

# 负增量关联规则更新算法研究

郭有强<sup>1</sup>, 胡学钢<sup>2</sup>

(1. 蚌埠学院 计算机科学与技术系, 安徽 蚌埠 233030;

2. 合肥工业大学 计算机与信息学院, 安徽 合肥 230009)

**摘要:**在增量式关联规则更新算法的研究中,关于负增量式更新算法的研究比较少。提出了一种实用的在支持度和置信度不变的情况下数据集规模减小的负增量关联规则更新算法。算法在如何减少数据集的扫描次数,如何充分利用现有的信息减少候选集的规模等方面进行了研究,给出了算法的具体实现。通过分析,算法是可行的。

**关键词:**关联规则;增量更新算法;剪枝

**中图分类号:**TP311

**文献标识码:**A

**文章编号:**1673-629X(2007)09-0048-03

## Study of Updating Algorithm for Negative Incremental Association Rule

GUO You-qiang<sup>1</sup>, HU Xue-gang<sup>2</sup>

(1. Computer Sci-Tech Department, Bengbu College, Bengbu 233030, China;

2. College of Computer and Info., Hefei Industrial University, Hefei 230009, China)

**Abstract:** In the study of updating algorithm for incremental association rules, little research has been done on the negative incremental updating algorithm. Provides a practical updating algorithm for negative incremental association rules in which the size of data sets is reduced, with the supporting and confidence limits unchanged. The algorithm explores how to diminish the number of scanning data sets, and how to make the best use of known information to shorten the size of candidate sets and so on. The concretization of the algorithm is also given. To sum up, the algorithm is feasible through analysis.

**Key words:** association rules; incremental updating algorithm; pruning

## 0 引言

模式维护是数据挖掘中一个具有挑战性的任务。关联规则<sup>[1]</sup>更新算法<sup>[2,3]</sup>就是在数据集规模、支持度  $s$ 、置信度  $c$  发生变化后,重新获得新的关联规则的高效算法。根据实际应用需求,关联规则的更新问题可以分为以下几种情况:

(1)事务数据库不变,最小支持度发生变化时,关联规则的高效更新问题;

(2)最小支持度不变,一个事务数据集添加到事务数据库时,如何生成最新事务数据库中的关联规则;

(3)最小支持度不变,从事务数据库删除一个事务数据集后,如何高效生成事务数据库中的关联规则。

至于其它情况,可由上述三种组合而成,因此,这三种情况是更新问题的基础和核心。

对于前两种情况,许多国外和国内的研究者已经提出了很多相关的算法,但对于后一种情况,研究的人员及提出的算法相对较少。

随着时间的推移,事务数据集中的一些较老的数据可能会过时,由他们产生的规则可能变得没有价值,有时还会成为噪音,为了保证挖掘结果的有效性,需要删除这些过时的数据。数据集发生了变化,已存在的规则也应随之更新。如何利用原来挖掘的结果高效生成更新后的关联规则成为维护算法讨论的焦点。关联规则维护算法高效的关键在于如何减少对数据集的扫描次数和生成较小的候选项集。文中研究的正是利用已经获得的频集结果,解决在支持度和置信度不变的情况下数据集规模减小的模式维护问题。

## 1 相关性质

$|D|$ 、 $|d|$ 、 $|D-d|$  分别为原数据库、欲减数据集、减后数据库的总事务条数; $L_D$  为原数据库  $D$  的频集, $L_d$  为欲减数据集  $d$  的频集, $L_{D-d}$  为减后数据库  $D-d$  的频集; $D$ 、 $d$  和  $D-d$  的最小支持数分别为

收稿日期:2006-11-27

基金项目:安徽省科技厅自然科学基金项目(050420207)

作者简介:郭有强(1966-),男,江苏邳江人,副教授,硕士,主要从事数据挖掘和可视化程序设计教学与研究;胡学钢,博士,教授,研究方向为知识工程、数据挖掘、数据结构。

$\text{support}_D, \text{support}_d, \text{support}_{D-d}$ ; 项目集  $X$  在数据集  $D, d, D-d$  中的支持数目分别记为  $X.\text{count}_D, X.\text{count}_d, X.\text{count}_{D-d}$ 。

性质1  $X \in L_D \cup L_d - L_D$  (见图1阴影部分), 则  $X$  在  $D-d$  中一定是不频繁的。

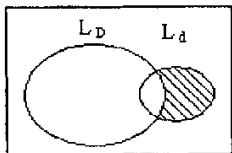


图1  $L_D \cup L_d - L_D$  部分的项目集

反证法: 显然  $X \in L_D, X \in L_d$ , 所以  $X.\text{count}_d \geq s \times |d|$  ①

假设  $X \in L_{D-d}$ , 则  $X.\text{count}_{D-d} \geq s \times |D-d|$  ②

① + ②  $X.\text{count}_{D-d} + X.\text{count}_d \geq s \times |D-d| + s \times |d|$

$X.\text{count}_D \geq s \times |D|$  得:  $X \in L_D$ , 与题意相反, 假设不成立。

性质2  $X \in L_D \cup L_d - L_d$  (见图2阴影部分), 则  $X$  在  $D-d$  中一定是频繁的。

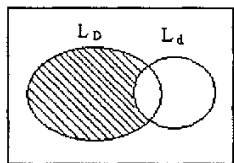


图2  $L_D \cup L_d - L_d$  部分的项目集

反证法: 显然  $X \in L_D, X \in L_d$ , 所以  $X.\text{count}_d < s \times |d|$  ①

假设  $X \in L_{D-d}$ , 则  $X.\text{count}_{D-d} < s \times |D-d|$  ②

① + ②  $X.\text{count}_D < s \times |D|$  得:  $X \notin L_D$ , 与题意相反, 假设不成立。

性质3  $X \in L_D \cap L_d$  (见图3阴影部分), 则  $X$  在  $D-d$  中不一定是频繁的。

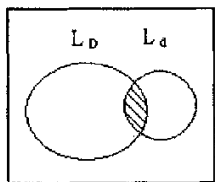


图3  $L_D \cap L_d$  部分的项目集

证明: 因为  $X \in L_D$ , 所以  $X.\text{count}_D \geq s \times |D|$

$X.\text{count}_{D-d} + X.\text{count}_d \geq s \times |D|$

$X.\text{count}_{D-d} \geq s \times |D| - X.\text{count}_d$

又因为  $X \in L_d$   $X.\text{count}_d \geq s \times |d| = s \times |d| + n$  ( $n = 0$  或正整数)

$X.\text{count}_{D-d} \geq s \times |D| - s \times |d| - n = s \times |D-d| - n$

所以不能保证  $X.\text{count}_{D-d} \geq s \times |D-d|$

## 2 更新算法基本思想及算法实现

### 2.1 更新算法基本思想

(1) 分离出欲减数据集  $d$ , 得到  $d$  中所有1项目的相关信息(每项目支持事务集合和支持事务计数), 得到频繁1-项集; 在此基础上使用连接项集  $L$  (即当前的频繁1-项集) 中的项目对频繁1-项集进行增长, 分别得到候选2-项集, 从而求得频繁2-项集, 将频繁2-项集中的所有项包含的单一项构成的集合, 与连接项集求交集, 动态更新连接项集  $L$  中的项目(即产生新的连接项集, 为以后增长做准备); ……; 在得到频繁  $K-1$  项集后, 利用求解过程中更新的连接项集中的项目进行增长, 分别得到候选  $K$  项集, 从而求得频繁  $K$  项集。此方法称为“项目增长法”。

(2) 扫描减后的数据集  $D-d$ , 得到  $D-d$  中所有1项目的相关信息(每项目支持事务集合和支持事务计数)。

(3) 所有项目集合由两部分组成:  $L_D \cup L_d$  和不属于  $L_D \cup L_d$  部分的项目集合。

a. 考虑  $L_D \cup L_d - L_D \cap L_d$ :

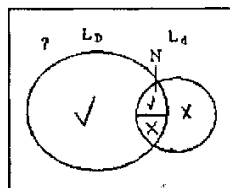
①  $L_D \cup L_d - L_D$ : 即  $X$  在  $d$  中是频繁的, 但在  $D$  中是不频繁的, 则在  $D-d$  中一定是不频繁的。(性质1)

②  $L_D \cup L_d - L_d$ : 即  $X$  在  $D$  中是频繁的, 但在  $d$  中是不频繁的, 则在  $D-d$  中一定是频繁的, 即  $X \in L_{D-d}$ 。(性质2)

b. 考虑  $L_D \cap L_d$ : 此部分的项集在  $D-d$  中可能频繁, 也可能不频繁。(性质3)

列出所有项目包含的单一项, 在  $D-d$  中查看单一项支持度, 将所有大于等于  $s^* \times |D-d|$  的作为一个集合  $M, M \in N$ ; 在  $L_D \cap L_d$  中去除有子项不包含在  $M$  中的项, 若  $X.\text{count}_{D-d} \geq s^* \times |D-d|, X \in N, N \in L_{D-d}$ 。

综上所述, 利用已知信息, 得到在  $D-d$  中项集是否频繁的分布如图4所示。



注: 图中 ✓ 为频繁, × 为不频繁, ? 为不一定

图4  $L_D \cup L_d$  中的项目集在  $D-d$  中是否频繁分布

c. 考虑不属于集合  $L_D \cup L_d$  的部分:

$D-d$  中所有 1 项集的支持度  $\geq s * |D-d|$  的为频繁 1 项集  $L_1, L_1 \in L_{D-d}$ ; 将  $L_1$  中的项进行两两组合得到二项集, 二项集中的项目若在  $L_{D-d}$  中, 将其放入  $C$ , 若不在, 则进一步判断是否在  $L_d - N$  中, 若不在, 则是候选二项集, 得到频繁二项集  $H$ , 则  $H \in L_{D-d}, L_2 = H \cup C$ , 清空  $H, C$ ; 将  $L_2$  中的单一项集合  $L$  作为连接项集, 与  $L_2$  组合, 得到三项集……。

d. 返回  $L_{D-d}$ 。

## 2.2 算法实现

$L_1 \in L_d; L = L_1$ ; // 第一步: 分离出欲减数据集  $d$ , 求频集  $L_d, L_1$  频繁 1 项集

for ( $k = 2; C_k \neq \emptyset; k++$ ) //  $C_k: L_{k-1}$  与  $L$  连接得到的候选  $k$  项集

{for all  $X \in C_k$  {得到频繁  $k$  项集  $L_k$ ;

$L_k \in L_d; L = \{L_k \text{ 中项目的单项集合} \}$  // 更新  $L$

for all  $X \in (L_D \cup L_d - L_d)$  // 第二步: 考虑  $L_D \cup L_d - L_D \cap L_d$  中的项集

$X \in L_{D-d}$ ;

扫描  $D-d$ , 得到  $L_1$ ; // 第三步: 考虑  $L_D \cap L_d$  中的项集

for all  $X \in (L_D \cap L_d)$  {

for each 所有项目包含的单一项  $x$  {

if ( $X.x.\text{count}_{D-d} \geq s * |D-d|$ )  $x \in M$ ; //  $X.x: X$  中的单一项

$M \in L_{D-d}$ ;

for each  $X \mid X.x \in M$

if ( $X.\text{count}_{D-d} \geq s * |D-d|$ )  $X \in N$ ;

$N \in L_{D-d}$ ; }

$L_1 \in L_{D-d}; L = L_1$ ; // 第四步: 考虑不属于集合  $L_D \cup L_d$  的部分

for ( $k = 2; C_k \neq \emptyset; k++$ ) { //  $C_k: L_{k-1}$  与  $L$  连接得到的候选项

for all  $X \in C_k$

{if ( $X \in L_{D-d}$ )  $X \in C$ ; //  $C$ : 已经确定过的频繁项集, 剪掉

else if ( $X \notin (L_d - N)$ )  $X \in C'_k$ ; //  $C'_k$ : 剪枝后的候选集

for all  $X \in C'_k$  // 确定候选集中的频繁项

if ( $X.\text{count}_{D-d} \geq s * |D-d|$ )  $X \in H$ ;

$H \in L_{D-d}; L_k = H \cup C; L = \{L_k \text{ 中项目的单项集合} \}$  // 更新  $L$

Answer =  $L_{D-d}$ ;

## 3 具体应用分析

假定有事务数据库  $D$  (支持度  $s = 40\%$ ) 如表 1 所示。

表 1 事务数据库  $D$

D			
TID	项目集	TID	项目集
1	ABCE	9	ABC
2	BCE	10	ACE
3	CDE	11	ACDE
4	ACD	12	BCE
5	ABCE	13	AE
6	ABC	14	ACD
7	ABD	15	AEF
8	CE		

$L_D = \{\{A\}, \{B\}, \{C\}, \{E\}, \{A, C\}, \{A, E\}, \{B, C\}, \{C, E\}\}$

(1) 分离出  $d$  (如表 2 所示) 得到  $d$  中每个项目支持事务集合及支持数, 使用项目增长法得到:

$L_d = \{\{A\}, \{C\}, \{D\}, \{E\}, \{A, C\}, \{A, D\}, \{A, E\}, \{C, D\}, \{C, E\}, \{A, C, D\}\}$ ;

表 2 欲减数据集  $d$

d	
TID	项目集
11	ACDE
12	BCE
13	AE
14	ACD
15	AEF

(2) 扫描  $D-d$ , 得到  $D-d$  中每个项目支持事务集合及支持数。

(3) 考虑  $L_D \cup L_d - L_D \cap L_d$ , 得到:  $L_D \cup L_d - L_D = \{\{B\}, \{B, C\}\} \in L_{D-d}$

(4) 考虑  $L_D \cap L_d$ , 得到:  $N = \{\{A\}, \{C\}, \{E\}, \{A, C\}, \{C, E\}\} \in L_{D-d}$

(5) 针对  $D-d$  数据集, 考虑不属于集合  $L_D \cup L_d$  的部分

$L_1 = \{\{A\}, \{B\}, \{C\}, \{E\}\} \in L_{D-d}$ , 当前  $L = \{\{A\}, \{B\}, \{C\}, \{E\}\}$

$C = \{\{A, C\}, \{B, C\}, \{C, E\}\}, C_2 = \{\{A, B\}, \{A, E\}, \{B, E\}\}$

$H = \{\{A, B\}\} \in L_{D-d}, L_2 = H \cup C = \{\{A, B\}, \{A, C\}, \{B, C\}, \{C, E\}\}$

…… $L_3 = H \cup C = \{\{A, B, C\}\}$

$L_{D-d} = \{\{A\}, \{B\}, \{C\}, \{E\}, \{A, B\}, \{A, C\}, \{B, C\}, \{C, E\}, \{A, B, C\}\}$

(下转第 54 页)

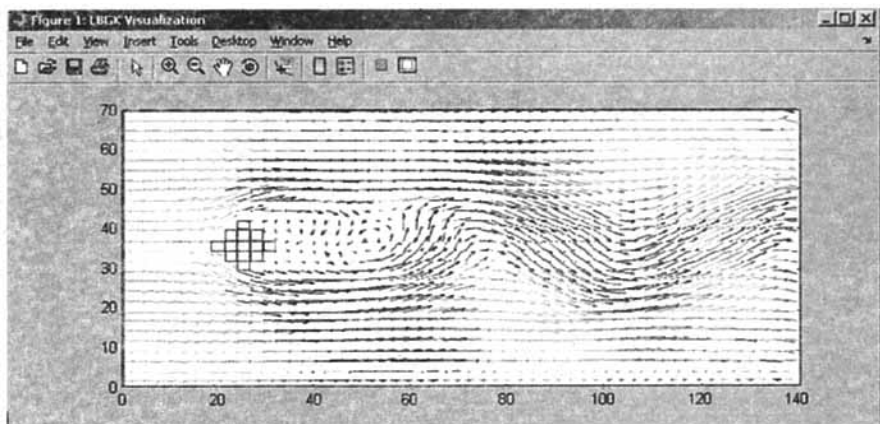


图 4 流场可视化结果

## 5 结 论

讨论了 LBGK D2Q9 模型在 Matlab 的分布式计算工具箱下计算与可视化的分布式并行处理方法及实现技术。实现了该模型的跟踪方式的可视化处理。利用 Matlab 的强大的科学运算和工具箱函数,便捷的接口的功能,高质量的图形可视化与界面设计,实现了 CFD 数值模拟中流场计算和可视化显示,对提高程序的运行效率、运算结果的分析效率以及改善分析计算的工作环境有着积极的意义。

### 参考文献:

- [1] 李晓梅,黄朝晖. 科学计算可视化导论[M]. 长沙: 国防科

技大学出版社,1996.

- [2] Qian Y, d'Humieres D, Lallemand P. Lattice BGK models for Navier - Stokes equation[J]. Europhys Lett,1992,17:479 - 484.  
 [3] Chen H, Chen S, Matthacus W H. Recovery of the Navier - Stokes equations using a lattice gas Boltzmann method[J]. Phys Rev A,1992,45:5339 - 5342.  
 [4] Li Q, Zheng C, Wang N, et al. LBGK simulations of Turing patterns in CIMA model[J]. J Scientific Computing,2001,16 (2):121 - 134.  
 [5] 郭照立,郑楚光,李 青,等. 流体动力学的格子 Boltzmann 方法[M]. 武汉:湖北科学技术出版社,2002.  
 [6] 张志勇. 精通 MATLAB5.3[M]. 北京:北京航空航天大学出版社,2000.

(上接第 50 页)

(6) 减后数据集的频繁项集为  $L_{D-d}$ 。

## 4 结束语

文中在对正负增量式关联规则更新技术<sup>[4-6]</sup>深入研究基础上,提出了相应的算法。通过分析可以看出:整个更新过程只扫描一次数据集;通过连接项集产生候选项,增强了产生候选项的针对性和有效性,提高了候选项的支持事务计数的效率;充分利用已知挖掘结果和正在挖掘过程中得到的信息,竭力获得无需计算支持数便已知的  $L_{D-d}$  中的项目,剪枝过程贯穿于整个算法,在算法的运行时耗上具有很明显的优势。历史数据集越大,剪枝效果越加明显,越能显现出本算法的优越性。由于本算法经常要判断一个项目集是否存在于另一个项目集中,可以在此问题上做一定的研究(如可以利用项集中项目有序的特点等),以进一步提高本算法的效率。

### 参考文献:

- [1] Han Jiawei, Kamber M. 数据挖掘概念与技术[M]. 北京:机械工业出版社,2001.  
 [2] DAVID W, CHEUNG J, NGVT, et al. Maintenance of Discovered Association Rules in Large Database: An Incremental Updating Technique[C]//In Proc 12th Int. Conf. on Data Engineering. New Orleans, Louisiana: IEEE Computer Society,1996:106 - 114.  
 [3] DAVID W, CHEUNG J, LEE S D, et al. A general Incremental Technique for Maintaining Discovered Association Rules [C]//In Proceedings of the Fifth International Conference on Database Systems for Advanced Applications. Melbourne, Australia:[s. n.],1997:185 - 194.  
 [4] 陈劲松,施小英. 一种关联规则增量更新算法[J]. 计算机工程,2002,28(7):106 - 107.  
 [5] 冯玉才,冯剑琳. 关联规则的增量式更新算法[J]. 软件学报,1998,9(4):301 - 306.  
 [6] 孙 浩,赵 霁. 一种关联规则增量更新算法[J]. 系统工程与电子技术,2004,26(5):676 - 677.