1

Suppose that a disk drive has 5,000 cylinders, numbered 0 to 4,999. The drive is currently serving a request at cylinder 2,150, and the previous request was at cylinder 1,805. The queue of pending requests, in FIFO order, is:
2,069, 1,212, 2,296, 2,800, 544, 1,618, 356, 1,523, 4,965, 3681
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 diskscheduling algorithms?

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

Solution:

Answer: The following table shows the access order of tracks and the total number of cylinders crossed.

Method	1	2	3	4	5	6	7	8	9	10	11	12	13	Total
FCFS	2150	2069	1212	2296	2800	544	1618	356	1523	4965	3681			13011
SSTF	2150	2069	2296	2800	3681	4965	1618	1523	1212	544	356			7586
SCAN	2150	2296	2800	3681	4965	4999	2069	1618	1523	1212	544	356		7492
C-SCAN	2150	2296	2800	3681	4965	4999	0	356	544	1212	1523	1618	2069	9917
LOOK	2150	2296	2800	3681	4965	2069	1618	1523	1212	544	356			7427
C-LOOK	2150	2296	2800	3681	4965	356	544	1212	1523	1618	2069			9137

2

Contrast the performance of the three techniques for allocating disk blocks(contiguous, linked, and indexed) for both sequential and random file access.

Solution:

There are three main disk space or file allocation methods.

- Contiguous Allocation
- Linked Allocation
- Indexed Allocation

By the performance level
Contiguous allocation > linked Allocation > Indexed Allocation

Contiguous is faster than linked allocation and both contiguous and linked are faster than indexed. As linked allocation does not support direct access and the indexed allocation, the pointer of it is bigger than linked allocation.

Explanation:
1. Contiguous Allocation :
In this scheme, each file occupies a contiguous set of blocks on the disk.

- It supports both accesses (sequential and direct)
- It's extremely fast as the number of seeks are minimal because of contiguous allocation of file blocks. but on the other hand it has some disadvantages as
- It can not deal with fragmentation whether its external or internal.
- And also increasing the size of it may get difficult.

2. Linked Allocation:
In this type of allocation, each file is a linked list of disk blocks and the disk blocks can be scattered anywhere on the disk.

- This type is very flexible with the file size.
- This method also does not suffer from external fragmentation. This makes it better in terms of memory utilization.
- It does not support random or direct access while it slightly support sequential access.

3. Indexed Allocation:
There is a special block known as the Index block contains the pointers to all the blocks occupied by a file in this type of allocation. Each file has its own index block.

- This type supports direct therefore provides fast access to the file blocks.
- It also overcomes with the problem of external fragmentation.

3
Consider a file system that uses inodes to represent files. Disk blocks are 8 KB in size,and a pointer to a disk block requires 4 bytes. This file system has 12 direct disk blocks, as well as single, double, and triple indirect disk blocks. What is the maximum size of a file that can be stored in this file system?

Solution:

Let F1 be the old file and F2 be the new file. A user wishing to access F1 through an existing link will access F2.Note that the access protection for file F1 is used rather than me one associated with F2.This problem can be avoided by insuring that all links to a deleted file are deleted also.This can be accomplished in several ways:
a. maintain a list of all Inks to a file, removing each of them when the file is deleted.
b. retain the links, removing them when an attempt is made to access a deleted file.
c. maintain a file reference list (or counter), deleting the file only after all links or references to that file have been deleted.

Result:

Let F1 be the old file and F2 be the new file. A user wishing to access F1 through an existing link will actually access F2.