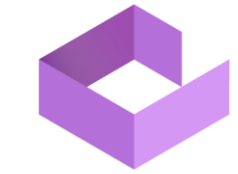
BDMI-01510243



Big Data and Machine Intelligence

AI system lab

iCenter

Tsinghua University

清華大學
Tsinghua University

HyperLogLog

讲解与样例

汇报人：李港新

基础概念

Background And Purpose

●○○○○

1. cardinality （基数、势）： 是指一个集合（这里的集合允许存在重复元素，与集合论对集合严格的定义略有不同，如不做特殊说明，本文中提到的集合均允许存在重复元素）中不同元素的个数。

A={1,2,3,4,5,2,3,9,7}

9个元素

基数为7

2.基数应用：

假设一个淘宝网店在其店铺首页放置了10个宝贝链接，分别从Item01到Item10为这十个链接编号。店主希望可以在一天中随时查看从今天零点开始到目前这十个宝贝链接分别被多少个独立访客点击过。所谓独立访客（Unique Visitor，简称UV）是指有多少个自然人，例如，即使我今天点了五次Item01，我对Item01的UV贡献也是1，而不是5。

区分要点：

1、对独立访客做标识

2、在访客点击链接时记录下链接编号及访客标记

3、对每一个要统计的链接维护一个数据结构和一个当前UV值，当某个链接发生一次点击时，能迅速定位此用户在今天是否已经点过此链接，如果没有则此链接的UV增加1

UV(独立访客)：即Unique Visitor,访问您网站的一台电脑客户端（User-Agent）为一个访客。00:00-24:00内相同的客户端只被计算一次。

几何平均数/调和平均数

$$G = \sqrt[n]{X_1 \times X_2 \times \cdots \times X_n} = \sqrt[n]{\prod_{i=1}^{N} X_i}$$

1、几何平均数受极端值的影响较算术平均数小。
2、如果变量值有负值，计算出的几何平均数就会成为负数或虚数。
3、它仅适用于具有等比或近似等比关系的数据。
4、几何平均数的对数是各变量值对数的算术平均数。
5、几何平均数是n个变量值连乘积的n次方根。
6、几何平均数多用于计算平均比率和平均速度。

$$H_n = \frac{1}{\frac{1}{n}\sum_{i=1}^{n}\frac{1}{x_i}} = \frac{n}{\sum_{i=1}^{n}\frac{1}{x_i}}$$

能干啥?

$$R_{并} = \frac{R_1 + R_2}{R_1 R_2}$$

1、调和平均数又称倒数平均数，是变量倒数的算术平均数和的倒数
2、调和平均数易受极端值的影响，且受极小值的影响比受极大值的影响更大。
3、只要有一个变量值为零，就不能计算调和平均数。
4、当组距数列有开口组时，其组中值即使按相邻组距计算了，假定性也很大，这时，调和平均数的代表性就很不可靠。
5、调和平均数应用的范围较小

基数估计的普遍方法：

算法	哈希函数	空间复杂度	准确度
Linear Counting	murmurhash32	$O(N_{max})$	完全
LogLog Counting	murmurhash32	$O(\log_2(\log_2(N_{max})))$	较高（大数据）
Adaptive Counting	murmurhash32	$O(\log_2(\log_2(N_{max})))$	较高
HyperLogLog Counting	murmurhash32	$O(\log_2(\log_2(N_{max})))$	较高
HyerLogLog++ Counting	murmurhash64	$O(\log_2(\log_2(N_{max})))$	较高

HyperLogLog:

Let $h : \mathcal{D} \to [0,1] \equiv \{0,1\}^\infty$ *hash data from domain* $\mathcal{D}$ *to the binary domain.*
Let $\rho(s)$*, for* $s \in \{0,1\}^\infty$*, be the position of the leftmost 1-bit* ($\rho(0001\cdots) = 4$).

Algorithm HYPERLOGLOG (**input** $\mathcal{M}$: multiset of items from domain $\mathcal{D}$).
assume $m = 2^b$ with $b \in \mathbb{Z}_{>0}$;
initialize a collection of m registers, $M[1], \ldots, M[m]$, to $-\infty$;

for $v \in \mathcal{M}$ **do**
  set $x := h(v)$;
  set $j = 1 + \langle x_1 x_2 \cdots x_b \rangle_2$; {*the binary address determined by the first* b *bits of* x}
  set $w := x_{b+1} x_{b+2} \cdots$; **set** $M[j] := \max(M[j], \rho(w))$;

compute $Z := \left(\sum_{j=1}^{m} 2^{-M[j]} \right)^{-1}$; {*the "indicator" function*}

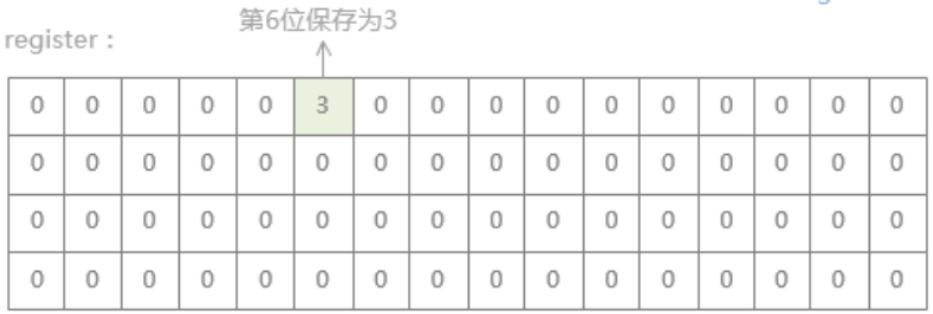
return $E := \alpha_m m^2 Z$ with α_m as given by Equation (3). $\alpha_m := \left(m \int_0^\infty \left(\log_2 \left(\frac{2+u}{1+u} \right) \right)^m du \right)^{-1}$

过程

添加元素过程

如果对添加元素过程还不太理解可以看这张图：

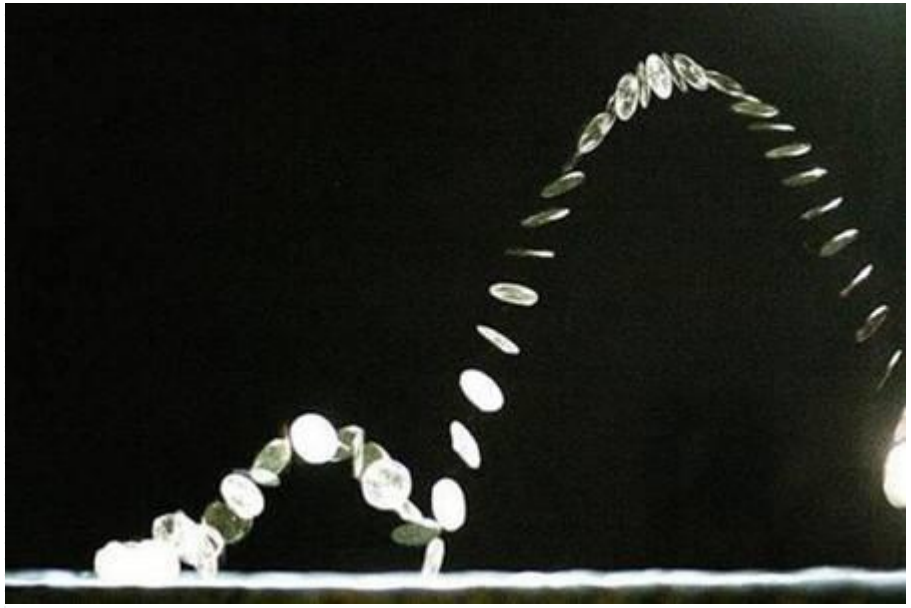
输入值：12125880　哈希值：1927385862（二进制：11100101110000110010011000000110）



1. 设 `b=6`，则桶数 `m=2^6=64`，输入元素值为 `12125880`，其哈希值为 `1927385862`
2. 将哈希值转换成二进制：`11100101110000110010011000000110`
3. 计算桶索引：取二进制最右边 6 位 `000110` 转换为十进制等于 `6`（和数据库中分库分表的取余法类似，平均分配到桶中）
4. 计算桶数值：从二进制最右边第 7 位开始，从右往左首次出现1经过的位数，数值等于 `3`
5. 将第 `6` 桶的值设置为 `3`（若第 6 桶原有值大于 3 则不设置），完成元素添加

伯努利过程

硬币



P(正面)=P(反面)=0.5

概率

小明：在我扔硬币的这几次过程中，我有一次连续扔出了两个反面，才扔出正面

001

扔了8次过程

疑问

为什么

基数估计定义规定：

整个集合中必须有不同的元素，
否则直接按照无处理
即000000视为不存在

凭什么

- n次伯努利过程的投掷次数都不大于k*max*
- n次伯努利过程，至少有一次投掷次数等于k*max*

对于第一个结论，n次伯努利过程的抛掷次数都不大于k_{max}的概率用数学公式表示为：

$$P_n(X \le k_{max}) = (1 - 1/2^{k_{max}})^n$$

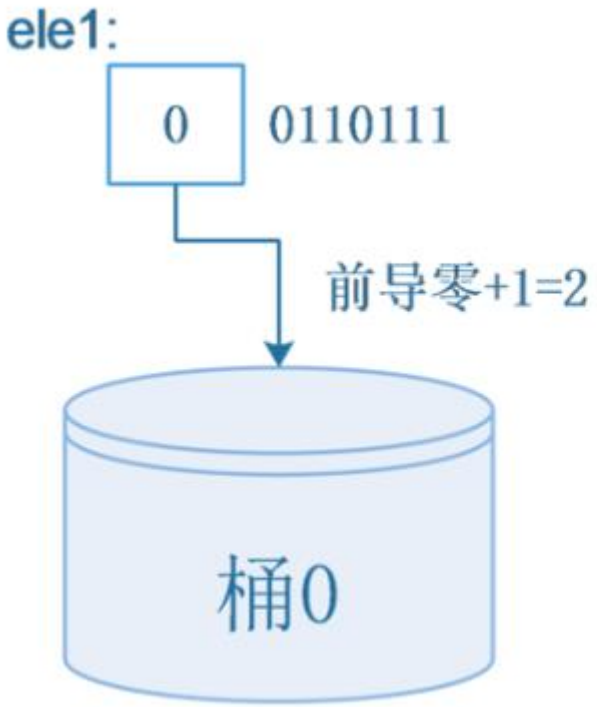
第二个结论至少有一次等于k_{max}的概率用数学公式表示为：

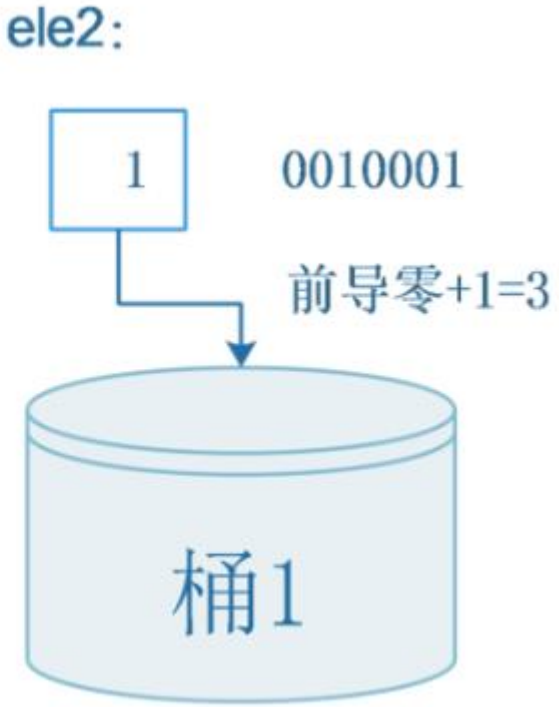
$$P_n(X \ge k_{max}) = 1 - (1 - 1/2^{k_{max}-1})^n$$

当$n \ll 2^{k_{max}}$时，$P_n(X \ge k_{max}) \approx 0$，即当$n$远小于$2^{k_{max}}$时，上述第一条结论不成立；

当$n \gg 2^{k_{max}}$时，$P_n(X \le k_{max}) \approx 0$，即当$n$远大于$2^{k_{max}}$时，上述第二条结论不成立。 因此，我们似乎就可以用$2^{k_{max}}$的值来估计n的大小。

以上结论可以总结为：进行了n次进行抛硬币实验，每次分别记录下第一次抛到正面的抛掷次数k，那么可以用n次实验中最大的抛掷次数k_{max}来预估实验组数量n：$\hat{n} = 2^{k_{max}}$





算术平均

基数估计：$2 \times 2^{\frac{2+3}{2}} = 11$

调和平均

基数估计：$2 \times \frac{2}{\frac{1}{2}+\frac{1}{3}} = 4.8$

$$DV_{HLL} = const * m * \left(\frac{m}{\sum_{j=1}^{m} \frac{1}{2^{R_j}}} \right)$$

每个桶的估计值

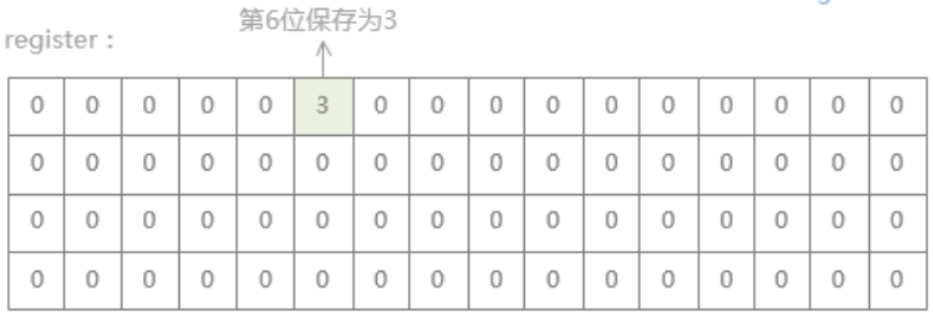
所有桶估计值的调和平均数

过程

添加元素过程

如果对添加元素过程还不太理解可以看这张图：

输入值：12125880　哈希值：1927385862（二进制：11100101110000110010011000000110）



1. 设 `b=6`，则桶数 `m=2^6=64`，输入元素值为 `12125880`，其哈希值为 `1927385862`
2. 将哈希值转换成二进制：`11100101110000110010011000000110`
3. 计算桶索引：取二进制最右边 6 位 `000110` 转换为十进制等于 `6`（和数据库中分库分表的取余法类似，平均分配到桶中）
4. 计算桶数值：从二进制最右边第 7 位开始，从右往左首次出现1经过的位数，数值等于 `3`
5. 将第 `6` 桶的值设置为 `3`（若第 6 桶原有值大于 3 则不设置），完成元素添加

HyperLogLog:

Let $h : \mathcal{D} \to [0,1] \equiv \{0,1\}^\infty$ *hash data from domain* $\mathcal{D}$ *to the binary domain.*
Let $\rho(s)$*, for* $s \in \{0,1\}^\infty$*, be the position of the leftmost 1-bit* ($\rho(0001\cdots) = 4$).

Algorithm HYPERLOGLOG (**input** $\mathcal{M}$: multiset of items from domain $\mathcal{D}$).
assume $m = 2^b$ with $b \in \mathbb{Z}_{>0}$;
initialize a collection of m registers, $M[1], \ldots, M[m]$, to $-\infty$;

for $v \in \mathcal{M}$ **do**
  set $x := h(v)$;
  set $j = 1 + \langle x_1 x_2 \cdots x_b \rangle_2$; {*the binary address determined by the first* b *bits of* x}
  set $w := x_{b+1} x_{b+2} \cdots$; **set** $M[j] := \max(M[j], \rho(w))$;

compute $Z := \left(\sum_{j=1}^{m} 2^{-M[j]} \right)^{-1}$; {*the "indicator" function*}

return $E := \alpha_m m^2 Z$ with α_m as given by Equation (3). $\alpha_m := \left(m \int_0^\infty \left(\log_2 \left(\frac{2+u}{1+u} \right) \right)^m du \right)^{-1}$

$$DV_{HLL} = const * m * \left(\frac{m}{\sum_{j=1}^{m} \frac{1}{2^{R_j}}} \right)$$

每个桶的估计值

所有桶估计值的调和平均数

$$RSD = \frac{1.04}{\sqrt{m}}$$

RSD

有了这个公式，你可以先确定你想要达到的RSD的值，然后再推出分桶的数目m。

CONSTANT

constant常数的选择与分桶的数目有关，具体的数学证明请看论文，这里就直接给出结论：

假设：m为分桶数，p是m的以2为底的对数

$$p = \log_2 m$$

p

```
switch (p) {
case 4: constant = 0.673 * m * m;
case 5: constant = 0.697 * m * m;
case 6: constant = 0.709 * m * m;
default: constant = (0.7213 / (1 + 1.079 / m)) * m * m; }
```

微调

当数据量很小的时候，会出现很多空桶

```
if DV < (5 / 2) * m:
        DV = m * log(m/V)
```

来表示当前基数的数量

Thanks!