ISSN 1000-1239/CN 11-1777/TP
47(1): 72-80, 2010

一种挖掘压缩序列模式的有效算法

童咏昕[1] 张媛媛[2] 袁 玫[3] 马世龙[1] 余 丹[1] 赵 莉[1]
[1](北京航空航天大学软件开发环境国家重点实验室 北京 100191)
[2](电信科学技术研究院 北京 100191)
[3](北京联合大学信息学院 北京 100084)
(yxtong@nlsde.buaa.edu.cn)

An Efficient Algorithm for Mining Compressed Sequential Patterns

Tong Yongxin[1], Zhang Yuanyuan[2], Yuan Mei[3], Ma Shilong[1], Yu Dan[1], and Zhao Li[1]
[1](*State Key Laboratory of Software Development Environment*, *Beihang University*, *Beijing* 100191)
[2](*China Academy of Telecommunication Technology*, *Beijing* 100191)
[3](*College of Information*, *Beijing Union University*, *Beijing* 100084)

Abstract Mining frequent sequential patterns from sequence databases has been a central research topic in data mining and various efficient algorithms for mining sequential patterns have been proposed and studied. Recently, many researchers have not focused on the efficiency of sequential patterns mining algorithms, but have paid attention to how to make users understand the result set of sequential patterns easily, due to the huge number of frequent sequential patterns generated by the mining process. In this paper, the problem of compressing frequent sequential patterns is studied. Inspired by the ideas of compressing frequent itemsets, an algorithm, CFSP (compressing frequent sequential patterns), is developed to mine a few representative sequential patterns to express all the information of all frequent sequential patterns and eliminate a large number of redundant sequential patterns. The CFSP adopts a two-steps approach: in the first step, all closed sequential patterns as the candidate set of representative sequential patterns are obtained, and at the same time most of the representative sequential patterns are obtained; in the second step, finding the remaining representative sequential patterns takes only a little time. An empirical study with both real and synthetic data sets proves that the CFSP has good performance.

Key words mining sequential pattern; compression; frequent pattern mining; association rule; data mining

摘 要 从序列数据库中挖掘频繁序列模式是数据挖掘领域的一个中心研究主题,而且该领域已经提出和研究了各种有效的序列模式挖掘算法.由于在挖掘过程中会产生大量的频繁序列模式,最近许多研究者已经不再聚焦于序列模式挖掘算法的效率,而更关注于如何让用户更容易地理解序列模式的结果集.受压缩频繁项集思想的启发,提出了一种CFSP(compressing frequent sequential patterns)算法,其可挖掘出少量有代表性的序列模式来表达全部频繁序列模式的信息,并且清除了大量的冗余序列模式.CFSP是一种two-steps的算法:在第1步,其获得了全部闭序列模式作为有代表性序列模式的候选集,与此同时还得到大多数的有代表性模式;在第2步,该算法只花费了少量的时间去发现剩余的有代表性序列模式.一个采用真实数据集与模拟数据集的实验研究也证明了CFSP算法具有高效性.

关键词 挖掘序列模式;压缩;频繁模式挖掘;关联规则;数据挖掘

中图法分类号 TP311.13

收稿日期:2009-06-26;**修回日期**:2009-10-12

基金项目:国家"九七三"重点基础研究发展计划基金项目(2005CB321902);北京市教委科技计划基金项目(KM200911417003)

由于频繁序列模式挖掘技术存在巨大的应用价值(例如网络日志分析、在 DNA 或蛋白质序列中进行频繁序列模式分析、基于程序执行跟踪分析程序 bug 等),故其成为数据挖掘研究中的重要主题之一.为了解决各类频繁序列挖掘问题,人们已经提出了许多有效的解决方案,例如,频繁序列模式挖掘[1-3]、极大序列模式挖掘[4]、闭序列模式挖掘[5-6]、top-k 闭序列模式挖掘算法[7]、基于大项集重用的序列模式挖掘[8]、基于约束的序列模式挖掘[9]和频繁偏序挖掘[10]等.因为已提出了众多有效的算法,故频繁序列模式挖掘算法的效率已不再是该研究领域最重要的挑战,取而代之的是应如何理解挖掘出的频繁序列模式结果集(为叙述方便,下面将频繁序列模式均简称为序列模式).

为使序列模式的挖掘结果易于理解,自然希望结果集的规模较小,仍可表达全部序列模式的信息.但 Agrawal 等人提出序列模式挖掘概念时主要依据于 Apriori 性质[11],即若序列模式 P 是频繁的,则 P 的全部子序列模式也都频繁,反之亦然.虽然 Apriori 性质可在搜索序列模式的过程中进行有效剪枝,但同时也导致了序列模式结果集的规模呈现指数爆炸的问题,因此不易于用户理解挖掘出的序列模式.为了解决该问题,目前主要提出了两类技术,即挖掘极大序列模式和挖掘闭序列模式.其中极大序列模式是指若一个频繁序列模式是极大序列模式,当且仅当该序列模式的任何超序列模式都不是频繁序列模式.挖掘极大序列模式是对全部频繁序列模式结果集进行了一种有损的压缩,其损失了所有极大序列模式的子序列模式的支持度信息,因此该方法不能表达出频繁序列模式结果集的完整信息.而另一类技术挖掘闭序列模式是指,若一个频繁序列模式是闭序列模式,当且仅当该序列模式的任何超序列模式的支持度都与其不相等.挖掘闭序列模式是对全部频繁序列模式结果集进行一种无损的压缩,其保存了频繁序列模式结果集的全部信息(包括序列模式的表达方式与模式的支持度这两种信息).虽然挖掘极大序列模式和挖掘闭序列模式的技术可以部分地缓解频繁序列模式结果集规模呈指数爆炸的问题,但由于闭序列模式的定义过于严格,因此当支持度阈值较小时闭序列模式的结果集依然很庞大,而极大序列模式虽然较大幅度压缩了结果集规模,但也损失了大量子序列的支持度信息.因此能否结合这两种技术的优点提出一种折中的方法,即由少量有代表性的序列模式组成一个集合,仍表达出全部序列模式的信息呢?

最近,在频繁项集挖掘的研究中遭遇了类似的问题[12-14],并已提出一些有效地压缩频繁项集算法,例如 RPglobal 算法和 RPlocal 算法[13].因此一个自然的想法是能否扩展压缩频繁项集的方法来解决压缩序列模式的问题.但如下两点原因使得扩展无法直接有效地进行.第一,在压缩项集的过程中采用 one-step 算法可以高效地得到一个较为精准的有代表性项集的集合,这是因为项集自身具有无序性的特点,使得判断每个项集是否具有代表性时只需在某个固定的搜索范围内进行.但序列模式由于自身的有序性,使得搜索树的规模趋近于一棵满叉树,无法直接在某固定范围内对序列模式进行是否具有代表性的判断.第二,采用的类似于 RPglobal 这类 two-steps 算法虽然可以直接扩展到压缩序列模式的问题,但是由于序列模式自身的有序性特点,使得判断过程效率极低,甚至在支持度阈值较低的情况下无法在一个合理时间范围内得到压缩结果.因此,能否设计一种求解压缩序列模式算法,使得其在运行效率上接近于压缩项集中的 one-step 算法,而在压缩准确程度上接近于 two-steps 算法?此处所谓 one-step 的方法是指从数据库直接得到目标结果集,无需在中间产生某个候选集.而 two-steps 的方法无法直接从数据库获得目标结果集,需先产生某个候选集方可获得目标结果集.

本文提出了一种高效的 two-steps 算法 CFSP (compressing frequent sequential patterns)来解决压缩序列模式的问题.虽然 CFSP 算法采用了 two-steps 的方式,但是在第 1 步挖掘全部闭序列模式的过程中,该算法采用 δ-dominate 序列模式检测机制同步挖掘出大部分有代表性序列模式,而在第 2 步只需要挖掘少量剩余的有代表性序列模式即可.所以在运行时间上该算法接近于一般的闭序列模式挖掘算法,因此达到了近乎于 one-step 方式的效率.与已有的研究工作相比,本文工作如下:

1) 介绍了压缩序列模式的问题,即挖掘出一个规模较小的有代表性序列模式的集合来表达全部序列模式结果集的信息.类比于压缩项集的问题,我们扩展了项集间的距离,提出了序列模式间的距离作为聚类的距离标准.并将选择有代表性的序列模式形式化为 δ-序列覆盖的概念,最后将压缩序列模式的问题规约为最小集合覆盖这个 NP-Hard 问题.

2) 提出了一种 δ-dominate 序列模式检测机制,并从理论与实现两个角度阐述了该机制可以在挖掘闭序列模式的同时产生出大多数有代表性的序列模式,从而大幅度提高了压缩序列模式的算法效率.

3) 基于 δ-dominate 序列模式检测机制,设计出一种 two-steps 的压缩序列模式挖掘算法 CFSP,

其既保证高的压缩准确程度,还具有近似于 one-step 方式的算法效率.并通过时间复杂性的理论分析和实验对比两方面证明了这一特点.

1 背景知识与相关工作

1.1 背景知识

设 $I=\{i_1,i_2,\cdots,i_n\}$ 为 n 个项的集合.一个序列 S 是一个有序的事件列表,记作 $\langle e_1,e_2,\cdots,e_m\rangle$,其中 e_i 是一个项集,即 $e_i\in I(1\leqslant i\leqslant m)$.也可对序列 S 简记为 $e_1e_2\cdots e_m$.此外,一个项可以多次出现于不同的多个事件中,一个序列中项的数目被称为序列的长度.当一个序列长度为 l 时也可称其为 l-序列.

定义 1. 序列间的包含关系.一个序列 $S_a=a_1a_2\cdots a_m$ 包含另一个序列 $S_b=a_1a_2\cdots a_n$,当且仅当存在一组整数 $1\leqslant i_1<i_2<\cdots<i_n\leqslant m$,使得 $a_1=b_{i_1},a_2=b_{i_2},\cdots,a_n=b_{i_n}$.

定义 2. 子序列/超序列.如果一个序列 S_a 包含另一个序列 S_b,则称 S_b 是 S_a 的子序列.反之称 S_a 是 S_b 的超序列.

对于一个给定的序列数据库 $SeqDB$,可将其视为一个二元组 (Sid,S) 的集合,其中 Sid 是一个序列的标示符,S 是该序列具体的内容,而该集合中二元组的数量被称为该序列数据库的规模,记作 $|SeqDB|$.

定义 3. 绝对支持度/相对支持度.给定一个序列数据库 $SeqDB$ 和一个序列 S_a,则序列 S_a 的绝对支持度是 $SeqDB$ 中包含 S_a 的二元组个数,记作 $sup(S_a)$.序列 S_a 的相对支持度是 $SeqDB$ 中包含 S_a 的二元组个数与 $SeqDB$ 所有二元组个数的比率,记作 $sup(S_a)/|SeqDB|$.

定义 4. 频繁序列模式.给定一个序列数据库 $SeqDB$ 和一个最小支持度阈值 min_sup,如果一个序列 S_a 的绝对支持度 $sup(S_a)\geqslant min\_sup$,则称 S_a 为频繁序列或频繁序列模式,简称为序列模式.

定义 5. 闭序列模式.如果 S_a 是一个频繁序列模式,且不存在序列模式 S_b,使得 $S_a\subset S_b$,且 $sup(S_a)=sup(S_b)$,则称 S_a 是闭序列模式.

1.2 相关工作

下面将对序列模式挖掘、压缩频繁模式和压缩的序列模式挖掘 3 类问题的相关研究进行介绍.

首先,Agrawal 等人最早提出序列模式挖掘的概念[1],随后产生了大量关于序列模式挖掘的有效算法,其中有代表性的有 PrefixSpan[2],SPADE[3] 等.但是由于序列模式挖掘基于 Apriori 特性,其导致了挖掘出的频繁序列模式的结果集呈指数爆炸的规模,严重地影响了频繁序列模式结果集的可用性.当前已提出两种方法来部分地解决该问题,即挖掘极大序列模式和挖掘闭序列模式.挖掘极大序列模式的思想来源自挖掘极大项集[4],所挖掘出的极大序列模式的任何超序列模式都不是频繁序列模式.其有代表性的算法有 MSPS[4] 等.同理,挖掘闭序列模式的思想来源自挖掘闭项集[5-6],所挖掘出的闭序列模式的任何超序列模式的支持度都小于该闭序列模式的支持度,其有代表性的算法有 CloSpan[5],BIDE[6] 等.虽然挖掘极大序列模式和挖掘闭序列模式的技术可部分地缓解频繁序列模式结果集规模呈指数爆炸的问题,但挖掘极大序列模式损失了极大序列模式所有子序列模式的支持度信息.而挖掘闭序列模式的定义过于严格.当支持度阈值较小时闭序列模式结果集仍很庞大.

其次,挖掘频繁项集的研究也遭遇过挖掘结果集呈指数爆炸规模的问题,除了挖掘极大项集和挖掘闭项集的方法外,最近提出了一些压缩的频繁项集挖掘策略[12-14].其中有代表性的方法有概括项集模式[14]和压缩频繁项集[13].概括项集模式采用一种称为概要模式的概念,从全部频繁项集中找出 K 个最具有代表性的项集来概括整个频繁项集的集合.而压缩频繁项集通过定义频繁项集两两之间的距离,并通过聚类的思想将频繁项集分成组内最大距离为 δ 的不同组,随后从每个组中选出一个项集组成有代表性项集的集合.该方法还提出了两种不同的实现算法 RPglobal 与 RPlocal,前者采用 two-steps 策略,即先挖掘出全部频繁项集,再从中得出有代表性的项集.该算法可以得到全局最优的压缩结果,但是计算的时间复杂度过高.后者采用 one-step 策略,即直接从给定的数据库中挖掘出有代表性的项集.虽然算法只得到了局部最优的压缩结果,但执行效率大幅提高.

最后,目前关于压缩序列模式问题的研究尚不充分,其中有代表性的研究是 CSP 算法[15],其扩展了概括项集模式的方法,提出了在序列模式中的概要模式的概念,并挖掘出 K 个最有代表性的序列模式.

2 基于序列覆盖的形式化建模

本节将对压缩序列模式的问题进行形式化建模:首先,从压缩频繁项集的问题中得到启发,提出了序列模式间的距离与 δ-序列覆盖的概念;随后介绍了一种 δ-dominate 序列模式检测机制,其可以在挖掘闭序列模式的同时检测出大多数有代表性的序

列模式;最后,压缩序列模式的问题被规约为最小集合覆盖这个 NP-Hard 问题.

如果一个序列模式 SP_1 可以代表另一个序列模式 SP_2,则 SP_1 与 SP_2 之间必然会存在较大的相似程度.在此,基于任意两项集间衡量相似程度的概念[13],提出一种序列模式间的 Jaccard 距离.

定义 6. 序列模式间的距离.给定一个序列数据库 $SeqDB$ 和从该数据库中挖掘出的两个序列模式 SP_1 与 SP_2.SP_1 与 SP_2 之间的距离如下所示:

$$D(SP_1,SP_2)=1-\frac{|S(SP_1)\cap S(SP_2)|}{|S(SP_1)\cup S(SP_2)|},$$

其中 $S(SP_i)$为 $SeqDB$ 中包含序列模式 SP_i 的序列的集合,$|S(SP_i)|$为该集合的势.

根据上述序列模式间距离的定义可观察出:如果一个序列模式 RSP 可合理地代表一个序列模式的集合 $\{SP_1,SP_2,\cdots,SP_n\}$,则 $SP_i\subseteq RSP$,且 $D(SP_i,RSP)\leqslant\varepsilon(1\leqslant i\leqslant n)$,$\varepsilon$ 是一很小的正实数.由此可定义出 δ-序列覆盖.δ 代表用户给定的一个序列模式可代表另一序列模式时两者距离的阈值.

定义 7. δ-序列覆盖.一个序列模式 RSP 可以 δ-序列覆盖另一个序列模式 SP,当且仅当 $SP\subseteq RSP$,且 $D(SP,RSP)\leqslant\delta(\delta\in[0,1])$.

由 δ-序列覆盖的定义可知,如果一个序列模式 RSP δ-序列覆盖另一个序列模式 SP,则 SP 一定会是 RSP 的子序列.故可简化序列模式间的距离公式为

$$D(RSP,SP)=1-\frac{|S(RSP)\cap S(SP)|}{|S(RSP)\cup S(SP)|}=1-\frac{sup\ (RSP)}{sup\ (SP)}.$$

虽然依据 δ-序列覆盖的定义可简化两序列模式间距离的计算难度,但是仍需计算任意两个序列模式间的距离才可判断两者是否存在 δ-序列覆盖关系,这将导致巨大的计算开销.能否提出一种高效的检测机制既加速 δ-序列覆盖的判断过程,又保证判断的准确性?下面介绍一种 δ-dominate 序列模式检测机制,可准确且高效地完成 δ-序列覆盖的判断过程.

定义 8. 最小序列覆盖.给定一个序列模式的集合 S,S 中的任意一个序列模式 SP 的最小序列覆盖 $MSC(SP)$定义如下:

$$MSC(SP)\begin{cases}\min\{D(SP,SP_i)\mid\forall SP_i\in S,\\ \quad SP\subseteq SP_i\},SP_i\in S,\\ +\infty,\forall SP_i\notin S.\end{cases}$$

定义 9. δ-dominate 序列模式.给定一个序列模式的集合 S,S 中的任意一个序列模式 SP 是一个 δ-dominate 序列模式,当且仅当 $MSC(SP)>\delta$.

定理 1. δ-dominate 序列模式检测机制.给定一个序列模式的集合 S,如果 S 中的一个序列模式 SP 是 δ-dominate 序列模式,则 S 中的任何序列模式均不能 δ-序列覆盖 SP.

证明.根据 δ-dominate 序列模式的定义,SP 是 δ-dominate 序列模式,则 $MSC(SP)>\delta$.则 S 中任意序列模式与 SP 距离中的最小值都大于阈值 δ,即 S 中的任意序列模式与 SP 的距离都大于 δ,故 S 中的任何序列模式均不能 δ-序列覆盖 SP. 证毕.

例 1 将进一步解释 δ-dominate 序列模式与 δ-dominate 序列模式检测机制的含义.

例 1. 给定一个频繁序列模式的集合 $S=\{(AB:100),(AC:80),(AD:60),(ABC:60),(ABD:50),(ABCD:20),(BABCD:5)\}$和距离阈值 $\delta=0.5$.其中括号内为一个序列模式和它相应的支持度,模式与其支持度之间用冒号分割.在该集合中,序列模式 AB 存在着多个超序列,如 ABC,$ABCD$,$BABCD$.根据最小序列覆盖与 δ-dominate 序列模式的定义可得 $MSC(AB)=D(AB,ABC)=1-60/100=1-0.6=0.4<0.5$,则 AB 不是 0.5-dominate 序列模式.而对于序列模式 $ABCD$,其只存在唯一的超序列 $BABCD$,计算 $MSC(ABCD)=D(ABCD,BABCD)=1-5/20=1-0.25=0.75>0.5$,则 $ABCD$ 是 0.5-dominate 序列模式,即在 S 中没有任何序列模式可以 0.5-序列覆盖 $ABCD$.

根据上述定义和定理,下面给出压缩序列模式的形式化定义和 δ-dominate 序列模式检测机制在压缩序列模式问题中完备性的定理.

定义 10. 压缩序列模式.给定一个序列数据库 $SeqDB$、最小支持度阈值 min_sup 和最小距离阈值 δ.压缩序列模式的问题旨在发现一个有代表性序列模式的集合 RSP,令基于 min_sup 所挖掘出的每个频繁序列模式都会被 RSP 中的一个有代表性的序列模式所 δ-序列覆盖,同时使得 RSP 集合的规模最小.

定理 2. δ-dominate 序列模式检测机制的完备性.给定一个序列数据库 $SeqDB$、最小支持度阈值 min_sup、集合 S(根据 min_sup 从 $SeqDB$ 中挖掘出的全部频繁序列模式的集合),RS 是任意一个可 δ-序列覆盖 S 的有代表性序列模式的集合,则 RS 必包含 S 中的全部 δ-dominate 序列模式.

证明.根据 δ-dominate 序列模式的定义,令 SP 是集合 S 中的任意一个 δ-dominate 序列模式,则 $MSC(SP)$必大于 δ,即 SP 与任何一个它的超序列之间的距离都大于 δ.故 S 中任何一个序列模式都

不能δ-序列覆盖SP,故任何一个可δ-序列覆盖S的有代表性的序列模式的集合RS都一定会包含SP. 证毕.

定理 3. 压缩序列模式问题计算复杂性. 压缩序列模式的问题是一个 NP-Hard 问题.

证明略,请参考文献[13]中类似的证明.

通过对压缩序列模式问题的复杂性δ-dominate序列模式检测机制完备性的讨论可知,如何有效地求解该问题成为关键所在. 又从定义 9 可知,检测δ-dominate序列模式的主要计算开销用于计算最小序列覆盖,下面给出两种新型的最小序列覆盖计算策略,可高效地检测出δ-dominate序列模式.

定义 11. Forward 扩展的最小序列覆盖. 给定一个序列数据库$SeqDB$和一个序列模式SP,在以SP为前缀的投影数据库中,与SP模式间距离最小的距离是 Forward 扩展的最小序列覆盖,记为$FMSC(SP)$.

定义 12. Backward 扩展的最小序列覆盖. 给定一个序列数据库$SeqDB$和一个序列模式SP,在SP的全部前缀序列最大周期中[6],与SP模式间距离最小的距离是 Backward 扩展的最小序列覆盖,记为$BMSC(SP)$.

3 CFSP 算法

通过上节的分析可知压缩序列模式的问题是发现一个有代表性序列模式的集合,该集合可δ-序列覆盖全部频繁序列模式,同时使该集合的规模最小.

本节将详细介绍 CFSP(compressing frequent sequential patterns)算法,该算法可高效地解决压缩序列模式的问题. 其采用 two-steps 的方式,且依据δ-dominate序列模式检测机制可在第 1 步挖掘闭序列模式的同时获得大部分δ-dominate序列模式,即有代表性的序列模式. 随后第 2 步只需进行很少的有代表性序列模式的检测. 此外,在算法实现上CFSP 算法将δ-dominate序列模式检测机制嵌入经典的闭序列模式挖掘算法 BIDE[6]中. 第 3.1 节先介绍 CFSP 算法的设计;第 3.2 节将对该算法核心部分进行时间复杂度的分析,从理论上证明该算法的高效性.

3.1 CFSP 算法的设计与实现

算法 1. CFSP.

输入:序列数据库$SeqDB=\{S_1,S_2,\cdots,S_n\}$;最小支持度阈值$min_sup$;距离阈值$\delta$;

输出:一个有代表性序列模式的集合RSP.

① E←数据库$SeqDB$中全部频繁 1-序列模式;$Dominate$←∅;$Closed$←∅;

② for each 1-SFP in E do

③ $SeqDBspi$←1-SFP 的投影数据库

④ if(用$BIDE$的$BackScan$检测不可剪枝)

⑤ $BMSC$←该序列模式$Backward$扩展的MSC(1-SFP)(定义 12);

⑥ else

⑦ $BMSC$←0

⑧ call $MDP(SeqDB_{\text{-}SFP},1\text{-}SFP,min\_sup,BMSC,\delta,Dominate,Closed)$;

⑨ $Covered$←$Closed$-$Dominate$;

⑩ RSP←$Compress(Dominate,Covered)$;

⑪ return RSP.

下面首先介绍 CFSP 算法,具体伪代码如算法 1 所示. 第 1 步,CFSP 算法获得全部 1-序列模式集合E,并将δ-dominate序列模式和闭序列模式的集合初始化为空;随后第 2～4 步,算法采用深度优先搜索策略,从每个 1-序列模式开始先计算出 Backward 扩展的最小序列覆盖$BMSC$,再调用 MDP 算法(算法 2)进行深度优先搜索;算法第 5 步将闭序列模式中除δ-dominate序列模式外的其他闭序列模式保存在$Covered$集合中. 然后调用 Compress 算法(算法 3)挖掘出全部压缩序列模式,并最终返回一个有代表性的序列模式集合.

算法 2 采用深度优先的搜索策略,递归地获得δ-dominate序列模式和闭序列模式. 其中第 1 步构造了当前序列模式SP为前缀的投影数据库,并获得其中的频繁项;第 2 步计算 Forward 扩展的最小序列覆盖 FMSC;算法第 3 步采用 BIDE 算法判断当前模式是否为闭序列模式;随后在第 4～5 步,判断 Forward 扩展的最小序列覆盖或 Backward 扩展的最小序列覆盖是否至少有一个大于δ,即SP为δ-dominate序列模式,并保存在$Dominate$集合中. 算法第 6～9 步则深度优先地递归该检测过程.

算法 2. MDP(mining δ-dominate patterns).

输入:序列数据库$SeqDB=\{S_1,S_2,\cdots,S_n\}$;序列模式$SP$;最小支持度阈值$min_sup$;Backward 扩展的最小序列覆盖$BMSC$;距离阈值$\delta$;$\delta$-dominate序列模式的集合$Dominate$;闭序列模式的集合$Closed$;

输出:无返回值.

① LE←SP为前缀的投影数据库中的频繁项;

② if(用 BIDE 算法检测出 SP 为闭序列模式)
③ $Closed \leftarrow Closed \cup SP$;
④ $FMSC \leftarrow SP$ 进行 Forward 扩展的 $MSC(SP)$(定义 11);
⑤ if($BMSC > \delta$ or $FMSC > \delta$)
⑥ $Dominate \leftarrow SP$;
⑦ for each FI in LE do
⑧ $SP_i = \langle SP, FI \rangle$
⑨ $SeqDB_{spi}$ 为 SP_i 的投影数据库
⑩ if(用 BIDE 的 BackScan 检测不可剪枝)
⑪ $BMSC \leftarrow SP_i$ 进行 Backward 扩展的 $MSC(SP_i)$(定义 12);
⑫ else
⑬ $BMSC \leftarrow 0$
⑭ call $MDP(SeqDB_{spi}, SP_i, min\_sup, \text{BIDE}, \delta, Dominate, Closed)$.

算法 3 是最终挖掘有代表性模式的过程. 由定理 2 可知 δ-dominate 序列模式必为有代表性序列模式,此为算法第 1 步. 第 2～3 步发现不能被 δ-dominate 序列模式进行 δ-序列模式的闭序列模式,并保存在 T 集合中. 第 4～8 步采用贪心策略,迭代选择出 $Covered$ 集合中能 δ-序列覆盖 T 集合中闭序列模式最多的模式作为新的有代表性序列模式并加入 RSP 集合中,删除被覆盖模式,直到 T 集合为空.

算法 3. Compress.

输入:δ-dominate 序列模式集合 $Dominate$;其他闭序列模式集合 $Covered$;

输出:一个有代表性序列模式的集合 RSP.

① $RSP \leftarrow RSP \cup Dominate$;
② for each $P \in Covered$ do
③ $T \leftarrow$不能被 $Dominate$ 集合中元素 δ-序列覆盖的 P;
④ while $T \neq \varnothing$ do
⑤ 选择一个序列模式 $CR \in Covered$,使得 CR 可以 δ-序列覆盖的 T 中序列模式的数目最大;
⑥ $RSP \leftarrow RSP \cup CR$;
⑦ 从 $Covered$ 删除 CR,从 T 中删除被 $CR\delta$-序列覆盖的序列模式;
⑧ return RSP.

3.2 时间复杂度分析

由第 3.1 节已知 MDP 算法的时间复杂度与经典的闭序列挖掘算法 BIDE[6] 的复杂度相似,在此不做赘述. CFSP 算法与传统的闭序列挖掘算法的主要区别在于 Compress 算法,本节主要分析了 Compress 算法的时间复杂度.

引理 1. 算法 Compress 的时间复杂度为 $|Dominate| \cdot |Covered| + |T| \cdot |Covered| + O(\sum_{CR \in Dominate\text{-}covered} Set(R))$.

4 实验与性能分析

4.1 测试环境与数据集

在本节我们将详细地报告在真实数据集和模拟数据集上算法测试的情况. 所有的实验都是在联想 ThinkPad T60 电脑上运行,其 CPU 为 Inter 4200,内存为 1 GB,操作系统为 Windows XP Professional.

所有的算法都是在 Microsoft Visual C++ 6.0 环境下实现,我们将 CFSP 算法与当今最快的闭序列模式挖掘算法 BIDE[6],以及另一个压缩频繁模式算法 RPglobal[13] 进行在真实数据集与模式数据集上分别进行比较. 下面我们将分别介绍这些数据集.

第 1 个数据集是 *Gazelle*,这是一个过去序列模式挖掘研究中一直作为 benchmark 的数据集. 其包含 29369 个来自客户的 Web 点击流序列和 1423 个不同的项,平均序列长度只为 3,最长序列长度是 651. 关于该数据集的详细信息请见文献[16].

第 2 个数据集是 *TCAS*,这是一个收集自交通预警与防冲突系统中软件执行跟踪记录的数据集. 其中包括 1578 个序列和 75 个不同的项,平均序列长度是 36,最长序列长度为 70. 关于该数据集的详细信息请见文献[17].

第 3 个数据集是 D5C20N10S20,这是一个产生自 IBM Sequence Data Generator 的模拟数据集[1]. 这个模式数据产生工具需要填充如下参数以控制产生数据集的特性. 其中 D 代表序列的数量,C 代表平均序列长度,N 代表不同项的个数,S 代表最长序列长度. 关于该数据集的详细信息请见文献[1].

实验是将 CFSP,BIDE,RPglobal 3 个算法在上述 3 个数据集上分别进行压缩质量和算法运行时间的比较. 在每次比较中都固定距离阈值 $\delta = 0.2$,因为这是一个合理的压缩质量标准.

在压缩质量的实验中,将 CFSP,BIDE,RPglobal 3 种算法在不同数据集上产生的序列模式的数量进行比较. 为了验证本文提出的 δ-dominate 序列模式检测机制的有效性,在图 1 至图 3 中 DSP 曲线记录了 δ-dominate 序列模式的数量. 此外,如果一个算法

不能够在 60 min 内完成计算,我们就不再显示它的输出结果,将其视为不可在合理的时间范围内完成计算.因此,在某些图中只给出了 RPglobal 算法的部分结果,就是当支持度较低时 RPglobal 算法无法在合理时间范围内给出序列模式的结果集.

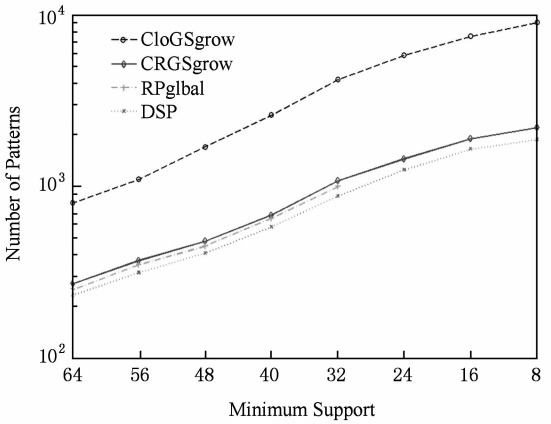


Fig. 1 Number of patterns in Gazelle.

图 1 Gazelle 数据集上的模式数量图

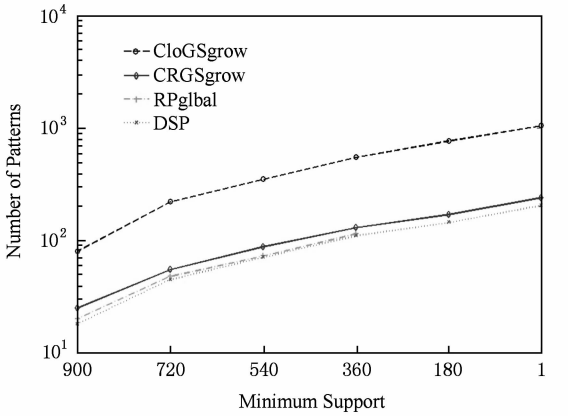


Fig. 2 Number of patterns in TCAS.

图 2 TCAS 数据集上的模式数量图

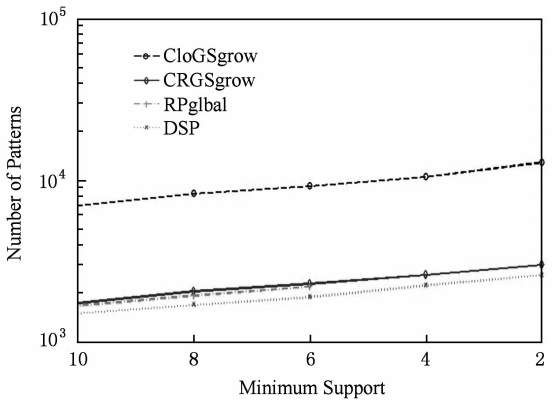


Fig. 3 Number of patterns in D5C20N10S20.

图 3 D5C20N10S20 数据集上的模式数量图

在运行时间的实验中,图 4 至图 6 显示了上述 3 种算法在不同数据集上的运行时间.特别说明的是 RPglobal 算法运行时间是通过闭序列挖掘算法 BIDE 产生出闭序列模式作为候选集的时间和在候选集中发现有代表性序列模式的时间这两部分的和.

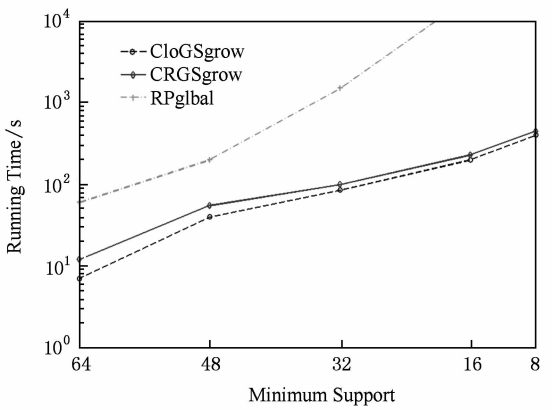


Fig. 4 Running time in Gazelle.

图 4 Gazelle 数据集上的运行时间

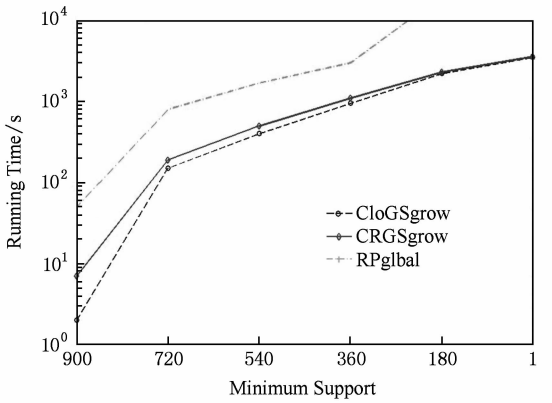


Fig. 5 Running time in TCAS.

图 5 TCAS 数据集上的运行时间

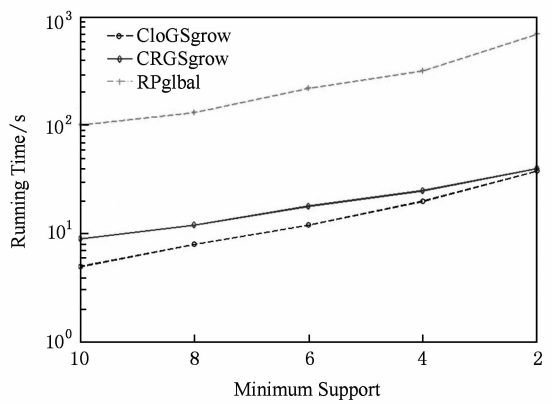


Fig. 6 Running time in D5C20N10S20.

图 6 D5C20N10S20 数据集上的运行时间

4.2 压缩质量分析

如图 1 至图 3 所示,我们可以清晰地观察到如下现象:第一,CFSP 算法产生的有代表性序列模式的数量比 RPglobal 算法产生的有代表性序列模式的数量略多一些,但是其却远少于 BIDE 算法产生的闭序列模式;第二,采用 δ-dominate 序列模式检测机制获得的 δ-dominate 序列模式的数量占 CFSP 算法产生的所有代表性序列模式的绝大部分.

第 1 种观察现象说明了 CFSP 算法的压缩质量远好于闭序列模式挖掘,并且十分接近于全局最优的压缩质量(这里认为 RPglobal 算法获得的压缩质量是全局最优的). 且 RPglobal 在支持度较低时无法在合理时间范围内完成,而 CFSP 算法可以完成. 第 2 种观察现象证明了 δ-dominate 序列模式检查机制的有效性.

4.3 运行时间分析

如图 4 至图 6 所示,可清晰地观察到如下现象:CFSP 算法的运行时间比 RPglobal 算法的运行时间减少了近一个数据量级,并接近于 BIDE 算法的运行时间. 这一观察现象解释了 δ-dominate 序列模式检测机制可在产生闭序列模式的同时获得占全部有代表性序列模式中大部分的 δ-dominate 序列模式,有效降低了 CFSP 算法的执行时间.

5 结 论

本文研究了如何从序列数据库中有效地压缩序列模式. 据我们所知,现存的工作未能很好地解决压缩序列模式的问题.

本文研究了压缩序列模式的问题,即挖掘出一个规模较小的有代表性序列模式的集合来表达全部序列模式结果集的信息. 首先,为了获得高质量的压缩,我们扩展了项集间的距离,提出了序列模式间的距离用来衡量任意两个序列模式间的相似程度,并将选择有代表性的序列模式形式化为 δ-序列覆盖. 其次,为了高效地挖掘出有代表性的序列模式,本文提出一种 δ-dominate 序列模式检测机制,可在挖掘闭序列模式的同时发现大多数的有代表性序列模式,从而大幅提高压缩序列模式的算法效率. 最后,基于 δ-dominate 序列模式检测机制,设计出一种高效的压缩序列模式挖掘算法 CFSP,不但保证压缩的准确性还具有很高的执行效率. 一个采用真实数据集与模拟数据集的实验研究也证明了 CFSP 算法可以获得很高的压缩质量,同时还具有很好的算法效率.

参 考 文 献

[1] Agrawal R, Srikant R. Mining sequential patterns [C] //Proc of the 11th Int Conf on Data Engineering. Los Alamitos, CA: IEEE Computer Society, 1995: 3-14

[2] Pei J, Han J, Mortazavi-Asl B, et al. Prefixspan: Mining sequential patterns efficiently by prefix-projected growth [C] //Proc of the 17th Int Conf on Data Engineering 2001. Los Alamitos, CA: IEEE Computer Society, 2001: 215-224

[3] Zaki M. SPADE: An efficient algorithm for mining frequent sequences [J]. Machine Learning, 2000, 42(3): 31-60

[4] Luo C, Chung S M. A scalable algorithm for mining maximal frequent sequences using a sample [J]. Journal of Knowledge and Information System, 2008, 15(2): 149-179

[5] Yan X, Han J, Afhar R. CloSpan: Mining closed sequential patterns in large datasets [C] //Proc of the 3rd SIAM Int Conf on Data Mining. San Francisco, CA: SIAM Press, 2003

[6] Wang J, Han J. BIDE: Efficient mining of frequent closed sequences [C] //Proc of the 20th Int Conf on Data Engineering. Los Alamitos, CA: IEEE Computer Society, 2004: 79-90

[7] Tzvetkov P, Yan X, Han J. TSP: Mining Top-K closed sequential patterns [C] //Proc of the 3rd Int Conf on Data Mining. Los Alamitos, CA: IEEE Computer Society, 2003: 347-354

[8] Song Shijie, Hu Huaping, Zhou Jiawei, et al. A sequential pattern mining algorithm based on large-itemset reuse [J]. Journal of Computer Research and Development, 2006, 43(1): 68-74 (in Chinese)
(宋世杰,胡华平,周嘉伟,等. 一种基于大项集重用的序列模式挖掘算法[J]. 计算机研究与发展, 2006, 43(1): 68-74)

[9] Pei J, Han J, Wang W. Constraint-based sequential pattern mining in large databases [C] //Proc of the 11th Int Conf on Information and Knowledge Management. New York: ACM, 2002: 18-25

[10] Pei J, Liu J, Wang H, et al. Efficiently mining frequent closed partial orders [C] //Proc of the 5th Int Conf on Data Mining. Los Alamitos, CA: IEEE Computer Society, 2005: 753-756

[11] Agrawal R, Srikant R. Fast algorithms for mining association rules [C] //Proc of the 20th Int Conf on Very Large Data Bases. San Francisco: Morgan Kaufmann, 1994: 487-499

[12] Afrati F N, Gionis A, Mannila H. Approximating a collection of frequent sets [C] //Proc of the 10th Int Conf on Knowledge Discovery and Data Mining 2004. New York: ACM, 2004: 12-19

[13] Xin D, Han J, Yan X, et al. On compressing frequent patterns [J]. Journal of Data and Knowledge Engineering, 2007, 60(1): 5-29

[14] Yan X, Cheng H, Han J, et al. Summarizing itemset patterns: A profile based approach [C] //Proc of the 11th Int Conf on Knowledge Discovery and Data Mining 2005. New York: ACM, 2005: 314-323

[15] Chang L, Yang D, Tang S, et al. Mining compressed sequential patterns [C] //Proc of the 2nd Int Conf on Advanced Data Mining and Applications 2006. Berlin: Springer, 2006: 761-768

[16] Kohavi R, Brodley C, Frasca B, et al. KDD cup 2000 organizers' report: Peeling the onion [J]. SIGKDD Explorations, 2000, 2(2): 86-98

[17] Lo D, Khoo S -C, Liu C. Efficient mining of iterative patterns for software specification discovery [C] //Proc of the 13th Int Conf on Knowledge Discovery and Data Mining 2007. New York: ACM, 2007: 460-469

Tong Yongxin, born in 1982. Master of Beihang Univerisity. His main research interests include data mining, data stream management, data warehouse, etc.

童咏昕,1982 年生,硕士,主要研究方向为数据挖掘、数据流管理、数据仓库等.

Zhang Yuanyuan, born in 1983. Engineer of China Academy of Telecommunication Technology. Her main research interests include data mining, data stream management, etc.

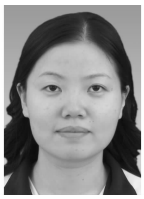
张媛媛,1983 年生,工程师,主要研究方向为数据挖掘、数据流管理等.

Yuan Mei, born in 1957. Professor of Beijing Union University since 2008. Her main research interests include data warehouse, data mining, etc.

袁 玫,1957 年生,教授,主要研究方向为数据仓库、数据挖掘等.

Ma Shilong, born in 1953. Professor and PhD supervisor of the School of Computer Science and Engineering in Beihang Univerisity. Senior member of China Computer Federation. His main research interests include automated reasoning and applications, data mining, computation model and logic in network, computation model in magnanimity information, etc.

马世龙,1953 年生,教授,博士生导师,中国计算机学会高级会员,主要研究方向为自动推理及其应用研究、网络环境下的计算模型和逻辑研究、海量信息处理的计算模型研究和数据挖掘的研究.



Yu Dan, born in 1979. Received her PhD degree in computer science from Wuhan University in 2007. Her current research interests include trusted computing, data mining and language design, etc.

余 丹,1979 年生,博士,主要研究方向为可信计算、数据挖掘、语言设计.

Zhao Li, born in 1985. Master of Beihang University. Her main research interests include magnanimity data processing, data mining, etc.

赵 莉,1985 年生,硕士,主要研究方向为海量数据处理、数据挖掘.

Research Background

This work is supported by the National 973 Basic Research Program of China under grant No. 2005CB321902 and the Beijing Municipal Commission Education Scientific and Technological Plan Fund under grant No. KM200911417003.

In the research of data mining, mining frequent sequential patterns has been a central topic. However, since the well-known downward closure property of Apriori leads to an explosive number of sequential patterns, users can not understand the result set of sequential patterns easily. In this paper, the problem of compressing frequent sequential patterns is studied. An efficient algorithm, CFSP (compressing frequent sequential patterns), is developed to mine a few representative sequential patterns to express all the information of all frequent sequential patterns and eliminate a large number of redundant sequential patterns. An empirical study with both real and synthetic data sets proves that the CFSP has good performance.