es usar dos bits M y R para decidir si se da una segunda oportunidad

Ejemplo algoritmo segunda oportunidad

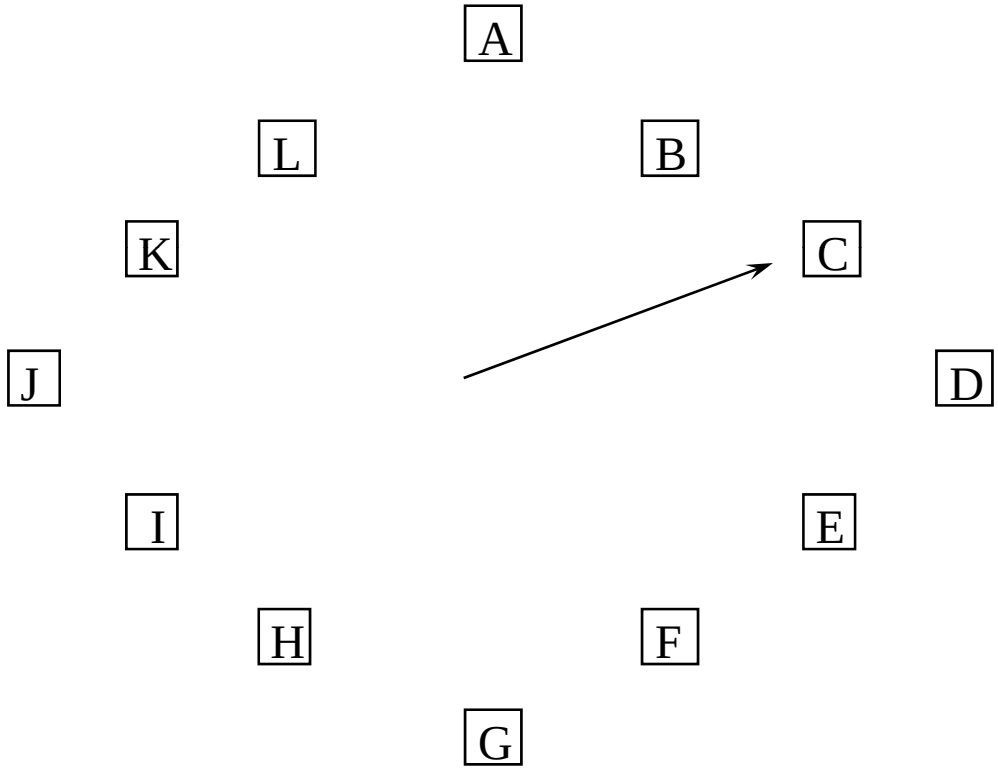




Algoritmo de reemplazo de reloj

- Mantener páginas en una lista circular con forma de reloj.
- Una manecilla apunta hacia la página más antigua.
- Al ocurrir un fallo de página se inspecciona la página a la que apunta la manecilla.
- Si bit R = 0 =>
  - página se retira de memoria
  - se inserta nueva página en su lugar en el reloj
  - manecilla avanza una posición
- si no se busca una página con R = 0.
- Difiere del anterior sólo por la implementación

Esquema reemplazo reloj



Algoritmo de la menor uso reciente

- Páginas uso frecuente en las últimas instrucciones se utilizan con cierta probabilidad en las siguientes.
- Es probable que las páginas que no hayan sido utilizadas durante mucho tiempo permanezcan sin uso por bastante tiempo.
- Esto induce al siguiente algoritmo:
  - *al ocurrir un fallo de página se elimina la página que no haya sido utilizada durante el tiempo más grande.*
- Nombre estrategia: LRU
- LRU: realizable en teoría, no es barato.

Aspectos implementación algoritmo

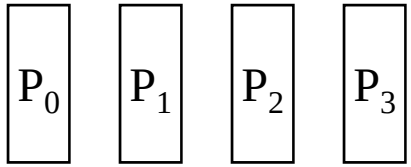
- Implementación:
  - necesario mantener una lista de todas las páginas en memoria, en donde la página de uso más reciente este al principio de la lista y la de uso menos reciente al final.
- Dificultad:
  - la lista debe actualizarse en cada referencia a la memoria.
- Búsqueda de la página en la lista, su eliminación y posterior traslado al frente de la misma NO puede ser una operación muy lenta.

Implementación contador 64 bits

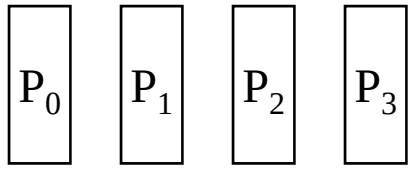
- Requiere de un contador de 64 bits, C, en hardware.
  - se incrementa en forma automática después de cada instrucción.
- Cada entrada en tabla de páginas debe contener espacio necesario para almacenar el contador.
- Después de cada referencia el valor actual de C se almacena en la entrada de la tabla de páginas correspondiente a la página a la que se hizo referencia.
- Fallo de página:
  - S.O. examina todos los contadores de la tabla de páginas y elige el mínimo, (i.e. página de uso más reciente).

Ejemplo de la implementación

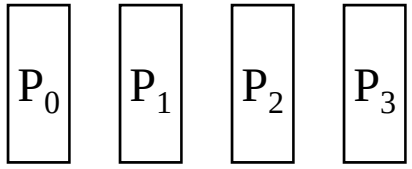
Valor inicial cont = 0;



cont = 1;

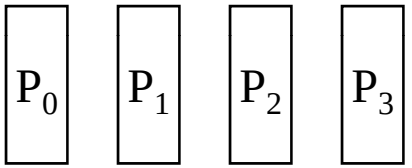


cont = 2;



cont = 3;

Referencias: P_0, P_1, P_3, P_1



cont = 4;

Página más recientemente usada: P_1, (cont = 6)
Página menos usada: P_2 , (cont = 0)

Implementacion LRU con matriz

- Máquina con *n* marcos para página, hardware LRU puede matriz de *n x n*, matriz inicializada en cero.
- Referencia al marco *k* hardware primero activa todos los bits del renglón *k* desactiva después todos los bits de la columna *k*.
- En cualquier instante:
  - renglón con valor binario mínimo es de uso menos frecuente,
  - renglón con el siguiente valor más pequeño es el segundo de uso menos reciente, etc.

Ejemplo LRU con matriz

	0	1	2	3
0	0	1	1	1
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0

(a)

	0	1	2	3
0	0	0	1	1
1	1	0	1	1
2	0	0	0	0
3	0	0	0	0

(b)

	0	1	2	3
0	0	0	0	1
1	1	0	0	1
2	1	1	0	1
3	0	0	0	0

(d)

	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0

(d)

	0	1	2	3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	1
3	1	1	0	0

(d)

	0	1	2	3
0	0	0	0	0
1	1	0	1	1
2	1	0	0	1
3	1	0	0	0

(f)

	0	1	2	3
0	0	1	1	1
1	0	0	1	1
2	0	0	0	1
3	0	0	0	0

(g)

	0	1	2	3
0	0	1	1	0
1	0	0	1	0
2	0	0	0	0
3	1	1	1	0

(h)

	0	1	2	3
0	0	1	0	0
1	0	0	0	0
2	1	1	0	1
3	1	1	0	0

(i)

	0	1	2	3
0	0	1	0	0
1	0	0	0	0
2	1	1	0	0
3	1	1	1	0

(j)

Máquina con cuatro marcos, con referencias a las páginas en el orden:

0, 1, 2, 3, 2, 1, 0, 3, 2, 3:

Roberto Gómez C.

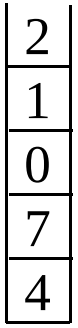
Implementación LRU con pila

- Mantener una pila de números de página
- Cuando se hace referencia a una página se saca de la pila y se coloca arriba
- De este modo, en el tope de la pila siempre esta la página más recientemente utilizada
- Ya que es necesario sacar entradas de en medio de la pila, la mejor forma de implementar la pila es con una lista doblemente encadenada y con apuntadores al principio y al final
- Cada actualización es costosa, pero no hay que buscar la página a reemplazar, uno de los apuntadores señala la página LRU
- Implementación en software o microcódigo

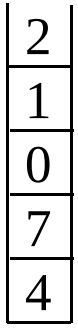
Ejemplo LRU con pila

- Ejemplo: *serie referencias: 4 7 0 7 1 0 1 0 1 2 1 2 7 1 2*

a *b*



pila antes de a



pila después de b

Algoritmo maduración

	Bits R para las páginas 0-5, en la marca de reloj	Bits R para las páginas 0-5, en la marca de reloj	Bits R para las páginas 0-5, en la marca de reloj	Bits R para las páginas 0-5, en la marca de reloj	Bits R para las páginas 0-5, en la marca de reloj	
	1 0 1 0 1 1	1 1 0 0 1 0	1 1 0 1 0 1	1 0 0 0 1 0	0 1 1 0 0 0	
Páginas:						
Página 1:	10000000	11000000	11100000	11110000	01111000	Página 1: 120
Página 2:	00000000	10000000	11000000	01100000	10110000	Página 2: 176
Página 3:	10000000	01000000	00100000	00010000	10001000	Página 3: 136
Página 4:	00000000	00000000	10000000	01000000	00100000	**Página 4: 32**
Página 5:	10000000	11000000	01100000	10110000	01011000	Página 5: 88
Página 6:	10000000	01000000	10100000	01010000	00101000	Página 6: 40
	(a)	(b)	(c)	(d)	(e)	

Ejemplos de tamaños de página

Computer	Page Size
Atlas	512 48-bit words
Honeywell-Multics	1024 36-bit word
IBM 370/XA and 370/ESA	4 Kbytes
VAX family	512 bytes
IBM AS/400	512 bytes
DEC Alpha	8 Kbytes
MIPS	4 Kbyes to 16 Mbytes
UltraSPARC	8 Kbytes to 4 Mbytes
Pentium	4 Kbytes or 4 Mbytes
IBM POWER	4 Kbytes
Itanium	4 Kbytes to 256 Mbytes

Roberto Gómez C.

Asignación marcos

- Asignación marcos monousuario
- Asignación equitativa vs proporcional
- Reemplazo global vs local

Asignación marcos monousuario

- Como repartir la cantidad fija de memoria libre entre los distintos procesos
- Si se tienen 93 marcos libres y dos procesos ¿cuantos marcos recibe cada proceso?
- Caso más sencillo: monousuario
  - memoria 128K y páginas de 1K
  - sistema operativo ocupa 35K => 93 marcos libres
  - 93 marcos se colocan en lista de marcos libres
  - proceso inicia y va a recibir los 93 marcos, cuando requiera un 94, se aplica un algoritmo de reemplazo
  - al final proceso regresa los 93 marcos

Roberto Gómez C.

Asignación marcos

- otras opciones: si sistema operativo no usa su espacio de buffers y tablas estos podrían apoyar la paginación de usuarios
- resumiendo estrategia: ***se asigna cualquier marco libre al proceso del usuario***

Asignación equitativa vs proporcional

- Asignación equitativa:
  - La forma más fácil de dividir m marcos entre n procesos es dar a cada proceso una porción de m/n marcos
  - Por ejemplo 93 marcos y cinco procesos: cada proceso recibirá 18 marcos, los tres marcos restantes pueden usarse como reserva
- Asignación proporcional
  - diferentes procesos requieren diferentes cantidades de memoria
  - si un proceso (p1) requiere 10K y otro (p2) 127K y hay 62 marcos no es lógico asignar 31 marcos a cada uno
  - se asigna memoria disponible a cada proceso según su tamaño, entonces:

s_i : memoria virtual requerida por el proceso p_i

$$S = \sum s_i$$

m : total de marcos disponibles

a_i : número marcos asignados al proceso ai

$$a_i = s_i / S \times m$$

Roberto Gómez C.

Ejemplo asignación proporcional

- Se debe ajustar a_i de modo que sea un entero mayor que el número mínimo de marcos requeridos por el conjunto de instrucciones y que la sumatoria no exceda *m*.
- Retomando ejemplo:
  - P_1 requiere 10K
  - P_2 requiere 127K
  - Existen 62 marcos
- Asignación:
  - Para el proceso P_1: (10/137) x 62 = 4
  - Para el proceso P_2: (127/137) x 62 = 57

Comentarios de ambos esquemas

- Tanto en asignación equitativa como en proporcional la asignación a cada proceso puede variar según el nivel de multiprogramación.
  - si el nivel se incrementa cada proceso perderá algunos marcos para proporcionar al nuevo proceso.
  - si el nivel disminuye los marcos que se habían asignado al proceso que salió se pueden repartir.
- Tanto en asignación equitativa como proporcional un proceso de alta prioridad se trata igual que uno de baja prioridad.

Reemplazo global vs local

- Reemplazo global
  - permite a un proceso seleccionar un marco de reemplazo del conjunto de todos los marcos,
  - incluso si ese marco esta asignado a otro proceso, un proceso puede arrebatarle un marco a otro
  - por ejemplo, procesos con más prioridad
  - problema: un proceso no puede controlar su propia frecuencia de fallos de página (proceso sólo selecciona marcos asignados a otros procesos)
- Reemplazo local
  - el número de marcos asignado a un proceso no cambia
  - el conjunto de páginas de un proceso que están en la memoria sólo depende del comportamiento de paginación de ese proceso
  - podría obstaculizar ejecución de un proceso por no dejar que aproveche otras páginas de memoria de poco uso

Roberto Gómez C.

Desempeño paginación por demanda

- Paginación por demanda puede tener un efecto importante sobre el desempeño de un sistema
- Necesario calcular el tiempo de acceso efectivo
- Tiempo acceso memoria (am) varía entre 10 y 200 nanosegundos
- Si no hay fallos: tiempo acceso efectivo es el mismo que el tiempo de acceso a memoria
- Si ocurre un fallo hay que leer del disco la página y luego acceder a la palabra deseada

El tiempo efectivo de acceso

- Entonces:
  - si p es la probabilidad de que ocurra un fallo,
  - cabe esperar que esta probabilidad es cercana a 0
  - el tiempo efectivo de acceso (tea) es:

$$tea = (1 - p) \times am + p \times (\, tiempo \;\; falló \; de \;\; página \,)$$

- Para calcular lo anterior necesitamos saber cuánto tiempo toma resolver un fallo de página

Secuencia fallo página

1. Notificación al sistema operativo
2. Guardar los registros del usuario y estado del proceso
3. Determinar que la interrupción fue un fallo de página
4. Verificar que la referencia a la página fue válida y determinar la posición de la página en disco
5. Leer del disco a un marco libre

   a. Esperar en la fila de espera de este dispositivo, hasta que se atienda la solicitud de lectura

   b. Esperar durante el tiempo de búsqueda y/o latencia del dispositivo

   c. Iniciar la transferencia de la página a un marco libre

Secuencia fallo página (cont)

6. Durante la espera asignar la CPU a algún otro usuario
7. Interrupción del disco (E/S terminada)
8. Guardar los registros y el estado de proceso del otro usuario (si se llevó a cabo el paso 6)
9. Determinar que la interrupción vino del disco
10. Corregir la tabla de páginas y las demás tablas de modo que indiquen que la página ya esta en memoria
11. Esperar que la CPU se asigne otra vez a este proceso
12. Restaurar los registros de usuario, el estado de proceso y la nueva tabla de páginas y reanudar la ejecución interrumpida

Tiempo servicio fallo página

- Tiene tres componentes
  - Atender la interrupción de fallo de página
  - Traer la página a la memoria
  - Reiniciar los procesos
- Primera y última tarea podrían tomar entre 1 y 100 microsegundos cada una
- Tiempo intercambio:
  - promedio latencia disco duro: 15 milisgs
  - tiempo de búsqueda: 8 milisgs
  - tiempo de transferencia: 1 milisg
  - total: 24 milisgs
- Tiempo total paginación: aprox. 25 milisgs

Tiempo servicio fallo página

- Solo se considera tiempo de servicio
  - si hay una cola de procesos hay que considerar el tiempo de espera para que el dispositivo este libre
- Si tiempo acceso memoria es de 100 nanosegs, el tiempo de acceso efectivo es:

tae = (1-p) x (100) + p (25 milisegs)

(1-p) x 100 + p x 25,000,000

100 + 24, 999,900 x p

Frecuencia mínima fallos página

- Tiempo de acceso efectivo es directamente proporcional a la frecuencia de los fallos de página
- Si un acceso de cada 1000 causa un fallo de página, el tiempo de acceso efectivo será de 25 microsegs

¡¡¡ Desempeño se reduce en un factor de 250 a causa de la paginación por demanda !!!

- Si se desea una degradación de menos del 10%, se necesita que:

$$110 > 100 + 25{,}000{,}000 \times p$$

$$10 > 25{,}000{,}000 \times p$$

$$p > 0.0000004$$

Concluyendo

Para mantener en un nivel razonable la reducción del desempeño sólo se puede permitir fallos de página en menos de uno de cada 2,500,000 accesos a memoria

- Es importante mantener baja la frecuencia de fallos de página, de lo contrario el tiempo de acceso aumentará y frenará la ejecución de procesos,
- Ejemplos: disco fragmentado y poca RAM en Windows 95 vs área de swap en Linux

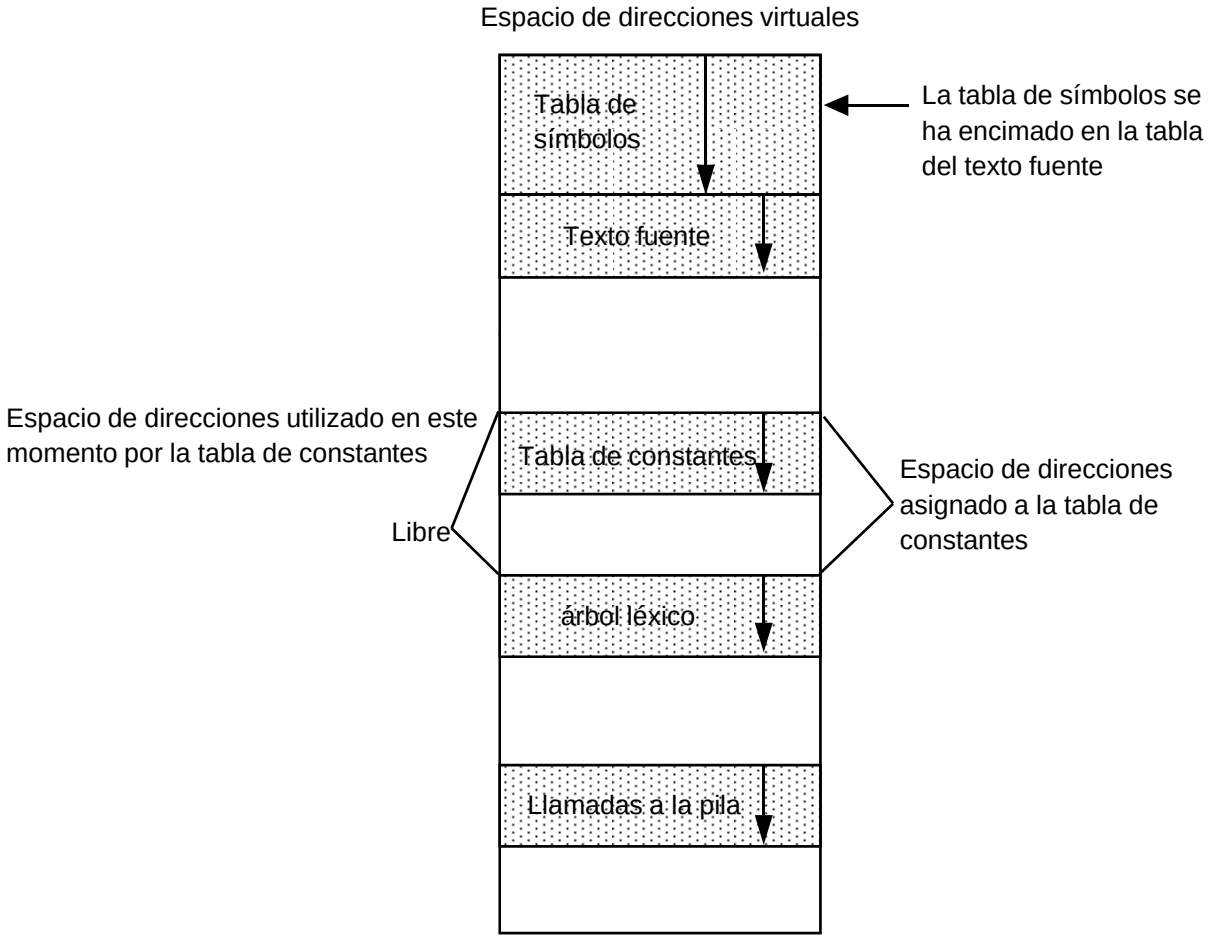
Evaluación algoritmos reemplazo páginas

- ¿Cómo escoger un algoritmo de reemplazo específico?
  - En general lo que se busca es el algoritmo con la frecuencia de fallos de página más baja
- Evaluación:
  - se ejecuta con una serie específica de referencias a memoria y se calcula el número de fallos de página
  - serie específica: serie de referencias
  - serie se puede generar en forma aleatoria o a partir del rastreo de un sistema dado
  - también se necesita saber el número de marcos que se dispone
  - a medida que aumenta el número de marcos, el número de fallos disminuye,
  - la adición de memoria física incrementa el número de marcos

La segmentación

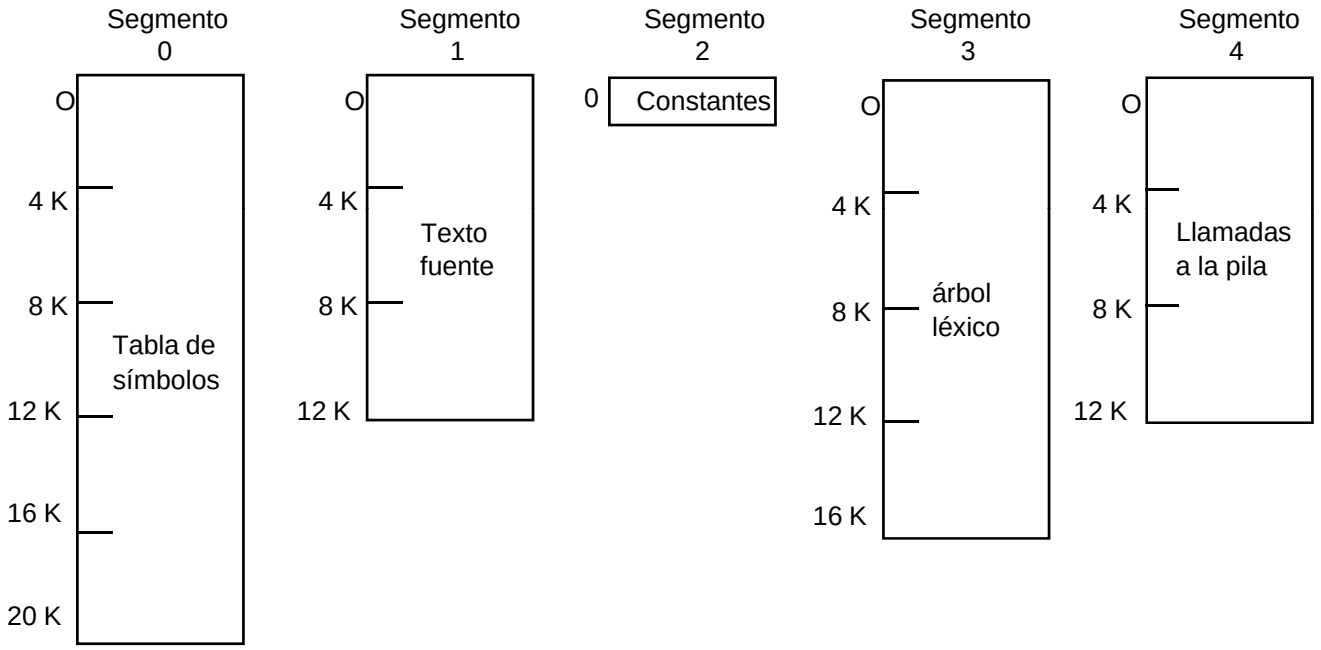
- Paginación
  - espacio direcciones unidimensional, (0..max)
- Dotar máquina de varios espacios independientes de direcciones, desde 0 hasta cierto máximo.
- La longitud de cada segmento puede ser distinta
- La longitud de un segmento puede variar durante su ejecución
- Acceso se hace en dos partes: número de segmento y una dirección dentro de este
- Un segmento puede tener la protección adecuada para el tipo de objeto almacenado

Ejemplo segmentación: compilador



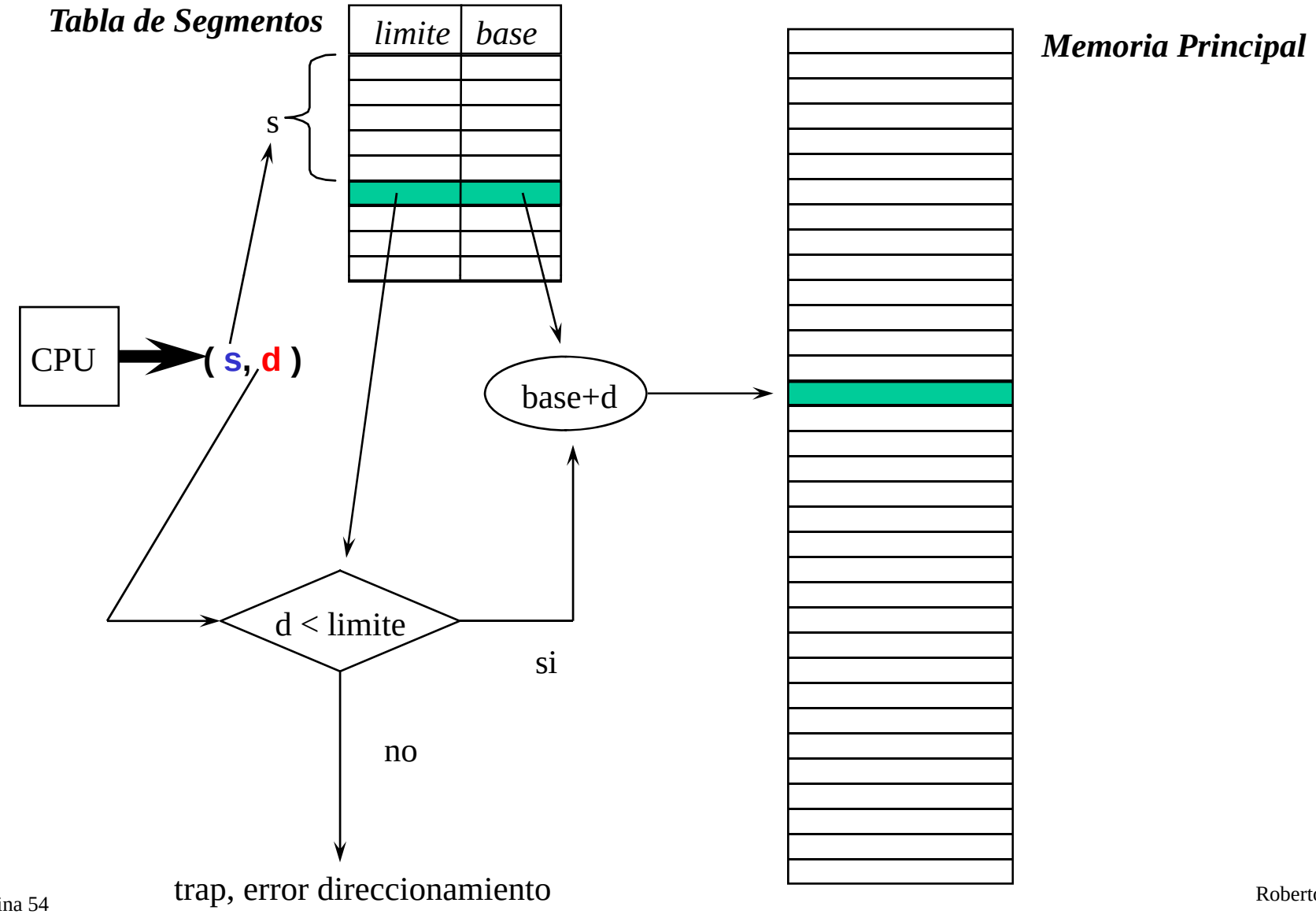
En un espacio unidimensional de direcciones con tablas crecientes, una tabla puede encimarse con otra.

Segmentos del ejemplo



Una memoria segmentada permite que cada tabla crezca o se reduzca en forma independiente de las demás

La tabla de segmentos



Ejemplo segmentación

Espacio direcciones virtuales

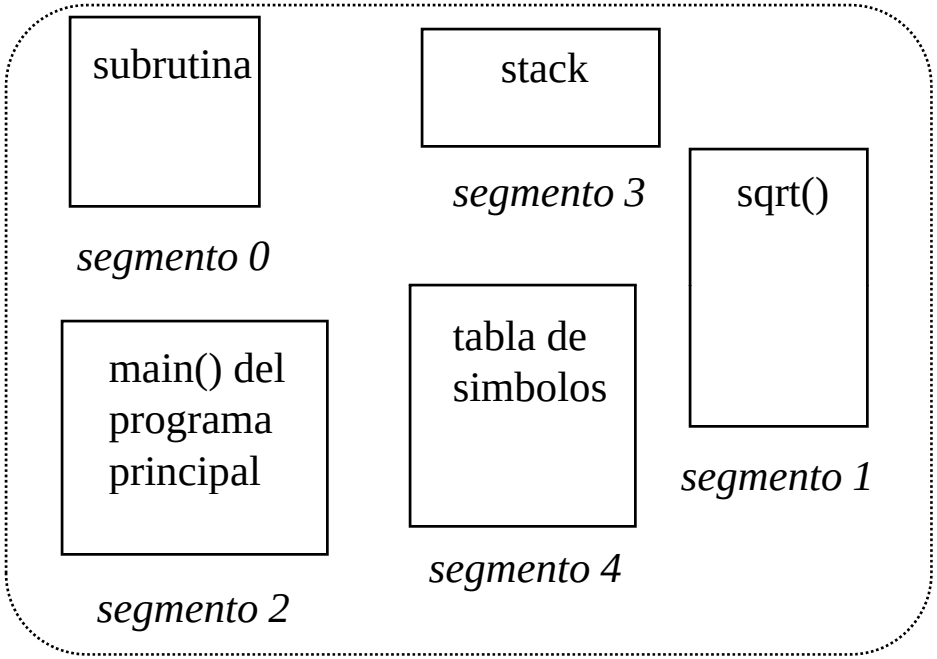
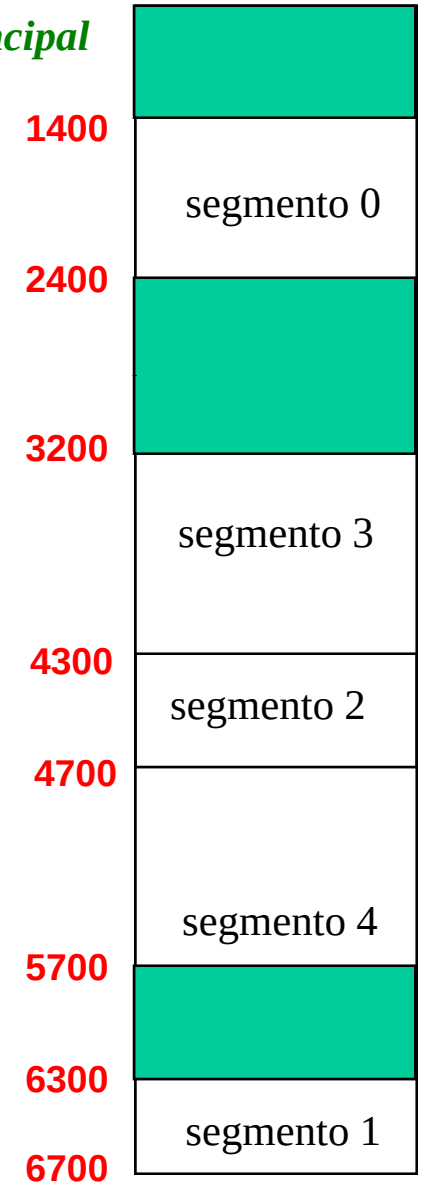


Tabla de Segmentos

	limite	*base*
0	1000	1400
1	400	6300
2	400	4300
3	1100	3200
4	1000	4700

Memoria Principal



Compartiendo segmentos

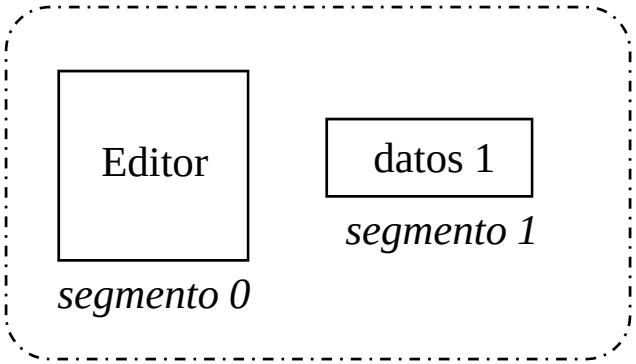


Tabla Segmentos P_1

	limite	*base*
0	25286	43062
1	4425	68348

Espacio direcciones virtuales de P_1

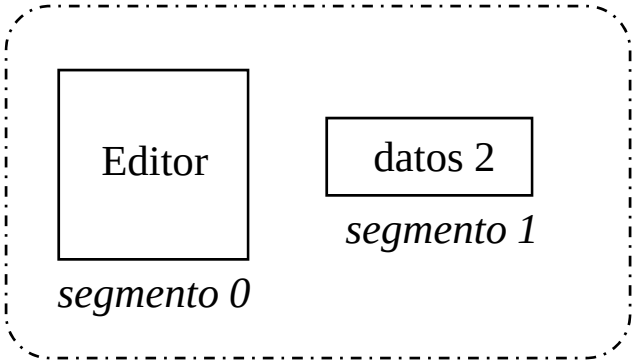
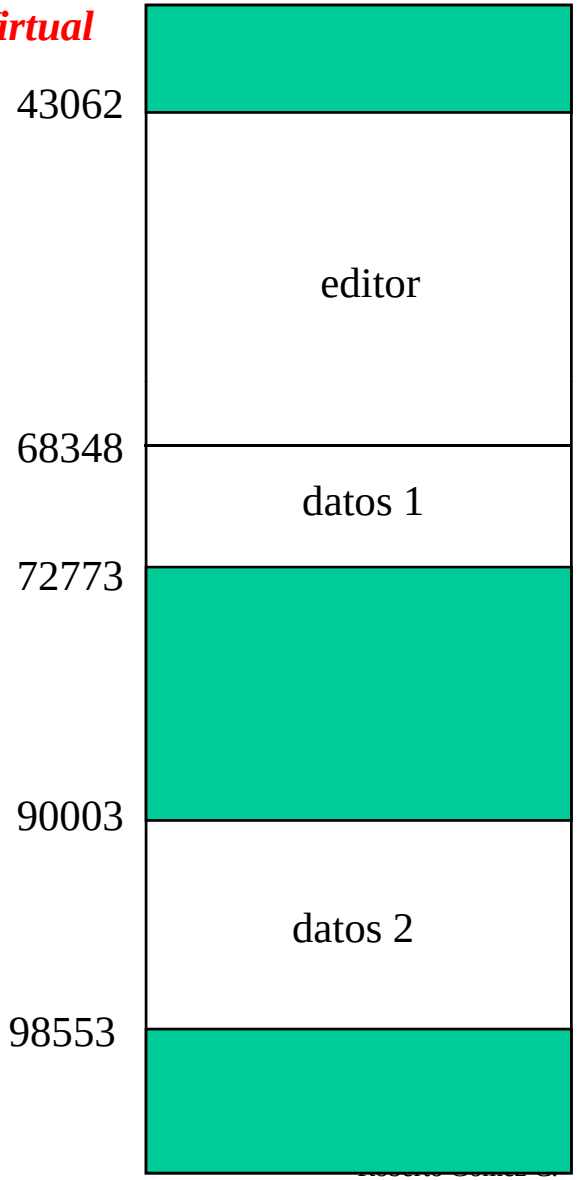


Tabla Segmentos P_2

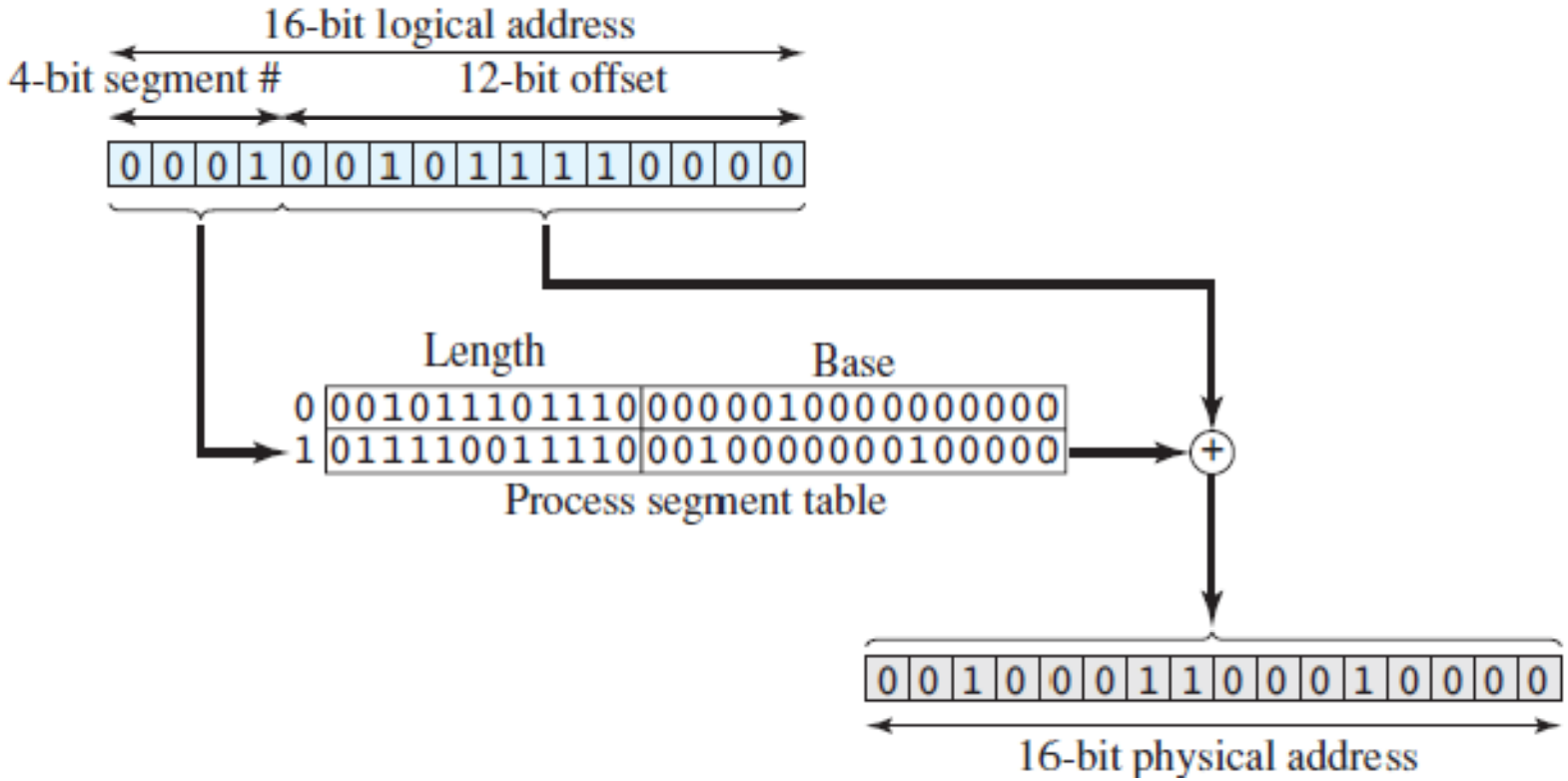
	limite	*base*
0	25286	43062
1	8550	90003

Espacio direcciones virtuales de P_2

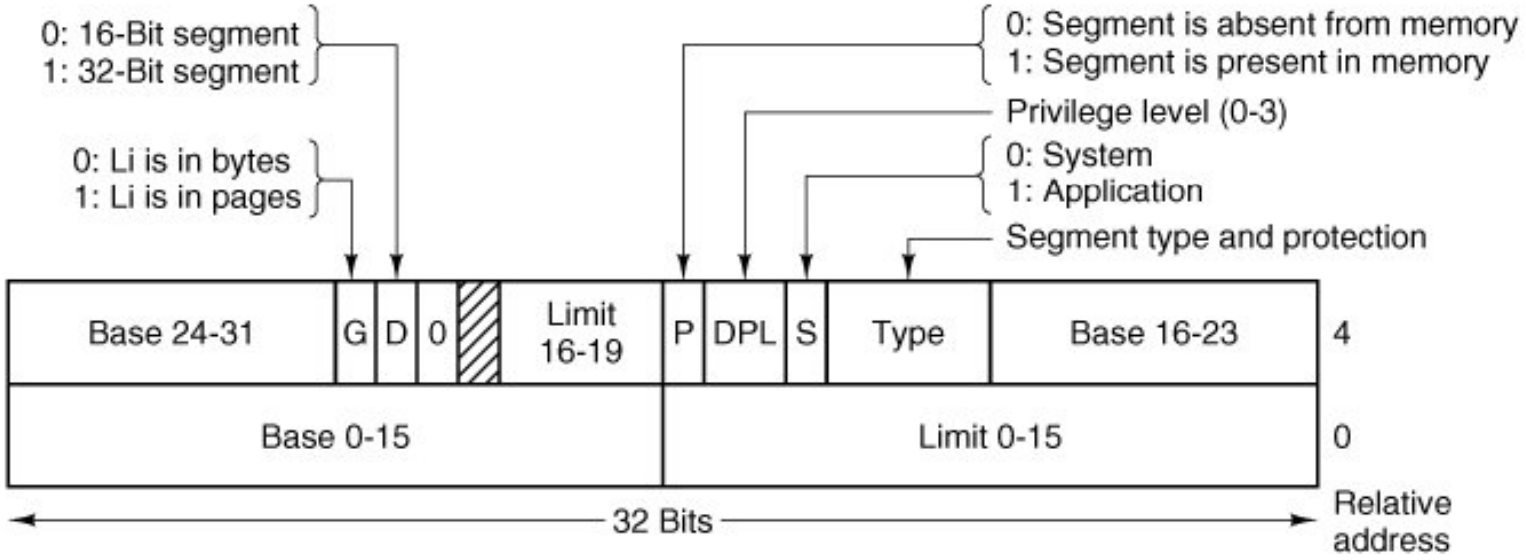
Memoria Virtual



Traducción dirección virtual a física



Ejemplo de descriptor de segmentos en un Pentium



Paginación vs Segmentación

Considerando	Páginación	Segmentación
¿Necesita saber el programador si está utilizando esta técnica?	**NO**	**SI**
¿Cuántos espacios lineales de direcciones existen?	**1**	**Muchos**
¿Puede el espacio total de direcciones exceder el tamaño de la memoria física?	**SI**	**SI**
¿Pueden distinguirse los procedimientos y los datos, además de protegerse en forma independiente?	**NO**	**SI**
¿Pueden adecuarse con facilidad las tablas con tamaños fluctantes?	**NO**	**SI**
¿Se facilita el uso de procedimientos compartidos entre los usuarios?	**NO**	**SI**