Graduate Operating Systems

Spring 2023

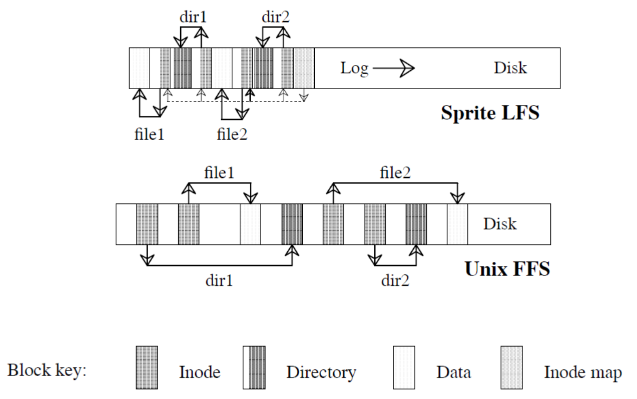
1

Paper “Sprite LFS”

- Does Sprite LFS improve READ or WRITE performance?
- What is the biggest challenge of a log-structured file system?
- Does Sprite LFS focus on large or small files?
- How does Sprite LFS keep reading performance acceptable?

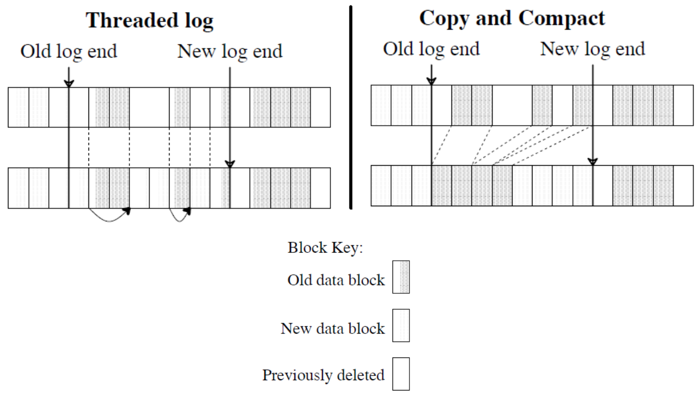
2

Paper "Sprite LFS"



3

Paper "Sprite LFS"



4

Paper “Sprite LFS”

- Segments, segment cleaning, segment summary block
  - When should the segment cleaner execute?
  - How many segments should it clean at a time?
  - Which segments should be cleaned?
  - How should the live blocks be grouped when they are written out?
- Cost-benefit policies
  - Cold segments should be cleaned at high utilization
  - Hot segments should be cleaned at low utilization
- Checkpoints
- Experimentation; metrics (“write cost”); microbenchmarks; pros/cons; overheads

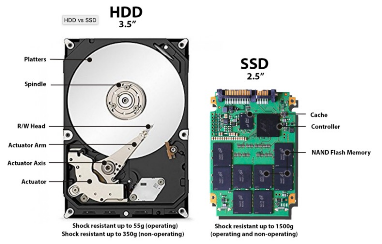
5

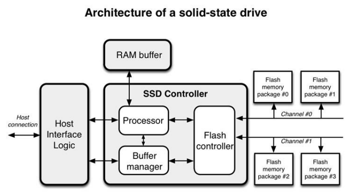
Summary “Sprite LFS”

- Improve write performance
- Crash recovery
- Concept of segments
- Segment cleaning (garbage collection)
- Pros & cons?

6

Paper “F2FS”


HDD vs SSD
HDD
3.5”
Platters
Spindle
R/W Head
Actuator Arm
Actuator Axis
Actuator
Shock resistant up to 55g (operating)
Shock resistant up to 350g (non-operating)
SSD
2.5”
Cache
Controller
NAND Flash Memory
Shock resistant up to 1500g
(operating and non-operating)

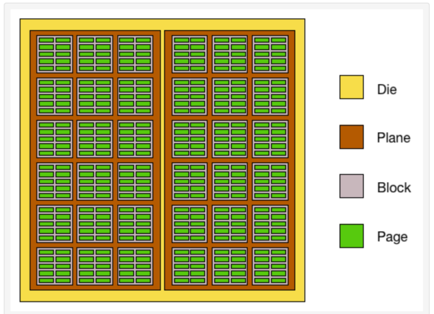


7

Paper “F2FS”

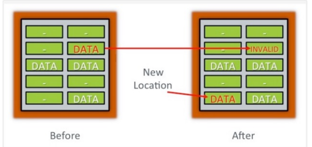
- NAND flash memory
- Sequential vs. random writes
  - Fragmentation, life time

8



NAND Flash Die Layout (image courtesy of AnandTech)

Operation	Area
Read	Page
Program (Write)	Page
Erase	Block

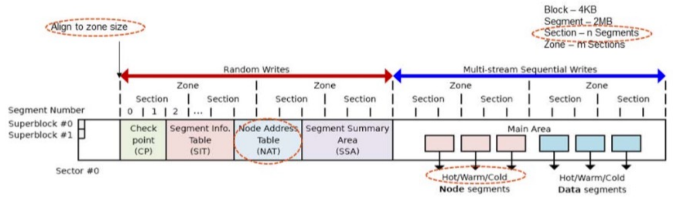


Updating a page in NAND flash. Note that the new page location does not need to be within the same block, or even the same flash die. It is shown in the same block here purely for ease of drawing.

9

Paper "F2FS"

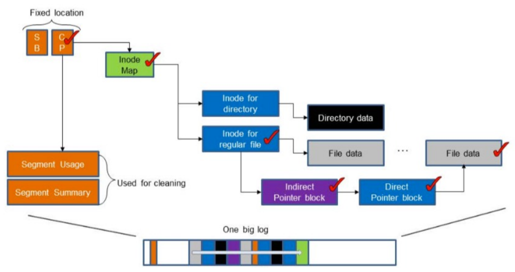
- Flash-Friendly On-Disk Layout
  - Segment as basic unit
  - Section: consecutive segments
  - Zone: consecutive sections
  - Node: inodes/indices; data: directory/user file data
  - Separation of meta data and "actual" data



10

Paper “F2FS”

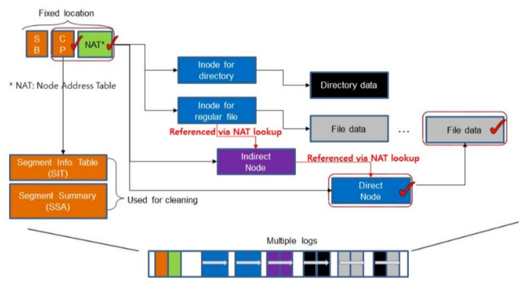
- Cost-Effective Index Structure
- LFS:



11

Paper “F2FS”

- Cost-Effective Index Structure
- F2FS:



12

Paper “F2FS”

- Wandering tree problem
  - LFS: when a file data is updated and written to the end of log, its **direct** pointer changes, its **indirect pointer block** is also updated, and upper index structures (inode, inode map, checkpoint block) are also changed

13

Paper “F2FS”

- Multi-Head Logging
  - Data temperature classification: hot, warm, cold
  - Six logging segments
- Cleaning
  - Section-level; foreground (need more sections) and background (periodic kernel thread)
  - Greedy for foreground, cost-benefit for background
- Adaptive Logging
  - Switch between normal and threaded logging

14