الگوریتم‌های خلاصه‌سازی برای مه‌داده — نیم‌سال اول سال تحصیلی ۱۳۹۹_۱۴۰۰

تمرین سری دوم

زمان پایان: ۲۹/۷/۹۹

۱ وزین‌ها با افزایش ℓ_1

In class we considered *turnstile streams* where a vector $x \in \mathbb{R}^n$ receives updates of the form "add Δ to x_i" in a stream. As mentioned, *insertion-only streams* are a special case of turnstile streams where $\Delta = 1$ always (so we can just imagine the stream is a sequence $i_1 i_2 \cdots i_L$ of integers in $[n]$). Also, recall in the point query problem that after several updates we are asked a query on i for some $i \in [n]$ and must output a value in $[x_i - (1/k)\|x\|_1, x_i + (1/k)\|x\|_1]$. Consider the algorithm CounterPointQuery below.

Algorithm COUNTERPOINTQUERY:

1. Initialize B counter/index pairs $(i_1, C_i), \ldots, (i_B, C_B)$ all to $(0, 0)$
2. **update(i):** **if** $i = i_j$ for some $j \in [B]$, **then** increment C_j
   else if none of the $i_j = i$ but some $C_j = 0$, then set $i_j = i, C_j = 1$
   else decrement every C_j by 1
3. **query(i):** **if** $i = i_j$ for some $j \in [B]$, then output C_j
   else output 0

(a) (10 points) Give an upper bound on what B needs to be to ensure that query(i) always outputs a value in $[x_i - (1/k)\|x\|_1, x_i + (1/k)\|x\|_1]$.

(b) (5 points) One can store the (i_j, C_j) pairs in an array so that they consume B space and updates take time $O(B)$ (since finding whether some $i_j = i$ or decrementing all counters would take $O(B)$ time). Devise a data structure taking $O(B)$ space to store the (i_j, C_j) pairs so that the **update(i)** operation above can be implemented in $O(1)$ time. Your data structure should probably use hashing, and the update time will be $O(1)$ *expected* time. Assume that integers in the range $1, \ldots, \max\{n, L\}$ can be stored in one unit of space, and that a computer can perform basic arithmetic operations on integers of this size in constant time.

Challenge problem (no credit): Suppose in part (a) you want to have error satisfying the tail guarantee, i.e. additive error $\pm(1/k)\|x_{tail(k)}\|_1$ (see Remark 4.1.1 of the lecture notes). Then what does B need to be?

۲ خلاصه‌سازی AMS با حافظه کمتر

Recall the AMS sketch from class for F_2 moment estimation: a random $m \times n$ matrix Π with entries $\pm 1/\sqrt{m}$ is drawn for $m = O(1/\varepsilon^2)$, and $\|x\|_2^2$ is estimated as $\|\Pi x\|_2^2$. Then with at least 2/3 probability,

$$(1-\varepsilon)\|x\|_2^2 \leq \|\Pi x\|_2^2 \leq (1+\varepsilon)\|x\|_2^2. \tag{1}$$

(a) (5 points) Imagine picking Π differently: for each $i \in \{1, \ldots, n\}$ we pick a uniformly random number $h_i \in \{1, \ldots, m\}$. We then set $\Pi_{h_i,i} = \pm 1$ for each $i \in \{1, \ldots, n\}$ (the sign is chosen uniformly at random from $\{-1, 1\}$), and all other entries of Π are set to 0. This Π has the advantage that in turnstile streams, we can process updates in constant time. Show that using this Π still satisfies the conditions of Equation 1 with 2/3 probability for $m = O(1/\varepsilon^2)$.

(b) (5 points) Show that the matrix Π from Problem 3(a) can be specified using $O(\log n)$ bits such that Equation 1 still holds with at least 2/3 probability, and so that given any $i \in \{1, \ldots, n\}$, $\Pi_{h_i,i}$ and h_i can both be calculated in constant time. You may assume that standard machine word operations take constant time (arithmetic, mod, bitwise operations, and bitshifts). **Hint:** Consider a hash function that does some arithmetic modulo a prime p for some choice of p.

۳ تقریب مختلطی از $F_۲$

In class we saw a particular way of producing estimates of frequency moments $F_k = \sum_{i=1}^{n} f_i^k$ and we briefly explored whether different estimators are possible. In this problem, you will see how one can use the field of complex numbers to achieve this. Let $R_k = \{x \in \mathbb{C} \,|\, x^k = 1\}$. be the set of k-roots of unity. For simplicity we will focus on the case of $k = 3$. The proposed basic estimator works as follows:

1. For each $i \in [m]$ we pick independently a uniform random element $x_i \in R_3$.
2. We form the random variable $Z = \sum_{i=1}^{n} f_i$, by adding x_i to Z each time we come across element $i \in [m]$.
3. We estimate F_3 as $Re\ Z^3$.

One can think of the mapping $i \to x_i$ as hash function, that instead of mapping to the 2-roots of unity $\{1, +1\}$ (in the original AMS scheme) maps to the 3-roots. You will analyze properties of this estimator:

(a) [10 points] Show that for any element $i \in [m]$, $\mathbb{E}[x_i^j] = \mathbb{E}[\bar{x}_i^j] = \begin{cases} 0 & \text{if } 1 \leq j < 3 \\ 1 & \text{if } j = 3 \end{cases}$.

$\bar{x}$ represents conjugate of x.

(b) [10 points] Show that $\mathbb{E}[Re\ Z^3] = F_3$. Hint: compute first $\mathbb{E}[Z^3]$.

(c) [20 points] Show that $Var[Re\ Z^3] = O(F_3^2)$. Hint: use the multinomial expansion.