

## Disk Performance

CIT 595  
Spring 2008

### Disk Performance

- Low CPU utilization can actually indicate a problem in the I/O subsystem
  - CPU spends more time waiting than running
- Disk drives are the slowest memory component
- A slow disk system can drag down the performance of all programs when virtual memory paging is involved
- Optimal disk performance is critical to system throughput

CIT 595

2

### Reducing Seek Time

- Disk arm motion is the greatest consumer of service time
- Disk specifications cite average seek time, which is usually in the range of 5 to 10ms
- However, a full-stroke seek can take as long as 15 to 20ms
- *Schedule* disk access such that we minimize seek time
- Scheduling can be done either by OS or Disk controller

CIT 595

3

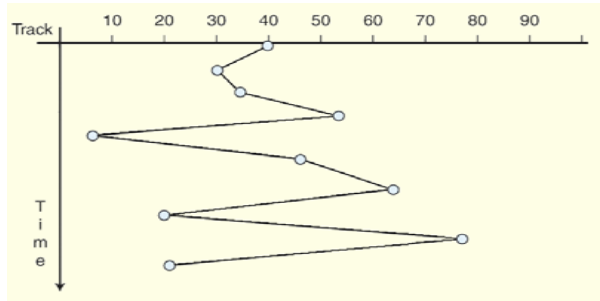
### Disk Scheduling

- Several algorithms exist to schedule the servicing of disk I/O requests
- Algorithms Minimize Seek time  $\approx$  Seek distance
  - i.e. how many tracks need to traveled between the current head position and the next request
- Example are illustrate with 0-99 tracks
  - Requests Order: 28, 35, 52, 6, 46, 62, 19, 75, 21
  - Current Track Serviced i.e. Current Head Position = 40

CIT 595

4

### First Come First Served (FCFS)

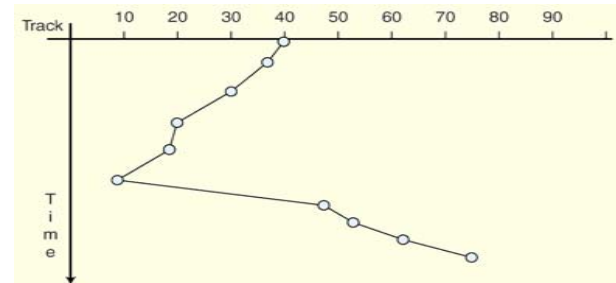


- Order of Service: 28, 35, 52, 6, 46, 62, 19, 75, 21
- Disk arm changes directions 6 times
- Traverses total of 291 tracks

CIT 595

5

### Shortest-Seek Time First



- Order of service : 35, 28, 21, 19, 6, 46, 52, 62, 75
- Disk arm changes direction only once for this example
- Traverses total of 103 tracks

CIT 595

6

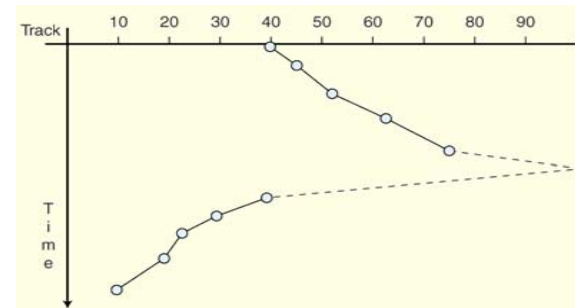
### SCAN

- Avoids SSTF starvation risk
- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests along the way
- When it gets to the other end of the disk the head movement is reversed and servicing continues
- Sometimes called the *elevator algorithm*
- While SCAN entails a lot of arm motion, the motion is constant and predictable

CIT 595

7

### SCAN (Cont.)



- Order of service : 46, 52, 62, 75, 35, 28, 21, 19, 6
- Disk arm changes direction only twice
- Traverses total of 99 tracks

CIT 595

8

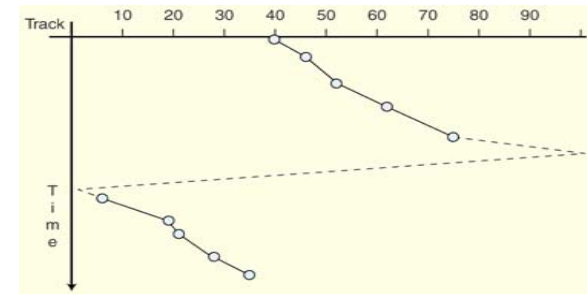
## C-SCAN

- A variant of SCAN
- The head moves from one end of the disk to the other, servicing requests as it goes
- When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip
  - E.g. start at 0, reaches 99, and then starts again at 0
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one

CIT 595

9

## C-SCAN (Cont.)



- Order of service : 46, 52, 62, 75, 6, 19, 21, 28, 35
- Disk reads in one direction only

CIT 595

10

## LOOK and C-LOOK

- Reduces the disk arm motion of SCAN and C-SCAN
- Instead of sweeping the entire disk, the disk arm travels only to the *highest*- and *lowest*-numbered tracks for which access requests are pending
- For example, the arm will traverse only 69 tracks
  - Traverse between 6 and 75
  - Saving 30% arm-motion

CIT 595

11

## RAID

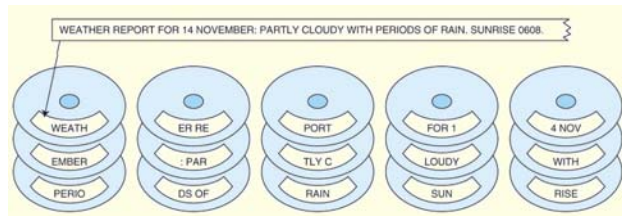
- RAID - Redundant Array of Independent/Inexpensive Disks
  - Invented to address problems of disk reliability, cost, and performance
- In RAID, data is stored across many disks, with extra disks added to the array to provide error correction (redundancy)
  - A RAID controller manages accesses to all drives

CIT 595

12

## RAID 0

- Data is written in blocks across the entire array
  - Block  $i$  of file goes to disk  $(i \bmod n)$ ,  $n = \#$  disks
- Advantage: Parallel read/write
- Disadvantage: No redundancy

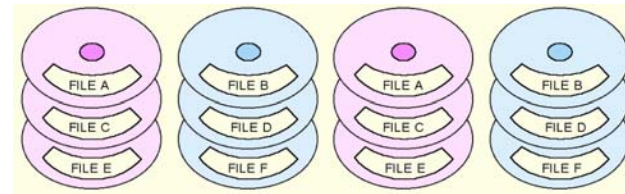


CIT 595

13

## RAID 1

- Mirrors or shadows every disk
- Every write happens to two disks
- Reads to the mirror may happen only when the primary disk fails
  - Or try to read both together and the quicker response is accepted
- Expensive solution: high reliability at twice the cost

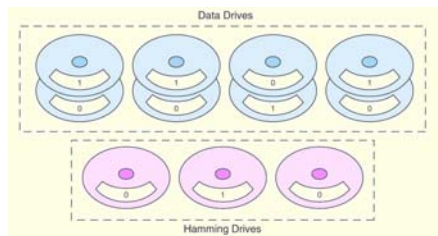


CIT 595

14

## RAID 2

- Uses a technique similar to “striping”
  - Split at the bit level instead of the block level
  - Each bit is written to a separate disk
- Hamming codes are generated for each word
  - Spread across separate Error Correcting Code disks
  - Data is cross-referenced with codes to insure data integrity



CIT 595

15

## RAID 2 (contd..)

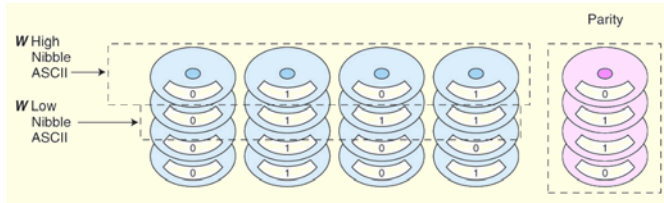
- Provides ECC
  - Overhead is too high
  - # Hamming Drives = #check bits need for 1-bit ECC
- Write performance is compensated since Hamming codes need to be calculated each time
- No commercial implementation
  - Performance is poor and the cost is relatively high

CIT 595

16

### RAID 3

- Stripes bits across a set of data drives and provides a separate disk for parity
- If a drive fails, the correction can be provided since the position of the bad bit is known
  - Controller pretends that all its bits are zero
  - If there is a parity error, then the dead drive value must have been 1



CIT 595

17

### RAID 3

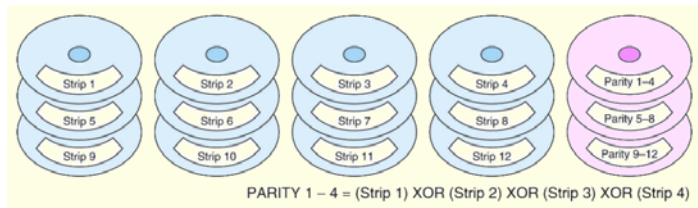
- Is Economical over RAID 1 and RAID 2
  - Uses only one drive for redundancy
- Supports lower number of disk requests as all drives need to participate on every request
- Good when large blocks data is read or written
  - E.g. image or video processing
    - Errors within stripe can be tolerated with such applications

CIT 595

18

### RAID 4

- Combine RAID 0 and RAID 3
  - Striping is done at block-level
  - Separate parity drive to hold



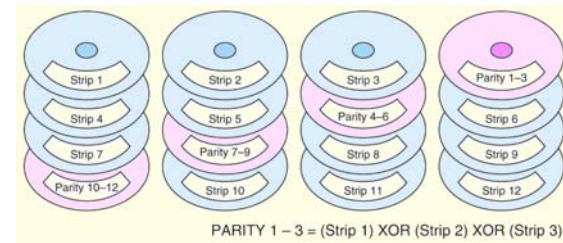
- Parity drive is bottleneck for independent writes
- No commercial implementation

CIT 595

19

### RAID 5

- Just like RAID 4 except parity stripe is spread throughout the entire array



- Can service some requests concurrently
- Used in file and application server, and databases

CIT 595

20

### RAID 5 (contd..)

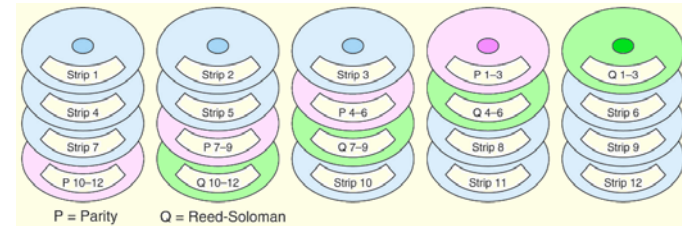
- Unlike RAID 2, RAID 5 requires extra check bits per stripes to correct soft errors
- Spare drives (“hot spares”) are pre-installed disks that do not take part in the RAID set until one of the active drives fail
- When a drive failure is detected, that drive is marked as bad, and reconstruction is immediately started on the first available spare drive without interrupting services

CIT 595

21

### RAID 6

- Able to tolerate more than one concurrent drive failure
- Parity + ECC
- Storage cost increase but reliability very high



CIT 595

22