CS91T Week 7 In-Lab Exercises

March 16, 2023

The learning goals of lab exercises this week is to build up comfort and intuition of how an algorithm can use randomness, and to work with a CS application of randomness.

1. **Coarse Bounds for the Harmonic Number.** The n-th Harmonic number H_n is defined as the sum of the *recipricols* of the first n natural numbers:

$$H_n = \sum_{k=1}^{n} \frac{1}{k} \ .$$

Similar to factorials, very tight approximations are known for H_n: $H_n = \ln(n) + O(1)$. In this exercise you will develop a quick and easy way of getting coarse bounds.

(a) Show that $H_n \leq \log(n)$. To show this inequality, let $n = 2^k - 1$ for some k.

- First, write out the terms of the Harmonic number: (I've done this for you)

$$H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{2^k} \ .$$

- Second, take each term, and "round the term up" to the nearest power of 2. For example, $\frac{1}{5} \leq \frac{1}{4}$, $\frac{1}{61} \leq \frac{1}{32}$, and $\frac{1}{1025} \leq \frac{1}{1024}$. Note also that $\frac{1}{2} \leq \frac{1}{2}$.
Give an upper bound for H_n by rounding up each term.

$$H_n \leq \text{\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_} \ .$$

- Next, add up terms with the same value. For example, there should be two terms in the sum of value $\frac{1}{2}$, so replace $\frac{1}{2} + \frac{1}{2}$ with $2 \cdot \frac{1}{2} = 1$.

$$H_n \leq \text{\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_} \ .$$

- Finally, add up terms in the above sum to get $H_n \leq \log(n)$.

(b) Use similar calculations to the one above to show that $H_n \geq 1 + \frac{1}{2}\log(n)$.

2. **Coupon Collectors.** Suppose we have the following balls-and-bins problem. There are n bins, and a number of balls, and as usual, each ball is independently and uniformly assigned a bin. This time, our task is to keep throwing balls into bins until each bin has at least one ball. **How many balls do we need until all bins are occupied?** This is known in the CS literature as the *Coupon Collectors Problem.*

(a) Let X be the number of balls thrown before all buckets are occupied. Show that

$$E[X] = nH_n \ .$$

Hint: For $1 \leq i \leq n$, let X_i denote the number of balls thrown while there are exactly $i-1$ occupied bins.

(b) Given $E[X] = nH_n$, we can use Markov's inequality to show that with high probability, the number of balls needed before all balls are occupied is $O(n \log(n))$. However, with a little more effort, it's possible to get a tighter bound.

Let $t = nH_n + cn$ for some constant c. Show that

$$\Pr[X > t] \leq e^{-c} \ .$$

Hint: for $1 \leq i \leq n$, let BAD_i be the event that none of the first t balls end up in bin B_i. Let $BAD = \cup_i BAD_i$. Show that $\Pr[BAD] \leq e^{-c}$ using the Union Bound.

3. **BucketSort.**

Suppose you have a List L of $n = 2^m$ random items. Each item in L is uniformly distributed over the universe $U = \{0, 1, \ldots, 2^k\}$ for some $k \geq m$. For this exercise you will design a Las Vegas algorithm $\mathcal{A}$ that sorts L in expected $O(n)$ time.

For each $z \in \{0,1\}^m$, define a bucket B_z. Algorithm $\mathcal{A}$ works as follows:

- First, scan through L. For each $1 \leq i \leq n$, look at the m most significant digits of $L[i]$, and place $L[i]$ into the bucket B_z whose most significant digits match z. (Assume you can place each item into a bucket in $O(1)$ timesteps.)
- Second, sort items in each bucket B_z using a any $O(n^2)$-time sorting algortihm you like.
- Finally, iterate through all buckets B_z in increasing order, and copy the items from each B_z back into L.

(a) Argue that $\mathcal{A}$ correctly sorts L.

(b) Let X_z denote the number of elements of L that end up in B_z. Argue that $E[X_z^2] = O(1)$.

(c) Next, conclude that the total expected runtime of step 2 is $O(n)$

4. **Asymptotic Analysis Proofs.**

(a) Let $f(n) = 3n^3 - 2n^2$ and $g(n) = 20n^2$. Prove that $f(n) = \Omega(g(n))$.

(b) Prove that $f = O(g)$ if and only if $g = \Omega(f)$.

5. **Comparing Functions Asymptotically.** Examine each of the following pairs of functions, and provide the tightest asymptotic comparison possible. For example, if $f = \Theta(g)$, say so. If $f = O(g)$ but not $f = \Theta(g)$, say $f = O(g)$.

(a) $f_1(n) = n^2$, $g_1(n) = 2n \log(n)$.

(b) $f_2(n) = \frac{10n}{\log(n)}$, $g_2(n) = \frac{n}{\log(n)} - n \log(e)$.

(c) $f_3(n) = n\sqrt{\frac{\log\log(n)}{\log(n)}}$, $g_3(n) = n$.