Review - 6 Great Ideas in Computer Architecture

1. Layers of Representation/Interpretation
2. Moore’s Law
3. Principle of Locality/Memory Hierarchy
4. Parallelism
5. Performance Measurement & Improvement
6. Dependability via Redundancy

Review - Great Idea #6: Dependability via Redundancy

- Redundancy so that a failing piece doesn’t make the whole system fail
- Increasing transistor density reduces the cost of redundancy
- 1+1=2
- 1+1=2
- 1+1=2

2 of 3 agree

FAIL!

Magnetic Disk – common I/O device

- A kind of computer memory
  - Information stored by magnetizing ferrite material on surface of rotating disk
  - Similar to tape recorder except digital rather than analog data
- Nonvolatile storage
  - Retains its value without applying power to disk.
- Two Types
  - Floppy disks – slower, less dense, removable.
  - Hard Disk Drives (HDD) – faster, more dense, non-removable.
- Purpose in computer systems (Hard Drive):
  - Long-term, inexpensive storage for files
  - “Backup” for main-memory. Large, inexpensive, slow level in the memory hierarchy (virtual memory)

Photo of Disk Head, Arm, Actuator
Disk Device Terminology

- Several platters, with information recorded magnetically on both surfaces (usually)
- Bits recorded in tracks, which in turn divided into sectors (e.g., 512 Bytes)
- Actuator moves head (end of arm) over track ("seek"), wait for sector rotate under head, then read or write

Where does Flash memory come in?

- Microdrives and Flash memory (e.g., CompactFlash going head-to-head
  - Both non-volatile (no power, data ok)
  - Flash benefits: durable & lower power
    (no moving parts, need to spin drives up/down)
  - Flash limitations: finite number of write cycles (wear on the insulating oxide layer around the charge storage mechanism).
    Most ≥ 100K; some ≥ 1M W/erase cycles.
- How does Flash memory work?
  - NMOS transistor with an additional conductor between gate and source/drain which "traps" electrons. The presence/absence is a 1 or 0.

Use Arrays of Small Disks...

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

Replace Small # of Large Disks with Large # of Small!

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

Disk Arrays potentially high performance, high MB per cu. ft., high MB per KW, but what about reliability?

Replace Small # of Large Disks with Large # of Small!

<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>??? Hrs</td>
</tr>
<tr>
<td>Cost</td>
<td>$250K</td>
<td>$2K</td>
<td>$150K</td>
</tr>
</tbody>
</table>

Disk Arrays potentially high performance, high MB per cu. ft., high MB per KW, but what about reliability?
Array Reliability

- Reliability - whether or not a component has failed
  - measured as Mean Time To Failure (MTTF)
- Reliability of N disks
  - Reliability of 1 Disk ÷ N (assuming failures independent)
  - 50,000 Hours ÷ 70 disks = 700 hour
- Disk system MTTF: Drops from 6 years to 1 month!
- Disk arrays too unreliable to be useful!

Redundant Arrays of (Inexpensive) Disks

- 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 info
  => Bandwidth penalty to update redundant info

RAID: Redundant Array of Inexpensive Disks

- Invented @ Berkeley (1989)
- A multi-billion industry
  - 80% non-PC disks sold in RAIDs
- Idea:
  - Files are “striped” across multiple disks
  - Redundancy yields high data availability
    - Disk will still fail
  - Contents reconstructed from data redundantly stored in the array
    - Capacity penalty to store redundant info
    - Bandwidth penalty to update redundant info

“RAID 0”: No redundancy = “AID”

- Assume have 4 disks of data for this example, organized in blocks
- Large accesses faster since transfer from several disks at once

RAID 1: Mirror data

- Each disk is fully duplicated onto its “mirror”
  - Very high availability can be achieved
- Bandwidth reduced on write:
  - 1 Logical write = 2 physical writes
- Most expensive solution: 100% capacity overhead

RAID 3: Parity

- Spindles synchronized, each sequential byte on a diff. drive
- Parity computed across group to protect against hard disk failures, stored in P disk
- Logically, a single high capacity, high transfer rate disk
- 25% capacity cost for parity in this example vs. 100% for RAID 1 (5 disks vs. 8 disks)
- Q: How many drive failures can be tolerated?
Inspiration for RAID 5 (RAID 4 block-stripping)

- Small writes (write to one disk):
  - Option 1: read other data disks, create new sum and write to Parity Disk (access all disks)
  - Option 2: since P has old sum, compare old data to new data, add the difference to P:
    
    \[ \text{1 logical write = 2 physical reads + 2 physical writes to 2 disks} \]

- Parity Disk is bottleneck for Small writes: Write to A0, B1 \( \Rightarrow \) both write to P disk

RAID 5: Rotated Parity, faster small writes

- Independent writes possible because of interleaved parity
  
  - Example: write to A0, B1 uses disks 0, 1, 4, 5, so can proceed in parallel
  - Still 1 small write = 4 physical disk accesses

“And in conclusion...”

- I/O gives computers their 5 senses
- I/O speed range is 100-million to one
- Processor speed means must synchronize with I/O devices before use: Polling vs. Interrupts
- Networks are another form of I/O
- Protocol suites allow networking of heterogeneous components
  
  - Another form of principle of abstraction
  
  - RAID
    
    \( \text{Higher performance with more disk arms per $} \)
    
    \( \text{More disks = More disk failures} \)
    
    \( \text{Different RAID levels provide different cost/speed/reliability tradeoffs} \)

Bonus: Disk Device Performance (1/2)

- Disk Latency = Seek Time + Rotation Time + Transfer Time + Controller Overhead
  
  - Seek Time? depends on no. tracks to move arm, speed of actuator
  
  - Rotation Time? depends on speed disk rotates, how far sector is from head
  
  - Transfer Time? depends on data rate (bandwidth) of disk (f(bit density/rpm)), size of request

Bonus: Disk Device Performance (2/2)

- Average distance of sector from head?
  
  \( \text{1/2 time of a rotation} \)
  
  \( \text{120 Rev/sec} \)
  
  \( \text{8.33 milliseconds} \)

- Average no. tracks to move arm?
  
  \( \text{Disk industry standard benchmark:} \)
  
  \( \text{Assumes average seek distance is random} \)

- Size of Disk cache can strongly affect perf!
  
  \( \text{Cache built into disk system, OS knows nothing} \)

BONUS: Hard Drives are Sealed. Why?

- The closer the head to the disk, the smaller the “spot size” and thus the denser the recording.
  
  \( \text{Measured in Gbit/in². ~60 is state of the art.} \)

- Disks are sealed to keep the dust out.
  
  Heads are designed to “fly” at around 5-20nm above the surface of the disk.
  
  \( \text{99.999% of the head/arm weight is supported by the air bearing force (air cushion) developed between the disk and the head.} \)