Lecture 43

05 November 2018 09:39

① in-memory FS state:

- inode cache
- dentry cache
- page cache
- FS logic + other state

on-disk

- super block
- meta data blocks (inode)
- data blocks.

② dentry cache ~ directory entries.

/home /data / file.txt ⤳ logical pathname

no explicit (data) description of hierarchy setup.

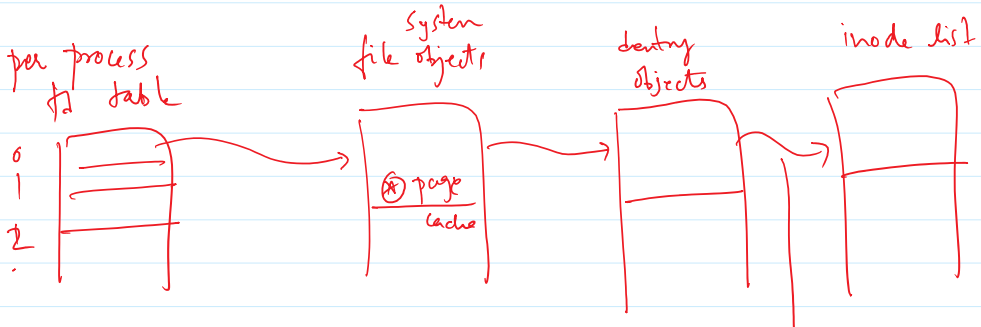
dentry cache — < entry, inode >

① lookup path component in dentry cache

- if hit
  get / use inode
- else
  create dentry and lookup inode from disk.

② - use inode to read contents (file) of directory.

- check for next path component
  - name, permissions etc.
    goto ①
  - else, error!



→ /home/data1/file1.txt

→ /home/data2/file2.txt
? ?

directory - is a file

< / ⇝ inode1 >

< /home ⇝ inode2 >

↑ path component

directory - is a file

datablocks of this file : <filename, inode #>

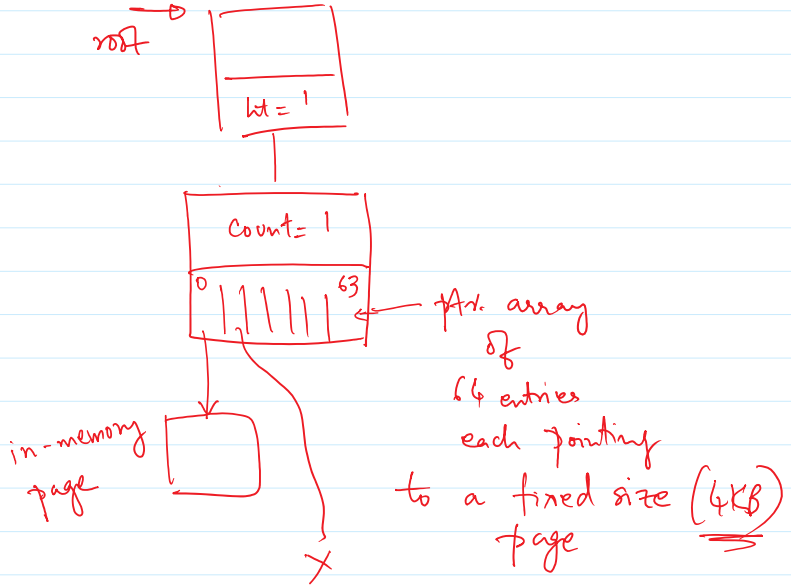
(3) page cache : 'data' cache.

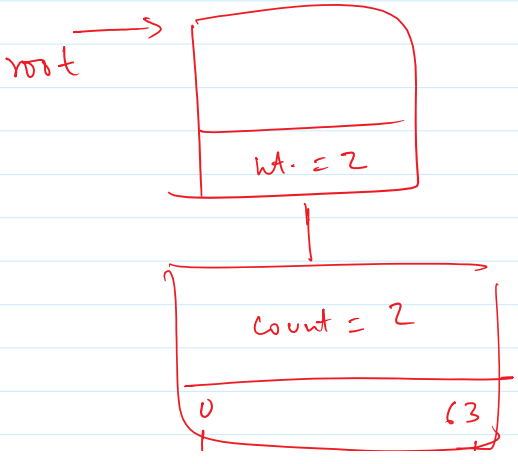
~ page cache (file/disk) —— <file, offset> ⊗

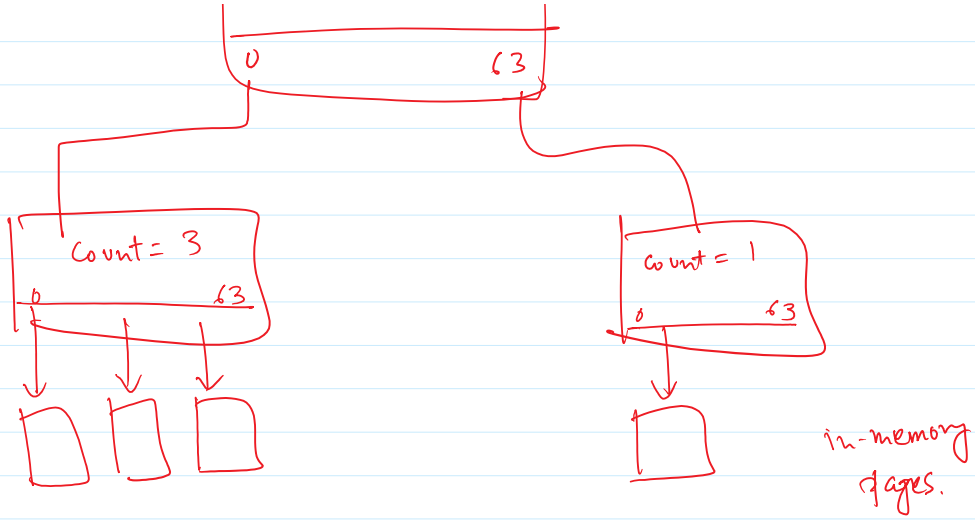
buffer cache —— <disk, block #>

~ challenge ~ how to get to a cached object quickly?
& store/build cache efficiently.

linux radix tree base page cache.







(*) all reads/writes go through page cache
└ one-mode of operation.

- cache consisting techniques
  - write-through — sync. writes to disk
  - write-back — async. writes to disk
    └ periodic flush to disk (Kernel threads).

(3) file system consistency. ~ JFS journaling/log file system.
- meta-data not in sync with data
- super-block not in sync with meta data.

e.g: tool 'fsck' file system check