第 20 卷 第 10 期　　控 制 与 决 策　　2005 年 10 月
Vol. 20 No. 10　　Control and Decision　　Oct. 2005

文章编号: 1001-0920(2005)10-1120-05

基于连接属性的元规则实例方法

朱恒民, 刘建国, 王宁生
(南京航空航天大学 CIMS 工程中心, 南京 210016)

摘　要: 针对元规则的实例是采用元规则制导数据挖掘的一个关键问题, 讨论了当前元规则的实例方法, 指出元规则中相同属性变元的实例不仅受关系中属性的数据类型约束, 而且受属性值集的约束. 提出了关系表的连接属性概念以及相关理论, 在此基础上设计了一种元规则的实例方法, 并通过实例验证了它的有效性. 该方法能够大大削减元规则的候选实例集.

关键词: 元规则; 数据挖掘; 连接属性; 元查询; 元模式
中图分类号: TP311　　**文献标识码**: A

Method to Instantiate Meta Rule Based on Connected Attributes

ZHU Heng-min, LIU Jian-guo, WANG Ning-sheng
(Research Center of CIMS Engineering, Nanjing University of Aeronautics and Astronautics, Nanjing 210016, China. Correspondent: ZHU Heng-min, Email: zhuhengmin@sina.com)

Abstract: To the meta rule instantiation of meta-rule-guided data mining, the current methods to instantiate meta rule are discussed. It is pointed out that, besides data type of attribute, the set of attribute value restricts the instantiation of the same attribute variable of meta rule. The notion of connected attributes between relation tables and relative theories are proposed, based on which a method of instan meta rule is given. An example shows that the proposed method is effective and can greatly reduce the size of meta rule candidate instantiation set.

Key words: Meta rule; Data mining; Connected attributes; Metaquery; Meta pattern

1 引　　言

数据挖掘(DM)亦称数据库中的知识发现(KDD), 是当今人工智能和数据库领域中非常具有活力的研究课题, 其目标是从大量数据中发现有趣的模式. 数据挖掘技术虽然日臻成熟, 然而从数据挖掘技术在各领域的一些应用中发现, 许多挖掘方法都产生了大量规则, 虽然其中不乏少量规则是有趣的, 但靠用户来发现有趣规则几乎是不可能的. 因此, 如何让挖掘方法产生用户感兴趣的规则是一个重要的研究课题.

Shen 等[1]提出采用元查询来提高发现规则的有趣性. 元查询, 又称元规则或元模式, 是一个重要的数据挖掘技术. 它通过指定待发现规则的期望形式来挖掘用户感兴趣的规则. 采用元规则有助于提高挖掘过程的效率, 而且能发现多个关系表之间蕴涵的规则.

元规则的实例是指产生与元规则具有相同形式的发现规则, 它是采用元规则来制导数据挖掘的关键技术. 目前, 国内外已有一些关于元规则实例方法的研究. 文献[2, 3]采用形如 $P_1(t: \mathrm{rel}, x_1) \wedge \cdots \wedge P_m(t, x_m) \rightarrow Q(t, y)$ 的这类元规则制导发现关联规则. 但该类元规则形式特殊, 公共变量 t 被指定为关系 rel 的主键, 而且这类元规则只用来发现关联规则, 因此文献中介绍的实例方法不具有通用性. 文献[4]提出了一种基于 CSP(Constraint Satisfaction Problem)的元规则实例方法. 该方法将相同属性变元绑定到相同的数据类型作为约束之一, 为每个谓词构造约束, 用满足约束的关系来逐个实例谓词.

收稿日期: 2004-11-05; **修回日期**: 2004-12-30.
基金项目: 国家科技部项目(2002ED691036).
作者简介: 朱恒民(1974-), 男, 南京人, 博士生, 从事数据挖掘、数据库技术等研究; 王宁生(1936-), 男, 南京人, 教授, 博士生导师, 从事 FMS, CIMS 和 ERP 等研究.

本文认为, 元规则中相同属性变元的实例, 不仅受关系中属性的数据类型约束, 而且受属性值集的约束. 例如属性"分数"和"年龄"都是整数型, 但显然不能用它们去实例元规则中相同的属性变元. 为此, 本文提出一种基于连接属性的元规则实例方法.

2　元规则基本概念

定义 1　假设 $L_1, \cdots, L_m, T$ 都是形如 $Q(Y_1, \cdots, Y_n)$ 的文字模式, 其中 Q 为谓词变量或特定关系名, Y_i 为变元(Y_i 要实例为关系中的属性, 文中在易混淆处称之为属性变元), 则元规则是具有如下形式的规则模板:

$$L_1 \wedge \cdots \wedge L_m \rightarrow T. \tag{1}$$

例如, $\mathrm{major}(s, X) \wedge Q(s, Y) \rightarrow R(s, Z)$ 和 $P(X, Y) \wedge Q(Y, Z) \rightarrow R(X, Z)$ 都是元规则. 元规则中已被实例的文字称为原子.

定义 2　R 为原子, $\mathrm{att}(R)$ 是原子 R 的所有属性变元组成的集合, J 是元规则 r 的前件中所有原子的自然连接, T 是 r 的后件中原子, $J \infty T$ 表示 J 与 T 自然连接, $|x|$ 表示 x 中的元组数, 则元规则的支持度为

$$s(r) = \max\left(\frac{|\pi_{\mathrm{att}(r_i)}(J)|}{|r_i|}\right), 1 \leq i \leq m; \tag{2}$$

元规则的可信度为

$$c(r) = \frac{|\pi_{\mathrm{att}(J)}(J \infty T)|}{|J|}. \tag{3}$$

元规则涉及多个关系表, 因此元规则的支持度计算不同于单个关系表的规则支持度计算. 文献[6]认为元规则的支持度应为 $|J| / \prod_{i=1}^{m} |r_i|$, 这样计算出的支持度随 $|r_i|$ 的增大而急剧减小. 本文采用的是文献[4, 5]的支持度计算公式, 它比较合理.

3　连接属性及相关理论

定义 3　对于数据类型相同的两个属性 A_x, A_y, 它们的值集分别为 V_x 和 V_y, $|V|$ 表示集合 V 的所有元素数目, 定义 A_x, A_y 的属性重叠度

$$\mathrm{Overlap}(A_x, A_y) = \max\left(\frac{|V_x \cap V_y|}{|V_x|}, \frac{|V_x \cap V_y|}{|V_y|}\right). \tag{4}$$

由于各种数据类型属性的值集都存在, 该公式适用于所有数据类型属性的重叠度计算. 例如: 假设属性 A_x, A_y 的值集分别为 $V_x = \{2, 4, 5\}$ 和 $V_y = \{2, 3, 4, 5\}$, 因为 $V_x \cap V_y = \{2, 4, 5\}$, 所以

$$\mathrm{Overlay}(A_x, A_y) = \max(3/3, 3/4) = 1.$$

关系表 T 中所有属性组成的集合称为 T 的属性集, 记为 $\mathrm{att}(T)$.

定义 4　考虑关系表 T_x, T_y 中的两个属性 A_x, A_y, 且满足 $A_x \in \mathrm{att}(T_x), A_y \in \mathrm{att}(T_y)$, 如果 $\mathrm{Overlap}(A_x, A_y) \geq k (0 \leq k \leq 1)$, k 为属性连接的门槛值, 由用户根据挖掘任务的具体情况确定, 则称 A_x 和 A_y 是可连接的, 记为 A_x-A_y, 亦称它们是关系表 T_x, T_y 的一对连接属性.

例如: 在定义 3 的例子中, 如果取 $k = 0.7$, 则由于 $\mathrm{Overlap}(A_x, A_y) = 1 > k$, A_x 与 A_y 是可连接的, 即 A_x-A_y.

定义 5　元规则 MQ 在数据库 DB 中是可实例的, 当且仅当 MQ 满足下面 2 个条件:

1) MQ 的每个谓词 P 都能实例为 DB 中的关系 rel, 且 P 的不同属性变元 $X_u, X_v (u \neq v)$ 能实例为 rel 的不同属性 $A_j, A_k, j \neq k$;

2) 如果 MQ 的两个谓词 $P(X_1, X_2), Q(Y_1, Y_2)$ 含有相同的属性变元, 即 $X_i = Y_j, 1 \leq i, j \leq 2$, 则 X_i, Y_j 的实例属性 $I(X_i), I(Y_j)$ 是可连接的.

定义 5 的条件 1) 说明了谓词实例为关系时必须满足的条件; 条件 2) 说明了当元规则含有相同属性变元时, 其实例属性要具有较高的属性重叠度, 这就避免了用属性"分数"和"年龄"去实例 MQ 中相同的属性变元.

例如: 考虑包含关系 t_1, t_2 和 t_3 的 DB 和 MQ: $P(X, Y) \wedge Q(Y, Z) \rightarrow R(X, Z)$. 如果谓词 $P(X, Y), Q(Y, Z)$ 和 $R(X, Z)$ 可分别被实例为关系 $t_1(a_1, a_2)$, $t_2(b_1, b_2)$ 和 $t_3(c_1, c_2)$, 且 a_1-c_1, a_2-b_1, b_2-c_2, 则称 MQ 在 DB 中是可实例的.

由定义 5 可得出如下性质:

性质 1　如果属性变元 X 出现在 $n (n \geq 2)$ 个谓词中, 则 X 在任意两个谓词中的实例属性都是可连接的.

定义 6　有 $n (n \geq 2)$ 个属性, 如果其中任意两个属性均为可连接的, 则称这 n 个属性构成了一个 n 项属性连通图, 用 $G_n(V)$ 表示, V 是所有属性的集合. 所有元素均为 n 项属性连通图的集合称为 n 项属性连通图集, 记为 GS_n.

一对连接属性可看成一个二项连通图, 例如 A_1-A_2 也可表达为 $G_2(V)$, 其中 $V = \{A_1, A_2\}$. 属性连通图可用图表示, 图的顶点对应于属性, 图的边对应于属性间的连接, 如图 1 所示.

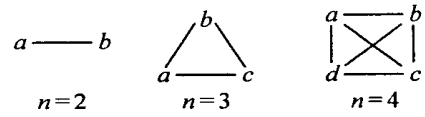

图 1　n 项属性连通图

n 项属性连通图概念的提出是针对属性变元 X 出现在 n 个谓词中的情况. 此时, X 的一个实例对应一个 $G_n(V)$, V 中的 n 个属性对应 X 在 n 个谓词中的实例属性, 这样便确保了 X 的实例满足性质 1.

例如: X 为元规则MQ 的属性变元, 令 X^{P_1}, X^{P_2} 分别表示谓词 P_1, P_2 中的 X, 假设 X 的实例对应图 1 中的二项属性连通图, 则 X^{P_1}, X^{P_2} 可分别实例为属性 a, b 或 b, a.

定理 1 $G_{n-1}(V_i)$, $G_{n-1}(V_j)$ 和 $G_{n-1}(V_k)$ 为 3 个互不相同的 $n-1$ 项属性连通图, 如果满足 $|V_i \cup V_j| = n$, 且 $V_k \subset (V_i \cup V_j)$, 则它们可以构成 n 项属性连通图.

证明 令 $V = V_i \cup V_j$, 因为 $V_k \subset (V_i \cup V_j)$, 所以 $V = V_i \cup V_j \cup V_k$; 因为 $|V_i| = |V_j| = |V_k| = n-1$, 可令属性 $\overline{v_i} = V - V_i$, $\overline{v_j} = V - V_j$, $\overline{v_k} = V - V_k$; 因为 V_i, V_j, V_k 互不相同, 所以 $\overline{v_i}, \overline{v_j}, \overline{v_k}$ 也互不相同.

从 V 中任取两属性 a, b, 因为 $V = V_i \cup V_j$, 所以相当于 a, b 从 $V_i \cup V_j$ 中任意选取.

1) 若 $\{a, b\} \subseteq V_i$ 或 V_j, 则显然 a, b 是可连接的;

2) 若 $\{a, b\} \not\subset V_i$ 且 $\{a, b\} \not\subset V_j$, 假设 $a \in V_i$, $b \in V_j$, 因为 $a \notin V_j$ (否则 $\{a, b\} \subseteq V_j$), 所以 $a = \overline{v_j}$. 同理, $b = \overline{v_i}$; 因为 $\overline{v_j} \neq \overline{V_k}$, 所以 $a = \overline{v_j} \in V_k$, 同理, $b = \overline{v_i} \in V_k$, 即 $\{a, b\} \subseteq V_k$, 从而 a 和 b 是可连接的.

从 V 中任取两属性都是可连接的, 由定义 6 知, V 中的 n 个属性构成了 n 项属性连通图 $G_n(V)$. □

定理 1 回答了如何由 $G_{n-1}(V)$ 生成 $G_n(V)$ 这个问题.

定义 7 如果 n 个属性连通图分别对应 n 个属性变元的实例, 且满足条件: 1) 任意两个连通图 $G_i(V_1)$ 和 $G_j(V_2)$, 均有 $V_1 \cap V_2 = \varnothing$; 2) 对于任意图 $G_i(V_1)$, 至少存在一个图 $G_j(V_2)$, 使得 V_1, V_2 分别包含的两个属性 a_u, a_v 是来源于DB 中同一个关系 (记为 $a_u \leftrightarrow a_v$); 则称它们构成了 n 项关系连通图, 记为 $RG_n(GS, R)$. 其中 GS 表示 n 个属性连通图组成的集合, R 是所有来自同一关系的属性对的集合.

关系连通图也可用图 2 表示. 关系连通图反映了元规则形式上的约束, 可用来确定谓词变量的实例关系.

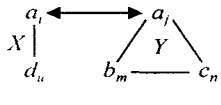

图 2 2 项关系连通图

例如 X, Y 为元规则MQ 的两个属性变元, X 出现在谓词 P_1, P_2 中, Y 出现在谓词 P_1, P_3, P_4 中. 假设图 2 为MQ 的一个关系连通图, 属性 a_i 和 a_j, b_m, c_n, d_u 分别是DB 中的关系 T_a, T_b, T_c, T_d 的属性. 由图 2 可知, X 的实例对应 $G_2(\{a_i, d_u\})$, Y 的实例对应 $G_3(\{a_j, b_m, c_n\})$. 因为 X, Y 都出现在谓词 P_1 中, 所以 X^{P_1}, Y^{P_1} 必须实例为同一关系的不同属性. 因为 $a_i \leftrightarrow a_j$, 所以 X^{P_1}, Y^{P_1} 分别实例为 a_i, a_j. 这样, X^{P_2} 只能实例为 d_u; Y^{P_3}, Y^{P_4} 可分别实例为 b_m, c_n 或 c_n, b_m. 相应地, 谓词 P_1, P_2 分别实例为关系 T_a, T_d; 谓词 P_3, P_4 可分别实例为关系 T_b, T_c 或 T_c, T_b.

4 元规则的实例方法

基于连接属性的相关概念和理论, 本文提出一种实例元规则的方法. 该方法通过DB 中的各项属性连通图集求出属性变元的实例集; 然后构造关系连通图, 从而求出元规则的候选实例集; 最后过滤掉支持度 $s < s_{min}$ 或可信度 $c < c_{min}$ 的候选实例, 产生元规则的实例集. 具体步骤如算法 1 所示.

若属性变元 X 在 n 个谓词中出现, 则称变元 X 的次为 n. 令 n_v 表示MQ 中所有属性变元数, d_i 表示第 i 个变元 X_i 的次, D 为所有变元的次的集合, $IA(X_i)$ 表示 X_i 的实例集.

算法 1 InitMQ (MQ, DB, s_{min}, c_{min})

input: 元规则MQ, 最小支持度 s_{min} 和最小可信度 c_{min};

output: 元规则的实例集 M;

Step 1: to compute n_v, D, and $D_{max} = max(D)$;

Step 2: Initialize Set RGS = $\varnothing$;

Step 3: for (int $j = 2$; $j \leq D_{max}$; $j = j + 1$) do {

Step 4: if ($j == 2$) then compute GS_2;

Step 5: else GS_j = Gen-LG(GS_{j-1}); }

Step 6: for (int $i = 1$; $i \leq n_v$; $i = i + 1$) do

Step 7: if ($d_i \geq 2$) then $IA(X_i) = GS_{d_i}$;

Step 8: for ($i = 1$; $i \leq n_v$; $i = i + 1$) do {

Step 9: $D_1 = max(D)$;

Step 10: if ($D_1 == 1$) then break;

Step 11: if ($D_1 == D_{max}$) then RGS = $IA(X_i^{d_i = D_1})$;

Step 12: else RGS = Gen-RG(RGS, $IA(X_i^{d_i = D_1})$);

Step 13: $D = D - \{D_1\}$; }

Step 14: produce MQ candidate instantiate set M_c from RGS;

Step 15: for each M_c^i do{

Step 16: if ($s(M_c^i) < s_{min}$) or ($c(M_c^i) < c_{min}$) then

Step17: $M_c = M_c - \{M_c^i\}$; }

Step18: return $M = M_c$.

算法1中, $X_i^{d_i = D_1}$ **表示满足** $d_i = D_1$ **的** X_i; Step3～Step5 **产生DB中各项属性连通图集**; Step6, Step7 **求出** $d_i \geq 2$ **的变元的实例集**; Step8～Step13 **是由** $d_i \geq 2$ **的所有变元的实例生成关系连通图**; Step14 **是由每个关系连通图求出各谓词变量的实例关系, 然后通过已实例的关系产生** $d_i = 1$ **的所有变元的实例属性集, 最终生成元规则的候选实例集**; Step15～Step18 **是经过** s_{min}, c_{min} **筛选, 生成元规则实例集**

其中, 算法 Gen-LG **是由** n **项属性连通图集产生** $n+1$ **项属性连通图集, 具体步骤如算法2所示; 算法** Gen-RG **是生成关系连通图, 具体步骤如算法3所示**

算法2　Gen-LG(GS_n)

input: n **项属性连通图集** GS_n;

output: $n+1$ **项属性连通图集** GS_{n+1};

Step1: Initialize Set $LG_0 = LG_1 = LG_2 = GS_n$; $GS_{n+1} = \varnothing$;

Step2: for each LG_0^i do {

Step3: $I_0 = LG_0^i, LG_1 = LG_0 - \{I_0\} - \bigcup_l G_n^{V_l \subset V^{n+1}(I_0)}(V_l)$;

Step4: for each LG_1^j do {

Step5: if ($|AS(I_0) \cup AS(LG_1^j)| = n+1$) then {

Step6: $I_1 = LG_1^j; LG_2 = LG_1 - \{I_1\} - \bigcup_l G_n^{V_l \subset V^{n+1}(I_1)}(V_l)$;

Step7: for each LG_2^k do {

Step8: $I_2 = LG_2^k$;

Step9: if ($AS(I_2) \subset (AS(I_0) \cup AS(I_1))$) then {

Step10: $GS_{n+1} = GS_{n+1} + G_{n+1}(V)$, and $V = AS(I_0) \cup AS(I_1)$;

Step11: $LG_1 = LG_1 - \{I_2\}$;

Step12: }break;

Step13: }}$LG_1 = LG_1 - \{I_1\}$;

Step14: }$LG_0 = LG_0 - \{I_0\}$;

Step15: } return GS_{n+1}.

算法2中, $AS(I_0)$ **表示** I_0 **的构成属性的集合**; $V^{n+1}(I_0)$ **表示构成属性包含** $AS(I_0)$ **的** $G_{n+1}(V)$ **的** V, $\bigcup_l G_n^{V_l \subset V^{n+1}(I_0)}(V_l)$ **表示满足条件** $V_l \subset V^{n+1}(I_0)$ **的所有** $G_n(V_l)$; I_0, I_1 **和** I_2 **是3个** n **项属性连通图, 根据定理1判断它们是否可以构成** $n+1$ **项属性连通图**

算法3　Gen-RG(RGS_n, $IA(X_k)$)

input: n **项关系连通图集** RGS_n, X_k **的实例集** $IA(X_k)$;

output: $n+1$ **项关系连通图集** RGS_{n+1};

Step1: Initialize Set $RGS_{n+1} = \varnothing$;

Step2: for each RGS_n^i do{

Step3: for each $IA^j(X_k)$ do{

Step4: if ($V^i \cap V^j = \varnothing$) then{

Step5: Initialize set $R = \varnothing$;

Step6: for each $a_u \in V^i$, each $a_v \in V_j$ do {

Step7: if ($a_u \leftrightarrow a_v$) and (not exiting other relation connections between $G_l(V(a_u))$ and $IA^j(X_k)$) then $R = R \cup \{a_u \leftrightarrow a_v\}$; }

Step8: if($R \neq \varnothing$) then

Step9: produce $RG_{n+1}(RGS_n^i \cup IA^j(X_k), R^i \cup R)$;

Step10: } $RGS_{n+1} = RGS_{n+1} + RG_{n+1}$;

Step11: }} return RGS_{n+1}.

算法3中, V^i, V^j **分别表示** RGS_n^i, $IA^j(X_k)$ **的所有构成属性的集合**; $G_l(V(a_u))$ **表示关系连通图** RGS_n^i **中以** a_u **为构成属性的属性连通图**

5　应用实例

图3是某数据库包含的4个关系表, 元规则的形式为 $P(X, Y) \wedge Q(Y, Z) \rightarrow R(X, Z)$.

Table T_a

a_1	a_2	a_3
jj	5	0.5
nn	5	0.8
ll	7	0.5
qq	5	0.5
kk	5	0.6
pp	4	0.6
mm	2	0.5
nn	4	0.6
kk	4	0.4
nn	5	0.4

Table T_b

b_1	b_2	b_3
14	0.5	mmm
16	0.6	jjj
14	0.3	jjj
12	0.7	nnn
12	0.1	lll
15	0.6	rrr
15	0.4	mmm
13	0.6	ooo
16	0.6	ooo
14	0.4	lll
14	0.6	kkk
15	0.3	mmm
12	0.5	mmm
15	0.4	nnn

Table T_c

c_1	c_2
14	qq
15	oo
16	nn
15	kk
16	ll
15	ll
15	mm
13	pp
16	oo
14	kk
13	mm
14	nn

Table T_d

d_1	d_2	d_3
nnn	0.0	13
mmm	0.0	13
rrr	0.1	14
mmm	0.0	15
ooo	0.1	15
jjj	0.0	16
kkk	0.0	12
mmm	0.0	14
jjj	0.1	15
mmm	0.0	15
jjj	0.0	13
jjj	0.0	16
lll	0.0	15
mmm	0.1	16

图3　某DB包含的关系表

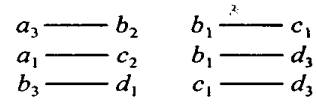

图4　2项属性连通图

采用本文方法求元规则的实例集 方法中连接属性的门槛值 k **的大小直接影响DB中的各项属性连通图和关系连通图的数目, 从而影响元规则的候**

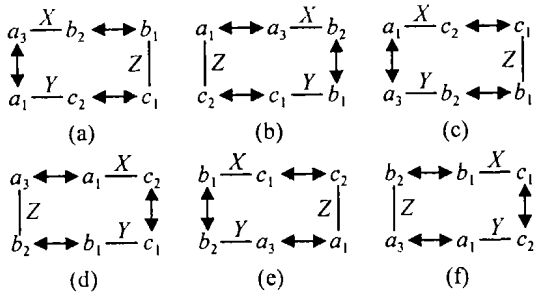

图 5 3 项关系连通图

选实例数目. k 取值越大, 元规则的候选实例越少. 本例中取连接属性的门槛值 $k = 0.6$, DB 中所有关系的二项属性连通图如图 4 所示. 变元 X, Y, Z 的实例集均为 GS_2, 由算法 1 共产生 6 个 3 项关系连通图, 如图 5 所示. 根据图 5(a), 包含变元 X, Y 的谓词变量 P 实例为关系 T_a, Q 实例为关系 T_c, R 实例为关系 T_b. 因为元规则的所有变元的实例属性都已确定, 所以元规则的候选实例为

$$\sigma_1 = \left\{\frac{T_a(a_3, a_1)}{P(X, Y)}, \frac{T_c(c_2, c_1)}{Q(Y, Z)}, \frac{T_b(b_2, b_1)}{R(X, Z)}\right\}$$

同理可得

$$\sigma_2 = \left\{\frac{T_b(b_2, b_1)}{P(X, Y)}, \frac{T_c(c_1, c_2)}{Q(Y, Z)}, \frac{T_a(a_3, a_1)}{R(X, Z)}\right\},$$

$$\sigma_3 = \left\{\frac{T_c(c_2, c_1)}{P(X, Y)}, \frac{T_b(b_1, b_2)}{Q(Y, Z)}, \frac{T_a(a_1, a_3)}{R(X, Z)}\right\},$$

$$\sigma_4 = \left\{\frac{T_a(a_1, a_3)}{P(X, Y)}, \frac{T_b(b_2, b_1)}{Q(Y, Z)}, \frac{T_c(c_2, c_1)}{R(X, Z)}\right\},$$

$$\sigma_5 = \left\{\frac{T_b(b_1, b_2)}{P(X, Y)}, \frac{T_a(a_3, a_1)}{Q(Y, Z)}, \frac{T_c(c_2, c_1)}{R(X, Z)}\right\},$$

$$\sigma_6 = \left\{\frac{T_c(c_1, c_2)}{P(X, Y)}, \frac{T_a(a_1, a_3)}{Q(Y, Z)}, \frac{T_b(b_1, b_2)}{R(X, Z)}\right\},$$

参数 s_{min} 和 c_{min} 影响着元规则的实例数目, 它们取值越大, 元规则实例数目越少. 如果取 $s_{min} = 0.6$, $c_{min} = 0.55$, 则元规则的实例集为 $\{\sigma_1, \sigma_6\}$.

采用文献[4]的方法求该元规则的实例集. 该方法将相同的属性变元绑定到相同的数据类型作为约束, 共产生 $\left\{\frac{T_a(a_1, a_2)}{P(X, Y)}, \frac{T_b(b_1, b_2)}{Q(Y, Z)}, \frac{T_c(d_1, d_2)}{R(X, Z)}\right\}$ 等 72 个元规则候选实例, 经过 $s_{min} = 0.6$, $c_{min} = 0.55$ 筛选, 只保留了 σ_1 和 σ_6 两个元规则实例. 由此可见, 本文提出的方法能显著削减元规则的候选实例集. 此外, 由于该方法是逐个实例谓词, 可能需要不断回溯才会得到元规则候选实例. 例如, 如果 $P(X, Y)$ 实例为 $T_a(a_1, a_2)$, 因为 $T_c(b_1, b_3)$ 满足 Q 谓词的约束, 所以 $Q(Y, Z)$ 实例为 $T_c(b_1, b_3)$, 但此时找不到满足 R 谓词约束的关系来实例 $R(X, Z)$, 只能重新实例 $Q(Y, Z)$. 这种回溯显然影响了元规则的实例效率.

6 结 论

文献[4, 5]证明了元规则的实例是 NP 难题. 虽然本文提出的方法并没有改变元规则实例问题的时间复杂性性质, 但运用元规则中相同属性变元的实例属性是可连接的这一实例条件, 可大大削减元规则的候选实例集; 而且该方法将元规则形式上的约束通过关系连通图表达出来, 直接由关系连通图得到元规则的候选实例, 避免了逐个实例谓词的弊端. 总之, 本实例方法为元规则的实例提供了一条新的途径.

参考文献(References)

[1] Shen W, Ong K, Mitbander B, et al. *Metaqueries for Data Mining* [M]. Cambridge, MA: AAAI/MIT Press, 1996: 375-398.

[2] Fu Y, Han J. Meta-rule-guided Mining of Association Rules in Relational Databases [A]. *Proc 1st Int'l Workshop on Integration of Knowledge Discovery with Deductive and Object-Oriented Databases (KDOOD'95)* [C]. Singapore, 1995: 39-46.

[3] 欧阳为民, 蔡庆生. 大型数据库中多层关联规则的元模式制导发现[J]. *软件学报*, 1997, 8(12): 920-927. (Ou-Yang W M, Cai Q S. Meta-pattern Guided Discovery of Multi-level Association Rule in Large Databases[J]. *J of Software*, 1997, 8(12): 920-927.)

[4] Ben-Eliyahu-Zohary R, Gudes E, Ianni G. Metaqueries: Semantics, Complexity, and Efficient Algorithms [J]. *Artificial Intelligence*, 2003, 149(1): 61-87.

[5] Angiulli F, Ben-Eliyahu-Zohary R, Ianni G, et al. Computational Properties of Metaquerying Problems [A]. *Proc of PODS-2000*[C]. Dallas, TX, 2000: 237-244.

[6] Shen W M, Leng B. A Metapattern-based Automated Discovery Loop for Integrated Data Mining-unsupervised Learning of Relational Patterns[J]. *IEEE Trans on Knowledge and Data Engineering*, 1996, 8(6): 898-910.