Buffalo Hird
CS124
Assignment 7

Problem 1:

We note that for this problem we are only concerned with the last 4 digits of the IDs as this is what passwords are constructed from, such that it does not matter how many digits precede the last 4. As a result, we note that the answer is precisely the same for any length ID number >= 4. We then note that this is an instance of the classic birthday problem. We note that this is a possibility of 10,000 possible passwords as they range from 0000 to 9999. We then note that there are given a person's password in a group of n, $n-1$ possibilities with $p = \frac{1}{10{,}000}$ for someone to match this person's password. We can more easily calculate the probability of no sharing, which for n people is $\sum_{i=0}^{n} \frac{10{,}000-k+1}{10{,}000^k}$. This is the case as for each new password we create, we have one fewer new passwords we can assign out of 10,000 that is not already allocated to a user. We note that this therefore becomes a 100% chance of shared password after 10,000 users. However, we can find the number such that we expect $p > 0.50$ by plugging in n values at which we find it more likely than not that there will be shared passwords.

Plugging in values for n we find

$$1 - \frac{\prod_{i=0}^{n} 10,000 - k + 1}{10,000^n} > 0.50, n \geq 119$$

For an 8 digit unique ID, we have probability of 2 matching of:

$$1 - \frac{\prod_{i=0}^{n} 100,000,000 - k + 1}{100,000,000^n} > 0.50, n \geq 11,775$$

For a 12 digit unique ID, we have probability of 2 matching of:

$$1 - \frac{\prod_{i=0}^{n} 1,000,000,000,000 - k + 1}{1,000,000,000,000^n} > 0.50, n \geq 1,177,411$$

Problem 2:

We note that this is similar to the previous problem. We want an instance of the birthday problem such that we are finding the q, or probability of no birthday matching (hashes) for value for $x \geq \sqrt{n}c_1$:

$$\prod_{k=0}^{x} \frac{n-k+1}{n} = \prod_{k=0}^{x} (1 - \frac{k}{n})$$

We note that as we are looking for values such that this is at most probability $\frac{1}{e}$.

We note that $e^{-x} \geq 1 - x$ such that for $x = \frac{k}{n}$ we expect the right hand to be larger. We therefore have:

$$\prod_{k=0}^{x1}(1 - \frac{k}{n}) \leq \prod e^{\frac{k}{n}}$$

We can consider how this is likewise a sum of exponents such that we have:

$$probability \leq e^{\Sigma_{k=0}^{x} - \frac{k}{n}} \rightarrow probability \leq e^{\frac{1}{n} * \frac{-x(x-1)}{2}} \rightarrow probability \leq e^{\frac{1}{n} * \frac{-x^2+x}{2}}$$

As previously stated we are attempting to bound this value by $\frac{1}{e}$ such that we have:

$$\frac{1}{n} * \frac{-x^2 + x}{2} \leq -1 \rightarrow x^2 - x - 2n \geq 2$$

We can then find the tighest bound for this value when $x = \sqrt{n} * c_1$. As we know this function is increasing we expected the worst case to occur when $n = 1$:

$$x^2 - x - 2n \geq 0 \rightarrow c_1^2 - c_1 \frac{1}{\sqrt{n}} - 2 \geq 0$$

$$n = 1, c_1^2 - c_1 - 2 \geq 0 \rightarrow c_1 \geq 2$$

We have therefore solved this problem to be the case when $c_1 \geq 2$.

We now consider c_2 such that when there are at most $c_2\sqrt{n}$ people in the room the q of no two having the same hash value is $\geq \frac{1}{2}$. Here we use the two identities and first write this for $x \leq c_2\sqrt{n}$:

$$\prod_{k=0}^{x} \frac{n - k + 1}{n} = \prod_{k=0}^{x}(1 - \frac{k}{n})$$

Here instead we want to bound this probability to $\geq \frac{1}{2}$, which by showing that something smaller than this probability satisfies this relation then our probability must then too. We use the identity $e^{-x-x^2} \leq 1 - x$ to compute:

$$\prod_{k=0}^{x}(1 - \frac{k}{n}) \geq \prod_{k=0}^{x} e^{-\frac{k}{n} - \frac{k^2}{n^2}}$$

We make the same product of exponents is equivalent to the sum of the exponents themselves again:

$$\prod_{k=0}^{x}(1 - \frac{k}{n}) \geq e^{\Sigma_{k=0}^{x} - \frac{k}{n} - \frac{k^2}{n^2}}$$

$$\geq e^{\frac{-x(x-1)}{2n} - \frac{x(x-1)(2x-1)}{6n^2}}$$

As previously stated we are attempting to bound this value by $\frac{1}{2}$ such that we have:

$$\geq \frac{1}{2}$$

$$\frac{-x(x-1)}{2n} - \frac{x(x-1)(2x-1)}{6n^2} \geq -ln(2)$$

$$\frac{x^2-x}{n} - \frac{x(x-1)(2x-1)}{3n^2} \leq 2ln(2)$$

We can then find the tighest bound for this value when $x = \sqrt{n} * c_2$. This function will be dominating in the squared term as we approach large value

$$\frac{c_2^2 n}{n} - \frac{c_2\sqrt{n}}{n} - \frac{c_2\sqrt{n}(c_2\sqrt{n}-1)(2c_2\sqrt{n}-1)}{3n^2} \leq 2ln(2) \to c_2^2 \leq 2ln(2) \to c_2 \leq \sqrt{2ln(2)}$$

We have therefore bounded this problem to $c_2 \leq \sqrt{2ln(2)}$

Problem 3:

Letting $X_{i,j}$ be a random variable which is 1 if the ith and jth element are compared on $QuickSelect(A, k)$,we note that the running time is proportional to $\Sigma_{i<j} X_{i,j}$. We give an exact expression for this expected value - using case analysis - given that $i < j$. We note that ith and jth will only even be compared if one is selected to be a pivot by the algorithm and the expected value relies on the probabiloty of this occuring. We therefore have 3 cases where this pivot is chosen such that we choose a pivot that divides i and j from the subarray A:

Case 1: We select a pivot from the interval $[i, k]$ where $i < j < k$

We note that in this case the two are only compared if either are selected as the pivot. If the pivot is greater than j then they will not be used. if the pivot point is between i and j then i will not be used, while if the point is after k and/or below i then will be compared. We therefore discover that this only occurs when one of them is chosen as a pivot initially, such that we have the expected value of this is:

$$\frac{2}{j-k+1}$$

As this occurs when i and j are both less than k, this occurs with probability

$$\frac{n-j}{n-2}$$

Case 2: We select a pivot from the interval $[k, j]$ where $k < i < j$

We note that this is similar to case 1, noting that we will have this comparison occur if either i or j is chosen to be a pivot, otherwise the subarray k in $QuickSelect(A, k)$ will be called without these two indices such that they will never be prepared. We therefore again have:

$$\frac{2}{j-k+1}$$

As this occurs only when both i and j are greater than k, we have similarly probability

$$\frac{i-1}{n-2}$$

Case 3: We select a pivot from the interval $[i, j]$ where $i < k < j$

Here we only have this occur if i or j is chosen to be a pivot, where the two possible subarrays worth noting are containing either 1) k and j or 2) k and i. This occurs once again with expected value:

$$\frac{2}{j-k+1}$$

This occurs when i and j are placed such that k is contained between them so that we have probability:

$$\frac{j-i-1}{n-2}$$

We cna then sum these probabilities to get a final value:

$$E(X_{i,j}) = \frac{2(n-j)}{(k-i+1)(n-2)} + \frac{2(i-1)}{(j-k+1)(n-2)} + \frac{2(j-i-1)}{(j-i+1)(n-2)}$$

3b) We determine, using a, that the expected runtime for all $i < j$ is $O(n)$. These sums are independent cases such that we can do the law of total probability for all 3 cases mentioned above:

$$E \sum_{i<j} = \sum_{i<j} E(X_{i,j})$$

Therefore this value is just the sum of all expected values, we therefore have:

$$\sum_{i=1}^{k-} \sum_{j=i+1}^{k-1} \frac{2}{k-i} + \sum_{i=k+1}^{n} \sum_{j=i+1}^{n} \frac{2}{j-k} + \sum_{i=1}^{k-1} \sum_{j=k+1}^{n} \frac{2}{j-i+1}$$

We then evaluate each of this subsums such that we can evaluate the whole expression's asymptotic runtime. For the first, we note that it is equivalent to:

$$\sum_{i=1}^{k-2} \frac{2(k-i-1)}{k-i+1} \leq \sum_{i=1}^{k-2} 2 = 2k-4$$

Of course, as this is a linear function of k, we expect a linear runtime bounded by $O(n)$.

For the second sum, we note that it is equivalently linear by symmetry, noting that we will achieve a similar linear value as we are concerned with k values below i and j now instead of above. For this reason, we expected a linear runtime bounded by $O(n)$.

For the third sub-sum we have:

$$\sum_{i=1}^{k-1}(j-i)\frac{2}{j-i+1}$$

We define $x = j - i$ as we need to consider when k is between these two indices and it is useful to have a variable to simplify this equation:

$$\sum_{i=1}^{k-1} x\frac{2}{x+1}$$

We therefore have a similar case as before such that we achieve:

$$\sum_{i=1}^{k-1} x\frac{2}{x+1} \leq \sum_{i=1}^{k-1} 2 = 2k-2$$

We therefore achieve a third sub-sum with linear runtime bounded by $O(n)$.

As we have three $O(n)$ runtimes, we expected an overall runtime of $O(n)$.

Problem 4:

We first define the probability that a specific bucket contains k items, noting that any item has a $\frac{1}{n}$ chance of being placed in a given bucket. We choose any k items and the rest to not be in this one bucket such that we have:

$$P(B=k) = \binom{n}{k}\frac{1}{n^k}(1-\frac{1}{n})^{n-k} \leq \binom{n}{k}\frac{1}{n^k}$$

Given this, we can easily calculate $P(B \geq k)$by simply enumerating this for cases higher than k, using stirling's approximation to greatly simplify this value:

$$P(B \geq k) \leq \sum_{i=k}^{n}\binom{n}{i}\frac{1}{n^i} \leq \sum_{i=k}^{n}(\frac{en}{i})^i\frac{1}{n^i} = \sum_{i=k}^{n}\frac{e^i}{i^i}$$

We note that this sum is strickly increasing, allowing us to bound it by some upper value. We note that if we include the largest value in the sum n times summed rather than the items themselves, we should achieve a strict bound on this interval. As the largest item of the sum of $n-k+1$ items is $\frac{e^k}{k^k}$, we have:

$$P(B \geq k) \leq n\frac{e^k}{k^k}$$

We now use the property of the union bond to determine given this probability of a given bucket containing at least k items, what is the probability there is a bucket containing at least k items. Clearly this probability of some bucket is less than m times $P(B \geq k)$ where m is the number of buckets, as this reduces the sample size from other buckets can pull and is therefore not independent. We then have, defining $\phi(n) = 2log(n) + k - klog(k)$:

$$P(aBucket) \leq m * P(B \geq k) \leq n^2\frac{e^k}{i^k} = e^{\phi(n)}(n = m)$$

We desire this expression to be strictly decreasing such that as $n \rightarrow \infty$ our probability $\rightarrow 0$. As our exponent is this function $\phi(n)$, we expect this to dominate such that we determine what dominants within that function. We set, as defined in the problem the table being at least of this size, $k = \frac{Clog(n)}{log(log(n))}$ such taht we find:

$$\phi(n) = (2-C)log(n) + \frac{C(1-log(C))log(n)}{log(log(n))} + \frac{Clog(n)log(log(log(n)))}{log(log(n))}$$

We note that this nested logs are going to increase very slowly and are therefore dominated by the first term as C increases. We note that for $C > 2$, $(2-C)log(n)$ should strictly decrease both quickly and unboundedly such that no linked list in the entire table has size larger than $O(\frac{log(n)}{log(log(n))})$ with probability 99%, as this expression has a probability $\leq 1\%$.