IT5002 Computer Systems and Applications

Tutorial 9

1. We consider a simple file system illustrated below:

Partition Information (Free Space Information)

- Bitmap is used, with a total of 32 file data blocks (1 = Free, 0 = Occupied):

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	1
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
1	0	1	0	1	0	1	0	1	1	1	0	0	1	0	0

Directory Structure + File Information:

- Directory structures are stored in 4 "directory" blocks. Directory entries (both files and subdirectories) of a directory are stored in a single directory block.
- Directory entry:
  - **For File:** Indicates the first and last data block number.
  - **For Subdirectory:** Indicates the directory structure block number that contains the subdirectory's directory entries.
  - The "**/**" root directory has the directory block number 0.

0	1	2	3
y |Dir | 3 f |File| 12| 2 x |Dir | 1	g |File| 0| 31 z |Dir | 2	k |File| 6| 6	i |File| 1| 3 h |File|27|28

File Data:

- Linked list allocation is used. The first value in the data block is the "next" block pointer, with "-1" to indicate the end of data block.
- Each data block is 1 KB. For simplicity, we show only a couple of letters/numbers in each block.

0	1	2	3	4	5	6	7
11 AL	9 TH	-1 S!	-1 ND	23 GS	-1 SO	-1 :)	10 TE
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
31 RE	3 EE	28 M:	31 OH	19 SE	13 AH	4 IN	17 NO
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
30 YE	2 OU	1 ON	17 RI	26 EV	14 AT	21 DA	7 YS
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
-1 HO	18 ME	0 AL	30 OP	-1 -(	5 LO	21 ER	-1 A!

a. (Basic Info) Give:
   - The current free capacity of the disk.
   - The current user view of the directory structure.

b. (File Paths) Walkthrough the file path checking for:
   - "**`/y/i`**"
   - "**`/x/z/i`**"

c. (File access) Access the entire content for the following files:
   - "**`/x/z/k`**"
   - "**`/y/h`**"

d. (Create file) Add a new file "**`/y/n`**" with 5 blocks of content. You can assume we always use the free block with the smallest block number. Indicate **all changes required to add the file.**

2. Let us compare and contrast the various ways of implementing files in this question.

Below are the hardware parameters:

a. Number of disk blocks = 2^{16} = 65,536 blocks (i.e. each disk block number = 2 bytes).
b. Disk block size = 512 bytes.

Points of comparison:

- **Total Number of Data Blocks:** Number of disk blocks needed to store the file data.
- **Overhead**: Extra bookkeeping information in bytes. Focus only on information directly related to keep track of file data. You can ignore the space needed for file name and other metadata for this question.
- **Worst Case Disk Access:** The worst case number of disk accesses needed to get to a particular block in the file. May include accessing book keeping information if they are in disk. Specify the block which causes the worst case.

File data allocation schemes under consideration:

A. Contiguous
B. Linked list
C. File allocation table
D. Index block

Fill in the following table for the indicated file size. State assumptions for your calculation (if any).

File size: 100 b	Total Number of Data Blocks	Overhead	Worst Case Disk Access
A			
B			
C			
D			

File size: 132,000 b	Total Number of Data Blocks	Overhead	Worst Case Disk Access
A			
B			
C			
D			

File size: 33,554,432 b	Total Number of Data Blocks	Overhead	Worst Case Disk Access
A			
B			
C			
D			

3. We are given the following table of contents in an inode. Suppose that each data block holds 2048 bytes of data. List down the block numbers that would be read/written by the following operations. Assume that fp points to a validly open file with this TOC.

Table of Contents
15
18
12
22