Numbers – Key Points

- there is a huge difference in performance between random and sequential disk accesses
  - transfer data sequentially if possible, and in as large chunks as possible otherwise

	Cheetah	Barracuda
$R_{I/O}$ Random	0.66 MB/s	0.31 MB/s
$R_{I/O}$ Sequential	125 MB/s	105 MB/s

- there is a substantial difference in performance between disks built for performance and those built for cheap capacity

Disk Scheduling

- I/O is slow and there's a big difference between random and sequential access, so there is potentially much to be gained by clever ordering of I/O requests
- can reasonably estimate the time required for a disk access
  - based on average seek time and rotational delay

Disk Scheduling – SSTF/SSF

- idea: *shortest seek (time) first* (SSTF/SSF)
  - order I/O requests by track – nearest track to the current track first
- problems
  - disk geometry isn't known to OS – disk is seen only as an array of blocks
    - *nearest block first* (NBF) – order I/O requests by block and choose the nearest block to the current block first
  - starvation
    - a steady stream of requests to the same group of nearby tracks means requests for farther-away tracks are never scheduled

Disk Scheduling – SCAN (C-SCAN, F-SCAN, Elevator)

- idea: repeatedly sweep from outer-to-inner (or inner-to-outer) tracks, handling queued requests according to sweep order
  - maintaining a fixed order rather than back-and-forth avoids privileging middle tracks
  - can freeze queue contents when a sweep begins so that nearby but later-arriving requests don't jump ahead of farther-away ones that have been waiting

Disk Scheduling - SPTF/SATF

- SSTF and SCAN only consider seek time
- idea: *shortest positioning time first* (SPTF) or *shortest access time first* (SATF)
- wrinkle
  - OS doesn't know internals of disk geometry and head positioning

Other Scheduling Issues

- who does disk scheduling?
  - OS picks a small group of the best requests to send to the drive
    - based on an approximation of seek time – SSTF, SCAN, or similar
  - SPTF is performed by the drive
- I/O merging
  - OS disk scheduler combines separate requests for consecutive blocks
    - reduces requests sent to disk
    - leverages greater sequential-access efficiency
- how quickly to send I/O requests to the disk?
  - *work-conserving* approach sends immediately, so that disk is not idle while there are pending I/O requests
  - holding on to the request allows additional requests to arrive which can potentially be handled more efficiently