الگوریتم‌های خلاصه‌سازی برای مه‌داده — نیم‌سال اول سال تحصیلی ۱۳۹۹ـ۱۴۰۰

تمرین سری اول

زمان پایان: ۲۹/۷/۹۹

۱ مثال‌هایی ساختی از توابع درهم‌ساز ۲ـطرفه

Let X and Y be finite sets and let Y^X denote the set of all functions from X to Y. We will think of these functions as “hash” functions. [The term “hash function” has no formal meaning; strictly speaking, one should say “family of hash functions” or “hash family” as we do here.] A family $H \subseteq Y^X$ is said to be 2-universal if the following property holds, with $h \in_R H$ picked uniformly at random:

$$\forall x, x' \in X \quad \forall y, y' \in Y \left(x \neq x' \Rightarrow \mathop{\mathbb{P}}_{h}[h(x) = y \& h(x') = y'] = \frac{1}{|Y|^2} \right)$$

We shall give two examples of 2-universal hash families from the set $X = \{0,1\}^n$ to the set $Y = \{0,1\}^k$ (with $k \leq n$).

1. Treat the elements of X and Y as column vectors with 0/1 entries. For a matrix $A \in \{0,1\}^{k \times n}$ and vector $b \in \{0,1\}^k$, define the function $h_{A,b} : X \to Y$ by $h_{A,b}(x) = Ax + b$, where all additions and multiplications are performed mod 2.
   Prove that the family of functions $H = \{h_{A,b} : A \in \{0,1\}^{k \times n}, b \in \{0,1\}^k\}$ is 2-universal.

2. Identify X with the finite field $GF(2^n)$. For elements $a, b \in X$, define the function $g_{a,b} : X \to Y$ as follows:

   $$\begin{aligned} g_{a,b}(x) = & \quad \text{rightmost } k \text{ bits of } f_{a,b}(x), \text{ where} \\ f_{a,b}(x) = & \quad ax + b, \text{with addition and multiplication performed in } GF(2^n). \end{aligned}$$

   Prove that the family of functions $G = \{g_{a,b} : a, b \in GF(2^n)\}$ is 2-universal. Is the family G better or worse than H in any sense? Why? [Note: If you are unfamiliar with finite fields, we should discuss this topic outside of class.]

نکته: تمرین‌های زیر قسمتی از تمرین‌های درس سال ۲۰۲۰ است و اشاره‌های به قضیه‌ها و جلسات،
اشاره به قضیه‌ها و جلسات در جزوه‌های درس در آدرس زیر است.

https://www.sketchingbigdata.org/fall20/lec/notes.pdf

۲ ماتریس‌های غیرمتجانس

A matrix $\Pi \in \mathbb{R}^{m \times n}$ is ϵ-*incoherent* if

1. For each column Π^i, $\|\Pi^i\|_2 = 1$.
2. For each $i \neq j$, $|\langle \Pi^i, \Pi^j \rangle| < \epsilon$.

In the lecture notes, Theorem 3.1.4 implies that for any $n > 1$ and $\epsilon \in (0, 1)$, there exists a code $\mathcal{C}$ with n codewords, alphabet size $q = O(1/\epsilon)$, block length $\ell = O(\epsilon^{-1} \log n)$, and relative distance $1-\epsilon$. This leads to an ϵ-incoherent matrix via the construction in Figure 1, where the columns of Π are in correspondence with codewords in $\mathcal{C}$.

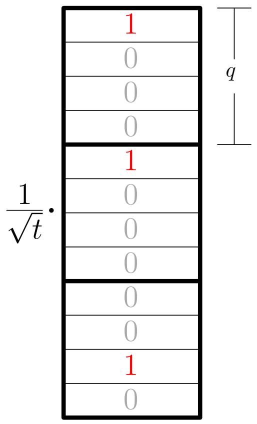

Figure 1: Each codeword gives one column of the incoherent matrix. Here $q = 4, t = 3$ and the codeword is $C_i = (1, 1, 3)$. The vector is $m = qt$ dimensional with the coordinates broken up into t blocks each of size q. A 1 is placed in the jth position in the location specified by $(C_i)_j$. The entire vector is normalized by $1/\sqrt{\ell}$ to have unit norm.

(a) (2 points) Consider the construction in Figure 1 to convert a code with n codewords, block length ℓ, alphabet size q, and relative distance ρ into a matrix $\Pi \in \mathbb{R}^{m \times n}$. For what ϵ is it ϵ-incoherent? How many rows m does it have? Your answers should be in terms of the code parameters.

It turns out one can achieve a better m than the code of Theorem 3.1.4 when ϵ is sufficiently small. Suppose the alphabet size q is a prime power and consider the finite field $\mathbb{F}_q$. Consider all polynomials $p_1, \ldots, p_N \in \mathbb{F}_q[x]$ of degree at most d where $N = q^{d+1}$. Define the *Reed-Solomon code* $C_1, \ldots, C_N$ as follows: $\ell = q$ where the jth entry of C_i is the evaluation of p_i on the jth element of $\mathbb{F}_q$ (so C_i is the evaluation table of p_i).

(b) (5 points) If we still want to have at least n codewords, we need $N \geq n$. Show how to choose d, q so $N \geq n$ and the relative distance is $1 - \epsilon$, and show what this gives (in big-Oh notation) for the number m of rows of the incoherent matrix Π we obtain.

(c) (3 points) How small does ϵ need to be as a function of n for the codes from part (b) to give smaller m than the code from Theorem 3.1.4? If the turning point is ϵ_T, you should provide an answer ϵ'_T such that $\log(1/\epsilon'_T) = \Theta(\log(1/\epsilon_T))$.

(d) (5 points) Suppose one has an ϵ-incoherent matrix Π. Show how ℓ_1-point queries to x being updated in the turnstile streaming model can be answered solely given $y = \Pi x$. (Note then that part (b) implies a *deterministic* ℓ_1-point query algorithm with low space in turnstile streams.) **Hint:** it may help to remember that $\Pi x = \sum_i x_i \Pi^i$.

OPEN PROBLEM: It is known that any ϵ-incoherent matrix with n columns must have $m = \Omega(\min\{n, \epsilon^{-2}(\log n)/\log(1/\varepsilon)\})$ [**?**, Section 9]. Can the gap between upper and lower bounds be closed? It is conceivable a better upper bound could be achieved by discovering a better code construction.

۳ شمارش اعداد متمایز با حذف

In Lecture 2 we showed how to estimate the number of distinct elements in a stream in $poly(\varepsilon^{-1} \lg n)$ bits of space with 2/3 success probability, where all integers in the stream are in $[n]$. In Lecture 3, we gave a different algorithm based on geometric sampling. Recall in the turnstile model, the distinct elements problem asks us to estimate $\|x\|_0 := |\{i : x_i \neq 0\}|$ when all updates have $\Delta = +1$. What if the updates in the stream are allowed to have $\Delta \in \{-1, 1\}$ though? Show how to alter the geometric sampling algorithm from Section 2.2.3 of the lecture notes to also handle such negative updates. What is the space complexity of your solution? Any solution using space $poly(\varepsilon^{-1} \lg(nL))$ for this modified problem, where L is the length of the stream, will receive full credit.

۴ مقایسه الگوریتم میانگین‌گیری و میانه‌گیری

فرض کنید الگوریتم A وجود دارد که خروجی آن یک متغیر تصادفی است از توزیع D با میانگین μ واریانس $\sigma^۲$. حال می‌خواهیم با استفاده از n بار اجرای A تخمین‌گری برای مقدار μ تولید کنیم به صورتی که با احتمال زیاد داشته باشیم $\mu - \epsilon\mu \leq \tilde{\mu} \leq \mu + \epsilon\mu$ که $\tilde{\mu}$ تخمین‌گر ماست.

فرض کنید مقدار ϵ به اندازه کافی بزرگ است مثلا داریم $\epsilon = ۲\sigma/\mu$.

الف) روش میانگین: در این روش میانگین n خروجی n بار اجرای الگوریتم به عنوان پاسخ در نظر گرفته می‌شود. تخمین‌گر این الگوریتم را $\tilde{\mu}_A$ می‌نامیم. نشان دهید $P[|\tilde{\mu}_A - \mu| \geq \epsilon\mu] \leq O(۱/n)$

ب) فرض کنید الگوریتم A با احتمال ثابت $\alpha > ۰$ جوابی که تولید می‌کند در بازه $[a, b]$ است. نشان‌دهید میانه خروجی‌های تولید شده از t بار اجرای الگوریتم A با احتمال $e^{-\Omega(n)}$ در بازه $[a, b]$ است.

ج) روش میانه: در این روش میانه n خروجی n بار اجرای الگوریتم به عنوان پاسخ در نظر گرفته می‌شود. تخمین‌گر این الگوریتم را $\tilde{\mu}_M$ می‌نامیم. نشان دهید $P[|\tilde{\mu}_M - \mu| \geq \epsilon\mu] \leq e^{-\Omega(n)}$