k-匿名数据集的增量更新算法

宋金玲[1,2] 赵 威[2] 刘 欣[1] 黄立明[1] 李金才[2] 刘国华[2]
(河北科技师范学院 秦皇岛 066004)[1] (燕山大学信息科学与工程学院 秦皇岛 066004)[2]

摘 要 发布k-匿名数据集可以起到有效保护隐私的目的,但如何保持k-匿名数据集与原始数据集的同步更新是一个亟待解决的问题。为了解决这个问题,在详细分析k-匿名数据集更新情况的基础上,给出了k-匿名数据集的增量更新算法:针对具体的更新操作,首先根据语义贴近度及元组映射等方法对更新元组在k-匿名数据集中进行定位,再对更新元组进行相应的更新操作。所提算法不仅保证了数据集的k-匿名约束性质,而且保证了k-匿名数据集与原始数据集的实时一致性。

关键词 k-匿名,增量更新,插入,删除,修改
中图法分类号 TP309.2 **文献标识码** A

Algorithm for Increment Update of k-Anonymized Dataset

SONG Jin-ling[1,2] ZHAO Wei[2] LIU Xin[1] HUANG Li-ming[1] LI Jin-cai[2] LIU Guo-hua[2]
(Hebei Normal University of Science & Technology, Qinhuangdao 066004, China)[1]
(Department of Computer Science and Engineering, Yanshan University, Qinhuangdao 066004, China)[2]

Abstract K-anonymity is an effective method to prevent linking attack and protect privacy. The main idea of k-anonymity is generalizing the values on a set of special attributes named quasi-identifier, so that gains a k-anonymized dataset in which the values of each tuple on quasi-identifier must repeat at least k occurrences. Although k-anonymized dataset guarantees privacy, the k-anonymized dataset needs to be updated constantly because the original dataset updates occasionally after a version of k-anonymized dataset has been existed. So, how to update the k-anonymized dataset as well as the original dataset becomes an urgent problem. To solve this problem, based on the detailed analysis to various update situations of the k-anonymized dataset, the increment update algorithms for the k-anonymized dataset were presented. Distinguishing the specific update operation, the position of the tuple to update was located firstly by different location methods such as Semantic Similarity Degree, tuple mapping. Then the corresponding update operation was processed to the updating tuple after the location. The above update algorithms not only can ensure the k-anonymized dataset achieving k-anonymity constraints, but also can guarantee the real-time consistency between k-anonymized dataset and original dataset.

Keywords k-anonymity, Increment update, Insert, Delete, Modify

1 引言

k-匿名[1]方法是防止连接攻击、保护发布数据隐私安全的一种有效方法,其主要思想是对原始数据集中称为准标识符的一组特殊属性上的值进行泛化,使每条元组在准标识符上的值至少重复k次,即满足k-匿名约束,并将得到的k-匿名数据集进行发布。现有的k-匿名研究只考虑了静态数据集的k-匿名化方法,忽略了数据集的动态更新特性。当一个k-匿名数据集版本发布后,原始数据集仍会不断更新,如插入新的元组、删除过时的元组或对某些错误的元组进行修改。如果k-匿名数据集不能及时更新,与原始数据集保持一致,则k-匿名数据集的有效性和可用性将会大大降低,同时制约了k-匿名方法的实用性。因此,k-匿名数据集的更新问题亟待解决。

k-匿名数据集的朴素更新方法是,针对更新后的原始数据集重新生成一个k-匿名数据集。对于数据更新量大的情况,该方法是比较实用的。但对于数据更新量小的情况,采用该方法不仅会大大增加系统开销,而且会导致多个k-匿名数据集版本的出现,从而可能造成新的信息泄露[2]。一般情况下,原始数据集的更新量都比较小,如果针对原始数据集的更新操作对已有k-匿名数据集进行增量更新,不仅能有效减轻

到稿日期:2009-05-20 返修日期:2009-07-27 本文受国家自然科学基金(No. 60773100),河北省自然科学基金(No. F2009000475),秦皇岛市科学技术研究与发展计划项目(2008-1-12)资助。

宋金玲(1973-),女,博士生,讲师,主要研究领域为信息安全,E-mail:songjinling99@126.com;赵 威(1974-),男,博士生,主要研究领域为数据库安全;刘 欣(1968-),女,讲师,主要研究领域为信息安全;黄立明(1972-),男,副教授,主要研究领域为信息安全、数据库理论;李金才(1984-),硕士生,主要研究领域为信息安全;刘国华(1966-),男,博士,教授,博士生导师,CCF高级会员,主要研究领域为数据库理论、数据库安全、Web数据管理。

系统负担，而且避免了多个 k-匿名数据集版本的出现。

由于 k-匿名数据集的某些属性值是原始数据集中值的泛化值，因此对 k-匿名数据集的增量更新，需要考虑如下问题：

①如何在 k-匿名数据集中定位更新元组，并进行相应的更新操作；

②如何保证更新后数据集的 k-匿名约束性质。

本文分别针对原始数据集的插入、删除、修改 3 种更新操作，对 k-匿名数据集的增量更新方法进行研究。首先，针对不同的更新操作，利用语义贴近度、元组和泛化元组的映射关系，对待更新元组在 k-匿名数据集中进行定位；然后再进行相应的更新操作。为了保证数据集的 k-匿名约束性质，更新后对小于 k 的 QI 组进行合并。所提算法不仅能保证 k-匿名数据集与原始数据集的实时一致性，而且能避免原始数据集的微小改动而导致 k-匿名数据集的较大变化。

2 相关工作

目前的 k-匿名研究主要集中在 k-匿名化方法上。文献[3]采用的是 Datafly 算法，文献[4]采用的是 μ-Argus 算法，Datafly 和 μ-Argus 算法为 k-匿名方法的进一步研究奠定了基础。为了提高生成表的数据精度，文献[5]采用了 Mingen 算法以求得到最高精度的生成表。文献[6]和文献[7]分别证明了获得最高精度的 k-匿名表是 NP 困难问题，并分别给出了泛化单元数为最小泛化 $O(k\log k)$ 倍和 $O(k)$ 的近似算法。文献[8]给出的是对所有属性值进行泛化的全域 k-匿名化算法——Incognito 算法。文献[9]给出的是对多个属性同时进行泛化的多维 k-匿名化算法。文献[10]给出了一种由上到下的 k-匿名化算法。Liu XY 等人在文献[11,12]中分别提出了 Classfly 算法和支持多 k-匿名约束的 Classfly+ 算法。A. Machanavajjhala 等人在文献[13]中提出了比 k-匿名模型功能更强的 ℓ-多样化模型。文献[14]指出除了对准标识符属性进行泛化外，某些敏感信息也需要进行不同层次的泛化。2007 年，Hyoungmin 等人在文献[15]中给出了泛化单元数为最小泛化 $O(\log k)$ 倍的近似算法。

虽然研究者们所提出的 k-匿名化算法各有所长，但是上述 k-匿名化算法都是针对静态数据集进行的，没有考虑到原始数据集动态更新情况。本文主要研究 k-匿名数据集的更新一致性问题，即在原始数据集更新的情况下，如何对 k-匿名数据集进行增量更新，研究成果是对现有 k-匿名算法的有效补充。

3 基本定义

本文的数据集为一个关系表，模式为 $R(A^{QI},A^S)$，其中 $A^{QI}=\{A_1^{QI},A_2^{QI},\cdots,A_n^{QI}\}$ 为准标识符属性集，A^S 为敏感属性。为了叙述简单，下面也用 R 表示数据集实例。对 $A\subseteq A^{QI}\cup A^S$，$R[A]$ 表示表 R 在属性集 A 上包含重复值的投影，$t[A]$ 表示元组 t 在属性集 A 上的值。

定义 1（k-匿名约束） 对数据集 $R(A^{QI},A^S)$，如果 $R[A^{QI}]$ 中每个元组的重复次数至少为 $k(k\geqslant 2)$，则称数据集 R 满足 k-匿名约束。

例 1 对表 1(b)所示数据集 $R^*(Age,Zip,Problem)$，当 R^* 的准标识符为 $\{Age,Zip\}$ 时，由于 $R^*[Age,Zip]=\{([21,25],[11k,20k]),([21,25],[11k,20k]),([41,50],[21k,30k]),([41,50],[21k,30k]),([51,55],[51k,60k]),([51,55],[51k,60k])\}$，其中元组 $([21,25],[11k,20k])$，$([41,50],[21k,30k])$ 和 $([51,55],[51k,60k])$ 的重复次数都为 2，因此 R^* 就满足 2-匿名约束。

表 1 原始数据集 R 及其 2-匿名数据集 R^*，其中 $A^{QI}=\{Age,Zip\}$

Tuple ID	Age	Zip	Problem
t1	21	12000	flu
t2	23	18000	gastritis
t3	48	28000	flu
t4	42	23000	gastritis
t5	49	25000	insomnia
t6	52	52000	flu
t7	53	59000	gastritis

(a)Microdata R

Tuple ID	QG	Age	Zip	Problem
t1*	1	[21, 25]	[11k, 20k]	flu
t2*	1	[21, 25]	[11k, 20k]	gastritis
t3*	2	[41, 50]	[21k, 30k]	flu
t4*	2	[41, 50]	[21k, 30k]	gastritis
t5*	3	[41, 50]	[21k, 30k]	insomnia
t6*	3	[51, 55]	[51k, 60k]	flu
t7*	3	[51, 55]	[51k, 60k]	gastritis

(b)2-anonymized dataset R^*

定义 2（泛化(Generalization)） 对关系 $R(A_1,A_2,\cdots,A_k)$，设属性 A_i 的域为 D，D 上的一个划分为 $\{u_1,u_2,\cdots,u_L\}$，其中 $u_i(1\leqslant i\leqslant L)$ 为一个整数区间。如果对任意元组 $t\in R$，属性 A_i 存在如下函数 $g:t[A_i]\rightarrow u_i$，其中 $t[A_i]\in u_i$，则 g 称为属性 A_i 的泛化函数，$g(t[A_i])$ 称为 $t[A_i]$ 的泛化操作。

同理，R 在属性集 $\{A_1,A_2,\cdots,A_k\}$ 上的泛化操作，则指 R 在每个属性上分别进行泛化所得到的结果，即 $g(t[A_1,A_2,\cdots,A_k])=(g(t[A_1]),g(t[A_2]),\cdots,g(t[A_k]))$。

注：定义 2 对泛化值同样适用，对泛化值的泛化操作是从当前泛化值到包含该泛化值的更大范围的泛化值的映射过程。

例 2 设属性 Age 的域为[21-60]，属性 Age 域的第一次划分为{[21-25],[26-30],…,[56-60]}，第二次划分为{[21-30],[31-40],…,[51-60]}，表 1(a)中数据集 R 在属性 Age 上的泛化操作为：属性值 21,23 分别泛化为[21-25]，属性值 52,53 分别泛化为[51-55]。属性值 48,42,49 则经过了两次泛化，第一次分别泛化为[46-50]，[41-45]，[46-50]，第二次分别泛化为[41-50]。泛化结果如表 1(b)所列。

定义 3（k-匿名数据集） 对数据集 $R(AQI,A^S)$，如果对 R 在 A^{QI} 上的值进行泛化得到数据集 R^*，且 R^* 在 A^{QI} 上满足 k-匿名约束，则 R 到 R^* 的泛化过程称为数据集 R 的 k-匿名化，数据集 R^* 称为 R 的 k-匿名数据集。

例 3 对数据集 R(表 1(a))，R^*(表 1(b))是对 R 中准标识符 $\{Age,Zip\}$ 上数据进行泛化所得到的数据集。由例 1 知，R^* 满足 2-匿名约束，因此从 R 到 R^* 的泛化过程就是数据集 R 的 2-匿名化，R^* 即为 R 的 2-匿名数据集。

为了区分 R 和 R^* 中元组，下面将 R 中元组称为元组，而将 R^* 中元组称为泛化元组。

定义 4（元组—泛化元组映射） 设原始数据集及其对应的 k-匿名数据集分别为 R 和 R^*，对任意元组 $t\in R$，如果存在 $t^*\in R^*$，使 $t[A_i^{QI}]\in t^*[A_i^{QI}](1\leqslant i\leqslant n)$，$t[A^S]=t^*[A^S]$，则

t^* 称为 t 的泛化元组。函数 $gf: t \to t^*$ 称为元组 t 到其泛化元组 t^* 的一对一映射函数。

例 4　对表 1(a)、表 1(b)所示的数据集 R 和 2-匿名数据集 R^*，R^* 中 $t1^*$ 是 R 中元组 $t1$ 的泛化元组，同理 $t2^*$ 是 $t2$ 的泛化元组，…，$t7^*$ 是 $t7$ 的泛化元组。

根据元组和泛化元组的一一对应关系，我们可以为 R 和 R^* 中元组建立索引表。

定义 5（QI 组）　对 k-匿名数据集 $R^*(A^{QI}, A^S)$，$R^*[A^{QI}]$ 中具有相同值的一组元组称为一个 QI 组，记为 QG。

将 R^* 所包含的 QI 组集合记为 $QG(R^*)=\{QG_1, QG_2, \cdots, QG_m\}$，其中 $|QG_i| \geqslant k$（根据 k-匿名约束定义），$QG_i \cap QG_j = \emptyset (1 \leqslant i, j \leqslant m, i \neq j)$ 且 $|QG_1| + |QG_2| + \cdots + |QG_m| = |R^*|$。

例 5　在 2-匿名数据集 R^*（表 1(b)）中，由于 $t_1[Age, Zip] = ([21, 25], [11k, 20k])$，$t_2[Age, Zip] = ([21, 25], [11k, 20k])$，因此元组 t_1 和 t_2 为一个 QI 组；同理，t_3, t_4, t_5 为一个 QI 组，t_6, t_7 为一个 QI 组。则 $QG(R^*) = \{QG_1 = \{t_1, t_2\}, QG_2 = \{t_3, t_4, t_5\}, QG_3 = \{t_6, t_7\}\}$。

对表 R 的插入、删除、修改 3 种更新操作分别表示如下。

$INSERT(R, T)$：向表 R 中插入元组集 $T = \{t_1, t_2, \cdots, t_k\}$，其中 $t_i (1 \leqslant i \leqslant k)$ 是作用在属性集 $\{A^{QI}, A^S\}$ 上的元组。

$DELETE(R, \varphi_D)$：删除表 R 中满足条件 φ_D 的元组。

$MODIFY(R, \varphi_M, F_M)$：将表 R 中满足条件 φ_M 的元组根据修改表达式 F_M 进行修改。

需要说明的是，φ_D 和 φ_M 是定义在属性集 $\{A^{QI}, A^S\}$ 上的布尔表达式集合，其一般形式是 $\varphi = \varphi_1 \wedge \varphi_2 \wedge \cdots \wedge \varphi_m$，其中 φ_i 是一个形如 $(x \theta y + c)$ 或 $(x \theta y)$ 的原子条件，x 或 y 代表属性变量，c 是一个常量，$\theta \in \{=, <, \leqslant, >, \geqslant\}$。修改表达式 F_M 是一个形如 $A = f(A_1, A_2, \cdots, A_k)$ 的表达式，其中 $A, A_1, A_2, \cdots, A_k$ 是 R 中的属性，f 是作用在 $A_1, A_2, \cdots, A_k$ 上的计算函数。下面用 $\alpha(\varphi)$ 表示表达式 φ 中所包含的变量。

4　k-匿名数据集的增量更新

由于原始数据集是一个关系表，因此当原始数据集插入、删除、修改元组时，k-匿名数据集也应插入、删除、修改相应的元组。但是，由于 k-匿名数据集中包含泛化值，必须考虑如何让泛化元组响应更新操作，且要考虑 k-匿名约束的维持问题，即每个 QI 组更新后都不能小于 k。

本节将分别研究 k-匿名数据集的插入、删除、修改 3 种更新操作。对插入操作，先根据插入元组与各 QI 组的语义贴近度确定要插入的 QI 组，再将其转化成泛化元组。对删除操作，根据元组与泛化元组的映射关系确定待删除元组在 k-匿名数据集中的位置，再进行删除；删除操作可能导致 QI 组变小，为了保证数据集的 k-匿名约束性质，对小于 k 的 QI 组，选择与它语义贴近度最大的 QI 组进行合并。对修改操作，则分解为删除和插入两步操作来完成。

在给出具体的更新算法之前，首先介绍元组-QI 组语义贴近度和 QI 组语义贴近度的定义。

定义 6（元组-QI 组语义贴近度，Tuple-QG Semantic Similarity Degree）　设原始数据集及其 k-匿名数据集分别为 $R(A^{QI}, A^S)$ 和 $R^*(A^{QI}, A^S)$，任意元组 $t \in R$ 与 R^* 中 QI 组 QG_j 的语义贴近度记为 $T\text{-}QGSSD(t, QG_j)$，用公式可表示为

$$T\text{-}QGSSD(t, QG_j) = \sum_{i=1}^{n} SSD(A_i^{QI})$$

其中，$SSD(A_i^{QI})$ 指元组 t 和 QG_j 在属性 A_i^{QI} 上的语义贴近度。令 $t[A_i^{QI}] = a$，QG_j 在属性 A_i^{QI} 上的值为一个区间 $[b, c]$，则 $SSD(A_i^{QI})$ 可表示为

$$SSD(A_i^{QI}) = \begin{cases} 1, & a \in [b, c] \\ \dfrac{(c-b)/2}{|a-(c+b)/2|}, & a \notin [b, c] \end{cases}$$

例 6　对 R（表 1(a)）中元组 $t_1 = (21, 12000, flu)$，t_1 与 R^*（表 1(b)）中 QI 组 QG_1 的语义贴近度计算如下：由于 $21 \in [21, 25]$，因此 $SSD(Age) = 1$；由于 $12000 \in [11k, 22k]$，因此 $SSD(Zip) = 1$；即 $T\text{-}QGSSD(t_1, QG_1) = 2$。$t_1$ 与 R^* 中 QG_2 的语义贴近度计算如下：由于 $21 \notin [41, 50]$，因此 $SSD(Age) = \dfrac{(50-41)/2}{|21-(50+41)/2|} = 0.19$；由于 $12000 \notin [11k, 22k]$，因此 $SSD(Zip) = \dfrac{(30-21)/2}{|12-(30+21)/2|} = 0.33$；即 $T\text{-}QGSSD(t_1, QG_2) = 0.19 + 0.33 = 0.52$。

如果 $T\text{-}QGSSD(t, QG_j) = n$，即 $SSD(A_i^{QI}) = 1 (1 \leqslant i \leqslant n)$，则表示 $t[A_i^{QI}] \in t^*[A_i^{QI}] (1 \leqslant i \leqslant n, t^*$ 为 QG_j 中任一元组）。

定义 7（QI 组语义贴近度，QG-QG Semantic Similarity Degree）　对 k-匿名数据集 $R^*(A^{QI}, A^S)$，R^* 中任意两个 QI 组 QG_i 与 QG_j 的语义贴近度记为 $QGSSD(QG_i, QG_j)$，用公式可表示为

$$QGSSD(QG_i, QG_j) = \sum_{i=1}^{n} QGSSD(A_i^{QI})$$

其中，$QGSSD(A_i^{QI})$ 指 QG_i 与 QG_j 在属性 A_i^{QI} 上的语义贴近度。令 QG_i 在属性 A_i^{QI} 上的值为一个区间 $[b_1, c_1]$，QG_j 在属性 A_i^{QI} 上的值为一个区间 $[b_2, c_2]$，则 $QGSSD(A_i^{QI})$ 可表示为

$$QGSSD(A_i^{QI}) = \frac{1}{|(b_1 + c_1)/2 - (b_2 + c_2)/2|}$$

例 7　对 2-匿名数据集 R^*（表 1(b)），QG_1 与 QG_2 的语义贴近度计算如下：$QGSSD(Age) = \dfrac{1}{|(21+41)/2-(50+25)/2|} = 0.15$；$QGSSD(Zip) = \dfrac{1}{|(11+21)/2-(20+30)/2|} = 0.11$；即 $QGSSD(QG_1, QG_2) = 0.15 + 0.11 = 0.26$。

4.1　插入操作

当执行插入操作 $INSERT(R, T)$ 时，k-匿名数据集 R^* 也应插入相应的元组。为了使更新后的 R^* 满足 k-匿名约束，对每条元组 $t \in T$，选择与 t 语义贴近度最大的 QI 组，转化成相应的泛化元组进行插入。如果 t 与某 QI 组 QG_i 的最大语义贴近度为 n，则 t 的泛化元组 t^* 取值如下：t^* 在 A^{QI} 上的值与 QG_i 中其他元组相同，t^* 在 A^S 上的值与 t 相同。如果 t 与某 QI 组 QG_i 的语义贴近度最大且不为 n，则 t 的泛化元组 t^* 取值如下：t^* 在 A^{QI} 上的值为 QG_i 与 t 相应属性值的泛化值，t^* 在 A^S 上的值与 t 相同。另外，当 QG_i 插入多个元组后，QG_i 可能过大（如等于 $2k$），此时可以按照一定原则，将此 QG_i 拆分成两个大小为 k 的 QI 组。

k-匿名数据集的插入操作方法如下。对任意元组 $t \in T$ 执行如下操作：1）计算 t 与每个 QI 组的语义贴近度。如果 t 与 QG_i 的语义贴近度为 n，则在 QI 组中直接添加一条元组 t^*，其中 $t^*[A^{QI}] = t^{*\prime}[A^{QI}] (t^{*\prime} \in QG_i)$，$t^*[A^S] = t[A^S]$；否

则，选择 R^* 中与 t 语义贴近度最大的 QI 组 QG_i，对任意元组 $t^{*\prime}\in QG_i$ 和 t 的泛化元组 t^*，使 $t^*[A^{QI}]=g(t[A^{QI}])=g(t^{*\prime}[A^{QI}])$，$t^*[A^S]=t[A^S]$。2)如果 $|QG_i|$ 等于 $2k$，则进行拆分。

算法描述如下：

$INS(R(A^{QI},A^S),T,R^*(A^{QI},A^S),INDT)$

输入：原始数据集 $R(A^{QI},A^S)$，及插入的元组 T，R 对应的 k-匿名数据集 $R^*(A^{QI},A^S)$，$INDT$ 为 R 与 R^* 的索引表

输出：R 插入更新后所对应的 k-匿名数据集 R^*

变量初始化：$i=0$；$ssd=0$；$ssdmax=0$；

1. for each $t\in T$

1.1 {for $i=1$ to m

{$ssd\leftarrow T\text{-}QGSSD(t,QG_i)$；

if $ssd>ssdmax$ then $ssdmax=ssd$；}

$QG\leftarrow$取与 t 语义贴近度为 $ssdmax$ 的 QI 组；

if $ssdmax=n$ then /* $|A^{QI}|=n^*$ /

{$t^{*\prime}\leftarrow$取 QG 中任意一条元组；

$R^*\leftarrow R^*\cup\{t^*\mid t^*[A_j^{QI}]=t^{*\prime}[A_j^{QI}],t^*[A^S]=t[A^S]\}$；

更新索引表 $INDT$；}

else

{$t^{*\prime}\leftarrow$取 QG_{ssdq} 中任意一条元组；

$R^*\leftarrow R^*\cup\{t^*\mid t^*[A_j^{QI}]=t^{*\prime}[A_j^{QI}]$与 $t[A_j^{QI}]$的泛化值，$t^*[A^S]=t[A^S]\}$；

for each $t^{*\prime}\in QG$

$t^{*\prime}[A_j^{QI}]=t^*[A_j^{QI}]$；

更新索引表 INDT；}

1.2 if $|QG|=2k$ then

{根据 $INDT$ 得到初始元组；

将 QG 拆分成 2 个大小为 k 的 QI 组；

更新索引表 $INDT$；}}

2. return (R^*)；

例 8 当 R(表 1(a))执行插入操作 $INSERT(R,\{(24,17000,\text{insomnia}),(55,62000,\text{insomnia})\})$时，$R^*$(表 1(b))的插入操作如下：首先对元组 $t=(24,17000,\text{insomnia})$进行插入，由于 $T\text{-}QGSSD(t,QG_1)=2$ 最大，令 $t^*=([21,25],[11k,20k],insomnia)$，$QG_1$ 插入后对应表 2 中 $t3^*$。对元组 $t=(55,62000,\text{insomnia})$，计算出 t 和各个 QI 组的语义贴近度为 $T\text{-}QGSSD(t,QG_1)=0.16$，$T\text{-}QGSSD(t,QG_2)=0.59$，$T\text{-}QGSSD(t,QG_3)=1.69$，由于 t 和 QG_3 的语义贴近度最大，因此 t^* 应插入 QG_3。由于 $55\in[51,55]$，则 $t^*[Age]=[51,55]$；由于 $62000\notin[51k,60k]$，因此将它们泛化为 $[51k,65k]$，即 $t^*[Zip]=[51k,65k]$，$t^*[Problem]=$insomnia 插入后对应表 2 中 $t9^*$。另外，$t5^*[Zip]$，$t6^*[Zip]$也应改为 $[51k,65k]$，修改后分别对应表 2 中 $t7^*$，$t8^*$。

表 2 R^* 的插入增量更新

Tuple ID	QG	Age	Zip	Problem
t1*	1	[21, 25]	[11k, 20k]	flu
t2*	1	[21, 25]	[11k, 20k]	gastritis
t3*	1	[21, 25]	[11k, 20k]	insomnia
t4*	2	[41, 50]	[21k, 30k]	flu
t5*	2	[41, 50]	[21k, 30k]	gastritis
t6*	2	[41, 50]	[21k, 30k]	insomnia
t7*	3	[51, 55]	[51k, 65k]	flu
t8*	3	[51, 55]	[51k, 65k]	gastritis
t9*	3	[51, 55]	[51k, 65k]	insomnia

4.2 删除操作

当执行删除操作 $DELETE(R,\varphi_D)$时，k-匿名数据集 R^* 也应删除相应的元组。由于 R^* 中准标识符上包含泛化值，敏感属性上是精确值，R^* 的删除操作有下列两种情况：①如果 φ_D 中只包含敏感属性 A^S，则 R^* 可以对删除条件直接进行判断，即 R^* 可以直接执行删除操作；②如果 φ_D 中包含准标识符中属性，则需要先从 R 中找出满足删除条件的元组，再映射为 R^* 中的泛化元组删除。另外，删除操作可能导致某些 QI 组小于 k。为了保证更新后 R^* 的 k-匿名约束性质，应对每个 QI 组进行检查，并对小于 k 的 QI 组进行泛化合并。

k-匿名数据集 R^* 的删除方法如下：①如果 φ_D 中只包含 A^S，则将 R^* 中使 φ_D 成立的元组直接删除；否则，找出 R 中满足 φ_D 的元组集 T，对每个元组 $t\in T$，将 t 映射为泛化元组 t^*，在 R^* 中将 t^* 删除。②检查 R^* 中各个 QI 组，如果存在 QG_i 小于 k，则选择与 QG_i 语义贴近度最大的另一个 QI 组 QG_j，将 QG_i 和 QG_j 泛化合并为一个 QI 组。

算法描述如下：

$DEL(R(A^{QI},A^S),\varphi_D,R^*(A^{QI},A^S),INDT)$

输入：表 $R(A^{QI},A^S)$及 R 的删除条件 φ_D，$R^*(A^{QI},A^S)$为 R 对应的 k-匿名数据集，$INDT$ 为 R 与 R^* 的索引表

输出：表 R 执行删除操作后对应的 k-匿名数据集 R^*

变量初始化：$ssd=0$；$ssdq=0$；

1. if $\alpha(\varphi_D)=A^S$ then

{$R^*\leftarrow R^*-$ {满足 φ_D 的所有元组}；

更新索引表 $INDT$；}

else

/* 从原始数据集中先找出要删除的元组，再到 R^* 中删除对应的元组 */

{$T\leftarrow\varphi_D$ 作用在 R 上得到的元组集；

for each $t\in T$ /* 删除 R^* 中 t 所对应的泛化元组 */

{$t^*\leftarrow$fg(t)；

$R^*\leftarrow R^*-\{t^*\}$；

更新索引表 INDT；}}

2. /* 检查 R^* 中所有的 QI 组，如果有小于 k 的 QI 组，则选择语义贴进度最大的 QI 组进行合并泛化，使得每个 QI 组的大小至少为 k；*/

for each $QG_i\in R^*$

if $|QG_i|<k$ then

{for each $QG_j\in R^*$ and $QG_j\neq QG_i$

if $QGSSD(QG_i,QG_j)>ssd$ then

{$ssd\leftarrow QGSSD(QG_i,QG_j)$；$ssdq=j$；}

将 QG_i 与 QG_{ssdq} 进行泛化合并，形成一个 QI 组；

更新索引表 $INDT$；}

3. return(R^*)；

例 9 当 R(表 1(a))要执行删除操作 $DELETE(R,(Problem=\text{“insomnia”}))$时，$R^*$(表 1(b))的删除操作如下：由于 R^* 能对删除条件进行判断，$t5^*$ 可以从 R^* 中直接删除，更新后的 R^* 如表 3(a)所列。当 R 要执行的删除操作为 $DELETE(R,(Age<25\wedge Zip>15000))$时，对 R^* 的删除操作如下：由于 φ_D 中包含准标识符中属性，因此将 $\varphi_D=(Age<25\wedge Zip>15000)$作用在 R 上，得到元组集 $T=\{(23,18000,\text{gastritis})\}$，由于元组 $t=(23,18000,\text{gastritis})$对应 R^* 中的元组为 $t2^*=([21,25],[11k,20k],\text{gastritis})$，因此将 $t2^*$ 删除。由于此时 QG_1 中元组个数小于 2，因此需要将此 QI 组与其他 QI 组进行合并，分别计算 QG_1 与 QG_2 和 QG_3 的语义贴近度 $QIGSSD(QG_1,QG_2)=0.14$，$QIGSSD(QG_1,QG_3)=0.06$。可知 QG_1 与 QG_2 的语义最接近，因此将 QG_1 与 QG_2 进行泛化，合并为同一个 QI 组，删除更新后的 R^* 如表 3(b)所列。

4.3 修改操作

当执行修改操作 $MODIFY(R,\varphi_M,F_M)$时，k-匿名数据集 R^* 也应对相应的元组进行修改。由于 R^* 中准标识符上包含泛化值，敏感属性上是精确值，对 R^* 的修改操作有下列两种情况：①如果 φ_M 和 F_M 中只包含 A^S，在 R^* 中直接修改元组。②如果 φ_M 或 F_M 包含准标识符中属性，则 R^* 的修改操作可以分解成删除和插入两步操作：对 R 中每个满足修改条件 φ_M 的元组 t，先在 R^* 中删除其对应的泛化元组，再向 R^* 中插入修改后的元组 t'。上述过程可能包括一种特殊情况：元组 t 和修改后的元组 t'所对应的泛化元组不变，此时 k-匿名数据集不需要改变。另外，修改过程中也包含了删除操作，因此修改完成后也应对每个 QI 组进行检查并对小于 k 的 QI 组进行合并处理。

表 3　R^* 的删除增量更新

Tuple ID	QG	Age	Zip	Problem
t1*	1	[21, 25]	[11k, 20k]	flu
t2*	1	[21, 25]	[11k, 20k]	gastritis
t3*	2	[41, 50]	[21k, 30k]	flu
t4*	2	[41, 50]	[21k, 30k]	gastritis
t6*	3	[51, 55]	[51k, 60k]	flu
t7*	3	[51, 55]	[51k, 60k]	gastritis

(a)

Tuple ID	QG	Age	Zip	Problem
t1*	1	[21, 50]	[11k, 30k]	flu
t2*	1	[21, 50]	[11k, 30k]	flu
t3*	1	[21, 50]	[11k, 30k]	gastritis
t4*	1	[21, 50]	[11k, 30k]	insomnia
t5*	2	[51, 55]	[51k, 60k]	flu
t6*	2	[51, 55]	[51k, 60k]	gastritis

(b)

k-匿名数据集 R^* 的修改操作主要包括下面两个步骤：①如果 φ_M 和 F_M 中只包含 A^S，则直接修改 R^* 中元组；否则，②找出 R 中满足 φ_M 的元组集 T，对每个元组 $t\in T$，如果 t 修改前后对应同一个泛化元组，则 R^* 不发生改变；否则，先删除 t 在 R^* 中的泛化元组，然后调用 INS 过程，将修改后的元组 t' 插入到 R^* 中；③对 R^* 中小于 k 的 QI 组进行合并处理，使每个 QI 组都大于等于 k。

算法描述如下：

$MOD(R(A^{QI},A^S),\varphi_M,F_M,R^*(A^{QI},A^S),INDT)$

输入：表 $R(A^{QI},A^S)$为原始数据集，$R^*(A^{QI},A^S)$为表 $R(A^{QI},A^S)$的 k-匿名数据集，φ_M 和 F_M 分别为表 R 的修改条件和修改表达式，$INDT$ 为 R 与 R^* 的索引表

输出：表 R 修改后对应的 k-匿名数据集 R^*

变量初始化：$ssd=0$；$ssdq=0$；

1. if $\alpha(\varphi_M)=A^S$ and $\alpha(F_M)=A^S$ then
   {R^*←按照修改表达式 F_M 修改 R^* 中满足 φ_M 的元组；
   更新索引表 $INDT$；}
   else
2. /* 将 R^* 的修改操作分解为删除旧的(修改前的)元组和插入新的(修改后的)元组两步操作 */
   T←φ_M 作用在 R 上得到的元组集；
   /* 对每个元组 $t\in T$ 执行修改操作 */
   for each $t\in T$
   {t^*←$fg(t)$；
   t'←t 按照表达式 F_M 进行修改；
   if $g(t'[A^{QI}])=t^*[A^{QI}]$ and $t'[A^s]=t^*[A^s]$, then
   更新索引表 $INDT$；/* t 修改前后对应同一个泛化元组，此时 k-匿名数据集无需改变 */
   else
   {R^*←$R^*-\{t^*\}$；/* 删除 R^* 中修改前的元组 */
   更新索引表 $INDT$；
   /* 在 R^* 中插入修改后的元组 */
   R^*←$INS(R(A^{QI},A^S),R^*,t',INDT)$；}}
3. /* 检查 R^* 中 QI 组，对小于 k 的 QI 组，选择语义贴进度最大的 QI 组进行合并泛化，直到 QI 组大于等于 k */
   for each $QG_i\in R^*$
   if $|QG_i|<k$ then
   {for each $QG_j\in R^*$ and $QG_j\neq QG_i$}
   if $QGSSD(QG_i,QG_j)>ssd$ then
   {ssd←$QGSSD(QG_i,QG_j)$；$ssdq=j$；}
   将 QG_i 与 QG_{ssdq} 进行泛化合并，形成一个 QI 组；
   更新索引表 $INDT$；}
4. return (R^*)；

例 10　当 R(表 1(a))要执行修改操作 $MODIFY(R,Age\leqslant 25,Age=Age+10)$时，$R^*$(表 1(b))的修改操作如下：首先将 $Age\leqslant 25$ 作用于 R，得到元组集 $T=\{(21,12000,flu),(23,18000,gastritis)\}$。对元组(21,12000,flu)，根据修改表达式修改后的元组为(31,12000,flu)。由于修改前后对应的泛化元组不同，先将(21,12000,flu)的泛化元组([21-25],[11k,20k],flu)在 R^* 中删除，然后调用 INS 过程，将(31,12000,flu)插入到 R^* 的 QG_1 中。根据 INS 过程，此时 QG_1 中元组如表 4 所列。对元组(23,18000,gastritis)，根据修改表达式修改后的元组为(33,12000,flu)，由于修改前后都对应泛化元组([31-35],[11k,20k],gastritis)，R^* 不再变化。

表 4　R^* 的修改增量更新

Tuple ID	QG	Age	Zip	Problem
t1*	1	[31, 35]	[11k, 20k]	flu
t2*	1	[31, 35]	[11k, 20k]	gastritis
t3*	2	[41, 50]	[21k, 30k]	flu
t4*	2	[41, 50]	[21k, 30k]	gastritis
t5*	2	[41, 50]	[21k, 30k]	insomnia
t6*	3	[51, 55]	[51k, 65k]	flu
t7*	3	[51, 55]	[51k, 65k]	gastritis

结束语　本文给出了原始数据集为关系表场景下的 k-匿名数据集的增量更新算法。在实际应用中，原始数据集可能为由一个或多个基本关系表导出的视图，此时，k-匿名数据集的增量更新情况会更加复杂，k-匿名数据集可能出现不同的更新状态。后期工作将主要对此进行研究。

参 考 文 献

[1] Sweeney L. K-Anonymity: a model for protecting privacy[J]. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2002, 10(5): 557-570

[2] Xiao Xiaokui, Tao Yufei. Dynamic Anonymization: Accurate Statistical Analysis with Privacy Preservation[C]// Proceedings of the ACM SIGMOD. New York: ACM, 2008: 107-120

[3] Sweeney L. Guaranteeing anonymity when sharing medical data, the Datafly system[C]// Proceedings Journal of the American Medical Informatics Association. Washington, DC: Hanley&Belfus, Inc, 1997: 51-55

(下转第 170 页)

其中，*card* 表示近似集的基数，即所包含的等价类的个数。对于图 2 所示的情形，$r_R(s)=(8-5)/8=0.38$。

论域 U 所包含的等价类 R_i 越多（即已有知识越细化），边界域 $bn_R(s)$ 的比重就会减小，粗糙度 $r_R(s)$ 也就降低了。图 5 是将图 2 的等价类增大至 4 倍的情形，粗糙度 $r_R(s)=(21-18)/21=0.14$，较先前下降了许多。

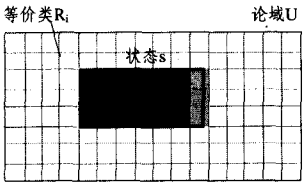

图 5　已有知识的丰富可以降低粗糙性

图 2 和图 5 的分析说明，粗糙性并不是客观事物本身具有的属性，而是人类对客观事物认知的“不精确性”程度。与其说是“事物的粗糙性”，不如说是“认知的粗糙性”。粗糙性的程度与认知主体已经拥有的知识存量直接相关。例如，同一个客观事物，对一位科学家而言的粗糙性就要比对一位农民而言的粗糙性小得多。可见，包含粗糙性的不精确性依赖于认知主体的已有知识，呈现一种反向变化的关联。

结束语　本文将不精确性合称模糊性与粗糙性，以便从人的认知活动和知识管理的视角对其深入考察，进而推进对“不精确性本体”、“不精确性推理”等的研究。本文研究的主要结论如下。

（1）不精确性源于人们对不确定性现象的认知活动。人们为了更好地认知不确定性现象，采取了两项有力措施：一方面是简化认知对象，将连续发展的客观事物人为地划分为若干类型。为了继续保持连续性，分类采用模糊子集的方式，即模糊子集之间的边界相互交迭，以致于某个具体事物同时隶属于多个模糊子集，模糊性由此而来。另一方面是利用已有知识，人类总是用已有知识来认知新事物的，新事物之“新”就在于该事物并不能由已有知识完全解释，可行的方案是采用与该新事物相毗邻的两个已有知识的集合（即下近似集和上近似集）来近似地说明，粗糙性由此而来。不精确性是模糊性与粗糙性之合称。

（2）减低不精确性的最好途径是有效的知识管理。不精确性的影响是负面的，对其的抑制无非是从模糊性和粗糙性两方面入手。模糊性就是由于分类而导致的“亦此亦彼性”。分类越细，越有利于判别对象隶属于何类而减小模糊性。然而，模糊性对应于隶属度函数，后者又完全由主观确定，分类越细小，需要确定的隶属度函数就越多，众人就更难取得一致，使原本模糊的事情更加模糊。模糊性的减少是困难的。粗糙性的情景就要乐观得多。粗糙性就是认知对象与认知主体（人或计算机系统）的已有知识的不吻合程度，通过有效的知识管理增加已有知识的存量、改善已有知识的有序性，就可减低粗糙性进而影响不精确性。

（3）不精确性的测度尚待研究。属性能够定量表达才有应用价值。模糊性采用可信性测度，并由此导出隶属度函数；粗糙性采用信赖性测度，并由此导出粗糙度函数[12]。由模糊性和粗糙性构成的不精确性应该采用怎样的测度？“不精确度函数”又应该如何定义？对此需要进一步探讨。

参 考 文 献

[1] Klinov P, Taylor J M, Mazlack L J. Interval rough mereology and description logic: an approach to formal treatment of imprecision in the Semantic Web ontologies[J]. Web Intelligence and Agent Systems, 2008(6): 157-174

[2] Klinov P, Mazlack L J. Fuzzy rough approach to handling imprecision in Semantic Web ontologies[EB/OL]. http://www.ece.uc.edu/~klinov p/doc/conferences/2006/nafips.2006.pdf

[3] Klinov P, Taylor J M, Mazlack L J. Formal treatment of imprecision in the Semantic Web ontologies[EB/OL]. http://www.ece.uc.edu/~klinov p/doc/proposals/nsf_apprx_onto.2006.pdf

[4] Zadeh L. Fuzzy sets[J]. Information and Control, 1965, 8(3): 338-353

[5] Pawlak Z. Rough sets[J]. International Journal of Computer and Information Sciences, 1982, 11(5): 341-356

[6] 张文修，吴伟志，梁吉业，等. 粗糙集理论与方法[M]. 北京：科学出版社，2001

[7] 李德毅，杜鹢. 不确定性人工智能[M]. 北京：国防工业出版社，2005

[8] 柳廷廷. 科学世界图景中的不确定性[J]. 哲学研究，1993(5): 58-64

[9] 鲁鹏. 论不确定性[J]. 哲学研究，2006(3): 3-10

[10] 罗承忠. 模糊集引论（上册）[M]. 北京：北京师范大学出版社，2007

[11] 刘清. Rough 集与 Rough 推理[M]. 北京：科学出版社，2001

[12] 刘宝碇，彭锦. 不确定性理论教程[M]. 北京：清华大学出版社，2005

（上接第 150 页）

[4] Hundepool A, Willenborg L. μ-and τ-argus: software for statistical disclosure control[C]// Proceedings of Third International Seminar on Statistical Confidentiality. 1996

[5] Sweeney L. Achieving k-anonymity privacy protection using generalization and suppression[J]. Int'l Journal on Uncertainty, Fuzziness and Knowledge-Based Systems, 2002, 10(5): 571-588

[6] Meyerson A, Williams R. On the complexity of optimal k-anonymity[C]// Proceedings of PODS 2004. New York: ACM, 2004, 6: 223-228

[7] Aggarwal G, Feder T, Kenthapadi K, et al. k-Anonymity: Algorithms and Hardness[R]. Stanford University, 2004

[8] Lefvre K, DeWitt D, Ramakrishnan R. Incognito: Efficient full-domain k-anonymity[C]// Proceedings of the ACM SIGMOD 2005. New York: ACM, 2005: 49-60

[9] LeFevre K, DeWitt D J, Ramakrishnan R. Mondrian Multidimensional K-Anonymity[C]// Proceedings of ICDE 2006. Washington, DC: IEEE Computer Society, 2006

[10] Bayardo R, Agrawal R. Data privacy through optimal k-anonymization[C]// Proceedings of ICDE 2005. Washington, DC: IEEE Computer Society, 2005: 217-228

[11] 刘向宇，杨晓春，于戈. 一种基于特征类的高精度隐私保护数据发布方法[J]. 计算机科学，2005, 32(7. 增 A): 368-373

[12] 杨晓春，刘向宇，王斌，等. 支持多约束的 K-匿名化方法[J]. 软件学报，2006, 17(5): 1222-1231

[13] Machanavajjhala A, Gehrke J, Kifer D. l-diversity: Privacy beyond k-anonymity[C]// Proceedings of ICDE 2006. Washington, DC: IEEE Computer Society, 2006: 1-12

[14] Xiao Xiaokui, Tao Yufei. Personalized Privacy Preservation[C]// Proceedings of the ACM SIGMOD 2006. New York: ACM, 2006: 229-240

[15] Hyoungmin P, Kyuseok S. Approximate Algorithms for k-Anonymity[C]// Proceedings of the ACM SIGMOD 2007. New York: ACM, 2007: 67-78