CSI 4500 Homework #5 – File system

Partial solution

Question 1. For each part of this question, assume that the disk has a total of 100 cylinders.

a. Suppose that the disk head scheduling policy is shortest seek time first (SSTF). The disk heads are currently positioned over cylinder 42 and requests are queue for data on cylinders

27, 40, 90, 37, 15, 67

What will be the total distance (in cylinders) traveled by the disk heads to serve the six queued requests?

The travel sequence would be 42 -> 40 -> 37 -> 27 -> 15 -> 67 -> 90, so the total distance is 42-15 + 90 – 15 = 102.

b. Suppose instead that the disk head scheduling policy is SCAN (elevator). The disk heads are initially positioned over cylinder 42 and the current direction of travel of the heads is "up", i.e., towards higher numbered cylinders. For the same set of queued requests given in part (a), what will be the total distance (in cylinders) traveled by the disk heads to service the requests?

The travel sequence would be 42 -> 67 -> 90 -> 99 -> 40 -> 37 -> 27, so the total travel distance is 99 – 42 + 99 – 27 = 129.

Question 2. Consider a File system that maintains unique index node for each file in the system. Each index node includes 8 direct pointers, a single indirect pointer, and a double indirect pointer. The file system block size is 1024 bytes, and a block pointer occupies 4 bytes.

Since the block size is 1024 bytes, and a pointer would use 4 bytes, which mean if a block is used to stored pointers then it would be able store 1024 bytes / 4 bytes = 256 pointers. Keep it in mind, this information would be used in the following questions.

a) What is the maximum file size that can be supported by the index node?

Maximum file size means the index node is fully utilized:

1) **8 directed pointers are used to link 8 data blocks**
2) **one single indirect pointer points to an index block (equal size of a data block) of 256 pointers, which are used to point 256 data blocks [corresponding to the 1+256 part in the final answer]**
3) **one double indirect pointer points to an index block (equal size of a data block) has 256 pointers, each of which pointers an second level index block (size of 1KB) and has 256 pointers, which are used to pointer 256 data blocks.**

So the double indirect block could link 256 * 256 blocks [corresponding to the last part in the final answer].

So the maximum file size, would have 8 + (1 + 256) + (1 + 256 + 256*256) blocks.

We also need to note that a file size reflects both the data size and the overhead (index blocked used in managing data blocks or other index blocks).

b) How many disk operations will be required if a process read data from the N^{th} block of a file? Assume that the file is already open, the buffer cache is empty, and each disk operation read a single file block. Your answer should be given in terms of N.

For the data blocks directly managed by the index node, the number of I/O operations would be 1, so when 1=< N <= 8, the number of IO ops is 1.

For the data blocks managed by the single indirect pointer, the first IO operation would read the index block into the memory, then the 2nd IO operation would bring the target block into memory. So when when 9=< N < 256 + 8, 2 IO operations would be required to read the data from the Nth block.

Use the same logic, when 256+8< N <= 8 + 256 + 256*256, there would be 3 IO operations.

b) How much space (including overhead) does a file that is 1025 bytes long?

If a file has 1025 bytes data, then there would be two data blocks with the first one has 1024 bytes, and the 1st byte of the second data block would still be used for this file. So the total space for this file would be include i-node and 2 data blocks.

d) How much space (including overhead) does a file that is 65536 (64KB) bytes long?

For a file to support 64KB data, the file system would manage 64KB/1KB = 64 data blocks, which means for the i-node, the 8 directed pointers would be used to support the 8 data blocks, and the single indirected pointer would be used point to a block serving as index block to manage the 56 (= 64 -8) data blocks (in another word, only the first 56 pointers are assigned values to point to the data blocks). So the total space for this file would be i-node, one index block + 64 data blocks.

Question 3. Suppose a file system can have three disk allocation strategies, contiguous, linked, and indexed. We have just read the information for a file from its parent directory. For contiguous and linked allocation, this gives the address of the first block, and for indexed allocation this gives the address of the index block. Now we want to read the 10^{th} data block into the memory. How many disk blocks (*N*) do we have to read for each of the allocation strategies? *For partial credit, explicitly list which block(s) you have to read.*

Contiguous allocation, it supports random access, so 1 disk block would be read.
Linked allocation, it supports sequential access, so 10 disk blocks would be read.
Indexed allocation, it needs to read the index block first, then the 10th data block, so two disk blocks would be read.