基于剪枝概念格的项集知识表示与挖掘

胡学钢,刘 卫,王德兴
HU Xue-gang,LIU Wei,WANG De-xing

合肥工业大学 计算机与信息学院,合肥 230009
School of Computer and Information,Hefei University of Technology,Hefei 230009,China
E-mail:jsjxhuxg@hfut.edu.cn

HU Xue-gang,LIU Wei,WANG De-xing.Representation and mining of itemsets knowledge based on pruned concept lattice.Computer Engineering and Applications,2007,43(22):176-178.

Abstract: The relationship between concept lattice and frequent itemsets is discussed,then the model of Pruned Concept Lattice (PCL) is introduced to represent itemsets in the database,and the scale of itemsets is compressed efficiently.The infrequent concepts is pruned timely and dynamically during the PCL's construction according to apriori property.The efficiency of the algorithm is shown in the experiments.

Key words: data mining;association rules;itemsets;concept lattice

摘 要:在研究概念格和项集关系的基础上,将剪枝概念格模型引入数据库中项集的表示与挖掘,利用概念间的关系性质,在构造过程中及时、动态地剪枝,删除与项集求解无关的概念,不丢失信息的同时能有效压缩频繁项集的规模,实验证实了算法良好的性能。

关键词:数据挖掘;关联规则;项集;概念格
文章编号:1002-8331(2007)22-0176-03 **文献标识码**:A **中图分类号**:TP274

1 引言

关联规则挖掘是数据挖掘领域中重要的研究分支,项目集求解是关联规则挖掘的基础和前提。基于 Apriori[1]及其改进算法[2]求解频繁项目集是采用逐层搜索迭代,因而求解频繁项目集的效率是非常低的;而 FP-growth[3]是基于 FP-tree 求解频繁项目集,不产生候选项目集,将频繁项目集的数据库压缩到频繁模式树 FP-tree, 然后将压缩后的数据库分成一组条件数据库。虽然很多学者对 FP-growth 算法进行深入地研究,提出许多改进算法,但仍存在很多不足:

(1)在最小支持度阈值较小、频繁项目集较多的情况下,基于投影的超集检测是 FP-growth 算法的最大耗时操作;

(2)FP-growth 算法在递归构造条件 FP-tree 及其遍历时的 CPU 和存储开销都很大;

(3)求解频繁项目集时,需要进行投影和裁剪操作,效率不高,对于密集型或存在特别长模式的数据集,FP-growth 算法效率与可伸缩性较差。

基于概念格的频繁项集与关联规则挖掘,虽然已经有些学者对此进行了深入地研究[4-6],但很少有系统地探讨概念格与项目集、剪枝概念格与频繁项目集之间的关系。因此,鉴于概念格具有完备而紧凑的知识表示结构, 在不丢失有效信息的情况下,基于概念格的频繁项目集表示和求解,能有效地压缩频繁项目集的表示规模。

本文结构如下:第 2 章给出了基本定义;第 3 章描述了概念格与项目集的表示关系;第 4 章描述了剪枝概念格与频繁项集、最大项集、闭合项集的关系;第 5 章对算法进行了实验性能比较与分析;第 6 章总结了相关工作。

2 相关定义

定义 1[7] 假设给定的上下文 $C=(O,D,R)$,其中 O 是对象集合,D 是属性集合,R 是 O 和 D 之间的一个二元关系, 则存在唯一的一个偏序关系与之对应,并且这个偏序关系产生一个格结构,这种由 C 所诱导的格就称为一个概念格(或 Galois 格,简记为 GCL)。格 GCL 中的每个节点(A,B)是一个二元组(即概念),其中 A 是幂集 $P(O)$的元素,B 是幂集 $P(D)$的元素,A 和 B 按如下运算关系建立连接:

$A'=\{m\in D|gRm$ 对所有的 $g\in A\}$

$B'=\{g\in O|gRm$ 对所有的 $m\in B\}$

并且 $A'=B,B'=A$。A 和 B 分别称为概念的外延(extention)和内涵(intention)。用 $Ext(C)$表示概念 C 的外延,用 $Int(C)$表示概念 C 的内涵。

基金项目:安徽省自然科学基金(the Natural Science Foundation of Anhui Province of China under Grant No.050420207);国家自然科学基金(the National Natural Science Foundation of China under Grant No.050504F)。

作者简介:胡学钢(1961-),男,博士,教授,主要研究领域为数据挖掘、人工智能;刘卫(1983-),男,硕士研究生,主要研究领域为数据挖掘;王德兴(1968-),男,博士,讲师,主要研究领域为数据挖掘。

定义 2 如果两个概念 $C1=(A1,B1)$ 和 $C2=(A2,B2)$ 满足 $A1\subset A2$，则称 $C1$ 和 $C2$ 之间有子概念—超概念关系，表示为 $C1<^*C2$ 或 $C2\in sup(C1)$；$C1$ 和 $C2$ 之间的直接子概念—直接超概念定义为 $C1<C2$：如果 $C1<^*C2$，并且不存在概念 C，满足 $C1<^*C<^*C2$。

定义 3 对概念 $C=(A,B)$，如果 $|A|\geq \min\_sup$，则 B 是一个频繁项集，对应的概念 C 称为频繁概念，否则称为非频繁概念，概念 C 的支持度 $support(C)=|A|/|D|$，其中 $|A|$ 和 $|D|$ 分别是概念 C 的外延基数及事务数据库 D 中的事务数。

3　概念格与项目集的表示与求解

定义 4 对任意项目集 F，称 F' 为 F 的支持集。对任意一个概念 $C=(A,B)$，将支持集为 A 的所有项目集的集合记为 $FI(A)$，即 $FI(A)=\{B_i|B_i'=A\}$。

定义 5 对于给定概念格中概念 $C=(A,B)$，设 C 的所有直接超概念为 $C_1,C_2,\cdots,C_k$，称 $New(C)=Intent(C)-(Intent(C_1)\cup\cdots\cup Intent(C_k))$ 为概念 C 的初始内涵。

下面定理 1 和定理 2 讨论了项目集和概念内涵之间的对应关系，为此，先给出相关的符号描述和定义。

定理 1 由上下文所导出的任意项目集必定是某概念的内涵或内涵的子集。

证明 对任意项目集 B_0，若 $B_0=(B_0')'$，则 (B_0',B_0) 构成一个概念，B_0 作为概念的内涵；否则意味着 $(B_0')'$ 中元素还存在共同满足另外的特征 B_1，即 $\exists B_1\subset I$ 且 $B_1'=B_0'$，此时若 $(B_0\cup B_1)=(B_0')'$，则 $(B_0',B_0\cup B_1)$ 构成一个概念，B_0 作为概念的初始内涵，否则以此类推，总存在两两不同的项目集 $B_1,B_2,\cdots,B_k$，使 $(B_0\cup B_1\cup\cdots\cup B_k)=(B_0')'$ 成立，从而使 $(B_0',B_0\cup B_1\cup\cdots\cup B_k)$ 构成概念，B_0 作为概念的初始内涵，证毕。

由定理 1 可知，每个项目集一定作为概念格中某个概念的内涵或内涵的子集出现，因而能完整地描述项目集，任意项目集一定存在于概念格的某一概念中，因而没有丢失任何项目集。

定理 2 对任意一个概念 $C=(A,B)$，$FI(A)$ 的求解可分为如下 4 种情况：

(1) $\forall B_s\subseteq New(C)$，且 $B_s\neq\emptyset$，则 $B_s\in FI(A)$

(2) $\forall C_i(C<C_i$ 或 $C<^*C_i)$，$\forall B_s\subseteq New(C),B_s\neq\emptyset,fc_i\in FI(A_i)$，则 $(B_s\cup fc_i)\in FI(A)$

(3) $\forall C_i,C_j(inf(C_i,C_j)=C\wedge fc_i\in FI(A_i)\wedge fc_j\in FI(A_j))$，则 $(fc_i\cup fc_j)\in FI(A)$，其中 $inf(C_i,C_j)=C$ 表示概念 C_i,C_j 的最大的公共下界概念为 C。

(4) $\forall X,Y\in FI(A)$，则 $X\cup Y\in FI(A)$

证明 由定义 4 知，对任意概念 $C=(A,B)$，$FI(A)$ 中每个元素的支持集为 A，因此，只需证明上述 4 种情况下内涵的支持集也等于 A 即可：

(1)对 $New(C)$ 的任意非空子集 B_s，由于在 C 中是第一次出现，故 B_s 的支持集只能是 A，即 $B_s'=A$，证毕。

(2)由于 $(B_s\in New(C))'=A$，因此，$(B_s\cup fc_i\in FI(A_i))'=B_s'\cap fc_i'=A$，证毕。

(3) $(fc_i\cup fc_j)'=(fc_i)'\cap(fc_j)'=Ext(C_i)\cap Ext(C_j)=A$，证毕。

(4)因为 $X,Y\in FI(A)$，故 $X'=A,Y'=A,(X\cup Y)'=X'\cap Y'=A$，证毕。

定理 2 表明借助于概念之间的关系可从每个概念导出多个项目集，其中每个项目集在相应的概念格中都有惟一概念的内涵或其子集与之相对应，并且每个概念的内涵或其子集至少有一个项目集与之相对应；同理，对于不小于最小支持度阈值的项目集，作为频繁项目集，其每个频繁项目集在相应的概念格中都有惟一频繁概念的内涵或其子集与之相对应，并且每个频繁概念的内涵或其子集至少有一个频繁项目集与之相对应。

综上所述的研究得知，每个项目集一定作为概念格中某个概念的内涵或内涵的子集出现，因而能完整地描述项目集，而借助于概念之间的关系可从每个概念导出多个项目集，因此，从项目集到相应的概念格中概念的内涵或其内涵子集间存在满射关系；同样，从频繁项目集到相应的概念格中频繁概念的内涵或其内涵子集间存在满射关系，因此基于概念格的概念内涵表示的项目集，所需要保存的信息规模要明显小于事务数据库中项目集的规模，且不会损失其中蕴涵的任何有效信息，因而适用于大规模数据库中的关联规则挖掘。

4　基于剪枝概念格的项目集表示

4.1　剪枝概念格的频繁项集表示

完备的概念格中包含了频繁概念与非频繁概念，由于非频繁概念与频繁项目集的求解无关，因而可从概念格中删除，得到面向频繁项目集的剪枝概念格模型，即剪枝概念格是在概念格中删除非频繁概念(空概念除外)得到的概念格结构。同样，由剪枝概念格可计算出所有的频繁项目集。

由于剪枝概念格是在概念格中删除非频繁概念(空概念除外)得到的概念格结构，根据频繁项目集与概念格中的频繁概念间的关系可知：

每个频繁项目集在相应的剪枝概念格中都有惟一频繁概念的内涵或其子集与之相对应，并且剪枝概念格中每个频繁概念的内涵或其子集至少有一个项目集与之相对应。从频繁项目集到相应的剪枝概念格中频繁概念的内涵或其内涵子集间存在满射关系，所需要保存的信息规模要亦小于相应的基于概念格表示项目集的规模，且不会损失其中蕴涵的有效信息。

4.2　剪枝概念格的最大项目集表示

定义 6 设 $t_k\subset I$ 是事务的项目集，如果 $support(t_k)$ 不小于给定的最小支持度阈值 $\min\text{-}support$，则频繁项目全集 $FIL=\{t_k\subset I|Support(t_k)\geqslant\min\text{-}support\}$。

定义 7 给定频繁项目集 $t_k\subset FIL$，如果不存在频繁项目集 J，使得 $t_k\subset J$，则 t_k 是最大频繁项目集，即最大频繁项目集 $max\text{-}frequent\text{-}itemsets=\{t_k\subset FIL|\neg\ \exists J\in FIL,t_k\subset J\}$。

根据剪枝概念格的定义知，剪枝概念格中底部的空概念 (Φ,all) 的所有直接超概念 $(\Phi,all).direct\text{-}super\text{-}concept$ 都是最大频繁项目集，且不会丢失原有全集的信息。

4.3　剪枝概念格的闭合频繁项目集表示

由定义 7 和定理 1 和定理 2 可知，剪枝概念格中概念的内涵及其子概念的内涵是频繁闭合项目集，因此，基于剪枝概念格表示的项目集不用检查局部闭合频繁模式是否为全局闭合

频繁模式,减小了闭合项集挖掘算法在模式匹配的同时仍需检查局部频繁模式是否全局频繁闭合模式所花费的高昂代价。

5 实验分析

对 FP-growth 及其改进算法的性能研究表明,基于 FP-growth 求解频繁项目集比 Apriori 算法大约快一个数量级[8],因此,在输入相同的的项目集的情况下,选取 FP-growth 算法求解频繁项目集的时间与基于剪枝概念格挖掘频繁概念的时间进行对比,而不再与基于 Apriori 算法求解频繁项目集进行时间对比。

实验平台为 Intel Celeron(R)2.53 G、256 M 内存的 PC 机,算法实现环境为 Windows XP Professional,用 Java JDK 1.4.2 实现基于 FP-growth 和剪枝概念格求解频繁项目集的挖掘时间对比,这里使用 FP-growth 程序的来源是:http://www.adrem.ua.ac.be/~goethals/software/主页中:Fp-growth(Download)This implementation is based on the Fp-growth algorithm(cfr.Han et al.,2000),FP-growth 算法需要输入的项目集数据是由 IBM Almaden 实验室的数据生成器产生的,依次是 412、816、1 220、1 626、2 044、4 075、8 221、16 427、32 925 个项目集,其中输入的项目集中有 20 个属性,即 20 个项目,项目集间的关系概率是 30%,最小支持度阈值依次是 10%、30%、50%,根据 FP-growth 算法求解频繁项目集的执行时间如表 1 所示。

表 1 剪枝概念格与 FP-growth 挖掘频繁项目集时间对比表

支持度阈值	FP-growth 挖掘频繁项目集的执行时间/ms								
10%	484	547	797	1 063	1 297	2 968	8 344	47 844	283 875
30%	343	516	765	1 031	1 274	2 937	7 906	45 703	281 609
50%	281	500	719	969	1 250	2 875	7 859	44 750	281 500
输入项目集数	412	816	1 220	1 626	2 044	4 075	8 221	16 427	32 925
剪枝概念格的挖掘频繁项目集的时间	≤27 ms								

这里的执行时间是递归遍历 FP-tree 生成频繁项目集的时间,时间单位是 ms,其中执行时间不包括构造 FP-tree 的时间;表 1 中的剪枝概念格的频繁项目集求解时间是遍历剪枝概念格的时间,都小于 27 ms,这里并不包含构造剪枝概念格的时间。由表 1 的求解频繁项目集的时间对比表明在剪枝概念格和 FP-tree 构造完毕后,基于剪枝概念格求解频繁项目集的时间比 FP-growth 的挖掘效率要好得多。

6 结论

项目集求解是关联规则挖掘的基础,Apriori 和 FP-growth 算法是经典的频繁项目集求解算法,在关联规则挖掘领域居首要地位。本文在深入研究概念格的基础上,提出基于概念格的项目集表示与求解,即每个项目集一定作为概念格中某个概念的内涵或内涵的子集出现,因而能完整地描述项目集,借助于概念之间的关系可从每个概念导出多个项目集,并且在构建概念格的同时使用了动态剪枝策略,有效缩减了概念格的规模,因而能实现大规模的数据库中关联规则挖掘。

(收稿日期:2007 年 3 月)

参考文献:

[1] Agrawal R,Imielinski T,Swami A.Mining association rules between sets of items in large databases [C]//Proceedings of ACM SIGMOD Conference on Management of Data,Washington DC,May 1993:207-216.

[2] Park J S,Chen M S,Yu P S.An effective hash-based algorithm for mining association rules[C]//Proc 1995 ACM-SIGMOD Int Conf Management of Data,San Jose,CA,May 1995:175-186.

[3] Han J,Pei J,Yin Y.2000 Mining frequent patterns without candidate generation [C]//Proc 2000 ACM-SIGMOD Int Conf Management of Data(SIGMOD'00),2000:1-12.

[4] 胡可云,陆玉昌,石纯一.基于概念格的分类及关联规则的集成挖掘算法[J].软件学报,2000,11(11):1478-1484.

[5] 王德兴,胡学钢,刘晓平,等.基于概念格和 Apriori 算法的关联规则挖掘分析[J].合肥工业大学学报,2006,29(6):699-702.

[6] 胡学钢,王媛媛.一种基于约简概念格的关联规则快速求解算法[J].计算机工程与应用,2005,41(22):180-183.

[7] Wille R.Restructuring lattice theory:an approach based on hierarchies of concepts[M]//RivalI.Dordrecht:Reidel,1982:445-470.

[8] Han J,Kamber M.Data mining-concepts and techniques[M].Sam Francisco:Morgan Kaufmann Publishers,2001.

(上接 175 页)

在数据集 1、2 的实验结果中,使用的预期最低支持度 δ 只要小于或等于真正使用的最低支持度 min_*supp*,则隐藏失败百分比 *HFR* 都会是 0%,符合预期的结果。以 δ=40%、min_*supp*=45%为例,表示预期接收端会以支持度 40%进行挖掘,从而将敏感序列的支持度都降至 40%以下,所以只要接收端以不小于 40%的最低支持度进行挖掘,都挖掘不出敏感序列,因此隐藏失败百分比是 0%。

4 结束语

本文提出了一种新颖的序列模式隐藏算法。序列模式的隐藏不同于关联规则的隐藏,需将所有的支持删除掉,方能达到隐藏的效果。对此,以预期最低支持度 δ 去做隐藏的程度依据,以大于或等于预期最低支持度 δ 的最低支持度去进行挖掘。实验表明,SDRF 算法能做到有较低的意外隐藏百分比及变动比例,符合预期设想。(收稿日期:2006 年 11 月)

参考文献:

[1] Atallah M,Bertino E,Elmagarmid A,et al.Disclosure limitation of sensitive rules [C]//Proc of IEEE Knowledge and Data Engineering Workshop,Chicago,Illinois,1999:45-52.

[2] Agrawal R,Srikant R.Mining sequential patterns [C]//ICDE'95,Taipei,Taiwan,Mar 1995:3-14.

[3] Oliveira S R M,Zaiane O R.A framework for enforcing privacy in mining frequent pattern,TR02-13[R].Computer Science Department,University of Alberta,Canada,2002.

[4] Zahidul,Md.Islam,Ljiljana Brankovic.A framework for privacy preserving classification in data mining [C]//Proceedings of the Second Workshop on Australasian Information Security,Data Mining and Web Intelligence,and Software Internationalization,Jan 2004.