Sistemas Operativos UCM 2012/2013

Soluciones Módulo 5 Gestión de memoria

1. Un sistema informático tiene sitio suficiente en memoria principal para contener …

Un primer razonamiento nos llevaría a deducir que si un proceso está inactivo la mitad del tiempo existe una probabilidad 0,5 de que la CPU esté desaprovechada por lo que a ese proceso respecta. Si hay 4 programas ejecutándose con el mismo comportamiento en cuanto a proporciones de actividad, para que la CPU no estuviera ocupada en ejecutar alguno de los programas deberían coincidir en sus períodos inactivos todos ellos; esta situación se dará con probabilidad (0,5)*(0,5)*(0,5)*(0,5) = $0,5^4$ = 0,0625. Es decir, la CPU estaría inactiva el 6,25% del tiempo total.

El razonamiento anterior adolece de un defecto: supone que pueden coincidir períodos de actividad simultánea de varios de los programas, cuando en un sistema informático de una sola CPU sólo puede haber ejecutándose un programa como máximo. Un razonamiento más ajustado es el que considera la situación como un caso de probabilidades de Markov

2. Encontrar una expresión que permita predecir el aumento de productividad …

El aumento de productividad lo podremos establecer comparando la actividad del procesador cuando hay un sólo proceso disponible, con la actividad cuando hay *n* procesos disponibles (el grado de multiprogramación es *n*).

Asumiendo el razonamiento más simple del problema 1, diremos que la probabilidad de que el procesador esté inactivo, para *n* procesos disponibles, es de p^n; por tanto la probabilidad de que esté activo será 1 - p^n. El aumento de productividad, como porcentaje, es por tanto, de

$$\left(\frac{1-p^n}{1-p}-1\right)*100$$

Los datos numéricos del problema nos dicen que p = 0,6 y que inicialmente n = 3, y al ampliar la memoria principal en 1M, n pasa a valer 5. El aumento de productividad será

$$\frac{1-p^5}{1-p^3}=\frac{1-0,6^5}{1-0,6^3}=\frac{0,922}{0,784}=1,176$$

es decir, la productividad ha aumentado un 17,6%.

3. Considerar un sistema de tiempo compartido con *swapping*, un disco …

a) Con tres particiones las posibilidades de ejecución y transferencia a disco son las tres siguientes, que mantienen ocupado el disco son:
   1. P1 ejecutándose; P2 leyéndose de disco (swap-in) y luego P3 escribiéndose a disco (swap-out)
   2. P1 en swap-out, mientras P2 se ejecuta; P3 en swap-in (después del swap-out de P1)
   3. P1 en swap-in y luego P2 en swap-out; mientras P3 en ejecución

   Luego el tiempo de ejecución continuada sobre una partición debe ser igual a l de dos transferencias (swap-in y swap-out) de discos seguidas que llevan 2 * (6 + 4) = 20 msg

b) Si dos turnos de ejecución del mismo programa deben estar separados 1 segundo, el número de turnos de ejecución de otros programas que pueden ejecutarse mientras siguiendo el esquema anterior es de 1s / 20ms = 50 turnos.

4. Considerar un sistema con 4200 palabras de MP que implementa particiones variables…

	Particiones ocupadas Inicio[Tamaño]	Huecos libres Inicio[Tamaño]
Inicialmente	P1: 1000 [1000] P2: 2900 [500] P3: 3400 [800]	H1: 0000 [1000] H2: 2000 [900]
Petición N1[500]: Usa el hueco H2	P1: 1000 [1000] N1: 2000 [500] P2: 2900 [500] P3: 3400 [800	H1: 0000 [1000] H3: 2500 [400]
Petición N2[1200]: Compactación mínima y uso del hueco resultante	P1: 0000 [1000] N1: 1000 [500] P2: 2900 [500] P3: 3400 [800]	H1: 1500 [1400]
	P1: 0000 [1000] N1: 1000 [500] N2: 1500 [1200] P2: 2900 [500] P3: 3400 [800	H1: 2700 [200]
Petición N3 [200]	P1: 0000 [1000] N1: 1000 [500] N2: 1500 [1200] N3: 2700 [200] P2: 2900 [500] P3: 3400 [800	(vacío)

5. El gestor de memoria recibe la siguiente secuencia de peticiones …

	Primer Ajuste		Mejor ajuste	
Situación inicial	(vacío)	000 [1024]	==	==
Entra B1 [100]	B1: 000 [100]	100 [924]	==	==
Entra B2 [500]	B1: 000 [100] B2: 100 [500]	600 [424]	==	==
Entra B3 [60]	B1: 000 [100] B2: 100 [500] B3: 600 [60]	660 [364]	==	==
Entra B4 [100]	B1: 000 [100] B2: 100 [500] B3: 600 [60] B4: 660 [100]	760 [264]	==	==
Sale B1	B2: 100 [500] B3: 600 [60] B4: 660 [100]	000 [100] 760 [264]	==	==
Sale B3	B2: 100 [500] B4: 660 [100]	000 [100] 600 [60] 760 [264]	==	==
Entra B5 [50]	B5: 000 [50] B2: 100 [500] B4: 660 [100]	050 [50] 600 [60] 760 [264]	B2: 100 [500] B5: 600 [50] B4: 660 [100]	000 [100] 650 [10] 760 [264]
Entra B6 [90]	B5: 000 [50] B2: 100 [500] B4: 660 [100] B6: 760 [90]	050 [50] 600 [60] 850 [174]	B6: 000 [90] B2: 100 [500] B5: 600 [50] B4: 660 [100]	090 [10] 650 [10] 760 [264]

6. Sea **A** el tamaño medio en bytes de los procesos que se ejecutan en un sistema …

El uso de páginas introduce fenómenos de fragmentación interna que podemos aproximar para cada proceso como p / 2, y de fragmentación de tabla, que podemos aproximar como (A/p) x b. Luego, si designamos D(p) al espacio desaprovechado:

$$D(p) = p/2 + (A/p)\, b$$

Tomando la derivada de D(p) con respecto a p, e igualando a 0, obtenemos:

$$D'(p)=1 / 2 - (A\, b) / p^2 = 0$$

de donde deducimos el valor de p que hace mínima/máxima la función D(p)

$$p = (2\, A\, b)^{1/2}$$

D''(p) para el p anterior es > 0, por lo que se trata de un mínimo.

Para A = 1MB y b = 4 bytes, p = (2 x 1024 x 1024 x 4) 1/2 = 2355,2 ≈ 2,83 Kb → 2 KB ó 4 KB

7. Un computador permite que cada proceso ocupe un espacio de direcciones …

Tendremos en cuenta que una página no puede albergar datos de segmentos distintos.

a) El espacio virtual está formado por 16 páginas de 4KB cada una
El segmento de TEXTO ocupará 32KB / 4KB = 8 páginas
El segmento de DATOS ocupará 16386 / 4096 = 4,0005 ⇒ 5 páginas
El segmento de PILA ocupará 15780 / 4096 = 3,8745 ⇒ 4 páginas
El número de páginas necesario es 8+5+4 = 17, por lo que el programa no cabe.

b) El espacio virtual está formado por 128 páginas de 512 bytes cada una
El segmento de TEXTO ocupará 32768 /512 = 64 páginas
El segmento de DATOS ocupará 16386 / 512 = 32,004 ⇒ 33 páginas
El segmento de PILA ocupará 15780 / 4096 = 30,82 ⇒ 31 páginas
El número de páginas necesario es 64+33+31 = 128, por lo que el programa cabe.

8. Un computador cuyos procesos tienen 1024 páginas en sus espacios de direcciones …

Sea h la tasa de aciertos de la memoria TLB

$$h\, t_{TLB} + (1 - h)\, (t_{TLB} + t_{TMP}) = 200 \implies h = 1 - (200 - 100) / 500 = 0{,}8 \implies \quad h = 80\%$$

9. Considerar los cuatro sistemas siguientes…

Sistema A: Una página tiene 512 palabras de 16 bits, e.d., 1024 bytes;
luego, cada dirección virtual tiene 6b de página y 10 de desplazamiento →64 elementos en cada TMP
Sistema B: Una página tiene 512 palabras de 32 bits, e.d., 2048 bytes;
luego, cada dirección virtual tiene 21b de página y 11 de desplazamiento → 2M elementos en cada TMP
Sistema C: Una página tiene 1024 palabras de 16 bits, e.d., 2048 bytes;
luego, cada dirección virtual tiene 5b de página y 11 de desplazamiento → 32 elementos en cada TMP
Sistema D: Una página tiene 1024 palabras de 32 bits, e.d., 4096 bytes;
luego, cada dirección virtual tiene 2b de página y 12 de desplazamiento → 1M elementos en cada TMP

10. Considerar la siguiente tabla de segmentos…

a) (0-430) → 219+430 = 649
b) (1-10) → 2300+10 = 2310
c) (1-11) → 2300+11 = 2311
d) (2-500) → error de acceso: máxima longitud, 100
e) (3-400) → 1327+400 = 1727
f) (4-112) → error de acceso: máxima longitud, 96

11. Un sistema de paginación pura tiene un tamaño de página de 512 palabras

a) TMP: entradas 0-8 : P=0 → no válidas
entrada 9: P=1, marco = 4
entrada 10: P=1, marco = 9
entradas 11-33: P=0 → no válidas
entrada 34: P=1, marco = 3
entradas 35-64: P=0 → no válidas
entrada 65: P=1, marco = 7
entradas 66-511: P=0 → no válidas

b) TMP: entradas 0-8 : P=0 → no válidas
entrada 9: P=1, marco = 4
entrada 10: P=1, marco = 9
entradas 11: P=0 → no válidas
entrada 12: P=1, marco = 3
entradas 13-48: P=0 → no válidas
entrada 49: P=1, marco = 0
entradas 50-64: P=0 → no válidas
entrada 65: P=1, marco = 7
entradas 66-511: P=0 → no válidas

c) dv: 4608, e.d, (9,0) → df: (4,0), e.d., 2048+0 = 2048
dv: 5119, e.d., (9,511) → df: (4,511), e.d., 2048+511 = 2559
dv: 5120, e.d., (10,0) → df: (9,0), e.d., 4608+0 = 4608
dv: 33300, e.d., (65,20) → df: (7,20), e.d., 3584+20 = 3604

d) dv: 33000, e.d., (64,232) → fallo de página

e) Debe estar en la misma página (10) para que las direcciones internas del procedimiento sean correctas. Pero si el procedimiento es PIC (Independiente de la posición, e.d, solo usa direccionamiento relativo), puede estar en cualquier página.

12. Considerar dos procesos P1 y P2 en un sistema segmentado puro …

a) Los segmentos 1 de P1 y P2, los segmentos 2 de P1 y P2, y los segmentos 3 de P1 y 0 de P2

b) Son direcciones generadas en un segmento compartido (el 1 en P1 y P2)
(0,0) MAL: si la genera un proceso (P2) apunta a un segmento compartido y si la genera el otro (P1) no
(1,0) BIEN: apunta al propio procedimiento
(2,0) BIEN: apunta a un segmento compartido con la misma numeración
(3,0) MAL: por la misma razón que (0,0), aunque el papel de los procesos (P1,P2) es el contrario
(4,0) MAL: si la genera un proceso (P1) apunta a un segmento definido, y si la genera otro (P2) no
(5,0) MAL: apunta a un segmento no definido (P1, P2)

c) P1: 0 → 4000, 1 → 6000, 2 → 9000, 3→ 1000, 4 → 8000
P2: 0 → 1000, 1 → 6000, 2 → 9000

13. Si se utilizase sustitución de páginas FIFO con cuatro marcos de página y…..

Estrategia FIFO

Cadena de referencias =	0	1	7	2	3	2	7	1	0	3
Fallo de página =	x	x	x	x	x				x	
Página sustituída =					0				1	

Estrategia LRU

Cadena de referencias =	0	1	7	2	3	2	7	1	0	3
Fallo de página =	x	x	x	x	x				x	x
Página sustituída =					0				3	2

14. Suponiendo una memoria física de cuatro marcos de página …

Cadena de referencias	a b g a d e a b a d e g d e	para 4 marcos

Política OPT	**a** a a **a** a a **a** a **a** a *a* **g** g g
	  b b b b b b **b** b b b b b b
	  g g *g* **e** e e e **e** e e **e**
	  d d d d d **d** d d **d** d
		6 fallos

Política FIFO	**a** a a **a** *a* **e** e e e **e** *e* **d** d
	  b b b b *b* **a** a **a** a a *a* **e**
	  g g g g *g* **b** b b b b b b
	  d d d d d **d** *d* **g** g g
		10 fallos

Política Reloj	**a** a a **a** a a **a** a **a** a a a a
	  b b b *b* **e** e e e e **e** e e **e**
	  g g g g *g* **b** b b *b* **g** g g
	  d d d d d **d** d d **d** d
		7 fallos

Política LRU	**a** a a **a** a a **a** a **a** a a a a
	  b b b *b* **e** e e e e **e** e e **e**
	  g g g g *g* **b** b b *b* **g** g g
	  d d d d d **d** d d **d** d
		7 fallos

15. Considerar la siguiente secuencia de referencias a memoria virtual …

a) La cadena de referencias es 0, 0, 1, 1, 0, 3, 1, 2, 2, 4, 4, 3
y reducida 0, 1, 0, 3, 1, 2, 4, 3

b)

OPT	0 0 0 0 0 3 3 3 3 3 3 3 1 1 1 1 1 2 2 4 4 4	5 fallos
FIFO	0 0 1 1 1 3 3 2 2 4 4 3 0 0 0 1 1 3 3 2 2 4	6 fallos
LRU	0 0 1 1 0 3 1 2 2 4 4 3 0 0 1 0 3 1 1 2 2 4	7 fallos
Reloj	**0 0’0’0’0’**3 3 2 2**’2’2’**3 1 1’1’**1 1’1 1** 4 4’**4**	6 fallos

16. Considera los siguientes programas …

1º) FIFO o LRU igual

(a)	2 fallos cada 100	⇒	2 x 10 = 20 fallos
(b)	2 fallos cada 100	⇒	2 x 10 = 20 fallos
(c)	2 fallos cada 100	⇒	2 x 10 = 20 fallos
(d)	2 fallos cada 500	⇒	2 x 2 = 4 fallos

2º) FIFO o LRU igual

(a) (d)	3 fallos cada instrucción	⇒ 3 x 1000 = 3000 fallos
(b)	3 fallos cada 100 hasta 500 2 fallos de 500 a 600* 3 fallos cada 100 hasta 1000	⇒ 15 + 2+ 12 = 2 fallos
(c)	3 fallos cada 100 hasta 500 4 fallos de 500 hasta 800 3 fallos cada 100 hasta 1000	⇒ 15 + 4 + 6 = 25 fallos

En i=500 no falla a[500] (estaba debido a acceso a[999-i] de la iteración anterior) pero sí b[500], que pisa el marco de a[400-499] y por tanto falla a[999-500], 2 fallos

17. Supongamos un sistema cuyo **TLB** (Translation Look-aside Buffer) y **TP** (Tabla de Páginas) contiene la siguiente información:

a) Indicar cuáles generan acierto o fallo en el TLB y en la TP así como los cambios que hace el SO en ambas tablas. ¿Cuáles son las direcciones físicas tras la conversión?
b) Suponga que un acceso al TLB supone 2ns, un acceso a la TP 50ns y un fallo de página 5ms. ¿Cuánto tiempo tardan en servirse las peticiones 0x40, 0x10 y 0xAF empleando los valores iniciales de las tablas?

NOTA: Las entradas del TLB están ordenadas de más antiguas a más nuevas, siendo la más antigua la primera (Pag. 4 -- Marco 2), en caso de reemplazamiento usar el algoritmo FIFO.

- 0x4A -> acierto en el TLB, dir física 0x2A
- 0x40 -> acierto en el TLB, dir física 0x20
  - Tiempo 2ns
- 0xC7 -> acierto en el TLB, dir física 0x47
- 0x10 -> fallo TLB, acierto TP, dir física 0xF0
  - saco [4-3] y meto [1-15] al final del TLB
  - tiempo 2ns + 50ns
- 0x43 -> fallo TLB, acierto TP, dir física 0x23
  - saco [15-3] y meto [4-2] al final del TLB
- 0xAF -> fallo TLB y TP, nuevo marco->**(X)** dir física 0x**(X)**F
  - saco [12-4] y meto [10-**(X)**]
  - tiempo 2ns + 50ns + 5ms
  -

18. Considerar la siguiente secuencia…

0x10, 0x1A, 0x1F4, 0x17C, 0x7C, 0x3B9, 0x185, 0x2FF, 0x24C, 0x434, 0x458, 0x36D

a) Deducir la cadena de referencias …

La cadena de referencias es **0, 0, 1, 1, 0, 3, 1, 2, 2, 4, 4, 3**
y reducida **0, 1, 0, 3, 1, 2, 4, 3**

b) Determinar razonadamente …

OPT — **5 fallos**

Marco 0	0	0	0	0	0	3	3	3	3	3	3	
Marco 1		1	1	1	1	1	2	2	4	4	4	

FIFO — **6 fallos**

Marco 0	0	0	0	0	0	3	3	3	3	4	4	4
Marco 1		1	1	1	1	1	2	2	2	2	3	

LRU — **7 fallos**

Marco 0	0	0	0	0	0	0	1	1	1	4	4	4
Marco 1		1	1	1	3	3	2	2	2	2	3	

Reloj — **6 fallos**

Marco 0	0	0’	0’	0’	0’	3	3	2	2’	2’	2’	3
Marco 1		1	1’	1’	1	1’	1	1	4	4’	4	