

SO_PED2_Oskar_Prieto_Arto

NOMBRE:	OSKAR PRIETO ARTO
DNI:	30.638.877-W
CENTRO :	BIZKAIA

1. Explique razonadamente si las siguientes afirmaciones son verdaderas o falsas:

I) La sobrepaginación aumenta el porcentaje de uso del procesador.

Falso.

Si se produce sobrepaginación quiere decir que constantemente se están produciendo fallos de página. Con lo cual constantemente hay que reemplazar páginas. Esto consume tiempo de procesador que no puede ser utilizado para atender procesos.

II) Se denomina buffering de página a la estrategia consistente en cargar un cierto número de páginas de un proceso antes de iniciar o continuar su ejecución.

Falso.

La estrategia consistente en cargar un cierto número de páginas de un proceso antes de iniciar o continuar su ejecución se denomina paginación por adelantado. El buffering de página consiste en reservar marcos de memoria principal para poder cargar rápidamente en ellos páginas cuando se produce un fallo de página. Estos marcos pueden ser marcos vacíos o pueden ser marcos ocupados que no requieren ser reescritos en memoria secundaria, con lo cual se pierde menos tiempo si se necesita hacer un reemplazo.

2. Un sistema con memoria virtual mediante demanda de páginas utiliza el algoritmo LRU para la sustitución de páginas. Un proceso genera la siguiente secuencia de referencias a páginas de memoria:

1 3 2 4 1 5 7 4 3 2 8 9 4 5 4 9 1 8 3 2

a) Determinar cuántos fallos de página se producen cuando se dispone de 4 o 5 marcos de página para este proceso.

El algoritmo LRU crea una lista de páginas referenciadas en orden de referencia. Cuando necesita cargar una página que no tiene en memoria, reemplaza la pagina que más tiempo haga que fue referenciada. Cuando tenemos cuatro marcos, se produce la siguiente secuencia:

	1	3	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3	2
	1	3	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3	2
		1	3	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3
			1	3	2	4	1	5	7	4	3	2	8	9	9	5	4	9	1	8
				1	3	2	4	1	5	7	4	3	2	8	8	8	5	4	9	1
	F	F	F	F	A	F	F	A	F	F	F	F	F	F	A	A	F	F	F	F

En el gráfico vemos que se han producido 16 fallos y 4 aciertos. Cuando tenemos 5 marcos, se produce la siguiente secuencia:

	1	3	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3	2
	1	3	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3	2
		1	3	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3
			1	3	2	4	1	5	7	4	3	2	8	9	9	5	4	9	1	8
				1	3	2	4	1	5	7	4	3	2	8	8	8	5	4	9	1
						3	2	2	1	5	7	4	3	2	2	2	8	5	4	9
	F	F	F	F	A	F	F	A	F	F	F	F	A	F	A	A	F	A	F	F

En el gráfico vemos que se han producido 14 fallos y 6 aciertos. Cuando tenemos 5 marcos, se produce la siguiente secuencia:

b) Explicar razonadamente si mejoraría la tasa de fallos de página si se aumentase el número de marcos de página a N, siendo $N > 5$.

Al aumentar el número de marcos asignados a un proceso, tenemos más páginas de dicho proceso cargadas en memoria, por lo cual la probabilidad de que la página necesitada se encuentre en memoria es mayor. Pero lo que aumenta es la probabilidad de acierto, lo cual no quiere decir que mejore la tasa de fallos. Poniéndonos en el peor de los casos, si la secuencia de petición de páginas hace que siempre se pida una página que no esté en memoria, la tasa de fallos sería del 100%, tanto con N como con N+1 marcos.

En el ejemplo anterior, la tasa de fallos ha sido del 80%. Aumentando un 25% el número de marcos, la tasa de fallos baja al 70%.

3. Explique razonadamente las funciones que realizan las capas de software de E/S del núcleo de un sistema operativo.

De forma general, el software del núcleo que gestiona las operaciones de E/S se divide en tres capas: Subsistema de E/S, Driver de dispositivos y manejador de interrupciones.

Cada capa tiene bien definida su interface para comunicarse con las capas adyacentes. Las funciones que realiza cada capa son:

SUBSISTEMA DE E/S

Este componente se encarga de todas las tareas de E/S que son comunes a todos los dispositivos de E/S e independiente de los mismos. Estas tareas son:

- Asignar y liberar dispositivos dedicados. Hay dispositivos que pueden ser accedidos por múltiples procesos simultáneamente, como los discos duros. Pero hay otros dispositivos, como las impresoras, que solo pueden ser accedidas por un proceso en cada momento. El subsistema de E/S se encarga de controlar este tema.
- Bloqueo de procesos en espera de una operación de E/S.
- Planificación de la E/S. Esto es necesario para asegurarse que un proceso no acapara el uso de un recurso y favorece que peticiones con prioridad alta puedan “saltarse” la cola.
- Invocación del driver apropiado. El subsistema de E/S no conoce el funcionamiento de los dispositivos. La capa de drivers es la encargada de esto, ofreciendo al subsistema de E/S una interface bien definida.
- Almacenamiento temporal de datos de E/S tanto en sentido procedimiento→dispositivo como en sentido dispositivo→procedimiento.
- Proporcionar tamaño uniforme de bloque a niveles superiores de software, adaptando el tamaño del bloque de datos del dispositivo y el tamaño del bloque de datos del proceso.
- Gestión de errores producidos durante el proceso de E/S.
- Proporcionar una interface uniforme a los drivers de los dispositivos.

DRIVERS DE DISPOSITIVOS DE E/S

Un driver de dispositivo contiene el código que permite al sistema operativo controlar dicho dispositivo. Un driver de dispositivo interactúa con el subsistema de E/S y con el controlador de E/S del dispositivo.

El driver recibe peticiones del subsistema de E/S a través de las funciones y estructuras de datos que le proporciona en su interface y se encarga de comunicárselo adecuadamente al dispositivo. Para el subsistema de E/S, un driver es una caja negra a la que se da órdenes y se reciben respuestas.

MANEJADOR DE INTERRUPCIONES

Las operaciones de E/S pueden ser muy lentas. Cuando se pide a un dispositivo que haga algo, el driver se queda esperando a que acabe, pero cediendo al sistema operativo el control, para no bloquear el procesador. Cuando el dispositivo acaba con lo que está haciendo, su controlador provoca una interrupción, que es gestionado por el manejador de interrupciones. De esta manera, el driver vuelve a tener control sobre el dispositivo y puede asignárselo a otro procedimiento.

4. En un computador con una capacidad de memoria principal de 64 kibipalabras se utiliza gestión de memoria mediante segmentación. La tabla de segmentos (todos los datos numéricos están en decimal) es la siguiente:

Nº de segmento	Base	Longitud
0	0	7230
1	16384	8191
2	32768	1024
3	8192	356
4	24576	4200

a) Supuesto que una dirección lógica tiene el mismo tamaño en bits que una dirección física y que consta de los campos [nº de segmento, desplazamiento], determinar el tamaño en bits de cada uno de estos campos.

El sistema tiene una memoria de 64 kibipalabras $\rightarrow 2^{16}$ palabras. A partir de aquí damos por supuesto que lo que direccionamos son palabras.

La dirección tendrá un tamaño de 16 bit, que hay que repartir entre nº de segmento y desplazamiento. El campo nº de segmento tendrá un tamaño S que se calcula con la siguiente desigualdad:

$$S = \min_x \{5 \leq 2^x\} = 3$$

Una vez que sabemos que S tiene un tamaño de 3 bits, y sabiendo que el tamaño total de la dirección son 16 bits, el tamaño del campo desplazamiento D será de 13 bits.

b) Determinar a qué direcciones físicas expresadas en decimal corresponden las siguientes direcciones lógicas expresadas en hexadecimal: i) $11AE_{16}$ ii) 6190_{16} .

$$11AE_{16} = 0001000110101110_2$$

0	0	0	1	0	0	0	1	1	0	1	0	1	1	1	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Segmento $000_2 \rightarrow 0 \rightarrow$ Base 0

Desplazamiento $1000110101110_2 \rightarrow$ desplazamiento 4526

Dirección física \rightarrow base + desplazamiento = $0 + 4526 = 4526$

$$6190_{16} = 0110000110010000_2$$

0	1	1	0	0	0	0	1	1	0	0	1	0	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Segmento $011_2 \rightarrow 3 \rightarrow$ Base 8192

Desplazamiento $0110000110010000_2 \rightarrow$ desplazamiento 800

Dirección física \rightarrow base + desplazamiento = $8192 + 800 = 8992$

5. La política de gestión de memoria de un cierto sistema es del tipo demanda de página. El tamaño de una página es de 1 KiB, el tamaño máximo de la memoria virtual es de 4 MiB y el tamaño de la memoria física es de 1 MiB. Se pide:

a) Determinar el tamaño de cada uno de los campos de una dirección virtual y de una dirección física.

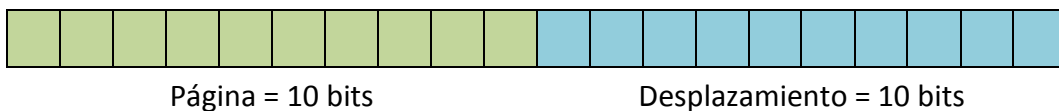
En el enunciado no se especifica, así que de aquí en adelante vamos a tomar como parte del enunciado que la longitud de una palabra es un byte y que vamos a direccionar palabras.

Memoria física $\rightarrow 1 \text{ MiB} \rightarrow 2^{20} \text{ bytes} \rightarrow$ la dirección tendrá una longitud de 20 bits

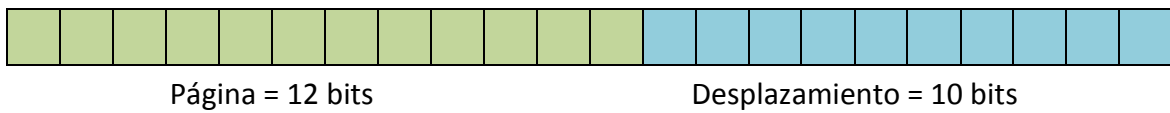
Memoria virtual $\rightarrow 4 \text{ MiB} \rightarrow 2^{22} \text{ bytes} \rightarrow$ la dirección tendrá una longitud de 22 bits.

Tamaño página $\rightarrow 1 \text{ KiB} \rightarrow 2^{10} \text{ bytes} \rightarrow$ necesitamos 10 bits para el desplazamiento.

DIRECCIONES FÍSICAS



DIRECCIONES VIRTUALES



b) Determinar la capacidad mínima que debe tener la tabla de páginas del proceso de mayor tamaño que se puede ejecutar en el sistema. ¿Qué tanto por ciento de la memoria principal ocuparía dicha tabla?

En la tabla de páginas tenemos que guardar para cada página, como mínimo, el marco en el que está en memoria, un bit de presente/ausente, un bit de referenciado, un bit de modificado y 3 bits para los permisos. Como tenemos 2^{10} marcos, también necesitamos 10 bits para referenciar el marco. En total necesitamos 16 bits.

Como el tamaño de la memoria virtual es de 4 MiB, el tamaño mayor que puede tener un programa será 4 MiB. Como la memoria virtual puede tener 2^{12} páginas (calculado en el apartado anterior), la tabla de páginas tendrá como máximo 2^{12} entradas de cómo mínimo 16 bits. Esto quiere decir que tendrá 2^{13} palabras.

Como la memoria principal tiene un tamaño de 2^{20} palabras y la tabla ocupa 2^{13} palabras, el porcentaje de memoria principal ocupada sería de:

$$\% \text{ Ocupación} = \frac{2^{13}}{2^{20}} = \frac{1}{2^7} = \frac{1}{128} = 0,0078125 \approx \mathbf{0,78\%}$$