UNIVERSITY of WISCONSIN-MADISON
Computer Sciences Department

CS 537
Introduction to Operating Systems

Andrea C. Arpaci-Dusseau
Remzi H. Arpaci-Dusseau

File Allocation

Questions answered in this lecture:

What are typical file access patterns?

What are different approaches for tracking which blocks belong to a file?

What are the advantages and disadvantages of each approach?

Workloads

Motivation: Workloads influence design of file system

File characteristics (measurements of UNIX and NT)

- Most files are small (about 8KB)
- Most of the disk is allocated to large files
  - (90% of data is in 10% of files)

Access patterns

- Sequential: Data in file is read/written in order
  - Most common access pattern
- Random (direct): Access block without referencing predecessors
  - Difficult to optimize
- Access files in same directory together
  - Spatial locality
- Access meta-data when access file
  - Need meta-data to find data

Goals

OS allocates LBNs (logical block numbers) to meta-data, file data, and directory data

- Workload items accessed together should be close in LBN space

Implications

- Large files should be allocated sequentially
- Files in same directory should be allocated near each other
- Data should be allocated near its meta-data

Meta-Data: Where is it stored on disk?

- Embedded within each directory entry
- In data structure separate from directory entry
  - Directory entry points to meta-data

Allocation Strategies

Progression of different approaches

- Contiguous
- Extent-based
- Linked
- File-allocation Tables
- Indexed
- Multi-level Indexed

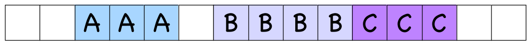
Questions

- Amount of fragmentation (internal and external)?
- Ability to grow file over time?
- Seek cost for sequential accesses?
- Speed to find data blocks for random accesses?
- Wasted space for pointers to data blocks?

Contiguous Allocation

Allocate each file to contiguous blocks on disk

- Meta-data: Starting block and size of file
- OS allocates by finding sufficient free space
  - Must predict future size of file; Should space be reserved?
- Example: IBM OS/360



Advantages

- Little overhead for meta-data
- Excellent performance for sequential accesses
- Simple to calculate random addresses

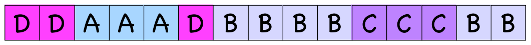
Drawbacks

- Horrible external fragmentation (Requires periodic compaction)
- May not be able to grow file without moving it

Extent-Based Allocation

Allocate multiple contiguous regions (extents) per file

- Meta-data: Small array (2-6) designating each extent
  - Each entry: starting block and size



Improves contiguous allocation

- File can grow over time (until run out of extents)
- Helps with external fragmentation

Advantages

- Limited overhead for meta-data
- Very good performance for sequential accesses
- Simple to calculate random addresses

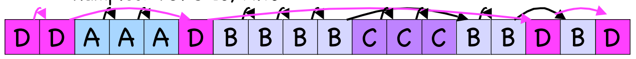
Disadvantages (Small number of extents):

- External fragmentation can still be a problem
- Not able to grow file when run out of extents

Linked Allocation

Allocate linked-list of fixed-sized blocks

- Meta-data: Location of first block of file
  - Each block also contains pointer to next block
- Examples: TOPS-10, Alto



Advantages

- No external fragmentation
- Files can be easily grown, with no limit

Disadvantages

- Cannot calculate random addresses w/o reading previous blocks
- Sequential bandwidth may not be good
  - Try to allocate blocks of file contiguously for best performance

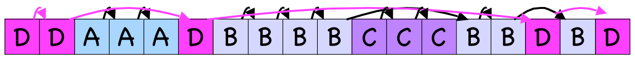
Trade-off: Block size (does not need to equal sector size)

- Larger --> ??
- Smaller --> ??

File-Allocation Table (FAT)

Variation of Linked allocation

- Keep linked-list information for all files in on-disk FAT table
- Meta-data: Location of first block of file
  - And, FAT table itself



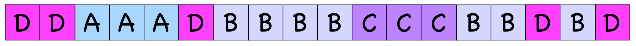
Comparison to Linked Allocation

- Same basic advantages and disadvantages
- Disadvantage: Read from two disk locations for every data read
- Optimization: Cache FAT in main memory
  - Advantage: Greatly improves random accesses

Indexed Allocation

Allocate fixed-sized blocks for each file

- Meta-data: Fixed-sized array of block pointers
  - Allocate space for ptrs at file creation time



Advantages

- No external fragmentation
- Files can be easily grown, with no limit
- Supports random access

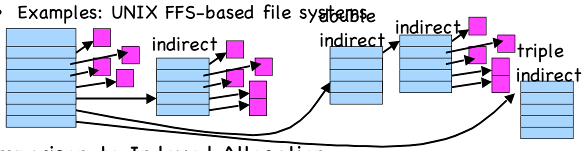
Disadvantages

- Large overhead for meta-data:
  - Wastes space for unneeded pointers (most files are small!)

Multi-Level Indexed Files

Variation of Indexed Allocation

- Dynamically allocate hierarchy of pointers to blocks as needed
- Meta-data: Small number of pointers allocated statically
  - Additional pointers to blocks of pointers
- Examples: UNIX FFS-based file systems



Comparison to Indexed Allocation

- Advantage: Does not waste space for unneeded pointers
  - Still fast access for small files
  - Can grow to what size??
- Disadvantage: Need to read indirect blocks of pointers to calculate addresses (extra disk read)
  - Keep indirect blocks cached in main memory