文章编号:1001 - 9081(2005)08 - 1856 - 02

基于概念格的关联规则算法

徐泉清,朱玉文,刘万春
(北京理工大学 信息科学技术学院,北京 100081)
(quanqingxu@163.com)

摘 要:对经典 Apriori 算法的优、缺点进行了剖析,在实际应用项目中,提出了一种基于概念格的关联规则算法 ACL (Apriori Algorithm Based On Concept Lattices)。在该算法中,引入了概念格和等价关系等概念,利用粗糙集相关方面的理论,计算得到频繁 2-项集 L_2。实验表明,ACL 算法是一种有效的快速的关联规则挖掘算法。

关键词:频繁项目集;支持度;信任度;概念格;等价关系
中图分类号: TP311.131 **文献标识码**:A

An efficient algorithm for mining association rules based on concept lattices

XU Quan-qing, ZHU Yu-wen, LIU Wan-chun
(*School of Information Science and Technology, Beijing Institute of Technology, Beijing* 100081, *China*)

Abstract: Apriori algorithm was researched and its merits and defects were analysed. A new association rule algorithm, Apriori algorithm based on concept lattices (ACL), was put forward. It adopted concept lattices and equivalence relationship, made use of the theory of rough sets, and then calculated to gain frequent 2-item L_2. Experiment results show that ACL is an efficient algorithm, and outperforms previous methods.

Key words: frequent itemset; support; confidence; concept lattice; equivalence relationship

0 引言

关联规则的挖掘是数据挖掘的一个重要研究方向。运用关联规则进行数据挖掘可以发现大量数据中的项集之间有趣的关联或相关联系。目前,关联规则的算法比较多,但大部分都是从经典算法 Apriori 演绎而来。文献[1]首先提出了在挖掘顾客事务数据库中项集间的关联规则问题,以后诸多的研究人员对关联规则的挖掘问题进行了大量的研究。其中文献[2]提出了一种新的方法,即基于经典的 Apriori 算法的频繁项目闭集挖掘方法:通过发现频繁闭项目集代替所有的频繁项目集,大大提高了挖掘的效率和准确性;文献[3]设计了一个基于划分的算法,它的做法是:先把数据库从逻辑上分成几个互不相交的块,每次单独考虑一个分块并对它生成所有的频集,然后把产生的频集合并,用来生成所有可能的频集,最后计算这些项集的支持度;文献[4]提出来基于 hash 的方法,是一种高效地产生频集的基于杂凑(hash)的算法。我们知道,寻找频集主要的计算是在生成频繁 2-项集 L_2 上,文献[4]就是利用了这个性质引入杂凑技术来改进产生频繁 2-项集的方法;文献[5]引入了修剪技术来减小候选集 C_k 的大小,也可以显著地改进 Apriori 算法的性能。

我们在项目中采用 Apriori 算法作为挖掘关联规则的基本算法,但发现深感经典 Apriori 算法有许多不足之处,因此提出了一个新的算法:ACL(Apriori algorithm based on Concept Lattices)算法。

1 关联规则基本概念

设 $I = \{i_1, i_2, \cdots i_m\}$ 是项的集合。设 D 是事务数据库的集合,其中每个事务 T 是项的集合,使得 $T \subseteq I$。每个事务有唯一的标识,如事务号,记作 TID。一个关联规则是形如 $A \Rightarrow B$ 的蕴涵式,这里 $A \subset I, B \subset I$,并且 $A \cap B = \varnothing$。

定义 1 规则的支持度

是事务集中包含 A 和 B 的事务数与所有事务数之比,记为 $support(A \Rightarrow B)$,即:

$$support(A \Rightarrow B) = P(A \cup B)$$

定义 2 规则的可信度

是指包含 A 和 B 的事务数与包含 A 的事务数之比,记为 $confidence(A \Rightarrow B)$,即:

$$confidence(A \Rightarrow B) = P(B \mid A)$$

定义 3 强关联规则

是指挖掘出支持度和可信度分别大于用户给定的最小支持度(min_supp) 和最小可信度(min_conf) 的关联规则。

定义 4 项集的出现频率

包含项集的事务数,简称项集的频率、支持计数或计数;

定义 5 频繁项集

如果项集的出现频率大于或等于 min_supp 与 D 中事务总数的乘积,则称它为频繁项集。

定理 1 频繁项集的所有非空子集都必须也是频繁的。

2 ACL 算法设计与实现

2.1 算法依据的主要性质

根据粗糙集相关性质和原理,设计了 ACL 算法。下面给出算法依据的性质和原理。

定义 6 概念格[6]

假定给定形式背景 为三元组 (T, I, R),其中 T 是事务集

收稿日期:2005 - 01 - 19;修订日期:2005 - 04 - 30

作者简介:徐泉清(1980 -),男,安徽东至人,硕士研究生,主要研究方向:数据挖掘、知识发现; 朱玉文(1950 -),女,四川成都人,副教授,主要研究方向:人工智能; 刘万春(1949 -),男,河北滦县人,教授,主要研究方向:模式识别.

合,I 是项目集合,R 是 T 和 I 之间的一个二元关系,则存在唯一的一个偏序集合与之对应,并且这个偏序集合产生一种格结构,称由(T,I,R) 所诱导的格 L 就称为一个概念格或 *Galois* 格。格 L 中的每个节点是一个序偶(即概念),记为(X,X'),其中,第一个分量 X 被称为此概念的外延,第二个分量 X' 被称为此概念的内涵。

定理 2　序偶$(X,X') \in P(O) \times P(D)$ 关于关系 R 是完备的,当且仅当满足下列两条性质:

(1)$X' = f(X)$,其中$f(X) = \{x' \in D \mid \forall x \in X, xRx'\}$;

(2)$X = g(X')$,其中$g(X') = \{x \in O \mid \forall x' \in X', xRx'\}$

定义 7　不可分辨(等价)关系

对于任何一个项目集合 $P \subseteq Q$,Q 为全体项目的集合,不可分辨关系用 IND 表示,定义如下:

$$IND(P) = \{(x,y) \in T \times T: f(x,a) = f(y,a)\ \forall a \in P\}$$

如果$(x,y) \in IND(P)$,则 x,y 称为相对于 P 是不可分辨的。针对项目集 P 上的不可分辨关系,T 可划分为几个等价类,用 T 的商集即 $T/IND(P)$ 表示。

2.2　算法设计思想

我们在研究 Apriori 算法时,发现生成频繁 2-项集 L_2 是算法的一个瓶颈,提出利用粗糙集理论来求 L_2。

在数据收集阶段,Apriori 算法不能辨别哪些项是相关的、哪些项是不重要的,认为所有项都是有用的,并全部存储,这大大增加了信息的存储量和处理量。然而,我们实际只对项目集中的某些子集感兴趣,通过对这些子集进行操作,就能达到目的,同时也减少了信息处理量,提高了效率。

求等价关系 $IND(I)$,采用概念格理论。由于概念格的结构特殊,使得在格上求等价关系相对简单。等价关系,即格 L 中包含项目集合 I 的节点(X,X') 的最小上界的并集。

$$IND(I) = \cap\ sup\{I \subseteq X', (X,X') \in L\}$$

下面给出在概念格 L 上求 *FrequentItem2*$(I,threshold)$ 的算法,在此算法中,用到了格结构对应的 Hasse 图[7] 概念,它们表示一种概念层次结构,本质上反映了 *E-R* 关系。算法描述:

FrequentItem2 (I, threshold)

输入:项目集 I,阈值 *threshold*

输出:结果集 R

1) 初始化:将结果集 R 置为 $\varnothing$,初始化格结构对应的 Hasse 图中各节点标志 $Flag = 0$;

2) 对于每一个 $H = (X, X') \in L$,X 不为 $\varnothing$,根据 $\| X' \|$ 升序,进行如下操作:

如果 $Flag(H)! = 1$ 并且 $I \subseteq Attrib(X')$,则遍历 Hasse 图中的每个节点 N,将 $Flag[N]$ 置为 1,同时将 X 加入到 R 中;

3) 遍历 R 中元素,两两一组,对于计数不小于 *threshold*,将其和其计数放入 L;

4) 返回 L;

2.3　算法描述

算法描述:

输入:事务数据库 D;最小支持度阈值 *min_supp*

输出:D 中的频繁项集 L

1) 对每一个事务 $t \in D$,对每一个 $item \in t$ 用 *itemid* 表示,寻找 D 中的频繁 1-项集 $L1$,生成 $L1$ 对应的概念格 L;

2) 对于频繁 1-项集 L_1,利用求 *FrequentItem2*(L_1, min_supp),得到频繁 2-项集 L_2;

3) 对于从 $k = 3$ 开始,一直到 L_{k-1} 不为空,作如下操作:

a) 根据 L_{k-1} 和最小支持度阈值 *min_supp* 产生候选集 C_k;

b) 对于每一个 $t \in D$,根据 C_k 得到候选集 t 的子集 C_t,并对每一个 $c \in C_t$ 进行计数 L_k;

c) 将 $c \in C_t$ 且 c 的计数大于等于 *min_supp* 加入到 L_k;

4) 返回 D 中的频繁项集 $L \cup_k L_k$;

2.4　算法实例

下面举个例子说明问题,从外存中读入事务数据库后,将项目集中的项目用项目标志(itemid) 唯一表示,即 $itemid(A) = 0$,$\cdots$,$itemid(F) = 5$。以下是 ACL 算法模拟执行(设最小事务支持计数为 2)。

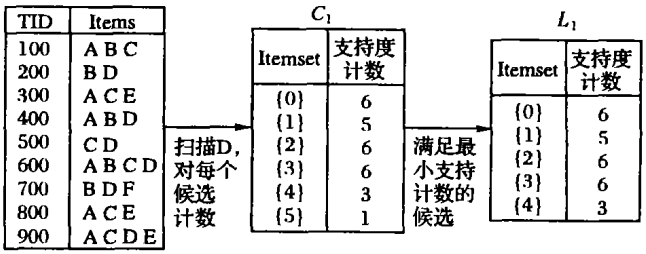

图 1　产生频繁 1-项集 L_1

利用在概念格上求 *FrequentItem2* (L_1, min_supp) 的算法,求出 $L_2 = \{\{0\ 1\}, \{0\ 2\}, \{0\ 3\}, \{0\ 4\}, \{1\ 2\}, \{1\ 3\}, \{2\ 3\}, \{2\ 4\}\}$。

表 1　利用 *FrequentItem2*(L_1, min_supp) 算法求 L_2

事务集	0	1	2	3	4	事务集	0	1	2	3	4
1	1	1	1	0	0	6	1	1	1	1	0
2	0	1	0	1	0	7	0	1	0	1	0
3	1	0	1	1	0	8	1	0	1	0	1
4	1	1	0	1	0	9	1	0	1	1	1
5	0	0	1	1	0						

最终表示:L_2

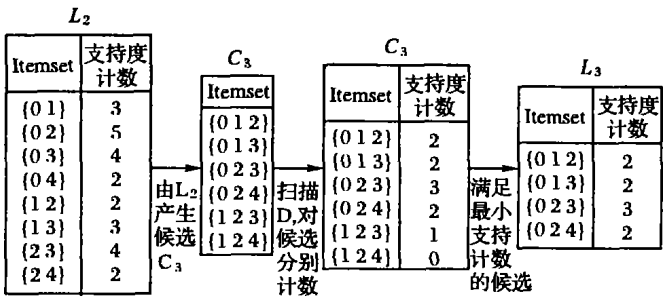

图 2　产生满足最小支持计数频繁 3-项集 L_3

2.5　算法性能分析

引入了粗糙集理论中的等价关系概念,利用在概念格上求 *FrequentItem2*(I, min_supp),代替 Apriori 算法中的求频繁 2-项集的集合 L_2 的过程。我们知道,Apriori 算法由频繁 1-项集的集合 L_1 计算频繁 2-项集的集合 L_2,其时间复杂度为 $O(m \times n^2)$,而采用在概念格上求 *FrequentItem2*(I, min_supp) 来计算得到 L_2,其时间复杂度仅为 $O(m \times n \times \log mn)$。如果仍然采用在概念格上求 *FrequentItem2*(I, min_supp),来计算得到 L_3,其时间复杂度较用 Apriori 算法不但不能下降,反而会大幅度地上升。这是因为,在 Apriori 算法中,由 L_1 到 L_2 是算法的瓶颈,由 L_2 到 L_3,$\cdots$,则 Apriori 算法的优势,可由定理 1 得知。

从以上的分析可以看到,结合粗糙集理论,很好地解决了由 L_1 到 L_2 这一 Apriori 算法的瓶颈,使得 ACL 算法性能比经典 Apriori 算法有了较大提高。

3　实验结果

下面仅对与经典 Apriori 算法不同之处进行了测试,即由频繁项目集 L_1 产生候选项目集 C_2,由 C_2 产生频繁项目集 L_2,

(下转第 1860 页)

难看出，x_3' 与精确值更远了。因此我们对这两种算法进行了合并，[11.5°,31.5°] 中采用优化后的弦割法，其他区间仍采用优化前的弦割法。优化后，平均迭代次数为 5.8779。

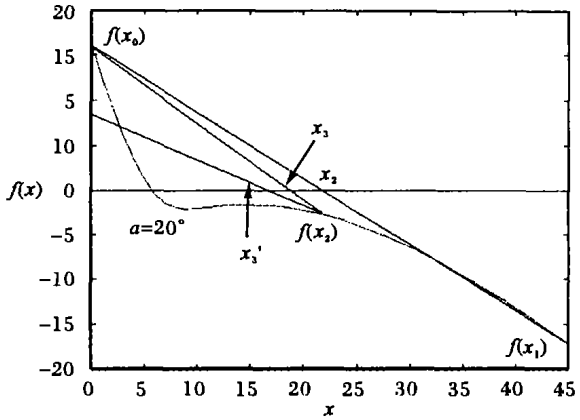

图6 球射出角度 20° 时的优化示意图

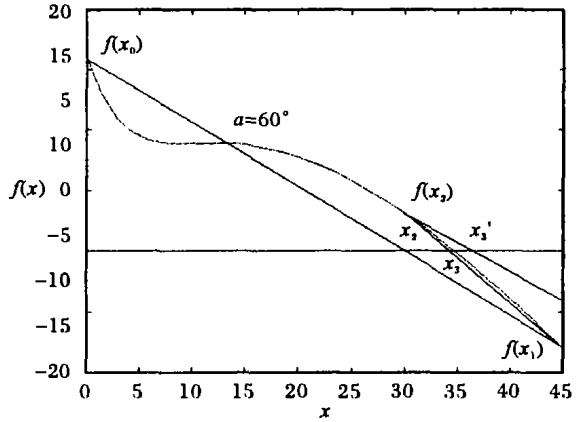

图7 球射出角度 60° 时的优化示意图

4 结语

从学习算法的角度来说，弦割法是一种搜索算法，它利用了人类常识作为启发信息，加速了搜索，它的物理意义比较直观，便于在决策中运用；在截球算法中，如果 $f(mrt) > 0$，认为失败。结合实际情况，认为该队员是追不上球的，即球员跑到球停止的地方所需时间超过了 45 个周期，认为截球失败，不进行计算，毕竟在比赛中，注重的是整体配合，由其他策略解决。当 $f(x) < 0$ 时，运动模型不是单调递减情况下，有可能出现多解的情况，因为多解情况仅限于特定条件，且出现情况不是很多，采用了其他算法来获取最佳解。通过实验发现由于方程 $f(x)$ 在 0 到 45 个周期内曲线斜率变化，在某些区间严重影响弦割法的收敛速度，造成震荡，最高迭代次数甚至达到 87。为此在迭代次数较高的区间对弦割法进行了优化。优化后，弦割法平均迭代次数 5.8779 次，效率比优化前弦割法快 37.3%，比二分法快 39.6%，很好满足了比赛截球要求。

致谢：东南大学李伟博士和南京邮电学院韦龙凤对本文的工作提供了好的建议和和思路，在此一并表示感谢。

参考文献：

[1] HIROKI K, MINORU A, YASUO K, *et al*. RoboCup: A Challenge Problem for AI and Robotics[A]. RoboCup-97: Robot Soccer World Cup I[C]. Berlin: Springer, 1998. 1 - 19.

[2] 李实，陈江. 清华机器人足球队的结构设计与实现[J]. 清华大学学报, 2001, 41(7): 94 - 97.

[3] Hagan mt. 神经网络设计[M]. 北京：机械工业出版社, 2002. 197 - 211.

[4] 菲利普斯，泰勒. 数值分析的理论及其应用[M]. 上海：上海科学技术出版社, 1972. 164 - 167.

[5] 何光渝. Visual C++ 常用数值算法集[M]. 北京：科学出版社, 2002. 528 - 530.

（上接第 1857 页）

这样一直下去，直到有某个 r 值使得 L_r 为空。对于刚开始，从外存中读入事务数据库，最后，写关联规则，经典 Apriori 算法与 ACL 算法一样。测试结果如表 2（设 support = 10%，confidence = 50%）。

表 2 ACL 算法与 Apriori 的比较

数据量	运行时间/s		数据量	运行时间/s	
	经典 Apriori	ACL 算法		经典 Apriori	ACL 算法
5000	0.53	0.04	20000	2.23	0.14
6000	0.65	0.05	50000	5.60	0.34
8000	0.88	0.06	80000	8.86	0.54
10000	1.10	0.07	100000	11.00	0.68
15000	1.68	0.11	150000	16.73	1.01

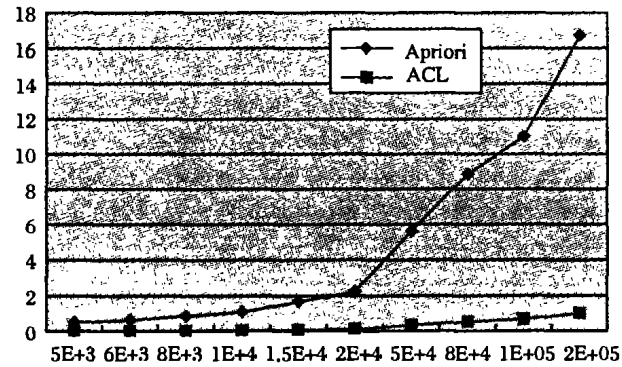

图3 两种算法运行时间曲线

以上测试是在 P4 1.6G，内存为 128M 的 PC 机上进行的。

参考文献：

[1] AGRAWAL R, IMIELINSKI T, SWAMI A. Mining association rules between sets of items in large databases[A]. Proceedings of the ACM SIGMOD Conference on Management of data[C]. Washington, DC, USA: ACM Press, 1993. 207 - 216.

[2] BASTIDE PY, TAOUIL R, LAKHAL L. Discovering frequent closed itemsets for association rules[A]. Proceedings of 7th International Conferece Database Theory (ICDT99)[C]. Jerusalem, Israel: Springer-Verlag, 1999. 398 - 416.

[3] SAVASERE A, OMIECINSKI E, NAVATHE S. An efficient algorithm for mining association rules in large databases[A]. Proceedings of the 21st International Conference on Very large Database[C]. San Francisco, CA, USA: Morgan Kaufmann, 1995. 432 - 444.

[4] PARK JS, CHEN MS, YU PS. An effective hash-based algorithm for mining association rules[A]. Proceedings of 1995 ACM-SIGMOD International Conference Management of Data[C]. San Jose: ACM Press, 1995. 175 - 186.

[5] KLEINBERG J, PAPADIMITRIOU C, RAGHAVAN P. Segmentation problems[A]. Proceedings of the 30th Annual Symposium on Theory of Computing[C]. NY, USA: ACM, 1998. 473 - 482.

[6] OOSTHUIZEN GD. Rough Sets and Concept Lattices[A]. Rough Sets, and Fuzzy Sets and Knowledge Discovery (RSKD'93)[C]. London: Springer-Verlag, 1993. 24 - 31.

[7] WILLE R. Restructuring lattice theory: An approach based on hierarchies of concepts[A]. Ordered Sets[C]. Dordrecht Boston: Reidel, 1982. 445 - 470.