基于多维护策略的物化视图选择方法*)

崔晓军[1,2] 薛永生[1] 张东站[1] 黄宗毅[1]
(厦门大学计算机科学系 厦门 361005)[1] (襄樊职业技术学院 襄樊 441050)[2]

摘 要 物化视图是数据仓库环境中提高OLAP查询效率的重要手段,因此,物化视图的选择是数据仓库设计中重要的决策之一。本文提出的物化视图选择方法目标是选择合适的视图进行物化,使得查询处理的总代价和物化视图的维护代价最低,提出了物化视图收益模型,并在此基础上基于视图的多维护策略提出了物化视图选择的方法:基于增量和重计算的物化视图选择算法IRMVS、基于增量策略的物化视图选择算法IMVS和基于重计算策略的物化视图选择算法RMVS和基于增量策略的物化后代视图选择算法IMDVS,理论分析和实验表明这些算法是有效可行的。

关键词 物化视图,收益模型,多查询优化,增量策略,重计算策略

Approaches for Selecting Views to Materialize Based on Multi-Maintenance Strategy

CUI Xiao-Jun[1,2] XUE Yong-Sheng[1] ZHANG Dong-Zhan[1] HUANG Zong-Yi[1]
(Department of Computer Science, Xiamen University, Xiamen 361005)[1]
(Xiang Fan Vocational and Technical College, Xiangfan 441050)[2]

Abstract A data warehouse stores materialized views, with the purpose of efficiently implementing OLAP queries or decision-support. Hence, selecting views to materialize is one of the most important decisions in designing a data warehouse. In this paper, we present a framework for analyzing the issues , which goal is to select an appropriate set of views so that the sum cost of processing queries and maintaining the materialized views is minimized. Based on the proposed benefit model, we proposed two approaches. The target of first approach is to solve the problem considering both multi-query optimization and the maintenance process optimization. In this approach, we present three algorithms: IRMVS, IMVS and RMVS. The second approach uses a simple search strategy that can cut down the time complexity to a linear. The comparative experiment indicates that these algorithms are efficient and feasible.

Keywords Materialized view, Benefit mode, Multi-query optimization, Incremental strategy, Recomputation strategy

1 引言

数据仓库中存放了大量的来自多个分布的、异质的数据源的数据,用于查询和分析。如何提高查询响应速度是当前数据仓库领域研究的热点之一。目前,一些查询优化技术[1~3]和索引[4]技术可以在一定程度上减少查询响应时间,但更常用的提高查询响应时间的技术是物化视图,即将一部分查询视图预先进行计算并物理存储,以有效地加快数据仓库对查询的响应速度,它已成为提高OLAP性能的重要手段之一。

物化视图的选择问题是如何最小化查询处理代价和物化视图维护代价之和,这个问题已经被证明为NP-完全问题[5]。因此,许多文献提出了利用启发式视图选择的算法框架,均根据特定的收益模型来进行计算。在文[6~10]中分别提出了不同的代价函数。

大多数算法在处理物化视图选择问题时没有考虑到不同的维护策略所带来的对视图维护优化过程的影响。文[8]为了降低已存在的物化视图集的维护代价,提出了物化额外的视图的方法,使用不同的维护策略的代价模型,但其模型没有考虑查询视图的代价。另一方面,文[10]提出了一种既考虑查询代价又考虑维护代价的代价模型,在此模型基础上提出了选择物化视图的算法,为了获得多查询优化,提出了一种搜索空间MVPP(Multi-View Processing Plan),但为了降低查询代价而进行多查询优化的同时忽略了视图维护过程的优化,只考虑了重计算的维护策略。

本文在上述文献的基础上提出了新的物化视图选择方法,既考虑了多查询优化,又考虑了视图维护过程的优化,这是本文研究的一个目标。

减少视图选择问题的搜索空间是本文研究的另一个方面。物化视图选择问题的最优解具有$O(2^n)$的时间复杂度,n是候选视图的个数。一些启发式的算法具有$O(n^2)$时间复杂度。因此,提出一种简单的能将复杂度降至$O(n)$的搜索策略是值得考虑的。

本文第2节提出物化视图选择方法的主要思想,并给出问题的定义和收益模型的描述;第3节给出具体的算法描述;第4节给出实验结果与分析;最后对本文的工作进行了总结。

2 物化视图选择方法

2.1 概述

基于选择-投影-连接(SPJ)视图的关系数据模型在物化

*)基金项目:国家自然科学基金项目(50474033),福建省自然科学基金项目(A0310008),福建省高新技术研究开放计划重点项目(2003H043)。
崔晓军 硕士研究生,副教授,主要研究方向为数据仓库、数据挖掘、XML等。**薛永生** 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、网络技术等。**张东部** 博士,讲师,主要研究方向为数据库理论与应用。**黄宗毅** 硕士,主要研究方向为数据仓库、数据挖掘。

视图的选择问题上，不仅有存储空间的限制，还要综合考虑查询处理的代价和物化视图的维护代价。对于物化视图选择问题的目标就是在一定的空间限制下，选择合适的视图集合，使得查询和维护的总代价最小。

本文的研究工作是在给定的多视图查询处理方案(MVPP)[10]基础上选择物化的视图，假定数据仓库采用定期更新(增量或重计算)的方式，提出了两种物化视图选择的方法。

在方法一中，提出了基于相同的搜索策略而不同维护策略的三种算法：(1)基于增量/重计算策略的物化视图选择算法 IRMVS(Incremental Recomputation Strategy Materialized Views Selection Algorithm)；(2)基于增量策略的物化视图选择算法 IMVS(Incremental strategy Materialized Views Selection Algorithm)；(3)基于重计算策略的物化视图选择算法 RMVS(Recomputation strategy Materialized Views Selection Algorithm)。

在方法二中提出了一种算法：基于增量策略的物化后代视图选择算法 IMDVS(Incremental strategy Materialized Descendants View Selection Algorithm)，该算法只考虑了增量更新策略。

2.2 MVPP

物化视图的选择可借助于 MVPP 实现。文[13]指出，MVPP 描述了数据仓库需要维护的所有视图(物化或虚拟的)，它用一个带有根结点的有向无环图来表示对仓库视图的查询处理策略。其中，叶子结点(R_i)代表基本关系，根结点(Q_i)代表对仓库进行的查询，所有的对应于查询结果或中间结果的顶点(V_i)定义为视图。图 1 给出了一个 MVPP 的例子。

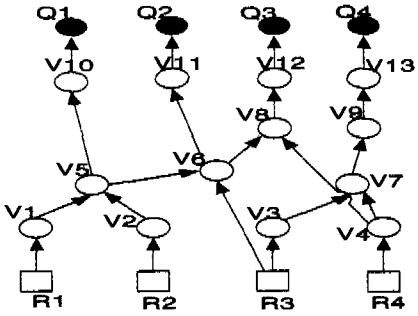

图 1　多查询处理方案 MVPP

2.3 物化视图的收益模型

给定一个 MVPP 及提交查询的频率 f_q 和视图更新的频率 f_u 作为输入，目标是获取视图 v 的收益。一个视图如果具有正的收益值则进行物化，否则保持虚拟。

视图收益值的计算依赖于总代价的计算。视图的总代价由两部分组成：所有查询的总查询代价和所有物化视图的总维护代价。

定义 1　视图 v 的收益 $B(v|M)$，表示在给定物化视图集合 M 的条件下，物化视图 v 所获取的收益：

$$B(v|M)=TC(M)-TC(M\cup\{v\}) \tag{1}$$

其中，$TC(M)$ 表示物化视图集合为 M 时的总代价，$TC(M\cup\{v\})$ 表示将 v 加入到集合 M 后的总代价。如果 $B(v|M)$ 为正值，则视图 v 的物化将导致总代价减少，即收益增加，应将视图 v 物化。

定义 2　物化视图集 M 的总代价 $TC(M)$，表示所有物化视图的集合为 M 时，总查询代价和总物化代价之和：

$$TC(M)=TQC(Q|M)+aTMC(M) \tag{2}$$

其中，$TQC(Q|M)$ 表示总的查询代价，$TMC(M)$ 表示总的维护代价，a 是权重，表示对于查询代价和维护代价关注程度的不同，本文取为 1。

定义 3　物化视图集 M 的总查询代价：

$$TQC(Q|M)=\sum_{q\in Q}(f_q(q))^*c(q|M)) \tag{3}$$

其中，$f_q(q)$ 是提交查询 q 的频率，$c(q|M)$ 是在给定物化视图集 M 的条件下查询 q 的查询代价。

定义 4　物化视图集 M 的总维护代价：

$$TMC(M)=\sum_{v\in M}(f_u(v)^*mc(v|M)) \tag{4}$$

其中，$f_u(v)$ 是视图 v 的更新频率，$mc(v|M)$ 是物化视图集为 M 时视图 v 的维护代价。

定义 5　视图 v 的维护代价 $mc(v|M)$：

$$mc(v|M)=\min(imc(v|M),rmc(v|M)) \tag{5}$$

其中，$imc(v|M)$ 是给定物化视图集 M 时视图 v 的增量更新代价，$rmc(v|M)$ 是给定物化视图集 M 时视图 v 的重计算更新代价，$mc(v|M)$ 取两者中低值。

定义 6　视图 v 的增量更新代价 $imc(v|M)$：

$$imc(v|M)=\begin{cases}\sum_{k=1}^{5}(c(\delta_v^k|M)+mergc(\delta_v^k)) & \text{(a)}\\ c(\delta v|M)+mergc(\delta v) & \text{(b)}\end{cases} \tag{6}$$

其中，(a)式表示连接操作(涉及多个子结点的连接操作可分解为多个只有 2 个子结点的连接操作的组合)时的更新代价；(b)式表示选择和投影操作时的更新代价。δ_v^k 用来计算连接操作时视图 v 的增量，计算如下：若视图 v 的子结点为 $v1$ 和 $v2$，则 $\delta_v^1:\delta_{v1}\infty_{v_2}$，$\delta_v^2:v1\infty\delta_{v2}$，$\delta_v^3:\delta_{v1}\infty\delta_{v2}$，$\delta_v^4:\delta_{v1}\infty\delta'_{v2}(v_2\in R)$，$\delta_v^5:\delta'_{v_1}\infty\delta_{v_2}(v_1\in R)$，其中 R 是基本关系的集合。视图 v 增量计算公式如下：

$$\delta_{v_1}\infty v_2+v_1\infty\delta_{v_2}-\delta_{v_1}\infty\delta_{v_2}-\delta_{v_1}\infty\delta'_{c_2}-\delta'_{v_1}\infty\delta_{v_2}$$

令 t_u 为维护开始时刻，t'_u 为上一次维护处理的开始时刻，v_1 和 v_2 是待更新的视图，δ_{v_1} 和 δ_{v_2} 是时刻 t'_u 和 t_u 间 v_1 和 v_2 的变化量，δ'_{c_1} 和 δ'_{v_2} 是在时刻 t_u 和查询 v_1 和 v_2 计算改变的时刻之间 v_1 和 v_2 的变化量。当 $v\notin R$ 时 $\delta'v=\{\}$。对于选择和投影操作而言，视图 v 由一个顶点 v_1 计算得到，视图 v 的增量计算如下：选择操作时 $\delta_v:\sigma(\delta_{v1})$，投影操作时 $\delta c:\pi(\delta_{v_1})$。$c(\delta_v^k|M)$ 是给定物化视图集合 M 时 δ_v^k 的计算代价，$mergc(\delta_v^k)$ 是合并 δ_v^k 和视图中原有数据的代价。

定义 7　视图 v 的重计算更新代价 $rmc(v|M)$：

$$rmc(v|M)=c(v|M)+mvc(v) \tag{7}$$

其中，$c(v|M)$ 是视图 v 的计算代价，$mvc(v)$ 是视图 v 的物化代价。

定义 8　视图 v 的计算代价 $c(v|M)$：

$$c(v|M)=\begin{cases}Localc(v)+\sum_{u\in S(v)}childc(u|M) & S(v)\neq\phi\\ commc(v) & S(v)=\phi\end{cases} \tag{8}$$

其中，$S(v)$ 是视图 v 的子结点的集合(MVPP 中有边指向 v 的结点)，$Localc(v)$ 是视图 v 的本地操作代价，$childc(u|M)$ 计算子结点 u 的代价，$commc(v)$ 是为了计算基本关系 v 的父结点时从基本关系 v 存储位置到数据仓库存储位置的通讯代价，它是传输数据量的线性函数，计算如下：$commc(v)=C_{MSG}+C_{TR}^*$(传输字节数)，其中 C_{MSG} 是建立通讯的固定代价，C_{TR} 是传输单位数据的代价。

定义 9 子视图 u 的计算代价 $childc(u|m)$：

$$childc(u|M)=\begin{cases}reuse(u) & u\in m\\ c(u|M) & u\notin M\end{cases} \quad (9)$$

其中，$reuse(u)$是物化视图 u 的重用代价，$c(u|M)$是视图 u 的计算代价。上式的计算是一个递归过程，当计算到达叶子结点(基本关系)或物化的子结点时，递归结束。

定义 10 查询代价 $c(q|M)$，表示给定一个视图 v 对应于查询 q 的最终结果，所计算的查询 q 的查询代价：

$$c(q|M)=\begin{cases}Access(v) & v\in M\\ c(v|M) & v\notin M\end{cases} \quad (10)$$

其中，$Access(v)$是访问物化视图 v 的访问代价，*Harinarayan* 在文[11]中验证了访问代价与视图的元组数成近似的线性关系。$c(v|m)$是给定物化视图集合 M 时视图 v 的计算代价。

定义 11 物化视图的空间约束为：

$$S(M)=\sum_{v\in M}size(v)<SPACE \quad (11)$$

其中，$SPACE$ 表示系统分配给物化视图的存储空间，$size(v)$是视图 v 物化所需的存储空间，它与 v 中元组数成近似线性关系[11]。

3 算法描述

3.1 IRMVS 算法及其变形

IRMVS(基于增量/重计算策略的物化视图选择)算法是方法一中的主要算法，可以变形为 IMVS 算法和 RMVS 算法，其算法描述见算法 1。

在该算法中使用到的集合有：(1)物化视图集合 M，初始为空；(2)候选视图集合 C，初始包括所有的视图；(3)顺序表 L，是按照视图初始收益值的降序排列的所有的视图集合。(4)临时集合 Mt，用于存储中间计算得到的物化视图，初始为空。

IRMVS 算法的策略：对于所有的候选视图，分别计算其单独的收益值，按照收益值的降序将所有的视图插入到顺序表 L 中。然后每次从 L 中取出一个视图，根据公式(1)计算将其物化的收益 $B(v|Mt)$，如果 $B(v|Mt)>0$ 则将其加入集合 Mt，并将其从顺序表 L 中删除，若 $B(v|Mt)<0$ 则直接将其从顺序表 L 中删除。重复上述过程，直至 L 为空。然后根据空间约束判断 Mt 中的视图结点可否物化，将可物化的结点加入到集合 M 中。

算法 1 基于增量/重计算策略的物化视图选择算法 IRMVS。

```
输入:候选视图集合 C //MVPP 中所有视图,分配给物化视图的存储
     空间 SPACE。
输出:物化视图集合 M
IRMVS(C,SPACE)
{/*(1)初始化*/
   Initialize(M)=∅; //集合 M 和 Mt 初始为 Φ
   Initialize(Mt)=∅; //集合 Mt 初始为 Φ
   Initialize(L) =∅; //顺序表 L 初始为空链表
/*(2)构造顺序链表 L*/
while(C≠∅)
{ GetV(C); //从集合 C 中顺序取一个结点 v
  Calc(B(v|M);
/*计算结点 v 的初始收益 B(v|M),此时 M 为空*/
   Insert(v,L);
/*将结点 v 插入链表 L 相应位置,按 B(v|M)降序排列*/
   Delete(v,C); //将结点 v 从 C 中删除
}
/*(3)构造临时物化视图集合 Mt*/
While(L≠∅)
{ GetFirstV(L); //从 L 中取出首结点
  Calc(B(v|Mt)); //计算结点 v 的收益 B(v|Mt)
  If(B(v|Mt)>0) Mt=Mt∪{v};
  Delete(v,L); //删除 L 中的当前结点
}
/*(4)检查空间约束*/
   Nm=count(Mt); //计算 Mt 中结点个数
   S(Mt)=0; //物化视图所需空间
   for(i=1;i<=Nm;i++)
   { S(Mt) = S(Mt)+size(Vi);
/*集合 Mt 中前 i 个视图所需的存储空间*/
     if(S(Mt)<SPACE) Copy(Mt,M,i);
/*将集合 Mt 的第 i 个结点拷入集合 M*/
       else break;
   }
Return M;
}
```

RMVS(基于重计算策略的物化视图选择)算法是 IRMVS 算法的一种变形，即在收益模型中只考虑重计算更新策略，式(5)简化为：

$$mc(v|M)=rmc(v|M) \quad (5')。$$

提出这个算法的目的是对于只使用重计算更新策略的系统给出一个解决方案。

IMVS(基于增量策略的物化视图选择)算法是 IRMVS 算法的另一种变形，即在收益模型中只考虑增量更新策略，式(5)简化为：

$$mc(v|M)=imc(v|M) \quad (5'')$$

提出这个算法的目的是对于只使用增量更新策略的系统给出一个解决方案。

3.2 IMDVS 算法

IMDVS(基于增量策略的物化后代视图选择)算法提出了一种快速剪枝的简单搜索策略，该算法只考虑增量更新维护策略。算法的基本思想是：当物化一个视图时，也同时物化其后代视图(不考虑存储空间的约束)。决定物化一个视图时，将以该结点为根的所有子结点同时从搜索空间中剪掉，这样可大大降低搜索时间。

当数据仓库中某个视图需要更新时，其所有的后代视图也必定需要更新，因为它们基于相同的基本关系。基本关系的更新导致子视图的更新计算，进而导致父视图的更新计算。考虑连接操作时，父视图的更新计算不仅用到子视图的增量，而且需要用到子视图中的所有元组。如果子视图物化，则它的计算代价就可以节省了，子视图的更新计算将被用于父视图的更新，所以物化子视图代价仅仅是其元组的物理更新(插入和删除)。如果增量更新时的主要代价是计算更新的代价而不是元组的物理更新，那么物化子视图所带来的维护代价的减少将超过其元组的物理更新所增加的代价。

收益模型：IMDVS 所使用的收益模型与 2.3 节的收益模型不同，每个视图 v 的收益均独立计算，视图 v 的收益计算见式(12)：

$$B(v)=\sum_{q\in Q}f_q(v)*c(v|M)-f_u(v)*imc(v|M) \quad (12)$$

其中，Q 表示查询的集合，$f_q(v)$表查询 q 访问结点 v 的频率，$f_u(v)$是视图 v 更新的频率，$c(v|M)$是视图 v 在物化视图集为 M 时的计算代价(见定义 8)，$imc(v|M)$是视图 v 的增量更新代价(见定义 6)。

IMDVS 算法的策略：给定一个 MVPP 查询树，树中每个结点代表一个候选视图，对于每个结点已知每个查询对该视图的访问频率 $\sum_{q\in Q}f_q(v)$，和该视图的更新频率 $f_u(v)$。对 MVPP 进行广度优先的搜索策略，即先遍历所有的根结点，然后再遍历它们的子结点，只到终止条件(该结点将被物化或其子结点属于基本关系 R 的集合)满足。对于每个结点，只有经过计算检查其不物化，才继续检查其子结点；如果该结点经计算需物化，则其所有的后代结点均不需要再访问，直接标记为物化，达到快速剪枝的目的。算法描述见算法 2，其中用到的集合有：(1)候选视图集合 C，即 MVPP 中所有的视图；

(2)物化视图集合 M,初始为所有视图;(3)待访问结点集合 VISIT,初始为空;(4)后代视图集 $\sum_{v\in C} D(v)$,存放每个视图的后代视图。

算法 2 基于增量策略的物化后代视图选择算法 IMDVS。

输入:候选视图集合 C,查询对视图的访问频率集合 $F_{qv}=\sum_{v\in C}\sum_{q\in Q} f_q(v)$,视图的更新频率集合 $F_u=\sum_{v\in C} f_u(v)$,视图的后代结点集合 $\sum_{v\in C} D(v)$

输出:物化视图集合 M

```
IMDVS(C,Fqv,Fu)
{/*(1)初始化*/
   Initialize(M)=C; //M 初始化为全部视图集合
   Initialize(VISIT)=∅; //VLIST 初始化为空
 *(2)广度优先遍历 MVPP*/
  while(C≠∅)
  { VISIT=C 中的顶层结点;
  /*每次存放 MVPP 中的一层结点,第一次为所有根结点*/
       for each(v) in VISIT
       { if(B(v)<0) //由式(12)计算
       {M=M-v;
       C=C-v ; }
       else
  /*从搜索空间 C 中删除结点 v 及后代结点 D(v)*/
      C=C-{v∪D(v)};
      VISIT=VISIT-v; }
      }
}
```

4 实验与性能分析

4.1 实验设计

本文的算法均在给定的 MVPP 上,选择合适的视图集进行物化。同类算法中具有代表性的选择算法有文[10]提出的 YKL 算法,因此,实验设计为本文中的几种算法与 YKL 算法在不同条件下进行比较。

实验方案设计为如下三组:(1)不同更新百分比下的性能比较;(2)不同查询负载下的性能比较;(3)不同记录个数下的性能比较。

在第(1)组实验中,数据仓库中的数据以 10%的增长比例从 10%变化到 100%。在第(2)组实验中,查询负载由 10 次/小时变化到 360 次/小时。在第(3)组实验中,数据仓库中记录个数在前次基础上每次增加 10%。

实验数据来源于 footmart2000,实验环境中的硬件平台为 DELL OPTIPLEX GX270 (P4 2.60GHz CPU, 512M RAM),运行 Windows 2000 Sever 操作系统,数据库平台为 SQL Server2000,算法用 C#.net 实现。

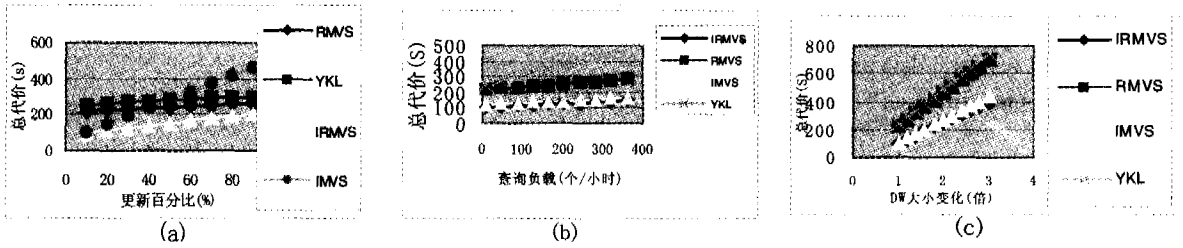

图 2 本文算法与 YKL 算法的性能比较

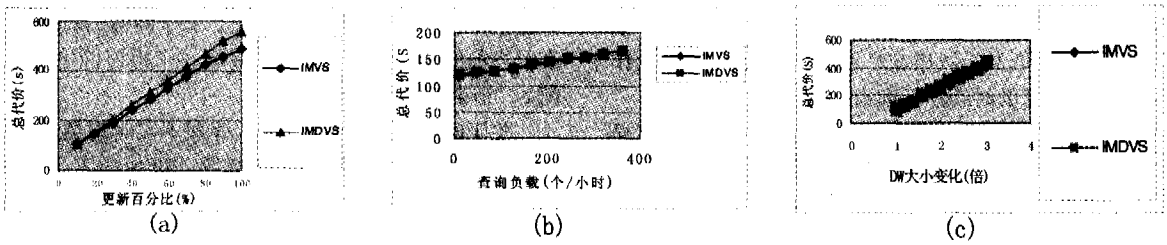

图 3 IMVS 与 IMDVS 性能比较

4.2 性能分析

实验结果如图 2 和图 3 所示,从结果中可看出:

(1) 对于只考虑重计算更新策略的算法而言,算法 RMVS 胜过算法 YKL。这是由于算法 RMVS 中收益模型是基于总代价的计算,而算法 YKL 中收益模型是基于独立代价的计算。

(2) 算法 IRMVS 的性能比其它算法都要优越,在更新百分比较小时性能与算法 IMVS 接近,在更新百分比较大时与算法 RMVS 接近。

算法 IRMVS 由于同样的原因胜过算法 YKL,另外,由于其考虑了维护代价的优化,而不是只考虑一种更新策略,所以它的性能比 IMVS 和 RMVS 优越,这也是其在更新比率较小时与 IMVS 接近,而在更新比率较大时与 RMVS 接近的原因。

(3) 对于只考虑增量更新策略的算法而言,算法 IMVS 和算法 IMDVS 性能基本相近,只是在更新百分比较大时,算法 IMDVS 性能略逊于算法 IMVS。

从算法策略上分析,算法 IMVS 要比算法 IMDVS 优越,但在某些情况下两者性能会比较接近,即两种算法都物化那些查询所需的视图,区别只在于 IMDVS 物化这些视图的所有后代视图,而 IMVS 只物化其部分后代视图。这时,两种算法的查询代价相同,而维护代价不同。如果物化后代视图带来的查询视图维护代价的减少与后代视图本身的维护代价相似,则两者性能接近。

结论 本文提出了两种在数据仓库中选择物化视图的方法,第一种方法综合考虑了多查询优化和维护代价的优化,给出了三个算法:IRMVS、RMVS 和 IMVS。第二种方法在考虑多查询优化的同时,重点放在如何减少问题的搜索空间,给出的 IMDVS 算法只考虑了增量更新的维护策略。实验结果表明这些算法是有效的。IRMVS 算法是一个综合性能最好,适用于各种维护策略;IMVS 算法适用于增量更新策略;RMVS 算法适用于重计算更新策略;IMDVS 算法可显著降低时间复杂度,并且能获得较好的性能,便于数据仓库管理员可根

(下转第 241 页)

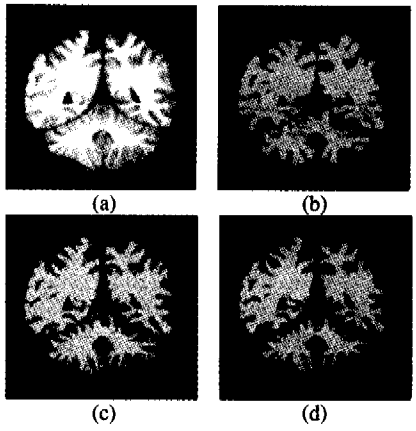

图4 真实图像分割 (a)原图像 (b)RFCM 聚类结果图 (c)仅用全局阈值二次分割后的结果图 (d)采用本文提出的二次分割方法处理后的结果图像

法，结果图像如图 4d。从视觉观察上比较这三个结果图像，本文的方法明显优于其它的两幅图像(图 4b 和 c)，尤其是在灰质与白质，灰质与脑脊液的过渡区域。

结论 医学图像分割问题一直是图像处理领域中的经典难题。大脑组织具有特别复杂的结构，为了得到脑部病变组织的尺寸、外观的量化信息和实现脑部结构的三维重构，脑组织图像分割显得格外关键。本文综合运用小波算法、分水岭算法及基于区域的模糊 C 均值算法，并提出了一种基于规则的二次分割方法实现对脑组织磁共振图像的分割。首先，采用一种基于小波的滤波器去除图像中的噪声；然后采用分水岭算法实现对图像的初始分割。通常，传统分水岭算法在对灰度尺度纹理图像，尤其是组织图像分割中，常常出现过度分割的现象。为了解决分水岭算法的过度分割问题，本文提出了一种基于区域的模糊 C 均值(RFCM)聚类算法实现对过度分割区域的合并。尽管，分水岭算法存在过度分割现象，仍有一些区域的分割并不完全，尤其是在脑脊液与灰质，或灰质与白质的过渡区域。而且，在 RFCM 聚类中每一个区域被假定与它的领域区域无关，也不考虑区域间的空间关系。然而，对于图像区域而言，领域区域间有着很强的相互关系。综合上述问题，本文提出一种局部区域连续性与全局信息相结合的基于规则的多阈值分割方法，对分水岭算法初始分割不完全的区域再次分割。文中多种方法巧妙地相互结合，实现了对脑组织磁共振图像分割，并通过对大量模拟数据和真实数据的分割实验证明本文所提出方法的有效性和精确性。

参 考 文 献

1 聂生东，聂斌，章鲁，等. 基于模糊 K-近邻规则的多谱磁共振脑图像分割方法的研究. 中国生物医学工程学报，2002，21(10)
2 Sahoo PK, Soltani S, Wong AKC, Chen YC. A survey of thresholding techniques. Computer Vision Graphical Image Process, 1988, 41(2): 233～260
3 Huang LK, Wang MJ. Image thresholding by minimizing the measures of fuzziness. Pattern Recognition, 1995, 28(1): 41～51
4 Tobias OJ, Seara R. Image segmentation by histogram thresholding using fuzzy sets. IEEE Trans. on Image Processing, 2002, 11(12): 1457～1465
5 Kundu A, Mitra SK. A new algorithm for image edge extraction using a statistical classifier approach. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1987, 9(4): 569～577
6 Liow YT. A contour tracing algorithm that preserves common boundaries between regions. CVGIP-Image Understanding, 1991, 53(3): 313～321
7 Adams R, Bischof L. Seeded region growing. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1994, 16(6): 641～647
8 Mehnert A, Jackway P. An improved seeded region growing algorithm. Pattern Recognition Letters, 1997, 18(10): 1065～1071
9 Vincent L, Soille P. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. IEEE Trans. on Pattern Analysis and Machine Intelligence, 1991, 13(6): 583～598
10 Pizurica A, Philips W, Lemahieu I, et al. A versatile wavelet domain noise filtration technique for medical imaging. IEEE Trans. Med. Imaging, 2003, 22 (3): 323～331
11 Kwan R S, Evans A, Pike G. MRI simulation-based evaluation of image-processing and classification methods. IEEE Trans. Med. Imaging, 1999, 18 (11): 1085～1097. Availableat: http://www.bic.mni.mcgill.ca/brainweb
12 Collins S H. Terrain parameters directly from a digital terrain model. Canadian Surveyor, 1975, 29(5): 507～518
13 Isaac E J, Singleton R C. Sorting by address calculation. J. ACM, 1956, 3: 169～174
14 Gerig G, Martin J, Kikinis R, et al. Unsupervised tissue type segmentation of 3D dual-echo MR head data. Image Vision Comput, 1992, 10: 349～360
15 Liew A W C, Yan H. An Adaptive Spatial Fuzzy Clustering Algorithm for 3-D MR Image Segmentation. IEEE Transaction on Medical Imaging, 2003, 22(9)
16 Kennedy DN, Filipek PA, Caviness VS. Anatomic segmentation and volumetric calculations in nuclear magnetic resonance imaging. IEEE Transactions on Medical Imaging, 1989, 8: 1～7. Available at: http://www.cma.mgh.harvard.edu/ibsr/
17 Zijdenbos A, Dawant B. Brain segmentation and white matter lession detection in MR images. Crit. Rev. Biomed. Eng. 1994, 22(5-6): 401～465

(上接第 117 页)

据实际应用选择使用。

参 考 文 献

1 Chaudhuri S, Shim K. Including Group by in Query Optimization. In: Proc. Int'l Conf. Very Large Database Systems, 1994
2 Gupta A, Harinarayan V, Quass D. Generalized Projections: A Powerful Approach to Aggregation. In: Proc. Int'l Conf. Very Large Database Systems, 1995
3 Yan W, Larson P. Eager Aggregation and Lazy Aggregation. In: Proc. 21st Int'l Conf. Very Large Databases (VLDB), 1995. 345～357
4 O'Neil P, Graefe G. Multi-Table Joins through Bitmapped Join Indices. SIGMOD Record, 1995, 24(3): 8～11
5 Harinarayan V, Rajaraman A, Ullman J D. Implementing data cubes efficiently. In: Proc of the 1996 ACM SIGMOD Int'l Conf on Management of Data1 New York : ACM Press , 1996. 205～227
6 Theodoratos D, Sellis T. Designing Data Warehouses. In: Proc. Data Knowledge Engineering(DKE), 1999. 279～301
7 Gupta H, Mumick I S. Selection of Views to Materialize Under a Maintenance Cost Constraint. In: Proc. of the Intl. Conf. on Database Theory(ICDT), 1999. 453～470
8 Mistry H, Roy P, Sundarshan S, et al. Materialized View Selection and Maintenance Using Multi-Query Optimization Algorithm. In: Proc of SIGMOD'01, 2001. 307～318
9 Ryousri N. New Algorithms for Selecting Materialized Views inga DW: [MSc Thesis. Computer Science Dept, Alexandia University]. Egypt, November 2002
10 Yang J, Karlapalem K LiQ. Algorithms for Materialized View Design in Data Warehousing Enviroment. In: Proc of the 23rd International Conf. of Very Large Data Bases (VLDB'97), Athens, Greece, August 1997. 136～145
11 Harinarayan V, Rajaraman A, Ullman J D. Implementing data cubes efficiently. In: Proc of the 1996 ACM SIGMOD Int'l Conf on Management of Data. New York: ACM Press, 1996. 205～227