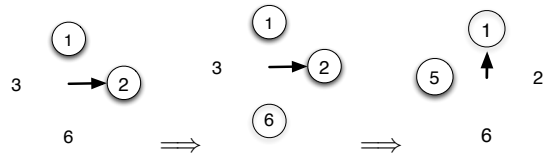
Clase Auxiliar 9

Prof: L. Mateu
Aux: J.Bustos

1. P1 Examen 2002/02

Parte a: La siguiente figura muestra los estados sucesivos de la memoria al realizar 2 accesos en un sistema de memoria virtual que usa la *estrategia del reloj para el reemplazo de páginas*.



Las páginas que tienen el bit de referencia en 1, aparecen encerradas en una circunferencia. La posición del puntero está señalada por la flecha. En la figura se observa una primera transición de estados cuando se accesa la página 6 (residente) y una segunda transición al accesar la página 5 (nor esidente). A continuación se accesan las siguientes páginas de memoria:

```
4 7 5 3 4 1 2
```

Siguiendo el mismo esquema de la figura, muestre los estados de la memoria después de realizar cada uno de los accesos indicados.

Parte b: En un computador se dispone de 10 páginas reales para los procesos y 4 para el núcleo del sistema operativo. El sistema operativo utiliza la estrategia del working set para administrar las páginas. En él se ejecutan 3 procesos simultaneamente. El proceso A ocupa 5 páginas virtuales y su working set es el conjunto de páginas {0,2,3,4}, el proceso B ocupa 6 páginas y su w-s es {1,2,4,5} y el proceso C ocupa 4 páginas y su w/s es {0,1,3}. Haga un diagrama mostrando una posible asignación de la memoria y escriba el contenido de las tablas de páginas de c/u de los procesos. En las tablas usted debe indicar si la página virtual se encuentra residente o no y el número de página real en que se encuentra.

2. P2 Examen 96

El siguiente procedimiento ordena un arreglo de n enteros usando bubble sort.

|p2.c

Este procedimiento se ejecuta co nun arreglo de 1MByte (256K palabras enteras) en un computador que posee memoria real para almacenar sólo la mitad.

1. ¿Podría correr este procedimiento si el computador posee una arquitectura segmentada. Explique
2. El procedimiento se ejecuta en una arquitectura que usa páginas de 4KBytes (o 1024 enteros) con un sistema operativo que implementa la estrategia de reemplazo LRU (least recently used). Haga una estimacion del número de pagefaults que se producirá debido a páginas de datos.
3. Concluya sobre la utilidad de un sistema de memoria virtual en este caso.

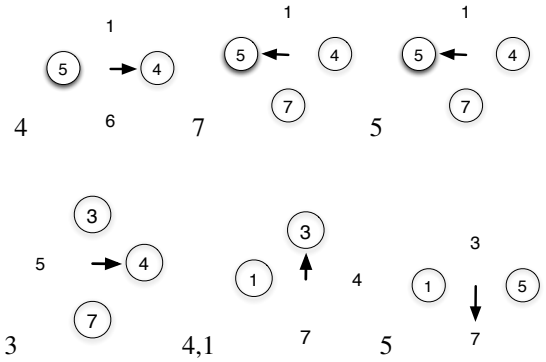
3. P2 Examen 97 (parte c)

Ud. tiene un dilema. Utilzar un hash con linear-probing o hashing-doble. Los análisis matemáticos indican que con linear-probing habrá que visitar en promedio 2.5 filas por cada búsqueda, mientras que con hashing doble se necesitará visitar 1.8 filas por cada búsqueda.

Elija un esquema. Haga supuestos razonables.

Solucion P1

Parte a:



Parte b:

En esta pregunta deben fijarse en tres cosas:

1. El *working-set* de cada proceso viene dado, no hay que calcularlo

2. No importa que "en total" hayan 14 páginas, las páginas del Sistema Operativo deben permanecer en memoria, así que sólo se utilizarán 10 páginas.

3. Si el sistema operativo no puede mantener el *working-set* de todos los procesos en memoria real, entonces se hace *swapping* de procesos (el proceso completo a disco).

Notar que sólo nos sirve el bit `V`, que nos indica si la página está en memoria o en disco. Dado que el WS de los tres procesos no cabe en memoria, supondremos que ejecutamos en estos momentos el proceso B y que el C fue llevado a disco:

Proceso A

	V	Real
0	1	0
1	0	
2	1	1
3	1	2
4	1	3

Proceso B

	V	Real
0	0	
1	1	4
2	1	5
3	0	
4	1	6
5	1	7

Proceso C

	V	Real
0	0	
1	0	
2	0	
3	0	

Solucion P2

1. No se puede, ya que con una arquitectura segmentada todo el segmento debe estar en memoria para correr, y en este caso el segmento de Datos o Pila tendria un tamaño mayor que la memoria real.

2. Considerando que solamente debe estar en memoria las paginas asociadas al arreglo, y despreciando paridad (par,impar):

$$n + (n-1) + (n-2) + ... + \frac{n}{2}$$

$$\frac{(n+1)\,n}{2} - \frac{(k+1)\,k}{2}, \quad k = \frac{n}{2}$$

$$\frac{(n+1)\,n}{2} - \frac{\left(\frac{n}{2}+1\right)\frac{n}{2}}{2}$$

$$\frac{\left(\frac{3n}{2}+1\right)n}{4}$$

Solución P3

De acuerdo a los apuntes del curso los tiempos de acceso a memoria son de $t_m = 60\,[ns]$ y $t_p = 12\,[ms]$. Luego,

$$t_e = (1-r)\cdot t_m + r \cdot t_p$$

Por lo que la diferencia entre un esquema y otro se da en la tasa r.

La idea es que en caso de colisiones, con linear probing, como sigues buscando secuencialmente, se cae en la misma pagina, disminuyendo así los page-faults.

Con hashing doble como vuelves a calcular un indice completamente distinto, es seguro que vas a caer en otra pagina, aumentando los page-faults.

Por lo tanto, si hay penuria de memoria, linear probing se comporta mejor.

El supuesto aca es que se trata de un arreglo en donde los elementos contienen directamente la informacion. Esto se puede hacer en C. En Java los arreglos contienen punteros, por lo que aun cuando se visite secuencialmente el arreglo, los elementos visitados pueden estar repartidos en la memoria, aumentando los page-faults. En general los lenguajes orientados a objetos (basados fuertemente en punteros) tienden a producir más page-faults.