CS162 Discussion #10: Disks and Queuing

Announcements

- Phase 2 initial design due tonight at 11:59pm (`submit proj3-initial-design`)
    - Drop off a hard copy of your doc in the bag at 346 Soda
- Sign up for design reviews!
    - Monday 11/8 in 278 Cory, Tuesday 11/9 in 310 Soda
- Midterm regrade requests (written) must be turned in on Monday 11/8 at latest
- Make sure to let Angela know if you're having any group issues!
- Terrible discussion handout puns are back. (People complained when they were gone.)

If you'll be my disk, I'll be your diskette

Last week with demand paging, we talked about "writing pages to disk" and "reading pages out of disk", but we haven't actually talked about what a disk is. A hard disk is a non-volatile storage device: it can retain data without power. A disk has the following components:

- *Platters*. A number of rotating circular platters, positioned in a column.
- *Read-Write Heads*. Heads are located above/between the platters, and read or write information to them using magnetization.

Here are some different ways we can look at the regions of the disk:

- *Track*. A fixed-radius circular band on a platter.
- *Cylinder*. The group of tracks over all platters corresponding to some fixed radius.
- *Sector*. A region of a track. This is the smallest independently addressable unit on the disk – each sector on the disk contains the same number of bits.
- *Block*. A group of sectors to be transferred together.

The rotation of the platters allows the head to read data off of the disk as the data passes by, by sensing the magnetization on the platter below the head. Therefore, without moving the head, we have access to an entire cylinder of information. We can change the angle of the arm on which the heads are located to change between tracks; this is called seeking. This combination of seek and rotation allows us to cover any part of the disk we need to.

Note that if we take sectors in the strictly mathematical sense (i.e., each sector is a region between two fixed angles from the center of the platter), we get sectors like pie slices, where the sectors on the outer tracks are wider than the ones on the inner tracks. However, since all sectors have to contain the same number of bits, the bits on the inside get squished and have little extra room, while the bits on the outside have a lot of extra room.

We can improve our usage of the disk by using a technique called zoned bit recording, which makes all sectors on the disk the same size so that the bit density stays constant. This means outer tracks will have more sectors than inner tracks. This brings us to another interesting point – since the outer tracks have more sectors using zoned bit recording, the data transfer rate there is higher. No matter how fast the platter is spinning, a greater distance will be traversed by the head on the outer tracks in comparison with the inner tracks within the same amount of time. Therefore, when reading from the outer tracks, we read a larger amount of data in the same amount of time.

QQ

Before a request to the disk can be serviced, it has to wait on a queue. The amount of time we'll be waiting on this queue after making the request is a factor in disk latency (Remember: Disk Latency = Queuing Time + Controller Time + Seek Time + Rotation Time + Transfer Time), so we'd like to be able to calculate what this might come out to. We use queuing theory to do this. Note that queuing theory applies to steady-state behavior only.

Before jumping into calculations, here are some parameters we will need:

λ	Mean number of arriving customers per second
T_{ser}	Mean time to service a customer
C	Squared coefficient of variance (σ^2 / T_{ser}^2)
μ	Service rate ($1 / T_{ser}$)
u	Server utilization ($\lambda / \mu = \lambda * T_{ser}$), must be between 0 and 1

All of the above parameters are observed or derived directly from observed parameters. From these parameters, we would like to compute T_q (time spent in the queue) and L_q (length of the queue). We have a handy rule called Little's Law to help us, which states:

Average # of tasks in the system = arrival rate * average response time

This tells us that:

$$L_q = \lambda * T_q$$

The general formula for computing T_q is:

$$T_q = T_{ser} * 0.5 (1 + C) * u / (1 - u)$$

Using these formulas with the observed parameters, we can calculate our desired quantities, as well as disk latency.