带有多个可转化约束的频繁项集挖掘算法*)

宋宝莉[1] 张帮华[2] 何炎祥[1] 朱骁峰[1]

(武汉大学计算机学院 软件工程国家重点实验室 武汉430072)[1] (广州军区司令部 广州510000)[2]

Mining Frequent Item Sets with Multiple Convertible Constraints

SONG Bao-Li ZHANG Bang-Hua HE Yan-Xiang ZHU Xiao-Feng

(School of Computer, State Key Lab of Software Engineering, Wuhan University, Wuhan, 430072)

Abstract Recent work has highlighted the importance of the constraint-based mining paradigm in the context of frequent item sets, associations, correlations. Some research has raised the notion "convertible constraints", and this method can push some constraints into the mining algorithm which essentially can't be pushed. This article has introduced a multiple convertible constraints mining algorithm, which analyzes the constraints by taking advantage of a sample database and then decides a optimal method to process data mining

Keywords Data mining, Multiple constraints, Convertible constraints, Sample database

1. 引言

频繁项集的挖掘是数据挖掘课题中的一个很重要的方面，然而频繁项集的挖掘过程通常会产生数目庞大的频繁项集，并且其中的绝大多数并不是客户所期望得到的，因而使挖掘过程的效果和效率都大打折扣。

近来的一些研究工作很重视带有约束条件的频繁项集的挖掘。在这种挖掘模式下，允许用户通过一组约束条件来表达其在频繁项集挖掘中的侧重点。除了用户可以对挖掘的方向和重点进行控制之外，还可以把这些条件放入到挖掘过程中进行检验，从而可以减少产生频繁项集的模式空间，进而提高整个挖掘过程的执行效率。

约束条件自身有一些可以遵循的规律，也正是这些规律才使我们可以把它们放入挖掘算法，以便在挖掘过程中进行检验。在文[2,3]中介绍了两大类约束条件：简明(succinct)约束和反单调(anti-monotone)约束。文[4,5]则在关联规则的挖掘中，既研究了上述两种约束，又研究了第三种类型的约束——单调(monotone)约束。尽管如此，还是有一些约束条件不在这三类约束条件之中，文[1]介绍了可转化约束的概念，其思想是：通过某种转换，使一些约束条件可以归为这三类之中，从而可以把它们放入挖掘算法中去。但是可转化约束的转换是基于项的某个排序，不同的约束可能基于不同的排序才是可转化的。在多约束条件的情况下，要找到某个序列使那些可转化约束条件在该序列下都是可转化的，几乎是不可能的。

本文采取了一种折衷的办法，把焦点集中在一些常用的序列上，并采用一个抽样数据库对约束条件进行检验，进而找出一个较优的方案来进行频繁项集的挖掘。

2. 问题定义

设 $I=\{i_1,i_2,\cdots,i_m\}$ 是所有项的集合，一个项是一个对象，有若干个预先定义好的属性，一般会经常用到项的"值"属性。事务 $T=\langle tid,I_t\rangle$，是一个固定的数据结构，其中 tid 是事务标识符，I_t 是一个项集，并且 $I_t\subseteq I$。一个事务数据库 D_T 就是由事务 T 构成的集合。一个 k-项集 S 是 I 的一个子集，它包含有 k 个项。

当 $S\subseteq I_t$ 时，称事务 T 包含 k-项集 S，S 的支持度 $sup(S)$ 等于事务数据库 D_T 中包含 S 的事务的个数。给定一个最小支持度阈值 ξ，当 $sup(S)\geqslant\xi$ 时，称 k-项集 S 为一个频繁项集。

单一约束条件 C 是一个 I 的幂集上的映射，$C:2^I\rightarrow\{true,false\}$，当且仅当 $C(S)=true$ 时，称 S 满足约束条件 C。集合 $sat_C(I)=\{S\mid S\subseteq I\wedge C(S)=true\}$ 是由项集 I 的所有满

*)本工作得到国家自然科学基金重大研究计划(项目编号：90104005)，国家教育部重点项目，软件工程国家重点实验室开放基金等资助。宋宝莉 博士研究生，主要研究方向为信息系统与数据挖掘，计算机应用等。张帮华 教授级高工，总工，主要研究方向为网络与分布并行处理。何炎祥 教授，博士生导师，主要研究方向为分布并行处理，数据挖掘等。朱骁峰 硕士研究生，主要研究方向为分布并行处理，数据挖掘。

chaos in a single delayed neuron equation with nonmonotonic activation function. Chaos, Soliton and Fractals, 2001, 21: 1535～1547

7 Liao X F, Wong K W, Wu Z. Bifurcation analysis in a two-neuron system with distributed delays. Phys. D, 2001, 149: 123～141

8 Liao X F, Wong K W, Wu Z. Asymptotic stability criteria for a two-neuron network with different time delays. IEEE Trans. Neural Networks, Jan. 2003, 14(1): 222～227

9 Liao X F, Yu J B. Robust interval stability analysis of Hopfield networks with time delay. IEEE Trans. Neural Networks, 1998, 9: 1042～1045

10 Liao X F, Wong K W, Wu Z, Chen G. Novel robust stability criteria for interval Hopfield neural networks with time delays. IEEE Trans. Circuits. Syst. I, 2001, 48: 1355～1359

11 Liao X F, Wong K W, Yu J B. Novel stability conditions for cellular neural networks with time delays. Int. J. Bifurcation Chaos, 2001, 11(7): 1853～1864

12 Liao X F, Chen G, Sanchez E N. LMI-based approach for asymptotically stability analysis of delayed neural networks. IEEE Trans. Circuits Syst. I, 2002, 49: 1033～1039

13 Liao X F, Chen G, Sanchez E N. Delay-dependent exponential stability analysis of delayed neural networks: an LMI approach. Neural Networks, 2002, 15: 855～866

足约束条件C的子集构成，当一个k-项集是这个集合的一个元素时，则称这个k-项集为“C有效项集”。

多约束条件C_{mul}可以表示成由多个单一约束条件组成的表达式，这些单一约束条件由逻辑“与”和逻辑“或”连接起来，并且可以用括号来表示逻辑运算的优先次序。我们可以分以下几步来定义多约束条件：

·由逻辑“与”关系或者逻辑“或”关系连接起来的两个单一约束条件是多约束条件。

·由逻辑“与”关系或者逻辑“或”关系连接起来的一个单一约束条件和一个多约束条件是多约束条件。

·由逻辑“与”关系或者逻辑“或”关系连接起来的两个多约束条件是多约束条件。

我们可以根据这个递归定义来求多约束条件C_{mul}的$sat_{C_{mul}}(I)$。给定两个条件C1和C2(C1和C2可以是简单约束条件，也可以是多约束条件)，根据集合运算的知识显然可得：$sat_{C1\wedge C2}(I)=sat_{C1}(I)\cap sat_{C2}(I)$，$sat_{C1\vee C2}(I)=sat_{C1}(I)\cup sat_{C2}(I)$。当一个k-项集$S\in sat_{Cmul}(I)$时，称这个$k$-项集为“$C_{mul}$有效项集”。

那么，我们的问题可以这样定义：给定一个事务数据库D_T，一个最小支持度阈值ξ，一个多约束条件C_{mul}，带有多约束条件的频繁项集的数据挖掘就是找出所有支持度大于ξ，并且满足多约束条件C_{mul}的频繁项集的完全集。即：

$$F_{Cmul}=\{S|S\in sat_{Cmul}(I)\wedge sup(S)\geqslant\xi\}$$

3. 约束条件分析

正如前文所述，现有的研究把约束条件分为三类：单调约束、反单调约束和简明约束。下面我们简单回顾一下这些概念。

单调约束条件：一个约束条件是单调约束条件，当且仅当任意一个项集S满足约束条件时，S的任何一个超集也满足该约束条件。

反单调约束条件：一个约束条件是反单调约束条件，当且仅当任意一个项集S不满足约束条件时，S的任何一个超集也不满足该约束条件。

简明约束条件，在这里就不再赘述它的递归定义，仅说明一下它的一个性质：任何一个简明(succinct)约束条件，可以被表示为用逻辑“与”和逻辑“非”连接起来的单调约束条件和反单调约束条件[1]。因此，简明约束条件也是一种多约束条件，并且可以把研究的重点只放在单调约束条件和反单调约束条件上来。

在常用的约束条件中，仍然有一部分约束条件不能被归为上述三类约束条件，文[1]提出了可转化约束条件的概念。有些约束条件可以被转换成为单调约束条件，这类约束条件被称为“可转化单调约束条件”；有些约束条件可以被转换成为反单调约束条件，这类约束条件被称为“可转化反单调约束条件”；还有些约束条件既可以被转换成为单调约束条件，又可以被转换成为反单调约束条件，这类约束条件就被称为“强可转化约束条件”。接下来，简单介绍一下这种转换的思想。

比如说这样一个约束条件：$avg(S)\geqslant i$，当项集I中的项没有经过任何排序时，这个约束条件没有任何单调和反单调的性质。但是，如果把I中的项按照项的值的升序或降序来排列的话，就可以表现出来一定的单调和反单调的性质。例如，假设$I=\{a,b,c,d,e\}$，并且a、b、c、d、e五个项的值分别为：20,50,10,30,100，显然，$avg(ab)\geqslant30$成立，但是项集abc(省略集合括号)却不能满足这个条件；反之，avg(abcde)≥30成立，但是它的子集abc却不能满足此约束条件，也就是说这个约束条件既不具有单调性，又不具有反单调性。但是我们把I中的项按值的升序排列，那么上述约束条件就会有单调性质；按降序排列，约束条件就会有反单调性质。由于Apriori类型的频繁项集挖掘算法对单调和反单调的性质的要求，相对于定义来讲要弱，即仅要项集S的超集中，那些以S为前缀的超集满足单调和反单调的要求即可。因此，通过对项集I中的项进行排序而得到的这些性质也就足够使用了。文[1]还定义了前缀单调函数来利用这些性质。项集S'是项集S的前缀，如果$f(S')\leqslant f(S)$，那么f就是前缀单调增函数，反之就是前缀单调减函数。把单个约束条件表示为$f(S)\theta\ v$的形式，其中θ是关系运算符，v是一个数值，表1[1]总结了函数单调性质和可转化约束分类之间的关系。

根据反单调的性质，子集不满足的约束，所有超集也不满足，可以很大程度上削减产生频繁项集的模式空间。单调的性质虽然不能减少模式空间，但某个子集满足的约束，其超集一定满足，所以某个项集的子集一旦被验证过满足约束条件，该项集就可以不再验证。文[1]分别针对这两种约束建立了两种算法FIC_A(anti-monotone)和FIC_M(monotone)。

表1

约束条件	可转化反单调	可转化单调	强可转化
$avg(S)\theta\ v\ \theta\in\{\leqslant,\geqslant\}$	是	是	是
$median(S)\theta\ v\ \theta\in\{\leqslant,\geqslant\}$	是	是	是
$sum(S)\leqslant v\ v\geqslant0$ S中的所有项值同号	是	否	否
$sum(S)\leqslant v\ v\leqslant0$ S中的所有项值同号	否	是	是
$sum(S)\geqslant v\ v\geqslant0$ S中的所有项值同号	否	是	是
$sum(S)\geqslant v\ v\geqslant0$ S中的所有项值同号	是	否	否
$f(S)\geqslant v$ f是前缀减函数	是	不定	不定
$f(S)\geqslant v$ f是前缀增函数	不定	是	不定
$f(S)\leqslant v$ f是前缀减函数	不定	是	不定
$f(S)\leqslant v$ f是前缀增函数	是	不定	不定

以上的一些内容都是针对单一约束条件的，多约束条件下的频繁项集的挖掘还存在一些难点。首先，约束条件的转换是依赖于项集I中项的排序，要找到某一个排序能使多约束中的所有可转化单一约束可以进行转换，几乎是不可能的。况且，一个项的个数为n的项集，其项的排列有n!个之多。其次，约束条件验证的繁简程度也不尽相同，我们期望能把那些验证方便、选择度高的约束条件放入到挖掘算法中去，最好是反单调的，以期减少模式空间。本文的以下部分主要讨论这些问题的解决方法。

4. 挖掘算法及其证明

4.1 挖掘算法思想及描述

对于带有约束条件的数据挖掘，最一般的解决办法是：先把频繁项集的完全集挖掘出来，然后用约束条件去过滤掉不符合约束的频繁项集，但是在频繁项集的挖掘中，会产生大量的候选项集。研究表明，如果有10^4个候选1-项集，则会产生10^7个候选2-项集。所以应当尽量地把约束条件，尤其是反单调的约束条件放入到挖掘算法中去。

在多约束条件下，问题变得更加复杂，使得很难将约束条

件都加进挖掘算法中，但是我们希望尽可能地把那些能极大提高效率的约束条件加入到算法中。下面将针对上一节提出的问题，介绍本文的解决方案。

多个可转化约束的频繁项集的挖掘算法的基本思想是，既然不能把所有的约束都放入算法中去，就把多个单一约束构成的多约束条件分成两个部分，这两部分之间是逻辑"与"的关系，把其中的一部分放入到挖掘算法中，另一部分对挖掘的结果进行过滤，从而得到最终结果。如何让计算机把第一部分从多约束条件中分离出来成为问题的关键。

含有 n 个项的 I 项集的排列有很多，如果按照每一个排序，对单一约束进行检验来断定它是否是可转化的，这样的代价太大，以至于还不如前面所述的最一般的方法有效。我们注意到这样一个事实：比如说，约束条件是关于项的值的，那么大多数可转化约束的转换是基于值的升序或降序两种排序。也就是说，有一个最小的常用排序集，我们仅需在这个集合内对约束进行检验。所谓检验就是按照某个排序，检查约束条件函数是前缀增函数，还是前缀减函数，然后根据表1进行判断。可能有个别的约束条件是在其他排序下才是可转化的，这种检验方式会把它漏掉，但这属于正常的牺牲范围。

这样所有可转换约束就按不同的排序和单调性质聚类，那些原本就是单调或反单调的约束条件，不论排序如何，都仍是单调或反单调的，因此它们可以归为任何一类。由于在进行数据挖掘的时候，仅能按照所有排序中的一个对 I 项集的项进行排序，因此只能从这些聚类后的约束条件中选出一组作为第一部分约束放入算法。另外还需注意的一点是：放入算法的那部分约束和其余部分的约束应该是逻辑"与"关系，所以被选中聚类中的约束条件并不是每一个都可以放入算法中的，应该把那些不能放入算法的约束从每一个聚类中剔除。例如，多约束条件为 $C_1 \wedge C_3 \wedge C_4 \vee C_1 \wedge C_2 \wedge C_4$，把它变成逻辑"与"连接的形式，即 $C_1 \wedge C_4 \wedge (C_2 \vee C_3)$，假设约束条件 $C_1C_2C_4$ 是某个排序上的反单调约束，则应该把 C_2 从该聚类中剔除。接着我们通过一个在原数据库上生成的抽样数据库来进行选择，首先用比最小支持度阈值 ξ 小得多的阈值 ξ' 产生抽样数据库的频繁项集的完全集，然后用这个集合检验每一个单一约束条件的选择性，选择性高的优先放入挖掘算法。

最后是选用算法 FIC_A（anti-monotone）和 FIC_M（monotone）中哪一个进行数据挖掘的问题。由于 FIC_M 算法不能削减模式空间，因此优先使用 FIC_A 算法，即反单调的约束条件优先。在无法使用 FIC_A 算法的情况下再使用 FIC_M 算法。多个可转化约束的频繁模式挖掘算法（多约束挖掘算法）描述如下：

输入：多约束条件 C_{mul}，最小支持度阈值 ξ，事务数据库 D_T
输出：满足约束 C_{mul} 的频繁项集的完全集。
方法：

1. 变换 C_{mul} 的形式，使之成为逻辑"与"连接的形式；生成 D_T 的抽样数据库 D_{TS}；根据项的信息产生最小常用排序集。
2. 按最小常用排序集中的每一个排序，对构成 C_{mul} 的所有单一约束条件进行判断，看它在该排序上是否具有单调或反单调的性质。若没有找到一个约束具有这样的性质，则退出算法，提示增加最小常用排序集的元素个数；否则，把具有单调或反单调性质的约束按排序和单调性质分类。剔除每一个聚类中不能加入到挖掘算法的约束。
3. 以比 ξ 小的阈值 ξ' 生成 D_{TS} 的所有频繁项集，然后针对以上每一个聚类计算其选择性。选出一组要加入到挖掘算法的约束条件，构成第一部分约束条件 C'。
4. 如果 C' 是反单调约束，则用参数 ξ、D_T 和 C' 调用 FIC_A 算法；反之，则用参数 ξ、D_T 和 C' 调用 FIC_M 算法。
5. 用余下的约束条件对上一步产生的结果进行过滤，得到最终结果。

4.2 挖掘算法正确性证明

引理4.1 多约束挖掘算法能挖掘出所有满足约束条件的频繁项集。

证明：多约束挖掘算法把多约束条件 C_{mul} 等价变换成 $C' \wedge C_{REST}$ 的形式，并把 C' 带入到挖掘算法 FIC_A 或 FIC_M 中，进行频繁项集的挖掘，即上述算法执行完第4步，得到中间结果 $F_{C'} = \{S \mid S \in sat_{C'}(I) \wedge sup(S) \geqslant \xi\}$。然后用余下的约束条件对这个结果进行过滤，即得到 $F = \{S \mid S \in sat_C(I) \wedge S \in sat_{CREST}(I) \wedge sup(S) \geqslant \xi\} = \{S \mid S \in sat_C(I) \cap sat_{CREST}(I) I \wedge sup(S) \geqslant \xi\}$。因为 C_{mul} 等价于 $C' \wedge_C REST$，所以 $sat_{C_{mul}}(I) = sat_C(I) \cap sat_{CREST}(I)$。因此，算法得到的最终结果为正确结果 $F = F_{C_{mul}} = \{S \mid S \in sat_{C_{mul}}(I) \wedge sup(S) \geqslant \xi\}$ 为正确结果，即挖掘出了所有满足条件的频繁项集。

引理4.2 多约束挖掘算法的效率，不会低于先挖掘出所有频繁项集，再用多约束条件过滤的一般解决办法。

证明：此算法的时间耗费主要有两部分，第一部分是对约束条件进行处理所耗费的时间，第二部分是调用挖掘算法进行挖掘的时间。由于把具有较高选择性的约束条件加入到算法当中，将极大限度地削减产生频繁项集的模式空间，这一部分的高效性文[1]已做论证。在此仅讨论第一部分的时间耗费，这一部分时间耗费的计算涉及到参数 N_C（单一约束的个数）、N_I（I 项集中项的个数）和 N_{DS}（抽样数据库中事务的个数）。挖掘算法的步骤2是检验条件的单调性和反单调性，其时间复杂度是 $O(N_C * N_I)$；步骤3是用抽样数据库检验条件的选择性，主要时间耗费是产生供测试用的频繁项集，其时间复杂度为 $O(N_{DS}^K)$，其中参数 K 根据不同的算法而不同，一般大于1。而用一般的解决办法，产生含有 N 个事务的数据库的频繁项集的时间复杂度为 $O(N^K)$。但是 N_I 和 N_{DS} 相对于 N 至少要小一个数量级，因此第一部分的时间耗费相对于产生频繁项集的耗费是可以忽略的，第二部分节省的时间足够包含第一部分耗费的时间，所以该算法是有效的。

结束语 本文主要是利用最小常用排序集和抽样数据库的方法，极大地减少用于条件检验的时间。在可承受的时间耗费之内，迅速对多约束条件进行分析和转换，并尽可能地把高选择性的约束条件加入到挖掘算法当中去，以期提高频繁项集的挖掘效率。带有约束的频繁项集的挖掘，尤其是多约束条件下的频繁项集挖掘仍是一个值得深入研究的课题。本文中所使用的最小常用排序集仍需要根据项的不同属性，事先人为地指定，如何让计算机自动地确定这个集合以及如何更高效地判断一个约束条件的单调性是我们今后要继续深入的工作。

参考文献

1 Pei J, Han J. Mining Frequent Itemsets with Convertible Contraints. In: Proc. 2001 Int. Conf. on Data Engineering (ICDE'01), Heidelberg, Germany, April 2001

2 Lakshmanan L V S, Ng R, Han J, Pang A. Optimization of Constrainted Frequent Set Queries with 2-variable Contraints. In: Proc. 1999 ACM-SIGMOD Int. Conf. Management of Data, Philadelph, ia, PA, June 1999. 157～168

3 Ng R, Lakshmanan L V S, Han J, Pang A. Exploratory mining and pruning optimizations of constrained associations rules. In: Proc. 1998 ACM-SIGMOD Int. Conf. Management of Data (SIGMOD'98), Seattle, WA, June 1998. 13～24

4 Grahne G, Lakshmanan L, Wang X. Efficient mining of constrained correlated sets. In: Proc. 2000 Int. Conf. Data Engineering (ICDE'00), San Diego, CA, Feb. 2000. 512～521

5 Pei J, Han J. Can we push more constraints into frequent pattern mining? In: Proc. 2000 Int. Conf. Knowledge Discovery and Data Mining (KDD'00), Boston, MA, Aug. 2000. 350～354