National University of Singapore

CS2106 Operating System

Second Half Summary Notes

Dong Shaocong A0148008J

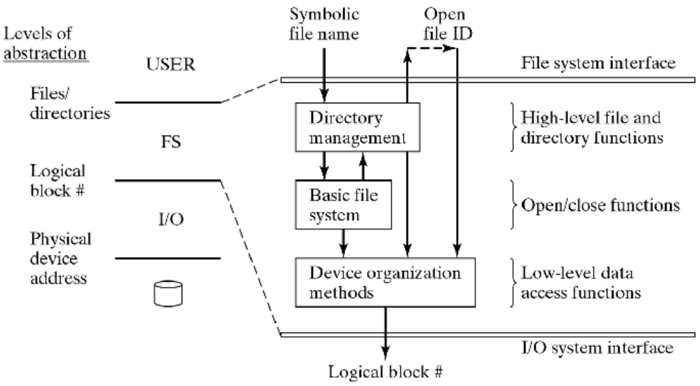
March 29, 2018

1 File System

Definition 1.1. File system

- Present logical (abstract) view of files and directories
  - Accessing a disk is very complicated: (2D or 3D structure, track/surface/sector, seek, rotation, ...)
  - Hide complexity of hardware devices
- Facilitate efficient use of storage devices: Optimise access e.g. to disk.
- Support sharing
  - Files persist even when owner/creator is not currently active (unlike main memory)
  - Key issue: Provide protection (control access)

Definition 1.2. Hierarchical View of File system



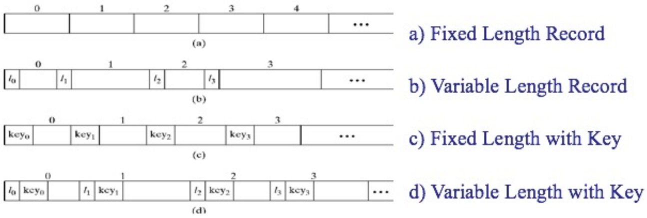
- **Directory management**: map logical name tounique Id, file descriptor
- **Basic file system**: open/close files
- **Physical device organization**: map file data to disk blocks

Definition 1.3. User-end view of File

- **File name and type**
  - Valid name: number or characters, lower or upper cases, illegal characters ...
  - Extension: tied to type of file, used by applications
  - File type is recorded in header
    - * Cannot be changed (even when extension changes)
    - * Basic types: text, object, load file, directory

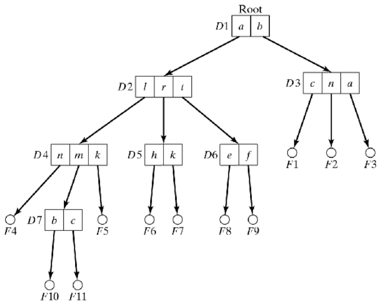
    - Application-specific types, e.g., .doc, .ps, .html

- **Logical file organization**
  - Most common: byte stream
  - Fixed-size or variable-size records
  - Addressed: **Implicitly** (sequential access to next record), or **Explicitly** by position (record#) or key

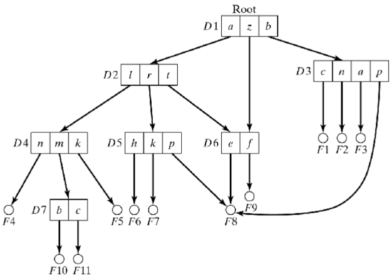


Definition 1.4. Directory Management

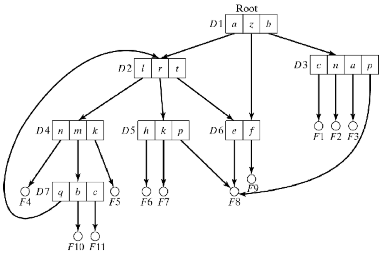
- **Main issues**:
  - **Shape** of the data structure
  - **What** info to keep about files
  - **Where** to keep the files in directory
  - **How** to organise entries for efficiency?
- **File directory data structure**:
  - **Tree-structured**
    - **Simple** search, insert, delete operations
    - Sharing is **asymmetric**



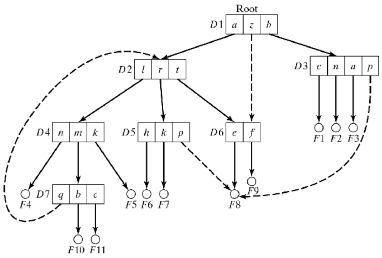
  - **DAG-structured**
    - **Symmetric** sharing
    - **Delete**: only last parent should remove files, so need **Reference count**



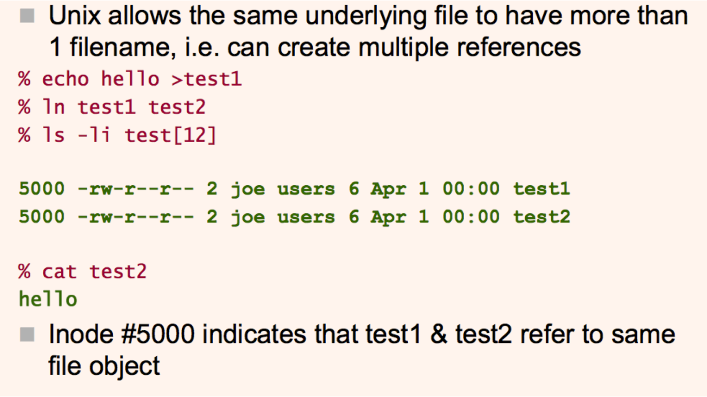
  - **DAG-structured with cycles**
    - Search is difficult with infinite loops
    - Deletion needs **garbage collection** (reference count not enough)



- **Symbolic links (shortcuts)**
  - **Compromise** to allow sharing but avoid cycles.
  - For **read/write**: symbolic link is the same as actual link.
  - For **deletion**: only symbolic link is deleted.



Definition 1.5. UNIX Hard Links



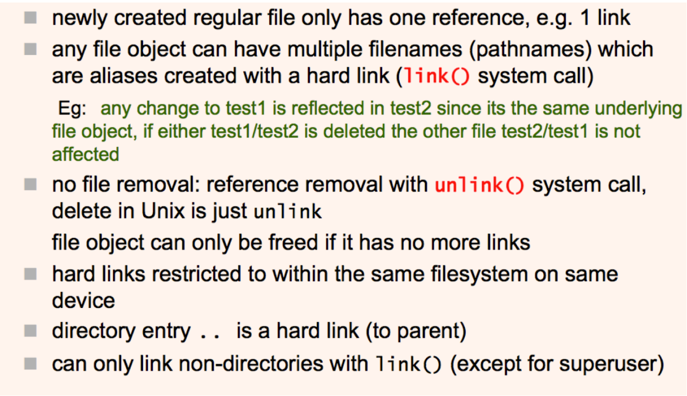
- Unix allows the same underlying file to have more than 1 filename, i.e. can create multiple references

```
% echo hello >test1
% ln test1 test2
% ls -li test[12]

5000 -rw-r--r-- 2 joe users 6 Apr 1 00:00 test1
5000 -rw-r--r-- 2 joe users 6 Apr 1 00:00 test2

% cat test2
hello
```

- Inode #5000 indicates that test1 & test2 refer to same file object

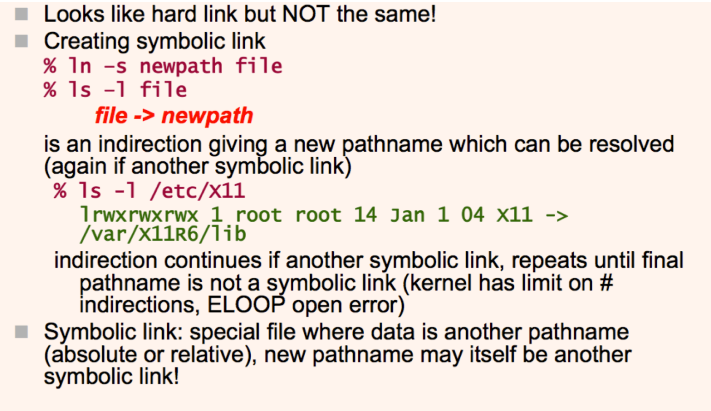


- newly created regular file only has one reference, e.g. 1 link
- any file object can have multiple filenames (pathnames) which are aliases created with a hard link (`link()` system call)

  Eg: any change to test1 is reflected in test2 since its the same underlying file object, if either test1/test2 is deleted the other file test2/test1 is not affected
- no file removal: reference removal with `unlink()` system call, delete in Unix is just `unlink`

  file object can only be freed if it has no more links
- hard links restricted to within the same filesystem on same device
- directory entry `..` is a hard link (to parent)
- can only link non-directories with `link()` (except for superuser)

Definition 1.6. Symbolic Links



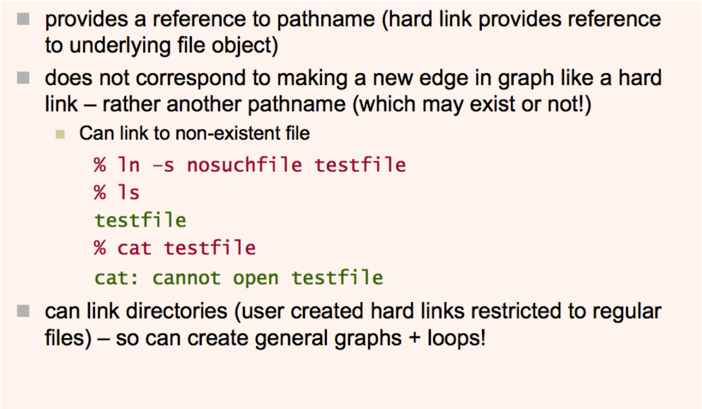
- Looks like hard link but NOT the same!
- Creating symbolic link

```
% ln -s newpath file
% ls -l file
      file -> newpath
```

is an indirection giving a new pathname which can be resolved (again if another symbolic link)

```
% ls -l /etc/X11
   lrwxrwxrwx 1 root root 14 Jan 1 04 X11 ->
   /var/X11R6/lib
```

indirection continues if another symbolic link, repeats until final pathname is not a symbolic link (kernel has limit on # indirections, ELOOP open error)
- Symbolic link: special file where data is another pathname (absolute or relative), new pathname may itself be another symbolic link!



- provides a reference to pathname (hard link provides reference to underlying file object)
- does not correspond to making a new edge in graph like a hard link – rather another pathname (which may exist or not!)
  - Can link to non-existent file

```
% ln -s nosuchfile testfile
% ls
testfile
% cat testfile
cat: cannot open testfile
```

- can link directories (user created hard links restricted to regular files) – so can create general graphs + loops!

Remark. Windows Shortcuts

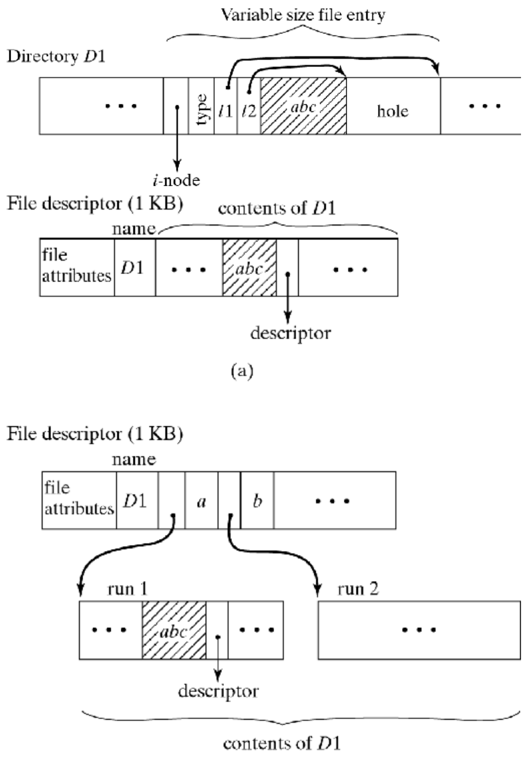
- Similar to symbolic links but does not re-expand further
- Only understood by GUI shell! Created also from GUI shell
- Win NTFS has hard links, symbolic links using junction mechanism (not normally used)

Remark. File Directories - Path name

- Concatenated local names with delimiter (. or / or \ )
- **Absolute** path name: start with root (/)
- **Relative** path name: start with current directory (.)
- Notation to move upward in hierarchy (..)

Method 1.1. Implementation of Directories

- **What information to keep in each entry**
  - **All** descriptive information, directory can become very large, searches are difficult / slow.
  - Only symbolic **name and pointer** to descriptor
    - ∗ Needs an extra disk access to descriptor
    - ∗ Variable name length
- **How to organise entries within directory**
  - **Fixed-size** entries: use array of slots
  - **Variable-size** entries: use linked list
  - **Size of directory**: fixed or expanding



(a)

(b)

red sftext tttext

Penghao is the best LaTeX writer.

```
def func():
    print("Penghao is cool!")
```

℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘℘

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

1. 123
2. 123
3. 123

	M1	M2	M3	M4	M5
M1	0	108	180	228	396
M2		0	72	168	288
M3			0	48	144
M4				0	128
M5					0

Algorithm 1 title

```
123
```

References

[1] Albert Einstein. *Zur Elektrodynamik bewegter Körper.* (German) [*On the electrodynamics of moving bodies* `www.google.com.sg`]. Annalen der Physik, 322(10):891921, 1905.