Worked out Examples

2.1. Consider a disk that is susceptible to developing bad sectors. Suggest a block based file organization technique that will make sure that no extra data (other than the corrupted block) is lost whenever a block gets corrupted. You should be able to obtain the whole file except for the corrupted block.

Ans: Maintain a combination of indexed allocation and linked list allocation so that whenever a block is corrupted, though the linked list chain is broken, the other blocks can be obtained from the index. Incase the index block itself is corrupted, the linked list can be used to retrieve the file and rebuild the index.

2.2. Suppose that a fast wide SCSI-II disk drive spins at 7200 RPM, has a sector size of 512 bytes, and holds 160 sectors per track.

a. Estimate the sustained transfer rate of this drive in megabytes per second.

b. Suppose that the drive has 7000 cylinders, 20 tracks per cylinder, a head switch time (from one platter to another) of 0.5 millisecond, and an adjacent cylinder seek time of 2 milliseconds. Use this additional information to give an accurate estimate of the sustained transfer rate for a huge transfer.

Ans:

a. The disk spins 120 times per second, and each spin transfers a track of 80 KB. Thus, the sustained transfer rate can be approximated as 9600 KB/s.

b. Suppose that 100 cylinders is a huge transfer. The transfer rate is total bytes divided by total time. Bytes: 100 cyl * 20 trk/cyl * 80 KB/trk, i.e., 160,000 KB. Time: rotation time + track switch time + cylinder switch time. Rotation time is 2000 trks/120 trks per sec, i.e., 16.667 s. Track switch time is 19 switch per cyl * 100 cyl * 0.5 ms, i.e., 950 ms. Cylinder switch time is 99 * 2 ms, i.e., 198 ms. Thus, the total time is 16.667 + 0.950 + 0.198, i.e., 17.815 s. (We are ignoring any initial seek and rotational latency, which might add about 12 ms to the schedule, i.e. 0.1%.) Thus the transfer rate is 8981.2 KB/s.

2.3. Suppose we use fixed blocking for a file with 10 byte logical records and 25 byte physical blocks. How much space do we waste per block? Does this result in internal or external fragmentation or both?

Ans: According to the problem only 10 bytes are used although the available space is 25 bytes. Therefore, the space wasted per block is 15 bytes.

Therefore, it results in *internal* fragmentation. But, if this block's data is deleted or transferred to other location then this block remains unutilized and it leads to external fragmentation.

2.4. Suppose in place of fixed logical records, we have variable logical records of lengths of 10 bytes, 20 bytes, and 30 bytes. Our physical block continues to be 25 bytes as in the previous exercise. What maximum and minimum fragmentation do we see in this case?

Ans: The **maximum** fragmentation that is observed is in the case of record length of 30 bytes and it is of **20 bytes.** This is explained as follows. Suppose we have a logical record of 30 bytes. Then, since the physical block length is only 25 bytes so we will require 2 physical blocks to store data. One block will be fully occupied and for the second block only 5 bytes will be used, resulting in unused 20 bytes.

25 bytes 5 bytes

As it is clear from the figure that if one has to incorporate 30 byte then 20 bytes has to be unused.

Note: The **blue** color shows the used portion and the **brown** portion shows the unused portion.

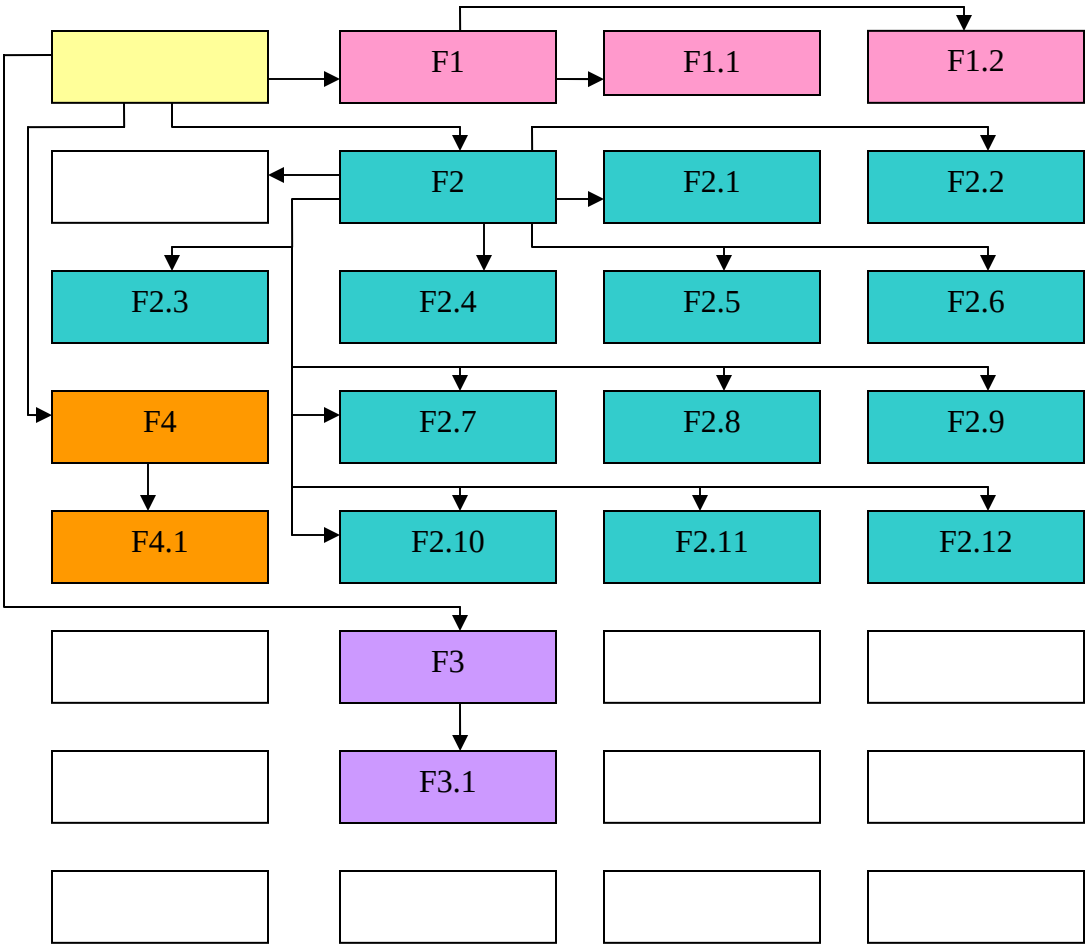
The minimum fragmentation that is observed in this case is **5 bytes** for the record length of 20 bytes**. Although in case of ideal utilization minimum is zero which is also there in the present design**. The values available are **10**, **20**, **30**, and **25**. The LCM of all these is 5. Therefore 5 byte is the obvious answer.

2.5. Using a diagram, show how an indexed allocation of a file may be done for a disked based system with the following characteristics. The disc size is 30

blocks each of 1024 bytes (may be modeled as 6 X 5 matrix). File f1 is 11 logical records of 112 bytes, file f2 is 890 logical records of 13 bytes, file f3 is 510 bytes of binary data stream and file f4 is 4 logical blocks of 95 bytes.

Ans: See the figure below.

Here, each block is of 1024 bytes.



- File f1 is 11 logical records each of 112 bytes; that means a total of 11 * 112 bytes = 1232 bytes. This implies it needs (1232 / 1024) ~ 2 blocks
- File f2 is 890 logical records each of 13 bytes; that means a total of 13 * 890 bytes = 11570 bytes. This implies it needs (11570 / 1024) ~ 12 blocks.
- File f3 is 570 bytes of binary data stream; that means it needs 1 block.
- File f4 is 4 blocks each of 95 bytes; that means a total of 4 * 95 bytes = 380 bytes.

This implies it needs 1 block.

2.6. We have a disc that has 10 tracks per platter with 8 sectors. The drive supports 10 writable platters on a common spindle. Each sector stores four 512 byte blocks. There is a read write head for every platter. The heads can be switched in 1ms. The track traversal is sequential and is at the rate of 10ms per track. Now attempt the following:

a. Draw a small figure to show how a 7.5 KB file could be stored ideally.

b. What is the time of retrieval for the file, assuming that (i) the head needs to be switched, (ii) the track needs to be traversed half way to retrieve the first sector of this file.

c. Suppose we decide to fill up this disk with 10 files of 1.1 KB, 20 files of 2.2KB, 30 files of 3.3 KB, and so on. Finally, how many files will be there in the disk? Also, what will be the internal and external fragmentations when the disk is filled up?

Ans:

There are 8 sectors/track

There are 10 tracks/ platter

Tswitch = 1ms

Ttraverse = 10ms/track

1 sectors = 4 blocks of 512 bytes

Total size of disk = 1638.4 KB

a. File of size 7.5 KB. Ideally it can be stored on 4 blocks of 512 bytes. That is on 4 consecutive sectors.

b. Time to access this file when it is stored on separate platters in 4 sectors will be –(10/8) *4 + 3 = 8 msTime to access this file when it is stored sequentially on a single platter will be –(1.25)/2 + 3*(1.25) = 4.375 ms

c. 10 files of 1.1.KB = 11KB

20 files of 2.2 KB = 44 KB

30 files of 3.3 KB = 99 KB

40 files of 4.4 KB = 176 KB

50 files of 5.5 KB = 275 KB

60 files of 6.6 KB =396 KB

70 files of 7.7 KB = 539 KB

Total number of files = 280

Total file size is 1540 KB

Total space on disk = 1638.4 KB

External fragmentation = 1638.4 – 1540 = 98.4 KB

Internal fragmentation =

For 1.1 KB files = (512*3) – 1.1 = 436 bytes, so total fragmentation for 10 files will be 4360 bytes.

Similarly for 20 files of 2.2 KB = 7200 bytes

For 30 files of 3.3 KB = 8530 bytes

For 40 files of 4.4. KB = 8320 bytes

For 50 files of 5.5 KB = 6600 bytes

For 60 files of 6.6 KB = 3360 bytes

For 70 files of 7.7 KB = 38440 bytes

2.7.What is a file system?

Ans: A file system is that software which allows users and applications to organize and manage their files. The organization of information may involve access, updates and movement of information between devices.

2.8.What is the information required for management of files?

Ans:

Nature of information	Significance	Uses in management
File name	Chosen by its creator	To check its uniqueness within a dir
File type	Text, Binary, Programme etc.	To check its correct usage
Date of creation and last usage	Time and Date	Useful for recording identity of users

2.9.Explain Inode Structure in UNIX?

Ans:

Item	Description
File type:	16 bit information Bits 14-12:filetypes Bits 11-9:execution flags Bits 8-6: owners rwx Bits 5-3: groups rwx Bits 2-0 : others rwx
Link count:	number of symbolic references to this file
Owners ID:	individual loginID who owns the file
File size :	Expressed in number of bytes
File address:	39 bytes of addressing format

2.10. What are different block based file organizations?

Ans:

1. File storage allocation policy
2. Chained list allocation
3. Indexed allocation

2.11. What is internal fragmentation?

Ans: The two blocks together having 2048 bytes capacity, we will fill suppose the first block completely and second block partially. Thus second remains mostly empty. Such non-utilization of space caused internally is termed as internal fragmentation.

2.12. What is external fragmentation?

Ans: Suppose we wish to insert moderately large sized file thinking with adequate space. Then it may happen that the free space list has shrunk so much that enough space is not available. This may be because there are many unutilized holes in the disk. Such non-utilization, which is outside of file space is regarded as external fragmentation.

2.13. Explain header of information storage organization on disks.

Ans:

Preamble: 25 bytes

Header: 8 bytes

Sync: 1 byte

Preamble: 25 bytes

Sync: 1 byte

Data bytes: 512 bytes

ECC: 6 bytes

Postamble: 22 bytes

2.14. What are portable storage mediums?

Ans: External medium storage devices like tapes, disks and floppies. These can be physically ported. Most file systems recognize these as online files when these are mounted on an I/O device like a tape drive or a floppy drive

2.15. What is root file system?

Ans: This file system is used for the organization and management for the files in the system. When OS is installed initially it creates a root file system, OS ensures about how the user and system files would be distributed for space allocations on the disk storage, this root is created as the initial directory which would be used later for further expansion of the directory structure.

Under UNIX following subdirectories are used….

usr --contains shareable binaries

bin --contain executables

sbin --contains system binaries

lib -- contains library

etc --it contains host related files

mnt --contains the device mount information

tmp --contains temporary files

var --contains files which have variable data

yp --contains network information

2.16. What is the type of fragmentation associated with each of memory allocation policies i.e. with chained list allocation, indexed allocation, file storage allocation policies if any?

1. pre allocation policy-- internal fragmentation
2. chained list allocation—external fragmentation

3. indexed allocation --- external fragmentation

2.17. What is inode in UNIX?

All file management operations in UNIX are maintained and controlled by information in inode structure. Typically it offers all information about acess rights .file size, its date of creation usage and modifications.

Inode structure:

File type	16 bit information Bits 14-12: file type(ordinary ,directory etc) 11-9: execution flags 9-6: owners rwx information 5-3: groups rwx information 2-0: other's rwx information
Link account	number of symbolic references to this file
Owner id	login id of person who owns the file
Group id	group id of the users
File size	expressed in number of bytes
File and address	39 bytes of addressing information
Last access to file	Date and time of last access
Last modified	Date and time of last modification
Last inode modification	Date and time of last inode modification