Comp 480/580: Mid-Term Exam
Total Time: 1 hour (+15 minutes slack) **Total Points:** 50

Name: ______________________________ NetID:________

1 Trivia (True or False) 15 points (3 points each)
Make sure to provide short explanation.

1. Given a set S, our goal is to be able to compress S. We want to answer whether any given x is an element of S or not. We can use bloom filters on S to compress it for this task.

2a. Given a random variable X, we know its mean and variance $E(X)$ and $Var(X)$. Given another random variable Y, which is possibly correlated with X. We also know its mean and variance $E(Y)$ and $Var(Y)$. The above information is sufficient to compute the mean of X + Y.

2b. The information in 2a is sufficient to compute the variance of X + Y

3. We have two random variables X and Y (and they are not constants). It is possible to have situations where $E(\frac{X}{Y}) = \frac{E(X)}{E(Y)}$.

4. Count-min sketch returns an underestimate.

5. Given any LSH function h, where $Pr(h(x) = h(y)) = p_{xy}$ for any given x and y. We can always construct another LSH g with probability $Pr(g(x) = g(y)) = p_{xy}^3$

2 Estimation and Minwise Hashing 10 points

Given two sets S_1 and S_2, minwise hashing given by h guarantees that the probability of hash agreement is Jaccard similarity J, i.e.,

$$Pr(h(S_1) = h(S_2)) = \frac{|S_1 \cap S_2|}{|S_2 \cup S_2|} = J$$

In the class we showed that if we generate K independent minwise hashes of S_1 and S_2, then simply counting the number of hash collisions, call it N_c, out of K gives us an estimator of J given by $\frac{N_c}{K}$. In addition, let us say that the variance of this counting estimator is given by $V = \frac{J(1-J)}{K}$

Questions:

1. Identify the random variable in the above problem. 1 point.

2. Given the values of the sizes of the sets, i.e., values of $|S_1|$ and $|S_2|$ (assume these sizes to be known constants). Can we get an estimator for $|S_1 - S_2|$ (Number of elements in S_1 but not in S_2)? (only in terms of N_c, K, $|S_1|$, and $|S_2|$). 5 points

3. What can we say about the variance of the estimator given in 2 above? (in terms of J, K, $|S_1|$, and $|S_2|$) 4 points

Known property: $Variance(aX) = a^2 Variance(X)$, where a is constant and X is any random variable.

3 Count-Min Sketch with weights 10 points

(Please see cheat-sheet for reference) Let us assume that every item i has a weight associated with it, call it v_i. We modify the count-min sketch algorithm as follows:

- **Addition:** Instead of "Each counter is initialized with zero, and every update c_i^t (to item i at time t) is added for all d rows to counters $M(j, h_j(i))$, where $j = \{1, 2, ..., d\}$", we update by multiplying every update of item i with v_i, i.e., for update c_i^t we update the counters $M(j, h_j(i))$, where $j = \{1, 2, ..., d\}$ with $v_i \times c_i^t$
- **Querying:** While estimating the count for element i, we use roughly the same estimation as count-min sketch, except that we divide the estimate by v_i. Essentially, a query for the count of item i reports the minimum of the corresponding d counters (after division by v_i) i.e., $\min\limits_{j \in \{1,2,...,d\}} \frac{M(j,h_j(i))}{v_i}$

Note, this is generalization of count-min sketch. We recover count-min sketch if we put $v_i = 1$ for all i
Questions:

1. Consider the element j that has the largest weight, i.e., $v_j = max_i v_i$. What can we say (more error or less error) about the estimate of this largest element compared with the estimate of usual count-min sketch (where all $v_i = 1$) 4 points.
Questions: Maybe just consider what happens with only 1 hash function (i.e. d =1).

2. Consider the element j that has the smallest weight, i.e., $v_j = min_i v_i$. What can we say (in one line) about its estimate of this element compared with the estimate of usual count-min sketch (where all $v_i = 1$) 4 points

3. In light of 1. and 2. above, can you describe (in few words) what do you think is going on. What is the significance of weights here? 2 points

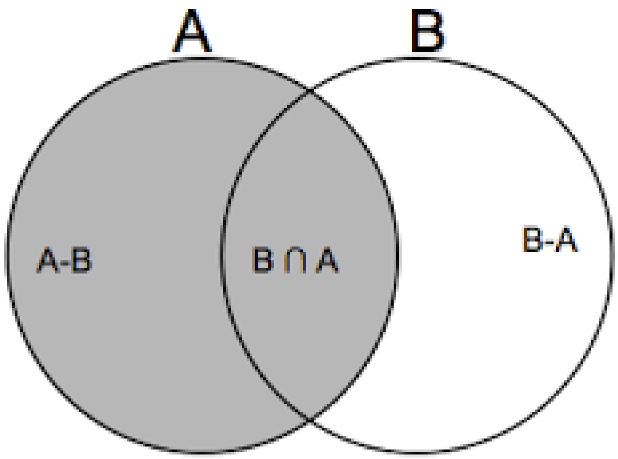

Figure 1: Illustration of Set and different regions.

4 Minwise Hashing (Advanced) 15 points

The proof of the fact

$$Pr(Minhash(S_1) = Minhash(S_2)) = \frac{|S_1 \cap S_2|}{|S_2 \cup S_2|} = J,$$

follows from the following arguments.

1. Given any set S, Apply hash mapping h (assuming perfect hashing i.e. for two different element $x \neq y$, implies $h(x) \neq h(y)$), to every element of S. The element that gets the minimum value (call it $Minhash(S)$) is a perfectly random draw of an element from S
2. Now consider set $S_1 \cup S_2$. Apply hash mapping h to this union of two sets. Thus, we expect $Minhash(S_1 \cup S_2)$ to give a random draw from $S_1 \cup S_2$.
3. We have $Minhash(S_1) = Minhash(S_2) = Minhash(S_1 \cup S_2)$, if an only if the minimum of $h(S_1 \cup S_2)$ comes from a token belonging to $S_1 \cap S_2$. Only when both the minimum values coming from the sets $h(S_1)$ and $h(S_2)$ will be equal. Essentially, the collision event indicates a random draw from $S_1 \cap S_2$. (Venn diagram may help visualise)
4. Thus, the collision event is a special event where $Minhash(S_1 \cup S_2)$ satisfies $Minhash(S_1) = Minhash(S_2) = Minhash(S_1 \cup S_2)$. The event has $|S_1 \cap S_2|$ choices. The total choices for $Minhash(S_1 \cup S_2)$ is $|S_1 \cup S_2|$ (any random draw irrespective of the collision).

A good idea is to look at standard Venn diagram in Figure 1.

Questions:

1. Consider the case $Minhash(S_1) \neq Minhash(S_2)$. What is the probability of this event? 2 points.

2. Can we come up with an expression of $Pr(Minhash(S_1) > Minhash(S_2))$, in terms of $|S_1|$, $|S_2|$, $|S_1 \cup S_2|$, and $|S_1 \cap S_2|$? Can you instead use this to estimate Jaccard similarity J (Generate K independent minwise hashes of S_1 and S_2 and .... (similar to Problem 2 define the estimator)) 8 points

3. Lets say we have three sets. S_1, S_2, and S_3. What is the expression for $Pr(Minhash(S_1) = Minhash(S_2) = Minhash(S_3))$ (in terms of set definitions). Can you generalize the expressions if we have instead n sets S_1, S_2, ..., S_n and we are looking for $Pr(Minhash(S_1) = Minhash(S_2) = ... = Minhash(S_n))$ (**Hint**: Draw a venn diagram with three sets). 5 points

4 (**Keep thinking, No points**) Which estimator for J is better? The one derived in Problem 2 previously or derived in the second part of this problem.

5 Cheat Sheet

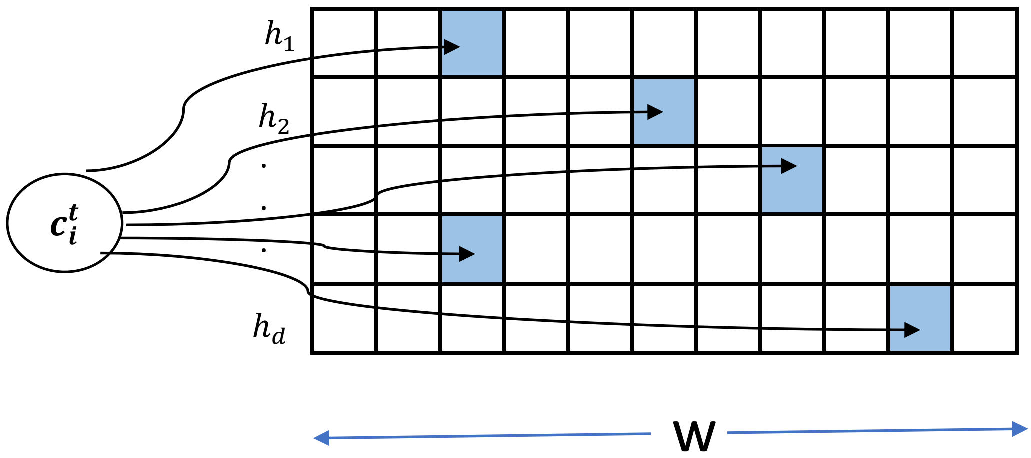

Figure 2: Count-Min Sketch Data Structure

5.1 Notations

The set of all items will be denoted by $I = \{1, 2, ..., N\}$. Time starts from $t = 0$, and the current time will be denoted by $T > 0$. The total increment to item i during time instance (or interval) t will be denoted by c_i^t, which is an aggregation of many streaming updates arriving at t. The total count of an item i will be given by $c_i = \sum_t c_i^t$

5.2 Count-Min Sketch (CMS)

The *Count-Min Sketch* (CMS) [?] algorithm is a generalization of Bloom filters [?] that is a widely popular in practice for estimating counts of items over data streams. CMS is a data structure with a two-dimensional array of counter cells M of width w and depth d, shown in Figure 2.

It is accessed via d pairwise-independent universal hash functions $h_1, h_2, ..., h_d : \{1, 2, ..., N\} \mapsto \{1, 2, ..., w\}$.

Each counter is initialized with zero, and every update c_i^t (to item i at time t) is added for all d rows to counters $M(j, h_j(i))$, where $j = \{1, 2, ..., d\}$. A query for the count of item i reports the minimum of the corresponding d counters i.e., $\min_{j \in \{1,2,...,d\}} M(j, h_j(i))$. This simple algorithm has strong error guarantees and is very accurate for estimating heavy hitters over entire streams, with its simplicity and easy parallelization contributing to its wide adoption.

5.3 Basic Analysis

Let us look at just one row, i.e., at h_1. For an element i, we can write down the value of $M(i, h_1(i))$ as

$$M(i, h_1(i)) = c_i + \sum_{j \neq i} \mathbf{1}_{h_1(j)=h_1(i)} \times c_j,$$

here $\mathbf{1}_{h_1(j)=h_1(i)}$ is an indicator vector for the event $h_1(j) = h_1(i)$. In expectation, we have

$$E[M(i, h_1(i))] = c_i + \sum_{j \neq i} E[\mathbf{1}_{h_1(j)=h_1(i)}] \times c_j,$$

which boils down to

$$E[M(i, h_1(i))] = c_i + \sum_{j \neq i} \frac{c_j}{w},$$

as $E[\mathbf{1}_{h_1(j)=h_1(i)}] = \frac{1}{w}$. Since the estimator for just 1 hash function is same as the value of $M(i, h_1(i))$ (minimum over 1 array), the expected error, for the count of i, in each row is given by $\sum_{j \neq i} \frac{c_j}{w}$