CS 423
Operating System Design: File Systems-II/Adv Storage 1

Ram Alagappan

Acks: Prof. Tianyin Xu and
Prof. Shivaram Venkataraman (Wisconsin) for the slides.

Recap

Very small file system (VSFS)

On-disk blocks: superblock, inode table, bitmaps, data blocks

File indexing: pointers, indirection, extents

MS FAT

Access methods: what happens on a file create/write/read

Page cache

Crash consistency – problem

Today’s lecture: solutions to CC and LFS

Crash Consistency

Basic problem:

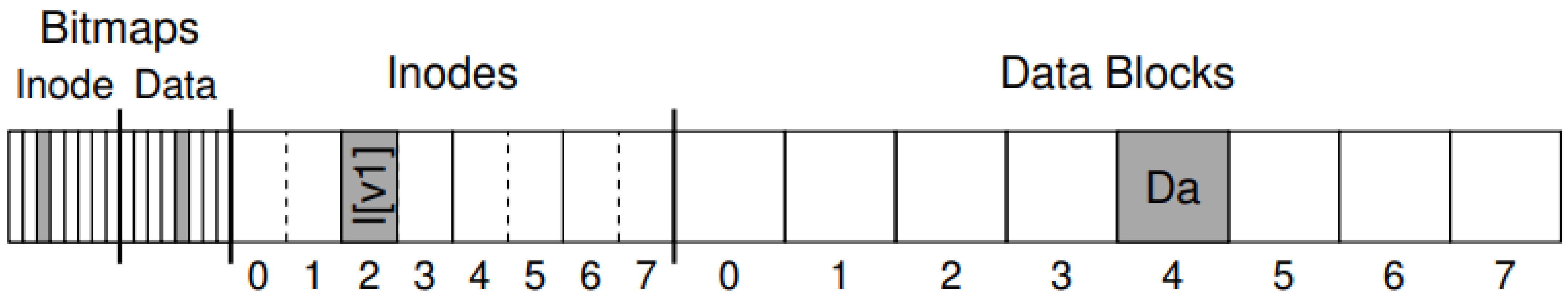
Must update many data structure on disk as a unit

What if failure happens in the middle

Types of failure:

- kernel panic
- power failures

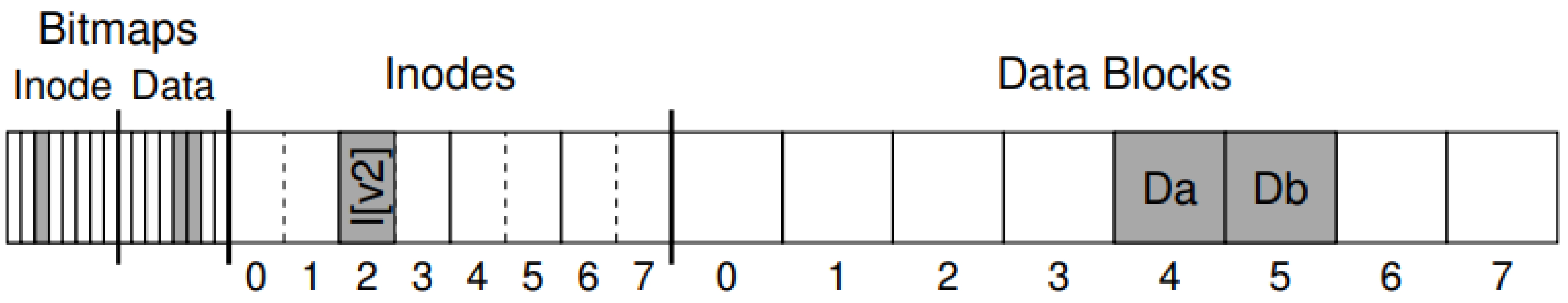
Append a Block Example



How many blocks do we need to write to accomplish the append?

Which ones?

Problems



What if only Db is written?
Only i[V2] is written to disk? (2 problems)
Data bitmap is alone written to disk?
Bitmap and data are written:
Data and inode are written:
Bitmap and inode are written:

What's special about the last case?

Metadata vs. Data

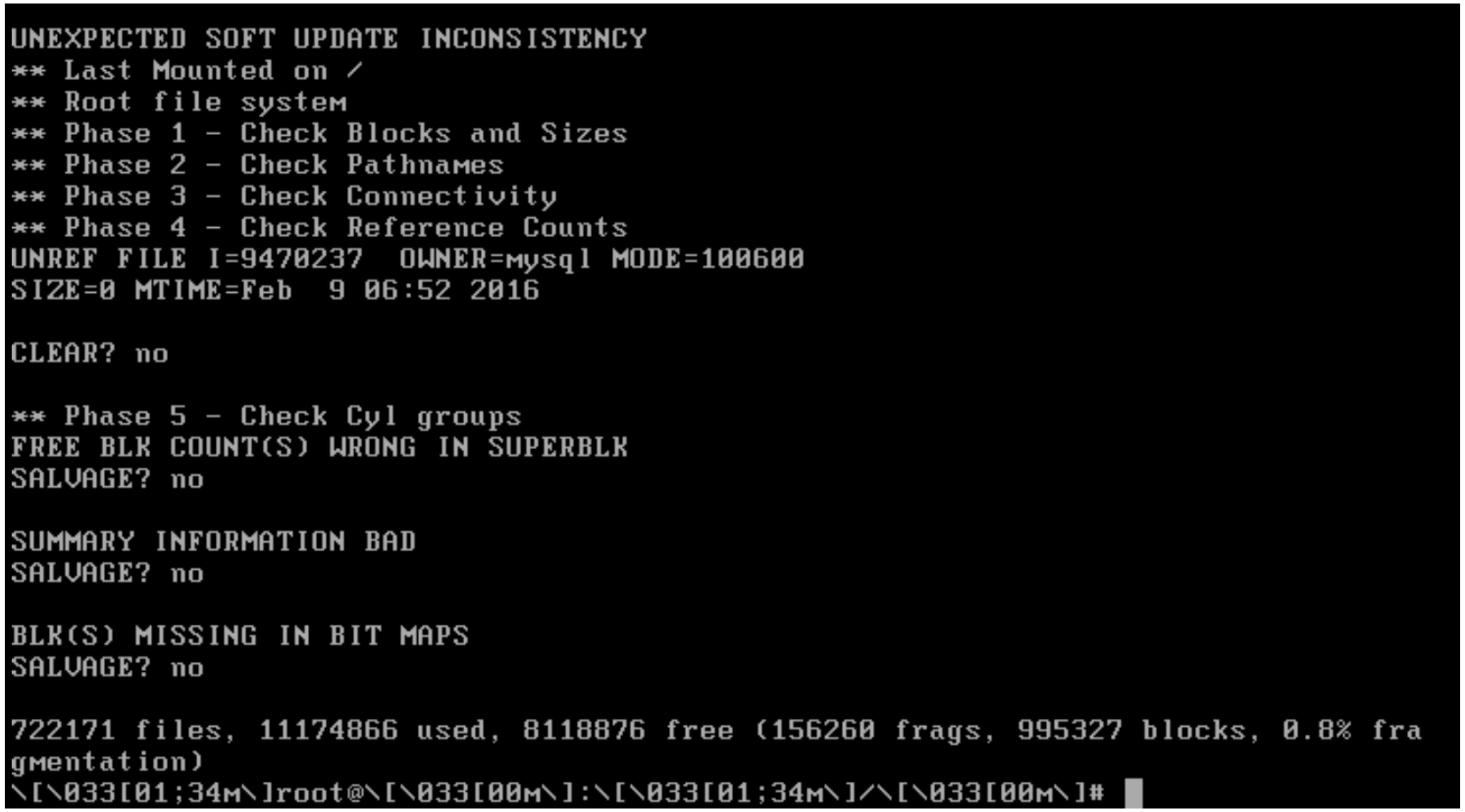
FS Metadata consistency vs. Data consistency

FS metadata consistency: internal structures agree with each other

Data consistency: additionally, the data must "make sense" to applications and users

FSCK

Let inconsistencies happen and take care during reboot



```
UNEXPECTED SOFT UPDATE INCONSISTENCY
** Last Mounted on /
** Root file system
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
UNREF FILE I=9470237  OWNER=mysql MODE=100600
SIZE=0 MTIME=Feb  9 06:52 2016

CLEAR? no

** Phase 5 - Check Cyl groups
FREE BLK COUNT(S) WRONG IN SUPERBLK
SALVAGE? no

SUMMARY INFORMATION BAD
SALVAGE? no

BLK(S) MISSING IN BIT MAPS
SALVAGE? no

722171 files, 11174866 used, 8118876 free (156260 frags, 995327 blocks, 0.8% fra
gmentation)
\[\033[01;34m\]root@\[\033[00m\]:\[\033[01;34m\]/\[\033[00m\]#
```

FSCK CHECKS

Do superblocks match?

Is the list of free blocks correct?

Do number of dir entries equal inode link counts?

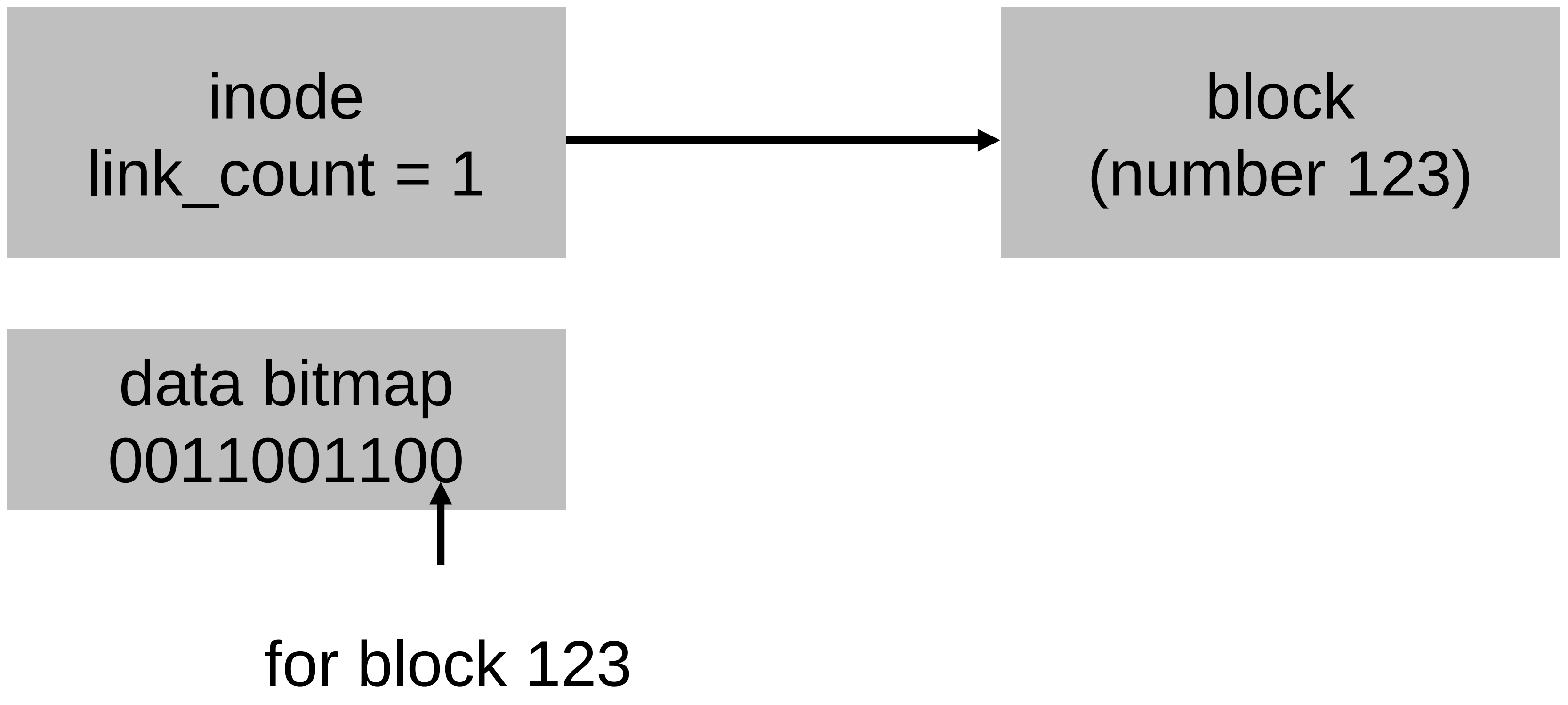
Do different inodes ever point to same block?

Are there any bad block pointers?

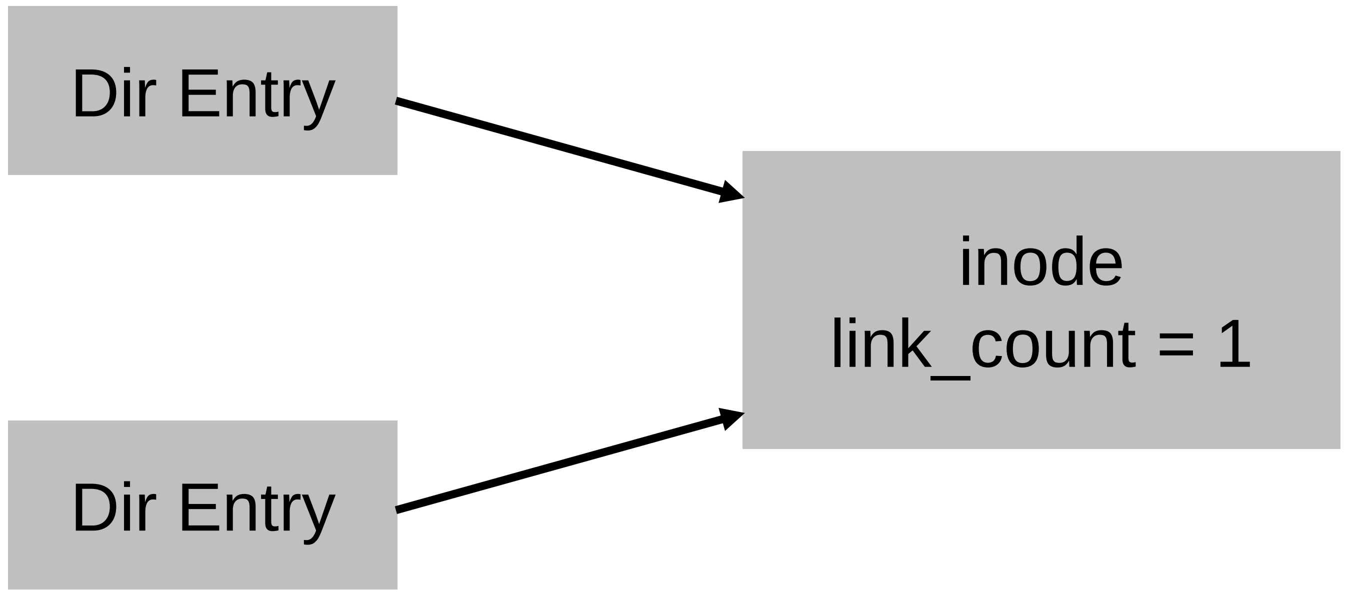
Do directories contain “.” and “..”?

…

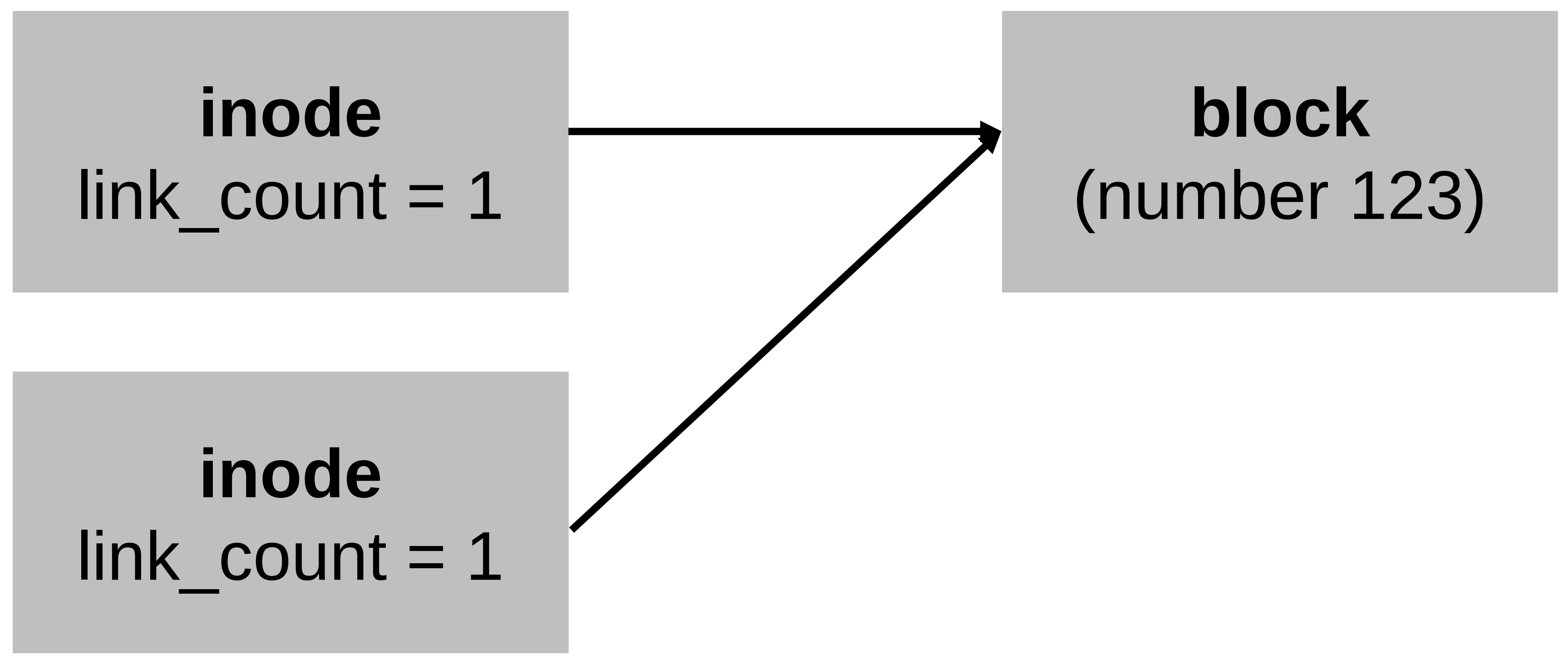
Free Blocks Example



Link Count Example



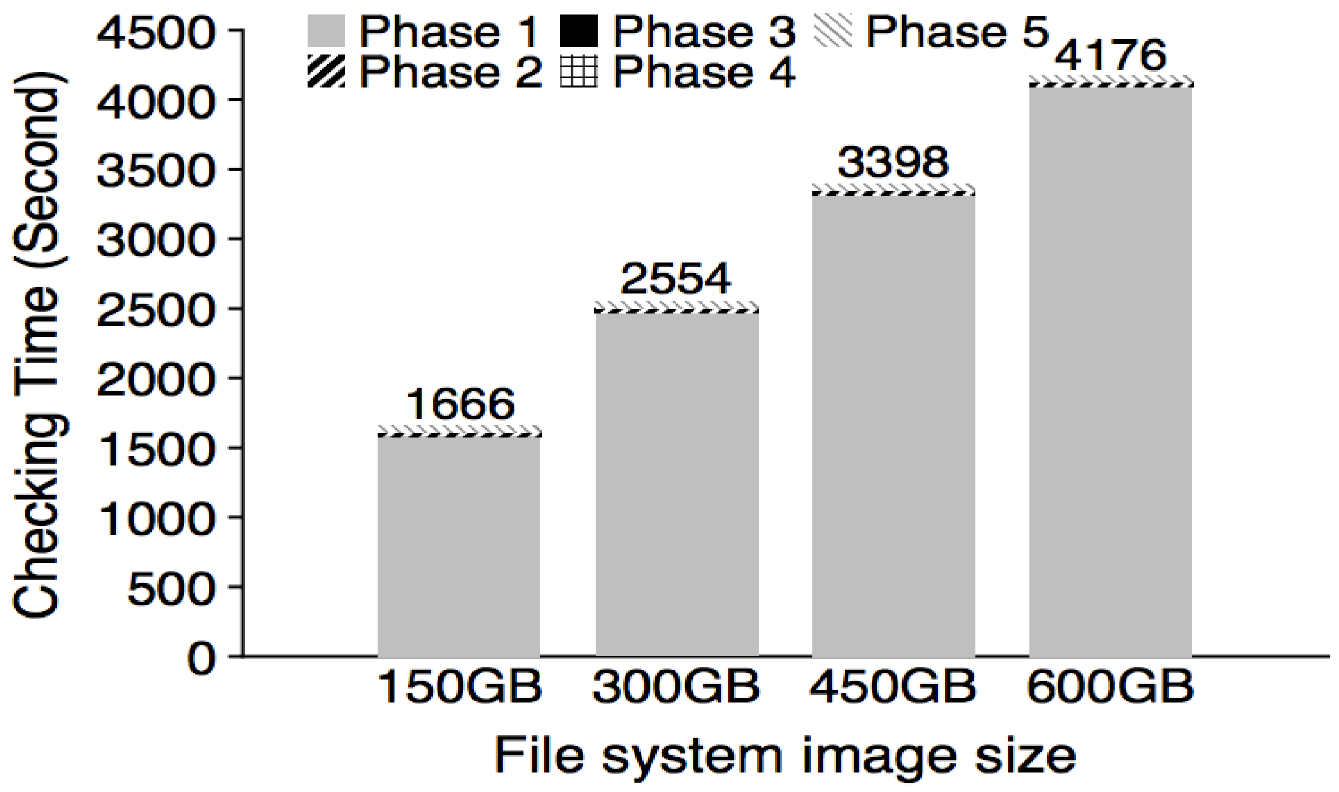
DUPLICATE POINTERS



FSCK PROBLEMS

Not always obvious how to fix file system image - don't know "correct" state, just consistent one

Simply too slow!



Checking a 600GB disk takes ~70 minutes

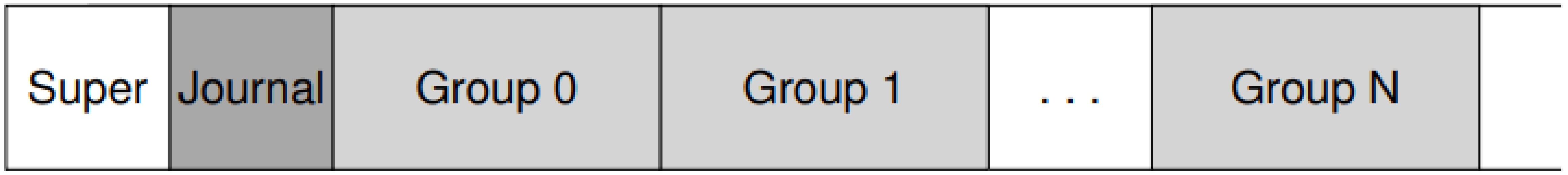
ffsck: The Fast File System Checker
Ao Ma, Chris Dragga, Andrea C. Arpaci-Dusseau, and Remzi H. Arpaci-Dusseau

Journaling or WAL

Main idea: write a “note” to a well-known location before actually writing the blocks

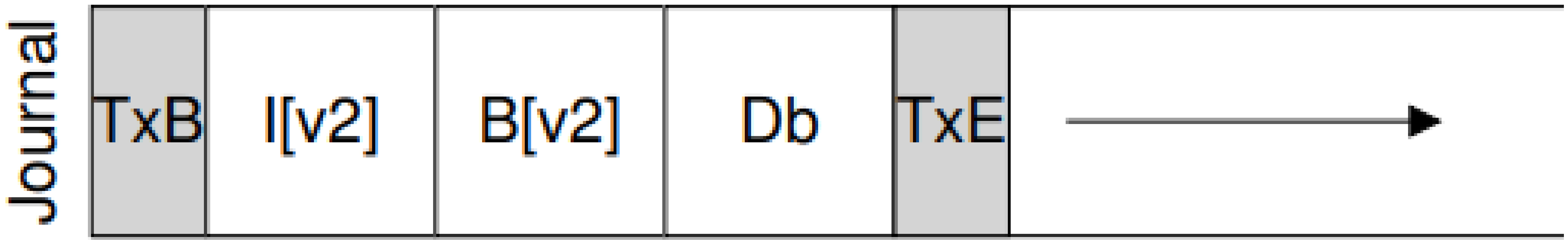
If crash, know what to fix and how to do so from the note (instead of scanning the entire disk)

Journaling in Linux ext3



Append a block to an existing file example

Journal Transaction



Data journaling vs. metadata journaling

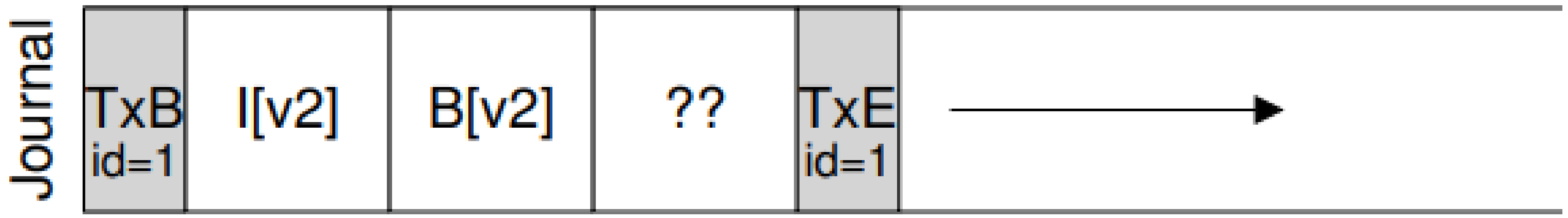
Journaling or WAL

First write the txn to journal
Once that is safe, write the actual blocks (this is called checkpointing)

What if crash happens during journal write?

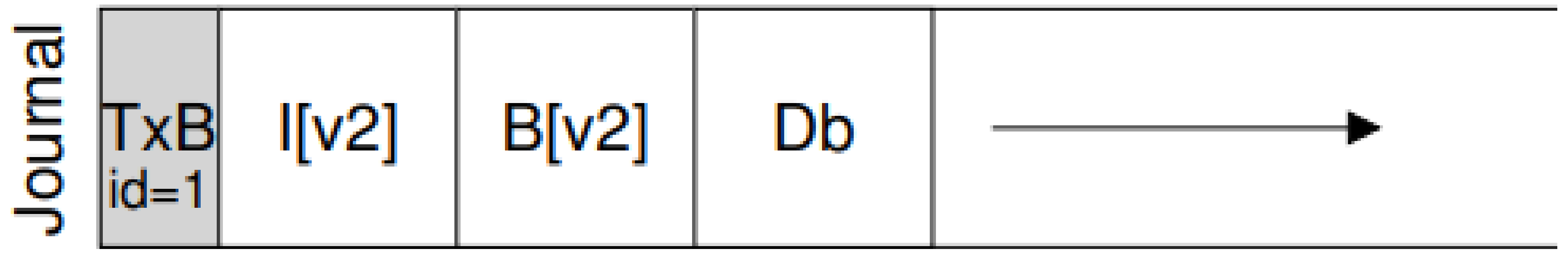
Journal Writes

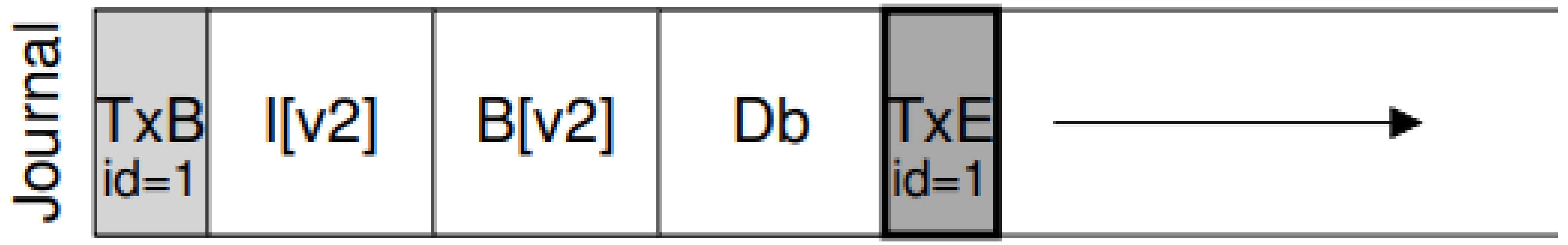
Can issue one write at a time but is too slow
Must maximize how many writes can be concurrently sent
But sending all 5 blocks together is problematic



How to solve this?

One solution





Barriers

Incurs a wait or flush between TxB + Data and TxE… How to do without waiting?

Solution without Wait

Recovery

Scan the journal

Checkpoint completed transactions

Discard otherwise

Will the system be safe if crash happens during recovery

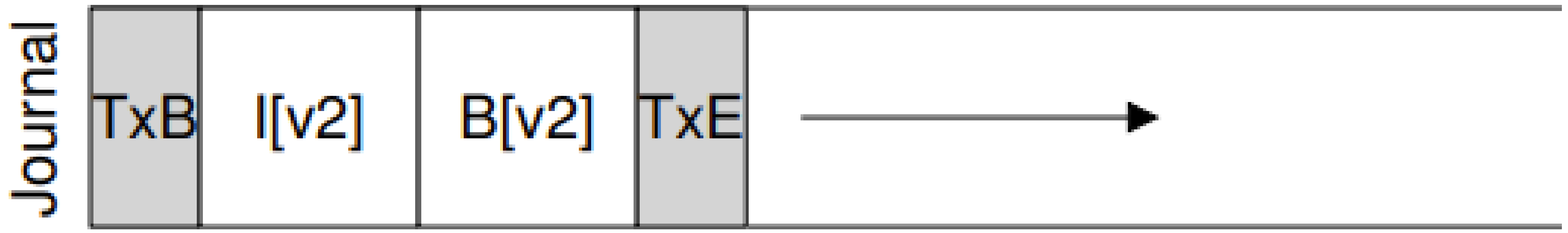
Batching for Efficiency

What is the problem with DJ?

Think about performance…

Which workload will suffer the most?

Metadata Journaling



Data blocks written in “FS proper” (in place)
Metadata goes via journal

What is the order of writes?

Order of Writes

D → JM → JC → M
First data, write metadata to journal, write commit block, then checkpoint metadata

D || JM → JC → M (|| means concurrent)
Is this safe?

Order of Writes

D → JM → JC → M

First data, write metadata to journal, write commit block, then checkpoint metadata

D || JM → JC → M (|| means concurrent)

Is this safe?

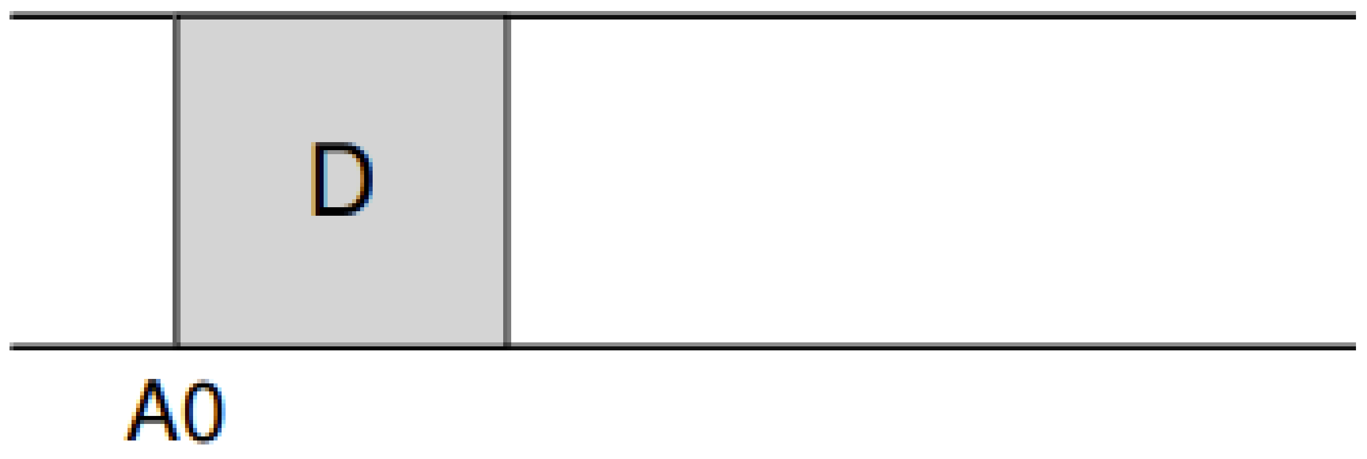
Log Structured FS

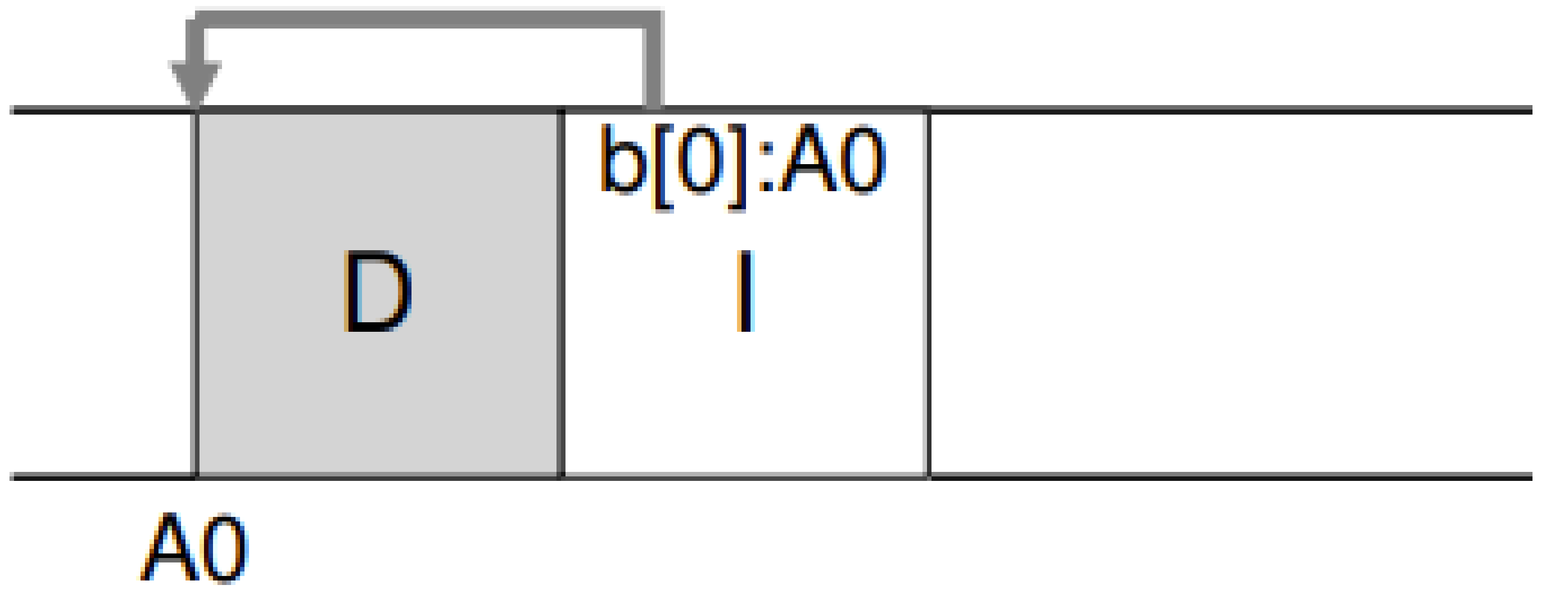
Motivation:

1. Sequential speed is much higher than random – all writes must be sequential ideally
2. Memory sizes are growing – write performance matters the most, can also buffer more

Goal

Make *all* writes sequential

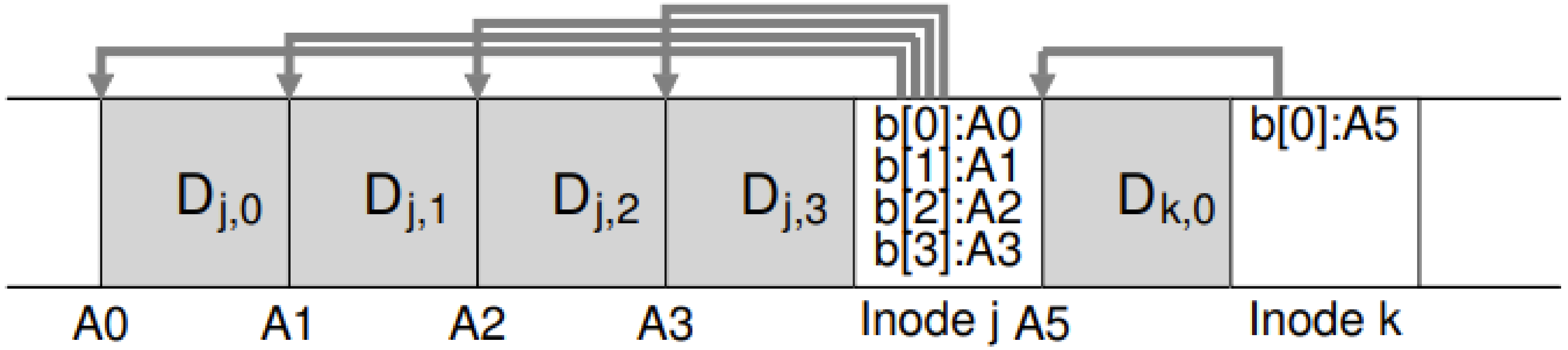




Buffering and Segments

Buffer and write in large chunks

Called a segment



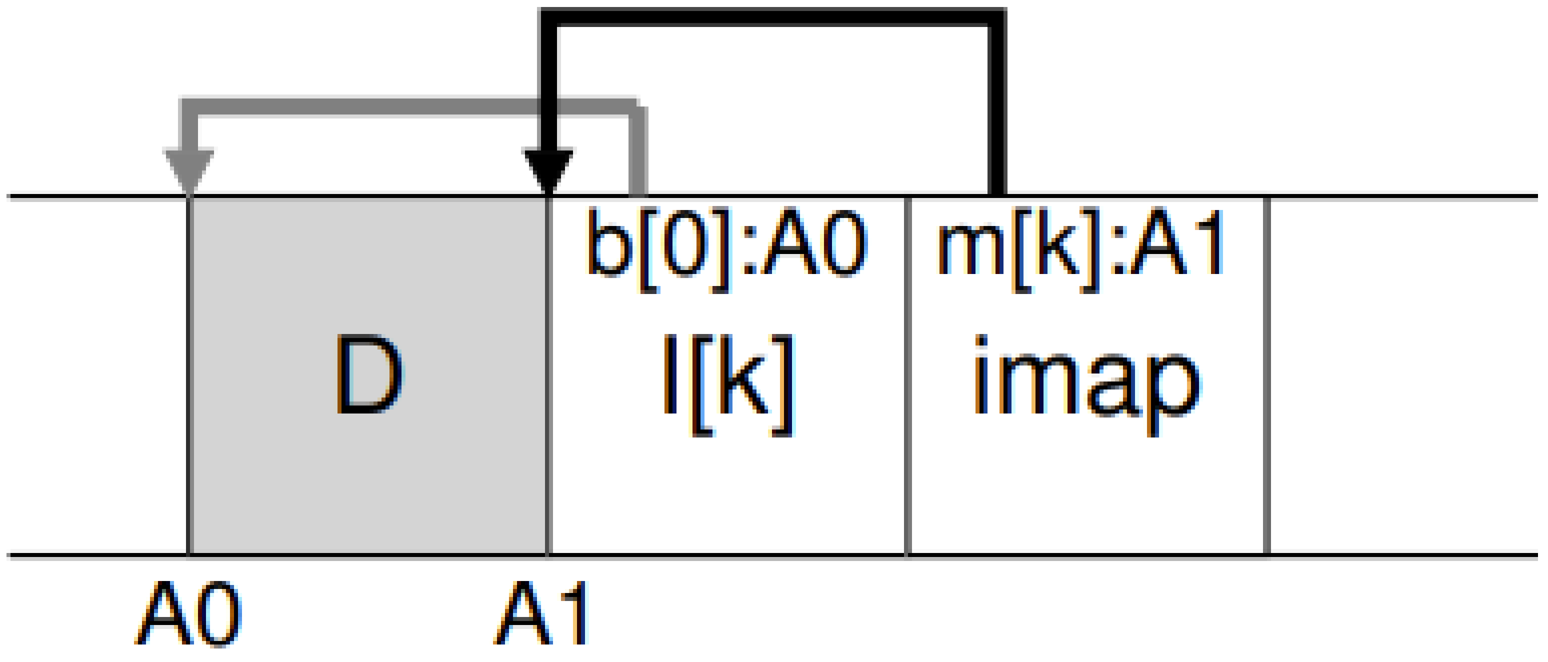
Finding an Inode

How VSFS (or most UNIX FS) does this?

LFS: what’s the problem?

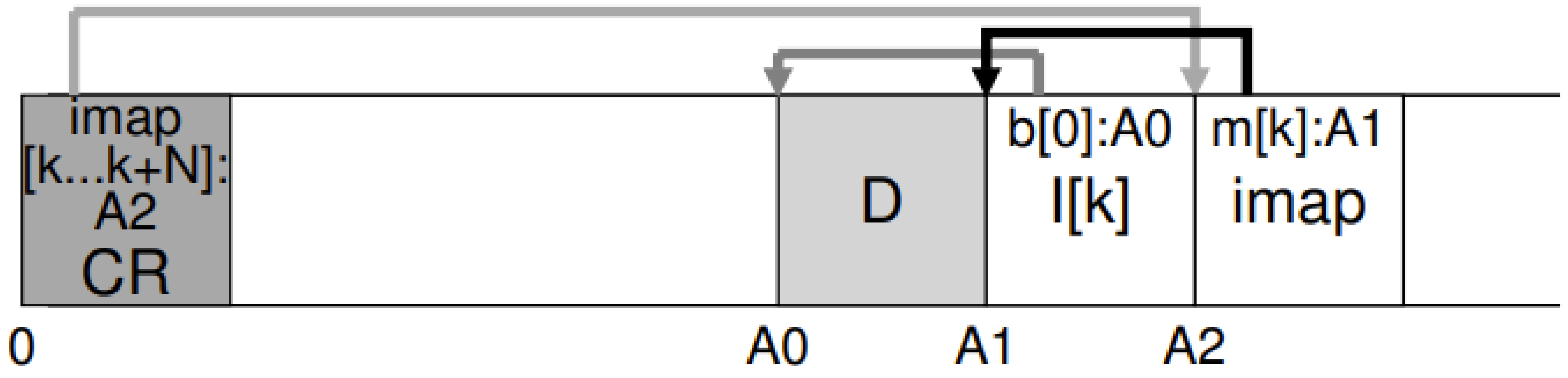
Finding an Inode

LFS uses a new structure called imap
Imap: take inode as input give disk address
Where should imap be?
Memory? Fixed location on disk?



How to Find IMAP!?

If in the log, how to find it?

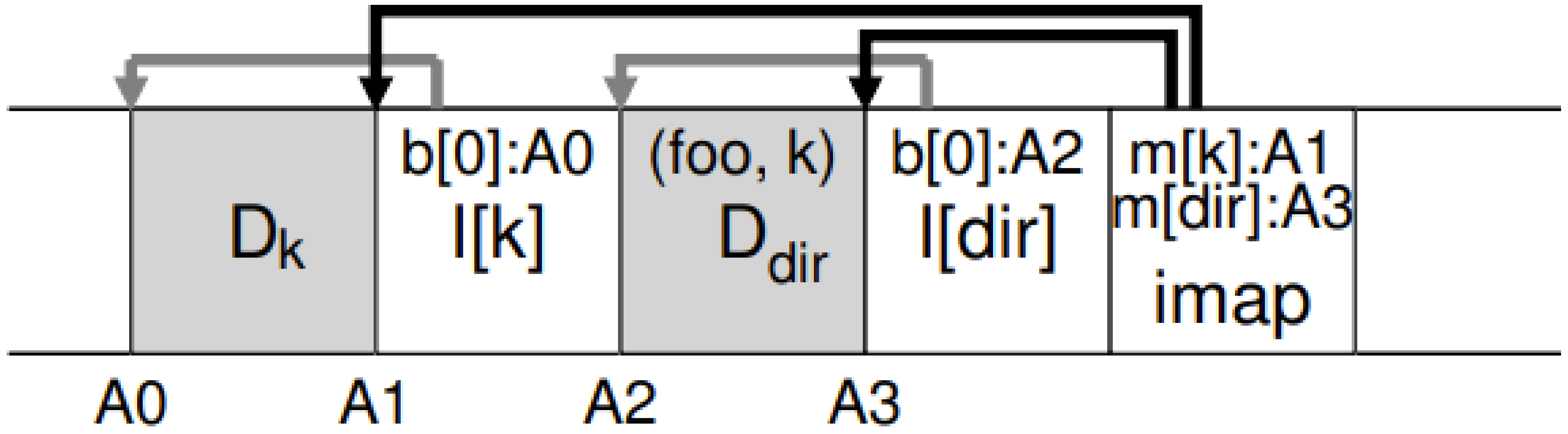


Reading a File

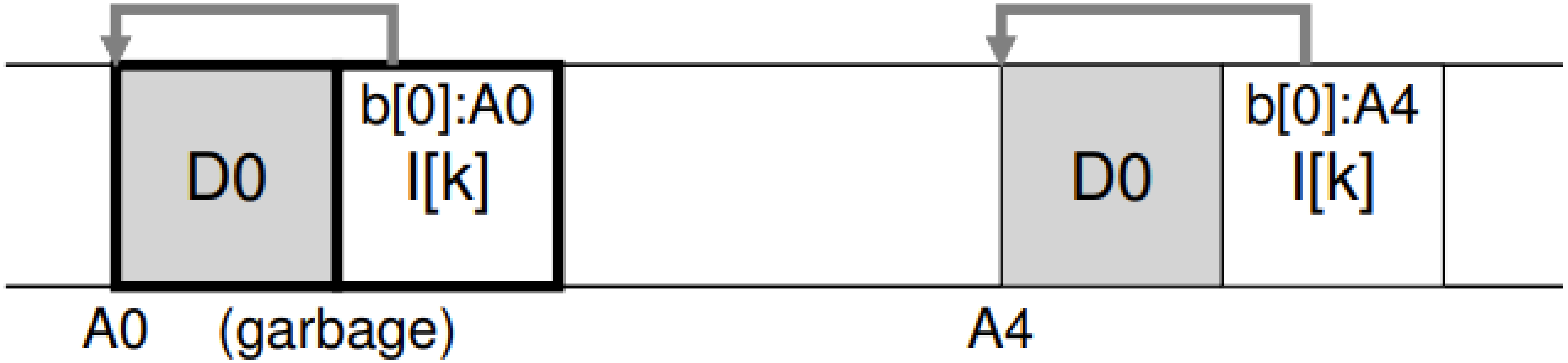
Assume nothing in memory
What are the steps?

Directories

Creating a file foo in a directory and appending a block to it



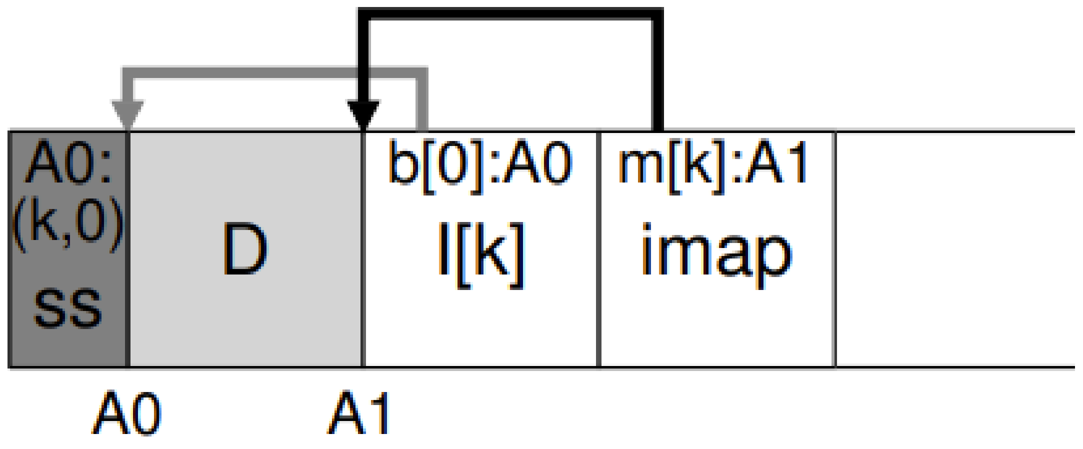
Garbage Collection



The problem?
The opportunity?

Cleaning

See which blocks are live within a segment
Write live blocks into a new segment, reuse the old segment
Segment summary block



```
(N, T) = SegmentSummary[A];
inode  = Read(imap[N]);
if (inode[T] == A)
    // block D is alive
else
    // block D is garbage
```

Crash Consistency in LFS

Next Lecture

RAID

Google File System