CSO TP5

1. La dirección lógica (o virtual) hace referencia a una dirección que enmascara o abstrae una dirección física, también hace referencia a una localidad en memoria y se la debe traducir a una dirección física.

Por otro lado, la dirección física hace referencia a una dirección real, que es con la que se accede efectivamente a la memoria. Representa la dirección absoluta en memoria.

1. a) En las particiones fijas la memoria se divide en particiones o regiones de tamaño fijo, estos pueden ser iguales o diferentes. En las particiones fijas se aloja un único proceso y cada proceso se coloca en alguna partición de acuerdo con algún criterio: First Fit, Best Fit, Worst Fit, y Next Fit.

Por otro lado, las particiones dinámicas varían en tamaño y número. Estas alojan un proceso cada una y cada partición se genera en forma dinámica del tamaño justo que necesita el proceso.

¿Qué problemas se generan en cada caso? --> La fragmentación se produce cuando una localidad de memoria no puede ser utilizada por no encontrarse en forma contigua:

La fragmentación interna se produce en el esquema de particiones fijas, esta es interna a la localidad asignada y es la porción de la localidad que queda sin utilizar.

La fragmentación externa se produce en el esquema de particiones dinámicas, son huecos que van quedando en la memoria a medida que los procesos finalizan. Al no encontrarse en forma contigua puede darse el caso de que tengamos memoria libre para alocar un proceso, pero que no podamos utilizar. La solución a esto es la compactación, la cual es muy cara.

b) Para gestionar la memoria con particiones, el sistema operativo debe disponer de: tamaño de memoria principal, tamaño y ubicación de particiones, tablas de asignación de memoria, estado de particiones, requisitos de memoria de los procesos.

3) a) En este caso, todas las particiones tienen el mismo tamaño predeterminado, independientemente de los procesos que se cargarán en ellas:

Ventajas:

* Simplicidad en la implementación: el SO no necesita calcular que partición asignar, ya que todas tienen el mismo tamaño.
* Facilidad de reutilización: las particiones libres pueden reutilizarse fácilmente, ya que no hay que considerar diferencias de tamaño.
* Rápida toma de decisiones: el proceso de asignación es directo y rápido

Desventajas:

* Fragmentación interna: si un proceso es significativamente más pequeño que la partición, se desperdicia mecha memoria.
* Ineficiencia en el uso de memoria: los procesos grandes que no caben en una partición no pueden ser cargados, incluso si la suma de las particiones libres podría alojarlos.

b) Las particiones se diseñan con tamaños variados para acomodar procesos de diferentes tamaños:

Ventajas:

* Reducción de fragmentación interna: las particiones más pequeñas pueden albergar procesos pequeños, y las grandes, procesos más grandes, optimizando el uso de memoria.
* Mayor adaptabilidad: es más probable que un proceso encuentre una partición adecuada, mejorando la flexibilidad.

Desventajas:

* Fragmentación externa: si se liberan varias particiones pequeñas y no contiguas, puede quedar memoria inutilizable para procesos grandes.
* Complejidad en la gestión: el SO necesita identificar la partición más adecuada para cada proceso, lo que puede aumentar el tiempo de procesamiento.
* Posible desperdicio: si solo hay procesos pequeños, las particiones grandes podrían quedar infrautilizadas.

4) a) La fragmentación interna ocurre cuando un bloque de memoria es más grande que el tamaño solicitado y se desperdicia el exceso de espacio, mientras que la externa ocurre cuando hay espacio entre los bloques asignados y la memoria no se puede usar para solicitudes más grandes.

b) Para subsanar la fragmentación externa existen dos métodos, uno es la compactación y otra es el algoritmo de ajuste. La compactación consiste en reorganizar la memoria moviendo bloques ocupados hacia un extremo de la memoria principal, y uniendo los bloques libres en una región contigua. Esto elimina los espacios vacíos dispersos y permite asignar bloques grandes para nuevos procesos.

5)a) La paginación es una estrategia de organización de memoria física que consiste en dividir la memoria en porciones de igual tamaño. Estas divisiones se las conoce como paginas físicas o marcos, estas divisiones facilitan la gestión de la memoria física.

Los marcos están identificados por un número, conocido como numero de página física, cada página física se asigna en exclusividad a un proceso.

El espacio de direcciones de los procesos se divide en porciones de igual tamaño denominadas página, el tamaño de página = tamaño marco = 512 bytes (generalmente).

b) El SO mantiene una tabla de páginas para cada proceso, la cual contiene el marco donde se encuentra cada página. Otra estructura adicional es el registro base de la tabla de páginas (PTR – Page Table Register), que guarda la dirección inicial de la tabla de páginas en memoria. Por último, las otras dos estructuras son el TLB (Translation Lookaside Buffer) que es una pequeña memoria cache utilizada para almacenar de manera temporal los mapeos recientes entre páginas y marcos; y el controlador de interrupciones que maneja excepciones como fallos de página.

c) A diagram of a computer system

Description automatically generated

d) En la paginación, la fragmentación puede ocurrir internamente ya que, si un proceso no ocupa completamente una página, el espacio restante dentro del marco asignado se desperdicia. En el caso de la externa no ocurre ya que la paginación elimina la necesidad de bloques contiguos de memoria, por lo que no quedan “huecos” inutilizables entre los marcos.

6) Diferencias:

|  |  |  |
| --- | --- | --- |
| Aspecto | Paginación | Particiones Fijas |
| Tamaño de bloques | Igual para todas las páginas y marcos | Puede ser igual o diferente, dependiendo del diseño |
| División de memoria | Se divide en páginas (lógicas) y marcos (física) | Solo se dividen particiones en la memoria física |
| Fragmentación interna | Menor, ya que el tamaño de las páginas suele ser pequeño | Mayor, si el tamaño de las particiones es grande |
| Fragmentación externa | No ocurre, ya que los bloques no necesitan estar contiguos | Ocurre si no hay particiones libres que se adapten al proceso |
| Mapeo de direcciones | Requiere una tabla de páginas para traducir direcciones lógicas a físicas | No necesita traducción, ya que cada particione tiene un tamaño fijo |
| Flexibilidad | Alta: las páginas de un proceso pueden ubicarse en cualquier marco libre | Baja: cada proceso necesita encajar en una partición especifica |
| Sobrecarga del SO | Mayor, debido al mantenimiento de las tablas de páginas y TLB | Menor, ya que la gestión es más simple. |

Similitudes:

* Tamaño fijo de los bloques: en ambas técnicas la memoria se divide en bloques de tamaño fijo
* Fragmentación interna: en ambas puede haber fragmentación interna, ya que un proceso o página puede no utilizar completamente el espacio asignado.
* Transparencia al usuario: en ambas técnicas, los programas no necesitan gestionar directamente cómo se asigna la memoria, ya que esto lo maneja el SO.
* Asignación predefinida: el tamaño de los bloques o particiones está predefinido y no cambia durante la ejecución.

7) Suponiendo un sistema donde la memoria es administrada mediante la técnica de paginación, y donde:

* El tamaño de la página es de 512 bytes
* Cada dirección de memoria referencia 1 byte
* Los marcos en memoria principal se encuentran desde la dirección física 0.

A table with numbers and letters

Description automatically generatedSuponiendo además un proceso con un tamaño de 2000 bytes y con la siguiente tabla de páginas:

a)

|  |  |  |  |
| --- | --- | --- | --- |
| #Marco | #Pagina | Dirección Virtual | Dirección Física |
| 0 | - | - | 0..511 |
| 1 | - | - | 512..1023 |
| 2 | 2 | 1024..1535 | 1024..1535 |
| 3 | 0 | 0..511 | 1536..2047 |
| 4 | - | - | 2048..2559 |
| 5 | 1 | 512..1023 | 2560..3071 |
| 6 | 3 | 1536..2047 | 3072..3583 |

b)

Dir. Lógica DIV Tam. Página = N° de Página

Dir. Lógica MOD Tam. Página = Desplazamiento

Dir. Física = Inicio o base del frame + deplazamiento

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dir. Lógica | DIV (512) N° Pag | MOD (512)  Desplazamiento | Marco  (base) | Dir. Física | Válida |
| 35 | 0 | 35 | M3 1536 | 1536 + 35 =  1525 | Si |
| 512 | 1 | 0 | M5 2560 | 2560 + 0 =  2560 | Si |
| 2051 | 4 | 3 | - | - | Error |
| 0 | 0 | 0 | M3 1536 | 1536 + 0 =  1536 | Si |
| 1352 | 2 | 301 | M2 1024 | 1024 + 301=  1325 | Si |
| 602 | 1 | 90 | M5 2560 | 2560 + 90 =  2650 | si |

c)

Dir. Física DIV Tam. Marco = N° de Marco

Dir. Física MOD Tam. Maco = Desplazamiento

Dir. Lógica = (N° página \* tam. Pagina) + Desplazamiento

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Dir. Física | DIV (512) N° Marco | MOD (512)  Desplazamiento | Página  (base) | Dir. Lógica | Válida |
| 509 | 0 | 509 | - | - | Error |
| 1500 | 2 | 476 | P2 1024 | 1024 + 476 = 1500 | Si |
| 0 | 0 | 0 | - | - | Error |
| 3215 | 6 | 143 | P3 1536 | 1536 + 143 = 1679 | Si |
| 1024 | 2 | 0 | P2 1024 | 1024 + 0 = 1024 | Si |
| 2000 | 3 | 464 | P0 0 | 0 + 464 = 464 | Si |

d) La fragmentación interna ocurre porque el tamaño de las páginas es fijo (512 bytes) y los procesos rara vez utilizan exactamente todos los bytes de una página, en este caso:

* Tamaño del proceso: 2000 bytes
* Tamaño de las páginas: 512 bytes
* Números de páginas utilizadas: 4 (3 completas + 1 parcial)

La última página del proceso (la 3) tiene un tamaño real de: 2000 mod 512 = 464 bytes usados 🡪 esto deja 512 – 464 = 48 bytes desperdiciados, la cual es la fragmentación interna total.

8) a)

- Bits para desplazamiento dentro de la página:

- Bits para el numero de página:

- Total de bits para la dirección lógica = Bits para el numero de página + Bits para el desplazamiento 🡪 3 + 10 = 13 Bits

b)

- Bits para desplazamiento dentro del marco:

- Bits para el numero de marcos:

- Bits de dirección física = Bits para el numero de marcos + Bits para el desplazamiento dentro de la página 🡪 10 + 5 = 16 Bits

9) a) La segmentación es un esquema que se asemeja a la visión del usuario. El programa es dividido en partes/sectores, donde en cada sección se guardan datos similares. Un programa es una colección de segmentos. Un segmento es una unidad lógica como:

- Programa principal, procedures, funciones, variables globales y locales, stack etc.

- Cada segmento tiene un registro base y un registro limite.

- Por lo que se genera una tabla de segmentos con esos dos valores para cada segmento por proceso.

La segmentación puede causar fragmentación externa. Todos los segmentos de un programa pueden no tener el mismo tamaño (códigos, datos, rutinas). La base y limite del segmento son dinámicos.

Las direcciones lógicas consisten en 2 partes:

* Selector de segmento
* Desplazamiento dentro del segmento (sobre registro base y límite)

Por último, la segmentación posee la ventaja sobre la paginación respecto de: la protección de espacios de memoria y la compartición de bloques de memoria.

b) Las estructuras adicionales que debe posee un SO para llevar a cabo su implementación son: una tabla de segmentos, donde por cada segmento hay dos valores: registro base y registro límite.

A diagram of a computer program

Description automatically generatedc)

d) En este esquema se puede producir fragmentación externa, ya que los segmentos tienen tamaños variables y se asignan en bloques de memoria contiguos. Con el tiempo, pueden quedar espacios pequeños (fragmentos) entre segmentos que no son lo suficientemente grande para alojar nuevos segmentos.

10) Similitudes:

* Tamaño variable: ambas técnicas asignan bloques de memoria de tamaño variable, ajustados según las necesidades del proceso o de sus componentes (segmentos)
* Fragmentación externa: ambas pueden sufrir de los mismo
* Gestión contigua: en ambos casos, los bloques asignados deben ser contiguos en la memoria física
* Requieren estructuras adicionales: ambas técnicas necesitan informacion de administración (como tablas de segmentos o listar particiones) para gestionar el uso de la memoria

Diferencias:

|  |  |  |
| --- | --- | --- |
| Aspecto | Segmentación | Particiones dinámicas |
| Unidad de asignación | Divide la memoria en segmentos lógicos definidos por el programador o sistema | Divide la memoria en particiones según el tamaño total solicitado por cada proceso |
| Base lógica | El espacio de direcciones lógicas esta dividido en segmentos que tienen tamaños independientes. | El proceso se considera como una única unidad, sin divisiones internas explícitas. |
| Traducción de direcciones | Se realiza usando una tabla de segmentos, donde cada segmento tiene una base y un límite. | Se realiza directamente usando una base (dirección inicial) y un tamaño para cada partición. |
| Protección | Ofrece protección a nivel de segmento; un proceso no puede acceder a segmentos fuera de su rango. | Solo se protege el proceso como una unidad; no hay separación interna de componentes lógicos. |
| Fragmentación interna | No ocurre, ya que los segmentos se ajustan exactamente al tamaño requerido. | Puede ocurrir si se asigna más memoria de la necesaria dentro de la partición. |
| Fragmentación externa | Sí ocurre, debido a la naturaleza variable de los segmentos. | También ocurre por el mismo motivo (particiones de tamaño variable). |
| Uso lógico | Refleja la estructura lógica del programa (por ejemplo, código, datos y pila separados). | No refleja la estructura lógica del programa, ya que el proceso se asigna como una unidad. |

11) Similitudes

* Traducción de direcciones: en ambas, las direcciones lógicas se transforman en direcciones físicas mediante una tabla administrada por el SO (tabla de páginas o tabla de segmentos)
* Asignación de memoria: ambas técnicas son métodos de asignación de memoria que dividen el espacio de direcciones lógicas y lo mapean en la memoria física.
* Independencia del proceso: tanto en paginación como en segmentación, cada proceso tiene su propio espacio de direcciones lógicas, aislado del de otros procesos
* Evitan fragmentación externa directa: aunque la segmentación puede sufrir fragmentación externa, su implementación combinada con paginación puede mitigarla. Por otro lado, paginación, al dividir la memoria física en marcos, evita directamente este problema.

Diferencias:

|  |  |  |
| --- | --- | --- |
| Aspecto | Paginación | Segmentación |
| Unidad de asignación | Divide la memoria lógica y física en bloques de tamaño fijo llamados páginas y marcos. | Divide la memoria lógica en unidades lógicas de tamaño variable llamadas segmentos. |
| Tamaño de unidad | Todas las páginas tienen el mismo tamaño fijo. | Los segmentos tienen tamaños variables, según las necesidades del programa. |
| Fragmentación | Puede producir fragmentación interna si no se utiliza completamente el espacio asignado a una página. | Puede producir fragmentación externa, ya que los segmentos necesitan espacio contiguo en memoria física. |
| Estructura lógica | No representa la estructura lógica del programa (código, datos, pila están en páginas arbitrarias). | Representa la estructura lógica del programa, asignando segmentos para código, datos, pila, etc. |
| Tabla utilizada | Usa una tabla de páginas, que mapea páginas lógicas a marcos físicos. | Usa una tabla de segmentos, que almacena la base (inicio) y el límite (tamaño) de cada segmento. |
| Traducción de direcciones | La dirección lógica se divide en dos partes: número de página y desplazamiento dentro de la página. | La dirección lógica se divide en número de segmento y desplazamiento dentro del segmento. |
| Flexibilidad en tamaños | Menos flexible, ya que las páginas tienen tamaños predefinidos. | Más flexible, ya que los segmentos pueden ajustarse a los tamaños requeridos por el programa. |
| Fragmentación externa | No ocurre, ya que los marcos son reutilizables sin importar su ubicación. | Sí ocurre, ya que los segmentos necesitan espacio contiguo en memoria física. |

12)

I) (2,1,1) = 1500 + 20 + 1 = 1521

II) (1, 3, 15) = 500 + 60 + 15 = 575

IIi) (3, 1, 10) = 5000 + 120 + 10 = 5130

IV) (2, 3, 5) = 1500 + 0 + 5 = 1505

13)

a) La memoria virtual es un esquema de administración que permite a un proceso trabajar con más memoria lógica de la que físicamente está disponible. Sus principales beneficios son:

* Ejecución de programas más grandes que la memoria física: La memoria virtual utiliza almacenamiento secundario (como un disco) para ampliar el espacio de direcciones disponible, permitiendo la ejecución de programas más grandes que la memoria física.
* Aislamiento entre procesos: Cada proceso tiene su propio espacio de direcciones, lo que evita que un proceso interfiera con la memoria de otro, mejorando la seguridad y estabilidad del sistema.
* Mejor utilización de la memoria: Solo se cargan en memoria física las partes del programa que están siendo utilizadas activamente, lo que reduce el uso innecesario de memoria.
* Multitarea eficiente: Al cargar y descargar páginas de forma dinámica, permite que varios procesos compartan la memoria física de manera eficiente.

b) Los elementos necesarios para implementar la memoria virtual son: Unidad de gestión de memoria (MMU), Tablas de página, Almacenamiento secundario (swap) y Mecanismos de manejo de fallos de página

c) Además del marco de memoria donde se encuentra la página (si está en memoria física), se requiere como mínimo un bit que indique la presencia de la página en memoria, debido a que solo a partir de este dato es que el HW puede generar la interrupción necesaria para resolver el fallo de página. Por otro lado, debe contarse con información sobre la presencia de modificaciones o no sobre las páginas cargadas, puesto que cualquier descarga de una página modificada implica la necesidad de actualizar los datos cargados en disco, para mantener la consistencia.

Otros bits pueden hacer referencia a si fue referenciado recientemente, esto es utilizado por algoritmos de reemplazo como LRU o Segunda Oportunidad. También se puede tener la dirección de almacenamiento secundario, que especifica donde está almacenada la página en el disco duro cuando no hay memoria libre en la física. Y por último permisos de acceso, que es información sobre los derechos de acceso de la página (lectura, escritura, ejecución), importante para la seguridad y protección.

14) a) Los fallos de página se producen cuando una instrucción ejecutada hace referencia a una dirección lógica cuya página no está cargada en memoria.

b) El responsable de detectar un fallo de página es el mismo HW, mediante una interrupción. Al momento de resolver la dirección, cuando recupera la entrada en la tabla de páginas correspondiente, debe analizar el bit de control V, que indica si la página encuentra o no cargada en memoria principal.

c)

1- Se genera el trap.

2- El SO bloquea al proceso (la CPU toma otro proceso).

3- El SO busca un marco libre en la memoria y genera una operación de E/S que le pide al disco, para copiar en dicho marco la pagina deseada.

4- La E/S avisa por interrupción cuando finaliza.

5- El SO actualiza la tabla de paginas del proceso:

* + Colocando el bit V en 1, en la página correspondiente.
  + Coloca la dirección base del marco donde se colocó la página.

6- El proceso pasa del estado bloqueado a listo para ejecutar.

7- Cuando vuelva a ser asignado al CPU, el proceso comenzará desde la instrucción que generó el fallo en primera instancia.

15)