0301304 FUNDAMENTAL OF OPERATIONG SYSTEM

UNIT	MODULES	WEIGHTAGE
1	INTRODUCATION TO OPERATING SYSTEM	20 %
2	PROCESS MANAGEMENT	20 %
3	PROCESS COMMUNICATION AND SYNCHRONIZATION	20 %
4	MEMORY MANAGEMENT	20 %
5	FILE MANAGEMENT , DISK MANAGEMENT , SECURITY AND PROTECTION	20 %

UNIT – 5 File Management, Disk Management, Security and Protection

- File System
  - Introduction
  - Files and File System
  - File Structure
  - File Naming and File Type
  - File Access
- Disk Management
  - Introduction
  - Disk Scheduling Algorithm
    - FCFS
    - SSTF
    - SCAN
    - LOOK and C-LOOK

UNIT – 5

File Systems

UNIT – 5 File Systems

- The file **system provides a convenient mechanism to store and retrive the data** and programs from this medium.
- **File are used as a collection of related information**, the meaning of which is defined by its creator.
- These **files are mapped to the disks or other storage media by the Os.**

UNIT – 5 File Systems

- Files, a c**onvenient environment is created that allows one to write, read, save and retrieve** the program and data on any type of store media.
- A Files, thus is a collection of related information that is **mapped on to a secondary storage.**
- The information stored in a file is in bits, bytes, lines or records.
- The **user only sees the logica view** of the file.
- The **sytem views all the work required to map the logical file to the secondary storage.**
- The **benefit of storeing** files on the secondary storage is the facility provided by the file system:
  - **To create, store and retrive the information in the form of files.**

UNIT – 5 File Systems

- The following are the primary constituents of a file system:
  - File Management
    - It manages how the files are stored, referenced, shared and secured.
  - File Allocation
    - It provides the methods to allocate files on the disk space.
  - File Access Methods
    - It provides the methods to access stored files.

UNIT – 5 File Systems – File Structure

- The basic element of data is **Field**, is a single value item.
- When multiple fields are combined to form a meaningful collection, it is known as a **record.**
- When such similar records are collected, it is known as a **file.**
- Os must support a required structure for a certain type of file.

UNIT – 5 File Systems – File Structure

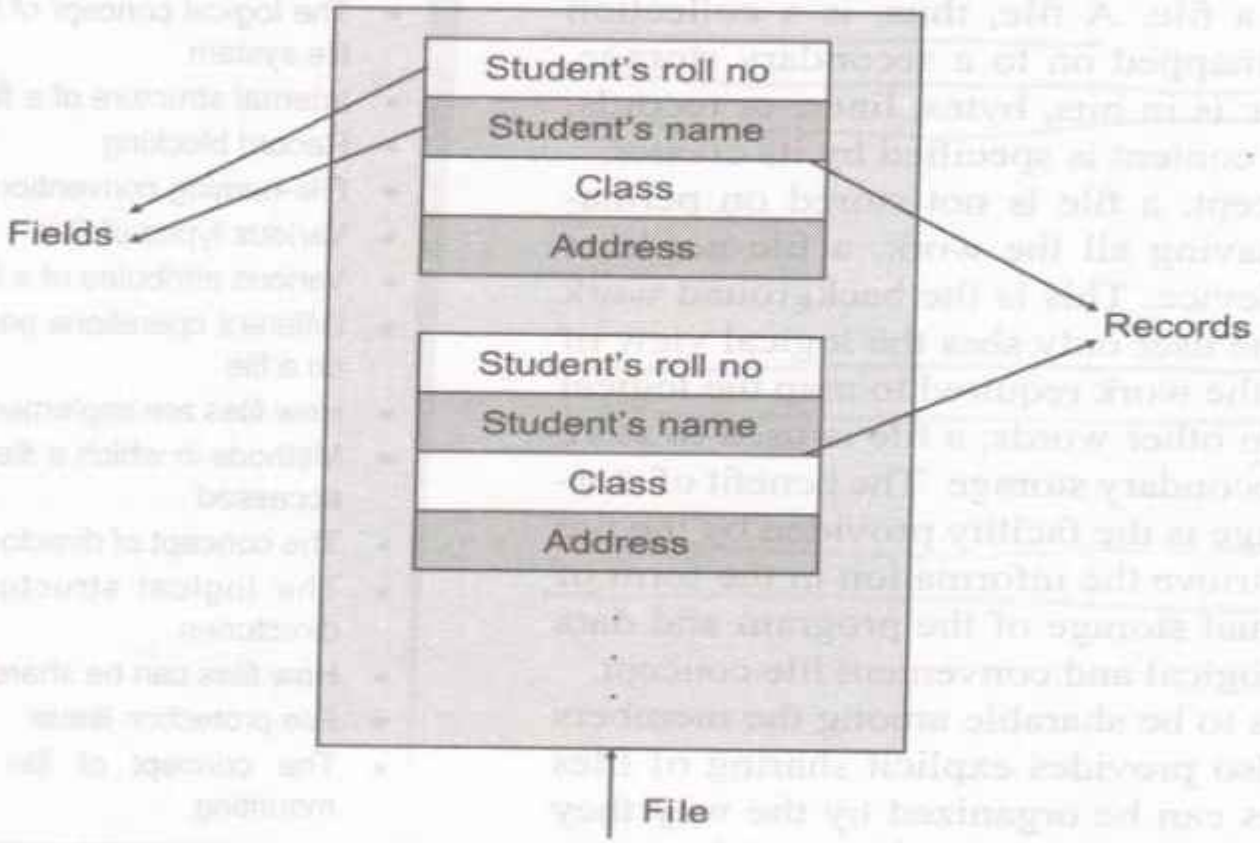
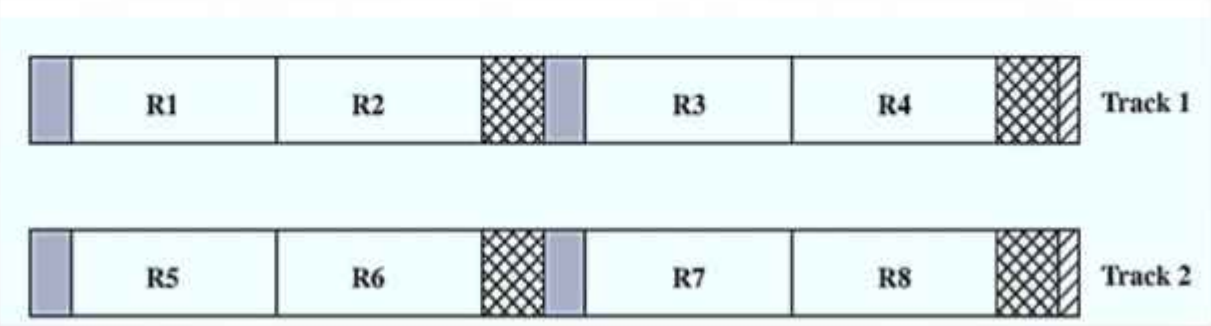


Fig. 12.1 File structure

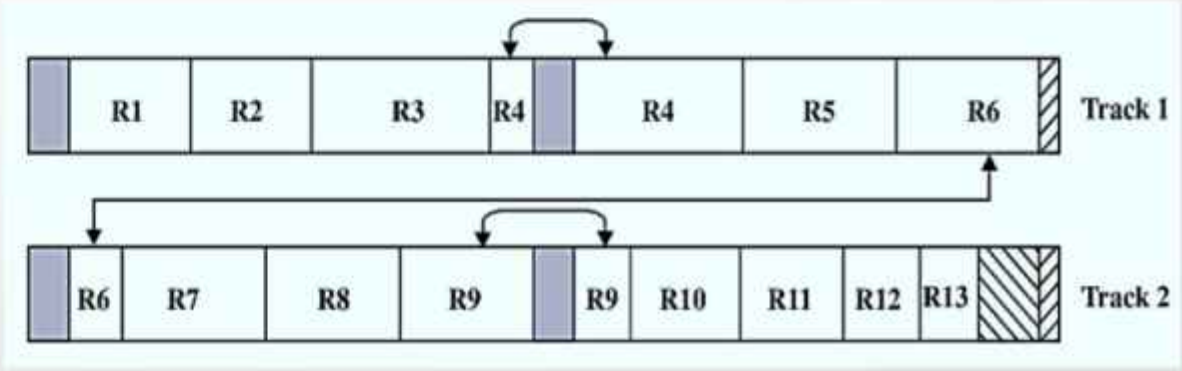
UNIT – 5 File Systems –Internal Structure & Record Blocking

- **Record Blockings**
  - A group of 512 bytes is packed into one block of disk. This is called block, all basic I/O functions are performed in terms of blocks, this is known as **record blockings.** Blocking conserves storage space on a volume by reducing the number of interblock gaps in the data set.
  - **Fixed Blocking**
  - **Variable – length Spanned Blocking**
  - **Varible – length Unspanned Blocking**

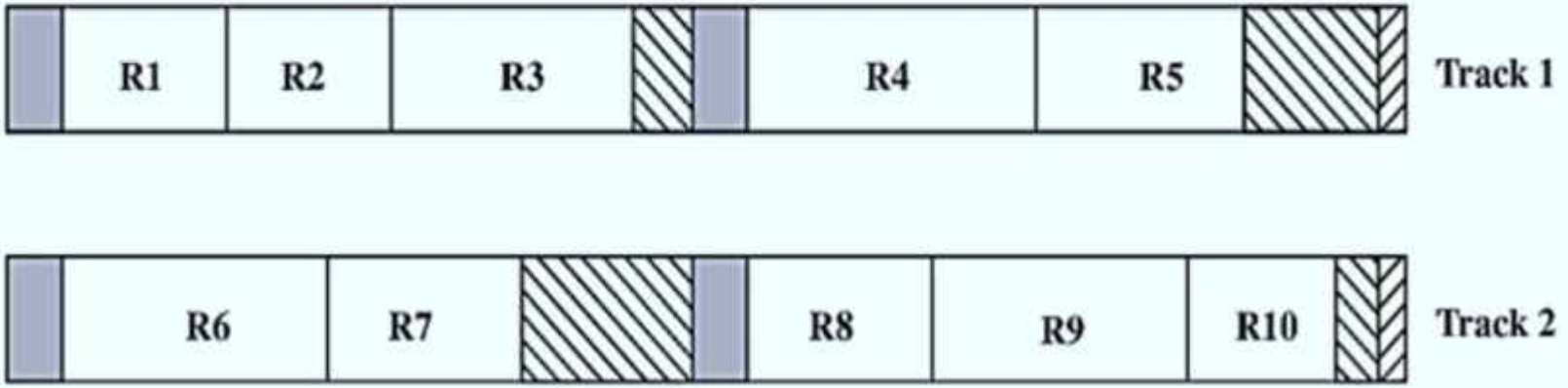
- **Fixed Blocking**
  - In this method, record lengths are fixed.
  - The prescribed number of records stored in a block. Internal fragmentation is stored in a block.
  - Fixed blocking is common for sequential files with fixed-length records.



- **Variable – length Spanned Blocking**
  - In this method, record sizes aren't same, Variable-length records packing into blocks with no unused space. So, some records may divide into two blocks.
  - In this type of situation, a pointer passes from one block to another block. So, the Pointers used to span blocks unused space.
  - It has efficient storage and doesn't limit record size, but it is more complicated to implement. So, the files are more difficult to update.



- **Variable – length Unspanned Blocking**
  - Variable lengths are used, but spanning is not considered in case of small size blocks.
  - A small size block is left unused, causing wastage of memory space and records are allocated to a bigger block.



UNIT – 5 File Systems – File Naming & File Types

- All the file name has two parts, seperated bya period (.).
- First part is the **name of the file**, define by the user.
- Second part is known as **extension.**

UNIT – 5 File Systems – File Naming & File Types

- An OS must recoganize the type of the file, because the operations performed on its depend on its type.
- **Source code file -** .c
- **Object file -** .obj or .o
- **Executable file -** .exe or .com or .bin
- **Text file -** .txt or .doc
- **Batch file -** .bat
- **Archive file -** .zip or .rar
- **Multimedia file -** .jpg or .mp3 or .mov

UNIT – 5 File Systems – File Naming & File Types

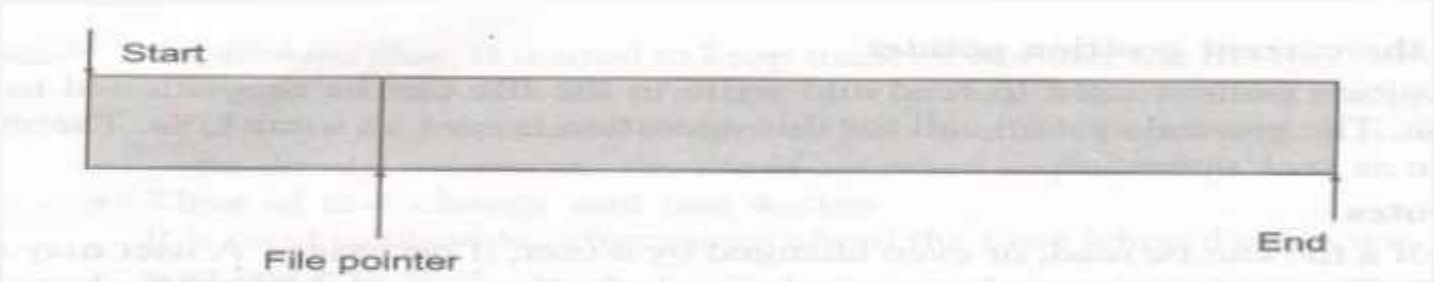
- In general files are the following types:
  - **Regular**
    - Contain the user information
  - **Directory**
    - Is a file type used to orgnize the list of files in a group.
  - **Special**
    - A special file contains no data but provides a mechanism that maps physical devices to file name.

UNIT – 5 File Systems – File Access

- The following are some file access methods:
  - Sequential File Access
  - Direct File Access
  - Indexed Sequential File Access

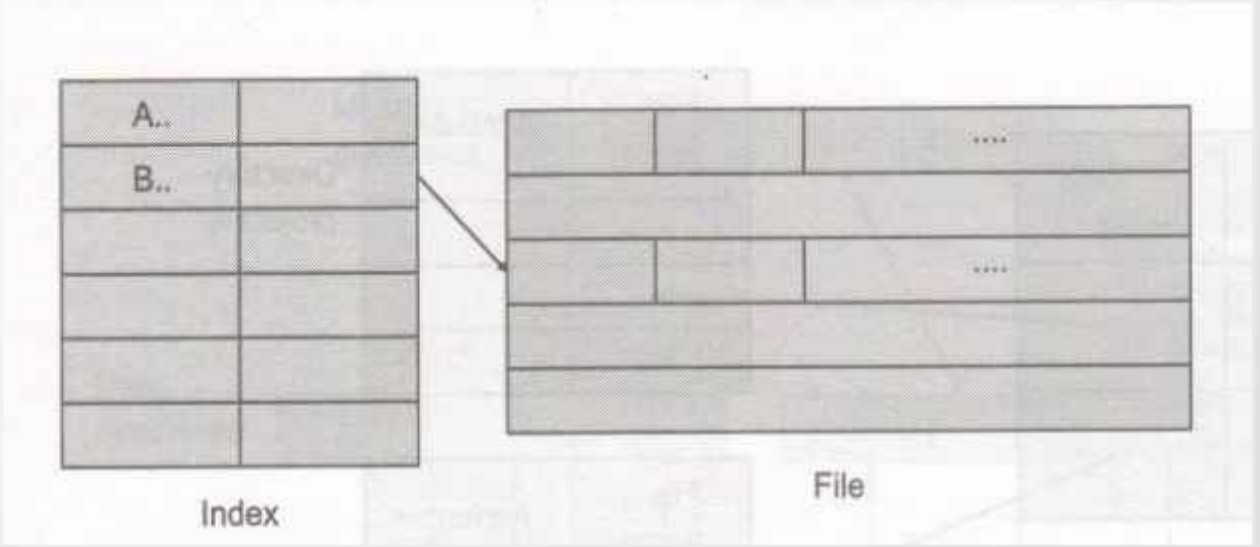
UNIT – 5 File Systems – File Access

- **Sequential File Access -** It is the simplest access method. Information in the file is processed in order, one record after the other. This mode of access is by far the most common; for example, editor and compiler usually access the file in this fashion.



- **Direct Access -** Another method is direct access method also known as relative access method. A filed-length logical record that allows the program to read and write record rapidly in no particular order.

- **Index sequential method –** It is the other method of accessing a file that is built on the top of the sequential access method. These methods construct an index for the file. The index, like an index in the back of a book, contains the pointer to the various blocks. To find a record in the file, we first search the index, and then by the help of pointer we access the file directly.



UNIT – 5

Disk management

UNIT – 5 Disk Management – Disk Scheduling

- There are several reason disk scheduling is important for I/O operations.
  - Many **process may send I/O** request and the processor can service one I/O at a time.
  - Seek time management.
  - When Disk is near to full.
  - Random requests from the user may also need disk scheduling.

UNIT – 5 Disk Management – Disk Scheduling Algorithms

- FCFS (First Come First Serve)
- SSTF (Shortest Seek Time First)
- SCAN (Elevator Algoritm)
- LOOK and C-LOOK

UNIT – 5 Disk Management – Disk Scheduling Algorithms - FCFS

- Consider a disk queue with I/O request

  54, 97, 73, 128, 15, 44, 110, 34, 45

- The disk head is assumed to be at Cylinder 23

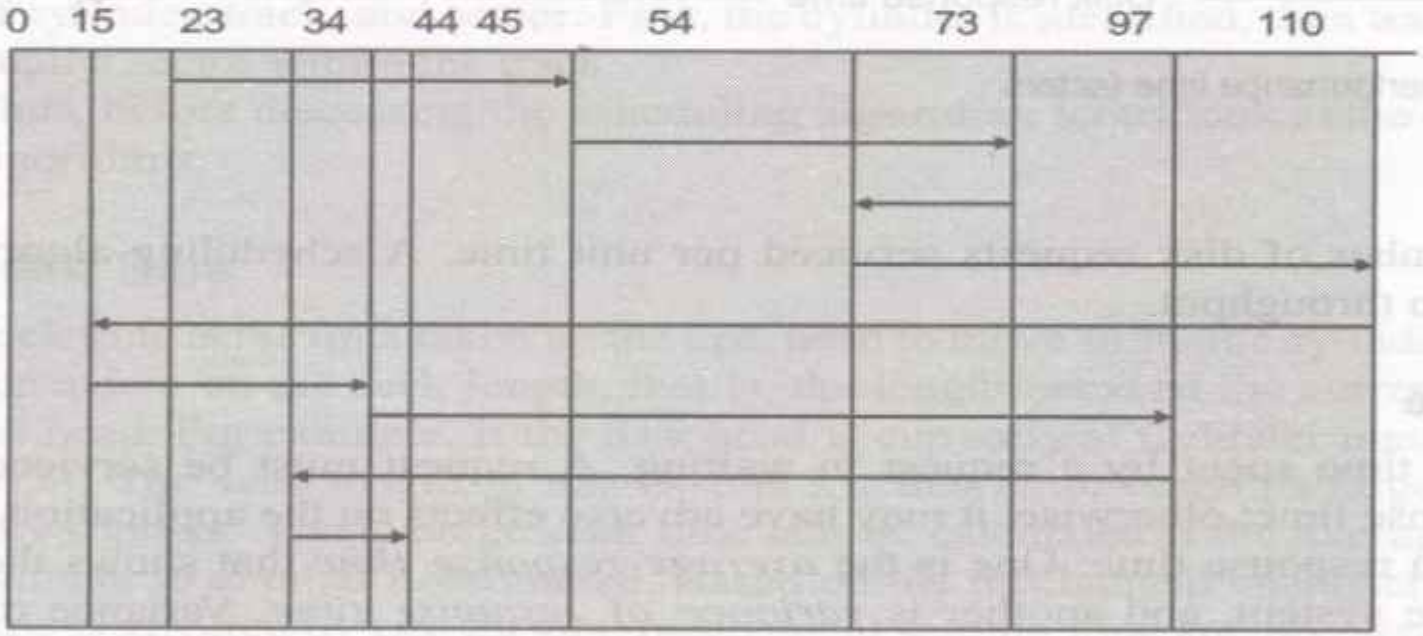


Fig. 15.2 FCFS disk scheduling for Example 15.1

Table 15.1 FCFS head movement for Example 15.1

UNIT – 5 Disk Management – Disk Scheduling Algorithms - FCFS

I/O request for cylinder	Head movement	Total head movement for the request
54	23–54	31
97	54–97	43
73	97–73	24
128	73–128	55
15	128–15	113
44	12–44	32
110	44–110	66
34	110–34	76
45	34–45	11
Total head movement = 451		

15.4.2 SSTF

UNIT – 5 Disk Management – Disk Scheduling Algorithms - FCFS

- Consider a disk queue with I/O request

  54, 97, 73, 128, 15, 44, 110, 34, 45

- The disk head is assumed to be at Cylinder 23

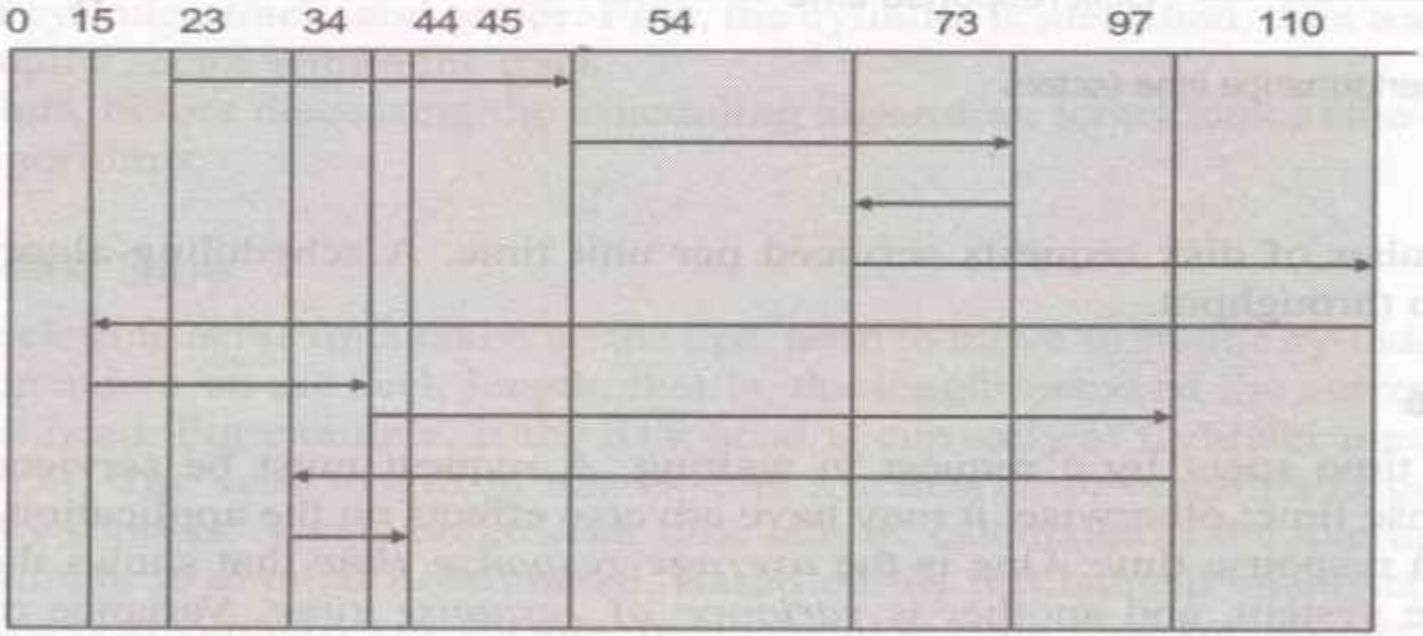


Fig. 15.2 FCFS disk scheduling for Example 15.1

Table 15.1 FCFS head movement for Example 15.1

UNIT – 5 Disk Management – Disk Scheduling Algorithms - SSTF

- Consider a disk queue with I/O request

  54, 97, 73, 128, 15, 44, 110, 34, 45

- The disk head is assumed to be at Cylinder 23

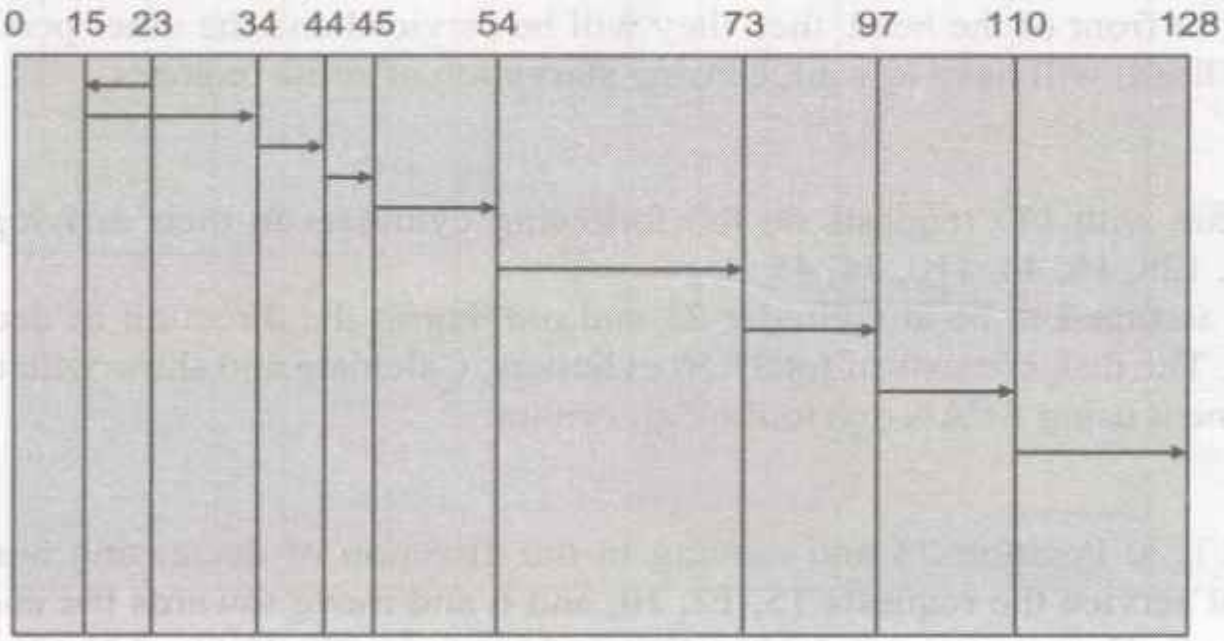


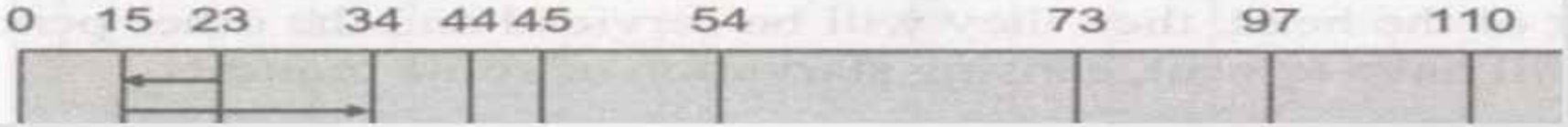
Fig. 15.3 SSTF disk scheduling for Example 15.2

UNIT – 5 Disk Management – Disk Scheduling Algorithms - SSTF

distance that a head needs to move. From the current positi
distance is the shortest is serviced, as shown in Table 15.2

Table 15.2 SSTF head movement for Example 15.2

Head movement	Total head movement for the request
23–15	8
15–34	19
34–44	10
44–45	1
45–54	9
54–73	19
73–97	24
97–110	13
110–128	18
Total head movement = 121	



UNIT – 5 Disk Management – Disk Scheduling Algorithms - SCAN

- Consider a disk queue with I/O request

  6, 10, 12, 54, 97, 73, 128, 15, 44, 110, 34, 45
- The disk head is assumed to be at Cylinder 23 and moving in the direction of decreasing number of cylinder.
- **The total head movement in this 23 + 150 = 173**

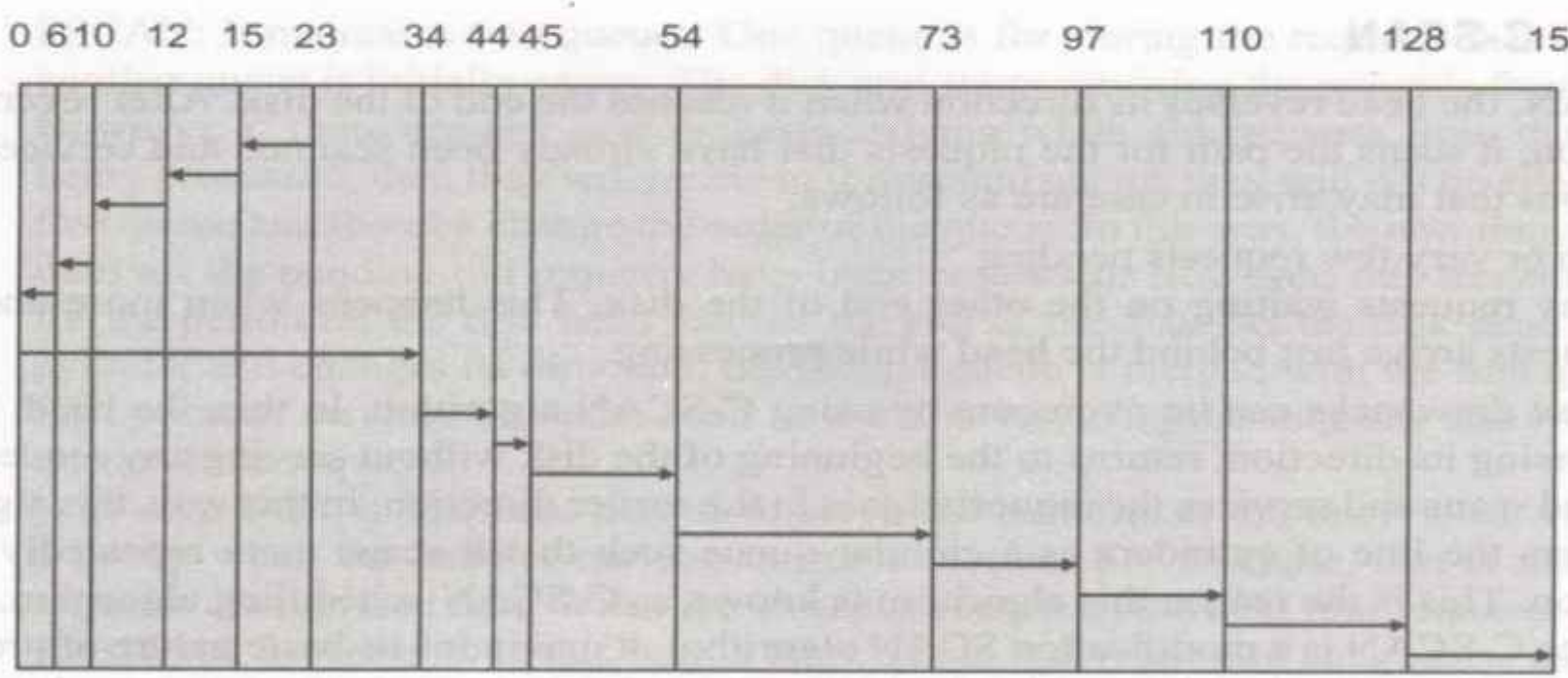


Fig. 15.4 SCAN disk scheduling for Example 15.4

UNIT – 5 Disk Management – Disk Scheduling Algorithms – SCAN

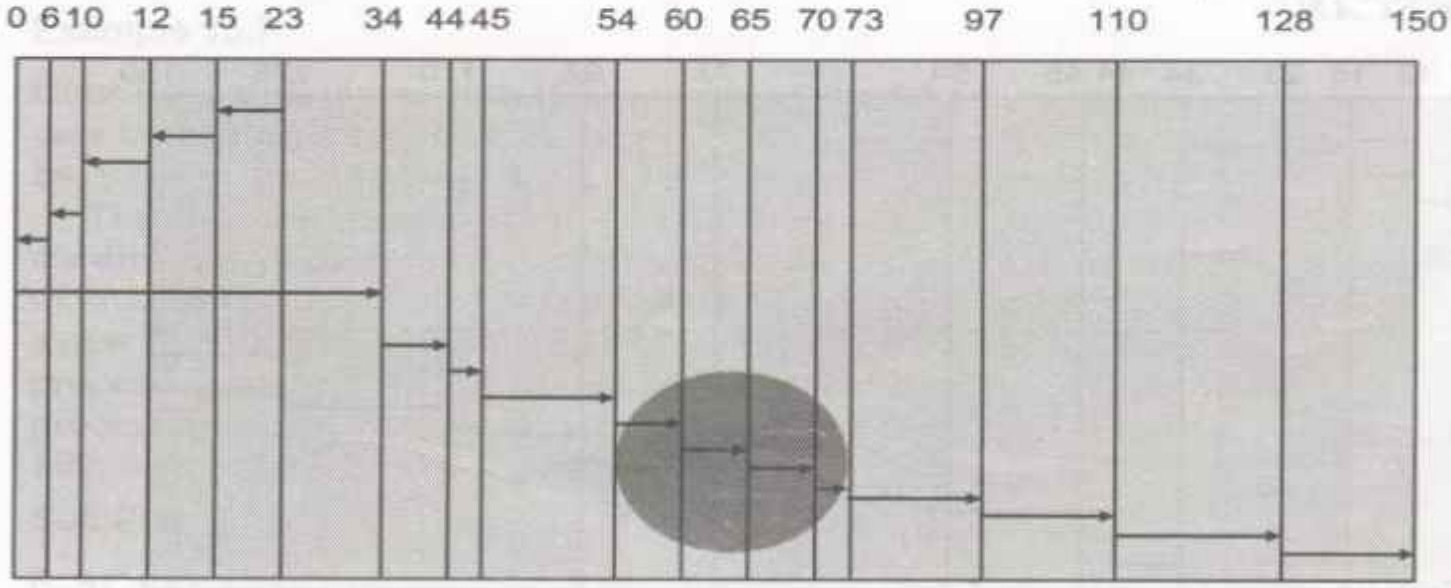
Cylinders 128 to 150 are unnecessary as there are no requests in these paths.

Example 15.5

In Example 15.4, suppose some new requests arrive at Cylinders 60, 65, and 70 while the disk head is processing Cylinder 54. What will happen to these new requests according to the SCAN-scheduling algorithm?

Solution

Since the new requests arrived are in the preferred direction of disk-head movement, all these requests will be serviced just after the processing at Cylinder 54 as shown in the highlighted portion in Fig. 15.5. However, due to arrival of these new requests, the waiting time of pending requests in the queue increases.



UNIT – 5 Disk Management – Disk Scheduling Algorithms – Look & C-Look

- Consider a disk queue with I/O request

  6, 10, 12, 54, 97, 73, 128, 15, 44, 110, 34, 45
- The disk head is assumed to be at Cylinder 23 moving the direction of decreasing cylinder number.
- **The total head movement in this 17 + 122 = 139**

The total head movement in this algorithm is 139.

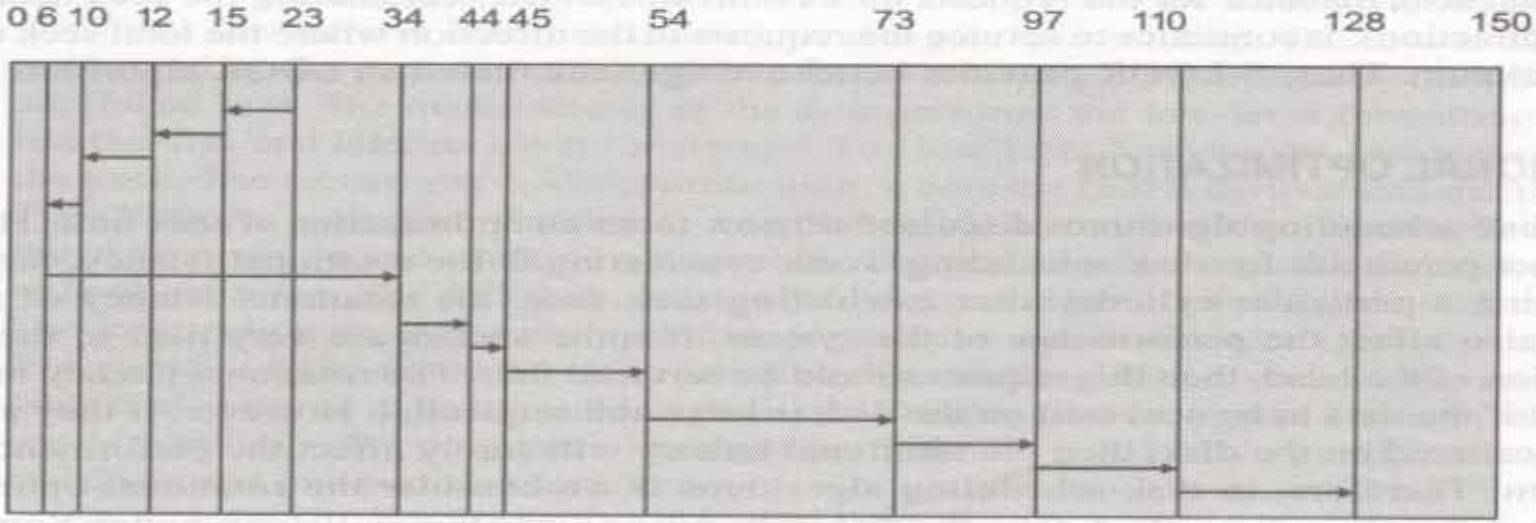
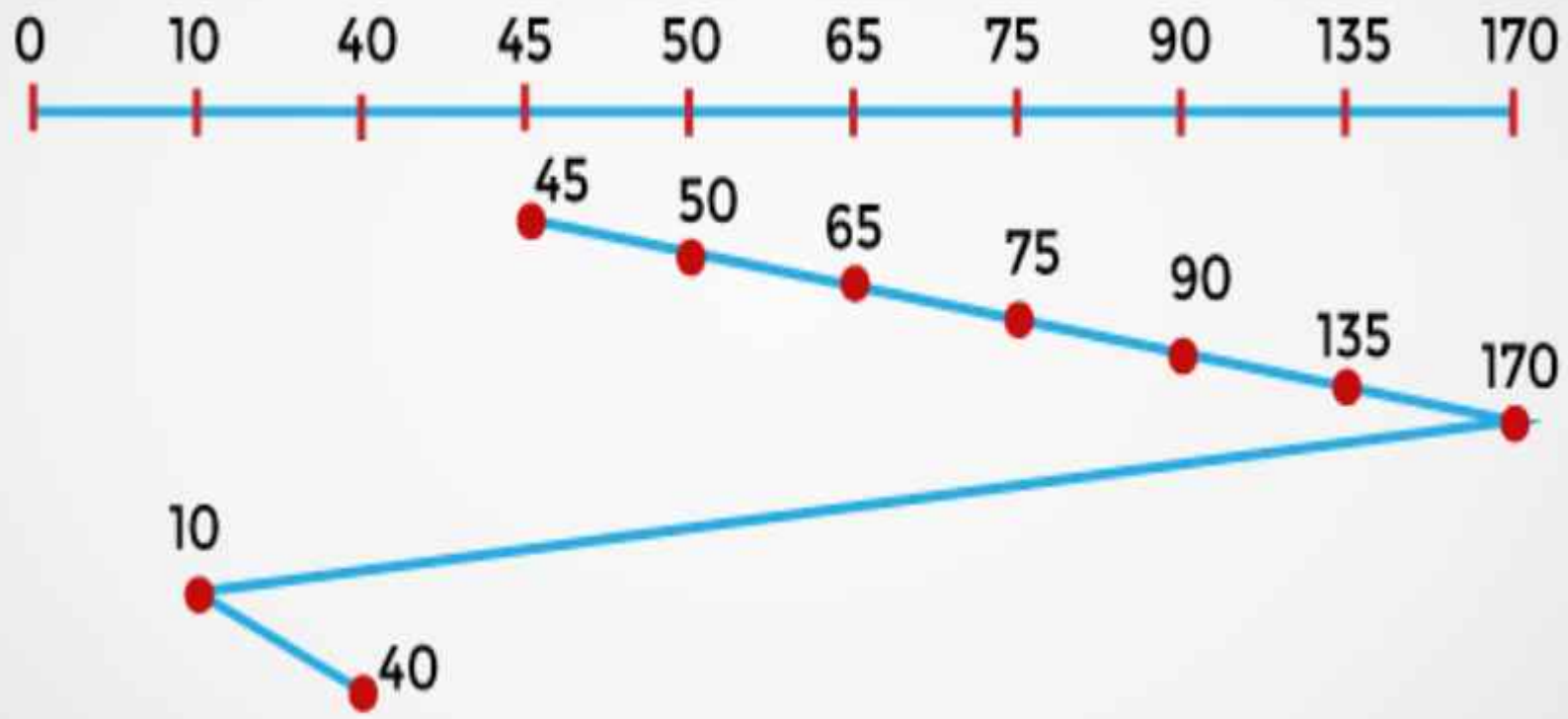


Fig. 15.13 LOOK disk scheduling for Example 15.9

C-Look Disk Scheduling Algorithm



C-Look Disk Scheduling Algorithm