一种复杂多维层次的连接和聚集算法

黄震华[1] 薛永生[1] 段江娇[1] 王劲波[2]
[1](厦门大学计算机科学系 厦门 361005)
[2](厦门大学计划统计系 厦门 361005)
(jukie@netease.com)

摘 要 由于数据仓库中存储着不同粒度、容量巨大的数据记录,所以如何有效地执行联机分析处理(OLAP)查询操作,特别是连接和聚集操作,便成为数据仓库领域的核心问题之一. 为此,提出了一种降低连接和聚集操作的新算法(join and aggregation based on the complex multi-dimensional hierarchies, JACMDH). 算法充分考虑了复杂多维层次的特点,在原有的位图连接索引(bitmap join index)的基础上,采用层次联合代理(hierarchy combined surrogate)和预先分组排序的方法,使得复杂的多维层次上的连接和聚集操作转化成事实表上的区域查询,从而在处理多维层次聚集的同时,提高了连接和聚集的效率. 算法性能分析和实验数据表明,JACMDH 算法和目前流行的算法相比,其性能有显著的提高

关键词 数据仓库; OLAP; 多维层次; 位图连接索引; 层次联合代理; 聚集查询

中图法分类号 TP311.13

A Join and Aggregate Algorithm for Complex Multi-Dimensional Hierarchies

HUANG Zhen-Hua[1], XUE Yong-Sheng[1], DUAN Jiang-Jiao[1], and WANG Jin-Bo[2]
[1](*Department of Computer Science, Xiamen University, Xiamen* 361005)
[2](*Department of Planning Statistics, Xiamen University, Xiamen* 361005)

Abstract Enormous volume of data reside in data warehouse, so it is important to process efficiently expensive queries including join and aggregate operation. In this paper, a new method (JACMDH algorithm) is proposed for processing time-consuming join and aggregate operation. This algorithm takes into consideration the characteristics of the complex multi-dimensional hierarchies and adopts hierarchy combined surrogate/pre-grouping and pre-sorting on the basis of bitmap join index. It improves the join and aggregate efficiency by translating join and aggregate operation of complex multi-dimensional hierarchies into range queries of fact table. The performance analysis and the experimental result, show that the performance of JACMDH algorithm can be improved dramatically, compared with current method for aggregation query evaluation.

Key words data warehouse; OLAP; multi-dimensional hierarchies; bit join index; hierarchy combined surrogate; aggregate query

1 引言及相关工作

通过对数据仓库中的低粒度数据的预聚集处理来生成高效的物化视图是联机分析处理(OLAP)的一个重要技术,而 OLAP 操作一般都是涉及大量数据的即席复杂查询[1]. 用户通过提交 OLAP 查询对数据进行分析,辅助决策,通常需要较快的查询响应速度. 提高 OLAP 查询处理的性能是数据仓库领域的关键性研究问题.

收稿日期: 2003-11-05; 修回日期: 2004-04-09
基金项目: 福建省自然基金项目(A0310008); 福建省高新技术研究开放计划重点基金项目(2003H043)

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

目前主要有 MOLAP(multi-dimensional OLAP) 和 ROLAP(relational OLAP) 两种方式可用于 OLAP 查询的实现. 近几年, 人们在 ROLAP 方面开展了大量的研究工作, 并且提出了若干技术来提高 ROLAP 查询的响应速度, 如新的索引技术[2]、实物化视图技术[3]、采样(sampling) 优化技术[4]等, 但是很多算法使用这些技术解决 OLAP 查询操作时都存在着某些不足之处. 在文献[5~7]中提出的基于采样查询的优化算法, 通过对维表和事实表中相似记录的采样来评估查询结果集, 从而有效减少了 OLAP 操作的时间开销; 但是它们不支持分组聚集操作. 文献[8]中提出的算法在进行连接和聚集操作时只需访问事实表, 从而提高了维表和事实表的连接和聚集效率; 但是它只能用于简单的星型模型中. 本文从优化维表和事实表的连接及聚集操作出发, 提出了一种新的基于复杂多维层次的连接和聚集算法——JACMDH (join and aggregation based on the complex multi-dimensional hierarchies).

JACMDH 算法充分考虑了复杂多维层次的特点, 在原有的位图连接索引(bitmap join index)[2] 的基础上, 采用层次联合代理(hierarchy combined surrogate) 和预先分组排序的方法, 使得复杂的多维层次上的连接和聚集操作转化成事实表上的区域查询, 从而在处理多维层次聚集的同时, 提高了连接和聚集的效率.

文章以下的部分是这样组织的: 第 2 节给出复杂多维层次的层次联合代理方法; 第 3 节描述 JACMDH 算法, 并给出性能分析; 第 4 节进行 JACMDH 算法实验结果分析; 最后对全文进行总结.

2 复杂多维层次的层次联合代理

在文献[9~11]中研究并提出了基于某一具体维的层次联合代理, 然而 OLAP 操作通常要结合多个维的属性, 所以我们必须将某一具体维的层次联合代理扩展为能够适用于多个维的情况.

2.1 有关多维层次的若干定义

定义 1. 复杂多维层次的一棵层次树 H-Tree 是一个以 ALL 为根结点的有向非循环图(directed acyclic graph, DAG), 可用二元组 $\Gamma=(\underline{V}, \partial)$ 表示. 其中, $\underline{V}=\{ALL, \underline{V}_1, \underline{V}_2, \ldots, \underline{V}_n\}$ 是 Γ 中结点集合, $\partial=\{\partial_{ij} \mid \partial_{ij}$ 表示 Γ 中有 $\underline{V}_i \rightarrow \underline{V}_j\}$ 是 Γ 中有向连线集合.

定义 2. 设维 D 的值域为 $\mathbb{R}=\{\sigma_1, \sigma_2, \ldots, \sigma_t\}$, 对应层次树 H-Tree 的深度记为 γ, 则它有 $\gamma+1$ 层的有序集族, 记为 $\Pi=\{\xi^0, \xi^1, \ldots, \xi^\gamma\}$. 如果 $\lambda=(X_1, X_2, \ldots, X_m)$ 满足下列条件, 则称 λ 为层次树 H-Tree 的第 i 层$(0 \leqslant i \leqslant \gamma)$ ξ^i 的成员组:

① $depth(X_j)=i, \ (1 \leqslant j \leqslant m)$;

② $X_j \subseteq \mathbb{R}$

③ $\xi^i=\bigcup_{0 \leqslant j \leqslant m} X_j$;

④对 $\forall X_p, X_q \in \xi^i$ 且 $X_p \neq X_q$, 则 $X_p \cap X_q=\varnothing$.

其中 $depth(X_j)$ 为 X 的深度, 第 i 层的第 j 个成员$(1 \leqslant j \leqslant m)$简记为 X_j^i. 显然, $\mathbb{R}=\bigcup_{\substack{0 \leqslant i \leqslant \gamma \\ 1 \leqslant j \leqslant m}} X_j^i$. 由定义 2 知, 处于同一层次上的各成员所表示的实体集不相互重叠.

定义 3. 成员 X_j^i 的子成员集定义为 $children(X_j^i)=\{X_l^{i+1} \in \xi^{i+1} \mid X_l^{i+1} \subseteq X_j^i\}$.

定义 4. 成员 X_j^i 的父成员集定义为 $parent(X_j^i)=\{X_k^{i-1} \in \xi^{i-1} \mid X_k^{i-1} \supseteq X_j^i\}$.

定义 5. 设$|children(X_j^i)|=\tau$, 则定义双射函数 $BOrd_X$ 为 $children(X_j^i) \rightarrow \{0, 1, \ldots, \tau-1\}$. 双射函数 $BOrd_X$ 为成员 X_j^i 的每个子成员 $X_l^{i+1} \in children(X_j^i)$赋予一个互不相同的有序码位, 从而定义了一种编码模式.

2.2 复杂层次的联合代理

为了有效地对复杂层次进行编码, 从而减少多个维表的连接和层次聚集操作的时间和空间消耗, 我们采取了联合代理的方法.

定义 6. 深度为 γ 的层次树 H-Tree 上的 $\gamma+1$ 个有序集为 $\xi^0, \xi^1, \ldots, \xi^\gamma$, $\xi^i(0 \leqslant i \leqslant \gamma)$ 的 m 个成员记为 $X_1^i, X_2^i, \ldots, X_m^i$, 赋予成员 X_j^i 的子成员集 $children(X_j^i)$的双射函数为 $BOrd_{X_j^i}$, 成员 X_j^i 的代理值 $f(H, X_j^i)$ 与其父成员 X_k^{i-1} 的代理值 $f(H, X_k^{i-1})$ 之间的连接记为$\oplus$. 我们可用递归形式来定义在层次树 H-Tree 上各成员的联合代理值:

$$f(H, X_j^i)=\begin{cases} BOrd_{parent(X_j^i)}(X_j^i), & \text{if } i=1, \\ f(H, parent(X_j^i)) \oplus BOrd_{parent(X_j^i)}(X_j^i), & \text{if } i \neq 1. \end{cases}$$

引理 1. 多维层次上的每个成员的联合代理值存在且惟一.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

证明. 由定义 5 知, 成员 X_j^i 根据双射函数 $BOrd_{X_j^i}$ 得到的子成员集 $children(X_j^i)$ 中的各成员 X_1^{i+1}, X_2^{i+1}, …, X_τ^{i+1} 的编码是有序且惟一的, 进而由定义 6 知从根成员到各个结点成员的编码联结也是惟一的, 即得每个成员的联合代理值存在且惟一.

证毕

定义 **7**. 深度为 γ 的层次树 H-Tree 上的 $\gamma+1$ 个有序集为 ξ^0, ξ^1, …, ξ^γ, $\xi^i(0 \leqslant i \leqslant \gamma)$ 的 m 个成员记为 X_1^i, X_2^i, …, X_m^i, 成员 X_j^i 的子成员集为 $children(X_j^i)$, 令 $\nabla(H, i+1) = \max\{Card(children(X_j^i)) \mid \forall X_j^i \in \xi^i\}$, 其中, $Card$ 为取集合容量的函数. 则我们定义第 $i+1$ 层的层跨距为 $\Rightarrow C^{i+1} = \lceil \log_2 \nabla(H, i+1) \rceil$.

据定义 7 知, 若成员 X_p, X_q 隶属层次树 H-Tree 的同一层 ξ^i, 那么它们所对应的层跨距 $\Rightarrow C^i$ 必相等, 从而成员 X_p 和 X_q 的编码长度是一致的. 因此, 可以用一种压缩的二进制编码方式进行统一管理各成员的联合代理值.

定义 **8**. 设路径 Φ 遍历层次树 H-Tree 的 t 个层的成员 X^1, X^2, …, X^t, 赋予成员 $X^i(1 \leqslant i \leqslant t)$ 的子成员集 $children(X^i)$ 的双射函数为 $BOrd_{X^i}$, 则定义路径 Φ 的联合代理值为

$$f(H, \Phi) = f(H, X^t) = BOrd_{parent(X^1)}(X^1) + BOrd_{parent(X^2)}(X^2) \times 2^{\Rightarrow C_1} + \cdots + BOrd_{parent(X^t)}(X^t) \times 2^{\Rightarrow C_1 + \Rightarrow C_2 + \cdots + \Rightarrow C_{t-1}}.$$

当以定义 6~8 来编码存储层次树 H 上的各结点时, 优点在于它能够用较少并且统一的位数来存储较多的数据, 并且减少搜索满足条件的记录的时间开销, 从而提高连接和聚集操作的效率.

3　算法描述

3.1　位图连接索引

位图连接索引[2] 在位图索引的基础上发展起来. 引入位图连接索引的目的在于减少维表和事实表连接操作的时间, 提高聚集操作的有效性.

定义 **9**. 数据仓库中 n 个维记为 d_1, d_2, …, d_n, 维 $d_i(1 \leqslant i \leqslant n)$ 的 δ 条记录所对应的主码值 $(DPKs)$ 为 ϑ_1, ϑ_2, …, ϑ_δ, 事实表 FT 的 ϖ 条事实记录所对应的元组标识符 $(TIDs)$ 为 $\hbar_1$, $\hbar_2$, …$\hbar_\varpi$, 定义维 d_i 和事实表 FT 间的位图连接索引 $\mathcal{J}^i$ 为具有 δ 行和 ϖ 列的二维矩阵 $\boldsymbol{R}[DPKs, TIDs]$; 该矩阵各分量按如下公式获得值:

$$\boldsymbol{R}[DPKs, TIDs] = \begin{cases} 1, & \text{如果元组标识符 } \hbar \text{ 所对应的事实表元组包含维表主码值 } \vartheta, \\ 0, & \text{否则.} \end{cases}$$

显然, 在 n 个维的数据仓库中存在 n 张维表和事实表间的位图连接索引, 记为 $\mathcal{J}^1$, $\mathcal{J}^2$, …, $\mathcal{J}^n$.

对两个或两个以上的位图连接索引执行适当的 AND 或 OR 操作, 在很大程度上缩短了对满足多个条件的结果集的搜索时间. 更重要的是位图连接索引可以使维表和事实表的连接和聚集操作只要访问事实表即可.

3.2　复杂多维层次的连接和聚集算法 JACMDH

近年来, 人们研究连接和聚集算法的方法可分为两类:

(1) 聚集查询的并行处理和结合聚集操作的查询优化方法, 其中有代表性是文献[12, 13];

(2) 从优化 OLAP 操作的本身出发来研究聚集算法, 其中有代表性是文献[14].

由方法(1), (2) 而得到的连接和聚集算法只能够优化某一方面, 有些算法可以有效提高连接的效率, 但是维表本身的搜索开销很大; 有些算法可以快速得到满足条件的结果集, 但是它只基于单维的, 从而限制了应用的范围.

本文从优化维表和事实表的连接和聚集操作出发, 提出了一种基于复杂多维层次的连接和聚集算法 JACMDH. JACMDH 算法的核心思想是:

① 把多维层次每个维上的约束通过层次联合代理转换成区域查询, 并把满足条件的属性值集放入临时表中;

② 根据分组属性排序结果集;

③ 根据位图连接索引, 获得每个分组的位图;

④ 根据每个分组的位图中的置 1 位, 选取事实表中的记录, 并通过期望的聚集函数来计算它们.

下面我们给出 JACMDH 算法的处理步骤:

输入: 事实表 FT, 维表 DT_1, …, DT_m, 分组属性 GA_1, …, GA_m, 层次树 H-Tree 的联合代理编码文件 CS_1, …, CS_m, 位图连接索引 $\mathcal{J}^1$, $\mathcal{J}^2$, …, $\mathcal{J}^n$, 聚集属性为 $Aggr(A)$;

输出: 具有分组属性的聚集度量表 $Agg\text{-}Mes\text{-}table(GA_1, \cdots, GA_m, M_1, \cdots, M_v)$.

(1) 将初始查询 Q 分解成单维查询 Q_1, …,

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

Q_m, 其中 Q_j ($1 \leqslant j \leqslant m$) 为对维表 DT_j 的简单查询, 仅包含原查询 Q 中与维表 DT_j 有关的查询条件和相关字段;

(2) for j = 1 to m

1) 对于查询 Q_j 的查询条件 C_{qj}, 查找编码文件 CS_j, 得到该条件字段所对应的联合代理编码 ω;

2) for i = 1 to ($l\_m(CS_j) - l\_o(CS_j)$)

① $\omega^{\#} = \omega \,||\, "0"$;

② $\omega^{\#\#} = \omega \,||\, "1"$;

/* $l\_m(CS_j)$ 和 $l\_o(CS_j)$ 分别为编码文件 CS_j 的最大编码长度和 ω 的编码长度, 符号" ||" 为字符串连接操作* /

3) 选择所有编码在 $\omega^{\#}$ 和 $\omega^{\#\#}$ 之间的记录插入到临时表 $Temp_j$ 中;

4) 根据查询 Q_j 中的分组属性 GA_j, 使用 K_ary 合并算法[15]来分组排序临时表 $Temp_j$;

5) for k = 1 to $Comp_j$ /* $Comp$ 等于 $Temp_j$ 中分组的组数* /

① 根据位图连接索引 $\mathcal{J}$ 对各组中每条记录所对应的在 $\mathcal{J}$ 中的列执行 OR 操作, 从而得到各分组的位图 Bm_{jk};

②将分组属性 GA_j 的各分组值和各分组的位图 Bm_{jk} 构成的元组 (GA_j, Bm_{jk}) 插入到临时表 # $Temp_j$ 中;

(3) 根据 PsJoin 连接算法[16]对 m 个临时表 # $Temp_1$, …, # $Temp_m$ 中的分组属性进行连接, 并把它们所对应的位图执行 AND 操作, 并删除那些位图矢量全为 0 的元组, 得到一新表: Grp_Agg_tab (GA_1, …, GA_m, Grp_Bitmap);

(4) 根据每个分组的位图中的置 1 位, 选取事实表中的记录, 并通过期望的聚集函数来计算它们, 并将结构插入到聚集度量表 Agg_Mes_table 中;

(5) 删除临时表 $Temp_1$, …, $Temp_m$, # $Temp_1$, …, # $Temp_m$, Grp_Agg_tab.

3.3 算法性能分析

对磁盘的访问是数据仓库聚集查询的主要开销. 在第 3.3 小节中, 将用磁盘访问次数来分析本文所提出的算法的性能. 由于维表中的数据相对较少, 所以访问编码文件可以在内存中进行; 但是因为事实表的数据量很大, 从而位图连接索引也将会很大, 所以访问位图连接索引将在磁盘中进行. 我们假设每个位图连接索引是一个二维矩阵, 从而可知磁盘的访问次数约为(满足条件的位图矢量大小)/(每个磁盘块的大小). 由于本文提出的算法是先按分组属性对满足条件的元组分组, 然后才访问每个分组中元组的位图连接索引, 所以每个分组的位图可以放在内存中. 表 1 是算法性能分析所用到的参数.

表 **1** 算法性能分析参数表

参数	描述
Num_d	参与连接和聚集操作的维的记录数
Num_f	事实表的记录数
Siz_d	维表记录的大小
Siz_f	事实表记录的大小
Siz_bk	每个磁盘块大小
Sel_r	选择度
Par_r	连接的参与率
Num_s	用来排序分组的磁盘数

算法的开销可分为 4 部分来计算:

(1) 根据联合代理, 把满足条件的元组放入临时表中. 这部分的开销为

$$Cost_1 = \left\lceil Siz\_d \times (Num\_d \times Sel\_r) / Siz\_bk \right\rceil .$$

(2) 根据 K_ary 合并算法, 对临时表中的记录按分组属性进行排序分组. 这部分的开销为

$$Cost_2 = 2 \times \left\lceil Siz\_d \times (Num\_d \times Sel\_r) / Siz\_bk \right\rceil \times \log_{Num\_s} \left\lceil Siz\_d \times (Num\_f \times Sel\_r) / Siz\_bk \right\rceil .$$

(3) 根据位图连接索引, 计算相应分组的位图, 这部分的开销为

$$Cost_3 = \left\lceil (Sel\_r \times Par\_r \times Num\_d) \right\rceil \times \left\lceil (Num\_f / (8 \times Siz\_bk) \right\rceil .$$

(4) 对每个分组, 根据它的位图中置 1 位来访问事实表, 对聚集属性进行聚集操作. 这部分的开销为

$$Cost_4 = (Num\_f \times Sel\_r) .$$

所以, 算法总的开销为 $Total\_Cost = Cost_1 + Cost_2 + Cost_3 + Cost_4$. 目前, 这方面比较有代表性的算法是文献[8]中提到的算法, 该算法的总开销为

$$Total\_Cost'' = \left\lceil (Sel\_r \times Par\_r \times Num\_d) \right\rceil \times \left\lceil Num\_f / (8 \times Siz\_bk) \right\rceil + (Num\_f \times Sel\_r) + (Num\_f \times Sel\_r) + \left\lceil (Siz\_d + Siz\_f) \times (Num\_f \times Sel\_r) / Siz\_bk \right\rceil + 2 \times \left\lceil (Siz\_d +$$

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

$$Siz\_f) \times (Num\_f \times Sel\_r)/Siz\_bk \rceil \times \log_{Num\_s}$$

$$\lceil (Siz\_d + Siz\_f) \times (Num\_f \times Sel\_r)/Siz\_bk \rceil +$$

$$\lceil (Siz\_d + Siz\_f) \times (Num\_f \times Sel\_r)/Siz\_bk \rceil .$$

比较 $Total\_Cost$ 和 $Total\_Cost''$ 两式可知, 本文提出的算法在很大程度上减少了磁盘的访问次数, 从而提高了维表和事实表的连接以及聚集操作的效率.

4　算法实验结果分析

我们在 OLAP 查询操作的研究中, 实现了基于复杂多维层次的连接和聚集算法 JACMDH, 并进行了算法实验. 实验所用环境为 P Ⅲ667(128 MB 内存), 数据库使用的是 Oracle9i 系统. 实验中用到的数据是我们通过人为构建的数据仓库, 其中包括 4 个维表(Time, Product, Store 和 Customer)和一个事实表(Sales). 图 1 给出了我们所构建的数据仓库中维表和事实表的结构:

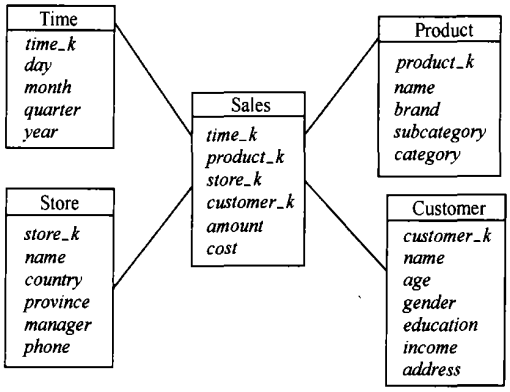

图 1　人为数据仓库的表结构

用于对该数据仓库进行 OLAP 聚集查询的 SQL 语句如下:

```
SELECT P.brand, St.name, C.income,
       T.month, Sum(S.cost)
FROM Product P, Store St, Customer C,
     Time T, Sales S
WHERE (P.product_k = S.product_k) AND
      (St.store_k = S.store_k)
  AND (C.customer_k = S.customer_k) AND
      (T.time_k = S.time_k)
  AND (P.category = 'Food') AND
      (St.manager = 'Smith')
  AND (C.education = 'College')
GROUP BY P.brand, St.name,
         C.income, T.month.
```

实验中用到的 4 个维表和一个事实表的部分数据如表 2 至表 6 所示.

维表和事实表的记录大小分别为 106 和 148 字节, 它们的记录数分别为 800000 和 6000000 条, 选择度 $Sel\_r = 0.05$, 连接的参与率 $Par\_r = 0.8$, 磁盘块大小 $Siz\_bk = 4$ KB, 用来排序分组的磁盘块数 $Num\_s = 100$.

表 2　时间维部分数据

time_k	day	month	quarter	year
344	21	3	1	1999
443	3	8	3	2002
123	12	4	2	2001
913	27	11	4	2003
522	15	9	3	1998
⋮	⋮	⋮	⋮	⋮

表 3　产品维部分数据

product_k	name	brand	subcategory	category
3	Kristiet	Markexs	Skirt	Clothing
7	Relox	Zduet	Jean	Clothing
21	Sweet Tooth	Chewy Industries	Candy	Food
11	Paomian	Kangshifu	Fastfood	Food
7	Paxit	Jeanes	Qixinshi	Cosmetic
1	Tipontx	Futxoge	Jean	Clothing
23	Rubirat	Uitondit	Candy	Food
17	Turkey	Frozen Bird	Frozen Foods	Food
⋮	⋮	⋮	⋮	⋮

表 4　商店维部分数据

store_k	name	country	province	city	manager	phone
3	Tafrei	China	Fujian	Fuzhou	Lifei	xxxx-xxxxxxx
6	Jexecf	China	Jiangshu	Shuzhou	Huangyi	xxxx-xxxxxxx
7	Fucklex	China	Jiangxi	Nanchan	Smith	xxxx-xxxxxxx
2	Renex	China	Shandong	Yantai	Chenyan	xxxx-xxxxxxx
5	Uyerwa	China	Guizhou	Guiyang	Yangkun	xxxx-xxxxxxx
⋮	⋮	⋮	⋮	⋮	⋮	⋮

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved.　http://www.cnki.net

表 5 顾客维部分数据

customer_k	name	age	gender	education	income	address
5	Liyang	32	M	College	3700	xxxxxxxxxxx
12	Chenli	25	M	High School	1370	xxxxxxxxxxx
4	Xuxian	41	FM	College	5000	xxxxxxxxxxx
24	Liuxiao	29	M	College	4100	xxxxxxxxxxx
8	Wangjin	38	FM	High School	2000	xxxxxxxxxxx
⋮	⋮	⋮	⋮	⋮	⋮	⋮

表 6 事实表部分数据

time_k	product_k	store_k	customer_k	amount	cost
443	7	2	12	20	400
123	3	11	7	10	2100
443	21	7	8	150	6578
443	23	19	24	21	4100
8	1	12	10	2	98
18	5	3	33	7	200
3	12	4	2	90	9543
9	1	2	23	5	120
72	23	8	7	11	310
⋮	⋮	⋮	⋮	⋮	⋮

在实验中, 我们分两种情况来评估 JACMDH 算法的性能:

(1) 当事实表记录数(6000000)不变的情况下, 维表记录数从 100000 条增加为 800000 条, 图 2 显示了 JACMDH 算法和目前较流行算法[14]的性能比较. 当聚集函数为记录记数 COUNT 时, 因为 JACMDH 算法只需对位图连接索引执行 OR 和 AND 操作即可, 此时性能提高最明显, 约为 42%.

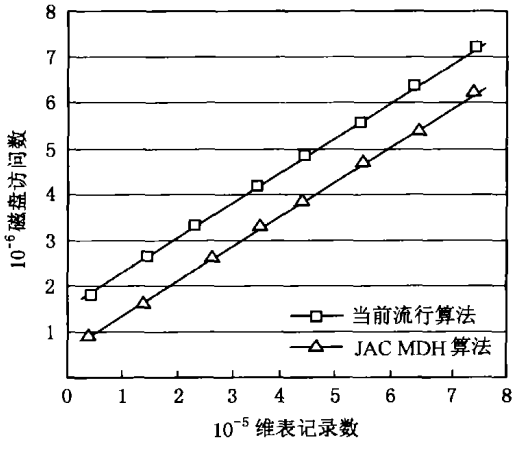

图 2 维表记录数变化时 JACMDH 算法和目前流行算法的性能比较

(2) 当维表记录数(800000)不变的情况下, 事实表记录从 1000000 条增至 6000000 条, 图 3 显示了 JACMDH 算法和目前流行算法[14]的性能比较. 当聚集函数为记录记数 COUNT 时, JACMDH 算法比当前算法性能提高 18%.

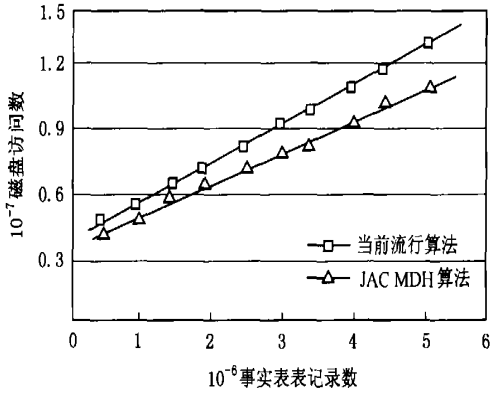

图 3 事实表记录数变化时 JACMDH 算法和目前流行算法的性能比较

5 结 论

由于海量数据的存在, 所以在数据仓库中执行复杂的 OLAP 操作的开销将会很大. 如何有效地执行这些耗时的 OLAP 操作是一个值得研究的课题. 通常的 OLAP 操作会涉及到维表和事实表的连接操作和对事实表中度量值的聚集操作.

本文提出一种使用层次联合代理的方法把多维层次每个维上的约束转换成区域查询, 减少了维表的查找时间. 引入位图连接索引, 从而使得维表和事实表的连接操作只需访问事实表即可. 在使用位图连接索引前进行分组和排序, 使得分组的位图可以存放在内存中, 从而减少了磁盘访问的次数.

为了显示 JACMDH 算法的有效性和优越性, 以磁盘访问次数的方式比较了 JACMDH 算法和目前流行算法之间的代价差异并进行了实验验证, 从而得出了结论: JACMDH 算法比目前流行算法的效率高.

参 考 文 献

1 S Chaudhuri, U Dayal. An overview of data warehousing and OLAP technology. ACM SIGMOD Record, 1997, 26(1): 65~74

2 O Neil, P D Quass. Improved query performance with variant indexes. ACM SIGMOD Record, 1997, 26(2): 38~49

3 D Srivastava, S Dar, H Jagadish, *et al*. Answering queries with aggregation using views. In: T M Vijayaraman, A P Buchmann, C Mohan, *et al* eds. Proc of the 22nd Int'l Conf on Very Large Data Bases. San Francisco: Morgan Kaufmann, 1996. 318~329

4 K Ushijima, S Fujiwara, I Nishizawa, *et al*. SUPRA: A sampling query optimization method for large scale OLAP. In: Proc of

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

the 9th Int'l Conf on Database and Expert Systems Applications. Oakland: IEEE Computer Press, 1998. 232~ 237

5 F Olken, D Rotem. Simple random sampling from relational databases. The 12th VLDB, Kyoto, Japan, 1986

6 F Olken, D Rotem. Random sampling from B+ trees. The 15th VLDB, Amsterdam, Netherlands, 1989

7 F Olken. Random sampling from databases: [Ph D dissertation]. Berkeley: University of California, 1993

8 K Sin Ht, K Yurr-Ht, K Sang-Wook, *et al*. Improving the processing of queries in data warehousing environment. In: Proc of the 9th Int'l Conf on Database and Expert Systems Applications. New York: Springer, 2002. 669~ 675

9 C Li, X S Wang. A data model for supporting on-line analytical processing. In: K Barker, U Manitoba, eds. Proc of the 5th Int'l Conf on Information and Knowledge Management CIKM'96. New York: ACM Press, 1996. 81~ 88

10 E Bertino, W Kim. Indexing technique for queries on nested objects. IEEE Trans on Knowledge and Data Engineering, 1989, 13(2): 196~ 214

11 C Zou, B Salzberg, R Ladin. Back to the future: Dynamic hierarchical clustering. In: Proc of the 5th Int'l Conf on Database and Expert Systems Applications. Montreal, Canada, 1998

12 W Peng, L Per-Ake. Eager aggregation and lazy aggregation. In: D Umeshwar, *et al* eds. Proc of the 21st Int'l Conf on Very Large Data Bases. Zurich: Morgan Kaufmann, 1995. 345~ 357

13 S Agarwal, R Agrawal, P M Deshpande. On the computation of multidimensional aggregates. In: T M Vijayaraman, A P Buchmann, C Mohan, eds. Proc of the 22nd Int'l Conf on Very Large Data Bases (VLDB'96). SanFransisco: Morgan Kaufmann, 1996. 506~ 521

14 蒋旭东, 冯建华. 联机分析查询处理中的一种聚集算法. 软件学报, 2002, 13(1): 65~ 70
(Jiang Xudong, Feng Jianhua. A novel aggregation algorithm for online analytical processing query evaluation. Journal of Software (in Chinese), 2002, 13(1): 65~ 70)

15 S Xiaojun, H Qing. Efficient embedding k-ary complete trees into hypercubes parallel processing symposium. In: Proc of the 12th Int'l Conf on Data Engineering. Los Alamitos: IEEE Computer Society Press, 1996. 24~ 31

16 A Datta, D VanderMeer, K Ramamritham. Parallel star join+ data Indexes: Efficient query processing in data warehouses and OLAP. IEEE Trans on Knowledge and Data Engineering, 2002, 14(6): 1299 ~ 1316

黄震华 男, 1980 年生, 硕士研究生, 主要研究方向为数据库理论与应用、分布式数据库、数据仓库、数据挖掘等.

薛永生 男, 1946 年生, 教授, 主要研究方向为数据库理论与应用、分布式数据库、数据仓库、数据挖掘等.

段江娇 女, 1972 年生, 讲师, 主要研究方向为数据库理论与应用、分布式数据库、数据仓库、数据挖掘、网络技术等.

王劲波 男, 1978 年生, 硕士, 助教, 主要研究方向为分布式数据库、数据挖掘、数据仓库等.

© 1994-2010 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net