Disk Scheduling

Disk Scheduling

Disk Scheduling

1- Disk System
2- Disk Scheduling

Disk Scheduling

A Disk System is in charge of:

1. *mechanical operation* of the disk and
2. *implementation of the disk I/O commands* given by CPU.

Disk Scheduling

A Disk System components

Disk Drive

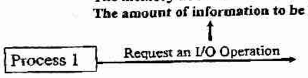
Mechanical parts including the device motor, read/write heads, and associated logic,

Disk Controller

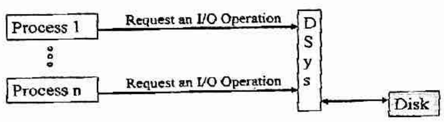
It takes the instructions from the CPU and orders the disk drive to carry out the instruction.

Disk Scheduling

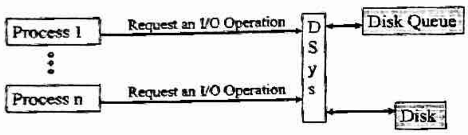
Type of operation (Input or output)
The disk address
The memory address
The amount of information to be transferred.



Disk Scheduling



Disk Scheduling



Disk Scheduling

Disk Scheduling is the process of scheduling the pending requests

Disk Scheduling Algorithms

Six algorithms are provided.

The number of tracks traveled by the read/write head (total head movements) is the basis for evaluating these algorithms.

Disk Scheduling Algorithms

Given: Current position of R/W head is track 53 and tracks to be visited are:

1- FCFS Scheduling

98	---H movement→ |98 – 53|	=	45
183	---H movement→ |183 – 98|	=	85
37	---H movement→ |37 – 183|	=	146
122	---H movement→ |122 – 37|	=	85
14	---H movement→ |14 – 122|	=	108
124	---H movement→ |124 – 14|	=	110
65	---H movement→ |65 – 124|	=	59
67	---H movement→ |67 – 65|	=	2
	Total of head movement:		640 tracks

Disk Scheduling Algorithms

2- Shortest-Seek-Time-First (SSTF)

Head moves to the closest track to the current position.

Example:

Current position of R/W head is track 53 and tracks to be visited are:

98	65	---H movement→ |65 – 53|	=	12
183	67	---H movement→ |67 – 65|	=	2
37	37	---H movement→ |37 – 67|	=	30
122	14	---H movement→ |14 – 37|	=	23
14	98	---H movement→ |98 – 14|	=	84
124	122	---H movement→ |122 – 98|	=	24
65	124	---H movement→ |124 – 122|	=	2
67	183	---H movement→ |183 – 124|	=	59
		Total of head movement:		236 tracks

Disk Scheduling Algorithms

Advantage of (SSTF)

- **Has a better total head movement than FCFS.**

Disadvantages:

- **Starvation**
- **The total head movement is not optimal**

Disk Scheduling Algorithms

3- SCAN (Elevator)

Example:

Current position of R/W head is track 53. R/W head moves toward higher track numbers and tracks to be visited are:

98	65	—H movement→ |65 – 53| =	12
183	67	—H movement→ |67 – 65| =	2
37	98	—H movement→ |98 – 67| =	31
122	122	—H movement→ |122 – 98| =	24
14	124	—H movement→ |124 – 122| =	2
124	183	—H movement→ |183 – 124| =	59
65	200	—H movement→ |200 – 183| =	17
67	37	—H movement→ |37 - 200| =	163
	14	—H movement→ |14 – 37| =	13
		Total of head movement:	323 tracks

Disk Scheduling Algorithms

4- LOOK (Elevator)

Example:

Current position of R/W head is track 53. R/W head moves toward higher track numbers and tracks to be visited are:

98	65	—H movement→ |65 – 53| =	12
183	67	—H movement→ |67 – 65| =	2
37	98	—H movement→ |98 – 67| =	31
122	122	—H movement→ |122 – 98| =	24
14	124	—H movement→ |124 – 122| =	2
124	183	—H movement→ |183 – 124| =	59
65	37	—H movement→ |37 – 183| =	146
67	14	—H movement→ |14 - 37| =	23
		Total of head movement:	299 tracks

Disk Scheduling Algorithms

5- C-SCAN (Circular Scan)

Example:

Current position of R/W head is track 53. R/W head moves toward higher track numbers and tracks to be visited are:

98	65	—H movement→ |65 – 53| =	12
183	67	—H movement→ |67 – 65| =	2
37	98	—H movement→ |98 – 67| =	31
122	122	—H movement→ |122 – 98| =	24
14	124	—H movement→ |124 – 122| =	2
124	183	—H movement→ |183 – 124| =	59
65	200	—H movement→ |200 – 183| =	17
67	0	—H movement→ |0 - 200| =	200
	14	—H movement→ |14 – 0| =	14
	37	—H movement→ |14 – 37| =	23
		Total of head movement:	384 tracks

Disk Scheduling Algorithms

6- C-Look (Circular Look)

Example:

Current position of R/W head is track 53. R/W head moves toward higher track numbers and tracks to be visited are:

98	65	—H movement→ |65 – 53| =	12
183	67	—H movement→ |67 – 65| =	2
37	98	—H movement→ |98 – 67| =	31
122	122	—H movement→ |122 – 98| =	24
14	124	—H movement→ |124 – 122| =	2
124	183	—H movement→ |183 – 124| =	59
65	14	—H movement→ |14 – 183| =	169
67	37	—H movement→ |37 - 14| =	23
		Total of head movement:	322 tracks

Disk Systems

RAID

(Redundant Array of Inexpensive Independent Disks)

To improve

- **Reliability (by Mirroring)**
- **Speed (by Stripping)**

Disk Systems

Mirroring

Duplication of a file on more than one disk.

Stripping

Dividing a "data unit" into a number of parts and store each part on a different disk for the purpose of reading data in parallel.

File Management System

File Management System

File Management System

- What is a file?
- Access Methods
- Directory Structure
- Protection

What is a File

Field: is a named data item.

(the name is type of the field and the data item is the instance of the field.)

Example:

Type

City

Instance

Savannah

What is a File

Record: a named collection of fields. (the name is type of the record and the collection of data items is the instance of the record.)

Example:

Type

Student

Instance

111-11-1111	28	Computer Science	Savannah
Student-ID	Age	Major	Address

What is a File

File: a named collection of one record type's instances.

Example:

SchoolComp

111-11-1111	28	Computer Science	Savannah
222-22-2222	36	Information Tech	Atlanta
333-33-3333	19	Computer Science	Savannah
111-33-4444	22	Engineering	Macon
222-44-1111	24	Computer Science	Macon
Student-ID	Age	Major	Address

What is a File?

Files may carry programs or data (numeric, alphabetic, alphanumeric, and binary)

File may be free or rigidly formatted

What is a File?

File attributes:

- **Name** The only information kept in human readable form.
- **Identifier** Non-human readable name for the file.
- **Type** for those systems supporting different file types. Presented as an extension that is added to the file name.
- **Location** A pointer to the file on a storage device and device itself.
- **Size** The current size of the file (in bytes, words, or blocks)
- **Protection** Access-control information.
- **Time, date, and User Identification** Information for last modification, last use, and creation.

What is a File?

File structure

An internal file structure is needed for OS to handle a file.

OS may be able to support a

minimal number of file structures (makes OS smaller in size) or

large number of file structures (makes OS bulkier).

What is a File

- In Unix each file is a sequence of 8-bit bytes.
- In Macintosh each file has two parts: a resource fork and a data fork.
- In DEC's VMS three different file structures are entertained.

Access Methods

Access method is a method by which the content of a file may be accessed.

Access Methods

Access methods:

- Sequential
- Direct
- Indexed

Access Methods: Sequential

Information in the file is processed one record after another, starting from the first record. A *file pointer, (or position pointer, or current position pointer)* keeps track of the current position in the file from which data (i.e. a record) will be read in. Initially, fp = 0

Access Methods: Sequential

Read operation reads from the "*position pointer*" and then set the pointer to the beginning of the next record.

To read n-th record of the file, then the pointer must go forward or backward to be positioned at the beginning of the n-th record.

Write operation always writes a new record at the end of the file.

Access Methods: Direct

Each record has a unique key.

Each record belongs to one physical record (block).

Records within a block are inserted in order of their keys.

File management system always is able to find block number, say i, for a given record with key =k.
$i = f(k)$

Access Methods: Direct

Read operation: The reading a record with key k, located on the block i, is completed as follow:

- The read-write head is moved to the beginning of block i and then
- The record with key k is found and read sequentially.

Access Methods: Direct

Write operation: The writing a record with key k into the file is completed as follow:

- $i= f(k)$
- The read-write head is moved to the beginning of block i
- The two adjacent records of p and q on block i with key values of k_p and k_q are found such that $k_p < k < k_q$.
- The record with key k is inserted between the two records of p and q.

Access Methods: Direct

Problems with direct access method:

- Maintaining orders of records based on their key values.
- Calculating physical block number related to a given record

Access Methods: Direct

Solutions for Maintaining orders of records based on their key values.

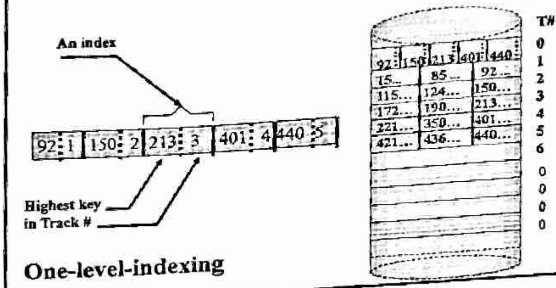
- Leave unused spaces on each block
- Provide a slot for each possible key value.
- Provide a slot for each record and not for each possible key value and then using a mapping mechanism to map a key to a slot. (e.g. using Hashing technique.)
- Use an overflow area. Data in the overflow area may be reached sequentially or by use of a linked list.

Access Methods: Direct

Solution for Calculating physical block number related to a given record

Use of Indexed Access Method.

Access Methods: Indexed

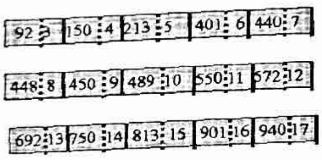


One-level-indexing

Access Methods: Indexed

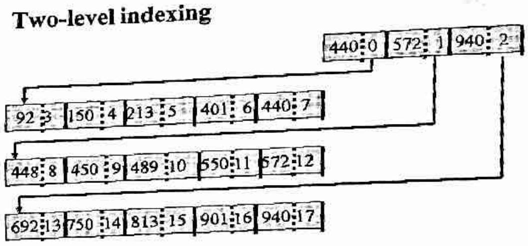
Two-level indexing

Indices go over several tracks:



Access Methods: Indexed

Two-level indexing



Directory Structure

Directory keeps particular information (file attributes) about a file and performs a "name mapping" function.

Directory Structure

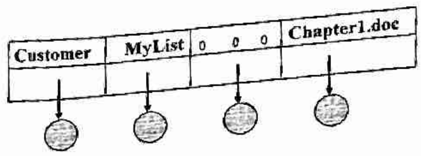
Performs "name mapping" function means directory translates file names into their directory entries.

Directory Structure

Directory is treated as a file. Therefore, it must be able to support insertion, deletion, updating, and retrieving of a new file attributes.

File Directory Structure

Single level directory



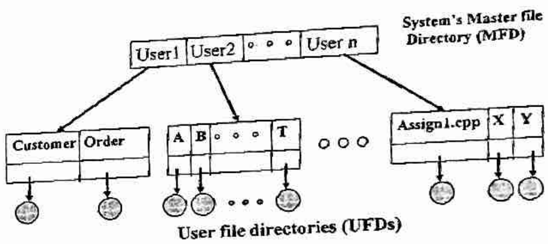
File Directory Structure

Single level directory

Advantage: Simplicity

Disadvantage: Two users can not have the same file name

File Directory Structure



File Directory Structure

Two- level directory

At the time of log-on, MFD is searched for the userid and then the corresponding UFD is loaded.

File Directory Structure

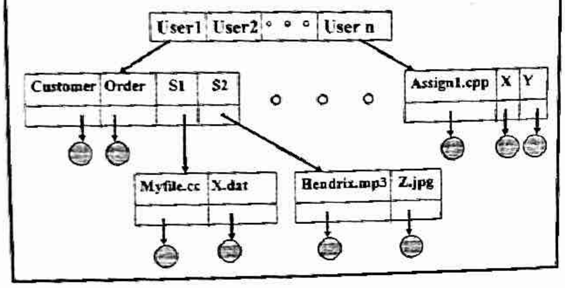
Problems with Two- level directory

- Subdirectories can not be created.
- Eliminates cooperation among the users.
- Makes waste of memory (i.e. System files must be copied in each UFD).

File Directory Structure

Solution to creation of sub-directory problem.

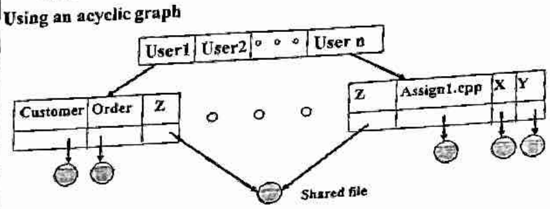
Use of tree-structured directories



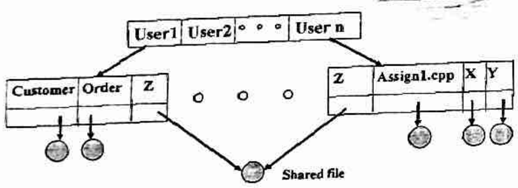
File Directory Structure

Solution to "lack of cooperation among the users" and "waste of memory."

Using an acyclic graph



File Directory Structure

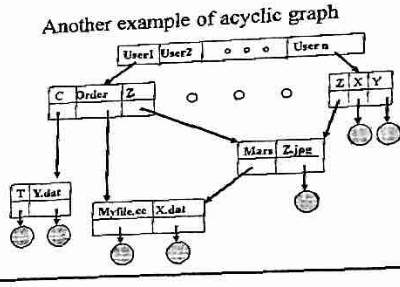


Reference Counter:

A counter that carries the number of users who use the shared file. It is needed for the purpose of deleting and updating the a shared file.

File Directory Structure

Another example of acyclic graph

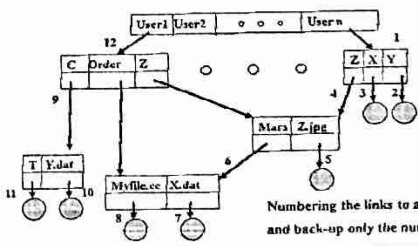


File Directory Structure

Problems with using acyclic graph

A shared file has more than one complete path name, consequently more than one distinct file name. Therefore, redundant copies of the same file is in the back-up storage.

File Directory Structure



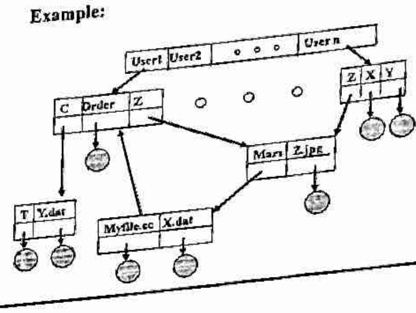
Numbering the links to a shared file once and back-up only the numbered links

File Directory Structure

Due to the need of users sometimes a cycle may appear in the acyclic graph directory structure. In this case, then directory structure becomes a "general graph".

File Directory Structure

Example:



File Directory Structure

Problems with having a cycle:

If a simple algorithm be used to handle the cycle, there is a chance for having:

1- An infinite loop during the search process for a file name.

2- A reference counter is not true representative of the number of users who access the shared file

(Because a cycle may refer to itself and, therefore, counter is incremented whereas the number of users using the shared file remains the same.)

∴ deletion of shared file is difficult.