Disk Scheduling

SSTF

SCAN

C-SCAN
Persistence

I/O devices < interface implementation.

Disk

Rate (Seq. Read) >> Rate (Random Reads)

Disk Scheduling

1) FIFO
2) SSTF \[\text{Seek}\] \[\text{starration}\] \[\text{ignores}\] \[\text{rotation}\]
3) SPTF/SATF
4) BSATF (Windows)

C-SCAN

F-SCAN

SCAN (Elevator)
1. File System APIs
2. File System Implementation

Data Structures  Access Methods
File System APIs

Block I/O
- Device Drivers
- Devices

Appln
- FS

Open, Create, Read, Write, Close

Basic Abstractions

File
- Regular File
- array of bytes
- low-level name (inode #)
- human-readable name

Directory
- special type of file
- human-readable name
- low-level name
- array of records

<table>
<thead>
<tr>
<th>Human Name</th>
<th>Low-level Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>a.txt</td>
<td>1231271</td>
</tr>
<tr>
<td>dir1</td>
<td>1470712</td>
</tr>
</tbody>
</table>
Files

1) Creation
   open( )
   creat( )

OS

Process - opens a file
   file table
   stdin
   stdout
   stderr
   files

file descriptors

Current offset

F:

\[ A \quad B \quad C \quad D \quad E \]

\[ 1 \quad 2 \quad 3 \quad 4 \quad 5 \]

\[ \text{write}(fd, 3); \]
\[ \text{read}(fd, 3); \]
Directories

- special type of file
  - create, read, delete
  - mkdir

Write - NOT possible

```
ls
rm -rf /
```

```
/ foo
```

```
file

<table>
<thead>
<tr>
<th>H-N.</th>
<th>Inode #</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td>1731</td>
</tr>
<tr>
<td>..</td>
<td>2</td>
</tr>
<tr>
<td>a.txt</td>
<td>1812</td>
</tr>
<tr>
<td>b.txt</td>
<td>1012</td>
</tr>
</tbody>
</table>
```

```
open(1, foo/a.txt)
```

Absolute pathname: `/foo/a.txt`

Relative pathname: `a.txt`

- → current directory
  - → parent directory

1 / (root)
2 foo
3 a.txt
Hard Links

- file1
- file2

ABCDEF

1. Hard Links

  → ln file1 file2

<table>
<thead>
<tr>
<th>A. N°</th>
<th>inode#</th>
</tr>
</thead>
<tbody>
<tr>
<td>file1</td>
<td>123</td>
</tr>
<tr>
<td>file2</td>
<td>123</td>
</tr>
</tbody>
</table>

→ No hardlinks for dir.
Symbolic links (soft links)

```
$ ln -s file1 soft
```
File System Implementation

FS → array of block (Disk) → VSFS

<table>
<thead>
<tr>
<th>16 KB</th>
<th>8 KB</th>
<th>4 KB</th>
</tr>
</thead>
<tbody>
<tr>
<td>D</td>
<td>1 2</td>
<td>3 4</td>
</tr>
<tr>
<td></td>
<td>5 6</td>
<td>7 8</td>
</tr>
<tr>
<td></td>
<td>9 10</td>
<td>11 12</td>
</tr>
<tr>
<td></td>
<td>13 14</td>
<td></td>
</tr>
</tbody>
</table>

Disk

1 block = 4 KB (8 sectors)

1 sector = 512 bytes

Size of HDD = 64 x 4 KB

= 2^6 x 2^2 KB = 256 KB

1. How to store user data?

"DATA blocks"
How to store per-file metadata?

- **Inode**
  - size = 128 or 256 bytes
  - max # of files: 5
    - 1 block → \( \frac{4 \text{ KB}}{256} = \frac{2^2}{2^9} = \frac{2}{16} = 4 \) blocks
    - Data block 7/9
    - Data block 21
    - Data block 30

- max file size = 15 × 4 KB = 60 KB

- log. block 14
  - 5 blocks → 80 inodes
  - 80 files
Inode

metadata

log. block

max file size = 14 x 4 KB + 

\[ 1024 \times 1024 \times 4 \text{ KB} \]

\approx 4 \text{ GB} \]

Data block

double indirect ptr.

4 bytes \rightarrow \text{ptr to data block.}

\# of ptrs in indirect block

= \frac{4 \text{ KB}}{4} = 1 \text{ KB} = 1024

max file size = 14 x 4 KB + 1024 x 4 KB

= 4 \text{ MB}
FS → read block 8?

\[ 8 \times 4\text{ KB} = 32\text{ KB} \]

sector # = \( \frac{32\text{ KB}}{256} = \frac{2^{15}}{2^8} = 2^7 = 128 \)

= \( \boxed{64} \) sector #

FS → read inode #32?

offset from the inode start = \( 32 \times 256 = 2^5 \times 2^8 = 8\text{ KB} \)

inode start from the beg of device = \( 12\text{ KB} + 8\text{ KB} \)

= \( \boxed{20}\text{ KB} \)

sector # = \( \frac{20\text{ KB}}{2^9} = \frac{20 \times 2^{10}}{2^9} \)

= \( \boxed{40} \) sector #
3) Directories?
   - Reuse inodes - type = d

Data block

<table>
<thead>
<tr>
<th>H.N.</th>
<th>Inode #</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>123</td>
</tr>
<tr>
<td></td>
<td>2</td>
</tr>
<tr>
<td>a.txt</td>
<td>75</td>
</tr>
</tbody>
</table>

2) data of dir.

4) How to track free inodes & data blocks?

inode bitmap

```
| 0 | 1 | 2 | ... | 79 |
```

allocated: 1
free: 0

data bitmap

```
| 0 | 1 | 1 | 0 | ... | 1 |
```

allocated: 1
free: 0
Read

\[ / \text{foo/bar.txt} \rightarrow \text{data blocks} \]

1. Read inode of / (root)
2. Read data of / (root)
3. Read inode of foo
4. Read data of foo
5. Read inode of bar.txt
6. Read data[0] of bar.txt
7. Write inode of bar.txt

<table>
<thead>
<tr>
<th>HN</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>1234</td>
</tr>
</tbody>
</table>

bar.txt (189)
<table>
<thead>
<tr>
<th>ib</th>
<th>db</th>
<th>root</th>
<th>node</th>
<th>foo</th>
<th>node</th>
<th>bar</th>
<th>txt</th>
<th>root</th>
<th>data</th>
<th>foo</th>
<th>data</th>
<th>bear</th>
<th>data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>