Chap. 6

FILE SYSTEMS

terminology + naming
system architecture
vis-a-vis I/O system
disk space management
user programs

"sequence of bytes"
⇒ and higher abstractions eg. record
see Fig 6-3 : executable file, archive

Fig 6-4 possible file attributes
protection, password
creator, owner, R/W, hidden, system
archive, temp.
lock
size : current / max
record / key attributes
time: creation / modn / last access

operations: create
delete
* open / close
read / write / append
seek
get / set attributes
rename

self study: 6.1.7. Example program using file system calls

DIRECTORIES

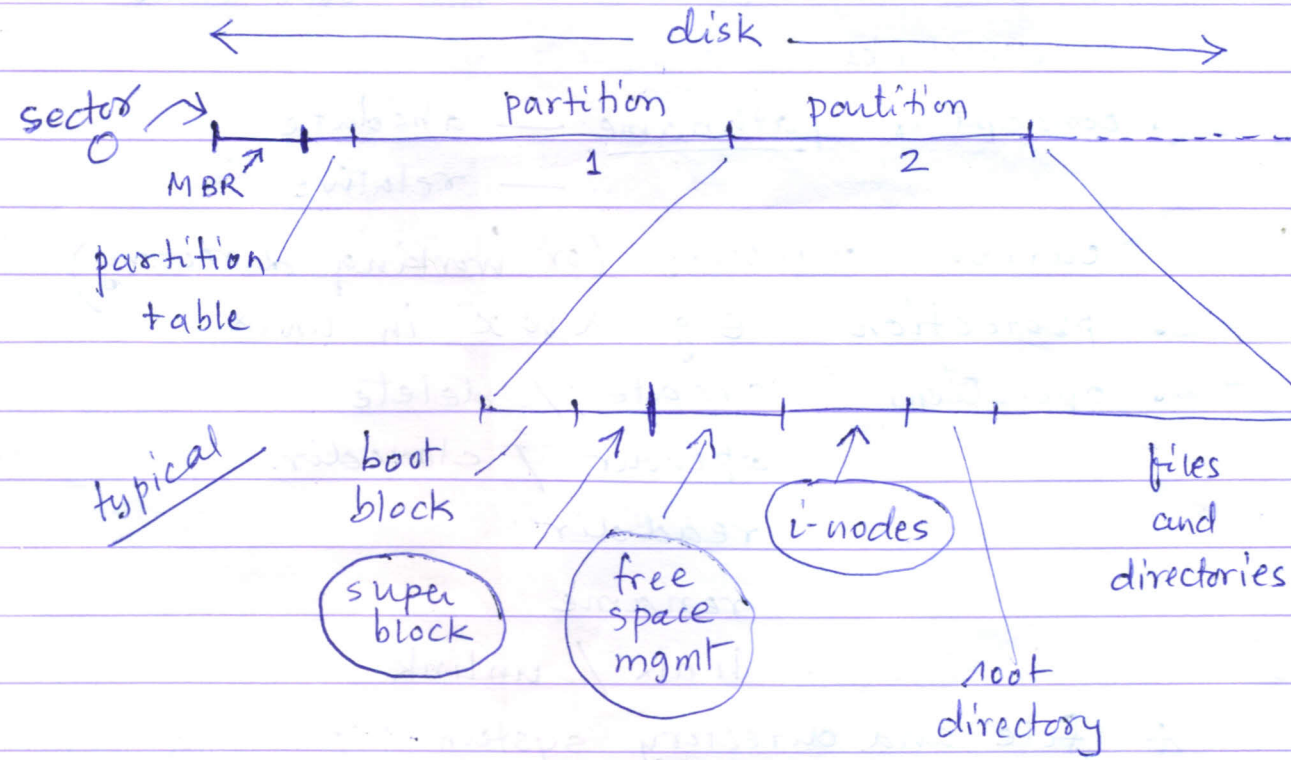
- a directory is also a type of file.
- but not to be used as a "sequence of bytes"
- a directory "contains" files and other directories
- concept of pathname — absolute
  — relative

  current directory (or working directory)
- protection e.g. rwx in UNIX
- operations create / delete
  opendir / closedir
  readdir
  rename
  link / unlink

* file and directory system defines a complex data structure on the secondary storage device / partition

6.3 FILE SYSTEM IMPLEMENTATION

6.3.1 File System Layout



goals - efficient utilization of space
- efficient access

6.3.2 Implementing Files

"... probably the most important issue ....."

contiguous allocation of disk blocks

* simple, easy to implement
* efficient access

but → * external fragmentation
* useful in CD-ROMs

linked list allocation

- linked list of blocks (disk blocks)
- (say) the 1st word of each block in the list is a pointer to the next block

* inefficient access esp. random access !!
* usable block not 2^n

[note difference between sequential and random access]

Linked list allocation but table in memory

↑ File Allocation Table (FAT)

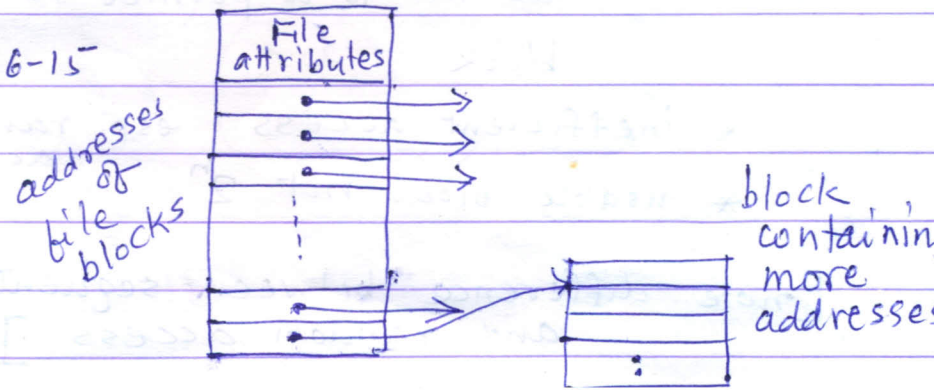
see Fig. 6-14

* what if the disk (or partition) is very big?
* random access - straight forward

I-nodes

* allocation information per file
* separate from file

see Fig 6-15



6.3.3 Implementing Directories

- self study

* how to accommodate filenames of arbitrary length

6.3.4 shared files → omit

→ BUT note the following

what are the issues when two or more processes open the same file?

what happens if a process opens a file more than once?

[see Fig. 6-5] → LAB

6.3.5 Disk SPACE MANAGEMENT

* structure of magnetic disk
* impact of block size on –
  - efficiency of space utilization
  - effective data transfer rate
* linked list or bitmap to track free blocks

6.3.6 File system reliability

- backups – full / incremental // logical / physical
- consistency

6.3.7 File system performance

- buffer cache or block cache
- block read ahead
- reducing disk arm motion

6.3.8 UNIX V7 File system

self study

Fig. 6-37 directory entry
6-38 i-node
6-39 steps in looking up