Introduction to Operating Systems and SQL for Data Science

Practice 5 – Files

Files implementation

- Consecutive allocation
- Linked list
- FAT
- i-node

Reichman University

Consecutive allocation

- Files allocated consecutive in disk. Each file gets several blocks.
  Data stored in consecutive blocks.
- **Advantages:**
  - Easy to implement
  - When reading all file to memory the read/write head of the disk doesn't move a lot (faster)
  - Random access – we can approach the file in any position directly. Reading block number N requires 1 reading operation.
- **Disadvantages**
  - Not supporting file changes

Reichman University

Linked list allocation

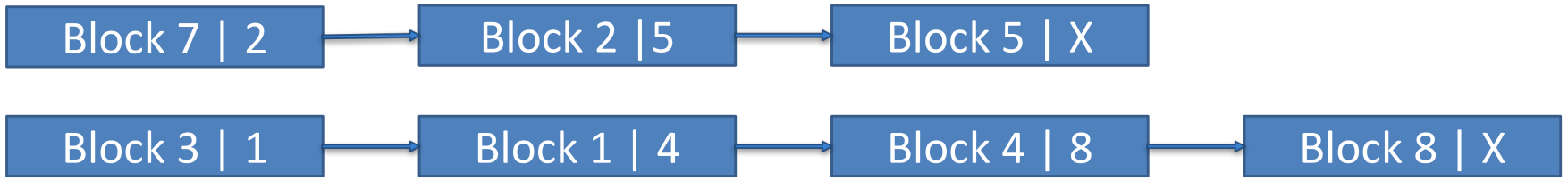
- Each file has a linked list of blocks, the blocks are not consecutive in the disk, each blocks points to the next block.
- **Advantages:**
  - Extending the file (add content) does not require any movement of the information
- **Disadvantages**
  - Reading block number N require N reading operations
  - Each blocks contain several bytes which are not related to the file's data, They contain the pointer to the next block.
  - Long access time – when blocks are on different tracks of the disk the read/write head of the disk moves a lot – takes time.
  - If one block is corrupted or lost, we lose the rest of the file (don't know where the next block is)

Reichman University

FAT = FILE ALLOCATION TABLE

- Separate file's structure (RAM) and file's data (Disk)
- Each block would have an entry in a table, the entry would give us the address of the next block of the file.
- X means this is the last block of the file.
- Tables saved as an array in the memory
- Reading block number N of a file is faster than in linked list since search is done in memory instead of in disk.

Block Number	Address of the next block
8	X
7	2
6	
5	X
4	8
3	1
2	5
1	4
0	



Reichman University

Some math

- If Cell size is k bits => there are max of 2^k blocks
- Number of blocks = $\frac{Disk\ Size}{Block\ Size}$
- A reminder:
  - 8 bit = 1 Byte
  - 1 KB = 2^{10} Byte
  - 1 MB = 2^{20} Byte
  - 1 GB = 2^{30} Byte
  - 1 TB = 2^{40} Byte

Question #1

- Disk size is 2GB
- Block size is 1KB
- Word size is 1 byte
- What is the size of a FAT entry?
- What is the total size of the table?

Question #1

Solution:

- Let's calculate first how many blocks there are on disk: $\frac{2^{31}}{2^{10}} = 2^{21}$
- There are 2^{21} blocks, so we will need 21 bits to represent a block index.
- $RoundUp(\frac{21}{8}) = 3$ so we'll need 3 words to represent block index => 3 words size entry point.
- Total FAT size is $3 * 2^{21} = 6 * 2^{20} = 6MB$

Question #2

- Disk size is 32GB Block size is 1MB Word size is 4 bytes
- What is the size of a FAT table?

Reichman University

Question #2

- Disk size is 32GB Block size is 1MB Word size is 4 bytes
- What is the size of a FAT table?

Solution:

- Number of blocks: $\frac{2^{35}}{2^{20}} = 2^{15}$
- There are 2^{15} blocks, so we will need 15 bits to represent a block index.
- $RoundUp\left(\frac{14}{32}\right) = 1$ so we'll need 1 word to represent block index => size of entry point is 1 word = 5 bytes.
- Total FAT size is $4 * 2^{15} = 128 * 2^{10} = 128KB$

Reichman University

i-node

Each i-node block contains:

- File’s meta data (Date created, last modified, size, read/write permissions, ownership, etc.)
- Pointers (addresses) to file’s blocks (ordered)

Reichman University

i-node – pointing methods

1. **Direct:** the pointer points to the next block of the file.
   A direct pointer points to one block of the file.
2. **Single indirect:** the pointer points to another block which contains a list of direct pointers.

   Given that an empty block can contain N pointers, a single indirect pointer can map to N blocks of the file.
3. **Double indirect:** the pointer points to another block which contains a list of single indirect pointers.

   Given that an empty block can contain N pointers, a single indirect pointer can map to N^2 blocks of the file.
4. **Triple indirect:** the pointer points to another block which contains a list of double indirect pointers.

   Given that an empty block can contain N pointers, a single indirect pointer can map to N^3 blocks of the file.

Question #3

- Given a system with i-node file management, block size is 8 words, word size is 1 byte.
- 4 first bytes are for file's metadata, there are 3 direct pointers and one single-indirect pointer.
- Given a table with the first blocks in the disk. Block #0 is the I-node of the file
- Write a list of the information blocks belong to that file (by order)

metadata direct Single- indirect

Block 0	1	2	7	4	6	3	8	5
Block 1	4	5	8	9	13	32	84	56
Block 2	1	7	19	3	6	8	12	30
Block 3	4	5	7	10	12	14	18	20
Block 4	35	34	33	32	31	30	29	10
Block 5	22	11	18	16	2	32	1	7
Block 6	16	0	0	13	22	32	8	1
Block 7	59	85	36	2	4	6	8	10
Block 8	2	4	6	8	10	12	14	16

Reichman University

Question #3

- Given a system with i-node file management, block size is 8 words, word size is 1 byte.
- 4 first bytes are for file's metadata, there are 3 direct pointers and one single-indirect pointer.
- Given a table with the first blocks in the disk. Block #0 is the I-node of the file
- Write a list of the information blocks belong to that file (by order)

metadata direct Single- indirect

Block 0	1	2	7	4	6	3	8	5
Block 1	4	5	8	9	13	32	84	56
Block 2	1	7	19	3	6	8	12	30
Block 3	4	5	7	10	12	14	18	20
Block 4	35	34	33	32	31	30	29	10
Block 5	22	11	18	16	2	32	1	7
Block 6	16	0	0	13	22	32	8	1
Block 7	59	85	36	2	4	6	8	10
Block 8	2	4	6	8	10	12	14	16

Solution:

6,3,8,22,11,18,16,2,32,1,7

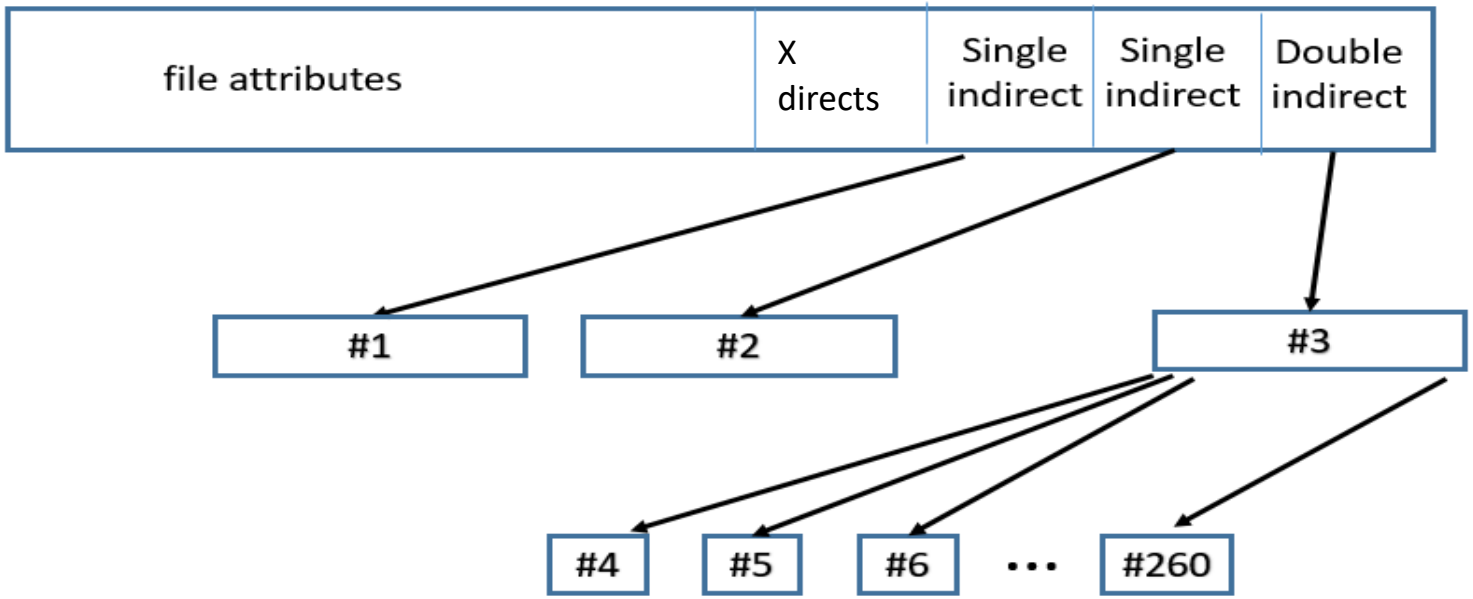
Reichman University

Question #4

- Given a system with i-node file management
- Block size is 1KB, word size is 2 bytes. 256 words are needed for file's metadata.
- The system supports disk with up to 1TB memory size
- In the i-node there are 2 single-indirect pointers, one double-indirect pointer, and the rest are direct pointers.
- For a maximum size file- how many blocks needed to represent file's structure?

Reichman University

Question #4



Reichman University

Solution

- Number of blocks: $\frac{2^{40}}{2^{10}} = 2^{30}$ => pointer size Is 4 bytes.
- How many pointers in empty block? $\frac{2^{10}}{2^{2}} = 2^{8} = 256$
- i-node + 1 single +2 double = 1+1+2+256
- Total = 260

Reichman University

Question #5

- Given a system with i-node file management
- Block size is 8KB, word size is 2 bytes. 2KB are needed for file's metadata.
- In the i-node there are 256 single-indirect pointers, 128 double-indirect pointers, and the rest are direct pointers.
- How much information can be saved in a file in this system?

Reichman University

Solution

- Number of blocks: $\frac{2^{40}}{2^{13}} = 2^{27}$ => pointer size requires 27 bits => 2 words => 4 bytes.
- How many pointers in empty block? $\frac{2^{13}}{2^{2}} = 2^{11}$
- 6KB for pointers => $\frac{6*2^{10}}{2^{2}} = 6 * 2^{8} = 1536$ pointers
- How many are direct? 1536-128-256=1152
- Hoy many data blocks? ? $1152 + 256 * 2^{11} + 128 * 2^{11*2} = 1152 + 2^{19} + 2^{29}$
- Block size is 8KB so total: $(1152 + 2^{19} + 2^{29})$*8KB

Reichman University