Chapter 12: Mass-Storage Systems

Operating System and Disk Mgmt

- Major OS jobs are to manage physical devices and to present a virtual machine abstraction to applications.
- For hard disks, the OS provides two abstractions:
  - Raw device: an array of data blocks.
  - File system: the OS queues and schedules the interleaved requests from several applications.

Data Transfer from Memory to Disk

- Reading and Writing Files:
  - Logical record: the unit of data as user thinks of it.
  - Physical record: the unit of data that OS processes.

Buffering

- OS must fill up one buffer before actually writing the block out to disk.

Writing Data Out to Disk

```c
write(fd, data_addr, count); /* fd = file descriptor */
for (i=0; i<count; i++) { /* write big array to disk */
    for (j=0; j<block_size; j++)
        write(fd, array[i][j], sizeof array[i][j]);
```
Buffering Schemes

(a) No buffering

(b) Single buffering

Figure 11.6 I/O Buffering Schemes (input)

Buffering Schemes (cont’d)

(c) Double buffering

(d) Circular buffering

Figure 11.6 I/O Buffering Schemes (input)

Write System Call

Pseudo-code for write system call:

```c
void write(int data_addr, count) {
    if (enough space remains in current buffer) {
        copy count bytes from data_addr to buffer
        buffer += count;
    } else {
        b = number of bytes still available in the buffer;
        copy b bytes of data into buffer;
        *buffer is now full */
        write buffer to disk;
        reset buffer pointer to beginning of buffer;
        reset buffer to zero;
        copy (count - b) bytes to buffer;
        buffer += (count-b);
    }
}
```

OS Buffer Pool

Overview of Disks

- Magnetic disks provide bulk of secondary storage of modern computers
- Drives rotate at 60 to 200 times per second
- Transfer rate is rate at which data flow between drive and computer
- Positioning time (random-access time) is time to move disk arm to desired cylinder (seek time) and time for desired sector to rotate under the disk head (rotational latency)
- Head crash results from disk head making contact with the disk surface
- Drive attached to computer via I/O bus
- Busses vary, including EIDE, ATA, SATA, USB, Fibre Channel, SCSI
- Host controller in computer uses bus to talk to disk controller built into drive or storage array

Moving-head Disk Mechanism
Disk Structure

- Disk drives are addressed as large 1-dimensional arrays of logical blocks, where the logical block is the smallest unit of transfer.
- The 1-dimensional array of logical blocks is mapped into the sectors of the disk sequentially:
  - Sector 0 is the first sector of the first track on the outermost cylinder.
  - Mapping proceeds in order through that track, then the rest of the tracks in that cylinder, and then through the rest of the cylinders from outermost to innermost.

Disk Attachment

- Host-attached storage accessed through I/O ports talking to I/O busses
- SCSI itself is a bus, up to 16 devices on one cable, SCSI initiator requests operation and SCSI targets perform tasks
  - Each target can have up to 8 logical units (disks attached to device controller)
- FC is high-speed serial architecture
  - the basis of storage area networks (SANs) in which many hosts attach to many storage units
  - Can be arbitrated loop (FC-AL) of 126 devices

Network-Attached Storage

- Network-attached storage (NAS) is storage made available over a network rather than over a local connection (such as a bus)
- NFS and CIFS are common protocols
- Implemented via remote procedure calls (RPCs) between host and storage
- New iSCSI protocol uses IP network to carry the SCSI protocol

Storage Area Network

- Common in large storage environments (and becoming more common)
- Multiple hosts attached to multiple storage arrays - flexible

Disk Scheduling

- The operating system is responsible for using hardware efficiently — for the disk drives, this means having a fast access time and disk bandwidth.
- Disk bandwidth is the total number of bytes transferred, divided by the total time between the first request for service and the completion of the last transfer.
- Access time has two major components
  - Seek time is the time for the disk arm to move the heads to the cylinder containing the desired sector.
  - Rotational latency is the additional time waiting for the disk to rotate the desired sector to the disk head.
- Algorithms to minimize seek time: FCFS, SSTF, SCAN, C-SCAN, C-LOOK

Disk Scheduling Algorithms

- Schedule the servicing of disk I/O requests wrt seek time.
- We illustrate them with a request queue of cylinder numbers (0-199).
  - 98, 183, 37, 122, 14, 124, 65, 67
- Head pointer 53
**FCFS**

Illustration shows total head movement of 640 cylinders.

```
queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
```

**SSTF**

- Selects the request with the minimum seek time from the current head position.
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests.
- Illustration shows total head movement of 236 cylinders.

**SSTF (Cont.)**

```
queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
```

**SCAN**

- The disk arm starts at one end of the disk, and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- Sometimes called the elevator algorithm.
- Illustration shows total head movement of 208 cylinders.

**SCAN (Cont.)**

```
queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
```

**C-SCAN**

- Provides a more uniform wait time than 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.
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.
C-SCAN (Cont.)

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53

0  14  37  53 65 67  98  122 124  183 196

C-LOOK

- Version of C-SCAN
- Arm only goes as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.

C-LOOK (Cont.)

queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53

0  14  37  53 65 67  98  122 124  183 196

Selecting a Disk-Scheduling Algorithm

- SSTF is common and has a natural appeal
- SCAN and C-SCAN perform better for systems that place a heavy load on the disk.
- Performance depends on the number and types of requests.
- Requests for disk service can be influenced by the file-allocation method.
- The disk-scheduling algorithm should be written as a separate module of the operating system, allowing it to be replaced with a different algorithm if necessary.
- Either SSTF or C-LOOK is a reasonable choice for the default algorithm.

Disk Management

- Low-level formatting, or physical formatting — Dividing a disk into sectors that the disk controller can read and write.
- To use a disk to hold files, the operating system still needs to record its own data structures on the disk.
  - Partition the disk into one or more groups of cylinders.
  - Logical formatting or "making a file system".
- Boot block initializes system.
  - The bootstrap is stored in ROM.
  - Bootstrap loader program.
- Methods such as sector sparing used to handle bad blocks.

Swap-Space Management

- Swap-space — Virtual memory uses disk space as an extension of main memory.
- Swap-space can be carved out of the normal file system, or, more commonly, it can be in a separate disk partition.
- Swap-space management
  - 4.3BSD allocates swap space when process starts; holds text segment (the program) and data segment.
  - Kernel uses swap maps to track swap-space use.
  - Solaris 2 allocates swap space only when a page is forced out of physical memory, not when the virtual memory page is first created.
RAID Structure

- RAID – multiple disk drives provides reliability via redundancy.
- RAID is arranged into six different levels.
- Randy Katz, UC Berkeley

RAID (cont)

- Several improvements in disk-use techniques involve the use of multiple disks working cooperatively.
- Disk striping uses groups of disks as one storage unit.
- RAID schemes improve performance and improve the reliability of the storage system by storing redundant data.
  - Mirroring or shadowing keeps duplicate of each disk.
  - Block interleaved parity uses much less redundancy.

RAID Levels

Data is Striped for improved performance
- Distributed data over multiple disks so that it appears as a single fast large disk
- Allows multiple I/Os to be serviced in parallel
- Multiple independent requests serviced in parallel
- A block request may be serviced by multiple disks

Data is Redundant for improved reliability
- Large number of disks in a single unit ensures reliability of the array
  - Reliability of N disks = Reliability of 1 disk / N
  - Example
    - 4000 hours / 70 disks = 57 hours
    - Disk System MTTF drops from 6 years to 1 month
  - Arrays without redundancy are unreliable

RAID

- RAID 0 (Non-redundant)
  - Stripes data; but does not employ redundancy
  - Lowest cost of any RAID
  - Best Write performance – can be done in parallel to multiple disks
  - Any single disk failure is catastrophic
  - Used in environments where performance is more important than reliability.
RAID 1 (Mirrored)
- Uses twice as many disks as non-redundant arrays - 100% Capacity Overhead - Two copies of data are maintained
- Data is simultaneously written to both arrays
- Data is read from the array with shorter queuing, seek and rotation delays - Best Read Performance.
- When a disk fails, mirrored copy is still available
- Used in environments where availability and performance (I/O rate) are more important than storage efficiency.

RAID (0 + 1) and (1 + 0)

RAID 2 (Memory Style ECC)
- Uses Hamming code - parity for distinct overlapping subsets of data
- # of redundant disks is proportional to log of total # of disks - Better for large # of disks - e.g., 4 data disks require 3 redundant disks
- If disk fails, other data in subset is used to regenerate lost data
- Multiple redundant disks are needed to identify faulty disk

RAID 3 (Bit Interleaved Parity)
- Data in a block is stored bit-wise over the data disks
- Uses Single parity disk to tolerate disk failures - Overhead is 1/N
- Logically a single high capacity, high transfer rate disk
- Reads access data disks only; Writes access both data and parity disks
- Used in environments that require high BW (Scientific, Image Processing, etc.), and not high I/O rates

RAID 4 (Block Interleaved Parity)
- Similar to bit interleaved parity disk array; except data is block-interleaved (Stripe Units)
- Read requests smaller than one striping unit, access one striping unit
- Write requests update the data block and parity block
- Generating parity requires 4 I/O accesses (RMW)
- Parity disk gets updates on all writes - a bottleneck

RAID 5 (Block-Interleaved Distributed Parity)
- Eliminates the parity disk bottleneck in RAID 4 - Distributes parity among all the disks
- Data is distributed among all disks
- All disks participates in read requests - Better performance than RAID 4
- Write requests update the data block and the parity block.
- Generating parity requires 4 I/O accesses (RMW)
- Left symmetry v.s. Right Symmetry - Allows each disk to be traversed once before any disk twice
12.43 RAID 6 (P + Q Redundancy)
- Uses Reed-Solomon codes to protect against up to 2 disk failures
- Data is distributed among all disks
- Two sets of parity P & Q
- Generating parity requires 6 I/O accesses (RMW) - update both P & Q
- Used in environments that require stringent reliability requirements

Comparisons
- Read/Write Performance
  - RAID 0 provides the best Write performance
  - RAID 1 provides the best Read performance
- Cost: Total # of Disks
  - RAID 1 is most expensive - 100% capacity overhead - 2N disks
  - RAID 0 is least expensive - N disks - no redundancy
  - RAID 2 needs N + ceiling(log₂N) + 1
  - RAID 3, RAID 4 & RAID 5 needs N + 1 disks

Preferred Environments
- RAID 0: Performance & capacity are more important than reliability
- RAID 0: High IO rate, high availability environments
- RAID 2: Large IO Data Transfer
- RAID 3: High I/O Applications (Scientific, Image Processing...)
- RAID 4: High I/O Applications
- RAID 5 & RAID 6: Mixed Applications

The table below, which shows Throughput per SS relative to RAID 0, assumes that G drives in an error correcting group.
What RAID for which application

- **Fast Workstation:**
  - Caching is important to improve I/O rate
  - If large files are installed, then RAID 0 may be necessary
  - It is preferred to put the OS and swap files in separate drives from user drives to minimize movement between swap file area & user area.

- **Small Server:**
  - RAID 1 is preferred
  - If more capacity is needed, then RAID 5 is recommended
  - Small Server: e.g. Database Servers
    - RAID 5 is preferred
    - Separate different I/Os in mechanically independent arrays; place index & data files in databases in different arrays.

- **Mid-Size Server:**
  - RAID 1 is preferred

- **Large Server:**
  - RAID 5 is recommended
  - Large Server: e.g. Database Servers
    - RAID 5 is recommended
    - Separate different I/Os in mechanically independent arrays; place index & data files in databases in different arrays.