Virtual Memory

Practice Exercises

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

Answer:
A page fault occurs when an access to a page that has not been brought into main memory takes place. The operating system verifies the memory access, aborting the program if it is invalid. If it is valid, a free frame is located and I/O is requested to read the needed page into the free frame. Upon completion of I/O, the process table and page table are updated, and the instruction is restarted.

10.2 Assume that you have a page-reference string for a process with m frames (initially all empty). The page-reference string has length p, and 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?

Answer:

a. n

b. p

10.3 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

Answer:

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

10.4 An operating system supports a paged virtual memory. The central processor has a cycle time of 1 microsecond. It costs an additional 1 microsecond to access a page other than the current one. Pages have 1,000 words, and the paging device is a drum that rotates at 3,000 revolutions per minute and transfers 1 million words per second. The following statistical measurements were obtained from the system:

- One 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.

Answer:

$$\begin{aligned}\text{effective access time} &= 0.99 \times (1\ \mu\text{sec} + 0.008 \times (2\ \mu\text{sec}) \\ &\quad + 0.002 \times (10{,}000\ \mu\text{sec} + 1{,}000\ \mu\text{sec}) \\ &\quad + 0.001 \times (10{,}000\ \mu\text{sec} + 1{,}000\ \mu\text{sec}) \\ &= (0.99 + 0.016 + 22.0 + 11.0)\ \mu\text{sec} \\ &= 34.0\ \mu\text{sec}\end{aligned}$$

10.5 Consider the page table for a system with 12-bit virtual and physical addresses and 256-byte pages.

Page	Page Frame
0	–
1	2
2	C
3	A
4	–
5	4
6	3
7	–
8	B
9	0

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). A dash for a page frame indicates that the page is not in memory.

Convert the following virtual addresses to their equivalent physical addresses in hexadecimal. All numbers are given in hexadecimal.

- 9EF
- 111
- 700
- 0FF

Answer:

- 9*EF* → 0*EF*
- 111 → 211
- 700 → *D*00
- 0*FF* → *EFF*

10.6 Discuss the hardware functions required to support demand paging.

Answer:

For every memory-access operation, the page table must be consulted to check whether the corresponding page is resident and whether the program has read or write privileges for accessing the page. These checks must be performed in hardware. A TLB could serve as a cache and improve the performance of the lookup operation.

10.7 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? Use LRU replacement, and assume

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;
```

Answer:

a. 5,000

b. 50

10.8 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, and seven frames? Remember that all frames are initially empty, so your first unique pages will cost one fault each.

- LRU replacement
- FIFO replacement
- Optimal replacement

Answer:

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

10.9 Consider the following page reference string:

7, 2, 3, 1, 2, 5, 3, 4, 6, 7, 7, 1, 0, 5, 4, 6, 2, 3, 0 , 1.

Assuming demand paging with three frames, how many page faults would occur for the following replacement algorithms?

- LRU replacement
- FIFO replacement

- Optimal replacement

Answer:

- 18
- 17
- 13

10.10 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.

Answer:
You can use the valid/invalid bit supported in hardware to simulate the reference bit. Initially set the bit to invalid. On first reference, a trap to the operating system is generated. The operating system will set a software bit to 1 and reset the valid/invalid bit to valid.

10.11 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.

Answer:
No. An optimal algorithm will not suffer from Belady's anomaly because—by definition—an optimal algorithm replaces the page that will not be used for the longest time. Belady's anomaly occurs when a page-replacement algorithm evicts a page that will be needed in the immediate future. An optimal algorithm would not have selected such a page.

10.12 Segmentation is similar to paging but uses variable-sized "pages." Define two segment-replacement algorithms, one based on the FIFO page-replacement scheme and the other on the LRU page-replacement scheme. Remember that since segments are not the same size, the segment that is chosen for replacement may be too small to leave enough consecutive locations for the needed segment. Consider strategies for systems where segments cannot be relocated and strategies for systems where they can.

Answer:

a. **FIFO**. Find the first segment large enough to accommodate the incoming segment. If relocation is not possible and no one segment is large enough, select a combination of segments whose memories are contiguous, which are "closest to the first of the list," and which can accommodate the new segment. If relocation is possible, rearrange the memory so that the first *N* segments large enough for the incoming segment are contiguous in memory. Add any leftover space to the free-space list in both cases.

b. **LRU**. Select the segment that has not been used for the longest time and that is large enough, adding any leftover space to the free-space list. If no one segment is large enough, and if relocation is not available, select a combination of the "oldest" segments that are contiguous in memory and are large enough. If relocation is available, rearrange the oldest *N* segments to be contiguous in memory and replace those with the new segment.

10.13 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 the CPU and the paging disk. Three alternative results are shown below. For each case, what is happening? Can the degree of multiprogramming be increased to increase the CPU utilization? Is the paging helping?

a. CPU utilization 13 percent; disk utilization 97 percent

b. CPU utilization 87 percent; disk utilization 3 percent

c. CPU utilization 13 percent; disk utilization 3 percent

Answer:

a. Thrashing is occurring.

b. CPU utilization is sufficiently high to leave things alone and increase the degree of multiprogramming.

c. Increase the degree of multiprogramming.

10.14 We have an operating system for a machine that uses base and limit registers, but we have modified the machine to provide a page table. Can the page table be set up to simulate base and limit registers? How can it be, or why can it not be?

Answer:

The page table can be set up to simulate base and limit registers provided that the memory is allocated in fixed-size segments. The base of a segment can be entered into the page table and the valid/invalid bit used to indicate that portion of the segment as resident in the memory. There will be some problem with internal fragmentation.