

# **CS241 System Programming**

## **File System Implementation (V)**

Klara Nahrstedt

Lecture 24

3/17/2006



# Content

- File System Layout
- Allocation of Disk Space
  - Contiguous Allocation
  - Linked List Allocation
  - Indexed Allocation

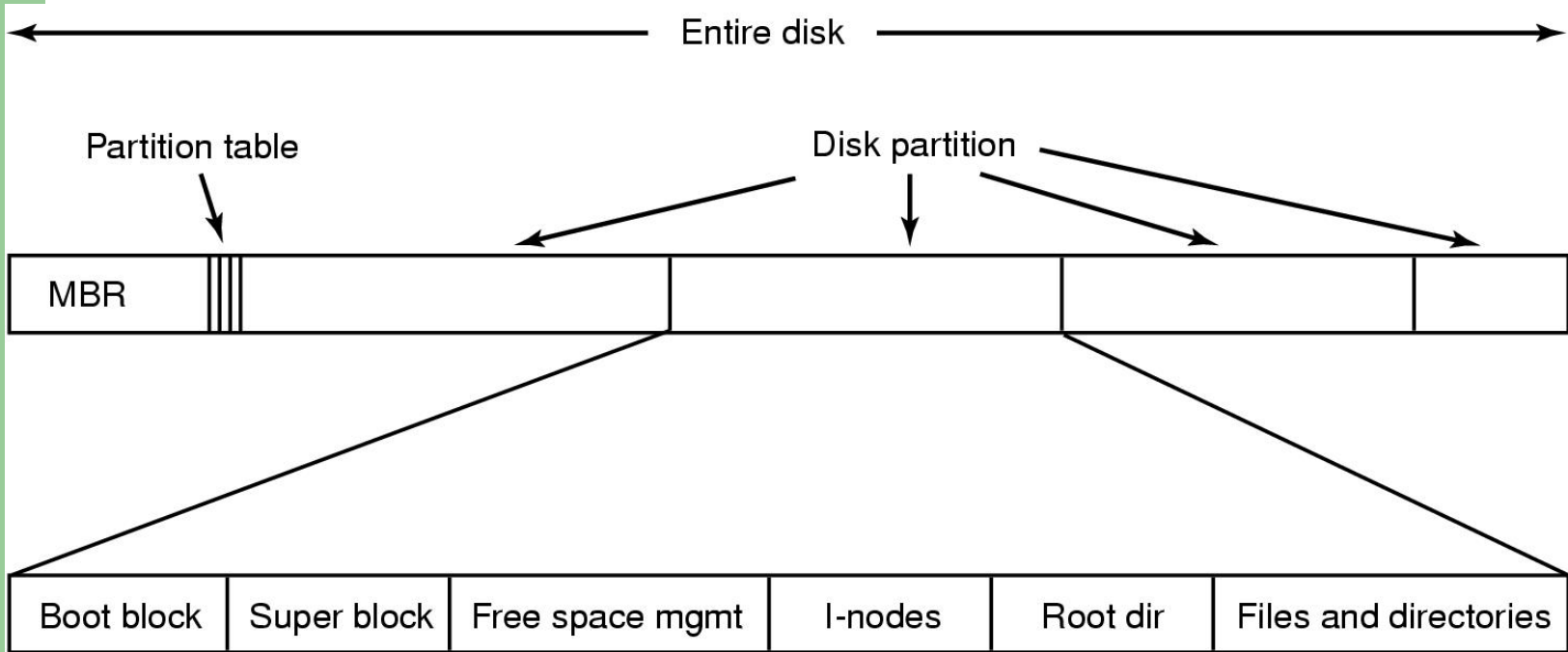
# Administrative

- R: Ch 4 pp92-135
- MP3 is posted, due April 3, 2006
- Quiz 6 is March 17, 2006

# File System Layout

- Most disks are divided up to one or more **partitions** with independent file systems in each partition
- Sector 0 is called **MBR – Master Boot Record**
  - Used to boot the computer
  - End of MBR contains partition table
  - MBR is executed by BIOS, BIOS is read when computer is booted
  - MBR locates active partition, reads **boot block**, executes it
  - Boot block includes OS in that partition
- **Superblock** – key parameters about the file system
  - Magic number to identify the file system type, number of blocks in the file system, other key admin information

# File System Implementation

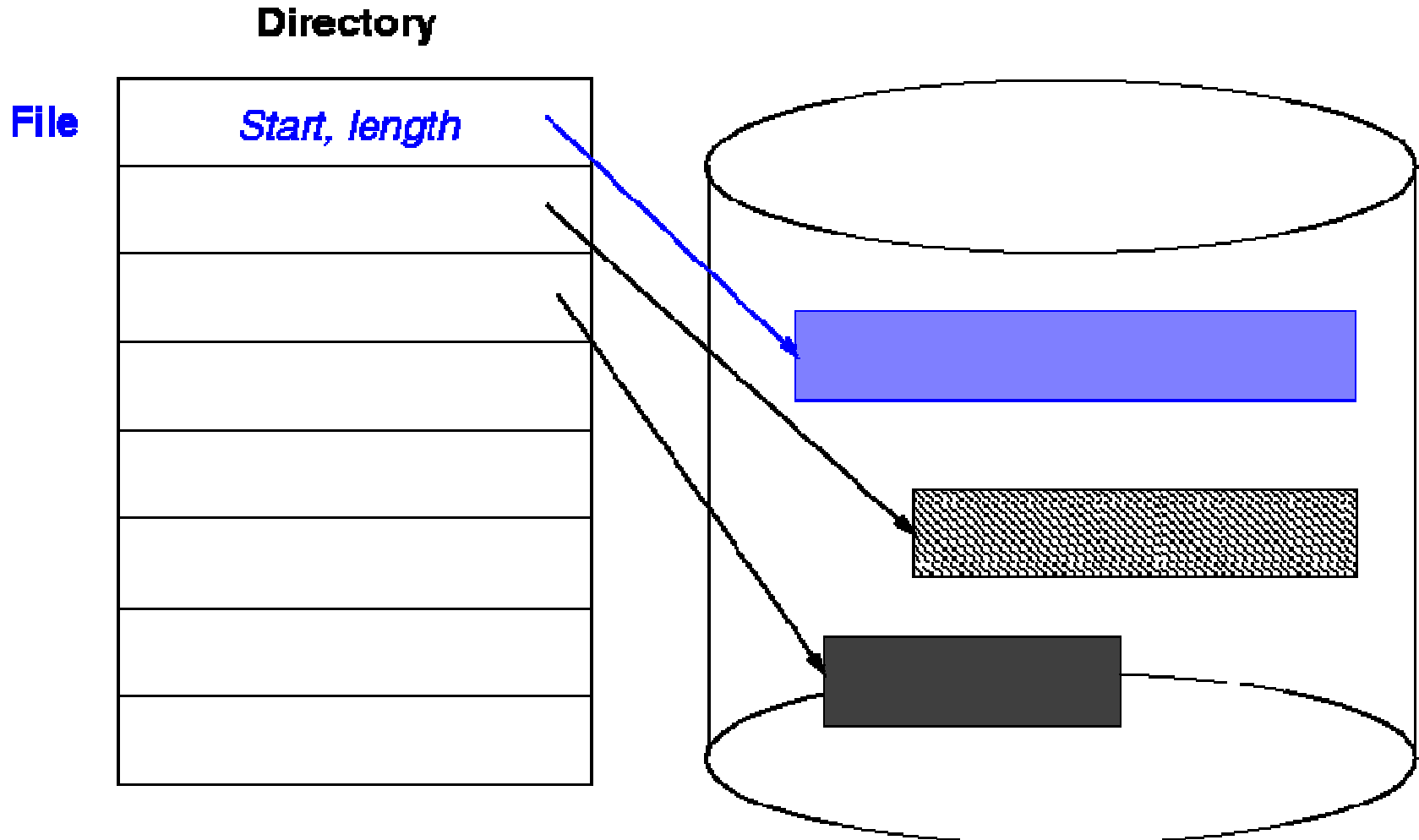


A possible file system layout

# Allocation of Disk Space

- Low level access methods depend upon the disk allocation scheme used to store file data
  - Contiguous allocation
  - Linked list allocation
  - Block allocation

# Contiguous Allocation



# Contiguous Allocation Issues

- Access method suits sequential and direct access
- Directory table maps files into starting physical address and length
- Easy to recover in event of system crash
- Fast, often requires no head movement and when it does, head only moves one track

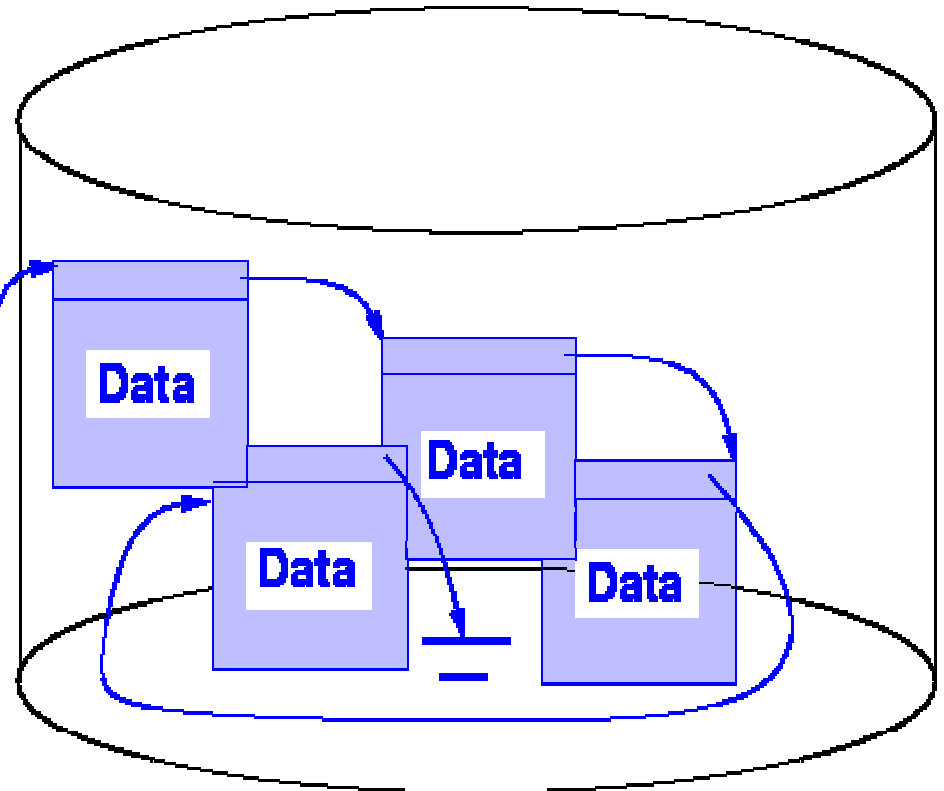
# Contiguous Allocation Issues

- File is allocated large contiguous chunks
- User knows size of the file
- Expanding the file requires copying
- Dynamic storage allocation - first fit, best fit
- External fragmentation occurs on disk
- Users tend to overestimate space => internal fragmentation

# Linked Allocation

Directory

<b>File</b>	<b>Address</b>



# Linked List Allocation Issues

- Each file is a linked list of chunks
- Pointers in list are not accessible to user
- Directory table maps files into head of list for a file
- A node in the list can be a fixed size physical block or a contiguous collection of blocks
- Easy to use - no estimation of size necessary

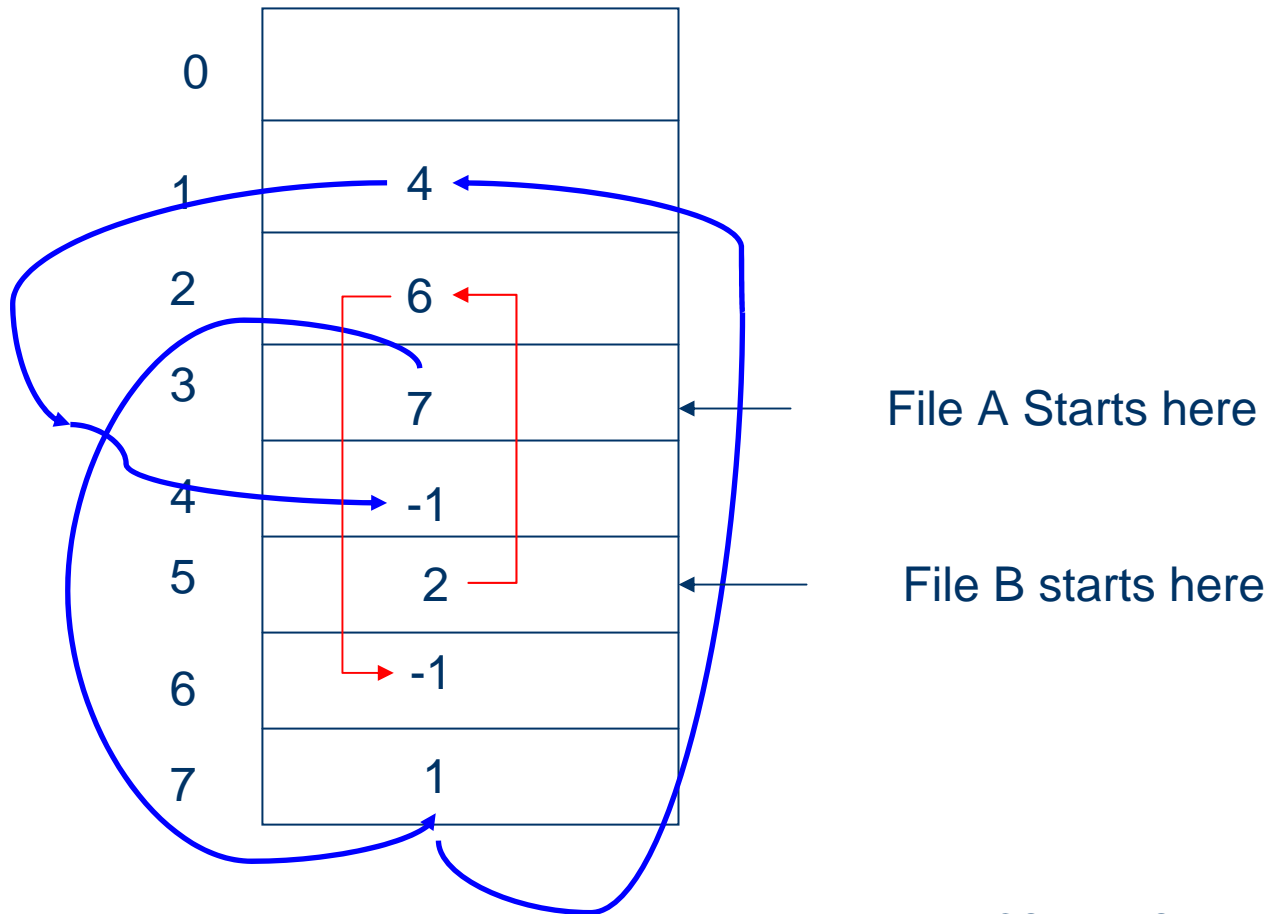
# Linked List Allocation Issues

- Can grow in middle and at ends
- Space efficient, little fragmentation
- Slow - defies the principle of locality. Need to read through linked list nodes sequentially to find record of interest blocks
- Suited for sequential access but not direct access

# Linked List Allocation Issues

- Disk space must be used to store pointers (if disk block is 512 bytes, and disk address requires 4 bytes, then the user sees blocks of 508 bytes)
- Not very reliable. System crashes can scramble files being updated
- Important variation on linked allocation method: 'file-allocation table' (FAT) - OS/2 and MS-DOS

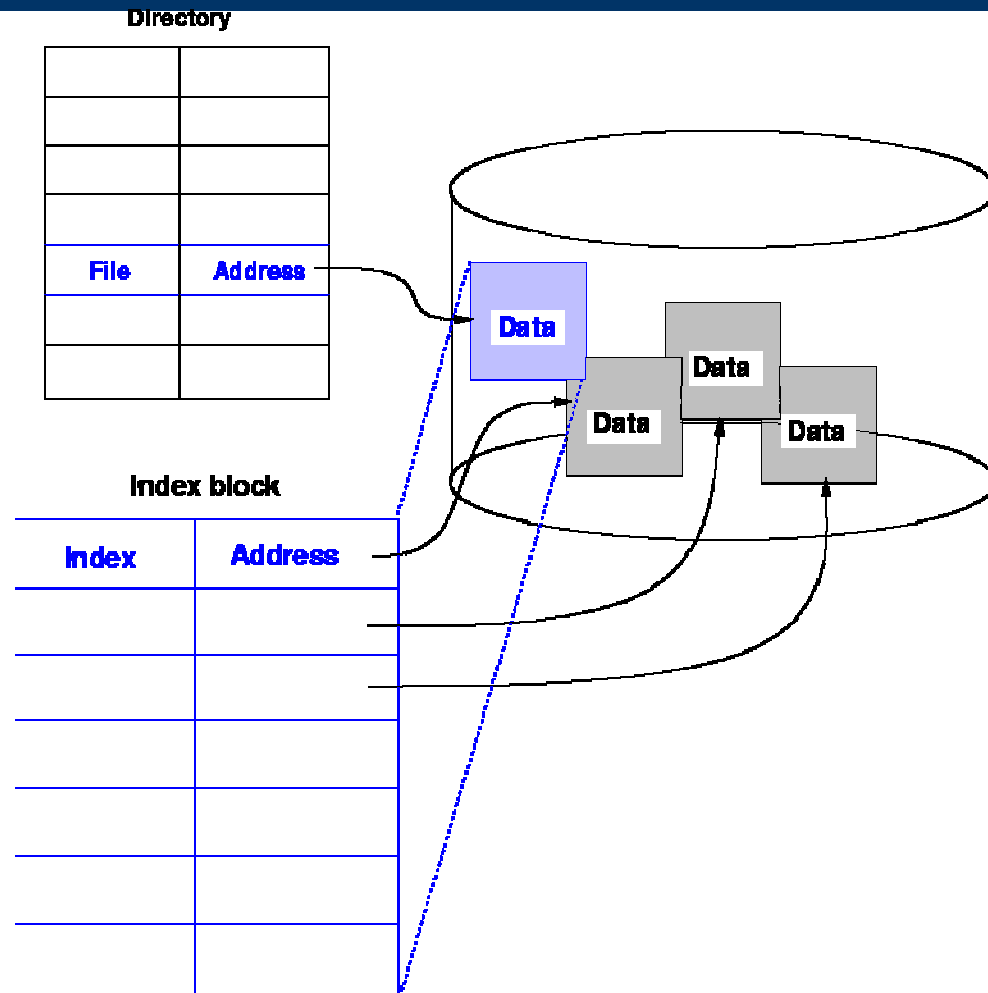
# File Allocation Table



# Linked List Allocation Issues

- Summary: linked allocation solves the external fragmentation and size-declaration problems of contiguous allocation,
- However, it can't support efficient direct access

# Indexed Allocation



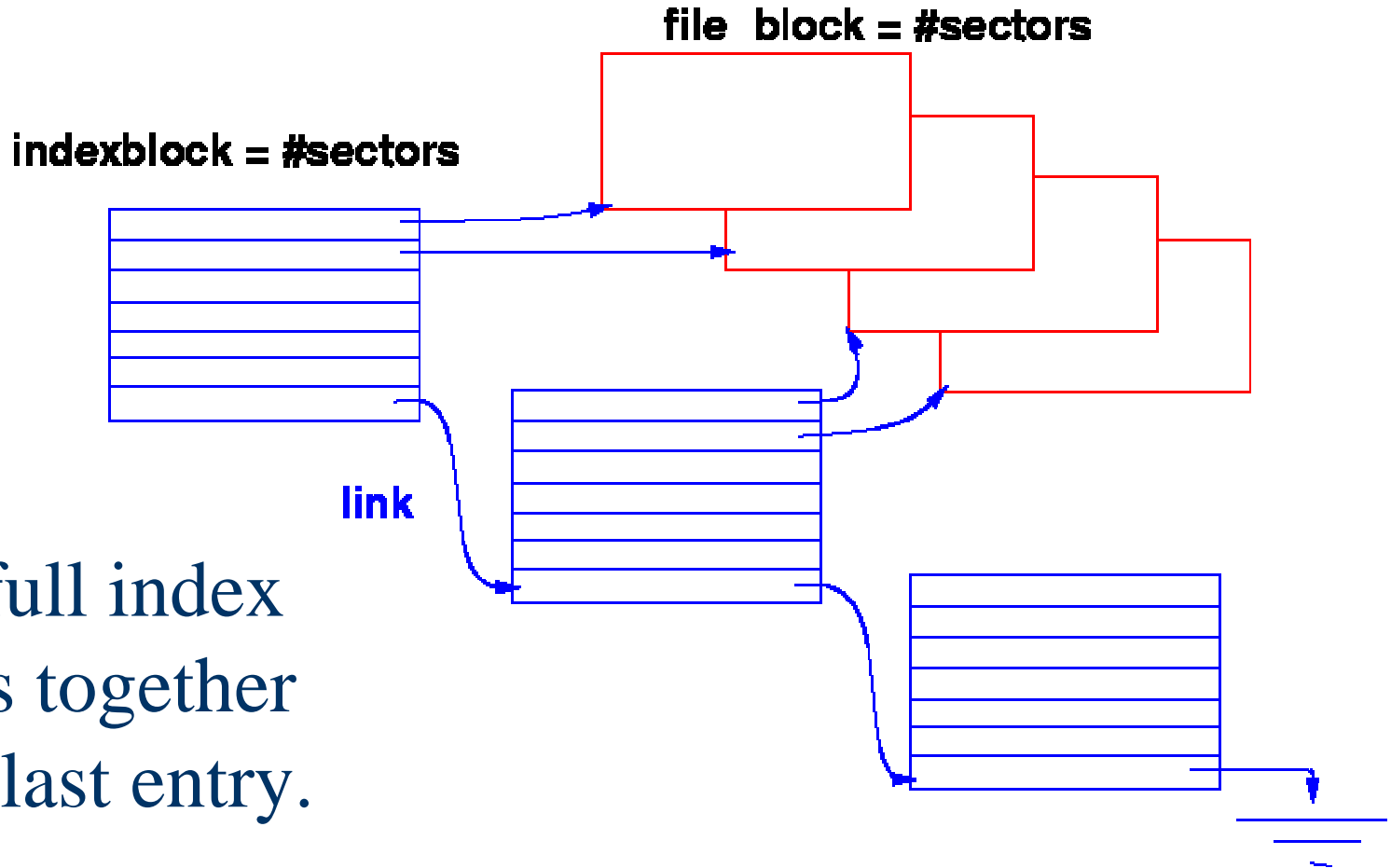
# Indexed Allocation

- Solves external fragmentation
- Supports sequential, direct and indexed access
- Access requires at most one access to index block first. This can be cached in main memory

# Indexed Allocation

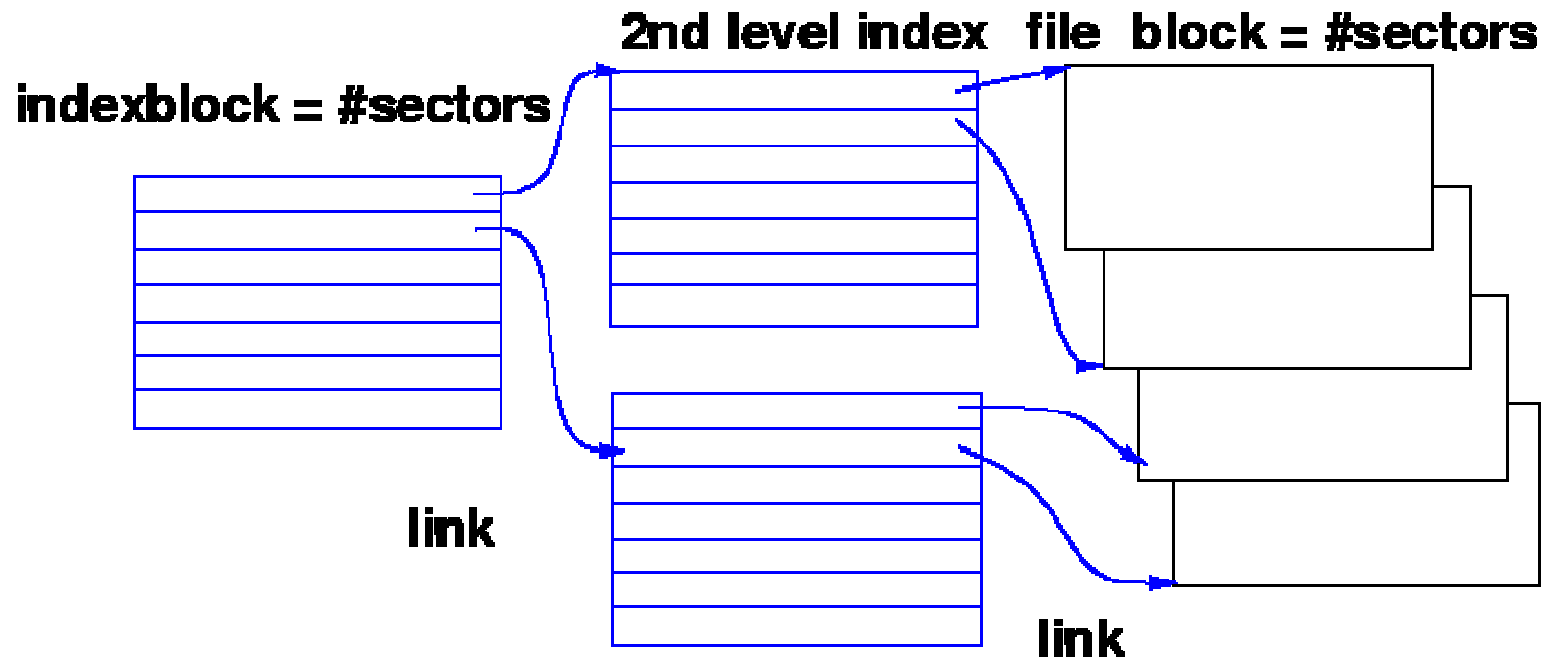
- File can be extended by rewriting a few blocks and index block
- Requires extra space for index block, possible wasted space
- Extension to big files issues

# Other Forms of Indexed File Linked



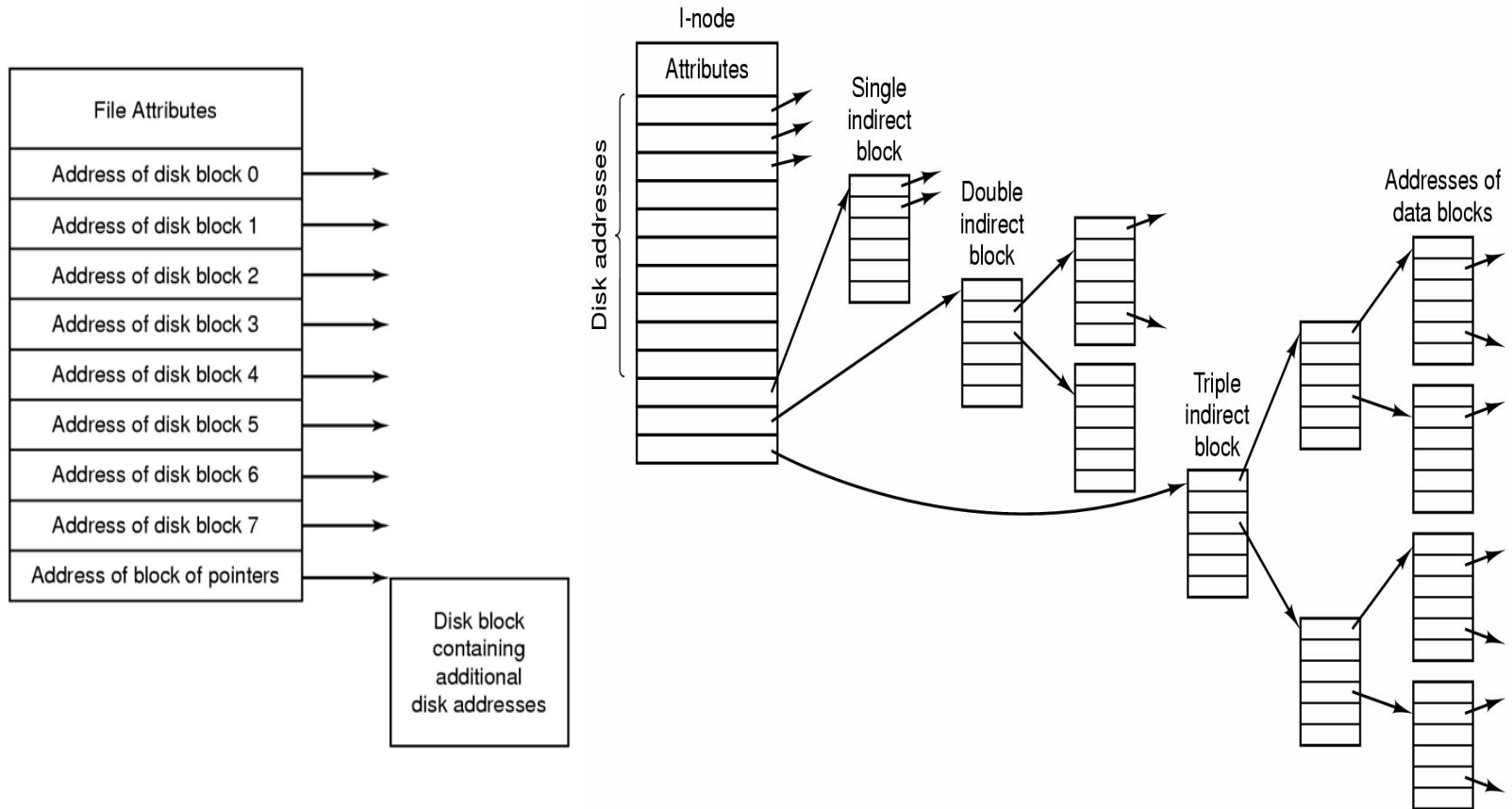
Link full index blocks together using last entry.

# Other Forms of Indexed File - Multilevel Index



Multiple levels of index blocks

# UNIX I-Node



# Questions

- Which of the following is not a problem associated with contiguous allocation of disk space for a file?
  - External fragmentation of disk space or
  - Frequent copying or
  - Direct access
- Which of the following methods would you choose if the file requires frequent direct access and also external fragmentation is to be avoided (to keep disk utilization high)?
  - Linked allocation or
  - Contiguous allocation or
  - Indexed allocation