QCS:一种OLAP预防多维推理方法的研究

蔡伟珊 陈启买 刘 海
(华南师范大学计算机学院 广州 510631)

摘 要 针对目前多数联机分析处理(OLAP)推理控制方法计算复杂性高、实用性不强的问题,在前人研究基础上,提出一种改进的基于查询单元集QCS(Query Cells Set)的OLAP预防多维推理方法。该方法把OLAP查询的多维推理威胁预防检测放在查询涉及到的底层不相交的单元集(即QCS),而不是单个单元上,从而降低了推理威胁检测算法的计算复杂性,这更符合OLAP的查询处理要求。同时给出QCS方法的有效性证明和算法的实现,并用实例进行说明。与以往的推理控制方法相比,QCS方法不仅可有效保护OLAP系统的隐私信息,而且具有较高的计算效率,能满足OLAP系统的实用性要求。

关键词 OLAP,多维推理,推理控制,隐私保护
中图法分类号 TP311.13 **文献标识码** A

QCS:A Preventing Multi-dimensional Inference Approach for OLAP

CAI Wei-shan CHEN Qi-mai LIU Hai
(School of Computer, South China Normal University, Guangzhou 510631, China)

Abstract For high complexity and low practicability of most on-line analytical processing(OLAP) system inference control approaches, the paper proposed an improved preventing multi-dimensional inference approach on the basis of previous researches. This approach is based on the QCS(Query Cells Set). It puts preventing detection of multi-dimensional inference threat on the cells set(not simple cell) that requested cells of the query depend on, so the complexity of detection algorithm is reduced greatly, which meets the normal query processing requirement of OLAP. Then the effectiveness proof and algorithm were provided, and an example was used to illustrate the algorithm as well. Compared with former inference control approaches, QCS approach not only protects the sensitive data in OLAP system effectively, but also has better computationally efficiency, which meets the practical requirements of OLAP system.

Keywords OLAP, Multi-dimensional inference, Inference control, Privacy protect

1 引言

联机分析处理(OLAP)系统是目前在决策系统领域中最流行的决策支持和知识发现技术,而隐私泄露是OLAP系统面临的严峻安全问题。传统的数据库访问控制和数据清理等技术并不能彻底解决OLAP中因数据聚集产生的推理问题,所以推理控制是数据仓库理论与应用研究者重点关注的安全问题[1-7]。

推理控制是Denning等[8]最早在统计数据库中提出的,主要有数据干扰和查询约束两种方法。这两种方法都能保护隐私数据,但准确的查询结果对于依赖于这些数据的重要决策是非常重要的,所以本文主要针对查询约束方法进行研究。OLAP系统类似统计数据库,传统的基于多维数据模型的推理控制方法多从统计数据库的推理控制方法[8-10]演变而来。但继承统计数据库的局限性和查询效率问题[9,11],使得这类方法只适用于一些特定的环境,如只支持单种聚集函数的查询[1,2,12-15],或只能防止特定敏感数据泄露[12,13,15](如基于基数推理控制方法[13]),很难满足OLAP系统实用性要求。

针对这些方法的不足,Wang等人先后提出基于格的推理控制方法[6]和基于查询驱动的推理控制方法[7],这两种方法都是基于预防的推理控制方法,能有效预防各种聚集数据的推理威胁。但是,前者是依靠事先预测用户查询来防止敏感信息的泄露,属于一种静态的方法;后者的在线计算时间和空间都与查询集的底层数据集大小紧密相关。OLAP的在线交互性和海量数据聚集性,使得这两种方法都不适合实际应用中的OLAP系统。

所以,本文针对以上这些方法的优点和不足,以文献[7]提出的推理控制方法为基础,保留了原有方法的特点:细粒度、广泛适用性、有效性等;针对其运算效率低、实用性不强等问题,从实际应用需求深入研究,提出一种改进的基于查询单元集的预防多维推理方法QCS(Query Cells Set)。该方法把OLAP查询的多维推理威胁预防检测放在查询涉及到的底层

到稿日期:2011-10-08 返修日期:2012-02-17 本文受广东省科技计划基金项目(2009B010800036),广东省教育科研基金项目(BKYBJG20060235)资助。

蔡伟珊(1987-),女,硕士生,主要研究方向为高性能数据库,E-mail:caiweishan@gmail.com;陈启买(1965-),男,教授,主要研究方向为高性能数据库、数据仓库与数据挖掘;刘 海(1974-),男,博士,讲师,主要研究方向为学术信息服务、本体工程、高性能数据库。

不相交的单元集(即 QCS),而不是单个单元上,不仅有效防止了因多维推理而造成的敏感数据泄露,保护了 OLAP 系统隐私信息,而且具有较低的算法复杂性,能满足 OLAP 系统的实用性要求。

2 形式化数据模型

数据立方体是 OLAP 系统的基本数据模型。下面参考文献[6]中的相关术语给出数据立方体及其相关概念的形式化定义。

定义 1 数据立方体是一个二元组,DataCube=$\langle \mathcal{L}, \mathcal{A} \rangle$,

(1)$\forall t \in \mathcal{A}$是属性值的一个 k 元组,称为单元(Cell);

(2)$\mathcal{A}=\bigcup_{c \in L} c$ 是数据立方体中所有单元的集合;$c=\langle d_1, d_2, \cdots, d_k \rangle$是维级别的一个 k 元组,称为方体(Cuboids);

(3)$\mathcal{L}=\prod_{i=1}^{k} D_i$ 是数据立方体中所有方体的集合;$D_i=\{d_i^j \mid 1 \leqslant j \leqslant |D_i|\}$是第 i 个维度,$k>0$ 是维数;

(4)若方体(或单元)$c_1 \preceq c_2$,称 c_1 是 c_2 的祖先,c_2 是 c_1 的后代;$c_1 \preceq c_2$,$c_2 \preceq c_1$ 都不成立,称 c_1 与 c_2 不可比;把 Data-Cube 中最细粒度的方体称为核心方体(Core cuboid),记作 C_c。

这里从维层次和属性值层次对数据立方体做双重定义,数据立方体既是方体集 $\mathcal{L}$,也是单元集 $\mathcal{A}$。依照文献[16],本文给每一维增加一个顶层属性 all(有唯一值 ALL)。由维层次间的偏序关系可诱导出数据立方体的依赖格[17](Lattice,简称为格)结构,这里用偏序集$\langle \mathcal{L}, \preceq \rangle$表示方体格(见图 1)。因为这种偏序关系同样存在于单元间,所以,同理$\langle \mathcal{A}, \preceq \rangle$表示单元格。格结构中的偏序关系说明了数据立方体中数据间的依赖关系。

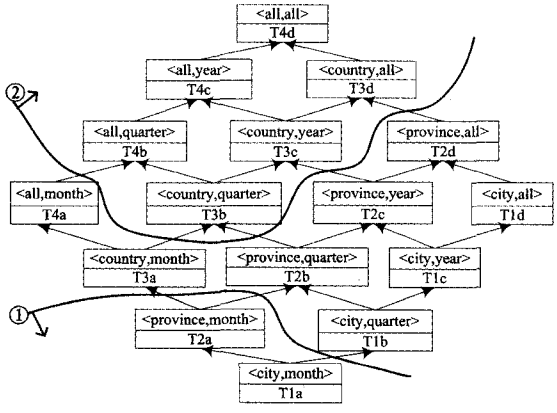

图 1 一个数据立方体方体格结构

图 1 为一个由核心方体〈city,month〉对应的事实表 T1a 诱导出的二维数据立方体方体格结构。其中包含 16 个方体(〈city,month〉,〈province,month〉,…,〈all,all〉);数据表(T1a,T2a,…,T4d)中的每个数据对就是一个单元(如〈Guangzhou,2009-10〉);〈city,quarter〉是〈city,month〉的后代,〈city,month〉是〈city,quarter〉的祖先,因为〈city,month〉 $\preceq$〈city,quarter〉。

敏感数据是相对概念,不被某用户或者角色直接访问的数据对该用户或者角色就是敏感、不可见的。作为推理控制的对象,敏感数据也是数据立方体的一部分。根据安全系统的设计需求,用户的敏感数据和查询数据都应该既可以是某些方体集,也可以是满足某些条件的单元集。从这个意义上讲,本文对敏感集和查询集作如下形式化定义。

定义 2 在 DataCube=$\langle \mathcal{L}, \mathcal{A} \rangle$中,

(1)定义 $Slice(r)=\{t \mid t \in A, \exists t' \in r \rightarrow t \preceq t' \vee t' \preceq t\}$,$Slice(r)$为单元集 r 的切片闭集;

(2)定义 $Below(B)=\{c \mid c \in L, \exists c' \in B \rightarrow c \preceq c'\}$,$Below(B)$为方体集 B 的后代闭集;

(3)定义 $Sensitive(S)=\{t \mid t \in Slice(r_i), t \in c \rightarrow c \in Below(B_i)\}$,$S=\{(r_i, B_i) \mid 1 \leqslant i \leqslant n\}$,$Sensitive(S)$是由规范集 S 定义的敏感集。

$Below(B)$可看作是数据立方体垂直维度上关于方体集 B 的一个划分,$Slice(r)$是数据立方体水平单元上关于单元集 r 的一个划分。敏感集 $Sensitive(\{(r, B)\})$是 $Slice(r)$和 $\bigcup_{c \in Below(B)} c$ 的交集,且 $Sensitive(S)=\bigcup_{i=1}^{n} Sensitive(\{(r_i, B_i)\})$。

例 1 假设用户要分析图 1 数据立方体中的数据,但受到这样的限制:不能看到各省 2010 年第一季度之前每月的销售量数据。因此方体〈province,month〉和〈city,month〉中 quarter<2010-Q2 的数据对于该用户都是敏感的。由定义 2 知,该敏感集形式化为 $Sensitive(S)$,$S=\{(r_1, B_1)\}$,$r_1=\{\langle s, q \rangle \mid s \in \text{province}, q \in \text{quarter}, q < \text{2010-Q2}\}$,$B_1=\{\langle \text{province}, \text{month} \rangle\}$;$Sensitive(S)$对应图 1 中①线以下的数据集,包含了该用户的所有不能访问的数据。

本文假设系统只需定义敏感集,敏感集里的数据对于用户都是不可访问的;而对应非敏感数据,即 DataCube-$Sensitive(S)$则默认为授权集,允许用户访问。

如果用户得到了查询数据 S,用户根据格结构中的偏序关系能从 S 推理得知 S 所有后代数据,因此,这里定义的查询集包含用户真正查询数据及其所有的后代数据。

定义 3 在 DataCube=$\langle \mathcal{L}, \mathcal{A} \rangle$中,

(1)定义 $Above(V)=\{c \mid c \in \mathcal{L}, \exists c' \in V \rightarrow c' \preceq c\}$,$Above(V)$为方体集 V 的祖先闭集;

(2)定义 $Query(Q)=\{t \mid t \in Slice(u_k), (t \in c \rightarrow c \in Above(V_k))\}$,$Q=\{(u_k, V_k) \mid 1 \leqslant k \leqslant m\}$,$Query(Q)$为由规范集 Q 定义的查询集。

查询集 $Q=\{(u, V)\}$是 $Slice(u)$和 $\bigcup_{c \in Above(V)} c$ 的交集,且有 $Query(Q)=\bigcup_{k=1}^{m} Query(\{(u_k, V_k)\})$。

例 2 设用户要查询 2010 年第一季度各国家的季度销售情况。由定义 3 知,该查询的查询集形式化为 $Query(Q1)$,$Q1=\{(u_1, V_1)\}$,$u_1=\{\langle c, q \rangle \mid c \in \text{country}, q \in \text{quarter}, q = \text{2010-Q1}\}$,$V_1=\{\langle \text{country}, \text{quarter} \rangle\}$;$Query(Q1)$对应图 1 中②线以上满足切片条件 $Slice(u_1)$的数据集,包含用户在这个查询中所有能直接查询和间接推理的数据。

用户通过查询除得到查询集的数据外,亦可结合多次历史查询结果多维推理出非历史查询集里的数据。为更好地描述由查询能推理出的所有数据,这里引入推理集[7]的概念,它包含查询集和通过查询集间接得到的所有单元集。

定义 4 在 DataCube=$\langle \mathcal{L}, \mathcal{A} \rangle$中,定义 $Inferable(Q)=\{t \mid t \in \bigcap_{k \in K \subseteq [1,m]} Slice(u_k) \rightarrow t \in c \wedge c \in Above(\text{GLB}(\bigcup_{k \in K} V_k))\}$,$Inferable(Q)$是关于查询集 $Query(Q)$的推理集[1)]。

1) 最大下届 $\text{GLB}(\mathcal{L})=\{x \mid \forall y (y \in \mathcal{L} \wedge x \preceq y \wedge \exists x' (x \preceq x' \preceq y) \rightarrow x'=x)\}$

例3　设用户在例2的 *Query*(Q1)查询后，又查询了2010年各省的年销售情况，这个查询表示为 $Q2=\{(u_2, V_2)\}$，$u_2=\{\langle p, y\rangle \mid p\in \text{province}, y\in \text{year}, y=2010\}$，$V_2=\{\langle \text{province}, \text{year}\rangle\}$，记 $Q=Q1\cup Q2$，则两次查询集的推理集 $Inferable(Q)=Query(Q)\cup\{\langle p, q\rangle \mid p\in \text{province}, q\in \text{quarter}, q=2010\text{-}Q1\}$；说明用户能通过这两次查询结果二维推理出祖先方体⟨province，quarter⟩的部分数据。

3　推理控制理论

当敏感集的某些数据被用户得到的查询集结合外部知识推理得到时，则存在推理通道。根据推理源的不同，推理通道可分为两种：一维推理(1-d Inference)和多维推理(m-d Inference)。一维推理是敏感数据通过其单个后代推理得到；而多维推理是一维推理的复杂形式，其通过两个或两个以上的后代数据推理得到敏感数据。

不同于一维推理，多维推理往往涉及非敏感集的多个后代和多种聚集数据的结合推理，其推理通道的检测必然需要检测所有后代和各种聚集数据的组合结果，显然这种检测是不可行的。所以，常见的基于检测推理控制方法一般是基于一些特殊情况提出的，如只考虑 SUM 聚集函数[1,12-15]，但一般 OLAP 系统不可能只涉及一种聚集函数。所以，本文提出的 QCS(Query Cells Set)方法是一种预防推理方法，该方法虽然一定程度上降低了系统可用性，却能有效地避免所有可能的多维推理威胁，使得推理控制在 OLAP 系统上的有效实施成为可能。

由格结构中的偏序关系可知，除非通过外部知识能够得到在不可比方体间或单元间的关系，否则数据立方体中只存在后代数据引起的推理，因此在消除外部知识威胁的前提下，可以只考虑由后代数据引起的推理；一般分析中存在的聚集函数有分布(如 sum，count)、代数(如 average，min)和整体(如 median，mode) 3类，整体函数一般是直接从事实表中计算结果，而代数函数一般可转换成分布函数来计算(如 *average*=*sum*/*count*)。因此，可以只考虑分布聚集函数的推理问题。在无外部知识威胁和只有分布聚集函数的假设前提下，对于源数据 S 关于敏感数据 x 的推理判断只需要考虑最小推理集 $Basis(S,x)$，因为 $S\text{-}Basis(S,x)$ 对这个推理判断取决于 $Basis(S,x)$。

定义 5[7]　在 DataCube=$\langle \mathscr{L}, \mathscr{A}\rangle$中，定义 $Basis(S,x)=\{t \mid t\in S, x\leq t, \exists t'(t'\in S\wedge t'\leq t\rightarrow t'=t)\}$，$Basis(S,x)$ 为单元集(或方体集)S 中关于 x 的最小推理集。

如例1的非敏感集 $\mathscr{L}\text{-}Sensitive(S)$ 关于敏感方体⟨province，month⟩的最小推理集为 $Basis(\mathscr{L}\text{-}Sensitive(S), \{\langle \text{province}, \text{month}\rangle\})=\{\langle \text{country}, \text{month}\rangle, \langle \text{province}, \text{quarter}\rangle\}$。

$Basis(S,x)$ 是推理源 S 对目标 x 的一个最小推理集合，显然当 $|Basis(S,x)|>1$ 时，可能存在从 S 到 x 的多维推理通道。因此，根据定理1结论，当且仅当 $Inferable(Q)\cap Sensitive(S)\neq\emptyset$ 时，敏感数据可被查询或存在多维推理威胁，也就是说，存在敏感数据泄露。

定理 1[7]　在 DataCube=$\langle \mathscr{L}, \mathscr{A}\rangle$中，给定查询集 *Query*(Q)，对于 $\forall t\in \bigcup_{k=1}^{m} Slice(u_k)$，当且仅当 $t\in inferable(Q)$ 时，下面两个结论必有一个成立：(1) $t\in Query(Q)$；(2) $\forall T\supseteq Query(Q)$ 满足 $|Basis(T,t)|>1|$。

因为一个 OLAP 查询往往涉及到核心方体层上一个连续的单元集而不是单个的单元，所以，本文把查询的推理集和敏感集的交集检测放在查询涉及到的 C_c 不相交的单元集(这里称为查询单元集 QCS)，而不是单个单元 Cell 上。这样做减少了敏感泄露威胁检测算法的计算复杂性，更具合理性，符合 OLAP 的查询处理要求。为预防从推理集到敏感集的多维推理，本文定义了两个函数。

定义 6　在 DataCube=$\langle \mathscr{L}, \mathscr{A}\rangle$中，定义 $Rsensitive(S,T)=\max(\{c \mid c\in \mathscr{L}, T\subseteq C_c, c\cap Sensitive(S)\cap Slice(T)\neq\emptyset\})$；$Rsensitive(S,T)$ 是敏感集 $Sensitive(S)$ 关于查询单元集 T 的极大方体集。

定义 7　在 DataCube=$\langle \mathscr{L}, \mathscr{A}\rangle$中，定义 $Rinferable(Q,T)=\min(\{c \mid c\in \mathscr{L}, T\subseteq C_c, c\cap Inferable(Q)\cap Slice(T)\neq\emptyset\})$；$Rinferable(Q,T)$ 是推理集 $Inferable(Q)$ 关于查询单元集 T 推测集的极小方体集。

由以上定义可知，任意包含 T 后代的敏感方体 c 都是 $Rsensitive(S,T)$ 部分元素的祖先；$Rinferable(Q,T)$ 只有一个元素，且任意包含 T 后代且与推理集 $Inferable(Q)$ 有相交的方体 c 都是 $Rinferable(Q,T)$ 的后代。因此，如果 $Rinferable(Q,T)$ 是部分 $Rsensitive(S,T)$ 的祖先时，则可能存在多维推理通道。

定理 2　对于 DataCube=$\langle \mathscr{L}, \mathscr{A}\rangle$，给定敏感集 $Sensitive(S)$ 和查询集 $Query(Q)$，当且仅当存在查询单元集 $r_c\subseteq C_c$，方体 $c_s\in Rsensitive(S,r_c)$ 满足 $Rinferable(Q,c_s)$ 时，$Inferable(Q)\cap Sensitive(S)\neq\emptyset$，即敏感数据可能被查询或存在多维推理威胁。

证明(先证充分性)：假设 $\exists r_c\in C_c, c_s\in Rsensitive(S,r_c)$ 满足 $Rinferable(Q,r_c)\leq c_s$，令 $\{c_q\}=Rinferable(Q,r_c)$，即 $c_q\leq c_s$。由以上定义知，$Slice(r_c)\cap c_s\neq\emptyset, Slice(r_c)\cap c_q\neq\emptyset$。令 $t_q\in Slice(r_c)\cap c_q$，则 $t_q\in Inferable(Q)$。再由 $c_q\leq c_s$，存在 $t_s\in Slice(r_c)\cap c_s$ 满足 $t_q\leq t_s$。故由 $t_q\in Slice(t_s), c_q\in Below(c_s)$ 推得 $t_q\in Sensitive(S)$。综上，$t_q\in Inferable(Q)\cap Sensitive(S)$，充分性得证。(必要性)假设 $Inferable(Q)\cap Sensitive(S)\neq\emptyset$，令 $t\in Inferable(Q)\cap Sensitive(S), t\in c, I=\{i \mid 1\leq i\leq n, t\in Slice(r_i)\}, K=\{k \mid 1\leq k\leq m, t\in Slice(r_k)\}$。由 $t\in Sensitive(S)$ 得，$c\leq c_s, c_s=\text{GLB}(\bigcup_{i\in I} B_i)$。由 $t\in Inferable(Q)$ 推得 $c_q\leq c, c_q=\text{GLB}(\bigcup_{k\in K} V_k)$，综上，$c_q\leq c_s$。令 $r_c=\{t_c \mid t_c\leq t, t_c\in C_c\}$，则 $r_c\in \bigcap_{i\in I} Slice(r_i)\bigcap_{k\in K} Slice(r_k)\cap C_c$，故 $\forall t_c\in r_c$。由 $c_s\cap Slice(t)\subseteq Sensitive(S), c_q\cap Slice(t)\subseteq Inferable(Q)$ 推得，$c_s\in Rsensitive(S,\{t_c\}), c_q\in Rinferable(Q,\{t_c\})$，且满足 $c_q\leq c_s$，必要性得证。

定理2表明，当且仅当存在查询单元集 T，$Rinferable(Q,T)$ 是 $Rsensitive(S,T)$ 部分元素的祖先时，推理集和敏感集相交。如果系统响应了该查询会令，敏感数据被直接(通过查询结果)或者间接泄露；反之，如不存在这样的 T，响应该查询不会对敏感数据造成多维推理威胁。由此，可通过检测 $Rinferable(Q,T)$ 和 $Rsensitive(S,T)$ 的祖先后代关系来预防多维推理威胁。

4　QCS 方法的实现

本文提出的预防多维推理方法是通过查找上节构造的查询单元集 QCS 来实现的，所以把这个方法称为 QCS 方法。基于查询的推理控制是一种动态推理控制，是把用户当前和历史查询集作为当前推理控制推理源，并及时更新历史查询

集。因此，QCS方法需包含两步：首先对新传入的查询请求，检查该查询集联合历史查询推理集是否对敏感集产生多维推理威胁(见4.1节)；其次若确定查询是安全的，除响应该查询请求外，需将该查询更新到用户历史查询中(见4.2节)。

4.1 检查新查询的多维推理威胁

算法1遍历每个与新查询集 $NewRq$ 相交的非空的QCS，计算基于QCS的推理集的极小方体集和 $NewRq$ 的最大上界方体 C_q；然后判断同样基于该QCS的敏感集的极大方体集的每个方体 C_b 是否是 C_q 的后代或与其重叠。

算法1 CheckQuery

```
CheckQuery(Rinferable[]HisRis,Query NewRq(newu,newV),Rsensitive Rsensitives)
输入：历史推理集 HisRis，新查询 NewRq(newu, newV)，敏感集 Rsensitives
输出：当新查询不存在多维推理威胁，返回 True；否则返回 False，表示拒绝响应该查询请求
Begin
newu=Slice(newu)∩Cc;// Cc 为核心单元集
Foreach HisRi in HisRis
If(newu∩HisRi. u≠∅)
  uunit=newu∩HisRi. u;// HisRi. u 是一个 QCS
  Cq=GLB(HisRi. V,newV);
  Foreach Rs in Rsensitives
    If(uunit∩Rs. r≠∅) //历史推理集和敏感集在某个 QCS 有重叠
      Foreach Cb in Rs. B
        If(Cq≤Cb) Return False;//可能引起多维推理威胁
Return True;
End Begin
```

算法分析：(1)有效性：算法1通过检测基于每个QCS的推理集和敏感集之间是否相交来预防多维推理威胁，根据定理2的结论可证明该算法是有效的；(2)复杂性：令 n=length({$HisRi$ | $HisRi \in HisRis, newu \cap HisRi.u \neq \emptyset$})，算法时间复杂度主要在第3、7、9行开始的3个嵌套循环上，其计算次数为 $n \cdot$ length($Rsensitives$) $\cdot$ length($Rs.B$)=O(n)(因相比 n，其它两项可看作常数)，所以算法的时间复杂度是O(n)；而空间复杂度则为O(length($HisRis$))。文献[7]算法的时间和空间复杂度分别是O(|$newu$|))和O(|C_c|)，因为 $n \ll$O(|$newu$|)，length($HisRis$)$<<$|C_c|，与文献[7]算法相比，算法1在时间和空间上的复杂度都有很大程度的降低，可更好满足OLAP系统的实用性要求[2)]。

4.2 更新历史查询推理集

更新历史查询推理集的过程是，把一个没有多维推理威胁的新查询添加到用户的历史查询集，形成新的历史推理集，作为下次推理威胁检查推理源的一部分。算法2的时间和空间复杂度都是O(length($HisRis$))[2)]。

算法2 UpdateRinferable

```
UpdateRinferable(Rinferable[] HisRis, Query NewRq(unew, Vnew))
输入：历史推理集 HisRis，单个新查询 NewRq
输出：更新后的历史推理集 HisRis
Begin
unew=Slice(unew) ∩Cc;// Cc 为核心单元集
Foreach HisRi in HisRis
If(unew∩HisRi. u ≠∅)
  uunit=unew∩HisRi. u;//HisRi. u 是一个 QCS
  Vunit=GLB(HisRi. V,Vnew);
  If(uunit=HisRi. u) //新查询单元包含某个 QCS
    If(Vunit ≥HisRi. V) unew=unew−uunit;
    Else HisRi. V=Vunit;unew=unew−uunit;
  ElseIf(uunit=unew) //新查询单元被某个 QCS 包含
    If(Vunit ≥HisRi. V) Return HisRis;
    Else HisRis. Add(Rinferable(uunit,Vunit));
      HisRi. u=HisRi. u− uunit;Return HisRis;
  Else //新查询切块与某 QCS 互不相包含
    If(Vunit ≥HisRi. V) unew=unew−uunit;
    Else HisRis. add(Rinferable(uunit,Vunit));
      HisRi. u=HisRi. u− uunit;unew=unew−uunit;
If(newu≠∅) HisRis. Add(Rinferable(unew,Vnew));
Return HisRis;
End Begin
```

5 举例说明

例4 假设事实表T1a中有100个城市，5年(2006−2010)共6000条数据。根据例1的假设，敏感集 $Rsensitives$={$Rs1$}，$Rs1.r$={⟨c,m⟩|$c \in$ city,$m \in$ month,$m<$2010-4}，$Rs1.B$=$B1$。并假设历史推理集为空，即 $HisRis=\emptyset$；

(1) 检测例2的查询 $Query(Q1)$：由算法1可检测到该查询是可以被允许的，并通过算法2得到更新后的历史推理集 $HisRis$={$HisRi1$}，$HisRi1.u$={2010-1≤$month$≤2010-3}，$HisRi1.V=V_1$；

(2) 检测例3的查询 $Query(Q2)$：由算法1，$uunit=Slice(u_2) \cap C_c \cap HisRi1.u=HisRi1.u$，$C_q$=GLB($HisRi1.V$,$V_2$)=⟨province,quarter⟩，$C_q \leq B_1$ 不成立，所以该查询是可以被允许的；更新后的历史推理集 $HisRis$={$HisRi1$,$HisRi2$}，其中 $HisRi1.u$={2010-1≤month≤2010-3}，$HisRi1.V$=⟨province,quarter⟩；$HisRi2.u$={2010-4≤month≤2010-6}，$HisRi2.V$=⟨province,year⟩；

(3)检测输入的新查询 $Query(Q3)$：$Q3$={(u_3,V_3)}，u_3={⟨a,m⟩| $a \in$ all,$m \in$ month,m=2010-3}，V_3={⟨country,month⟩}。由算法1，$uunit=Slice(u_3) \cap C_c \cap HisRi1.u$={month=2010-3}，$C_q=GLB(HisRi1.V,V2)$=⟨province,month⟩=$B_1$，所以新查询可能引起多维推理威胁，必须拒绝该查询。

表1 算法1和文献[7]算法相关数据对比

次数/大小 / 记录数		算法1		文献[7]算法	
		时间	空间	时间	空间
Query(Q1)	300	1	1	300	6000
Query(Q2)	1200	2	2	1200	6000
Query(Q3)	100	1	2	100(最坏)	6000

表1为关于以上3个查询检测多维推理威胁的过程算法1与文献[7]算法的计算次数、所用空间大小和查询涉及到的底层数据表T1a记录数的对比结果。

6 性能优化策略

任何安全机制的实施都会影响系统的服务性能，推理控

(下转第190页)

2) $length(v)$表示向量 v 的长度；|T|表示单元集 T 中的单元个数。

[R]. TR-703-04. Princeton University, 2004

[5] Wang Hao-jun, Zimmermann R, Ku W-S. ASPEN: An Adaptive Spatial Peer-to-Peer Network[C]// Proceedings of the 13th Annual ACM International Workshop on Geographic Information Systems. Bremen, Germany, 2005: 230-239

[6] Ratnasamy S, Francis P, Handley M, et al. A Scalable Content-Addressable Network[C]// Proceedings of the ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications. San Diego, California, United States, 2001: 161-172

[7] Ganesan P, Yang B, Garcia H. One Torus to Rule them All: Multidimensional Queries in P2P Systems[C]// Proceedings of the 7th International Workshop on the Web and Databases(WebDB '04). Pairs, France, 2004: 19-24

[8] Schutt T, Schintke F, Reinefeld A, et al. Structured Overlay without Consistent Hashing: Empirical Results[C]// Proceedings of the 6th IEEE International Symposium on Cluster Computing and the Grid Workshops (CCGRIDW'06). Singapore, 2006: 8

[9] Guttman A. R-trees: A Dynamic Index Structure for Spatial Searching[C]// Proceedings of the 1984 ACM SIGMOD International Conference onManagement of Data. Boston, Massachusetts, United States, 1984: 47-57

[10] Samet H. The quadtree and related hierarehieal data struetures [J]. ACM Computer Surveys, 1984, 16(2): 187-260

[11] Bentley J L. Multidimensional binary search trees used for associative searching[J]. Communication of ACM, 1975, 18(9): 508-517

[12] Aspnes J, Shah G. Skip Graphs[J]. ACM Transactions on Algorithms, 2007, 3(4): 37

[13] Faloutsos C, Roseman S. Fractals for Secondary Key Retrieval [C]// Proceedings of the 8th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS '89). Philadelphia, Pennsylvania, United States, 1989: 247-252

(上接第 181 页)

制也不例外。所以在考虑系统安全性的同时，需要兼顾信息损失和代价两个因素对系统的影响。本节从 OLAP 系统实际应用的特殊性，给出了 QCS 方法的两种性能优化策略。

(1)优化历史推理集。基于查询的推理控制方法大多和历史查询集大小有关，系统的查询响应效率会随着查询集的增大而降低，QCS 方法也不例外。因为大多数用户一般不会基于一个长时间、大量的查询历史来推导敏感数据，所以在安全许可情况下，可定时删除过期历史查询集，或设定优化阈值来限制查询集大小等。当然这些策略必须对用户透明，以避免被用户恶意利用。

(2)基于角色的推理控制。基于角色的访问控制(RBAC)模型本质是把权限指派给角色而不是单个用户，所以，引用这种思想，在 QCS 方法中以角色为单位来维护查询历史推理集，这样既减少系统的性能负担，又可防止同角色里的用户间的串谋推理威胁。

结束语 本文着重讨论 OLAP 系统多维数据集的多维推理控制问题。提出的 QCS 方法能有效预防多维推理威胁，有较好的多维推理威胁检测效率。其算法的计算复杂性只与历史推理集长度有关，与底层数据集大小无关。QCS 方法是基于查询的动态推理控制，在提高系统安全性的同时牺牲了一定的在线查询响应时间。鉴于静态推理方法在这方面的优势，今后，我们将结合静动态推理控制各自的优势做进一步的优化研究；另外，本文的 QCS 方法没有考虑不同用户或角色间的串谋推理威胁，如何防止串谋推理威胁也是今后的研究重点。

参 考 文 献

[1] Hua M, Zhang S, Wang W, et al. FMC: An approach for privacy preserving OLAP[J]. Lecture Notes in Computer Science, 2005, 3589: 408-417

[2] Agrawal R, Srikant R, Thomas D. Privacy Preserving OLAP [C]// Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data. Baltimore: Maryland, 2005: 251-262

[3] Zhang N, Zhao W. Privacy-Preserving OLAP: An Information-Theoretic Approach [J]. Knowledge and Data Engineering, 2011, 23(1): 122-138

[4] 周海清，陈启买，刘海. 基于数据立方体的数据仓库安全控制[J]. 计算机工程，2010，36(10)：152-154

[5] 毛奇正，柏文阳，刘奇志. 元数据相关推理研究[J]. 计算机科学，2004，31(11)：86-88

[6] Wang L, Wijesekera D, Jajodia S. Lattice-based Inference Control in Data Cubes [J]. Journal of Computer Security, 2004, 12(5): 655-692

[7] Wang L, Li Y, Jajodia S, et al. Preserving Privacy in On-Line Analytical Processing(OLAP)[J]. Advances in Information Security, 2007, 29: 147-167

[8] Denning D E, Schlorer J. Inference controls for statistical databases [J]. IEEE Computer, 1983, 16(7): 69-82

[9] Chin F, Ozsoyoglu G Y. Auditing and Inference Control in Statistical Databases[J]. IEEE Transactions on Software Engineering, 1982, 8(6): 574-582

[10] Chin F. Security Problems on Inference Control for Sum, Max, and Min Queries[J]. Journal of the ACM, 1986, 33(3): 451-464

[11] Kleinberg J, Papadimitriou C. Raghavan PAuditing Boolean Attributes[J]. Journal of Computer and System Sciences, 2003, 66(1): 244-253

[12] Wang L, Li Y, Wijesekera D, et al. Precisely Answering Multi-Dimensional Range Queries without Privacy Breaches[J]. Computer Security, 2003, 2808: 100-115

[13] Wang L, Li Y, Wijesekera D, et al. Cardinality-Based Inference Control in Data Cubes[J]. Journal of Computer Security, 2004, 12(5): 655-692

[14] Sung Y, Liu Y, Xiong H, et al. Privacy Preservation for Data Cubes[J]. Knowledge and Information Systems, 2006, 9(1): 38-61

[15] Li Y, Lu H, Deng R H. Practical Inference Control for Data Cubes[C]// Proceedings of IEEE Transactions on Dependable and Secure Computing. Canana: IEEE Computer Society, 2006: 115-120

[16] Gray J, Bosworth A, Bosworth A, et al. Data cube: A relational aggregation operator generalizing group-by, cross-tab and sub-totals[J]. Data Mining and Knowledge Discovery, 1997, 1(1): 29-53

[17] Donnellan T. Lattice Theory[M]. Pergamon Press, Oxford, 1968