Tema 7. Memòria Virtual Problemes

Joan Manuel Parcerisa

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

Problema 7.1

7.1. Es disposa d'un sistema amb memòria virtual paginada, amb les característiques següents:

- Espai d'adreces dels programes (grandària de la memòria virtual)= 16 Mbytes
- Espai de memòria física= 16 Kbytes
- Mida de les pàgines = 256 bytes

Cada programa pot tenir a memòria física un màxim de 3 pàgines simultàniament. En cas que una d'aquestes 3 pàgines hagués de ser sacrificada per deixar lloc a una altra, es fa servir l'algorisme de reemplaçament LRU.

Es demana:

a) Indica les dimensions de la taula de pàgines de cada programa, tenint en compte que cada entrada de la taula té un bit de presència (indica si la pàgina corresponent està carregada a memòria física) i un bit de modificació (indica si la pàgina corresponent ha estat modificada durant la seva estada a la memòria física).

Problema 7.1

b) A la taula següent apareix una llista d'adreces virtuals (en hexadecimal) que són generades pel processador. Omple la taula indicant, per a cadascuna de les adreces, el número de pàgina en hexadecimal corresponent (columna A), si l'accés produeix una fallada de pàgina o no (columna B), i, en cas que la fallada de pàgina hagi produït un reemplaçament, el número de pàgina reemplaçada (columna C). Considera que inicialment no hi ha cap pàgina carregada a memòria física.

Adreça (hex)	A Núm. pàgina (hex)	B Fallada de pàgina (SI o NO)	C Núm. pàgina reemplaçada (hex)
000023			
000101			
000102			
000200			
000010			
000111			
000416			
000520			
000101			

Problema 7.2

7.2. Un sistema experimental es gestiona amb memòria virtual sota paginació amb algorisme de reemplaçament LRU. Aquest sistema té 64 KB de memòria virtual i 8 KB de memòria física. Les pàgines tenen 1 KB de mida. La taula de pàgines està carregada sempre a memòria física, ocupant l'últim marc de pàgina.

Suposem que una certa aplicació és ubicada dins l'espai lògic en les següents adreces:

- El codi del programa se situa en el rang d'adreces: 0x0000 - 0x03FF
- Les dades del programa se situen en el rang d'adreces: 0x0400 - 0x47FF

Dins les dades del programa hi ha una matriu que en alt nivell ha estat declarada com:

```
int MAT[32][256];
```

Aquesta matriu s'emmagatzema a memòria a partir de l'adreça base 0x0400. Tingues en compte que els enters es codifiquen en 16 bits. La resta de variables del programa, així com la pila, s'emmagatzemen a partir de l'adreça 0x4400.

Considerem el següent fragment del codi de l'aplicació:

```
for (i=0; i<32; i++)
     for (j=0; j<256; j++)
          MAT[i][j] = i+j;
```

on:

- El compilador reserva dos registres del processador per a les variables `i` i `j`.
- La taula de pàgines abans de començar a executar l'anterior codi indica que a memòria física només tenim la pàgina corresponent al codi de l'aplicació.

Problema 7.2

Es demana contestar els següents apartats, que fan referència a l'execució d'aquest fragment de codi:

a) Quins elements de MAT es porten a memòria física en la primera fallada de pàgina que es produeix?

b) Indica, justificant-ho raonadament, el nombre total de fallades de pàgina que es produiran.

c) En alguns llenguatges, com ara FORTRAN, els elements de les matrius es guarden en posicions consecutives però *per columnes* en lloc de *per files*. Considerant aquesta altra manera d'emmagatzemar les matrius, quins elements de MAT es portarien a memòria física en la primera fallada de pàgina?

d) Considerant l'emmagatzematge per columnes de l'apartat anterior, indica, justificant-ho raonadament, el nombre total de fallades de pàgina que es produirien.

Problema 7.4

7.4. Tenim un processador amb memòria virtual basada en paginació. El sistema de memòria virtual té les següents característiques:

- 16 bits d'adreça lògica, i 15 bits d'adreça física
- mida de pàgines: 8 KB
- reemplaçament LRU

El contingut inicial de la taula de pàgines es mostra a continuació, on P és el bit de presència, D és el bit de pàgina modificada i PPN el número de pàgina física.

	P	D	PPN
0	0	0	-
1	0	0	-
2	1	0	0
3	1	0	1
4	1	0	2
5	0	0	-
6	0	0	-
7	0	0	-

A continuació també es mostra el contingut inicial de la memòria, en la qual la pàgina física 1 (lògica 3) és la que fa menys temps que ha estat accedida, la següent és la 0 (lògica 2) i la que fa més temps és la 2 (lògica 4).

pàgina física	pàgina lògica
0	2
1	3
2	4
3	-

Problema 7.4

La següent taula mostra una seqüència de referències a memòria (E: escriptura/ L: lectura). Emplena la taula indicant, per cada referència el número de pàgina lògica (VPN) i el número de pàgina física resultant de la traducció (PPN). Indica també amb una creu (X) si es produeix fallada de pàgina, si es llegeix del disc, i si s'escriu al disc. Indica també, en cas de reemplaçar una pàgina, el VPN i PPN de la pàgina reemplaçada. Per últim, indica també el contingut final de la taula de pàgines i de la memòria física.

adr. lògica (hex)		VPN (hex)	PPN (hex)	fallada de pàgina	lectura disc	escriptura disc	pàg. reemplaçada	
							VPN (hex)	PPN (hex)
E	F458							
E	8666							
L	1BBF							
E	5C44							
L	6600							
L	4000							

Problema 7.3

7.3. Considerem un computador amb processador MIPS com l'estudiat a classe, que té una memòria cache de dades amb les següents característiques:

- associativa per conjunts de 4 vies (és a dir, de 4 blocs per conjunt)
- 64 conjunts
- 32 bytes per bloc
- algorisme de reemplaçament LRU

Prescindirem de la part de cache (idèntica al problema 6.11, ja resolt)

Considerem també que té un TLB de dades amb les següents característiques:

- completament associatiu
- 32 entrades
- mida de pàgina: 8 KB
- algorisme de reemplaçament LRU

Sobre aquest sistema s'executen 2 versions diferents d'una mateixa aplicació:

```
int A[128][1024];/* emmagatzemada a partir de l'adreça 0 */

/* versió A */
sumA = 0;
for (i=0; i<128; i++)
        for (j=0; j<1024; j++)
                sumA = sumA + A[i][j];

/* versió B */
sumA = 0;
for (j=0; j<1024; j++)
        for (i=0; i<128; i++)
                sumA = sumA + A[i][j];
```

Suposant que les variables `i`, `j`, i `sumA` s'emmagatzemen en registres, indica quantes fallades hi ha a la cache i al TLB, per a cada versió.

Problema 7.5

7.5. Considerem el següent codi escrit en assemblador del MIPS, en què a l'etiqueta A li correspon l'adreça 0x10010000:

```
      la      $s0, A
      li      $s1, 0
      li      $s2, 512000
for:  bge     $s1, $s2, fi
      sll     $t0, $s1, 2
      addu    $t0, $s0, $t0
      lw      $t1, 0($t0)
      lw      $t2, 8192($t0)
      addu    $t2, $t2, $t1
      sw      $t2, 8192($t0)
      sw      $t2, 16384($t0)
      addiu   $s1, $s1, 512
      b       for
fi:
```

Suposant que el sistema de memòria té les pàgines de mida 8 KB i que utilitza un TLB de 4 entrades (completament associatiu, i amb reemplaçament LRU), respon les següents preguntes:

a) Per a cadascun dels accessos a dades de memòria (instruccions en negreta) i per a les primeres 5 iteracions del bucle, indica a quina pàgina de la memòria virtual s'està accedint.

b) Indica la quantitat de fallades de TLB en tot el bucle.

c) Respon a les mateixes preguntes anteriors, suposant ara que el sistema de memòria té les pàgines de mida 4 KB.

Problema 7.6

7.6. Considerem un sistema amb memòria virtual que té pàgines de 4 KB. A més, té un TLB de 4 entrades completament associatiu, on s'utilitza un reemplaçament LRU.

Considerem els següents continguts inicials del TLB (per al funcionament de l'algorisme LRU, considerem que les tres entrades vàlides han estat recentment accedides en el mateix ordre que ocupen dins el TLB):

TLB		
V	VPN	PPN
1	11	12
1	7	4
1	3	6
0	9	0

Considerem també els següents continguts inicials de la taula de pàgines. Per a noves pàgines assignades a MP, s'assignaran números de pàgina física (PPN) correlatius, a partir del número major (és a dir, els números 13, 14, etc.).

	TP	
	P	PPN
0	1	5
1	0	-
2	0	-
3	1	6
4	1	9
5	1	11
6	0	-
7	1	4
8	0	-
9	0	-
10	1	3
11	1	12

Problema 7.6

a) Per a cada una de les dues llistes de referències d'adreces virtuals que es mostren a continuació, indica quin serà l'estat final del TLB i de la taula de pàgines (TP). A més, per a cada referència, indica si es produeix una fallada al TLB, i si es produeix una fallada de pàgina.

- Llista 1: `4095, 31272, 15789, 15000, 7193, 4096, 8912`
- Llista 2: `9452, 30964, 19136, 46502, 38110, 17653, 47480`

b) Repeteix l'apartat anterior però en lloc de considerar les pàgines de 4 KB considera-les ara de 16 KB.

c) Considerant els paràmetres següents de dos sistemes que gestionen memòria virtual, indica quina serà la mida de la taula de pàgines en cada cas.

Mida adreces virtuals	Mida pàgina	Mida entrades TP
32 bits	4 KB	4 bytes
64 bits	16 KB	8 bytes

Examen 2010/11-Q1 (adaptat)

Donat el següent programa en C

```
int M[144][256], i, tmp;
for (i=0; i< 128; i++) {
        tmp = M[i][0];
        M[i+4][0] = tmp;
        M[i+16][0] = M[i+16][0] + tmp;
}
```

S'ha compilat per a un processador de 32 bits amb memòria virtual paginada

- Les variables *i* i *tmp* s'han guardat en registres
- L'adreça lògica inicial de M és 0x00000000
- La mida de les pàgines és 4KB
- El TLB té 4 entrades (associatiu, amb reemplaçament LRU)

a) Per a cada un dels 4 accessos a la matriu M (en negreta), indica a quina pàgina virtual (VPN) s'accedeix en cada una de les 16 primeres iteracions

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
M[i][0]																
M[i+4][0]																
M[i+16][0]																
M[i+16][0]																

b) Calcula el nombre d'encerts i fallades del TLB, en tot el bucle, per a cada instrucció

c) Suposant que tinguéssim un TLB amb un nombre infinit d'entrades, ¿quin seria el nombre total de fallades del TLB en aquest bucle?