Gestion de memoria - Solución de ejercicio de libro de sistemas operativos

Sistemas Operativos (Universidad de Cundinamarca)

Nombre:

GESTIÓN DE MEMORIA EN LOS SISTEMAS OPERATIVO

Cuestionario.

1) ¿Qué requisitos se intenta satisfacer en gestión de la memoria?

R/: Los requisitos que se intenta satisfacer en la gestión de memoria son:

- Reubicación.
- Protección.
- Compartición.
- Organización lógica.
- Organización física.

2) ¿Por qué es deseable la capacidad para reubicar procesos?

R/: Es deseable la capacidad de reubicar procesos porque no se puede conocer de forma anticipada dónde se va a colocar un programa y se debe permitir que los programas se puedan mover en la memoria principal, debido al intercambio o swap.

3) ¿Por qué no es posible forzar la protección de la memoria en tiempo de compilación?

R/: Esto es debido a que el sistema operativo no puede anticipar todas las referencias de memoria que

un programa hará. Incluso si tal anticipación fuera posible, llevaría demasiado tiempo calcularlo para cada programa a fin de comprobar la viola-ción de referencias de la memoria.

4) ¿Qué razones existen para permitir que dos o más procesos accedan a una misma región de la memoria?

R/: Las razones que existen para permitir que dos o más procesos accedan a una misma región de la memoria es que si varios programas están ejecutando el mismo programa, es ventajoso permitir que cada proceso pueda acceder a la misma copia del programa en lugar de tener su propia copia separada. Procesos que estén cooperando en la misma tarea podrían necesitar compartir el acceso a la misma estructura de datos. Por tanto, el sistema de gestión de la memoria debe permitir el acceso controlado a áreas de memoria compartidas sin comprometer la protección esencial.

5) En un esquema de particionamiento fijo, ¿cuáles son las ventajas de utilizar particiones de distinto tamaño?

R/: Las ventajas de implementar particiones de distinto tamaño son

1. Los módulos se pueden escribir y compilar independientemente, con todas las referencias de un módulo desde otro resueltas por el sistema en tiempo de ejecución.
2. Con una sobrecarga adicional modesta, se puede proporcionar diferentes grados de protección a los módulos (sólo lectura, sólo ejecución)

3. Es posible introducir mecanismos por los cuales los módulos se pueden compartir entre los procesos. La ventaja de proporcionar compartición a nivel de módulo es que se corresponde con la forma en la que el usuario ve el problema, y por tanto es fácil para éste especificar la compartición deseada.

6) ¿Cuál es la diferencia entre fragmentación interna y externa?

R/: A medida que pasa el tiempo, la memoria se fragmenta cada vez más y la utilización de la memoria se decrementa. Este fenómeno se conoce como fragmentación externa, indicando que la memoria que es externa a todas las particiones se fragmenta de forma incremental, por contraposición a lo que ocurre con la fragmentación interna, descrita anteriormente.

En el ejemplo, podría haber un programa cuya longitud es menor que 2 Mbytes; ocuparía una partición de 8 Mbytes cuando se lleva a la memoria. Este fenómeno, en el cual hay espacio interno malgastado debido al hecho de que el bloque de datos cargado es menor que la partición, se conoce con el nombre de fragmentación interna.

7) ¿Cuáles son las distinciones entre direcciones lógicas, relativas y físicas?

R/: Una dirección lógica es una referencia a una ubicación de memoria independiente de la asignación actual de datos a la memoria; se debe llevar a cabo una traducción a una dirección física antes de que se alcance el acceso a la memoria. Una dirección relativa es un ejemplo particular de

dirección lógica, en el que la dirección se expresa como una ubicación relativa a algún punto conocido, no-malmente un valor en un registro del procesador. Una dirección física, o dirección absoluta, es una ubicación real de la memoria principal.

8) ¿Cuál es la diferencia entre una página y un marco?

R/: La diferencia es que la memoria principal se divide en porciones de tamaño fijo relativamente pequeños, y que cada proceso también se divide en porciones pequeñas del mismo tamaño fijo. A dichas porciones del proceso, conocidas como páginas, se les asigna porciones disponibles de memoria, conocidas como marcos, o marcos de páginas. Esta sección muestra que el espacio de memoria malgastado por cada proceso debido a la fragmentación interna corresponde sólo a una fracción de la última página de un proceso.

9) ¿Cuál es la diferencia entre una página y un segmento?

R/: La segmentación elimina la fragmentación interna, pero, al igual que el particionamiento dinámico, sufre de fragmentación externa. Sin embargo, debido a que el proceso se divide en varias piezas más pequeñas, la fragmentación externa debería ser menor. Mientras que la paginación es invisible al programador, la segmentación es normalmente visible y se proporciona como una utilidad para organizar programas y datos.

PROBLEMAS

1) En la Sección 2.3, se listaron cinco objetivos de la gestión de la memoria y en la Sección 7.1 cinco requisitos.

R/: Los cinco objetivos de la gestión de memoria:

- Proceso de Aislamiento
- Asignación y gestión automáticas
- Soporte de programación modular
- Protección y control de acceso
- Almacenamiento a largo plazo

Los cinco requisitos de memoria:

- Relocalización
- Protección
- Compartición
- Organización Local
- Organization Fisica

2) Considérese un esquema de particionamiento fijo con particiones de igual tamaño de 2 bytes y una memoria principal total de tamaño 224 bytes. Por cada proceso residente, se mantiene una tabla de procesos que incluye un puntero a una partición. ¿Cuántos bits necesita el puntero?

R/: El número de particiones es igual al número de bytes de la memoria principal dividido por el número de bytes en cada partición: $2^{24}/2^{16} = 2^{8}$ Se necesitan ocho bits para identificar una de las 2^{8} particiones.

3) Considérese un esquema de particionamiento dinámico. Demostrar que, en media, la memoria contiene la mitad de los huecos que de segmentos.

R/: Deje que s y h denoten la cantidad promedio de segmentos y agujeros, respectivamente. La probabilidad de que un segmento dado sea seguido por un agujero en la memoria (y no por otro segmento) es 0.5, porque eliminaciones y creaciones son igualmente probables en equilibrio. entonces, con los segmentos s en la memoria, el número promedio de agujeros debe ser s / 2. Es intuitivamente razonable que el número de agujeros debe ser menor que el número de segmentos porque los segmentos vecinos se pueden combinar en un solo agujero al eliminarlos.

4) Para implementar los diferentes algoritmos de colocación discutidos para el particionamiento dinámico (Sección 7.2), se debe guardar una lista de los bloques libres de memoria.

R/:

- Best-Fit: Best-Fit busca la lista completa de bloques disponibles en la memoria. Cuando llega un proceso con un tamaño, el algoritmo Best-fit necesita buscar toda la memoria. Si una memoria que tiene el tamaño del algoritmo de mejor ajuste de n Mbytes debe buscar n bytes para los bloques de memoria disponibles. Por lo tanto, la duración promedio de la búsqueda para un mejor ajuste es n.
- First-Fit: First-Fit escanea la memoria del bloque disponible desde el principio y selecciona el primer bloque de memoria lo suficientemente grande. Si una memoria tiene el tamaño de n bytes, suponga que llega un nuevo proceso con un tamaño de 12 Mbytes, primero ajusta escanea la memoria desde el principio y encuentra un bloque que tiene suficiente bloque de memoria de 12 Mbytes en el medio del escaneo. Luego, first-fit asigna el bloque de memoria al proceso y detiene el escaneo de los bloques de memoria restantes. Por lo tanto, la duración promedio de búsqueda para first-fit es n / 2.
- Next-fit: Next-fit escanea la memoria del último bloque asignado Esto significa que un proceso llega con un tamaño de 12 Mbytes, next-fit escanea la memoria desde el principio y encuentra el bloque de memoria suficiente en el medio del escaneo. Ahora, llega un nuevo proceso con un tamaño igual o superior al del proceso anterior, next-

fit inicia el escaneo desde donde se asignó el proceso anterior. Porque, el proceso de búsqueda del bloque de memoria y encontrado justo ahora, no hay bloques disponibles antes, por lo tanto, el análisis de proceso recién llegado se inicia desde la asignación del anterior.

El escaneado del proceso recién llegado comienza desde n / 2, la búsqueda de longitud se reduce a n / 2. Ahora que el proceso recién llegado encuentra el bloque suficiente en el medio, la longitud promedio para esto es n / 4.

5) Otro algoritmo de colocación para el particionamiento dinámico es el de peor ajuste (*worst-fit*). En este caso, se utiliza el mayor bloque de memoria libre para un proceso. Discutir las ventajas e inconvenientes de este método comparado con el primer, próximo y mejor ajuste. ¿Cuál es la longitud media de la búsqueda para el peor ajuste?

R/: El peor ajuste necesita buscar n cantidad de páginas hasta que se encuentre el bloque grande. Por lo tanto, la búsqueda de longitud promedio para el peor ajuste es n.

6) Si se utiliza un esquema de particionamiento dinámico y en un determinado momento la configuración de memoria es la siguiente:

Las áreas sombreadas son bloques asignados; las áreas blancas son bloques libres. Las siguientes tres peticiones de memoria son de 40M, 20M y 10M. Indíquese la dirección inicial para cada uno de los tres bloques utilizando los siguientes algoritmos de colocación:

a) Primer ajuste

b) Mejor ajuste

c) Siguiente ajuste. Asúmase que el bloque añadido más recientemente se encuentra al comienzo de la memoria.

d) Peor ajuste

R/: Este diagrama muestra un ejemplo de configuración de memoria en partición dinámica, después de que se llevaron a cabo varias operaciones de colocación y cambio. Las direcciones van de izquierda a derecha; las áreas grises indican bloques ocupados por procesos; las áreas blancas indican bloques de memoria libres. El último proceso realizado es de 2 Mbytes y está marcado con una X. Solo se cambió un proceso después de eso.

7) Un bloque de memoria de 1 Mbyte se asigna utilizando el sistema *buddy*:

a. Mostrar los resultados de la siguiente secuencia en una figura similar a la Figura 7.6:

Petición 70; Petición 35; Petición 80; Respuesta A; Petición 60; Respuesta B; Respuesta

D; Respuesta C.

R/: Dado que se asigna un bloque de 1Mb para el sistema Buddy

Solicitud: 70 (A)

Solicitud: 35 (B)

Solicitud: 80 (C)

Devolución: A

Solicitud: 60 (D)

Regreso: B

Regreso: D

Regreso: C

Supongamos que bloques con letras mayúsculas A para 70, 35 para B, C para 80 y D para 60.

1 Mbyte-Block

b. Mostrar la representación de árbol binario que sigue a Respuesta B.

R/: El tamaño del bloque es 16, entonces La dirección binaria de buddy es 011011100000

8) Considérese un sistema *buddy* en el que un determinado bloque en la asignación actual tiene la dirección 011011110000.

a) Si el bloque es de tamaño 4, ¿cuál es la dirección binaria de su bloque compañero o *buddy*?

b) Si el bloque es de tamaño 16, ¿cuál es la dirección binaria de su bloque compañero o *buddy*?

9) Sea *buddyk(x)* = dirección del bloque de tamaño 2k, cuya dirección es *x*.

Escribir una expresión general para el bloque compañero *buddy de buddyk(x)*.

R/: Tomar buddy k (x) = direccion del buddy del tamaño del bloque 2^{K} (la direccion es x).

La expresión general para buddy buddy k (x) se puede escribir de la siguiente manera:

buddy k (x) es dependiente de x mod $2^{(k+1).}$

Si el x mod $2^{(k+1)}$ es 0, entonces el buddy k (x) es $x+2^{k.}$

Si el x mod $2^{(k+1)}$ es 2^{k}, entonces el buddy k (x) es $x+2^{k.}$

10) La secuencia Fibonacci se define como sigue:

$$Fo=0, F1=1, Fn+2= Fn+1 + Fn, n \geq 0$$

a) ¿Podría utilizarse esta secuencia para establecer un sistema *buddy*?

R/: Esta secuencia se puede usar para establecer un sistema de buddy. Porque, los tamaños de bloque deberían satisfacer esta ecuación

b) ¿Cuál sería la ventaja de este sistema respecto al sistema *buddy* binario descrito en este

R/: La ventaja de este sistema sobre el sistema de compañero binario es que el sistema de compañero ofrece más tamaño de bloque que el sistema

de compañero binario. Esto tiene el potencial de una menor fragmentación interna, pero puede causar fragmentación externa adicional porque muchos bloques inútiles pequeños son creado creado.

11) Durante el curso de ejecución de un programa, el procesador incrementará en una palabra los contenidos del registro de instrucciones (contador de programa) después de que se cargue cada instrucción, pero alterará los contenidos de dicho registro si encuentra un salto o instrucción de llamada que provoque la ejecución de otra parte del programa. Ahora considérese la Figura 7.8. Hay dos alternativas respecto a las direcciones de la instrucción:

- Mantener una dirección relativa en el registro de instrucciones y hacer la traducción de direcciones dinámica utilizando el registro de instrucciones como entrada. Cuando se encuentra un salto o una llamada, la dirección relativa generada por dicho salto o llamada se carga en el registro de instrucciones.
- Mantener una dirección absoluta en el registro de instrucciones. Cuando se encuentra un salto o una llamada, se emplea la traducción de direcciones dinámica, almacenando los resultados en el registro de instrucciones. ¿Qué opción es preferible?

R/: El uso de direcciones absolutas reduce la cantidad de veces que se debe realizar la traducción dinámica de direcciones. Sin embargo, deseamos que el programa sea reubicable. Por lo tanto, podría ser preferible usar direcciones relativas en el registro de instrucciones. Alternativamente, la dirección en el

registro de instrucciones se puede convertir a relativa cuando un proceso se intercambia fuera de la memoria.

12) Considérese un sistema de paginación sencillo con los siguientes parámetros: 232 bytes de memoria física; tamaño de página de 210 bytes; 216 páginas de espacio de direccionamiento lógico.

a) ¿Cuántos bits hay en una dirección lógica?

R/: El número de bytes en el espacio de direcciones lógicas es (2^16 páginas) × (2^10 bytes / página) = 2^26 bytes. Por lo tanto, se requieren 26 bits para la dirección lógica.

b) ¿Cuántos bytes hay en un marco?

R/: Un marco tiene el mismo tamaño que una página, 2^{10} bytes.

c) ¿Cuántos bits en la dirección física especifica el marco?

R/: El número de cuadros en la memoria principal es (2^{32} bytes de memoria principal) / (2^{10} bytes / cuadro) = 2^{22} cuadros. Entonces se necesitan 22 bits para especificar el marco.

d) ¿Cuántas entradas hay en la tabla de páginas?

R/: Hay una entrada para cada página en el espacio de direcciones lógicas. Por lo tanto, hay 2^16 entradas.

e) ¿Cuántos bits hay en cada entrada de la tabla de páginas?

R/: Además del bit válido / inválido, se necesitan 22 bits para especificar la ubicación del cuadro en la memoria principal, para un total de 23 bits.

13) Considérese un sistema de segmentación sencillo que tiene la siguiente tabla de segmentos:

Dirección inicial Longitud (bytes)

660 248

1752 422

222 198

996 604

Para cada una de las siguientes direcciones lógicas, determina la dirección física o indica si

se produce un fallo de segmento:

a) 0,198

b) 2,156

c) 1,530

d) 3,444

Cada dirección física tiene 12 bits, de ellos 10 se emplean para desplazamiento dentro de la trama (log21024 = 10), y el resto 2 para especificar el número de trama. La memoria principal estará formada por: 22 = 4 tramas. Las direcciones lógicas que especifica el enunciado están en el formato de proceso, segmento y desplazamiento dentro del segmento. Para que una dirección lógica sea correcta ha de ser menor que el tamaño del segmento al que pertenece. Como se trata de segmentación paginada es necesario identificar la página del segmento a la que corresponde cada

dirección lógica para ubicarla en memoria. Considerando la memoria principal inicialmente vacía y aplicando el algoritmo LRU GLOBAL se obtiene la siguiente ocupación de memoria para las referencias anteriores: marco B, 3, 2100 (pág. 2) A, 2, 1100 (pág. 1) B, 0, 800 (pág. 0) B, 2, 300 (pág. 0) A, 0, 50 (pág. 0) B, 0, 300 (pág. 0) 0 B,3,2 B,3,2 B,3,2 B,3,2 A,0,0 A,0,0 1 A,2,1 A,2,1 A,2,1 A,2,1 A,2,1 2 B,0,0 B,0,0 B,0,0 B,0,0 3 B,2,0 B,2,0 B,2,0

14) Considérese una memoria en la cual se colocan segmentos contiguos S1, S2, …, Sn en su orden

de creación, desde un extremo del dispositivo al otro, como se sugiere en la siguiente

figura:

Cuando se crea el segmento contiguos Sn+1, se coloca inmediatamente después de Sn, incluso

si algunos de los segmentos S1, S2, …, Sn ya se hubieran borrado. Cuando el límite entre

segmentos (en uso o borrados) y el hueco alcanzan el otro extremo de memoria, los segmentos

en uso se compactan.

a) Mostrar que la fracción de tiempo F utilizada para la compactación cumple la siguiente

inigualdad:

donde

s = longitud media de un segmento, en palabras

t = tiempo de vida medio de un segmento, en referencias a memoria

f = fracción de la memoria que no se utiliza bajo condiciones de equilibrio

Sugerencia: Encontrar la velocidad media a la que los límites cruzan la memoria y

b) Encontrar F para f=0,2, t=1000 y s=50.

R/: Sistema de segmentos contiguos

a. Considere la siguiente ecuación de desigualdad:

Donde:

s = longitud promedio de un segmento, en palabras.

t = vida media de un segmento, en memoria se refiere.

f = fracción de la memoria que no se utiliza en condiciones de equilibrio.

Paso 2: Ahora, exprese la fracción F del tiempo de compactación. Observe que surge una referencia a algún segmento en la memoria de cada unidad de tiempo, y que un segmento se elimina cada t referencias. Debido a que el sistema está en equilibrio, se debe insertar un nuevo segmento cada t referencias. Por lo tanto, la velocidad del movimiento del límite es s / t palabras por unidad de tiempo. La operación del sistema timeo, luego el tiempo requerido para que el límite cruce el agujero es: El tiempo de operación del sistema to, entonces el tiempo requerido para que el límite cruce el agujero.