ISSN 1000-9825, CODEN RUXUEW
*Journal of Software*, Vol.19, No.2, February 2008, pp.301−313
DOI: 10.3724/SP.J.1001.2008.00301

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

# 实时主动数据仓库中多维数据实视图的选择*

林子雨[1], 杨冬青[1], 宋国杰[2+], 王腾蛟[1], 唐世渭[2]

[1](北京大学 信息科学技术学院,北京 100871)

[2](北京大学 视觉与听觉信息处理国家重点实验室,北京 100871)

# Materialized Views Selection of Multi-Dimensional Data in Real-Time Active Data Warehouses

LIN Zi-Yu[1], YANG Dong-Qing[1], SONG Guo-Jie[2+], WANG Teng-Jiao[1], TANG Shi-Wei[2]

[1](School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

[2](State Key Laboratory of Machine Perception, Peking University, Beijing 100871, China)

+ Corresponding author: Phn: +86-10-62755440, E-mail: gjsong@pku.edu.cn, http://db.pku.edu.cn

**Lin ZY, Yang DQ, Song GJ, Wang TJ, Tang SW. Materialized views selection of multi-dimensional data in real-time active data warehouses. *Journal of Software*, 2008,19(2):301−313.** http://www.jos.org.cn/1000-9825/19/301.htm

**Abstract**: In this paper, data mining based on the log of active decision engine is introduced to find the CUBE using pattern of analysis rules, which can be used as important reference information for materialized views selection. Based on it, a 3A probability model is designed, and the greedy algorithm, called PGreedy (probability greedy), is proposed, which takes into account the probability distribution of CUBE. Also view keeping rule is adopted to achieve better performance for dynamic view adjusting. Experimental results show that PGreedy algorithm can achieve better performance than BPUS (benefit per unit space) algorithm in real-time active data warehouses environment.

**Key words**: view selection; materialized view; data warehouse; active decision engine; analysis rule; OLAP (online analytical processing)

**摘 要**: 通过基于主动决策引擎日志的数据挖掘来找到分析规则的CUBE使用模式,从而为多维数据实视图选择算法提供重要依据;在此基础上设计了3A概率模型,并给出考虑CUBE受访概率分布的视图选择贪婪算法PGreedy (probability greedy),以及结合视图挽留原则的视图动态调整算法.实验结果表明,在实时主动数据仓库环境下,PGreedy算法比BPUS(benefit per unit space)算法具有更好的性能.

**关键词**: 视图选择;实视图;数据仓库;主动决策引擎;分析规则;联机分析处理

**中图法分类号**: TP311 **文献标识码**: A

实时主动数据仓库是数据仓库演化过程中的一个全新阶段,它融合了实时数据集成技术和主动规则机制,

* Supported by the National Natural Science Foundation of China under Grant No.60473051 (国家自然科学基金); the China HP Co. and Peking University Joint Project (北京大学-惠普(中国)合作项目)

Received 2007-02-01; Accepted 2007-06-29

既可以支持战略决策,也可以支持战术(实时)决策.主动决策引擎[1]是实时主动数据仓库的重要组成部分,其主要功能是监测事件的发生、执行分析规则[2]、作出主动决策,从而指导企业业务的开展.分析规则执行时需要使用多维数据进行分析,由于受到空间的限制,多维数据实视图中不可能包含所有级别的数据,只能有选择地实化一部分多维数据,这就产生了"实视图的选择问题"[3].

目前已有不少视图选择的相关研究,但都存在一些不足之处.有一些视图选择方法,比如文献[3−7],在分析和查询相对稳定的情况下表现出了很好的性能,但是通常来讲,OLAP(online analytical processing)查询变化不定,这使得这些方法在实际应用中的性能有待改进.而另一些视图选择方法,比如文献[8,9],作了一些关于动态视图选择有益的尝试,这些方法所基于的原理与内存管理原理相似,视图可以根据需要被选择,或者通过某种策略被预先实化.但是,OLAP 查询的难以预测性(系统不可能事先知道所有可能的查询)又使得这些方法在 OLAP 环境下无法取得较好的效果.

OLAP 查询的变化不定,在很大程度上影响了以上这些方法的性能.相反,在实时主动数据仓库中,分析规则所使用的 CUBE 集合却是相对稳定的,因为分析规则是预先设计的,使用哪些 CUBE 也是确定的.因此,本文把分析规则 CUBE 集合和 OLAP 查询集合分开处理,只研究针对前者的视图选择,从而可以充分利用前者的特性,避免这种特性由于 OLAP 查询的干扰而变得难以发现.

当视图选择对象是分析规则 CUBE 集合时,以前的视图选择方法可以获得更好的性能.但是,它们都是针对数据仓库提出的比较通用的方法,并没有与实时主动数据仓库环境紧密结合,未能发现和利用分析规则使用 CUBE 的模式,从而在实时主动数据仓库环境中无法取得更好的性能改善.

本文提出了基于主动决策引擎日志的数据挖掘,以发现分析规则使用多维数据的规律,从而指导多维数据实视图的选择.结合从日志挖掘得到的 CUBE 频率矩阵和系统实时统计信息,我们提出了 3A 概率模型,并引入多维数据格理论,设计了包含结点受访概率分布的 CUBE 的多维数据格.在此基础上,本文给出了考虑 CUBE 受访概率分布的视图选择贪婪算法 PGreedy(probability greedy),并对 PGreedy 算法做了性能分析.在视图的动态调整过程中,我们根据日志挖掘得到的近期受访 CUBE 设计了 CUBE 挽留原则,实现了更合理的视图动态调整.通过进行一系列的实验,结果表明,在实时主动数据仓库环境下,PGreedy 算法比已有的其他视图选择算法具有更好的性能.

本文第 1 节给出问题描述.第 2 节介绍如何从日志中挖掘 CUBE 使用模式.第 3 节阐述利用 CUBE 使用模式进行多维数据实视图的选择,详细论述 3A 概率模型和考虑 CUBE 受访概率分布的视图选择贪婪算法 Ggreedy.第 4 节给出实验设计与结果.相关工作在第 5 节进行介绍.最后在第 6 节给出结论.

## 1 问题描述

实时主动数据仓库是一个集成的信息存储仓库,既具备批量和周期性的数据加载能力,也具备数据变化探测、新数据的连续加载和更新能力,并能结合历史数据和新颖数据实现查询分析和自动规则触发,从而提供对战略决策和战术决策的双重支持.

主动决策引擎[1]是实时主动数据仓库的关键组成部分,它包括事件、规则、动作和元数据等几个组成部分.主动决策引擎实时监测事件的发生,并使用分析规则执行具体分析任务,如果满足某个条件,就触发特定的动作,则该过程就被称为主动决策.主动决策的结果会反馈到源系统当中.用户(分析人员)可以为主动决策引擎自行定义一系列分析规则,并负责处理分析规则执行时产生的问题和冲突.分析规则执行多维分析时,可以从 OLAP 服务器获得所需要的多维数据(CUBE).关于分析规则的细节可以参考文献[2].

本文所论述的实时主动数据仓库环境下的视图选择,只把分析规则使用的 CUBE 集合作为选择的对象,而不考虑 OLAP 查询.这种处理原则非常关键,因为分析规则使用的 CUBE 集合和 OLAP 查询具有很大的区别.前者是事先确定的,容易进行统计和预测,而且很多分析规则的触发有一定的规律;而后者则变化不定,统计和预测的难度较大.把分析规则 CUBE 集合分离出来,更容易发现关于 CUBE 的使用模式,从而为视图选择提供重要依据.

问题描述:在对分析规则 CUBE 集合和 OLAP 查询集合进行了划分之后,实时主动数据仓库环境下的视图选择需要解决的关键问题包括:

(1) 如何发现 CUBE 使用模式;

(2) 如何利用 CUBE 使用模式指导视图选择.

## 2 发现CUBE使用模式

CUBE 使用模式是指主动决策引擎的分析规则使用 CUBE 的规律,它为有效的视图选择提供了重要的依据.我们的研究采用基于日志的挖掘发现 CUBE 使用模式,为此,我们把分析规则执行的相关信息按照一定的格式记录并保存到日志中,从而为数据挖掘建立数据源.

CUBE 使用模式包括 CUBE 使用周期和 CUBE 概率矩阵,其中,CUBE 使用周期可以利用周期模式挖掘方法从主动决策引擎日志中获得.目前,针对周期模式挖掘已经有很多研究成果可以利用,这里不再赘述.下面我们来介绍 CUBE 概率矩阵.

**定义 1(CUBE 概率矩阵)**. 假设 CUBE 集合是 $C=\{c_0,c_1,\ldots,c_m\}$,时间段 $I=\bigcup_{j=0}^{n-1}[t_j,t_{j+1})$,其中,$[t_j,t_{j+1})$是一个单位统计区间.CUBE 概率矩阵是一个 $m\times n$ 的矩阵 $\boldsymbol{A}=(a_{ij})$,其中,$a_{ij}=r_{ij}\Big/\sum_{k=0}^{m}r_{kj}$,$r_{ij}$ 表示 $c_i$ 在$[t_j,t_{j+1})$内被访问的总次数.

CUBE 概率矩阵是一种重要的 CUBE 使用模式信息,它显示了在时间段 $I$ 内的不同单位统计区间内,各个 CUBE 被使用的概率情况(即概率分布情况).这里,时间段 $I$ 通常是一天(24 小时),每个单位统计区间$[t_j,t_{j+1}]$的长度相等,通常等于 1 小时.之所以建立 CUBE 概率矩阵,是因为在一天的不同时间区间内,CUBE 的使用概率分布是不同的,比如,在$[t_1,t_2]$内,CUBE $c_1$ 和 $c_2$ 的使用概率分别是 0.37 和 0.12;而在$[t_3,t_4]$内,则可能分别是 0.52 和 0.35.

主动决策引擎日志中通常包含大量的记录,我们可以利用数据挖掘和统计工具从日志中获取 CUBE 概率矩阵.需要指出的是,CUBE 概率矩阵的计算并不是实时进行的,而是按照一定的周期进行,比如一天一次.

## 3 利用CUBE使用模式进行多维数据实视图的选择

本节将介绍如何利用 CUBE 使用模式信息来进行多维数据实视图的选择和动态调整.这里,我们将用基于多维数据格的理论讨论实视图选择问题.

### 3.1 CUBE的多维数据格

**定义 2(多维数据格[3])**. 多维数据格$\langle M,\leq\rangle$是由多个数据结点 $T$ 构成的,其中,$T=(a_1,a_2,\ldots,a_n)$,$M$ 是多个结点 $T$ 的集合,每个 $a_i$ 代表第 $i$ 维上的一个级别,并且有:

(1) 对于两个结点 $T_1=(a_1,a_2,\ldots,a_n)$和 $T_2=(b_1,b_2,\ldots,b_n)$,$T_1\leq T_2$(即 $T_1$ 依赖于 $T_2$),当且仅当对于所有的 $i$ 都有 $a_i\leq b_i$,即 $T_2$ 在各维上的级别均低于或等于 $T_1$ 在相应维上的级别,并且利用 $T_2$ 可以计算得到 $T_1$.

(2) 格中基本元表示为 $DB=(c_1,c_2,\ldots,c_n)$,其中,$c_i$ 表示第 $i$ 维上最低的一个级别;通常假定 $DB$ 是已知的,从而可以根据 $DB$ 计算格中各个其他结点的数据.

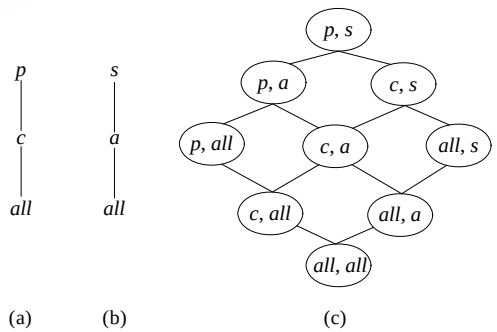



Fig.1 An example of lattice

图 1 一个多维数据格的例子

例 1:多维数据模式包含两个维,即 *Product* 和 *Location*,每个维又具有各自的层次结构.*Product* 维的层次结构是 *ProductId*→*Category*→*All*;*Location* 维的层次结构是 *StoreId*→*Area*→*All*.该多维数据格如图 1 所示,其中,*p*,*c*,*a*,*s* 分别表示 *ProductId*,*Category*,*StoreId*,*Area*,根据多维数据格的定义,基本元 $DB=(p,s)$.

**定义 3(分析规则的 CUBE)**. 假设有一个多维数据集合 *MD*,分析规则的 CUBE *c* 是 *MD* 的数据格中某一数

据结点的切片或切块,可以将 $c$ 表示成由 $d$ 个二元组组成的 $d$ 元组[10]:$\{(l_1,R_1),(l_2,R_2),\ldots,(l_d,R_d)\}$,其中,$d$ 为 $M$ 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 $R_i$ 是 $dom(l_i)$,即取值范围没有限制,则可以将$(l_i,R_i)$记为 $l_i$;当 $R_i$ 被表示为$\langle r_{i1},r_{i2}\rangle$时,表示在$\langle r_{i1},r_{i2}\rangle$范围上的一个切块,而当 $R_i$ 被表示为$\{r_{i1},r_{i2},\ldots,r_{in}\}$时,则表示在$\langle r_{i1},r_{in}\rangle$范围上的 $n$ 个切片.若 $l_i=all$,则维 $i$ 的二元组可以不出现在 CUBE 中.

**定义 4(CUBE 间依赖关系)**. 如果 CUBE $c$ 是多维数据集合 $MD$ 的数据格中的数据结点 $u$ 的切片或切块,就把数据结点 $u$ 称为 $c$ 的基,记为$\beta_c$,若 $c=\{(l_1,R_1),(l_2,R_2),\ldots,(l_d,R_d)\}$,则$\beta_c=\{l_1,l_2,\ldots,l_d\}$.CUBE 间的依赖关系定义为:对于 CUBE $p$ 和 $q$,$p\le q$($p$ 依赖于 $q$)当且仅当可以从 $q$ 计算得到 $p$.显然,若 $p\le q$,则不但要求$\beta_p\le\beta_q$,而且用来生成 $p$ 的数据均应包含在 $q$ 中[10].

分析规则的 CUBE 之间存在依赖关系,这种依赖关系对于视图选择至关重要.根据这种依赖关系,可以从已经被实化的 CUBE 中计算得到其他还未实化的 CUBE.在进行视图选择代价评估时,需要利用视图之间的依赖关系,计算由一个视图被实化以后给其他视图的生成代价带来的影响.

**定义 5(完整聚集视图)**. 在定义 3 中,当对于所有的 $i$,或者 $l_i=all$ 成立,或者 $R_i=\langle r_{i1},r_{i2}\rangle=dom(l_i)$成立时,则 CUBE 被称为完整聚集视图.

**定义 6(部分聚集视图)**. 在定义 3 中,当至少存在一个 $i$ 使得 $l_i\ne all$ 且 $R_i=\langle r_{i1},r_{i2}\rangle\ne dom(l_i)$时,则 CUBE 被称为部分聚集视图.

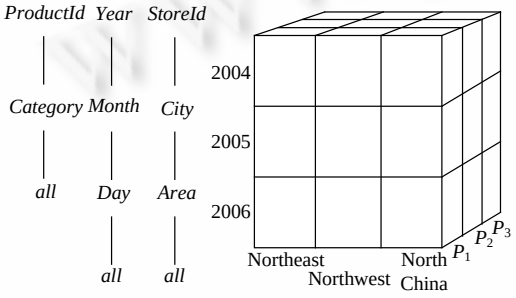



Fig.2 An example of multidimensional data set

图 2 一个多维数据集例子

例 2:我们以一个销售应用的多维数据为例,对 CUBE 的表示方法以及完整聚集视图和部分聚集视图的概念进行说明.该多维数据模式包括 3 个维:*Time*,*Product* 和 *Store*,各维的层次结构如图 2 所示.我们定义如下 CUBE:

$c_1=\{(ProductId),(Area,\{Northwest\}),(Year,\{2004,2006\})\}$,表示“各个产品在西北区域内分别在 2004 年和 2006 年的销售量”.

$c_2=\{(ProductId),(Area),(Year)\}$,表示“各个产品在各个区域内分别在 2004 年、2005 年和 2006 年的销售量”.

很显然,根据定义 5 和定义 6,$c_1$ 属于部分聚集视图,$c_2$ 属于完整聚集视图.容易看出,例 1 所示的多维数据格中的数据结点属于完整聚集视图.

基于上面的定义,根据 CUBE 之间的依赖关系,我们就可以为 CUBE 建立数据格.CUBE 可能属于完整聚集视图,也可能属于部分聚集视图,为了便于讨论问题,这里假设 CUBE 都属于完整聚集视图.

### 3.2 CUBE受访概率分布

在实际应用中,多维数据格中的各个结点被访问的概率是不同的.因此,为了更加合理地对视图选择的代价进行评估,需要综合考虑各个结点的受访概率分布.由于我们已经假设参与视图选择的候选视图都是分析规则使用的 CUBE,利用主动决策引擎日志,我们就可以很容易得到不同 CUBE 的受访概率.在已有的视图选择方法中,有些方法假设视图受访概率分布情况是已知的,或者假设是均匀分布的,或者假设用户能够给出视图的受访概率分布,比如文献[3–5];另外一些方法假设由系统收集视图受访概率分布情况,比如文献[10,11].

对视图受访概率的统计基本上包括两大类:对最近较短时间内的概率统计和对较长时间跨度内的概率统计.第 1 类方法通常只包含最近一段时间内的概率分布信息,一些距离当前时间点比较久远的访问则不予考虑,这类统计方法的理论基础是“最近的概率分布也是未来一段时间内最可能的概率分布”.但是该方法有一个缺点,就是由于统计时段过短,一些“噪音”(比如突发的一系列查询)会严重“扭曲”当前的概率分布信息,使得它不能很好地代表未来一段时间内的查询分布;第 2 类方法则会把较长时间跨度内的访问情况都考虑在内,与第 1 种方法相比,可以更好地处理“噪音”.但是该方法也存在缺点,因为它统计的是长时间内的概率分布,所以它与当

前真实的概率分布可能存在偏差,有时这种偏差甚至很大.

本文中,我们提出采用"3A 概率模型"来更好地反映在某一个视图更新周期内(视图更新发生在周期的结束点),主动决策引擎所用 CUBE 的受访概率分布情况.在该模型中,系统需要维护 3 类信息,包括 CUBE 概率矩阵 $\boldsymbol{A}_1$、当前单位统计区间内 CUBE 受访实时概率分布 $\boldsymbol{A}_2$ 和当前视图更新周期内 CUBE 受访实时概率分布 $\boldsymbol{A}_3$.

**定义 7(3A 概率模型)**. 假设$\langle M,\leq\rangle$是 CUBE 的多维数据格,CUBE 概率矩阵 $\boldsymbol{A}_1$,当前单位统计区间$[t_j,t_{j+1}]$内 CUBE 受访概率分布 $\boldsymbol{A}_2$ 和当前视图更新周期$[u_k,u_{k+1}]$内 CUBE 受访概率分布 $\boldsymbol{A}_3$,则下一个视图更新周期$[u_{k+1},u_{k+2}]$内 CUBE 受访预测概率分布为 $\boldsymbol{A}=\alpha\times\boldsymbol{A}_3+\beta\times\pi_d(\boldsymbol{A}_1)$,其中,$[u_{k+1},u_{k+2}]\in[t_d,t_{d+1}]$,$\pi_d(\boldsymbol{A}_1)$表示矩阵 $\boldsymbol{A}_1$ 的第 $d$ 列元素,$\alpha$和$\beta$满足如下关系:(1) $\alpha+\beta=1$;(2) $\alpha=\gamma\times\beta$,其中,$\gamma=\boldsymbol{D}(\pi_j(\boldsymbol{A}_1),\boldsymbol{A}_2)$,表示矩阵 $\boldsymbol{A}_1$ 的第 $j$ 列元素与 $\boldsymbol{A}_2$ 之间的距离.

$\boldsymbol{D}(\pi_j(\boldsymbol{A}_1),\boldsymbol{A}_2)$可以采用欧氏距离(Euclidean distance)进行计算.对于两个数值序列,$\boldsymbol{X}=\{x_0,x_1,\ldots,x_{n-1}\}$和 $\boldsymbol{Y}=\{y_0,y_1,\ldots,y_{n-1}\}$,两者之间的欧氏距离 $\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$计算方法如下:

$$\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})=\sqrt{\sum_{i=0}^{n-1}(x_i-y_i)^2}\ .$$

从定义 7 我们可以得到$\alpha=\gamma(1+\gamma)$和$\beta=1/(1+\gamma)$,由于 $\boldsymbol{A}_1$ 与 $\boldsymbol{A}_2$ 都用于描述概率分布,所以,二者包含的元素的值都位于区间[0,1]内,因此,$\gamma=\boldsymbol{D}(\pi_j(\boldsymbol{A}_1),\boldsymbol{A}_2)\in[0,1]$,从而可以得到,$\alpha=\gamma(1+\gamma)\in[0,0.5]$和$\beta=1/(1+\gamma)\in[0.5,1]$.也就是说,虽然$\pi_d(\boldsymbol{A}_1)$和 $\boldsymbol{A}_3$ 在 $A$ 中所占比重存在一个浮动区间,但是总体来说,$\pi_d(\boldsymbol{A}_1)$的比重要大于 $\boldsymbol{A}_3$ 的比重.

3A 概率模型综合考虑了视图更新周期内的概率分布($\boldsymbol{A}_3$)和单位统计区间内的概率分布($\boldsymbol{A}_1$ 和 $\boldsymbol{A}_2$),体现了长期概率分布与短期概率分布的结合;同时,还考虑了当前单位统计区间内的实时概率分布($\boldsymbol{A}_2$)和当前单位统计区间内的历史概率分布($\boldsymbol{A}_1$),体现了历史概率分布与当前概率分布的结合.因此,3A 概率模型比已有的概率统计方法更加合理、有效,我们将在实验部分对此进行验证.

例 3:假设$\langle M,\leq\rangle$是 CUBE 的多维数据格,格中各个结点的 CUBE 概率矩阵如图 3(c)所示.图 3(c)中只显示了本例需要使用的矩阵元素.另外,我们假设基本元 $DB$ 总是被实化的(因为它不能从任何其他结点计算得到),在进行受访概率统计时不考虑基本元,所以,在图 3(c)中没有关于结点 $a$ 的概率信息.又假设当前单位统计区间是$[t_6,t_7]$,当前视图更新周期是$[u_0,u_1]$,由系统统计得到的$[u_0,u_1]$内各个结点的实时概率分布是[0.07,0.32,0.17,0.25,0.19,0,0,0],$[t_6,t_7]$内各个结点的实时概率分布是[0.17,0.16,0.17,0.11,0.13,0.08,0.16,0.02],下一个视图更新周期是$[u_1,u_2]$,并且有$[u_1,u_2]\in[t_7,t_8]$,则$[u_1,u_2]$的预测概率分布计算方法如下:

$\pi_6(\boldsymbol{A}_1)$=[0.20,0.14,0.12,0.16,0.09,0.04,0.20,0.05],

$\boldsymbol{A}_2$=[0.17,0.16,0.17,0.11,0.13,0.08,0.16,0.02],

$\gamma=\boldsymbol{D}(\pi_6(\boldsymbol{A}_1),\boldsymbol{A}_2)=\sqrt{(0.20-0.17)^2+(0.14-0.16)^2+\ldots+(0.20-0.16)^2+(0.05-0.02)^2}$ =0.1149,

$\alpha=\gamma(1+\gamma)$=0.103,

$\beta=1/(1+\gamma)$=0.897,

$\pi_7(\boldsymbol{A}_1)$=[0.12,0.23,0.07,0.27,0.04,0.15,0.10,0.02],

$\boldsymbol{A}_3$=[0.07,0.32,0.17,0.25,0.19,0,0,0],

$\boldsymbol{A}=\alpha\times\boldsymbol{A}_3+\beta\times\pi_d(\boldsymbol{A}_1)$=[0.11,0.24,0.08,0.27,0.06,0.13,0.09,0.02].

### 3.3 多维数据实视图的选择

实视图的选择问题就是在满足约束条件的前提下,根据代价评估模型,以代价(查询代价或视图维护代价)最小为原则,从给定的视图集合中选择一部分视图进行实化[12].代价模型是实视图的选择问题中的一个重要部分,在计算代价时,有些方法只考虑查询代价,而有些方法则综合查询代价和视图维护代价.本文假设在计算代价时,只考虑查询代价而不考虑视图维护代价.

在计算查询代价时,目前较多的研究工作采用线性代价模型(比如文献[3,10]).假设$\langle M,\leq\rangle$是 CUBE 的多维数据格,$c$ 和 $d$ 是$\langle M,\leq\rangle$中的两个结点,$c\leq d$,其中,$c$ 是一个未实化的 CUBE,$d$ 是一个已经实化的 CUBE,如果采用线性代价模型,那么从 $d$ 计算得到 $c$ 的代价是 $d$ 中包含的记录总数.

有了线性代价模型,我们就可以进行视图选择的收益评估,从而决定选择哪些视图进行实化.这里,我们假设多维数据格中的基本元 *DB* 总是被实化的,因为它不能从任何其他结点计算得到.

**定义 8(收益评估模型)**. 假设$\langle M,\leq\rangle$是 CUBE 的多维数据格,并且格中结点都有受访概率信息,$V$ 表示$\langle M,\leq\rangle$中已经实化的结点的集合,CUBE $c$ 是$\langle M,\leq\rangle$中未被实化的结点,则 $c$ 相对于 $V$ 的净收益 $B(c,V)=\omega_c\sum_{d\leq c}B_d$,其中,$\omega_c$ 表示结点 $c$ 的受访概率,对于每一个满足 $d\leq c$ 条件的$\langle M,\leq\rangle$中的结点 $d$,定义 $B_d$ 如下:

(1) 假设 $u$ 是 $V$ 中满足 $d\leq u$ 条件的、代价最小的 CUBE.

(2) 如果 $Cost(c)<Cost(u)$,则 $B_d=Cost(u)-Cost(c)$;否则,$B_d=0$.其中,$Cost(u)$表示 $u$ 的查询代价.

在收益评估模型的基础上,我们可以决定选择哪些视图进行实化.一般来说,视图选择问题属于 NP 问题,需要借助于启发算法来产生近似最优解.目前有些研究采用贪婪算法,比如文献[3,4,7].本文给出的贪婪算法 PGreedy(见算法 1)以文献[3]中的贪婪算法 Greedy 为基础,区别在于,后者在收益评估时没有考虑多维数据格中结点的受访概率分布;而本文的贪婪算法则考虑了这个因素,并且采用了 3A 概率模型.因此,二者的收益评估模型不一样.

**算法 1**. 视图选择贪婪算法 *PGreedy*(*M*,*k*).

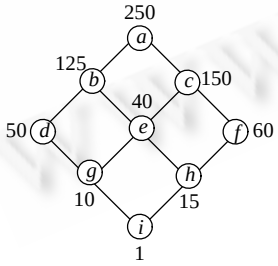
Input: 1. 包含结点受访概率分布和空间代价的多维数据格$\langle M,\leq\rangle$.

2. 视图个数 $k$.

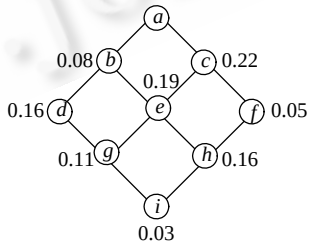
Output: 1. 所选择的实视图集合 $V$.

```
begin
    V←{DB};   //DB 是多维数据格⟨M,≤⟩中的基本元,总是被实化
    ←多维数据格⟨M,≤⟩中除了基本元 DB 以外的所有结点;
    for (i=0; i++; i<k) do
        c←选择不在集合 V 中并且使 B(c,V)最大的 C 中的结点;
        V←V∪{c}; C←C−{c};
    return V;
end
```

例 4:假设$\langle M,\leq\rangle$是 CUBE 的多维数据格,格中结点的空间代价和受访概率分布如图 3(a)和图 3(b)所示.现在使用 PGreedy 算法从$\langle M,\leq\rangle$中选择实化视图.表 1 给出了 PGreedy 算法在各轮选择中选择不同结点时产生的毛收益(没有考虑结点的受访概率时的收益)和净收益(考虑结点的受访概率时的收益),并给出了每轮选择的结果.在本例中,我们只演示前 3 轮的选择过程.




(a) Space cost (a) 空间代价




(b) Access probability distribution (b) 受访概率分布

| | ... | $[t_6,t_7)$ | $[t_7,t_8)$ | ... |
|---|---|---|---|---|
| *b* | ··· | 0.20 | 0.12 | ··· |
| *c* | ··· | 0.14 | 0.23 | ··· |
| *d* | ··· | 0.12 | 0.07 | ··· |
| *e* | ··· | 0.16 | 0.27 | ··· |
| *f* | ··· | 0.09 | 0.04 | ··· |
| *g* | ··· | 0.04 | 0.15 | ··· |
| *h* | ··· | 0.20 | 0.10 | ··· |
| *i* | ··· | 0.05 | 0.02 | ··· |

(c) CUBE probability matrix (c) CUBE 概率矩阵

Fig.3 Space cost and access probability distribution of lattice nodes

图 3 多维数据格结点的空间代价和受访概率分布

由于基本元 $a$ 总是被实化的,所以最初的实化视图集合只包含 $a$,其他结点都需要从 $a$ 生成,因此代价都是 250.在第 1 轮选择中,如果先选择 $b$,则 $b$ 将不再需要从 $a$ 生成,而只要自己生成,所需代价是 125,代价减少了 125.

同理,其他位于 $b$ 下面的与 $b$ 有依赖关系的结点 $d,e,g,h$ 和 $i$ 也可以直接从 $b$ 生成,而不必从 $a$ 生成,所以每个代价分别减少了 125.亦即,选择 $b$ 带来的毛收益 $B^*(b,V)=6\times125=750$,由此带来的净收益 $B(b,V)=750\times0.08=60$.再比如,如果先选择 $c$,$c$ 自己的生成代价减少了 100,其他位于 $c$ 下面的结点 $e,f,g,h$ 和 $i$ 也可以直接从 $c$ 生成,而不必从 $a$ 生成,每个代价分别减少了 100,所以,选择 $c$ 带来的毛收益 $B^*(c,V)=6\times100=600$,由此带来的净收益 $B(c,V)=600\times0.22=132$.同理可以计算得到选择其他结点时的收益.最终,结点 $e$ 在第 1 轮选择中胜出,因为由它带来的净收益最大.按照同样的方式,在第 2 轮选择中 $c$ 胜出,第 3 轮选择中 $d$ 胜出.

**Table 1** View selection benefits of PGreedy algorithm

**表 1** PGreedy 算法视图选择收益

| | Choice 1 | | | Choice 2 | | | Choice 3 | | |
|---|---|---|---|---|---|---|---|---|---|
| | Gross benefit | Access probability | Net benefit | Gross benefit | Access probability | Net benefit | Gross benefit | Access probability | Net benefit |
| *b* | 750 | 0.08 | 60 | 250 | 0.08 | 20 | 250 | 0.08 | 20 |
| *c* | 600 | 0.22 | 132 | 200 | 0.22 | 44 | | | |
| *d* | 600 | 0.16 | 96 | 200 | 0.16 | 32 | 200 | 0.16 | 32 |
| *e* | 840 | 0.19 | 159.6 | | | | | | |
| *f* | 570 | 0.05 | 28.5 | 190 | 0.05 | 9.5 | 90 | 0.05 | 4.5 |
| *g* | 480 | 0.11 | 52.8 | 60 | 0.11 | 6.6 | 60 | 0.11 | 6.6 |
| *h* | 470 | 0.16 | 75.2 | 50 | 0.16 | 8 | 50 | 0.16 | 8 |
| *i* | 249 | 0.03 | 7.47 | 39 | 0.03 | 1.17 | 39 | 0.03 | 1.17 |
| Choice result | *e* | | | *c* | | | *d* | | |

### 3.4 贪婪算法性能分析

文献[3]证明了在没有考虑结点受访概率的情况下,贪婪算法 Greedy 与最优解的收益比大于等于 $1-(1-1/k)^k$,并认为可以把 Greedy 算法扩展到考虑视图受访概率分布的情况,但是并没有给出详细论证.而且,即使考虑了视图受访概率分布的情况,文献[3]所采用的收益评估模型与本文的也仍有区别.前者在考虑概率分布时的收益计算方法是 $B(c,V)=\sum_{d\le c}\omega_d B_d$,而本文的计算方法是 $B(c,V)=\omega_c\sum_{d\le c}B_d$.因此,这里将证明,在本文采用的收益评估模型下,贪婪算法 PGreedy 与最优解的收益比也大于等于 $1-(1-1/k)^k$.

**定理 1**. 假设$\langle M,\le\rangle$是多维数据格,$S=\{s_1,s_2,\ldots,s_k\}$是贪婪算法 PGreedy 选择得到的实化视图集合,$p_i$ 是选择视图 $s_i$ 产生的收益,$V=\{v_1,v_2,\ldots,v_k\}$是实化视图选择的最优解,$q_i$ 是选择视图 $v_i$ 产生的收益,$\omega_i$ 表示视图的受访概率,则贪婪算法 PGreedy 获得的总收益为

$$P=\sum_{i=1}^{k}\omega_i p_i \tag{a}$$

最优解获得的总收益为

$$Q=\sum_{i=1}^{k}\omega_i q_i \tag{b}$$

则有 $$P/Q\ge1-(1-1/k)^k.$$

证明:已知 $S=\{s_1,s_2,\ldots,s_k\}$是贪婪算法选择得到的实化视图集合,假设 $x_{ij}$ 是由选择视图 $s_j$ 而为视图 $v_i$ 带来的收益,根据文献[3]的结论,有如下不等式成立:

$$\sum_{i=1}^{k}\omega_i x_{ij}\le\omega_j p_j \tag{c}$$

下面我们采用文献[3]的证明思路来证明几个重要的不等式.因为在贪婪算法中,除非 $v_i=s_i$,否则,算法的选择结果只能是 $s_i$ 而不是 $v_i$.因此在第 1 轮选择中,对于所有的 $i$ 都有 $\omega_i q_i\le\omega_1 p_1$,因为对于任何一个 $i$,如果这个关系不成立,就意味着 $v_i$ 会被贪婪算法选中,而不是 $s_i$.在 $s_1$ 被选中的情况下,在第 2 轮选择中,由选择 $v_i$ 带来的收益是 $\omega_i q_i-\omega_i x_{i1}$,其中,$\omega_i x_{i1}$ 是 $v_i$ 和 $s_1$ 重叠部分的收益,需要被扣除,则对所有的 $i$ 都有 $\omega_i q_i-\omega_i x_{i1}\le\omega_2 p_2$.因为对于任何一个 $i$,如果这个关系不成立,就意味着 $v_i$ 会被贪婪算法选中,而不是 $s_2$.同理可以得到,对所有的 $i$ 都有 $\omega_i q_i-\omega_i x_{i1}-\omega_i x_{i2}-\omega_i x_{i3}-\ldots-\omega_i x_{i,j-1}\le\omega_j p_j$.根据已知条件(a),(b)和以上讨论的多个不等式,可以得到以下不等式:

$$Q\le k\omega_1 p_1 \tag{1}$$

$$Q\le k\omega_2 p_2+\omega_1 p_1 \tag{2}$$

$$Q\le k\omega_3 p_3+\omega_1 p_1+\omega_2 p_2 \tag{3}$$

…

$$Q\le k\omega_k p_k+\omega_1 p_1+\omega_2 p_2+\ldots+\omega_k p_k \tag{k}$$

对于确定的 $P$,当 $Q$ 最大程度收敛时,需要满足的条件是不等式(1),(2),…,($k$)右边全部相等,又因为不等式($i$)和不等式($i$+1)的右边的差值是 $k\omega_{i+1}p_{i+1}-(k-1)\omega_i p_i$,如果不等式($i$)和不等式($i$+1)的右边相等,则必须满足 $k\omega_{i+1}p_{i+1}-(k-1)\omega_i p_i=0$,我们令 $u_i=\omega_i p_i$,由此可以得到

$$u_i=\frac{k}{(k-1)}u_{i+1} \tag{d}$$

由公式(a)和公式(d)可以得到

$$P=\sum_{i=0}^{k-1}\left(\frac{k}{(k-1)}\right)^i u_k \tag{e}$$

根据公式(d)和不等式(1),我们又可以得到

$$Q\le k\left(\frac{k}{(k-1)}\right)^{k-1}u_k \tag{f}$$

因此,根据公式(e)和公式(f)可以得到:

$$P/Q\ge\sum_{i=0}^{k-1}\left(\frac{k}{k-1}\right)^{i-k+1}\ge\frac{1}{k}\left(1+\frac{k-1}{k}+\left(\frac{k-1}{k}\right)^2+\ldots+\left(\frac{k-1}{k}\right)^{k-1}\right)\ge 1-(1-1/k)^k \qquad \square$$

### 3.5 多维数据实视图的动态调整

随着主动决策引擎的运行,各个 CUBE 的受访概率分布是动态变化的.因此需要对实视图集合进行动态调整,以满足分析规则使用 CUBE 的需求.算法 2 显示了选择前 $k$ 个视图时的视图动态调整的过程,带空间限制条件的视图动态调整与此类似,因此这里不再给出算法.算法 2 中使用了 CUBE 挽留原则,其定义如下:

**定义 9(CUBE 挽留原则)**. CUBE 挽留原则是指,对于两个不同的 CUBE $A$ 和 $B$,$A$ 在实化视图集合中,$B$ 在候选视图集合中.在进行实视图的动态调整时,虽然 $B$ 的收益大于 $A$ 的收益,但是 $A$ 是近期受访的 CUBE(下一个视图更新周期内使用的 CUBE),则 $A$ 仍然保留在实视图集合中,不会被 $B$ 取代.

利用基于主动决策引擎日志的周期模式挖掘,可以获得近期受访的 CUBE.对周期模式的挖掘就是在时序数据库中找出重复出现的模式.由于动作日志数据库包含了 *Time* 字段,所以可以作为一个时序数据库来进行周期模式挖掘.比如,每天上午 9:00,分析规则都会使用某一个 CUBE 进行多维分析.

采用 CUBE 挽留原则,在很大程度上减少了视图动态调整过程中可能出现的“震荡”问题——已经实化的 CUBE 被删除后,在短暂时间内(比如 5 分钟以后),有分析规则再次请求同样的 CUBE,OLAP 服务器就必须为它重新生成一次,从而导致不必要的系统资源浪费.

**算法 2**. 视图动态调整.

Input: 1. $\boldsymbol{A}_1$, $\boldsymbol{A}_2$, $\boldsymbol{A}_3$, $k$;

2. 当前视图更新周期内的实视图集合 $V_0$;

3. 近期受访 CUBE 集合 $C$.

Output: 1. 下一个视图更新周期内的实视图集合 $V$.

**begin**

//根据 3A 概率模型计算下一个视图更新周期的 CUBE 受访概率分布

$\boldsymbol{A}\leftarrow CalculateProbability(\boldsymbol{A}_1,\boldsymbol{A}_2,\boldsymbol{A}_3)$;

$UpdateProbablitiy(M,\boldsymbol{A})$; //更新 $M$ 中各结点的受访概率

$R\leftarrow C\cap V_0$; //获得满足视图挽留原则的视图集合

$V\leftarrow R\cup\{DB\}$; //基本元和 $R$ 中的视图仍然作为实化视图被保留

```
    C←多维数据格⟨M,≤⟩中除了 R 和基本元以外的所有结点;
    for (i=0; i++; i<k–|R|) do
        c←选择不在集合 V 中并且使 B(c,V)最大的 C 中的结点;
        V←V∪{c};
        C←C–{c};
    //选出 V 以后,需要对 V 中视图进行实化,同时物理删除 V0 中被淘汰的实视图
    for each v0∈V0 do
      if v0∉V then   //说明 v0 被淘汰
        DeleteView(v0);   //把 v0 物理删除
    for each v∈V do
      if v∉V0 then   //说明是新选中的视图,还未被实化
        Materialize(v);   //对视图 v 进行实化
    return V;
end
```

## 4 实验设计与结果

本节介绍实验设计和实验结果,目的在于说明,结合日志挖掘得到的 CUBE 使用模式来进行多维数据实视图的选择能够取得比以前的方法更好的性能,并且能够使 OLAP 服务器节省资源消耗,避免一些不合理的实视图生成和删除.

### 4.1 实验设计

实验基于我们设计的 DM-LADE 系统,DM-LADE 系统是我们设计的实时主动数据仓库原型系统 MPP-RTADW 的一个子系统,用来完成基于主动决策引擎日志的数据挖掘功能.在 MPP-RTADW 原型系统的初步设计中,采用 ORCALE 10g 作为数据库源和数据仓库的数据库管理系统.

在数据源部分,使用 TPC-H(www.tpc.org)测试基准完成数据建模和数据填充工作,并使用 TPC-H 测试基准提供的并发数据修改来模拟数据源中发生的变化.在 MPP-RTADW 子系统中,由主动决策引擎的事件代理实时探测事件的发生,并在特定事件发生以后,使用规则进行分析.如果条件满足,就作出主动决策,同时会把分析规则执行的相关信息记录到动作日志数据库中.数据挖掘模块会根据预先设定的周期或人工命令执行数据挖掘任务,并把数据挖掘结果(比如关于各个 CUBE 的使用周期和 CUBE 概率矩阵)写入到 CUBE 使用模式表 $T$ 中,并把 $T$ 作为 OLAP 服务器的多维数据实视图的选择算法的输入参数.

实验的硬件环境是 3 台 HP Proliant DL585 服务器,分别作为数据源、数据仓库和 OLAP 服务器,每台服务器的配置为 4 个 AMD 皓龙 2.4G CPU,32GB 内存和 1.20TB 硬盘.实验的软件环境是 Windows Server 2003 操作系统和 ORACLE 10g 数据库管理系统.

### 4.2 实验1:分析规则的规律性对算法性能的影响

本实验的目的在于比较当存在实视图空间限制时,分析规则运行的规律性对 FPUS(frequency per unit space)算法[10]、BPUS(benefit per unit space)算法[3]和 PGreedy 算法的性能影响.当存在空间限制时,我们让 PGreedy 采用单位空间的净收益为选择标准.我们共设计了 100 条分析规则,使用到了 50 个不同类型的 CUBE,总数据量为 120M,实视图空间取为 15M.分析规则在多维分析过程中会使用 1 个或多个 CUBE.分析规则会被多次触发,其中涉及的 CUBE 会被多次调用,对于某个 CUBE,我们计算 OLAP 服务器多次响应该 CUBE 的时间的平均值 $t$.对于两种算法 FPUS(或 BPUS)和 PGreedy,我们可以分别得到某个 CUBE 的响应时间 $t_1$ 和 $t_2$,$t_1/t_2$ 的值就是两种算法对某个 CUBE 的响应时间比 $r$,计算所有的 CUBE 的 $r$ 的平均值就可以得到平均响应时间比.在实验中,我们把有规律的分析规则(按一定周期执行)占所有分析规则的百分比从 0 增加到 100%,从而观察分析规

则执行的规律性对算法性能的影响,因为分析规则的周期性运行会导致 CUBE 的周期性被使用.

图 4 显示了实验的结果,从中可以看出,当分析规则按一定规律运行时,算法 PGreedy 比算法 BPUS 和 FPUS 取得了更好的性能,而且按照一定的规律执行的分析规则越多,PGreedy 算法取得的性能优势就越明显.这说明 PGreedy 可以充分利用 CUBE 使用模式进行更加合理的视图选择.

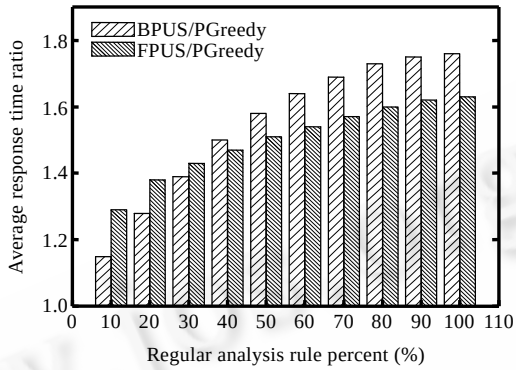



Fig.4 Influence of analysis rule regularity on algorithm performance

图 4 分析规则执行的规律性对算法性能的影响

### 4.3 实验2:CPU使用率

本实验的目的在于比较当存在实视图空间限制时,采用 BPUS 算法和 PGreedy 算法时的 CPU 资源消耗程度.为了测试的合理性,我们选择触发的分析规则中有规律运行的分析规则与无规律运行的分析规则各占 50%,分析规则的规律性事先已经写入到 CUBE 使用模式表中,这里可以直接提供给 PGreedy 算法使用.在 15 分钟内,我们连续触发这些分析规则,同时观察 CPU 资源的使用情况.

图 5 显示了在此期间的 CPU 使用率变化情况,从中可以看出,在大部分时间里,PGreedy 算法的 CPU 使用率都比 BPUS 算法要低,这是因为 PGreedy 算法是根据 CUBE 使用模式作出实视图选择决定的,避免了一些不必要和不合理的实视图生成和删除.

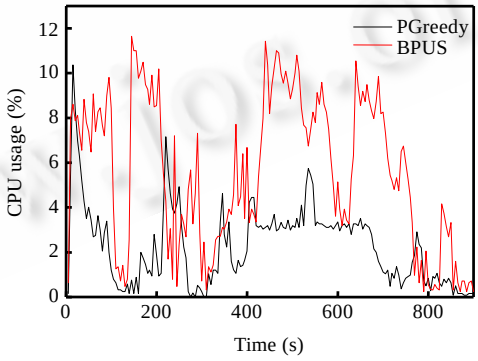



Fig.5 CPU usage of PGreedy and BPUS

图 5 PGreedy 和 BPUS 的 CPU 使用率

### 4.4 实验3:3A概率模型性能

本实验的目的在于观察 3A 概率模型预测 CUBE 受访概率分布的准确程度.我们设计了两种场景:在第 1 种场景中,我们使用 3A 概率模型计算得到的概率分布 $P_{3A}$ 作为下一个视图更新周期的预测概率分布;第 2 种场景中,直接用系统统计得到的当前视图更新周期内实际概率分布 $P_{normal}$ 作为下一个视图更新周期内预测概率

分布.我们用 $P_{real}$ 表示由系统统计得到的下一个视图更新周期的实际概率分布,用 $\boldsymbol{D}_{3A}$ 表示 $P_{3A}$ 和 $P_{real}$ 之间的欧氏距离,即二者的偏离程度,用 $\boldsymbol{D}_{normal}$ 表示 $P_{normal}$ 和 $P_{real}$ 之间的欧氏距离.图 6 显示了在连续 20 个视图更新周期内 $\boldsymbol{D}_{3A}$ 和 $\boldsymbol{D}_{normal}$ 的变化情况,从中可以看出,$\boldsymbol{D}_{normal}$ 的值明显比 $\boldsymbol{D}_{3A}$ 要大,而且不稳定,这说明 3A 概率模型得到的预测概率分布比较接近实际概率分布,而且性能比较稳定.

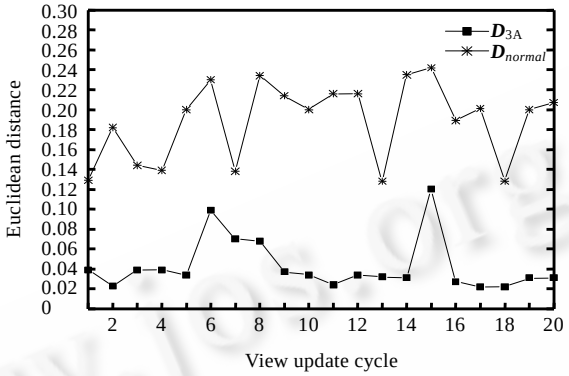



Fig.6 Performance of 3A probability model

图 6 3A 概率模型的性能

### 4.5 实验4:3A概率模型对PGreedy算法性能的改善

本实验的目的在于观察3A概率模型对贪婪算法性能的改善.首先,除了分析规则不再按照人为设置的规律运行以外,本实验的其他设计与实验 1 相同.在此基础上,我们设计了 3 种场景:第 1 种场景没有考虑 CUBE 受访概率分布的 Greedy 算法[3];第 2 种场景考虑 CUBE 受访概率分布的 Greedy*算法(为了方便区分,我们临时命名的算法),但是直接用系统统计得到的当前视图更新周期内实际概率分布 $P_{normal}$ 作为下一个视图更新周期内预测概率分布;第 3 种场景采用 3A 概率模型 PGreedy 算法.假设第 1 种场景与第 2 种场景的 CUBE 响应时间比为 $r_1$,第 1 种场景与第 3 种场景的 CUBE 响应时间比为 $r_2$.

图 7 显示了在不同的单位统计区间内 CUBE 响应时间比 $r_1$ 和 $r_2$ 的变化情况.可以看出,在连续的 10 个单位统计区间内,由于考虑了概率分布,PGreedy 算法和 Greedy*算法比没有考虑概率分布的 Greedy 算法取得了更好的性能,而且在性能改进程度上,采用 3A 概率模型的 PGreedy 算法比 Greedy*算法幅度更大.

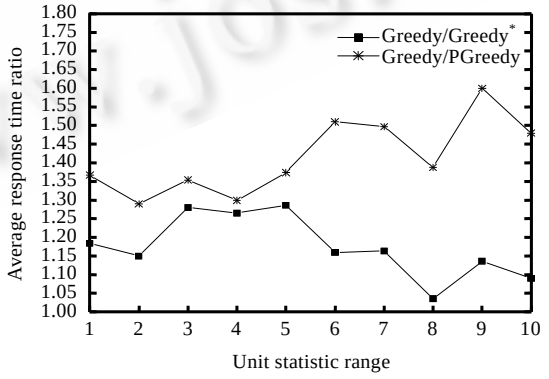



Fig.7 Performance improvement by 3A probability model over greedy algorithm

图 7 3A 概率模型对贪婪算法性能的改善

## 5 相关工作

视图选择一直是数据仓库领域的研究热点,也产生了丰硕的研究成果.Harinarayan 等人在文献[3]中最早提出了基于数据立方体格(lattice of data cube)模型的实视图选择方法,并给出了一种时间复杂度为 $O(kn^2)$的贪婪算法 BPUS.但是,文献[3]在进行视图选择时并没有考虑索引的问题.因此,文献[4]引入了带有 B-树索引的实视图选择问题.文献[5]提出了一种基于特殊多维数据格的、简单而快速的 PBS(pick by size)算法来选择实视图.PBS 的运行速度要比 BPUS 快几个数量级,可以很快生成“实视图空间-查询响应时间曲线”.PBS 继承了 BPUS 的线性代价模型,与 BPUS 不同的是,PBS 采用了单位空间收益为视图选择标准.文献[10]提出以单位空间的频率为标准、时间复杂度为 $O(n\log_2 n)$的 FPUS 算法.文献[11]提出了最近实化父视图的概念,并用 $B^+$树创建对聚集视图的索引,还给出一个以相对效益为标准的 PVMA(progressive view materialization algorithm)算法.

另外,文献[6]等还提出了在不考虑资源限制和性能保证的情况下筛选候选视图的方法;文献[7]提出了一个视图选择常见问题的理论框架;文献[13]提出采用遗传算法来解决视图选择问题;文献[14]引入了人工智能领域的模拟退火算法,取得了较好的性能改进;文献[15]中提出的多用户多窗口方法充分考虑了每个用户的类型和在发起一系列查询时所具有的特殊规律,可以充分利用查询规律改善视图选择性能.

## 6 结束语

多维数据实视图的选择问题在传统数据仓库领域已有较多可行的解决方案,但在实时主动数据仓库环境下还缺少与之紧密结合的视图选择方法.

本文提出采用基于主动决策引擎日志的数据挖掘来获得 CUBE 频率矩阵和近期受访 CUBE 等参考信息,为改进视图选择算法的性能奠定了坚实的基础.在此基础上,我们提出了考虑 CUBE 结点的受访概率分布的贪婪算法 PGreedy.与其他同类方法不同,我们采用 3A 概率模型来获得概率分布信息,使获得的概率分布更加接近实际值.为了避免不必要的视图进出实视图窗口,我们采用视图挽留原则对近期受访 CUBE 进行保留,进一步改善了视图选择方法的性能.实验证明,在实时主动数据仓库环境下,我们的 PGreedy 算法比 FPUS 算法和 BPUS 算法具有更好的性能.

在未来的研究工作当中,我们将根据分析规则使用 CUBE 的特点,对分析规则进行分类,建立分类的标准,并设计有效的分类算法以及分类的动态调整算法.

**References**:

[1] Schrefl M, Thalhammer T. On making data warehouses active. In: Kambayashi Y, Mohania MK, Tjoa AM, eds. Proc. of the 2nd Int’l Conf. on Data Warehousing and Knowledge Discovery (DaWaK 2000). London: Springer-Verlag, 2000. 34−46.

[2] Thalhammer T, Schrefl M, Mohania M. Active data warehouses: Complementing OLAP with analysis rules. Data & Knowledge Engineering, 2001,39(3):241−269.

[3] Harinarayan V, Rajaraman A, Ullman JD. Implementing data cubes efficiently. In: Jagadish HV, Mumick IS, eds. Proc. of the 1996 ACM SIGMOD Int’l Conf. on Management of Data (SIGMOD’96). Montreal: ACM Press, 1996. 205−216.

[4] Gupta H, Harinarayan V, Rajaraman A, Ullman JD. Index selection for OLAP. In: Gray A, Larson P, eds. Proc. of the 13th Int’l Conf. on Data Engineering (ICDE’97). Birmingham: IEEE Computer Society Press, 1997. 208−219.

[5] Shukla A, Deshpande P, Naughton JF. Materialized view selection for multidimensional datasets. In: Gupta A, Shmueli O, Widom J, eds. Proc. of the 24th Int’l Conf. on Very Large Data Bases (VLDB’98). New York: Morgan Kaufmann Publishers, 1998. 488−499.

[6] Baralis E, Paraboschi S, Teniente E. Materialized view selection in a multidimensional database. In: Jarke M, Carey MJ, Dittrich KR, eds. Proc. of the 23rd Int’l Conf. on Very Large Data Bases (VLDB’97). Athens: Morgan Kaufmann Publishers, 1997. 156−165.

[7] Gupta H, Mumick IS. Selection of views to materialize in a data warehouse. IEEE Trans. on Knowledge and Data Engineering, 2005,17(1):24−43.

[8] Sapia C. PROMISE: Predicting query behavior to enable predictive caching strategies for OLAP systems. In: Kambayashi Y, Mohania MK, Tjoa AM, eds. Proc. of the 2nd Int'l Conf. on Data Warehousing and Knowledge Discovery (DaWaK 2000). London: Springer-Verlag, 2000. 224−233.

[9] Yao QS, An AJ. Using user access patterns for semantic query caching. In: Marik V, ed. Proc. of the 14th Int'l Conf. on Database and Expert System Applications (DEXA 2003). Prague: Springer-Verlag, 2003. 737−746.

[10] Tan HX, Zhou LX. Dynamic selection of materialized views of multi-dimensional data. Journal of Software, 2002,13(6): 1090−1096 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/13/1090.pdf

[11] Uchiyama H, Runapongsa K, Teorey TJ. A progressive view materialization algorithm. In: Proc. of the 2nd ACM Int'l Workshop on Data Warehousing and OLAP (DOLAP'99). Kansas: ACM Press, 1999. 36−41.

[12] Xu WG, Theodoratos D, Zuzarte C. Computing closest common subexpressions for view selection problems. In: Song IY, Vassiliadis P, eds. Proc. of the 9th ACM Int'l Workshop on Data Warehousing and OLAP (DOLAP 2006). Arlington: ACM Press, 2006. 75−82.

[13] Wang ZQ, Zhang DX. Optimal genetic view selection algorithm under space constraint. Int'l Journal of Information Technology, 2005,11(5):44−51.

[14] Derakhshan R, Dehne F, Korn O, Stantic B. Simulated annealing for materialized view selection in data warehousing environment. In: Hamza MH, ed. Proc. of the 24th IASTED Int'l Conf. on Database and Applications (DBA 2006). Innsbruck: IASTED/ACTA Press, 2006. 89−94.

[15] Xue YS, Lin ZY, Duan JJ, Lü XH, Zhang W. Dynamic selection of materialized views of multi-dimensional data with multi-users and multi-windows method. Journal of Computer Research and Development, 2004,41(10):1703−1711 (in Chinese with English abstract).

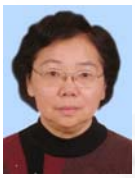
## 附中文参考文献:

[10] 谭红星,周龙骧.多维数据实视图的动态选择.软件学报,2002,13(6):1090−1096. http://www.jos.org.cn/1000-9825/13/1090.pdf

[15] 薛永生,林子雨,段江娇,吕晓华.用多用户多窗口处理多维视图动态选择.计算机研究与发展,2004,41(10):1703−1711.

**林子雨**(1978－),男,吉林柳河人,博士生,主要研究领域为数据库,实时主动数据仓库,数据挖掘.

**王腾蛟**(1973－),男,博士,副教授,CCF 高级会员,主要研究领域为数据库,数据仓库,Web 数据集成,数据挖掘.



**杨冬青**(1945－),女,教授,博士生导师,CCF 高级会员,主要研究领域为数据库,数据仓库,Web 数据集成,移动数据挖掘.

**唐世渭**(1939－),男,教授,博士生导师,CCF 高级会员,主要研究领域为数据库,半结构化数据,Web 数据集成,数据挖掘.

**宋国杰**(1975－),男,博士,讲师,主要研究领域为数据仓库,数据挖掘.