数据仓库中物化视图选择策略

林小静， 薛永生
（厦门大学 计算机系，福建 厦门 361005）

摘 要：为了提高决策支持和 OLAP 查询的响应效率，数据仓库多采用物化视图的思想。因此，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。提出一个以 MVPP(multi-view processing plan)为视图选择的搜索空间的物化视图选择新算法——VSMF(views selection base on multi-factor)算法。该算法在存储空间约束下同时实现多查询最优化和视图维护最优化。

关键词：数据仓库；物化视图；选择策略；维护策略；存储空间约束
中图法分类号：TP311.131 **文献标识码**：A **文章编号**：1000-7024 (2007) 13-3056-04

Selection strategy of materialized views in data warehouse

LIN Xiao-jing, XUE Yong-sheng
(Department of Computer Science, Xiamen University, Xiamen 361005, China)

Abstract: A set of materialized views are stored in the data warehouse for the purpose of efficiently implementing decision-support or OLAP queries. The selection of materialized views is one of the most important issues in the data warehouse development. The goal is to select an appropriate set of views so that the total cost of storage, maintenance and query is minimized. A new algorithm named VSMF (views selection base on multi-factor) algorithm using multi-view processing plan structure as search space is proposed, which solve the problem considering both multi-query optimization and the maintenance process optimization under the storage space constrain.

Key words: data warehouse; materialized view; selection strategy; maintenance strategy; storage space constrain

0 引 言

当前，数据仓库领域的一个研究热点就是物化视图的选择问题。物化视图是指将查询视图预先计算并以表的形式存储在数据仓库中，当执行OLAP查询时，可直接从物化视图中获取查询结果，避免了对底层数据作复杂的综合操作，从而有效提高查询响应速度。因此，物化视图是提高系统多维分析性能的有效手段。

但是，物化视图也带来了大量存储空间和视图维护的开销，必须在缩短响应时间和资源限制二者之间权衡。由此，物化视图选择的目标就是，在空间限制下，选出一组恰当的视图物化，使得其对一组查询的总查询代价和其自身的维护代价之和为最小。该问题为组合优化问题，其已经被证明为是NP-完全问题。这个问题的最优解的复杂度是 $O(2^n)$，其中，n 是数据仓库中视图的总数。目前存在许多算法，基于各自不同的代价计算模型，通过各种途径求解该问题的近似最优解。

1 相关工作

文献[1]以视图大小为选择准则提出 PBS 算法，在特定前提下，使复杂度降低为 O(n log n)；文献[2]和文献[3]将遗传算法的获取最优解的能力应用于最优物化视图集的选取，并在降低算法复杂度方面进行了研究；文献[4]提出基于单位空间上查询频率的 FPUS。

以上算法都仅仅从视图所占空间大小这一限制条件着眼，而忽略了物化视图的维护时间。而在实际应用中，随着数据存储技术的飞速发展，视图的维护时间逐渐成为限制数据仓库不能物化所有视图的主要因素。

文献[5]提出了以物化视图总维护时间为约束条件的贪心算法 ITGA；文献[6]综合考虑了查询代价和维护代价，并提出了 MVPP(multi-view processing plan)作为视图选择的搜索空间以获得最优解；文献[7]在 MVPP 的基础上，应用遗传算法求解；文献[8]提出了一种结合遗传算法和模拟退火算法的混合算法；文献[9]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。以上算法，都同时考虑了查询代价和维护代价，但都集中在获取最小代价的多查询最优化，而忽略了物化视图维护策略的优化对这个问题的影响。

文献[10]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，然而，在该代价模型中，没有考虑这

收稿日期：2006-07-20 E-mail：lovebysea@126.com
基金项目：福建省自然科学基金项目（A0310008）；福建省重点科技基金项目（2003H043）。
作者简介：林小静（1982－），女，福建连江人，硕士研究生，研究方向为数据仓库、数据挖掘及分布式数据库等；薛永生，男，教授，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。

些视图的查询代价。

因此，Naha A.R.Yousri 和 Khalil M.Ahmed 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[11]，但这两个算法都忽略存储空间的约束。此外，IMDVSA 算法只考虑，使用增量更新策略的情况；而 IRVSA 算法虽然同时考虑两种维护策略，但其与 BPUS 算法主要缺陷相同：首先，其每一步选择都要重新计算所有待选择视图的增益；其次，其每一步选择的视图都将作为将来要物化的视图，没有考虑每选择出一个新的视图后，已选视图的增益将出现衰减，而这一变化可能会导致其应该从已选视图中删除。

综上所述，目前已有的算法大多只考虑一个条件约束，或者只考虑存储空间约束，或者只考虑视图维护时间约束，而后者又大多忽略了视图维护策略的优化所带来的影响。在存储空间和视图维护时间共同制约下，同时考虑查询代价、视图维护代价及视图维护策略优化对维护代价的影响的物化视图选择问题，目前没有提出相应的算法。

本文在上述问题上进行了更为深入的探索，基于 SPJ(select-project-join)视图假设的关系数据库模型，以 MVPP 为搜索空间，综合考虑存储空间、视图维护开销、视图维护策略优化及查询性能，提出了 IMDVSA 的改造算法——VSMF 算法。

2 问题描述

基于 SPJ 视图假设的数据仓库中，在存储空间 Space 的限制下，从 MVPP 中选择出一个视图集合 M 加以物化，使得查询集合 Q 的查询代价和物化视图集 M 的维护代价之和最小。在此，我们假设数据仓库会周期性地更新，并且增量更新和重新计算两种策略在该数据仓库中同时使用。

3 物化视图选择算法—VSMF 算法

3.1 相关定义

定义 1　MVPP(multi-view processing plan)：MVPP 是通过结合给定查询集 Q 中每个查询的最优方案构建起来的，它以有向无环图的形式来描述针对查询集 Q 的一个查询处理策略。

如图 1 所示，MVPP 的叶子结点相当于数据仓库中的基表，其根结点相当于一个查询的最终结果，其所有中间结点及根结点都定义为一个视图。以下的讨论都将基于 MVPP。

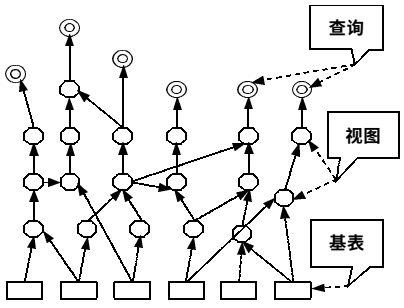

图 1　MVPP

定义 2　子孙结点：视图 u 为视图 v 的子孙结点，当且仅当在 MVPP 中，从视图 u 有一条通路到达视图 v。特别的，若通路中只有一条边，则称 u 为 v 的儿子结点，v 为 u 的父亲结点。

定义 3　一个查询 q 的查询代价 $C(q,M)$：当数据仓库中物化视图集为 M 时，q 的查询代价，等于查询 q 对 M 中所有视图的查询代价的最小值。即：$C(q, M)=MIN\{c(q, v), v\in M\}$。

定义 4　查询集 Q 的总查询代价 TQC(Q,M)：当数据仓库中的物化视图集为 M 时，Q 中所有查询的查询代价和该查询提交频率的乘积的总和。即：$TQC(Q,M)=\sum_{q\in Q}(f_q(q)*C(q, M))$。这里，$f_q(q)$ 是查询 q 的提交频率。

定义 5　物化视图 v 的维护代价 $MC(v,M)$：基于数据仓库中同时使用增量更新和重新计算两种维护策略，因此 v 的维护代价为当数据仓库中物化视图集为 M 时，分别使用两种策略所需代价的最小值。即：$MC(v,M)=\min\{IMC(v,M),RMC(v,M)\}$。

其中，IMC(v, M) 为使用增量更新策略所需的维护代价，RMC(v,M)为使用重新计算策略所需的维护代价。二者均可由文献[10]的计算公式求得。

此处特别提到以数据仓库中物化视图集 M 为前提，这是因为，当使用增量更新策略时，物化视图集 M 是否包含视图 v 的子孙结点，与视图 v 的维护代价大小关系密切，这将后文中详细说明。

定义 6　物化视图集 M 的总维护代价 TMC(M)：M 的总维护代价为 M 中每个视图 v 的维护代价与该视图更新频率的乘积的总和。即：$TMC(M)=\sum_{v\in M}(f_v(v)*MC(v,M))$。其中，$f_v(v)$ 是视图 v 的更新频率。

定义 7　物化视图集 M 的总代价 TC(M)：M 的总代价为当 M 被物化时，M 的总维护代价和查询集 Q 的总查询代价的加权和。即：$TC(M)=\delta*TMC(M)+TQC(Q,M)$。

将 TMC(M)乘以权重因子 $\delta(\delta>0)$，是因为不同的系统对查询性能和视图维护性能的要求不同，系统管理员可以自行设定权重。

定义 8　视图 v 的增益 $B(v, M)$：视图 v 的增益为当物化视图集为 M 时的总代价与物化视图集为 $M\cup\{v\}$ 时的总代价的差值。即：$B(v,M)=TC(M)-TC(M\cup\{v\})$。

定义 9　视图 v 的大小 $S(v)$：视图 v 的大小即其物化所需的存储空间，可采用文献[12]提出的基于数学估算和无重复采样的 Sample Frequency 算法估算。

定义 10　视图 v 单位空间的效益 $BS(v, M)$：物化视图集为 M 时，视图 v 的增益与其所占存储空间的商。即：$BS(v, M)=B(v, M)/S(v)$。

3.2 算法理论基础

当采用增量更新策略进行物化视图的维护时，物化一些额外的视图将降低维护视图集的代价[10]。当更新视图集时，一个物化视图的所有子孙结点由于基表的改变而产生的更新将传播到该物化视图。儿子结点的更新被用于计算父亲结点的更新。在某些情况下，例如：连接操作，不单是儿子结点的更新数据参与计算，而是儿子结点的全部数据都需要参与计算。于是，如果儿子结点已经被物化，则它的计算代价就节省了。同时，由于儿子结点的更新要被用来计算父亲结点的更新，因此物化儿子结点而产生的维护代价就成为父亲结点维护代价的一部分，即不论儿子结点是否物化该结点，都会产生这部分代价。

IMDVSA 算法正是基于这个理论提出的。该算法的优点

是复杂度低，仅为 O (*n*)。但是，该算法仅仅考虑增量更新策略，一旦一个视图确定将被物化，其所有的子孙结点都将被物化。其忽略了重新计算策略更优的可能性，若一个视图重新计算的维护代价更低，则物化该视图所有子孙结点只会增加系统的维护开销。

另外，该算法还忽略了存储空间的约束。在极端情况下，该算法所选出的物化视图集 M 可能包含几乎所有的候选视图，从而导致庞大的存储空间和视图维护代价。

基于上述考虑，提出了 IMDVSA 算法的改进算法——VSMF(views selection base on multi-factor)算法及考虑存储空间约束对物化视图进行调整的算法——MVSCA（modulation of views under space constraint algorithm)算法。

3.3 物化视图的选择

对于一组查询集合 Q，首先使用文献[6]中提到的算法构造出 MVPP，接着使用 VSMF 算法，从 MVPP 中求得视图集后，运行 MVSCA 算法调整物化视图集，使之满足存储空间约束。

算法中 M 表示选出的物化视图集，N 表示未搜索的物化视图集，*n* 表示 MVPP 中的层次，C 表示候选视图集，D(*v*)表示视图 *v* 的所有子孙结点，TS 表示物化视图集 M 所需的存储空间。

算法 1：VSMF 算法

输入：MVPP,查询集 Q

输出：应被物化的视图集 M

过程：按广度优先策略搜索 MVPP 中的视图，计算访问的当前结点(视图)的增益(按定义 8 的公式计算)，增益大于 0，则物化该视图，同时判断该视图采用哪种维护策略更优，若使用增量更新策略更优，则物化该视图的所有子视图，接着将访问过的结点及要物化的结点从搜索空间中删除。

```
VSMF(MVPP,Q)
Begin
  M={MVPP 中所有的视图};
  N={MVPP 中所有的视图};
  C={}；n=0；
  Repeat
    C={所有第n层的视图}∩N;  //未搜索的视图中最上层的视图集
      对 C 中的每个视图 v
        If B (v, M)<0
        Then {
          M=M-{v};
          N=N-{v}; }
        Else  If IMC (v,M)<RMC (v,M)
              Then
                  N=N-{v}∪D (v);
        Else  N=N-{v};
      n=n+1;
  Until n=MVPP 的高;
  Return MVSCA(Space, M);
End
```

算法 2：MVSCA 算法

输入：物化视图集 M，存储空间限制 S;

输出：满足空间约束的物化视图集 M;

过程：估算物化视图集 M 需要的存储空间，若超过空间限制，则采用贪心算法对 M 进行调整。首先计算每个视图的单位空间的增益。一个视图增益越小，所占空间越大，则其单位空间的增益越小，因此，每次选择单位空间的增益最小的视图从 M 中删去，直到 M 满足空间约束。

```
MVSCA(Space, M)
Begin
  TS=Σ S (v),v∈M;
  If TS<Space
  Then Return M;
  Else
    Repeat
      计算 M 中每个视图v单位空间增益 BS (v, M);
      v=M 中 BS (v, M)最小的视图;
      M=M-{v};
      TS=TS-S (v);
  Until TS<Space;
  Return M;
End
```

3.4 算法分析

VSMF 算法综合考虑了查询性能、存储空间约束、物化视图维护时间约束及不同更新策略的影响等因素。算法使用的增益计算模型较其它算法更为合理。算法采用从视图集 M 中逐个删去增益为负的视图的方法，从视图维护代价方面分析，M 中保留的视图或者是采用重新计算策略，或者是采用增量更新策略，后者的所有子孙结点皆保留物化视图中，因此，删去的视图不会影响保留下视图的维护代价；从查询性能方面分析，物化视图集中视图总数减少，保留下的每个视图对系统查询性能的贡献将提高或者不变。

由此可知，M 中保留下的视图的增益不会随着 M 中视图数量的减少而降低，从而避免了 BPUS 等算法中出现的视图增益衰减的弊病。算法同时考虑增量更新和重新计算两种维护策略，对使用重新计算策略更优的视图，避免了物化其对提高查询性能没有贡献的子孙视图，节省了大量视图维护开销，同时也降低了存储空间代价。最后，VSMF 算法调用 MVSCA 算法，调整物化视图集，使之满足空间限制 Space。VSMF 算法的复杂度为 O(n)+ MVSCA 算法的复杂度。MVSCA 算法的复杂度主要来自于选择最小单位空间增益视图的操作，选择恰当的算法，复杂度为 O(n log n)。因此，VSMF 算法的复杂度最差情况下为 O(n log n)，最优情况下，即当给定的空间约束足够大而无需调整视图集，VSMF 算法的复杂度为 O(n)。

4 实验及比较

4.1 实验设计

目前各种物化视图选择算法中，使用 MVPP 为搜索空间且同时考虑查询代价和维护代价的算法主要有 YKL 算法[6]、IMDVSA 算法[11]、IRVSA 算法[11]等，为了实验结果更具代表性，选择考虑不同更新策略的 IRVSA 算法及只考虑增量更新策略的 IMDVSA 算法与 VSMF 算法做比较。

测试环境：硬件平台：P4 3.0GHz，1G RAM；操作系统：Windows 2003 Server；数据库平台：Microsoft SQL Server 2000；算法使用 Jbuilder2006 实现。

4.2 实验分析与对比

我们所选择的测试数据集包含一个事实表，每个维表都有 3 个层次。我们每次都利用模拟查询发生器产生 2 000 次查询时间，查询的分布满足 2~8 原则，即 80%的查询量产生于 20%的查询。对基表更新频率的设定，采用赋予 0~1 之间的随机数的方法。我们从算法时间开销，结果集 M 对查询的平均响应速度，及维护时间 3 个方面进行比较分析。从算法时间开销比较，实验结果如图 2 所示。

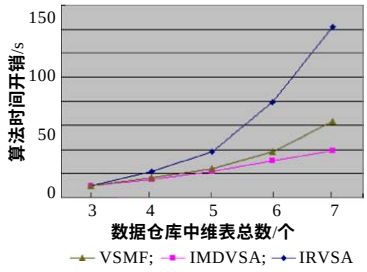

图 2 算法时间开销比较

可以看出，VSMF 算法相对于 IRVSA 算法，算法时间开销很低，相对于 IMDVSA 算法，随着维表的数目增长，VSMF 算法时间开销增长要快些，这是由于当选出的视图集不满足空间约束时，算法会对视图集进行调整。因此，如果给定的空间约束足够大，VSMF 算法的时间开销将不会比 IMDVSA 算法大。结果集对查询的平均响应速度比较，实验结果如图 3 所示。可以看出，3 种算法得到的物化视图集的平均查询响应速度相差不大。

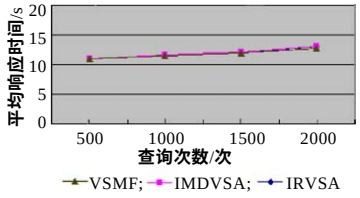

图 3 平均响应时间比较

维护时间的比较：首先从数据仓库的基表中随机选取 10%作为更新基表，之后每次更新基表的数量以 10%递增。设定每张表更新数据量为该表数据量的 10%，可得到实验结果如图 4 所示，可以看出，VSMF 算法的视图维护性能优于 IMDVSA 算法，而较 IRVSA 算法略差，但其在算法时间开销上的优势足以弥补这一缺陷。

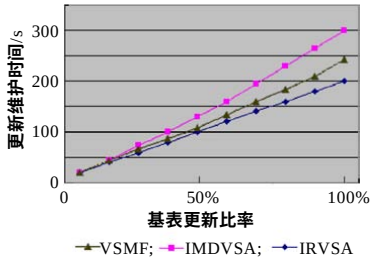

图 4 更新维护时间比较

5 结束语

本文提出的 VSMF 算法综合考虑数据仓库中影响系统查询性能和维护性能的各种因素，使用更为合理的增益估算模型和广度优先搜索，并使用 MVPP 削减算法的搜索空间。实验和分析表明算法的复杂度低而得到结果集具有较好的查询响应性能和较低维护开销。

参考文献:

[1] Clarke I, Sandberg O, Wiley B, et al. Freenet: A distributed anonymous information storage and retrieval system[C]. Proc of the Workshop on Design Issues in Anonymity and Unobservability. Berlin: Springer-Verlag, 2001: 46-66.

[2] Joseph S R H. NeuroGrid: Semantically routing queries in peer-to-peer networks[C].Pisa: International Workshop on Peer-to-Peer Computing, 2002:78-90.

[3] Tang Chunqiang, Xu zhichen, Dwarkada S. Peer-to-peer information retrieval using self-organizing semantic overlay networks [C]. Karlsruhe, Germany: Proc of SIGCOMM Conf, 2003.

[4] Cohen E, Fiat A, Kaplan H. Associative search in peer to peer networks: harnessing latent semantics [C]. The 22nd Annual Joint Conf of the IEEE Computer and Communications Societies. California: IEEE Computer Society Press, 2003: 1261-1271.

[5] Sripanidkulchai K, Maggs B, Zhang H. Efficient content location using interest-based locality in peer-to-peer systems [C]. Proc of Infocom, 2003.

[6] Yang J, Karlapalem K, Li Q. Algorithms for materialized view design in data warehousing environment[C]. Athens, Greece: Proc of the 23rd International Conference of Very Large Data Bases,1997:136-145.

[7] Horng Jorng-Tzong,Chang Yu-Jan,Lin Baw-Jhiune, et al. Materialized view selection using genetic algorithms in a data warehouse system[C]. Washington: Proc of the Congress of Evolutionary Computation, 1999: 2221-2227.

[8] 徐海涛,郑宁.数据仓库中物化视图选择的一种混合算法[J].计算机工程与设计,2005,26(10):194-197.

[9] Himanshu Gupta, Inderpal Singh Mumick. Selection of views to materialize in a data warehouse[J]. IEEE, 2005(17):24-43.

[10] Mistry H, Roy P, Sudarshan S, et al. Materialized view selection and maintenance using multi-query optimization [C]. Proceedings of SIGMOD'01, 2001: 307-318.

[11] Noha A R Yousri, Khalil M Ahmed, Nagwa M Ei-Makky. Algorithms for selecting materialized views in a data warehouse[J]. IEEE, 2005(5):27-35.

[12] Runapongsa K, Nadeau T P, Teorey T J. Storage estimation for multidimensional aggregates in OLAP [C]. Proc of the 10th CASCON Conf. Orlando, USA: IBM Press, 1999:40-54.