Opositando

Sistemas y Aplicaciones Informáticas

Volumen práctico

7 de noviembre de 2026

Síguenos en Instagram y entérate de nuestras últimas actualizaciones:

opositando.shop

Índice

1. Bloque 1: Análisis y resolución de problemas de arquitectura, hardware y representación digital . . . 1
   - 1.1 Ejercicio 1 . . . 1
   - 1.2 Ejercicio 2 . . . 7
   - 1.3 Ejercicio 3 . . . 14
   - 1.4 Ejercicio 4 . . . 24
   - 1.5 Ejercicio 5 . . . 28
   - 1.6 Ejercicio 6 . . . 36
   - 1.7 Ejercicio 7 . . . 45
   - 1.8 Ejercicio 8 . . . 52
   - 1.9 Ejercicio 9 . . . 58
   - 1.10 Ejercicio 10 . . . 66
2. Bloque 2: Diseño y manipulación de estructuras de datos, ficheros y algoritmos . 70
   - 2.1 Ejercicio 11 . . . 70
   - 2.2 Ejercicio 12 . . . 78
   - 2.3 Ejercicio 13 . . . 86
   - 2.4 Ejercicio 14 . . . 89
   - 2.5 Ejercicio 15 . . . 94
   - 2.6 Ejercicio 16 . . . 103
   - 2.7 Ejercicio 17 . . . 109
   - 2.8 Ejercicio 18 . . . 116
   - 2.9 Ejercicio 19 . . . 123
   - 2.10 Ejercicio 20 . . . 127
3. Bloque 3: Programación, implementación, depuración y verificación de programas 133
   - 3.1 Ejercicio 21 . . . 133
   - 3.2 Ejercicio 22 . . . 141
   - 3.3 Ejercicio 23 . . . 145
   - 3.4 Ejercicio 24 . . . 151
   - 3.5 Ejercicio 25 . . . 159
   - 3.6 Ejercicio 26 . . . 162
   - 3.7 Ejercicio 27 . . . 170
   - 3.8 Ejercicio 28 . . . 178
   - 3.9 Ejercicio 29 . . . 184
   - 3.10 Ejercicio 30 . . . 189

4. Bloque 4: Instalación, configuración y administración de sistemas operativos y equipos . . . 197
4.1 Ejercicio 31 . . . 197
4.2 Ejercicio 32 . . . 201
4.3 Ejercicio 33 . . . 207
4.4 Ejercicio 34 . . . 216
4.5 Ejercicio 35 . . . 222
4.6 Ejercicio 36 . . . 228
4.7 Ejercicio 37 . . . 236
4.8 Ejercicio 38 . . . 241
4.9 Ejercicio 39 . . . 247
4.10 Ejercicio 40 . . . 254
5. Bloque 5: Diseño, consulta y desarrollo de aplicaciones con bases de datos relacionales . . . 259
5.1 Ejercicio 41 . . . 259
5.2 Ejercicio 42 . . . 263
5.3 Ejercicio 43 . . . 271
5.4 Ejercicio 44 . . . 276
5.5 Ejercicio 45 . . . 282
5.6 Ejercicio 46 . . . 291
5.7 Ejercicio 47 . . . 295
5.8 Ejercicio 48 . . . 301
5.9 Ejercicio 49 . . . 310
5.10 Ejercicio 50 . . . 317
6. Bloque 6: Análisis, diseño, documentación y aseguramiento de la calidad de aplicaciones . . . 324
6.1 Ejercicio 51 . . . 324
6.2 Ejercicio 52 . . . 333
6.3 Ejercicio 53 . . . 337
6.4 Ejercicio 54 . . . 344
6.5 Ejercicio 55 . . . 353
6.6 Ejercicio 56 . . . 358
6.7 Ejercicio 57 . . . 365
6.8 Ejercicio 58 . . . 374
6.9 Ejercicio 59 . . . 382
6.10 Ejercicio 60 . . . 390
7. Bloque 7: Diseño e implementación de interfaces gráficas, aplicaciones visuales y multimedia . . . 401
7.1 Ejercicio 61 . . . 401
7.2 Ejercicio 62 . . . 405
7.3 Ejercicio 63 . . . 413
7.4 Ejercicio 64 . . . 422
7.5 Ejercicio 65 . . . 427

7.6 Ejercicio 66 . . . 435
7.7 Ejercicio 67 . . . 444
7.8 Ejercicio 68 . . . 450
7.9 Ejercicio 69 . . . 458
7.10 Ejercicio 70 . . . 465
8. Bloque 8: Instalación, explotación e integración de aplicaciones de propósito general y gestión . . . 471
8.1 Ejercicio 71 . . . 471
8.2 Ejercicio 72 . . . 479
8.3 Ejercicio 73 . . . 489
8.4 Ejercicio 74 . . . 495
8.5 Ejercicio 75 . . . 503
8.6 Ejercicio 76 . . . 514
8.7 Ejercicio 77 . . . 518
8.8 Ejercicio 78 . . . 526
8.9 Ejercicio 79 . . . 534
8.10 Ejercicio 80 . . . 541
9. Bloque 9: Diseño y análisis de arquitecturas, comunicaciones e interconexión de redes . . . 550
9.1 Ejercicio 81 . . . 550
9.2 Ejercicio 82 . . . 559
9.3 Ejercicio 83 . . . 564
9.4 Ejercicio 84 . . . 571
9.5 Ejercicio 85 . . . 581
9.6 Ejercicio 86 . . . 586
9.7 Ejercicio 87 . . . 595
9.8 Ejercicio 88 . . . 602
9.9 Ejercicio 89 . . . 607
9.10 Ejercicio 90 . . . 614
10. Bloque 10: Implantación, administración, seguridad y optimización de sistemas en red . . . 622
10.1 Ejercicio 91 . . . 622
10.2 Ejercicio 92 . . . 629
10.3 Ejercicio 93 . . . 638
10.4 Ejercicio 94 . . . 649
10.5 Ejercicio 95 . . . 653
10.6 Ejercicio 96 . . . 661
10.7 Ejercicio 97 . . . 671
10.8 Ejercicio 98 . . . 675
10.9 Ejercicio 99 . . . 683
10.10 Ejercicio 100 . . . 692

1. Bloque 1: Análisis y resolución de problemas de arquitectura, hardware y representación digital

1.1 Ejercicio 1

Un dispositivo utiliza palabras de 16 bits sin signo.

- Calcula el valor decimal del número binario 1011010010110110.
- Expresa dicho valor en hexadecimal.
- Indica el valor máximo que puede representarse con 16 bits sin signo.

Solución

Paso 1: Identificación y estrategia

El ejercicio trata sobre la **representación interna de números enteros sin signo**, contenido vinculado especialmente al tema de representación interna de los datos. Se proporciona una palabra binaria de 16 bits:

1011010010110110

Se pide obtener su equivalente decimal, expresarlo en hexadecimal y determinar el mayor valor posible en una palabra de 16 bits sin signo.

Al ser un número **sin signo**, todos los bits contribuyen positivamente al valor. El bit situado más a la izquierda no representa un signo: tiene peso 2^{15}, mientras que el bit situado más a la derecha tiene peso 2^0.

La estrategia es la siguiente:

- Para pasar de binario a decimal, se asocia a cada posición el peso de una potencia de 2 y se suman únicamente los pesos cuyas posiciones contienen un bit 1.
- Para pasar de binario a hexadecimal, se agrupan los bits de cuatro en cuatro desde la derecha, porque cada dígito hexadecimal equivale exactamente a 4 bits.
- Para calcular el máximo número sin signo representable con n bits, se emplea la fórmula $2^n - 1$.

Paso 2: Resolución razonada

Conversión de binario a decimal

Una palabra de 16 bits tiene posiciones numeradas, de izquierda a derecha, desde la posición 15 hasta la posición 0. Cada posición posee como peso una potencia de 2:

$$2^{15}, 2^{14}, 2^{13}, \ldots, 2^1, 2^0$$

El número dado es:

$$1011010010110110$$

Escribimos cada bit junto a su peso correspondiente:

- El primer bit es 1 y ocupa la posición 15: aporta $1 \cdot 2^{15}$.
- El segundo bit es 0 y ocupa la posición 14: aporta $0 \cdot 2^{14}$.
- El tercer bit es 1 y ocupa la posición 13: aporta $1 \cdot 2^{13}$.
- El cuarto bit es 1 y ocupa la posición 12: aporta $1 \cdot 2^{12}$.
- El quinto bit es 0 y ocupa la posición 11: aporta $0 \cdot 2^{11}$.
- El sexto bit es 1 y ocupa la posición 10: aporta $1 \cdot 2^{10}$.
- El séptimo bit es 0 y ocupa la posición 9: aporta $0 \cdot 2^9$.
- El octavo bit es 0 y ocupa la posición 8: aporta $0 \cdot 2^8$.
- El noveno bit es 1 y ocupa la posición 7: aporta $1 \cdot 2^7$.
- El décimo bit es 0 y ocupa la posición 6: aporta $0 \cdot 2^6$.
- El undécimo bit es 1 y ocupa la posición 5: aporta $1 \cdot 2^5$.
- El duodécimo bit es 1 y ocupa la posición 4: aporta $1 \cdot 2^4$.
- El decimotercer bit es 0 y ocupa la posición 3: aporta $0 \cdot 2^3$.
- El decimocuarto bit es 1 y ocupa la posición 2: aporta $1 \cdot 2^2$.
- El decimoquinto bit es 1 y ocupa la posición 1: aporta $1 \cdot 2^1$.
- El último bit es 0 y ocupa la posición 0: aporta $0 \cdot 2^0$.

Por tanto, la expresión completa es:

$$1{\cdot}2^{15}{+}0{\cdot}2^{14}{+}1{\cdot}2^{13}{+}1{\cdot}2^{12}{+}0{\cdot}2^{11}{+}1{\cdot}2^{10}{+}0{\cdot}2^9{+}0{\cdot}2^8{+}1{\cdot}2^7{+}0{\cdot}2^6{+}1{\cdot}2^5{+}1{\cdot}2^4{+}0{\cdot}2^3{+}1{\cdot}2^2{+}1{\cdot}2^1{+}0{\cdot}2^0$$

Se eliminan los términos cuyo bit es 0 y se calculan las potencias que sí intervienen:

$$2^{15} + 2^{13} + 2^{12} + 2^{10} + 2^7 + 2^5 + 2^4 + 2^2 + 2^1$$

$$32768 + 8192 + 4096 + 1024 + 128 + 32 + 16 + 4 + 2$$

Realizando las sumas sucesivas:

$$32768 + 8192 = 40960$$

$$40960 + 4096 = 45056$$

$$45056 + 1024 = 46080$$

$$46080 + 128 = 46208$$

$$46208 + 32 = 46240$$

$$46240 + 16 = 46256$$

$$46256 + 4 = 46260$$

$$46260 + 2 = 46262$$

En consecuencia, el valor decimal es **46262**.

Conversión de binario a hexadecimal

El sistema hexadecimal es de base 16. Como $16 = 2^4$, cada dígito hexadecimal se representa mediante un grupo de 4 bits. Por ello, para convertir directamente de binario a hexadecimal se divide la secuencia binaria en grupos de cuatro bits, comenzando siempre por la derecha.

En este caso ya hay 16 bits, que forman exactamente cuatro grupos:

$$1011\ 0100\ 1011\ 0110$$

Ahora se convierte cada grupo binario a su valor hexadecimal:

- El grupo 1011 equivale a $8 + 2 + 1 = 11$. En hexadecimal, el valor 11 se representa con la letra **B**.
- El grupo 0100 equivale a 4. En hexadecimal se escribe **4**.
- El grupo 1011 vuelve a equivaler a 11, es decir, **B**.
- El grupo 0110 equivale a $4 + 2 = 6$. En hexadecimal se escribe **6**.

Por tanto, el número hexadecimal es:

$$B4B6_{16}$$

También es habitual escribirlo como **0xB4B6**, donde el prefijo '0x' indica que la constante está expresada en hexadecimal. En una respuesta de examen conviene indicar explícitamente la base, por ejemplo: **B4B6 en base 16**.

Como comprobación, puede observarse que la primera cifra hexadecimal, B, representa los cuatro bits más significativos, 1011. Dado que este grupo comienza por 1, el número es superior a 2^{15}, es decir, superior a 32768, lo que resulta coherente con el valor decimal calculado, 46262.

Máximo valor con 16 bits sin signo

Con n bits sin signo pueden representarse 2^n combinaciones diferentes, desde la combinación formada por todos los bits a 0 hasta la combinación formada por todos los bits a 1.

El mínimo valor es:

$$0$$

El valor máximo se obtiene cuando los 16 bits valen 1:

$$1111111111111111$$

Su valor es la suma de todas las potencias de 2 desde 2^0 hasta 2^{15}:

$$2^{15} + 2^{14} + \cdots + 2^1 + 2^0$$

Esta suma equivale a:

$$2^{16} - 1$$

Por tanto:

$$2^{16} - 1 = 65536 - 1 = 65535$$

El **máximo valor representable con 16 bits sin signo es 65535**. En hexadecimal corresponde a:

$$FFFF_{16}$$

El rango completo de representación de una palabra de 16 bits sin signo es, por tanto, desde 0 hasta 65535, ambos inclusive.

Paso 3: Claves para el examen

El primer aspecto que debe comprobarse siempre es si el enunciado indica una representación **sin signo** o **con signo**. En este ejercicio es sin signo: el bit más significativo es parte del valor numérico y no debe interpretarse como indicador de negatividad. Confundirlo con una representación en complemento a dos llevaría a un resultado decimal incorrecto.

Para convertir de binario a decimal, conviene escribir las potencias de 2 de forma ordenada, desde 2^{n-1} hasta 2^0, y sumar solo las asociadas a bits 1. Este procedimiento es algo más largo que aplicar atajos mentales, pero es fácilmente comprobable y demuestra razonamiento ante el tribunal.

Al agrupar para pasar de binario a hexadecimal, debe hacerse desde la **derecha**. Si el número de bits no fuese múltiplo de cuatro, se añadirían ceros a la izquierda, nunca a la derecha. Añadir ceros a la derecha cambiaría el valor numérico, pues equivaldría a multiplicar por una potencia de 2.

Es importante recordar las equivalencias hexadecimales superiores a 9: 10 es A, 11 es B, 12 es C, 13 es D, 14 es E y 15 es F. En este ejercicio, el grupo 1011 vale 11 y por ello se expresa como B, no como el número decimal 11 escrito literalmente junto a los demás dígitos.

Para el máximo sin signo no debe escribirse 2^{16} como resultado final. 2^{16} es el número de valores distintos que pueden codificarse, pero los valores comienzan en 0; por eso el valor máximo es $2^{16} - 1$. Esta distinción es una fuente muy habitual de errores.

Una presentación clara en examen consiste en indicar primero la fórmula posicional binaria, desarrollar la suma de los pesos activos, efectuar el agrupamiento hexadecimal y terminar con la fórmula general del rango sin signo. Deben señalarse las bases cuando pueda existir ambigüedad: base 2, base 10 o base 16.

Paso 4: Qué debes aprender

Debes dominar que un número binario sin signo de n bits se interpreta mediante potencias de 2, asignando al bit situado en la posición i el peso 2^i. El bit de menor peso es el de la derecha y corresponde a 2^0; el de mayor peso es el de la izquierda y corresponde a 2^{n-1}.

Para convertir binario a decimal, hay que sumar los pesos correspondientes a todos los bits cuyo valor sea 1. Para convertir binario a hexadecimal, hay que dividir la cadena binaria en grupos de cuatro bits desde la derecha y transformar cada grupo en un dígito hexadecimal.

También debes retener la regla general de capacidad para representaciones sin signo: con n bits se codifican 2^n valores y el rango es:

$$0 \leq x \leq 2^n - 1$$

En particular, para 16 bits sin signo, el rango es de 0 a 65535. Estas reglas permiten resolver con seguridad ejercicios equivalentes aunque cambien el número binario, el número de bits o la base de salida solicitada.

1.2 Ejercicio 2

Un ordenador dispone de una CPU con una caché de primer nivel, memoria RAM y almacenamiento SSD. Durante la ejecución de una aplicación de edición de vídeo se observa que la CPU mantiene una utilización moderada, pero el sistema presenta pausas frecuentes al cargar recursos. El administrador comprueba que la memoria RAM disponible es escasa y que el sistema realiza numerosas operaciones de paginación sobre el SSD. La aplicación accede repetidamente a bloques de datos de vídeo de gran tamaño, mientras otros procesos mantienen abiertos navegadores, herramientas de comunicación y servicios en segundo plano. Se pretende determinar qué componente limita principalmente el rendimiento y qué mejoras serían técnicamente razonables.

Explica la relación entre jerarquía de memoria, paginación y rendimiento en este caso, identificando el cuello de botella principal y justificando dos medidas de mejora priorizadas.

Solución

Paso 1: Identificación y estrategia

El ejercicio pide diagnosticar un problema de rendimiento en un ordenador y proponer mejoras justificadas. No basta con afirmar que «el equipo va lento»: hay que relacionar los síntomas observados con la **jerarquía de memoria**, la **memoria virtual por paginación** y el comportamiento de una aplicación que trabaja con datos voluminosos.

Intervienen principalmente los contenidos relativos a memoria interna, almacenamiento externo, gestión de memoria por el sistema operativo y evaluación del rendimiento. Deben distinguirse adecuadamente la caché de primer nivel, la RAM y el SSD, así como sus funciones y sus diferencias de capacidad, latencia y velocidad de transferencia.

La estrategia de resolución consiste en interpretar cada dato del enunciado:

- La CPU presenta una utilización moderada, por lo que no parece ser el recurso que está saturado ni el límite inmediato del sistema.
- La RAM disponible es escasa y existen numerosas operaciones de paginación: ambos hechos apuntan a presión de memoria.
- La aplicación maneja bloques grandes de vídeo, que requieren una cantidad elevada de memoria de trabajo y pueden tener poca reutilización efectiva en caché.
- Otros procesos consumen simultáneamente RAM, reduciendo el espacio disponible para la aplicación principal.
- La paginación se realiza sobre SSD, que es mucho más rápido que un disco mecánico, pero sigue siendo enormemente más lento que la RAM.

A partir de ello se identificará el cuello de botella y se priorizarán dos medidas: aumentar la RAM y reducir la presión de memoria mediante la gestión de procesos y de la carga de trabajo.

También se explicará por qué una mejora del SSD, aunque pueda ser complementaria, no sería la primera actuación si el origen del problema es la insuficiencia de memoria principal.

Paso 2: Resolución razonada

Jerarquía de memoria y criterio de rendimiento

La **jerarquía de memoria** organiza los medios de almacenamiento según su proximidad a la CPU, velocidad, capacidad y coste por bit. En términos generales, la caché de primer nivel está integrada o muy próxima al núcleo de procesamiento; la RAM constituye la memoria principal; y el SSD forma parte del almacenamiento secundario o externo.

La caché L1 es muy pequeña y extremadamente rápida. Guarda instrucciones y datos a los que la CPU accede de forma reciente o frecuente. La RAM tiene mucha más capacidad que la caché, pero una latencia superior. El SSD proporciona una capacidad aún mayor y conserva los datos sin alimentación eléctrica, pero no puede sustituir a la RAM como memoria de ejecución ordinaria debido a su latencia mucho mayor.

Para expresar la idea de rendimiento puede utilizarse el tiempo medio de acceso a memoria. En una simplificación que considere caché y RAM:

$$T_{medio} = T_{L1} + m_{L1} \cdot T_{RAM}$$

En esta expresión, T_{L1} es el tiempo de consulta de la caché de primer nivel, m_{L1} es su tasa de fallos y T_{RAM} representa la penalización asociada a tener que obtener el dato desde RAM tras un fallo en L1. En un sistema real hay más niveles de caché y otros factores, pero la idea esencial es que un fallo en un nivel obliga a consultar un nivel inferior y más lento.

Si el dato que necesita un proceso no está en RAM, se produce un **fallo de página**. El sistema operativo debe localizar esa página en el área de intercambio o en el fichero de paginación ubicado en el SSD, cargarla en RAM y, si no existe un marco libre, desalojar otra página. El tiempo efectivo pasa entonces a estar condicionado por el almacenamiento secundario:

$$T_{efectivo} = (1 - p) \cdot T_{RAM} + p \cdot T_{pagina}$$

Aquí, p es la probabilidad o tasa de fallos de página que requieren acceso al almacenamiento y T_{pagina} es el tiempo de servicio de ese fallo. Este último incluye la latencia de entrada/salida, la transferencia del bloque, la intervención del sistema operativo, la actualización de tablas de páginas y la posible escritura previa de una página modificada que deba ser desalojada.

Aunque p sea aparentemente pequeño, T_{pagina} es varios órdenes de magnitud mayor que T_{RAM}. Por ello, una frecuencia apreciable de fallos de página puede elevar radicalmente el tiempo de respuesta percibido por el usuario. Este efecto explica las pausas al cargar recursos.

Interpretación de los síntomas del caso

La aplicación de edición de vídeo trabaja con bloques de datos de gran tamaño. Según el formato, la resolución, la profundidad de color, el número de pistas, los efectos aplicados y los fotogramas almacenados para previsualización, puede consumir una cantidad importante de memoria. Además, el software puede mantener en RAM fotogramas, búferes, formas de onda, cachés de medios, datos de efectos y estructuras internas de edición.

El enunciado indica que hay poca RAM disponible. Esto significa que los conjuntos de trabajo de los procesos activos, es decir, las páginas que necesitan utilizar con cierta frecuencia, no caben simultáneamente en la memoria principal disponible. El editor de vídeo compite por la RAM con navegadores, aplicaciones de comunicación y servicios en segundo plano.

El sistema operativo usa memoria virtual para proporcionar a cada proceso un espacio de direcciones amplio y aislado. Cuando la RAM es insuficiente, debe elegir páginas para desalojarlas temporalmente al espacio de intercambio. Si una aplicación vuelve a necesitar una página desalojada, se genera un fallo de página y se recupera desde el SSD.

El problema se agrava porque la edición de vídeo suele procesar datos grandes y puede recorrer numerosos bloques. Si se cargan bloques de vídeo, después otros procesos requieren sus propias páginas, y posteriormente el editor vuelve a necesitar los bloques anteriores, pueden producirse desalojos y recargas continuadas. Cuando el sistema dedica una proporción relevante de tiempo a intercambiar páginas entre RAM y SSD, se habla de **hiperpaginación** o *thrashing*.

La hiperpaginación no exige necesariamente que la CPU esté al cien por cien. De hecho, el síntoma típico es una CPU con actividad moderada o incluso baja, mientras los procesos permanecen bloqueados esperando operaciones de entrada/salida. La CPU no está limitada por su capacidad de cálculo: queda infrautilizada porque debe esperar a que lleguen las páginas requeridas desde el SSD.

Por tanto, el diagnóstico es el siguiente: el **cuello de botella principal es la insuficiencia de memoria RAM disponible**, que provoca paginación intensa hacia el SSD. El SSD interviene en la manifestación del problema, ya que atiende las operaciones de intercambio, pero no es la causa primera. Incluso un SSD rápido es demasiado lento frente a la RAM para sostener de forma continuada el acceso a páginas de ejecución.

Papel de la caché de primer nivel

La presencia de caché L1 no evita el problema descrito. La caché L1 está diseñada para atender datos e instrucciones de uso muy inmediato por parte de la CPU y tiene una capacidad muy reducida. Los bloques de vídeo son grandes en relación con esa capacidad; además, si el dato no está siquiera en la RAM residente, no puede estar disponible de forma útil en L1.

Una vez que una página se ha recuperado desde el SSD a la RAM, el procesador podrá ir llevando a sus cachés las porciones de datos que utilice. Sin embargo, esto ocurre después de haber pagado la enorme penalización del fallo de página. Por eso sería erróneo concluir que el problema se soluciona aumentando la caché de primer nivel o cambiando la CPU, especialmente cuando el enunciado afirma que la utilización de CPU es moderada y destaca la paginación.

También conviene comprender que el tamaño de los recursos de vídeo reduce la eficacia relativa de la caché. El procesamiento secuencial de grandes volúmenes puede expulsar rápidamente datos anteriores de una caché pequeña. Pero este hecho afecta a la optimización fina del procesamiento; no altera el diagnóstico principal, que se sitúa entre la RAM insuficiente y el uso reiterado del almacenamiento como respaldo de memoria virtual.

Primera medida priorizada: ampliar la memoria RAM

La primera medida debe ser **instalar más memoria RAM**, siempre que la placa base, el sistema operativo y la arquitectura del equipo lo admitan. Es la actuación prioritaria porque ataca directamente la causa: permite mantener residentes simultáneamente un mayor número de páginas del editor de vídeo, del sistema operativo y de los procesos auxiliares.

Con más RAM disminuye la tasa de fallos de página, es decir, disminuye p en la expresión:

$$T_{efectivo} = (1 - p) \cdot T_{RAM} + p \cdot T_{pagina}$$

Como T_{pagina} es muy superior a T_{RAM}, una reducción de p produce una mejora notable del tiempo efectivo, incluso si la reducción no parece muy grande en términos porcentuales. El objetivo no es usar toda la RAM libre permanentemente, pues los sistemas operativos aprovechan la memoria no usada para caché de archivos; el objetivo es evitar que los conjuntos de trabajo activos compitan de forma continua y obliguen a paginar intensivamente.

Antes de ampliar, el administrador debe comprobar la cantidad de RAM instalada, los módulos y ranuras disponibles, la memoria máxima soportada por placa base y procesador, la compatibilidad de generación y frecuencia, y si el sistema operativo es de 64 bits. Asimismo, debe analizar el consumo real durante una sesión de edición representativa, pues la capacidad necesaria depende de la resolución de los proyectos y de las aplicaciones concurrentes.

La ampliación de RAM es técnicamente razonable porque proporciona una mejora estructural: no solo acelera una operación concreta, sino que reduce la dependencia del SSD para el intercambio y mejora la capacidad de multitarea general.

Segunda medida priorizada: reducir la presión de memoria

La segunda medida debe consistir en **reducir el consumo concurrente de memoria**, cerrando o limitando los procesos no esenciales durante la edición de vídeo. En el caso expuesto, deben revisarse los navegadores, especialmente si mantienen muchas pestañas, herramientas de comunicación, aplicaciones residentes y servicios en segundo plano no necesarios para la tarea.

Esta actuación libera memoria para el conjunto de trabajo del editor y reduce la competencia por marcos de página. Puede aplicarse de forma inmediata, incluso antes de ampliar el hardware, y sigue siendo recomendable después de la ampliación, especialmente al trabajar con proyectos exigentes.

La medida debe realizarse con criterio. No se deben finalizar indiscriminadamente servicios críticos del sistema operativo, componentes de seguridad o procesos cuya función se desconozca.

Procede identificar el consumo mediante las herramientas de monitorización del sistema: administrador de tareas, monitor de recursos, visor de actividad u otras equivalentes. Deben observarse el uso de RAM por proceso, la memoria comprometida, los fallos de página, la actividad de lectura y escritura del SSD y la evolución de estos valores durante las pausas.

También pueden configurarse adecuadamente las aplicaciones. Por ejemplo, se puede reducir la cantidad de medios precargados, limitar la memoria asignada a procesos auxiliares, utilizar archivos proxy o formatos de previsualización menos exigentes y evitar la ejecución simultánea de tareas que demanden mucha memoria. Estas acciones no sustituyen necesariamente a la ampliación de RAM, pero disminuyen el tamaño del conjunto de trabajo y, por tanto, la probabilidad de paginación.

Por qué no se prioriza inicialmente el cambio de CPU o SSD

Sustituir la CPU no es la medida prioritaria. La utilización moderada indica que no existe saturación sostenida de cálculo. Una CPU más potente podría acelerar la codificación, los efectos o el renderizado cuando esos procesos sean realmente dependientes de cómputo, pero no elimina las esperas de entrada/salida causadas por fallos de página. Durante una pausa por paginación, una CPU más rápida puede seguir esperando a que se carguen las páginas necesarias.

Tampoco sería prioritario cambiar el SSD si ya es funcional y razonablemente moderno. Un SSD más rápido puede disminuir parcialmente T_{pagina}, por lo que puede amortiguar las consecuencias de la paginación. Sin embargo, no elimina los fallos de página ni evita el intercambio continuo. La jerarquía correcta de intervención es reducir primero p, aumentando la RAM y reduciendo la presión de memoria; solo después tendría sentido valorar un SSD más rápido si persiste una carga elevada de entrada/salida o si el dispositivo actual presenta un rendimiento insuficiente.

Además, el área de intercambio debe mantenerse en un SSD con espacio libre suficiente y en buen estado, pero no conviene presentar el aumento del archivo de paginación como solución al rendimiento. Ampliarlo puede prevenir errores por falta de memoria comprometida, pero puede empeorar la experiencia si permite que el sistema dependa aún más del SSD en lugar de disponer de RAM suficiente.

Paso 3: Claves para el examen

El error más frecuente es confundir almacenamiento secundario rápido con memoria principal. Un SSD es considerablemente más veloz que un disco duro mecánico, pero sigue siendo muy lento respecto de la RAM en acceso aleatorio y en latencia. Por ello, que el intercambio se realice sobre SSD no convierte la paginación intensa en una situación normal ni deseable.

También es habitual interpretar erróneamente la CPU moderada como ausencia de problema. En este supuesto es una pista diagnóstica relevante: la CPU no está ocupada porque los procesos pueden estar bloqueados esperando entrada/salida. Debe explicarse que una baja o moderada utilización de CPU puede coexistir con una grave degradación de la respuesta del sistema.

Otro error consiste en atribuir automáticamente el problema a la caché L1. La caché mejora los accesos repetidos a escala muy pequeña y muy rápida, pero no resuelve que el dato haya

sido expulsado de RAM y deba recuperarse desde almacenamiento secundario. Es importante distinguir los niveles de memoria y no presentar la caché como una alternativa a ampliar la RAM.

En la respuesta conviene respetar un orden lógico. Primero se describen los síntomas; después se relacionan con la gestión de memoria virtual; a continuación se identifica el cuello de botella; finalmente se proponen medidas priorizadas y se justifica por qué actúan sobre la causa y no solo sobre un efecto.

Una formulación sólida puede incluir estas ideas, integradas en párrafos y no como afirmaciones aisladas:

- La RAM escasa ocasiona expulsión de páginas pertenecientes a conjuntos de trabajo activos.
- Los accesos posteriores a páginas expulsadas causan fallos de página y operaciones de entrada/salida sobre el SSD.
- La diferencia de rendimiento entre RAM y SSD explica las pausas percibidas.
- La actividad de intercambio puede mantener a la CPU esperando, de modo que la utilización moderada no descarta un cuello de botella grave.
- La ampliación de RAM es prioritaria porque disminuye la tasa de fallos de página.
- El cierre o limitación de procesos no esenciales es la segunda medida porque reduce inmediatamente la competencia por la memoria principal.

Si el enunciado aportara datos cuantitativos, deben utilizarse con prudencia. Por ejemplo, si facilitase la tasa de fallos de página y los tiempos de acceso, podría calcularse el tiempo efectivo mediante la fórmula indicada. En ausencia de valores numéricos, no deben inventarse porcentajes, velocidades o tamaños de memoria; basta con exponer la relación cualitativa y, si procede, la fórmula general.

Paso 4: Qué debes aprender

Debes dominar la **jerarquía de memoria**: cachés, RAM y almacenamiento secundario tienen funciones distintas. Cuanto más próximo está un nivel a la CPU, normalmente menor es su capacidad y mayor su velocidad. La caché acelera accesos locales y repetidos; la RAM sostiene la ejecución de procesos; el SSD almacena datos persistentes y puede servir de respaldo a la memoria virtual, pero no ofrece el rendimiento de la memoria principal.

Debes comprender que la **paginación** permite ejecutar procesos cuyo espacio de direcciones excede la RAM física, trasladando páginas entre RAM y almacenamiento secundario. Es un mecanismo necesario de memoria virtual, pero su uso intenso reduce drásticamente el rendimiento. La situación crítica aparece cuando los conjuntos de trabajo activos no caben en RAM y las páginas son expulsadas y recuperadas repetidamente.

Para diagnosticar ejercicios similares, asocia los síntomas a los recursos:

- CPU muy alta y poca actividad de entrada/salida puede indicar un límite de cómputo.
- RAM saturada, fallos de página frecuentes y actividad intensa de disco o SSD suele indicar presión de memoria o hiperpaginación.
- Alta actividad de almacenamiento sin paginación relevante puede deberse a lectura o escritura propia de la aplicación, copias, bases de datos o procesamiento de archivos.
- Pausas con CPU no saturada exigen analizar si los procesos esperan memoria, entrada/salida, red u otros recursos.

Finalmente, aprende a priorizar medidas según la causa. Si la paginación es consecuencia de RAM insuficiente, la respuesta principal es aumentar la RAM; la respuesta complementaria inmediata es reducir el consumo de memoria de procesos y cargas no esenciales. La mejora del SSD o de la CPU puede ser útil en otros escenarios, pero debe justificarse con los indicadores disponibles y no proponerse de forma automática.

Opositando

Fin de la muestra

Has visto los dos primeros ejercicios de esta relación.

La relación completa, con todos los ejercicios resueltos paso a paso, está disponible en **opositando.shop**.

Síguenos en Instagram y entérate de nuestras últimas actualizaciones:

opositando.shop