基于多维数据集的异常子群发现技术

张静恬[1] 伍 赛[2] 陈 刚[2] 寿黎但[3] 陈 珂[2]

[1](浙江大学计算机科学与技术学院 杭州 310027)

[2](浙江省大数据智能计算重点实验室 杭州 310027)

[3](浙江大学计算机辅助设计与图形学国家重点实验室 杭州 310027)

摘 要 非频繁项集是未被标准化的频繁项集产生算法(如 APRIORI 以及 FP-Growth 算法)提取的所有项集.在数据集上挖掘有意义的非频繁项集是数据挖掘的重要工作之一.目前,基于传统数据集的非频繁项集挖掘研究主要集中在负相关、负模式以及间接关联等方面,且主要是对整个数据集上的性质进行分析,而没有对数据集的切片进行分析.该文提出了一种新的模式,试图找到符合如下条件的特定子群,其描述的数据集切片上存在某些特殊项集,这些项集在整个数据集上并非频繁项集,但是在该数据集切片上却是频繁项集.根据用户要求自动找出这些异常子群以及其对应项集的算法在数据分析中有着十分重要的意义.该文提出的解决方案由两部分组成:候选产生阶段以及查询交互阶段.前者是一个脱机处理的过程,而后者则是在线实时反馈的过程.在候选产生阶段,该文提出了一种基于多维数据集高效产生频繁项集以及显著子群并有效建立索引的算法.根据索引,在查询交互阶段,该文提出的算法框架可以快速准确地返回给定查询对应的异常子群以及对应项集.基于多个真实数据集的实验表明,该文提出的方案可以根据用户要求实时返回有意义的异常子群以及对应项集.此外,该文提出的算法在多维数据集上的挖掘效率比 UTMTU 算法提升了数倍.

关键词 频繁项集挖掘;子群发现;多维数据集;数据挖掘;异常检测

中图法分类号 TP391 **DOI 号** 10.11897/SP.J.1016.2019.01671

Unexpected Subgroup Mining in Multi-Dimensional Database

ZHANG Jing-Tian[1] WU Sai[2] CHEN Gang[2] SHOU Li-Dan[3] CHEN Ke[2]

[1](*College of Computer Science and Technology, Zhejiang University, Hangzhou* 310027)

[2](*Key Laboratory of Big Data Intelligent Computing of Zhejiang Province, Hangzhou* 310027)

[3](*State Key Laboratory of CAD&CG, Zhejiang University, Hangzhou* 310027)

Abstract Traditional pattern mining generates too many frequent patterns. Many of those patterns are not very useful and too generative. For a familiar dataset, it is useless to return generative patterns (e.g., increasing profits with sales growth are frequent patterns in the original dataset). However, some unexpected patterns are more interesting and require further study. The infrequent itemsets are all the itemsets which are not extracted by standardized frequent itemsets algorithm (such as the APRIORI and FP-Growth algorithm). Mining interesting infrequent itemsets from the database is an important task in data mining. So far, infrequent itemsets mining research in the traditional database mainly focus on the negative association, negative association rules and indirect associations, and ignore the automatic analysis of the dataset slices. In this paper, we

收稿日期:2017-09-27;在线出版日期:2018-04-04.本课题得到国家"九七三"重点基础研究发展规划项目(2015CB352400)、国家自然科学基金(61661146001,61472348,61672455)、浙江省自然科学基金(LY18F020005)资助.**张静恬**,博士研究生,主要研究方向为数据挖掘和实体识别.E-mail:zyzjsz@126.com.**伍 赛**(通信作者),博士,讲师,硕士生导师,主要研究领域为 P2P 系统、分布式数据库、云系统和索引技术.E-mail:wusai@zju.edu.cn.**陈 刚**,博士,教授,博士生导师,"长江学者奖励计划"特聘教授,国家"万人计划"科技创新领军人才,中国计算机学会(CCF)会员,主要研究领域为数据库、信息安全、协同设计.**寿黎但**,博士,教授,博士生导师,中国计算机学会(CCF)会员,主要研究领域为知识工程、分布式和并行数据库.**陈 珂**,博士,副研究员、硕士生导师,中国计算机学会(CCF)会员,主要研究领域为隐私保护、数据库技术、海量数据处理和检索.

propose a new mining job, trying to find some specific subgroups, denoted as unexpected subgroups. There is at least one unexpected itemset which is not frequent in the original dataset, but frequent enough in the slice described by the unexpected subgroup. Typically, we use "unexpected patterns" to represent the combination of unexpected subgroups and unexpected itemsets. The method of finding these unexpected subgroups and corresponding unexpected patterns plays an important role in the real-life data analysis. For example, both of "earn less than $5000 per year" and "earn more than $5000 per year" are not frequent in the original dataset, but "earn less than $5000 per year" is frequent in the slice described by unexpected subgroup "high school graduate" and "earn more than $5000 per year" is frequent in the slice described by unexpected subgroup "doctorate". It is impossible to find them using traditional mining works unless you choose a very low support. But choosing a very low support may return many useless patterns at the same time and make the mining results quite confusing. In particular, our scheme consists of an offline process and an online process. For the offline process, we introduce how to find an efficient and effective algorithm that can find significant subgroup and frequent itemsets in the multi-dimensional database. When generating new subgroups, we do not need to combine two k-subgroups which share the same attributes. Besides, we only need to combine k-subgroups with the same $(k-1)$ selectors as proved in the APRIORI algorithm, because other combinations only result in redundant results. But, the computation overhead of comparing selectors and attribute sets grows dramatically. Hence, we introduce a pruning strategy to reduce the computation. In addition, we use *diffset* to check if a generated k-subgroup is significant. With this algorithm, we can generate candidate subgroups and built an index table. Then, in the online process, we can use the index and retrieve all unexpected patterns for a given user query efficiently. Besides, we develop a series of optimization techniques in order to further reduce the processing overhead. We compare our method of finding frequent itemsets in the multi-dimensional database with the UTMTU algorithm. The experiments show that even if we do not use other optimization techniques and *diffset* to increase the effectiveness, our method is several times faster than UTMTU algorithm. As the minimal support decreases, performance improvements are even more obvious. Furthermore, we perform a series of experiments to show the effectiveness of each optimization. For each real-life dataset, we listed five examples unexpected patterns and analyzed the effects of different support thresholds. Finally, experiments on the real datasets show that our approach can find unexpected patterns in a very short time (less than 1 second).

Keywords frequent pattern mining; subgroup discovery; multi-dimensional database; data mining; anomaly detection

1 引 言

现代企业在长期的生产经营中积累了大量的数据,例如大型购物广场每天都会收集到大量的用户购物数据,通过分析这些购物数据,可以了解和理解用户的购买行为,从而提高企业竞争力.随着数据规模的逐渐扩大,对强有力的数据分析工具的需求越发迫切.如何从大规模数据集中自动挖掘出潜在的有用信息是当前信息技术领域的重要工作之一.

传统的关联分析致力于对以购物篮数据集为代表的二维数据集进行分析,挖掘在整个数据集上成立的频繁项集,而直接忽略了支持度低于给定阈值的非频繁项集[1-3].频繁项集提供了一种简单有效的方式帮助企业理解用户行为模式,发现新的交叉销售商机.然而在非频繁模式中同样存在着很多值得挖掘的内容.

当前对非频繁项集的研究主要集中在三个方面:负相关[4]、负模式[5]和间接关联[6].其中负相关模式与正相关模式对应,可以用来识别竞争项[7],也

可以用来有效抑制正相关匹配过程中产生的噪声[8]. 负模式则是类似于“喝茶的人倾向于不喝咖啡”的规则[9-11]. 间接关联则试图找到满足如下条件的项集：AB 组合的支持度很低，但是 AC 组合和 BC 组合的支持度却都很高[12].

这些基于非频繁项集的方法致力于在整个数据集上找到有意义的模式，而在多维数据库中，我们可以对数据选择维度中特定的值，比如只选择性别为男性的数据，并对得到的数据切片进行分析. 当前已经有一些数据分析软件可以实现该操作，但如何找到有意义的数据切片以及对得到的数据切片进行自动分析依然是目前亟待解决的问题.

本文提出了一种新的多维数据库挖掘工作，试图找到符合如下条件的特定子群：其描述的数据集切片上存在某些频繁项集，这些项集在整个数据集上并非频繁项集，但是在该数据集切片上却是频繁项集. 本文将这样的特定子群称为异常子群，将其上存在的特殊项集称为异常项集. 异常子群与异常项集的组合构成异常模式.

根据用户要求自动找出这些异常子群以及其对应项集的算法在数据分析中有着十分重要的意义. 例如在 Adult 数据集①中，“年收入少于 5000 美元”和“年收入在 5000 美元以上”均非频繁项集. 但是在异常子群“高中毕业”描述的数据集切片中，“年收入少于 5000 美元”是频繁项集. 而在异常子群“有博士学位”描述的数据集切片中，“年收入在 5000 美元以上”是频繁项集. 只有选择一个非常低的支持度，才有可能用传统的关联挖掘算法找到如上所述的关联关系. 但与此同时，算法会同时返回很多无用的关联模式，使得用户需要查看大量无效的关联模式，才能从中找到可能有意义的规则.

在对多维数据集进行数据分析时，随着数据集维度的增长，对数据集进行挖掘的时空成本以及产生的挖掘结果数量呈现指数增长的趋势. 然而在实际查询交互的过程中，用户通常希望能够在很短时间内得到想要的结果. 如何在保证能够快速响应的同时，返回所有符合用户要求的查询结果是一个很有挑战性的问题.

为了快速有效地针对用户要求找出符合条件的异常子群以及对应的异常模式，本文提出的第一种算法框架(GCS 算法)由两部分组成：候选产生阶段以及查询交互阶段. 在候选产生阶段，本文通过一种全新的、基于多维数据集的显著子群产生算法找到候选子群并建立索引. 在查询交互阶段，根据索引可快速找到候选子群，从而在候选子群对应的切片上找到可能存在的异常项集.

GCS 算法在多数情况下已经能够在一秒以内响应在线查询需求，但是对于数据倾斜严重或者数据量更大的高维数据集，依旧需要数十秒时间才能返回结果. 为此，我们在 GCS 算法的基础上提出了 GCSM 算法，将候选项集产生过程中的支持度计算从查询交互阶段迁移到候选产生阶段. 先前提出的显著子群产生算法经过改造同样可以用来在多维数据集上挖掘频繁项集从而产生候选项集并建立索引. 实验表明，GCSM 算法可以将查询交互时间继续维持在一秒以内.

本文的主要贡献如下：

(1) 提出了一种新的多维数据集挖掘工作：在特定子群描述的切片上存在某些特殊项集，这些项集在整个数据集上并非频繁项集，但是在该数据集切片上却是频繁项集.

(2) 提出了一种基于多维数据集直接对频繁项集以及显著子群进行挖掘的高效算法，算法效率在 UTMTU 算法基础上提升了数倍.

(3) 在以上算法的基础上提出了两种可以实时针对用户要求实时返回对应异常子群以及异常项集的算法(GCS 算法和 GCSM 算法)以及一系列优化技术.

(4) 在多个真实数据集上进行实验，实验结果表明本文提出的算法可以根据用户要求快速准确地发现多维数据集中对应的异常子群以及异常项集.

本文第 2 节对子群发现技术以及多维数据集中的频繁项集挖掘算法进行总结；第 3 节对本文涉及到的概念进行定义，并提出具体的问题定义；第 4 节介绍一种可以直接在多维数据集上进行显著子群挖掘的算法，并在该算法基础上提出 GCS 算法，用于实时返回所需的异常子群以及对应异常项集；第 5 节在 GCS 算法的基础上提出 GCSM 算法. GCSM 算法可以更好适应大规模数据集以及数据倾斜严重的数据集；第 6 节提出可以应用于 GCS 以及 GCSM 算法的三种优化技术；第 7 节通过多个真实数据集上进行的一系列实验，验证算法以及优化技术的实际效果；第 8 节给出全文总结.

2　相关工作

本文提出的两种算法框架(GSS 算法和 GCSM

① https://archive.ics.uci.edu/ml/datasets/Adult

算法)在候选产生阶段都需要从多维数据集中找出符合覆盖率阈值的显著子群,并将找到的显著子群作为候选子群.2.1 节对子群发现技术进行了介绍.

在 GCSM 算法的候选产生阶段,本文提出的算法框架从多维数据集中找到频繁项集.在 GCS 算法的查询交互阶段则要从数据集的切片找到在该切片上为频繁项集但在整个数据集上非频繁项集的异常项集.2.2 节对基于多维数据集的频繁项集挖掘技术进行了介绍.

2.1 子群发现技术

当前对多维进行分析的操作主要有钻取、上卷、切片(切块)以及旋转[13].其中切片(切块)指的是对于某个(某些)维度选取特定的值或者值域,然后对得到的一部分数据进行分析.在子群发现领域中,这一部分数据被称为数据集的一个切片(slice)[14].一系列属性-属性值组成的子群描述了这个切片.

子群发现技术可以分为三种:集束搜索、穷举搜索以及基于遗传算法的方法.集束搜索虽然速度较快,但是只对部分搜索空间进行了挖掘,不符合本文的要求[15].而基于遗传算法的方法虽然可以用于找到有用的知识,却无法用于异常子群的检索[16].

而穷举搜索算法则搜索整个空间,并且在搜索过程中同时可以得到子群描述的切片大小,以便下一步在子群描述的切片上找到异常项集.此外通过穷举搜索发现子群的方法通常由频繁项集挖掘算法转化而成.例如 APRIORI-SD 算法[17]是由 APRIORI-C 分类规则学习算法[18]改进而来,而 APRIORI-C 算法是通过对原始的 APRIORI[1]频繁项集挖掘算法进行修改得到的.SD4TS 算法[19]则是在 APRIORI-SD 算法的基础上进一步改进,使用获得的规则质量去修剪搜索空间.SD-Map 算法[20]是频繁项集挖掘算 FP-Growth[2]的扩展,无需借助于其它的中间结果评估获得的规则质量.DpSubgroup 算法[21]同样是使用 FP-Growth 算法[2]的 FP-tree 结构,使用乐观估计方法[22]进行剪枝.Merge-SD 算法[23]利用变量的数值界限将搜索空间分割成不同的部分,其核心思想同样来源于 FP-Growth 算法[2].而像 IMR 算法[24]一样致力于寻找子群描述等价类的算法,参考的则是频繁项集挖掘算法 Eclat 算法[3]所提出的等价类概念.迄今为止,基于穷举搜索的子群发现算法主要都是以上算法的衍生,而应用于以上子群发现算法的优化技术很多也同样可以用于频繁项集搜索.

本文提出的基于多维数据集发现显著子群的算法同样基于穷举搜索,不仅可以用于发现显著子群,经过改造之后同样可以用于在多维数据集上发现频繁项集.该算法的核心思想具有良好的适应性,在显著子群描述的切片上查找对应的异常项集时同样可以发挥作用.此外,本文提出的优化技术既可以用于发现显著子群,也可以用于挖掘频繁项集.

2.2 多维数据集中的频繁项集挖掘算法

传统频繁项集挖掘算法如 APRIORI 算法[1]、FP-Growth 算法[2]以及 Eclat 算法[3]等的研究目标都是在二维数据集(如购物篮数据集)上对频繁项集进行挖掘,而无法直接应用到多维数据集上进行频繁项集的挖掘.部分学者选择先对数据集进行二元化处理,通过为每一对不同的属性-属性值对创造一个新的项,将多维数据集转化为二维数据集,然后再进行接下来的研究工作[25-26].

Ji 等人[27]提出了可以直接在三维数据集上挖掘频繁闭项集的方法,然而该方法很难扩展到更高维度的数据集.QARMM 算法对属性布尔位标注,再进行向量运算,但只能应用于少量属性且属性层次较少的集合[28].Xu 等人[29]提出了一种在多维数据集上逐层进行频繁项集挖掘的办法,但是该方法依旧会对属性完全相同的项集进行合并,产生了许多无效候选项集并带来极大的无意义开销.Vo 等人[30]避免了对属性完全相同的项集进行合并,但是带来了大量在属性比对上的开销.UTMTU 算法将原数据集进行编码后映射为上三角矩阵然后再进行多维频繁项集挖掘,根据有效层次对属性进行了排序,但是计算支持度采用布尔矩阵,计算量较大,而且在挖掘过程中仍需对数据集进行一次扫描[31].

本文提出的多维频繁项集挖掘算法没有维度局限性,可以应用于维度极高的数据集进行挖掘,在减少无意义计算开销的同时无需进行属性以及属性值比对.在计算项集支持度时,本文采用 *diffset* 方法,大幅度减少了计算开销.本文使用的动态排序方法,比仅根据有效层次进行一次排序的 UTMTU 算法效果更好.此外,本文提出的等价属性值以及过滤项集(过滤子群)等技术同样进一步提升了算法效率.

3 问题定义

本节首先介绍一些背景知识,然后给出详细的问题定义.

如表 1 所示,一个 n 维数据集(D)通常以表的

形式呈现，由若干条记录组成，每一行为一条记录，每条记录包含编号(tid)以及 n 个属性值. 表格的每一列对应一条属性. 本文用 a_i 和 v_i 来分别表示这些属性以及属性上的属性值. 编号的集合用 $tidset$ 表示，编号的数量用 $|tidset|$ 表示. 例如对于属性 sex 上的属性值 $Male$，其对应的 $tidset$ 为 $\{1,3,5,10\}$，$|tidset|=4$. 整个数据集中编号的数量用 $|D|$ 表示. 在如表 1 所示的样例数据集中，$|D|=10$.

表 1　样例数据集

tid	sex	$education$	$education\text{-}num$	$class$
1	*Male*	*HS-grad*	9	>50K
2	*Female*	*HS-grad*	9	≤50K
3	*Male*	*Bachelors*	13	≤50K
4	*Female*	*Bachelors*	13	≤50K
5	*Male*	*Bachelors*	13	>50K
6	*Female*	*Bachelors*	13	≤50K
7	*Female*	*Doctorate*	16	>50K
8	*Female*	*Doctorate*	16	>50K
9	*Female*	*HS-grad*	9	≤50K
10	*Male*	*Some-college*	10	≤50K

子群(subgroup)由若干个选择器(selector)的连接表达式 $\{e_1,e_2,\cdots,e_n\}$ 组成，其中每个选择器 $e_i:(a_i=v_i)$ 是在属性 a_i 值域上的滤波器(filter)[20]. 子群中所有属性 a_i 的集合称为子群的属性集(attribute set)，用 AS 表示. 每个子群都描述了多维数据集的一个切片. Lavrac 等人[32]定义了覆盖率来评估子群的重要性：给定子群 S，其覆盖率为对应切片中记录数量占原数据集中记录数量的比例，用 $cov(S)$ 表示. 若其覆盖率 $cov(S)$ 不低于给定覆盖率阈值 $mincov$，则称 S 为显著子群. 子群 S 对应的切片中所有记录的编号的集合用 $tidset(S)$ 表示. 例如对于表 1 中的原始数据集，子群 $S=\{(class=$“>50K”$)\}$ 描述了如表 2 所示的数据集切片. S 的覆盖率 $cov(S)=0.4$. S 的属性集 $AS=\{class\}$. $tidset(S)=\{1,5,7,8\}$.

表 2　样例数据集的切片

tid	sex	$education$	$education\text{-}num$
1	*Male*	*HS-grad*	9
5	*Male*	*Bachelors*	13
7	*Female*	*Doctorate*	16
8	*Female*	*Doctorate*	16

在多维数据集中，每个项(Mitem)为一对 (a_i,v_i)，其中 a_i 为属性，v_i 为 a_i 值域上的属性值. 项集(Mitemset)由一系列属性不重合的项组成. 项集中所有属性 v_i 的集合称为项集的属性集(attribute set)，用 AS 表示. n 维数据集的每一条记录由编号以及一个 n-Mitemset 组成. 若项集 I 是某条记录中 n-Mitemset 的子集，则称该条记录支持 I，其编号属于集合 $tidset(I)$. 给定项集 I，其支持度为支持该项集的记录占数据集中全部记录的比例，用 $sup(I)$ 表示. 若 $sup(I)$ 不低于给定支持度阈值 $minsup$，则称 I 在整个数据集上为频繁项集.

项集(Mitemset)的定义同样可以用到数据集的切片中. 假定数据集的切片中有 k 个属性(剩余的属性被用于描述该切片的子群)，则切片的每一行由编号以及一个 k-Mitemset 组成. 给定项集 I，其在切片中的支持度为切片中支持该项集的记录占切片中全部记录的比例，用 $sup(I,S)$ 表示. 若 $sup(I,S)$ 不低于给定支持度阈值 $minsup$，则称项集 I 在子群 S 描述的切片上为频繁项集.

根据以上定义和描述，本文给出异常子群、异常项集以及异常模式的定义.

定义 1.　对于特定显著子群 S，若存在项集 I 满足以下条件：(1) I 在整个数据集上非频繁项集；(2) I 在子群 S 描述的切片上为频繁项集，则称 S 为异常子群，I 为异常项集，(S,I) 组合为异常模式.

例如，当给定支持度阈值 $minsup$ 为 0.5 时，项集 $I=\{(sex,Male)\}$ 在如表 1 所示的原始数据集上非频繁项集，但是在如表 2 所示的子群 $S=\{(class=$“>50K”$)\}$ 描述的切片上为频繁项集.

本文的挖掘旨在根据用户给定的参数，能够自动检索出用户需要的异常模式. 然而随着维度的增长，异常模式的数量以指数速度增长，而异常模式挖掘通常应用于交互式的数据挖掘过程中，意味着结果必须在极短时间内返回. 因此，本文选择针对给定属性集上的子群进行异常模式挖掘，而非一次性检索全部的异常模式.

根据以上内容，本文给出如下问题定义：

给定多维数据集 D，覆盖率阈值 $mincov$ 以及支持度阈值 $minsup$，根据用户输入的属性集 AS，返回子群在给定属性集上的全部异常模式.

4　GCS 算法

如前所述，对异常模式的检索可以分为两个阶段：候选产生阶段以及查询交互阶段. 在 GCS 算法的候选产生阶段，本文在脱机情况下对数据集进行预处理，计算出可能是异常模式组成部分的候选子群. 在 GCS 算法的查询交互阶段，本文根据这些候选子群以及用户输入的属性集 AS，实时返回对应

的异常模式. 4.1 节对候选产生阶段进行介绍. 而 4.2 节则介绍了查询交互阶段.

4.1 候选产生阶段

根据定义,构成异常模式的候选子群应在给定覆盖率阈值$mincov$上为显著子群. 根据覆盖率 $mincov$ 的定义可得 $|tidset(S)|=cov(S)\times|D|$. 因此,可以用 $|tidset(S)|\geq mincov\times|D|$ 取代 $cov(S)\geq mincov$ 来判断新产生的子群 S 是否是显著子群.

本节将介绍如何在多维数据集中产生符合要求的显著子群并将其保存在本地文件系统中,同时对所有生成的子群建立索引,以便在查询交互阶段能够快速找到给定属性集上的全部候选子群,从而快速找到对应的异常模式.

首先,本文从原始数据集中生成一系列基本选择器. 对于离散属性 a_d 以及 a_d 上的属性值 v_d,生成选择器($a_d=v_d$). 而对于连续属性 a_c,如年龄等,显然不能对于其中出现的每个属性值都生成一个选择器. 因此本文选择将其值域划分为 L 个离散化区间:$\{[l_0,l_1),[l_1,l_2),\cdots,[l_{L-2},l_{L-1}),[l_{L-1},l_L]\}$,然后再生成一系列选择器. 此外,用户同样可以进行定制化处理,例如在属性 $class$ 上定义“>50 K”和“≤50 K”,对于表 1 中的样例数据集,可以生成表 3 中所示的一系列基本选择器.

表 3 基本选择器列表

基本选择器
$(sex=Male)$
$(sex=Female)$
$(education\text{-}num=HS\text{-}grad)$
$(education\text{-}num=Bachelors)$
$(education\text{-}num=Doctorate)$
$(education\text{-}num=Some\text{-}college)$
$(education\text{-}num=9)$
$(education\text{-}num=13)$
$(education\text{-}num=16)$
$(education\text{-}num=10)$
$(class=$“>50 K”$)$
$(class=$“≤50 K”$)$

随后对每个基本选择器建立对应的 $tidset$. 对于每个选择器 e_i,若 e_i 满足 $|tidset(e_i)|\geq mincov\times|D|$,则对 e_i 生成一个 1 阶子群 $S_i=\{e_i\}$,容易知道,$|tidset(S_i)|=|tidset(e_i)|\geq mincov\times|D|$,即生成的 1 阶子群为数据集上的全部 1 阶显著子群.

接下来本文采取逐层生成并检测的策略来找到数据集中全部的显著子群. 为了减少不必要的开销,本文通过结合两个 k 阶子群生成 $k+1$ 阶子群. 显而易见的,这两个子群应含有 $k-1$ 个相同的选择器. 此外,若两个 k 阶子群的属性集相同,则无需对这两个 k 阶子群进行结合,因为它们要么共享全部的属性值,实质上是相同的子群,要么基于其生成的 $k+1$ 阶子群覆盖率为 0. 但覆盖率为 0 的子群不可能是显著子群.

对属性集以及属性值进行对比的时间开销增长十分迅速,为了避免这些开销,本文定义了 S-universe 以及 Attribute Set Subgroup Tuple(简写为 AS-Tuple)的概念以方便剪枝. 前 $k-1$ 个属性以及属性值完全相同且仅在最后一个选择器的属性或属性值上有所不同的 k 阶子群属于同一个 k 阶 S-universe. 而 k 阶 AS-Tuple 由两部分组成:属于同一个 k 阶 S-universe 且属性集相同的子群,以及这个属性集本身.

如图 1 所示,2 阶子群 $S_1=\{(a_1=v_2),(a_2=v_4)\}$ 与 $S_2=\{(a_1=v_2),(a_3=v_6)\}$ 共享同一个选择器($a_1=v_2$)因而 S_1 与 S_2 在同一个 S-universe 中,但是 S_1 的属性集为 $\{a_1,a_2\}$,而 S_2 的属性集为 $\{a_1,a_3\}$,它们的属性集并非完全相同,因而 S_1 与 S_2 落在不同的 AS-Tuple 中.

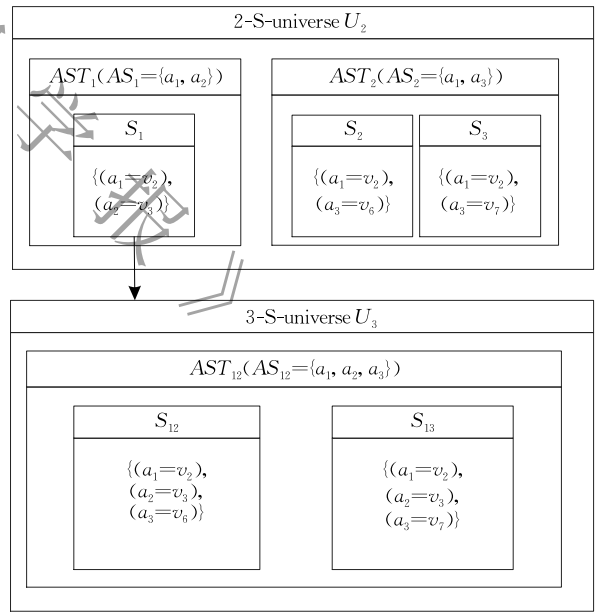

图 1 S-universe 生成样例

要达到如前所述的剪枝效果,只需在生成 $k+1$ 阶子群时,对相同 S-universe 中来自不同 AS-Tuple 的 k 阶子群进行结合,而无需对属性和属性值进行对比. 基于不同 k-S-universe 的 $k+1$ 阶子群的生成过程是互相独立的,这种独立性使得显著子群的生成可以通过非常简单的方式扩展到分布式计算. 算法 1 详细说明了如何从 k-S-universe 中的显著子群

产生一系列$(k+1)$-S-universe 以及其中的显著子群的详细过程.

算法 1.　S-universe 生成算法.

输入：k-S-universe U_k；

输出：一系列$(k+1)$-S-universe

1. FOR U_k 中的每个 AS-Tuple AST_i
2. AS_i 为 AST_i 中的属性集
3. FOR AST_i 中的每个子群 S_m
4. $(k+1)$-S-universe $U_{k+1}=\varnothing$
5. FOR U_k 中的每个 AS-Tuple $AST_j(j>i)$
6. AS_j 为 AST_j 中的属性集
7. $AS_{ij}=AS_i \cup AS_j$
8. 建立新 AS-Tuple AST_{ij}
9. $AST_{ij}.setAS(AS_{ij})$
10. FOR AST_j 中的每个子群 S_n
11. $S_{mn}=S_m \cup S_n$
12. 计算 $|tidset(S_{mn})|$
13. IF $|tidset(S_{mn})| \geq minsup \times |D|$ THEN
14. $AST_{ij}.addsubgroup(S_{mn})$
15. END IF
16. END FOR
17. IF $|AST_{ij}|>0$ THEN
18. $U_{k+1}.add(AST_{ij})$
19. 存储 AST_{ij} 并建立索引
20. END IF
21. END FOR
22. END FOR
23. END FOR

如图 1 所示，对于 k-S-universe U_k 中的每个子群 S_m 建立一个新的$(k+1)$-S-universe U_{k+1}. S_m 中的 k 个属性以及属性值即为 U_{k+1} 中相同的 k 个属性以及属性值. 在 U_{k+1} 中，由相同的两个 AS-Tuple AST_i 和 AST_j 中的子群结合成的新子群均属于 AS-Tuple AST_{ij}，因为这些新子群有着相同的属性集 $AS_{ij}=AS_i \cup AS_j$，其中 AS_i 和 AS_j 分别为 AST_i 和 AST_j 中的属性集. 例如子群 S_{12} 和 S_{13} 均属于 AS-Tuple AST_{12}，因为它们共享属性集$\{a_1,a_2,a_3\}$.

为了有效地寻找特定属性集上的显著子群，本文在存储 AS-Tuple 的同时对其建立索引. k 阶 AS-Tuple 存储在文件名为 k-subgroup 的文件中. 该文件的每条索引由两部分组成：属性集 AS_k 以及属性集为 AS_k 的每条 AS-Tuple 在文件中存储的位置. 根据索引，对于给定属性集 AS_k，可以快速得到属性集为 AS_k 的 AS-Tuple，进而得到属性集为 AS_k 的全部子群. 显然，在显著子群中，每个属性（维度）最多只可能出现一个属性值. 因此，假设基础选择器中含有 w 个属性，则有 $0 \leq k \leq w$.

在传统数据集上的频繁项集挖掘过程中，对于新产生的项集 I_{mn}，Zaki 等人[3] 提出了一种利用 $diffset$ 来快速计算 $|tidset(I_{mn})|$ 的方法并给出了证明：

$$diffset(I_{mn})=diffset(I_n)-diffset(I_m),$$
$$|tidset(I_{mn})|=|tidset(I_m)|-|diffset(I_m)-diffset(I_n)|.$$

其中项集 I_m 和项集 I_n 的前 $k-1$ 个项完全相同，$I_{mn}=I_m \cup I_n$.

根据本文第 3 节中给出的定义，容易知道，若项集 I_{mn} 与子群 S_{mn} 共享相同的属性以及属性值，则有 $tidset(S_{mn})=tidset(I_{mn})$. 鉴于 $diffset$ 的计算完全取决于 $tidset$，除了 $|tidset(S_{mn})|=|tidset(I_{mn})|$ 之外，还可以得到 $diffset(S_{mn})=diffset(I_{mn})$.

因此，在判断子群 S_{mn} 是否是显著子群时，同样可以使用 $diffset$ 来减少计算量，加快计算速度. 通过 $diffset$ 计算 $|tidset(S_{mn})|$ 的具体计算公式如下：

$$diffset(S_{mn})=diffset(S_n)-diffset(S_m),$$
$$|tidset(S_{mn})|=|tidset(S_m)|-|diffset(S_m)-diffset(S_n)|.$$

基于 k-S-universe U_k 进行 S-universe 生成新子群 S_{mn} 时，可以通过 S_m 和 S_n 的 $diffset$ 以及 $|tidset|$ 来计算 $diffset(S_{mn})$ 和 $|tidset(S_{mn})|$. 对于每个生成的显著子群 S_{mn}，本文利用 hash 算法根据其属性以及属性值生成一个编号（$sid(S_{mn})$），并用 $sid(S_{mn})$ 建立 $diffset(S_{mn})$ 以及 $|tidset(S_{mn})|$ 的索引，使得在迭代过程中可以快速得到上一层子群的 $diffset$ 以及 $|tidset|$.

4.2　查询交互阶段

在查询交互阶段，根据用户指定的属性集 AS_k 可得所求子群的属性以及属性值数量 k，根据属性 k 找到文件 k-subgroup，即可根据索引快速读出所有的 k 阶 AS-Tuple，从而得到满足要求的全部候选子群. 对于每个候选子群 S，为了计算对应的异常模式，本节需要找到所有在子群 S 描述的切片上为频繁项集但是在整个数据集上非频繁项集的项集.

我们可以使用类似产生候选子群的方式来找到所有在子群 S 描述的切片上为频繁项集的全部项集. 首先，找到在子群 S 描述的切片上为频繁项集的 1 阶项集，然后逐层计算在子群 S 描述的切片上为频繁

项集的 k 阶项集.类同于 S-universe 以及 Attribute Set Subgroup Tuple,本文对项集建立 I-universe 以及 Attribute Set Mitemset Tuple(AI-Tuple)以减少计算过程中对项集中的属性以及属性值的比对开销.在生成 $k+1$ 阶项集时,只需对相同 k-I-universe 中来自不同 k-AI-Tuple 的 k 阶项集进行组合.

对于每个在给定子群 S 描述的切片上为频繁项集的项集 I,若其在整个数据集上非频繁项集,则(S,I)为异常模式.I 在整个数据集上非频繁项集等价于 $|tidset(I)|<minsup\times|D|$,而对于 $I=I_1\cup I_2$ 可以利用 $diffset$ 通过以下公式计算 $|tidset(I)|$[3]:

$$diffset(I)=diffset(I_2)-diffset(I_1),$$
$$|tidset(I)|=|tidset(I_1)|-|diffset(I_1)-diffset(I_2)|.$$

项集 I 在子群 S 描述的切片上为频繁项集等价于 $|tidset_s(I)|>minsup\times|tidset(S)|$.给定子群 S,对于项集 $I=I_1\cup I_2$ 容易知道:

$$\begin{aligned}tidset_s(I)&=tidset(I)\cap tidset(S)\\&=tidset(I_1)\cap tidset(I_2)\cap tidset(S)\\&=tidset(I_1)\cap tidset(S)\cap tidset(I_2)\cap tidset(S)\\&=tidset_s(I_1)\cap tidset_s(I_2).\end{aligned}$$

因此 $|tidset_s(I)|$ 同样可以利用 $diffset$ 通过以下公式计算:

$$diffset_s(I)=diffset_s(I_2)-diffset_s(I_1),$$
$$|tidset_s(I)|=|tidset_s(I_1)|-|diffset_s(I_1)-diffset_s(I_2)|.$$

对于每个新生成的且在给定子群 S 描述的切片上为频繁项集的项集 I,可以利用与 4.1 节中同样的 hash 算法生成一个基于其属性以及属性值的编号($smid(I)$),并用该编号对 $diffset(I)$、$|tidset(I)|$、$diffset_s(I)$ 以及 $|tidset_s(I)|$ 建立索引,以便在迭代过程中快速得到上一层的信息.

5 GCSM 算法

当数据集规模较小且数据倾斜不严重时,GCS 算法可以提供足够优秀的计算效果,且在用户需要频繁调整阈值时,候选产生阶段耗时较短.然而在针对含有较多属性且数据倾斜严重的大规模数据集进行挖掘时,GCS 算法无法继续提供实时返回的结果.因此,本文在 GCS 算法的基础上提出了一种新的算法 GCSM,其核心思路是把候选项集产生过程中的支持度计算从查询交互阶段迁移到候选产生阶段.

根据定义,对于异常模式(S,I),子群 S 应在给定阈值 $mincov$ 上为显著子群,即 $|tidset(S)|\geq mincov\times|D|$,而项集 I 应在显著子群 S 描述的切片上为频繁项集,$|tidset_s(I)|\geq minsup\times|tidset(S)|$.因此项集 I 应满足 $|tidset_s(I)|\geq minsup\times|tidset(S)|\geq minsup\times mincov\times|D|$.

如果在候选产生阶段事先产生了一系列满足 $|tidset(I)|\geq minsup\times mincov\times|D|$ 的候选项集并对每个符合要求的项集 I,通过如前所述的 hash 算法根据其属性以及属性值计算其编号($mid(I)$),利用编号对项集以及 $|tidset(I)|$ 建立索引.则在查询交互阶段,可以通过计算 $mid(I)$ 直接从索引中获得 $|tidset(I)|$,而对于 $|tidset_s(I)|$,只要产生一个与 S 属性以及属性值完全相同的项集 I_S,易得出:

$$\begin{aligned}tidset_s(I)&=tidset(I)\cap tidset(S)\\&=tidset(I)\cap tidset(I_S)\\&=tidset(I\cup I_S).\end{aligned}$$

因为 $|tidset_s(I)|\geq minsup\times mincov\times|D|$,所以 $|tidset(I\cup I_S)|\geq minsup\times mincov\times|D|$.因此,可以知道 $I\cup I_S$ 必然在候选产生阶段产生过的项集中,即在候选产生阶段,已经计算过 $|tidset(I\cup I_S)|$ 并对 $|tidset(I\cup I_S)|$ 建立了索引,因而在查询交互阶段,只需根据相同的 hash 算法对项集 $I\cup I_S$ 的编号进行计算,即可通过索引直接得到 $|tidset_s(I)|$,而无需再进行任何集合运算.

鉴于集合运算是查询交合阶段中的主要时间开销来源,将项集支持度计算迁移到候选产生阶段可以大幅度减少查询时间.

令 $minsup'=minsup\times mincov$,GCSM 算法在候选产生阶段的目标是在整个数据集上产生以 $minsup'$ 为最小支持度的频繁项集并建立索引记录每个符合要求的项集的支持度.

算法 2 的主体内容与先前所述的产生显著子群的算法 1 相似,首先找出在整个数据集上符合要求的 1 阶项集,然后逐层计算符合 $|tidset(I_{mn})|\geq minsup'\times|D|$ 的全部项集.如前所述,在逐层生成 $k+1$ 阶频繁项集的过程中,只需对相同 I-universe 中来自不同 AI-Tuple 的 k 阶项集进行组合.用 $diffset$ 计算 $|tidset(I)|$ 的具体方法可见 4.2 节.对于每个符合要求的项集 I,本文通过如前所述的 hash 算法根据其属性以及属性值计算其编号($mid(I)$),并利用编号对项集以及 $|tidset(I)|$ 建立索引.

算法 2. I-universe 生成算法.

输入：k-I-universe U_k

输出：一系列$(k+1)$-I-universe

```
1.  FOR U_k中的每个 AI-Tuple AIT_i
2.    AS_i为AIT_i中的属性集
3.    FOR AIT_i中的每个项集 I_m
4.    (k+1)-I-universe U_{k+1}=∅
5.    FOR U_k中的每个 AI-Tuple AIT_j(j>i)
6.      AS_j为AIT_j中的属性集
7.      AS_ij=AS_i∪AS_j
8.      建立新 AI-Tuple AIT_ij
9.      AIT_ij.setAS(AS_ij)
10.     FOR AIT_j中的每个项集 I_n
11.       I_mn=I_m∪I_n
12.       计算|tidset(I_mn)|
13.       IF|tidset(I_mn)|≥minsup'×|D| THEN
14.         AIT_ij.addMitemset(I_mn)
15.           建立索引记录|tidset(I_mn)|
16.         END IF
17.       END FOR
18.       IF |AIT_ij|>0 THEN
19.         U_{k+1}.add(AIT_ij)
20.       END IF
21.     END FOR
22.   END FOR
23. END FOR
```

在真实数据集 Census-Income 上进行实验得到的结果表明，GCS 算法在属性较多且数据倾斜严重的大规模数据集上需要数十秒才能返回查询结果，而 GCSM 算法依旧能够在一秒以内得到查询结果. 具体实验结果参看第 7 节.

6 优化技术

本节提出三种可以应用于 GCS 算法以及 GSCM 算法中进一步减少运行时间的优化技术. 其中 6.1 节详细描述了动态排序，6.2 节介绍了等价属性值的概念以及其具体用法，6.3 节对过滤项集和过滤子群进行了定义. 并将其应用于 GCS 以及 GCSM 算法中.

6.1 动态排序

模式挖掘过程中，元素的排列顺序是一个很重要的考量. 在传统基于二维数据集进行频繁项集挖掘的领域，Bayardo 给出了如下推论：根据支持度将元素进行升序排序能够尽可能减少计算量[33].

为了将这个结论应用到本文的算法中，我们对 AS-Tuple 和 AI-Tuple 的 $|tidset|$ 进行定义. 其中 AS-Tuple 的 $|tidset|$ 为其中所有子群 $|tidset|$ 的总和，而 AI-Tuple 的 $|tidset|$ 则为其中所有项集 $|tidset|$ 的总和. 根据 Bayardo 给出的推论，为了尽可能地减少计算量，在 S-universe(I-universe)逐层生成过程中，对于每个新生成的 S-universe(I-universe)，对其中的 AS-Tuple(AI-Tuple)根据其 $|tidset|$ 进行升序排序. 根据本文的算法描述，因为在新的子群(项集)产生过程中，无需对 AS-Tuple(AI-Tuple)内部的子群(项集)进行组合，所以对 AS-Tuple(AI-Tuple)中的子群(项集)进行排序没有任何意义.

6.2 等价属性值

在多维数据集中，若两个分别存在于不同属性上的属性值在整个数据集上分布完全相同，即其 *tidset* 完全相同，则本文认为这两个属性值是等价属性值. 例如在表 1 中所示的数据集中，属性 *education* 上的属性值 *HS-grad* 以及属性 *education-num* 上的属性值 9，它们对应的 *tidset* 均为{1,2,9}，因而这两个属性值为等价属性值.

鉴于这两个属性值对应的 *tidset* 完全相同，在任何一个属性上都不可能存在第二个属性值使得其与另一个属性值组合生成的子群是显著子群，也不可能存在第二个属性值使得其与另一个属性值组合生成的项集是频繁项集. 而只要显著子群或者频繁项集含有二者中的一个属性值，将其换成另一个属性值得到的结果必然是显著子群或者频繁项集. 因此，在生成候选子群以及项集的过程中，这两个属性值带来的集合运算也是完全相同的.

为了避免进行重复地集合运算，在开始候选子群以及候选项集计算前，首先根据符合要求的项进行等价项计算，对于若干个等价的项，只保留其中一个，并记录等价关系，保留下来的等价项称为保留等价项. 鉴于拥有相同属性以及属性值的选择器和项，其 *tidset* 必然相同，故而可以直接根据等价项关系，进行基于 *mincov* 的过滤，得到对应的等价选择器关系，对于若干个等价的选择器只保留其中一个，并记录等价关系，保留下来的等价选择器称为保留等价选择器.

对于含有保留等价项 $Mitem_1$ 的频繁项集 I 以及与 $Mitem_1$ 等价的项 $Mitem_2$，容易知道，$I'=(I-\{Mitem_1\})\cup\{Mitem_2\}$ 和 $I''=I\cup\{Mitem_2\}$ 均为频繁项集，这些频繁项集可以直接通过等价关系进行恢复. 同理可得，对于含有保留等价选择器 e_1 的

显著子群 S 以及与 e_1 等价的项 e_2，容易知道，$S'=(S-\{e_1\})\cup\{e_2\}$ 以及 $S''=S\cup\{e_2\}$ 均为显著子群，而这些显著子群同样可以直接通过等价关系进行恢复.

6.3 过滤项集(过滤子群)

若 2 阶项集 $I_2=\{Mitem_1,Mitem_2\}$ 且 $Mitem_1$ 和 $Mitem_2$ 满足 $tidset(Mitem_1)\subset tidset(Mitem_2)$，则称 I_2 为一个过滤项集.

定理 1. 对于任意项集 I 和 I'，若存在过滤项集 $I_2=\{Mitem_1,Mitem_2\}\subseteq I$ 且 $I'=I-\{Mitem_2\}$，则有 $tidset(I)=tidset(I')$.

证明. 令 $I''=I-\{Mitem_1,Mitem_2\}$.

$$\begin{aligned}tidset(I)&=tidset(I'')\cap tidset(Mitem_1)\cap tidset(Mitem_2)\\&=tidset(I'')\cap tidset(Mitem_1)\\&=tidset(I').\end{aligned}$$

证毕.

对于任何 k 阶频繁项集 I，若存在过滤项集 $I_2=\{Mitem_1,Mitem_2\}\subseteq I$，则根据定理 1 可得，$|tidset(I)|=|tidset(I')|$，因此 I' 是频繁项集与 I 是频繁项集是等价的. 在先前项集的逐层生成过程中，必然已经对 $k-1$ 阶项集 I' 计算过 $|tidset(I')|$.

而对于给定子群 S，同样可以得出：

$$\begin{aligned}|tidset_s(I)|&=|tidset(I)\cap tidset(S)|\\&=|tidset(I')\cap tidset(S)|\\&=|tidset_s(I')|.\end{aligned}$$

即项集 I 在子群 S 描述的切片上为频繁项集与项集 I' 在该切片上为频繁项集也是等价的. 在切片上生成候选项集过程中，必然已经对 $k-1$ 阶项集 I' 计算过 $|tidset_s(I')|$.

为了减少不必要的重复计算，在项集的逐层产生过程中，可以直接对含有过滤项集的项集进行剪枝. 而对于含有过滤项集的频繁项集，可以直接通过键值对 $(Mitem_1,Mitem_2)$ 以及含有 $Mitem_1$ 的频繁项集进行恢复.

若 2 阶子群 $S_2=\{e_1,e_2\}$ 满足 $tidset(e_1)\subset tidset(e_2)$，则称 S_2 为一个过滤子群. 鉴于若项集 I 与子群 S 共享相同的属性以及属性值，则有 $tidset(S)=tidset(I)$，因此根据定理 1 的证明可得对于任何 k 阶子群 S，若存在过滤子群 $S_2=\{e_1,e_2\}\subseteq S$ 且 $S'=S-\{e_2\}$，则 $tidset(S)=tidset(S')$. 据此，在显著子群的逐层产生过程中，同样可以对含有过滤子群的子群进行剪枝以减少无效的集合计算. 而对于含有过滤子群的显著子群，则可以直接通过键值对 (e_1,e_2) 以及含有 e_1 的候选子群进行恢复.

7 实验结果

本节在多个真实数据集上进行了实验来分析算法的表现. 实验环境为 Intel(R) Xeon(R) 2.0 2.00 GHz，8 GB 内存，Ubuntu 操作系统. 所有的代码用 java 编程语言实现，用 java 1.7.079 编译.

本次实验使用了三个真实数据集，即 Adult 数据集，HADS 数据集以及 Census-Income 数据集. Adult 数据集由巴里·贝克尔(Barry Becker)从 1994 年的美国人口普查数据库中提取出的干净数据构成，含有 14 个属性，两万多个属性值，近三万条记录. HADS 是 2011 年美国国家住房调查(AHS)中提供的住房负担能力数据，含有 99 个属性，两百万个属性值，14 万条记录. Census-Income 数据集是从 1994 年和 1995 年美国人口普查局进行的人口调查中提取的加权人口普查数据，含有 40 个属性，近十三万个属性值，近三十万条记录.

7.1 基于多维数据集的频繁项集挖掘

UTMTU 算法[31]对原始数据集进行编码筛选，然后映射为上三角矩阵和布尔矩阵，根据有效层次对属性进行排序，通过布尔矩阵对新产生的项集进行过滤.

为了更好地将本文提出的基于多维数据集的频繁项集挖掘算法与 UTMTU 算法进行比较，在本节中，使用仅应用了动态排序技术的 I-universe 生成算法与之对比. 在判断新产生的项集是否为频繁项集时，本文实现了与布尔矩阵原理较为相似但效率低于 $diffset$ 的 $tidset$ 方法. Zaki 已经给出 $tidset$ 与 $diffset$ 的效率对比[3]，这里不再赘述.

基于多个真实数据集的实验结果如图 2 到图 4 所示. 可以看到，在仅使用动态排序算法和 $tidset$ 的

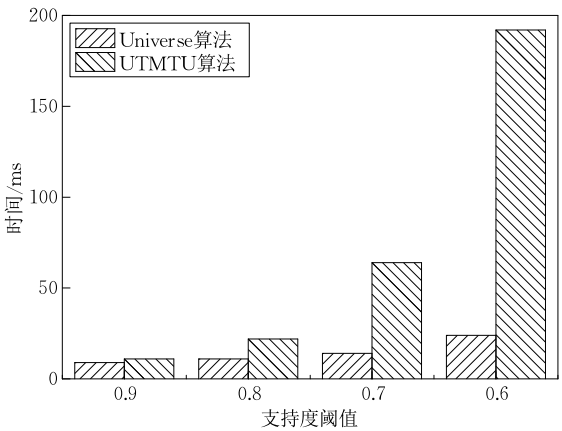

图 2 Adult 数据集上的时间对比

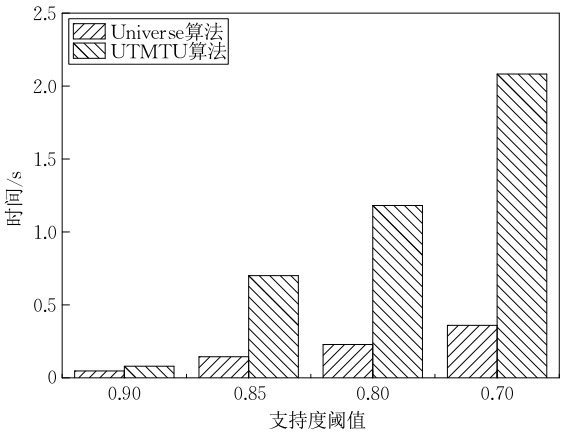

图 3　HADS 数据集上的时间对比

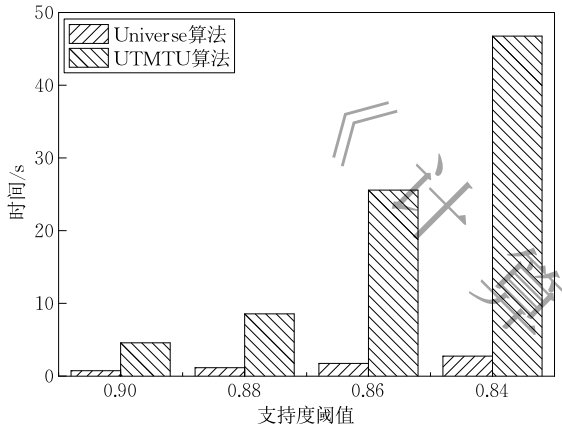

图 4　Census-Income 数据集上的时间对比

情况下，本文提出的算法（图中用 Universe 算法指代）计算效率明显高于 UTMTU 算法.

主要原因在于本文提出的算法不仅需要对更少的项集进行是否频繁的判断，而且将迭代算法开始前的排序拓展到在每一层计算之后都进行排序，使得剪枝机制更加高效.

7.2　优化方案

本文中的优化技术主要由四部分组成：动态排序、等价属性值、过滤项集以及过滤子群. 本节通过在 Census-Income 数据集上进行多组实验，对优化技术在算法各个组成部分上产生的效果进行分析. 其中 7.2.1 节对优化技术在 GCS 算法以及 GCSM 算法的候选产生阶段中产生的效果进行分析，7.2.2 节对优化技术在 GCS 算法的交互阶段中产生的效果进行分析. 在本节中，支持度阈值默认设置为 0.80.

7.2.1　候选产生阶段

本节对优化算法在候选子群产生阶段以及候选项集产生阶段的效果进行分析.

在候选子群产生阶段，可以使用的优化方法包括动态排序、等价属性值以及过滤子群. 候选子群产生阶段的耗时如图 5 所示. 可以看到从覆盖率阈值低于 0.85 开始，每一种优化技术的加入都在先前基础上对算法产生的效果有了较大提升，其中动态排序的提升效果最为明显. 且随着覆盖率阈值的降低，优化技术的提升效果越来越显著.

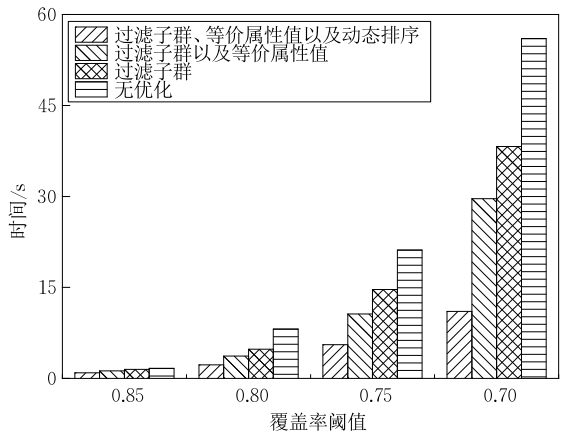

图 5　候选子群产生阶段的时间对比

在候选项集产生阶段，可以使用的优化方法包括动态排序、等价属性值以及过滤项集. 候选项集产生阶段的耗时如图 6 所示. 可以看到从覆盖率阈值低于 0.85 开始，每一种优化技术的加入都在先前基础上对算法产生的效果有了较大提升. 随着覆盖率阈值的降低，优化技术的提升效果越来越显著.

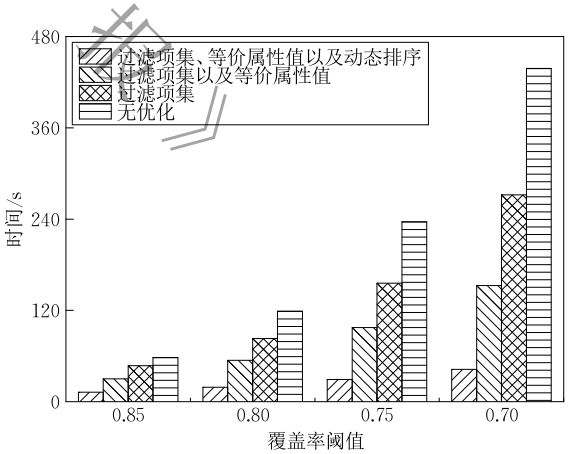

图 6　候选项集产生阶段的时间对比

7.2.2　查询交互阶段

本节对优化算法在 GCS 算法中查询交互阶段的效果进行分析. 而在 GCSM 算法中，鉴于可以直接从索引中进行查询，而无需进行支持度运算，本文提出的优化方法很难在此基础上有明显提升. 在 GCS 算法的查询交互阶段中，对全部有查询结果的属性集进行测试，取总时间作为测试标准. 可以使用

的优化方法包括动态排序、等价属性值以及过滤项集.实验结果如图 7 所示.可以看到从覆盖率阈值低于 0.90 开始,每一种优化技术的加入都在先前基础上提升了算法效率.随着覆盖率阈值的降低,优化技术的提升效果越来越显著.

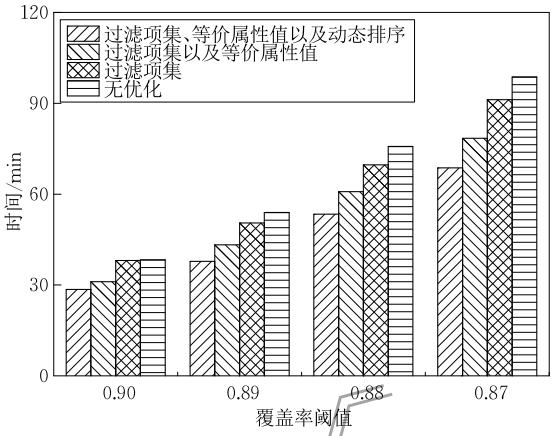

图 7 查询交互阶段的时间对比

7.3 GCS 和 GCSM 算法的实际效果

本节通过在三个真实数据集上运行 GCS 和 GCSM 算法对优化后的算法效果进行了分析,在 GCSM 算法候选产生阶段,对候选项集以及候选子群产生过程的时间总和进行统计.而在两个算法的查询交互阶段,对全部有查询结果的属性集进行测试,取平均时间作为测试标准.其中,GCS 脱机和 GCSM 脱机分代表 GCS 算法和 GCSM 算法在候选产生阶段的运行时间.GCS 查询和 GCSM 查询分别代表 GCS 算法和 GCSM 算法在查询交互阶段平均每次查询的运行时间.

首先,本文在 Adult 数据集上取支持度阈值为 0.7 进行测试.时间单位为 ms.结果如表 4 所示.可以看到在 Adult 数据集上,GCS 算法已经可以在一秒以内返回查询结果,而 GCSM 算法虽然进一步提升了查询效率,但是在候选产生阶段耗时较长.

表 4 Adult 数据集的耗时

覆盖率阈值	GCS 脱机	GCSM 脱机	GCS 查询	GCSM 查询
0.80	5	13	35	5
0.60	7	21	17	3
0.40	17	48	24	5
0.20	110	245	19	3

然后,本文在 HADS 数据集上取支持度阈值为 0.85 进行测试.时间单位为 ms.结果如表 5 所示.可以看到在 HADS 数据集上,GCS 算法同样可以在一秒以内返回查询结果,而 GCSM 算法虽然进一步提升了查询效率,但是在候选产生阶段耗时较长.

表 5 HADS 数据集的耗时

覆盖率阈值	GCS 脱机	GCSM 脱机	GCS 查询	GCSM 查询
0.90	1	8	142	16
0.85	4	13	173	64
0.80	6	64	320	53
0.70	42	210	189	44

最后,本文在 Census-Income 数据集上取支持度阈值为 0.8 进行测试.时间单位为 s.结果如表 6 所示.可以看到,在 Census-Income 数据集上 GCS 算法需要数十秒来返回查询结果,而 GCSM 算法依旧可以在一秒以内返回查询结果.

表 6 Census-Income 数据集的耗时

覆盖率阈值	GCS 脱机	GCSM 脱机	GCS 查询	GCSM 查询
0.90	0.305	8.31	35.1	0.402
0.89	0.362	9.14	33.4	0.381
0.88	0.450	10.00	35.2	0.368
0.87	0.574	11.30	34.5	0.379

此外,从表 4 到表 6 中可以看到,无论是 GCS 算法还是 GCSM 算法的候选产生阶段,所需要的实际均随着覆盖率阈值的降低而上升,但由于支持度阈值保持不变,查询交互阶段所需的时间会在一定范围内来回摆动.

7.4 异常模式样例

对于 Adult 数据集,本文列出如下 5 条符合支持度阈值为 0.7 的异常模式样例:

(1)“年收入高于 50 000 美元”在“学历为拥有博士学位”等子群描述的切片上为频繁项集;

(2)“性别为男性”在“出生地为墨西哥”,“年收入高于 50 000 美元”,“年龄在 35 到 39 之间”,“每周工作 45～60 小时”等子群描述的切片上为频繁项集;

(3)“婚姻状况为已婚”在“学历为专业学校”,“年收入高于 50 000 美元”等子群描述的切片上为频繁项集;

(4)“职业为自由职业者”在“性别为女性”,“婚姻关系为未婚”等子群描述的切片上为频繁项集;

(5)“婚姻关系为已婚且性别为男性”在“年收入高于 50 000 美元”等子群描述的切片上为频繁项集.

对于 HADS 数据集,本文列出如下 5 条符合支持度阈值为 0.85 的异常模式样例:

(1)“房屋结构为独栋”在“每单元有 6 个房间”,“每个单元有 4 间卧室”等子群描述的切片上为频繁项集;

(2)“房主(房东数量为 2)”在“每单元有 3 个房

间”，“每个单元有 1 间卧室”，“房屋结构为每栋 5～19 个单位”等子群描述的切片上为频繁项集；

(3)“收入低于 50%的公平市场租金(FMR)”在“收入低于贫困线”，“收入低于 30%地区平均收入”等子群描述的切片上为频繁项集；

(4)“房屋开销低于收入 30%”在“收入高于 300%的公平市场租金(FMR)”，“收入超过 120%的地区平均收入”等子群描述的切片上为频繁项集；

(5)“收入超过 200%的贫困线”在“每单元有 7 个房间”，“每个单元有 4 间卧室”等子群描述的切片上为频繁项集.

对于 Census-Income 数据集，本文列出如下 5 条符合支持度阈值为 0.8 的异常模式样例：

(1)“种族为白人且年收入不高于 50000 美元”在“性别为女性”等子群描述的切片上为频繁项集；

(2)“性别为女性”在“房屋产权情况为伴侣拥有房屋产权”等子群描述的切片上为频繁项集；

(3)“父亲出生于美国”在“母亲出生于美国”，“学历为高中”等子群描述的切片上为频繁项集；

(4)“房屋产权情况为伴侣拥有房屋产权”在“性别为女性”等子群描述的切片上为频繁项集；

(5)“婚姻关系为未婚”在“家庭情况为父母双全”等子群描述的切片上为频繁项集.

7.5　阈值设定对异常模式的影响

本节以 Adult 数据集为例，对不同阈值设定下，异常模式挖掘的结果进行分析.

在支持度阈值提升后，可能存在原先成立的异常模式在新的阈值下不成立的情况. 例如，在子群“学历为拥有博士学位”描述的切片上，“年收入高于 50000 美元”的支持度为 0.74，因此支持度阈值为 0.70 时，“年收入高于 50000 美元”在子群“学历为拥有博士学位”描述的切片上为频繁项集，但是当支持度阈值为 0.80 时，“年收入高于 50000 美元”在子群“学历为拥有博士学位”描述的切片上为非频繁项集.

在支持度阈值提升后，也可能存在原先不成立的异常模式在新的阈值下成立的情况. 例如，“年收入高于 50000 美元”的支持度为 0.76，因此支持度阈值为 0.70 时，“年收入低于 50000 美元”在子群“学历为高中毕业”描述的切片上为非频繁项集，但是支持度阈值为 0.80 时，“年收入低于 50000 美元”在子群“学历为高中毕业”描述的切片上为频繁项集.

8　结　语

本文提出了一种新的模式，由描述数据集特定切片的子群以及只在其描述的数据集切片上为频繁项集而在整个数据集上并非频繁项集的特殊项集. 为了更好地根据用户需求挖掘出这种特殊模式，本文提出一种基于多维数据集高效产生频繁项集以及显著子群并有效建立索引的算法. 实验结果表明，该算法在多维数据集上的挖掘效率比 UTMTU 算法提升了数倍.

在此基础上，本文提出了两种在多维数据集上挖掘异常模式的算法框架：GCS 算法和 GCSM 算法. 其中候选产生阶段耗时较短的 GCS 算法在数据倾斜不严重的高维大规模数据集上即可在一秒以内返回查询结果. 即使在数据倾斜严重的大规模高维数据集上，GCSM 算法虽然候选产生阶段耗时比 GCS 算法要长，但是依旧能在一秒以内返回查询结果.

同时本文还提出了三种优化技术，这些优化技术均可作用于 GCS 算法和 GCSM 算法的候选产生阶段以及 GCS 算法的查询交互阶段. 实验结果表明，每种优化技术的加入都能够提升算法效率.

参　考　文　献

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

[2] Han J, Pei J, Yin Y. Mining frequent patterns without candidate generation//Proceedings of the ACM SIGMOD International Conference on Management of Data. Dallas, USA, 2000: 1-12

[3] Zaki M J, Gouda K. Fast vertical mining using diffsets//Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Washington, USA, 2003: 326-335

[4] Savasere A, Omiecinski E, Navathe S. Mining for strong negative associations in a large database of customer transactions//Proceedings of the 14th International Conference on Data Engineering. Orlando, USA, 1998: 494-502

[5] Boulicaut J F, Bykowski A, Jeudy B. Towards the tractable discovery of association rules with negations//Proceedings of the 4th International Conference on Flexible Query Answering

Systems. Warsaw, Poland, 2000: 425-434

[6] Tan P N, Kumar V. Indirect association: Mining higher order dependencies in data//Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery. Freiburg, Germany, 2000: 632-637

[7] Houari A, Ayadi W, Yahia S B. Mining negative correlation biclusters from gene expression data using generic association rules. Procedia Computer Science, 2017, 112: 278-287

[8] Fang Ming, Xu Jing. Negative correlation based on reliable pattern registration. Journal of Changchun University of Science and Technology (Natural Science Edition), 2011, 34(3): 149-151(in Chinese)
(方明, 徐晶. 基于负相关的可信赖模式匹配. 长春理工大学学报(自然科学版), 2011, 34(3): 149-151)

[9] Yu Ru, Huang Ming-Xuan, Huang Li-Xia. Discovery of matrix-weighted positive and negative association patterns from educational data based on mutual information. Journal of Data Acquisition and Processing, 2015, 30(1): 219-230 (in Chinese)
(余如, 黄名选, 黄丽霞. 基于互信息的教育数据矩阵加权正负关联模式发现. 数据采集与处理, 2015, 30(1): 219-230)

[10] Gao Liang, Xia Bing, Huang Ming-Xuan. Efficient algorithm AWAPM _SPRMII for mining all-weighted positive and negative association patterns. Application Research of Computers, 2015, 32(6): 1642-1648(in Chinese)
(高亮, 夏冰, 黄名选. 一种有效的完全加权正负关联模式挖掘算法 AWAPM_SPRMI. 计算机应用研究, 2015, 32(6): 1642-1648)

[11] Huang Ming-Xuan, Huang Fa-Liang, Yan Xiao-Wei, et al. Weighted positive and negative association rules mining based on dynamic item weight and SCCI framework. Control and Decision, 2015, 43(10): 1729-1741(in Chinese)
(黄名选, 黄发良, 严小卫等. 基于项权值变化和 SCCI 框架的加权正负关联规则挖掘. 控制与决策, 2015, 43(10): 1729-1741)

[12] Duan Qiao-Ling, Li Fen, Zhang Li. An indirect association rule mining algorithm in multiple databases. Software Guide, 2016, 15(9): 49-51(in Chinese)
(段巧灵, 李芬, 张莉. 多数据库中的间接关联规则挖掘算法. 软件导刊, 2016, 15(9): 49-51)

[13] Han J W, Kamber M, Pei J. Data Mining: Concepts and Techniques. 3rd Edition. Waltham, UK: Morgan Kaufmann, 2012

[14] Atzmueller M, Puppe F. SD-Map — A fast algorithm for exhaustive subgroup discovery//Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery. Berlin, Germany, 2006: 6-17

[15] Leeuwen M, Knobbe A. Diverse subgroup set discovery. Data Mining & Knowledge Discovery, 2012, 25(2): 208-242

[16] Carmona C J, Ruiz-Rodado V, del Jesús M J, et al. A fuzzy genetic programming-based algorithm for subgroup discovery and the application to one problem of pathogenesis of acute sore throat conditions in humans. Information Sciences, 2015, 298(C): 180-197

[17] Kavšek B, Lavrač N, Jovanoski V. APRIORI-SD: Adapting association rule learning to subgroup Discovery//Proceedings of the International Symposium on Intelligent Data Analysis. Berlin, Germany, 2003: 230-241

[18] Jovanoski V, Lavrač N. Classification rule learning with APRIORI-C//Proceedings of the Portuguese Conference on Artificial Intelligence on Progress in Artificial Intelligence, Porto, Portugal, 2001: 44-51

[19] Mueller M, Rosales R, Steck H, et al. Subgroup discovery for test selection: A novel approach and its application to breast cancer diagnosis//Proceedings of the International Symposium on Intelligent Data Analysis. Lyon, France, 2009: 119-130

[20] Atzmueller M, Puppe F. SD-Map — A fast algorithm for exhaustive subgroup discovery//Proceedings of the European Conference on Principles of Data Mining and Knowledge Discovery. Berlin, Germany, 2006: 6-17

[21] Grosskreutz H, Rüping S, Wrobel S. Tight optimistic estimates for fast subgroup discover//Proceedings of the Machine Learning and Knowledge Discovery in Databases. Antwerp, Belgium, 2008: 440-456

[22] Klösgen W. Advances in Knowledge Discovery and Data Mining. CA, USA: AAAI Press, 1996

[23] Grosskreutz H. On subgroup discovery in numerical domains//Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases. Bled, Slovenia, 2009: 30-30

[24] Boley M, Grosskreutz H. Non-redundant subgroup discovery using a closure system//Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases. Bled, Slovenia, 2009: 179-194

[25] Lee Y S, Yen S J. Mining multidimensional frequent patterns from relational database//Proceedings of the Asian Conference on Intelligent Information and Database Systems. Quang Binh, Vietnam, 2013: 51-60

[26] Wang Dong-Yan. Application of multidimensional association rules method in psychological measurement. Intelligent Computers and Applications, 2015, 5: 8-11(in Chinese)
(王冬燕. 多维关联规则在心理测量中的应用. 智能计算机与应用, 2015, 5: 8-11)

[27] Ji L, Tan K L, Tung A K H. Mining frequent closed cubes in 3D datasets//Proceedings of the International Conference on Very Large Data Base. Seoul, South Korea, 2006: 811-822

[28] Li Guo-Yan, Shen Xia-Jiong. A quantitative association rules mining algorithm based on matrix. Computer Engineering & Science, 2008, 30(5): 72-74(in Chinese)
(李国雁, 沈夏炯. 一种基于矩阵的多值关联规则的挖掘算法. 计算机工程与科学, 2008, 30(5): 72-74)

[29] Xu W X, Wang R J. A novel algorithm of mining multidimensional association rules//Proceedings of the International Conference on Intelligent Computing. Kunming, China, 2006: 771-777

[30] Vo B, Le B, Nguyen T N. Mining frequent itemsets from multidimensional databases//Proceedings of the International Conference on Intelligent Information and Database Systems. Daegu, Korea, 2011: 177-186

[31] Ye Tao, Yu Li-Xia, Zhang Ya-Ping. A multi-dimensional association rules mining algorithm based on upper triangular matrix-tree union. Software Engineering, 2017, 20(6): 8-11 (in Chinese)
(叶涛，于利霞，张亚平. 基于上三角矩阵构造多叉树的多维关联规则挖掘算法. 软件工程，2017，20(6)：8-11)

[32] Lavrač N, Kavšek B, Flach P, et al. Subgroup discovery with CN2-SD. Journal of Machine Learning Research, 2004, 5(2): 153-188

[33] Bayardo Jr R J. Efficiently mining long patterns from databases. ACM SIGMOD Record, 1998, 27(2): 85-93

ZHANG Jing-Tian, Ph.D. candidate. Her research interests include data mining and entity recognition.

WU Sai, Ph.D., lecturer, M.S. supervisor. His research interests include P2P systems, distributed database, cloud systems, and indexing techniques.

CHEN Gang, Ph.D., professor, Ph.D. supervisor. His research interests include database, information security, and collaborative design.

SHOU Li-Dan, Ph.D., professor, Ph.D. supervisor. His research interests include knowledge engineering, distributed and parallel database.

CHEN Ke, Ph.D., associate professor, M.S. supervisor. Her research interests include privacy protection, database technology, massive data processing and retrieval.

Background

The infrequent itemsets are all the itemsets which are not extracted by standardized frequent itemsets algorithm (such as the APRIORI and FP-growth algorithm). Mining interesting infrequent itemsets from the database is an important task in data mining. So far, infrequent itemsets mining researches are mainly conducted in the traditional dataset and focus on the negative association, negative association rules and indirect association, which only care about the patterns on the entire dataset.

This paper introduces the unexpected subgroup and some related concepts in the multi-dimensional database, aims to find itemsets which are not frequent in the total database but frequent in a slice of the original database, expands the meaningful infrequent itemsets research from the traditional 2D dataset to the multi-dimensional dataset. We proposed a method of finding unexpected patterns related to user's query in real-time with a series of optimization techniques. The experimental results show that our proposed method can detect unexpected patterns in multi-dimensional database effectively and efficiently.

In order to find unexpected subgroup and unexpected itemsets quickly, we introduce an efficient and effective algorithm that can find significant subgroup and frequent itemsets in the multi-dimensional database. Some authors choose to transform the multi-dimensional database into the traditional database and then mine from it, which will lose the relationship of items in the same attribute and require larger space. Although some efficient methods have been conducted on the dataset with more than 2 dimensions, these methods are either difficult to be extended to the higher dimensional dataset or based on the pattern mining of traditional 2D dataset in essence. A lot of the existing methods on the multi-dimensional dataset generate a large amount of invalid overhead during the calculation. To avoid the overhead, we introduce a brand new algorithm with special pruning strategy and reduce the computation.

Our work is a partly research of project "The Research of Theorem and Method in Big City Data". This project mainly focuses on extracting useful information from the city data in order to provide monitoring early warning, decision support and other services.

This research was supported by the National Basic Research 973 Program of China under Grant No. 2015CB352400, the Natural Science Foundation of Zhejiang Province under Grant Nos. LY18F020005, and the National Natural Science Foundation of China under Grant Nos. 61661146001, 61472348 and 61672455.