Storage Systems

Jin-Soo Kim (jinsookim@skku.edu)
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu
Today’s Topics

- Disks
- Disk scheduling policies
Secondary Storage

- **Secondary storage usually**
  - is anything that is outside of “primary memory”.
  - does not permit direct execution of instructions or data retrieval via machine load/store instructions.

- **Characteristics**
  - It’s large: 100GB and more
  - It’s cheap: 1TB SATA2 disk costs ₩100,000.
  - It’s persistent: data survives power loss.
  - It’s slow: milliseconds to access.
Disk Device (1)

- Seek time
- Rotational latency
- Transfer rate
Disk Device (2)

- **Seagate Barracuda ST31000528AS (1TB)**
  - 4 Heads, 2 Discs
  - Max. recording density: 1413K BPI (bits/inch)
  - Avg. track density: 236K TPI (tracks/inch)
  - Avg. areal density: 329 Gbits/sq.inch
  - Spindle speed: 7200rpm (8.3ms/rotation)
  - Average seek time: < 8.5ms (read), < 9.5ms (write)
  - Max. internal data transfer rate: 1695 Mbits/sec
  - Max. I/O data transfer rate: 300MB/sec (SATA-2)
  - Max. sustained data transfer rate: 125MB/sec
  - Internal cache buffer: 32MB
  - Max power-on to ready: < 10.0 sec
Disks (1)

- Physical disk structure
  - platters
  - surfaces
  - tracks
  - sectors
  - cylinders
  - arm
  - heads
Disks (2)

Disks and the OS

- Disks are messy physical devices:
  - Errors, bad blocks, missed seeks, etc.
- The job of the OS is to hide this mess from higher-level software:
  - Low-level device drivers (initiate a disk read, etc)
  - Higher-level abstractions (files, databases, etc.)
- The OS may provide different levels of disk access to different clients:
  - Physical disk block (surface, cylinder, sector)
  - Disk logical block (disk block #)
  - Logical file (filename, block or record or byte #)
Interacting with disks

- Specifying disk requests requires a lot of info:
  - Cylinder #, surface #, track #, sector #, transfer size, etc.
- Older disks required the OS to specify all of this
  - The OS needs to know all disk parameters.
- Modern disks are more complicated.
  - Not all sectors are the same size, sectors are remapped, etc.
- Current disks provide a higher-level interface (e.g., SCSI)
  - The disks exports its data as a logical array of blocks [0..N-1]
  - Disk maps logical blocks to cylinder/surface/track/sector.
  - Only need to specify the logical block # to read/write.
  - As a result, physical parameters are hidden from OS.
Disks (4)

- **Disk performance**
  - Performance depends on a number of steps
    - **Seek**: moving the disk arm to the correct cylinder
      → depends on how fast disk arm can move (increasing very slowly)
    - **Rotation**: waiting for the sector to rotate under head
      → depends on rotation rate of disk (increasing, but slowly)
    - **Transfer**: transferring data from surface into disk controller, sending it back to the host.
      → depends on density of bytes on disk (increasing, and very quickly)
  - **Disk scheduling:**
    - Because seeks are so expensive, the OS tries to schedule disk requests that are queued waiting for the disk.
FCFS

- **FCFS (= do nothing)**
  - Reasonable when load is low.
  - Long waiting times for long request queues.
SSTF

- **Shortest seek time first**
  - Minimizes arm movement (seek time)
  - Maximizes request rate
  - Unfairly favors middle blocks
  - May cause starvation of some requests

![Diagram](image)
SCAN

- **Elevator algorithm**
  - Service requests in one direction until done, then reverse
  - Skews wait times non-uniformly

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

- Like SCAN, but only go in one direction (e.g. typewriters)
- Uniform wait times

```plaintext
queue = 98, 183, 37, 122, 14, 124, 65, 67
head starts at 53
```
LOOK / C-LOOK

- Similar to SCAN/C-SCAN, but the arm goes only as far as the final request in each direction.

<table>
<thead>
<tr>
<th>queue</th>
<th>98, 183, 37, 122, 14, 124, 65, 67</th>
</tr>
</thead>
<tbody>
<tr>
<td>head starts at</td>
<td>53</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>0</th>
<th>14</th>
<th>37</th>
<th>53</th>
<th>65</th>
<th>67</th>
<th>98</th>
<th>122</th>
<th>124</th>
<th>183</th>
<th>199</th>
</tr>
</thead>
</table>

C-LOOK
Disk Scheduling (1)

- 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.
  - Either SSTF or LOOK is a reasonable choice for the default algorithm.
  - Performance depends on the number and types of requests.
  - Requests for disk service can be influenced by the file allocation method.
  - In general, unless there are request queues, disk scheduling does not have much impact.
    - Important for servers, less so for PCs
  - Modern disks often do the disk scheduling themselves.
    - Disks know their layout better than OS, can optimize better.
    - Ignores, undoes any scheduling done by OS.
Disk Scheduling (2)

- **Intelligent controllers**
  - Nowadays, most disk controllers are built around a small CPU and have many kilobytes of memory.
  - They run a program written by the controller manufacturer to process I/O requests from the CPU and satisfy them.
  - Intelligent features:
    - Read-ahead: the current track
    - Caching: frequently-used blocks
    - Command queueing
    - Request reordering: for seek and/or rotational optimality
    - Request retry on hardware failure
    - Bad block identification
    - Bad block remapping: onto spare blocks and/or tracks