- *Better utilization of computing resources*
- *Automation (convenience and scheduling)*

Chapter 1 — Multi-Programming

Chapter 2 — OS and CA for Multi-Programming

- *Sharing Resources between Multiple Processes*
- *Management of Multiple Processes*

Chapter 3 — Process Management for Multi-Programming

- *Sharing CPU between Multiple Processes*
- *Management of Performance Requirements of Processes*

Chapter 4 — CPU Scheduling for Many Processes

- *Shared Data and Variables between Multiple Processes*
- *Race Conditions*

Chapter 5 — Process Synchronization

Chapter 6 — Deadlock

- *Using Synchronization Tools such as Semaphores*
- *Deadlock due to Inappropriate Use of Synchronization*
- *Deadlock Handling*

- *Multi-programming*
- *Sharing of Main Memory*

Chapter 7 — Memory Management

Chapter 8 — Virtual Memory

- *Loading and Execution of Multiple Processes at the same time*
- *Main Memory Management and Allocation to Multiple Processes*
- *Virtualization of Memory*

Chapter 9 — File Systems

- *Data storage as Files*
- *Files as a Virtual data unit*

Chapter 10 — IO Systems

- *Dealing with different IO devices*
- *Virtualization of devices*
- *Mechanism hard disk request Scheduling*

Chapter 11 — Distributed Systems

- *Virtualization of computer systems*

COMPS267F Chapter 9

File Systems

Dr. Andrew Kwok-Fai LUI

concept of files

Concept of Files

- A file is a set of data stored in a secondary device under one name and a set of attributes
  - Files represent both data and programs in common operating systems
  - A file is usually seen as a sequence of data bytes
  - The meaning of the data not carried with the file itself
- The data stored in a file is in sequence
  - The sequential order in the data is kept

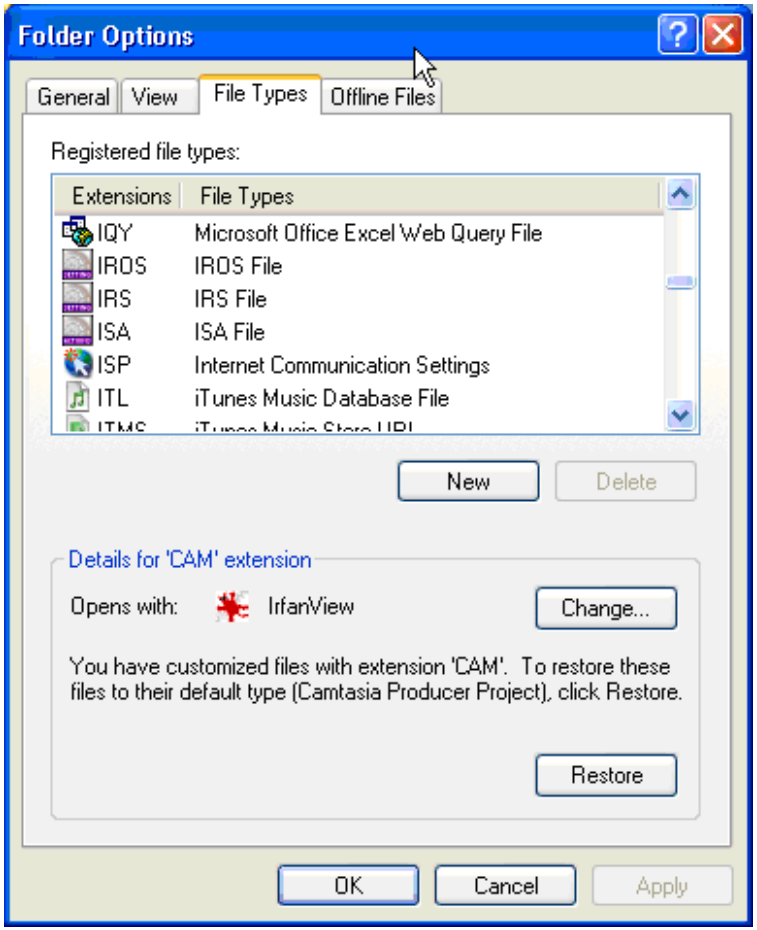
Meaning and Attributes of Files

- There are various types of data stored in a file
  - Executable code
  - Constants for a program
  - A Word document
  - An image
  - A video
  - A ZIP file

Meaning and Attributes of Files

- The file itself does not necessarily carry information about its purpose
  - Meta-information
- A set of attributes that could indicate to the operating systems some indications of its purpose
  - Examples of file attributes include
    - Name, identifier (internal record), type, location, size, protection, time stamp, user identification

Meaning and Attributes of Files



Meaning and Attributes of Files

- Different OS carry different sets of attributes for files
  - Directories, network connection and device drivers are often regarded as a file
    - Special attributes must exist to distinguish the type
- Windows has tried to support file meaning interpretation
  - A program is distinguished with the execute bit in the file attribute in UNIX and with the file extension in Windows
  - A registration scheme for file suffix to know the relevant handling applications

Meaning and Attributes of Files

- MIME based mail or HTTP web browsing supports intelligent applications
  - Meaning of a file is carried with the file externally as a MIME content type
  - Web browser and servers generally knows how to handle the file it receives

```
POST /foo HTTP/1.1
Content-Length: 68137
Content-Type: multipart/form-data; boundary=---------------9747672998524989295316105 75

-----------------------------9747672998524989295316105 75
Content-Disposition: form-data; name="description"

some text
-----------------------------9747672998524989295316105 75
Content-Disposition: form-data; name="myFile"; filename="foo.txt"
Content-Type: text/plain

(content of the uploaded file foo.txt)
-----------------------------9747672998524989295316105 75--
```

file operations

Essential File Operations

- Creating a new file (allocating space, associate the space with the file)
- Writing a file
- Reading a file
- Repositioning within a file (or traversing in a file)
- Deleting a file (free the allocated space)
- Truncating a file (removing part of a file)

File Open

- Most operating systems require an explicit open operation on a file before the other operations could be applied
  - The key advantage is efficiency
  - The location and attributes of the named file can first be found before actual reading begins
  - Allocation of buffers and other data structures

File Open

- An open operation is to reduce this time consuming search to once and then the record of opened file is kept in an open-file table for further references
  - Some systems open file implicitly when a file is used for the first time
  - Each process has its own open-file table, which points to entries in the system-wide open-file table
  - Synchronization rules apply to the reading and writing of files

File Open

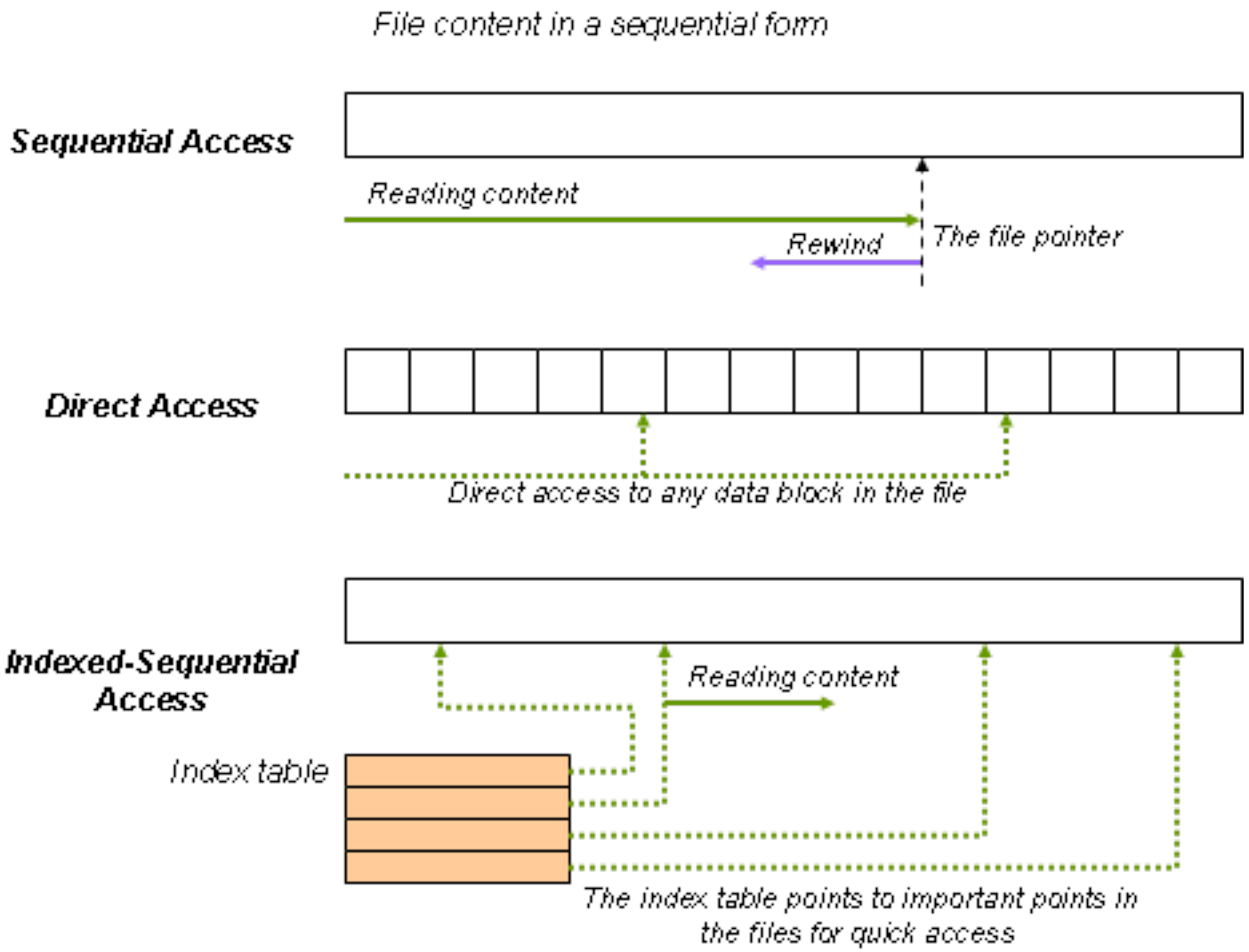
- An open file generally has associated attributes kept in memory for efficient access
  - File pointer to keep track of the current read/write head (current read/write location)
  - File open count to keep track of number of process that has opened a particular file
  - Disk location of the file
  - Access rights

file access methods

File Data Access

- Methods to access the file content
- The most common methods are given in the following
  - Sequential
  - Direct
  - Indexed-sequential
- Give different performance characteristics
- Requires explicit file system design for support

File Data Access



File Data Access

- Sequential
  - The file is processed one data at a time from start to end
  - Rewind operation to move back to the beginning.
- Direct
  - Arbitrary data block access directly using a block number
    - Similar to array indexing
- Indexed-sequential
  - Sequential access + bookmarks at key places
    - The file pointer can be relocated.
    - Useful for a very long file.
  - The index itself can become too large
  - Two level indexing
    - Master index (in memory) and secondary index (in hard disk)

directory systems

Directory Systems

- The OS provides directory as a means to manage named files
  - Files of the same name can co-exist in a system if they are placed in a different directory
  - Essential for multi-users system
- Objectives of directory system include the following
  - Efficient location and access to file
  - File naming is convenient to users
  - Logical grouping of files

Directory Systems

- Directory can be viewed as a file that has entries listing all of the member files
  - Each directory entry for a file can contain information such as file name, file type, location, and size

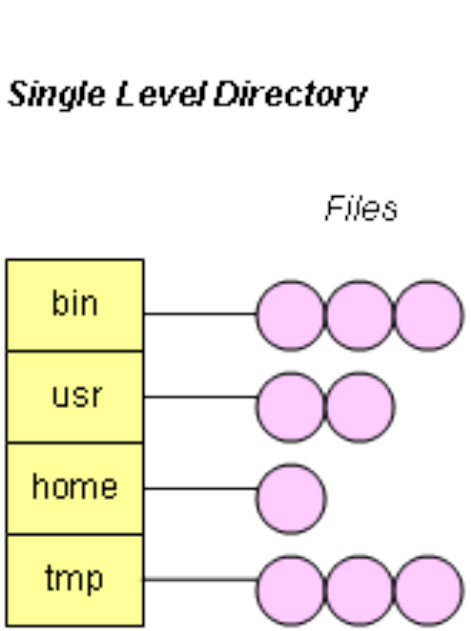
Directory Systems

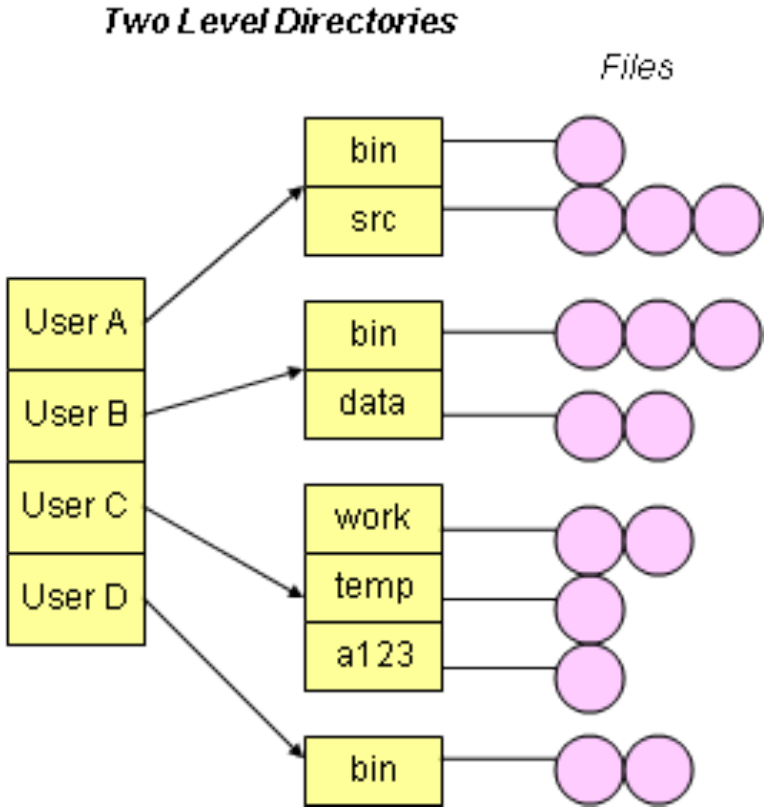
- Directory systems can be implemented in a number of ways
  - Single level directory
  - Two level directories
  - Tree structured directories
  - Acyclic-graph directories

Single Level Directory and Two Level Directory

- Single level directory
  - All the users share the same directory structure.
  - File-naming problem
- Two level directories
  - Each user has own user file directory
  - When a user logs on, OS searches the master file directory that is indexed by user name or account number.
  - Users are isolated from each other and they cannot share files.
    - A major disadvantage when users cooperate on a task that requires accessing other user files.

Single Level Directory and Two Level Directory





Tree structured directories

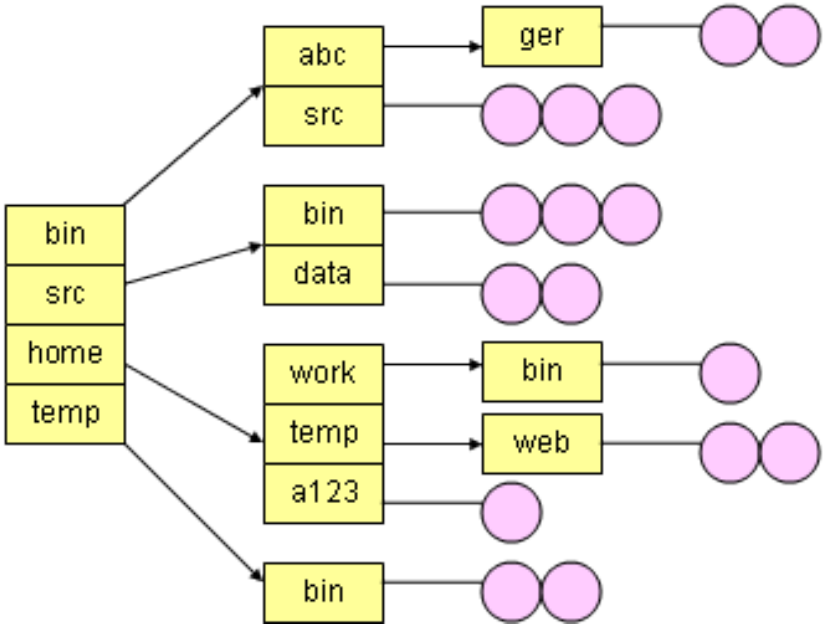
- Expending the two-level directory to a tree of multi-levels
  - Users can create their own sub-directories
  - Directories in both UNIX and MSDOS are tree-structured.
  - The tree has a root directory
  - Each file in the system has a unique path name
  - To uniquely specify a file, we have to use a path name such as root/progs/progB

Acyclic-graph directories

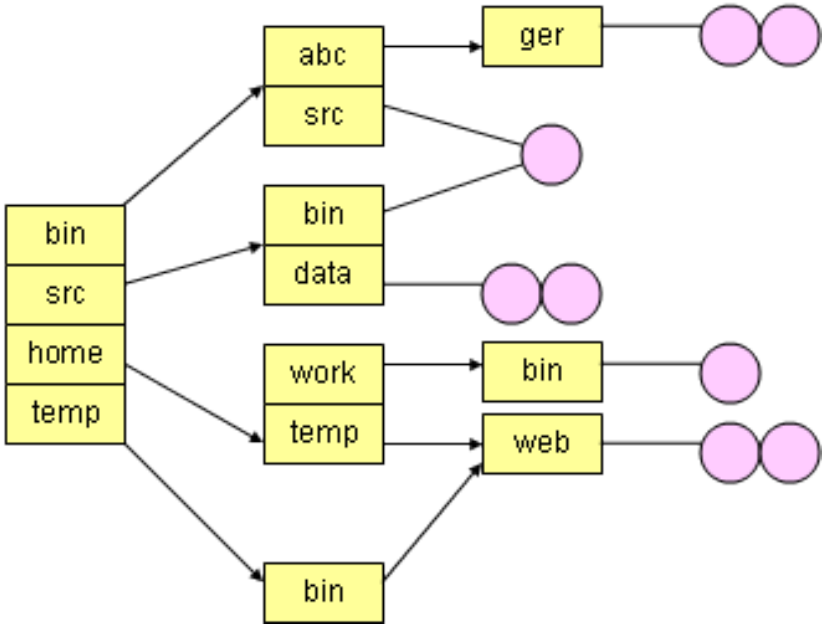
- An extension of the tree-structured directory system
  - Allowing directories to share sub-directories and files
  - Implementation can be tricky
  - BSD UNIX uses a directory entry called a link
    - A link is actually a pointer to another file or sub-directory

Tree Directories and Acyclic-graph Directories

Tree Structured Directories



Acyclic Graph Directories



file protection

File Protection

- Two aspects of file protection
  - Protection from physical damage
    - Accomplished by making duplicate copies of files
  - Protection from improper or illegal access
    - Implementing limiting different types of file access.
    - Some of the more common operations on a file are read, write, execute, append, delete, and list attribute.
    - A better implementation allows user level privilege differentiation.

File Protection

- Three ways of file protection
  - Passwords
  - Access Lists
    - An user access list is associated with each file and directory
  - Access Groups
    - Similar to Access Lists, but users are put into groups and access privilege is specified for a group

File Protection

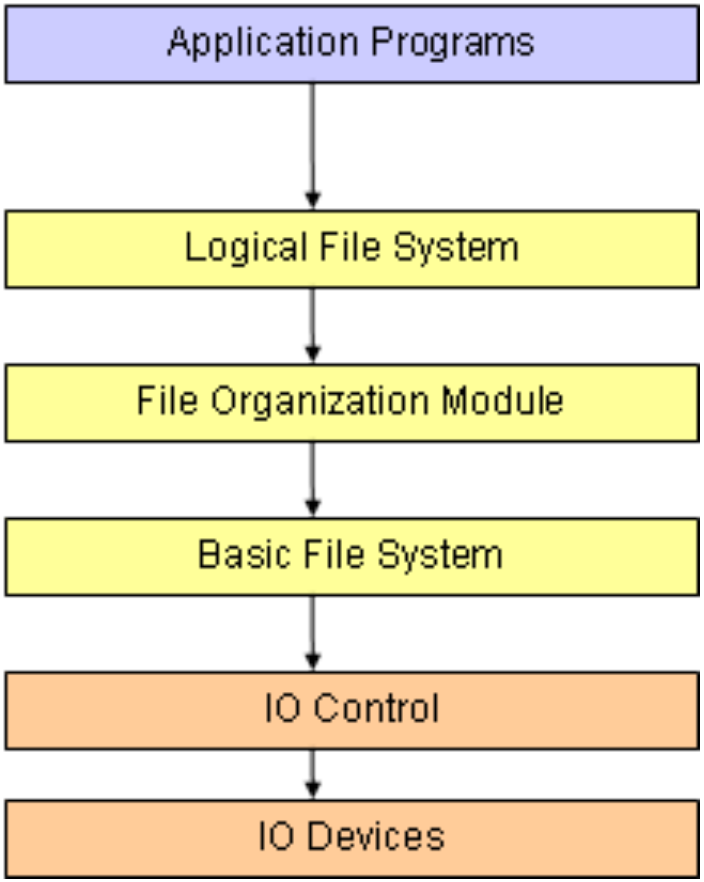
```
drwxr-xr-x   2 mt258    mt258        8192 May 31  2002 guest
-rw-r--r--   1 mt258    mt258         460 Oct 17  2003 index.htm
-rw-------   1 mt258    mt258         460 Oct 16  2003 index.htm.local
-rw-------   1 mt258    mt258         488 Oct 16  2003 index.htm.plbpc005
drwxr-xr-x   6 mt258    mt258        8192 Nov 16  2004 mt258
drwxr-xr-x   7 mt258    mt258        8192 Sep 22 11:41 mt258-2004
drwxr-xr-x   9 mt258    mt258        8192 Sep 22 11:41 mt258-2005
```

file system

File System

- The module in an OS that supports file operations
  - Interface between users (programs and end-users) and storage
  - Supported by secondary storage
    - In-place rewritten of data for efficiency
    - Direct access to any data block on hard disk for efficiency
    - Data redundancy for protection

File System Structure



Layers in File System Structure

- Logical File System manages metadata information of files
  - Also manages the directory structure.
  - File structure is managed using file control blocks
- File Organization Module knows their logical and physical structure.
  - Translate logical location (0, 1, 2, ...) into physical location
- Basic File System needs to distinguish appropriate device driver for a particular instruction.
  - Each physical block across multiple devices is uniquely identified by a numeric disk address (drive, cylinder, track, and sector)
- Device Drivers are responsible for transferring information between the main memory and the disk system.
  - Translates high-level instructions such as retrieve block A into lower level instruction that involves hard disk controls

file space management

File Space

- The file space is the place where the file content and metadata are actually stored
  - Hard disks are common
  - Tape drive, CDROM, and even RAM

File Space Management

- File space management is similar to memory management
  - Book-keeping the free and used parts of the file space
  - Recording the owner of a file space portion
- Operations of file space management
  - Allocating and de-allocating file space to newly created files
  - Efficient access
  - High file space utilization
  - Minimal overhead

File Space Management

- Issues of file space management
  - Capacity of storage devices such as hard disks has grown rapidly due to technological advances
  - Average size of files has grown even more dramatically
    - Multimedia applications

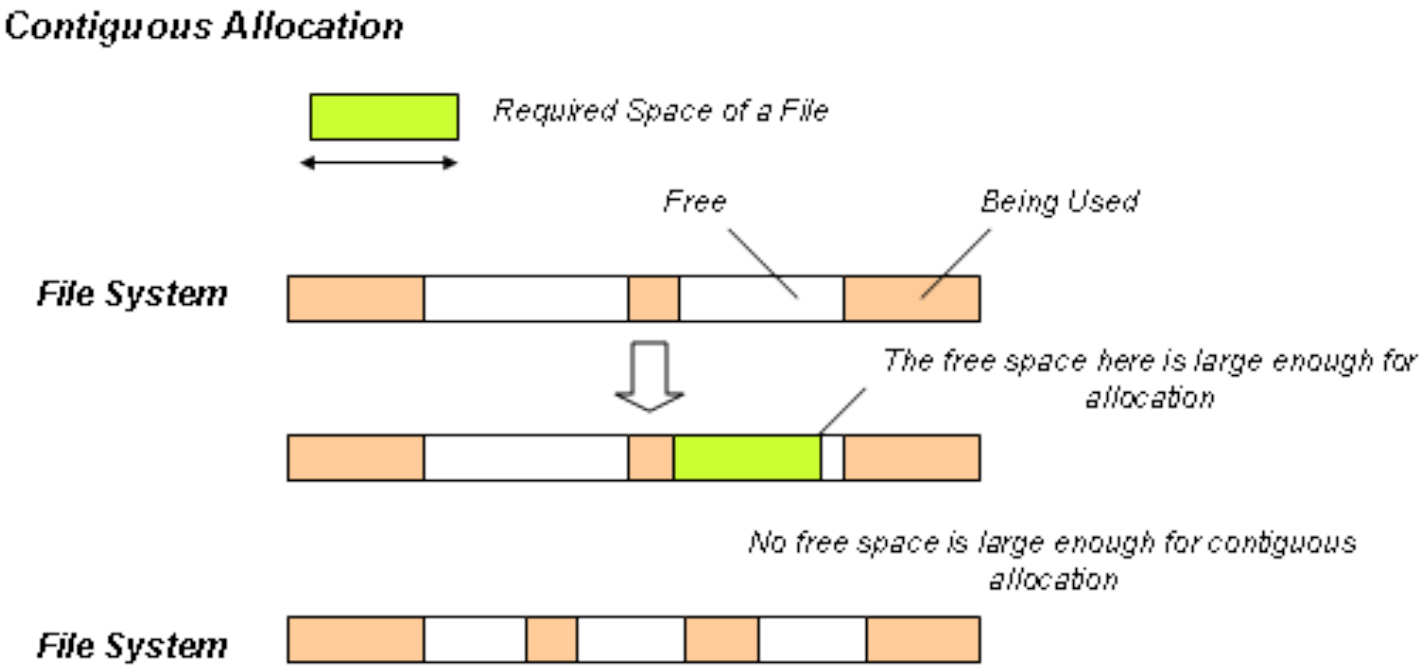
File Space Allocation

- Common approaches of allocating file space for new files
  - Contiguous Allocation
  - Linked Allocation
  - Indexed Allocation

Contiguous Allocation

- Each file is a set of contiguous blocks on the disk
  - Number of disk seeks for accessing contiguous allocated files is minimal (only the search for the first block is required)
  - Finding space for new files is difficult due to fragmentation
  - Determining how much space is to be allocated to a file is often not known at file creation
    - File data are written after a file is created
  - If some kind of best-fit strategy is used then, when the file is to be extended, the space around the file might well have been allocated to other files

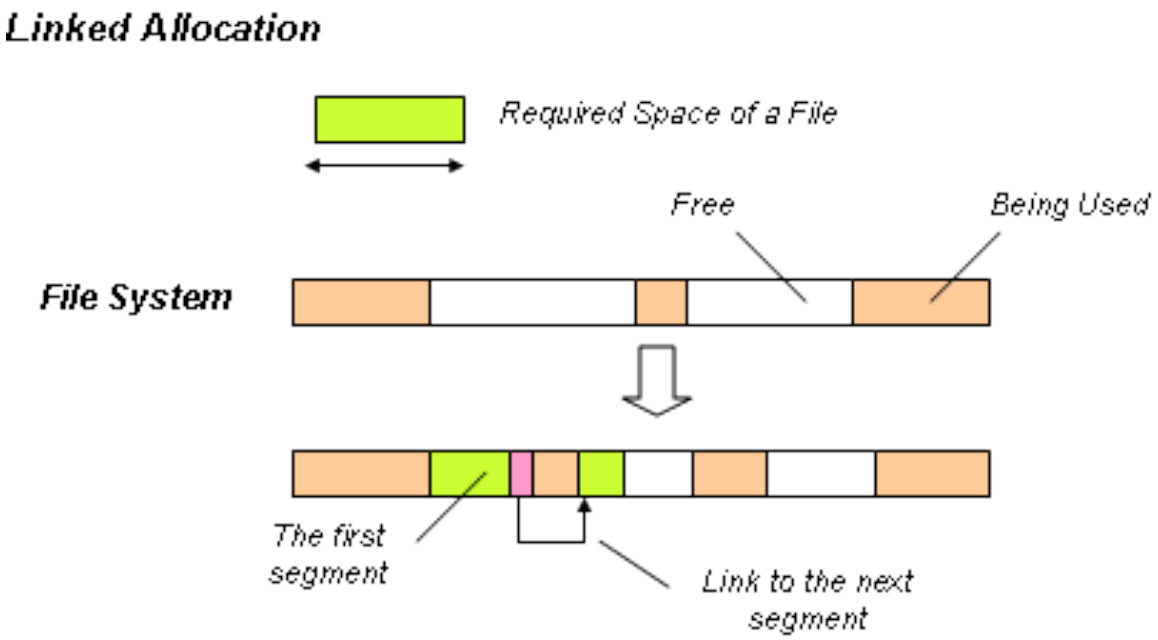
Contiguous Allocation



Linked Allocation

- A file consists of a linked list of disk blocks
  - Disk blocks may be located anywhere on the disk.
  - File extension is handled easily
    - Additional free disk block is simply allocated and linked to the list
  - Support sequential-access only by following the pointers
    - No random access
  - Pointers also take space
  - Potential problem with the pointer when corrupted

Linked Allocation



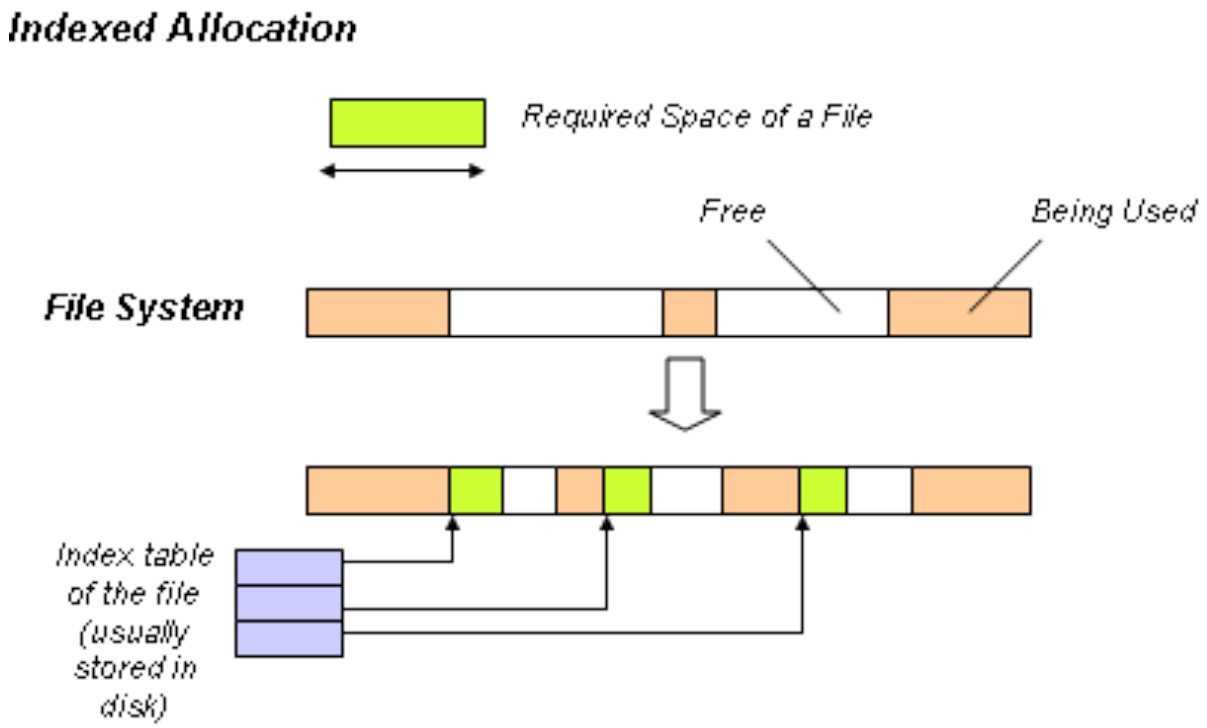
Indexed Allocation

- A file includes a index block that collects all the links to various data blocks
  - Each file has its own index block, a list of disk block addresses.
  - The directory contains the address of the index block
  - No need to traverse the linked disk blocks sequentially
  - An index block always needs to be allocated, even if the file is a small one and only a few pointers are actually needed in the index block

Indexed Allocation in MSDOS

- MS-DOS is an interesting variant of linked and indexed allocation.
  - A file-allocation table (FAT) is used for each file
  - FAT contains an entry for each disk block in the system
    - Indexed by the block number.
  - Similar to linked allocation
    - Entries in the FAT are linked together in a list
    - FAT can be considered an index block

Indexed Allocation



Performance Issues

- Efficient use of available disk space
- Effect on the performance of the given memory allocation scheme and adopted directory structure
- There are two key issues
  - Minimizing the overhead such as pointer size and fixed structures
  - Minimizing disk accesses

Performance Issues

- Caching as a solution
  - Local memory in the disk controllers large enough to store an entire track at a time
  - Once a seek is made, a whole track is read into the cache
  - Eliminating latency time for subsequent sector requests
- Caching can also be maintained in memory, which is known as a disk cache
  - UNIX treats all unused memory as a buffer pool to be used for paging and disk caching

Performance Issues

- RAM disk is a section of memory is mapped to a virtual disk, and all disk operations will actually take place in memory.
  - This needs user intervention to set up.
  - Cache replacement strategies must be carefully designed
- Techniques for sequential access
  - Free-behind: a block is removed in the buffer as soon as the next buffer is requested, so as to free up buffer space
  - Read-ahead: retrieve and cache several subsequent blocks when a block is read