Problemas Resueltos

Capítulo 3. Memoria Virtual.

1.- Notas y criterios para los problemas:

-En las soluciones de los problemas se ha representado la división entera por "div" y el resto de la división entera por la siglas "mod".

-Para el bit de validez de los descriptores de páginas el criterio seguido es: 1 página en memoria y 0 la página no está en memoria.

-Se utiliza indistintamente marco, trama o "frame" para referencia un fragmento lógico de igual tamaño que la página en la memoria física.

2.- NOTA GENERAL: Tema 3 "Gestión de memoria: memoria virtual".

3.- Problemas y cuestiones de Paginación por demanda: algoritmo de reemplazo y traducción de direcciones.

1. ¿Existe alguna incompatibilidad entre los algoritmos de asignación (reparto) de marcos entre procesos (asiganción equitativa, proporcional, prioritaria) y el ámbito de reemplazo de páginas (local o global)?. Justifíque la respuesta.

Solución

El algoritmo de asignación de marcos establece la política de reparto de la memoria física a los procesos. Es decir, el número de marcos que debe tener asignado cada proceso. Si el algoritmo de reemplazo utilizado es global, el número de marcos asignado a cada proceso se puede modificar dinámicamente, por lo que el algoritmo de asignación podría no respetarse, o bien sólo tendría sentido inicialmente. Por el contrario, si el ámbito del algoritmo de reemplazo es local (sólo afecta a los marcos asignados al proceso), no hay ninguna incompatibilidad, ya que en todo momento se respeta la asignación de memoria.

2. Sea un sistema de gestión de memoria virtual con paginación por demanda, con un tamaño de memoria principal de 5000 palabras y un tamaño de página de 1000 palabras. En un momento determinado se tienen 3 procesos P1, P2 y P3 en el sistema que generan la siguiente secuencia de direcciones lógicas (se han representado pares compuestos por proceso y la dirección lógica): (P1,1023) (P2,0224) (P1,0783) (P3,3848) (P3,1089) (P3,0098) (P2,2345) (P1,0787) (P1,1654) (P3,2899) (P3,3008) (P3,1111). Realice un diagrama de la situación de cada página en memoria física en los siguientes casos:

a) Utilizando un algoritmo de reemplazo global según la política óptima.

b) Utilizando un algoritmo de reemplazo local según la política óptima.

Solución

La memoria física estará formada por 5 marcos que inicialmente supondremos vacíos.
El algoritmo óptimo elige como página a reemplazar aquella que tardará más tiempo en ser referenciada.
Mientras no se encuentren ocupados los marcos de memoria no será necesario reemplazar, por tanto, si indicamos con t el instante de tiempo, tendremos que las 5 primeras a páginas ocuparan los 5 marcos.

	t=0	t=1	t=3	t=4	t=5
Dir.	(P1,1023)	(P2,0224)	(P1,0783)	(P3,3848)	(P3,1089)
Nº marco	P1, Pág. 1	P2, Pág. 0	P1, Pág. 0	P3, Pág. 3	P3, Pág. 1
0	P1, Pág. 1	P1, Pág. 1	P1, Pág. 1	P1, Pág. 1	P1, Pág. 1
1	--	P2, Pág. 0	P2, Pág. 0	P2, Pág. 0	P2, Pág. 0 (*a)
2	--	--	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0
3	--	--	--	P3, Pág. 3	P3, Pág. 3
4	--	--	--	--	P3, Pág. 1 (*b)

A partir de este instante será necesario hacer reemplazos y seleccionar victima. Los * representan las victimas seleccionadas en el próximo instante de tiempo. Así (*a) es para la solución del apartado a) y (*b) para la solución del apartado b).

a) Algoritmo de reemplazo global.

t=6	t=7	t=8	t=9	t=10	t=11	t=12
(P3,0098) P3, Pág. 0	(P2,2345) P2, Pág. 2	(P1,0788) P1, Pág. 0	(P1,1654) P1, Pág. 1	(P3,2899) P3, Pág. 2	(P3,3008) P3, Pág. 3	(P3,1111) P3, Pág. 1
P1, Pág. 1	P1, Pág. 1	P1, Pág. 1	P1, Pág. 1*	P3, Pág. 2	P3, Pág. 2	P3, Pág. 2
P3, Pág. 0 *	P2, Pág. 2	P2, Pág. 2	P2, Pág. 2	P2, Pág. 2	P2, Pág. 2	P2, Pág. 2
P1, Pág. 0	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0
P3, Pág. 3	P3, Pág. 3	P3, Pág. 3	P3, Pág. 3	P3, Pág. 3	P3, Pág. 3	P3, Pág. 3
P3, Pág. 1	P3, Pág. 1	P3, Pág. 1	P3, Pág. 1	P3, Pág. 1	P3, Pág. 1	P3, Pág. 1

Los * representan las victimas seleccionadas en el próximo instante de tiempo.
En los instantes t=8, t=9,t=11 y t=12 no es necesario elegir victima ya que la página referenciada se encuentra en memoria.

b) Algoritmo de reemplazo local. Se selecciona la victima entre marcos ocupados por el mismo proceso.

t=6	t=7	t=8	t=9	t=10	t=11	t=12
(P3,0098) P3, Pág. 0	(P2,2345) P2, Pág. 2	(P1,0788) P1, Pág. 0	(P1,1654) P1, Pág. 1	(P3,2899) P3, Pág. 2	(P3,3008) P3, Pág. 3	(P3,1111) P3, Pág. 1
P1, Pág. 1	P1, Pág. 1	P1, Pág. 1	P1, Pág. 1	P1, Pág. 1	P3, Pág. 2	P3, Pág. 2
P2, Pág. 0 *	P2, Pág. 2	P2, Pág. 2	P2, Pág. 2	P2, Pág. 2	P2, Pág. 2	P2, Pág. 2
P1, Pág. 0	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0	P1, Pág. 0
P3, Pág. 3	P3, Pág. 3	P3, Pág. 3	P3, Pág. 3	P3, Pág. 3	P3, Pág. 3	P3, Pág. 3
P3, Pág. 0	P3, Pág. 1	P3, Pág. 1	P3, Pág. 1*	P3, Pág. 2	P3, Pág. 2*	P3, Pág. 1

3. Un determinado sistema operativo gestiona la memoria virtual mediante paginación por demanda. La dirección lógica tiene 24 bits, de los cuales 14 indican el número de página. La memoria física tiene 5 marcos. El algoritmo de reemplazo de páginas es el LRU LOCAL, y se ha implementado mediante un contador asociado a cada página que indica el instante de tiempo en que se referenció la página por última vez. Las tablas de páginas en el instante 16 son:

Tabla de páginas proceso A

	Marco	Bit de validez	Contador
0	1	v	10
1	2	v	15
2	-	i	6
3	-	i	5

Tabla de páginas proceso B

	Marco	Bit de validez	Contador
0	0	v	7
1	-	i	2
2	-	i	3
3	3	v	4
4	4	v	11

Indique las direcciones físicas generadas para la siguiente secuencia de direcciones lógicas: (A, 2900) (B, 1200) (A, 1850) (A, 3072) (B, 527) (B, 2987) (A, 27) (A, 2000) (B, 4800) (B, 1500).

Solución

Para poder traducir las direcciones lógicas a físicas hay que conocer el marco donde se ha ubicado la página correspondiente.

En este caso el tamaño de página es de 1024 palabras, ya que la dirección lógica es de 24 bits de los cuales 14 son para indicar la página ⇒ quedan 10 bits para el desplazamiento dentro de la página.

Consultando la tabla de páginas podemos componer el estado inicial de la memoria. Por comodidad al lado de cada marco se representará el contado LRU. Por lo tanto en la memoria principal tendremos:

Nº marco	Inicial t=15	Cont. LRU	A,2 t=16	Cont. LRU	B,1 t=17	Cont. LRU	A,1 t=18	Cont. LRU	A,3 t=19	Cont. LRU
0	B,0	7	B,0	7	B,0	7	B,0	7	B,0	7
1	A,0	10	A,2	16	A,2	16	A,2	16	A,3	19
2	A,1	15	A,1	15	A,1	15	A,1	18	A,1	18
3	B,3	4	B,3	4	B,1	17	B,1	17	B,1	17
4	B,4	11	B,4	11	B,4	11	B,4	11	B,4	11

	B,0 t=20	Cont. LRU	B,2 t=21	Cont. LRU	A,0 t=22	Cont. LRU	A,1 t=23	Cont. LRU	B,4 t=24	Cont. LRU	B,1 t=25	Cont. LRU
0	B,0	20	B,0	20	B,0	20	B,0	20	B,0	20	B,1	25
1	A,3	19	A,3	19	A,3	19	A,1	23	A,1	23	A,1	23
2	A,1	18	A,1	18	A,0	22	A,0	22	A,0	22	A,0	22
3	B,1	17	B,1	17	B,1	17	B,1	17	B,4	24	B,4	24
4	B,4	11	B,2	21	B,2	21	B,2	21	B,2	21	B,2	21

Dirección Lógica = U	Página = U div 1024	Desplazamiento= U mod 1024	Dirección física= Marco * 1024 + Desplazamiento
(A, 2900)	2	852	1*1024+852=1876
(B,1200)	1	176	3*1024+176=3248
(A,1850)	1	826	2*1024+826=2874
(A, 3072)	3	0	1*1024+0=1024
(B,527)	0	527	0*1024+527=527
(B,2987)	2	939	4*1024+939=5035
(A,27)	0	27	2*1024+27=2075
(A,2000)	1	976	1*1024+976=2000
(B,4800)	4	704	3*1024+704=3776
(B,1500)	1	476	0*1024+476=476

4. Un sistema de memoria virtual con paginación por demanda, tiene un tamaño de página de 512 palabras, una memoria virtual de 16 páginas numeradas del 0 al 15 una memoria física de 4 marcos *(frames*) numerados de 0 a 3. El contenido actual de la memoria es:

Nº marco	
0	Pág. 4 del proceso P
1	Pág. 9 del proceso P
2	Pág. 5 del proceso P
3	Pág. 1 del proceso P

a) Mostrar el contenido de la tabla de páginas .

b) Suponiendo un algoritmo de reemplazo de páginas con estrategia óptima, mostrar el contenido de la tabla de páginas, tras generar cada una de las siguientes direcciones lógicas: 1112, 1645, 2049, 622, 2776 .

c) Direcciones físicas equivalentes a las lógicas 1628, 851, 2700 y 2432.

d) ¿Qué pasa cuando se referencia la dirección lógica 1330?.

e) Si la página cargada en el marco 3 es un procedimiento que otro proceso Q quiere compartir, ¿dónde debe aparecer en la tabla de páginas de Q?

Solución

a) Mirando el contenido de la memoria principal y teniendo en cuenta que la tabla de páginas para el proceso tendrá un descriptor por cada página obtenemos que:

Tabla de páginas

	Marco	Bit de validez
0	-	0
1	3	1
2	-	0
3	-	0
4	0	1
5	2	1
6	-	0
7	-	0

Tabla de páginas

	Marco	Bit de validez
8	-	0
9	1	1
10	-	0
11	-	0
12	-	0
13	-	0
14	-	0
15	-	0

b) En primer lugar será necesario conocer a que página pertenece cada una de las direcciones lógicas, para ello realizaremos la operación:
Página = dirección_lógica *div* tamaño_página= dirección_lógica *div* 512
Desplazamiento = dirección_lógica *mod* tamaño_página

Dirección lógica	1112	1645	2049	622	2776
Página	2	3	4	1	5
desplazamiento	88	109	1	110	216

A continuación representaremos la memoria tras estas referencias, aplicando el algoritmo óptimo para reemplazo de páginas.

marco	inicial	Referencia Pág. 2 *	Referencia Pág. 3 *	Referencia Pág. 4	Referencia Pág. 1	Referencia Pág. 5
0	Pág. 4	Pág. 4	Pág. 4	Pág. 4	Pág. 4	Pág. 4
1	Pág. 9	Pág. 2	Pág. 3	Pág. 3	Pág. 3	Pág. 3
2	Pág. 5	Pág. 5	Pág. 5	Pág. 5	Pág. 5	Pág. 5
3	Pág. 1	Pág. 1	Pág. 1	Pág. 1	Pág. 1	Pág. 1

El * indica que hay fallo de página, por lo tanto hay que seleccionar una victima. Se elige como victima aquella que se tarda más tiempo en volver a referenciar o que ya no se vuelve a referenciar. Cuando se hace referencia a la página 2, se elige como victima la página 9, que ya no se va a volver a referenciar. Cuando se referencia a la página 3 se elige como victima a la 2 que ya no se vuelve a referenciar.
En este instante el contenido de la tabla de páginas será el siguiente:

Tabla de páginas

	Marco	Bit de validez
0	-	0
1	3	1
2	-	0
3	1	1
4	0	1
5	2	1
6	-	0
7	-	0

Tabla de páginas

	Marco	Bit de validez
8	-	0
9	1	0
10	-	0
11	-	0
12	-	0
13	-	0
14	-	0
15	-	0

c) La traducción de direcciones se realiza consultando la tabla de páginas:

Dir. lógica	Página	Desplazamiento	Dir. Física Marco*512 + desplazamiento
1628	3	92	1*512 + 92 = 604
851	1	339	3*512+339= 1875
2700	5	140	2*512+140= 1164
2432	4	384	0*512+384= 384

d) Si a continuación se referencia la dirección lógica 1330 tendremos un fallo de página, ya que esta dirección lógica pertenece a la página 2, cuyo bit de validez en la tabla de páginas está a invalido.

e) Depende del tipo de direccionamiento que se este empleando en el código de la página a compartir. Si se trata de direccionamiento relativo, esta procedimiento podría estar ubicado en cualquier página del proceso Q, y por tanto el descriptor correspondiente debería contener el marco 3. Si se trata de de direccionamiento absoluto, debería aparecer en la misma entrada donde aparece para el proceso P, ya que en caso contrario el código presente en ella podría no funcionar bien (Si hay una instrucción de salto que no es relativa al valor del contador de programa, y la posición destino de ese salto se encuentra en la misma página, se está obligando a que ese marco esté asociado siempre a la misma página para todos los procesos que lo utilicen). Por tanto, ha de aparecer asociado a la página 1 del proceso Q.

5. En un determinado sistema con memoria virtual con paginación por demanda, una dirección lógica consta de 16 bits, 10 de offset (desplazamiento) y 6 para el número de página. Se dispone de 4 marcos. Dada la siguiente secuencia de direcciones lógicas:

512 1102 2147 3245 5115 5200 4090 4207 1070 6200

7168 8200 7200 8300 9300 7410 8525 9700 5300 4387 1007

Se pide:

a) Dar la secuencia de referencias a páginas.

b) Contar el número de fallos de página suponiendo los algoritmos de reemplazo FIFO, LRU y ÓPTIMO.

Solución

Tamaño de página es de 1024 bytes = 2^{10}

a) La secuencia de referencia de páginas representa la secuencia de páginas que han sido accedidas durante cierto periodo de tiempo.

Página = dirección_lógica *div* 1024

Secuencia de referencia páginas = 0, 1, 2, 3, 4, 5, 3, 4, 1, 6, 7, 8, 7, 8, 9, 7, 8, 9, 5, 4, 0.

b) Se supone inicialmente la memoria vacía, con lo que habrá 4 fallos de página hasta llenarla.

Algoritmo de reemplazo FIFO.

Se elige como victima la página que mas tiempo lleva en memoria.

marco	Pág. 0,1,2,3	Pág. 4	Pág. 5	Pág. 3	Pág. 4	Pág. 1	Pág. 6	Pág. 7	Pág. 8
0	**Pág.0**	**Pág.4**	Pág.4	Pág.4	Pág.4	Pág.4	Pág.4	**Pág.7**	Pág.7
1	**Pág.1**	Pág.1	**Pág.5**	Pág.5	Pág.5	Pág.5	Pág.5	Pág.5	**Pág.8**
2	**Pág.2**	Pág.2	Pág.2	Pág.2	Pág.2	**Pág.1**	Pág.1	Pág.1	Pág.1
3	**Pág.3**	Pág.3	Pág.3	Pág.3	Pág.3	Pág.3	**Pág.6**	Pág.6	Pág.6

marco	Pág. 7	Pág. 8	Pág. 9	Pág. 7	Pág. 8	Pág. 9	Pág. 5	Pág. 4	Pág. 0
0	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	**Pág.4**	Pág.7
1	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	**Pág.0**
2	Pág.1	Pág.1	**Pág.9**	Pág.9	Pág.9	Pág.9	Pág.9	Pág.9	Pág.9
3	Pág.6	Pág.6	Pág.6	Pág.6	Pág.6	Pág.6	**Pág.5**	Pág.5	Pág.5

Los fallos de página se han resaltado en negrita. Total de fallos de página = 14.

Algoritmo de reemplazo LRU.
Se selecciona como victima la página de memoria que hace más tiempo que no se ha referenciado.

marco	Pág. 0,1,2,3	Pág. 4	Pág. 5	Pág. 3	Pág. 4	Pág. 1	Pág. 6	Pág. 7	Pág. 8
0	**Pág.0**	**Pág.4**	Pág.4	Pág.4	Pág.4	Pág.4	Pág.4	Pág.4	**Pág.8**
1	**Pág.1**	Pág.1	**Pág.5**	Pág.5	Pág.5	Pág.5	**Pág.6**	Pág.6	Pág.6
2	**Pág.2**	Pág.2	Pág.2	Pág.2	Pág.2	**Pág.1**	Pág.1	Pág.1	Pág.1
3	**Pág.3**	Pág.3	Pág.3	Pág.3	Pág.3	Pág.3	Pág.3	**Pág.7**	Pág.7

marco	Pág. 7	Pág. 8	Pág. 9	Pág. 7	Pág. 8	Pág. 9	Pág. 5	Pág. 4	Pág. 0
0	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	**Pág.0**
1	Pág.6	Pág.6	Pág.6	Pág.6	Pág.6	Pág.6	**Pág.5**	Pág.5	Pág.5
2	Pág.1	Pág.1	**Pág.9**	Pág.9	Pág.9	Pág.9	Pág.9	Pág.9	Pág.9
3	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	**Pág.4**	Pág.4

Los fallos de página se han resaltado en negrita. Total de fallos de página = 14.

Algoritmo de reemplazo ÓPTIMO.
Se selecciona como victima aquella página de memoria que más tiempo se va a tardar en referenciar o que nunca se va a referenciar.

marco	Pág. 0,1,2,3	Pág. 4	Pág. 5	Pág. 3	Pág. 4	Pág. 1	Pág. 6	Pág. 7	Pág. 8
0	**Pág.0**	Pág.0	**Pág.5**	Pág.5	Pág.5	Pág.5	Pág.5	Pág.5	Pág.5
1	**Pág.1**	Pág.1	Pág.1	Pág.1	Pág.1	Pág.1	**Pág.6**	Pág.6	**Pág.8**
2	**Pág.2**	**Pág.4**	Pág.4	Pág.4	Pág.4	Pág.4	Pág.4	Pág.4	Pág.4
3	**Pág.3**	Pág.3	Pág.3	Pág.3	Pág.3	Pág.3	Pág.3	**Pág.7**	Pág.7

marco	Pág. 7	Pág. 8	Pág. 9	Pág. 7	Pág. 8	Pág. 9	Pág. 5	Pág. 4	Pág. 0
0	Pág.5	Pág.5	Pág.5	Pág.5	Pág.5	Pág.5	Pág.5	Pág.5	**Pág.0**
1	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8	Pág.8
2	Pág.4	Pág.4	**Pág.9**	Pág.9	Pág.9	Pág.9	Pág.9	Pág.9	Pág.9
3	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	Pág.7	**Pág.4**	Pág.7

Los fallos de página se han resaltado en negrita. Total de fallos de página = 12.

6. Supongamos un sistema de gestión de memoria virtual basado en segmentación paginada, con un tamaño de página de 1000 palabras y una memoria principal de 3000 palabras. Las tablas de segmentos y las de página asociadas a cada segmento están ubicadas en registros (no ocupan espacio en memoria principal). Cada proceso consta de un espacio de direcciones lógicas dividido en 3 segmentos, uno para los datos (seg 0), otro para el código (seg 1) y el último para la pila (seg 2). En el sistema tenemos actualmente un solo proceso de las siguientes dimensiones (en palabras):

segmento	longitud
0	1300
1	2100
2	1000

Suponiendo que ha sido generada la siguiente secuencia de direcciones lógicas durante la ejecución del proceso en cuestión: (0, 0300) (0, 1200) (1, 0058) (0, 1112) (1, 1048) (2, 0354) (0, 1035) (0, 0036) (1, 2050) (1, 0128)

Calcular la correspondiente secuencia de direcciones físicas que se generará si se utiliza un algoritmo de reemplazo de páginas LRU.

Solución

Tamaño de página de 1000 palabras

	(0, 0300)	(0, 1200)	(1, 0058)	(0,1112)	(1, 1048)	(2, 0354)	(0, 1035)
Seg.	0	0	1	0	1	2	0
Pag.	0	1	0	1	1	0	1
Desp.	300	200	58	112	48	354	35

	(0, 0036)	(1, 2050	(1, 0128)
Seg.	0	1	1
Pag.	0	2	0
Desp.	36	50	128

Supongamos que la memoria principal se encuentra inicialmente vacía, con lo que insertaremos en ella 3 páginas. Los fallos de página se han resaltado en negrita

marco	Seg, pag (0,0), (0,1) (1,0)	Seg,pag 0,1	Seg,pag 1,1	Seg,pag 2,0	Seg,pag 0,1	Seg,pag 0,0	Seg,pag 1,2	Seg,pag 1,0
0	**(0,0)**	(0,0)	**(1,1)**	(1,1)	(1,1)	(0,0)	(0,0)	(0,0)
1	**(0,1)**	(0,1)	(0,1)	(0,1)	(0,1)	(0,1)	(0,1)	(1,0)
2	**(1,0)**	(1,0)	(1,0)	**(2,0)**	(2,0)	(2,0)	**(1,2)**	**(1,2)**

7. En un sistema paginado con 4 marcos en memoria, se hacen las siguientes referencias a páginas: {c, a, d, b, e, b, a, b, c, d}. Suponiendo que tenemos inicialmente en memoria las páginas {a, b, c, d} (en ese orden), analícese el contenido en memoria para cada una de las siguientes políticas de reemplazo:

a) ÓPTIMO.

b) FIFO.

c) LRU.

Solución

Las referencias a página son: {c, a, d, b, e, b, a, b, c, d}.

a) Algoritmo de reemplazo óptimo.

marco	inicial	Ref. c	Ref. a	Ref. d	Ref. b	Ref. e	Ref. a	Ref. b	Ref. c	Ref. d
0	a	a	a	a	a	a	a	a	a	**d**
1	b	b	b	b	b	b	b	b	b	b
2	c	c	c	c	c	c	c	c	c	c
3	d	d	d	d	d	**e**	e	e	e	e

Los fallos de página se han resaltado en negrita

b) Algoritmo de reemplazo FIFO.

marco	inicial	Ref. c	Ref. a	Ref. d	Ref. b	Ref. e	Ref. a	Ref. b	Ref. c	Ref. d
0	a	a	a	a	a	**e**	e	e	e	**d**
1	b	b	b	b	b	b	**a**	a	a	a
2	c	c	c	c	c	c	c	**b**	b	b
3	d	d	d	d	d	d	d	d	**c**	c

Los fallos de página se han resaltado en negrita

c) Algoritmo de reemplazo LRU.

marco	inicial	Ref. c	Ref. a	Ref. d	Ref. b	Ref. e	Ref. a	Ref. b	Ref. c	Ref. d
0	a	a	a	a	a	a	a	a	a	a
1	b	b	b	b	b	b	b	b	b	b
2	c	c	c	c	c	**e**	e	e	e	**d**
3	d	d	d	d	d	d	d	d	**c**	c

Los fallos de página se han resaltado en negrita

8. Sea un sistema de memoria virtual por paginación por demanda, en el que:

- Una dirección lógica consta de 12 bits, de los cuales 3 son para el número de página.
- Una dirección física contiene 11 bits.
- Existen dos procesos (A y B), y toda la memoria física se reparte entre estos dos procesos por igual.
- Se utiliza un algoritmo de reemplazo local de páginas LRU.

Dar, para la siguiente secuencia de direcciones lógicas, el correspondiente mapa de memoria: (A, 1035) (B, 312) (A, 530) (B, 780) (A, 600) (A, 2000) (B, 1400) (B, 927) (A, 1030) (A, 1720)

Nota: el par (a,x) indica (proceso, dir. lógica en decimal)

Solución

Dirección lógica de 12 bits = 3 bits número de pág. + 9 bits de desplazamiento.

Tamaño de página de 512 palabras.

La dirección física es de 11 bits = 2 bits número de marco + 9 bits de desplazamiento.

Los marcos se reparten por igual $\Rightarrow$ 2 marcos para el proceso A y 2 marcos para el proceso B.

	A, 1035	B, 312	A, 530	B,780	A, 600	A, 2000	B, 1400
marco	A, pág. 2	B, pág. 0	A, pág. 1	B, pág. 1	A, pág. 1	A, pág. 3	B, pág.2
0	**A,2**	**A,2**	A,2	A,2	A,2	**A,3**	A,3
1		**B,0**	B,0	B,0	B,0	B,0	**B,2**
2			**A,1**	A,1	A,1	A,1	A,1
3				**B,1**	B,1	B,1	B,1

	B, 927	A,1030	A, 1720
marco	B, pág. 1	A, pág. 1	A, pág. 3
0	A,3	A,3	A,3
1	B,2	B,2	B,2
2	A,1	A,1	A,1
3	B,1	B,1	B,1

Los fallos de página se han resaltado en negrita

9. Un sistema de gestión de memoria por segmentación paginada posee las siguientes características:

- Una dirección lógica se compone de 12 bits.
- Una dirección física se compone de 10 bits.
- El máximo núnmero de segmentos en un programa es 4 (segmentos 0 a 3).
- Hay 4 marcos en memoria.

Se pide:

a) ¿Cuál es el máximo número posible de páginas en un segmento?

b) Rellena las tablas de segmentos para los procesos A y B:

PROCESO A
TABLA DE SEGMENTOS

Seg 0	0	tamaño	Base tabla páginas
	419		
Seg 1	420		
	1299		

PROCESO B
TABLA DE SEGMENTOS

Seg 0	0	tamaño	Base tabla páginas
	300		
Seg 1			
	829		
Seg 2	1019		

c) Suponiendo el formato (proceso, segmento, desplazamiento dentro del segmento), se genera la siguiente secuencia de referencias a memoria: (A, 1, 27) (B, 2, 48) (A, 0, 411) (B, 2, 132) (B, 1, 400) (A, 1,428) (B, 0, 25) (B, 2, 177) (A, 1, 186) (B, 1, 280). La memoria está inicialmente vacía, y la política de reemplazo es FIFO con ámbito local. Determine a partir de la situación final cuál es la dirección física correspondiente a cada una de las siguientes direcciones lógicas: (A, 0, 350) (B, 2, 112) (A, 1, 422) (B, 0, 277) (A, 1, 604).

NOTAS:

- si la dirección es errónea, escriba ERROR.
- si está en disco, escriba DISCO.
- suponga que la memoria se llena de menor a mayor número de marco.

Solución

a) La respuesta es 4 páginas.

La dirección física está formada por 10 bits = 2 bits número marco + 8 bits desplazamiento.
La dirección lógica está formada por 12 bits = 2 para el segmento + 2 bits para el número de página + 8 bits para el desplazamiento de página.
Con dos bits para el número de página, el número máximo de páginas de un segmento es 4.

b) El tamaño de página es de 256 palabras. Suponiendo una única tabla de páginas para cada proceso, la tabla de segmentos quedará:

PROCESO A

TABLA DE SEGMENTOS

Seg 0	0	tamaño	Base tabla páginas
	419	420	Base + 0
	420	880	Base + 2
Seg 1			
	1299		

PROCESO B

TABLA DE SEGMENTOS

Seg 0	0	tamaño	Base tabla páginas
	300	300	Base + 0
Seg 1		530	Base + 2
	829	190	Base + 5
Seg 2	1019		

c) El algoritmo de reemplazo es FIFO local.

	A, 1, 27	B, 2, 48	A, 0, 411	B, 2, 132	B, 1, 400	B, 1, 428
marco	A, 1, pág. 0	B, 2, pág. 0	A, 0, pág. 1	B, 2, pág. 0	B, 1, pág. 1	B, 1, pág. 1
0	**A, 1, pág. 0**	A, 1, pág. 0	A, 1, pág. 0	A, 1, pág. 0	A, 1, pág. 0	A, 1, pág. 0
1		**B, 2, pág. 0**	B, 2, pág. 0	B, 2, pág. 0	B, 2, pág. 0	B, 2, pág. 0
2			**A, 0, pág. 1**	A, 0, pág. 1	A, 0, pág. 1	A, 0, pág. 1
3					**B, 1, pág. 1**	B, 1, pág. 1

	B, 0, 25	B, 2, 177	A, 1, 186	B, 1, 280
marco	B, 0, pág. 0	B, 2, pág. 0	A, 1, pág. 0	B, 1, pág. 1
0	A, 1, pág. 0	A, 1, pág. 0	A, 1, pág. 0	A, 1, pág. 0
1	**B, 0, pág. 0**	B, 0, pág. 0	B, 0, pág. 0	**B, 1, pág. 1**
2	A, 0, pág. 1	A, 0, pág. 1	A, 0, pág. 1	A, 0, pág. 1
3	B, 1, pág. 1	**B, 2, pág. 0**	B, 2, pág. 0	B, 2, pág. 0

Los fallos de página se han resaltado en negrita

La traducción de las direcciones lógicas a físicas será la siguiente:

Dirección lógica	A, 0, 350 pág.1,des. 94	B, 2, 112 pág.0,des. 112	A, 1, 422 pág.1,des. 166	B, 0, 277 pág.1,des. 21	A, 1, 604 pág.2,des. 92
Dirección física	2*256+94= 606	3*256+112= 880	DISCO	DISCO	DISCO

10. Sea un sistema con memoria virtual por segmentación paginada, con algoritmo de reemplazo óptimo local. La memoria principal está organizada en 4 marcos de 512 palabras cada uno. En un instante T_i únicamente tenemos a los procesos A y B en el sistema, y el S.O. mantiene la siguiente información:

Proceso A

Tabla segmentos

tamaño	btp
710	0
200	2
1040	3

Tabla de páginas

0	DISCO
1	DISCO
2	0
3	DISCO
4	3
5	DISCO

Proceso B

Tabla segmentos

tamaño	btp
100	0
600	1
250	3

Tabla de páginas

0	1
1	DISCO
2	2
3	DISCO

NOTA.- btp (base de la tabla de páginas) representa el desplazamiento en la tabla de páginas (una tabla de páginas para todos los segmentos) de un mismo proceso.

a) Con esta situación, ¿qué porcentaje de memoria se desperdicia debido a la fragmentación externa?; ¿y debido a la interna?.

b) A partir de ese instante, continúa la ejecución concurrente de ambos procesos, y se generan las siguientes direcciones lógicas: (A, 0, 100) (A, 1, 25) (B, 2, 200) (A, 2, 1035) (B, 0, 10) (A, 0, 450) (B,1, 580). Considere el instante T_j, que es el que ocurre tras generar las direcciones lógicas anteriores. Calcule, para cada una de las siguientes direcciones lógicas, la dirección física equivalente, considerando la situación del instante T_j: (A, 2, 1030) (B, 0, 400) (A, 0, 700) (B, 2, 100) (A, 0, 300) (B, 0, 92)

NOTA.- Si es una dirección errónea, conteste ERROR, y si es correcta pero la dirección equivalente no está en memoria física, escriba DISCO.

Solución

a) En segmentación paginada sólo hay fragmentación interna. El tamaño de página es de 512 palabras, por tanto la segmentación interna vendrá dada por la diferencia entre el tamaño de página y el número de palabras ocupadas.

marco	Contenido de memoria	Fragmentación interna
0	Proceso A, seg 1, pág. 0	512-200=312 palabras
1	Proceso B, seg 0, pág. 0	412 palabras
2	Proceso B, seg 1, pág. 1	1024-600= 424 palabras
3	Proceso A, seg 2, pág. 0	0 palabras

El tamaño total de memoria física es de 512 + 4= 2 K palabras.
La fragmentación interna que hay en la memoria es de 312+424=736 palabras.
El porcentaje de fragmentación interna vendrá dado por 736/2048 *100= 35,93 %.

b) El algoritmo de reemplazo es OPTIMO LOCAL.

marco	Contenido de memoria	A, 0, 100 (pág. 0)	A, 1, 25 (pág. 0)	B, 2, 200 (pág. 0)	A, 2, 1035 (pág. 2)
0	A, 1, pág. 0	A, 1, pág. 0	A, 1, pág. 0	A, 1, pág. 0	**A, 2, pág. 2**
1	B, 1, pág. 0	B, 1, pág. 0	B, 1, pág. 0	**B, 2, pág. 0**	B, 2, pág. 0
2	B, 1, pág. 1	B, 1, pág. 1	B, 1, pág. 1	B, 1, pág. 1	B, 1, pág. 1
3	A, 2, pág. 0	**A, 0, pág. 0**	A, 0, pág. 0	A, 0, pág. 0	A, 0, pág. 0

marco	B, 0, 10 (pág. 0)	A, 0, 450 (pág. 0)	B, 1, 580 (pág. 1)
0	A, 2, pág. 2	A, 2, pág. 2	A, 2, pág. 2
1	**B, 0, pág. 0**	B, 0, pág. 0	B, 0, pág. 0
2	B, 1, pág. 1	B, 1, pág. 1	B, 1, pág. 1
3	A, 0, pág. 0	A, 0, pág. 0	A, 0, pág. 0

Los fallos de página se han resaltado en negrita.

En esta situación de la memoria la traducción de direcciones lógicas a físicas quedará:

Dirección lógica	A, 2, 1030 pág.2,des. 6	B, 0, 400 pág.0,des. 400	A, 0, 700 pág.1,des. 188	B, 2, 100 pág.0,des. 100
Dirección física	512*0+6=6	512*1+400=912	DISCO	DISCO

Dirección lógica	A, 0, 300 pág.0,des. 300	B, 0, 92 pág.0,des. 92
Dirección física	512*3+300= 1836	512*1+92= 604

11. Dado un sistema con memoria virtual con paginación por demanda y el siguiente fragmento de código, con su correspondiente traducción a ensamblador:

```
FOR i:= 1 TO 10 DO
FOR j:= 1 TO 10 DO
C[i, j] := C[i, j] + A[i, j] + B[i, j];
```

En ensamblador

```
INICIO:      Inicializar R0 a 11
BUCLE_i:     Decrementar R0 una unidad
             Salto si cero a FIN_BUCLE_i
             Inicializar R1 a 11
BUCLE_j:     Decrementar R1 una unidad
             Salto si cero a FIN_BUCLE_j
             Cargar R3 con A[R0, R1]
             Cargar R4 con C[R0, R1]
             Cargar R5 con B[R0, R1]
             R3 <- R3 + R5
             R4 <- R4 + R3
             Grabar en C[R0, R1] el contenido de R4
             Salto a BUCLE_j
FIN_BUCLE_j: Salto a BUCLE_i
FIN_BUCLE_i: ...
```

Donde cada instrucción ocupa una palabra y R0 ... R5 son registros del procesador. Suponiendo que la paginación del programa ha quedado como:

MATRIZ C página 0
MATRIZ A página 1
MATRIZ B página 2
Código y variables página 3

a) ¿Qué secuencia forman las páginas referenciadas a cada iteración del BUCLE_j?

b) Suponiendo que cuando se vaya a ejecutar este fragmento de código (desde INICIO hasta FIN_BUCLE_i), la tabla de páginas contiene la siguiente información:

Tabla de páginas

0	1	v
1	-	i
2	2	v
3	0	v

y que se tienen únicamente 3 marcos en memoria, ¿cuántos fallos de página provocaría esta suma de matrices cuando se utiliza una política de reemplazo de páginas ÓPTIMO GLOBAL?. Calcúlese el valor de la tasa de fallos de página (p).

Solución

a) En primer lugar necesitaremos conocer cuál es la secuencia de direcciones emitida en cada iteración del BUCLE_j. Por cada instrucción hay que acceder a la página del código, que es la 3 (para poder leer la instrucción y llevarla al procesador). Cuando la instrucción accede a algún registro no hay que llevar a cabo ningún acceso adicional a memoria, ya que los registros se encuentran en el procesador. Sin embargo, cuando se acceda a una matriz habrá que leer la página correspondiente (la 0 para la matriz C, la 1 para la A y la 2 para la B).

Por tanto, la secuencia de accesos a páginas que se efectúa es la siguiente:

Instrucción	Páginas
Decrementar R1 una unidad	3
Salto si cero a FIN_BUCLE_j	3
Cargar R3 con A[R0, R1]	3,1
Cargar R4 con C[R0, R1]	3,0
Cargar R5 con B[R0, R1]	3,2
R3 <- R3 + R5	3
R4 <- R4 + R3	3
Grabar en C[R0, R1] el contenido de R4	3,0
Salto a BUCLE_j	3

Con ello, la secuencia ha sido 3, 3, 3, 1, 3, 0, 3, 2, 3, 3, 3, 0, 3. Es decir, un total de 13 accesos. Esto da origen a la siguiente serie de referencias: 3, 1, 3, 0, 3, 2, 3, 0, 3.

b)Las instrucciones que están incluidas en el BUCLE_i, pero no el BUCLE_j, son instrucciones que no acceden a ninguna de las matrices. Por tanto, únicamente necesitan un acceso a la página 3 de código, cada una de ellas. Por tanto la serie de referencias es la misma que la del apartado anterior. Los fallos de página que se dan en cada iteración del BUCLE_j siguiendo el algoritmo de reemplazo ÓPTIMO GLOBAL vendrá dado por:

marco	inicio	Pág. 3	Pág. 1	Pág. 3	Pág. 0	Pág. 3	Pág. 2	Pág. 3	Pág. 0	Pág. 3
0	3	3	3	3	3	3	3	3	3	3
1	0	0	0	0	0	0	0	0	0	0
2	2	2	**1**	1	1	1	**2**	2	2	2

Los fallos de página se han resaltado en negrita
Solo hay dos fallos de página en cada iteración del BUCLE_i. El contenido de la memoria al final de cada iteración es el mismo que el del inicio, por tanto como el bucle se realiza10 veces, esto implica que el número de fallos de página en cada ejecución del código será de 20.
La tasa de fallos viene dada por:

$$p = \frac{fallos\ de\ página}{número\ total\ de\ referencias}$$

Necesitaremos conocer por tanto el número total de referencias que se han de realizar para ejecutar este código.

Instrucción	Páginas	Cuantas páginas	Veces ejecuta	total
Inicialiar R0 a 11	3	1	1	1
Decrementar R0 en una unidad	3	1	11	11
Salto si cero a FIN_BUCLE_i	3	1	11	11
Inicializar R1 a 11	3	1	10	10
Decrementar R1 una unidad	3	1	110	110
Salto si cero a FIN_BUCLE_j	3	1	110	110
Cargar R3 con A[R0, R1]	3,1	2	100	200
Cargar R4 con C[R0, R1]	3,0	2	100	200
Cargar R5 con B[R0, R1]	3,2	2	100	200
R3 <- R3 + R5	3	1	100	100
R4 <- R4 + R3	3	1	100	100
Grabar en C[R0, R1] el contenido de R4	3,0	2	100	200
Salto a BUCLE_j	3	1	100	100
Salto a BUCLE_i	3	1	10	10
Total				1363

Obsérvese que las dos primeras instrucciones de cada bucle (tanto del interno como del externo) se ejecutan una vez más que el resto. Esto es debido a que, para comprobar la finalización del bucle, se necesita efectuar de nuevo la operación de decremento del registro y la de comprobación. Esto hace que las instrucciones *Decrementar R0 una unidad* y *Salto si cero a FIN_BUCLE_i* se ejecuten 11 veces, mientras que *Decrementar R1 una unidad* y *Salto si cero a FIN_BUCLE_j* deban ejecutarse 110 (ya que iteran 11 veces cada vez que se ejecuta el BUCLE_j, pero al estar encerrado dentro del BUCLE_i, se repite su ejecución 10 veces).
Por tanto, la tasa de fallos de página para este código vendrá dada por:

$$p = \frac{fallos\ de\ página}{número\ total\ de\ referencias} = \frac{20}{1363} = 0.014735 = 1.46\%$$

12. Sea un sistema de memoria virtual con segmentación paginada y Algoritmo de Reemplazo LRU Global. El tamaño de página es de 512 palabras y en memoria física hay en total 3 tramas, inicialmente vacías. Suponiendo que hay dos procesos en el sistema (A y B) con las siguientes tablas de segmentos (especificando únicamente la longitud de cada uno):

PROCESO A
TABLA DE SEGMENTOS

	tamaño
0	160
1	200
2	1000

PROCESO B
TABLA DE SEGMENTOS

	tamaño
0	300
1	640

Se generan las siguientes direcciones lógicas: (B,0,209) (A,0,27) (A,1,171) (B,0,180) (A,1,25) (A,2,638) (A,2,815) (B,0,200) (A,0,155) (B,0,193) (B,1,608) (A,2,715). Supóngase que las tramas se asignan de menor a mayor (primero la 0). Calcular las direcciones físicas a que dan lugar ,y el número de fallos de página provocados.

Solución

El algoritmo de reemplazo es el LRU GLOBAL y el tamaño de página de 512 palabras.

marco	B, 0, 209 (pág. 0)	A, 0, 27 (pág. 0)	A, 1, 171 (pág. 0)	B, 0, 180 (pág. 0)	A, 1, 25 (pág. 0)	A, 2, 638 (pág. 1)	A, 2, 815 (pág. 1)
0	**B, 0, 0**	B, 0, 0	B, 0, 0	B, 0, 0	B, 0, 0	B, 0, 0	B, 0, 0
1		**A, 0, 0**	A, 0, 0	A, 0, 0	A, 0, 0	**A, 2, 1**	A, 2, 1
2			**A, 1, 0**	A, 1, 0	A, 1, 0	A, 1, 0	A, 1, 0

marco	B, 0, 200 (pág. 0)	A, 0, 155 (pág. 0)	B, 0, 193 (pág. 0)	B, 1, 608 (pág. 1)	A, 2, 715 (pág. 1)
0	B, 0, 0	B, 0, 0	B, 0, 0	B, 0, 0	B, 0, 0
1	A, 2, 1	A, 2, 1	A, 2, 1	**B, 1, 1**	B, 1, 1
2	A, 1, 0	**A, 0, 0**	A, 0, 0	A, 0, 0	**A, 2, 1**

El numero total de fallos de página son 7 y han sido resaltados en negrita.

Dirección Lógica	Dirección física	Dirección Lógica	Dirección física	Dirección Lógica	Dirección física
B, 0, 209	209	A,1,25	1049	A,0,155	1179
A, 0, 27	539	A,2,638	638	B,0,193	193
A,1,171	1195	A,2,815	815	B,1,608	608
B,0,180	180	B,0,200	200	A,2,715	1227

13. Un determinado sistema operativo gestiona la memoria virtual mediante segmentación paginada. Una dirección lógica tiene 24 bits, de los cuales 5 indican el número de segmento. Una dirección física tiene 12 bits, y el tamaño de la trama es de 1024 octetos. Suponga que hay 2 procesos en el sistema, con las siguientes tabla de segmentos:

Proceso A	
Tabla de segmentos	
	tamaño
0	100
1	2100
2	1120
3	3450

Proceso B	
Tabla de segmentos	
	tamaño
0	950
1	4120
2	512

Las tablas de segmentos y páginas no consumen memoria física. Inicialmente la memoria está vacía. Indique las direcciones físicas generadas, si se utiliza un algoritmo de reemplazo de páginas LRU GLOBAL, para la siguiente secuencia de direcciones lógicas: (A, 3, 2100) (A, 2, 1100) (B, 0, 800) (B, 2, 300) (A, 0, 50) (B, 0, 300) (A, 2, 1024) (A, 1, 2000) (B, 1, 3120) (A, 2, 1050) (B, 0, 800) (A, 1, 2100).

Solución

Cada dirección física tiene 12 bits, de ellos 10 se emplean para desplazamiento dentro de la trama ($\log_2 1024 = 10$), y el resto 2 para especificar el número de trama. La memoria principal estará formada por: $2^2 = 4$ tramas.

Las direcciones lógicas que especifica el enunciado están en el formato de proceso, segmento y desplazamiento dentro del segmento. Para que una dirección lógica sea correcta ha de ser menor que el tamaño del segmento al que pertenece. Como se trata de segmentación paginada es necesario identificar la página del segmento a la que corresponde cada dirección lógica para ubicarla en memoria.

Considerando la memoria principal inicialmente vacía y aplicando el algoritmo LRU GLOBAL se obtiene la siguiente ocupación de memoria para las referencias anteriores:

marco	B, 3, 2100 (pág. 2)	A, 2, 1100 (pág. 1)	B, 0, 800 (pág. 0)	B, 2, 300 (pág. 0)	A, 0, 50 (pág. 0)	B, 0, 300 (pág. 0)
0	**B,3,2**	B,3,2	B,3,2	B,3,2	**A,0,0**	A,0,0
1		**A,2,1**	A,2,1	A,2,1	A,2,1	A,2,1
2			**B,0,0**	B,0,0	B,0,0	B,0,0
3				**B,2,0**	B,2,0	B,2,0

marco	A, 2, 1024 (pág. 1)	A, 1, 2000 (pág. 1)	B, 0, 3120 (pág. 3)	A, 2, 1050 (pág. 1)	B, 0, 800 (pág. 0)
0	A,0,0	A,0,0	**B,0,3**	B,0,3	B,0,3
1	A,2,1	A,2,1	A,2,1	A,2,1	A,2,1
2	B,0,0	B,0,0	B,0,0	B,0,0	B,0,0
3	B,2,0	**A,1,1**	A,1,1	A,1,1	A,1,1

El numero total de fallos de página son 7 y han sido resaltados en negrita. De estos fallos de páginas sólo 3 han sido con reemplazo.

Para traducir direcciones lógicas a físicas se emplea la siguiente expresión:

$$dirección\_física = marco * 1024 + desplazamiento\_de\_página$$

Dirección lógica			Dirección lógica		
Pr, seg, desp_seg.	Pr, seg, pág, desp_pag	Dirección Física	Pr, seg, desp_seg.	Pr, seg, pág, desp_pág	Dirección Física
(A, 3, 2100)	(A, 3, 2, 52)	52	(A, 2, 1024)	(A, 2, 1, 0)	1024
(A, 2, 1100)	(A, 2, 1, 76)	1100	(A, 1, 2000)	(A, 1, 1, 976)	4048
(B, 0, 800)	(B, 0, 0, 800)	2848	(B, 1, 3120)	(B, 1, 3, 48)	48
(B, 2, 300)	(B, 2, 0, 300)	3372	(A, 2, 1050)	(A, 2, 1, 26)	1050
(A, 0, 50)	(A, 0, 0, 50)	50	(B, 0, 800)	(B, 0, 0, 800)	2848
(B, 0, 300)	(B, 0, 0, 300)	2348	(A, 1, 2100)	ERROR	ERROR

14. Sea un sistema de memoria virtual con paginación por demanda, con algoritmo de reemplazo LRU local y con las siguientes características:

- Tamaño de Página: 1024 palabras.
- Número de tramas en memoria: 5.
- Asignación de tramas: proporcional al tamaño del proceso (medido en páginas).
- El tamaño de las tablas de páginas está ajustado al tamaño de cada proceso.

Suponiendo dos procesos (A y B) en el sistema, donde las tablas de páginas contiene inicialmente (siendo v=válido, i=no válido):

	Proceso A	
	Tabla de Páginas	
	marco	Bit validez
0		i
1	4	v
2		i
3	3	v
4		i
5		i
6		i
7		i

	Proceso B	
	Tabla de Páginas	
	marco	Bit validez
0		i
1	0	v
2		i
3		i
4	2	v

Siendo, hasta este momento, las últimas referencias a páginas: (B,1), (A,3). Se generan las siguientes direcciones lógicas: (B,2109) (A,207) (A,4444) (B,2222) (A,2345) (B,3471) (A,88) (B,2000) (A,1055) (B,1093) (B,5008) (A,7815)

a) Aplicar el algoritmo de reemplazo, mostrando el contenido de memoria en una tabla.

b) Calcular las direcciones físicas a que dan lugar esas mismas direcciones lógicas cuando se atienden, y el número de fallos de página provocados.

Solución

a) EL proceso A tiene 8 páginas y el proceso B 5 páginas, por tanto la asignación de tramas proporcional al tamaño del proceso quedará:

$$N^o\_tramas\_A = \frac{núm\_pág\_A}{núm\_pág\_A + núm\_pág\_B} * núm\_tramas = \frac{8}{8+5} * 5 = 3{,}07$$

$$N^o\_tramas\_B = \frac{núm\_pág\_B}{núm\_pág\_A + núm\_pág\_B} * núm\_tramas = \frac{5}{8+5} * 5 = 1{,}92$$

Por tanto de las 5 tramas que tiene la memoria principal 3 se asignan al proceso A y 2 al proceso B.

Teniendo en cuenta las tablas de páginas, podemos saber que páginas hay en memoria y en que marcos a partir de ese momento se aplica para las referencias propuestas un algoritmo de reemplazo LRU local obteniéndose:

marco	Inicial	B ,2109 (pág. 2)	A,207 (pág. 0)	A,4444 (pág. 4)	B,2222 (pág 2)	A,2345 (pág. 2)	B,3471 (pág. 3)
0	B,1	B,1	B,1	B,1	B,1	B,1	**B,3**
1			**A,0**	A,0	A,0	A,0	A,0
2	B,4	**B,2**	B,2	B,2	B,2	B,2	B,2
3	A,3	A,3	A,3	A,3	A,3	**A,2**	A,2
4	A,1	A,1	A,1	**A,4**	A,4	A,4	A,4

marco	A,88 (pág. 0)	B,2000 (pág. 1)	A,1055 (pág. 1)	B,1093 (pág. 1)	B,5008 (pág. 4)	A,7815 (pág. 7)
0	B,3	B,3	B,3	B,3	**B,4**	B,4
1	A,0	A,0	A,0	A,0	A,0	A,0
2	B,2	**B,1**	B,1	B,1	B,1	B,1
3	A,2	A,2	A,2	A,2	A,2	**A,7**
4	A,4	A,4	**A,1**	A,1	A,1	A,1

El numero total de fallos de página son 9 y han sido resaltados en negrita. De estos fallos de páginas sólo 7 han sido con reemplazo.

Las direcciones físicas resultantes se calculan en la tabla siguiente:

Dirección lógica (pág., despla)	B ,2109 (2, 61)	A,207 (0,207)	A,4444 (4, 348)	B,2222 (2,174)	A,2345 (2,297)	B,3471 (3,399)
Dirección física	DISCO	1024+207= 1231	DISCO	DISCO	DISCO	DISCO

Dirección lógica (pág., despla)	A,88 (0,88)	B,2000 (1,976)	A,1055 (1,31)	B,1093 (1,69)	B,5008 (4,912)	A,7815 (7,647)
Dirección física	1024+88= 1112	2048+976= 3024	4096+31= 4127	2048+69= 2117	0+912= 912	3072+647= 3719

15. Sea un sistema de memoria virtual por segmentación paginada. La memoria está organizada en 5 marcos de 512 palabras cada uno. En este momento únicamente se tienen 2 procesos: A y B. La situación inicial de los marcos es:

Memoria	
Nº marco	(X,S,P)
0	—
1	A,2,0
2	B,1,0
3	B,3,1
4	A,1,1

correspondiendo (X,S,P) al proceso X, segmento S, página P (dentro del segmento).

a) Represente las tablas de segmentos y páginas correspondientes a esta situación inicial, sabiendo que:

- A está formado por 3 segmentos de tamaños 800, 600 y 500.
- B está formado por 4 segmentos de tamaños 450, 600, 100 y 2000.

b) Si se utiliza un algoritmo de reemplazo ÓPTIMO global, y a partir de la situación inicial se generan las siguientes direcciones lógicas: (B,3,1257) (A,0,12) (A,0,789) (B,3,789) (A,1,514) (A,2,3) (B,0,415) (B,3,1100). Represente el mapa de ocupación de la memoria e indique el número de fallos de página producido.

Solución

a) Tabla de páginas y de segmentos.

Proceso A

Tabla de segmentos

	tamaño	base
0	800	0
1	600	2
2	500	4

Tabla de páginas

	marco	Bit de validez
0		i
1		i
2		i
3	4	v
4	1	v

Proceso B

Tabla de segmentos

	tamaño	base
0	450	0
1	600	1
2	100	3
3	2000	4

Tabla de páginas

	marco	Bit de validez
0		i
1	2	v
2		i
3		i
4		i
5	3	v
6		i
7		i

b)Utilizando el algoritmo de reemplazo óptimo global se obtiene:

marco	Inicial	B ,3,1257 (pág. 2)	A,0,12 (pág. 0)	A,0,789 (pág. 1)	B,3,789 (pág 1)	A,1,514 (pág. 1)
0		**B,3,2**	B,3,2	B,3,2	B,3,2	B,3,2
1	A,2,0	A,2,0	A,2,0	A,2,0	A,2,0	A,2,0
2	B,1,0	B,1,0	**A,0,0**	**A,0,1**	A,0,1	A,0,1
3	B,3,1	B,3,1	B,3,1	B,3,1	B,3,1	B,3,1
4	A,1,1	A,1,1	A,1,1	A,1,1	A,1,1	A,1,1

marco	A,2,3 (pág. 0)	B,0,415 (pág. 0)	B,3,1100 (pág. 2)
0	B,3,2	B,3,2	B,3,2
1	A,2,0	**B,0,0**	B,0,0
2	A,0,1	A,0,1	A,0,1
3	B,3,1	B,3,1	B,3,1
4	A,1,1	A,1,1	A,1,1

El numero total de fallos de página son 4 y han sido resaltados en negrita. De estos fallos de páginas sólo 3 han sido con reemplazo.

16. Sea un sistema de memoria virtual por paginación por demanda. Las direcciones lógicas tienen 11 bits, de los cuales 2 se interpretan internamente como número de página. La memoria está organizada en 3 marcos. En este momento únicamente tenemos 2 procesos: A y B. La situación inicial de las páginas es:

Proceso A
Tabla de páginas

	marco
0	0
1	1
2	i
3	i

Proceso B
Tabla de páginas

	marco
0	2
1	i
2	i
3	i

a) Si se obtuviera de la situación inicial cada una de las siguientes direcciones físicas (sin relación de orden entre sí) 845, 623, 1024, 1603, calcule las direcciones lógicas de las que proceden.

b) Si se utiliza un algoritmo de reemplazo LRU global, y a partir de la situación inicial se generan las siguientes direcciones lógicas: (A, 632), (A, 1130), (B, 555), (B, 28), (A, 1333), (B, 446), (A, 501), (A, 1422), (B, 111), (A, 999), (A, 1222), (A, 888) realice el mapa de ocupación de la memoria e indique el número de fallos de página producido.

Nota importante: Las últimas referencias a páginas antes de la situación inicial han sido: B0, A1, A0 (en este orden).

Solución

a) La dirección lógica es de 11 bits de los cuales 2 bits son para el número de página y 9 bits para el desplazamiento dentro de la página, por tanto el tamaño de página es de 512 palabras.

Las direcciones lógicas a las que corresponden las direcciones físicas del enunciado serán:

Dirección Física	Cálculo	Marco	Página	Dirección Lógica
845	512+333	1	A,1	A,845
623	512+111	1	A,1	A,623
1024	1024+0	2	B,0	B,0
1603	1536+67	Error	Error	Error

El tamaño de la memoria física es 1536, por tanto la dirección física 1603 excede el tamaño de la memoria y es por tanto imposible acceder ha dicha posición.

b) Partiendo de la situación inicial reflejada en las tablas de páginas y utilizando un algoritmo de reemplazo LRU global, para las referencias del enunciado se obtiene la siguiente ocupación de memoria:

marco	Inicial	A,632 (pág. 1)	A,1130 (pág. 2)	B,555 (pág. 1)	B,28 (pág 0)	A,1333 (pág. 2)	B,446 (pág. 0)	A,501 (pág. 0)
0	A,0	A,0	A,0	**B,1**,	B,1,	B,1,	B,1,	**A,0**
1	A,1	A,1	A,1	A,1	**B,0**	B,0	B,0	B,0
2	B, 0	B, 0	**A,2**	A,2	A,2	A,2	A,2	A,2

marco	A,1422 (pág. 2)	B,111 (pág. 0)	A,999 (pág. 1)	B,28 (pág 0)	A,1222 (pág. 2)	A,888 (pág. 1)
0	A,0	A,0	**A,1**	A,1	A,1	A,1
1	B,0	B,0	B,0	B,0	B,0	B,0
2	A,2	A,2	A,2	A,2	A,2	A,2

El numero total de fallos de página son 5 y han sido resaltados en negrita.

17. Sea un sistema con doble nivel de paginación donde las direcciones lógicas tienen 15 bits, el tamaño de página es de 1Kbyte y cada tabla de primer nivel puede contener cuatro descriptores de tablas de segundo nivel. Las tablas del primer y segundo nivel las gestiona el sistema operativo y por tanto no esta en el área de memoria destinada a los procesos. Se tiene 4 marcos ubicados en las direcciones más baja de la memoria física asignable a los procesos, dichos marcos se encuentran inicialmente vacíos, los marcos libres se asignan en orden creciente. Suponga que se emite la siguiente secuencia de direcciones lógicas: 7168, 26624, 16508, 5120, 7168, 26624, 12288, 7168, 26624, 16508, 5120, 12288. Esta secuencia se repite indefinidamente.

Diga qué página ocupará cada marco de la memoria física después de la primera repetición de la secuencia suponiendo que el algoritmo de reemplazo de páginas es el OPTIMO y utilizando la siguiente notación: `(i,j)`: identifica la página cuyo descriptor de primer nivel es el `i`, y cuyo descriptor de segundo nivel es el `j`.

Solución

Las direcciones lógica son de 15 bits.

El tamaño de páginas es de 1Kbyte=2^{10} bytes $\Rightarrow$ 10 bits para el desplazamiento dentro de la páginas.

Las tablas de páginas del primer nivel contienen 4 descriptores $\Rightarrow$ 2 bits para utilizarlo de índice en las tablas de primer nivel. Por tanto la dirección lógica la podemos considerar formada por los siguientes campos:

Dirección lógica		
1er nivel paginación	2º nivel paginación	desplazamiento
2 bits	3 bits	10 bits

Por tanto, dada una dirección lógica en decimal el valor de los bits correspondientes al primer nivel de paginación vendrá dado por el cociente de la siguiente división entera:

$$primer\_nivel = \frac{dirección\_lógica}{8\times 1024} = \frac{dirección\_lógica}{8192}$$

Por tanto, dada una dirección lógica en decimal el valor de los bits correspondientes al segundo nivel de paginación vendrá dado por el cociente de la siguiente división entera:

$$segundo\_nivel = \frac{(dirección\_lógica \mod 8192)}{1024}$$

donde mod representa el resto de la división entera.

Dirección lógica	7168	26624	16508	5120	12288
Índice descriptor 1er nivel	0	3	2	0	1
Índice descriptor 2º nivel	7	2	0	5	4
(i,j)	(0,7)	(3,2)	(2,0)	(0,5)	(1,4)

Como inicialmente la memoria se encuentra vacía, las cuatro primeras referencias a páginas distintas serán las que ocupen los marcos. A partir de ese momento se emplea un algoritmo de reemplazo óptimo:

	7168, 26624, 16508, 5120	7168 (0,7)	26624 (3,2)	12288 (1,4)	7168 (0,7)	26624 (3,2)	16508 (2,0)	5120 (0,5)	12288 (1,4)
0	**(0,7)**	(0,7)	(0,7)	(0,7)	(0,7)	(0,7)	(0,7)	(0,7)	(0,7)
1	**(3,2)**	(3,2)	(3,2)	(3,2)	(3,2)	(3,2)	(3,2)	(3,2)	(3,2)
2	**(2,0)**	(2,0)	(2,0)	(2,0)	(2,0)	(2,0)	(2,0)	**(0,5)**	(0,5)
3	**(0,5)**	(0,5)	(0,5)	**(1,4)**	(1,4)	(1,4)	(1,4)	(1,4)	(1,4)

El numero total de fallos de página son 6, de los cuales dos son con reemplazo de página. Los fallos de página se encuentran resaltados en negrita en la tabla anterior.

18. Se tiene un sistema con memoria virtual que utiliza la técnica de la segmentación paginada por demanda. En este sistema se ejecuta un proceso A con tres segmentos T (código), D (datos) y S (pila). Los segmentos de datos y de pila pueden crecer. Las longitudes de estos segmentos se especifican en la tabla 1. La memoria física consta de 4 marcos de 512 bytes cuyo contenido en un instante *t* es el de la tabla 2, donde la nomenclatura T*i* indica la página lógica *i* del segmento T (ej: D0 es la página 0 del segmento D).

Tabla-1	
segmento	longitud
T	2500
D	1500
S	300

Tabla-2	
marco	Segmento,página
0	T2
1	D0
2	T1
3	S0

Si en ese instante *t* se produjese la referencia a memoria que se especifica en los siguientes casos, diga si se produciría fallo de página y, en caso afirmativo, el tipo de fallo (violación de protección,

dirección fuera de rango, página en disco, ...) y el manejo o tratamiento que recibiría por parte del sistema operativo.

c) Acceso a la dirección (D,1000) para escribir en una variable.

d) Acceso a la dirección (T, 3510) para ejecutar una instrucción.

e) Acceso a la dirección (D, 1510) para crear una variable dinámica.

f) Acceso a la dirección (T,950) para ejecutar una instrucción.

Solución

Referencia y Tipo de fallo	Manejo
(D,1000) → Fallo de página.	La página D1 debe cargarse en memoria reemplazando a una de las cuatro cargadas.
(T,3510)→ Dirección fuera de rango.	La dirección no está dentro del tamaño del segmento. El sistema abortará el proceso.
(D,1510) → Fallo de página.	Aunque la dirección está fuera del rango actual del segmento D, el sistema permite que crezca dicho segmento. La página D2 será asignada al proceso (puede que no se necesite cargarla desde disco si todavía no ha sido utilizada y pertenece al área de datos sin valores iniciales), reemplazando a una de las ya cargadas.
(T,950): Sin fallo.	Es una dirección de la página T1, ya presente en memoria. Por tanto, se atenderá la solicitud de dicha dirección

19. Suponga que en un sistema de memoria virtual se desea realizar una implementación de los siguientes algoritmos:

- LRU (Last Recently Used)
- SEGUNDA OPORTUNIDAD
- LFU (Least Frequently Used)

Diga, para cada uno de ellos, qué registros hardware sería estrictamente necesario que la MMU actualizase cada vez que se referencia una página, para poder realizar una implementación de dichos algoritmos. Los posibles registros hardware son: Bit de modificación, Bit de referencia, Bit de validez, Contador del número de veces que una página ha sido referenciada, Tiempo de la última referencia a una página, Registro de desplazamiento en el que el bit de referencia se introduce como bit de mayor peso.

Solución

LRU (*least recently used*): Se elige como victima aquel que hace más tiempo que se ha referenciado, por tanto sería necesario actualizar el registro del "Tiempo de la última referencia a página". El registro de desplazamiento para el bit de referencia no sería siempre utilizable porque para implementar exactamente este algoritmo debería disponer de un número muy grande de bits y el desplazamiento debería realizarse tras cada acceso, no periódicamente como se explicó en clase.

SEGUNDA OPORTUNIDAD: Este algoritmo sólo necesitaría actualizar el del "Bit de referencia".

LFU (*least frequently used*): Se elige como victima aquel que se ha referenciado menos veces por tanto es necesario llevar la contabilidad del número de referencias que se han realizado a una página, el registro que se necesita para ello es del "Contador de referencias".

20. **D**ado un sistema de gestión de memoria basado en múltiples niveles de paginación, se trata de determinar el número de niveles necesarios para que la tabla de primer nivel pueda caber en una TLB de 256 bytes. Se supone que la dirección lógica tiene un formato de 32 bits, el tamaño de página es de 1024 bytes, el tamaño de los descriptores de cualquiera de las tablas de páginas es de 8 bytes y que cada tabla de nivel superior al primero ocupa el tamaño de una página.

Solución

La dirección lógica es de 32 bits y el tamaño de página es de 1024 bytes, por tanto tendremos una estructuración de la dirección lógica como sigue:

22 bits	10 bits
Niveles de paginación	Desplazamiento de página

Hay que calcular cuantos niveles de paginación se necesitan.
El primer nivel de paginación ha de caber en la TLB, el tamaño de la TLB es de 256 bytes y el descriptor de página es de 8 bytes, por tanto tendremos:

$$Número\_descriptores\_TLB = \frac{256}{8} = \frac{2^8}{2^3} = 2^5 = 32 \quad descriptores\ en\ TLB$$

Por tanto de los bits de la dirección lógica dedicados a niveles de paginación los 5 de mayor peso será para el primer nivel:

5 bits	17 bits	10 bits
Primer nivel	Otros niveles de paginación	Desplazamiento de página

Las páginas son de 1024 bytes, por tanto el número de descriptores que puede contener una página vendrá dado por:

$$Número\_descriptores\_página = \frac{1024}{8} = \frac{2^{10}}{2^3} = 2^7 = 128 \quad descriptores\ en\ una\ página$$

Con lo que la composición final de la dirección lógica será:

22 bits				10 bits
5 bits	7 bits	7 bits	3 bits	10 bits
Primer nivel	Segundo nivel	Tercer nivel	Cuarto nivel	Desplazamiento de página

Se necesitan por tanto 4 niveles de paginación distribuidos como se aprecia en la tabla anterior.

21. **S**e tiene un sistema de gestión de memoria segmentado paginado con páginas cuya longitud es 1Kb. Este sistema dispone de una tabla de descriptores de página donde cada descriptor guarda, entre otros, la siguientes datos:

RO	LIST-SEG	etc.	Nº Marco

- RO: bit de protección frente escrituras. (1: sólo lectura, 0:lectura / escritura).
- LIST-SEG: Lista de segmentos que comparten la página.
- Nº Marco: Número de marco.

En este sistema cuando se produce un fallo de página no se aplica el algoritmo de reemplazo ya que un proceso del sistema ("demonio") se encarga de mantener una lista de marcos libres quitándoselos, cuando sea preciso, a los procesos ubicados en memoria. La lista de marcos libres que gestiona el demonio es: 15, 23, 40, 75, 90, 110, 125, 140, 142, 156, 200, 215 ...

Sobre el sistema anterior existe definida una operación de copia de segmentos

```
copiar(Id: Identificador_De_Segmento) return
        Identificador_De_Segmento
```

Esta operación toma como parámetro el identificador del segmento a copiar y devuelve el identificador de un nuevo segmento que es copia del anterior. La copia de segmentos se realiza utilizando la técnica de *copy-on-write* (copiar al escribir) consistente en lo siguiente: Cuando se invoca `s2 := copiar(s1)` el sistema operativo crea un segmento `s2` que *comparte* todas sus páginas con las del segmento `s1`. Sólo en el instante en que se intente realizar una operación de modificación de un segmento, se producirá un fallo de protección y el sistema operativo realizará una copia privada de la página que se desea modificar para el segmento en el que va a producirse la modificación. El sistema operativo debe fijar los bits de protección de una página de tipo "copy-on-write" para que los intentos de escritura provoquen un fallo de protección. La rutina de tratamiento de fallo de protección debe ocuparse de realizar la copia de la página.

En el sistema de gestión de memoria descrito existe un segmento de 3500 bytes de longitud, que se representará como segmento A. Los descriptores de segmento en este sistema son una lista de referencias a una tabla de páginas global. De acuerdo con ésto, el descriptor del segmento A y el estado de la tabla de páginas en el instante que se considera son:

Se Pide: El estado de los descriptores de segmento implicados y de la tabla de descriptores de después de cada una de las siguientes operaciones:

- **1.-** B := Copiar(A)
- **2.-** Escribir en la dirección (A , 365)
- **3.-** Escribir en la dirección (B , 1100)
- **4.-** C := Copiar(B)
- **5.-** Escribir en la dirección (B , 3000)
- **6.-** Escribir en la dirección (C , 3000)

- **7.-** Escribir en la dirección (A , 3000)

Solución

1.- B := Copiar(A)

Seg.A	SegB	Seg. C
0	0	
1	1	
2	2	
3	3	

	RO	seg	marco
0	1	A B	10
1	1	A B	11
2	1	A B	12
3	1	A B	13
4			
5			
6			

	RO	seg	marco
7			
8			
9			
1			
1			
1			
13			

2.- Escribir en la dirección (A , 365)

Seg.A	Seg. B	Seg. C
4	0	
1	1	
2	2	
3	3	

	RO	seg	marco
0	0	B	10
1	1	A B	11
2	1	A B	12
3	1	A B	13
4	0	A	15
5			
6			

	RO	seg	marco
7			
8			
9			
10			
11			
12			
13			

3.- Escribir en la dirección (B , 1100)

Seg. A	Seg. B	Seg. C
4	0	
1	5	
2	2	
3	3	

	RO	Lista-seg	Nº marco
0	0	B	10
1	0	A	11
2	1	A B	12
3	1	A B	13
4	0	A	15
5	0	B	23
6			

	RO	Lista-seg	Nº marco
7			
8			
9			
10			
11			
12			
13			

4.- C := Copiar(B)

Seg. A	Seg. B	Seg. C
4	0	0
1	5	5
2	2	2
3	3	3

	RO	Lista-seg	Nº marco
0	1	B C	10
1	0	A	11
2	1	A B C	12
3	1	A B C	13
4	0	A	15
5	1	B C	23
6			

	RO	Lista-seg	Nº marco
7			
8			
9			
10			
11			
12			
13			

5.- Escribir en la dirección (B , 3000)

Seg. A
4
1
2
3

Seg. B
0
5
6
3

Seg. C
0
5
2
3

	RO	Lista-seg	Nº marco
0	1	B C	10
1	0	A	11
2	1	A C	12
3	1	A B C	13
4	0	A	15
5	1	B C	23
6	0	B	40

	RO	Lista-seg	Nº marco
7			
8			
9			
10			
11			
12			
13			

6.- Escribir en la dirección (C , 3000)

Seg. A
4
1
2
3

Seg. B
0
5
6
3

Seg. C
0
5
7
3

	RO	Lista-seg	Nº marco
0	1	B C	10
1	0	A	11
2	0	A	12
3	1	A B C	13
4	0	A	15
5	1	B C	23
6	0	B	40

	RO	Lista-seg	Nº marco
7	0	C	75
8			
9			
10			
11			
12			
13			

7.- Escribir en la dirección (A , 3000)

Seg. A
4
1
2
3

Seg. B
0
5
6
3

Seg. C
0
5
7
3

	RO	Lista-seg	Nº marco
0	1	B C	10
1	0	A	11
2	0	A	12
3	1	A B C	13
4	0	A	15
5	1	B C	23
6	0	B	40

	RO	Lista-seg	Nº marco
7	0	C	75
8			
9			
10			
11			
12			
13			

22. Sea un sistema con Memoria Virtual que utiliza paginación multinivel con dos niveles de paginación. El tamaño de la dirección física es de 13 bits , el tamaño de marco es de 1024 bytes y el número máximo de entradas de las tablas de páginas de ambos niveles es de 4 descriptores de página en cada nivel.

a) Indique cada uno de los campos de la dirección lógica y física de este sistema, así como el número de bits de dichos campos.

b) En la actualidad en dicho sistema se están ejecutando dos procesos P1 y P2, y el contenido de la memoria principal es el que se muestra a continuación:

Memoria Principal

Núm. marco	Contenido (proceso,página)
0	(P1, 4)
1	(P2, 8)
2	(P1, 7)
3	(P2, 10)
4	(P2, 11)
5	(P1, 5)
6	(P1, 6)
7	Tabla pág.

B1)Teniendo en cuenta el contenido de la memoria y el de la tabla de páginas del primer nivel del proceso P1, indique el contenido de las tablas de páginas del 2º nivel para el proceso P1. Indique también el descriptor de página de primer nivel que sería necesario utilizar con el fin de acceder a las páginas que están ubicadas en memoria. Cada descriptor de páginas ocupa dos bytes

Proceso P1
Tabla de páginas
1er nivel

8000
8008
8016
8024

B2)Teniendo en cuenta el contenido de la memoria y el de la tabla de páginas del primer nivel del proceso P2, indique el contenido de las tablas de páginas del 2º nivel para el proceso P2. Indique también el descriptor de página de primer nivel que sería necesario utilizar con el fin de acceder a las páginas que están ubicadas en memoria. Cada descriptor de páginas ocupa dos bytes.

Proceso P2
Tabla de páginas
1er nivel

8032
8040
8048
8056

c) Ambos procesos utilizan los mismos datos de entrada, sobre los que únicamente acceden para lectura. Dichos datos corresponden a los contenidos de las páginas 7 para ambos procesos P1 y P2. Indique como se habrían de modificar los contenidos de las tablas de páginas del apartado anterior para que sea posible la compartición de los datos de entrada.

d) En dicho sistema, utilizando la técnica de paginación por demanda con un algoritmo de reemplazo FIFO se produce hiperpaginación para una secuencia de solicitudes dada de un conjunto de procesos. Para eliminar la hiperpaginación se decidió incrementar la memoria principal de dicho sistema y por tanto el número de marcos, finalmente se comprobó que con

dicho incremento no se solucionaba el problema de hiperpaginación. Razone si es posible o no que ocurriera lo anteriormente descrito y porque.

Solución

a) Dirección Lógica

Bit 13 Bit 0

1er nivel	2º nivel	Desplaz. Página
2 bits	2 bits	10 bits

Dirección Física

Bit 12 Bit 0

Marco	Desplazamiento
3 bits	10 bits

b1)

Valor del descriptor de página
8008

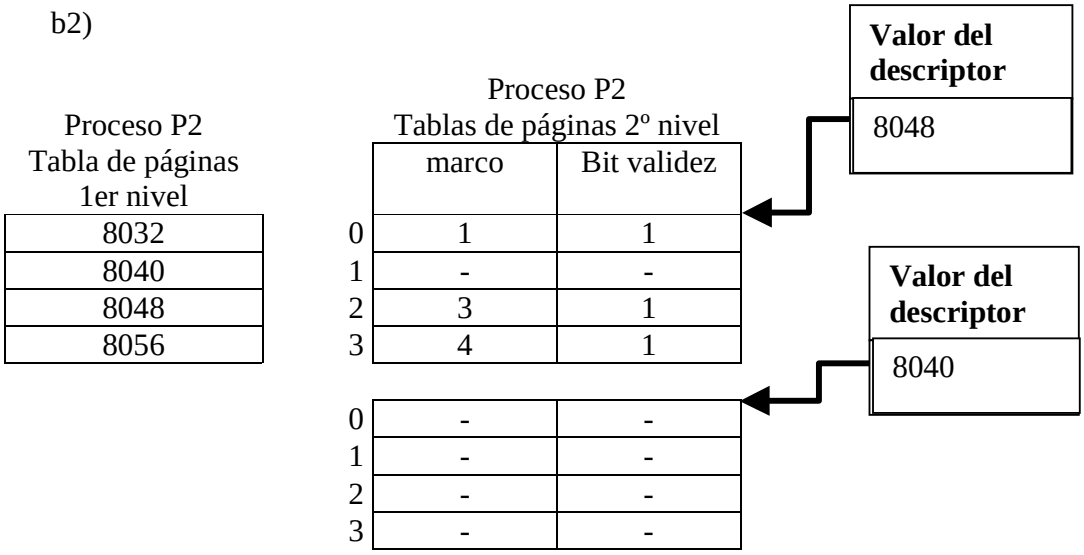
Proceso P1
Tabla de páginas 1er nivel

8000
8008
8016
8024

Proceso P1
Tablas de páginas 2º nivel

	marco	Bit validez
0	0	1
1	5	1
2	6	1
3	2	1

b2)



Proceso P2
Tabla de páginas 1er nivel

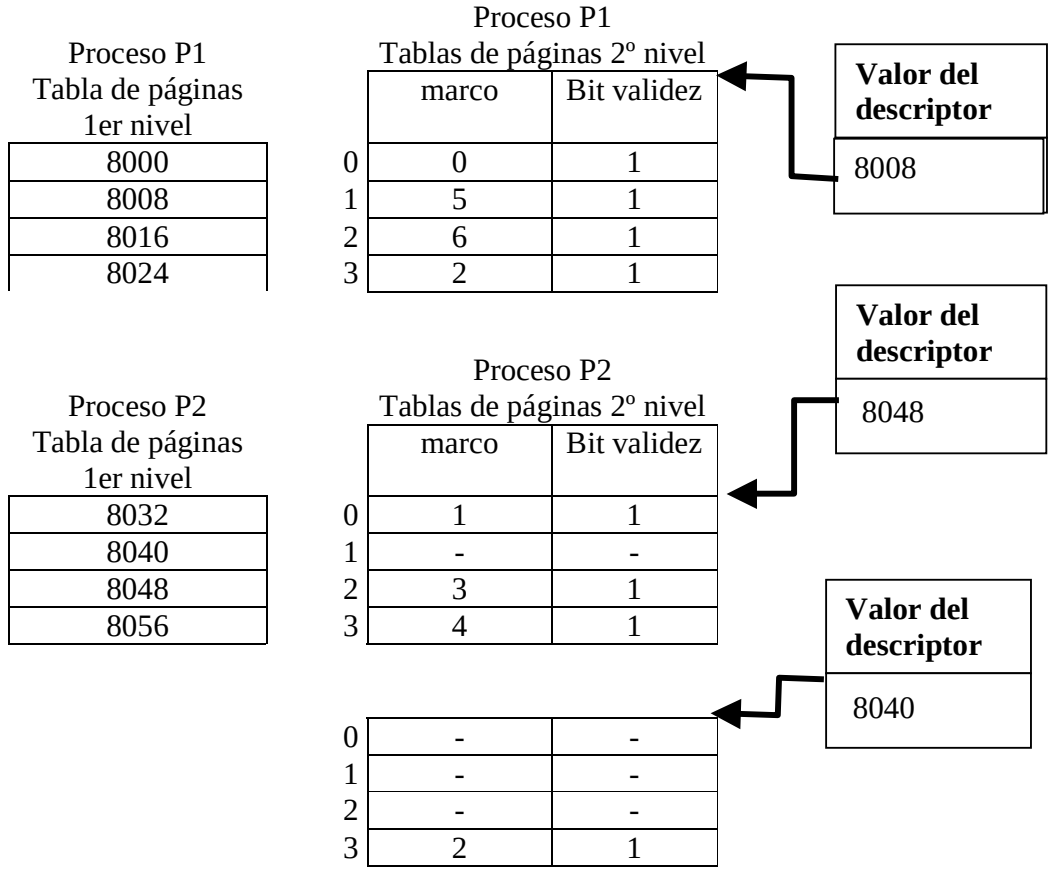
8032
8040
8048
8056

Proceso P2
Tablas de páginas 2º nivel

	marco	Bit validez
0	1	1
1	-	-
2	3	1
3	4	1

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

c)



d) Porque presenta la anomalía de Belady o no se ha incrementado suficientemente la memoria.

23. A continuación se describe un modelo de memoria basado en la técnica de "segmentación". En dicho modelo (ver figura), la imagen interna de un proceso está compuesta por cuatro segmentos (código, datos, espacio intermedio y pila). Este modelo, similar al que utiliza el sistema operativo Minix, se caracteriza porque los cuatro segmentos residen siempre de forma contigua en memoria, utilizándose el segmento de espacio intermedio para prever las ampliaciones de los segmentos de datos y/o pila. Por ejemplo si un proceso tiene 4K de Código, 2K de Datos y 1 de Pila y en la cabecera del ejecutable ("a.out") se ha marcado un tamaño total del proceso de 40 K, el Espacio intermedio dispondrá de 33K.

Pila	1K
Espacio Intermedio	33K
Datos	2K
Código	4K

Se definen los siguientes comandos:

- CHMEM: que permite reajustar el espacio dedicado a pila y espacio intermedio de un fichero ejecutable. Un ejemplo de llamada: "chmem 10240 a.out" cambiaría el

tamaño total del proceso de 40K a 16K (10240 bytes + tamaño código y datos), de los que el bloque de 1K más alto pertenecería a la pila.

- BRK: que permite ampliar el segmento de datos según el tamaño que se especifique en la llamada. Esta ampliación la realiza a costa de reducir el segmento de espacio intermedio. Un ejemplo: "brk(2048)" aumentaría en 2K el tamaño del segmento de datos y reduciría en 2K el segmento de espacio intermedio. En caso de no existir el suficiente espacio intermedio la llamada se abortaría y devolvería un mensaje de error.

Otros comandos relacionados con el manejo de procesos son: FORK (que realiza un duplicado del proceso que lo invoca p.e. a.out y lo pone en ejecución, a'.out), EXEC (que ubica el ejecutable invocado en memoria y lo pone en ejecución) y EXIT (que finaliza la ejecución del proceso y libera su memoria).

Se supone que la memoria está disponible a partir de la dirección 204800 y que la asignación de memoria sigue una política del "primer hueco disponible" y se realiza en orden de direcciones ascendente. Se consideran los siguientes ejecutables disponibles:

- "a.out"con 2K de código, 1K de datos y 1K de pila. El espacio total del proceso es de 9K.
- "b.out"con 2K de código, 2K de datos y 1K de pila. El espacio total del proceso es de 12K.
- "c.out"con 1K de código, 3K de datos y 1K de pila. El espacio total del proceso es de 15K.

Se pide completar para la siguiente la tabla

Proceso	Segmento	Tamaño	Dir. base

Para cada operación de la siguiente secuencia:

a) Se realiza la operación "exec b.out".

b) 2) Se realiza la operación "exec c.out".

c) 3) El proceso b.out realiza una operación "brk(512)".

d) 4) Se realiza una operación "chmem 8192 c.out".

e) 5) El proceso b.out realiza como última instrucción de su código una operación "exit".

f) 6) Se realiza la operación "exec c.out".

Solución

a) Se realiza la operación "exec b.out".

Proceso	Segmento	Tamaño	Dir. base
b.out	código	2 Kb	204800
b.out	datos	2 Kb	206848
b.out	intermedio	7 Kb	208896
b.out	pila	1 Kb	216064

b) Se realiza la operación “exec c.out”.

Proceso	Segmento	Tamaño	Dir. base
b.out	código	2 Kb	204800
b.out	datos	2 Kb	206848
b.out	intermedio	7 Kb	208896
b.out	pila	1 Kb	216064
c.out	código	1 Kb	217088
c.out	datos	3 Kb	218112
c.out	intermedio	10 Kb	221184
c.out	pila	1Kb	231424

c) El proceso b.out realiza una operación “brk(512)”.

Proceso	Segmento	Tamaño	Dir. Base
b.out	código	2 Kb	204800
b.out	datos	2,5 Kb	206848
b.out	intermedio	6,5 Kb	209408
b.out	pila	1 Kb	216064
c.out	código	1 Kb	217088
c.out	datos	3 Kb	218112
c.out	intermedio	10 Kb	221184
c.out	pila	1Kb	231424

d) Se realiza una operación “chmem 8192 c.out”.

Proceso	Segmento	Tamaño	Dir. Base
b.out	código	2 Kb	204800
b.out	datos	2,5 Kb	206848
b.out	intermedio	6,5 Kb	209408
b.out	pila	1 Kb	216064
c.out	código	1 Kb	217088
c.out	datos	3 Kb	218112
c.out	intermedio	10 Kb	221184
c.out	pila	1Kb	231424

e) El proceso b.out realiza como última instrucción de su código una operación “exit”.

Proceso	Segmento	Tamaño	Dir. base
c.out	código	1 Kb	217088
c.out	datos	3 Kb	218112
c.out	intermedio	10 Kb	221184
c.out	pila	1Kb	231424

f) Se realiza la operación “exec c.out”.

Proceso	Segmento	Tamaño	Dir. base
c.out-2	código	1 Kb	204800
c.out-2	datos	3 Kb	205824
c.out-2	intermedio	7 Kb	208896
c.out-2	pila	1Kb	216064
c.out	código	1 Kb	217088
c.out	datos	3 Kb	218112
c.out	intermedio	10 Kb	221184
c.out	pila	1Kb	231424

24. En la versión 1.0 del sistema operativo OS/2 se utilizó la técnica de la **segmentación por demanda (sin paginación)**, dado que el procesador destino de este operativo fue el Intel 80286, el cual fue diseñado para soportar esta técnica de gestión de memoria. De esta forma, un proceso puede ejecutarse sin necesidad de que todos sus segmentos estén presentes en memoria, cargándose estos bajo demanda cuando sea necesario.

En la versión 1.0, para la gestión de la segmentación por demanda, se utilizan los siguientes criterios:

- Los procesos en OS/2 disponen de tres segmentos (código, datos y pila).
- El segmento de código de un proceso debe estar **siempre** presente en memoria central.
- Búsqueda de hueco: utiliza la técnica del **mejor ajuste.**
- Compactación: cuando se aplica, se genera un **único hueco** en memoria a partir de la dirección física **0**, desplazando **todos** los segmentos hacia direcciones altas.
- Selección de segmento víctima: se elige el segmento al cual hace más tiempo que no se accede **(LRU GLOBAL**).
- Algoritmo de fallo de página es el que se expone a continuación:

```
hueco_encontrado:= falso
    mientras no hueco_encontrado
       si existe un hueco en memoria suficientemente grande
       entonces
                   hueco_encontrado := cierto
                   H := MEJOR_AJUSTE
       si_no
             si el sumatorio de los huecos en memoria
                                es lo suficientemente grande
             entonces
                  compactar
                 hueco_encontrado:= cierto
                  H := HUECO_EN_INICIO_MEMORIA
             si_no
                  desalojar de memoria un segmento completo según LRU GLOBAL
             fin_si
       fin_si
    fin_mientras
    retorna H
```

En un determinado instante, la situación del sistema es la siguiente

Existen cuatro procesos en el sistema, siendo los tamaños de sus segmentos:

proceso	código	datos	pila
A	57500	14000	13000
B	3100	2400	6900
C	3300	3300	4300
D	6400	9800	19000

La memoria central instalada es de 128.000 bytes y ocupación actual es:

Posición inicial	proceso	segmento
0	A	código
57500	B	código
60600	C	código
63900	D	código
70300	A	datos
84300	C	datos
87600	A	pila
100600	B	pila

Asuma que las últimas direcciones generadas han sido (B,pila,398) (A,pila,1274) (C,datos,2091) (A,datos,97).

Se pide :

a) **C**ompleten las tablas de segmentos para que reflejen la situación descrita anteriormente indicando el bit de validez..

b) Tras producirse la siguiente secuencia de direcciones lógicas, describa el estado de la memoria central : (A, código, 1020) (B, pila, 300) (A, código , 37200) (C, código, 200) (D, pila, 500) (A, pila, 32)

c) **S**uponga que ahora se crea al proceso E, de tal forma que sea una réplica exacta del proceso A. La creación obliga al sistema a ubicar en memoria únicamente el segmento de código de este proceso. Describa cómo queda el estado de la memoria central .

Solución

a) Consultando el contenido de la memoria principal y las características de cada proceso se rellenan las tablas de segmento para cada proceso.

	proceso A		Proceso B		Proceso C		Proceso D	
	Dir. Base	Bit validez	Dir .base	Bit validez	Dir. base	Bit validez	Dir. base	Bit validez
código	0	V	57.500	V	60.600	V	63.900	V
datos	70.300	V	--	I	84.300	V	--	I
pila	87.600	V	100.600	V	--	I	--	I

c) Todas las referencias son a segmentos que ya se encuentran en memoria excepto la referencia al segmento de pila del proceso D, que es necesario ubicarlo en memoria. Como hay hueco para ubicar el segmento de pila del proceso D, el contenido de la memoria queda de la siguiente manera:

posición inicial	proceso	segmento
0	A	código
57500	B	código
60600	C	código
63900	D	código
70300	A	datos
84300	C	datos
87600	A	pila
100600	B	pila

posición inicial	proceso	segmento
107.500	D	pila

c) El segmento de código del nuevo proceso no cabe en memoria, por lo tanto hay que buscarle un hueco y aplicar el algoritmo de fallo de página. Finalmente el contenido de la memoria que como sigue:

posición inicial	proceso	segmento
0	A	código
57500	B	código
60600	C	código
63900	D	código
70300	E	código

posición inicial	proceso	segmento

25. **J**ustifique por qué se puede emplear un algoritmo LRU para gestionar la caché de bloques del sistema operativo, pero no hay apenas ningún sistema que emplee ese mismo algoritmo para reemplazar páginas en la gestión de la memoria virtual (se utilizan aproximaciones en este segundo caso, pero no el algoritmo tal cual es).

Solución

En la caché de bloques es muy fácil anotarse cuándo ha sido accedido un bloque (que es la información que necesita el algoritmo LRU), pues para ello hay que efectuar una llamada al sistema y poco se notará si el sistema operativo se anota esta información en alguna estructura de datos (por ejemplo, en una cola ordenada de bloques que se va modificando a medida que se realizan los accesos, ubicando el bloque accedido al final de la cola y tomando como víctima aquel bloque que esté en la cabeza).

En el caso del reemplazo de páginas resulta extremadamente caro anotar por cada acceso a memoria la ocurrencia de tal acceso. Ninguna MMU se permite el lujo de guardar tal información en un contador, cola o alguna estructura similar. Como máximo proporcionan un bit de referencia con el que se puede saber si cada página ha sido accedida o no, pero se desconoce cuántos accesos ha sufrido y en qué momento. Con este bit de referencia se pueden implementar aproximaciones al LRU, pero no el algoritmo exacto.

26. En un sistema de gestión de memoria virtual se decide utilizar paginación y un algoritmo de reemplazo LOCAL basado en una aproximación al LRU, denominada SEGUNDA OPORTUNIDAD. En el sistema se ejecutan actualmente 3 procesos A, B y C. Cada acceso se

codifica de forma que el primer símbolo representa el proceso que lo realiza y el segundo la página accedida (se supone que todas las páginas accedidas están dentro del espacio lógico del proceso).

La secuencia de accesos a analizar es la siguiente: A1, A1, B0, C3, A0, A1, B0, B1, A3, B3, C0, C1, C1, A1, C0, A0, B0, A2, A2, A1, C0, C1, B1, C2, A3, C3, A4, C3, A2, B0, B0, C2, C3, A0, C2, C0, A4, C0, A1, A0.

Se pide:

a) Determinar el contenido de la tabla de páginas asignada a cada proceso, una vez que la memoria se ha llenado por primera vez. La memoria consta de 8 marcos que se asignan en orden de direcciones crecientes (desde el marco 0 al 7) e inicialmente están vacíos. Cada descriptor de página consta de dos campos: primero el **número de marco** asignado a la página y segundo el **bit de referencia** asociado. Si la página no está en memoria, ello se indica con un guión (-) en el número de marco correspondiente a dicha página.

b) Determinar el contenido de los descriptores de página, cada vez que se produzca un fallo de página (indicar qué página falla y a qué proceso pertenece). El contenido de dichos descriptores debe reflejar el estado posterior al fallo y su actualización se limitará a los descriptores de página del proceso que ha producido el fallo de página.

c) A qué algoritmo de reemplazo equivale el de SEGUNDA OPORTUNIDAD, si todos los bits de referencia de la tabla de páginas están puestos a 1 en el momento previo al fallo de página.

Solución

a) Dada la secuencia de referencias propuesta en el enunciado la memoria se llenara con las 8 primeras referencia a páginas distintas. Esto es: A1, A1, B0, C3, A0, A1, B0, B1, A3, B3, C0,.... En las tablas de páginas todas estas páginas aparecerán con el bit de referencia a 1.

Memoria

0	A1
1	B0
2	C3
3	A0
4	B1
5	A3
6	B3
7	C0

Tabla Pag. A

Pg.	marco	Bit ref.
0	**3**	**1**
1	**0**	**1**
2	-	-
3	**5**	**1**
4	-	-
5	-	-
6	-	-

Tabla Pag. B

marco	Bit ref.
1	**1**
4	**1**
-	-
6	**1**
-	-
-	-
-	-

Tabla Pag. C

marco	Bit ref.
7	**1**
-	
-	
2	**1**
-	
-	
-	

b) El primer fallo de página ocurre al referenciar por primera vez C1, en la siguiente tabla se ha sombreado en gris las referencias ya realizadas.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

El algoritmo de fallo de reemplazo es el de SEGUNDA OPORTUNIDAD LOCAL, por lo tanto sólo afectará a los marcos de memoria que contienen páginas del proceso C. El orden FIFO en que la páginas del proceso C se ubicaron en memoria es C3, C0. Por

tanto se comienza dando una segunda oportunidad a la página 3 y a continuación a la 0. Se pone en primer lugar el bit de referencia de la página 3 a 0 y a continuación el de la 0, eligiéndose como victima la página 3.

Memoria	
0	A1
1	B0
2	C1
3	A0
4	B1
5	A3
6	B3
7	C0

	Tabla Pag. A	
Pg.	marco	Bit ref.
0	**3**	**1**
1	**0**	**1**
2	-	-
3	**5**	**1**
4	-	-
5	-	-
6	-	-

Tabla Pag. B	
marco	Bit ref.
1	**1**
4	**1**
-	-
6	**1**
-	-
-	-
-	-

Tabla Pag. C	
marco	Bit ref.
7	**0**
2	**1**
-	
-	**0**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso C queda C0,C1.

Orden FIFO	A	A1	A0	A3
	B	B0	B1	B3
	C	C0	C1	-

Se continua con las referencias mientras no haya fallo de página actualizando los bits de referencia correspondientes.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

Memoria	
0	A1
1	B0
2	C1
3	A0
4	B1
5	A3
6	B3
7	C0

	Tabla Pag. A	
Pg.	marco	Bit ref.
0	**3**	**1**
1	**0**	**1**
2	-	-
3	**5**	**1**
4	-	-
5	-	-
6	-	-

Tabla Pag. B	
marco	Bit ref.
1	**1**
4	**1**
-	-
6	**1**
-	-
-	-
-	-

Tabla Pag. C	
marco	Bit ref.
7	**1**
2	**1**
-	
-	**0**
-	
-	
-	

Fallo de página al referenciar A2. El orden FIFO en que la páginas del proceso A se ubicaron en memoria es A1, A0, A3. Se comienza dando una segunda oportunidad a la página A1. Se pone por tanto en primer lugar el bit de referencia de la página A1 a 0, continuando con el de A0, y A3 y eligiéndose como victima la página 1.

Memoria	
0	A2
1	B0
2	C1
3	A0
4	B1
5	A3
6	B3
7	C0

	Tabla Pag. A	
Pg.	marco	Bit ref.
0	**3**	**0**
1	**-**	**0**
2	**0**	**1**
3	**5**	**0**
4	**-**	**-**
5	-	-
6	-	-

Tabla Pag. B	
marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	**-**
-	-
-	-

Tabla Pag. C	
marco	Bit ref.
7	**1**
2	**1**
-	
-	**0**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso A queda A0,A3,A2.

Orden FIFO	A	A0	A3	A2
	B	B0	B1	B3
	C	C0	C1	-

Se continua con las referencias mientras no haya fallo de página actualizando los bits de referencia correspondientes.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

Fallo de página al referenciar A1. El orden FIFO en que la páginas del proceso A se ubicaron en memoria es A0, A3, A2. Se comienza mirando el bit de referencia de la página A0, este bit está a 0 y por tanto la página A0 se elige como victima.

Memoria	
0	A2
1	B0
2	C1
3	A1
4	B1
5	A3
6	B3
7	C0

	Tabla Pag. A	
Pg.	marco	Bit ref.
0	**-**	**0**
1	**3**	**1**
2	**0**	**1**
3	**5**	**0**
4	-	-
5	**-**	**-**
6	**-**	**-**

Tabla Pag. B	
marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	-
-	-
-	-

Tabla Pag. C	
marco	Bit ref.
7	**1**
2	**1**
-	
-	**0**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso A queda A3,A2, A1.

Orden FIFO	A	A3	A2	A1
	B	B0	B1	B3
	C	C0	C1	-

Se continua con las referencias mientras no haya fallo de página actualizando los bits de referencia correspondientes.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

Fallo de página al referenciar C2. El orden FIFO en que la páginas del proceso C se ubicaron en memoria es C0, C1. Se comienza mirando el bit de referencia de la página C0, este bit está a 1, y por tanto se le da una segunda oportunidad y se pone su bit de referencia a 0, igualmente se hace para la página C1, se elige como victima, la página C0.

Memoria

0	A2
1	B0
2	C1
3	A1
4	B1
5	A3
6	B3
7	C2

Tabla Pag. A

Pg.	marco	Bit ref.
0	**-**	**0**
1	**3**	**1**
2	**0**	**1**
3	**5**	**0**
4	**-**	**-**
5	**-**	**-**
6	**-**	**-**

Tabla Pag. B

marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	**-**
-	**-**
-	**-**

Tabla Pag. C

marco	Bit ref.
-	**0**
2	**0**
7	**1**
-	**0**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso C queda C1,C2.

Orden FIFO	A	A3	A2	A1
	B	B0	B1	B3
	C	C1	C2	-

Se continua con las referencias mientras no haya fallo de página actualizando los bits de referencia correspondientes.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

Memoria

0	A2
1	B0
2	C1
3	A1
4	B1
5	A3
6	B3
7	C2

Tabla Pag. A

Pg.	marco	Bit ref.
0	**-**	**0**
1	**3**	**1**
2	**0**	**1**
3	**5**	**1**
4	**-**	**-**
5	**-**	**-**
6	**-**	**-**

Tabla Pag. B

marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	**-**
-	**-**
-	**-**

Tabla Pag. C

marco	Bit ref.
-	**0**
2	**0**
7	**1**
-	**0**
-	
-	
-	

Fallo de página al referenciar C3. El orden FIFO en que la páginas del proceso C se ubicaron en memoria es C1, C2. Se comienza mirando el bit de referencia de la página C1, este bit está a 0. Se elige como victima, la página C1.

Memoria	
0	A2
1	B0
2	C3
3	A1
4	B1
5	A3
6	B3
7	C2

	Tabla Pag. A	
Pg.	marco	Bit ref.
0	**-**	**0**
1	**3**	**1**
2	**0**	**1**
3	**5**	**1**
4	**-**	**-**
5	-	-
6	-	-

Tabla Pag. B	
marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	**-**
-	-
-	-

Tabla Pag. C	
marco	Bit ref.
-	**0**
-	**0**
7	**1**
2	**1**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso C queda C2,C3.

Orden FIFO				
	A	A3	A2	A1
	B	B0	B1	B3
	C	C2	C3	-

Se continua con las referencias mientras no haya fallo de página actualizando los bits de referencia correspondientes.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

Fallo de página al referenciar A4. El orden FIFO en que la páginas del proceso A se ubicaron en memoria es A3,A2, A1. Se comienza mirando el bit de referencia de la página A3, este bit está a 1 y por tanto se le da una segunda oportunidad y se pone su bit de referencia a 0, igualmente se hace para la página A2, y A1. Se elige como victima, la página A3.

Memoria	
0	A2
1	B0
2	C3
3	A1
4	B1
5	A4
6	B3
7	C2

	Tabla Pag. A	
Pg.	marco	Bit ref.
0	**-**	**0**
1	**3**	**0**
2	**0**	**0**
3	**-**	**0**
4	**5**	**1**
5	-	-
6	-	-

Tabla Pag. B	
marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	**-**
-	-
-	-

Tabla Pag. C	
marco	Bit ref.
-	**0**
-	**0**
7	**0**
2	**1**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso A queda A2,A1,A4.

Orden FIFO				
	A	A2	A1	A4
	B	B0	B1	B3
	C	C2	C3	-

Se continua con las referencias mientras no haya fallo de página actualizando los bits de referencia correspondientes.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

Memoria

0	A2
1	B0
2	C3
3	A1
4	B1
5	A4
6	B3
7	C2

Tabla Pag. A

Pg.	marco	Bit ref.
0	**-**	**0**
1	**3**	**0**
2	**0**	**1**
3	**-**	**0**
4	**5**	**1**
5	**-**	**-**
6	**-**	**-**

Tabla Pag. B

marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	**-**
-	**-**
-	**-**

Tabla Pag. C

marco	Bit ref.
-	**0**
-	**0**
7	**1**
2	**1**
-	
-	
-	

Fallo de página al referenciar A0. El orden FIFO en que la páginas del proceso A se ubicaron en memoria es A2, A1, A4. Se comienza mirando el bit de referencia de la página A2, este bit está a 1 y por tanto se le da una segunda oportunidad y se pone su bit de referencia a 0, se mira el bit de referencia de la página A1, este bit está a 0. Se elige como victima, la página A1.

Memoria

0	A2
1	B0
2	C3
3	A0
4	B1
5	A4
6	B3
7	C2

Tabla Pag. A

Pg.	marco	Bit ref.
0	**3**	**1**
1	**-**	**0**
2	**0**	**0**
3	**-**	**0**
4	**5**	**1**
5	**-**	**-**
6	**-**	**-**

Tabla Pag. B

marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	**-**
-	**-**
-	**-**

Tabla Pag. C

marco	Bit ref.
-	**0**
-	**0**
7	**1**
2	**1**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso A queda A1,A4,A0.

Orden FIFO	A	A2	A4	A0
	B	B0	B1	B3
	C	C2	C3	-

Se continua con las referencias mientras no haya fallo de página actualizando los bits de referencia correspondientes.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

Fallo de página al referenciar C0. El orden FIFO en que la páginas del proceso C se ubicaron en memoria es C2,C3. Se comienza mirando el bit de referencia de la página C2, este bit está a 1 y por tanto se le da una segunda oportunidad y se pone su bit de referencia a 0, igualmente con el bit de referencia de la página C3. Se elige como victima, la página C2.

Memoria

0	A2
1	B0
2	C3
3	A0
4	B1
5	A4
6	B3
7	C0

Tabla Pag. A

Pg.	marco	Bit ref.
0	**3**	**1**
1	**-**	**0**
2	**0**	**0**
3	**-**	**0**
4	**5**	**1**
5	**-**	**-**
6	**-**	**-**

Tabla Pag. B

marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	-
-	-
-	-

Tabla Pag. C

marco	Bit ref.
7	**1**
-	**0**
-	**0**
2	**0**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso C queda C3,C0.

Orden FIFO	A	A2	A4	A0
	B	B0	B1	B3
	C	C3	C0	-

Se continua con las referencias mientras no haya fallo de página actualizando los bits de referencia correspondientes.

A1	A1	B0	C3	A0	A1	B0	B1	A3	B3	C0	C1	C1	A1
C0	A0	B0	A2	A2	A1	C0	C1	B1	C2	A3	C3	A4	C3
A2	B0	B0	C2	C3	A0	C2	C0	A4	C0	A1	A0.		

Fallo de página al referenciar A1. El orden FIFO en que la páginas del proceso A se ubicaron en memoria es A2, A4, A0. Se comienza mirando el bit de referencia de la página A2, este bit está a 0. Se elige como victima, la página A2.

Memoria

0	A1
1	B0
2	C3
3	A0
4	B1
5	A4
6	B3
7	C0

Tabla Pag. A

Pg.	marco	Bit ref.
0	**3**	**1**
1	**0**	**1**
2	**-**	**0**
3	**-**	**0**
4	**5**	**1**
5	**-**	**-**
6	**-**	**-**

Tabla Pag. B

marco	Bit ref.
1	**1**
4	**1**
-	**-**
6	**1**
-	-
-	-
-	-

Tabla Pag. C

marco	Bit ref.
7	**1**
-	**0**
-	**0**
2	**0**
-	
-	
-	

Con lo que el orden FIFO de las páginas para el proceso A queda A4, A0, A1.

Orden FIFO	A	A4	A0	A1
	B	B0	B1	B3
	C	C3	C0	-

Se continua con la ultima referencia a A0 y como no haya fallo de página y su bit de referencia se encuentra a 1, las ultimas tablas representan el estado de la memoria.

c) Al algoritmo FIFO.

27. Explique brevemente cual es el principal inconveniente que puede presentar la segmentación por demanda.

Solución

El algoritmo de fallo de segmento es más complicado que el de fallo de página: Puede que no baste con seleccionar un solo segmento víctima (el espacio requerido por el segmento que provocó el fallo puede ser mayor que el liberado por el segmento víctima). También puede necesitarse compactación tras haber extraído de la memoria múltiples segmentos víctimas.

Al igual que ocurre con la paginación por demanda, otro inconveniente es el "thrashing" (hipersegmentación): Si no hay suficiente memoria física para el conjunto de procesos actualmente en funcionamiento, puede que se pierda más tiempo gestionando fallos de segmento que ejecutando procesos.

4.- Hiperpaginación y Área activa.

-Tanto en los enunciados como en las soluciones de los problemas de este apartado se emplean un conjunto de siglas que pasamos a definir:

- TAA = Tamaño de área activa.

28. Proporcione una definición precisa de la hiperpaginación o thrashing.

Solución

Si consideramos que un programa está compuesto por diferentes localidades que se pueden solapar, la hiperpaginación se puede definir formalmente de la siguiente manera:

$$hiperpaginación = (\sum tamaños\_de\_localidad) > (tamaño\_de\_memoria\_total)$$

donde se considera:

- Localidad: como el conjunto de páginas que un proceso utiliza conjuntamente.
- Principio de localidad de referencia: un proceso va pasando de una localidad a otra de una forma lenta y progresiva.

29. En un sistema de gestión de memoria virtual se decide utilizar un modelo de área activa para controlar la demanda de memoria. En el sistema se ejecutan actualmente 3 procesos A, B y C. Cada acceso se codifica de forma que el primer símbolo representa el proceso que lo realiza y el segundo la página accedida (se supone que todas las páginas accedidas están dentro del espacio lógico del proceso).

Suponiendo que el tamaño de la ventana de área activa es 4, calcule el mínimo y máximo tamaño de área activa de cada proceso después de la siguiente secuencia de accesos y suponiendo que el muestreo se inicia inmediatamente antes del primer acceso a cada proceso: A1, A1, B0, C3, A0, A1, B0, B1, A3, B3, C0, C1, C1, A1, C0, A0, B0, A2, A2, A1, C0, C1, B1, C2, A3, C3, A4, C3, A2, B0, B0, C2, C3, A0, C2, C0, A4, C0, A1, A0, A0, C0.

Solución

Área activa es conjunto de páginas accedidas en las últimas D (en nuestro caso D=4) referencias. El tamaño del área activa del proceso p_i es el número de páginas referenciadas en la ventana más reciente.

De las referencias a páginas propuestas en el enunciado, si se extraen ordenadamente para cada proceso y se analizan en intervalos de 4 referencias se obtiene lo siguiente:

Referencias al proceso A
A1, A1, A0, A1, A3, A1, A0, A2, A2, A1, A3, A4, A2, A0, A4, A1, A0, A0.
Se observa que para el proceso A en esta secuencia de referencias el número mínimo de páginas distintas que se referencia tomando una ventana de 4 referencias es 2 páginas y el máximo es de 4 páginas distintas. Por tanto tendremos:

$$máx\,TAA_A = 4; \quad \min TAA_A = 2$$

Referencias al proceso B
B0, B0, B1,B3, B0, B1,B0, B0.
Se observa que para el proceso B en esta secuencia de referencias el número mínimo de páginas distintas que se referencia tomando una ventana de 4 referencias es 2 páginas y el máximo es de 3 páginas distintas. Por tanto tendremos:

$$máx\,TAA_B = 3; \quad \min TAA_B = 2$$

Referencias al proceso C
C3, C0, C1, C1, C0, C0, C1, C2, C3, C3, C2, C3, C2, C0, C0, C0.
Se observa que para el proceso C en esta secuencia de referencias el número mínimo de páginas distintas que se referencia tomando una ventana de 4 referencias es 2 páginas y el máximo es de 3 páginas distintas. Por tanto tendremos:

$$máx\,TAA_C = 4; \quad \min TAA_C = 2$$

30. Justifique la relación entre el principio de localidad y el concepto de área activa.

Solución

El área activa de un proceso trata de reflejar la localidad de dicho proceso definida como el conjunto de páginas que se están usando activamente juntas.

31. Indicar brevemente en qué consiste el área activa o working-set.

Solución

Es un modelo usado para determinar la localidad de los procesos. Utiliza un parámetro ventana del área activa que representa el número de referencias a considerar hasta el momento actual, tal conjunto de páginas constituye el área activa.
El S.O., conociendo el área activa de cada proceso, ha de asignar marcos suficientes para que cada uno pueda mantenerla en memoria. Si la suma de los tamaños de las áreas activas excede del total de marcos disponibles, el S.O. seleccionará un proceso, guardará sus páginas en el dispositivo de paginación y reasignará sus marcos a los demás procesos.

32. Diga si las siguientes afirmaciones sobre el modelo del área activa son verdaderas (V) o falsas (F) y justifique su respuesta:

a) Si la suma de los tamaños de las ventanas del área activa de los distintos procesos de un sistema es mayor que el número de marcos de memoria física, entonces en el sistema se produce hiperpaginación.

b) Si el espacio de direcciones lógicas de un proceso es de 1024 páginas y el tamaño del área

activa en un instante dado es de 512 páginas, entonces puede asegurarse que el proceso provocará hiperpaginación en todos los casos.

c) Una MMU que pudiese provocar una interrupción cada cierto número fijo de referencias a memoria y gestionase el bit de referencia, podría calcular de forma exacta el tamaño del área activa.

d) En un sistema cuyo tamaño de ventana de área activa es 1024, el tamaño del área activa en un instante *t* vale 1024 si, y solo si, las últimas 1024 páginas de una serie de referencias son distintas.

e) Si se aumenta el número de marcos que tiene asignados un proceso, entonces aumentará el tamaño del área activa.

Solución

a) FALSA. El tamaño del área activa del proceso p_i es el número de páginas referenciadas en la ventana más reciente, por tanto la suma de los tamaños de ventana de área activa no tiene sentido.

d) FALSA. El que provoque hiperpaginación dependerá de cuantas páginas necesite el proceso tener en memoria a la vez para poder ser ejecutado.

e) FALSA: Lo podría calcular únicamente de forma aproximada.

d) VERDADERO. Ya que el tamaño del área activa del proceso p_i es el número de páginas referenciadas en la ventana más reciente. Sólo sería falso si alguna de esas referencias fuese incorrecta.

f) FALSO. EL tamaño de área activa depende de la referencias que haya solicitado ese proceso y del tamaño de la ventana que se escoja para calcularlo.

33. En un sistema operativo se ha decidido utilizar el modelo del área activa para controlar el número de páginas que deben residir en memoria por cada proceso. La estrategia utilizada es la siguiente:

- Cada vez que se realiza un acceso a memoria por parte de un proceso, su área activa es recalculada.
- En todo momento sólo deben estar presentes en memoria las páginas que forman las áreas activas de todos los procesos, siempre y cuando exista disponibilidad de memoria física.
- Si al recalcular el área activa de un proceso P, se detecta que no hay suficiente memoria física, este proceso P es eliminado completamente de memoria y es suspendido.
- Cuando se produzca nueva disponibilidad de memoria física, debe cargarse en memoria el área activa del primer proceso que se encuentre suspendido

En este sistema los parámetros que lo caracterizan son los siguientes:

- El número de referencias que se utilizan para calcular el área activa es 3
- Existen 8 marcos en memoria física

Inicialmente la memoria física esta vacía y llegan simultáneamente, pero en el orden indicado, cuatro procesos (A, B, C y D). En la siguiente tabla se indica la secuencia de accesos a **páginas** que realiza cada proceso dentro de su espacio lógico de direcciones. En el último acceso aparece la letra "T", la

cual indica que el proceso termina, lo que implica que sus páginas deben ser eliminadas de memoria y debe antenderse la reanudación de un proceso suspendido si fuera necesario.

A	0	0	0	7	8	T
B	8	3	5	1	5	T
C	3	4	5	6	7	T
D	0	1	2	2	1	T

El sistema atiende a los procesos siguiendo un turno rotatorio, de tal forma que permite a un proceso emitir una dirección y luego se lo permite al siguiente. Ejemplo: A0, B8, C3, D0, A0, B3 .....

Complete la siguiente tabla de tal forma que describa para cada acceso a memoria en que estado queda la memoria física tras dicho acceso, expresando para cada marco el identificativo del proceso y el número de página que alberga.

Instante	Proceso Página	Marco 0	Marco 1	Marco 2	Marco 3	Marco 4	Marco 5	Marco 6	Marco 7

Solución

Se tienen cuatro procesos en el sistema y 8 marcos, la ventana de área activa es de 3.
Se ubican las páginas en memoria por orden de referencia, teniendo en cuenta que ningún proceso puede tener más de tres páginas en memoria (la ventana de área activa es de 3) y mientras queden marcos libres.
En el instante 9 y después de hacer referencia a la página 5 del proceso B, la memoria se encuentra totalmente llena. En el instante 10 el área activa del proceso C no puede ser ubicada en memoria y el proceso C es suspendido y libera memoria.

Tabla que muestra el contenido de memoria en cada instante:

Instante	Proceso Página	Marco 0	Marco 1	Marco 2	Marco 3	Marco 4	Marco 5	Marco 6	Marco 7
0	A0	A0							
1	B8	A0	B8						
2	C3	A0	B8	C3					
3	D0	A0	B8	C3	D0				
4	A0	A0	B8	C3	D0				
5	B3	A0	B8	C3	D0	B3			
6	C4	A0	B8	C3	D0	B3	C4		
7	D1	A0	B8	C3	D0	B3	C4	D1	
8	A0	A0	B8	C3	D0	B3	C4	D1	
9	B5	A0	B8	C3	D0	B3	C4	D1	B5
10	D2	A0	B8	D2	D0	B3		D1	B5
11	A7	A0	B8	D2	D0	B3	A7	D1	B5
12	B1	A0	B1	D2	D0	B3	A7	D1	B5
13	D2	A0	B1	D2		B3	A7	D1	B5
14	A8	A0	B1	D2	A8	B3	A7	D1	B5
15	B5	A0	B1	D2	A8		A7	D1	B5

16	D1	A0	B1	D2	A8		A7	D1	B5
17	AT	C3	B1	D2	C4			D1	B5
18	BT	C3		D2	C4			D1	
19	C5	C3	C5	D2	C4			D1	
20	DT	C3	C5		C4				
21	C6	C6	C5		C4				
22	C7	C6	C5		C7				
23	CT								

Tabla que muestra el área activa de cada proceso en cada instante:

	Área activa Proceso A	Área activa Proceso B	Área activa Proceso C	Área activa Proceso D
Instante 10	0	8, 3, 5	3, 4, 5 Se suspende C	0, 1
Instante 11	0, 7	8, 3, 5	Suspendido	0, 1, 2
Instante 12	0, 7	3, 5, 1	Suspendido	0, 1, 2
Instante 13	0, 7	3, 5, 1	Suspendido	1, 2
Instante 14	0, 7, 8	3, 5, 1	Suspendido	1, 2
Instante 15	0, 7, 8	5, 1	Suspendido	1, 2
Instante 16	0, 7, 8	5, 1	Suspendido	1, 2
Instante 17	Libera memoria	5, 1	Se ubica su área activa	1, 2
Instante 18	-----	Libera memoria	Se ubica su área activa	1, 2
Instante 19	-----	-----	3, 4, 5	1, 2
Instante 20	-----	-----	3, 4, 5	Libera memoria
Instante 21	-----	-----	4, 5, 6	-----
Instante 22	-----	-----	5, 6, 7	-----
Instante 23	-----	-----	Libera memoria	-----

34. Dada la siguiente lista de referencias emitida por cierto proceso: ... 5 5 8 3 2 2 3 2 3 5 8 2 9 7 (instante$_1$) 7 9 2 2 9 10 9 10 9 10 9 10 10 2 (instante$_2$) ... si se utiliza el modelo del área activa con Δ=10. Indique si las siguientes afirmaciones son verdaderas (V) o falsas (F) y justifique su respuesta.

a) El tamaño del área activa en instante$_1$ es 10 y en instante$_2$, también.

b) El área activa en instante$_2$ está formada por los marcos {2, 9, 10}.

c) El área activa en instante$_1$ está formada por las páginas {2, 3, 5, 7, 8, 9}.

d) Con la información presentada no se puede saber qué páginas forman las áreas activas de este proceso en los instantes 1 y 2, deberíamos conocer también su localidad de referencia.

e) El tamaño del área activa en instante$_2$ es 4.

f) Si este proceso tiene 6 marcos asignados en el instante 1 a las páginas de su área activa (y no tiene otros marcos), no dará ningún fallo de página entre los instantes 1 y 2.

Solución

a) FALSO. El área activa en el instante 1 es de 6 páginas y en el instante 2 de 3 páginas, que son el número de páginas distintas que aparecen en las últimas 10 referencias.
b) FALSO. El área activa no esta formada por marcos, si no por páginas.
c) VERDADERO. El conjunto de páginas distintas que aparecen en las últimas 10 referencias antes del instante 1 son: {2, 3, 5, 7, 8, 9}.
d) FALSO. Es suficiente con la información de las referencias y el tamaño de ventana para poder calcular el área activa.
e) FALSO. El tamaño de área activa en el instante 2 es de 3 páginas.
f) FALSO. Si que da fallo de página, ya que hay referencias a páginas distintas a las que componen el área activa en el instante 1, a partir de dicho instante.

5.- Paginación por demanda: Análisis de eficiencia.

-Tanto en los enunciados como en las soluciones de los problemas de este apartado se emplean un conjunto de siglas que pasamos a definir:

- TAE = TAEF=Tiempo de acceso efectivo.
- MMU= Unidad de manejador de memoria.
- TLB= registros asociativos, la búsqueda en ellos es por contenido.
- TAM= tiempo de acceso a memoria.

-El Tiempo de Acceso Efectivo (TAE) en los sistemas donde se utiliza paginación por demanda, se define como:

$$TAE = (1 - p) * TAM + p * (1 + pm) * TTP$$

donde:
–TAM: Tiempo de acceso a memoria.
–TTP: Tiempo de transferencia página/disco.
–p: Tasa de fallos de página.
–pm: Probabilidad de que la página a reemplazar haya sido modificada.

35. Sea un esquema de paginación por demanda, donde la tabla de páginas reside en memoria y los descriptores de página poseen un bit de modificación (dirty bit). Discuta cuáles son los números máximo y mínimo de accesos a memoria y disco posibles tras la emisión de una dirección por parte de un proceso.

Solución

Se utilizaran en la solución las siguientes abreviaturas:
N= número de palabras de una página
TAM = tiempo de acceso a una posición de memoria para lectura.
2*TAM = tiempo de acceso a una posición de memoria para escritura.
TAMR=Tiempo de lectura de una página en memoria = N * TAM
TAMW= tiempo de escritura de una página en memoria =N *2* TAM.
TADR= Tiempo de lectura de un página en disco.
TADW= Tiempo de escritura de un página en disco = 2*TADR.

Se considera un descriptor de páginas de tamaño igual a una palabra.

Se consideraremos el siguiente criterio para el dirty_bit:

- dirty_bit=0 la página no ha sido modificada, mientras se encontraba en memoria y por tanto si es reemplazada no es necesario actualizarla en el disco.
- dirty_bit=1 la página si ha sido modificada, mientras se encontraba en memoria y por tanto si es reemplazada es necesario actualizarla en el disco.

Como se trata de un sistema de memoria paginado cada vez que se haga referencia a una dirección lógica habrá que consultar las tablas páginas, que en nuestro caso se encuentra en memoria, para localizar el marco de la dirección física correspondiente.

Mejor caso en el número de accesos, no hay fallo de página:

La página referenciada se encuentra en memoria⇒ 1 acceso a memoria para consultar tabla de páginas y otro acceso a memoria a la dirección física correspondiente. Total 2 accesos a memoria = 2 TAM.

Casos con fallo de página y reemplazo:

Pueden ocurrir dos posibilidades que la página a reemplazar tenga el dirty_bit a 0 ó a 1. Es decir que sea necesario actualizar o no la página en el disco.

- Fallo de página con reemplazo y dirty_bit=0.
  - Leer descriptor de página en tabla de páginas en memoria = 1 TAM.
  - Leer página referenciada en el DISCO= TADR.
  - Escribir página en memoria = TAMW = 2N*TAM.
  - Actualizar tabla de páginas en memoria = 1TAM.
  - Se reinicia la solicitud de referencia.

  $Total\_accesos = 2 \cdot (1+N) \cdot TAM + TADR$
- Fallo de página con reemplazo y dirty_bit=1.
  - Leer descriptor de página en tabla de páginas en memoria = 1 TAM.
  - Actualizar página victima en el DISCO= TADW = 2 TADR.
  - Leer página referenciada en el DISCO= TADR.
  - Escribir página en memoria = TAMW =2N*TAM .
  - Actualizar tabla de páginas en memoria = 1TAM.
  - Se reinicia la solicitud de referencia.

  $Total\_accesos = 2 \cdot (1+N) \cdot TAM + 3 \cdot TADR$

36. En un sistema de gestión de memoria segmentado-paginado se tiene una MMU donde únicamente se permiten 8 segmentos por proceso y se emplean dos niveles de paginación, implementando ambos en memoria principal. La tabla de segmentos se ha implementado en registros, con un tiempo de acceso igual a 2 ns. El tiempo de acceso a memoria (TAM) es igual a 20 ns. Para mejorar el rendimiento, se tiene un conjunto de registros asociativos (TLB), donde se guarda como clave tanto el número de segmento como los dos números de página. El tiempo de acceso para estos registros asociativos es igual a 10 ns. Indique cuál será el tiempo de acceso efectivo (TAE) de este sistema, si las direcciones lógicas a traducir se encuentran en el TLB en el 90% de los accesos.

Solución

Tiempo de acceso a registros= Treg= 2 ns

TAM=20 ns

Tiempo de acceso a TLB=Ttlb=10 ns.
Porcentaje de aciertos en TLB= 90% ⇒ tasa de aciertos= p= 0.9
Porcentaje de fallos en TLB=10% ⇒ tasa de fallos=1- p= 0.1
Como se trata de un sistema de memoria segmentado paginado (no hay paginación por demanda), se supone que todo el proceso se encuentra en memoria, por tanto no hay fallos de página, sino simplemente fallos en la búsqueda de descriptores en TLB.
Para poder traducir la dirección lógica a física habrá que conocer el segmento a que pertenece, consultar la tabla de segmentos para comprobar si la dirección es correcta y acceder a tablas de los dos niveles de paginación.
Cuando se emite una dirección lógica en el sistema se sigue el siguiente esquema:

- Se accede a los registros de la MMU para consultar la tabla de segmentos⇒Treg.
- A continuación se realiza una búsqueda en las TLB con el número de segmento y los número de páginas de los dos niveles⇒Ttlb.
  - Si hay acierto en la búsqueda, la TLB contiene el marco donde se encuentra la página referenciada y se accede a memoria para satisfacer la solicitud ⇒TAM.
  - Si no hay acierto en la búsqueda de las TLB, entonces hay que consultar las dos tablas de páginas que se encuentran en memoria, para encontrar el marco donde se encuentra la página referenciada y se accede a memoria para satisfacer la solicitud ⇒3TAM.

Con todo ello el tiempo efectivo de acceso a memoria vendrá dado:

$$TAE = (Treg + TAM + Ttlb)\cdot p + (1-p)(3\cdot TAM + Ttlb + Treg)$$
$$TAE = (2+20+10)\cdot 0.9 + 0.1(3\cdot 20 + 10 + 2) = 36\ ns$$

En este ejercicio se ha considerado que las páginas referenciadas siempre estaban en memoria, y la diferencia radicaba en el camino que se ha seguido para buscar el marco donde se encuentra la página.

37. Se tiene un sistema de memoria virtual con paginación a un nivel que no utiliza intercambios y cuya tabla de páginas se encuentra en memoria principal. Este sistema utiliza una TLB (*Translation Look-aside Buffers*) cuyo porcentaje de aciertos es del 85%. ¿Cuánto empeora el TAE respecto a un sistema que tuviese la tabla de páginas íntegramente en registros del procesador?. Considere que el tiempo de acceso a los registros internos del procesador y a la TLB es despreciable frente al TAM (Tiempo de Acceso a Memoria).

Solución

Caso de que la tabla de páginas se encuentre en registros del procesador:

- En este caso no hay posibilidad de fallo a la hora de encontrar el descriptor de páginas, por tanto cada vez que se emita una dirección lógica se consultan los registros del procesador y se accede a la dirección física solicitada ⇒ Treg_proc +TAM≅TAM

Caso con TLB y tabla de páginas en memoria:

- En este caso cada vez que se emita una dirección lógica se ha de consultar en TLB ⇒ Ttlb≅0.
  - Si hay acierto en la búsqueda, lo cual ocurre con una probabilidad p (p=0.85), se accede a memoria a la dirección física solicitada TAM⇒ pTAM.
  - Si no hay acierto en la búsqueda, lo cual ocurre con una probabilidad 1-p (1-p=0.15), hay que acceder dos veces a memoria una para encontrar el descriptor de página y otra para acceder a la dirección física solicitada 2*TAM⇒ (1-p)TAM.

$$TAE_1 = TAM;$$
$$TAE_2 = p \cdot TAM + (1-p) \cdot 2 \cdot TAM = 0.85 \cdot TAM + 0.15 \cdot 2 \cdot TAM = 1.15 \cdot TAM;$$
$$TAE_2 - TAE_1 = 0.15 \cdot TAM$$

El porcentaje de mejora que se consigue utilizando los registros del procesador vendrá dado por:

$$\frac{TAE_2 - TAE_1}{TAE_2} \cdot 100 = \frac{0.15}{1.15} \cdot 100 \cdot TAM = 13.04\% \, TAM$$

38. En un sistema de gestión de memoria paginado, se han incluido unos registros asociativos (TLB) para mejorar el rendimiento del sistema. La búsqueda en estos registros cuesta 20 ns, mientras que su búsqueda y actualización cuesta 40 ns. El tiempo de acceso a memoria (TAM) es igual a 100 ns. Suponiendo que la tasa de acierto en el TLB es de un 80%, calcular el tiempo de acceso efectivo a memoria en este sistema.

Solución

TAM=100 ns
Tiempo de búqueda en TLB=Ttlbusq=20 ns.
Tiempo de búqueda y actualización en TLB=Ttlbusq_act=40 ns.
Porcentaje de aciertos en TLB= 80% ⇒ tasa de aciertos= p= 0.8
Porcentaje de fallos en TLB=20% ⇒ tasa de fallos=1- p= 0.2
Como se trata de un sistema de memoria paginada, sin memoria virtual, para poder traducir la dirección lógica a física habrá consultar las tablas páginas.
Cuando se emite una dirección lógica se realiza una búsqueda en las TLB con el número de página de dicha dirección lógica ⇒ Ttlbusq.

- Si hay un acierto en la búsqueda, lo cual ocurre con una probabilidad p, la TLB contiene el descriptor de la página referenciada y a continuación se accede a la posición de memoria referenciada TAM ⇒ p. TAM.
- Si no hay acierto en la búsqueda de las TLB, lo cual ocurre con una probabilidad 1-p, entonces hay que consultar la tabla de páginas en memoria TAM y actualizar las TLB para que contenga el descriptor de la nueva página referenciada Ttact. Una vez se conoce el marco donde se encuentra la página se ha de acceder a memoria para satisfacer la solicitud TAM⇒ (1-p) (2TAM+Ttact).

Con todo ello el tiempo efectivo de acceso a memoria vendrá dado:

$$TAE = (TAM + Ttlbusq)\cdot p + (1-p)(2\cdot TAM + Ttlbusq\_act)$$
$$TAE = (100+20)\cdot 0.8 + 0.2\,(2\cdot 100+40) = 144\ ns$$

En este ejercicio se ha considerado que las páginas referenciadas siempre estaban en memoria.

39. **D**ado un esquema de gestión de memoria basado en paginación y donde parte de la tabla de páginas se almacena en una TLB, indique cuál sería el porcentaje de aciertos en la TLB para obtener un TAE inferior a 50 ns. El tiempo de acceso a la TLB es de 5 ns y el tiempo de acceso a memoria (TAM) es 30 ns.

Solución

TAM=30 ns

Tiempo de búqueda en TLB=Ttlbusq=5 ns.

TAE<50ns

Como se trata de un sistema de memoria paginada, sin memoria virtual, para poder traducir la dirección lógica a física habrá consultar las tablas páginas.

Cuando se emite una dirección lógica se realiza una búsqueda en las TLB con el número de página de dicha dirección lógica ⇒ Ttlbusq.

- Si hay un acierto en la búsqueda, lo cual ocurre con una probabilidad p, la TLB contiene el descriptor de la página referenciada y a continuación se accede a la posición de memoria referenciada TAM ⇒ p. TAM.
- Si no hay acierto en la búsqueda de las TLB, lo cual ocurre con una probabilidad 1-p, entonces hay que consultar la tabla de páginas en memoria TAM. Una vez se conoce el marco donde se encuentra la página se ha de acceder a memoria para satisfacer la solicitud TAM⇒ (1-p) (2TAM).

Con todo ello el tiempo efectivo de acceso a memoria vendrá dado:

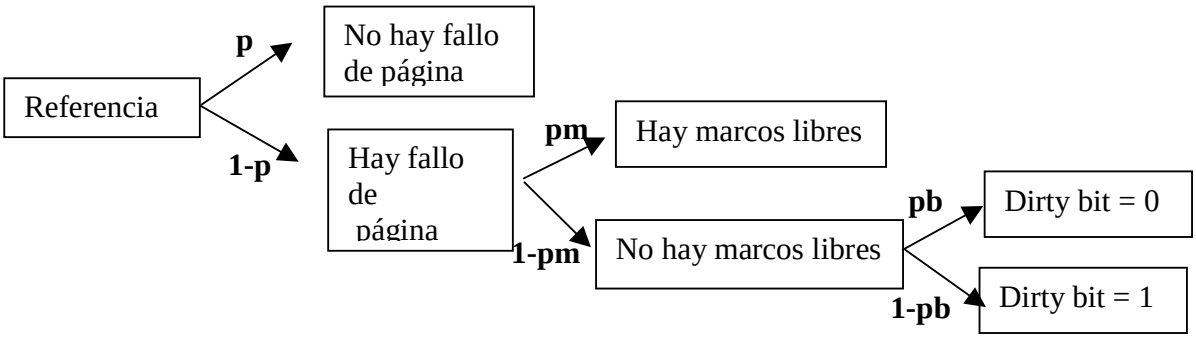
$$TAE = (TAM + Ttlbusq)\cdot p + (1-p)(2\cdot TAM + Ttlbusq) < 50ns$$
$$TAE = (30+5)\cdot p + (1-p)\,(2\cdot 30+5) = 35p + 65 - 65p = 65 - 30p < 50\ ns$$
$$p > \frac{15}{30} = 0{,}5$$

En este ejercicio se ha considerado que las páginas referenciadas siempre estaban en memoria.

40. Sea un sistema con paginación por demanda donde la tabla de páginas se implementa en registros. El tiempo de servicio de un fallo de página en caso de que haya marcos libres o la página reemplazada no ha sido modificada (dirty bit a 0) es de 8 ms, mientras que el tiempo de servicio en caso de que la página a reemplazar haya sido modificada (dirty bit a 1), el tiempo de servicio es de 20 ms. La probabilidad de encontrar un marco libre cuando se produce un fallo de página es de 0.5. La página a reemplazar tiene el dirty bit a 1 el 70% de las ocasiones. El tiempo de acceso a memoria física es de 100µs. ¿Cuál es la mayor tasa de fallos de página (p) aceptable para que el tiempo de acceso efectivo no exceda de 200 µs?.

Solución

Consideraremos que la tabla de páginas está completa en los registros y por tanto sólo puede haber fallo de página de la página donde se encuentra la referencia solicitada (no puede haber fallo de página de las páginas de la tabla de páginas).
Cada vez que se emita una dirección lógica en este sistema las situaciones que pueden aparecer son las siguientes:



Consideraremos el tiempo de acceso a estos registros es despreciable ⇒ Treg≈0.
Tiempo de servicio de fallo de página cuando hay marcos libres= TFP1=8ms.
Tiempo de servicio de fallo de página con dirty bit a 0= TFP1=8ms
Tiempo de servicio de fallo de página con dirty bit a 1= TFP2=20ms.
Pm= 0.5
1-pb=0.7; pb=0.3;
TAM = 100μseg.
Como se trata de un sistema de memoria con paginación por demanda para traducir direcciones lógicas a física habrá que consultar las tablas páginas.
Como la tabla de páginas se encuentra en registros, cada vez que se emite una dirección lógica se realiza una búsqueda en registros ⇒ Treg≈0.

- Si no hay fallo de página, lo cual ocurre con una probabilidad p, se accede a la posición de memoria referenciada TAM ⇒ p. TAM.
- Si hay fallo de página, lo cual ocurre con una probabilidad 1-p, entonces hay que traerse la página de disco y ubicarla en memoria
  - Si hay marcos libres lo cual ocurre con una probabilidad pm ⇒ (pm *TFP1)
  - Si no hay marcos libre lo cual ocurre con una probabilidad 1-pm tendremos dos casos:
    - Dirty bit =0 ⇒ (1-pm)*pb*TFP1
    - Dirty bit =0 ⇒ (1-pm)*(1-pb)*TFP2.

Con todo lo cual el tiempo de acceso efectivo a memoria quedará:

$$TAE = p\,TAM + (1-p)(pm \cdot TFP1 + (1-pm)(pb \cdot TFP1 + (1-pb)TFP2)) < 200\mu s$$

$$TAE = p\,TAM + (1-p)(pm \cdot TFP1 + (1-pm)(2.4ms + 14ms)) < 200\mu$$

$$TAE = p\,TAM + (1-p)(4ms + 8.2ms) < 200\mu s$$

$$TAE = p\,100\mu s + (1-p)(12200\mu s) = 12200 - 12100\,p < 200\mu s$$

$$p > \frac{12000}{12100} = 0{,}9917$$