

para leer en la entrada i de la tabla de páginas el marco de página j donde se aloja la página i . Para acelerar la traducción de direcciones se puede utilizar un banco de registros o un TLB.

Preguntas de autoevaluación tema 7

7.1. ¿Qué ventajas e inconvenientes presenta el uso de memoria virtual?

Permite aumentar el grado de multiprogramación del sistema, ya que en memoria principal caben más procesos si solo hay algunas partes de cada proceso que si sus espacios de direcciones virtuales estuviesen cargados por completo. Además permite ejecutar procesos con un espacio de direcciones virtuales de tamaño superior al tamaño de la memoria física disponible para procesos.

Otra ventaja es que reduce el número de operaciones de E/S que son necesarias para intercambiar un proceso bloqueado desde memoria principal a secundaria o viceversa.

Se debe diseñar con cuidado ya que de lo contrario el rendimiento del sistema puede verse seriamente afectado. Si un proceso está causando continuamente excepciones porque requiere cargar en memoria partes del proceso que no están cargadas, su ejecución se ralentiza enormemente y además estará provocando una fuerte sobrecarga al sistema que tendrá que ser atendido por continuas excepciones.

7.2. ¿Qué requisitos debe reunir el hardware de un computador para poder implementar la memoria virtual?

Que soporte el reinicio de las instrucciones de su repertorio. Además cuando se implementa la mv también se suele utilizar un componente hardware llamado unidad de gestión de memoria (MMU) que se encarga de realizar la traducción de una dirección virtual en una dirección física. Una MMU puede implementarse en un circuito integrado independiente o dentro del circuito integrado del procesador. Consta de registros, sumadores y comparadores. También puede tener un banco de registro o un TLB.

7.3. Describir brevemente el funcionamiento de la técnica de paginación por demanda.

Al igual que en la técnica de paginación simple divide el espacio de la memoria principal en bloques de igual tamaño

denominados marcos de página. Además divide el espacio de un proceso en bloques del mismo tamaño denominado páginas. En un determinado instante de tiempo un marco de página j puede estar libre u ocupado por una página i de un proceso X .

En la paginación por demanda, a diferencia de la simple, para que un proceso pueda ejecutar no es necesario que todas las páginas del mismo estén cargadas en memoria principal, únicamente se cargan las páginas que se van referenciando durante la ejecución de un proceso. Cuando se referencia a una página que no está cargada en la memoria principal el hardware produce una excepción denominada fallo de página. El SO se encarga de atender a los fallos de página. Básicamente el tratamiento de un fallo de página requiere que se busque un marco libre o se elija uno ocupado para ser reemplazado, y se copie en el mismo desde memoria secundaria la página que produjo el fallo de página.

7.4. ¿Qué tareas debe realizar el sistema operativo cuando usa la técnica de paginación por demanda?

- ❖ **Asignación de marcos de memoria principal.** En un sistema multiprogramado se suelen cargar simultáneamente en memoria dos o más procesos. El SO tiene que decidir cuántos marcos asigna inicialmente a cada uno.
- ❖ **Control de carga.** El SO debe controlar el grado de multiprogramación del sistema.
- ❖ **Copia en la memoria secundaria de páginas modificadas.** El SO debe decidir en qué momento y de qué forma copiará las páginas que han sido modificadas en la memoria principal y secundaria.

7.5. Enumerar y describir brevemente los campos que suele contener una entrada de la tabla de páginas de un proceso cuando se utiliza paginación por demanda.

Como mínimo, una entrada de una tabla de páginas asociada a la página i de un proceso, aparte del marco de página j donde se aloja la página, debe contener un bit denominado presente/ausente o validez/invalidéz para indicar si la página está cargada en la memoria principal.

Cuando se crea un proceso y éste entra en el estado preparado para ejecución, el SO crea una tabla de páginas para dicho proceso que también reside en memoria principal. El tamaño de la tabla de páginas coincide con el tamaño del espacio virtual de

un proceso. La entrada i de la tabla se descompone en un número fijo de campos que contienen información sobre la página i del espacio de direcciones virtuales de un proceso. El número, la longitud y el nombre de estos campos depende de cada SO, pero generalmente se suelen encontrar al menos los siguientes campos.

- ❖ **Número de marco de página.** Contiene el número de marco j de memoria principal donde se encuentra almacenada la página i . El tamaño de este campo depende del número de marcos en que se descomponga la memoria principal.
- ❖ **Validez o presencia.** Consta de un único bit v que se activa ($v=1$) si la página i está cargada en el marco j , es decir, la entrada de la tabla se considera que es válida y puede ser utilizada. Si la página no pertenece actualmente al espacio de direcciones virtuales del proceso o la página no está cargada en memoria principal este bit está desactivado ($v=0$). En cada referencia a memoria hay que comprobar este bit. Si está desactivado entonces el hardware provoca un fallo de página.
- ❖ **Protección.** Este campo en su forma más simple consta de un único bit que en función de su valor indica si la página es de solo lectura o lectura-escritura. Usualmente suele contar de 3 bits ($p_2p_1p_0$) lectura, escritura y ejecución respectivamente.
- ❖ **Referenciada.** Consta de un único bit que se activa cuando la página i es referenciada por una dirección virtual.
- ❖ **Modificada.** Consta de un único bit que se activa cuando una página es modificada en una operación de escritura.

Los campos nº de marco de página, validez y protección son configurados por el SO y consultados por la MMU cada vez que hay que traducir una dirección virtual en una dirección física.

Por su parte, los campos referenciada y modificada son activados por el hardware. Inicialmente cuando una página i de un proceso es cargada en memoria estos campos de la entrada i de la tabla de páginas del proceso son configurados a 0 por el SO. Además cada cierto tiempo durante la ejecución de un proceso el SO pone a 0 el campo referenciada de todas las páginas cargadas en memoria.

7.6. ¿Qué elementos hardware son necesarios para lograr el reinicio de instrucciones en la técnica de paginación por demanda?

Las arquitecturas que soportan memoria virtual por demanda de página disponen de un registro especial donde copian el valor del registro contador del programa antes de iniciar la fase de búsqueda de una instrucción.

7.7. En la técnica de paginación por demanda ¿qué eventos provocan que el sistema operativo tenga que acceder a la memoria secundaria para leer o escribir páginas de procesos?

Cuando la referencia a una página i produce un fallo de página, el SO debe leer dicha página en memoria secundaria y copiarla a la memoria principal. También, cuando se produce un fallo de página y no existen marcos libres en memoria principal el SO debe seleccionar una página para ser reemplazada. Si la página ha sido modificada o no existe una copia de la misma en el área de intercambio dicha página debe ser escrita allí.

7.8. ¿En qué localizaciones de memoria secundaria puede buscar el sistema operativo una página de un proceso?

Bloque de disco de un archivo ejecutable. Parte de las páginas en que se divide el espacio de direcciones virtuales de un proceso, en concreto las asociadas a las regiones de código y datos, son creadas por el SO a partir del archivo ejecutable del cual es instancia dicho proceso. Conviene aclarar que los bloques de disco que ocupa el archivo no contienen a las páginas ya formadas propiamente dichas sino que poseen el contenido que permite su creación.

Bloque de disco del área de intercambio. El área de intercambio se utiliza para el almacenamiento de las páginas de un proceso presentes en la memoria principal cuando éste es intercambiado fuera para hacer sitio a otros procesos. También se utiliza para el almacenamiento de las páginas que han sido seleccionadas para ser reemplazadas en memoria principal por otras durante el tratamiento de fallos de página.

7.9. Señala las ventajas e inconvenientes de que el sistema operativo copie por adelantado en el área de intercambio todas las páginas del espacio de direcciones virtuales de un proceso.

El objetivo de esta estrategia es evitar tener que acceder al sistema de archivos cuando se produzcan fallos de página y acelerar así su tratamiento.

Si la copia de la imagen del proceso ocupa un rango de direcciones contiguas del disco y además cada bloque del área de

intercambio contiene una página, entonces el SO únicamente necesita conocer la dirección de disco donde comienza la copia de la imagen del proceso en el área de intercambio. Dicha dirección se almacena en el bloque de control de proceso.

El problema es que el espacio de direcciones virtuales de un proceso puede aumentar su tamaño debido al crecimiento de la región de pila o la región de datos. Si esto sucede hay que buscar un hueco lo suficientemente grande en el área de intercambio y copiar allí de nuevo todo el espacio del proceso.

Otro problema es que consume bastante espacio del área de intercambio, ya que todas las páginas de un proceso son copiadas por adelantado en dicha área, independientemente de que se vaya a utilizar o no.

7.10. ¿Qué significa que un marco esté bloqueado o pinchado? Señalar una situación donde se realice el bloque de marcos.

Si su contenido no puede ser remplazado. El SO bloquea un marco de página activando un bit de bloqueo en la entrada asociada a dicho marco en la tabla de marcos de páginas. Los Nmp marcos en que se divide la memoria principal en un conjunto V marcos están reservados para alojar el espacio del núcleo. También se reservan marcos para alojar buffers de E/S. Los marcos reservados están bloqueados.

7.11. ¿Qué acciones realiza el sistema operativo para el tratamiento de un fallo de página?

Si el bit de validez está desactivado entonces el hardware produce una excepción denominada fallo de página y guarda en la pila del núcleo el valor del contador de programa y de los registros especiales. A continuación se cede el control al SO para que atienda el fallo de página. Cómo se realiza el tratamiento de un fallo de página depende de cada SO. Pero de forma general se puede decir que en primer lugar el SO guarda el contexto del proceso A cuya ejecución produjo la excepción. En segundo lugar comprueba la causa de la excepción, ya que existen diferentes motivos que pueden causar una excepción y la identifica como fallo de página.

A continuación intenta averiguar cuál fue la dirección virtual que produjo el fallo de página, consultando típicamente los registros ocultos del procesador. Si el procesador no dispone de dichos registros el SO tendrá que simular la ejecución de la instrucción que produjo el fallo hasta descubrirla.

Después de determinar la dirección virtual que produjo el fallo de página, el SO comprueba que se trata de una dirección legal, es decir, que pertenece al espacio de direcciones virtuales del proceso y que el tipo de acceso que se pretende no viola el tipo de acceso permitido según el campo de protección de su entrada asociada en la tabla de páginas. Si no es una dirección legal entonces enviará al proceso una señal de aviso o finalizará el proceso.

Si la dirección es legal, el SO consulta la lista de marcos de páginas libres. Si no hay ningún marco libre debe invocar al algoritmo de reemplazamiento de página para seleccionar un marco de página *j*, que no esté bloqueado, donde cargar la página *i* que produjo el fallo.

Si la página *k* que estaba contenida en el marco *j* tenía su bit de modificada activado, entonces la página *k* es planificada para ser copiada al disco duro. El SO pasa al proceso *A* cuya ejecución produjo fallo de página al estado bloqueado mientras se completa la copia de la página *k* al disco duro. También bloquea el marco *j* en la tabla de marcos de página para impedir que pueda ser utilizado para almacenar una página distinta de la página *i*. Además, se realiza un cambio de proceso, para permitir que otro proceso *B* se pueda ejecutar mientras tanto.

Una vez que el marco *j* está limpio, es decir, la página *k* se ha copiado en el disco, el SO localiza la ubicación de la página *i* en memoria secundaria y planifica su copia a memoria principal.

Cuando la página *i* ha sido copiada en memoria principal se produce una interrupción de disco para avisar de tal evento al SO. Éste actualiza las tablas de páginas y desbloquea el marco *j* en la tabla de marcos de página. A continuación el SO modifica el contador de programa y otros registros guardados en el contexto de proceso *A* para que cuando sea planificado de nuevo pueda reiniciarse justo desde el comienzo de la instrucción que produjo el fallo de página.

El SO despierta el proceso *A*, que pasa al estado preparado para ejecución. Cuando el proceso *A* sea planificado para ejecución se cargará su contexto y se reiniciará su ejecución desde el principio de la instrucción que produjo el fallo de página.

En el caso de que el SO hubiese encontrado un marco libre en la lista de marcos libres, entonces en la gestión del fallo se habrían ahorrado los pasos de invocar al algoritmo de reemplazamiento de página y la operación E/S para copiar la página k desde memoria principal al disco.

7.12. ¿Qué afirma el principio de localidad de referencias?

También denominado ***principio de proximidad*** afirma que durante un determinado intervalo de tiempo de ejecución de un proceso éste referencia a direcciones virtuales próximas entre sí, es decir, a páginas cercanas. Por tanto una vez que dichas páginas se han traído a memoria, el proceso se ejecutará durante dicho intervalo de tiempo sin necesidad de provocar fallos de página. Dicho conjunto de páginas forma una zona de localidad del proceso. Transcurrido dicho intervalo el proceso pasará a referenciar durante otro intervalo de tiempo a otro conjunto de páginas cercanas, es decir, a otra zona de localidad, y así sucesivamente.

¿Qué es una zona de localidad?

Conjunto de páginas cercanas.

7.13. ¿Qué es el conjunto de trabajo de un proceso?

Si el proceso se está ejecutando dentro de una zona de localidad, por el principio de proximidad, las referencias futuras serán parecidas a las referencias pasadas. Esta idea es la que subyace en el concepto de *conjunto de trabajo de un proceso*, que se puede definir para el instante T_i de tiempo virtual como el conjunto de páginas distintas que el proceso en ejecución ha referenciado en sus últimas referencias a memoria principal.

7.14. ¿Qué conclusiones se derivan del estudio del conjunto de trabajo de un proceso?

- ❖ ***Una página que es miembro del conjunto de trabajo de un proceso no debe ser seleccionada como página víctima para ser reemplazada, ya que ello provocaría rápidamente un nuevo fallo de página.***
- ❖ ***Un proceso debe tener asignados el suficiente número de marcos para poder tener cargado en memoria su conjunto de trabajo.*** En caso contrario estará constantemente provocando fallos de página, es decir, se produce el fenómeno conocido como sobrepaginación o trasiego.

7.15. ¿Qué es la sobrepaginación?

Un proceso debe tener asignados el suficiente número de marcos para poder tener cargado en memoria su conjunto de trabajo. En caso contrario estará constantemente provocando fallos de página, es decir, se produce el fenómeno conocido como sobrepaginación o trasiego.

7.16. Definir los siguientes términos: a) Página víctima. b) Conjunto de marcos candidatos.

De entre el conjunto de páginas candidatas, el algoritmo de reemplazamiento debe elegir una para ser reemplazada. A la página elegida se le denomina *Página víctima*.

En relación con el conjunto de páginas candidatas a ser reemplazadas, se define el término *conjunto de marcos candidatos*, que hace referencia a los marcos de memoria principal donde se encuentran almacenadas las páginas candidatas.

7.17. Señalar las diferencias entre las estrategias de reemplazamiento de páginas local y global.

El conjunto de páginas candidatas a ser reemplazadas puede estar formado únicamente por las páginas cargadas en memoria principal en marcos no bloqueados pertenecientes al proceso que ha producido fallo de página. En dicho caso se dice que el SO utiliza una estrategia de reemplazamiento de páginas local.

Otra posibilidad es que el conjunto de páginas candidatas a ser reemplazadas esté formado por todas las páginas cargadas en memoria principal ubicadas en marcos no bloqueados. En este caso se dice que el SO utiliza una estrategia de reemplazamiento de páginas global.

7.18. ¿Qué criterios se tienen en cuenta para establecer el rendimiento de un algoritmo de reemplazamiento?

El número de fallos de página que permite evitar y al tiempo de procesador que requiere su ejecución. Lo deseable es que el algoritmo de reemplazamiento que utilice el SO minimice el número de fallos de página y produzca una sobrecarga mínima. El problema es que estos dos requisitos son opuestos, por lo que hay que tratar de llegar a un compromiso entre ambos.

También es deseable que el algoritmo de reemplazamiento de páginas que se utilice disminuya el número de fallos de página a

medida que aumenta el número de marcos de página de la memoria principal. Es decir, cumpla el principio de que a más recursos mejor comportamiento.

7.19. ¿Qué es la anomalía de Belady?

Aquellos algoritmos de reemplazamiento que pueden aumentar el número de fallos de página si aumenta el número de marcos en memoria principal.

7.20. Describir el algoritmo de reemplazamiento óptimo, sus ventajas e inconvenientes.

Selecciona para ser reemplazada aquella página del conjunto de páginas candidatas a ser reemplazadas que tardará más en volver a ser referenciada por una dirección virtual. Minimiza el número de fallos de página pero no se puede implementar en la práctica ya que requiere conocer la cadena de direcciones virtuales futuras a las que hará referencia el procesador.

7.21. Describir el algoritmo de reemplazamiento LRU, sus ventajas e inconvenientes.

El algoritmo de reemplazamiento de la página usada menos recientemente o algoritmo de reemplazamiento LRU (Least Recently Used) selecciona para ser reemplazada aquella página del conjunto de páginas candidatas a ser reemplazadas que lleva más tiempo sin ser referenciada, es decir, sin ser utilizada. La idea de este algoritmo es la de intentar predecir el futuro próximo teniendo en cuenta el pasado reciente.

Produce un número de fallos de página bastante cercano al algoritmo óptimo: Sin embargo, su principal inconveniente es que su implementación es compleja y produce bastante sobrecarga al sistema.

7.22. Describir dos posibles implementaciones del algoritmo de reemplazamiento LRU.

Una posible implementación es mediante el uso de lista enlazada. Cada entrada de la lista contiene el número de página de una página del conjunto de páginas candidatas a ser reemplazadas. Cada vez que una página es referenciada su número es colocado al principio de la lista. Es decir, la lista funciona como una pila. Cada vez que el procesador referencia a una dirección virtual, la lista debe ser ordenada, lo cual consume un tiempo importante.

Otras implementaciones requieren el apoyo de hardware. Por ejemplo, el algoritmo LRU se puede implementar utilizando un contador hardware que es incrementado cada vez que se referencia una dirección virtual. Además en cada entrada de una tabla de páginas hay que añadir un campo adicional para copiar el valor del contador, cada vez que una página es referenciada. Cuando se produce un fallo de página el SO examina el campo contador de cada entrada de la tabla de páginas y selecciona para ser reemplazada la página cuyo campo contador contiene el menor valor, ya que será la página que lleve más tiempo sin ser referenciada. Esta solución produce bastante sobrecarga, ya que en cada referencia a una dirección virtual hay que realizar un acceso a memoria principal para copiar el valor del contador en el campo de entrada correspondiente de la tabla de páginas. Además hay que realizar una búsqueda en todas las entradas de la tabla de páginas del campo contador de menor valor. Cuanto mayor sea el número de entradas más tiempo empleará la búsqueda.

7.23. Describir el algoritmo de reemplazamiento mediante envejecimiento. ¿Qué ventaja presenta este algoritmo frente al LRU?

Es una aproximación al algoritmo LRU que introduce una sobrecarga mucho más pequeña ya que se puede implementar de forma bastante eficiente.

Básicamente se asigna un registro de desplazamiento software de n bits a cada página cargada en memoria principal. El registro se inicializa a 0 cuando la página es cargada en memoria. Cada cierto tiempo T preestablecido, el SO desplaza un bit a la derecha el contenido del registro asignado a cada página cargando el bit referenciada en el bit más significativo de cada registro. Además pone a 0 el bit referenciada de cada página.

7.24. Describir el algoritmo de reemplazamiento PIFO, sus ventajas e inconvenientes.

Primero en entrar primero en salir selecciona para ser reemplazada aquella página del conjunto de páginas candidatas a ser reemplazadas que lleva más tiempo cargada en la memoria principal.

Para implementar este algoritmo el SO debe mantener una lista enlazada o cola FIFO. Cada entrada de la lista contiene el número de página de una página del conjunto de páginas

candidatas a ser reemplazadas. Cuando una página se carga en memoria su número de página se coloca al final de la lista. Así el número de página que se encuentra al principio de la lista indica la página que lleva más tiempo cargada en memoria y será la elegida para ser reemplazada. Cuando una página k que ocupaba el marco j es reemplazada por una página i entonces su número se elimina del principio de la lista y se coloca al final el número de la página que ha sido cargada en el marco j reemplazando a la página k .

Solo es necesario actualizar esta estructura de datos cuando se carga una página en un marco de memoria, por lo que su gestión requiere muy poco tiempo, además es sencillo de implementar y programar por lo que produce poca sobrecarga. Sin embargo, su rendimiento (número de fallos) no suele ser muy bueno. Además adolece de la anomalía de Belady.

7.25. Describir el algoritmo de reemplazamiento de la segunda oportunidad.

Es una variante del algoritmo FIFO que busca la página que lleva más tiempo cargada en memoria y no ha sido referenciada recientemente.

Básicamente este algoritmo consulta el bit referenciado (r) de la página que se encuentra al principio de la cola FIFO. Si $r=0$, entonces la página es seleccionada para ser reemplazada y el algoritmo finaliza. Si $r=1$ el algoritmo pone el bit a 0 y coloca el número de la página al final de la cola.

Tiene un mejor rendimiento en cuanto a la disminución del número de fallos de página que el algoritmo FIFO. Sin embargo produce mayor sobrecarga al estar moviendo páginas en la cola. Una forma de reducir esta sobrecarga es con una cola circular donde si $r=1$ este pasa a 0 y el puntero apunta a la siguiente número de página de la lista.

7.26. Describir el algoritmo de reemplazamiento del reloj mejorado.

El rendimiento del algoritmo del reloj puede mejorarse si además del bit referenciada (r) tiene en cuenta también el de modificada (m).

- $R = 0$ y $m = 0$. Son las candidatas ideales para ser reemplazadas.
- $R = 0$ y $m = 1$. Si se elige una página de este tipo para ser reemplazada, será necesario realizar una operación de

escritura en memoria secundaria para actualizar la copia de la página en el área de intercambio.

- $R = 1$ y $m = 0$. No son buenas candidatas para ser reemplazadas ya que puede que vuelvan a ser referenciadas próximamente.
- $R = 1$ y $m = 1$. No son buenas candidatas porque aparte de poder ser referenciadas próximamente deben ser salvadas en el área de intercambio.

Este algoritmo primero hace una ronda buscando $R=0$ y $m=0$, si no encuentra una segunda con 01, Si no encuentra una tercera con 00 y si no encuentra una última con 01.

7.27. Describir el algoritmo de reemplazamiento WSClock.

Para implementar WSClock es necesario que en cada entrada i de la tabla de página de un proceso se mantenga un campo denominado tiempo de último uso que contiene el tiempo virtual de último uso de página i . Si $r = 1$ entonces se escribe en el campo tiempo de último uso el tiempo virtual del proceso, es decir el tiempo que hace que la página no fue referenciada.

Si la edad de la página es menor o igual al tiempo de proceso entonces la página pertenece al conjunto de trabajo del proceso. Si no es una buena candidata a ser reemplazada.

En el caso 01 se planifica para ser escrita en el área de intercambio y el puntero pasa a la siguiente página.

El algoritmo WSClock busca páginas con $r = 0$, $m = 0$ y que no pertenezcan al conjunto de trabajo del proceso. Si durante una vuelta completa se da la situación de que todas las paginas pertenecen al conjunto de trabajo del proceso, entonces seleccionará la primera página que encuentre con $r = 0$ y $m=0$ aunque pertenezca al grupo de trabajo.

7.28. ¿Qué condiciones debería cumplir un buen algoritmo de reemplazamiento?

El mejor algoritmo realizable de reemplazamiento de páginas es aquél que produce una tasa de fallos cercana al producido por el algoritmo de reemplazo óptimo y además su implementación produce poca sobrecarga.

¿Cuáles son los algoritmos de reemplazamiento más utilizados en los sistemas operativos modernos?

Los más utilizados son el algoritmo de envejecimiento y WSClock ya sean en la forma descrita o con modificaciones.

WSClock solo se puede usar en local ya que se basa en el conjunto de trabajo de un determinado proceso.

7.29. ¿En qué consiste la tarea de asignación de memoria principal que realiza el sistema operativo para implementar la paginación por demanda?

Consiste en decidir cuántos marcos de página de la memoria principal reserva para un proceso que se tiene que ejecutar.

7.30. ¿Cuál es el número mínimo de marcos que es necesario asignar a un proceso?

Queda definido por el repertorio de instrucciones del computador. Se deben reservar como mínimo los marcos suficientes para contener todas las páginas a las que puede referenciar cualquier instrucción en el repertorio de la máquina.

7.31. Describir las dos posibles estrategias de asignación de memoria principal en paginación por demanda.

- ❖ **Asignación fija.** Consiste en asignar a cada proceso un número prefijado de marcos. Su principal ventaja es que produce poca sobrecarga ya que el SO solo tiene que decidir cuántos marcos asigna cuando se va a cargar el proceso. Su principal inconveniente es que si el conjunto de trabajo del proceso crece por encima del número de marcos reservados, entonces se produciría sobrepaginación, aunque existieran marcos libres en memoria. Por otro lado, si el conjunto de trabajo disminuye por debajo del número de marcos reservados se estaría desperdiciando memoria. Si el SO utiliza una estrategia de asignación fija, entonces la estrategia de reemplazo de páginas tiene que ser local. Si se reemplaza una página, la página víctima se elige de entre el conjunto de marcos asignados al proceso.
- ❖ **Asignación variable.** Consiste en ir variando a lo largo del tiempo de vida del proceso el número de marcos que tiene asignados. Su principal ventaja es su flexibilidad, lo que permite disminuir la sobrepaginación y la memoria desperdiciada. Además, permite que la política de reemplazo pueda ser local o global. El principal inconveniente de la asignación variable es que introduce una mayor sobrecarga

ya que el SO debe decidir continuamente cuantos marcos reserva a cada proceso.

7.32. Enumerar y describir brevemente los algoritmos de asignación de marcos más conocidos que se pueden emplear en la asignación variable.

- ❖ *Asignación equitativa.* Cada cierto tiempo comprueba el número de procesos en ejecución y asigna a cada proceso el mismo número de marcos. Se trata de un algoritmo justo ya que todos los procesos son tratados por igual Sin embargo desperdicia memoria.
- ❖ **Asignación proporcional.** Asigna a cada proceso un número de marcos en proporción a su tamaño o prioridad, o una combinación de ambos.
- ❖ **Por frecuencia de fallos de página (PPF).** Se basa en el principio que cuanto mayor sea el número de marcos asignados a un proceso menor será su tasa de fallos. Este principio lo cumplen todos aquellos algoritmos de reemplazamiento que no sufren la anomalía de Belady. El algoritmo mide la frecuencia de fallos de página en un proceso. Si la frecuencia de fallos de página se encuentra por encima de un valor límite prefijado, significa que la tasa de fallos es alta y asigna más marcos al proceso. Por otra parte, si la frecuencia está por debajo de un límite inferior prefijado se asignarán los marcos de este proceso a otros que lo necesiten. En el caso de que varios procesos pasen los límites y no existan marcos libres se intercambiarán en el área de intercambio.

7.33. Representar gráficamente la influencia del grado de multiprogramación sobre el porcentaje de uso del procesador.

7.34. ¿Qué es la política de limpieza?

El SO debe decidir en qué momento y de qué forma copiará las páginas que han sido modificadas de la memoria principal a la memoria secundaria. A la estrategia o política seguida por el SO para realizar esta tarea se le conoce comúnmente como política de limpieza.

7.35. Señalar las principales ventajas e inconvenientes de las políticas de limpieza por demanda y por adelantado.

- ❖ **Por demanda.** La página modificada se escribe en memoria cuando es seleccionada como página víctima para ser

reemplazada por otra página que ha producido un fallo de página. Con esta política se aplaza la escritura hasta que es estrictamente necesaria, con lo que se minimiza el número de operaciones de escritura. Sin embargo, cuando se produce un fallo de página hay que realizar dos operaciones de E/S, una de escritura de la página que se va a reemplazar y la otra de lectura de la página que ha producido el fallo de página, con lo que aumenta el tiempo que transcurre hasta que puede reiniciarse la ejecución del proceso que ha producido el fallo de página. El fallo de página se atendería más rápidamente si la página víctima ya hubiera sido copiada en memoria secundaria con anterioridad a que se produjese el fallo de página.

- ❖ **Por adelantado.** Las páginas que han sido modificadas se agrupan en lotes y se planifica cada cierto tiempo su escritura en memoria secundaria antes de que sean elegidas por el algoritmo de reemplazamiento. La principal ventaja es que permite atender los fallos de página más rápidamente al ahorrar, normalmente, la operación de escritura de la página víctima. Su principal inconveniente es que desperdicia tiempo de E/S, ya que una página que ha sido modificada una vez y que todavía no ha sido elegida como página víctima, puede volver a ser modificada en un futuro próximo por lo que tendrá que ser copiada varias veces. Una posible solución a las escrituras innecesarias consiste en seleccionar para ser copiadas solamente aquellas páginas modificadas que son candidatas a ser reemplazadas.

7.36. ¿Qué factores hay que tener en cuenta a la hora de seleccionar el tamaño de página?

- ❖ **Fragmentación interna.** En promedio la fragmentación interna es de media página por proceso. En consecuencia, cuanto mayor sea el tamaño de la página mayor será la fragmentación interna y mayor será el desperdicio de espacio de la memoria principal.
- ❖ **Tamaño de una tabla de páginas.** Cada proceso tiene una tabla de páginas cargada en memoria principal que posee un número de entradas igual al número de páginas en que se descompone el espacio de direcciones virtuales del proceso. Cuanto menor sea el tamaño de una página, en más páginas se descompondrá el espacio virtual de proceso, y las tablas de páginas serán de mayor tamaño. Por lo tanto, consumirán más espacio de memoria. Además, si el tamaño de la tabla de

páginas aumenta, el tiempo de cambio de proceso será mayor ya que el SO tendrá que cargar o vaciar más registros si se tiene una MMU con banco de registro o con TLB.

- ❖ **Número de fallos de página.** Si el tamaño de página es pequeño el espacio de direcciones virtual constará de más páginas por lo que la probabilidad de que se produzca un fallo de página será mayor.
- ❖ **Tiempo de uso de E/S.** El tiempo de una lectura o escritura en un disco se descompone en tiempo de búsqueda, tiempo de latencia rotacional y tiempo de transferencia. De estos tres tiempos el tiempo de transferencia es mucho más pequeño que los tiempos de búsqueda y de latencia. Por ello se tarda menos tiempo en transferir una página de tamaño grande que en transferir varias páginas de tamaño más pequeño. Por otra parte las páginas se escriben o se leen en memoria secundaria una a una. Si el tamaño de página es pequeño, el espacio de direcciones virtuales del proceso se descompondrá en un mayor número de páginas, por lo que la probabilidad de realizar operaciones de E/S será mayor, tiempo de ES que no puede ser utilizado por la ejecución de los procesos.

7.37. ¿Cuáles son las ventajas e inconvenientes de tener un tamaño de página grande?

Disminuye el tamaño de las tablas de páginas, el número de fallos de página y el tiempo de uso de la E/S. Sin embargo aumenta la fragmentación interna.

7.38. ¿En qué consiste la estrategia conocida como paginación por adelantado? ¿Cuándo se considera que esta estrategia es efectiva?

En un SO con memoria virtual mediante demanda de página cuando comienza la ejecución de un proceso o se continúa con la ejecución de un proceso que estaba bloqueado o intercambiado, se producen un conjunto de fallos de página iniciales hasta que el proceso tiene cargadas las páginas de su conjunto de trabajo.

Para evitar este conjunto de fallos iniciales se puede usar la estrategia conocida como paginación por adelantado (prepaging) que consiste en cargar un cierto número de páginas asociadas a un proceso antes de iniciar o continuar con su ejecución. También se puede utilizar la paginación por adelantado para cada vez que se

produce un fallo de página, con el objetivo de evitar futuros fallos de página.

El punto clave de esta estrategia radica en cargar las páginas adecuadas para evitar los fallos de página, ya que si se cargan varias páginas que no van a ser referenciadas se estaría perdiendo tiempo en cargar dichas páginas y no se evitarían fallos.

En general si se cargan Npa páginas por adelantado, se referenciarán Npr páginas y no se referenciarán Npa-Npr páginas. Cada página referenciada cargada por adelantado evita un fallo de página. La paginación por adelantado se considera que es efectiva si el tiempo ahorrado en tratar Npr fallos de páginas es superior al tiempo empleado en cargar Npa – Npr que no son referenciadas. En caso contrario no compensa utilizar paginación por adelantado.

7.39. ¿Qué opciones se emplean en relación a la selección de páginas que van a ser cargadas por adelantado?

- ❖ **Cargar el conjunto de trabajo del proceso**, que debe ser mantenido por el SO. Esta opción se puede emplear cuando se va a continuar con la ejecución un proceso que fue intercambiado o bloqueado.
- ❖ **Cargar un conjunto de páginas ubicadas de forma contigua en el área de intercambio**, ya que el tiempo de una operación de E/S para traer a memoria desde el disco Npa páginas contiguas es más pequeño que el tiempo empleado en realizar Npa operaciones de E/S para traerlas una a una cuando sean referenciadas.

7.40. ¿En qué consiste la reserva de marcos libres y qué ventajas ofrece?

El tiempo de tratamiento de un fallo de página se reduce si se dispone de un marco libre de memoria principal donde cargar la página referenciada por la dirección virtual que ha producido el fallo de página. Por este motivo el SO puede mantener una reserva de marcos libres implementada en la forma de una lista de marcos libres. En cada entrada de la lista de marcos libres se mantiene el número de marco j y la pagina i que contiene actualmente. Nótese que un marco j calificado como libre puede o bien estar vacío o contener una página i que es reemplazable.

Si la reserva de marcos libre cae por debajo de un cierto límite se utiliza el algoritmo de reemplazamiento para seleccionar páginas reemplazables, cuyos marcos se agregan a la lista hasta alcanzar cierto límite máximo.

Cuando se necesita un marco libre se coge el situado en la cabecera de la lista de marcos libres. Por otro lado, si se selecciona alguna página *i* como reemplazable su marco *j* es colocado al final de la lista si la página no ha sido modificada. Además el campo válido de la entrada *i* de la tabla de páginas correspondiente se desactiva. Si la página ha sido modificada, entonces ingresa en una lista de páginas modificadas que serán escritas por lotes en la memoria secundaria. Una vez escritas pasan a ser colocadas al final de la lista de marcos libres.

Mientras la página *i* siga cargada en el marco *j* puede ser utilizada. Por este motivo cuando se produce un fallo de página se comprueba la lista de marcos libres por si alguno de ellos contuviera la página que ha producido el fallo, y de este modo ahorrarse la lectura de la página en memoria secundaria. Así, la lista de marcos libres se utiliza como una memoria caché software de páginas.

Preguntas de autoevaluación tema 8

8.1. Explicar qué es un dispositivo modo bloque y un dispositivo modo carácter. Señalar algunos ejemplos de cada tipo.

Modo bloque almacena información en bloques de tamaño fijo, normalmente 512 bytes o una potencia de dos múltiplos de la anterior, cada uno de los cuales tiene asignado un determinado número identificador o dirección. Ejemplos de dispositivos modo bloque son los discos y unidades de cinta.

Modo carácter envía o recibe información como una secuencia o flujo lineal de bytes, en ellos la información no se organiza con una estructura concreta por lo que no es direccionable, y en consecuencia no permite la realización de operaciones de búsqueda.

8.2. ¿Qué información debe pasar un proceso a la llamada al sistema que invoque para solicitar una petición de E/S?