Tecnológico de Costa Rica

Escuela de Ingeniería en Computación
Principios de Sistemas Operativos

12 de octubre de 2017

Samantha Arburola León
2013101697

Operating System Concepts

Chapter 9

9.1 Under what circumstances do page faults occur? Describe the actions taken by the operating system when a page fault occurs.

Se produce un error de página cuando tiene lugar un acceso a una página que no se ha llevado a la memoria principal. El sistema operativo verifica el acceso a la memoria, cancelando el programa si no es válido. Si es válido, se encuentra un marco libre y se solicita a E / S que lea la página necesaria

9.2 Assume that you have a page-reference string for a process with m frames (initially all empty). The page-reference string has length p; n distinct page numbers occur in it. Answer these questions for any page-replacement algorithms:
a. What is a lower bound on the number of page faults?
b. What is an upper bound on the number of page faults?

A -> N
b -> P

9.3 Consider the page table shown in Figure 9.30 for a system with 12-bit virtual and physical addresses and with 256-byte pages. The list of free page frames is D, E, F (that is, D is at the head of the list, E is second, and F is last). Convert the following virtual addresses to their equivalent physical addresses in hexadecimal. All numbers are given in hexadecimal. (A dash for a page frame indicates that the page is not in memory.)

- **9EF** - 0EF
- **111** - 211
- **700** - D00
- **0FF** - EFF

9.4 Consider the following page-replacement algorithms. Rank these algorithms on a five-point scale from "bad" to "perfect" according to their page-fault rate. Separate those algorithms that suffer from Belady's anomaly from those that do not.

a. LRU replacement
b. FIFO replacement
c. Optimal replacement
d. Second-chance replacement

Rank	Algorithm	Suffer from Belady's anomaly
1	Optimal	no
2	LRU	no
3	Second-chance	si
4	FIFO	si

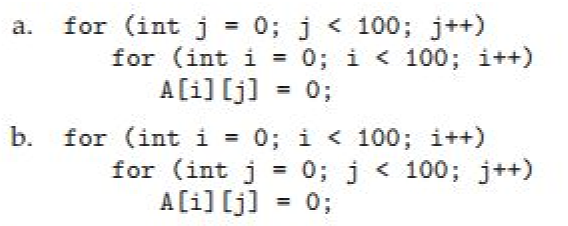
9.5 Discuss the hardware support required to support demand paging.

Para cada operación de acceso a la memoria, se debe consultar la tabla de páginas para verificar si la página correspondiente es residente o no y si el programa tiene privilegios de lectura o escritura para acceder a la página. Un TLB podría servir como caché y mejorar el rendimiento de la operación de búsqueda

9.6 An operating system supports a paged virtual memory, using a central processor with a cycle time of 1 microsecond. It costs an additional 1 microsecond to access a page other than the current one. Pages have 1000 words, and the paging device is a drum that rotates at 3000 revolutions per minute and transfers 1 million words per second. The following statistical measurements were obtained from the system: • 1 percent of all instructions executed accessed a page other than the current page. • Of the instructions that accessed another page, 80 percent accessed a page already in memory. When a new page was required, the replaced page was modified 50 percent of the time. Calculate the effective instruction time on this system, assuming that the system is running one process only and that the processor is idle during drum transfers.

effective access time = 0.99 × (1 sec + 0.008 × (2 sec) + 0.002 × (10,000 sec + 1,000 sec) + 0.001 × (10,000 sec + 1,000 sec)
effective access time =(0.99 + 0.016 + 22.0 + 11.0) sec
effective access time = 34.0 sec

9.7 Consider the two-dimensional array A: int A[][] = new int[100][100];
where A[0][0] is at location 200 in a paged memory system with pages of size 200. A small process that manipulates the matrix resides in page 0 (locations 0 to 199). Thus, every instruction fetch will be from page 0. For three page frames, how many page faults are generated by the following array-initialization loops, using LRU replacement and assuming that page frame 1 contains the process and the other two are initially empty?



```
a.  for (int j = 0; j < 100; j++)
        for (int i = 0; i < 100; i++)
            A[i][j] = 0;

b.  for (int i = 0; i < 100; i++)
        for (int j = 0; j < 100; j++)
            A[i][j] = 0;
```

a -> 5000
b -> 50

9.8 Consider the following page reference string:

1, 2, 3, 4, 2, 1, 5, 6, 2, 1, 2, 3, 7, 6, 3, 2, 1, 2, 3, 6.

How many page faults would occur for the following replacement algorithms, assuming one, two, three, four, five, six, or seven frames? Remember all frames are initially empty, so your first unique pages will all cost one fault each. • LRU replacement • FIFO replacement • Optimal replacement

Number of frames	**LRU**	**FIFO**	**Optimal**
1	20	20	20
2	18	18	15
3	15	16	11
4	10	14	8
5	8	10	7
6	7	10	7
7	7	7	7

9.9 Suppose that you want to use a paging algorithm that requires a reference bit (such as second-chance replacement or working-set model), but the hardware does not provide one. Sketch how you could simulate a reference bit even if one were not provided by the hardware, or explain why it is not possible to do so. If it is possible, calculate what the cost would be

Puede usar el bit válido / inválido admitido en el hardware para simular el bit de referencia. Inicialmente establezca el bit a inválido. En la primera referencia, se genera una trampa para el sistema operativo. El sistema operativo configurará un bit de software en 1 y restablecerá el bit válido / inválido a válido.

9.10 You have devised a new page-replacement algorithm that you think may be optimal. In some contorted test cases, Belady's anomaly occurs. Is the new algorithm optimal? Explain your answer.
No. Un algoritmo óptimo no sufrirá la anomalía de Belady porque, por definición, un algoritmo óptimo reemplaza a la página que no se usará por más tiempo. La anomalía de Belady se produce cuando un algoritmo de sustitución de páginas expulsa una página que se necesitará en el futuro inmediato Un algoritmo óptimo no habría seleccionado dicha página.

9.11 Segmentation is similar to paging but uses variable-sized "pages."Define two segment-replacement algorithms based on FIFO and LRU pagereplacement schemes. Remember that since segments are not the same size, the segment that is chosen to be replaced may not be big enough to leave enough consecutive locations for the needed segment. Consider strategies for systems where segments cannot be relocated, and those for systems where they can

FIFO. Encuentre el primer segmento lo suficientemente grande como para acomodar el segmento entrante. Si la reubicación no es posible y ningún segmento es lo suficientemente grande, seleccione una combinación de segmentos cuyas memorias sean contiguas, que estén "más cerca de la primera de la lista" y que puedan acomodar el nuevo segmento. Si es posible la reubicación, reorganice la memoria para que los primeros N segmentos sean lo suficientemente grandes como para el segmento entrante es contiguo en la memoria.

LRU. Seleccione el segmento que no se haya utilizado durante el período de tiempo más largo y que sea lo suficientemente grande, agregando cualquier espacio sobrante a la lista de espacio libre. Si ningún segmento es lo suficientemente grande, seleccione una combinación de los segmentos "más antiguos" que estén contiguos en la memoria (si la reubicación no está disponible) y que sean lo suficientemente grandes. Si la reubicación está disponible, reorganice los segmentos N más antiguos para que queden contiguos en la memoria y reemplácelos con el segmento nuevo.

9.12 Consider a demand-paged computer system where the degree of multiprogramming is currently fixed at four. The system was recently measured to determine utilization of CPU and the paging disk. The results are one of the following alternatives. For each case, what is happening? Can the degree of multiprogramming be increased to increase the CPU utilization? Is the paging helping?

cpu 13%, DISK 97%
cpu 87%, DISK 3%
cpu 13%, DISK 3%