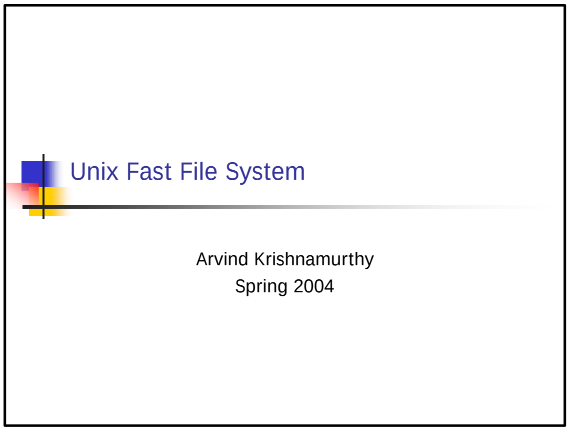
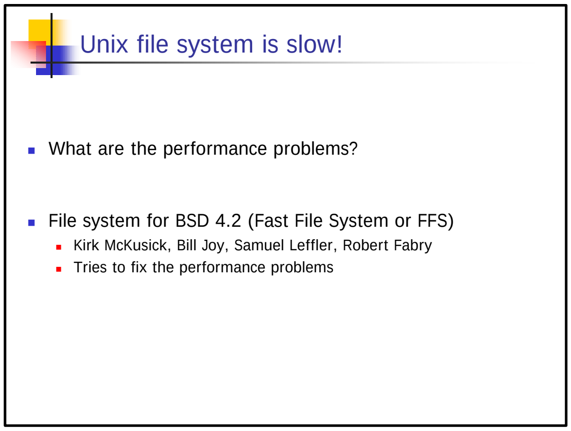


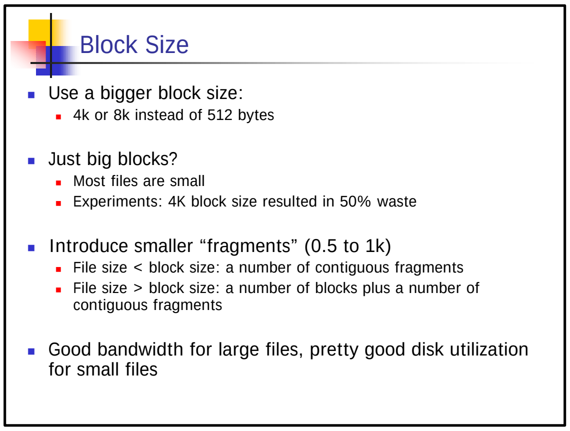
Unix Fast File System

Arvind Krishnamurthy
Spring 2004



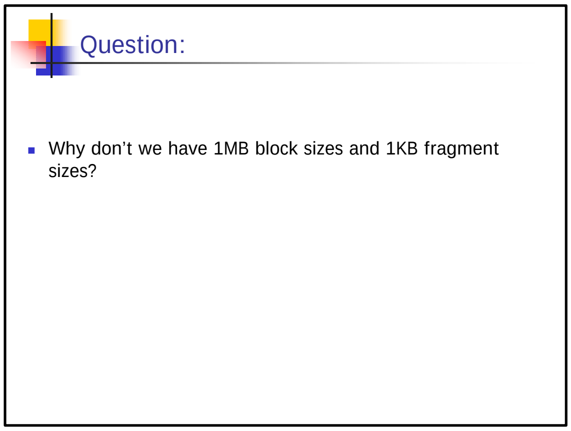
Unix file system is slow!

- What are the performance problems?
- File system for BSD 4.2 (Fast File System or FFS)
  - Kirk McKusick, Bill Joy, Samuel Leffler, Robert Fabry
  - Tries to fix the performance problems



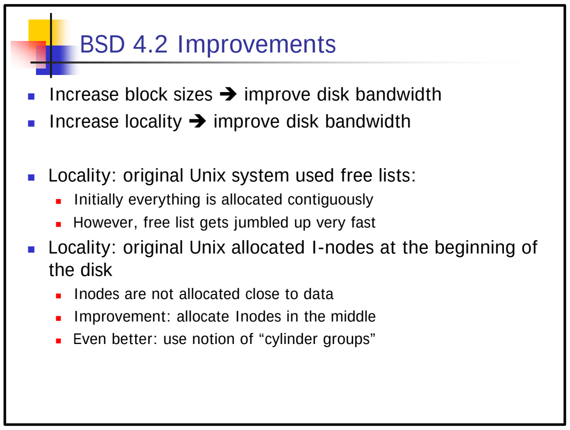
Block Size

- Use a bigger block size:
  - 4k or 8k instead of 512 bytes
- Just big blocks?
  - Most files are small
  - Experiments: 4K block size resulted in 50% waste
- Introduce smaller “fragments” (0.5 to 1k)
  - File size < block size: a number of contiguous fragments
  - File size > block size: a number of blocks plus a number of contiguous fragments
- Good bandwidth for large files, pretty good disk utilization for small files



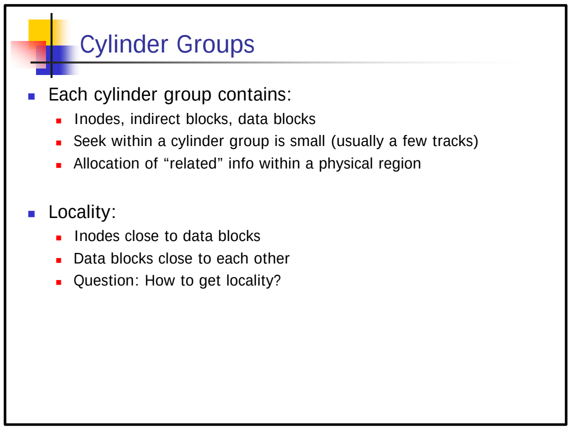
Question:

- Why don’t we have 1MB block sizes and 1KB fragment sizes?



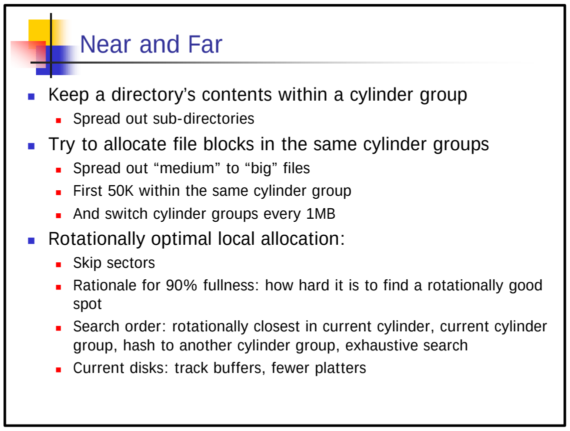
BSD 4.2 Improvements

- Increase block sizes ➔ improve disk bandwidth
- Increase locality ➔ improve disk bandwidth
- Locality: original Unix system used free lists:
  - Initially everything is allocated contiguously
  - However, free list gets jumbled up very fast
- Locality: original Unix allocated I-nodes at the beginning of the disk
  - Inodes are not allocated close to data
  - Improvement: allocate Inodes in the middle
  - Even better: use notion of “cylinder groups”



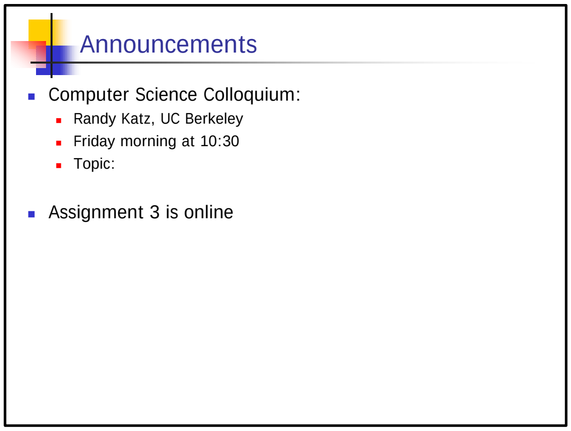
Cylinder Groups

- Each cylinder group contains:
  - Inodes, indirect blocks, data blocks
  - Seek within a cylinder group is small (usually a few tracks)
  - Allocation of “related” info within a physical region
- Locality:
  - Inodes close to data blocks
  - Data blocks close to each other
  - Question: How to get locality?



Near and Far

- Keep a directory's contents within a cylinder group
  - Spread out sub-directories
- Try to allocate file blocks in the same cylinder groups
  - Spread out “medium” to “big” files
  - First 50K within the same cylinder group
  - And switch cylinder groups every 1MB
- Rotationally optimal local allocation:
  - Skip sectors
  - Rationale for 90% fullness: how hard it is to find a rotationally good spot
  - Search order: rotationally closest in current cylinder, current cylinder group, hash to another cylinder group, exhaustive search
  - Current disks: track buffers, fewer platters



Announcements

- Computer Science Colloquium:
  - Randy Katz, UC Berkeley
  - Friday morning at 10:30
  - Topic:
- Assignment 3 is online