Case for storage

- Shift in focus from computation to communication & storage of information
  - E.g., Cray Research/Thinking Machines vs. Google/Yahoo!
  - “The Computing Revolution” (1960s to 1980s) ⇒ “The Information Age” (1990s to today)
- Storage emphasizes reliability and scalability as well as cost-performance
- What is “software king” that determines which hardware features are actually used?
  - OS for storage
  - Compiler for processor
- Also has own performance theory–queuing theory–balances throughput vs. latency time
Outline

- Magnetic disks
- RAID
- Dependability/reliability/availability
- I/O benchmarks, performance, and dependability

Magnetic disks

- Spindle, platters, head, head arm, head arm actuator

- Tracks, sectors
  - Tracks: concentric circles
  - Sectors: unit of data access

- Standard interface
  - ATA: parallel (old), serial (SATA, new)
  - SCSI: parallel (old), serial (SAS, new)

- Performance determined by
  - Seek (mechanical), rotation speed (mechanical)
  - Queuing/buffering, interface
Disk figure of merit: Areal density

- Bits recorded along a track
  - Metric is *bits per inch* (BPI)
- Number of tracks per surface
  - Metric is *tracks per inch* (TPI)
- Disk designs brag about bit density per unit area
  - Metric is *bits per square inch (areal density) = BPI × TPI*

Specific design ideas

- Zoned recording
  - Outer tracks are longer than inner tracks; why not allocate more sectors on outer tracks?
  - Drives usually have 15 to 25 zones
  - Defect management in zones

- Serpentine ordering of tracks; skewed ordering of sectors
  - Minimize arm movement!

- Command queuing

- Security support (data protection)
Historical perspective

- 1956 IBM RAMAC – early 1970s Winchester
  - Developed for mainframe computers, proprietary interfaces
  - Steady shrink in form factor: 27 inches to 14 inches

IBM RAMAC
Historical perspective

- 1956 IBM RAMAC – early 1970s Winchester
  - Developed for mainframe computers, proprietary interfaces
  - Steady shrink in form factor: 27 inches to 14 inches
- Form factor and capacity drives market more than performance
- 1970s developments
  - 5.25-in. floppy disk form factor
  - Emergence of industry standard disk interfaces
- Early 1980s: PCs and first generation workstations
- Mid 1980s: Client-server computing
  - Centralized storage on file server ⇒ accelerates disk downsizing: 8 in. to 5.25 in.
  - Mass market disk drives become a reality ⇒ 5.25 in. to 3.5 in. drives for PCs, end of proprietary interfaces
- 1990s: Laptops ⇒ 2.5-in. drives
- 2000s: Continued areal density improvement, new solid-state devices entering hard disk designs

Solid-state drive
Solid-state drive

<table>
<thead>
<tr>
<th>Time frame</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>2007–2008</td>
<td>4-way, 4 channels, 30–80 MB/s R/W performance; mostly SLC flash based;</td>
</tr>
<tr>
<td>2008–2009</td>
<td>8–10 channels, 150–200+ MB/s performance (SATA, consumer); 16+ channels, 600+ MB/s performance (PCI-e, enterprise); use of MLC flash in consumer products;</td>
</tr>
<tr>
<td>2009–2010</td>
<td>16+ channels, 200–300+ MB/s performance (SATA 6 Gbps); 20+ channels, 1+ GB/s performance (PCI-e); adoption of MLC in enterprise products;</td>
</tr>
<tr>
<td>2010–</td>
<td>16+ channels; wider acceptance of PCI-e;</td>
</tr>
</tbody>
</table>
Past and current design trends

- Fewer platters, lower diameter (thanks to high areal density)
  - 1~3 platters
  - 2.5-inch form factor
  - Slower rotational speeds

- More intelligence in the drive
  - Access scheduling (w/ help from native command queuing)
  - Integrated defect management

- Larger sector size (512B to 4kB)

Future disk sizes and performance

- Continued advance in capacity (60%/year) and bandwidth (40%/year)
- Slow improvement in seek, rotation (8%/year)

- Time to read whole disk

<table>
<thead>
<tr>
<th>Year</th>
<th>Sequentially</th>
<th>Randomly (1 sector/seek)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1990</td>
<td>4 minutes</td>
<td>6 hours</td>
</tr>
<tr>
<td>2000</td>
<td>12 minutes</td>
<td>1 week</td>
</tr>
<tr>
<td>2006</td>
<td>56 minutes</td>
<td>3 weeks (SCSI)</td>
</tr>
<tr>
<td>2006</td>
<td>171 minutes</td>
<td>7 weeks (SATA)</td>
</tr>
</tbody>
</table>
Use arrays of small disks?

- Katz and Patterson of Berkeley asked in 1987:
  - Can smaller disks be used to close gap in performance between disks and CPUs?

**Conventional:**
4 disk designs

- 3.5"
- 5.25"
- 10"
- 14"

**Disk Array:**
1 disk design

Advantages of small form-factor disk drives

- Low cost/MB
- High MB/volume
- High MB/watt
- Low cost/Actuator

Cost and Environmental Efficiencies
Replace small number of large disks with large number of small disks! (1988 disks)

<table>
<thead>
<tr>
<th></th>
<th>IBM 3390K</th>
<th>IBM 3.5&quot; 0061</th>
<th>x70</th>
</tr>
</thead>
<tbody>
<tr>
<td>Capacity</td>
<td>20 GBytes</td>
<td>320 MBytes</td>
<td>23 GBytes</td>
</tr>
<tr>
<td>Volume</td>
<td>97 cu. ft.</td>
<td>0.1 cu. ft.</td>
<td>11 cu. ft.</td>
</tr>
<tr>
<td>Power</td>
<td>3 KW</td>
<td>11 W</td>
<td>1 KW</td>
</tr>
<tr>
<td>Data Rate</td>
<td>15 MB/s</td>
<td>1.5 MB/s</td>
<td>120 MB/s</td>
</tr>
<tr>
<td>I/O Rate</td>
<td>600 I/Os/s</td>
<td>55 I/Os/s</td>
<td>3900 I/Os/s</td>
</tr>
<tr>
<td>MTTF</td>
<td>250 Khrs</td>
<td>50 Khrs</td>
<td>??? Khrs</td>
</tr>
<tr>
<td>Cost</td>
<td>$250K</td>
<td>$2K</td>
<td>$150K</td>
</tr>
</tbody>
</table>

Disk Arrays have potential for large data and I/O rates, high MB per cu. ft., high MB per KW, **but what about reliability?**

**Array reliability**

- Reliability of N disks = (Reliability of 1 disk)/N
  - 50,000 hours/70 disks = 700 hours
  - Disk system MTTF: drops from 6 years to 1 month!

- Arrays (without redundancy) too unreliable to be useful!

- Hot spares support reconstruction in parallel with access: Very high media availability can be achieved
**Redundant array of (inexpensive) disks (RAID)**

- Files are striped across multiple disks
- Redundancy yields high data availability
  - Availability: Service still provided to user, even if some components failed
- Disks will still fail
- Contents reconstructed from data redundantly stored in the array
  - Capacity penalty to store redundant information
  - Bandwidth penalty to update redundant information

**RAID 1: Disk mirroring/shadowing**

- Each disk is fully duplicated onto its “mirror”
  - High availability can be achieved
- Bandwidth sacrifice on write:
  - Logical write = two physical writes
- Most expensive solution: 100% capacity overhead
RAID 3: Parity disk

- P contains sum of other disks per stripe mod 2 ("parity")
- If disk fails, subtract P from sum of other disks to find missing information

RAID 3

- Sum computed across recover group to protect against hard disk failures, stored in P disks
- Logically, a single high capacity, high transfer rate disk: Good for large transfers
- Wide arrays reduce capacity costs, but decreases availability
- 33% capacity cost for parity if 3 data disks and 1 parity disk
Inspiration for RAID 4

- RAID 3 relies on parity disk to discover errors on read
- But every sector has an error detection/correction field
- To catch errors on read, rely on error detection/correction field vs. the parity disk

- Allows independent reads to different disks simultaneously

RAID 4: High I/O rate parity

Example:
- small read D0 & D5
- large write D12-D15
Inspiration for RAID 5

- RAID 4 works well for small reads
- Small writes (write to one disk):
  - Option 1: Read other data disks, create new sum and write to parity disk
  - Option 2: Since P has old sum, compare old data to new data, add the difference to P
- Small writes are limited by parity disk: Write to D0, D5 both also write to parity disk

RAID 5: High I/O rate, interleaved parity

Independent writes possible because of interleaved parity

Example: write to D0, D5 uses disks 0, 1, 3, 4
**Small writes**

*RAID 5: Small Write Algorithm*

1 Logical Write = 2 Physical Reads + 2 Physical Writes

- D0'
- D0
- D1
- D2
- D3
- P

- new data
- old data
- old parity
- XOR

(1. Read) + D0'
(2. Read) + D1
(3. Write) + D2
(4. Write) + D3

**RAID 6: Recovering from 2 failures**

- Why > 1 failure recovery?
  - Operator accidentally replaces the wrong disk during a failure
  - Since disk bandwidth is growing more slowly than disk capacity, the mean time to repair a disk in a RAID system is increasing \( \Rightarrow \) increases the chances of a 2\textsuperscript{nd} failure during repair since repair takes longer
RAID 6: Recovering from 2 failures

- NetApp’s row-diagonal parity or RAID-DP
- Like the standard RAID schemes, it uses redundant space based on parity calculation per stripe
- Since it is protecting against a double failure, it adds two check blocks per stripe of data
  - If \( p+1 \) disks total, \( p-1 \) disks have data
- Row parity disk is just like in RAID 4
- Each block of the diagonal parity disk contains the even parity of the blocks in the same diagonal

Example w/ \( p = 5 \)

- Row diagonal parity starts by recovering one of the 4 blocks on the failed disk using diagonal parity
  - Since each diagonal misses one disk, and all diagonals miss a different disk, 2 diagonals are only missing 1 block
- Once the data for those blocks are recovered, then the standard RAID recovery scheme can be used to recover two more blocks in the standard RAID 4 stripes
- Process continues until two failed disks are restored
Berkeley RAID-I

- RAID-I (1989)
  - Sun 4/280 w/ 128MB of DRAM
  - Four dual-string SCSI controllers
  - 28 5.25-in. SCSI disks and specialized disk striping software
- Today, RAID is $24B industry, 80% non-PC disks sold in RAIDs

Summary: RAID techniques

- Disk mirroring (RAID 1)
  - Each disk is fully duplicated onto its “mirror”
  - Logical write = two physical writes
  - 100% capacity overhead
- Parity data bandwidth array (RAID 3)
  - Parity computed horizontally
  - Logically a single high data bandwidth disk
- High I/O rate parity array (RAID 5)
  - Interleaved parity blocks
  - Independent reads and writes
  - Logical write = 2 reads + 2 writes