Lecture 7: 频繁元素：Frequent Items

Lecturer: 彭攀 *Scribe:* 徐怡，申冉冉

本节课我们将继续学习在数据流模型下的算法设计。特别地，我们将学习**估计频繁元素**的算法。这个问题的应用场景有很多，比如我们想抵御 DDoS 攻击，可以分析截获的网络数据包（packet）的 IP 地址，如果有一些 IP 地址出现了很多次，我们就可以拦截对应的包。

1 众数

问题: 考虑一串以数据流形式输入的整数 $i_1, \ldots, i_m \in [n] = \{1, \ldots, n\}$，要求输出数据流中的众数（我们这节课定义为出现次数大于 $m/2$ 的元素），使用空间越少越好。

以下是一个简单的算法，Misra-Gries 算法：

1. 维护两个数，ID 和计数器 c。初始时刻设 $c = 0$，ID$=\perp$（空白）
2. 当某个元素 x 到来时，根据以下规则更新维护的两个数：
   - 如果 ID$= x$，则 $c \leftarrow c + 1$
   - 否则，如果计数器 $c = 0$，则设置 ID$\leftarrow x$，$c \leftarrow 1$
   - 否则，$c \leftarrow c - 1$
3. 当数据流结束时，输出 ID 和计数器 c

首先我们用一个例子演示一下这个算法。

时刻	初始值	1	2	3	4	5	6	7	8	9	10	11
数据流		1	3	10	3	1	3	10	3	3	3	3
ID	$\perp$	1	1	10	10	1	1	10	10	3	3	3
c	0	1	0	1	0	1	0	1	0	1	2	3

表 1: Misra-Gries 算法的演示-众数

按照上表的演示，算法最终会输出 ID$= 3$，而 3 的确是这个数据流的众数。下面我们来分析这个算法。

断言 1. 如果数据流 $i_1, \ldots, i_m$ 中存在众数 i，则 Misra-Gries 算法会输出 ID$= i$ 和计数器 $c \geq f_i - m/2$。其中，f_i 表示 i 这个数在数据流中出现的次数，或者说是 i 的**频率**。这个算法只需要 $O(\log n + \log m)$ 个比特。

注意 2. 如果数据流里没有众数，会输出错误的数。需要第二遍读取数据流来检验算法的输出。

2 频繁元素

下面我们考虑更一般的场景。

问题: 考虑一串以数据流形式输入的整数 $i_1, \ldots, i_m \in [n] = \{1, \ldots, n\}$，给定一个整数 $k \geq 1$。要求输出数据流中所有满足 $f_i > m/(k+1)$ 的元素 $i \in [n]$（即出现次数大于 $m/(k+1)$ 的元素）。使用空间越少越好。

众数问题是频繁元素问题的一个特例，即 $k = 1$ 的情形。

思路: 首先观察到，满足 $f_i > m/(k+1)$ 的元素最多有 k 个。这是因为，满足上式的元素，其频率之和应该不超过 m。记这些元素的个数为 x，即：

$$m \geq \sum_{i: f_i > \frac{m}{k+1}} f_i > \sum_{i: f_i > \frac{m}{k+1}} \frac{m}{k+1} = x \cdot \frac{m}{k+1}$$

所以 $x < k+1$，即 $x \leq k$。因此，算法的空间复杂度应该和 k 成正比。我们可以用改进的 Misra-Gries 算法解决这个问题：

1. 用数组 A 来维护 k 个 ID 和 k 个计数器 c。初始时刻把所有计数器设为 $c = 0$，所有 ID 设为 $\perp$（空白）
2. 当某个元素 x 到来时，根据以下规则更新维护的数组 A：
   - 如果 x 被某个 ID 保存了，就把对应的计数器增加 1，即 $c \leftarrow c + 1$
   - 否则，如果存在某个计数器 $c = 0$，则把对应的 ID 设为 x，并把这个计数器设为 1
   - 否则，A 中的每个计数器都减 1。这一个操作被称为**全部递减**（decrement-all）
3. 当数据流结束时，输出 A 中所有的 ID 和计数器

我们用一个例子演示一下这个算法。

时刻	初始值	1	2	3	4	5	6	7	8	9	10	11
数据流		1	3	10	3	1	3	10	3	3	3	3
ID_1	$\perp$	1	1	1	1	1	1	1	1	1	1	1
c_1	0	1	1	0	0	1	1	0	0	0	0	0
ID_2	$\perp$	$\perp$	3	3	3	3	3	3	3	3	3	3
c_2	0	0	1	0	1	1	2	1	2	3	4	5

表 2: Misra-Gries 算法的演示-频繁元素

断言 3. 数据流中所有满足 $f_i > m/(k+1)$ 的元素，在数据流结束时，都会被保存在数组 A 里。这个算法只需要 $O(k(\log n + \log m))$ 个比特。

注意 4. 如果数据流里没有 $f_i > m/(k+1)$ 的元素，会输出错误的数。需要第二遍读取数据流来检验算法的输出。

对断言 3的证明. 当数据流结束时，对于被记录在数组 A 里的元素 i，记其对应的计数器的值为 $\hat{f}_i$，表示该元素的频率的估算值。对于没有被记录在 A 里的元素，设 $\hat{f}_i = 0$。这样一来，对于每一个 $i \in [n]$，$\hat{f}_i$ 都有定义。

我们只需证：对于所有的 $i \in [n]$，有 $f_i - \frac{m}{k+1} \le \hat{f}_i \le f_i$。如果这个命题得证，那么只要 $f_i > \frac{m}{k+1}$，就有 $\hat{f}_i \ge f_i - \frac{m}{k+1} > 0$，即元素 i 一定会被记录在数组 A 里，断言 3就得证了。

下面我们来证明：对于所有的 $i \in [n]$，有 $f_i - \frac{m}{k+1} \le \hat{f}_i \le f_i$。

首先，右边的不等式是平凡的：每当数据流的一个新元素出现时，算法有可能把该元素存入 A，并把对应计数器加 1，也有可能不加 1，因此计数器的值总是不会超过它真实的频率，即 $\hat{f}_i \le f_i$。

对于左边的不等式，考虑算法的另一种视角：假设我们维护了一个长度为 n 的数组，记为 C，并把数组里每个元素的初始值都设为 0。每个时刻，我们都保证数组里最多有 k 个非零元素。在这个等效的视角下，算法的执行流程变为：

在第 j 个时刻，数据流中到来一个新的元素 e_j 时，根据以下规则更新数组 C：

- 如果 $C[e_j] > 0$，就把 $C[e_j]$ 加 1
- 否则，如果 C 里面的非零元素的个数小于 k，就令 $C[e_j] \leftarrow 1$
- 否则，把 C 中所有非零元素减 1（**全部递减**）

上述算法和前面的 Misra-Gries 算法是等价的，但我们不能花费 $O(n)$ 的空间来存储数组，因此 Misra-Gries 算法进一步简化，只保留非零元素。

对任意一个元素 i，考虑 $\alpha = f_i - \hat{f}_i$，并且假设 f_i 的初始值也是 0，随着数据流的到来和算法的执行，f_i 和 $\hat{f}_i$ 都会发生变化。我们观察 α 的变化情况。

- 如果在第 j 个时刻，到来的元素是 $e_j = i$
  - 如果此时 $C[i] > 0$，或者 C 的非零元素个数小于 k，则 $C[i]$ 加 1。那么 $\hat{f}_i$ 和 f_i 都会加 1，α 保持不变
  - 否则，对 C 的所有非零元素执行**全部递减**操作，$C[i]$ 不变（因为 $C[i] = 0$）。但 f_i 加 1，所以 α 加 1
- 如果在第 j 个时刻，到来的元素 e_j 不是 i，α 有可能不变，也有可能加 1（加 1 一定发生在执行**全部递减**的时候；但执行**全部递减**不一定导致 α 加 1）

因此，在算法结束的时候，$\alpha \le$ 执行**全部递减**的次数，记这个次数为 ℓ。

下面我们来证明：$\ell \le \frac{m}{k+1}$。

考虑两个求和，$(1) = \sum_{i=1}^{n} \hat{f}_i$，$(2) = \sum_{i=1}^{n} f_i$，它们是随着时间变化的量。当某个时刻执行**全部递减**操作时，(1) 减少 k，(2) 增加 1，则 $(2)-(1)$ 增加了 $k+1$。如果没有执行**全部递减**操作，则 $(2)-(1)$ 不变。

当算法结束时，$(2) = m$，$(1) \ge 0$，因此 $(2)-(1) \le m$。由于**全部递减**执行了 ℓ 次，每次执行时 $(2)-(1)$ 增加了 $k+1$，则：$\ell \cdot (k+1) \le m$，因此 $\ell \le \frac{m}{k+1}$。

因此，对于任意一个元素 $i \in [n]$，$\alpha = f_1 - \hat{f}_i \le \ell \le \frac{m}{k+1}$，即 $f_i - \frac{m}{k+1} \le \hat{f}_i$。 ■

标注 5. Misra-Gries 算法的好处是，这是一个确定性的算法（即运行算法若干次，每次都能得到同样的结果），而且使用的空间很少。坏处是，不能处理数据流中存在删除元素的情况。

3 数据流的推广模型

前面提到的数据流模型，都是只允许插入操作，不允许删除操作。现在我们考虑更一般的数据流模型。

问题: 考虑一个向量/数组 $x \in \mathbb{R}$，初始时刻是一个全零向量。每一个时刻，数据流到来的元素 e_j 不再是单一的整数，而是一个元组，即 $e_j = (i_j, \Delta_j)$。其中，$i_j \in [n]$，$\Delta_j \in \mathbb{R}$。当 e_j 到来时，我们对向量 x 进行更新：$x_{i_j} \leftarrow x_{i_j} + \Delta_j$。$\Delta_j$ 可以是正数，也可以是负数。

根据 Δ_j 不同的取值情况，数据流模型有以下变种：

- 收银机模型（cash register model）：$\Delta_j > 0$；特殊情形下，$\Delta_j = 1$，即我们前面研究的数据流模型
- 旋转门模型（turnstile model）：Δ_j 可以是任意值
- 严格的旋转门模型（strict turnstile model）：Δ_j 可以是任意值，但 x 的每一项在任意时刻都必须大于 0

4 频繁元素-严格旋转门模型

我们接下来在严格的旋转门模型下，研究频繁元素问题的推广版本。首先考虑以下两个查询问题：

问题 6 ((k, ℓ_1)-点查询问题)**.**
输入：待查询的整数 $i \in [n]$
输出：$\tilde{x}_i$，满足 $\tilde{x}_i = x_i \pm \frac{1}{k} \cdot \|x\|_1$，即，$x_i - \frac{1}{k} \cdot \|x\|_1 \le \tilde{x}_i \le x_i + \frac{1}{k} \cdot \|x\|_1$

其中，$\|x\|_1 = |x_1| + |x_2| + \cdots + |x_n|$，是向量 $x = (x_1, x_2, \ldots, x_n)$ 的一范数。

问题 7 ((k, ℓ_1)-频繁元素查询问题)**.**
当查询时，返回一个集合 $L \subset [n]$，使得 $|L| = O(k)$，并且如果元素 $x_i > \frac{1}{k} \cdot \|x\|_1$，则 $i \in L$

问题 6的算法能用于解决问题 7，具体体现为以下引理：

引理 8. 假设存在一个解决 $(3k, \ell_1)$-点查询的算法 $\mathcal{A}$，犯错概率是 $\frac{\delta}{n}$，占用的空间是 s 个比特。那么存在一个解决 (k, ℓ_1)-频繁元素查询的算法 $\mathcal{A}'$，犯错概率是 δ，占用的空间是 $s + O(k \log n)$ 个比特。

对引理 *8*的证明. 有了算法 $\mathcal{A}$ 之后，我们这样设计算法 $\mathcal{A}'$：

1. 对每个 $i \in [n]$，调用 $\mathcal{A}(i)$，对 i 进行查询
2. 对于值最大的 $3k$ 个查询结果（即 $\tilde{x}_i$），保留它们的下标（即 $L \leftarrow L \cup i$）
3. 返回集合 L

根据 (k,ℓ_1)-点查询算法的定义，对任意整数 $i\in[n]$，有 $x_i-\frac{1}{3k}\cdot\|x\|_1\le\tilde{x}_i\le x_i+\frac{1}{3k}\cdot\|x\|_1$。根据 $\mathcal{A}$ 的犯错概率，这个事件发生的概率至少是 $1-\frac{\delta}{n}$。那么，对于所有 $i\in[n]$，查询 i，返回的 $\tilde{x}_i$ 都满足这个近似比的概率至少是 $1-\delta$。下面我们假设这个事件发生（即所有的 i 都满足该近似比），继续我们的分析。

一方面，当 $x_i>\frac{\|x\|_1}{k}$ 时，$\tilde{x}_i\ge x_i-\frac{\|x\|_1}{3k}>\frac{\|x\|_1}{k}-\frac{\|x\|_1}{3k}=\frac{2\|x\|_1}{3k}$，因此我们需要保留 $\tilde{x}_i>\frac{2\|x\|_1}{3k}$ 对应的 i。

另一方面，当 $x_i\le\frac{\|x\|_1}{3k}$ 时，$\tilde{x}_i\le x_i+\frac{\|x\|_1}{3k}\le\frac{\|x\|_1}{3k}+\frac{\|x\|_1}{3k}=\frac{2\|x\|_1}{3k}$。反之，如果 $\tilde{x}_i>\frac{2\|x\|_1}{3k}$，则一定有 $x_i>\frac{\|x\|_1}{3k}$。

注意到，满足 $x_i>\frac{\|x\|_1}{3k}$ 的整数 i 的个数，最多有 $3k$ 个。因此满足 $\tilde{x}_i>\frac{2\|x\|_1}{3k}$ 的整数 i 的个数也最多有 $3k$ 个。由于 $\mathcal{A}'$ 保留了 $\mathcal{A}$ 的返回值（即 $\tilde{x}_i$）中 $3k$ 个最大的值，那么所有满足 $\tilde{x}_i>\frac{2\|x\|_1}{3k}$ 对应的整数 i 都被存下来了。

■

4.1 CountMin Sketch-离线版本

下面我们给出一个解决 $(3k,\ell_1)$-点查询的算法 $\mathcal{A}$，叫做 CountMin Sketch。应用引理引理 8，就能得到解决 (k,ℓ_1)-频繁元素查询的算法 $\mathcal{A}'$ 了。

我们先看离线版本（数据存储在离线磁盘里，而不是以数据流的形式到来）下的 CountMin Sketch 算法。

思路: 记 $b_1,\dots,b_k\in[n]$ 为 k 个频繁元素。如果我们选取一个哈希函数 $h:[n]\to[c\cdot k]$，其中 c 是一个大于 1 的常数，那么 h 就会把 $b_1,\dots,b_k$ 映射到这 $c\cdot k$ 个篮子里。理想情况下，我们可以在 $b_1,\dots,b_k$ 被映射的篮子里存放它们的频率；但实际上哈希函数会有冲突（可能两个不同的元素 $i,j\in[n]$ 会被映射到同一个篮子，那它们的频率就混在一起了。我们可以用多个独立的哈希函数来改进这个问题。CountMin Sketch 算法描述如下：

1. 选 d 个独立的哈希函数，$h_1,\dots,h_d$
   - 每个哈希函数 h_ℓ 都是 2 方独立的（参见 Notes Lec6 不同元素个数-k 方独立哈希函数族），并且 $h_\ell:[n]\to w$，即把元素 $i\in[n]$ 映射到 w 个篮子里
2. 每个篮子存一个数（表示某个元素的频率）
   - 我们一共有 $d\cdot w$ 个篮子，存放 $d\cdot w$ 个数（初始值设为 0），构成了一个二维数组 $C_{d\times w}$。哈希函数决定了篮子（二维数组）的下标。比如 $h_\ell(i)=s$，就把 i 的频率存到 $C[\ell,s]$ 里
3. 令 $x\in\mathbb{R}^n$ 为给定的数组，每个 x_i 表示 i 的频率。对每个 $1\le\ell\le d$，$1\le s\le w$，定义 $C[\ell,s]=\sum_{i:h_\ell(i)=s}x_i$
   - 从这个定义可以看出，哈希函数有可能把两个不同的 $i,j\in[n]$ 映射到同一个 s，从而把它们的频率累加到同一个篮子 $C[\ell,s]$

4. 当查询整数 $i \in [n]$ 时，输出 $\tilde{x}_i = \min_{\ell=1,\dots,d} C[\ell, h_\ell(i)]$

二维数据 $C_{d\times w}$ 就是一个 sketch，所需要的空间大小为 $d \times w$。通过对 d, w 合理取值，可以使得 $d \times w \ll n$，即我们可以用很小的空间去总结很大的向量；在查询时，使用 sketch 可以快速给出回答。

4.2 CountMin Sketch-数据流版本

CountMin Sketch 数据流版本的算法描述如下：

1. 选 d 个 2-方独立（2-wise independent）的哈希函数 $h_1, \dots, h_d : [n] \to [w]$；
2. 初始化 $C[\ell, s] = 0$，其中 $1 \le \ell \le d, 1 \le s \le w$；
3. 对流中的每一个元素 $e_t = (i_t, \Delta_t)$：
   - 对每个 ℓ（$1 \le \ell \le d$），更新 $C[\ell, h_\ell(i_t)] = C[\ell, h_\ell(i_t)] + \Delta_t$；
4. 对每个 $i \in [n]$，令 $\tilde{x}_i = \min_{\ell=1,\dots,d} C[\ell, h_\ell(i)]$；
5. 查询 i 时，输出 $\tilde{x}_i$。

引理 9. *考虑严格旋转门模型（strict turnstile model），即在任意时刻都有 $x \ge 0$。令 $d = \Omega\left(\log \frac{1}{\delta}\right)$、$w > 2k$。对任意固定的 $i \in [n]$，有*

$$x_i \le \tilde{x}_i$$

和

$$\Pr\left[\tilde{x}_i \ge x_i + \frac{\|x\|_1}{k}\right] \le \delta.$$

注意：引理 9 中固定了 $i \in [n]$，而不是对所有的 $i \in [n]$ 都有上述式子成立。我们可以令 $\delta = \frac{1}{n^c}, c \ge 2$，针对 i 使用 union bound，即可得到对于所有的 $i \in [n]$，$\Pr\left[\tilde{x}_i < x_i + \frac{\|x\|_1}{k}\right] \ge 1 - n\delta = 1 - \frac{1}{n^{c-1}}$。

接下来我们首先分析在 $d = \Omega\left(\log \frac{1}{\delta}\right)$、$w > 2k$ 时，算法的空间复杂度；其次对引理 9 进行证明。

空间复杂度：

- 算法使用 d 个 2-方独立的哈希函数，需要 $d \cdot \log n = \Omega(\log \frac{1}{\delta} \cdot \log n)$ 比特；
- 算法维护一个大小为 $d \times w$ 的二维数据 $C_{d\times w}$；数组中 $C[\ell, s] \le \|x\|_1$；因此 $d \times w$ 的二维数据需要 $d \cdot w \cdot \log \|x\|_1 = \Omega\left(\log \frac{1}{\delta} \cdot k \cdot \log \|x\|_1\right)$ 比特。

现在我们对引理 9 进行证明。

Proof. 因为 $\tilde{x}_i = \min_{\ell=1,\dots,d} C[\ell, h_\ell(i)]$，对于每一个 $C[\ell, h_\ell(i)]$，这个格子中至少保存了 x_i，此外还有可能保存了 x_j，j 满足 $h_\ell(i) = h_\ell(j)$。因此 $\tilde{x} \ge x_i$。

固定一个 $i \in [n]$，固定一个 $\ell \in [d]$，令

$$Z_\ell = C[\ell, h_\ell(i)] = \sum_{j\in[n], h_\ell(j)=h_\ell(i)} x_j.$$

则有

$$\begin{aligned}
\mathbb{E}\left[Z_\ell\right] &= \mathbb{E}\left[\sum_{j\in[n], h_\ell(j)=h_\ell(i)} x_j\right] \\
&= x_i + \sum_{i'\in[n], i'\neq i, h_\ell(i')=h_\ell(i)} \mathbb{E}\left[x_{i'}\right] && \text{（固定了 } i \in [n]\text{）} \\
&= x_i + \sum_{i'\in[n], i'\neq i} x_{i'} \cdot \Pr\left[h_\ell(i') = h_\ell(i)\right] \\
&= x_i + \sum_{i'\in[n], i'\neq i} x_{i'} \cdot \frac{1}{w} && \text{（} h_\ell \text{ 是 2-方独立的哈希函数）} \\
&\le x_i + \frac{1}{w}\|x\|_1 \\
&< x_i + \frac{1}{2k}\|x\|_1 && \text{（} w > 2k \text{）}.
\end{aligned}$$

则 $\mathbb{E}\left[Z_\ell - x_i\right] < \frac{1}{2k}\|x\|_1$，由 Markove 不等式，有

$$\Pr\left[Z_\ell - x_i \ge \frac{\|x\|_1}{k}\right] \le \frac{\mathbb{E}\left[Z_\ell - x_i\right]}{\frac{\|x\|_1}{k}} < \frac{\frac{\|x\|_1}{2k}}{\frac{\|x\|_1}{k}} = \frac{1}{2}.$$

至此，我们证明了在固定一个 $\ell \in [d]$ 时，有 $\Pr\left[Z_\ell - x_i \ge \frac{\|x\|_1}{k}\right] < \frac{1}{2}$。因此，对于任意 $\ell \in [d]$，有

$$\Pr\left[C[\ell, h_\ell(i)] \ge x_i + \frac{\|x\|_1}{k}\right] = \Pr\left[C[\ell, h_\ell(i)] - x_i \ge \frac{\|x\|_1}{k}\right] = \Pr\left[Z_\ell - x_i \ge \frac{\|x\|_1}{k}\right] < \frac{1}{2^d} \le \delta.$$

综上，在 $d = \Omega\left(\log \frac{1}{\delta}\right)$、$w > 2k$ 时，对任意固定的 $i \in [n]$，有

$$\Pr\left[\tilde{x}_i \ge x_i + \frac{\|x\|_1}{k}\right] \le \delta.$$

■

4.3 Count Sketch-数据流版本

本节我们给出一个相比较 CountMin Sketch 误差更小（但使用的空间更多）的算法，即 Count Sketch。在给出算法之前，先回顾一个事实：

事实 10.

$$\|x\|_2 \le \|x\|_1 \le \sqrt{n} \cdot \|x\|_2.$$

Proof. 分别证明两个不等式：

- $\|x\|_2^2 = \sum_{i=1}^n x_i^2 \le (\sum_{i=1}^n |x_i|)^2 = \|x\|_1^2$，因此 $\|x\|_2 \le \|x\|_1$；
- 由柯西不等式，有 $\|x\|_1^2 = (\sum_{i=1}^n |x_i|)^2 \le \sum_{i=1}^n x_i^2 \cdot \sum_{i=1}^n 1^2 = n \cdot \|x\|_2^2$，因此 $\|x\|_1 \le \sqrt{n} \cdot \|x\|_2$。

■

注意：在一些情况下，有 $\|x\|_2 \ll \|x\|_1$。例如在 x 是一个定义在 $[n]$ 上的均匀分布时，有 $\|x\|_2 = \frac{1}{\sqrt{n}} \ll \|x\|_1 = 1$。

Count Sketch 数据流版本的算法描述如下：

1. 选 d 个 2-方独立（2-wise independent）的哈希函数 $h_1, \ldots, h_d : [n] \to [w]$，选 d 个 2-方独立的哈希函数 $g_1, \ldots, g_d : [n] \to \{-1, 1\}$；
2. 初始化 $C[\ell, s] = 0$，其中 $1 \le \ell \le d, 1 \le s \le w$；
3. 对流中的每一个元素 $e_t = (i_t, \Delta_t)$：
   - 对每个 ℓ（$1 \le \ell \le d$），更新 $C[\ell, h_\ell(i_t)] = C[\ell, h_\ell(i_t)] + g_\ell(i_t) \cdot \Delta_t$；
4. 对每个 $i \in [n]$，令 $\tilde{x}_i = \text{median}_{\ell=1,\ldots,d}\{g_\ell(i) \cdot C[\ell, h_\ell(i)]\}$；
5. 查询 i 时，输出 $\tilde{x}_i$。

引理 11. *考虑严格旋转门模型（strict turnstile model），即在任意时刻都有* $x \ge 0$。*令* $d \ge 18 \log \frac{1}{\delta}$、$w > 3k^2$。*对任意固定的* $i \in [n]$，*有*

$$\Pr\left[|\tilde{x}_i - x_i| \ge \frac{\|x\|_2}{k}\right] \le \delta.$$

注意：

- 引理 11 中以至少 $1 - \delta$ 的概率对固定的 i 满足 $\tilde{x}_i \le x_i + \frac{\|x\|_2}{k}$；引理 9 中以至少 $1 - \delta$ 的概率对固定的 i 满足 $\tilde{x}_i \le x_i + \frac{\|x\|_1}{k}$；由事实 10 可知 $x_i + \frac{\|x\|_2}{k} \le x_i + \frac{\|x\|_1}{k}$，因此 Count Sketch 相比较 CountMin Sketch 有更小的误差。
- 引理 11 中固定了 $i \in [n]$，而不是对所有的 $i \in [n]$ 都有上述式子成立。我们可以令 $\delta = \frac{1}{n^c}, c \ge 2$，针对 i 使用 union bound，即可得到对于所有的 $i \in [n]$，$\Pr\left[\tilde{x}_i < x_i + \frac{\|x\|_2}{k}\right] \ge 1 - n\delta = 1 - \frac{1}{n^{c-1}}$。

接下来我们首先分析在 $d \ge 18 \log \frac{1}{\delta}$、$w > 3k^2$ 时，算法的空间复杂度；其次对引理 11 进行证明。

空间复杂度：

- 算法使用 d 个 2-方独立的哈希函数 h 和 d 个 2-方独立的哈希函数 g，需要 $2d\cdot\log n=\Omega(\log\frac{1}{\delta}\cdot\log n)$ 比特；
- 算法维护一个大小为 $d\times w$ 的二维数据 $C_{d\times w}$；数组中 $C[\ell,s]\le\|x\|_1$；因此 $d\times w$ 的二维数据需要 $d\cdot w\cdot\log\|x\|_1=\Omega\left(\log\frac{1}{\delta}\cdot k^2\cdot\log\|x\|_1\right)$ 比特：
  - CountMin Sketch 中维护二维数据需要 $\Omega\left(\log\frac{1}{\delta}\cdot k\cdot\log\|x\|_1\right)$ 比特；
  - 因此 Count Sketch 需要更多的空间。

现在我们对引理 11 进行证明。

Proof. 在 Count Sketch 算法中，有 $\tilde{x}_i=\text{median}_{\ell=1,\dots,d}\{g_\ell(i)\cdot C[\ell,h_\ell(i)]\}$。固定一个 $i\in[n]$、固定一个 $\ell\in[d]$，令

$$Z_\ell=g_\ell(i)\cdot C[\ell,h_\ell(i)].$$

此时有 $\tilde{x}_i=\text{median}_{\ell=1,\dots,d}Z_\ell$。为了方便分析，我们对任意 $i'\in[n]$，引入随机变量 $Y_{i'}$：

$$Y_{i'}=\begin{cases}1, & \text{If } h_\ell(i')=h_\ell(i)\\ 0, & \text{otherwise}\end{cases}.$$

首先分析 $C[\ell,h_\ell(i)]$ 这个格子：

- 包含 $g_\ell(i)\cdot x_i$；
- 若有 $i'\neq i$，且 $h_\ell(i')=h_\ell(i)$，那么格子中也包含 $g_\ell(i')\cdot x_{i'}$。

由于我们引入了随机变量 $Y_{i'}$，因此我们可以将 $C[\ell,h_\ell(i)]$ 写为

$$C[\ell,h_\ell(i)]=g_\ell(i)\cdot x_i+\sum_{i'\in[n],i'\neq i}g_\ell(i')\cdot x_{i'}\cdot Y_{i'}.$$

则 $Z_\ell=g_\ell(i)\cdot C[\ell,h_\ell(i)]$ 可以写为：

$$Z_\ell=(g_\ell(i))^2\cdot x_i+\sum_{i'\in[n],i'\neq i}g_\ell(i)\cdot g_\ell(i')\cdot x_{i'}\cdot Y_{i'}=x_i+\sum_{i'\in[n],i'\neq i}g_\ell(i)\cdot g_\ell(i')\cdot x_{i'}\cdot Y_{i'}. \tag{1}$$

因此

$$\begin{aligned}\mathbb{E}[Z_\ell]&=\mathbb{E}\left[x_i+\sum_{i'\in[n],i'\neq i}g_\ell(i)\cdot g_\ell(i')\cdot x_{i'}\cdot Y_{i'}\right]\\&=x_i+\sum_{i'\in[n],i'\neq i}x_{i'}\cdot\mathbb{E}[g_\ell(i)\cdot g_\ell(i')\cdot Y_{i'}] &&(\text{固定了 } i\in[n])\\&=x_i+\sum_{i'\in[n],i'\neq i}x_{i'}\cdot\mathbb{E}[g_\ell(i)\cdot g_\ell(i')]\cdot\mathbb{E}[Y_{i'}] &&(Y_{i'}\text{ 与 } h \text{ 有关，和 } g \text{ 独立})\end{aligned}$$

$$
\begin{aligned}
&= x_i + \sum_{i' \in [n], i' \neq i} x_{i'} \cdot 0 \cdot \mathbb{E}\left[Y_{i'}\right] && (\mathbb{E}\left[g_\ell(i) \cdot g_\ell(i')\right] = 0) \\
&= x_i .
\end{aligned}
$$

注：$\mathbb{E}\left[g_\ell(i) \cdot g_\ell(i')\right] = 1 \cdot \Pr\left[g_\ell(i) = g_\ell(i')\right] + (-1) \cdot \Pr\left[g_\ell(i) \neq g_\ell(i')\right] = \frac{1}{2} - \frac{1}{2} = 0.$
至此我们证明了在固定一个 $i \in [n]$、固定一个 $\ell \in [d]$ 时，有 $\mathbb{E}\left[Z_\ell\right] = x_i$。
与分析 $\mathbb{E}\left[Z_\ell\right]$ 类似，我们分析 $\mathrm{Var}\left[Z_\ell\right]$：

$$
\begin{aligned}
\mathrm{Var}\left[Z_\ell\right] &= \mathbb{E}\left[\left(Z_\ell - \mathbb{E}\left[Z_\ell\right]\right)^2\right] \\
&= \mathbb{E}\left[\left(Z_\ell - x_i\right)^2\right] && (\mathbb{E}\left[Z_\ell\right] = x_i) \\
&= \mathbb{E}\left[\left(\sum_{i' \in [n], i' \neq i} g_\ell(i) \cdot g_\ell(i') \cdot x_{i'} \cdot Y_{i'}\right)^2\right] && (\text{根据式 } 1\ ) \\
&= \mathbb{E}\left[\sum_{i' \in [n], i' \neq i} x_{i'}^2 \cdot Y_{i'}^2 + \sum_{i', i'' \in [n], i' \neq i, i'' \neq i, i' \neq i''} g_\ell(i') g_\ell(i'') \cdot x_{i'} x_{i''} \cdot Y_{i'} Y_{i''}\right] && (\text{平方项展开}) \\
&= \sum_{i' \in [n], i' \neq i} \mathbb{E}\left[x_{i'}^2 \cdot Y_{i'}^2\right] + \sum_{i', i'' \in [n], i' \neq i, i'' \neq i, i' \neq i''} x_{i'} x_{i''} \cdot \mathbb{E}\left[g_\ell(i') g_\ell(i'') \cdot Y_{i'} Y_{i''}\right] \\
&= \sum_{i' \in [n], i' \neq i} \mathbb{E}\left[x_{i'}^2 \cdot Y_{i'}^2\right] + \sum_{i', i'' \in [n], i' \neq i, i'' \neq i, i' \neq i''} x_{i'} x_{i''} \cdot \mathbb{E}\left[g_\ell(i') g_\ell(i'')\right] \cdot \mathbb{E}\left[Y_{i'} Y_{i''}\right] \\
&= \sum_{i' \in [n], i' \neq i} \mathbb{E}\left[x_{i'}^2 \cdot Y_{i'}^2\right] + \sum_{i', i'' \in [n], i' \neq i, i'' \neq i, i' \neq i''} x_{i'} x_{i''} \cdot 0 \cdot \mathbb{E}\left[Y_{i'} Y_{i''}\right] \\
&= \sum_{i' \in [n], i' \neq i} \mathbb{E}\left[x_{i'}^2 \cdot Y_{i'}^2\right] \\
&= \sum_{i' \in [n], i' \neq i} x_{i'}^2 \cdot \Pr\left[h_\ell(i') = h_\ell(i)\right] \\
&= \sum_{i' \in [n], i' \neq i} x_{i'}^2 \cdot \frac{1}{w} \\
&\leq \frac{\|x\|_2^2}{w} .
\end{aligned}
$$

由 Chebyshev 不等式，有

$$
\begin{aligned}
\Pr\left[\left|Z_\ell - x_i\right| \geq \frac{\|x\|_2}{k}\right] = \Pr\left[\left|Z_\ell - \mathbb{E}\left[Z_\ell\right]\right| \geq \frac{\|x\|_2}{k}\right] &\leq \frac{\mathrm{Var}\left[Z_\ell\right]}{\frac{\|x\|_2^2}{k^2}} \\
&\leq \frac{\frac{\|x\|_2^2}{w}}{\frac{\|x\|_2^2}{k^2}} = \frac{k^2}{w} \\
&< \frac{1}{3} && (w > 3k^2)\ .
\end{aligned}
$$

因此，对任意 $\ell \in [d]$，有 $\Pr\left[|Z_\ell - x_i| \geq \frac{\|x\|_2}{k}\right] < \frac{1}{3}$。又因为 $\tilde{x}_i = \text{median}_{\ell=1,\ldots,d} Z_\ell$，即使用了 median trick。用 X_ℓ 表示 Z_ℓ 是否满足 $|Z_\ell - x_i| \leq \frac{\|x\|_2}{k}$，并定义 X 如下：

$$X_\ell = \begin{cases} 1, & |Z_\ell - x_i| \leq \frac{\|x\|_2}{k} \\ 0, & \text{otherwise} \end{cases} \qquad X = \sum_{\ell=1}^{d} X_\ell.$$

注意到这 d 个 X_ℓ 是独立的且 $\mathbb{E}[X] = \sum_{\ell=1}^{d} \mathbb{E}[X_\ell] \geq \frac{2d}{3}$。若最终输出的 $\tilde{x}_i$ 不满足 $|\tilde{x} - x_i| \leq \frac{\|x\|_2}{k}$（即 $\tilde{x}_i$ 满足 $|\tilde{x} - x_i| \geq \frac{\|x\|_2}{k}$），则至少有半数的 Z_ℓ 不满足 $|Z_\ell - x_i| \leq \frac{\|x\|_2}{k}$，即 $X = \sum_{\ell=1}^{d} X_\ell < \frac{d}{2}$。根据 Hoeffding 不等式，

$$\Pr\left[X - \frac{2d}{3} \leq -\frac{d}{6}\right] \leq \exp\left(-\frac{d}{18}\right).$$

在 $d \geq 18 \ln \frac{1}{\delta}$ 时有

$$\Pr\left[X - \frac{2d}{3} \leq -\frac{d}{6}\right] \leq \delta.$$

即在 $d \geq 18 \log \frac{1}{\delta}$、$w > 3k^2$。对任意固定的 $i \in [n]$，有

$$\Pr\left[|\tilde{x}_i - x_i| \geq \frac{\|x\|_2}{k}\right] \leq \delta.$$

■