CS469/CS569 Final Exam

May 11th, 2009

Write your answers in the blue book(s). Justify your answers. Work alone. Do not use any notes or books.

There are four problems on this exam, each worth 20 points, for a total of 80 points. You have approximately three hours to complete this exam.

1 Randomized mergesort (20 points)

Consider the following randomized version of the mergesort algorithm. We take an unsorted list of n elements and split it into two lists by flipping an independent fair coin for each element to decide which list to put it in. We then recursively sort the two lists, and merge the resulting sorted lists. The merge procedure involves repeatedly comparing the smallest element in each of the two lists and removing the smaller element found, until one of the lists is empty.

Compute the expected number of comparisons needed to perform this final merge. (You do not need to consider the cost of performing the recursive sorts.)

Solution

Color the elements in the final merged list red or blue based on which sublist they came from. The only elements that do no require a comparison to insert into the main list are those that are followed only by elements of the same color; the expected number of such elements is equal to the expected length of the longest monochromatic suffix. By symmetry, this is the same as the expected longest monochromatic prefix, which is equal to the expected length of the longest sequence of identical coin-flips.

The probability of getting k identical coin-flips in a row followed by a different coin-flip is exactly 2^{-k}; the first coin-flip sets the color, the next $k-1$ must follow it (giving a factor of 2^{-k+1}, and the last must be the opposite color (giving an additional factor of 2^{-1}). For n identical coin-flips, there is a probability of 2^{-n+1}, since we don't need an extra coin-flip of the opposite color. So the expected length is $\sum_{k=1}^{n-1} k2^{-k} + n2^{-n+1} = \sum_{k=0}^{n} k2^{-k} + n2^{-n}$.

We can simplify the sum using generating functions. The sum $\sum_{k=0}^{n} 2^{-k}z^k$ is given by $\frac{1-(z/2)^{n+1}}{1-z/2}$. Taking the derivative with respect to z gives $\sum_{k=0}^{n} 2^{-k}kz^{k-1} = (1/2)\frac{1-(z/2)^{n+1}}{1-z/2}^2 + (1/2)\frac{(n+1)(z/2)^n}{1-z/2}$. At $z = 1$ this is $2(1 - 2^{-n-1}) - 2(n +$

$1)2^{-n} = 2 - (n+2)2^{-n}$. Adding the second term gives $\mathrm{E}[X] = 2 - 2 \cdot 2^{-n} = 2 - 2^{-n+1}$.

Note that this counts the expected number of elements for which we do not have to do a comparison; with n elements total, this leaves $n-2+2^{-n+1}$ comparisons on average.

2 A search problem (20 points)

Suppose you are searching a space by generating new instances of some problem from old ones. Each instance is either good or bad; if you generate a new instance from a good instance, the new instance is also good, and if you generate a new instance from a bad instance, the new instance is also bad.

Suppose that your start with X_0 good instances and Y_0 bad instances, and that at each step you choose one of the instances you already have uniformly at random to generate a new instance. What is the expected number of good instances you have after n steps?

Hint: Consider the sequence of values $\{X_t/(X_t + Y_t)\}$.

Solution

We can show that the suggested sequence is a martingale, by computing

$$\begin{aligned}
\mathrm{E}\left[\frac{X_{t+1}}{X_{t+1}Y_{t+1}}|X_t, Y_t\right] &= \frac{X_t}{X_t + Y_t}\frac{X_t+1}{X_t+Y_t+1} + \frac{Y_t}{X_t+Y_t}\frac{X_t}{X_t+Y_t+1} \\
&= \frac{X_t(X_t+1)Y_tX_t}{(X_t+Y_t)+(X_t+Y_t+1)} \\
&= \frac{X_t(X_t+Y_t+1)}{(X_t+Y_t)+(X_t+Y_t+1)} \\
&= \frac{X_t}{X_t+Y+t}.
\end{aligned}$$

From the martingale property we have $\mathrm{E}\left[\frac{X_n}{X_n+Y_n}\right] = \frac{X_0}{X_0+Y+0}$. But $X_n + Y_n = X_0 + Y_0 + n$, a constant, so we can multiply both sides by this value to get $\mathrm{E}[X_n] = X_0\left(\frac{X_0+Y_0+n}{X_0+Y_0}\right)$.

3 Support your local police (20 points)

At one point I lived in a city whose local police department supported themselves in part by collecting fines for speeding tickets. A speeding ticket would cost 1 unit (approximately \$100), and it was unpredictable how often one would get a speeding ticket. For a price of 2 units, it was possible to purchase a metal placard to go on your vehicle identifying yourself as a supporter of the police union, which (at least according to local legend) would eliminate any fines for subsequent speeding tickets, but which would not eliminate the cost of any previous speeding tickets.

Let us consider the question of when to purchase a placard as a problem in on-line algorithms. It is possible to achieve a strict[1] competitive ratio of 2 by purchasing a placard after the second ticket. If one receives fewer than 2 tickets, both the on-line and off-line algorithms pay the same amount, and at 2 or more tickets the on-line algorithm pays 4 while the off-line pays 2 (the off-line algorithm purchased the placard before receiving any tickets at all).

1. Show that no deterministic algorithm can achieve a lower (strict) competitive ratio.

2. Show that a randomized algorithm can do so, against an oblivious adversary.

Solution

1. Any deterministic algorithm essentially just chooses some fixed number m of tickets to collect before buying the placard. Let n be the actual number of tickets issued. For $m = 0$, the competitive ratio is infinite when $n = 0$. For $m = 1$, the competitive ratio is 3 when $n = 1$. For $m > 2$, the competitive ratio is $(m + 2)/2 > 2$ when $n = m$. So $m = 2$ is the optimal choice.

2. Consider the following algorithm: with probability p, we purchase a placard after 1 ticket, and with probability $q = 1 - p$, we purchase a placard after 2 tickets. This gives a competitive ratio of 1 for $n = 0$, $1 + 2p$ for $n = 1$, and $(3p + 4q)/2 = (4 - p)/2 = 2 - p/2$ for $n \geq 2$. There is a clearly a trade-off between the two ratios $1+2p$ and $2-p/2$. The break-even point is when they are equal, at $p = 2/5$. This gives a competitive ratio of $1 + 2p = 9/5$, which is less than 2.

[1] I.e., with no additive constant.

4 Overloaded machines (20 points)

Suppose n^2 jobs are assigned to n machines with each job choosing a machine independently and uniformly at random. Let the load on a machine be the number of jobs assigned to it. Show that for any fixed $\delta > 0$ and sufficiently large n, there is a constant $c < 1$ such that the maximum load exceeds $(1+\delta)n$ with probability at most nc^n.

Solution

This is a job for Chernoff bounds. For any particular machine, the load S is a sum of independent indicator variables and the mean load is $\mu = n$. So we have

$$\Pr[S \geq (1+\delta)\mu] \leq \left(\frac{e^\delta}{(1+\delta)^{1+\delta}}\right)^n.$$

Observe that $e^\delta/(1+\delta)^{1+\delta} < 1$ for $\delta > 0$. One proof of this fact is to take the log to get $\delta - (1+\delta)\log(1+\delta)$, which equals 0 at $\delta = 0$, and then show that the logarithm is decreasing by showing that $\frac{d}{d\delta} \cdots = 1 - \frac{1+\delta}{1+\delta} - \log(1+\delta) = -\log(1+\delta) < 0$ for all $\delta > 0$.

So we can let $c = e^\delta/(1+\delta)^{1+\delta}$ to get a bound of c^n on the probability that any particular machine is overloaded and a bound of nc^n (from the union bound) on the probability that any of the machines is overloaded.