基于概率的数据库计数算法

陈嘉杰
计 72 2017011484

2018 年 6 月

摘要

随着时代的发展，数据库中的数据量快速增长，远远超出了单台机器的存储和计算能力，因此，现代的数据库大多采用分布式存储数据的方法，但这也为一些基本的数据库操作添加了困难。本文主要讨论在分布式数据库上，如何实现一个基于概率的计数方法，它对一个存储在分布于不同机器上的一个列表进行计数，得出该列表中相异元素个数的一个估计。由于数据量十分巨大，原本的将所有数据集中在一台机器上进行排序和去重的方案不再现实。但在一些现实需求下，可以牺牲一定的准确性，在较短的时间内得到一个较好的估计。本文对解决该问题的不同算法进行了介绍和评估。

关键词：算法，哈希，数据库，计数算法

1 引言

获取数据库中一个列中所有数据中相异元素的个数在许多地方有应用。通过获取这个数据，可以反映出数据的一些特征，针对该特征，数据库开发者可以根据这个特性进行优化。[3] 在数学上，由于数据库分布式地存储在不同的机器上，这个问题可以转化为：

1. 记我们要统计的所有数据的多重集合为 M 。存储在机器 i 上为多重集合 M 的多重子集 M_i 。
2. 在每台机器上分别对 M_i 进行处理，将处理结果统一到一台机器，在该机器上对 $|M|$ 进行估计，即该多重集合中相异元素的个数 n ，记估计值为 $\hat{n}$。

容易想出解决这个问题的算法，假设只有两个多重子集 A B [3]：

1. 对 A 进行排序，然后对 B 中每一个元素在 A 中进行二分查找，将两个集合中重复的元素去掉
2. 对 A 和 B 分别进行排序，然后进行合并，遇到相同元素时跳过
3. 对 A 和 B 进行哈希等预处理，简化相同元素的比较过程，再采用以上方案

以上这些方案也可以推广到更多的多重子集，但他们的空间复杂度都较高。当数据量极其大，无法在单台机器中容纳时，以上的精确计数方案不在适用。但在允许一定错误率的情况下，通过后文将提到的 HyperLogLog 算法，可以大大减少空间要求，如对一个 10^9 个元素的计数，在 2% 的错误率下也只需要 1.5KB 的内存空间。[2]

本文将介绍几个解决该问题的算法，并对他们进行比较和分析。

2 正文

2.1 前序知识

由于数据本身结构的复杂性，通过直接比较数据是否相同是一个十分低效的过程，并且由于数据本身的特性十分不同，很难设计出一个适用于不同数据类型的普适算法，而且也无法保证数据有一定的概率分布的性质。为此，我们使用哈希函数对数据进行预处理：

定义 2.1.1 (哈希函数) *哈希函数是一个任意数据到 $[0 \cdots 2^L - 1]$ 的映射，其中 L 为哈希的位长，是定值。*

性质 2.1.1 (哈希函数) *哈希函数将任意数据等可能地映射到 $[0 \cdots 2^L - 1]$，每一个二进制位上为 1 和 0 的概率相同*

性质 2.1.2 (哈希函数) *对数据的微小改变，会导致哈希值的巨大变化。反过来说，如果两个数据的哈希值相同，那么几乎可以认为这两个数据相同。*

通过将数据每一个函数进行哈希，我们得到了一个新的多重集合，其中重复的数据，我们可以认为对应着原来数据的重复。因此，统计出新的这个集合的相异元素个数，即得原集合中相异元素的个数。

常见的一些哈希函数如下：

哈希函数	位长（L/bit）
MD5	128
RIPEMD-160	160
SHA-1	160
SHA-256	256
SHA-512	512

通过这一步操作，我们获得了便于处理的二进制数据，也提供了一些概率上的性质。

2.2 PC（Probablisitic Counting）算法 [1]

1985 年，Philippe Flajolet 提出了基于概率的简易计数算法，我们称之为 PC（Probablistic Counting）算法，并通过随机平均（stochastic averaging）提出了改进，为 PCSA（Probablistic Counting with Stochastic Averaging），允许根据需要增加计算量，从而缩小估计值的标准差。

首先定义函数 $\mathrm{bit}(y,k)$ 和 $\rho(y)$ ：

定义 2.2.1 ($\mathrm{bit}(y,k)$) 定义 $\mathrm{bit}(y,k)$ 为 y 的二进制表示中，从右向左第 k 位（从 0 开始）的二进制。

引理 2.2.1

$$y=\sum_{k\geq 0}\mathrm{bit}(y,k)2^k$$

定义 2.2.2 ($\rho(y)$)

$$\begin{aligned}\mathrm{rho}(y)&=\min_{k>=0}\mathrm{bit(y,k)}\neq 0 && \text{if}\quad y>0\\ &=L && \text{if}\quad y=0.\end{aligned}$$

可以观察到，由于哈希的均匀分布，我们知道 $\rho(y)$ 为指数分布。对不同的 $\rho(y)$ 进行统计，我们可以得到 PC 算法。

由于哈希函数的分布特性，$BITMAP[0]$ 被更新的期望为 n, $BITMAP[1]$ 被更新的期望为 $n/2$，故我们采用 $BITMAP$ 中最左边一个为 0 的位的位置来估计 $\log_2(n)$ ，并且进行常数的修正，作为 n 的估计值。

我们不加证明地给出，$\sigma(R)\approx 1.12$ 。详细证明见 [1] 。

input : the multiset M whose cardinality is sought
output the estimated $\hat{n}$
:
for $i \leftarrow 0$ **to** $L-1$ **do**
  BITMAP $[i] \leftarrow 0$;
end
for *all* x *in* M **do**
  $index \Leftarrow \rho(\text{hash}(x))$;
  if BITMAP $[index] = 0$ **then**
    BITMAP $[index] \leftarrow 1$;
  end
end
$n \leftarrow$ the position of the leftmost zero in BITMAP;
$\phi \leftarrow 0.77351\cdots$;
$\hat{n} \leftarrow \log_2(\phi n)$;

Algorithm 1: Probablistic Counting

2.2.1 PCSA（Probablistic Counting Stochastic Averaging）算法

为了进一步缩小标准差，使得我们的估计能够更加接近精确值，通过采用不同的哈希函数，分别求出对应的 $\hat{n}_i$ ，并且取 $\hat{n} = \frac{\sum_{i=1}^{m} \hat{n}_i}{m}$ 为估计值，那么，标准差变为 $\sigma(R) \approx \frac{1.12}{\sqrt{m}}$ 。这样，通过 Stochastic Averaging（随机平均）的方法，我们可以通过改变 m 使得我们可以进一步控制估计的标准差。

引用文献

[1] Philippe Flajolet and G Nigel Martin. “Probabilistic counting algorithms for data base applications”. In: *Journal of Computer and System Sciences* 31.2 (1985), pp. 182–209.

[2] Philippe Flajolet et al. “Hyperloglog: The analysis of a near-optimal cardinality estimation algorithm”. In: *Conference on Analysis of Algorithms* (2007).

[3] Kyu-Young Whang, Brad T Vander Zanden, and Howard M Taylor. “A Linear-Time Probabilistic Counting Algorithm for Database Applications.” In: *ACM Trans. Database Syst.* 15.2 (1990), pp. 208–229.