多维数据实视图的动态选择*

谭红星[1,2], 周龙骧[1]

[1](中国科学院 数学与系统科学研究院 数学研究所,北京 100080);
[2](河南大学 计算机科学学院,河南 开封 475001)
E-mail: {hxtan,lxzhou}@math08.math.ac.cn
http://www.amss.ac.cn

摘要: 提出了多维数据中实视图的动态选择方案.其基本思想是由系统跟踪查询的分布情况,并据此动态地调整实视图集合.具体实现了该方案,并证明了在一定条件下,单次选择算法的效果与最优效果的差具有一定的上限.实验结果表明,动态方案的效果优于已有的其他选择方案.

关 键 词: 实视图;OLAP(online analytical processing);多维数据;数据仓库
中图法分类号: TP311 文献标识码: A

实视图是提高 OLAP(online analytical processing)系统查询效率的有效手段.由于 OLAP 系统中的查询是随机的,而选择实视图时需要确知系统中的查询集合,因而现有 OLAP 系统中实视图的选择方案均假设这些查询在综合数据上是均匀分布的,或者用户可以提供其分布概率.而实际上,均匀分布的假设常常不能成立,由用户提供查询的分布概率也是强人所难.另外,由于不能确知系统中的查询,多维数据的实视图常常不得不存储某个综合级别上的所有数据.这固然有助于提高查询效率,但毕竟不能确切地反映查询的需求,难以更好地提高查询的效率.

克服这些问题的方法是不进行假设或要求用户估计,而是根据系统中实际发生的查询来选择实视图.如果我们认为无论何时已收集的查询总可以在一定程度上反映查询的分布情况,那么可以在运行过程中收集查询,并根据已收集到的查询来选择实视图.在此后的运行过程中,继续收集查询,当发现已有的实视图不再适合新的查询集合时,再根据新的查询集合重新选择实视图.

本文考虑以这种方案解决多维数据实视图的选择问题.我们将这种方案称为动态选择方案.实视图的动态选择以实际接受到的查询为基础,因而可以选择出与查询更为接近的实视图集合.我们将给出单次选择的具体方法、实视图集合的动态调整方法及其性能分析.本文第 1 节介绍动态选择方案中单次选择算法 FPUS(frequency per unit space).第 2 节分析 FPUS 的性能.第 3 节介绍实视图的动态调整方案.第 4 节是相关的实验结果.第 5 节给出本文的结论、与相关工作的比较以及对有关问题的讨论.

1 多维数据实视图的选择算法

首先来看一个例子.

例 1:考虑销售业绩的多维数据模式[1],其中的维有两个:产品和销售地点,其度量数据为销售量(sales).两个维的层次结构分别如图 1 所示.其中 *ALL* 表示对维中所有成员的综合.

定义 **1(多维数据格[2])**. 不同综合程度的多维数据集合称为一个数据结点.一个多维数据模式中的所有数

* 收稿日期: 2000-08-23; 修改日期: 2001-01-15
基金项目: 国家自然科学基金资助项目(2008100)
作者简介: 谭红星(1969－),男,河南兰考人,博士,副教授,主要研究领域为分布式数据库系统,数据仓库技术;周龙骧(1938－),男,浙江温州人,研究员,博士生导师,主要研究领域为分布式数据库系统,多媒体数据库系统,电子商务技术.

据结点构成一个格,其中

(1) 数据结点间的偏序$\preceq$定义为:给定结点 u 和 v,$u\preceq v$ 当且仅当仅利用 v 即可计算出 u.这意味着 v 的各维上的级别均低于或等于 u 的相应维上的级别;

(2) 格中最大元记为 DB.DB 上各维的级别为该维中最低的.一般假定 DB 的数据是已知的,可利用它计算格中任意结点的数据.

Dimension Product①: *ProductId*②→*Catagory*③→*ALL* Dimension Location④: *StoreId*⑤→*Area*⑥→*ALL*

①产品维,②产品,③种类,④地点维,⑤商店,⑥地区.

Fig.1 The dimension hierarchies

图 1 维的层次结构

例 1 中多维数据的格如图 2 所示,其中 P,S,C,R,A 分别表示 *ProductId*,*StoreId*,*Category*,*Area* 和 *ALL*,$DB=(P,S)$.

在实视图的选择过程中,需要再估计一个数据结点 v 的尺寸 $|v|$.文献[3]讨论了解决该问题的多种方案.

定义 **2(**多维数据上的查询**)**. 多维数据集合 MD 上的查询 q 是对 MD 中某一数据结点的切片或切块.可以将 q 表示为由 d 个二元组组成的 d 元组:$\{(l_1,R_1),(l_2,R_2),\ldots,(l_d,R_d)\}$,其中 d 为 MD 的维数,l_i 表示维 d_i 的某一级别,R_i 表示在 l_i 上的选择范围.若没有对 l_i 的范围进行限制,可以将(l_i,R_i)简记为 l_i.若 $l_i=ALL$,则维 i 的二元组可以不出现在查询中.

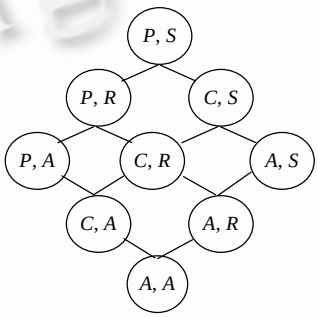


Fig.2 The lattice of multi-dimensional data in Example 1

图 2 例 1 中多维数据的格

查询 q 所访问的数据结点称为 q 的基,记为β_q.若 $q=\{(l_1,R_1),(l_2,R_2),\ldots,(l_d,R_d)\}$,则$\beta_q=\{l_1,l_2,\ldots,l_d\}$.

查询间的偏序关系$\preceq$定义为:对于查询 p 和 q,若 q 的结果可以用来响应 p,则有 $p\preceq q$.显然,若 $p\preceq q$,不但要求$\beta_p\preceq\beta_q$,而且用来生成 p 的数据均应包含在 q 中.

例如,查看产品 P_1,P_2 在地区 A_1 的销售情况的查询 $q_1=\{(ProductId,\{P_1,P_2\}),(Area,\{A_1\})\}$.若地区 A_1 中共有 3 家商店 S_1,S_2 和 S_3.比较这 3 家商店中产品 P_1,P_2 的销售量的查询 $q_2=\{(ProductId,\{P_1,P_2\}),(StoreId,\{S_1,S_2,S_3\})\}$.我们有 $q_1\preceq q_2$.

由于实视图的目的是保存数据供查询访问,因而视图的表示与查询是一致的.我们常将查询和视图混用.

系统所收集的查询包括查询的集合 Q 和每个查询 q 的发生频率 $f(q)$.

定义 **3(**实视图的选择问题**)**. 实视图的选择问题就是在给定存储空间 *Space* 的限制下,选择哪些实视图可以最大限度地提高总体查询的效率.

由于该问题是 NP-完全的,我们采用贪婪算法求解.这里我们从系统收集的查询集合 Q 中选出将被实化的视图,选择标准是单位空间的频率,即 $f(q)/|q|$.其中 $f(q)$是已知的,而$|q|=|\beta_q|\times\prod_{i=1}^{d}|q.R_i|/\prod_{i=1}^{d}|q.l_i|$,其中 d 为多维数据的维数,$|q.l_i|$表示级别 l_i 中不同元素的个数,$|q.R_i|$表示 R_i 的基数.

算法 **1**. 按照单位空间的频率选择实视图(FPUS)

输入:可利用空间 *Space*,查询集合 Q;

输出:所选择的实视图集合 V.

```
V:=∅;
WHILE Space>0 DO
    v:=Q 中 f(q)/|q|最大的 q;
    IF Space≥|v| THEN
        V:=V∪{v};Q:=Q−{v};
```

Space:=*Space*−|*v*|;

ELSE *Space*:=0;

ENDWHILE

RETURN *V*

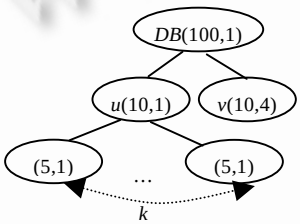
FPUS 的时间复杂度为 $O(n\log_2 n)$.如果采用适当的辅助存储,FPUS 的时间复杂度可以大大降低.例如,由于系统中查询数量相当庞大,可以为这些查询按照 $f(q)/|q|$ 建立索引.这样其时间复杂度仅相当于所选择的实视图的个数.

2 FPUS 的效果分析

本节分析算法 FPUS 的效果.在给定的存储空间的限制下,不同的选择算法会得到不同的实视图集合.由于一旦遇到剩余空间容纳不下的视图,选择过程就结束,因而给定空间可能会有剩余,不同的选择算法得到的剩余也不尽相同.当视图的尺寸与给定空间相比很小时,空间剩余可以忽略不计.我们假设各种算法所选择的实视图总会占满给定的存储空间.

考察一个算法效果的基本方法是将它所得到的效益与最优效益 B_{opt} 相比较.我们希望 FPUS 的效益与 B_{opt} 之差有一个下限.然而遗憾的是,在一般情况下,这个下限是不存在的.

如图 3 所示,如果可利用空间 *Space*=10,FPUS 选择的实视图集合为{*v*}.*v* 的效益 *B*(*v*)=400(效益的计算方法见定义 8).若选择 *u*,则 *u* 的效益 *B*(*u*)=100+90*k*.随着 *k* 的增加,*B*(*u*)会无限制地超过 *B*(*v*).



Of the digit pair of each view, the former indicates the view size and the later indicates the query frequency①.

①在视图边的一对数字中,前者表示视图的尺寸,后者表示视图的频率.

Fig.3 Query set breaking the sub-view restriction

图 3 不满足子视图限制的查询集合

可以看出,*B*(*u*)之所以超过 *B*(*v*),是由于 *u* 有过多子视图的缘故,而且这些子视图的频率都相当小,不足以被 FPUS 选中.如果抛开这种极端的情形,FPUS 就能获得较好的效果.下面,我们在假设上述极端情形不发生的情况下考察 FPUS 的效果.

定义 4(访问者集合**)**. 对于实视图 *v*,为响应其上的查询而访问 *v* 但不同于 *v* 的那些视图的集合称为 *v* 的访问者集合,记为 *Q*(*v*).显然,若 $u\in Q(v)$,则 *u* 没有实化,$u\prec v$ 且不存在另外的实视图 *w* 使得 $u\prec w$ 但 $|w|<|v|$.

若 $u\in Q(v)$,我们称 $v=Q^{-1}(u)$.

一般情况下,仅对已经实化的视图才有必要考虑其访问者集合,否则其访问者集合为空.但我们常关注一个待实化视图的情况.因而当提到一个未实化视图的访问者集合时,是指若将其实化则会发生的情形.

定义 5(访问频率**)**. 对于实视图 *v*,需在 *v* 上响应的查询的频率之和称为 *v* 的访问频率,记为 *F*(*v*),显然 $F(v)=f(v)+\sum_{u\in Q(v)} f(u)$.

定义 6(子视图限制**)**. 设用 FPUS 选中的实视图集合为 *V*,其中频率/尺寸最小的元素为 *w*.对于 *V* 外的视图 *z*,对 *z* 作如下的限制:$F(z)/|z|\le f(w)/|w|$.该限制称为子视图限制.

子视图限制用于保证频率小的视图不会有过多频率更小的子视图.若它的某一个子视图的频率足够大,则会被 FPUS 选中,从而不再属于其访问者集合.

定义 7(相对效益**)**. 设已选择的实视图的集合为 $V, v\notin V$,则相对于 *V* 而言,实化 *v* 所带来的效益:

$$B(v,V)=|Q^{-1}(v)|\,f(v)+\sum_{u\in Q(v)}(|Q^{-1}(u)|-|u|)\,f(u).$$

定义 8(绝对效益**)**. 视图 *v* 的绝对效益 *B*(*v*)是指 *v* 相对于基视图 *DB* 的效益:

$$B(v)=|DB|\,f(v)+(|DB|-|v|)\sum_{u\in Q(v)} f(u).$$

亦即计算 *v* 的绝对效益时不考虑 *v* 和 *DB* 间是否有已经实化的视图,而仅认为在实化 *v* 前为响应 *v* 及 *Q*(*v*)上的

查询需访问 DB,而在实化后访问 v.利用绝对效益便于对效益进行分析,但它要与选择过程中所用到的相对效益保持一致.我们已经证明[4],各当选视图的相对效益之和与绝对效益之和是相等的.因而在考察视图的效益时,利用其绝对效益即可.

我们在分析 FPUS 的效果时,主要是将它与文献[2]中采用的选择算法 BPUS 的效果相比较.BPUS 每次选择单位空间效益最大的视图 v,亦即 $B(v,V)/|v|$最大的视图,将其加入到 V 中,直到所选择的视图在剩余空间内容纳不下为止.设利用穷举法获得的最优解的效益为 B_{opt},v 为格中尺寸最大者,令 $f=|v|/Space$,则利用算法 BPUS 得到的实视图集合的效益不低于 B_{opt} 的 $1-1/e-f$ 倍,其中 e 为自然对数的底.

引理 **1**. 设算法 FPUS 和 BPUS 所选择的实视图带来的效益分别为 B_F 和 B_B,且给定的查询集合满足子视图限制,则 $B_F \geq B_B/2$.

证明:设 FPUS 和 BPUS 选择的实视图集合分别为 F 和 B.这两个集合可能有交集,令 $BF=B\cap F$.

对于 F 和 B,其中视图 v 的访问者集合分别记为 $Q_F(v)$和 $Q_B(v)$.值得指出的是,即便对于同一个实视图 v,即 $v\in BF$,$Q_F(v)$和 $Q_B(v)$也可能是不同的.

F 可分为 BF,F_1,F_2 和 F_3 四个部分,其中 F_1 中任一元素属于 BF 中某元素的访问者集合,F_2 中任一元素属于 $B-F$ 中某元素的访问者集合,$F_3=F-BF-F_1-F_2$.

B 和 F 中视图 v 的效益分别记为 $B_B(v)$和 $B_F(v)$.给定实视图集合 S,将 S 的效益记为其中各元素的效益之和,即 $B_B(S)=\sum_{v\in S}B_B(v)$,$B_F(S)=\sum_{v\in S}B_F(v)$.

显然,BF 和 $B-F$ 构成了对 B 的划分,BF,F_1,F_2 和 F_3 构成了对 F 的划分,即

$$B_B(B)=B_B(BF)+B_B(B-F),\quad B_F(F)=B_F(BF)+B_F(F_1)+B_F(F_2)+B_F(F_3).$$

下面,我们来考察上述各个集合的效益.

F_1,F_2,F_3:对于其中的任意元素 v,

$$B_F(v)=|DB|\,f(v)+(|DB|-|v|)\sum_{u\in Q(v)}f(u)\geq|DB|f(v).$$

因此

$$B_F(F_i)=\sum_{v\in F_i}B_F(v)\geq|DB|\sum_{v\in F_i}f(v),\quad i\in\{1,2,3\}. \tag{1}$$

V_{BF}:对于其中的任意元素 v,

$$Q_B(v)-F_1\subseteq Q_F(v),\text{则有}\sum_{u\in Q_B(v)-F_1}f(u)\leq\sum_{u\in Q_F(v)}f(u),$$

$$\begin{aligned}B_B(v)&=|DB|f(v)+(|DB|-|v|)\sum_{u\in Q_B(v)}f(u)\\&=|DB|f(v)+(|DB|-|v|)\sum_{u\in Q_B(v)-F_1}f(u)+(|DB|-|v|)\sum_{u\in Q_B(v)\cap F_1}f(u)\\&\leq|DB|f(v)+(|DB|-|v|)\sum_{u\in Q_F(v)}f(u)+|DB|\sum_{u\in Q_B(v)\cap F_1}f(u)\\&=B_F(v)+|DB|\sum_{u\in Q_B(v)\cap F_1}f(u).\end{aligned}$$

因此,
$$B_B(BF)=\sum_{v\in BF}B_B(v)\leq\sum_{v\in BF}B_F(v)+|DB|\sum_{v\in BF}\sum_{u\in Q_B(v)\cap F_1}f(u)=B_F(BF)+|DB|\sum_{v\in F_1}f(v).$$

由式(1)可得:

$$B_B(BF)\leq B_F(BF)+B_F(F_1). \tag{2}$$

$B-F$:对于其中的任意元素 v,

$$\begin{aligned}B_B(v)&=|DB|f(v)+(|DB|-|v|)\sum_{u\in Q_B(v)}f(u)\\&\leq|DB|f(v)+|DB|\sum_{u\in Q_B(v)}f(u)\\&=|DB|f(v)+|DB|\sum_{u\in Q_B(v)-F_2}f(u)+|DB|\sum_{u\in Q_B(v)\cap F_2}f(u)\\&=|DB|F(v)+|DB|\sum_{u\in Q_B(v)\cap F_2}f(u).\end{aligned}$$

因此,

$$\begin{aligned}B_B(B-F)&=\sum_{v\in B-F}B_B(v)\\&\leq\sum_{v\in B-F}F(v)+|DB|\sum_{v\in B-F}\sum_{u\in Q_B(v)\cap F_2}f(u)\\&=\sum_{v\in B-F}F(v)+|DB|\sum_{v\in F_2}f(u)\\&\leq\sum_{v\in B-F}F(v)+B_F(F_2).\end{aligned} \tag{3}$$

由式(2)和式(3)可得:

$$B_B(B)-B_F(F)=B_B(BF)+B_B(B-F)-B_F(BF)-B_F(F_1)-B_F(F_2)-B_F(F_3)\le|DB|\sum_{v\in B-F}F(v)-B_F(F_3)\le|DB|\sum_{v\in B-F}F(v).$$

设 w 是 F 中 $f(w)/|w|$最小的视图,对于 $B-F$ 中的元素 v,由于 $F(v)/|v|\le f(w)/|w|$,即 $F(v)\le f(w)/|w|\times|v|$,因此,

$$B_B(B)-B_F(F)\le|DB|\sum_{v\in B-F}F(v)\le|DB|f(w)/|w|\times\sum_{v\in B-F}|v|. \tag{4}$$

对于 F 中任意元素 v,我们有 $f(v)/|v|\ge f(w)/|w|$,即 $f(v)\ge f(w)/|w|\times|v|$,因而

$$B_F(v)\ge|DB|f(v)\ge|DB|f(w)/|w|\times|v|,$$

$$B_F(F)=\sum_{v\in F}B_F(v)\ge|DB|\sum_{v\in F}f(v)\ge|DB|\sum_{v\in F}f(v)/|w|\times\sum_{v\in F}|v|. \tag{5}$$

由于算法 BPUS 和 FPUS 都选择了占满可利用空间 *Space* 的实视图,则

$$\sum_{v\in F}|v|=\sum_{v\in B}|v|\ge\sum_{v\in B-F}|v|. \tag{6}$$

由式(4)~式(6)可得:$B_F(F)\ge B_B(B)-B_F(F)$,亦即 $B_F(F)\ge B_B(B)/2$. □

定理 **1**. 在满足子视图限制的条件下,FPUS 算法的效益不低于最优效益的$(1-1/e-f)/2$ 倍,其中 e 为自然对数的底, f 为视图集合中尺寸最大的视图所占的可利用空间比例.

这是由于 B_B 不低于最优效益的 $1-1/e-f$ 倍[2].

值得指出得是,虽然 FPUS 的效益在一般情况下并不优于 BPUS,但这是指对相同的集合中选取实视图的情形.如果 PBUS 利用基于格的选择方法,而 FPUS 利用基于查询的方法,从我们的实验结果来看,后者仍优于前者.这是由于 PBUS 的选择过于“粗放”的缘故.

3 实视图的动态调整

根据收集的查询选择实视图的集合后,系统就可以利用它们来响应新的查询,并继续收集这些查询.随着新查询的增加,系统中查询的分布情况亦可能发生变化,使得原有的实视图集合不再适应新的查询分布情况,这时需要选择新的实视图集合.

为了尽量避免重复计算查询的结果(实视图中的数据),可以在计算查询后立即调整实视图集合.这种方案称为即时调整,它发生在执行一个查询之后.

算法 **2**. 实视图集合的即时调整.

输入:当前的查询 q,剩余的可利用空间 *Space*,当前的实视图集合 V;$q\notin V$.

S:=*Space*; //收集到的可利用空间

V_{remove}:=∅; //将要被删除的实视图集合

search:=TRUE; //继续搜索标记

WHILE $S<|q|$ AND *search* DO

  v:=V 中 $f(v)/|v|$最小的视图;

  IF $f(v)/|v|<f(q)/|q|$ THEN

    S:=S+$|v|$;

    V_{remove}:=$V_{\text{remove}}\cup\{v\}$; V:=$V-\{v\}$;

  ELSE *search*:=FALSE;

ENDWHILE

IF $S\ge|q|$ THEN

  删除 V_{remove} 中的视图;*Space*:=S;

  实化 q; *Space*:=*Space*$-|q|$; V:=$V\cup\{q\}$;

ELSE V:=$V\cup V_{\text{remove}}$;

在该算法中,是否实化 q 由以下两点决定:

(1) 若剩余空间尚够,则不需要删除任何实视图(V_{remove}=∅),直接实化即可;

(2) 否则,需要从实视图集合中选取一批单位空间频率小的实视图,将它们删除,为 q 腾出空间.这些视图的单位空间频率均应小于 q,且其空间之和加上剩余空间可以容下 q.

4 实验结果

我们用人工数据对 FPUS 的效果进行了实验,主要是与 BPUS 的效果相比较.结果表明前者优于后者.

实验数据包括两部分:多维数据和查询.

(1) 多维数据的设计.其生成方法与文献[5]相似:多维数据有 4 个维 D_0,D_1,D_2 和 D_3,它们分别有 4,3,4,3 个级别,各级别的成员个数见表 1.

Table 1 Count of the members of multi-dimensional data in each level

表 1 多维数据各级别的成员个数

Levels①	Dimensions②			
	D_0	D_1	D_2	D_3
3	1	—	1	—
2	25	1	5	1
1	50	25	25	10
0	100	50	50	50

①级别,②维.

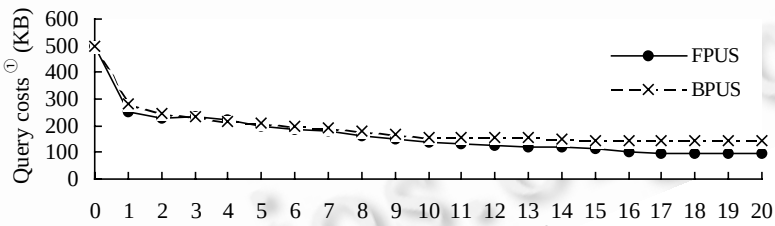
多维数据的基视图 *DB* 中共有 500K 行,假设多维数据的分布是均匀的,这样估计出 CUBE 的尺寸为 15M 行.

(2) 查询的设计.在现实中,对多维数据的访问常是不均匀的.这表现为对某一数据区域或某种综合程度的数据的访问较为频繁.我们设计的查询反映了这一情况.我们假定,在所有的查询中,有 90%访问多维数据格中 10%的结点.对于任意被访问的级别,在访问它的所有查询中,有 80%的查询访问其中 20%的成员.

实验中共使用了 10 000 个查询.这些查询被分为 10 组,其中每一组按上述比例在结点和维的成员间重新随机分布.最后取这些查询的平均代价作为查询的代价.

需要指出的是,基于格的实视图选择时所需要的查询分布与这里的查询分布在概念上是不同的.前者所要求的是具体的查询出现频率,而后者仅要求一个分布比例,具体的分布则是完全随机的.

我们比较了在不同的存储空间内实化视图后查询的代价.存储空间取 CUBE 总尺寸的 0%~20%.两种算法的效果如图 4 所示.



①查询代价,②实视图空间/Cube 空间.

Fig.4 Query cost based on different evaluation algorithms

图 4 基于不同求解算法的查询代价

5 结论及与相关工作的比较

本文讨论了多维数据实视图的选择问题.在既不对查询分布进行均匀性假设也不要求用户提供查询分布情况的条件下,我们给出了基于查询的实视图的动态选择方案及相关算法,并证明了在满足一定条件下,该算法的效果与最优效果的差具有一定的下限.我们还提出了即时调整方案,在利用该方案选择实视图时,既不需要重新计算实视图的数据,也不需要估计视图的尺寸,因而效率更高,结果也更为准确.

对多维数据实视图选择问题的研究是当前数据库研究界的热门话题.文献[2]提出了多维数据组织的格模型,给出了以单位空间的效益为选择标准、时间复杂度为 $O(kn^2)$的贪婪算法 BPUS.在此基础上,文献[6]讨论了带有 B-树索引的实视图的选择问题;文献[7]提出了以实视图的尺寸为选择标准、时间复杂度为 $O(n\log_2 n)$的选择算法 PBS;文献[8]提出了筛选候选视图的方法,以提高选择的效率.

总之,已有的各种方案均基于两点,即查询的分布情况是已知的,而且实视图内容以整个结点为单位.本文与这些方案的主要区别在于不要求上述两点,而以系统收集的查询集合及其频率为选择实视图集合的基础,并在运行的过程中动态调整实视图的集合.

References:

[1] Agrawal, R., Gupta, A., Sarawagi, S. Modeling multidimensional databases. In: Gray, A., Larson, Per-Åke, eds. ICDE'97, Proceedings of the 13th International Conference on Data Engineering. Birmingham, U.K.: IEEE Computer Society Press, 1997. 232~243.

[2] Harinarayan, V., Rajaraman, A., Ullman, J.D. Implementing data cubes efficiently. In: Jagadish, H.V., Mumick, I.S., eds. SIGMOD'96, Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data. Montreal: ACM Press 1996. 205~216.

[3] Shukla, A., Deshpande, P., Naughton, J.F., *et al*. Storage estimation for multi-dimensional aggregates in the presence of hierarchies. In: Vijayaraman, T.M., Buchmann, A.P., Mohan, C., eds. VLDB'96, Proceedings of the 22nd International Conference on Very Large Data Bases. Bombay: Morgan Kaufmann Publishers, Inc., 1996. 522~531.

[4] Qi, Wen-wen, Xu, Bin, Tan, Hong-xing. Selecting materialized views within data cubes. Journal of He'nan University (Natural Science edition), 2001,31(1):20~25 (in Chinese).

[5] Deshpande, P.M., Ramasamy, K., Shukla, A., *et al*. Caching multidimensional queries using chunks. In: Haas, L.M., Tiwary, A., eds. SIGMOD'98, Proceedings of the ACM SIGMOD International Conference on Management of Data. Seattle: ACM Press, 1998. 259~270.

[6] Gupta, H., Harinarayan, V., Rajaraman, A., *et al*. Index Selection for OLAP. In: Gray, A., Larson, Per-Åke, eds. ICDE'97, Proceedings of the 13th International Conference on Data Engineering. Birmingham, U.K.: IEEE Computer Society Press, 1997. 208~219.

[7] Shukla, A., Deshpande, P., Naughton, J.F. Materialized view selection for multidimensional datasets. In: Gupta, A., Shmueli, O., Widom, J., eds. VLDB'98, Proceedings of the 24th International Conference on Very Large Data Bases. New York: Morgan Kaufmann Publishers, Inc., 1998. 488~499.

[8] Baralis, E., Paraboschi, S., Teniente, E. Materialized view selection in a multidimensional database. In: Jarke, M., Carey, M.J., Dittrich, K.R., *et al*., eds. VLDB'97, Proceedings of the 23rd International Conference on Very Large Data Bases. Athens: Morgan Kaufmann Publishers, Inc., 1997. 156~165.

附中文参考文献:

[4] 祁文文,徐彬,谭红星.数据立方中实视图的选择.河南大学学报(自然科学版),2001,31(1):20~25.

Dynamic Selection of Materialized Views of Multi-Dimensional Data*

TAN Hong-xing[1,2], ZHOU Long-xiang[1]

[1](*Institute of Mathematics, Academy of Mathematics and System Sciences, The Chinese Academy of Sciences, Beijing* 100080, *China*);

[2](*School of Computer Science, He'nan University, Kaifeng* 475001, *China*)

E-mail: {hxtan,lxzhou}@math08.math.ac.cn

http://www.amss.ac.cn

Abstract: A novel method is proposed to select materialized views of multi-dimensional data called dynamic selection. The idea of dynamic selection is that the system is in charged of collecting the queries to obtain their distribution. The set of materialized views is adjusted dynamically according to the queries' distribution. The method is given in detail including the algorithm of single-step selection and the instant adjusting method. It is also proved that under certain constraints, the performance of the single-step algorithm is guaranteed to be no worse than that of the optimal one by a certain bound. The experimental results show that the dynamic selection is more effective than other solutions.

Key words: materialized view; OLAP (online analytical processing); multi-dimensional data; data warehousing

* Received August 23, 2000; accepted January 15, 2001
Supported by the National Natural Science Foundation of China under Grant No.2008100