•信息工程•

DOI:10.15961/j.jsuese.201601076

云数据库中等宽直方图的分布式并行构造方法

王　阳[1,2], 钟　勇[1,2], 周渭博[1,2], 杨观赐[3*]

(1.中国科学院 成都计算机应用研究所, 四川 成都 610041; 2.中国科学院大学, 北京 100049;
3.贵州大学 现代制造技术教育部重点实验室, 贵州 贵阳 550003)

摘　要:直方图能够直观描述数据分布,在数据库查询优化中起着重要作用。然而在分布式云数据库场景中,现有直方图构建方法存在并行资源利用率低,网络传输量较高的问题。针对该问题,基于关系型云数据库提出一种等宽直方图的分布式并行构造方法。首先,根据集群中分布式存储的数据无关性,基于master-slave架构在直方图任务开始前由集群中请求发起节点对经RPC(remote procedure call)协议获取到的多个工作节点最值数据比较得到数据表在整个集群的全局最大值、最小值;然后,考虑到算法运行过程中数据传输量的优化,集群中工作节点对本地数据扫描、排序,划分至依据全局最值信息构建的直方图桶内,实现聚合子直方图的并行构建以提高集群计算资源利用率;最后,请求发起节点对并行构建的多个子直方图中边界值相等的桶频率值聚合得到全局直方图。算法利用分布式思想实现了关系型云数据库中直方图的构建,将计算任务划分成多个子任务并行执行,子直方图信息代替数据分片的传输大幅优化了网络带宽的负载。算法已应用于关系型云数据库内核以优化SQL语句执行路径的初始扫描开销、数据选择率等关键参数。人工合成数据与评分数据的实验结果证明,算法运行过程中的网络传输量与数据库表中元组个数无关,且具有良好的可拓展性。

关键词:关系型云数据库;等宽直方图;数据分布;并行计算;网络传输量

中图分类号:TP311.133.1　　**文献标志码**:A　　**文章编号**:2096-3246(2018)02-0133-08

Distributed and Parallel Construction Method of Equi-width Histogram in Cloud Database

WANG Yang[1,2], *ZHONG Yong*[1,2], *ZHOU Weibo*[1,2], *YANG Guanci*[3*]

(1.Chengdu Inst. of Computer Applications,Chinese Academy of Sciences,Chengdu 610041,China; 2.Univ. of Chinese Academy of Sciences,Beijing 100049,China; 3.Key Lab. of Advanced Manufacturing Technol.,Ministry of Education,Guizhou Univ.,Guiyang 550003,China)

Abstract: Description of data distribution has been commonly used in databases to support query optimization,among which histograms are of particular interest.The existing conventional histogram construction methods have the problem of low efficiency of parallel resource utilization and high data transmission.To address these issues,a distributed and parallel constructing method was proposed for equi-width histogram in relational cloud database.Since the data ranges of different storage nodes are not the same,firstly,the local maximum and minimum values of working nodes were transferred to the application request node using the RPC protocol.These values were compared with each other to get the global maximum and minimum values based on master-slave model before the start of the histogram construction.Secondly,considering the optimization of data transmission during the histogram estimation,each working node scanned,sorted and partitioned the data into buckets according to the global maximum and minimum values which were transmitted from the application request node.Sub-histograms that were constructed in parallel improved the efficiency of resource utilization in the cluster.Finally,sub-histograms were aggregated to obtain the global histogram in the application request node.The algorithm divided the histogram task into multiple small tasks that could be simultaneously executed in a distributed cluster.During the histogram estimation,only a small portion of information about buckets and a few necessary data need to be transmitted over the network.The algorithm has been applied to the relational cloud database to optimize the initial scanning cost of the SQL statement,the data selec-

收稿日期:2016 – 09 – 26

基金项目:国家自然科学基金资助项目(61640209;91746116);四川省科技创新苗子工程资助项目(SCMZ2006012);贵州省科技计划资助项目(黔科合JZ字[2014]2004号;黔科合人字(2015)13号)

作者简介:王　阳(1987—),男,博士生. 研究方向:分布式并行计算. E-mail: wangyang2014casit@outlook.com

* 通信联系人 E-mail: guanci_yang@163.com

网络出版时间:2018 – 03 – 20 16 : 21 : 49　　**网络出版地址**:http://kns.cnki.net/kcms/detail/51.1773.TB.20180320.1621.008.html

tion rate and other key parameters.The experimental results of the synthetic data and the real data demonstratedthat the amount of the data transmission was unrelated with the table size and the proposed algorithm achievedgood scalability.

Key words: velation cloud databased;equi-width histogram;data distribution;parallel computing;data transmission

直方图是数据统计分析中一种直观、简单的形式，是描述数据库中数据分布的流行且有效的方法。绝大多数商用数据库系统在关系上维护一个或多个直方图。例如Oracle数据库中，查询优化器应用直方图评估数据分布，以精确条件查询中数据选择率，从而优化查询操作。直方图在基于代价的查询优化、聚集近似查询、数据挖掘等领域有着广泛的应用[1]，在数据库查询优化处理中的重要作用已经得到广泛认识。

传统的关系型数据库领域，已有较多学者对直方图的构建进行了研究。Ioannidis[2]介绍了直方图的历史，并总结了直方图在传统数据库领域的应用。Poosala等[3]对比分析了不同类型的直方图及其构造方法。Chaudhuri等[4]采用元组采样的方法构建近似直方图，并给出了抽样数据大小与近似直方图准确性的精确关系。骆吉洲等[5]通过对查询缓冲池内查询的调度反馈提出了一种基于压缩数据库的直方图自适应构建方法。张龙波等[6]基于合并–分裂策略提出等深直方图的近似增量维护算法。Bruno等[7]针对多维数据进行模拟，为了避免估计结果集中采用数据独立性假设导致估计误差过大问题，提出了一种称为STHole的多维自适应直方图。Kanne等[8]中对存储桶边界值、频率的直方图和桶内添加新存储内容的直方图进行对比，为直方图桶的多样化提供了思路。传统数据库领域，数据集中存储于一个高性能的服务器，因此可以直接从单节点服务器读取数据执行不同类型直方图的构建。而云数据库中数据分布式存储于大规模集群中多个存储节点，存储方式的差别导致集中式数据库的直方图构建方法无法直接应用于云数据库。

随着大数据时代的到来，一些学者开始研究MapReduce架构下直方图的构建算法。Jestes等[9]利用元组抽样方法提出基于MapReduce的小波直方图构建算法。Tang[10]分别基于值模型和元组模型的语义期望介绍了MapReduce架构中V-Optimal类型的近似直方图构建方法。Shi等[11]对MapReduce架构进行了拓展，在Map阶段之前和Reduce阶段之后分别增加了数据采样和统计阶段，对基于MapReduce的等宽和等深直方图构建算法进行改进。针对数据流快速、时变、不可预测等特点，Guha[12]中提出了基于滑动窗口的实时数据流直方图的构建方法。基于MapReduce架构的直方图构建方法需要将数据经Mapper处理后转换为key-value对，将数据以key-value形式发送至Reducer节点进行直方图桶的构建，或将key-value对按照一定的概率在节点间传输，这会导致较高的网络传输量。

为了提高直方图构建方法的资源利用率并减少网络传输量，作者提出一种等宽直方图的分布式并行构造方法。通过对直方图类型、云数据库中分片存储、数据流转方式、任务的并行执行特性的分析，将直方图任务划分为能够直接合并的多个子任务下压至计算集群中工作节点并行执行；工作节点依据与请求发起节点提前一轮交互获得的最值信息独立的对本地数据扫描、排序、构建子直方图，直方图中每个桶仅包含桶边界值及桶内频率信息，请求发起节点与工作节点间直方图的传输避免了因分片数据传输导致的网络负载过重问题。

1 等宽直方图分布式并行构建方法

1.1 相关定义

不失一般性，假设一个数据库表T包含t个元组，表中数据分布式存储于关系型云数据库中的N个节点，表T包含属性A，A的值是整数或者实数，其值域为D，直方图相关定义如下：

定义1 设V为表T中属性A的值，对于D中的任意一个值x，用$f(x)$表示$A=x$在表T中出现的频率，定义$B=\{\langle B_{\mathrm{L}},B_{\mathrm{R}},f(B)\rangle,B_{\mathrm{L}},B_{\mathrm{R}}\in D\}$为直方图中的桶，$B_{\mathrm{L}}$为桶$B$的左边界值，$B_{\mathrm{R}}$为桶$B$的右边界值，$f(B)$为桶$B$中左边界至右边界范围内值的频率和。

定义2 表T属性A上一系列互不相交的桶$B_1,B_2,\cdots,B_m$构成的集合$H=\{B_i;i=1,2,\cdots,m\}$为关系T上属性A的直方图。

定义3 数据库表T的数据分布式存储于计算集群中N个节点，表T的属性A在各工作节点的最大值定义为Max_{L}，最小值定义为Min_{L}，表T属性A在整个集群的最大值定义为Max_{G}，最小值定义为Min_{G}。

1.2 等宽直方图的分布式并行构造思想

关系型云数据库的存储和HDFS（Hadoop distributed file system）的存储方式类似，虚谷云数据库中表以分片（Tablet）形式存储，每个数据分片保存1个主版本和2个副版本，数据片分布式存储于集群中不同节点上，数据库表的分布式存储方式如图1所示。需要指出的是，关系型云数据库中计算集群采用share-nothing体系架构，集群中任意节点都可以作为请求发起节点。

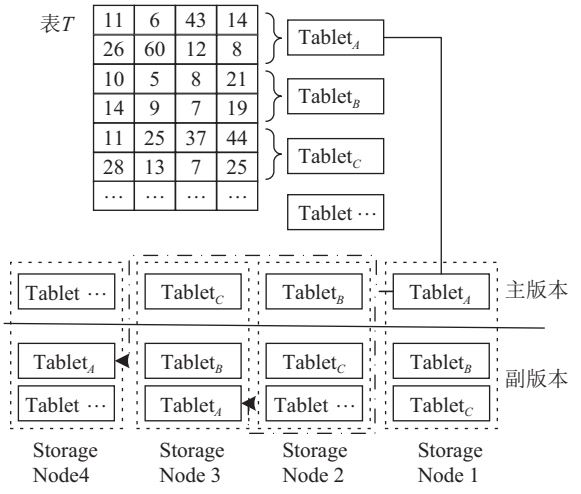


图 1　数据库表分布式存储方式

Fig.1　Distributed storage of table in database

由表的分布式存储方式可知$\forall \text{Tablet}_i \subseteq T, i \le n$, 且对于表$T$的所有同一版本分片满足:

$$\begin{aligned}&\text{Tablet}_i \cap \text{Tablet}_j = \varnothing,\ 0 \le \forall i, j \le n;\\ &\text{Tablet}_1 \cup \text{Tablet}_2 \cup \cdots \cup \text{Tablet}_n = T\end{aligned} \tag{1}$$

通过对不同类型直方图特征的分析, 等宽直方图具有桶边界值差相等的特点:

$$B_i.B_R - B_i.B_L = B_j.B_R - B_j.B_L,\ 1 \le \forall i, j \le m \tag{2}$$

若能够在分布式节点上构建具有相同边界值的子直方图, 则请求发起节点直接对子直方图聚合即可得到表T上相关属性的全局直方图, 且集群中不需要传输表中具体数据, 只需传输直方图桶的边界值信息和频率值信息。由数据库表数据存储方式结合第1.1节定义3可知$Max_L \le Max_G$, $Min_L \ge Min_G$。集群中请求发起节点接收到直方图构建请求后, 各工作节点只拥有本地最大值Max_L、最小值Min_L, 不同存储节点的数据范围可能没有任何关系, 因此各节点构造的子直方图间无规律可循, 也就无法在请求发起节点对工作节点的子直方图进行聚合。

一种解决方法是请求发起节点接收到直方图构建任务后, 各工作节点将本地存储分片数据发送至请求发起节点, 在请求发起节点统一对数据进行排序、分桶, 构造等宽直方图, 然而这种方法会导致极大的网络传输量, 并且大数据环境下, 在请求发起节点对数据进行统一处理在时间和空间上都是不切实际的。考虑到关系型云数据库中带宽资源是极其宝贵的资源, 直方图构建过程中降低网络传输量是较为理想的方案。数据库表中数据以分片形式分布式存储于集群的工作节点, 各工作节点间数据没有必然联系。基于等宽直方图特点, 使工作节点不按照本地存储的数据范围构建直方图, 而是依据相同的数据范围构建直方图可以满足既利用分布式并行计算的优势, 又不传输表中具体数据的需求。

基于关系型云数据库的等宽直方图分布式并行构造方法的基本思想是: 首先, 请求发起节点在接收到任务请求后, 请求发起节点与计算集群中工作节点间建立RPC通道, 将数据库表、属性名、直方图桶数等基本信息发送至相关工作节点, 各工作节点对本地存储数据扫描、排序, 得到本地节点的最大值Max_L、最小值Min_L; 然后, 各工作节点将本地的最值信息通过RPC协议传输至请求发起节点, 请求发起节点对各工作节点发送的最值信息再次进行排序, 得到全局最大值Max_G、最小值Min_G, 再将全局最值信息经RPC协议传送至工作节点, 各工作节点依据全局最值信息在本地构建等宽直方图; 最后, 只需将工作节点的直方图信息发送至请求发起节点, 请求发起节点依据直方图桶边界值和桶内频率直接累加构建全局等宽直方图。直方图构建过程中数据流转示意图如图2所示。

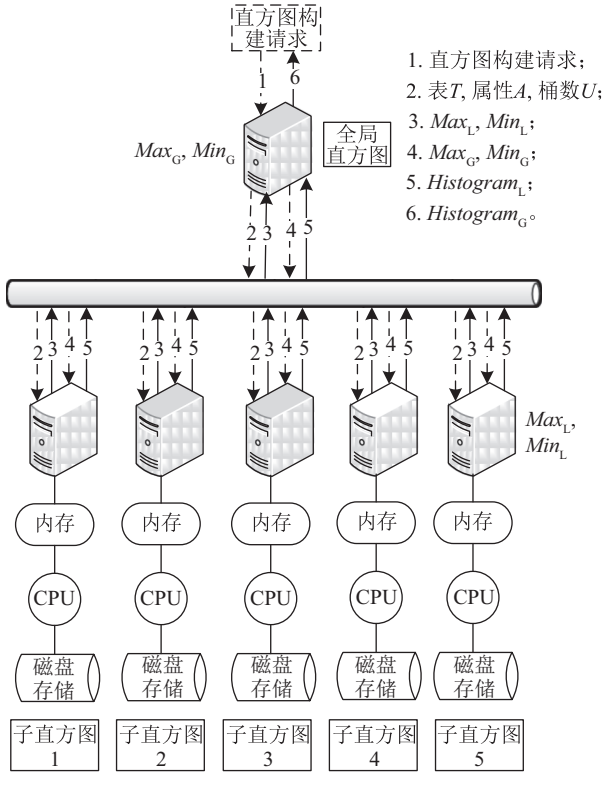


图 2　等宽直方图构造方法示意图

Fig.2　Process flow of constructing equi-width histogram

1.3　等宽直方图的分布式并行构造算法

基于上文所述, 提出了关系型云数据库中等宽直方图的分布式并行构造方法, 算法实现的伪代码为:

Algorithm DPMHistogram(T, A, U);

Input: Table T, Attribute A, Bucketnumber U

Output: Equi-width histogram

```
Begin
1 Exec SYSDBA.DBMS_STAT.ANALYZE_TABLE
2 ANALYZE_RPC_PIPE(T, A, U);
3 P_SORT(T, A);
4 RPC To MainNode(Max_L, Min_L);
5 For i = 1 to N
    If Max_G ≤ Max_Li Then
    Max_G = Max_Li;
    End If
    If Min_G ≥ Min_Li Then
    Min_G = Min_Li;
    End If
6 For i = 1 to N
    RPC To SubNode(Max_G, Min_G);
  End For
7 For i = 0 to U do
    B_i.B_L = Min_G + i×(Max_G − Min_G)/U;
    B_i.B_R = Min_G + (i+1)×(Max_G − Min_G)/U;
      If B_L ≤ V(x) ≤ B_R
        B_i.f(B)++;
  End For
8 RPC To MainNode(Histogram_L);
9 For i = 0 to U do
    For j = 0 to N do
      B_i.f(B) += B_j.f(B);
      j++;
    End For
    i++;
  End For
End ANALYZE_TABLE
```

云数据库中等宽直方图通过DBMS_STAT包的存储过程实现，存储过程名为ANALYZE_TABLE。

伪代码中步骤2为直方图的参数传递。计算集群中请求发起节点与工作节点间建立RPC通道，将需构建直方图的关系名T、属性A、直方图包含桶数U通过RPC协议发送至集群中相关工作节点。步骤3中，各工作节点对本地存储数据扫描、排序，得到本地最大值Max_{L}、最小值Min_{L}。

步骤4中，工作节点分别将本地最大值、最小值经RPC协议发送至集群中请求发起节点。步骤5中请求发起节点对各工作节点$Max_{\mathrm{L}1}$、$Min_{\mathrm{L}1}$，$Max_{\mathrm{L}2}$，$Max_{\mathrm{L}2}$，$\cdots$，$Max_{\mathrm{L}i}$、$Min_{\mathrm{L}i}$，$\cdots$，$Max_{\mathrm{L}N}$、$Min_{\mathrm{L}N}$进行比较，得到整个集群中属性A的全局最大值Max_{G}、最小值Min_{G}。于步骤6再次通过RPC协议将全局最值信息依次发送至各工作节点。

步骤7中，各工作节点依据全局最大值Max_{G}、最小值Min_{G}和直方图桶数U，基于排序后数据并行构造等宽直方图。

步骤8中，工作节点将直方图的左右边界值、桶内频率信息发送至请求发起节点。假设集群中第k个工作节点构建的子直方图用$H_{\mathrm{L}k}$表示：

$$H_{\mathrm{L}k}=\{\langle B_{1\mathrm{L}}^{k},B_{1\mathrm{R}}^{k},f\left(B_{1}^{k}\right)\rangle,\langle B_{2\mathrm{L}}^{k},B_{2\mathrm{R}}^{k},f\left(B_{2}^{k}\right)\rangle,\cdots,\langle B_{m\mathrm{L}}^{k},B_{m\mathrm{R}}^{k},f\left(B_{m}^{k}\right)\rangle\} \tag{3}$$

式中，$B_{1\mathrm{L}}^{k}$为子直方图$H_{\mathrm{L}k}$中第1个桶的左边界值，$B_{1\mathrm{R}}^{k}$对应第1个桶的右边界值，$f(B_{1}^{k})$为第1个桶内的频率值。

集群中请求发起节点接收到N个工作节点的子直方图集合为式(4)：

$$\begin{aligned}Dataset=\{&\langle B_{1\mathrm{L}}^{1},B_{1\mathrm{R}}^{1},f\left(B_{1}^{1}\right)\rangle,\langle B_{2\mathrm{L}}^{1},B_{2\mathrm{R}}^{1},f\left(B_{2}^{1}\right),\cdots,\\&\langle B_{m\mathrm{L}}^{1},B_{m\mathrm{R}}^{1},f\left(B_{m}^{1}\right)\rangle,\langle B_{1\mathrm{L}}^{2},B_{1\mathrm{R}}^{2},f\left(B_{1}^{2}\right)\rangle,\\&\langle B_{2\mathrm{L}}^{2},B_{2\mathrm{R}}^{2},f\left(B_{2}^{2}\right)\rangle,\cdots,\langle B_{m\mathrm{L}}^{2},B_{m\mathrm{R}}^{2},f\left(B_{m}^{2}\right)\rangle,\cdots,\\&\langle B_{1\mathrm{L}}^{N},B_{1\mathrm{R}}^{N},f\left(B_{1}^{N}\right)\rangle,\langle B_{2\mathrm{L}}^{N},B_{2\mathrm{R}}^{N},f\left(B_{2}^{N}\right)\rangle,\cdots,\\&\langle B_{m\mathrm{L}}^{N},B_{m\mathrm{R}}^{N},f\left(B_{m}^{N}\right)\rangle\}\end{aligned} \tag{4}$$

最后，步骤9对N个子直方图信息合并，输出等宽直方图H。假设请求发起节点聚合的全局直方图用H_{G}表示，则对工作节点子直方图合并的公式为：

$$\begin{aligned}&H_{\mathrm{G}}.B_{i}.B_{i\mathrm{L}}=H_{\mathrm{L}k}.B_{i}^{k}.B_{i\mathrm{L}}^{k},\\&H_{\mathrm{G}}.B_{i}.B_{i\mathrm{R}}=H_{\mathrm{L}k}.B_{i}^{k}.B_{i\mathrm{R}}^{k},\\&H_{\mathrm{G}}.B_{i}.f(B_{i})=H_{\mathrm{L}1}.B_{i}^{1}.f\left(B_{i}^{1}\right)+\cdots+H_{\mathrm{L}k}.B_{i}^{k}.f\left(B_{i}^{k}\right)+\\&\qquad\cdots+H_{\mathrm{L}N}.B_{i}^{N}.f\left(B_{i}^{N}\right)\end{aligned} \tag{5}$$

式中：$i=1,2,\cdots,m$；$k=1,2,\cdots,N$；B_i为全局直方图H_{G}中第i个桶；$B_{i\mathrm{L}}$为第i个桶的左边界值；$B_{i\mathrm{R}}$为第i个桶的右边界值；$f(B_i)$为第i个桶内数据的频率值；B_i^k对应于第k工作节点子直方图$H_{\mathrm{L}k}$的第i个桶；$B_{i\mathrm{L}}^k$、$B_{i\mathrm{R}}^k$和$f(B_i^k)$分别对应于子直方图$H_{\mathrm{L}k}$的第i个桶的左边界值、右边界值和频率值。

1.4 算法网络传输量分析

云数据库中直方图的分布式并行构造方法采用并行计算思想进行等宽直方图的构建，直方图构造过程中不需要传输表中具体数据，只传输工作节点的直方图桶信息，减少了算法运行过程中的网络传输量。

直方图构建过程中只涉及到节点最值信息、直方图中桶边界值、频率值的传输，经分析得到算法网络传输量为$Q=7N+3UN$。其中，N个工作节点将本地子直方图信息发送至请求发起节点，每个子直方

图包含U个桶，每个桶$\langle B_{\mathrm{L}}, B_{\mathrm{R}}, f(B)\rangle$包含3条数据，依据式（4）可知$N$个工作节点传输子直方图信息至请求发起节点的网络传输量为$3UN$。请求发起节点传输表名T、属性名A与桶数U至N个工作节点的网络传输量为$3N$，N个工作节点将本地最大值Max_{L}、最小值Min_{L}传送至请求发起节点的传输量为$2N$，请求发起节点将全局最大值Max_{G}、最小值Min_{G}传输至N个工作节点的数据量为$2N$，因此算法运行过程中总体的数据传输量为$Q=3N+2N+2N+3UN=7N+3UN$。

文献[11]中对MapReduce架构拓展后构建等宽、等深直方图，其他类型直方图的构建只是在Map阶段和Reduce阶段对数据的处理方式不同。基于MapReduce构建精确的直方图需对所有存储数据执行全扫描，经Mapper处理后产生了BucketID，与数据值一起构成key-value对经Hash定位发送至相应Reducer节点处理。考虑最坏情况下所有工作节点的数据经处理后都需要发送至集群中其他工作节点的Reducer处理，则直方图构建过程中网络传输量为表中所有元组t和各工作节点构建的直方图信息；最好情况下每个工作节点中所有数据经Mapper处理后的BucketID恰好为本地Reducer的节点ID，网络传输量为需要发送至所有Reducer节点的直方图信息。近似直方图构建方法通过对数据进行抽样减少网络传输量，但仍然需要在Shuffle阶段传输抽样数据。假设抽样概率为p，基于MapReduce架构的直方图构建方法与本文方法的网络传输量对比如表1所示。

表 1　本文算法与HEDC++算法网络传输量对比

Tab.1　Comparison of network transmission between the proposed algorithm and HEDC++

直方图构建方法	网络传输量		
	最优	最差	平均
MapReduce精确直方图	$3UN$	$t+3UN$	$t/2+3UN$
MapReduce近似直方图	$3UN$	$pt+3UN$	$pt/2+3UN$
本文算法	$7N+3UN$	$7N+3UN$	$7N+3UN$

大数据环境下，数据库表中记录数上百万、千万已是常态。与数据量t相比，计算集群中工作节点数N、直方图包含的桶数U几乎可以忽略不计。分布式关系型数据库中，带宽资源是极其宝贵的资源，本文算法在网络传输量上有较大改善，可缓解云数据库中带宽占用率较高的问题。

2　性能测试实验与分析

2.1　实验环境及数据

为了测试算法构建直方图性能，在Visual Studio 2013环境下采用C++语言完成了算法在数据库内核的实现，测试环境使用3台DELL PowerEdge R730机架式服务器集群，其配置为：Xeon E5-2603 v3，8 GB DDR4。

实验采用两个测试数据集，即一组人工合成的符合Gauss分布的数据集和一组评分数据集。符合高斯分布数据集包含10万条数据，其最大值为4.289 1，最小值为–4.248 6，数据集分布如图3所示。评分数据是美国Minnesota大学计算机科学与工程学院的GroupLens项目组搜集的用于推荐系统的100万条评分数据集。

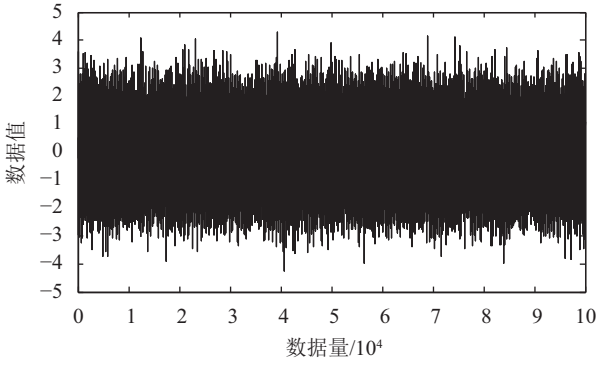


图 3　高斯分布数据集

Fig.3　Gaussian distribution dataset

由图3可以看出，10万条数据绝大多数位于[–2，2]区间内，但是像[–2，–2.5]、[1，1.5]区间内数据频率无法估计。

2.2　实验设置

为了测试算法性能，设计了3组实验：

1）算法实现效果。针对10万条人工合成数据集，使用本文算法分别建立包含不同桶数的直方图。对于100万条评分数据集，只建立包含5个桶的直方图，每个桶内为1类评分数据的频率。

2）可拓展性分析。为了验证算法在云数据库中的可拓展性，分别为计算集群增加一个和减少一个计算节点，并对存储数据进行均衡化处理后，验证算法构造直方图时间随计算集群中节点个数变化的关系。

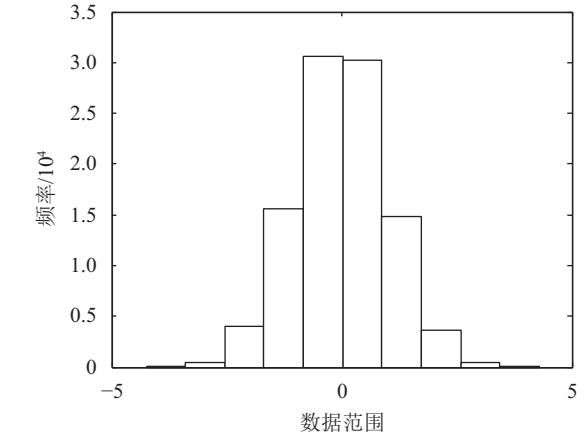
3）与相似算法进行性能比较。对比本文算法与文献[11]中提出的HEDC++方法在两个数据集上构建等宽直方图所消耗时间与网络传输量。

2.3　实验结果与分析

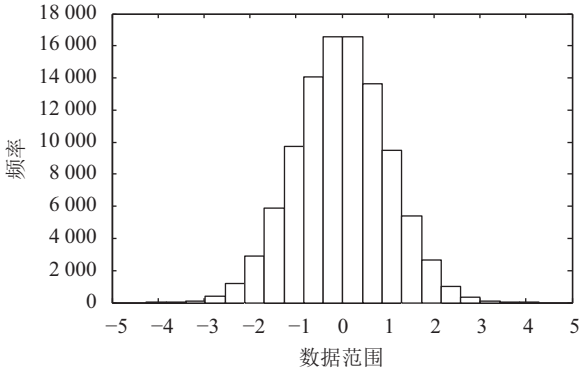
2.3.1　实验1

1）人工合成数据集直方图

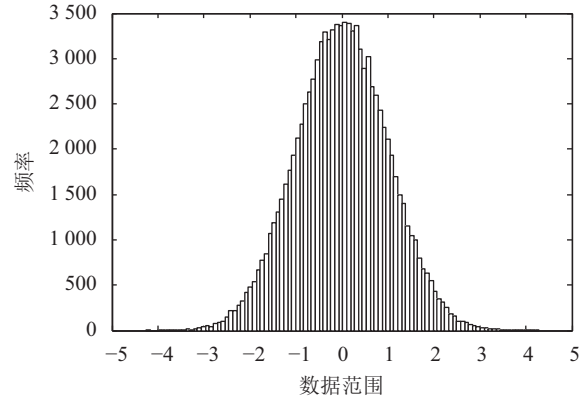
实验将数据集以数据库表形式插入到虚谷云数据库，使用分布式集群并行构建包含10、20和100个桶的等宽直方图如图4所示。



(a) 包含 10 个桶的等宽直方图



(b) 包含 20 个桶的等宽直方图



(c) 包含 100 个桶的等宽直方图

图 4　包含不同桶桶个数的高斯分布数据集直方图

Fig.4　Histograms of Gaussian distribution dataset with different number of buckets

与图3相比，直方图更精确地描述了数据分布情况，依据直方图中桶的左右边界信息和频率值可以估计数据在更精确范围的分布百分比。

由图4中同一数据集构造的3种等宽直方图可以看出，直方图桶数越多，数据分布的描述就越精细，直方图桶的增加会导致计算任务量的增加，但数据集大小确定的情况下桶数无限增加对数据评估优化的提高是有限的。因此，实际应用时需根据数据集大小、数据特征、应用请求精度等具体情况确定直方图应包含的桶数。

2）评分数据集直方图

实验使用的评分数据集为包含6 000个用户对4 000部电影的100万条评分数据，评分范围为1 ~ 5的整数。算法对评分数据构建直方图如图5所示。

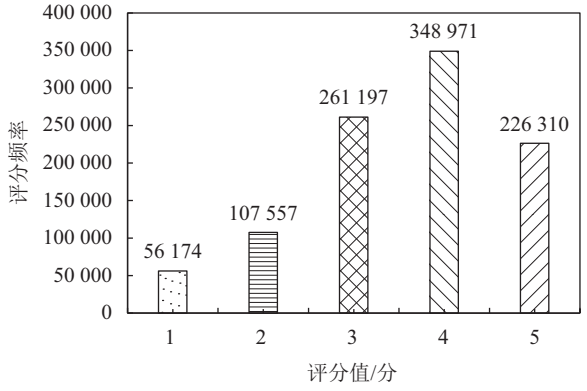


图 5　评分数据集直方图

Fig.5　Histogram of rating dataset

由图5直方图结果可以看出，1 000 209条评分数据中评分为4的数据最多，将近35%，评分为3分和5分的数据在26%和22%左右。直方图的数据分布信息对推荐系统缺失值的评分预测、用户行为分析等提供了重要依据。

2.3.2　实验2

为了进一步分析算法的可拓展性，对比单节点直方图构建方法和本文提出的分布式并行构建方法在10万条高斯分布数据集和100万条评分数据集上的运行时间，并分别为集群增加和减少一个计算节点后测试算法的运行时间如图6所示。

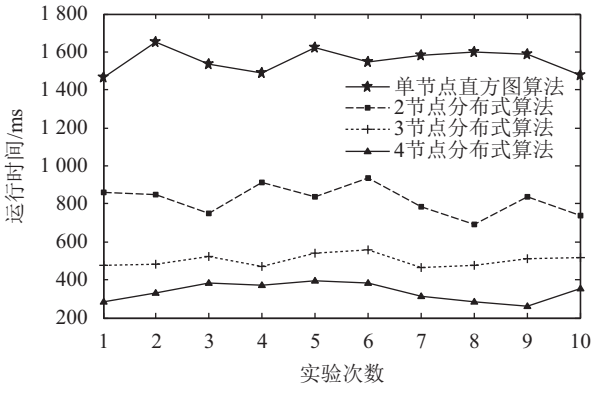
由图6可知，直方图的分布式并行构造方法对比单节点构造方法在性能上有了成倍的提高，随着计算集群中节点数量的增加，本文算法运行时间逐渐减小。特别是100万条评分数据集使用单节点构建直方图算法的时间开销超过20 s。而算法利用集群的并行计算优势，降低了云数据库中直方图构建任务的处理时间。

2.3.3　实验3

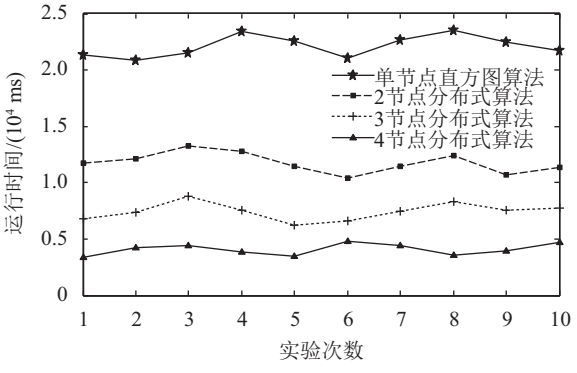
1）算法运行时间对比

将人工合成数据和评分数据在3节点集群中进行均衡化处理后，对比文献[11]中的HEDC++算法与本文方法分别构建精确等宽直方图所需运行时间，如图7所示。

实验结果表明，关系型云数据库中直方图构建时间比基于MapReduce架构构建直方图时间提高了20倍以上。这是因为同等硬件条件下，MapReduce性能远低于并行数据库，关系型数据库中数据是经过



(a) 10 万条数据



(b) 100 万条评分数据

图 6　单节点直方图算法与分布式直方图算法运行时间对比

Fig.6　Comparison of running time between the single node algorithm and the distributed algorithm on histogram construction

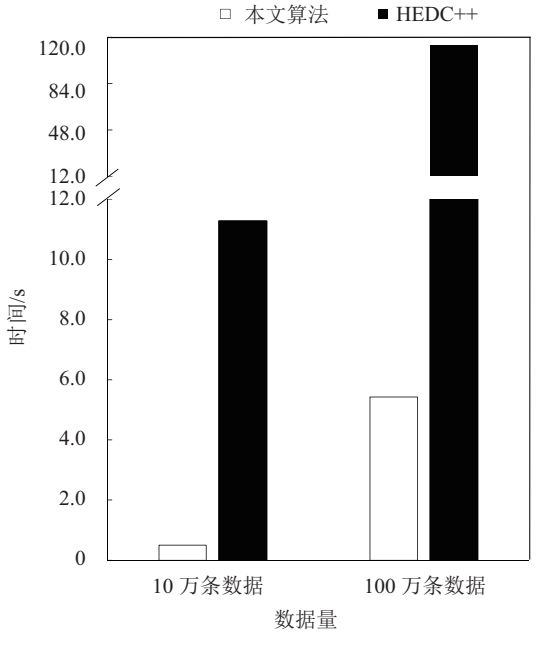


图 7　本文算法与HEDC++算法运行时间对比

Fig.7　Comparison of running time between the proposed algorithm and HEDC++

预处理的高度结构化的数据, Hadoop中没有对数据做任何预处理, 且数据的访问需要直接从文件系统中读入原始数据文件。基于MapReduce的精确直方图构建方法需要将大量key-value对数据经哈希定位发送至集群中其他节点, 而本文算法只需要将少量直方图信息在集群中进行传输。

2)网络传输量对比

云数据库直方图分布式并行构造过程中, 网络传输量为$Q=7N+3UN$, 与数据库表中元组个数无关, 只与存储节点数N和直方图桶数U有关, 而基于MapReduce的直方图构建方法需传输具体数据, 与数据量大小有直接关系。本文算法与基于MapReduce的方法在10万条数据和100万条数据集构造等宽直方图的网络传输量对比, 如表2所示。其中, p为近似直方图的抽样概率。

表 2　本文算法与HEDC++算法网络传输量对比

Tab.2　Comparison of network transmission between the proposed algorithm and HEDC++

直方图方法	10万条数据			100万条数据		
	最优	最差	平均	最优	最差	平均
HEDC++精确直方图	90	100 090	50 090	90	10 000 900	500 090
HEDC++近似直方图	90	100 000p+90	50 000p+90	90	1 000 000p+90	500 000p+90
本文算法	111	111	111	111	111	111

由表2可知, 本文算法网络传输量平均情况下远远低于基于MapReduce架构的直方图构建方法, 基于MapReduce架构的方法在最优情况下比本文算法略低, 但是这要求每个存储节点的数据经Mapper处理后恰好全部定位至本地Reducer节点。

本文算法与数据量大小无关, 在直方图桶数和集群节点数确定的情况下, 无论是10万条数据, 还是100万条数据的网络传输量相同。构建包含20个桶的等宽直方图的网络传输量在任意情况下均为$Q=7\times3+3\times20\times3=201$。

3　结　论

基于关系型云数据库集群中的分布式计算资源, 设计并实现了等宽直方图的并行构建方法, 所采用的主从式架构与数据统计结果的传输改善了直方图构建过程中的网络传输量。实验结果验证了本文方法能够高效地完成云数据库中直方图的构建, 算法性能随集群中计算资源的增加具有较好的可拓展性。目前, 本文算法已应用于成都欧冠信息技术有限公司研发的关系型云数据库平台–虚谷DBMS, 为查询优化提供数据支持。需要指出的是, 云数据库中针

对不同场景的复杂查询，直方图提供的数据分布评估准确性会有所不同。下一步的研究工作中，将针对复杂场景特殊查询，增加如V-Optimal、Maxdiff等更多类型直方图的并行构建方法。同时，由于精确直方图的构建需对数据库表数据进行全扫描，因此选择高效的采样算法构建一定误差范围内的近似直方图也是下一步研究方向。

参考文献:

[1] Yıldız B,Büyüktanır T,Emekci F.Equi-depth histogram construction for big data with quality guarantees[R].New York: Cornell University Library,2016.

[2] Ioannidis Y.The history of histograms (abridged)[C]//Proceedings of the 29th International Conference on Very Large Data Bases.California:Very Large Data Base Endowment Inc Endowment,2003:19–30.

[3] Poosala V,Ioannidis Y E,Haas P J,et al.Improved histograms for selectivity estimation of range predicates[C]//Proceedings of the 1996 ACM Sigmod International Conference on Management of Data.New York:ACM,1996: 294–305.

[4] Chaudhuri S,Motwani R,Narasayya V.Random sampling for histogram construction:How much is enough?[C]//Proceedings ACM Sigmod International Conference on Management of Data.New York:ACM,1998:436–447.

[5] Luo Jizhou,Li Jianzhong,Wang Hongzhi.Construction of an adaptive histogram in compressed database[J].Journal of Software,2009,20(7):1785–1799.[骆吉洲,李建中,王宏志.压缩数据库中一种自适应直方图的构建[J].软件学报,2009,20(7):1785–1799.]

[6] Zhang Longbo,Li Zhanhuai,Wang Yong.Incremental maintenance of approximate equal-depth histograms based on merge-split strategy[J].Computer Science,2009,36(8): 182–184.[张龙波,李战怀,王勇.基于合并–分裂策略的近似等深直方图增量维护[J].计算机科学,2009,36(8): 182–184.]

[7] Bruno N,Chaudhuri S,Gravano L.STHoles:A multidimensional workload-aware histogram[C]//Proceedings of the 2001 ACM Sigmod International Conference on Management of Data.New York:ACM,2001:211–222.

[8] Kanne C C,Moerkotte G.Histograms reloaded:The merits of bucket diversity[C]//Proceedings of the 2010 ACM Sigmod International Conference on Management of Data.New York:ACM,2010:663–674.

[9] Jestes J,Yi Ke,Li Feifei.Building wavelet histograms on large data in mapreduce[J].Proceedings of the VLDB Endowment,2011,5(2):109–120.

[10] Tang Mingwang.Efficient and scalable monitoring and summarization of large probabilistic data[C]//Proceedings of the 2013 Sigmod/pods Ph.D.Symposium.New York:ACM,2013: 61–66.

[11] Shi Yingjie,Meng Xiaofeng,Wang Fusheng,et al.HEDC++: An extended histogram estimator for data in the cloud[J]. Journal of Computer Science and Technology,2013,28(6): 973–988.

[12] Guha S,Koudas N,Shim K.Approximation and streaming algorithms for histogram construction problems[J].ACM Transactions on Database Systems (TODS),2006,31(1): 396–438.

（编辑 张 琼）

引用格式: Wang Yang,Zhong Yong,Zhou Weibo,et al.Distributed and parallel construction method of equi-width histogram in cloud database[J].Advanced Engineering Sciences,2018,50(2):133–140.[王阳,钟勇,周渭博,等.云数据库中等宽直方图的分布式并行构造方法[J].工程科学与技术,2018,50(2):133–140.]