CS 3650 – Computer Systems
Spring 2024
Peter Desnoyers

Lecture 18, Thur Mar 14, 2024

File systems

file: array of bytes

hier. namespace:

obj = file (dir

dir = { "name": obj,

: }

open

close

read

write

(create)

unlink (delete)

mkdir

rmdir

<list dir>

open("/home/pjd/file.txt",

...

other stuff in the file system

/dev/null ← fake

/dev/tty ← terminal ("teletypewriter")

/dev/sda ←→ disk ("SCSI device A")

/dev/random ← random #s

/zero

"CON:"

ioctl ("i/o control")

→ the hack function

character & block devices

Symbolic links

"libc.so" ← C runtime library

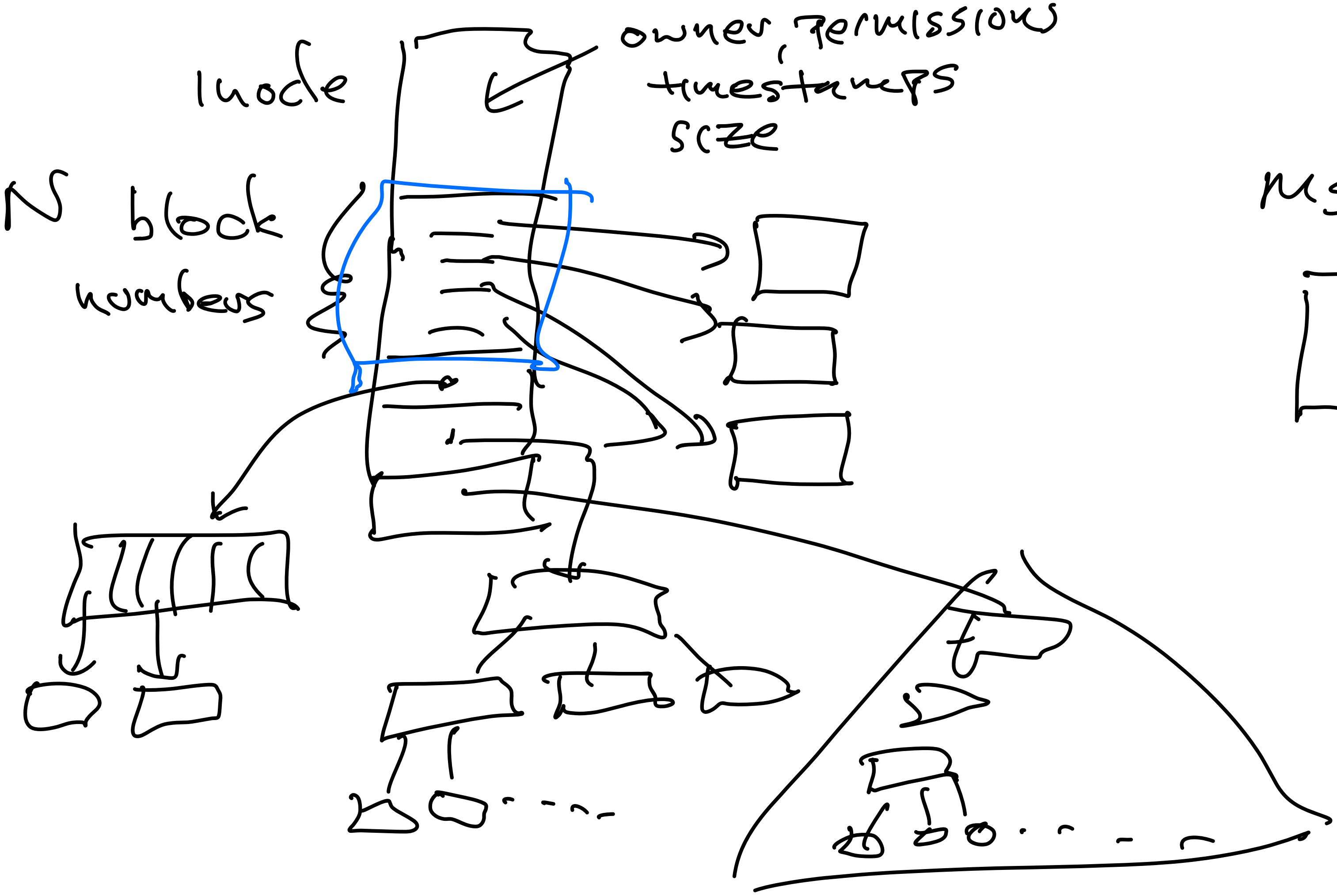
/usr/x86-64/lib/libc.so → libc.so.10.2.1

ls -l /tmp

" → /private/tmp"

File system formats

Unix (ext2)



CD-ROM

dirent:

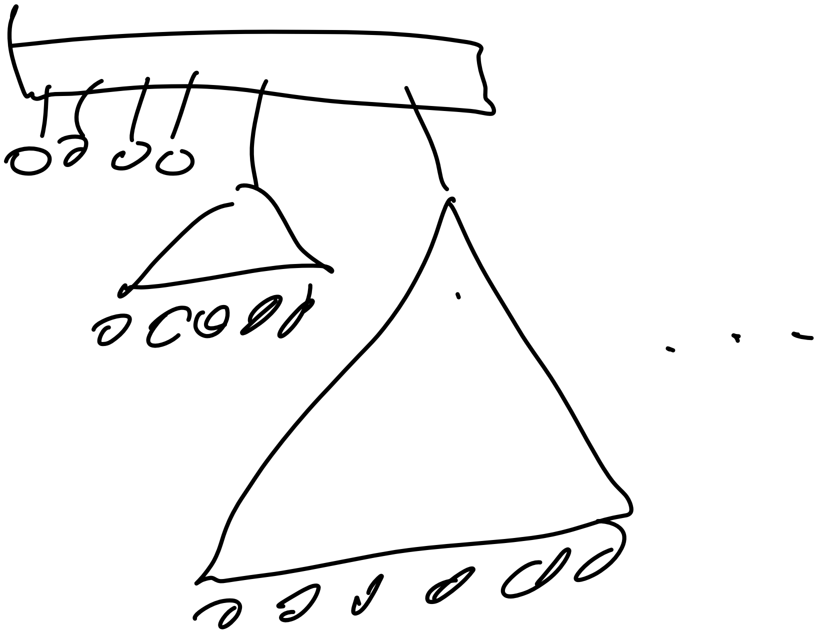
name	start block	len

↑ sequential

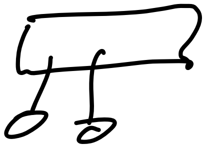
MS-DOS

name	start block	len

↑ linked list



"skewed" tree



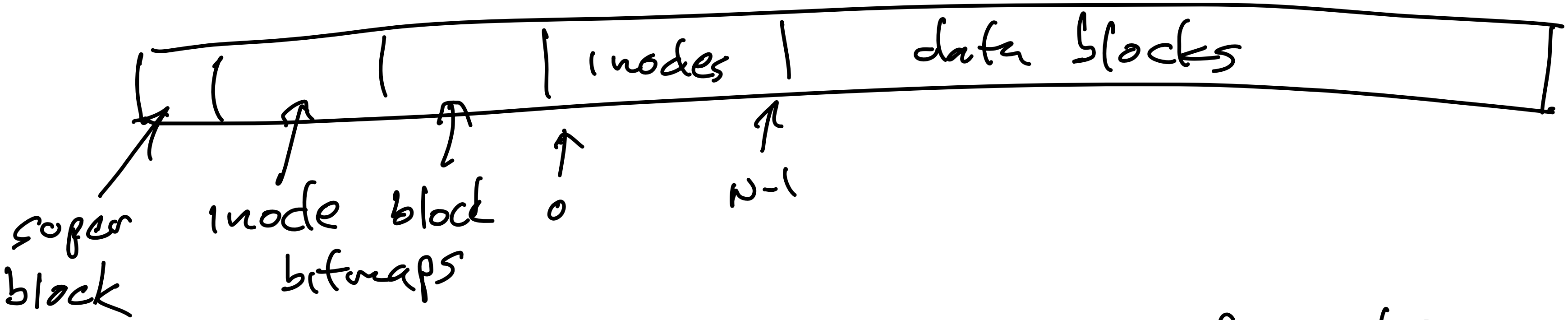
inode identifier:
-number

directory:

```
{ "name": inode#,
  "name": inode#
     ....
}
```

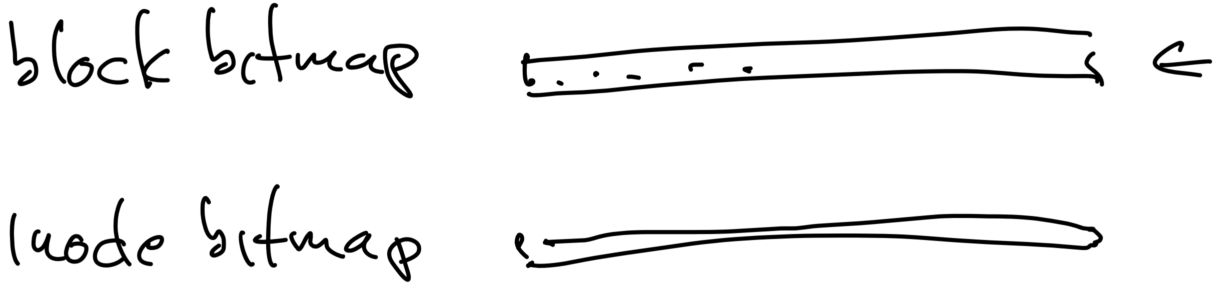
```
struct dirent {
  bool valid
  int inode#
  char name[..]
}
```

ext2 layout (not to scale)



root inode #

allocation / free space



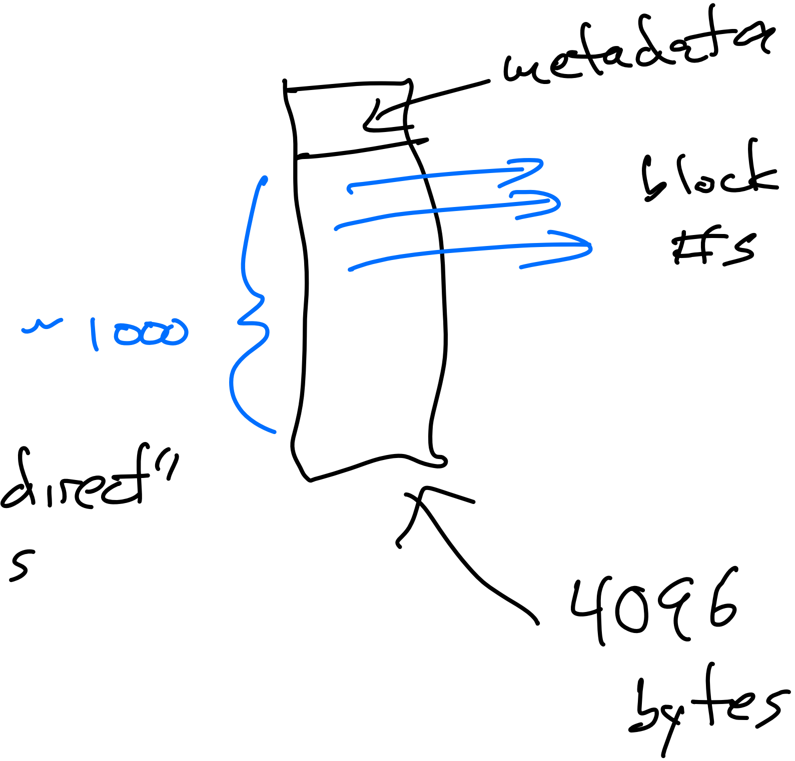
← bit i :
1 if block i
is in use

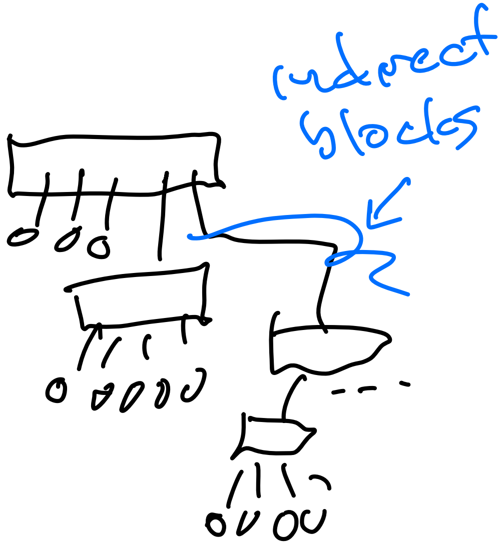
3650 file system

1) read-only
2) in-inode pointers only
3) full-block inodes

no "indirect" blocks

no allocation bitmaps





Variations

block organization:

- contiguous (CDROM)
- linked list (MSDOS)
- tree (ext2)
- extent (NTFS, ext4, ...)

1001, 1002, 1003 . . . . . .

"extent" = (start block#,
length)

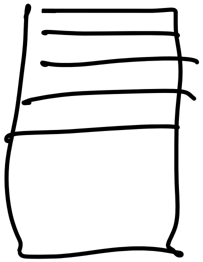
file fragmentation:

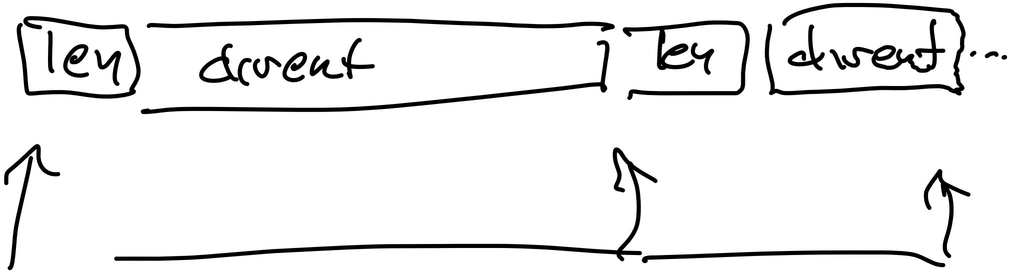
file 1, blocks
10, 100, 70, 210, 5

file2:
101, 102, 103, 104, 105

[101, 105]

Long file names



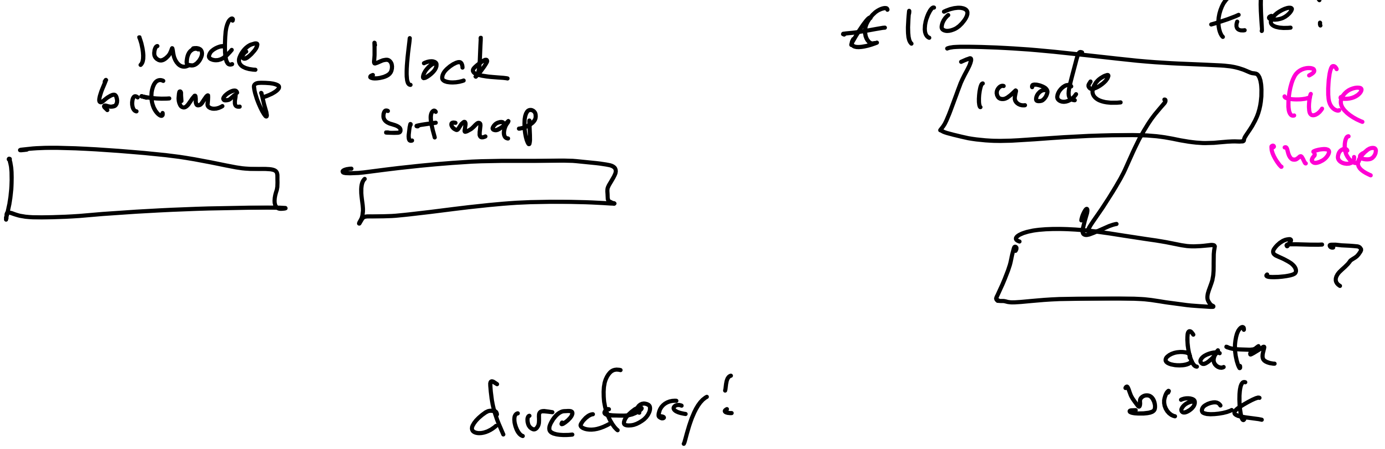


hashed & tree-structured directories

ext4

Crash resilience

rm /dir/file



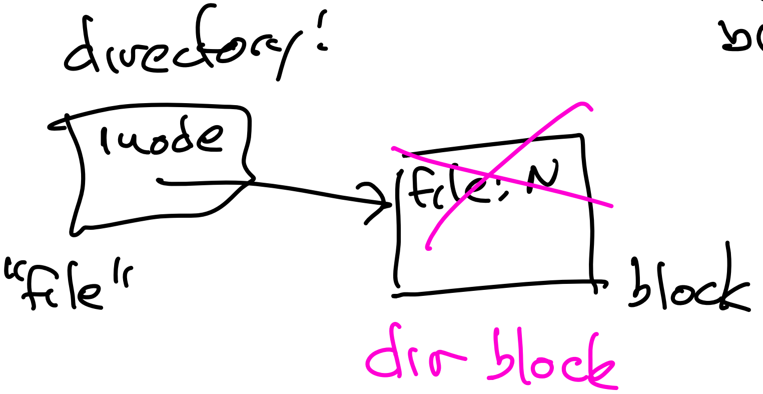
to delete:

a) — write dir block → remove "file"

b) — write inode map → now free

c) — write block map ①

② crash



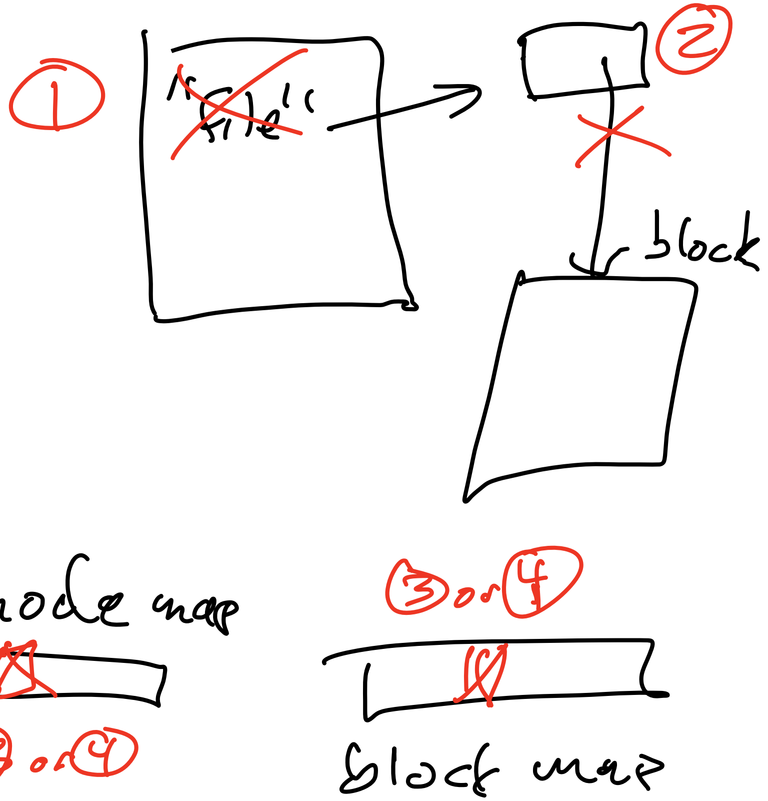
"/dir/file" → inode 110

inode 110 → block 57

block_map[57] = FREE

Careful writes

1. zero out dir. entry
2. zero out inode
3. clear inode map entry
4. " block map



+: prevents corruption

−: slow

Journaling

write-ahead log

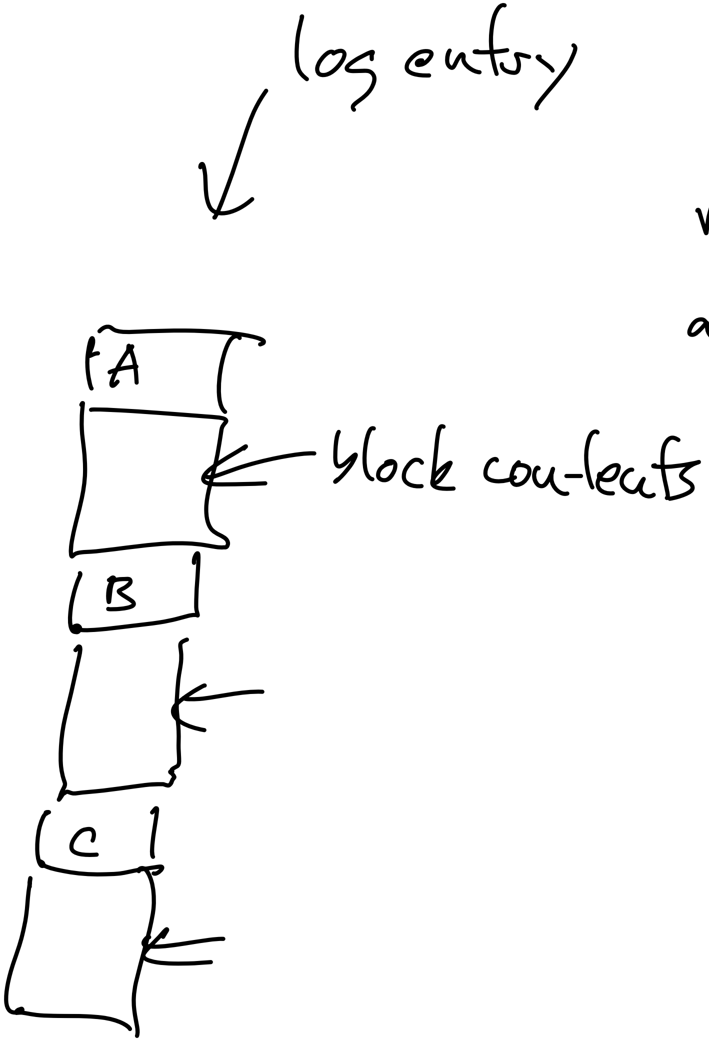
1) "I'm going to X" → log

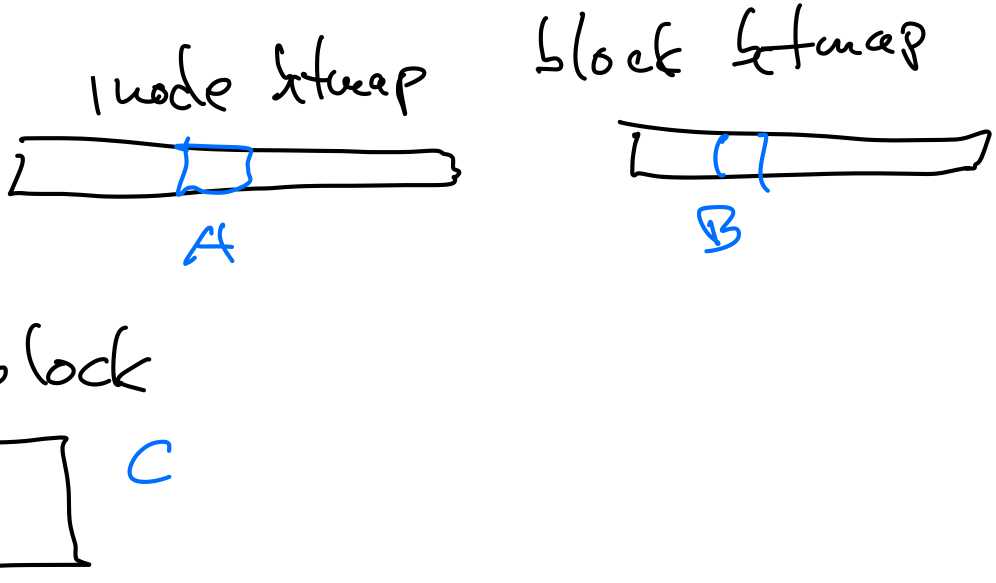
2) X

crash

read log

apply


log entry
A
block contents
B
C


inode bitmap
A
block bitmap
B
dir block
C