Administrivia

- Reminder: Quiz 5 Monday.

Slide 1

Minute Essay From Last Lecture

- Several sensible answers involving filename and attribute conventions. Also, maximum filesize could be different.

- One person mentioned how Linux doesn't insist (as much?) on filename extensions. True, but various programs sort of do — e.g., graphical file manager might take into account the "extension".

- One person mentioned differences in text file conventions — different line separators (right?).

Slide 2

Filesystem Implementation — Overview

- Last time we talked about many aspects of filesystem abstraction. After making decisions about what to implement — how?

- Recall(?) basic organization of disk:

  - Master boot record (includes partition table)

  - Partitions, each containing boot block and lots more blocks.

- How to organize/use those "lots more blocks"? Must keep track of which blocks are used by which files, which blocks are free, directory info, file attributes, etc., etc.

  Typically start with superblock containing basic info about filesystem, then some blocks with info about free space and what files are there, then the actual files.
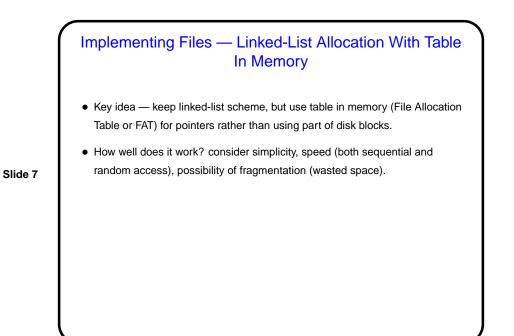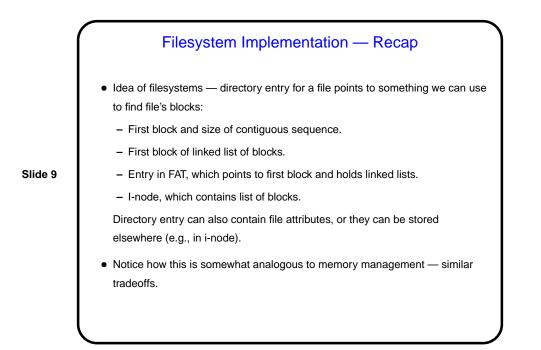
Implementing Files

- One problem is keeping track of which disk blocks belong to which files.

- No surprise — there are several approaches. (All assume some outside "directory"-type structure with some information about each file — a starting block, e.g.)
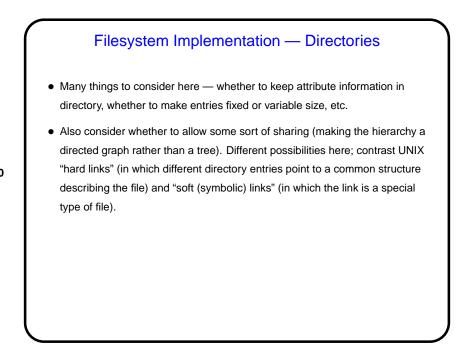
Implementing Files — Contiguous Allocation

Slide 5

- Key idea — what the name suggests, much like analogous idea for memory management.

- How well does it work? consider simplicity, speed (both sequential and random access), possibility of fragmentation (wasted space).

- Widely used long ago, abandoned, but now maybe useful again.

Implementing Files — Linked-List Allocation

Slide 6

- Key idea — organize each file's blocks as a linked list, with pointer to next block stored within block.

- How well does it work? consider simplicity, speed (both sequential and random access), possibility of fragmentation (wasted space).

Implementing Files — Linked-List Allocation With Table In Memory

- Key idea — keep linked-list scheme, but use table in memory (File Allocation Table or FAT) for pointers rather than using part of disk blocks.

- How well does it work? consider simplicity, speed (both sequential and random access), possibility of fragmentation (wasted space).
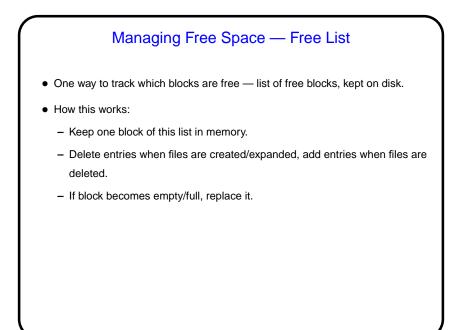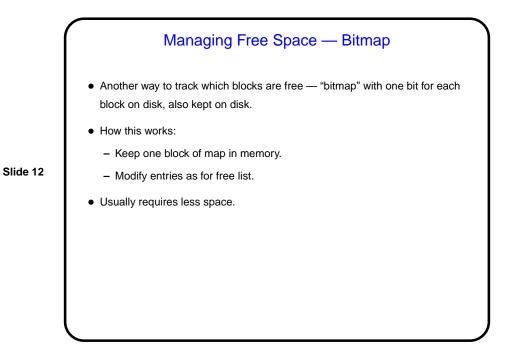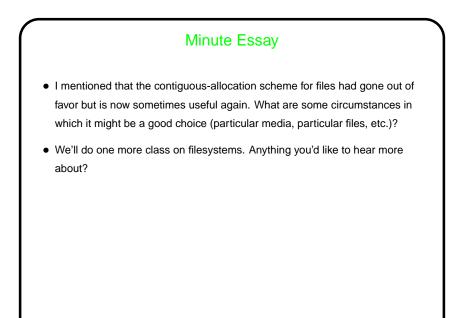
Slide 7

Implementing Files — I-Nodes

- Key idea — associate with each file a data structure ("index node" or i-node) containing file attributes and disk block numbers, keep in memory.

- How well does it work? consider simplicity, speed (both sequential and random access), possibility of fragmentation (wasted space).

Slide 8

Filesystem Implementation — Recap

Slide 9

- Idea of filesystems — directory entry for a file points to something we can use to find file's blocks:
  - First block and size of contiguous sequence.
  - First block of linked list of blocks.
  - Entry in FAT, which points to first block and holds linked lists.
  - I-node, which contains list of blocks.

  Directory entry can also contain file attributes, or they can be stored elsewhere (e.g., in i-node).

- Notice how this is somewhat analogous to memory management — similar tradeoffs.

Filesystem Implementation — Directories

Slide 10

- Many things to consider here — whether to keep attribute information in directory, whether to make entries fixed or variable size, etc.

- Also consider whether to allow some sort of sharing (making the hierarchy a directed graph rather than a tree). Different possibilities here; contrast UNIX "hard links" (in which different directory entries point to a common structure describing the file) and "soft (symbolic) links" (in which the link is a special type of file).

Slide 11

Managing Free Space — Free List

- One way to track which blocks are free — list of free blocks, kept on disk.

- How this works:
  - Keep one block of this list in memory.
  - Delete entries when files are created/expanded, add entries when files are deleted.
  - If block becomes empty/full, replace it.

Slide 12

Managing Free Space — Bitmap

- Another way to track which blocks are free — "bitmap" with one bit for each block on disk, also kept on disk.

- How this works:
  - Keep one block of map in memory.
  - Modify entries as for free list.

- Usually requires less space.

Minute Essay

Slide 13

- I mentioned that the contiguous-allocation scheme for files had gone out of favor but is now sometimes useful again. What are some circumstances in which it might be a good choice (particular media, particular files, etc.)?

- We'll do one more class on filesystems. Anything you'd like to hear more about?

Minute Essay Answer

Slide 14

- It could be a good choice for a write-once medium, assuming every files is written all at once (rather than, say, writing some blocks of one file, then some blocks of another, and so forth).

- It might even be a good choice in situations in which faster access outweighs other considerations (such as difficulties in making files bigger, or the potential for fragmentation).