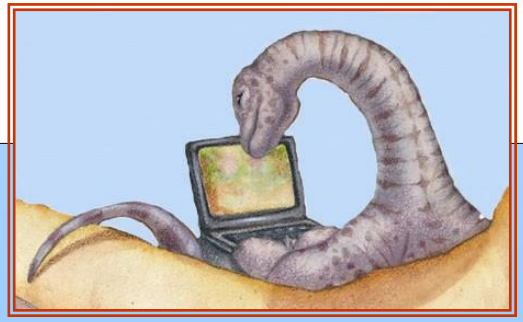
File System Implementation



Allocation Methods

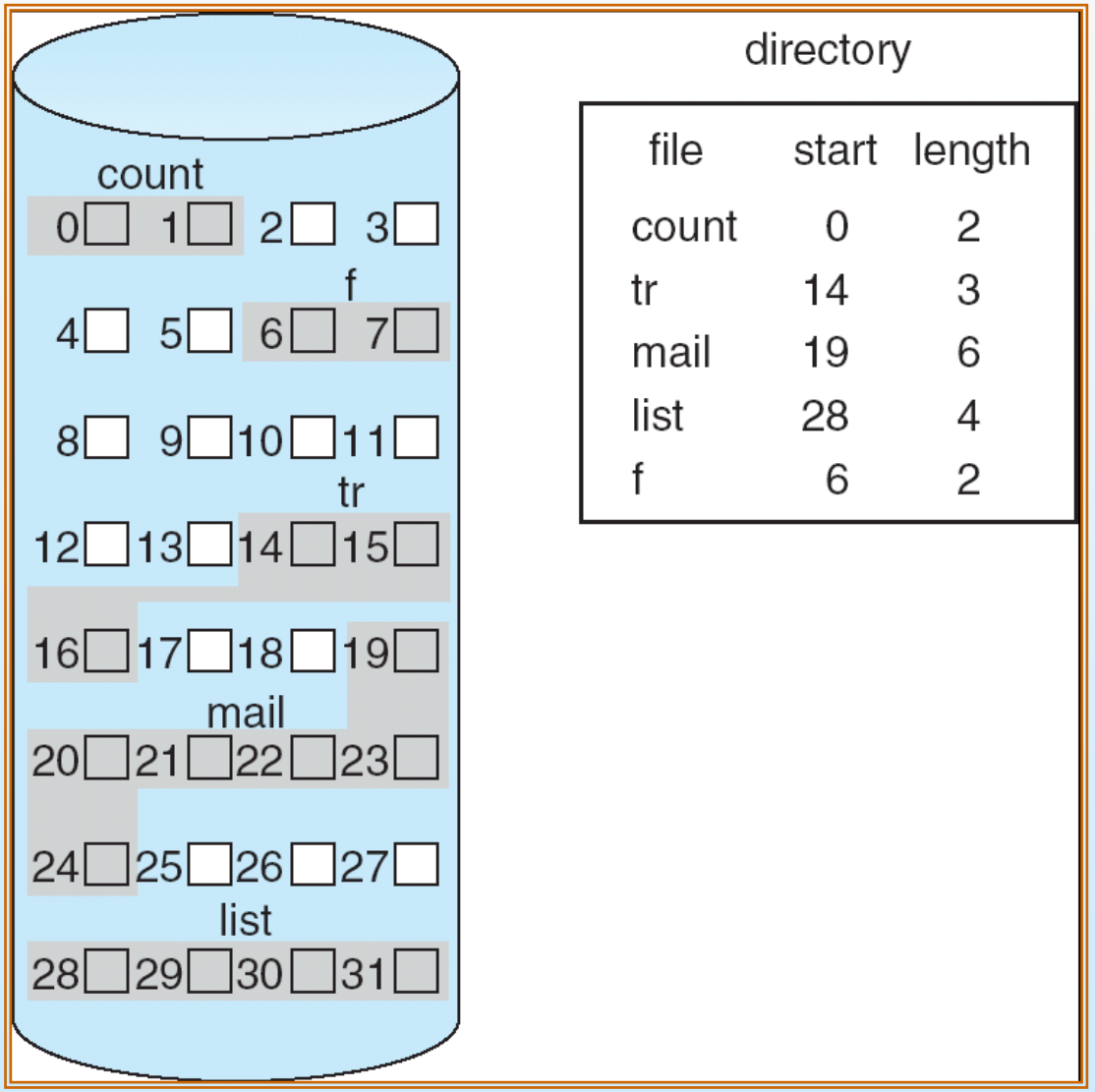
- An allocation method refers to how disk blocks are allocated for files:
- **Contiguous allocation**
- **Linked allocation**
- **Indexed allocation**

Contiguous Allocation

- Each file occupies a set of contiguous blocks on the disk
- Simple – only starting location (block #) and length (number of blocks) are required
- Sequential and Random access

If a file is n block long, and starts at location b, then it occipies b,b+1,b+2,… b+n-1

Contiguous Allocation of Disk Space



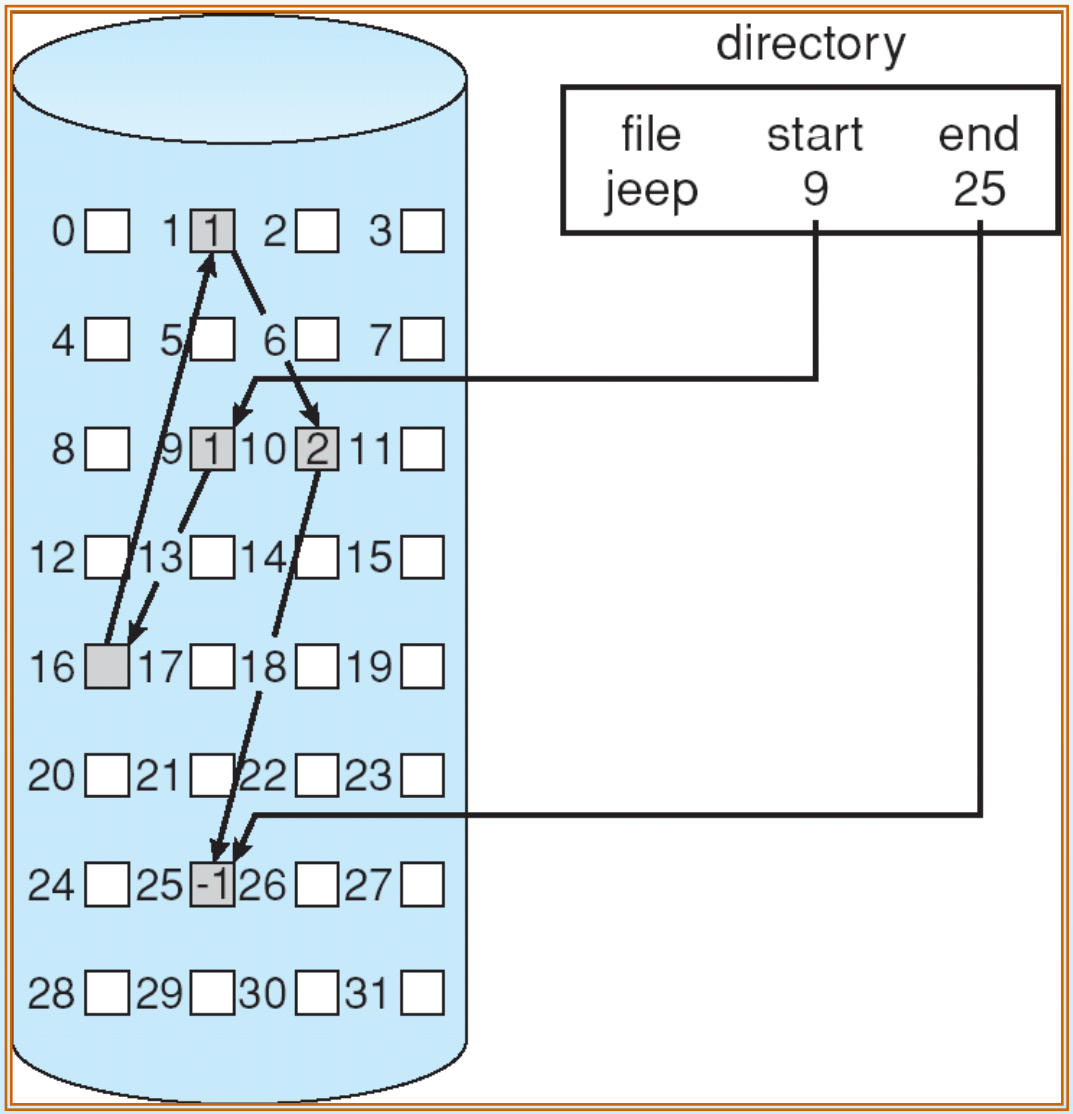
Problems

- Difficulty is finding space for a new file
- Determining how much space is needed for a file. When a file is created, the total amount of space it will need must be found and allocated.
- Wasteful of space
- Files cannot grow

Linked Allocation (Cont.)

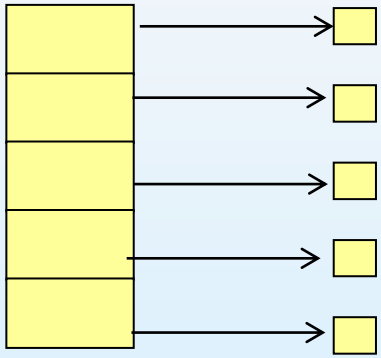
- Simple – need only starting address
- Free-space management system – no waste of space
- No random access

Linked Allocation



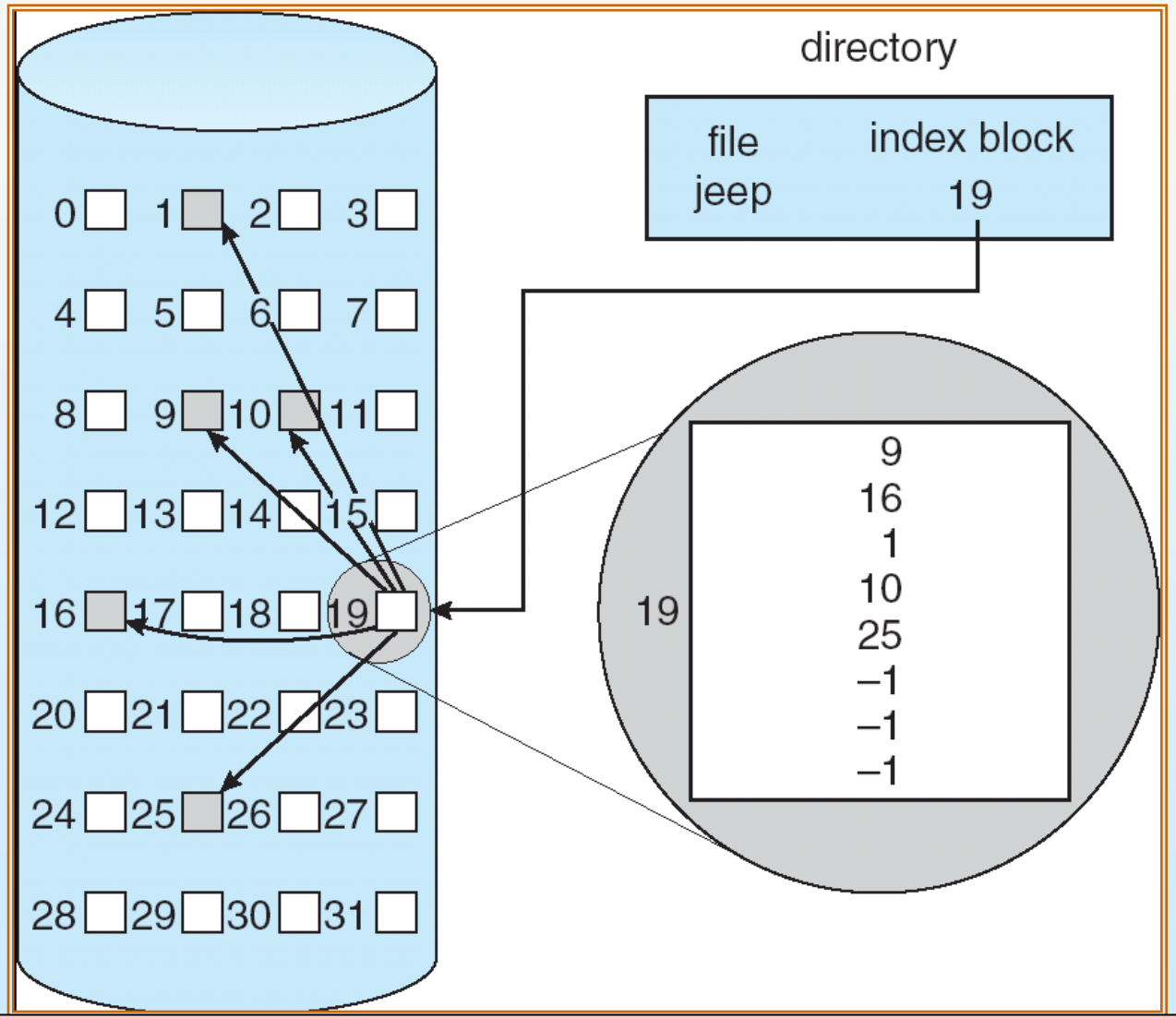
Indexed Allocation

- Linked allocation solves external fragmentation and size-declaration problem of contiguous allocation.
- Brings all pointers together into the *index block.*
- Logical view.



index table

Example of Indexed Allocation



Disadvantages

A bad index block could cause the lost of entire file.
Size of a file depends upon the number of pointers, a index block can hold.
Having an index block for a small file is totally wastage.
More pointer overhead

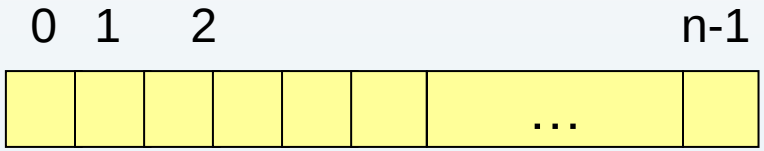
Free-Space Management

A file system is responsible to allocate the free blocks to the file therefore it has to keep track of all the free blocks present in the disk.

Free-Space Management

000011100000110.

- Bit vector (*n* blocks)



$$bit[i] = \begin{cases} 1 \Rightarrow block[i] \text{ free} \\ 0 \Rightarrow block[i] \text{ occupied} \end{cases}$$

Block number calculation

(number of bits per word) *
(number of 0-value words) +
offset of first 1 bit

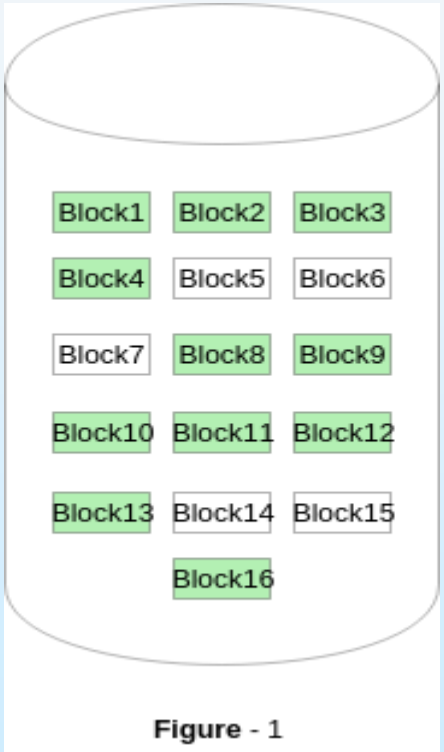
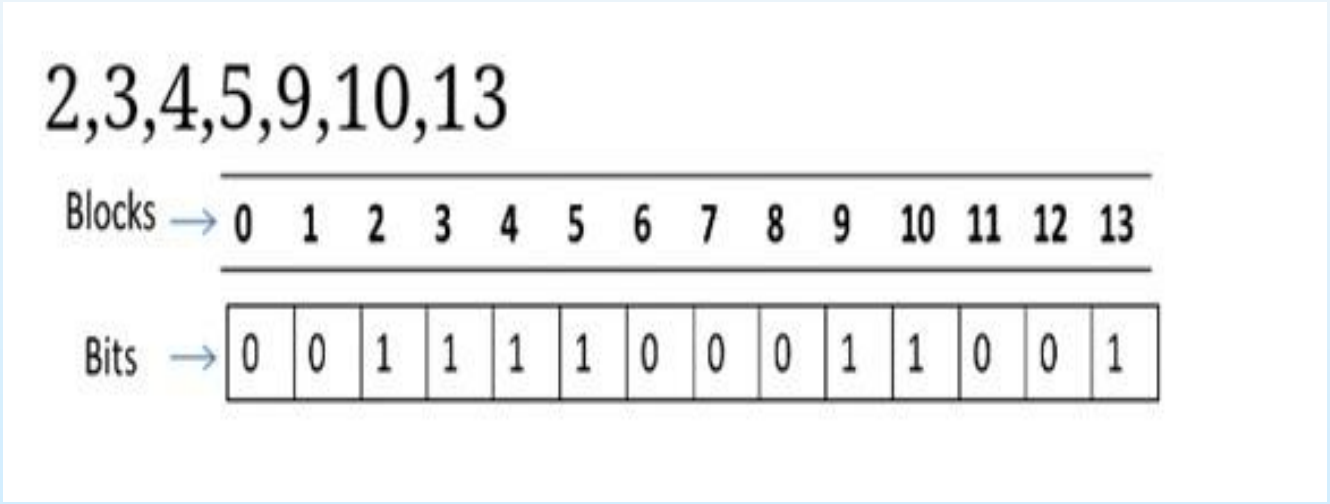


Figure - 1

- Assume the following are free. Rest are allocated:
- 2, 3,4,5,9,10,13



Free-Space Management (Cont.)

- Bit map requires extra space
  - Example:

    block size = 2^{12} bytes

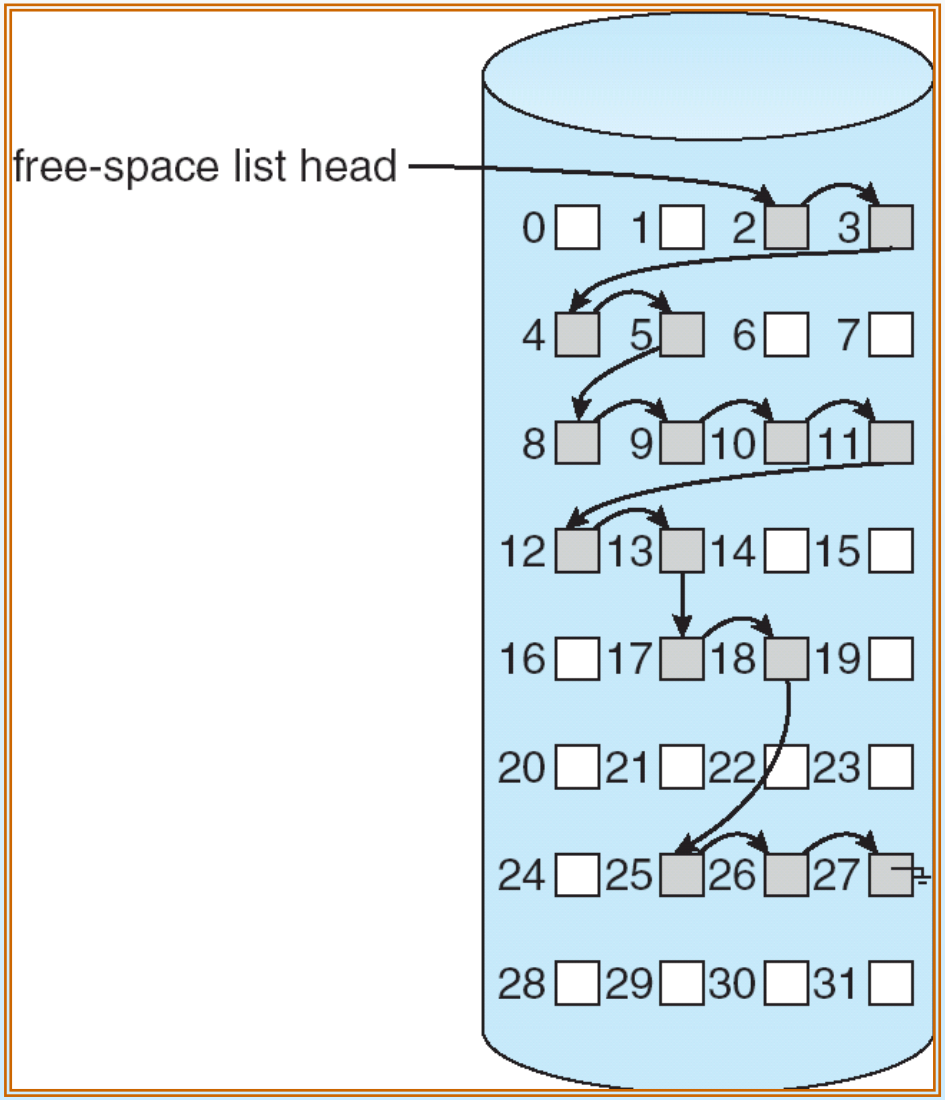
    disk size = 2^{30} bytes (1 gigabyte)

    $n = 2^{30}/2^{12} = 2^{18}$ bits (or 32K bytes)
- Easy to get contiguous files

Linked Free Space List on Disk

In this approach, the free disk blocks are linked together i.e. a free block contains a pointer to the next free block. The block number of the very first disk block is stored at a separate location on disk and is also cached in memory.

Linked Free Space List on Disk



Directory Implementation

- Linear list of file names with pointer to the data blocks
  - simple to program
  - time-consuming to execute
- Hash Table – linear list with hash data structure
  - decreases directory search time
  - **collisions** – situations where two file names hash to the same location
  - fixed size

Grouping

- A free block contains *n pointers to free blocks*
- The last block among *n pointers contains another free blocks.*

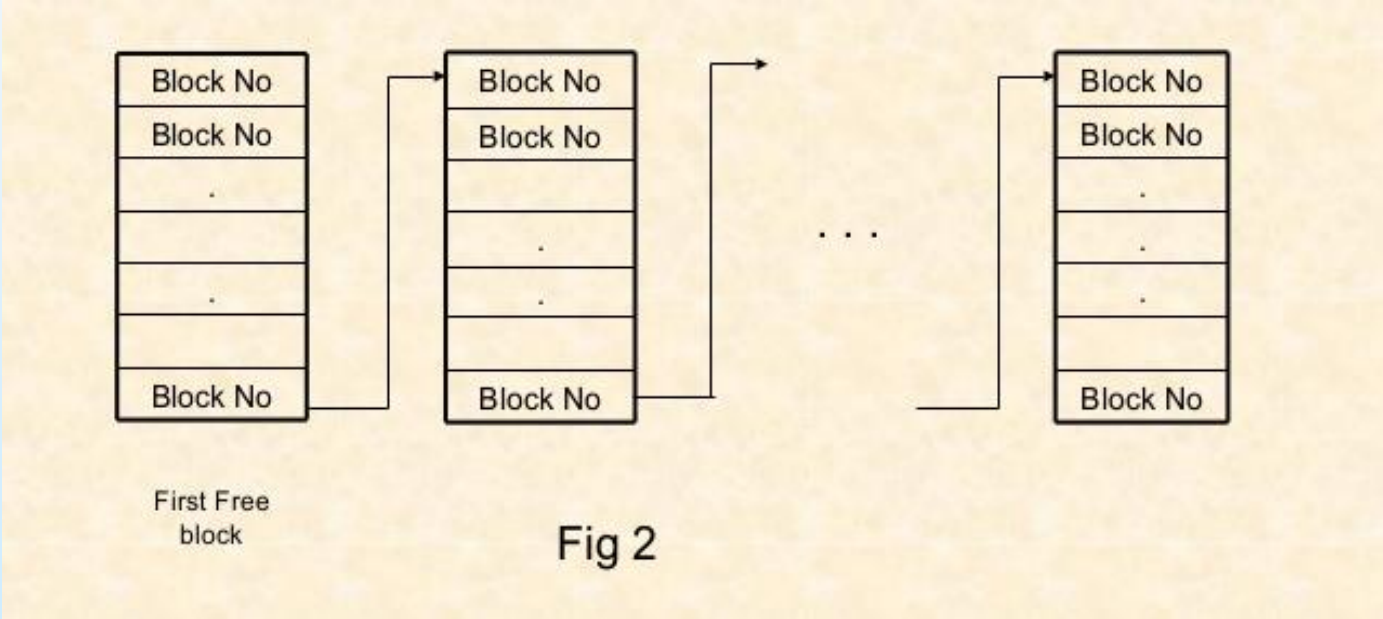


Fig 2

Example :

- Suppose we have a disk with some free blocks and some occupied blocks.
- The free block numbers are 3,4,5,6,9,10,11,12, and 14. And occupied block numbers are 1,2,7,8,15, and 16 i.e., they are allocated to some files.

Block 3 -> 4, 5, 6
Block 6 -> 9, 10, 11
Block 11 -> 12, 13, 14

Counting

- This approach stores the address of the first free disk block and a number n of free contiguous disk blocks that follow the first block.
- Every entry in the list would contain:
- Address of first free disk block
- A number n
- For example, *in Figure-1*, the first entry of the free space list would be: ([Address of Block 5], 2), because 2 contiguous free blocks follow block 5.

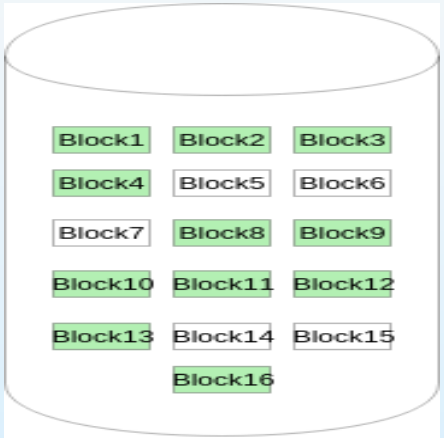
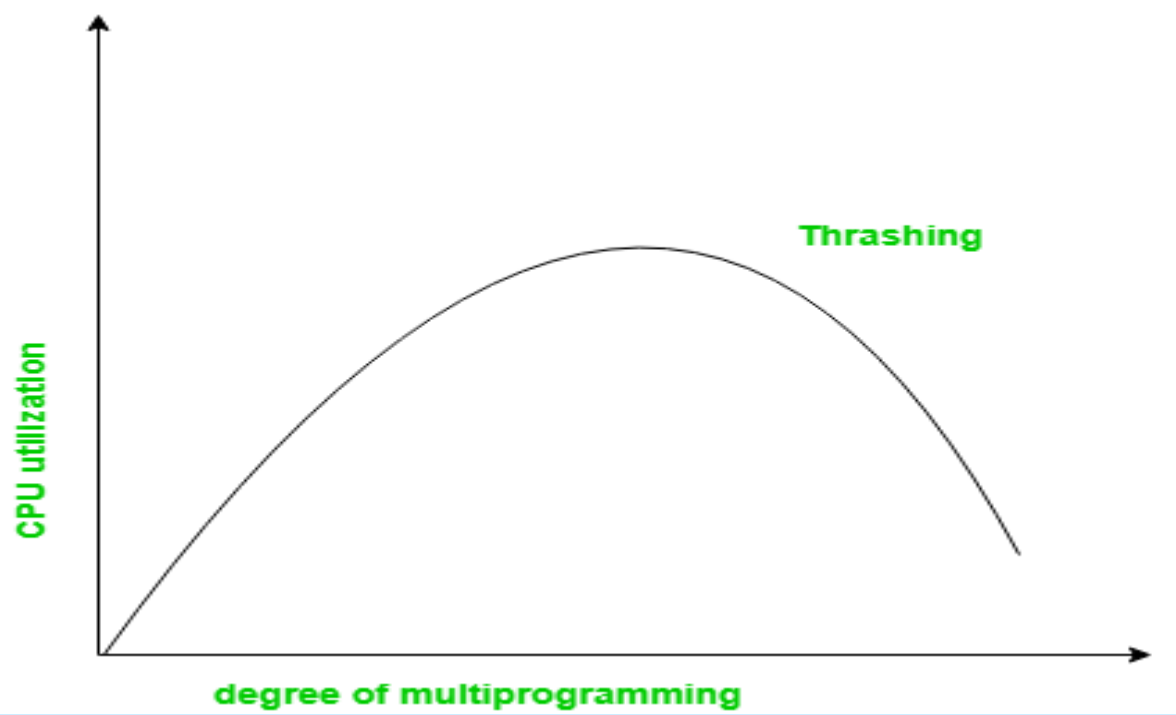


Figure - 1

Thrashing

- **Thrashing** is a condition or a situation when the system is spending a major portion of its time in servicing the page faults, but the actual processing done is very negligible.



- The effect of Thrashing is

1) CPU becomes idle.
2) Decreasing the utilization increases the degree of multiprogramming and hence bringing more processes at a time which in fact increases the thrashing exponentially.

Techniques to Handle Thrashing

- **Working Set Model**
- This model is based on locality.
- What locality is saying, the page used recently can be used again and also the pages which are nearby this page will also be used.
- Working set means set of pages in the most recent D time.

Working Set Model

- Page reference 2 6 1 5 7 7 7 5 1 6 2 3 4 1 2 3 4 4 4 Maximum working set window: 5

Requested page	Time	Working set windows	Page fault (Hit/Miss)	Current windows
2	t_1	{2}	Page fault (Miss)	1
6	t_2	{2, 6}	Page fault (Miss)	2
1	t_3	{2, 6, 1}	Page fault (Miss)	3
5	t_4	{2, 6, 1, 5}	Page fault (Miss)	4
7	t_5	{2, 6, 1, 5, 7}	Page fault (Miss)	5
7	t_6	{6, 1, 5, 7}	Page fault (Hit)	4
7	t_7	{1, 5, 7}	Page fault (Hit)	3
5	t_8	{5, 7}	Page fault (Hit)	2
1	t_9	{1, 5, 7}	Page fault (Miss)	3
6	t_{10}	{1, 5, 7, 6}	Page fault (Miss)	4
2	t_{11}	{1, 5, 7, 6, 2}	Page fault (Miss)	5
3	t_{12}	{1, 5, 6, 2, 3}	Page fault (Miss)	5

4	t_{13}	{1, 6, 2, 3, 4}	Page fault (Miss)	5
1	t_{14}	{1, 6, 2, 3, 4}	Page fault (Miss)	5
2	t_{15}	{1, 2, 3, 4}	Page fault (Hit)	4
3	t_{16}	{1, 2, 3, 4}	Page fault (Hit)	4
4	t_{17}	{1, 2, 3, 4}	Page fault (Hit)	4
4	t_{18}	{1, 2, 3, 4}	Page fault (Hit)	4
4	t_{19}	{2, 3, 4}	Page fault (Hit)	3

Average Frame requirement=71/19=3.73

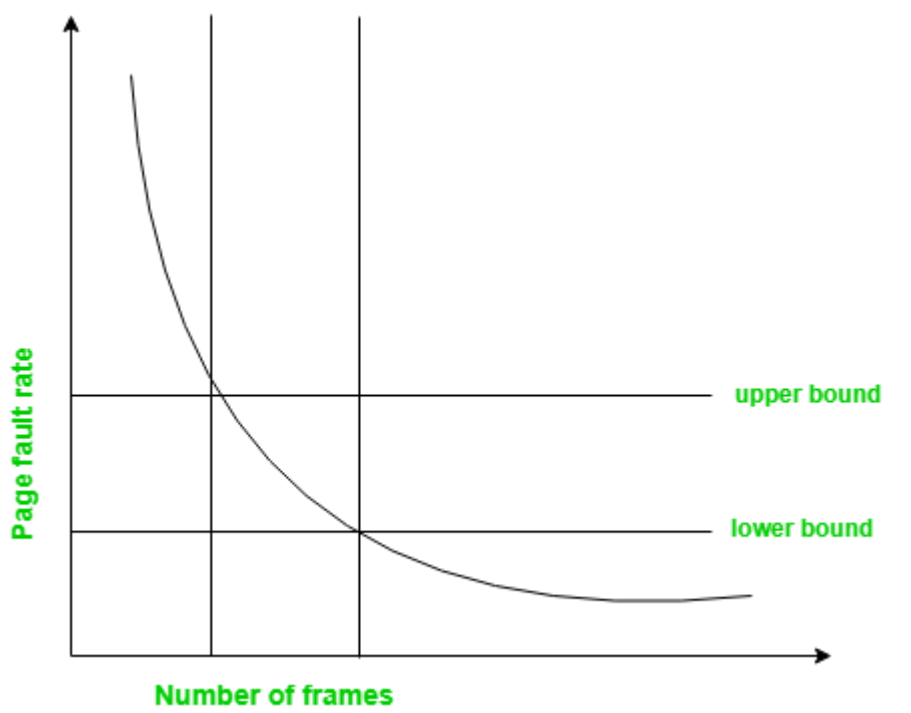
- If D is the total demand for frames and is the working set size for a process i,

$$D = \sum_{n} WSS_i$$

- Now, if ‘m’ is the number of frames available in the memory, there are 2 possibilities:

- (i) D>m i.e. total demand exceeds the number of frames, then thrashing will occur as some processes would not get enough frames.
- (ii) D<=m, then there would be no thrashing.
- If there are enough extra frames, then some more processes can be loaded in the memory. On the other hand, if the summation of working set sizes exceeds the availability of frames, then some of the processes have to be suspended(swapped out of memory).

Page Fault Frequency



- It is some direct approach than working set model.
- When thrashing occurring we know that it has few number of frames.
- And if it is not thrashing that means it has too many frames.
- Based on this property we assign an upper and lower bound for the desired page fault rate.
- According to page fault rate we allocate or remove pages.
- If the page fault rate become less than the lower limit, frames can be removed from the process.

- Similarly, if the page fault rate become more than the upper limit, more number of frames can be allocated to the process.
- And if no frames available due to high page fault rate, we will just suspend the processes and will restart them again when frames available.

Q1

Q) Disk requests are received by a disk drive for cylinders 5, 25, 18, 3, 39, 8, and 35 in that order. A seek takes 5 ms per cylinder moved. How much seek time is needed to serve these requests if serviced in the order that they are received (FCFS)? Assume that the arm is at cylinder 20 when the last of these requests is made with none of these requests yet served.

FCFS=151

SSTF=59

SCAN=57

LOOK=55

Q2

- **Consider a reference string: 4, 7, 6, 1, 7, 6, 1, 2, 7, 2. the number of frames in the memory is 3. Find out the number of page faults respective to:**
- Optimal Page Replacement Algorithm — 5
- FIFO Page Replacement Algorithm — 6
- LRU Page Replacement Algorithm — 6

Q3

- Consider a paging scheme with a TLB. Assume no page fault occurs. It takes 20 ns to search the TLB and 100 ns to access the physical memory. If TLB hit ratio is 80%, the effective memory access time is _____ msec.

- hit ratio * (TLB access time + Main memory access time) +
(1 – hit ratio) * (TLB access time + 2 * main memory time)

1)

- In _____ information is recorded magnetically on platters.
  a) magnetic disks
  b) electrical disks
  c) assemblies
  d) cylinders

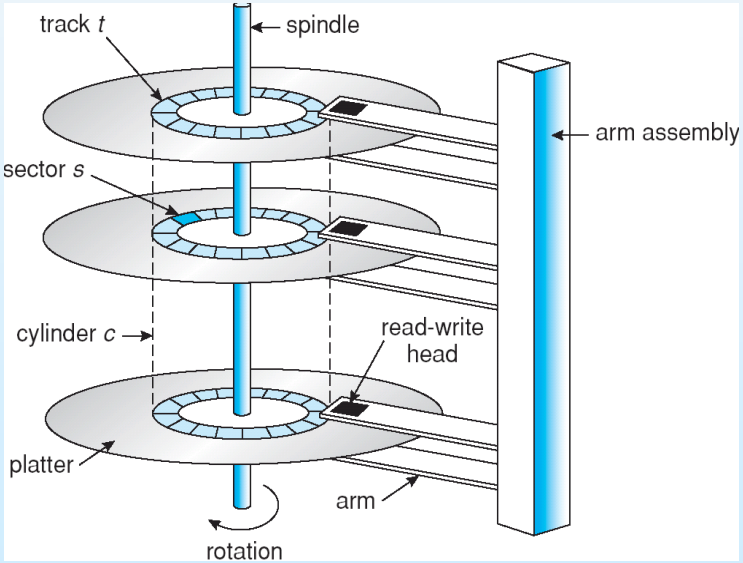
2)

- The heads of the magnetic disk are attached to a _____ that moves all the heads as a unit.

  a) spindle
  b) disk arm
  c) track
  d) none of the mentioned

3)

- The set of tracks that are at one arm position make up a _____
  a) magnetic disks
  b) electrical disks
  c) assemblies
  d) cylinders



4)

- The time taken to move the disk arm to the desired cylinder is called the _____
  a) positioning time
  b) random access time
  c) seek time
  d) rotational latency

5)

- The time taken for the desired sector to rotate to the disk head is called _____
  a) positioning time
  b) random access time
  c) seek time
  d) rotational latency

6)

- . _____ is a unique tag, usually a number identifies the file within the file system.
  a) File identifier
  b) File name
  c) File type
  d) None of the mentioned

7)

- File type can be represented by _____
  a) file name
  b) file extension
  c) file identifier
  d) none of the mentioned

8)

- Which one of the following is the address generated by CPU?
  a) physical address
  b) absolute address
  c) logical address
  d) none of the mentioned

9)

- Run time mapping from virtual to physical address is done by _____
  a) Memory management unit
  b) CPU
  c) PCI
  d) None of the mentioned

10)

- What is compaction?
  a) a technique for overcoming internal fragmentation
  b) a paging technique
  c) a technique for overcoming external fragmentation
  d) a technique for overcoming fatal error

11)

- DMA is used for _____
  a) High speed devices(disks and communications network)
  b) Low speed devices
  c) Utilizing CPU cycles
  d) All of the mentioned

12)

- In an interrupt driven input/output _____
  a) the CPU uses polling to watch the control bit constantly, looping to see if a device is ready
  b) the CPU writes one data byte to the data register and sets a bit in control register to show that a byte is available
  c) the CPU receives an interrupt when the device is ready for the next byte
  d) the CPU runs a user written code and does accordingly

13)

- At least one resource must be held in a non sharable mode. Only one process at a time can use a resource. These statements are true for which deadlock necessary condition?
- A. Mutual Exclusion
- B. Hold and wait
- C. Circular wait
- D. No preemption

- What is convoy effect?
- What is Seek time and rotational latency time?
- What is Belady's Anomalies?
- What is thrashing? What are the different techniques to handle it?

Q14.

Let m[0]…m[4] be mutexes (binary semaphores) and P[0]…P[4] be processes. Suppose each process P[i] executes the following:

wait(m[i]);

wait(m[(i+1) mod 4]);

…….

release(m[i]);

release(m[(i+1) mod 4]);

This could cause-

(a) Thrashing (b) Deadlock (c) Starvation but not deadlock

(d) None of the above

MCQ

- A solution to the critical-section problem must satisfy the following requirements :
- **Mutual Exclusion :**
- **Progress:**
- **Bounded Waiting**
- **Circular wait**
- **Hold and wait**

Consider the following C code for process P1 and P2. a=4, b=0, c=0 (initialization)

```
   P1                    P2
if (a < 0)            b = 10;
  c = b-a;            a = -3;
else
  c = b+a;
```

If the processes P1 and P2 executes concurrently (shared variables a, b and c), which of the following can be the value of ‘c’ after both processes complete?

(A) 4 **(B)** 7 **(C)** 10 **(D)** 13

Answer: (C)
Explanation:
P1 : 1, 3, 4 -> c = 0+4 =4 {hence option a}
P2 : i, ii and P1 : 1, 2 -> c = 10-(-3) = 13 {hence option d}
P1 : 1 , P2 : i, ii and P1 : 3, 4 -> c= 10+(-3) = 7 { hence option b}

- Thank You. All the Best