3
GESTIÓN DE MEMORIA

Problema 3.1

Sea un sistema monousuario con memoria virtual, que planteamos de muy pequeñas dimensiones para no complicar el problema, que dispone de:

- *Una memoria principal de 8 páginas de 4KB*
- *Un SO cuya parte residente ocupa 4KB*
- *Una zona de swap con capacidad para 10 páginas.*
- *Una política de no preasignación de swap.*
- *Un disco con dos particiones una para ficheros y otra para swap.*

El SO operativo arranca un único proceso shell y el usuario solicita la ejecución de un proceso compuesto por:

- *Segmento de texto de 3,4KB*
- *Segmento de pila, con un tamaño inicial de 200B*
- *Segmento de datos de sólo lectura de 14KB*
- *Segmento de datos de escritura y lectura con un tamaño inicial de 6KB*

Suponer que el sistema de ficheros es de tipo UNIX y que la capacidad de datos útil que se consigue es de 160 KB.

Se pide:

a) Establecer la tabla de páginas del proceso de usuario. Justificar su estructura y proponer su contenido completo.
b) Indicar el contenido de la memoria principal y del swap al iniciar la ejecución del proceso.
c) Suponer que, en un instante determinado, el proceso tiene asignado una sola página para la pila, que la página está llena (la pila ocupa toda la página) y que se ejecuta una instrucción de PUSH. Exponer las acciones que se producen.

Problema 3.2

Un sistema tiene un disco de 128KB organizado en bloques e intercalado simple y una memoria principal de 44KB. La gestión de la memoria virtual se realiza con páginas y con tablas de páginas de dos niveles. Las direcciones virtuales son de 24bits, de los cuales 6bits sirven para acceder a la tabla de páginas de primer nivel y 6bits para acceder a la tabla de páginas de segundo nivel.

Además, en nuestro sistema existe una biblioteca dinámica lib.so que contiene la función func. Se tiene, además, el siguiente fragmento de programa.

```
char x[1024];
char y[1024];
...

main()
{
   int i, pid;

   /* A  Situación inicial */
   for (i=0; i<1024; i++)
      x[i]=y[i];
```

```
    func("datos");
    /* B  Después del for y de usar la biblioteca dinámica */
    ...
}
```

a) *Sabiendo que una vez compilado el programa el código ocupa 5K, los datos con valor inicial (d.v.i.) ocupan 6K y que los datos sin valor inicial (d.s.v.i.) tienen un tamaño de 2K, se pide* ***representar gráficamente*** *el fichero* ***ejecutable*** *resultante, suponiendo que utilizamos el formato ELF y que el tamaño de la cabecera del ejecutable es de 1KB.*

b) ***Representar gráficamente*** *el* ***mapa de memoria*** *del proceso en la situación inicial A.*

c) *Con el sistema de gestión de memoria propuesto inicialmente, indicar qué* ***tipo de fragmentación*** *se está generando.*

Estado inicial del disco

0	*1*	*2*	*3*
4 *lib.so texto*	*5* *lib.so texto*	*6* *lib.so datos*	*7* *lib.so datos*
8 *lib.so datos*	*9*	*10*	*11*
12	*13*	*14*	*15*
16 *SWAP*	*17* *SWAP*	*18* *SWAP*	*19* *SWAP*
20 *SWAP*	*21* *SWAP*	*22* *SWAP*	*23* *SWAP*
24 *SWAP*	*25* *SWAP*	*26* *SWAP*	*27* *SWAP*
28 *SWAP*	*29* *SWAP*	*30* *SWAP*	*31* *SWAP*

Problema 3.3

Partiendo de la situación descrita en el problema 3.2, resolver las siguientes cuestiones.

a) *Sabiendo que se tiene un sistema sin preasignación de swap, se pide* ***representar las tablas de páginas*** *tanto de primer como de segundo nivel en el instante A. Para cada Entrada de la Tabla de Páginas (ETP) se tendrá en cuenta la siguiente información: bit de validez, bit presente/ausente, bit modificado, bit referenciado, bit COW, bit Rellenar a Ceros (RC), bit de Rellenar de Fichero (RF), protección y número de marco o bloque de disco.*

b) *Responder* ***brevemente*** *a las siguientes preguntas: ¿qué ocurre si una misma página es expulsada varias veces al área de swap?, ¿quién y cuándo escribe el bit de modificado y quién y cuándo lo inicializa?*

Problema 3.4

Partiendo de la situación descrita en el problema 3.2, resolver las siguientes cuestiones.

a) *Indicar qué* ***cambios*** *se realizarían en el mapa de memoria del proceso y en la tabla de páginas al llegar el proceso al instante B. Suponer que la llamada a función* func *necesita solamente la primera página de la región de texto y la primera página de la región de datos de la biblioteca dinámica.*

b) *Explicar* ***brevemente*** *qué ocurre si el proceso crea un proceso hijo y éste escribe en la zona de datos de la biblioteca dinámica.*

Problema 3.5

Sea el código adjunto que se monta en modo dinámico para ser ejecutado en una máquina con memoria virtual y páginas de 2KB. Considere que el código del programa ocupa 3.573 B.

```
01 #include <stdio.h>   /* fichero ejemplo.c */
02 #include <stdlib.h>
03
04 int a;
05 int b = 50;
06
07 void funcion (int c)
08 {
09     int d;
10     static e = 2;
11     d = e
12     c = d + 5;
13     printf ("Esto lo imprime la funcion funcion()\n");
14 }
15
16 main ()
17 {
18     char *f;
19     f = (char *) malloc (5120);
20     funcion (b);
21     free (f);
22     exit (0);
23 }
```

a) *Haga un gráfico con las diferentes secciones del fichero ejecutable correspondiente, especificando en qué sección del mismo se almacena cada una de las variables del programa y por qué se almacena en dicha sección.*

b) *Haga un gráfico con las distintas regiones de memoria del proceso que ejecuta el programa anterior hasta la línea 16, indicando:*

- Dirección de comienzo y fin de cada región.
- Características de la región.
- Contenido de cada región, indicando expresamente las variables incluidas en cada una de ellas, así como su valor.

Como no se indican las necesidades de memoria de la librería dinámica libc utilizada por el programa, especifique los tamaños necesarios mediante nombres simbólicos.

Problema 3.6

Repetir el apartado b) del problema 3.5 cuando el programa ejecuta hasta las líneas 12 y 21.

Problema 3.7

Sea un sistema operativo que utiliza páginas de 4KB, tablas de páginas de 2 niveles y regiones de texto compartidas.

Existen simultáneamente dos procesos A y B que ejecutan el mismo programa y de los que sabemos que en un instante de tiempo determinado Ta tienen la siguiente situación:

Proceso	*Región*	*Nº páginas*	*Nº páginas presentes*	*Dirección comienzo*
A	*Texto*	*24*	*7*	*0*
	Datos	*13*	*5*	2^{23}
	Texto Biblioteca dinámica Z	*45*	*12*	2^{24}
	Datos Biblioteca dinámica Z	*5*	*3*	$2^{24}+2^{23}$
	Fichero M proyectado compartido	*27*	*7*	2^{25}
	Pila	*11*	*3*	2^{30}
B	*Texto*	*24*	*7*	*0*
	Datos	*15*	*5*	2^{23}
	Fichero M proyectado compartido	*27*	*7*	2^{24}
	Texto Biblioteca dinámica Z	*45*	*12*	2^{25}
	Datos Biblioteca dinámica Z	*5*	*3*	$2^{25}+2^{23}$
	Pila	*16*	*7*	2^{30}

a) Calcular el nº total de marcos de páginas que tienen asignados entre los dos procesos en ese instante.

b) Seguidamente, A ejecuta un bucle de lectura que recorre todo el fichero proyectado. Suponiendo que no se reemplaza ninguna página de los procesos A y B, indicar el nº de fallos de página que se producen así como el total de marcos de página que tienen ahora asignados entre los dos procesos.

Tomando como referencia el instante de tiempo Ta, Indicar si se produce un error de ejecución y, en su caso, el valor de la variable v en cada uno de los procesos en los supuestos c, d, e, y f siguientes. (NOTA: recuerde que la notación C 1<<x equivale a 2^x).

*c) El proceso A ejecuta: v = *p (donde p vale 24); e inmediatamente el proceso B ejecuta el mismo trozo de código, pero ahora p vale 25. En este caso, indicar, además, si se puede producir un fallo de página considerando que las variables v de cada proceso están cada una en su correspondiente marco de página.*

*d) El proceso A ejecuta: p = (1<<24) + 1; *p = 234; v = *p; e inmediatamente el proceso B ejecuta v = p.*

*e) El proceso A ejecuta: p = (1<<24) + (1<<23) + 12; *p = 234; e inmediatamente el proceso B ejecuta p = 1<<24) + (1<<23) + 12.*

*f) El proceso A ejecuta: p = (1<<25) + 10012; *p = 234; e inmediatamente el proceso B ejecuta p = (1<<24) + 10012; v = *p.*

Problema 3.8

Un sistema operativo con memoria virtual sigue el modelo clásico UNIX con una región de datos que engloba los datos estáticos y los dinámicos (heap). Además, asigna por defecto un heap de, al menos, 16KB y una pila de 32KB. El tamaño de página es de 4 KB.

Mediante el mandato size obtenemos los siguientes valores:

```
# size mi_programa
      text    data     bss       dec      hex      filename
     12096    1080    8724     21900     558C      mi_programa
# size mi_biblio
      text    data     bss       dec      hex      filename
    645024   14720   13424    658448    A0C10      mi_biblio
```

a) Dibujar el mapa de memoria del proceso que ejecuta mi_programa al comienzo de la ejecución, considerando que el montaje de las bibliotecas se hace al invocar una de sus funciones. Indicar para cada región el espacio que ocupa, los permisos de acceso y el contenido. Marcar los huecos existentes.

b) mi_programa incluye una línea que utiliza la función mi_funcion incluida en la biblioteca dinámica mi_biblio. Dibujar el mapa de memoria del proceso justo después de la llamada a mi_funcion, de igual forma que en el caso anterior.

Problema 3.9

Continuando con el entorno descrito en el problema 3.8, resolver las siguientes cuestiones:

a) mi_programa incluye las siguientes líneas de código:

```
p = mmap(NULL, 8192, PROT_READ | PROT_WRITE, MAP_SHARED, fd, 0);
q = p;
n = fork();
if (n == 0) {
   for (i = 0; i < 2; i++){*q = 1; q++;}
   m = *p;
   ......
}else{
   for (i = 0; i < 2; i++){*q = 3; q++;}
   r = *p;
   ......
}
```

Indicar el valor que tendrán las variables m y r después de que el padre y el hijo hayan ejecutado completamente las sentencias for.

*b) mi_programa tiene definida como global la variable int *p; Además tiene dos threads que ejecutan las líneas de código siguientes:*

Thread 1

```
p = mmap(NULL, 1024, PROT_READ | PROT_WRITE, MAP_PRIVATE, fd, 0);
for (i = 0; i < 2; i++){*p = i; p++;}
```

Thread 2

```
for (j = 0; j < 5; j++){*p = j+5; p++;}
```

Suponiendo que primero el thread 1 ejecuta completamente las dos líneas indicadas y luego el thread 2 completa la ejecución del for, indicar el contenido de la región generada con el mmap.

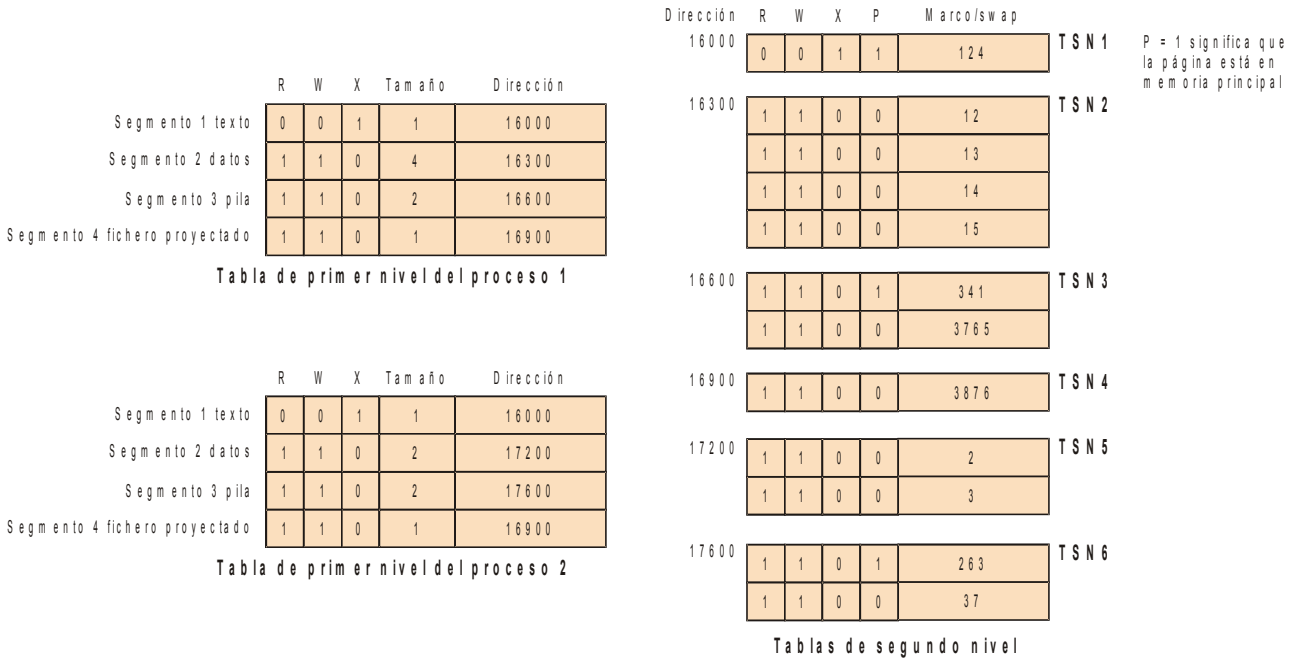
c) mi_programa realiza montaje explícito de la biblioteca mis_numeros, utilizando la función max, que se define en dicha biblioteca. Esta función recibe dos números enteros y devuelve otro número entero. Escriba un fragmento de código que utilice dicha función max.

Problema 3.10

En un sistema con páginas de 4KB y palabras de 32 bits se tienen las tablas de páginas de la figura.

a) *Compare las tablas de los dos procesos e indique las conclusiones a las que llegue.*

b) *Supongamos que el proceso 1 está ejecutando, que el tamaño ocupado de la pila es de 400 B y que realiza una llamada a un procedimiento que contiene exclusivamente la declaración de una matriz local de 100x10 enteros con valor inicial. Indicar los cambios que se producirían en las tablas de páginas al ejecutarse dicha llamada. ¿Qué ocurre si la matriz es de 100x100 enteros con valor inicial? ¿Y si es de 100x100 enteros sin valor inicial?*



Problema 3.11

Continuando con el supuesto planteado en el problema 3.10, resolver las siguientes cuestiones.

a) *Supóngase que el proceso 1, partiendo de la situación de la figura, ejecuta un bucle que recorre todo su segmento 4. Justo después de ello el proceso 2 ejecuta a su vez un bucle que recorre todo su segmento 4. ¿Es probable que el proceso 2 requiera el intercambio de alguna página? Justificar la respuesta.*

b) *Sea ahora una máquina con palabras de 64 bits y que requiere datos alineados. Los enteros ocupan 4 bytes mientras que los double ocupan 8 bytes. Calcular el espacio real ocupado por la siguiente estructura suponiendo que el compilador no realiza ninguna optimización.*

```
struct ditur {
   char b;
   double c;
   int v[5];
   double m;
   char n;
   char s;
   int r;
}
```

¿Qué optimización se podría realizar?

Problema 3.12

Para la traza formada por las páginas:

4, 2, 1, 5, 4, 2, 3, 4, 2, 3, 4, 2, 1, 5, 3

Calcular el funcionamiento de los algoritmos de reemplazamiento FIFO y LRU con 3 y 4 marcos de página, obteniendo el número de fallos de página producidos en cada uno de los cuatro casos.

Comentar los resultados obtenidos.

Problema 3.13

Se desea diseñar un gestor de memoria para un sistema con memoria virtual que soporta tablas de páginas multinivel. El modelo de memoria de proceso que ha de proporcionar este gestor ha de ser de múltiples segmentos de asignación libre y dinámica.

Se pide definir la ***secuencia ordenada*** *de acciones que debe realizar el gestor de memoria para cada uno de los casos indicados. Destacar las* ***llamadas*** *al sistema de ficheros, las acciones que se solicitan del núcleo, así como las* ***estructuras de información*** *necesarias en el gestor de memoria.*

a) El proceso A solicita un servicio EXEC sobre el fichero xxxx.

b) ¿Qué implicaciones tendría el uso de la técnica de ficheros proyectados en memoria en el caso del EXEC?