

**Fall 2009**  
**CSC 4103 – Operating Systems**  
**Homework Assignment #4**

The due date is: November 19<sup>th</sup>, Thursday, before the class.

**Problem 1:**

Consider a system where free space is kept in a free-space list.

- a. Suppose that the pointer to the free-space list is lost. Can the system reconstruct the free-space list? Explain your answer.
- b. Consider a file system similar to the one used by UNIX with indexed allocation. How many disk I/O operations might be required to read the contents of a small local file at `/a/b/c`? Assume that none of the disk blocks is currently being cached.

**Problem 2:**

Some file systems allow disk storage to be allocated at different levels of granularity. For instance, a file system could allocate 4 KB of disk space as a single 4-KB block or as eight 512-byte blocks. How could we take advantage of this flexibility to improve performance? What modifications would have to be made to the free-space management scheme in order to support this feature?

**Problem 3:**

Consider a file system on a disk that has both logical and physical block sizes of 512 bytes. Assume that the information about each file is already in memory. For each of the three allocation strategies (contiguous, linked, and indexed), answer these questions:

- a. How is the logical-to-physical address mapping accomplished in this system? (For the indexed allocation, assume that a file is always less than 512 blocks long.)
- b. If we are currently at logical block 10 (the last block accessed was block 10) and want to access logical block 4, how many physical blocks must be read from the disk?

**Problem 4:**

None of the disk-scheduling disciplines, except FCFS, is truly fair (starvation may occur).

- a. Explain why this assertion is true.
- b. Describe a way to modify algorithms such as SCAN to ensure fairness.
- c. Explain why fairness is an important goal in a time-sharing system.
- d. Give three or more examples of circumstances in which it is important that the operating system be *unfair* in serving I/O requests.

**Problem 5:**

Suppose that a disk drive has 5000 cylinders, numbered 0 to 4999. The drive is currently serving a request at cylinder 143, and the previous request was at cylinder 125. The queue of pending requests, in FIFO order, is

86, 1470, 913, 1774, 948, 1509, 1022, 1750, 130

Starting from the current head position, what is the total distance (in cylinders) that the disk arm moves to satisfy all the pending requests, for each of the following disk-scheduling algorithms?

- a. FCFS
- b. SSTF
- c. SCAN
- d. LOOK
- e. C-SCAN

**Problem 6:**

From elementary physics, we know that when an object is subjected to a constant acceleration  $a$ , the relationship between distance  $d$  and time  $t$  is given by  $d = \frac{1}{2} at^2$ . Suppose that, during a seek, the disk in the “*Problem 5 (above)*” accelerates the disk arm at a constant rate for the first half of the seek, then decelerates the disk arm at the same rate for the second half of the seek. Assume that the disk can perform a seek to an adjacent cylinder in 1 millisecond and a full-stroke seek over all 5000 cylinders in 18 milliseconds.

- a. The distance of a seek is the number of cylinders that the head moves. Explain why the seek time is proportional to the square root of the seek distance.
- b. Write an equation for the seek time as a function of the seek distance. This equation should be of the form  $t = x + y\sqrt{L}$ , where  $t$  is the time in milliseconds and  $L$  is the seek distance in cylinders.
- c. Calculate the total seek time for each of the schedules in “*Problem 5*”. Determine which schedule is the fastest (has the smallest total seek time).
- d. The *percentage speedup* is the time saved divided by the original time. What is the percentage speedup of the fastest schedule over FCFS?

**Problem 7:**

Consider a RAID Level 5 organization comprising five disks, with the parity for sets of four blocks on four disks stored on the fifth disk. How many blocks are accessed in order to perform the following?

- a. A write of one block of data
- b. A write of seven continuous blocks of data