计算机研究与发展 ISSN 1000-1239/CN 11-1777/TP
Journal of Computer Research and Development 45(5): 901~908, 2008

NDSMMV——一种多维数据集物化视图动态选择新策略

张东站 黄宗毅 薛永生
(厦门大学计算机科学系 厦门 361005)
(ysxue@xmu.edu.cn)

NDSMMV—A New Dynamic Selection Strategy of Materialized Views for Multi-Dimensional Data

Zhang Dongzhan, Huang Zongyi, and Xue Yongsheng
(*Department of Computer Science, Xiamen University, Xiamen* 361005)

Abstract The selection strategy of materialized view is one of the important issues of data warehouse research. Its goal is to elect a group of materialized views, which could cut down the cost of the query greatly on the basis of the limited storage space. The cost model is proposed at first. Then, a new dynamic selection strategy of materialized views for multi-dimensional data (NDSMMV) is presented, which is composed of four algorithms: CVGA (candidate view generation algorithm), IGA (improved greedy algorithm), MAMV (modulation algorithm of materialized views) and DMAMV (dynamic modulation algorithm of materialized views). CVGA generates the candidate view set based on multi-dimensional data lattice, which reduces the number of candidate views to decrease the space search cost and time consumption of the following algorithm. IGA selects materialized views taking account of view query, view maintenance and space constraint. MAMV modulate the materialized views according to the change of the materialized view profit, which improves the capability of querying materialized views. DMAMV uses the sample space to judge whether it is necessary to change the view set which can avoid sharp dither. The comparative experiment indicates that NDSMMV operates more effectively than BPUS and FPUS in the respect that CVGA reduces the amount of views beforehand. IGA selects the materialized views quickly, MAMV modulates the materialized views accurately, and the query expense decreases further with the modulation of the DMAMV on line, which validates the efficiency of NDSMMV.

Key words materialized view; dynamic selection; multi-dimensional data; candidate view; data warehouse

摘 要 物化视图的选择策略是数据仓库研究的重要问题之一.通过深入研究提出了一种多维数据集中物化视图动态选择的新策略——NDSMMV,包括候选视图生成算法CVGA、物化视图选择算法IGA、物化视图调整算法MAMV和物化视图动态调整算法DMAMV. CVGA基于多维数据格生成候选视图集,对候选视图数量进行压缩以减少后续算法的视图空间搜索代价和时间复杂度; IGA基于视图查询、视图维护和存储空间三元评价标准在候选视图集上进行物化视图的选择;MAMV基于物化视图选择过程已选视图的收益变化情况对物化视图进行进一步调整以提高查询的响应性能;DMAMV定时地判断查询视图类型分布是否变化来决定是否进行物化视图的动态调整,从而避免了物化视图集的"抖动".理论分析和实验结果表明该策略是有效可行的.

收稿日期: 2007-08-28; 修回日期: 2008-02-01
基金项目: 国家自然科学基金项目(50604012);福建省高新技术研究计划重点项目(2003H043)

关键词 物化视图; 动态选择; 多维数据集; 候选视图; 数据仓库

中图法分类号 TP311.13

物化视图的选择策略是数据仓库研究的重要问题之一. 其目标是选出一组存储、维护代价与查询代价的总和为最小的物化视图. 由于空间的限制, 物化视图如何选择以最大限度地提高总体查询的效率已成为数据仓库领域的研究热点. 文献[1]适用于已知系统所需要处理的查询类型以及可能对系统中数据进行更新的情况, 算法局限于特定的已知环境. 文献[2]提出了基于多维数据格模型的BPUS算法; 文献[3]提出了以物化视图的尺寸为选择标准, 算法时间复杂度为 $O(n\log_2 n)$ 的PBS算法; 文献[4]将遗传算法获取最优解的能力用于最优物化集的选择; 文献[5]采用随机算法的思想设计了选择物化视图的算法, 其特点是速度快, 适合高维的情况, 但不能从理论上保证得到的结果的精度. 文献[6]结合与或图、贪心算法以及 A^* 启发式算法探讨该问题的求解方法.

由于不可能确定系统中的查询集合, 以上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查询分布概率, 本质上都可归结为静态算法. 而实际随着数据仓库系统的运行, 查询请求动态变化会导致物化视图集的一部分视图收益下降, 部分未被物化的视图收益上升, 使得物化视图集的总收益下降. 必须通过动态选择算法加以解决.

动态选择算法在查询过程中, 根据查询类型的分布动态选择视图物化, 克服了以上静态选择算法的缺点. 文献[7]提出一种物化视图选择的预处理算法; 文献[8]把物化视图选择问题当做类似于Cache中内存管理问题. 文献[9]提出一种动态Cache优化算法DCO(dynamic cache optimization); 文献[10]根据用户查询多样性的特点, 提出了基于粗糙集聚类的物化视图的动态调整算法(RSCDMV). 文献[11]提出的代价模型考虑了使用不同视图维护策略而产生的最小维护代价; 文献[12]同时考虑多查询优化和视图维护优化这两个问题, 提出了IRVSA算法和IMDVSA算法. 文献[13]提出的基于单位空间上的查询频率的视图选择方法(FPUS), 能较好地反映查询的需求. 但FPUS算法没有考虑视图间的依赖关系[14], 其采用的即时调整策略可能会导致物化视图集存在频繁的"抖动"[7].

本文通过深入分析基于用户查询分布和多维数据格, 综合考虑存储空间、视图维护开销、视图维护策略优化和查询性能以及物化视图动态调整, 提出了一种物化视图动态选择的新策略——NDSMMV. 它包括候选视图格图生成算法CVGA、改进的BPUS算法——IGA算法、物化视图集调整算法MAVM和物化视图的动态调整算法DMAVM.

1 物化视图代价模型

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(视图结点). 一个多维数据模式中的所有数据结点构成一个格, 其中:

1) 数据结点间的偏序 $\leqslant$ 定义为给定结点 u 和 v, $u\leqslant v$ 当且仅当仅利用 v 即可计算出 u. 即 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 V_{base}, V_{base} 上各维的级别为该维中最低的. 一般假定 V_{base} 的数据是已知的, 可利用它计算格中任意结点的数据.

在物化视图的选择过程中, 需要再估计数据结点 v 的尺寸, 记为 $|v|$.

在系统构建阶段, 我们粗略地给出一个初始用户查询集 $Q_{\text{set}}=\{q_1, q_2, \cdots, q_n\}$; 而查询集 Q_{set} 中每个查询 q_i 的发生概率为 f_{q_i} ($f_{q_i}\in F_{Q_{\text{set}}}$, $F_{Q_{\text{set}}}$ 为查询概率集)在没有统计数据条件下, 可以根据用户需求和管理员的经验确定.

定义2. 影响视图. 一个查询 q_i 的查询结果通常可以从多维数据格图的多个视图结点计算出, 其中计算代价最小的视图称为查询 q_i 的影响视图 v_i.

对数据仓库的每一个查询 q_i, 可以按其group by子句在多维数据格图上找到相应的影响视图 v_i, 设 v_i 和 v_i 的所有父结点构成的集合为 $father(v_i)$, q_i 可以从 $father(v_i)$ 中的任何一个视图得到查询结果.

定义3. 视图的查询概率. 以视图 v_i 为其影响视图的查询的概率之和称做该视图的查询概率, 用 $p(v_i)$ 表示, 则 $p(v_i)=\sum_{q_i\in Q_i} f_{q_i}$, 其中 $f_{q_i}\in F_{Q_{\text{set}}}$, Q_i 为初始用户查询集 Q_{set} 中以视图 v_i 为其影响视图的查询的集合.

定义4. 物化视图的访问者集合. 对于物化视图 v, 为响应其上的查询而访问 v 但不同于 v 的那

些视图的集合称为物化视图 v 的访问者集合, 记为 $Q(v)$. 显然, 若 $u \in Q(v)$, 则 u 没有被物化, $u \leqslant v$ 且不存在另外的物化视图 w, 使得 $u \leqslant w$ 但 $|u| < |w|$. 若 $u \in Q(v)$, 我们称 $v = Q^{-1}(u)$.

定义 **5.** 物化视图的查询概率. 对于物化视图集 V 中的一个物化视图 v_i, 在 v_i 上响应查询而使得计算代价最小的查询概率之和称为物化视图 v_i 的查询概率, 记为 $P(v_i)$, 显然, $P(v_i) = p(v_i) + \sum_{u \in Q(v_i)} p(u)$.

定义 **6.** 物化视图集 V 的总体查询代价. 若物化视图集 V 中每个物化视图 v_i 的查询概率为 $P(v_i)$, 文献[4]中验证了查询代价与其生成视图的元组数成近似的线性关系, 故查询代价与所依赖的物化视图的尺寸大小具有单调特性. 则整个物化视图的查询代价为 $C_Q(Q_{set}, F_{Q_{set}}, V) = \sum_{v_i \in V} P(v_i) \times |v_i|$.

定义 **7.** 物化视图集 V 的总体更新代价. 若物化视图集 V 中每个视图 v_i 的更新概率为 $R(v_i)$ ($R(v_i) \in F_{U_{set}}$, $F_{U_{set}}$ 为更新概率集), 文献[15]中证明物化视图的更新代价与其自身元组数成近似的线性关系, 故物化视图的更新代价与其尺寸大小也具有单调特性, 则全部物化视图的更新代价为

$$C_U\left(V, F_{U_{set}}\right) = \sum_{v_i \in V} R(v_i) \times |v_i|.$$

定义 **8.** 物化视图集 V 的总体代价. 若给定一个多维数据集 MD 和初始用户查询集 Q_{set}、相应的查询概率集 $F_{Q_{set}}$ 以及视图更新概率集 $F_{U_{set}}$, 则选择生成一个物化视图集 V 的总体代价为

$$TC_V(Q_{set}, F_{Q_{set}}, V, F_{U_{set}}) = C_Q(Q_{set}, F_{Q_{set}}, V) + C_U(V, F_{U_{set}}).$$

2 物化视图动态选择新策略——NDSMMV

2.1 候选视图生成算法——CVGA

文献[16]认为 BPUS 算法及其改进算法的时间复杂度都是指数级, 算法效率较低, 其原因有二: 一是在每一轮迭代中会对物化视图格中的所有视图结点进行评估; 二是在对每个结点评估时考虑了视图对所有后裔产生的影响. 故 NDSMMV 策略把整个物化视图选择过程分成两个阶段: 候选视图选择阶段和视图选择阶段. 在候选视图选择阶段通过调用本文讨论的基于多维数据格候选视图生成算法——CVGA, 选出多维数据格中有希望的部分, 从而大幅度降低了视图选择阶段算法的复杂度. 使策略完全适用于物化视图的动态调整.

定义 **9.** 子视图. 对于视图 v, 若 u 与 v 至少满足下列条件之一:

1) $u \leqslant v$, 即 u 与 v 满足偏序关系且 $|u| < |v|$;

2) $u \in v$, 即 u 可由 v 进行投影或选择操作直接得到, 则称 u 为 v 的子视图, v 的子视图的集合称为 v 的子视图集合, 记为 $H(v)$.

通常用户查询集 Q_{set} 只是对应所有视图结点中的一小部分, 查询稀疏性这一特点表明, 使物化视图总代价取得最小的相关视图只是所有视图中的一部分. 这部分视图即为下面定义的候选视图 CV (candidate views).

定义 **10.** 候选视图. 若一个视图 v_i 满足以下两个条件之一, 则称为候选视图 CV:

① v_i 的查询概率大于零, 即 $p(v_i) > 0$;

② $p(v_i) = 0$, 至少存在两个不同的候选视图 u_1, u_2. 使得 $u_1 \in H(v_i)$, $u_2 \in H(v_i)$, 且 $u_1 \notin H(u_2)$, $u_2 \notin H(u_1)$.

借鉴 PMVS 中候选立方格图的生成算法 CVLC[7], 基于多维数据格的候选视图生成算法如下:

算法 **1.** 候选视图生成算法——CVGA.

Procedure $CVGA(Q_{set}, F_{Q_{set}}, V_{set})$.

输入: 给定的常用查询集合 Q_{set}、常用查询集对应的查询概率集 $F_{Q_{set}}$ 以及多维数据格图 V_{set}.

输出: 候选视图集 CV、候选视图查询概率集 P.

```
CV= ∅; P= ∅;
for each q_i ∈ Q_set
  {根据 q_i 的分组属性, 找到其相应的影响视图
   v_i ∈ V_set;
   p(v_i)= p(v_i) + f_qi;/* f_qi ∈ F_Qset * /
   P= P ∪ p(v_i);}
for each v_i ∈ V_set
  {if p(v_i)> 0 L= L ∪ v_i;}
  M= L; N= ∅;
  while M ≠ ∅
    {for each v_i ∈ M
      {for each v_j ∈ L and v_i ≠ v_j
        {F_i= father(v_i);
         /* each w ∈ L, v_i ⩽ w* /
         F_j= father(v_j);
         /* each w ∈ L, v_j ⩽ w* /
         if v_i ∈ F_j continue;
         if v_j ∈ F_i continue;
```

$R = F_i \cap F_j$;
for each $v_k \in R$
{if $v_k \notin L$ $N = N \cup v_k$;}}}
$M = N$; $L = L \cup N$; $N = \varnothing$;}
$CV = L$;
return (CV, P);

本算法借鉴了 CVLC 的思想[7], 生成候选视图对物化总代价最小的充分性和必要性证明可参阅文献[7].

2.2 物化视图选择算法——IGA

物化视图的选择问题已经被证明是 NP-hard 问题, 不存在多项式时间的算法. 我们采用改进的 BPUS 算法作为算法的第 1 部分, 然后根据查询概率对第 1 步计算得到的物化视图集合进行调整.

定义 **11.** 视图物化的相对单位空间效益. 设已选择的物化视图的集合为 V, $v \notin V$, 则相对于 V 而言, 物化 v 所带来的相对效益为

$$B(v, V) = ((|Q^{-1}(v)| - |v|) \times p(v) + \sum_{u \in Q(v)} (|Q^{-1}(u)| - |v|) \times p(u) - c \times R(v) \times |v|)/|v|.$$

公式中 c 为权重, 由设计者根据视图维护开销相对于查询计算开销的重要程度来指定. 本文指定为 1. $R(v)$ 表示视图 v 的更新概率.

BPUS 算法仅考虑了视图的查询因素, 而我们提出的改进算法 IGA(improved greedy algorithm)则采用基于三元(视图查询、维护和系统存储空间)的综合评价标准. 此外, BPUS 算法虽然体现了视图之间的依赖关系, 但忽略了查询概率分布, 而 IGA 算法从这两方面进行了综合考虑.

算法 **2.** 物化视图选择算法——IGA.

Procedure *IGA (Space, CV)*.

输入: 算法 1 得到的候选视图集合 CV、可利用的空间 $Space$.

输出: 应该被物化的视图集合 V.

$V = V_{base}$; /* 事实表是第 1 个应被物化的视图*/
$Space = Space - |V_{base}|$; $Search =$ True;
$CV = CV - \{V_{base}\}$;
while ($Search ==$ True and $CV \neq \varnothing$)
{select $v \in CV$, 使得 $B(v, CV)$ 最大
if $B(v, CV) < 0$
$Search =$ False;
else
if $Space \geqslant |v|$
{$V = V \cup v$;
$Space = Space - |v|$;
$CV = CV - \{v\}$;}
else
$Search =$ False;}
return $MAMV(Space, CV, V)$;

假设多维数据格图共有 n 个视图结点, 经过算法 1 的处理后, 参与算法 2 的视图数目为 $n/\xi(\xi \geqslant 1)$, 时间复杂度变为 $O(n^2/\xi)$, 即 $\lim_{n\to\infty}(n^2/\xi)/n^2 = 1/\xi$. 随着 ξ 的增加, 算法时间复杂度呈指数趋势急剧下降, 极大地降低了时间复杂度.

2.3 物化视图调整算法——MAMV

BPUS 算法的一个重要缺陷是没有考虑每选择一个新的视图后已选视图的效益值出现衰减, 而这一变化可能会使该视图应从已选视图集中删除. 所以需要根据查询概率对算法 2 得到的物化视图集进行调整, 用查询概率高的视图替代已选出的物化视图集 V 中收益减少太多的视图. 使结果集进一步接近最优解.

算法 **3.** 物化视图调整算法——MAMV.

Procedure *MAMV(Space, CV, V)*.

输入: 算法 2 得到的候选视图集合 CV、可利用的空间 $Space$.

输出: 最终应该被物化的视图集合 V.

$Search =$ True; $V_{temp} = \varnothing$; $V_{remove} = \varnothing$;
while ($CV \neq$ Null and $V \neq$ Null and $Search ==$ True)
{$v = CV$ 中 $f(v)/|v|$ 最大的视图;
$u = V$ 中 $f(u)/|u|$ 最小的视图;
if $(f(v)/|v|) < (f(u)/|u|)$
$Search =$ False;
else
if $Space > |v|$
if $C(V \cup \{v\}) < C(V)$ /* 比较代价*/
{$V = V \cup \{v\}$; $CV = CV - \{v\}$;
$Space = Space - |v|$;}
else
$Search =$ False;
else
{$Search1 =$ True;
while ($Search1 ==$ True and $V \neq$ Null)
{$w = V - V_{remove}$ 中第 1 个视图;
if $((v \in H(w)$ or $w \in H(v))$ and $C(V \cup \{v\} - \{w\}) < C(V))$
if $(Space + |w| \geqslant |v|)$

```
                {V= V∪{v}- {w};
                 CV= CV- {v};
                 Space= Space+ |w|- |v|;
                 Search1= False;
                 V_remove= ∅;}
            else
              if (C(V)> C(V- {w}))
                {V= V- {w};
                 Space= Space+ |w|;}
                  else
                    {V_remove= V_remove ∪{w};
                     if V== V_remove
                       Search1= False;}
                else
                  {V_temp= V_temp ∪{w};
                   V= V- {w};}}
              if (Search1== True and V==
                 Null) or (Search1== False
                 and V_remove ≠Null)
                 CV= CV- {v};
              V= V∪V_temp;
    }
  }
  return V;
```

MAMV 算法的时间复杂度为 $O(n\log_2 n)$. 当系统中经过算法 1 筛选后的候选视图数量还较多时, 可以为这些候选视图按照 $p(v)/|v|$ 建立索引. 这样其时间复杂度仅相当于所选择的物化视图的个数.

2.4 物化视图动态调整算法 ——DMAMV

由于数据仓库的时变性特点, 随着用户的使用, 系统中的查询的分布情况可能发生变化, 使得原有的物化视图集不再适应新的查询分布情况, 为此要求物化视图集及时作出必要的调整. 文献[13] 采用一种实时调整策略——FPUS 算法, 所谓实时调整就是在执行一个查询后立即调整物化视图集合. 但该策略会导致物化集存在频繁的"抖动".

为防止"抖动"发生, 我们选择在一定的统计周期内, 观察多维数据格模型中视图集合的查询分布概率有没有变化, 若当前的统计周期与上一统计周期内视图集合的查询分布概率没有发生变化, 则无须进行物化视图的动态调整; 否则调用物化视图动态调整算法进行物化视图的调整. 如果以一个固定时间间隔为统计周期, 则由于查询发生不是均匀分布, 会导致调整计算具有很大的随意性, 本文以固定的查询发生次数作为一个统计周期. 查询概率 $p_i(n)$ 指的是在第 n 个统计周期 $T(n)$ 里视图 v_i 发生查询的频率(查询次数/统计周期).

定义 **12.** 有效样本空间. 在当前的统计周期 $T(n)$ 内, 发生的 n 个查询事件集合 $\{q_1, q_2, \cdots, q_n\}$, 称为有效样本空间, 记做 $Q_{set}(n)$.

$Q_{set}(n)$ 用来描述当前的查询趋势, 预示未来可能发生查询趋势. 因为对于随机分布的查询, 选取最新的数据才能反映出查询变化, 并可以用来预测未来可能发展的趋势.

定义 **13.** 有效样本集合. 在 $Q_{set}(n)$ 内, 查询事件相对应的影响视图集合为 $\{v_1, v_2, \cdots, v_k\}$, 称为有效样本集合, 记做 $V_{set}(n)$.

通过观察 $Q_{set}(n)$ 内查询代价的数学期望的变化, 对查询视图类型分布变化来进行定量的估计, 数学期望为 $E_n(V) = \sum_{i=1}^{n} p_i(n) \times |v_i|$. 此外还需要通过计算均方差来判断查询代价同 $E(V)$ 的偏离程度. 有效样本空间 $Q_{set}(n)$ 内的方差公式为

$$D_n(V) = \sum_{i=1}^{n} (|v_i| - E_n(V))^2 \times p_i(n).$$

在两个相邻的样本空间内, 若两个视图 v_i 和 v_j, $|v_i| = |v_j|$, 在 $T(n-1)$ 周期中, $p_i(n-1) = p_1$, $p_j(n-1) = p_2$, $p_1 \gg p_2$, 在 $T(n)$ 周期中, $p_i(n) = p_2$, $p_j(n) = p_1$ 而其他的视图的查询概率在两个周期中都不改变, 则其查询代价的数学期望和方差都相等, 但是其视图集合的查询分布概率也发生了本质变化, 所以还需要计算相邻两个样本空间中视图查询概率变化约束条件:

$$G(v) = \sum_{i=1}^{n} ((p_i(n) - p_i(n-1)) \times |v_i|)^2.$$

所以在两个相邻样本空间内如果数学期望 $E_n(V) \approx E_{n-1}(V)$, 且 $D_n(V) \approx D_{n-1}(V)$, $G(v) < up\_limit$, 则两个样本空间查询视图分布基本一致, 反之, 物化视图就需要调整.

算法 **4.** 物化视图动态调整算法 DMAMV.

Procedure $DMAMV(Space, Q_{set}(n), F_{Q_{set}}(n), V_{set}, E_{n-1}(V_{set}(n-1)), D_{n-1}(V_{set}(n-1)))$.

输入: 用户查询集 $Q_{set}(n)$、统计出的该周期内各查询的查询概率集 $F_{Q_{set}}(n)$、多维数据格图 V_{set}、可用空间 $Space$、样本空间 $Q_{set}(n-1)$ 的数学期望 $E_{n-1}(V)$ 和方差 $D_{n-1}(V)$.

输出: 调整后的物化视图 V.

```
(V_set(n), P(n)) = CVGA (Q_set(n), F_Q_set(n), V_set);
    {if Check_Modify (V_set(n), P(n), E_n-1(V_set(n
      - 1)), D_n-1(V_set(n- 1)))
```

IGA(Space, $V_{set}(n)$);
return *V;*}

Check_Modify() 计算数学期望 $E_n(V)$ 和方差 $D_n(V)$, 并同上一个样本空间 $Q_{set}(n-1)$ 的数学期望 $E_{n-1}(V)$ 和方差 $D_{n-1}(V)$ 比较, 此外还要计算 $G(v)$ 并与约束 *up_limit* 比较.

3 实验与性能分析

3.1 实验设计

在现有的各种物化视图静态选择算法中, 较具代表性的有 BPUS 和 PBS, 其中 PBS 算法较快达到 $O(n\log_2 n)$, 然而该算法需要一定的前提条件(要求其格图属于 SR-hypercube lattice), 使该算法在实际应用中受到较大的限制. 而在现有的物化视图动态选择算法中, 最具代表性的是 FPUS 算法. 因此我们为了使算法对比更具有普遍性, 实验采用 NDSMMV 策略与 BPUS 算法以及 FPUS 算法相比较.

测试环境中的硬件平台为联想启天 M4600 (Pentium Ⅳ 2.93GHz CPU, 1GB RAM), 运行 Windows 2003 Sever 操作系统, 数据库平台为 Oracle 9i, 算法用 JBuilder2006 实现.

3.2 性能分析与对比

BPUS 算法的时间复杂度为 $O(kn^2)$, 在允许物化视图数 k 相同的条件下, 算法的时间消耗与结点总数 n 的平方成正比, 而 FPUS 算法的时间复杂度为 $O(n\log_2 n)$. NDSMMV 策略的开销包括 CVGA 生成候选视图格图 $O(d^2+dm)$, d 为与用户查询直接对应的视图数量, m 为生成候选视图的数量, $m=n/\xi$; IGA 算法第 1 步进行初步物化视图选择为 $O(km^2)$; IGA 算法第 2 步调用 MAMV 算法对初步选择的物化视图集进行调整 $O(m\log_2 m)$; 物化视图动态调整算法 DMAMV 中判断是否要动态调整的 *Check_Modify* 算法为 $O(m)$. 则策略的总开销为 $O(km^2)$, m 为生成候选视图的数量.

我们的测试数据集包含 1 个事实表, 每个维表都有 4 个层次. 在 3 个实验中均利用模拟的查询发生器产生 2000 次查询事件, 统计周期为 100 次, 其查询的分布满足 2~8 原则, 即 80% 查询量产生于 20% 的查询.

实验 1 中我们不断增加数据集的维数, 从 3 个维逐渐增加到 7 个维. 每次统计周期结束后分别调用 BPUS 算法、FPUS 算法和 NDSMMV 策略进行比较, 具体的算法实验对比效果如表 1 所示:

Table 1 Comparison of Running Time

表 1 运行时间对比 s

Dimension	Algorithm		
	BPUS	NDSMMV	FPUS
3	0.26	0.37	0.07
4	1.44	0.42	0.10
5	8.43	0.51	0.37
6	46.61	0.61	1.18
7	453.77	0.70	4.03

由表 1 可见, NDSMMV 策略相对于单纯的 BPUS 算法其算法时间开销很低, 完全可以适用于物化视图的在线动态调整. 此外, 在维数较少时三者之中 FPUS 算法时间开销最小, NDSMMV 策略开销最大. 这是因为候选视图生成算法 CVGA 减少的视图数量有限, 使得 $m\approx n$, 故 NDSMMV 策略的算法复杂度为 $O(km^2)\approx O(kn^2)$, 大于 FPUS 算法的 $O(n\log_2 n)$. 此时的 NDSMMV 策略速度甚至不如 BPUS 算法, 主要原因有二, 一是候选视图生成算法 CVGA 处理后减少的视图数量有限, 二是 NDSMMV 策略还要执行候选视图生成算法 CVGA 和物化视图调整算法 MAMV, 所以在时间开销上反而大于 BPUS 算法.

随着维数的增加, BPUS 算法的时间开销增长极为快速, 呈指数增加, 而 NDSMMV 策略的时间开销增加则非常缓慢, 基本上是线性的. 当数据集维数增加后, 由于 NDSMMV 策略中的候选视图算法对视图的筛选作用, 使得参与 NDSMMV 策略选择的视图数得以大幅度减少, 从而大大降低了算法的时间开销. 此时的 NDSMMV 策略的时间开销甚至小于 FPUS 算法.

实验 2 中我们对 3 种算法得到的物化视图集实际的查询响应性能进行比较, 在此本文采用平均响应时间来衡量查询响应能力. 实验中我们在每 500 次查询后, 计算其平均响应时间并进行比较. 具体结果如图 1 所示:

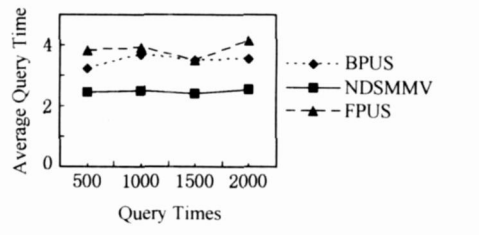

Fig. 1 Comparison of average query time.

图 1 平均查询时间对比

由图 1 可见, NDSMMV 策略所选择的物化视图集在对查询的响应性能方面明显优于 BPUS 算法和 FPUS 算法. 其主要原因是 BPUS 算法是概率无关算法, 选择的视图集合相对固定, 没有考虑查询分布的影响; 而 FPUS 算法虽然考虑了概率因素, 但是忽略了视图之间的内在依赖关系, 而 NDSMMV 策略的 IGA 算法综合考虑了视图之间的依赖关系和查询分布, 而且通过调用物化视图集调整算法 MAMV, 进一步提高了物化视图集的响应查询能力.

实验 3 中我们希望判断 DMAMV 算法中查询代价的数学期望和方差以及修正值是否能够反映用户查询类型分布的变化, 误差是否在容忍范围内. 同时判断当查询类型分布发生变化时, 算法对物化视图集的调整是否改善了查询代价. 实验中我们在每次统计周期结束时执行物化视图的动态调整算法 DMAMV, 为方便结果显示, 我们仅列出了前 1000 次的结果. 如图 2 所示:

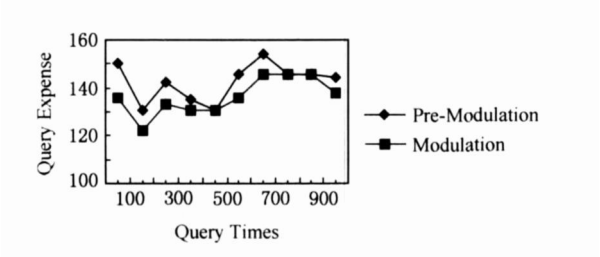

Fig. 2 Query expense comparison fore-and-aft modulation algorithm.

图 2 调整算法前后查询代价比较

我们发现调整后的数学期望总是比调整前要小, 说明算法有效地降低了查询代价. 图 2 中 3 个查询区间没有对物化视图进行调整, 分别在 400~500, 700~800 和 800~900 三个查询区间, 这是因为在这几个区间查询视图分布同上一个查询区间的分布近似, 所以没有进行物化视图调整.

4 结 论

本文提出的 NDSMMV 策略通过减少搜索空间降低了时间复杂度, 同时通过改进 BPUS 算法, 并增加调整算法提高了物化视图集对查询的响应性能, 该策略通过定时地判断视图查询分布是否变化来决定是否进行物化视图的动态调整, 避免了物化视图集的"抖动". 通过分析和实验可以看到该策略是有效可行的.

参 考 文 献

[1] Yu J Xu, Yao Xin, Gou Gang. Materialized view selection as constrained evolutionary optimization [J]. IEEE Trans on Systems, Man, Cybernetics, 2003, 33(4): 458-467

[2] V Harinarayan, A Rajaraman, J D Ullman. Implementing data cubes efficiently [C]. In: Proc of the 1996 ACM SIGMOD Int'l Conf on Management of Data. New York: ACM Press, 1996. 205-216

[3] A Shukla, P M Deshpande, J F Naughton. Materialized view selection for multidimendional datasets [C]. In: Proc of the 24th Int'l Conf on VLDB. San Francisco: Morgan Kaufmann, 1998. 488-499

[4] C Zhang, X Yao, J Yang. An evolutionary approach to materialized views selection in a data warehouse environment [J]. IEEE Trans on Systems, Man and Cybernetics, Part C, 2001, 31(3): 282-294

[5] P Kalnis, N Mamoulis, S Papadias. View selection using randomized search [C]. In: Proc of ACM SIGMOD Int'l Conf on Management of Data. Amsterdam: Elsevier Science Publishers, 2002. 322-333

[6] Himanshu Gupta. Inderpal Singh Mumick. Selection of views to materialize in a data warehouse [J]. IEEE Trans on Knowledge and Data Engineering, 2005, 17(1): 24-43

[7] Zhang Baili, Sun Zhihui, Sun Xiang. Preprocessor of materialized views selection [J]. Journal of Computer Research and Development, 2004, 41(10): 1645-1651 (in Chinese)
(张柏礼, 孙志挥, 孙翔. 物化视图选择的预处理算法[J]. 计算机研究与发展, 2004, 41(10): 1645-1651)

[8] Y Kotidis, N Roussopoulos. A case for dynamic view management [J]. ACM Trans on Database Systems, 2001, 26(4): 388-423

[9] Zhang Baili, Sun Zhihui, Zhou Xiaoyun, *et al*. A dynamic cache optimized algorithm of static materialized views [J]. Journal of Software, 2006, 17(5): 1213-1221 (in Chinese)
(张柏礼, 孙志挥, 周晓云, 等. 静态物化视图的动态 Cache 优化算法[J]. 软件学报, 2006, 17(5): 1213-1221)

[10] Feng Shaorong, Xiao Wenjun. Dynamic materialized view algorithm based on rough set clustering [J]. Computer Engineering, 2007, 33(23): 185-188 (in Chinese)
(冯少荣, 肖文俊. 基于粗糙集聚类的物化视图动态调整算法[J]. 计算机工程, 2007, 33(23): 185-188)

[11] H Mistry, P Roy, S Sudarshan, *et al*. Materialized view selection and maintenance using multi-query optimization [C]. In: Proc of SIGMOD' 01. New York: ACM Press, 2001. 307-318

[12] Noha A R Yousri, Khalil M Ahmed, Nagwa M El-Makky. Algorithms for selecting materialized views in a data warehouse [C]. In: Proc of the ACS/IEEE 2005 International Conference, Los Alamitos, CA: IEEE Computer Society Press, 2005. 27-35

[13] Tan Hongxing, Zhou Longxiang. Dynamic selection of materialized views of multi-dimensional data[J]. Journal of Software, 2002, 13(6): 1090-1096 (in Chinese)
(谭红星, 周龙骧. 多维数据实视图的动态选择[J]. 软件学报, 2002, 13(6): 1090-1096)

[14] C Hurtado, C Gutierrez. Computing cube view dependences in OLAP datacubes[C]. In: Proc of the 15th Int'l Conf on SSDBM. Los Alamitos, CA: IEEE Computer Society Press, 2003. 33-42

[15] Thomas P Nadeau, Toby J Teorey. Achieving scalability in OLAP materialized view selection[C]. In: Proc of DOLAP' 2002. New York: ACM Press, 2002

[16] E Baralis, S Paraboschi, E Teniente. Materialized view selection in a multidimensional database[C]. In: Proc of the 23rd Int'l Conf on VLDB. San Francisco, CA: Morgan Kaufmann, 1997. 156-165

Zhang Dongzhan, born in 1974. He has been teacher of Xiamen University since 2003. Ph. D. and associate professor. His main research interests involve theory and application of database, data warehouse and data mining.
张东站, 1974 年生, 博士, 副教授, 主要研究方向为数据库理论与应用、数据仓库、数据挖掘等.

Huang Zongyi, born in 1981. Master candidate in the Department of Computer Science of Xiamen University. His current research interests involve theory and application of database, distributed database system, data warehouse and data mining.
黄宗毅, 1981 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.

Xue Yongsheng, born in 1946. He has been professor of Xiamen University since 2000. His main research interests involve theory and application of database, distributed database system, data warehouse and data mining.
薛永生, 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.

Research Background

Materialized views for multi-dimensional data can improve the performance of data warehouse, but numbers of materialized views will cost lots of space and time. The materialized view selection problem is one of the most important decisions in designing a data warehouse. In this paper, we present a new dynamic selection strategy of materialized views for multi-dimensional data (NDSMMV), which is composed of four algorithms: CVGA (candidate view generation algorithm), IGA (improved greedy algorithm), MAMV (modulation algorithm of materialized views), and DMAMV (dynamic modulation algorithm of materialized views). CVGA generates the candidate view set based on multi-dimensional data lattice, which reduces the number of candidate views to decrease the space search cost and time consumption of the following algorithm. IGA selects materialized views taking account of view query, view maintenance and space constraint. MAMV modulate the materialized views according to the change of the materialized view profit, which improves the capability of querying materialized views. DMAMV used the sample space to judge whether it is necessary to change the view set and avoid sharp dither. Our work is supported by the National Natural Science Foundation of China (50604012) and the Advanced Technology Foundation of Fujian Province (2003H043).