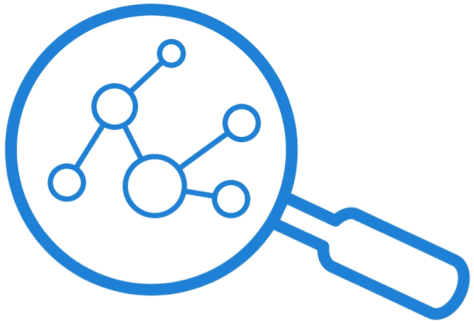
数据科学与大数据技术的数学基础

第三讲



计算机学院
余皓然

2023/4/25

课程内容

Part1 随机化方法
一致性哈希 布隆过滤器 **CM Sketch方法** 最小哈希 欧氏距离下的相似搜索 Jaccard相似度下的相似搜索
Part2 谱分析方法
主成分分析 奇异值分解 谱图论
Part3 最优化方法
压缩感知

CM Sketch方法

最高频元素寻找问题

最高频元素寻找问题

最高频元素寻找问题（Finding the Majority Element）

给定长度为n的数组A，如何寻找出现次数超过n/2的元素？

应用举例：

- 统计获得选票过半的候选人

最高频元素寻找问题

最高频元素寻找问题（Finding the Majority Element）

给定长度为n的数组A，如何寻找出现次数超过n/2的元素？

方法一：依次查看每个元素并对每个元素计数

Algorithm 1: Find the majority element (if exists) of an array

```
Data: Array x
Function Naive-Approach(x):
    dict = {};
    n = length(x);
    for each element m in x do
        if m exists in dict then
            dict[m] = dict[m] + 1;
        else
            dict[m] = 1
        end
    end
k, freq = max(dict);
if freq ≥ ⌈n/2⌉ then
    return k;
else
    return 'Not exist';
end
```

空间复杂度$O(n)$

伪代码取自Baeldung <Find the Majority Element of an Array>

最高频元素寻找问题

最高频元素寻找问题（Finding the Majority Element）

给定长度为n的数组A，如何寻找出现次数超过n/2的元素？

方法二：对数组A排序再检查中值（满足出现次数超过n/2条件的最高频元素一定是中值）

Algorithm 2: Find the majority element (if exists) of an array

```
Data: Array x
Function Improved-Approach(x):
    n = length(x);
    sort(x);
    med = x[(n+1)/2];
    freq = 0;
    for each element m in x do
        if m == med then
            freq = freq + 1;
        end
    end
    if freq ≥ ⌈n/2⌉ then
        return k;
    else
        return 'Not exist';
    end
```

排序的最低时间复杂度$O(n\log n)$

伪代码取自Baeldung <Find the Majority Element of an Array>

最高频元素寻找问题

方法三：Boyer-Moore算法

Algorithm 3: Find the majority element (if exists) of an array

```
Data: Array x
Function Boyer-Moore(x):
    k = x[0];
    c = 1;
    n = length(x);
    for each element m in x starting from x[1] do
        if c == 0 then
            k = m;
            c = 1;
        else
            if k == m then
                c = c + 1
            else
                c = c - 1
            end
        end
    end
freq = 0;
for each element m in x do
    if m == k then
        freq = freq + 1;
    end
end
if freq ≥ ⌈n/2⌉ then
    return k;
else
    return 'Not exist';
end
```

k：当前猜测的最高频元素（出现次数超过n/2的元素）

c：每检查一个元素，该值+1或-1

伪代码取自Baeldung <Find the Majority Element of an Array>

最高频元素寻找问题

方法三：Boyer-Moore算法

Algorithm 3: Find the majority element (if exists) of an array

```
Data: Array x
Function Boyer-Moore(x):
    k = x[0];
    c = 1;
    n = length(x);
    for each element m in x starting from x[1] do
        if c == 0 then
            k = m;
            c = 1;
        else
            if k == m then
                c = c + 1
            else
                c = c - 1
            end
        end
    end
freq = 0;
for each element m in x do
    if m == k then
        freq = freq + 1;
    end
end
if freq ≥ ⌈n/2⌉ then
    return k;
else
    return 'Not exist';
end
```

k：当前猜测的最高频元素（出现次数超过n/2的元素）

c：每检查一个元素，该值+1或-1

如果已知数组A一定包含一个最高频元素，则k即该元素（不用执行后续命令，即只需扫一遍数组）

伪代码取自Baeldung <Find the Majority Element of an Array>

最高频元素寻找问题

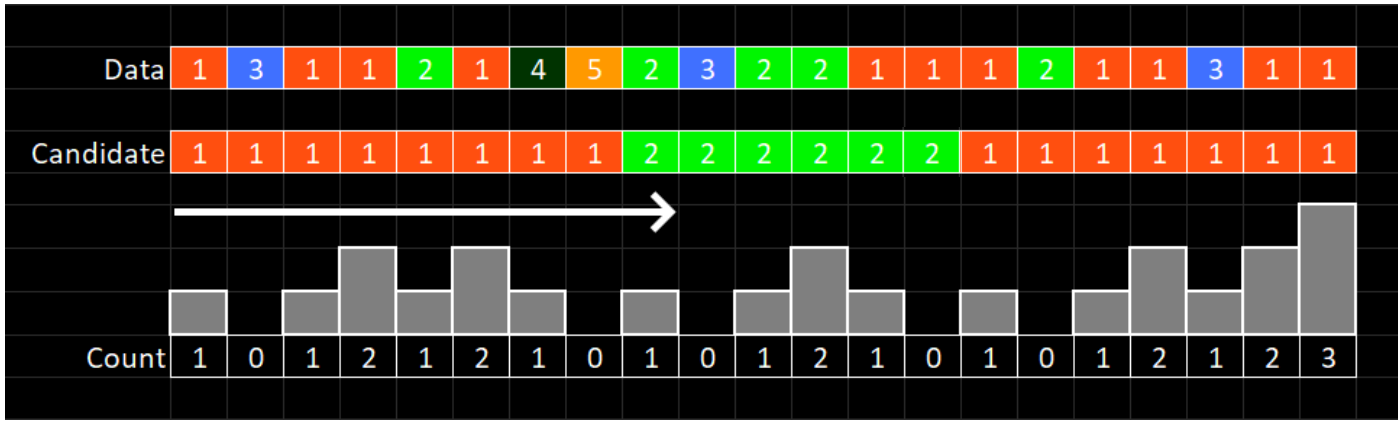
方法三：Boyer-Moore算法

Algorithm 3: Find the majority element (if exists) of an array

```
Data: Array x
Function Boyer-Moore(x):
    k = x[0];
    c = 1;
    n = length(x);
    for each element m in x starting from x[1] do
        if c == 0 then
            k = m;
            c = 1;
        else
            if k == m then
                c = c + 1
            else
                c = c - 1
            end
        end
    end
freq = 0;
for each element m in x do
    if m == k then
        freq = freq + 1;
    end
end
if freq ≥ ⌈n/2⌉ then
    return k;
else
    return 'Not exist';
end
```

k：当前猜测的最高频元素（出现次数超过n/2的元素）

c：每检查一个元素，该值+1或-1



图例修改自DataGenetics < Heavy Hitters Algorithm >

最高频元素寻找问题

方法三：Boyer-Moore算法

Algorithm 3: Find the majority element (if exists) of an array

```
Data: Array x
Function Boyer-Moore(x):
    k = x[0];
    c = 1;
    n = length(x);
    for each element m in x starting from x[1] do
        if c == 0 then
            k = m;
            c = 1;
        else
            if k == m then
                c = c + 1
            else
                c = c - 1
            end
        end
    end
freq = 0;
for each element m in x do
    if m == k then
        freq = freq + 1;
    end
end
if freq ≥ ⌈n/2⌉ then
    return k;
else
    return 'Not exist';
end
```

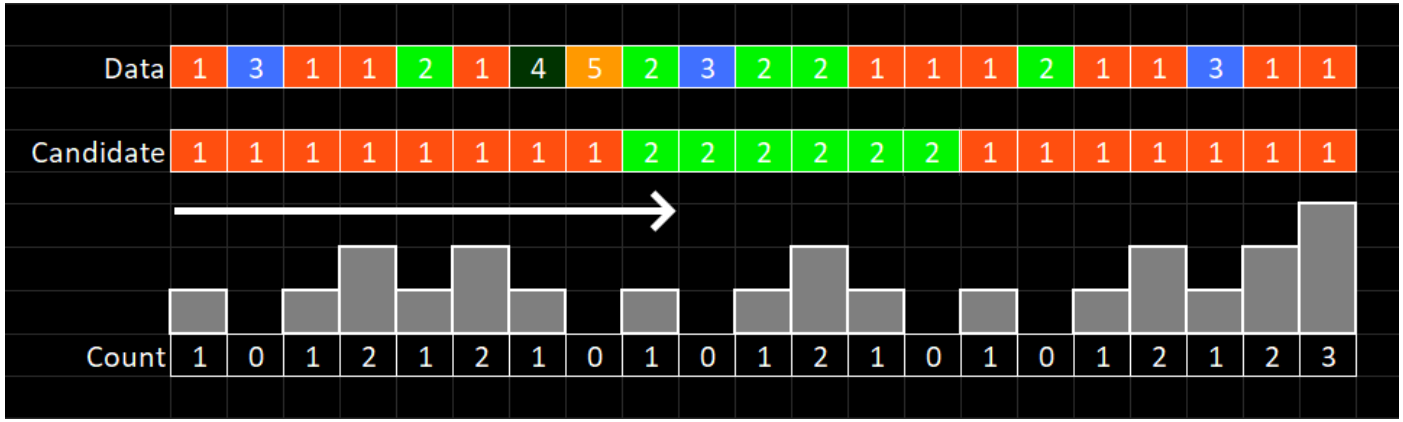
k：当前猜测的最高频元素（出现次数超过n/2的元素）

c：每检查一个元素，该值+1或-1

时间复杂度$O(n)$：优于方法二（排序）

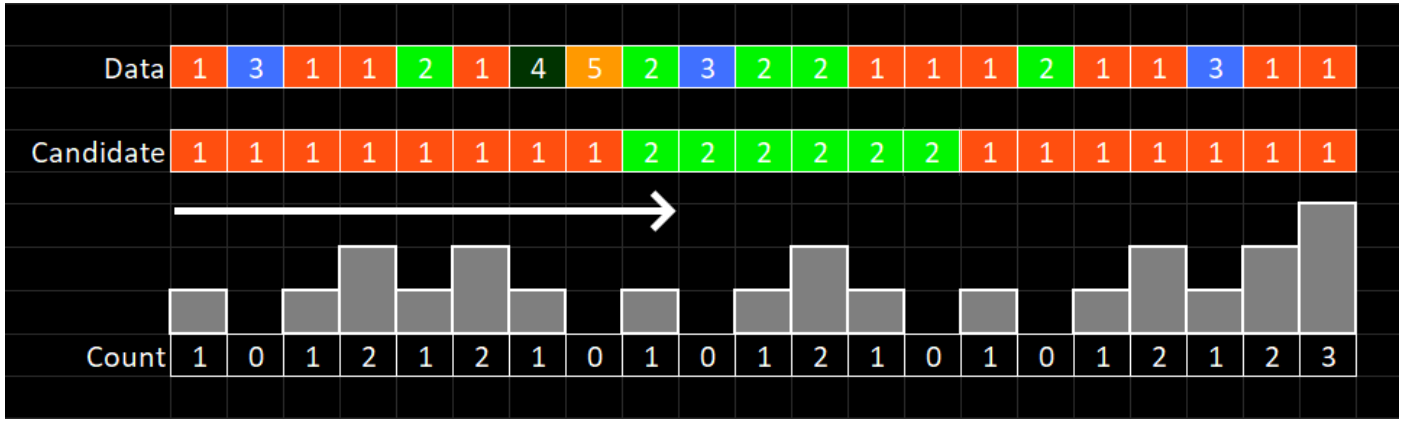
空间复杂度 $O(1)$：即不随n变化，优于方法一（计数）

最高频元素寻找问题



如何证明若存在最高频元素，最后的candidate一定是该最高频元素？

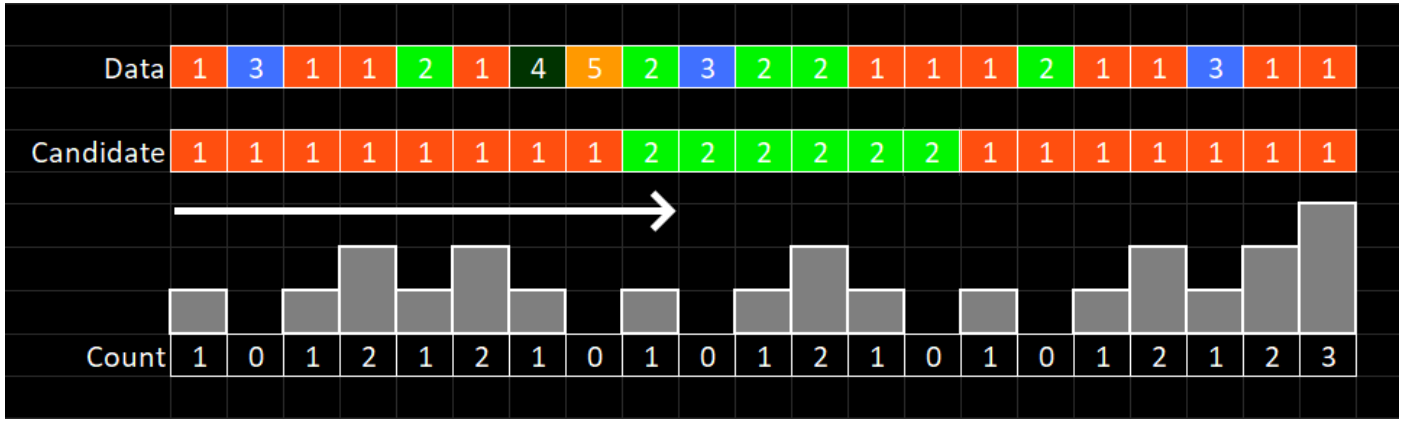
最高频元素寻找问题



如何证明若存在最高频元素，最后的candidate一定是该最高频元素？

如何证明最后一次count=0出现前的信息不影响对最高频元素的判断？

最高频元素寻找问题



如何证明若存在最高频元素，最后的candidate一定是该最高频元素？

如何证明最后一次count=0出现前的信息不影响对最高频元素的判断？

关键步骤：证明如果数组存在最高频元素，那么从数组中拿掉两个不一样的元素，剩余元素中的最高频元素不变

CM Sketch方法

高频元素寻找问题

高频元素寻找问题

最高频元素寻找问题（Finding the Majority Element）

给定长度为n的数组A，如何寻找出现次数超过n/2的元素？

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

当$k = 2 - \varepsilon$（ε为极小正数）时，高频元素寻找问题即变为最高频元素寻找问题

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

应用举例：

- 电商平台统计历史上（如过去一天内）用户浏览频繁的商品
- 搜索引擎统计历史上（如过去一天内）用户搜索频繁的内容
- 交换机统计历史上流量大的TCP流
- ...

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

根据解决“最高频元素寻找问题”的方法，最直接的求解方法是：

依次查看每个元素并对每个元素计数，找出高频元素

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

根据解决“最高频元素寻找问题”的方法，最直接的求解方法是：

依次查看每个元素并对每个元素计数，找出高频元素

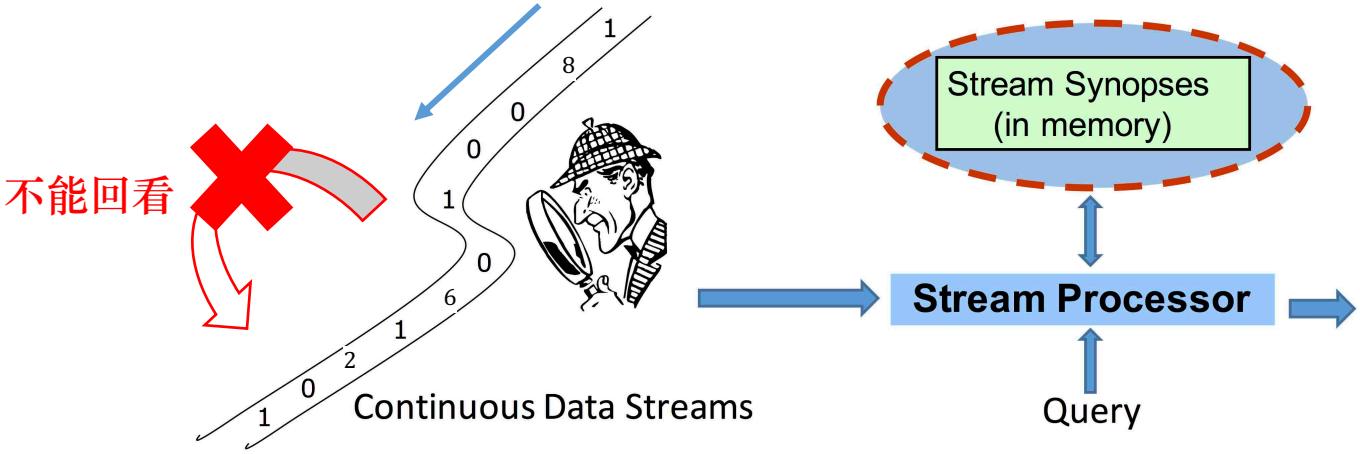
空间复杂度$O(n)$，当n很大（$10^7 \sim 10^9$，如电商平台浏览商品总数），需占用大量存储空间

在电商平台高频商品寻找、搜索引擎高频内容判断等应用中，经常需要实时、在线解决问题

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？



数据流任务要求在消耗较少的存储资源且仅浏览数据一遍的约束下解决“高频元素寻找问题”

图片取自S. Raskhodnikova and A. Smith <Scalable, Private Algorithms for Continual Data Analysis >

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

不可能定理（Impossibility Result）

在仅浏览数据一遍的约束下，不存在任何算法可以在保证次线性空间复杂度的同时解决高频元素寻找问题。

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

不可能定理（Impossibility Result）

在仅浏览数据一遍的约束下，不存在任何算法可以在保证次线性空间复杂度的同时解决高频元素寻找问题。

次线性复杂度（sublinear complexity）：$o(n)$

注意是小写o不是大写O

$$f(x) = o(g(x)) \quad \text{as } x \to \infty$$

if for every positive constant ε there exists a constant x_0 such that

$$|f(x)| \leq \varepsilon g(x) \quad \text{for all } x \geq x_0.$$

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

不可能定理（Impossibility Result）

在仅浏览数据一遍的约束下，不存在任何算法可以在保证次线性空间复杂度的同时解决高频元素寻找问题。

次线性复杂度（sublinear complexity）：$o(n)$

The difference between the definition of the big-O notation and the definition of little-o is that while the former has to be true for *at least one* constant *M*, the latter must hold for *every* positive constant ε, however small.[17] In this way, little-o notation makes a *stronger statement* than the corresponding big-O notation: every function that is little-o of *g* is also big-O of *g*, but not every function that is big-O of *g* is also little-o of *g*. For example, $2x^2 = O(x^2)$ but $2x^2 \neq o(x^2)$.

As *g*(*x*) is nonzero, or at least becomes nonzero beyond a certain point, the relation $f(x) = o(g(x))$ is equivalent to

$\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0$ (and this is in fact how Landau[16] originally defined the little-o notation).

比如$c\log n$、$c\sqrt{n}$满足次线性复杂度的要求；cn不满足

图片取自Wikipedia <Big O notation>

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

不可能定理（Impossibility Result）

在仅浏览数据一遍的约束下，不存在任何算法可以在保证次线性空间复杂度的同时解决高频元素寻找问题。

如何证明？

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

不可能定理（Impossibility Result）

在仅浏览数据一遍的约束下，不存在任何算法可以在保证次线性空间复杂度的同时解决高频元素寻找问题。

证明的大致思路（非严格证明）：考虑k=n/2的情况

当数组的n个元素满足：$\underbrace{|x_1|x_2|x_3|\cdots|x_{n-1}|}_{\text{set } S \text{ of distinct elements}} y|,$

除非完整存储集合S的信息，否则不能判断y是否是高频元素

存储集合S信息所需空间随n线性增长（参考上一讲中的从属问题）

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A和数值k，如何寻找出所有出现次数大于等于n/k的元素？

不可能定理（Impossibility Result）

在仅浏览数据一遍的约束下，不存在任何算法可以在保证次线性空间复杂度的同时解决高频元素寻找问题。

如何以较小的存储空间解决高频元素寻找问题？

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A、数值k，如何寻找出所有出现次数大于等于 $\frac{n}{k}$ 的元素？

如何以较小的存储空间解决高频元素寻找问题？

采用“以精确度换存储空间”的思想

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A、数值k，如何寻找出所有出现次数大于等于 $\frac{n}{k}$ 的元素？

如何以较小的存储空间解决高频元素寻找问题？

采用“以精确度换存储空间”的思想

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；“不漏掉高频元素”

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。“允许混入次高频元素”

求解该问题对于电商平台/搜索引擎等同样有重要的意义

高频元素寻找问题

高频元素寻找问题（Heavy Hitters Problem）

给定长度为n的数组A、数值k，如何寻找出所有出现次数大于等于 $\frac{n}{k}$ 的元素？

如何以较小的存储空间解决高频元素寻找问题？

采用“以精确度换存储空间”的思想

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；CM Sketch满足（1）

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。CM Sketch以大概率满足（2）

CM Sketch方法

CM Sketch方法

CM Sketch方法

An improved data stream summary: the count-min sketch and its applications

G Cormode, S Muthukrishnan - Journal of Algorithms, 2005 - Elsevier

We introduce a new sublinear space data structure—the count-min sketch—for summarizing data streams. Our sketch allows fundamental queries in data stream summarization such as point, range, and inner product queries to be approximately answered very quickly; in addition, it can be applied to solve several important problems in data streams such as finding quantiles, frequent items, etc. The time and space bounds we show for using the CM sketch to solve these problems significantly improve those previously known—typically from …

☆ Save 99 Cite Cited by 2124 Related articles All 48 versions »

Graham Cormode

Shan Muthukrishnan

Muthukrishnan receives 2014 Imre Simon award

Referenced People:

Muthukrishnan-Shan

The paper "An improved data stream summary: The count-min sketch and its applications," authored by Graham Cormode and S. Muthukrishnan, published in LATIN 2004, has been awarded the 2014 Imre Simon Test-of-Time Award. The award will be presented in the conference Latin American Theoretical Informatics (LATIN) 2014. Congratulations to Professor Muthukrishnan and his collaborator, Graham Cormode.

The Imre Simon Award was created in 2012, with the aim of recognizing the papers published in LATIN which have had the most relevant and lasting impact. Since then, each edition of the conference awards a paper published in LATIN that is at least 10 years old, in order to assess its long-term impact in the area of Theoretical Computer Science. See **http://www.latintcs.org/prize** for more information. Incidentally, Professor Martin Farach-Colton was one of the recipients of the inaugural Imre Simon award in 2012 together with his collaborator Michael Bender.

CM Sketch方法

已经得到广泛应用：

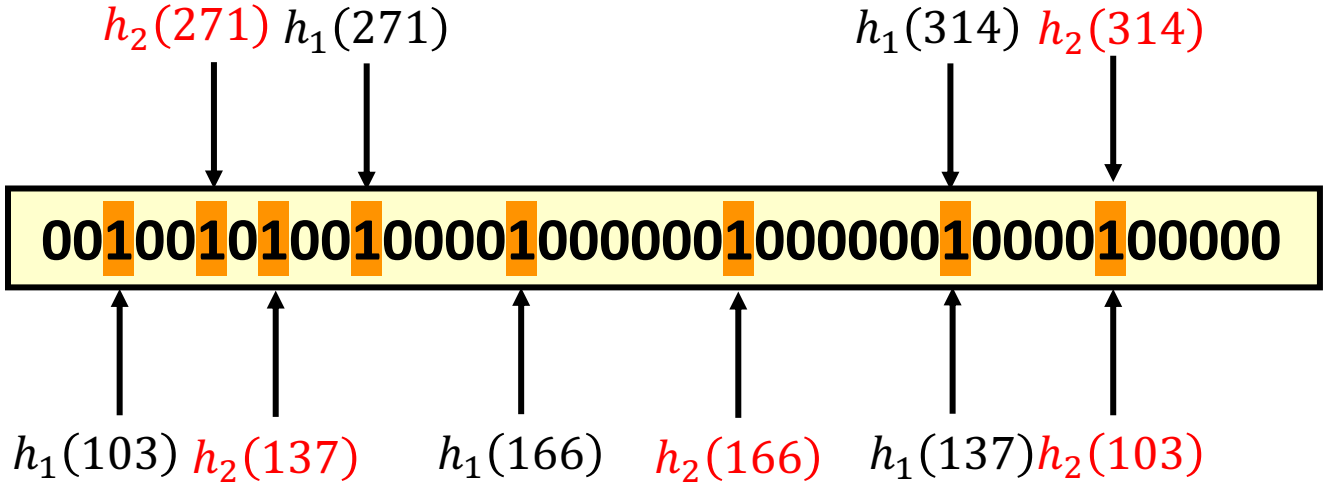
- AT&T将CM (Count-Min) Sketch用于交换机上的网络流量分析
- Google将CM Sketch用于其MapReduce系统
- ...

CM Sketch方法

已经得到广泛应用：

- AT&T将CM (Count-Min) Sketch用于交换机上的网络流量分析
- Google将CM Sketch用于其MapReduce系统
- ...

设计思路与布隆过滤器有相似之处



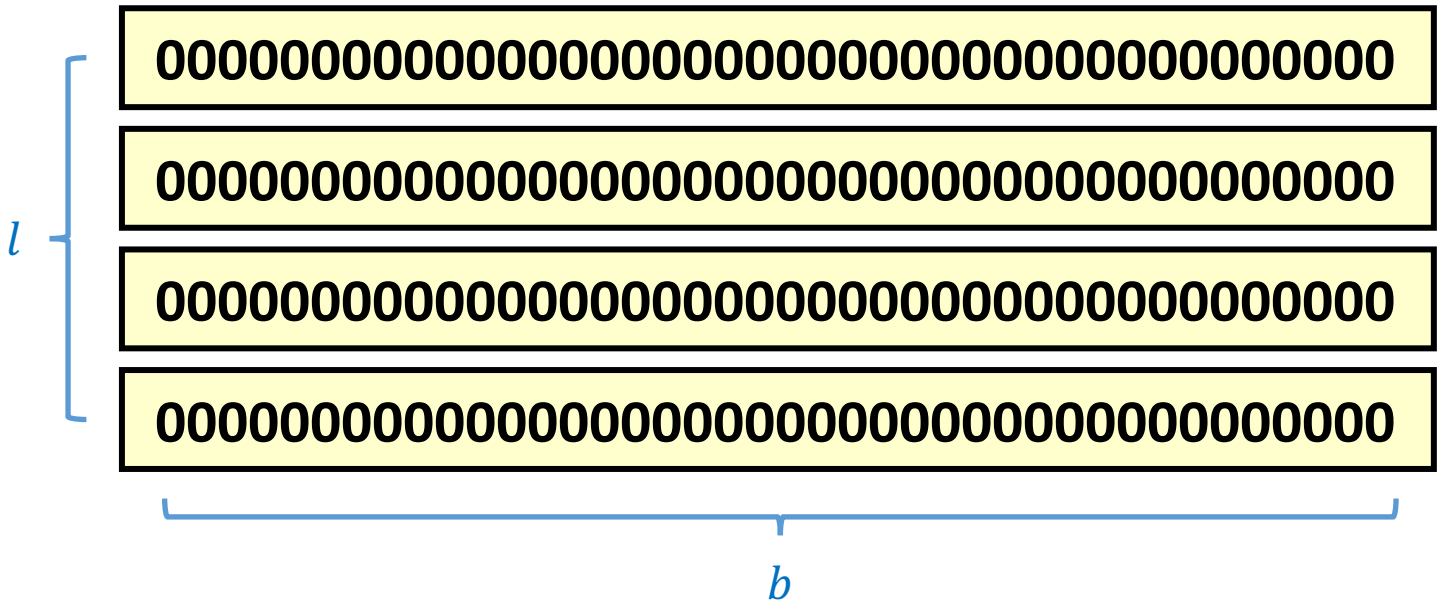
布隆过滤器的思路：（1）利用哈希进行映射（2）采用多个哈希函数提高准确率

区别：高频元素问题涉及到对元素出现频次的统计（整数），而非存在性的判断（0/1）

CM Sketch方法

假设采用l个哈希函数，每个哈希函数将依次输入的n个元素分别映射到$\{0,1,\ldots,b-1\}$

准备一个$l \times b$大小的全零数组（比如$l=4$，$b=40$，$n=10^9$）



不同于布隆过滤器：此处为每个哈希函数分别准备一个数组

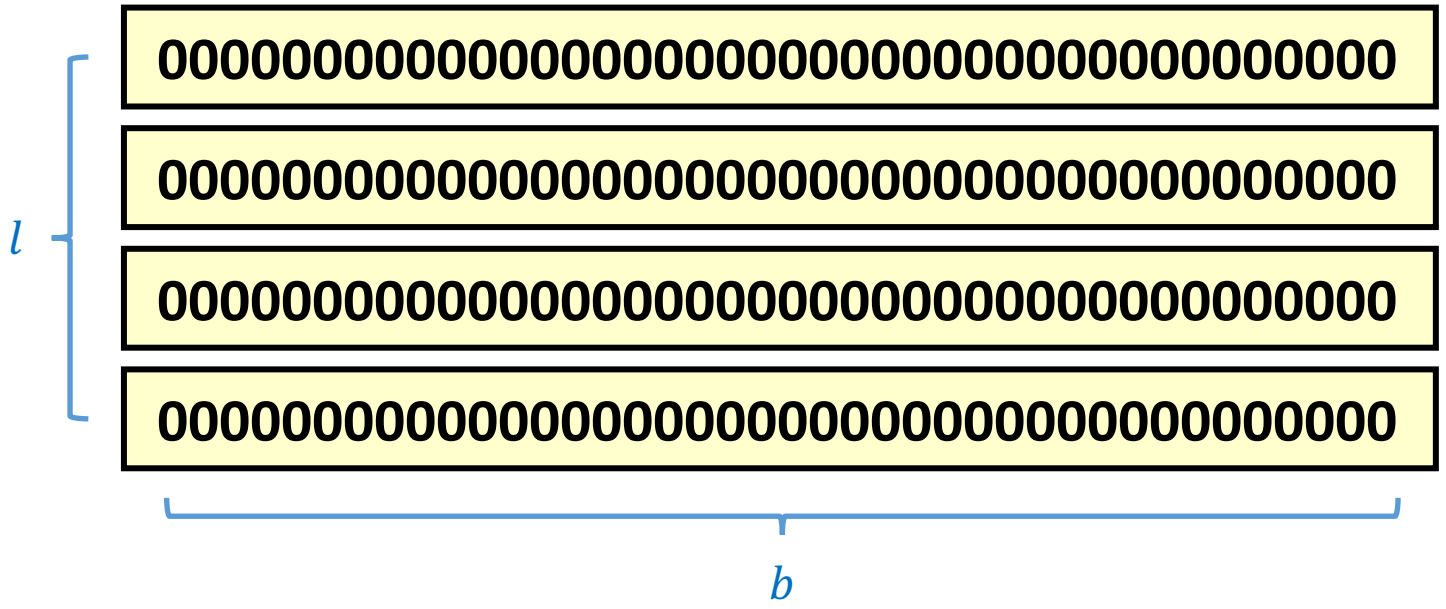
* 在介绍布隆过滤器时，所采取符号不同：k个哈希函数，数组长为m

CM Sketch方法

对每一个元素$x \in A$：（1）计算$h_1(x), h_2(x), \ldots, h_l(x)$；（2）对数组相应的$l$个位置分别加1

$\mathrm{CMS}[1][h_1(x)] \leftarrow \mathrm{CMS}[1][h_1(x)] + 1$，…，$\mathrm{CMS}[l][h_l(x)] \leftarrow \mathrm{CMS}[l][h_l(x)] + 1$

例：数组A为 [2, 6, 2, 5, 2, ...]

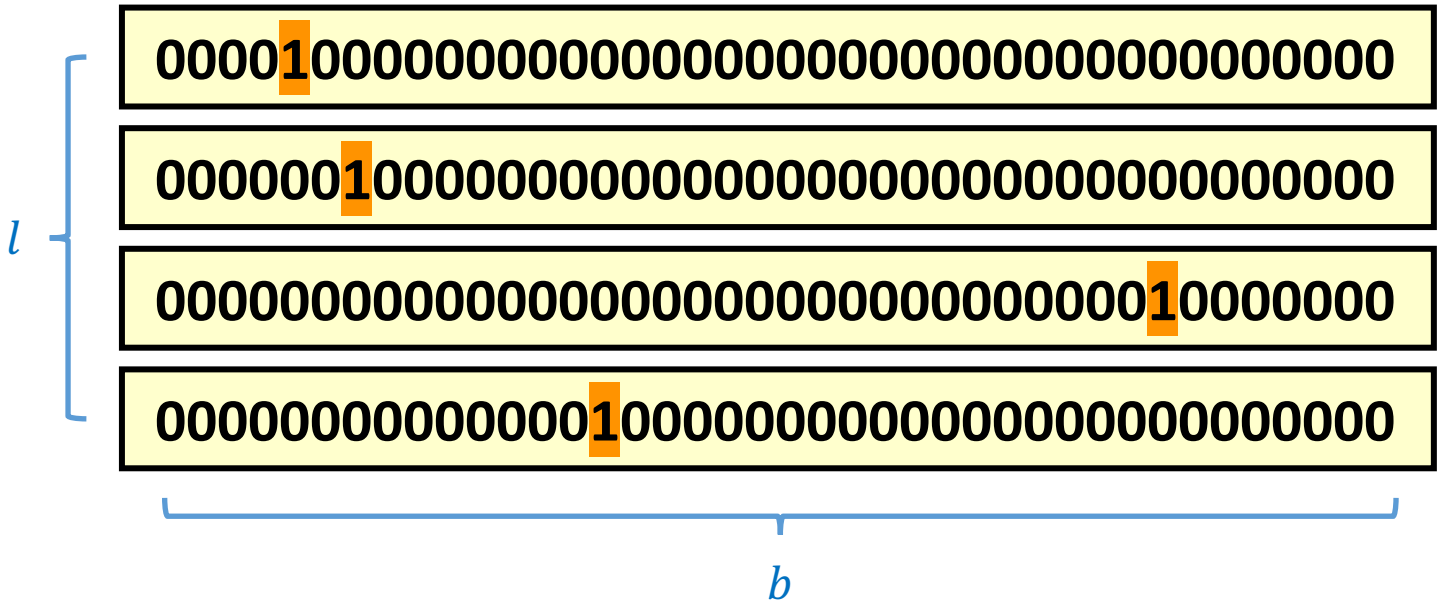


CM Sketch方法

对每一个元素$x \in A$：（1）计算$h_1(x), h_2(x), \ldots, h_l(x)$；（2）对数组相应的$l$个位置分别加1

$\mathrm{CMS}[1][h_1(x)] \leftarrow \mathrm{CMS}[1][h_1(x)] + 1$，…，$\mathrm{CMS}[l][h_l(x)] \leftarrow \mathrm{CMS}[l][h_l(x)] + 1$

例：数组A为 [2, 6, 2, 5, 2, ...]

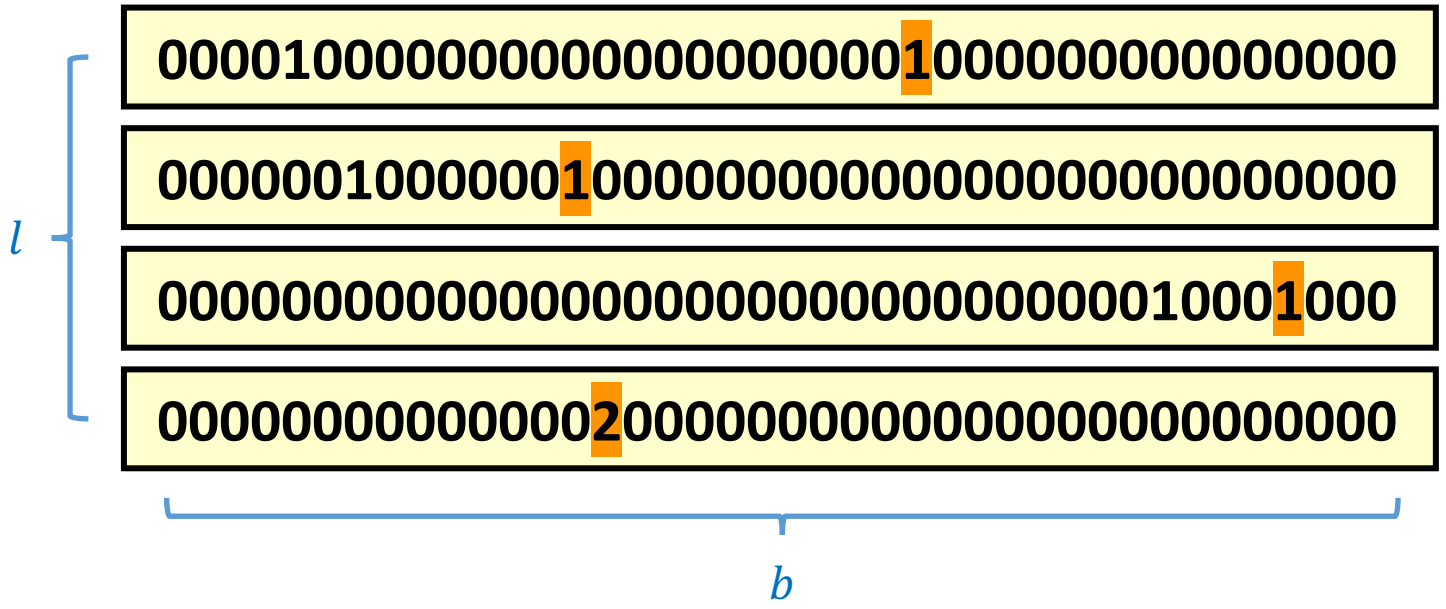


CM Sketch方法

对每一个元素$x \in A$：（1）计算$h_1(x), h_2(x), \ldots, h_l(x)$；（2）对数组相应的$l$个位置分别加1

$\mathrm{CMS}[1][h_1(x)] \leftarrow \mathrm{CMS}[1][h_1(x)] + 1$，…，$\mathrm{CMS}[l][h_l(x)] \leftarrow \mathrm{CMS}[l][h_l(x)] + 1$

例：数组A为 [2, 6, 2, 5, 2, ...]

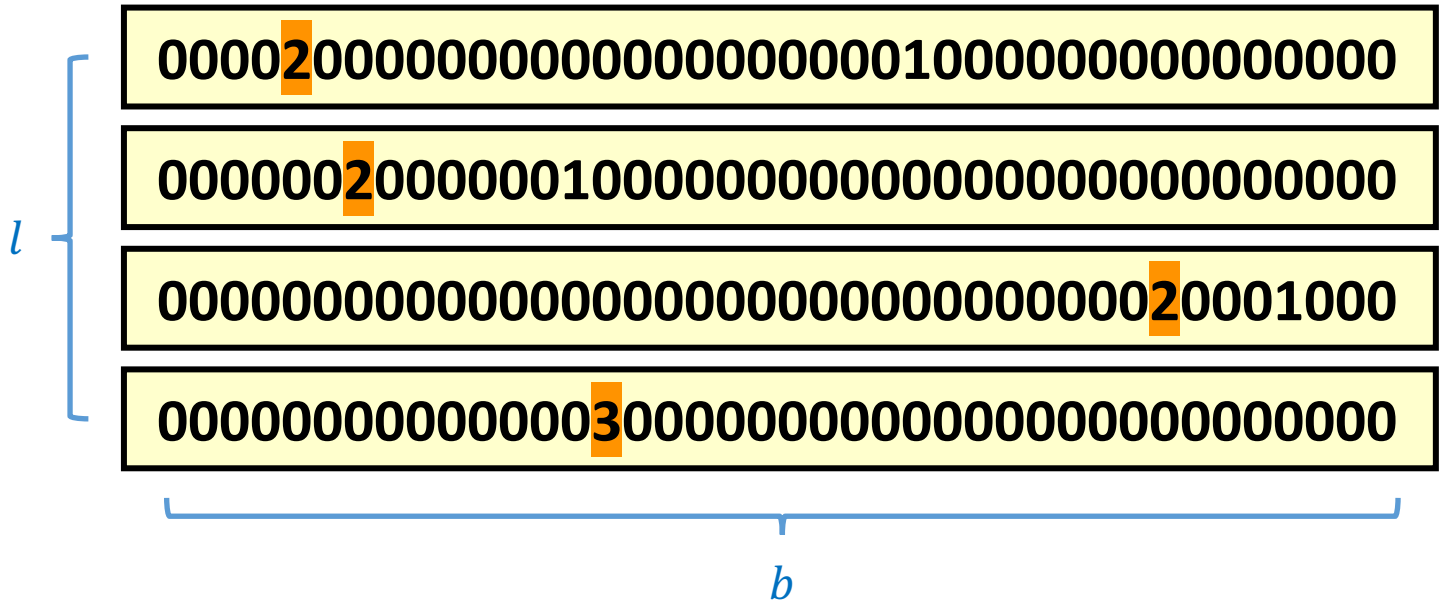


CM Sketch方法

对每一个元素$x \in A$：（1）计算$h_1(x), h_2(x), \ldots, h_l(x)$；（2）对数组相应的$l$个位置分别加1

$\mathrm{CMS}[1][h_1(x)] \leftarrow \mathrm{CMS}[1][h_1(x)] + 1$，…，$\mathrm{CMS}[l][h_l(x)] \leftarrow \mathrm{CMS}[l][h_l(x)] + 1$

例：数组A为 [2, 6, 2, 5, 2, ...]

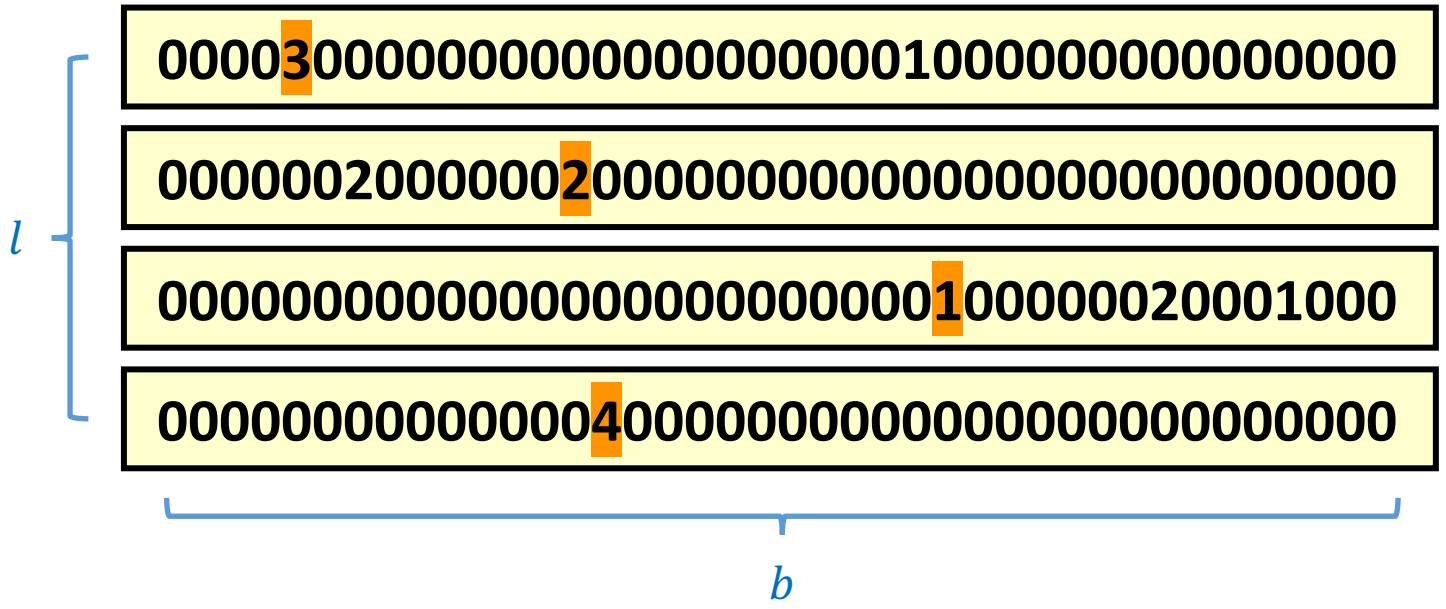


CM Sketch方法

对每一个元素$x \in A$：（1）计算$h_1(x), h_2(x), \ldots, h_l(x)$；（2）对数组相应的$l$个位置分别加1

$\mathrm{CMS}[1][h_1(x)] \leftarrow \mathrm{CMS}[1][h_1(x)] + 1$，…，$\mathrm{CMS}[l][h_l(x)] \leftarrow \mathrm{CMS}[l][h_l(x)] + 1$

例：数组A为 [2, 6, 2, 5, 2, ...]

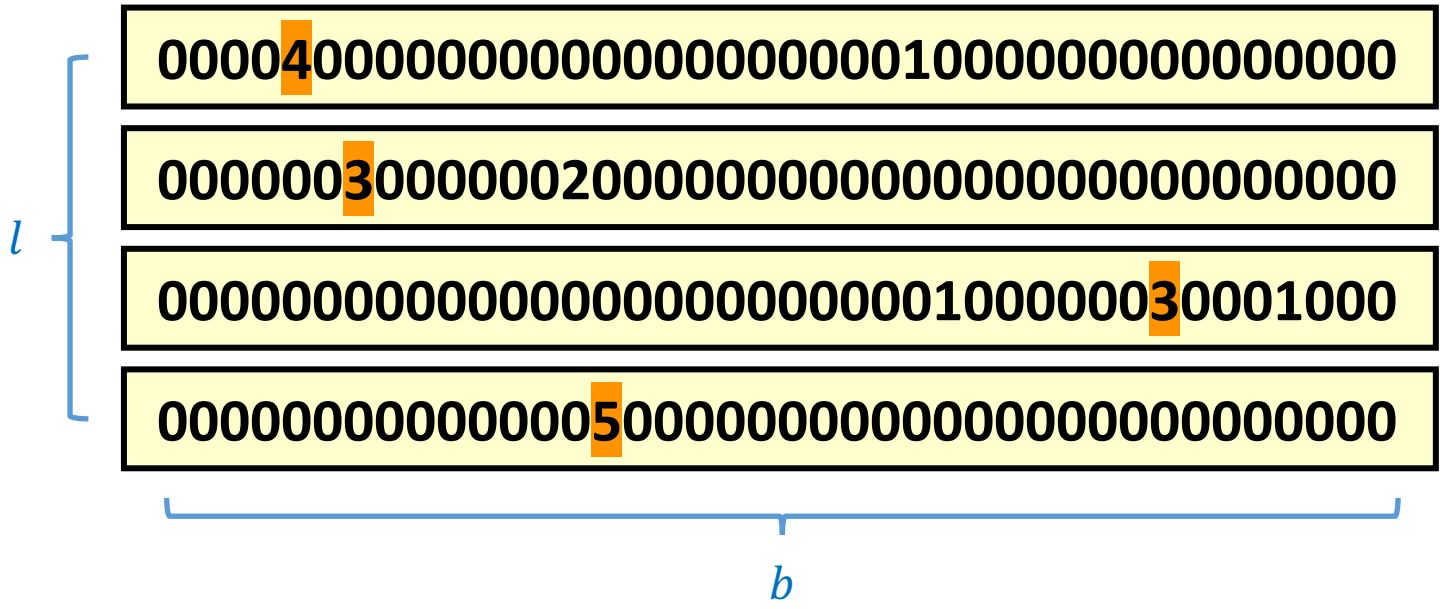


CM Sketch方法

对每一个元素$x \in A$：（1）计算$h_1(x), h_2(x), \ldots, h_l(x)$；（2）对数组相应的$l$个位置分别加1

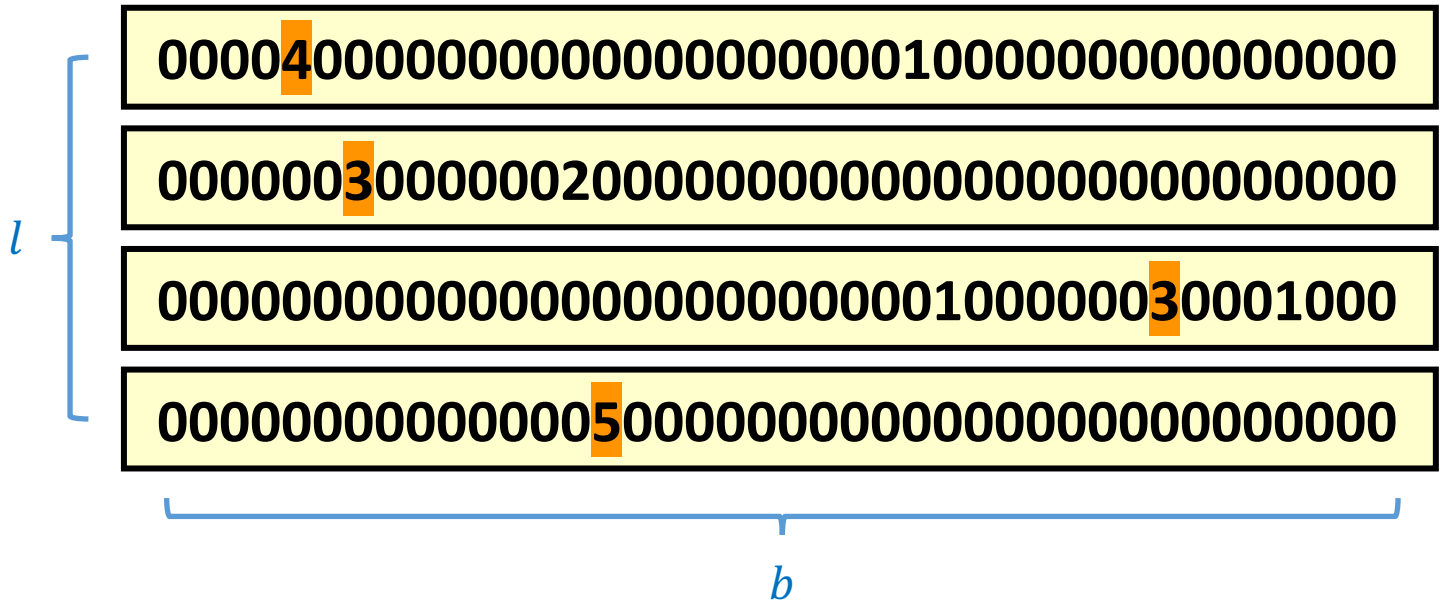
$\mathrm{CMS}[1][h_1(x)] \leftarrow \mathrm{CMS}[1][h_1(x)] + 1$，…，$\mathrm{CMS}[l][h_l(x)] \leftarrow \mathrm{CMS}[l][h_l(x)] + 1$

例：数组A为 [2, 6, 2, 5, 2, ...]



CM Sketch方法

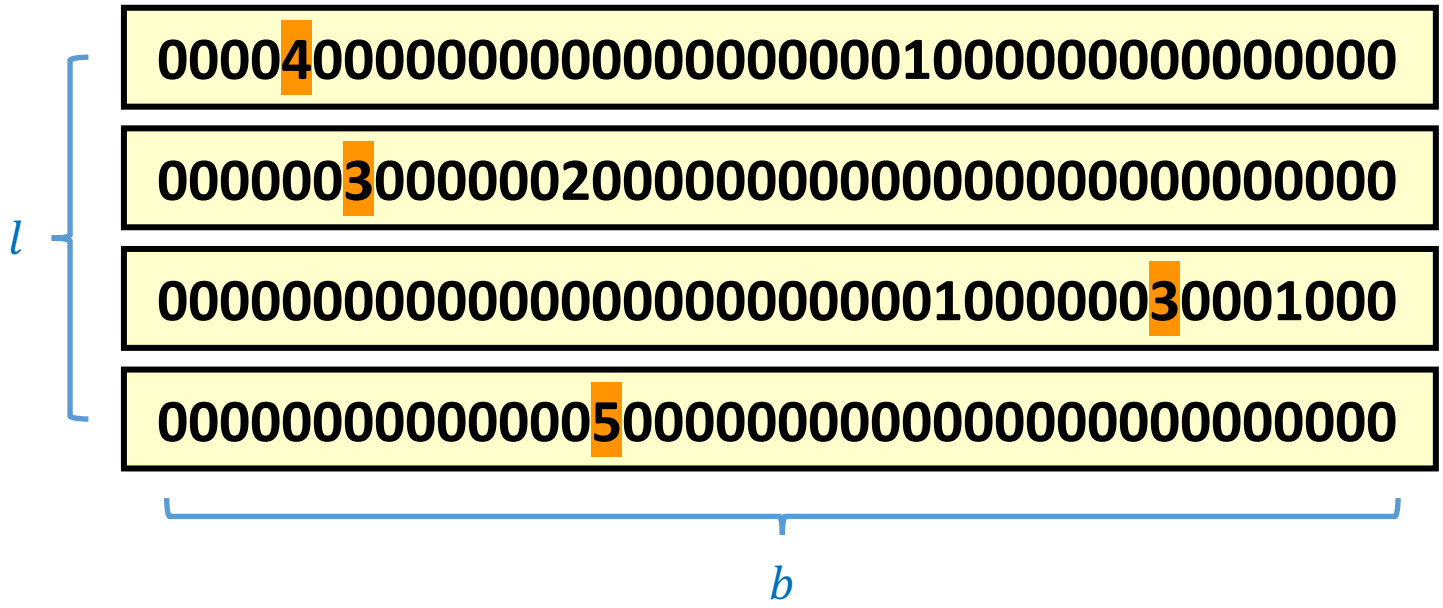
例：数组A为 [2, 6, 2, 5, 2, ...]



如何利用CMS数组判断任一元素x在数组A中的出现次数?

CM Sketch方法

例：数组A为 [2, 6, 2, 5, 2, ...]

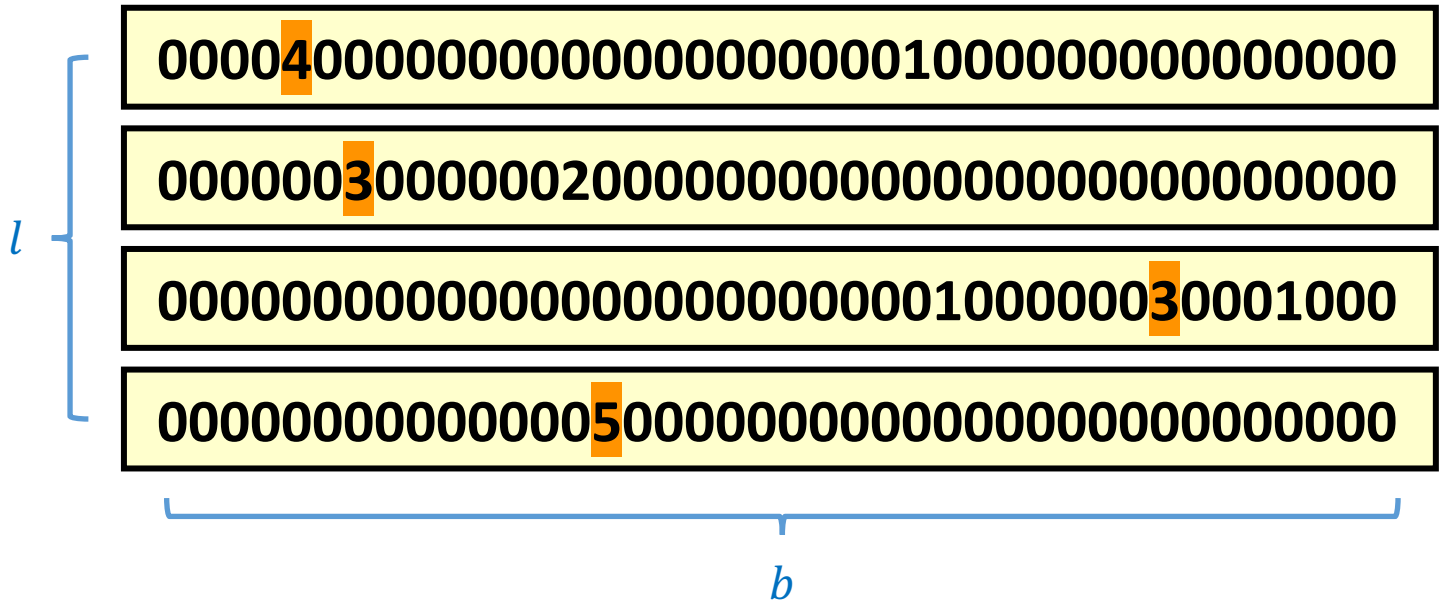


如何利用CMS数组判断任一元素x在数组A中的出现次数?

(1) 计算$h_1(x), h_2(x), \ldots, h_l(x)$;
(2) 读取CMS[1][$h_1(x)$]，…，CMS[l][$h_l(x)$]
(3) ?

CM Sketch方法

例：数组A为 [2, 6, 2, 5, 2, ...]



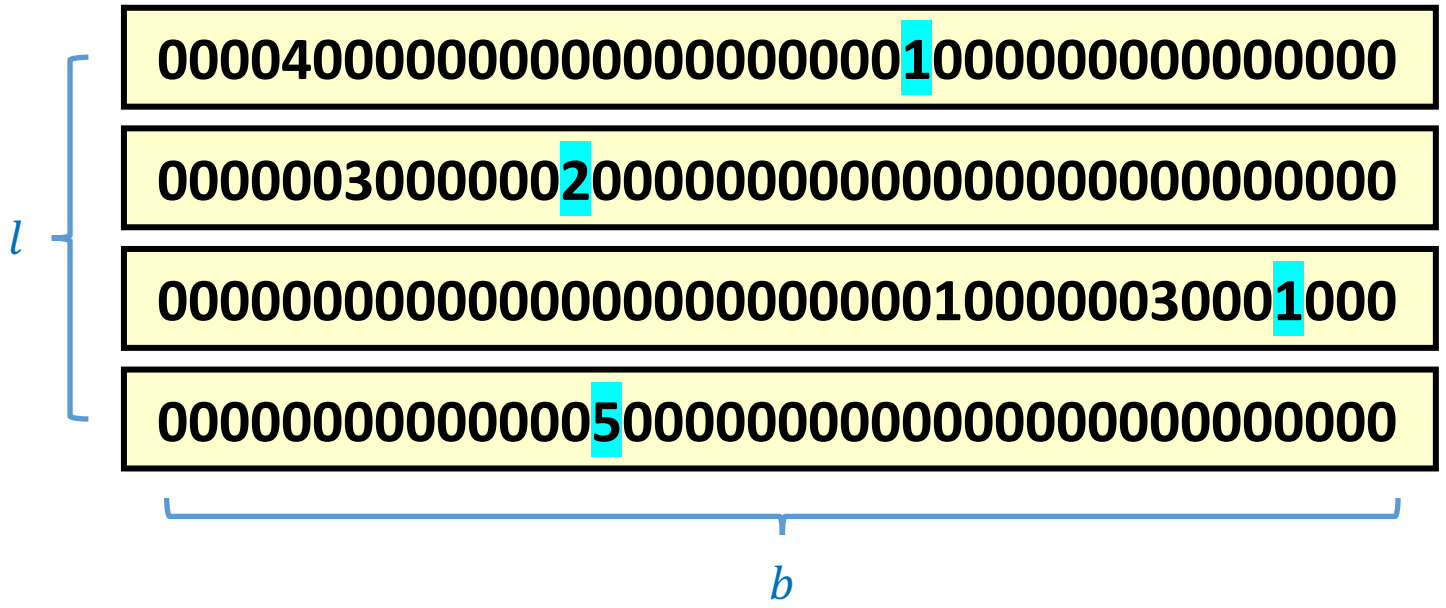
如何利用CMS数组判断任一元素x在数组A中的出现次数?

(1) 计算$h_1(x), h_2(x), \ldots, h_l(x)$;
(2) 读取CMS$[1][h_1(x)]$，…，CMS$[l][h_l(x)]$
(3) $\min_i$ CMS$[i][h_i(x)]$

$x = 2$时?

CM Sketch方法

例：数组A为 [2, 6, 2, 5, 2, ...]



如何利用CMS数组判断任一元素x在数组A中的出现次数?

(1) 计算$h_1(x), h_2(x), \ldots, h_l(x)$;
(2) 读取CMS$[1][h_1(x)]$，…，CMS$[l][h_l(x)]$
(3) $\min_i$ CMS$[i][h_i(x)]$

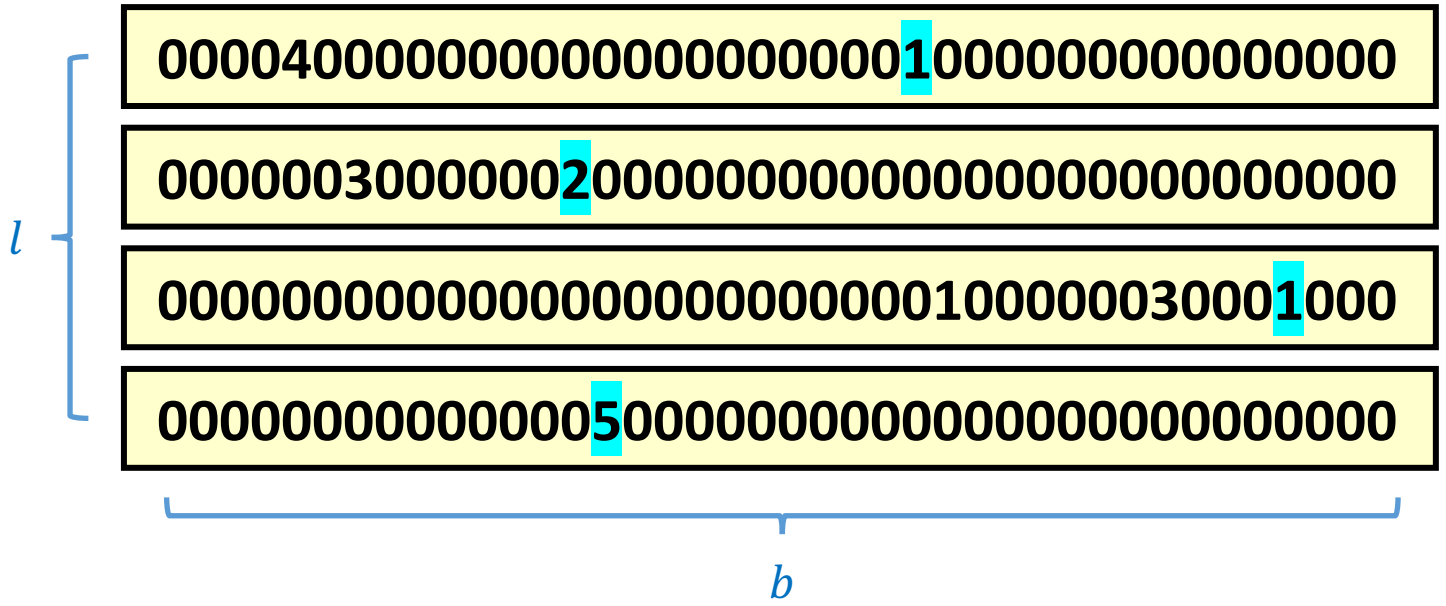
$x = 2$时？ $x = 6$时？

记元素x的实际出现次数为f_x

有$\min_i$ CMS$[i][h_i(x)] \geq f_x$

CM Sketch方法用$\min_i$ CMS$[i][h_i(x)]$近似f_x

CM Sketch方法



CM Sketch方法用$\min_i \mathrm{CMS}[i][h_i(x)]$近似$f_x$

当b增大时？当l增大时？该近似方法的精确度如何变化？

CM Sketch方法

CM Sketch方法的精度分析

CM Sketch方法

分析在给定l与b值下，$\min_i \mathrm{CMS}[i][h_i(x)]$与$f_x$的近似程度

先分析单个哈希函数（第i个哈希函数）下 $\mathrm{CMS}[i][h_i(x)]$ 与 f_x 的近似程度

CM Sketch方法

分析在给定l与b值下，$\min_i \mathrm{CMS}[i][h_i(x)]$与$f_x$的近似程度

先分析单个哈希函数（第i个哈希函数）下 $\mathrm{CMS}[i][h_i(x)]$ 与 f_x 的近似程度

数组A有n个元素

CMS数组第i行

00100060102002100015000020020043050050040

b

$$\mathrm{CMS}[i][h_i(x)] = f_x + \sum_{y \in A: y \neq x, h_i(y) = h_i(x)} f_y$$

随机值的期望的上界?

CM Sketch方法

分析在给定l与b值下，$\min_i \mathrm{CMS}[i][h_i(x)]$与$f_x$的近似程度

先分析单个哈希函数（第i个哈希函数）下 $\mathrm{CMS}[i][h_i(x)]$ 与 f_x 的近似程度

数组A有n个元素

CMS数组第i行 **00100060102002100015000020020043050050040**

b

$$\mathrm{CMS}[i][h_i(x)] = f_x + \sum_{y\in A: y\neq x, h_i(y)=h_i(x)} f_y$$

对哈希值是否冲突求期望

$$\mathbb{E}\left\{\sum_{y\in A: y\neq x, h_i(y)=h_i(x)} f_y\right\} = \mathbb{E}\left\{\sum_{y\in A: y\neq x} \mathbf{1}_{\{h_i(y)=h_i(x)\}} f_y\right\} = \sum_{y\in A: y\neq x} \Pr[h_i(y)=h_i(x)] f_y$$

$$= \sum_{y\in A: y\neq x} \frac{1}{b} f_y = \frac{1}{b} \sum_{y\in A: y\neq x} f_y \leq \frac{n}{b}$$

* x不一定属于数组A

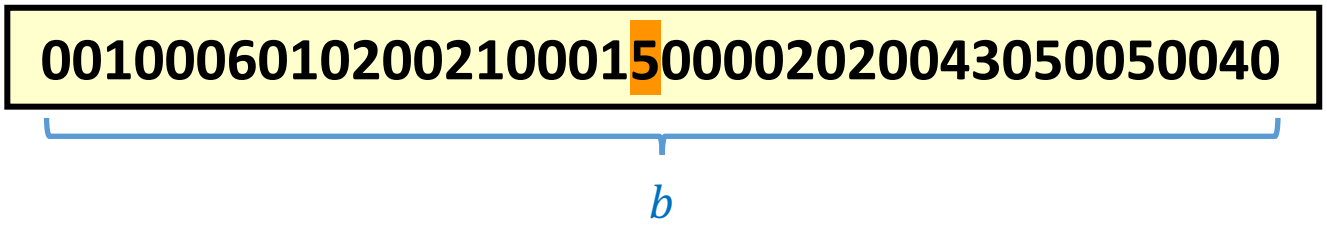
CM Sketch方法

分析在给定l与b值下，$\min_i \mathrm{CMS}[i][h_i(x)]$与$f_x$的近似程度

先分析单个哈希函数（第i个哈希函数）下 $\mathrm{CMS}[i][h_i(x)]$ 与 f_x 的近似程度

数组A有n个元素

CMS数组第i行



$$\mathbb{E}\{\mathrm{CMS}[i][h_i(x)] - f_x\} \le \frac{n}{b}$$

估测值　　真实值

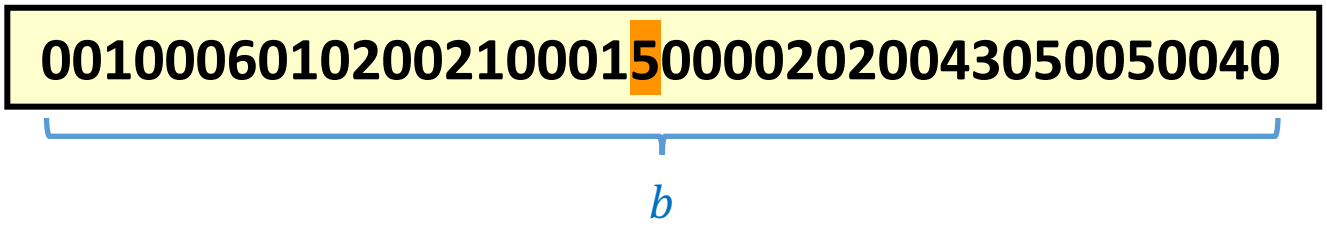
CM Sketch方法

分析在给定l与b值下，$\min_i \mathrm{CMS}[i][h_i(x)]$与$f_x$的近似程度

先分析单个哈希函数（第i个哈希函数）下 $\mathrm{CMS}[i][h_i(x)]$ 与 f_x 的近似程度

数组A有n个元素

CMS数组第i行



$$\mathbb{E}\{\mathrm{CMS}[i][h_i(x)] - f_x\} \leq \frac{n}{b}$$

估测值　　真实值

这是对于估测误差的期望的分析，如何分析估测误差过大的概率

CM Sketch方法

马尔可夫不等式（Markov's Inequality）

若X是一个非负随机变量且$\mathbb{E}[X] \neq 0$，对任何常数$c > 0$，有 $\Pr[X \geq c\mathbb{E}[X]] \leq \frac{1}{c}$.

例如已知抽奖的平均奖金是10元，问抽到的奖金大于20元的概率与0.5的关系?

答案：$\Pr[\text{奖金} \geq 20] \leq \frac{1}{2}$　即$c = 2$的特例

CM Sketch方法

马尔可夫不等式（Markov's Inequality）

若X是一个非负随机变量且$\mathbb{E}[X] \neq 0$，对任何常数$c > 0$，有 $\Pr[X \geq c\mathbb{E}[X]] \leq \frac{1}{c}$.

$$\mathbb{E}[X] = \Pr[X \geq c\mathbb{E}[X]]\mathbb{E}[X|X \geq c\mathbb{E}[X]] + \Pr[X < c\mathbb{E}[X]]\mathbb{E}[X|X < c\mathbb{E}[X]]$$

因为X非负，有$\Pr[X < c\mathbb{E}[X]]\mathbb{E}[X|X < c\mathbb{E}[X]] \geq 0$，即$\mathbb{E}[X] \geq \Pr[X \geq c\mathbb{E}[X]]\mathbb{E}[X|X \geq c\mathbb{E}[X]]$

又因为$\mathbb{E}[X|X \geq c\mathbb{E}[X]] \geq c\mathbb{E}[X]$，

有$\mathbb{E}[X] \geq \Pr[X \geq c\mathbb{E}[X]]\mathbb{E}[X|X \geq c\mathbb{E}[X]] \geq \Pr[X \geq c\mathbb{E}[X]]\ c\mathbb{E}[X]$

CM Sketch方法

马尔可夫不等式（Markov's Inequality）

若X是一个非负随机变量且$\mathbb{E}[X] \neq 0$，对任何常数$c > 0$，有 $\Pr[X \geq c\mathbb{E}[X]] \leq \frac{1}{c}$.

$$\mathbb{E}\{\mathrm{CMS}[i][h_i(x)] - f_x\} \leq \frac{n}{b}$$

估测值　　真实值

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k} - \varepsilon n$。

分析在给定数值ε下 $\Pr[\mathrm{CMS}[i][h_i(x)] - f_x \geq \varepsilon n]$

CM Sketch方法

马尔可夫不等式（Markov's Inequality）

若X是一个非负随机变量且$\mathbb{E}[X] \neq 0$，对任何常数$c > 0$，有 $\Pr[X \geq c\mathbb{E}[X]] \leq \frac{1}{c}$.

$$\mathbb{E}\{\mathrm{CMS}[i][h_i(x)] - f_x\} \leq \frac{n}{b}$$

估测值　　真实值

分析在给定数值ε下 $\Pr[\mathrm{CMS}[i][h_i(x)] - f_x \geq \varepsilon n]$

令$c\frac{n}{b} = \varepsilon n$，即$c = \varepsilon b$

因为$\mathrm{CMS}[i][h_i(x)] - f_x$是非负数，有

$$\Pr[\mathrm{CMS}[i][h_i(x)] - f_x \geq \varepsilon b\mathbb{E}\{\mathrm{CMS}[i][h_i(x)] - f_x\}] \leq \frac{1}{\varepsilon b}$$

$$\Pr\left[\mathrm{CMS}[i][h_i(x)] - f_x \geq \varepsilon b\frac{n}{b}\right] \leq \Pr[\mathrm{CMS}[i][h_i(x)] - f_x \geq \varepsilon b\mathbb{E}\{\mathrm{CMS}[i][h_i(x)] - f_x\}] \leq \frac{1}{\varepsilon b}$$

即得到 $\Pr[\mathrm{CMS}[i][h_i(x)] - f_x \geq \varepsilon n] \leq \frac{1}{\varepsilon b}$

CM Sketch方法

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。

$$\Pr[\mathrm{CMS}[i][h_i(x)]-f_x\geq \varepsilon n]\leq \frac{1}{\varepsilon b}$$

CM Sketch方法

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。

$$\Pr[\mathrm{CMS}[i][h_i(x)]-f_x\geq\varepsilon n]\leq\frac{1}{\varepsilon b}$$

这仅是第i个哈希函数下 $\mathrm{CMS}[i][h_i(x)]$ 与 f_x 的近似程度

CM Sketch方法用$\min_i \mathrm{CMS}[i][h_i(x)]$近似$f_x$

需要分析 $\Pr\left[\min_i \mathrm{CMS}[i][h_i(x)]-f_x\geq\varepsilon n\right]$

CM Sketch方法

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。

由$\Pr[\mathrm{CMS}[i][h_i(x)] \geq \varepsilon n + f_x] \leq \frac{1}{\varepsilon b}$可得

$$\Pr\left[\min_i \mathrm{CMS}[i][h_i(x)] \geq \varepsilon n + f_x\right] = \prod_i \Pr[\mathrm{CMS}[i][h_i(x)] \geq \varepsilon n + f_x] \leq \left(\frac{1}{\varepsilon b}\right)^l$$

独立事件（要求l个哈希函数是独立的）

CM Sketch方法

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。

$$\Pr\left[\min_i \mathrm{CMS}[i][h_i(x)] \geq \varepsilon n + f_x\right] \leq \left(\frac{1}{\varepsilon b}\right)^l \Rightarrow \Pr\left[\min_i \mathrm{CMS}[i][h_i(x)] < \varepsilon n + f_x\right] \geq 1 - \left(\frac{1}{\varepsilon b}\right)^l$$

检查要求一：对于在数组A中出现次数大于等于$\frac{n}{k}$的元素x，$\min_i \mathrm{CMS}[i][h_i(x)] \geq \frac{n}{k}$，即不会遗漏

检查要求二：对于满足$\min_i \mathrm{CMS}[i][h_i(x)] \geq \frac{n}{k}$的元素$x$，

以大概率满足要求二

$$\Pr\left[f_x > \frac{n}{k} - \varepsilon n\right] = \Pr\left[\frac{n}{k} - f_x < \varepsilon n\right] \geq \Pr\left[\min_i \mathrm{CMS}[i][h_i(x)] - f_x < \varepsilon n\right] \geq 1 - \left(\frac{1}{\varepsilon b}\right)^l$$

CM Sketch方法

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k} - \varepsilon n$。

CM Sketch方法完全满足要求一，以至少$1 - \left(\frac{1}{\varepsilon b}\right)^{l}$的概率满足要求二

例如，要求$\varepsilon = 0.01$，并且以0.99的概率满足条件二，那么应该如何选取CMS大小（b和l）？

CM Sketch方法

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k} - \varepsilon n$。

CM Sketch方法完全满足要求一，以至少$1 - \left(\frac{1}{\varepsilon b}\right)^{l}$的概率满足要求二

例如，要求$\varepsilon = 0.01$，并且以0.99的概率满足条件二，那么应该如何选取CMS大小（b和l）？

$$\left(\frac{1}{0.01\ b}\right)^{l} = 0.01$$

可以取$b = 200$，$l = 6.644$。即7个哈希函数、每个函数映射到$\{0,1,\ldots,199\}$即可以达到要求

整个CMS数组占用存储空间大小为1400个单位

n可以为10^4, 10^8, 10^{12}, ...

占用空间大小不受数组A的规模n影响！

CM Sketch方法

CM Sketch方法的应用

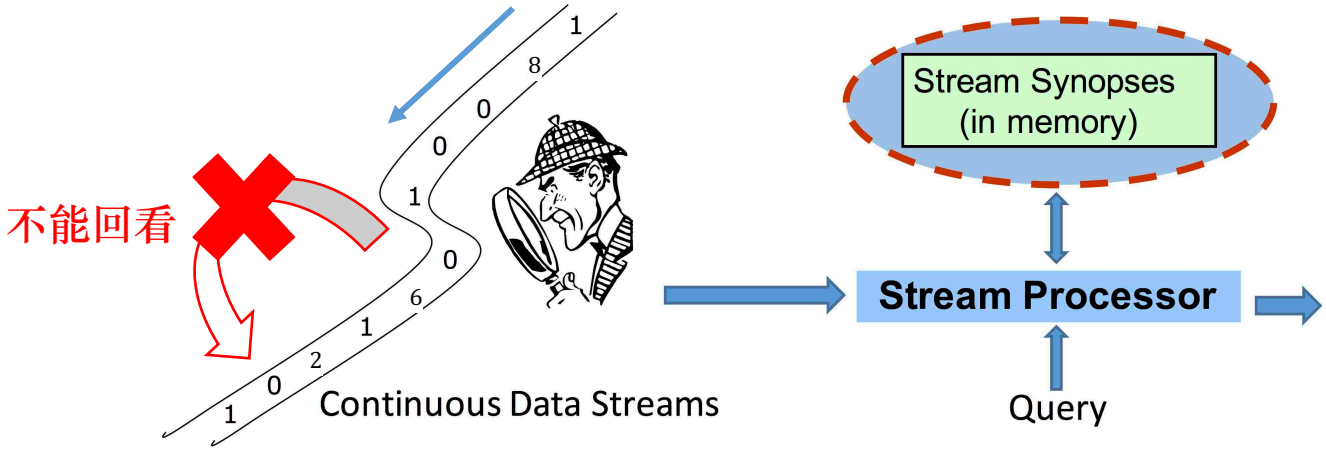
CM Sketch的应用

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。



面对数据流，如何输出满足要求的元素?

不能等数组A完全输入、得到完整的CMS后，再依次检查数组A中的元素 空间复杂度$O(n)$

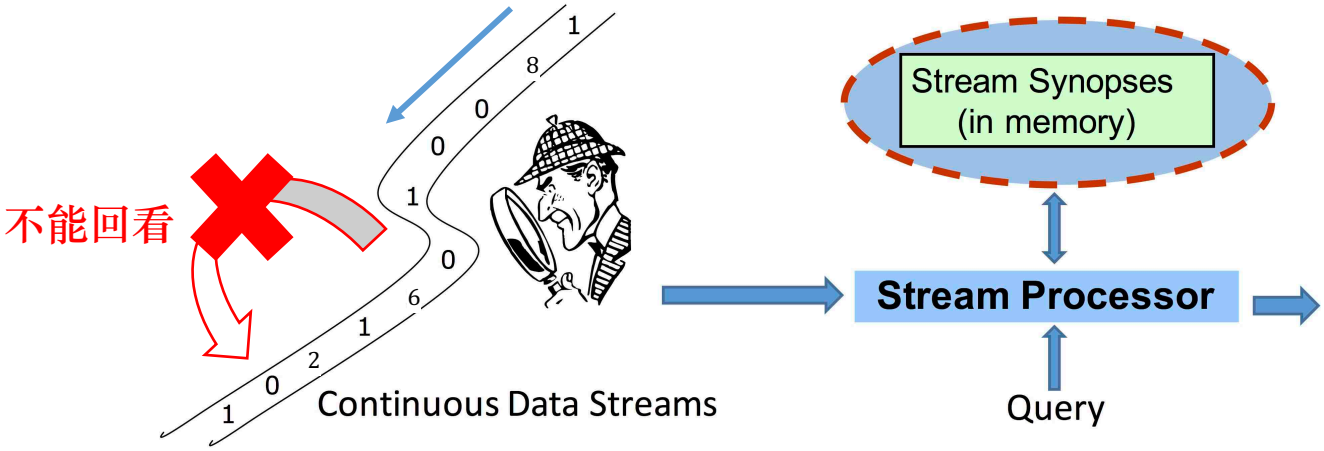
CM Sketch的应用

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。



面对数据流，如何输出满足要求的元素？

数组A元素依次输入、CMS实时更新、若当前输入的元素x对应的$\min_i \mathrm{CMS}[i][h_i(x)] \geq \frac{n}{k}$则输出$x$

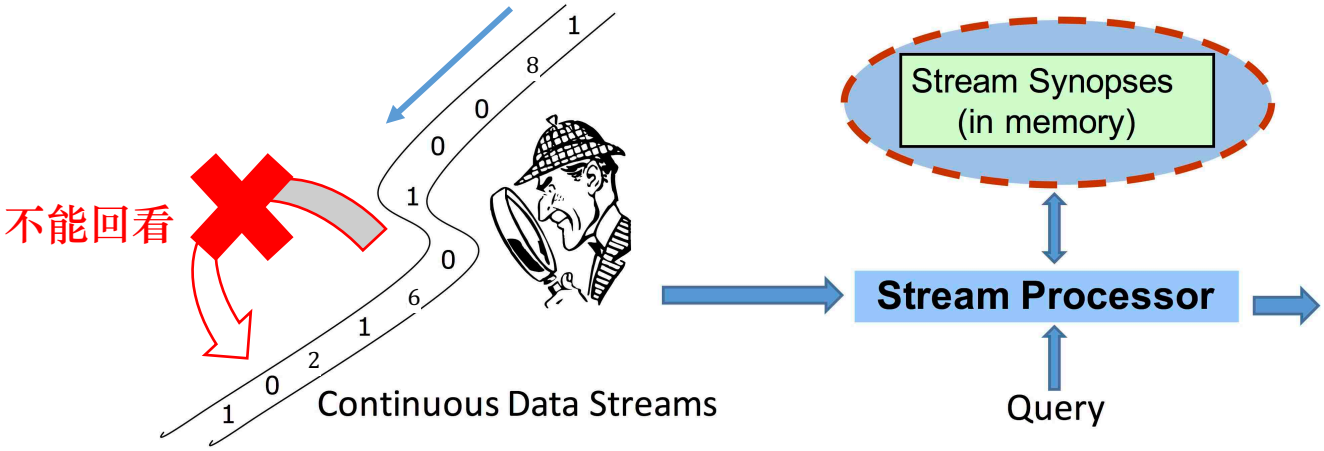
CM Sketch的应用

近似高频元素寻找问题（Approximate Heavy Hitters Problem）

给定长度为n的数组A、数值k、数值ε，如何输出一列元素并满足：

（1）该列元素包含了所有在数组A中出现次数大于等于$\frac{n}{k}$的元素；

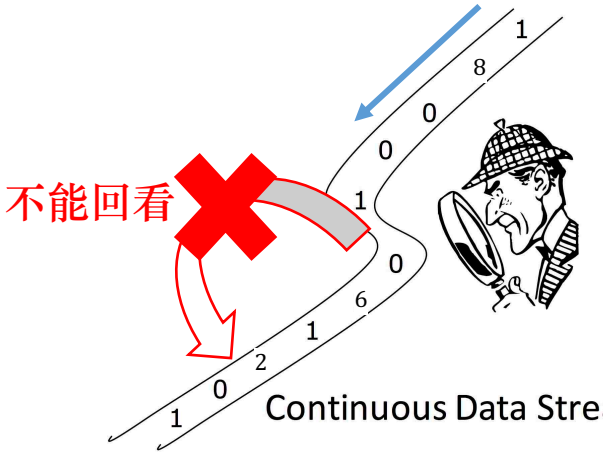
（2）该列的所有元素在数组A中出现次数大于$\frac{n}{k}-\varepsilon n$。



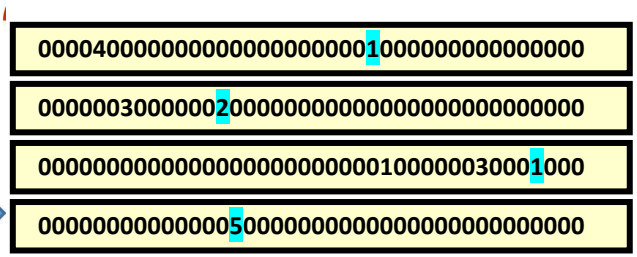
面对数据流，如何输出满足要求的元素？ 如果n事先未知怎么办？如无法预知当天用户浏览总量

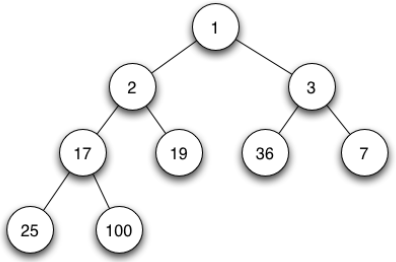
数组A元素依次输入、CMS实时更新、若当前输入的元素x对应的$\min_i \mathrm{CMS}[i][h_i(x)] \geq \frac{n}{k}$则输出$x$

CM Sketch的应用



如果n事先未知怎么办？如无法预知当天用户浏览总量





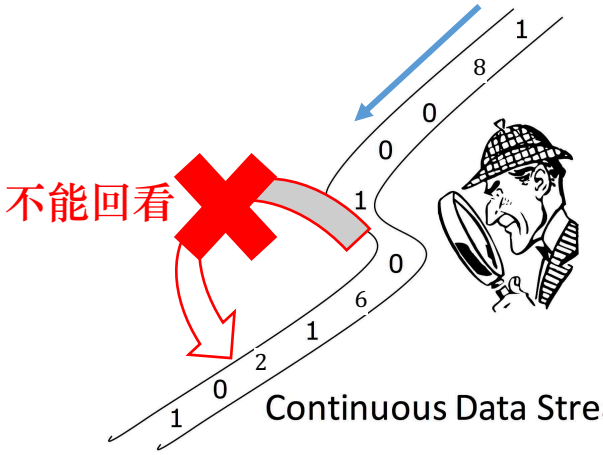
设已输m个元素，用堆记录潜在近似高频元素

第一步（判断当前输入元素x），若x满足$\min_i \mathrm{CMS}[i][h_i(x)] \geq \frac{m}{k}$

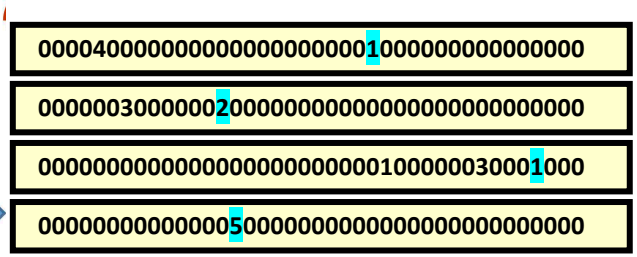
（1）如果x已经存在堆中，删去堆中原有x，插入新的x（值为当前的$\min_i \mathrm{CMS}[i][h_i(x)]$）

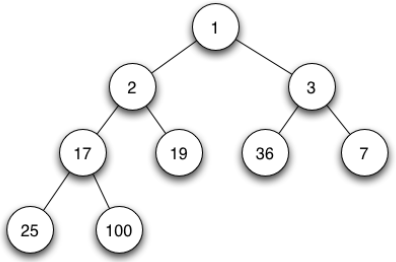
（2）如果x不在堆中，插入x，其值记为当前的$\min_i \mathrm{CMS}[i][h_i(x)]$

CM Sketch的应用



如果n事先未知怎么办？如无法预知当天用户浏览总量



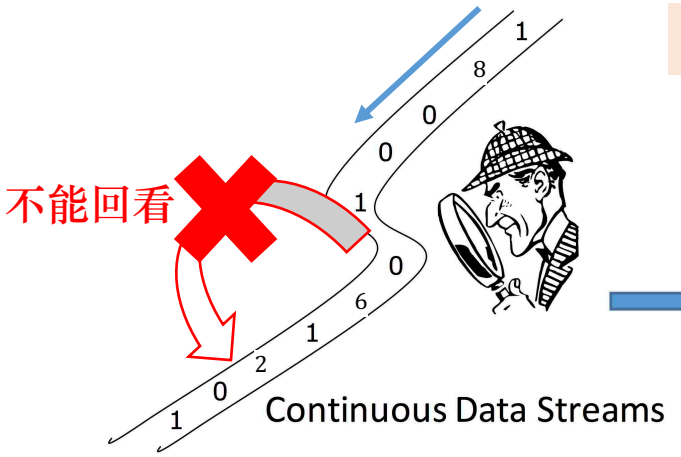


设已输m个元素，用堆记录潜在近似高频元素

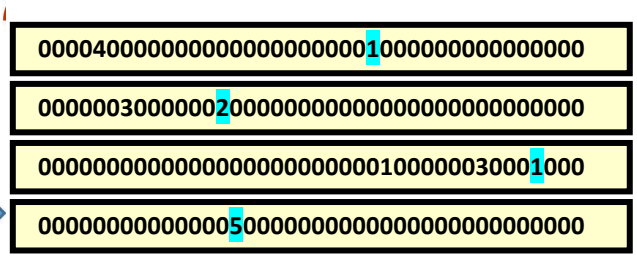
第二步（删去堆中当前m值下的低频元素），若堆中已有的元素存在记录值小于$\frac{m}{k}$的情况：

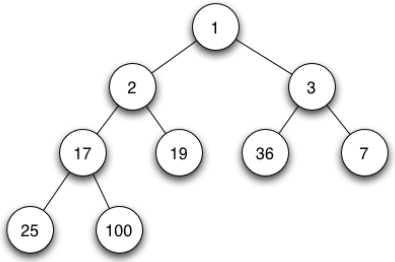
删去相应元素

CM Sketch的应用



如果n事先未知怎么办？如无法预知当天用户浏览总量





设已输m个元素，用堆记录潜在近似高频元素

第一步（判断当前输入元素x）

第二步（删去堆中当前m值下的低频元素）

当所有n个元素输入完，将堆中存储的元素输出

本讲小结

- 最高频元素寻找问题、（近似）高频元素寻找问题
- CM Sketch的方法、分析及应用

计算机学院
余皓然

主要参考资料

Tim Roughgarden and Gregory Valiant <CS 168 - The Modern Algorithmic Toolbox> Lecture Notes

Cameron Musco <COMPSCI 514 - Algorithms for Data Science> Slides

Baeldung <Find the Majority Element of an Array> Article

DataGenetics <Heavy Hitters Algorithm> Article

S. Raskhodnikova and A. Smith <Scalable, Private Algorithms for Continual Data Analysis > Webpage

Wikipedia <Big O notation> Webpage

计算机学院
余皓然

2023/4/25

谢谢！

计算机学院
余皓然
2023/4/25