Operating Systems: Quiz 2

1. Explain the difference between internal and external fragmentation.

Answer:

Internal Fragmentation is the area in a region or a page that is not used by the job occupying that region or page. This space is unavailable for use by the system until that job is finished and the page or region is released.

2. Explain the difference between paging and segmentation.

	Paging	Segmentation
Length	Fixed length	Variable length
Address space	One dimensional address	Two dimensional address
DAT	Page # → frame #	Segment # + offset → base address + offset
Protection	Not easy	Good
Sharing	Not easy	Good
Fragment	Internal fragment	External fragment
Linking	Static linking	Dynamic linking
Loading	Dynamic loading	Dynamic loading

3. Given memory partitions of 100K, 500K, 200K, 300K, and 600K (in order), how would each of the First-fit, Best-fit, and Worst-fit algorithms place processes of 212K, 417K, 112K, and 426K (in order)? Which algorithm makes the most efficient use of memory?

Answer:

First-fit:

1. 212K is put in 500K partition
2. 417K is put in 600K partition
3. 112K is put in 288K partition (new partition 288K = 500K - 212K)
4. 426K must wait

Next-fit

1. 212K is put in 500K partition
2. 417K is put in 600K partition
3. 112K is put in 183K partition (new partition 183K = 600K - 417K)
4. 426K must wait

Best-fit:

1. 212K is put in 300K partition
2. 417K is put in 500K partition
3. 112K is put in 200K partition
4. 426K is put in 600K partition

Worst-fit:

1. 212K is put in 600K partition

2. 417K is put in 500K partition
3. 112K is put in 388K partition
4. 426K must wait

In this example, Best-fit turns out to be the best.

4. Why are page sizes always powers of 2?

Answer:

Recall that paging is implemented by breaking up an address into a page and offset number. It is most efficient to break the address into X page bits and Y offset bits, rather than perform arithmetic on the address to calculate the page number and offset. Because each bit position represents a power of 2, splitting an address between bits results in a page size that is a power of 2.

5. Consider a logical address space of eight pages of 1024 words each, mapped onto a physical memory of 32 frames.

A. How many bits are there in the logical address?

B. How many bits are there in the physical address?

Answer:

A. Logical address: 13 bits
B. Physical address: 15 bits

6. Assume a page reference string for a process with m frames (initially all empty). The page reference string has length p with n distinct page numbers occurring in it. 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

7. Which of the following programming techniques and structures are "good" for a demand-paged environment? Which are "not good"? Explain your answers.

A. Stack
B. Hashed symbol table
C. Sequential search
D. Binary search
E. Pure code
F. Indirection

Answer:

A. Stack—good.

B. Hashed symbol table—not good.
C. Sequential search—good.
D. Binary search—not good.
E. Pure code—good.
F. Indirection—not good.

8. Consider a demand-paging system with the following time-measured utilizations:

CPU utilization	**20%**
Paging disk	**97.7%**
Other I/O devices	**5%**

For each of the following, say whether it will (or is likely to) improve CPU utilization? Explain your answer answers.

A. Install a faster CPU.
B. Install a bigger paging disk.
C. Increase the degree of multiprogramming.
D. Decrease the degree of multiprogramming.
E. Install more main memory.
F. Install a faster hard disk.
G. Increase the page size.

Answer:

The system obviously is spending most of its time paging, indicating over-allocation of memory. If the level of multiprogramming is reduced resident processes would page fault less frequently and the CPU utilization would improve. Another way to improve performance would be to get more physical memory or a faster paging drum.

A. Get a faster CPU—No.
B. Get a bigger paging drum—No.
C. Increase the degree of multiprogramming—No.
D. Decrease the degree of multiprogramming—Yes.
E. Install more main memory—Likely to improve CPU utilization as more pages can remain resident and not require paging to or from the disks.
F. Install a faster hard disk—Also an improvement, for as the disk bottleneck is removed by faster response and more throughput to the disks, the CPU will get more data more quickly.
G. Increase the page size—Increasing the page size will result in fewer page faults if data is being accessed sequentially. If data access is more or less random, more paging action could ensue because fewer pages can be kept in memory and more data is transferred per page fault. So this change is as likely to decrease utilization as it is to increase it.

9. 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

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. What is the cause of thrashing? How does the system detect thrashing? Once it detects thrashing, what can the system do to eliminate this problem?

Answer:

Thrashing is caused by under-allocation of the minimum number of pages required by a process, forcing it to continuously page fault. The system can detect thrashing by evaluating the level of CPU utilization as compared to the level of multiprogramming. It can be eliminated by reducing the level of multiprogramming.