

SISTEMAS OPERATIVOS - SEGUNDA PRUEBA DE EVALUACIÓN A DISTANCIA (PED2)

January 6, 2012

Octavio Martínez García
NIF 71280002-N
Centro Asociado Burgos

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

I) (1 p) La sobrepaginación aumenta el porcentaje de uso del procesador. La sobrepaginación es un fenómeno que se produce cuando el número de páginas de un proceso cargadas en memoria principal es menor al número de páginas que forman el conjunto de trabajo de un proceso, de manera que se producen demasiados fallos de página durante la ejecución del proceso, los cuales provocan que el SO tenga que estar constantemente copiando páginas de memoria secundaria a memoria principal, con la consiguiente sobrecarga que esto produce, lo cual a su vez reduce el porcentaje de utilización del procesador en favor de un aumento del porcentaje de uso de las operaciones de E/S necesarias para la escritura de las páginas en memoria principal.

De modo que la afirmación es falsa.

II) (1 p) Se denomina buffering de páginas a la estrategia consistente en cargar un cierto número de páginas de un proceso antes de iniciar o continuar su ejecución. La estrategia que se define en la anterior afirmación se refiere a la paginación por adelantado o prepaging, utilizada comunmente para evitar la serie de fallos de página iniciales que se producen cuando se ejecuta de nuevo un proceso, y hasta el momento en que este tiene cargado en memoria principal su conjunto de trabajo.

El buffering de páginas se refiere a una estrategia que utiliza la lista de marcos libres como si fuera una memoria caché de páginas, comprobando dicha lista

cuando se produce un fallo de página para ver si alguno de esos marcos señalados como libres pero que aún contienen una página cargada en ellos (la cual es reemplazable, por lo que el marco a todos los efectos es como si estuviera vacío), contiene la página buscada. para no tener que acceder a memoria secundaria a por ella.

2. (2 p) 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:

13241574328945491832

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

	1	3	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3	2
j=0	1	1	1	1	1	1	1	1	3	3	3	3	4	4	4	4	4	4	3	3
j=1		3	3	3	3	5	5	5	5	2	2	2	2	5	5	5	5	8	8	8
j=2			2	2	2	2	7	7	7	7	8	8	8	8	8	8	1	1	1	1
j=3				4	4	4	4	4	4	4	4	9	9	9	9	9	9	9	9	2
F/A	F	F	F	F	A	F	F	A	F	F	F	F	F	F	A	A	F	F	F	F
q4				4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3	2
q3			2	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3
q2		3	3	3	2	4	1	5	7	4	3	2	8	9	9	5	4	9	1	8
q1	1	1	1	1	3	2	4	1	5	7	4	3	2	8	8	8	5	4	9	1

Como se puede ver en la tabla adjunta, se producen un total de 4 aciertos y 16 fallos, es decir, un tasa de acierto y de fallo del 20% y 80% respectivamente si se dispone de 4 marcos de página.

Las variables q_i representan una estructura de datos de tipo pila modificada utilizada para representar el algoritmo de reemplazamiento LRU, en la cual, cuando se llena, el elemento de la base q₁ (usado menos recientemente) se elimina, y se produce la inserción de un nuevo elemento por la cima, q₄ (elemento usado más recientemente).

Con 5 marcos la tabla quedaría de la siguiente forma.

	1	3	2	4	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3	2
j=0	1	1	1	1	1	1	1	1	1	2	2	2	2	2	2	2	1	1	1	1
j=1		3	3	3	3	3	7	7	7	7	7	9	9	9	9	9	9	9	9	9
j=2			2	2	2	2	2	2	3	3	3	3	3	5	5	5	5	5	3	3
j=3				4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	2
j=4						5	5	5	5	5	8	8	8	8	8	8	8	8	8	8
F/A	F	F	F	F	A	F	F	A	F	F	F	F	A	F	A	A	F	A	F	F
q5						5	7	4	3	2	8	9	4	5	4	9	1	8	3	2
q4				4	1	1	5	7	4	3	2	8	9	4	5	4	9	1	8	3
q3			2	2	4	4	1	5	7	4	3	2	8	9	9	5	4	9	1	8
q2		3	3	3	2	2	4	1	5	7	4	3	2	8	8	8	5	4	9	1
q1	1	1	1	1	3	3	2	2	1	5	7	4	3	2	2	2	8	5	4	9

Como se puede observar de la tabla adjunta la tasa de aciertos aumenta al añadir un marco extra, pasando de 4 a 6 aciertos (se reducen los fallos de 16 a 14), que suponen una tasa de acierto del 30% y de fallo del 70%.

b) Explicar razonadamente si mejoraría la tasa de fallos de página si se aumentase el número de marcos de página a N, siendo $N > 5$. De las conclusiones obtenidas en el ejercicio anterior se deduce que si aumenta el número de marcos de página se debería reducir la tasa de fallos de página utilizando el algoritmo LRU, lo cual se sabe que es cierto para la mayoría de algoritmos de reemplazamiento, excepto aquellos como el FIFO que cumplen la anomalía de Belady, es decir, que aumenta su tasa de fallos conforme aumenta el número de marcos de página.

En general, si se aumenta el número de marcos se dispone de una “cache de páginas” de mayor tamaño, y tal como sucede con las auténticas memorias caché, la tasa de fallos disminuye conforme aumenta su tamaño (aunque empeora el rendimiento debido al mayor tiempo de búsqueda), debido a que hay mas oportunidades de que la página siguiente que se necesite se encuentre ya cargada en memoria.

3. (2 p) Explique razonadamente las funciones que realizan las capas de software de E/S del núcleo de un sistema operativo. De forma general el software de E/S del núcleo de un SO se pueden dividir en 3 capas, cada una de las cuales realiza unas funciones determinadas y se puede comunicar con las capas adyacentes mediante una interfaz bien definida:

1-Subsistema de E/S: Se encarga de gestionar la parte independiente del dispositivo de todas las operaciones de E/S, mediante la realización de las siguientes tareas:

- Asignación y liberación de dispositivos dedicados: determina si una petición de E/S de un proceso A debe ser aceptada o rechazada, o si debe poner al proceso en una cola de procesos bloqueados en espera de la utilización del dispositivo de E/S
- Bloqueo de procesos que solicitan una operación de E/S: dependerá de si el proceso invocó una llamada de tipo bloqueante, y del estado y tipo del dispositivo
- Planificación de E/S: teniendo como objetivo la disminución del tiempo de espera promedio para la finalización de una operación de E/S, y la distribución equitativa del uso de los recursos entre todos los procesos.
- Invocación del driver de dispositivo adecuado: se encarga de traducir la cadena de caracteres que utilizan los procesos para hacer referencia a un dispositivo determinado, en la dirección de inicio del driver correspondiente, usando unas tablas de correspondencia para ello.
- Almacenamiento temporal de datos de E/S o buffering:
- Proporcionar un tamaño de bloque uniforme a los niveles superiores de software para que puedan trabajar con tamaños de bloque lógico uniformes independientemente del tamaño de bloque físico
- Gestión de errores de E/S: ya sean errores de programación (debidos a un fallo de programación que hacen que un proceso solicite operaciones de E/S imposibles), o bien errores en el dispositivo, cuando estos realizan operaciones de E/S legales, producidos por causas técnicas (p.e. sector defectuoso en disco, etc)
- Proporcionar una interfaz para los drivers de los dispositivos: la cual debe funcionar permitiendo que el subsistema de E/S pueda solicitar algún servicio a un driver, e inversamente, que un driver pueda invocar rutinas del núcleo del SO.

2-Drivers de los dispositivos: Contiene el código que permite al subsistema de E/S controlar un determinado tipo de dispositivo de E/S. Las tareas que realiza se pueden resumir en:

- Comprueba que los parámetros de la función invocada por el subsistema de E/S son correctos y que dicha función se puede realizar
- Traduce los parámetros de dicha función a parámetros del dispositivo
- Comprueba si el dispositivo está ocupado atendiendo alguna petición de E/S anterior o si se encuentra preparado.

- Genera un conjunto de órdenes para el controlados del dispositivo las cuales son cargadas en los registros del controlador
- Una vez transmitidas las órdenes debe esperar a que el controlador las ejecute. Cuando la operación de E/S se complete el controlador activará una interrupción. El manejador o rutina de servicio de la interrupción despertará al driver si este se hubo bloqueado.
- Examina la cola de peticiones de E/S pendientes para atender la siguiente si existe.

3-Manejadores de interrupción:

- Son la parte del software de E/S que se encarga de despertar al driver, cuando un controlador de E/S active una interrupción, mediante el uso del mismo mecanismo que usara este para bloquearse.
- Además, en caso de que no se use DMA, también se puede encargar de transferir datos desde un registro del controlador a un buffer del espacio del núcleo.

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

Se pide:

a) (1 p) 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.

b) (1 p) Determinar a qué direcciones físicas expresadas en decimal corresponden las siguientes direcciones lógicas expresadas en hexadecimal: i) 11AE16, ii) 619016, a) Como tenemos 5 segmentos cargados en memoria principal, el campo s de número de segmento debe tener 3 bits, ya que con 2 bits solo podemos direccionar 2^2 segmentos, de modo que necesitamos 3 bits, con los que podemos direccionar hasta $2^3 = 8$ segmentos.

En cuanto al campo desplazamiento de una D_L variará según el tamaño del segmento que se considere en función de los bits necesarios para poder direccionar todas las palabras que abarca la longitud de dicho segmento.

En nuestro caso tendríamos:

Nº de segmento	Longitud	s ($\min_s\{N_s \leq 2^s\}$)	d ($\min_d\{S_s \leq 2^d\}$)	longitud $D_L = s + d$
0	7230	3	13	16
1	8191	3	13	16
2	1024	3	10	13
3	356	3	9	12
4	4200	3	13	16

También se podría haber considerado un único tamaño de D_L tomando el mayor valor del desplazamiento ($d = 13$, $D_L = 16$), o en su defecto, la capacidad de la memoria principal, con 64 Ki, para direccionar las cuales necesitaríamos una longitud de 16 bits ($2^{16} = 64Ki$), 3 para el número de segmento y 13 para el desplazamiento.

b) Si convertimos las siguientes direcciones a binario tenemos:

- $11AE16_{16} = 000 \parallel 1\ 0001\ 1010\ 1110\ 0001\ 0110$; esta dirección hace referencia al segmento 0 (3 msb), y a un desplazamiento de 1158678 (13 lsb) lo cual está fuera del rango posible de direcciones, por lo que no constituye una dirección física válida
- $619016_{16} = 011 \parallel 0\ 0001\ 1001\ 0000\ 0001\ 0110$; segmento 3 (011) y desplazamiento de 102422 (13 l.s.bits) palabras, lo cual vuelve a estar fuera del rango, por lo que produciría una excepción por violación del tamaño del segmento, no constituyendo una D_F válida.

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) (1 p) Determinar el tamaño de cada uno de los campos de una dirección virtual y de una dirección física.

b) (1 p) 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?

a) Considerando 1 palabra de 1 byte como unidad direccionable, para hallar la longitud de una dirección física D_F necesitamos:

La capacidad de la memoria principal es de $1MiB = 2^{20}B$, con lo que considerando la UD a la palabra de 1B, necesitamos 20 bits para direccionar todas la palabras de memoria.

El campo desplazamiento, por su parte, se obtiene a partir del tamaño de página establecido, $1KiB = 2^{10}B$, para el cual necesitamos 10 bits para poder acceder a todas las direcciones de marcos de página.

De este modo el campo número de marco de página de una D_F es de $20 - 10 = 10$ bits, quedando el formato de una dirección física de la siguiente forma

número de marco de página f = 10 bits	desplazamiento d = 10 bits
---------------------------------------	----------------------------

Por su parte una dirección virtual D_V tendría el mismo formato para el campo desplazamiento, 10 bits, necesarios para acceder a todas las UD de una página.

El tamaño del campo número de página se calcularía:

$\log_2 2^{22} = 22$ bits necesarios para direccionar todo el espacio de direcciones virtuales disponibles ($4MiB = 2^{22}B$), menos el tamaño del campo desplazamiento, 10 bits, nos da un campo de página de 12 bits, quedando la D_V :

número de página p = 12 bits	desplazamiento d = 10 bits
------------------------------	----------------------------

b) El proceso de mayor tamaño ejecutable ocuparía la totalidad del espacio de memoria virtual, es decir, 4MiB, de modo que su tabla de páginas deberá tener la siguiente capacidad:

Consideramos que los campos presentes en cada entrada de la tabla son los campos de número de marco de página (10 bits), referenciado (1 bit), modificado (1 bit) y validez (1 bit), de modo que cada entrada tendrá 13 bits.

Ahora si consideramos que la tabla debe tener una entrada por cada página del proceso, tenemos que:

$$2^{22}bytes / 2^{10}bytes / pág = 2^{12}páginas$$

de modo que la tabla tendrá $2^{12}entradas * 13bits/entrada = 53.248bits = 6.656bytes$

los cuales ocuparían $6.656bytes / 1MiB = 0,00634$

La tabla de páginas ocuparía un 0,63% de la capacidad total de memoria principal.