CSE 312: Foundations of Computing II — Winter 2021

Problem Set 6 (due Wednesday, February 24, 11:59pm)

Directions:
***Answers**: For each problem, remember you must briefly explain/justify how you obtained your answer, as correct answers without an explanation will receive **no credit**. Moreover, in the event of an incorrect answer, we can still try to give you partial credit based on the explanation you provide. It is fine for your answers to include summations, products, factorials, exponentials, or combinations; you dont need to calculate those all out to get a single numeric answer, for instance 26^7 or $26!/7!$ or $26 \cdot \binom{26}{7}$.*

Your solutions need to be concise and clear. We will take off points for lack of clarity or for excess verbosity. Please see section worksheet solutions (posted on the course website) to gauge the level of detail we are expecting.

Please clearly indicate your final answer, in such a way as to distinguish it from the rest of your explanation.

***Groups:** This homework is to be completed in groups of 1 or 2. Specific guidelines about collaboration are available on the syllabus, but every group will be submitting their own submission. Please cite any collaboration at the top of your submission. Instructions are available here as to how to add groupmates to your submission.*

Before you start your homework, write down the list of people you collaborated with. Remember, you can only collaborate outside your group by discussing the problems at a high level. Provide names and email addresses.

***Submission**: You must upload a **pdf** of your written solutions to Gradescope under "PSet 6 [Written]". Problem 6 is a coding problem, so under "Pset 6 [Coding]" you will be uploading a .py file called cse312_pset6_dist_elts.py. Problems 7 and 8 are extra credit problems, so if you answer them upload your solution to "Pset 6 [Extra]". (Instructions as to how to upload your solutions to Gradescope are on the course web page.) The use of latex is highly recommended.*

Note that if you want to hand-write your solutions, you'll need to scan them. We will take off points for hand-written solutions that are difficult to read due to poor handwriting and neatness.)

Please cite any collaboration at the top of your submission (beyond your group members, who should already be listed).

1. Confidence intervals in "real life" (30 points)

In all of the following use the Central Limit Theorem. Use continuity correction if you're working with a discrete approximation.

(a) [10 Points] You're trying to figure out your finances over the upcoming year. Based on your recent spending patterns, you know that you spend \$1200 a month on average, with a standard deviation of \$400, and each month's spending is independent and identically distributed. In addition, because these are hard times, you don't have any income. How much money should you have in your bank account right now if you don't want to go broke in the next 12 months, with probability at least 95%? You should treat the amount of money you'll spend as continuous.

(b) [10 Points] You've decided to try starting a company together with your 312 pset partner. After hours of brainstorming (in between solving 312 problems), you've cut your list of ideas down to 10, all of which you want to implement at the same time. An angel investor has agreed to back all 10 ideas, as long as your net return from implementing the ideas is positive with at least 95% probability.

Suppose that implementing an idea requires 50 thousand dollars, and your start-up then succeeds with probability p, generating 150 thousand dollars in revenue (for a net gain of 100 thousand dollars), or fails with probability $1-p$ (for a net loss of 50 thousand dollars). The success of each idea is independent of every other. What is the condition on p that you need to satisfy in order to secure the angel investor funding? You should treat the amount of revenue you'll get as discrete (since you're only ever adding up multiples of \$100,000 and -\$50,000).

(c) [10 Points] One of your start-ups uses error-correcting codes[1], which can recover the original message as long as at least 1000 packets are received (not erased). Each packet gets erased independently with probability 0.8. How many packets should you send such that you can recover the message with probability at least 99% You should treat the number of packets received as discrete.

2. Polling again (8 points)

In class on 11/4, we discussed an **i**dealized polling procedure and analysis to determine the fraction p of a population that is planning to vote to legalize the therapeutic use of magic mushrooms[2]
Specifically, we analyzed how we could choose the number n of people to sample in order to guarantee that 98% of the time, it will be the case that

$$p \in [\overline{X} - 0.05, \overline{X} + 0.05].$$

You probably know that many recent polls (that follow similar methodology) have been **way off**. Briefly discuss two reasons that real-world polling may not work as well as the idealized polling that we discussed.
Note: This is not a math question, and there can be many different answers.

3. Exponential in all directions (30 points)

A continuous random variable X has a density function with parameter λ given by:

$$f_X(x) = ce^{-\lambda|x|} \qquad -\infty < x < \infty,$$

for some constant c.

(a) [8 Points] If λ is equal to 0 or negative, this is not a valid density function. Explain what property of pdfs is violated when $\lambda \leq 0$.

For the rest of this problem, assume $\lambda > 0$.

(b) [4 Points] Compute the constant c in terms of λ. Graph the pdf for $\lambda = 1$ and $\lambda = 5$. You can use Wolfram Alpha or any other graphing tool.

(c) [4 Points] Compute the mean and variance of X in terms of λ.

(d) [6 Points] Compute $Pr(X \geq x)$ in terms of x and λ. (Note that x can be positive or negative or 0. Consider all cases.)

(e) [4 Points] For $s, t > 0$, compute $Pr(X \geq s + t | X \geq s)$.

(f) [4 Points] Let $Y = |X|$. Compute the pdf of Y.

[1] From the Wikipedia page: "In computing, telecommunication, information theory, and coding theory, an error correcting code (ECC) is used for controlling errors in data over unreliable or noisy communication channels. The central idea is the sender encodes the message with redundant information in the form of an ECC. The redundancy allows the receiver to detect a limited number of errors that may occur anywhere in the message, and often to correct these errors without retransmission."

[2] FYI: This measure passed in Oregon.

4. Analyze code snippet (12 points)

Let "data[ ]" be an array of 100 random variables, where the ith entry in data, i.e. data$[i]$, for $1 \leq i \leq 100$ is a sample from an independent exponential distribution with parameter $\lambda = 5$. The following is a program to compute the minimum of these numbers.

Compute-Min:

```
min := infinity
for t := 1 to 100
    if data[t] < min
        min := data[t]
        print "The new minimum is " min            (*)
```

(a) [2 Points] What is the worst case number of times line (*) is executed?

(b) [10 Points] What is the expected number of times line (*) is executed?

Hint: Use linearity of expectation. You will also need to apply the law of total probability in the continuous case (to be discussed in an upcoming class).

Here is all you need to know for now (and you should use this to solve the problem even before we discuss the following in class): Suppose that Y and W are independent random variables. Then as you recall, if they are both discrete random variables, we can use the law of total probability to see that

$$\begin{aligned} Pr(Y \leq W) &= \sum_{x \in \Omega_W} Pr(Y \leq W | W = x) Pr(W = x) \\ &= \sum_{x \in \Omega_W} \frac{Pr(Y \leq W, W = x)}{Pr(W = x)} Pr(W = x) \\ &= \sum_{x \in \Omega_W} \frac{Pr(Y \leq x, W = x)}{Pr(W = x)} Pr(W = x) \\ &= \sum_{x \in \Omega_W} Pr(Y \leq x, W = x) \\ &= \sum_{x \in \Omega_W} Pr(Y \leq x) Pr(W = x) \end{aligned}$$

Notice that the last line uses the independence of Y and W. Similarly (and the following is what you will use in this problem), if they are continuous random variables then the **continuous law of total probability** says that for independent Y and W:

$$Pr(Y \leq W) = \int_{-\infty}^{\infty} Pr(Y \leq W | W = x) f_W(x) dx = \int_{-\infty}^{\infty} Pr(Y \leq x) f_W(x) dx.$$

Notice the analogy to the discrete case. We are integrating over the range of the random variable W (rather than summing over it), and we are conditioning on the probability of W being equal to x. We are also replacing $Pr(W = x)$ in the discrete case with $f_W(x)dx$ in the continuous case. We will justify this formula in an upcoming lecture, but for now you can simply use it.

5. Distinct Elements Analysis (9 points)

YouTube wants to count the number of ***distinct*** views for a video, but doesn't want to store all the user ID's. In class on 11/9 and 11/13, we described a way for them to get a good estimate of this number without storing everything.

We modelled the problem as follows: we see a **stream** of 8-byte integers (user ID's), $x_1, x_2, \ldots, x_N$, where x_i is the user ID of the i-th view to a video, but there are only n *distinct* elements ($1 \leq n \leq N$), since some people rewatch the video, even multiple times. We don't know what the number of views N is; we can't even store the number n of distinct views (i.e., the number of distinct views).

(a) [5 Points] Let $U_1, \ldots, U_m$ be m iid samples from the continuous $Unif(0,1)$ distribution, and let $X = \min\{U_1, \ldots, U_m\}$. We know from lecture (and the textbook) that $E[X] = \frac{1}{m+1}$. Compute $Var[X]$.

(b) [2 Points] If we were to solve the Distinct Elements problem naively using a Set, what would the big-Oh space complexity be (in terms of N and/or n)? If a video had $N = 2$ billion views, with only $n = 900$ million of them being distinct views, how much storage would we need for this one video to keep track of the distinct users? Your answer should be in bytes.

(c) [2 Points] Determine how much space is saved from using the MultDistElts class if YouTube wants to use $K = 10,000$ hash functions compared to using the Set in part (b)? Assume for simplicity MultDistElts only stores $K = 10,000$ floats val_i (each float is 8 bytes, and corresponds to a certain DistElts class). What is the multiplicative savings factor (e.g., 10x, 2x, etc)?

6. Distinct Elements [Coding] (11 points)

In this problem we are going to be writing the code to go with the Distinct Elements Analysis. It continues off the same setup as the previous question.

(a) [5 Points] Suppose the universe of user ID's is the set $\mathcal{U}$ (think of this as all 8-byte integers), and we have a single ***uniform*** hash function $h : \mathcal{U} \to [0,1]$. That is, for an integer y, pretend $h(y)$ is a **continuous** $Unif(0,1)$ random variable. That is, $h(y_1), h(y_2), ..., h(y_k)$ for any k **distinct** elements are iid continuous $Unif(0,1)$ random variables, but since the hash function always gives the same output for some given input, if, for example, the i-th user ID x_i and the j-th user ID x_j are the same, then $h(x_i) = h(x_j)$ (i.e., they are the "same" $Unif(0,1)$ random variable).

In class we discussed how to (approximately) solve this distinct elements problem using a single floating point variable (8 bytes), instead of the amount of memory the naive approach from question 5b requires. Pseudocode is provided which explains the two key functions:

(a) UPDATE(x): How to update your variable when you see a new stream element.

(b) ESTIMATE(): At any given time, how to estimate the number of distinct elements you've seen so far.

Algorithm 1 Distinct Elements Operations

```
function INITIALIZE()
    val ← ∞
function UPDATE(x)
    val ← min {val, hash(x)}
function ESTIMATE()
    return round (1/val − 1)
for i = 1, ..., N: do                    ▷ Loop through all stream elements
    update(x_i)                          ▷ Update our single float variable
return estimate()                        ▷ An estimate for n, the number of distinct elements.
```

Implement the functions UPDATE and ESTIMATE in the DistElts class of cse312_pset6_dist_elts.py.

(b) [6 Points] The estimator we used in (a) has high variance, so isn't great sometimes. To solve this problem, we will keep track of K DistElts classes, take the mean of our K mins, and then apply the same trick as earlier to give an estimate. This will reduce the variance of our estimate significantly.

In a bit more detail, this means that after n different users have viewed the video, we have updated K different values

$$\text{val}_1, \text{val}_2, \ldots, \text{val}_K$$

where each of val_i, $1 \leq i \leq K$ is an i.i.d. random variable with the distribution of the minimum of n independent $Unif(0,1)$ variables. Our final estimate will then be

$$\hat{n} = \frac{1}{\widehat{\text{val}}} - 1 \quad \text{where} \quad \widehat{\text{val}} = \frac{1}{K}\sum_{i=1}^{K} \text{val}_i.$$

Implement the functions UPDATE and ESTIMATE in the MultDistElts class of cse312_pset6_dist_elts.py using the improved estimator.

Refer to Section 9.4 of the book for more details on the distinct elements algorithm.

7. Extra Credit: Incommunicado (5 points)

n people are brought into a room. A single bit (either 0 or 1) gets pasted on each person's forehead. They cannot see the bit on their own forehead, but they can see the bits on everyone else's foreheads.

Without communicating and without seeing what anyone else writes, they must each write down a bit (either 0 or 1) which should be equal to the number on their own forehead. If they all get it right, they will each earn a prize of one million dollars, but if any of them get it wrong, there will be no prizes at all.

They can agree ahead of time on a protocol, that is before the numbers are placed on their foreheads. But once they are brought into the room, no further communication is allowed. Come up with a protocol that guarantees they will each win a million dollars with probability $1/2$.

8. Extra Credit: Beyond distinct elements (5 points)

Suppose we have a stream $x_1, \ldots, x_N$, with only $n < N$ distinct elements. The **distinct sum** problem is: instead of returning n, return the *sum* of the n distinct elements. When asked for pseudocode, the LaTeX pseudocode from problem 5 is included in the LaTeX template linked with this pset on the calendar. Feel free to copy it and modify it.

(a) [2 Points] Suppose the stream elements are all *positive integers*. Explain how we can solve this "positive integer *distinct sum*" problem. Then, explicitly give pseudocode for the update and estimate functions. (Hint: You can "reduce" this problem to the distinct elements problem. This means transforming the stream somehow and using instance(s) of the distinct elements class. You can assume you have a distinct elements implementation which handles any datatype (int, string, etc.).)

(b) [3 Points] Suppose the stream elements are all *integers* (could be positive or negative). Explain how we can solve this "integer *distinct sum*" problem. Then, explicitly give pseudocode for the update and estimate functions. (Hint: Reduction.)