

Ordinaria2023-Resuelto.pdf



AMoreno03



Sistemas Operativos



2º Grado en Ingeniería Informática



**Escuela Técnica Superior de Ingenierías Informática y de Telecomunicación
Universidad de Granada**



[Accede al documento original](#)



**El momento villancico
en familia, no renta.**

webuy.com

CAMBIAR
tu Play 4 por una
Play 5 y jugar
con amigos, sí.



Convocatoria ordinaria enero 2023

Examen del tema 3 y 4

- 1) En un SO con una organización de memoria con paginación, siempre existe una tabla de páginas por cada proceso y con los mismos contenidos en cada entrada de dicha tabla independientemente de que dicho SO adicionalmente lleve a cabo o no gestión de memoria virtual. ¿Verdadero o falso?
- 2) ¿En qué consiste la hiperpaginación (thrashing) y por qué se produce? ¿Qué algoritmos de gestión de memoria son los más adecuados para mitigar la hiperpaginación y por qué?
- 3) En caso de que un bloque de datos no aparezca registrado en la lista de bloques libres ni en ningún archivo actual del sistema de archivos, entonces es más fácil encontrar a qué archivo debe pertenecer dicho bloque si el método de asignación de bloques a archivos es enlazado en lugar de indexado. ¿Verdadero o falso?
- 4) El tamaño del bloque lógico (o también conocido como cluster) tiene alto impacto tanto en rendimiento como en capacidad de almacenamiento especialmente en sistemas de archivos de tipo FAT 32. ¿Verdadero o falso?
- 5) Dada la siguiente cadena: 1, 2, 3, 4, 5, 2, 1, 2, 3, 4. Sabiendo que tenemos un sistema de paginación por demanda con 4 marcos en memoria principal. ¿Cuántos fallos de página se producen aplicando el algoritmo de planificación de reloj? Importante: en el resultado debe mostrar en cada momento los valores de los bits de referencia, la posición del puntero y donde se producen los fallos de página.
- 6) (2 puntos) Suponga un espacio virtual de direcciones de 16 bits (2 bits para la tabla de páginas de primer nivel, 3 bits para la tabla de páginas del segundo nivel y once bits para el desplazamiento). El mapa de memoria virtual de un proceso se muestra en la figura. Construya la tabla de páginas de dos niveles suponiendo que la primera página virtual del proceso está cargada en el marco 5 y el resto de sus páginas válidas se cargan de forma consecutiva tras ella en la memoria principal (no hay problema de espacio en memoria principal). Indique el total de páginas y cree una tabla de páginas con todos los campos que debe incluir. A continuación, realice la traducción de 7200 y 63550 como dirección virtual a direcciones en memoria principal.

Texto (Código)	6144 B
Datos	8192 B
40960 Bytes (sin usar)	
Pila	10240 B

- 7) Dado un sistema de archivos FFS o ext2, represente, explique y justifique cómo se lleva a cabo la localización del byte de datos situado en la posición 10MB de un archivo. Suponga que las direcciones de bloques de datos ocupan 4B, y el tamaño del bloque de datos es de 4KB.

- 1) En un SO con una organización de memoria con paginación, siempre existe una tabla de páginas por cada proceso y con los mismos contenidos en cada entrada de dicha tabla independientemente de que dicho SO adicionalmente lleve a cabo o no gestión de memoria virtual. ¿Verdadero o falso?

Falso, aunque es cierto que un 'SO' con paginación tiene una tabla de páginas por cada proceso, los contenidos de las entradas de tabla son muy distintos si se usa o no Memoria Virtual:

- Si no se usa Memoria Virtual el programa deberá estar cargado por completo en memoria RAM si quiere ejecutarse, esto trae consigo cambios en el contenido de la tabla, ya no necesitamos un bit de validación pues todo el programa está en memoria ni tampoco bits de estado para algoritmos de reemplazo.
- En cambio, si se usa Memoria Virtual, el espacio de direcciones lógico del proceso puede ser mayor que la memoria física disponible, lo que quiere decir que en memoria solo habrá una parte de las páginas del programa y las otras estarán en disco (área de swap).

Para gestionar esta la tabla de páginas necesita campos adicionales como:

- Bit de Validez: Indica si está o no cargada una página en memoria.
- Bit de Modificado: Indica si una página ha sido modificada desde que se cargó, esto es importante para los algoritmos de reemplazo de páginas.
- Bit de Referencia: Indica si la página se ha leído recientemente.

- 2) ¿En qué consiste la hiperpaginación (thrashing) y por qué se produce? ¿Qué algoritmos de gestión de memoria son los más adecuados para mitigar la hiperpaginación y por qué?

La hiperpaginación se produce cuando el sistema operativo dedica más tiempo a mover páginas entre la memoria principal y la secundaria que a ejecutar instrucciones reales de programa.

(Extra)

El rendimiento cae en picado y se observan una tasa de utilización del procesador muy baja mientras que un dispositivo de paginación saturado.

La hiperpaginación se produce cuando el nivel de multiprogramación es demasiado alto para la memoria física disponible. En ese estado los procesos no tienen suficientes marcos para mantener su 'conjunto de trabajo' en RAM y por tanto se generan muchas faltas de página.

Los algoritmos más adecuados son:

- WS (Working Set): Este algoritmo intenta mantener en M. Principal solo las páginas que el proceso ha referenciado en una ventana de tiempo reciente, asegurando que un proceso solo se ejecute si tiene su conjunto mínimo de páginas necesario.
- FFT: Utiliza intervalos de tiempo entre faltas de página consecutivas para ajustar el número de marcos asignados a un proceso. Por tanto, si un proceso tiene faltas de páginas muy frecuentes el algoritmo entiende que necesita más y se las asigna.

- 3) En caso de que un bloque de datos no aparezca registrado en la lista de bloques libres ni en ningún archivo actual del sistema de archivos, entonces es más fácil encontrar a qué archivo debe pertenecer dicho bloque si el método de asignación de bloques a archivos es enlazado en lugar de indexado. ¿Verdadero o falso?

Verdadero, para estos problemas de recuperación de datos la asignación enlazada es mucho más eficaz para encontrar el 'bloque huérfano', esto es así por como se estructura. Este método no solo contiene la información del archivo, sino también reserva un espacio para un puntero al siguiente bloque, al existir esta cadena de punteros el 'SO' tiene muy fácil recuperar el archivo. En cambio con la asignación indexada el proceso se complica mucho más puesto que se guardan bloques de datos puros, no tienen ninguna información de control.

- 4) El tamaño del bloque lógico (o también conocido como cluster) tiene alto impacto tanto en rendimiento como en capacidad de almacenamiento especialmente en sistemas de archivos de tipo FAT 32. ¿Verdadero o falso?

Verdadero, el tamaño de los clusters dentro de un sistema de archivos como FAT32 que están basados en tablas tiene un gran impacto.

• Si el tamaño de los clusters es muy grandes, aumentara significativamente la fragmentación interna, lo que es ineficiente si el sistema contiene muchos archivos pequeños.

• Si el tamaño de los clusters es muy pequeño la tabla de asignación, que tiene una entrada por cada clusters se hará muy grande. Una tabla FAT demasiado grande consume más espacio en disco y más memoria principal.

- 5) Dada la siguiente cadena: 1, 2, 3, 4, 5, 2, 1, 2, 3, 4. Sabiendo que tenemos un sistema de paginación por demanda con 4 marcos en memoria principal. ¿Cuántos fallos de página se producen aplicando el algoritmo de planificación de reloj? Importante: en el resultado debe mostrar en cada momento los valores de los bits de referencia, la posición del puntero y donde se producen los fallos de página.

	↓					↓					
M1	1*	1*	1*	1*	5*	5*	5*	5*	5*	4*	
M2	-	2*	2*	2*	2	2*	2*	2*	2*	2	
M3	-	-	3*	3*	3	3	1*	1*	1*	1	
M4	-	-	-	4*	4	4	4	4	3*	3	
Puntero	→M2	→M3	→M4	→M1	→M2	→M3	→M4	→M4	→M1	→M2	
Fallo	✓	✓	✓	✓	✓	✗	✓	✗	✓	✓	

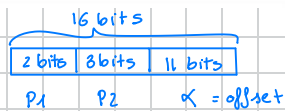
↓ = Hoy vuelta completa dejando el bit de referencia a 0

* = Bit de referencia a 1



- 6) (2 puntos) Suponga un espacio virtual de direcciones de 16 bits (2 bits para la tabla de páginas de primer nivel, 3 bits para la tabla de páginas del segundo nivel y once bits para el desplazamiento). El mapa de memoria virtual de un proceso se muestra en la figura. Construya la tabla de páginas de dos niveles suponiendo que la primera página virtual del proceso está cargada en el marco 5 y el resto de sus páginas válidas se cargan de forma consecutiva tras ella en la memoria principal (no hay problema de espacio en memoria principal). Indique el total de páginas y cree una tabla de páginas con todos los campos que debe incluir. A continuación, realice la traducción de 7200 y 63550 como dirección virtual a direcciones en memoria principal.

Texto (Código)	6144 B
Datos	8192 B
40960 Bytes (sin usar)	
Pila	10240 B



- Espacio de direcciones de 16 bits = $2^{16} = 65,536 \text{ bytes} = 64 \text{ KB}$
- Tamaño página = $2^{11} = 2048 \text{ bytes} = 2 \text{ KB}$
- Total páginas = $\frac{64 \text{ KB}}{2 \text{ KB}} = 32 \text{ páginas}$

Texto (Código)

$$\frac{6144 \text{ Bytes}}{2048 \text{ Bytes}} = 3 \text{ páginas}$$

Sin usar

$$\frac{40960 \text{ Bytes}}{2048 \text{ Bytes}} = 20 \text{ páginas}$$

Pila

$$\frac{10240 \text{ Bytes}}{2048 \text{ Bytes}} = 5 \text{ páginas}$$

Datos

$$\frac{8192 \text{ Bytes}}{2048 \text{ Bytes}} = 4 \text{ páginas}$$

Marco	Valida	Protección
0	5	V R
1	6	V R
2	7	V R
3	8	V RW
4	9	V RW
5	10	V RW
6	11	V RW
7		I

Texto (Código)

Datos

Marco	Valida	Protección
0	17	V RW
1		I
2		I
3	18	V RW

Marco	Valida	Protección
0		I
1		I
2		I
3	12	V RW
4	13	V RW
5	14	V RW
6	15	V RW
7	16	V RW

Pila

Marco	Valida	Protección
0		I
1		I
2		I
3		I
4		I
5		I
6		I
7		I

⑥ Direcciones a Traducir = 7200, 63550

D.L (7200)

$$\text{Primer Nivel} = \left\lfloor \frac{\text{D.L}}{\text{Direcciones nivel 1}} \right\rfloor = \frac{7200}{4 \cdot 2048} = 0$$

$$\text{Segundo Nivel} = \left\lfloor \frac{(\text{D.L mod } \overset{\text{nivel 2}}{\text{Direcciones}})}{\text{Direcciones} \times \text{Entrada}} \right\rfloor = \frac{7200 \text{ mod } 16384}{2048} = 3$$

2048 · 8

$$\text{Desplazamiento} = (\text{D.L mod Direcciones}) \text{ mod } \text{Direcciones} \times \text{Entrada} = (7200 \text{ mod } 8192) \text{ mod } 2048 = 1056$$

$$\text{D. Física} = (\text{Marco} \times \text{Tamaño Página}) + \text{Desplazamiento} = (8 \times 2048) + 1056 = \underline{17440}$$

D.L (63550)

$$\text{Primer Nivel} = \frac{63550}{8192} = 7 \text{ mod } 4 = 3$$

$$\text{Segundo Nivel} = \frac{(63550 \text{ mod } 16384)}{2048} = \frac{14398}{2048} = 7$$

$$\text{Desplazamiento} = (63550 \text{ mod } 8192) \text{ mod } 2048 = 62$$

$$\text{D. Física} = (16 \times 2048) + 62 = \underline{32830}$$

7) Dado un sistema de archivos FFS o ext2, represente, explique y justifique cómo se lleva a cabo la localización del byte de datos situado en la posición 10MB de un archivo. Suponga que las direcciones de bloques de datos ocupan 4B, y el tamaño del bloque de datos es de 4KB.

Posición del byte = 10 MB = 10.485.760 bytes

Tamaño bloque = 4 KB = 4.096 bytes

Direcciones = 4 bytes

$$\text{Punteros por bloque} = \frac{4096}{4} = 1.024 \text{ punteros}$$

• Vamos a buscar el bloque del archivo que buscamos

$$\left\lfloor \frac{10.485.760}{4096} \right\rfloor = 2.560$$

• Desplazamiento

$$(10.485.760 \text{ mod } 4096) = 0$$

• La estructura del i-nodo estándar en los sistemas ext2/FFS es:

- 12 punteros directos : Cubre de 0 a 11 1024 bloques
- 1 puntero Indirecto simple : Cubre de 12 a 1.025
- 1 puntero Indirecto doble : Cubre de 1.026 a 1.048.576 1024 × 1024 bloques

WUOLAH



**El momento villancico
en familia, no renta.**

CAMBIAR
tu Play 4 por una
Play 5 y jugar
con amigos, sí.



webuy.com



Entonces el bloque 2560 esta en el puntero doble porque $1036 \leq 2560 < 1.048.576$

Ahora tenemos que encontrar el índice relativo: $2560 - 12(\text{directos}) - 1024(\text{simple}) = 1.524$

Primer Nivel = $\left\lfloor \frac{1524}{1024} \right\rfloor = 1$ (Entrada 1 en la tabla)
 Recordar que se empieza en 0

Segundo Nivel = $1524 \bmod 1024 = 500$

• En la 500 se encuentra la D.Física del Bloque de datos

I-nodo

...
Ptr(Directos)
Ptr(Simple)
Ptr(Dobles)

Nivel 1

Entrada 0
Entrada 1
...

Nivel 2

Entrada 0
...
Entrada 600
...

Bloque de Datos

Byte 0
Byte 1
...