Announcements

- Project #5 is due on May 1st at 5:00 pm
- Midterm #2 was returned
- Stats
  - Min: 46
  - Max 99
  - Mean: 72
  - Standard Dev: 12.2

Disk Scheduling Exercise

- **After talking about the goals of disk head scheduling**
  - The class split into 7 groups
  - Each group created a scheduling algorithm
  - Groups presented their solutions
- **Algorithms presented**
  - Shortest request first
  - Scan scheduling
  - Circular Scan
  - Time weighted scheduling
    - Schedule by distance + time weighting
  - Sorted Batch Scheduling
    - Batch up to n requests
    - Service batch with minimum seek time ordering

Disk Scheduling

- **First come, first served**
  - ordering may lead to lots of disk head movement
  - i.e. 1, 190, 3, 170, 4, 160 etc.
  - total number of tracks traversed : 863

- **Shortest seek time first: select request with the minimum seek time from current head position**
  - move head to closest track
  - i.e. 1,3,4,160,170,190
  - total number of tracks traversed: 189
  - potential problem with distant tracks not getting service for an indefinite period

Disk Scheduling

- Scan scheduling - read-write head starts at one end of the disk, moves to the other, servicing requests as it reaches each track
  - Consider example: 1, 190, 3, 170, 4, 160
  - If head starts at track 64 and moves towards 0, the ordering would be 4,3,1,160,170,190
  - Total distance 265

- C-Scan (circular scan)
  - disk head sweeps in only one direction
  - when the disk head reaches one end, it returns to the other
  - Consider example: 1, 190, 3, 170, 4, 160
  - If head starts at track 64 and moves towards 0, the ordering would be 4,3,1,190,170,160
  - Total distance 282

Disk Cache

- Buffer in main memory for disk sectors

- Cache contains copy of some of the sectors on a disk. When I/O request is made for a sector, a check is made to find out if sector is in the disk cache

- Replacement strategy:
    - Least recently used: block that has been in the cache longest with no reference gets replaced
    - Least frequently used: block that experiences fewest references gets replaced