一种面向数据流 top-k 频繁模式发布的差分隐私保护方案

梁文娟[1),2),3)] 陈 红[1),2)] 赵素云[1),2)] 李翠平[1),2)]

[1)](中国人民大学数据工程与知识工程教育部重点实验室 北京 100872)

[2)](中国人民大学信息学院 北京 100872)

[3)](河南大学计算机与信息工程学院 河南 开封 475001)

摘 要 频繁模式挖掘是事务数据分析的常用技术,面向数据流的频繁模式挖掘具有重要的应用价值.然而当事务为敏感信息时,直接发布频繁模式及支持度会导致个体隐私泄露.差分隐私是一种严格且可证明的隐私保护模型,目前虽然已有基于差分隐私的频繁模式发布方案,但它们大都是面向静态数据做一次性发布的隐私保护.本文是面向数据流频繁模式发布的隐私保护,旨在设计一种兼顾可用性和发布效率的持续发布的差分隐私保护方案.与静态发布方案不同,面向数据流的隐私保护处理面临两大挑战:一是持续发布过程中隐私预算的累计消耗会造成发布结果可用性较低;二是候选模式集增大会造成发布结果误差较大和发布效率较低.为解决隐私预算的累计消耗问题,方案设计了满足 event 级差分隐私的保护机制.该机制可以最大化隐私预算利用率,提高发布结果可用性.为降低候选模式集大小,从而提高发布结果可用性和发布效率,方案首先设计了一种基于模式估计的长事务拆分预处理策略,并对拆分所致的信息丢失率进行了分析和弥补.然后在持续发布阶段,在基于 Cantree 的挖掘中,先基于支持度阈值对候选模式集进一步缩减.基于缩减后的候选模式集,本文设计了一种蓄水池抽样和指数机制(EM)相结合的持续更新发布策略,该策略通过一遍扫描抽样集,在保证可用性和隐私保护级别的前提下提高了发布效率.最后,理论证明了该方案满足 ε-差分隐私,实验结果验证了该方案具有较好的可用性和较高的工作效率.

关键词 模式估计;差分隐私;蓄水池抽样;频繁模式挖掘;事务拆分

中图法分类号 TP392 **DOI 号** 10.11897/SP.J.1016.2021.00741

A Differentially Private Scheme for Top-k Frequent Itemsets Mining Over Data Streams

LIANG Wen-Juan[1),2),3)] CHEN Hong[1),2)] ZHAO Su-Yun[1),2)] LI Cui-Ping[1),2)]

[1)](*Key Laboratory of Data Engineering and Knowledge Engineering of Ministry of Education, Renmin University of China, Beijing* 100872)

[2)](*School of Information, Renmin University of China, Beijing* 100872)

[3)](*College of Computer and Information Engineering, Henan University, Kaifeng, Henan* 475001)

Abstract Frequent Itemset Mining (FIM) is a popular technique for analyzing transactional data. Currently, continually mining frequent patterns over data streams have important application value. However, if these data are sensitive, directly releasing frequent patterns and their true supports may disclose the privacy of individuals. The protection of user privacy while obtaining statistical information is important. Differential privacy, as a rigid and provable privacy model, has recently gained significant attention in frequent itemset mining. Several differentially private schemes for FIM have been proposed. However, most of them are designed for static datasets and focus on privacy preservation for one-shot release of frequent patterns. In this paper, we focus on

收稿日期:2019-07-19;在线发布日期:2020-05-04.本课题得到国家自然科学基金(61532021,61772537,61772536,61702522)资助.**梁文娟**,博士研究生,讲师,主要研究方向为数据隐私保护、数据挖掘和物联网数据管理.E-mail:liangwenjuan@ruc.edu.cn.**陈 红**(通信作者),博士,教授,中国计算机学会(CCF)高级会员,主要研究领域为数据库、数据隐私保护、大数据系统.E-mail:chong@ruc.edu.cn.**赵素云**,博士,副教授,主要研究方向为模糊集、不确定性数据和隐私保护.**李翠平**,教授,博士生导师,主要研究领域为社会网络分析和推荐系统.

differentially private FIM over data streams, and aim to design a private continual release scheme which can not only achieve high data utility and a high degree of privacy, but also offer high time efficiency. Compared to the static release, the privacy preservation processing in FIM over data streams should meet the real-time requirement while achieving high data utility. Therefore, two challenges will be faced when designing the private scheme for FIM over data streams: the first is the cumulative consumption of privacy budget in the continual release process, which may result in low data utility. The second is the enlargement of candidate itemsets, which may cause a high error of the released result and low release efficiency. Some strategies need to be designed to solve the two challenges to promote the data utility and release efficiency. To solve the first challenge, a strategy of event level privacy for FIM over data streams is designed. Benefitting from this strategy, the utilization of the privacy budget for the continual release can be maximized, and thus the data utility can be improved. To solve the second challenge, an efficient transaction splitting method based on count estimation under differential privacy is first proposed in the preprocessing stage. Since splitting may cause a certain amount of information loss, the information loss is then analyzed and compensated in the publishing stage. Benefitting from transaction splitting, both the candidate itemsets and the information loss can be reduced, and thus the error of the released result can also be significantly reduced. Then in the continually publishing stage, when mining based on Cantree, the generated candidate patterns are compressed based on the support threshold to further reduce the error of the released result. Only the patterns whose support is smaller than the threshold can be retained in the sampling set. Based on the reduced sampling set, an incremental update release scheme is designed by combining weighted reservoir sampling with exponential mechanism (EM). The advantage of this combination is the sampling set can be traversed only once when finding the top-k frequent patterns, which can promote the release efficiency while not affecting the privacy level and data utility. By formal privacy analysis, we theoretically show that our scheme is ε-differentially private. Finally, extensive experiments on real-world datasets illustrate that our scheme outperforms the state-of-the-art comparison methods.

Keywords pattern estimation; differential privacy; weighted reservoir sampling; frequent itemsets mining; transaction splitting

1 引 言

频繁模式挖掘(即 FIM)是事务数据分析的常用技术,也是数据挖掘应用的基本构件.随着信息技术的发展,从数据流中挖掘频繁模式的应用越来越广泛,如网络趋势分析、行为模式分析、电信数据欺诈检测等[1-2].然而,当事务数据流中包含敏感信息时,直接发布频繁模式及支持度会泄露用户隐私.下面给出一个直接发布频繁模式造成隐私泄露的示例.假设 t 时刻事务数据流 $S_t=\{\{p_1,p_2,p_3,p_4\},\{p_1,p_3\},\{p_2,p_4\},\{p_1,p_4\}\}$,$S_t$ 中事务分别代表用户 $\{u_1,u_2,u_3,u_4\}$ 的网页浏览记录,p_i 代表特定网页.设定支持度阈值为 0.7,则 t 时刻的频繁模式集为 $\{\{p_1,0.75\},\{p_4,0.75\}\}$.假设攻击目标为第 4 条事务,且攻击者已经掌握了除攻击目标之外的所有事务,通过对比分析已发布频繁模式集,可以推断出 u_4 一定访问过 p_1 和 p_4,即用户 u_4 的隐私被泄露.因此,在面向数据流的频繁模式发布问题中,如何保护用户隐私具有实际的应用背景和重要的研究意义.

差分隐私[3-5]是一种具有严谨理论证明的隐私保护模型.它通过向查询过程或结果中添加随机噪声实现隐私保护,添加的噪声量取决于隐私预算 ε 和敏感度.隐私预算越小,敏感度越大,所需噪声越多,反之越少.相较于密码学技术[6-8]或 k-匿名[9-12]等隐私保护模型,差分隐私优点是不需要提供特殊的攻击假设,即使攻击者拥有最大的背景知识,它也能提供有力的隐私保护.

近年来已有一些频繁模式发布的差分隐私保护方案[13-20]. 在这些方案中，发布任务敏感度为挖掘过程中生成的候选频繁模式数. 因其值较大，所需添加的噪声量很大，从而影响发布结果的可用性. 为降低发布敏感度，现有方案大都通过降低长事务维度来缩减候选频繁模式集. 如文献[13,17]采用基于随机抽样的截断方式对事务降维. 由于截断事务会造成信息丢失，文献[14]提出一种启发式策略来降低截断事务所致的信息丢失率. 文献[15]则提出找一组跟频繁模式相关的基，将事务映射到基上进行降维，从而降低信息丢失率. 此外，文献[16]提出用加权拆分事务的方法降维，以降低信息丢失率. 基于降低维度后的事务集，上述方案大都结合经典的 Apriori[14] 或 FP-Growth[16] 频繁模式挖掘算法来设计频繁模式发布的隐私保护方案.

上述方案的共同特点是它们都是基于静态数据上的 FIM 挖掘算法设计隐私保护方案. 其研究核心是如何在保证差分隐私的前提下提高发布结果可用性，没有兼顾发布效率. 如文献[16]中，为提高拆分事务的准确性，要先找出所有项目的关联关系树 CRT-Tree，然后基于 CRT-Tree 进行事务拆分. 该过程中涉及多次事务扫描，影响发布效率. 再如文献[15,17]中，要先扫描一次事务找出真实频繁模式集的统计信息，然后基于统计结果设计 FIM 挖掘过程中的噪声添加方案，目的是提高隐私预算利用率. 这些方案都涉及多次事务遍历，发布效率低，不适用于面向数据流的处理. 不同于静态发布，面向数据流的隐私保护处理具有以下问题：(1) 面向数据流的频繁模式发布为一个持续发布过程，需考虑持续发布中隐私预算的累计消耗问题；(2) 由于数据流的无限、快速、变化和有序特点，面向数据流的隐私保护处理方案必须高效，具有实时性.

对于问题(1)，现有面向数据流发布的隐私保护研究中，提供了 3 种隐私保护级别的隐私预算分配方案：event 级[21]、user 级[21] 和 w-event 级[22]. event 级只对持续发布过程中用户的一次参与事件进行保护，其隐私保护级别最低. 但其优点是不需考虑隐私预算在多次参与事件中的分配问题，因此用于每次发布的隐私预算会较大，发布结果可用性较高. 其余两种隐私保护方案是对整个时序上或滑动窗口内用户的多次参与事件进行保护，隐私保护级别较高，但可用性较低. 为降低持续发布中隐私预算的累积消耗，本文隐私保护目标为实现 event 级差分隐私保护. 对于问题(2)，面向数据流处理的差分隐私保护机制必须满足以下三条约束：(1) 结合隐私保护处理的发布过程必须实现对事务的单次扫描，以满足数据流处理的特点；(2) 持续发布过程中，由于候选项集的持续增大，会造成发布结果误差较大，因此需设计满足差分隐私的候选模式空间缩减方案，以降低发布结果误差；(3) 在对每个发布时刻的候选模式空间遍历挖掘 top-k 模式时，需设计高效且满足差分隐私的 top-k 持续发布策略以满足实时发布要求.

虽然静态发布方案中都有候选模式集的缩减策略，但它们都是面向静态 FIM 算法设计，存在多次事务扫描、发布效率低等问题，不能满足数据流处理的要求. 因此，本文针对面向数据流频繁模式发布中的隐私泄露问题，基于一种经典的用于增量更新发布频繁模式的数据结构——Cantree[23]，提出了一种兼顾可用性和效率的 event 级隐私保护方案. 主要贡献如下：

(1) 提出了一种面向数据流 top-k 频繁模式发布的 event 级差分隐私保护机制，机制中设计了两种策略，保证在满足差分隐私的前提下提高发布结果可用性及发布效率.

(2) 为缩减候选模式空间，从而降低发布结果误差和提高发布效率，基于模式估计定义，提出了一种满足差分隐私的长事务拆分预处理策略. 其核心思想是基于模式估计先找出待拆分长事务的频繁模式集 CS，基于 CS 设计了一种贪心事务拆分算法来降低信息丢失率. 为满足差分隐私，上述处理中添加了精心计算的拉普拉斯噪声. 为提高发布结果可用性，对拆分所致的信息丢失率进行了分析，并在后续发布阶段对丢失的信息进行了弥补.

(3) 为提高发布效率，在基于 Cantree 的挖掘过程中，首先基于支持度阈值对候选模式集进一步缩减，然后基于缩减后的模式集，提出了一种结合权重蓄水池抽样和指数机制(EM)的持续更新发布策略. 由于发布过程中只需遍历一次候选频繁模式集，发布效率得到了提高.

(4) 从隐私性上证明了所提方案满足 event 级 ε-差分隐私. 通过真实数据集上的实验分析验证了方案具有较高的可用性及工作效率.

本文第 2 节对相关工作进行讨论，并说明我们工作与现有工作的不同；第 3 节介绍差分隐私预备知识、全文所涉及的符号以及问题定义；第 4 节首先详细描述 top-k 模式挖掘的隐私保护方案以及两个关键隐私保护策略，然后证明所提方案满足 ε-差分隐私；第 5 节给出实验验证；第 6 节对本文进行总结，并指出下一步研究工作.

《计算机学报》

2 相关工作

本节主要介绍3类相关工作:面向静态数据频繁模式发布的隐私保护研究、面向数据流数据发布的差分隐私保护研究及非隐私保护的数据流频繁模式挖掘研究.

2.1 面向静态数据频繁模式发布的隐私保护

近年来已有很多差分隐私下频繁模式发布的研究工作[13-20].根据差分隐私的实现机制[3-4]定义,添加的噪声量由隐私预算和发布任务敏感度决定.在频繁模式发布的隐私保护研究中,敏感度为挖掘过程中生成的候选模式数.由于数据库中事务的高维性,频繁模式挖掘过程中生成的候选频繁模式集很大,发布敏感度很高.为提高发布结果可用性,文献[13]分别基于LM机制[2]和的EM机制[4]提出两种top-k频繁模式发布的隐私保护方案.该方案规定候选频繁模式集中只保留长度为l的候选频繁模式,从而降低了敏感度.文献[14]提出一种启发式策略来截断数据库中的长事务,从而缩减候选频繁模式集,降低发布任务敏感度.文献[15]则提出先找出一组Basis set,然后将事务映射到这组基上来降低敏感度.文献[16]提出一种事务拆分方法来降低发布敏感度.由于事务拆分会造成信息丢失,为降低信息丢失率,他们先找出所有项之间的关系,根据这些关系对长事务进行拆分,从而保证拆分后经常一起出现的频繁项保留在同一短事务中.文献[17]提出先找最大频繁模式集以节约隐私预算,提高发布结果可用性.文献[18-19]则基于抽样和截断事务思想,针对高维事务集和大数据上的FIM提出了对应的差分隐私保护方案.此外,文献[20]基于本地化差分隐私技术提出一种频繁模式发布的隐私保护策略.

上述方案中,方案[13-19]是中心化差分隐私保护方案,前提是假设数据管理者为可信第三方;方案[20]则是本地化差分隐私方案,假设数据管理者为不可信第三方.所有方案都只关注静态数据上一次发布的隐私保护,不适用于面向数据流的频繁模式发布问题.

2.2 面向数据流数据发布的差分隐私保护

根据提供的隐私保护强度,现有方案主要分为三种:event级隐私保护[21]、user级隐私保护[21]和w-event级隐私保护[22].Event级隐私保护提供对持续发布中用户一次参与事件的保护,因此不需考虑隐私预算的累计消耗问题.现有event级隐私保护方案[24-29]大都针对无限数值型数据流,围绕简单频数统计任务(如单值计数和、直方图等)的持续发布进行研究.为提高发布结果可用性,很多方案设计二叉树或自适应分区策略来降低发布任务的敏感度.与event级隐私保护不同,User级隐私保护方案是对时序上用户多次参与事件进行保护,需考虑隐私预算在多次事件中的累计消耗问题.为提高发布结果可用性,方案[30-33]大都结合发布任务特点设计自适应抽样策略来最大化时序上隐私预算利用率,提高发布结果可用性.第三种方案为w-event级隐私保护方案[22],它是对w宽度滑动窗口内用户参与事件进行保护,其隐私保护强度介于event级和user级之间.其核心也是时序上隐私预算的分配及最大化利用问题.上述方案都是基于简单频数统计任务研究,所以不适用于频繁模式挖掘这类复杂分析任务.

与本文研究工作最为接近的是文献[34],它提出了一种面向数据流关键模式发布的差分隐私保护方案DP-CPM.DP-CPM实现滑动窗口内关键模式发布的差分隐私保护,而本文是实现对增量更新发布top-k模式的差分隐私保护,两者面向的问题场景不相同.DP-CPM方案中,为了最大化滑动窗口内隐私预算,设计了基于差异计算的抽样发布策略.为了提高数据可用性,设计了缩减候选模式集的策略,并添加拉普拉斯噪声以实现差分隐私.而本文设计了event-级隐私保护的隐私预算分配方案.为了提高发布效率和可用性,本文先设计了高效的长事务拆分预处理策略,然后基于经典的增量更新频繁模式挖掘的数据结构Cantree,设计了基于指数机制和蓄水池抽样发布的隐私保护机制.

2.3 非隐私保护的数据流频繁模式挖掘

不考虑隐私保护的面向数据流频繁模式挖掘的算法很多,本文将这类研究分为两种:(1)精确频繁模式挖掘研究,指挖掘算法返回准确的频繁模式集.Cantree[23]是一种经典的精确增量挖掘算法代表,Cantree树的构建可满足对事务一次扫描的限制,扫描的事务按照预定项顺序插入到前缀树.基于构建好的前缀树,自底向上递归挖掘频繁模式集.Moment[35]、CloStream[36]及TMoment[37]是几种经典的基于窗口模型精确挖掘频繁模式的算法.为降低内存空间利用率,提高挖掘效率,它们对前缀树结构进行改进或设计新型的树结构来减少内存树所占空间,从而降低插入和删除模式时的时间和空间消耗;(2)近似频繁模式挖掘研究.有很多面向数据流的频繁

模式挖掘算法都属于近似挖掘算法，该类算法为提高挖掘效率，利用统计抽样、草图结构或衰减模型快速找出频繁模式，但允许返回结果有一定的误差. 代表性的经典算法有 Lossy Counting[38]、CountSketch[39]、estDec[40]、estDec+[41]等.

本文将基于精确增量挖掘频繁模式的数据结构——Cantree 数据结构，研究持续发布 top-k 频繁模式的差分隐私保护方案.

3　预备知识及问题定义

3.1　差分隐私

设数据集 D 和 D' 具有相同属性结构，两者的对称差记为 $D\Delta D'$，$|D\Delta D'|$ 表示对称差 $D\Delta D'$ 中的记录个数，若 $|D\Delta D'|=1$，则 D 和 D' 为邻居数据集.

定义 1(差分隐私)[3]. 给定一个随机化函数 M，P_M 为 M 所有可能输出构成的集合. 对任意两个邻居数据集 D 和 D' 及 P_M 的任何子集 O，如果算法 M 满足：

$$\Pr[M(D)\in O]\leq \Pr[M(D')\in O]\times e^{\varepsilon},$$

则称算法满足 ε 差分隐私. ε 称为隐私预算，ε 值越大，隐私保护强度越低；ε 值越小，隐私强度越高.

在实践中，为了满足差分隐私保护要求，对不同的问题有不同的实现机制. Laplace 机制(Laplace Mechanism，LM)和指数机制(Exponential Mechanism，EM)是两种基础的差分隐私保护实现机制. 拉普拉斯机制适用于对数值型数据的保护，指数机制则适用于对非数值型数据的隐私保护.

定义 2(Laplace 机制)[4]. 设有函数 $f:D\to R^d$，输入为数据集 D，输出为一 d 维实数向量，则随机算法 $M(D)=f(D)+Y$ 提供 ε-差分隐私保护，其中，$Y\sim \mathrm{Lap}(\Delta f/\varepsilon)$ 为随机噪声，服从位置参数为 0，尺度参数为 $\Delta f/\varepsilon$ 的 Laplace 分布；$\Delta f=\max_{D,D'}[f(D)-f(D')]_1$ 是 $f(D)$ 和 $f(D')$ 之间的一阶范数距离，表示函数 f 的全局敏感度；令 $\sigma=\Delta f/\varepsilon$，其概率密度函数为 $p(x)=\frac{1}{2\sigma}\exp\left(-\frac{|x|}{\sigma}\right)$.

由于 Laplace 机制仅适用于数值型的查询结果，在实际应用中，很多查询结果为实体对象，为此，McSherry 等人[5]提出了指数机制.

定义 3(指数机制 EM)[5]. 设随机算法 M 输入为数据集 D，输出为一实体对象 $r\in O$，O 为 M 的输出域，$u:(D\times O)\to R$ 为可用性打分函数，若算法 M 以正比于 $\exp\left(\frac{\varepsilon u(D,r)}{2\Delta u}\right)$ 的概率从 O 中选择并输出 r，即

$$r:\Pr[r\in O]\propto \exp\left(\frac{\varepsilon u(D,r)}{2\Delta u}\right),$$

则称算法 M 满足 ε-差分隐私. r 表示从输出域 O 中所选择的输出项；$u(D,r)(r\in O)$ 为打分函数，目的是评估输出 r 的优劣程度，分值越高，被选择输出的概率越大；Δu 为打分函数的全局敏感度.

定理 1(序列组合性)[4]. 假设算法 $M(D)$ 有 n 个算法 $M_i(D;z_i)$ 组成，每个 $M_i(D;z_i)$ 满足 ε_i-差分隐私，其中 z_i 代表 M_i 的输出；并且 $z_1=M_1(D)$，$z_2=M_2(D)$，$z_3=M_3(D)$，…，$M(D)=z_n$，则算法 $M(D)$ 满足 $\left(\sum_{i=1}^{n}\varepsilon_i\right)$-差分隐私保护.

3.2　常用符号定义

表 1 总结了文中的常用符号及其含义.

表 1　常用符号表

常用符号	含义	常用符号	含义
S_t	t 时刻的事务流	l_{opt}	事务拆分长度
T	特定事务	ε	隐私预算
I	项目全集	$\hat{F}$	频繁 1-项集
k	top-k 挖掘参数	$\hat{P}$	频繁 2-项集
e	模式	λ	支持度阈值
$C(e)$	e 在事务中出现频数	δ	top-k 模式最大长度
$C_{\max}(e)$	e 最大出现频数	$\widehat{FI}_t$	t 时刻发布的频繁模式集
$C_{\min}(e)$	e 最小出现频数	ρ	错误控制参数($0<\rho<1$)
α	隐私预算分配比例系数	φ	错误控制参数
CS	估计的频繁项集		

3.3　问题定义

3.3.1　频繁模式挖掘相关定义

模式 e，也称项集，是项目全集 I 的任意子集，即 $e\subseteq I$. $C(e)$ 代表数据集中包含模式 e 的事务数，即 e 的出现频次. 支持度指 $C(e)$ 与事务总数的比值. 频繁模式挖掘是指寻找一组模式集，其中每个模式的支持度都大于等于支持度阈值 λ；top-k 频繁模式挖掘是在频繁模式集中找出 k 个支持度最高的频繁模式. 频繁 i-项集是指对于 I 的任意非空子集 I_1，如果 I_1 为频繁模式且包含 i 个项，则称 I_1 为频繁 i-项集.

3.3.2　问题场景描述

一个可信的管理者不断接受一个无限次更新的事务流 S，S 中每个元素为一个事务 $T(T\subseteq I)$，代表某用户的一次参与记录. S 可以用数组形式表示，$S[i]$ 代表 S 中第 i 个元素，如图 1 所示，$S[i]=T_i$. 令 S_t 代表 t 时刻对应的数据流前缀，则在图 1 中可以看出，发布时刻 1 的数据流前缀为 $S_1=\{S[1],$

《计算机学报》

…,$S[4]\}=\{T_1,T_2,T_3,T_4\}$. 时刻 2 对应 $S_2=\{S[1],\cdots,S[7]\}=\{T_1,T_2,T_3,T_4,T_5,T_6,T_7\}$. 令 ΔS_t 代表在发布时间 $t-1$ 到 t 之间更新的事务集合;面向数据流的频繁模式发布是指在一个离散的发布时间序列上,对于每个发布时刻 t,基于 t 时刻的数据流 S_t,持续更新发布 top-k 频繁模式集 $\widehat{FI}_t$.

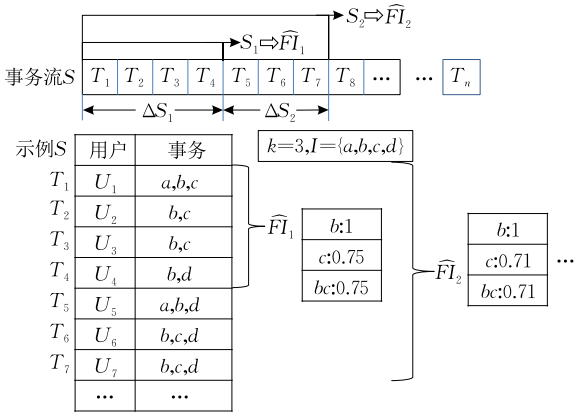

图 1 问题场景描述示例

3.3.3 Event 级邻居数据流定义

为定义数据流上的差分隐私保护,需先定义邻居数据流的概念. 与静态数据上邻居数据集定义类似,event 级邻居数据流是指在发布时刻 t,事务流 S 所对应的两个数据流前缀 S_t 和 S_t' 中,只有一个位置上的元素不同.

定义 4(event 级邻居数据流)[21]. 设 S_t 和 S_t' 为 t 时刻对应的两个数据流前缀,若其只在一个位置 $i(i\in[|S_t|])$ 上对应的元素不同,即

(1) $S_t[j]=S_t'[j],j\in[|S_t|]$ 且 $1\leq j\leq i-1$;

(2) $S_t[j]\neq S_t'[j],j\in[|S_t|]$ 且 $j=i$;

(3) $S_t[j]=S_t'[j],j\in[|S_t|]$ 且 $i<j\leq|S_t|$;

则称 S_t 和 S_t' 为邻居数据流.

例如:图 1 中,发布时刻 1 对应的数据流前缀 $S_1=\{T_1:\{a,b,c\},T_2:\{b,c\},T_3:\{b,c\},T_4:\{b,d\}\}$,则 $S_1'=\{T_1:\{a,b,c\},T_2:\{b\},T_3:\{b,c\},T_4:\{b,d\}\}$ 为 S_1 的一个邻居数据流,两个数据流只有第二个元素不同,即 $S_1[2]\neq S_1'[2]$.

3.3.4 隐私保护目标

本文隐私保护目标为面向事务流持续更新发布 top-k 频繁模式集,持续发布结果需满足 event 级 ε-差分隐私.

定义 5(event 级 ε-差分隐私)[21]. 对于所有 event 级邻居数据流前缀 S_t 和 S_t',给定一个随机化机制 M,若对所有的输出结果 $O\in\mathrm{Range}(M)$,算法 M 满足:

$$\Pr[M(S_t)\in O]\leq\Pr[M(S_t')\in O]\times e^{\varepsilon},$$

则称算法 M 满足 event 级 ε-差分隐私,ε 为隐私预算. 令 $\widehat{FI}_t$ 代表本文算法的输出结果(top-k 频繁模式集),根据定义 5,对于所有的邻居数据流前缀 S_t 和 S_t',本文面向数据流持续发布的隐私保护机制必须满足 $\Pr[M(S_t)\in\widehat{FI}_t]\leq\Pr[M(S_t')\in\widehat{FI}_t]\times e^{\varepsilon}$.

4 面向数据流 top-k 频繁模式发布的隐私保护方案 PrivCE

本文将所设计的隐私保护方案命名为 PrivCE.

4.1 PrivCE 方案核心思想

面向数据流频繁模式发布的差分隐私保护机制必须满足以下三个约束:(1) 一次事务扫描的约束:PrivCE 基于 Cantree[23] 结构实现,Cantree 是一种经典的用于增量更新发布频繁模式的数据结构,基于 Cantree 的隐私保护策略可满足一次事务扫描的处理要求;(2) 缩减持续发布中的候选频繁模式空间,从而降低发布结果误差:PrivCE 中设计了差分隐私下基于模式估计的长事务拆分策略,该预处理策略只需一次事务扫描和较少的处理时间;(3) 提高发布效率以满足实时响应:基于发布时刻的 Cantree 树,PrivCE 中设计了基于权重蓄水池抽样和指数机制相结合的持续更新发布机制,满足差分隐私的同时提高 top-k 模式的发布效率.

具体 PrivCE 隐私保护方案可见算法 1. 该算法中主要包含两个策略:一个策略是基于模式估计的长事务拆分预处理策略来缩减候选模式空间(第2~6 行),基于拆分后事务,构建 t 时刻的 $\mathrm{Cantree}_t$(第 7 行);第二个策略是基于 $\mathrm{Cantree}_t$,设计权重蓄水池抽样和指数机制(EM)结合的 top-k 发布策略,在保护隐私的同时提高发布效率(第 8~9 行). 为满足 event 级 ε-差分隐私,对 t 时刻事务拆分处理分配隐私预算 $\varepsilon_1=\alpha\cdot\varepsilon(0<\alpha<1)$,对 t 时刻基于指数机制的发布分配隐私预算 $\varepsilon_2=(1-\alpha)\cdot\varepsilon$(第 1 行). 根据定理 1,发布方案满足 $\varepsilon=\varepsilon_1+\varepsilon_2$ 差分隐私. 两种策略具体描述参见 4.1 节和 4.2 节,详细隐私分析参见 4.3 节.

算法 1. PrivCE 隐私保护方案.

输入:$\Delta S_t,\lambda,\varepsilon,l_{\mathrm{opt}},k$

输出:$\widehat{FI}_t$

1. $\varepsilon_1=\alpha\cdot\varepsilon;\varepsilon_2=(1-\alpha)\cdot\varepsilon;(0<\alpha<1)$//拆分隐私预算

2. FOR 每一事务 T in ΔS_t

3. 更新频繁 1-项集 $\hat{F}$ 和频繁 2-项集 $\hat{P}$;

4. IF T 的长度$>l_{opt}$ THEN
　R=基于模式估计的长事务拆分($\hat{F}$,$\hat{P}$,T,l_{opt},λ,ε_1);
5. END IF;
6. END FOR;
7. Cantree$_t$=将短事务集 R 更新到 Cantree 树;
8. $\widehat{FI}_t$=基于蓄水池抽样和 EM 发布(Cantree$_t$,λ,ε_2,k,ρ);
　//对 Cantree$_t$ 进行满足 ε_2-差分隐私的挖掘
9. Return $\widehat{FI}_t$;
END 算法 1.

4.2　基于模式估计的长事务拆分预处理策略

随着事务流的持续增量更新,候选频繁模式会出现组合爆炸问题.该问题会给差分隐私保护机制的设计造成两个弊端:(1)候选模式数增加会造成发布结果误差增大(见 4.3 节定理 3 分析);(2)候选模式空间增大也会增加遍历时间,从而降低发布效率.为提高发布结果可用性和发布效率,必须缩减候选频繁模式集大小.现有静态数据上的 FIM 隐私保护方案中,常采用降低长事务维度的策略缩减候选频繁模式集.由于这些策略涉及多次事务扫描,不适用于面向数据流的处理.因此,本节提出一种基于模式估计[40]的长事务拆分策略.该策略只需一遍扫描事务和较少的预处理时间.下面先给出模式估计定义:模式估计指采用统计学方法,基于已读取事务的统计信息,对长事务中的频繁模式集 CS 进行估计.具体定义如下.

4.2.1　模式估计定义

定义 6(模式估计)[40].　一个 n-项集 $e(n\geq 2)$,e 的子集 Sub(e)、e 的长度为 m 的子集 Sub$_m$(e)、e 的长度为 m 的子集的计数集 Sub$_m^C$(e)定义如下:

(1) Sub(e)=$\{a|a\in 2^e-\{e\},a\neq\varnothing\}$,$a$ 由 e 的一个或多个项组成;

(2) Sub$_m$(e)=$\{a|\forall a$ s.t $a\in$Sub(e),$|a|=m\}$;

(3) Sub$_m^C$(e)=$\{C(a)|\forall a$ s.t $a\in$Sub$_m$(e)$\}$,$C(a)$代表模式 a 在事务中出现频次.

对于一个模式 e,当 e 包含的项在事务中经常一起出现时,e 的出现频次和 e 子集的出现频次相同.由此可见,e 的出现频次依赖于其子集一起出现的次数.如果 e 的子集在事务中一起出现的次数越多,e 的出现频次会越高;反之,当 e 的子集在事务中经常独立出现时,e 的出现频次最低.因此,项集 e 的最大出现频次 $C_{max}(e)$为 e 的所有$(n-1)$-子集出现频次的最小值,见式(1):

$$C_{max}(e)=\min(\text{Sub}_{n-1}^C(e)) \tag{1}$$

如果项集,$e=e_1\cup e_2$,$C_{min}(e_1\cup e_2)$代表 $e_1\cup e_2$ 的最小频次,其计算可表示为式(2):

$$C_{min}(e_1\cup e_2)=\begin{cases}\max(0,C(e_1)+C(e_2)-C(e_1\cap e_2)), & \text{if } e_1\cap e_2\neq\varnothing\\ \max(0,C(e_1)+C(e_2)-|D|), & \text{if } e_1\cap e_2=\varnothing\end{cases} \tag{2}$$

令 e_i,e_j代表 e 的任意两个不同的 $(n-1)$-子集,即 $\forall e_i,e_j\in\text{Sub}_{n-1}(e)$,$e=e_i\cup e_j$,则 $C_{min}(e)$为根据式(2)的所有计算结果中的最大值:

$$C_{min}(e)=\max(C_{min}(e_i\cup e_j)) \tag{3}$$

下来给出基于模式估计的长事务拆分过程.

4.2.2　长事务拆分策略描述

长事务拆分是指将一个超出预设长度上限的事务拆分成多个短事务的集合,每个短事务为长事务的子集且满足设定的上限值.拆分可能会导致频繁模式的支持度遭受某种程度的信息丢失.假设事务拆分长度为 l_{opt},l_{opt}值越小,拆分后的候选模式集越小,频繁模式及其支持度的信息丢失程度会越大,发布结果可用性会越低.但根据定理 3 分析,候选模式集越小,噪声误差会越小,发布结果可用性会越高.因此,综合衡量拆分长度对信息丢失率和噪声误差的影响很重要,我们希望找出较好的拆分长度 l_{opt}值来提高发布结果可用性.

基于上述目的,拆分长度 l_{opt}的设置方法如下:假设数据集中事务最大长度为 l_{max},每个长度的事务所占比例为$\langle z_1,z_2,\cdots,z_{l_{max}}\rangle$,设定一个 η 值($0<\eta<1$,表示短事务所占比例),选取满足 $\sum_{i=1}^{j}z_i\geq\eta$ $(1\leq j\leq l_{max})$的最小 j 值为拆分长度 l_{opt}.如在 WV1 数据集中,假设 $\eta=0.8$,则满足条件的 j 值为[20-267],l_{opt}设置为 20.设置方法原理分析:由于数据集中事务长度分布满足短事务较多,长事务较少的特点,同时占大多数比例的短事务的最大长度往往远小于最大事务长度.因此,通过调整 η 值-短事务所占比例(具体见实验 5.1),按照上述方法可以找出一个较好的 j 值,使得基于这个 j 值的拆分后,信息丢失率趋于较低且稳定的状态(原因是当 η 增大时,占 $1-\eta$ 比例的长事务对频繁模式的统计影响会逐渐降低,当 η 增大到一定程度,信息丢失率趋于一个较低水平);同时候选模式集有较大幅度的下降:从 $\sum_{i=1}^{l_{max}}\binom{|I|}{i}$ 到$\sum_{i=1}^{j}\binom{|I|}{i}$(原因是由于数据集长度满足上述分布特点,占 η 比例的短事务的长度上限值小于事务最大长度 $j\ll l_{max}$).

以 WV1 中频繁 1-项集的统计为例,假设 $\eta=0.8$,我们选取 80% 的短事务长度上限为 $j=20$,即

《计算机学报》

80%的事务长度都小于20，20%的事务长度大于20并被认定为长事务，最大事务长度为267.下面分析在不考虑噪声时，j 值对信息丢失率的影响：(1) 随着 j 从1变为20(即 η 逐渐增加至0.8)，频繁1-项集支持度的相对误差由0.33逐步降到0.005(原因：认定的长事务比例在降低，拆分所致的信息丢失率下降)；(2) 当 j 从20逐渐增加至170时(即 η 从0.8逐渐增大)，相对误差由0.005降为0.002，下降幅度不大，基本趋于稳定(原因分析：从0.8开始，长事务比例都比较小，这些较少的长事务对频繁模式的挖掘的影响较小).因此，基于20对WV1中占20%比例的长事务进行拆分，信息丢失率可以降到较低程度，趋于稳定状态.再考虑加上噪声误差：(1) 当 j 从20逐步减小至1时，候选模式空间的降幅不大：从 $\sum_{i=1}^{20}\binom{|I|}{i}$ 降低至 $\sum_{i=1}^{1}\binom{|I|}{i}$(原因：根据数据集中事务长度分布特点，$j\ll l_{\max}$)，但信息丢失率增幅却很大：由0.005逐步增加至0.33；(2) 当 j 从20逐步增加至170时，候选模式空间增幅却很大：从 $\sum_{i=1}^{20}\binom{|I|}{i}$ 大幅度增加至 $\sum_{i=1}^{170}\binom{|I|}{i}$(原因：根据数据集中事务长度分布特点，$j\ll l_{\max}$)，但信息丢失率降幅不大：从0.005到0.002.因此，当 j 为20时，信息丢失率和候选模式空间都会有较好的指标：分别为0.005和 $\sum_{i=1}^{20}\binom{|I|}{i}$.

下面给出基于 l_{opt} 的拆分过程，先给出事务拆分定义.

定义7(事务拆分). 对长事务 T 的拆分处理可形式化为一个 $2^I\to 2^I$ 的概率函数 r：

$$\sum_{T_1\subseteq I,\cdots,T_s\subseteq I}\Pr(r(T)=\{T_1,\cdots,T_s\})=1,$$

$\{T_1,\cdots,T_S\}$ 为 T 拆分后对应的短事务集，I 为项目全集.$r(T)$ 表示对长事务 T 的拆分.长事务 T 被拆分成一组短事务的集合，集合中每个短事务满足 l_{opt} 长度约束，所有短事务的并集等于长事务.下面介绍满足差分隐私的长事务拆分处理过程，包含以下三个步骤：先是频繁1-项集 $\hat{F}$ 和2-项集 $\hat{P}$ 的噪声统计，然后是基于 $\hat{F}$ 和 $\hat{P}$ 的频繁模式集 CS 的估计，最后是基于 CS 的长事务拆分.

频繁1-项集 $\hat{F}$ 和2-项集 $\hat{P}$ 的噪声统计.频繁1-项集是指包含1个项的频繁模式集；频繁2-项集指包含两个项的频繁模式集.在扫描事务流过程中，首先需要统计更新频繁1-项集 $\hat{F}$ 和2-项集 $\hat{P}$ 及计数信息，以备接下来的模式估计过程使用.在统计过程中，为满足差分隐私，需对 $\hat{F}$ 和 $\hat{P}$ 集合中所有模式支持度加上拉普拉斯噪声.由LM机制可知，添加的噪声量由隐私预算和计算任务的敏感度决定.由于该过程分配到的隐私预算为 ε_1，则 $\hat{F}$ 和 $\hat{P}$ 的计算平均分配到的隐私预算为 $\varepsilon_1/2$.下面分析计算任务敏感度：在计算 i-项集过程中，假设 m 为所有可能生成的 i-项集总数，令 $Q=\langle q_1,\cdots,q_m\rangle$ 为计算所有 m 个 i-项集频数的计算函数，$q_j(1\leq j\leq m)$ 表示对第 j 个 i-项集计数值的计算.基于 Q 计算函数定义，频繁 i-项集的计算过程可以表示为首先计算所有 m 个 i-项集的计数值(执行 Q 函数)，然后找出大于阈值的 i-项集集合.如在计算 $\hat{F}$ 时，$i=1,m=|I|$，即首先计算 $|I|$ 个1-项集的计数值，然后从中找出大于阈值的频繁1-项集；在计算 $\hat{P}$ 时，$i=2,m=\binom{|I|}{2}$，计算过程相同.

定理2. 频繁 i-项集计算函数的全局敏感度为 $\binom{l_{\text{opt}}}{i}$.

证明. 由于事务最大长度限制为 l_{opt}，任意添加(删除)一个事务后，受影响的1-项集个数最多为 $\binom{l_{\text{opt}}}{1}=l_{\text{opt}}$，受影响的2-项集个数最多为 $\binom{l_{\text{opt}}}{2}$，以此类推，受影响的 i-项集个数为最多为 $\binom{l_{\text{opt}}}{i}$.因此，$i$-项集计算函数的全局敏感度为 $\binom{l_{\text{opt}}}{i}$. 证毕.

根据定理2，在 $\hat{F}$ 的计算过程中，每个模式的计数所需添加的噪声为 $\text{Lap}(2\cdot l_{\text{opt}}/\varepsilon_1)$；$\hat{P}$ 的计算过程每个模式的计数所需添加的噪声量为 $\text{Lap}\left(2\cdot\binom{l_{\text{opt}}}{2}\Big/\varepsilon_1\right)$.

频繁模式集CS的估计.基于 $\hat{F}$ 和 $\hat{P}$ 的统计信息，对数据流中每个长事务 T 的频繁模式CS进行估计.首先，由于 $\hat{F}$ 和 $\hat{P}$ 代表的是长度小于2的频繁模式集，则先将其加入到CS中，然后基于 $\hat{F}$ 和 $\hat{P}$ 估计长度大于2的频繁模式.由于 T 中可以生成的候选模式个数为 $2^{|T|}$，如果对 T 中所有可能生成的候选模式都做检查，效率较低.因此，首次发布时，算法中估计待拆分长事务中长度不大于 l_{opt} 的频繁模式用于提高拆分效率，后续发布时，根据上个时刻的发布结果 $\widehat{FI}_{t-1}$，获取top-k 模式的最大长度 δ.如果候选模式 e 长度大于 δ，因其对发布结果没有影响，不再估计其是否为频繁模式.在每个长事务的拆分

《计算机学报》

过程中，如果候选模式 e 的长度大于 2 且小于等于 δ，基于式(1)计算 e 的最大计数值 $C_{\max}(e)$. 如果大于支持度阈值 $\lambda|S_t|$，该模式被认定为一个频繁模式. 基于式(3)，计算其最小估计值 $C_{\min}(e)$. 然后将 $(e, C_{\max}(e), C_{\min}(e))$ 作为元素，添加到 CS 中.

基于 CS 的长事务拆分. 当拆分长事务时，为降低频繁模式的信息丢失率，拆分过程需遵循下述原则：如果两个频繁模式在事务中经常在一起出现，拆分时应尽量将其放在一个短事务中. 每个拆分过程本质上是在当前待拆分的长事务 T 中找出一个最优短事务的过程，该过程需执行 $\lceil |T|/l_{\text{opt}} \rceil$ 次. 为了衡量一个短事务是否最优，对每个生成的候选模式 e，都设计了一个权重值：

$$f(e).\text{weight} = \gamma C_{\max}(e) + (1-\gamma) C_{\min}(e) \quad (4)$$

式中 $\gamma(0<\gamma<1)$ 为调整两个估计值的比例参数.

问题 1(最优短事务查找). 给定一个长事务 T 及其频繁模式估计集 CS，最优短事务 t_{temp} 查找过程可形式化如下：

目标：最优短事务 t_{temp}

约束：

$$\begin{cases} (1)\ \max(f(t_{\text{temp}}).\text{weight}); \\ (2)\ f(t_{\text{temp}}).\text{weight} = \sum\limits_{X_j \in CS \wedge X_j \subseteq t_{\text{temp}}} (f(X_j).\text{weight}); \\ (3)\ t_{\text{temp}} \subseteq T \wedge |t_{\text{temp}}| \leq l_{\text{opt}}. \end{cases}$$

最优短事务的查找为在当前待拆分的长事务 T 中找到一个权重最大且满足上述两个约束条件的短事务. 该问题是一种背包问题，属于一种经典的 NP-hard 问题. 因此，本文采用一种贪心策略：拆分过程中，重复在 T 和 CS 中找到具有最高权重的模式，并将其加入到 t_{temp} 中，直到 T 中所有项都已分配完. 由于拆分时需要考虑上述原则：经常共同出现的模式尽量拆分到相同短事务中. 因此，拆分过程中，对 CS 中的任一模式 e，如果 e 与正在生成的短事务 t_{temp} 有相交项，则增加 e 的权重，以提高 e 分配到 t_{temp} 中的概率. 假设 e 与 t_{temp} 相交项个数为 p，e 权重调整为

$$f(e).\text{weight} = f(e).\text{weight} + (f(e).\text{weight}/|e|) \cdot p \quad (5)$$

具体拆分过程可见算法 2：首先估计待拆分长事务 T 的频繁模式集 CS，计算拆分后应生成的短事务个数(1～12 行)，然后重复执行以下步骤构造每一个最优短事务(13～20 行)：对 t_{temp} 初始化为 $\varnothing$，q 代表 t_{temp} 的长度. 如果 q 小于 l_{opt}，根据式(5)对 CS 中的各个模式 e 调整权重；然后在当前的 CS 和 T 中找出共同包含且权重最高的模式 $e_{\max}$，将其加入到 t_{temp}，对 CS 和 T 进行更新. 重复上述过程直至 T 中所有元素都分配完.

算法 2.　基于模式估计的长事务拆分.

输入：$\delta, \hat{F}, \hat{P}, T, l_{\text{opt}}, \lambda, \varepsilon_1$

输出：一组短事务集合 R

1. 将 $\hat{F}$ 和 $\hat{P}$ 中元素加入 CS：$CS=\hat{F}\cup\hat{P}$；
2. FOR $l=3$ TO δ DO
3. FOR 每一个 T 中长度为 l 的模式 e DO
4. 基于 CS 中($l-1$)-模式及 ε_1 计算 $C_{\max}(e), C_{\min}(e)$；
5. IF $C_{\max}(e)>\lambda|S_t|$ THEN
6. $CS=CS\cup\{(e, C_{\max}(e), C_{\min}(e))\}$
7. END IF
8. END FOR
9. END FOR
10. 计算 T 被拆分成的短事务数目 $m=\lceil |T|/l_{\text{opt}} \rceil$；
11. FOR $p=1$ TO m DO
12. $t_{\text{temp}}=\varnothing$；$q=|t_{\text{temp}}|$；
13. WHILE $q< l_{\text{opt}}$ DO
14. 根据式(5)更新 CS 中每个模式 e 的权重 $f(e).\text{weight}$；
15. 找出 CS 和 T 中共同包含且权重最高的模式 $e_{\max}$；
16. IF $|t_{\text{temp}}\cup e_{\max}| < l_{\text{opt}}$ THEN
17. $t_{\text{temp}}=t_{\text{temp}}\cup e_{\max}$；$T=T-\{e_{\max}\}$；
$q^{+}=|e_{\max}|$；将 $e_{\max}$ 及子集从 CS 中删除；
18. ELSE
19. 添加 t_{temp} 到 R；Break；
20. END IF；END WHILE；END FOR；

END 算法 2.

为帮助理解上述拆分过程，给出一个拆分示例：假设 $S_t=\{\{abcde\},\{abcd\},\{abc\},\{ab\},\{a\}\}$，$\lambda=0.3$，$\gamma=0.5$，$l_{opt}=3$. 在下面事务拆分示例过程中，由于示例事务流 S_t 中包含的事务总数太少，造成频繁模式的真实计数值太小. 如果按照实验设定的隐私预算 ε 值(值域为[0.5,1.5])对 $\hat{F}$ 和 $\hat{P}$ 中所有模式计数添加 Laplace 噪声，较大噪声会干扰较小的计数值，造成示例事务拆分准确性低. 为更清晰地展示基于模式估计拆分长事务的贪心算法，示例过程中先忽略噪声添加过程，包含如下步骤：(1) 基于给定的 S_t，得到频繁 1-项集 $\hat{F}$ 和频繁 2-项集 $\hat{P}$ 的统计信息，$\hat{F}$ 为 $\{(a,5),(b,4),(c,3),(d,2)\}$. 统计得到 $\hat{P}$ 为 $\{(ab,4),(ac,3),(ad,2),(bc,3),(bd,2),(cd,2)\}$，并将其添加到 CS 中. 步(1)对应算法 2 的第 1 行；(2) 假设对第一个长事务 $T=\{abcde\}$ 进行拆分，由于 $\delta\leq 3$，则基于 $\hat{F}$ 和 $\hat{P}$ 依次生成 T 的所有长度小于等于 3 的候选模式，并基于式(1)和式(3)对其计数值进行估计，生成 T 对应的频繁模式集

《计算机学报》

CS,最终 CS 中包含的元素及最大和最小计数为 $\{(abc,3,3),(acd,2,2),(bcd,2,2),(abd,2,2)\}\cup\hat{F}\cup\hat{P}$. 步(2)对应算法 2 的 2～9 行;(3)接下来基于 CS 对 T 进行拆分:首先找出 T 和 CS 中包含的最高权重的模式,结果为$(a,5)$,将 a 添加到 t_{temp} 中并在 CS 中将其删除. 由于目前 t_{temp} 长度小于 l_{opt},继续执行选择最高权重模式的步骤:根据现有 t_{temp} 与 CS 中所有元素的相交元素个数 p,按照式(5)对 CS 中所有元素的权重值进行更新. 更新后 CS 中包含的模式及权重依次为$\{(b,4),(c,3),(d,2)\}\cup\{(ab,6),(ac,4.5),(ad,3),(bc,3),(bd,2),(cd,2)\}\cup\{(abc,4),(acd,2.67),(bcd,2),(abd,2.67)\}$. 基于更新后的 CS,找到最高权重模式$(ab,6)$,将 ab 合并到 t_{temp} 中,并将 ab 与其子集从 CS 中删除. 由于此时 t_{temp} 长度仍小于 l_{opt},则重复上述权重更新过程,找到$(ac,6.75)$,将其并入 t_{temp}. 此时 t_{temp} 长度已达到 l_{opt},则 T 的第一个短事务生成为$\{abc\}$. 此时 T 变为$\{de\}$,由于长度小于 l_{opt},第二个短事务为$\{de\}$. 步(3)对应算法 2 的 11—20 行. 上述贪心拆分的目的是尽量将出现频率高且常在一起出现的频繁模式分在同一个短事务中,从而降低拆分后频繁模式的信息丢失率,提高发布结果可用性.

4.2.3 拆分准确率和效率分析

准确性分析. 假设 $C(e)$, $err(e)$代表模式 e 的真实计数值和估计错误,$err(e)=C_{\max}(e)-C_{\min}(e)$. $\forall e_1,e_2\in CS$,如果能证明根据估计值得出的两个模式之间关系与真实计数值之间的关系一致,就能保证依据式(4)和(5)拆分的正确性. 令 a,b,c,d 分别代表 $C_{\max}(e_1)$, $C_{\min}(e_1)$, $C_{\max}(e_2)$, $C_{\min}(e_2)$. 令 $\gamma=0.5$,由于 $a\geq b,c\geq d$,则 4 个变量 a,b,c,d 之间可能存在的不等关系有以下 6 种情况:(1) $a>b>c>d$; (2) $a>c>b>d$; (3) $a>c>d>b$; (4) $c>a>b>d$; (5) $c>a>d>b$; (6) $c>d>a>b$. 在前面两种情况下,$C(e_1)$一定大于 $C(e_2)$,根据式(4),基于估计值的权重比较为 $0.5a+0.5b>0.5c+0.5d$. 由此可见,基于真实值的比较和基于权重的比较结果一致;在第 3 种情况下,如果权重之间的关系是 $0.5a+0.5b>0.5c+0.5d$,$C(e_1)$也会大于 $C(e_2)$,因为 e_2 的估计错误比 $err(e_2)$低,所以 e_2 两个估计值的均值小于 e_1 的均值;同样,在 $0.5a+0.5b<0.5c+0.5d$ 的情况下,$C(e_1)$也会小于 $C(e_2)$. 剩下的 3 种情况与前 3 种情况分析类似. 除了上述不等值的 6 种情况,还有等值情况,当 $a=c$ 或 $b=d$ 时,如果 $err(e_1)$ 小于 $err(e_2)$,则 $C(e_1)$大于 $C(e_2)$,权重之间的关系比较与真实值比较一致,反之依然. 式(5)的目的是将经常在一起出现的项目拆分到一个短事务中. 假设 e_1 已经被分配到当前短事务中,而 e_2 经常和 e_1 一起出现,在这种情况下,基于式(5)提高 e_2 权重,使得拆分时选择 e_2 的概率得到提高. 因此,基于式(4)和(5)的事务拆分会提高频繁模式发布的准确性.

效率分析. 和现有的拆分方法比,本文的拆分过程效率更高. 现有长事务拆分方法 PFP[16] 实现原理如下:第一遍扫描数据库事务统计 2-项集的支持度;基于统计结果,创建一个无向权重图;采用社区发现算法识别图中项之间的关系,并据此结果建立项目关系树(CR-Tree); 基于 CR-Tree,第二次扫描数据库事务进行拆分. 影响 PFP 效率的原因主要为三点:(1)事务的扫描次数太多;(2)社区挖掘算法所耗费的预处理时间较长,极大影响了拆分效率;(3)拆分时,需要搜索 CR-Tree 中所有项之间的依赖关系来指导拆分过程,搜索空间太大所以影响拆分效率. 而本文长事务拆分方法效率更高,原因如下:(1)事务的扫描次数只需 1 次;(2)模式估计的拆分方法只需估计当前拆分事务的频繁模式集而非全部项之间的依赖关系;(3)拆分时只需根据当前事务的频繁模式集 CS 进行拆分,搜索空间也大大降低,从而提高了发布效率. 因此适用于面向事务流挖掘的特点.

4.2.4 拆分信息丢失率分析

拆分可能导致一些频繁模式支持度降低,因此会带来信息丢失. 为提高发布准确率,本节拟对信息丢失率进行分析并在发布阶段对其进行修正. 假设待拆分事务为 T,其长度为 $l(l>l_{\text{opt}})$,假设 T 包含一个 i-模式 X. 根据文献[14]得知,如果对事务进行截断,模式 X 留在拆分后事务中的概率如下:

$$\Pr_{\text{truncate}(i,l)}(X)=\frac{\binom{l-i}{l_{\text{opt}}-i}}{\binom{l}{l_{\text{opt}}}}.$$

基于上述公式,下面分析拆分后 X 留在$\lceil l/l_{\text{opt}}\rceil$个短事务的概率. T 经过拆分后,会有$\lfloor l/l_{\text{opt}}\rfloor$个长度为 l_{opt} 的短事务和一个长度小于 l_{opt} 的短事务,该事务的长度为 $a=l-(\lfloor l/l_{\text{opt}}\rfloor\times l_{\text{opt}})$.

如果 $a<i$, 则 i-模式 X 拆分后保留在$\lfloor l/l_{\text{opt}}\rfloor$个短事务中某一个事务的概率为

$$\binom{\lfloor l/l_{\text{opt}}\rfloor}{1}\frac{\binom{l-i}{l_{\text{opt}}-i}}{\binom{l}{l_{\text{opt}}}}=\frac{\lfloor l/l_{\text{opt}}\rfloor\binom{l-i}{l_{\text{opt}}-i}}{\binom{l}{l_{\text{opt}}}}.$$

《计算机学报》

如果 $a\geq i$，X 留在最后一个短事务中的概率为 $\dfrac{\binom{l-i}{a-i}}{\binom{l}{a}}$，因此 X 留在所有短事务中某一个的概率为 $\dfrac{\lfloor l/l_{\text{opt}}\rfloor\binom{l-i}{l_{\text{opt}}-i}}{\binom{l}{l_{\text{opt}}}}+\dfrac{\binom{l-i}{a-i}}{\binom{l}{a}}$. 根据上述分析，拆分后 X 留在$\lceil l/l_{\text{opt}}\rceil$个短事务的概率如下：

$$\Pr_{\text{split}(i,l)}(X)=\begin{cases}\dfrac{\lfloor l/l_{\text{opt}}\rfloor\binom{l-i}{l_{\text{opt}}-i}}{\binom{l}{l_{\text{opt}}}}, & \text{if } a<i\\ \dfrac{\lfloor l/l_{\text{opt}}\rfloor\binom{l-i}{l_{\text{opt}}-i}}{\binom{l}{l_{\text{opt}}}}+\dfrac{\binom{l-i}{a-i}}{\binom{l}{a}}, & \text{if } a\geq i\end{cases}.$$

假设所有事务长度满足均匀分布，当前发布时刻事务总数为 n，令 f_k 代表长度为 k 且包含 i-模式 X 的事务数. 根据上述分析，拆分后 X 保留在短事务中的概率为

$$R(X)=\sum_{k=i}^{l_{\text{opt}}}\frac{f_k}{\sum_{j=1}^{n}f_j}+\sum_{k=l_{\text{opt}}+1}^{n}\frac{f_k}{\sum_{j=1}^{n}f_j}\Pr_{\text{split}(i,l)}(X)\quad(6)$$

在后续发布过程中，将基于式(6)对拆分所致的频繁模式支持度的丢失信息进行弥补.

4.3　基于指数机制和权重蓄水池抽样的发布策略

本节将设计基于指数机制和权重蓄水池抽样的发布策略. 在现有基于指数机制 EM 的频繁模式发布文献[13,17]中，普遍采用权重随机抽样选取 top-k 频繁模式. 权重随机抽样需要遍历集合两遍：第一遍计算各元素的权重值 w_i，然后基于权重值计算各元素的抽样概率 $pr(i)=w_i\Big/\sum_{j=1}^{n}w_j$；第二遍基于各元素的抽样概率，执行 k 遍抽样出 top-k 元素. 如果抽样集合较大或者元素个数不确定，遍历两次集合效率太差或不可能实现. 由于持续发布过程中，候选频繁模式空间会逐渐增大，因此将采用蓄水池抽样发布 top-k 模式. 蓄水池抽样优点是对抽样集合只进行一次遍历，效率较高.

为满足差分隐私，蓄水池抽样需与 EM 机制相结合. EM 机制的设计核心是为每个候选频繁模式设计打分函数. 基于模式的分数值和蓄水池抽样技术，可以使抽样出的 top-k 噪声模式满足差分隐私要求. 每个候选模式 e 的打分函数设计如下：

$$e.\text{score}=\exp\left(\frac{\varepsilon' C(e)}{2k}\right)\quad(7)$$

此处分配的隐私预算为 $\varepsilon'=\varepsilon/2k$，发布敏感度为 k，因为添加或删除一个事务最多影响 k 个频繁模式. $C(e)$为模式 e 的出现计数.

top-k 模式发布过程设计的蓄水池抽样算法基于文献[42]的算法思想：为每个元素的抽样权重值定义为 $w_s=r^{1/w_i}$，r 是一个(0～1)之间的随机数，w_i 是元素的权重. 基于算法思想，结合打分函数，每个模式 e 的蓄水池抽样权重定义为

$$e.\text{sw}=r^{1/e.\text{score}}.$$

下面介绍基于上述定义的持续发布过程：本节基于 Cantree[23] 设计持续更新发布 top-k 模式的隐私保护方案. 叙述方便起见，先给出一个事务流 S_t 的 Cantree$_t$ 构建示例(图 2)，然后结合 Cantree$_t$，给出一个条件模式基的构建及挖掘示例，最后分析满足差分隐私的 top-k 模式发布设计. Cantree 是一种持续更新发布频繁模式的经典数据结构. 为实现对事务的一次遍历，Cantree 按照预定义的项顺序建立 Cantree 头表，并按此项顺序依次扫描事务并将其插入到前缀树. Cantree 优点在于头表中包含所有项，每次更新事务时，不需重复调整头表中项的顺序和前缀树中节点及之间的链接关系，因此构建效率较高. 假设 $S_t=\{\{abcf\},\{bch\},\{abcef\},\{bcdh\},\{ag\},\{afg\}\}$，则 S_t 对应的 Cantree 树如图 2 所示.

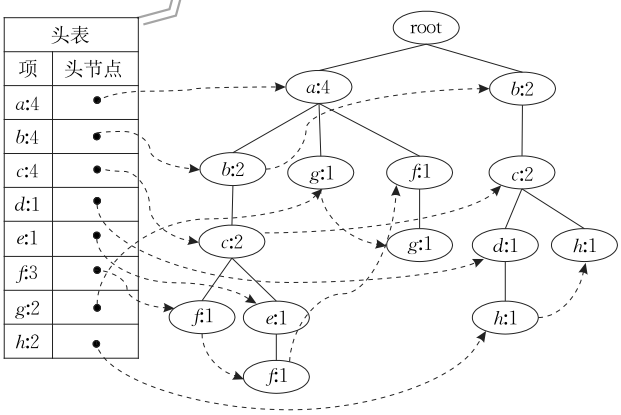

图 2　S_t 对应的 Cantree 树

在每个发布时刻 t，根据构建好的 Cantree$_t$，自底向上对 Cantree$_t$ 递归挖掘频繁模式集. 挖掘过程如下：(1) 挖掘 Cantree$_t$ 头表中的频繁 1-模式；(2) 对头表中每个频繁 1-模式 e_1，首先将其添加到频繁模式集，然后基于 Cantree$_t$ 构建 e_1 的条件模式基 CPB$_{e_1}$——基于 Cantree$_t$ 找出的模式 e_1 的所有前缀路径，以 FP-Tree 结构组织(FP-Tree 特点：过滤

掉前缀路径中小于阈值的项，其头表和前缀树中按频繁项计数值降序排列)；(3)对 CPB_{e_1} 递归挖掘，找出以 e_1 为前缀的频繁模式集. 递归过程描述如下：如果 CPB_{e_1} 为空，则对 CPB_{e_1} 挖掘结束. 如果 CPB_{e_1} 只剩一条路径，则该路径上所有项的组合为以 e_1 为前缀的频繁模式集. 否则对 CPB_{e_1} 头表中每一项 e_2，添加 $\{e_1e_2\}$ 到频繁模式集，构建项集 $\{e_1e_2\}$ 的条件模式基 $CPB_{\{e_1e_2\}}$，对 $CPB_{\{e_1e_2\}}$ 按步(3)继续递归挖掘. 下面给出一个基于 $Cantree_t$ 的挖掘过程示例.

假设阈值为 0.3，基于图 2 示例中的 $Cantree_t$，递归挖掘示例过程如下：(1)挖掘 $Cantree_t$ 中的频繁项：a,b,c,d,f,g,h；(2)自底向上构建 $Cantree_t$ 头表中各频繁项的条件模式基. 首先构建最后一项 h 的条件模式基 CPB_h——即 $Cantree_t$ 中所有包含 h 的前缀路径(以 FP-tree 形式存储)；(3)基于 CPB_h 递归挖掘以 h 为前缀的频繁模式；图 3(a)给出了模式 h 的条件模式基 CPB_h，递归挖掘该条件模式基，由于 CPB_h 只剩一条路径，则该路径上所有项的组合都是频繁模式，即以 h 为前缀的频繁模式为 $\{\{hb:0.33\},\{hc:0.33\},\{hbc,0.33\}\}$. 然后自底向上基于 Cantree 树依次构造 g,f 等项的条件模式基，递归挖掘条件模式基找出以这些项为首的频繁模式集.

为实现差分隐私，在上述递归挖掘过程中，每生成一个频繁模式(如 hb)，根据设计的指数机制(式(7))和该模式的计数值计算其蓄水池抽样权重(见图 3(b))；根据该模式的权重值，判断是否将其添加或替换到蓄水池中，使得最后蓄水池中保留的 top-k 模式满足差分隐私要求. 在上述基于 Cantree 的 top-k 频繁模式挖掘中，在相同的隐私预算下，最终发布结果的误差大小依赖于挖掘过程中生成的候选频繁模式集中的模式个数，候选频繁模式个数越多，发布结果误差越大，反之越小.

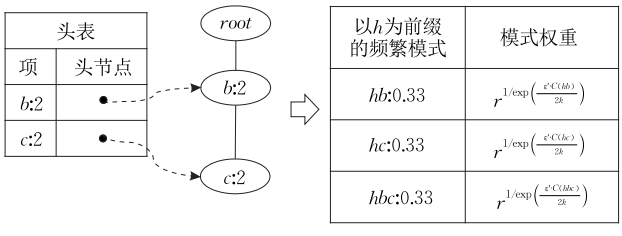

图 3 h 的条件模式基 CPB_h

由于在每个发布时刻 t，经过长事务拆分处理，$Cantree_t$ 中包含的候选频繁模式长度不会超过 l_{opt}，因此抽样集中只包含长度小于 l_{opt} 的候选模式，已经得到了缩减. 如果基于此时的候选模式集抽样发布 top-k 模式，发布结果可用性分析如定理 3.

定理 3(抽样可用性分析). 设 ρ 和 φ 是抽样发布过程的错误控制参数. 对于任意给定的 $\rho(0<\rho<1)$，$\forall e\in\widehat{FI}_t$，$e$ 的发布计数与其真实计数 $C(e)$ 之间小于 φ 的概率不小于 $1-\rho$，其中 $\varphi=O\left(\frac{2k}{\varepsilon'}\ln\left(\frac{k}{\rho}+\ln(|I|^{l_{opt}})\right)\right)$.

证明. 由式(7)可知，抽样过程中，抽取一个计数不小于 $C(e)-\varphi$ 的模式的概率为 $\frac{\exp\left(\frac{\varepsilon'(C(e)-\varphi)}{2k}\right)}{\exp\left(\frac{\varepsilon'C(e)}{2k}\right)}=\exp\left(-\frac{\varepsilon'\varphi}{2k}\right)$. 经过长事务拆分，候选频繁模式数上界为 $O(|I|^{l_{opt}})$，从抽样集中采样一个模式的概率不大于 $|I|^{l_{opt}}\cdot\exp\left(-\frac{\varepsilon'\varphi}{2k}\right)$. 由于最终发布模式数为 k，则抽样 k 个模式的概率不大于 $k\cdot|I|^{l_{opt}}\cdot\exp\left(-\frac{\varepsilon'\varphi}{2k}\right)$. 因此，$\rho\geq k\cdot|I|^{l_{opt}}\cdot\exp\left(-\frac{\varepsilon'\varphi}{2k}\right)$. 经等价变化可得 $\varphi\geq\frac{2k}{\varepsilon'}\left(\ln\frac{k}{\rho}+\ln(|I|^{l_{opt}})\right)$. 证毕.

由定理 3 可知，抽样结果中每个模式 e 的发布计数的误差大小跟 ε',k,ρ 和抽样集的大小相关. ε' 和 ρ 越小，发布计数误差越大；k 和抽样模式集越大，发布计数误差越大. 由于 ε',k,ρ 是计算或预定义参数值，不能修改，为进一步提高发布结果的可用性，可以对拆分后的候选模式集进行进一步缩减，降低抽样集空间. 策略如下，每个发布时刻 t，先基于 ρ 计算抽样计数误差 φ，然后基于 φ 对支持度阈值 λ 进行调整：$\lambda=\lambda-\varphi/|S_t|$($|S_t|$ 为 t 时刻数据流前缀中包含的事务总数，$\varphi/|S_t|$ 是根据给定的错误控制率 ρ 计算的支持度误差). 然后在对 $Cantree_t$ 树的挖掘过程中，每生成一个候选模式，判断其支持度是否小于修正后的 λ. 如果是，则将其排除在抽样集外，如果不小于 λ，才纳入抽样集，计算其抽样权重，结合蓄水池算法将其加入或替换到蓄水池中.

具体发布过程的隐私保护方案如算法 3：在每个发布时刻 t，首先计算抽样一个模式需要分配的隐私预算(第 1 行)；然后基于给定的错误控制参数 ρ 计算计数误差 φ(第 2 行)，根据 φ 计算缩减候选模式的支持度阈值 λ(第 3 行)；基于 λ 对 $Cantree_t$ 进行挖掘(第 4～23 行)：首先挖掘 $Cantree_t$ 头表中的频繁项 $H_{Cantree_t}$(第 4 行)；对 $H_{Cantree_t}$ 中包含的每一个频繁项 e_1，基于式(6)对其进行信息丢失修正(第 5、6

行)；对修正后的候选模式 e_1，基于式(7)和蓄水池抽样算法，计算 e_1 的抽样权重，如果蓄水池不满 k 个模式，则将 e_1 及计数和权重添加到蓄水池，如果蓄水池中已满 k 个模式，且 e_1 的权重比现有蓄水池中模式的最小权重大，就将 e_1 替换入蓄水池中(第 7～13 行)；然后递归挖掘以 e_1 为前缀的频繁模式集(第 14～23 行)：先构造 e_1 的条件模式基 CPB_{e_1}(第 14 行)；如果 CPB_{e_1} 为空，则对 CPB_{e_1} 的挖掘结束(第 15 行)；如果 CPB_{e_1} 只剩一条路径，则重复 6～13 行将以 e_1 为前缀的模式替换或添加到蓄水池中(第 16、17 行)；否则找出 CPB_{e_1} 头表中的频繁项列表 H_{e_1}(第 18、19 行)；对于 H_{e_1} 中的每一项 e_2，递归挖掘以 $\{e_1e_2\}$ 为前缀的条件模式基(第 20～23 行)．在上述挖掘过程可以看出，只有大于 λ 的候选模式才会计算其蓄水池抽样权重，进入到抽样集，参与到蓄水池 top-k 抽样过程．在发布最终结果前，对发布结果集中的频繁模式支持度添加 LM 噪声(第 24～28 行)，所需添加的噪声量为 $\mathrm{Lap}(2k/\varepsilon')$．详细隐私分析可见 4.4 节．

算法 3.　基于蓄水池抽样和 EM 的发布．

输入：$\mathrm{Cantree}_t$，λ，ε_2，k，ρ

输出：$\widehat{FI}_t$

1. $\varepsilon'=\varepsilon_2/2k$；
2. 根据定理 3 基于 ρ 计算 φ；
3. 设定 $\lambda=\lambda-\varphi/|S_t|$；
4. H_{Cantree_t} = 基于 λ 挖掘 $\mathrm{Cantree}_t$ 头表中的频繁 1-项集；
5. For 每一个项 e_1 in H_{Cantree_t} DO
6. 　基于式(6)修正 e_1 的计数值 $C(e_1)=\dfrac{C(e_1)}{R(e_1)}$；
7. 　计算 e_1 的分值 $e_1.score=\exp\left(\dfrac{\varepsilon'.C(e_1)}{2k}\right)$；
8. 　生成(0～1)随机数 $r=\mathrm{Random}()$；
9. 　计算 e_1 的蓄水抽样权重 $e_1.\mathrm{sw}=r^{1/e_1.\mathrm{score}}$；
10. 　找出 reservoir 中最高权重 $e_{\max}$；
11. 　IF $|reservoir|<k$ or $e_1.sw>e_{\max}$ THEN
12. 　$(e_1,C(e_1),e_1.\mathrm{sw})$ 添加或替换到蓄水池 reservoir 中；
13. 　END IF；
14. 　基于 λ 构建 e_1 的条件模式集 CPB_{e_1}；
15. 　IF CPB_{e_1} 为空 Then Continue；
16. 　ELSE IF CPB_{e_1} 只剩一条路径 THEN
17. 　生成所有以 e_1 为前缀的频繁模式，并重复 6～13 于 EM 将这些模式添加或替换到 reservoir；Continue；
18. 　ELSE
19. 　$H_{e_1}=CPB_{e_1}$ 头表中的所有项；
20. 　For 每一个项 e_2 in H_{e_1} DO
21. 　模式 $e_n=e_1\cup e_2$；
22. 　重复步 6～13 基于 EM 将 e_n 添加或替换到 reservoir；基于步 14～23 递归挖掘以 e_n 为前缀的频繁模式集；
23. End FOR；End IF；END FOR；
24. $\widehat{FI}_t=\{e|(e,C(e))\in\mathrm{reservoir}\}$；
25. FOR EACH e in $\widehat{FI}_t$ DO
26. 　$C(e)^+=\mathrm{Lap}(2k/\varepsilon')$；
27. END FOR；
28. RETURN $\widehat{FI}_t$；

END 算法 3.

定理 4(发布算法的复杂度)．　发布算法的时间复杂度远小于 $O(|I|^{l_{\mathrm{opt}}})$，空间复杂度为 $O(k)$．

证明．　假设事务中所包含的项目总数为 $|I|$，事务的最大长度为 $l_{\max}$．未做拆分时，发布过程中的候选模式的搜索空间为 $O(|I|^{l_{\max}})$．拆分后，发布过程的搜索空间降为 $O(|I|^{l_{\mathrm{opt}}})$．经过缩减的候选频繁模式集空间，只有支持度大于 λ 的频繁模式会保留在抽样集中，因此抽样过程的时间复杂度远小于 $O(|I|^{l_{\mathrm{opt}}})$．由于抽样过程中只需空间为 k 的蓄水池来保留过程中的 top-k 模式，因此空间复杂度为 $O(k)$．

证毕．

4.4　隐私分析

定理 5.　算法 3 满足 ε_2-差分隐私．

证明．　用 $(e_1,e_2,\cdots,e_k)$ 代表 t 时刻数据流 S_t 中采用蓄水池抽样抽取出的 top-k 模式，则

$$\Pr(e_1,e_2,\cdots,e_k|S_t)=\prod_{i=1}^{k}\frac{\exp\left(\dfrac{\varepsilon'C(e_i,S_t)}{2\Delta C}\right)}{\sum\limits_{e_i\in\widehat{FI}_t}\exp\left(\dfrac{\varepsilon'C(e_i,S_t)}{2\Delta C}\right)}\tag{8}$$

式(8)中 S_t，S_t' 为邻居数据流，$C(e_i,S_t)$ 指 e_i 模式在数据流 S_t 中的计数，ΔC 为全局敏感度．现在需证明，$\dfrac{\Pr(e_1,e_2,\cdots,e_k|S_t)}{\Pr(e_1,e_2,\cdots,e_k|S_t')}$ 要小于等于 $\exp(\varepsilon_2/2)$，即算法 3 的抽样过程满足 $\varepsilon_2/2$ 差分隐私．证明分为两个过程：(1) 由于是采用蓄水池抽样抽取出 k 个频繁模式，所以需要证明每抽取一个模式的概率小于 $\exp(\varepsilon')$，即 $\dfrac{\Pr(e_i|S_t)}{\Pr(e_i|S_t')}\le\exp(\varepsilon'))$；(2) 由差分隐私的序列组合性，抽样出 k 模式的概率小于等于 $\exp(k\cdot\varepsilon')=\exp(\varepsilon_2/2)$．证明如下：

$$\frac{\Pr(e_i\mid S_t)}{\Pr(e_i\mid S_t')}=\frac{\dfrac{\exp\left(\dfrac{\varepsilon'C(e_i,S_t)}{2\Delta C}\right)}{\sum\limits_{e_i\in\widehat{FI}_t}\exp\left(\dfrac{\varepsilon'C(e_i,S_t)}{2\Delta C}\right)}}{\dfrac{\exp\left(\dfrac{\varepsilon'C(e_i,S_t')}{2\Delta C}\right)}{\sum\limits_{e_i\in\widehat{FI}_t}\exp\left(\dfrac{\varepsilon'C(e_i,S_t')}{2\Delta C}\right)}}$$

《计算机学报》

$$=\frac{\exp\left(\frac{\varepsilon' C(e_i,S_t)}{2\Delta C}\right)}{\exp\left(\frac{\varepsilon' C(e_i,S_t')}{2\Delta C}\right)}\times\frac{\sum_{e_i\in\widehat{FI}_t}\exp\left(\frac{\varepsilon' C(e_i,S_t')}{2\Delta C}\right)}{\sum_{e_i\in\widehat{FI}_t}\exp\left(\frac{\varepsilon' C(e_i,S_t)}{2\Delta C}\right)}$$

$$=\exp\left(\frac{\varepsilon'(C(e_i,S_t)-C(e_i-S_t'))}{2\Delta C}\right)\times\exp\left(\frac{\varepsilon'}{2}\right)\times\frac{\sum_{e_i\in\widehat{FI}_t}\exp\left(\frac{\varepsilon' C(e_i,S_t)}{2\Delta C}\right)}{\sum_{e_i\in\widehat{FI}_t}\exp\left(\frac{\varepsilon' C(e_i,S_t)}{2\Delta C}\right)}\leq\exp(\varepsilon').$$

由于从邻居数据流 S_t,S_t'中抽取某一个频繁模式的概率比值小于 $\exp(\varepsilon')$，则抽取 top-k 频繁模式的概率比值如下：

$$\frac{\Pr(e_1,e_2,\cdots,e_k\mid S_t)}{\Pr(e_1,e_2,\cdots,e_k\mid S_t')}=\frac{\prod_{j=1}^{k}\frac{\exp\left(\frac{\varepsilon' C(e_i,S_t)}{2\Delta C}\right)}{\sum_{e_i\in\widehat{FI}_t}\exp\left(\frac{\varepsilon' C(e_i,S_t)}{2\Delta C}\right)}}{\prod_{j=1}^{k}\frac{\exp\left(\frac{\varepsilon' C(e_i,S_t')}{2\Delta C}\right)}{\sum_{e_i\in\widehat{FI}_t}\exp\left(\frac{\varepsilon' C(e_i,S_t')}{2\Delta C}\right)}}$$

$$=\prod_{j=1}^{k}\frac{\exp\left(\frac{\varepsilon' C(e_i,S_t)}{2\Delta C}\right)}{\exp\left(\frac{\varepsilon' C(e_i,S_t')}{2\Delta C}\right)}\times\frac{\sum_{e_i\in FI_i}\exp\left(\frac{\varepsilon' C(e_i,S_t')}{2\Delta C}\right)}{\sum_{e_i\in FI_i}\exp\left(\frac{\varepsilon' C(e_i,S_t)}{2\Delta C}\right)}$$

$$=\prod_{j=1}^{k}\frac{\Pr(e_i\mid S_t)}{\Pr(e_i\mid S_t')}$$

$$\leq\prod_{j=1}^{k}\exp(\varepsilon')=\exp(k\times\varepsilon')=\exp(\varepsilon_2/2).$$

由此得证，蓄水池抽样满足 $\varepsilon_2/2$ 差分隐私；由定义 2 可知对 top-k 模式加 LM 噪声满足 $\varepsilon_2/2$ 差分隐私，因此，算法 3 满足 ε_2差分隐私. 证毕.

定理 6. 算法 1 满足 ε-差分隐私.

证明. 算法 1 由算法 2、3 组成，由定义 2 可知算法 2 满足 ε_1差分隐私，由定理 5 可知算法 3 满足 ε_2差分隐私，因此，由定理 1 可知，算法 1 满足 $\varepsilon=\varepsilon_1+\varepsilon_2$差分隐私. 证毕.

5 实验与分析

本节将基于真实数据集对所提方案的可用性及效率进行评估和验证. 实验环境为 Intel(R) Core(TM) i7-3540M CPU (3.00GHz)，8GB 内存，Windows 7 操作系统.

本文将发布方案命名为 PrivCE，对比方案命名为 PFP 和 PrivSuper，分别基于文献[16]和文献[17]的思想设计 top-k 发布的隐私保护方案. 所有算法都用 Java 实现. 因发布过程都包含随机性，每种方案都执行 10 次来获取其平均性能指标.

(1) 评估指标

为评估发布结果可用性，采用 F-Score[16] 来衡量发布频繁模式的准确性.

$$F\text{-Score}=2\times\frac{\text{精确度}\times\text{召回率}}{\text{精确度}+\text{召回率}},$$

精确度 $=|\widehat{FI}_t\cap FI_t|/|\widehat{FI}_t|$，召回率 $|\widehat{FI}_t\cap FI_t|/|FI_t|$，$\widehat{FI}_t$是经隐私保护处理后发布的频繁模式集，$\widehat{FI}_t$是真实频繁模式集.

为评估发布方案的发布效率，采用发布算法的运行时间来进行衡量.

(2) 数据集

实验数据集为 PUMSB①、POS[43]、BMS-WebView-1(WV1)和 BMS-WebView-2(WV2)[43]，具体信息见表 2.

表 2 数据集信息表

Dataset	$\lvert D\rvert$	$\lvert I\rvert$	Max$\lvert t\rvert$	AVG$\lvert t\rvert$
PUMSB	49046	2088	63	50.0
POS	515597	1657	164	6.5
WV1	59602	497	267	2.5
WV2	77512	3340	161	5.0

表中$|D|$表示数据集中事务总数，$|I|$表示数据集包含的项目总数，Max$|t|$表示事务的最大长度，AVG$|t|$表示事务的平均长度.

持续发布过程中，每次发布随机选择 $5\%\times|D|$ 的事务作为 ΔS_i. 评估指标值为多次发布的平均值.

5.1 l_{opt}对可用性的影响

从 4.2 节中(2)可知，事务拆分会缩减候选模式集，从而提高发布结果可用性，但同时拆分也会造成模式计数信息丢失，从而降低发布结果可用性. 因此需找出一个较优 l_{opt}，使得在降低信息丢失率的同时降低候选模式集空间. 在本文发布方案中，l_{opt} 设置为至少占所有事务数 η 比例的短事务的长度下界值. 我们在实验中观察发布结果可用性如何随着 η 值变动进行变化.

实验参数设置情况如下：PUMSB 为稠密数据集，因此 PUMSB 上 λ 为 0.7，其余三个数据集为稀疏数据集，因此 POS 上 λ 为 0.08，WV1 上 λ 为 0.01，WV2 上 λ 为 0.012；α 和 γ 为比例系数，因此设置为 0.5；ρ 为 0.01，ε 为 1，η 从 0.5 变为 0.9. 从图 4 可知，当 η 较小($\leq$0.6)时，F-Score 值相对较

① Frequent itemset mining dataset repository. http://mi.ua.ac.be/data

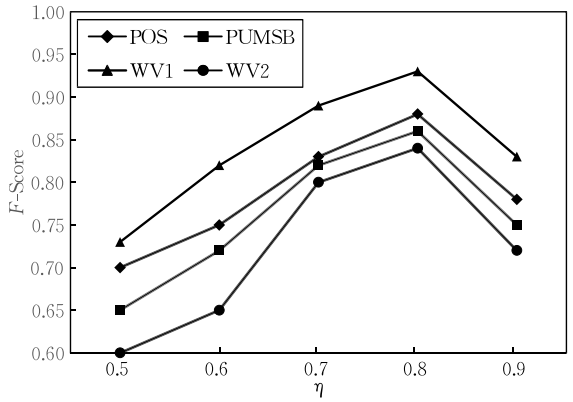

图 4　η vs. F-Score

低，在大部分数据集上可用性都小于 0.75. 原因如下：拆分长事务时，设置的拆分长度越大，模式计数信息丢失率越低；反之，设置的拆分长度越小，模式计数的信息丢失率会越高. 当 $\eta \leq 0.6$ 时，意味着 t 时刻有至少 40%的事务都被认定为长事务，需要进行拆分，从而造成信息丢失过大，影响了发布结果可用性. 当 η 逐渐增加时，可以发现可用性在改善. 当 η 位于 0.7～0.8 之间时，可用性在数据集上已经超过 0.8. 当 η 到达 0.8 时，可用性达到最高. 原因是只有 20%的事务被认定为长事务，拆分后对候选模式计数影响降低. 同时，候选模式空间也有了一定的缩减，从而兼顾了降低信息丢失率和降低候选模式集的双重目标. 当 η 大于 0.8 时，虽然被拆分的事务数减少了，信息丢失率降低了，但是没有明显的缩减候选模式集，所以可用性又有了回落.

5.2　可用性评估

本节将对三个评估方案的可用性进行比较. 参数设置如下：PUMSB 为稠密数据集，因此 PUMSB 上 λ 为 0.7，其余 3 个数据集为稀疏数据集，因此 POS 上 λ 为 0.08，WV1 上 λ 为 0.01，WV2 上 λ 为 0.012；α 和 γ 为比例系数，因此设置为 0.5；ρ 为 0.01，k 值从 10 变化为 200，ε 从 0.5～1.5.

(1) k 的影响

设置 $\varepsilon=1$，观察 k 值变化在 4 个数据集上对发布结果的影响. 从图 5(a)～(d)可以看出，PrivCE 的发布结果可用性最好. 在 4 个数据集上，PrivCE 要比 PrivSuper 的可用性数值在所有情况下高出 3%～5%. 其主要原因在于 PrivCE 中采用的是基于模式估计的拆分，而 PrivSuper 中是基于随机抽样的拆分. 后者因拆分所导致的信息丢失率较大. 在所有情况下，PrivCE 要比 PFP 的可用性高 5%～9%. 原因如下：在拆分时，PrivCE 方案只需根据拆分事务的频繁模式集进行拆分，而 PFP 则需要提前找出所有事务包含的项目之间的依赖关系，并根据结果进行拆分. 因此，PrivCE 拆分会更准确. 一般情况下，PrivSuper 发布结果可用性要比 PFP 高 2%～4%. 原因在于 PrivSuper 直接查找最大频繁模式集，然后将其子集加入到发布结果中，第 2 个步骤没有额外的消耗隐私预算，因此准确性相对较高. 随着 k 值的增加，F-Scores 值都在降低，原因是噪声量逐渐增大所致.

(2) ε 的影响.

设置 $k=100$，观察随着 ε 值的变化，3 种方案发布结果的变化. 从图 5(e)～(h)看出，在相同的隐私预算下，PrivCE 要比其他两种方案性能好 3%～10%，PrivSuper 要比 PFP 高 2%～5%. 随着隐私预算的降低，所有发布方案的 F-Score 值都有所降低. 原因是隐私预算越低，发布过程所需添加的噪声越多. 在相同的隐私预算下，POS 和 WV1 上的 F-Score 值要比 PUMSB 和 WV2 上的值高，原因是 POS 和 WV1 在拆分过程中信息丢失率相对较低.

5.3　效率评估

本节将对 3 种方案的效率进行评估. 所用参数设置如下：POS 数据集上 λ 为 0.08，PUMSB 上 λ 为 0.7，WV1 上 λ 为 0.01，WV2 上 λ 为 0.012，α 和 γ 为 0.5，ρ 为 0.01，η 为 0.8. 当 k 值从 10 增加到 100 时，观察 3 种方案在 4 个数据集上的运行时间.

(1) k vs. 运行时间

从图 6(a)～(d)可以看出，PrivCE 效率比 PrivSuper 高，一般情况下，其运行时间要少 8 s～50 s. 原因是每次发布 PrivSuper 需要执行频繁模式挖掘算法两次，第一次执行非隐私保护的挖掘算法找出真实 top-k 频繁模式的最大长度等统计信息，第二次挖掘算法执行中利用上述统计信息节约隐私预算来提高发布结果可用性. PrivCE 执行时间要比 PFP 少 145 s～255 s. PFP 效率低是因为它的预处理时间较长. 为提高发布可用性，降低信息丢失率，需要提前找出全部项之间的关系，造成预处理时间过长. 而 PrivCE 对预处理过程进行了改进，拆分长事务时，只需基于该事务的估计频繁模式集进行拆分，搜索空间大大降低，从而提高了发布效率；同时，基于蓄水池抽样的发布策略也能进一步提高发布效率. 因此，3 种发布方案中，PrivCE 的执行效率最高. 随着 k 值的增大，3 种方案的执行时间都有所增加，原因是发布处理过程耗时增加.

(2) 基于模式估计拆分优点评估

为了更好地评估基于模式估计拆分优点，本节将基于模式估计的拆分方法应用到 PFP 方案中，并

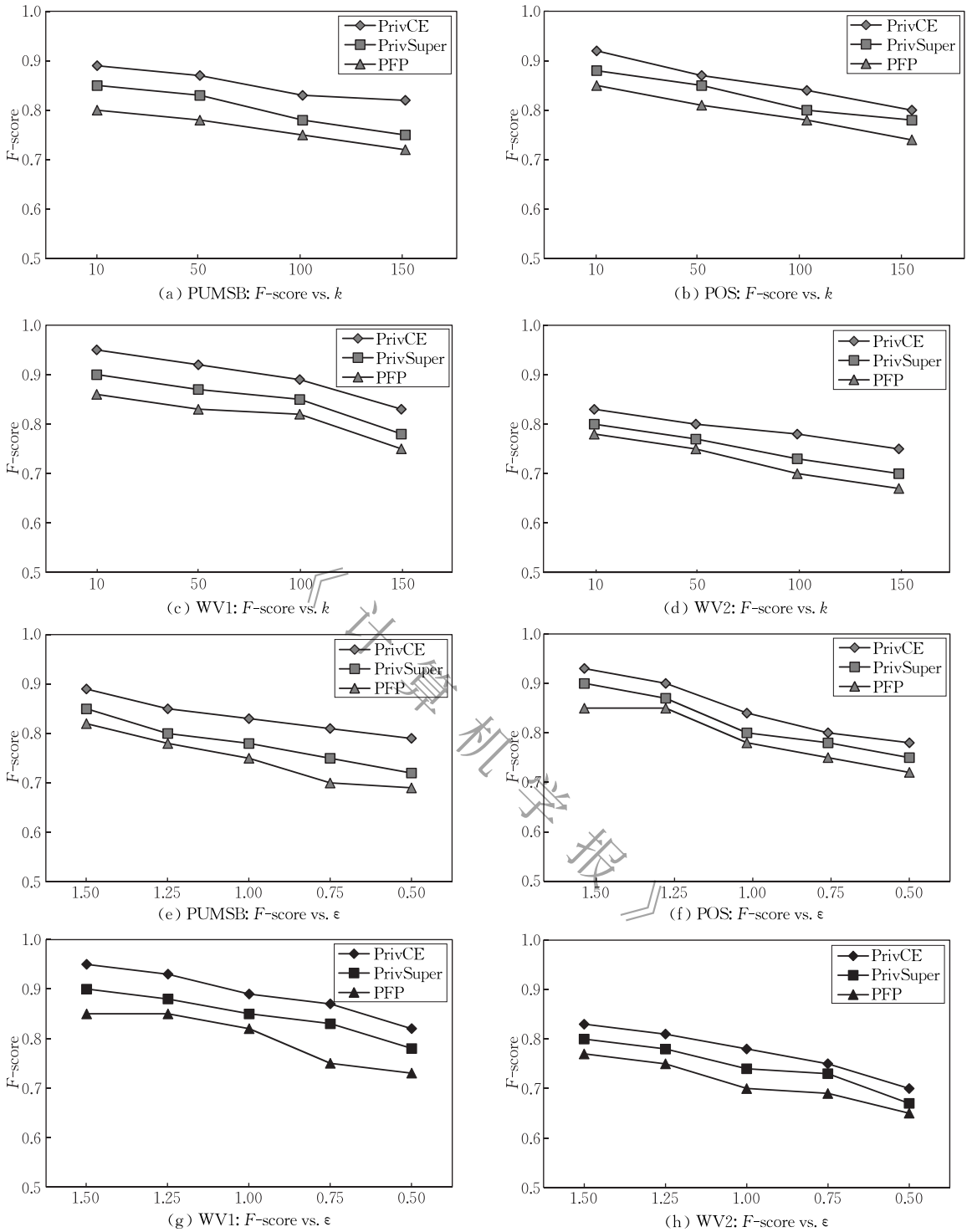

图 5 可用性评估

与原有方案作对比. 参数设置如下:ε 为 1.0, k 值变化从 10～200, λ 在 POS 上为 0.08, 在 PUMSB 上为 0.7, 在 WV1 上为 0.01, 在 WV2 上为 0.012, α 和 γ 为 0.5, ρ 为 0.01, η 为 0.8. 令 PFP+CE 代表修改后的方案, 从图 6(e)～(f)中可以看出, 当 k 值从 10 增加到 200 时, 改进后的方案 PFP+CE 运行时间平均要比 PFP 少 200 s. 其主要原因是 PFP+CE 只需基于拆分事务的频繁模式集进行拆分, 相比于 PFP, 减少了查找全部项之间关系的时间, 因此效率得到了提高.

(3) 基于蓄水池抽样发布优点评估

为了更好地评估蓄水池抽样发布优点, 本节将基于蓄水池抽样的发布方法应用到 TF[13] 方案中, 并与原有方案作对比. 参数设置如下:ε 为 1.0, k 值

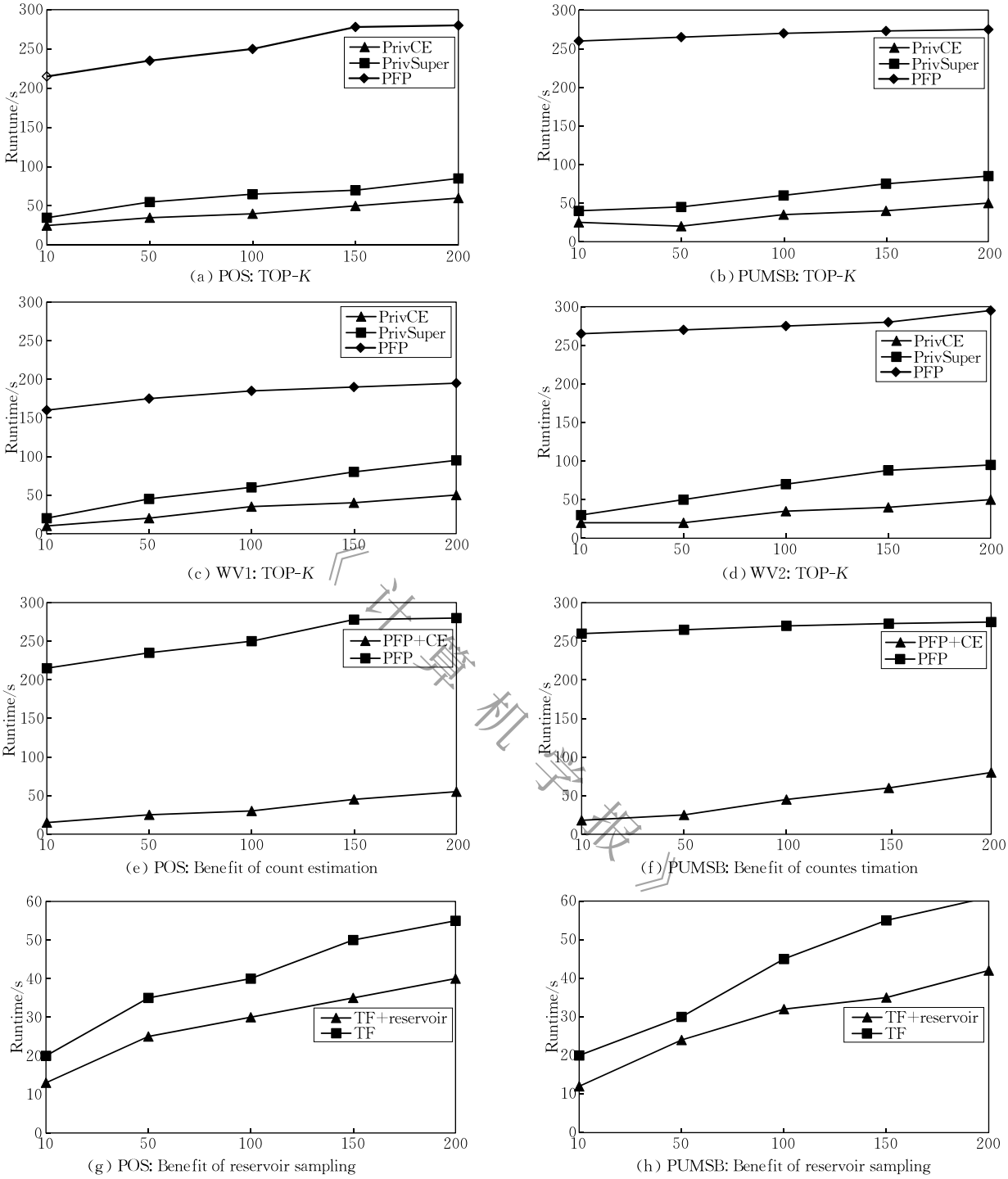

图 6　运行时间评估

变化从 10 到 200，λ 在 POS 上为 0.04，在 PUMSB 上为 0.5. 令 TF+reservoir 代表修改后的方案，从图 6(g)～(h)可以看出，修改后方案在每个数据集上执行时间会减少 8 s～20 s. 原因是基于蓄水池抽样的发布方案只需扫描候选频繁模式集一遍，而传统 TF 方案需要扫描两遍，因此改进后的方案提高了发布效率.

6　总　结

针对面向数据流持续发布 top-k 频繁模式中的隐私泄露问题，本文提出了一种满足 event 级 ε-差分隐私的保护方案. 方案中设计了两种策略提高发布结果可用性和发布效率：一是基于模式估计的长

事务拆分策略,在降低候选模式空间的同时提高发布效率;二是蓄水池抽样和 EM 机制相结合的发布策略提高 top-k 模式发布效率.我们不仅理论证明了该方案满足 ε-差分隐私,并且实验验证了该方案的高可用性和效率.下一步工作将面向数据流频繁模式挖掘任务,提出具有更高隐私保护级别和可用性的持续发布方案.

参 考 文 献

[1] Garofalakis M N, Gehrke J, Rastogi R. Querying and mining data streams: You only get one look a tutorial//Proceedings of the International Conference on Management of Data (SIGMOD). Madison, USA, 2002: 635

[2] Tanbeer S K, Ahmed C F, Jeong B, Lee Y. Sliding window-based frequent pattern mining over data streams. Information Sciences, 2009, 179(22): 3843-3865

[3] Dwork C, Roth A. The algorithmic foundations of differential privacy. Foundations and Trends in Theoretical Computer Science, 2014, 9(3/4): 211-407

[4] Dwork C, McSherry F, Nissim K, Smith A D. Calibrating noise to sensitivity in private data analysis//Proceedings of the International Conference on Theory of Cryptography (TCC). New York, USA, 2006: 265-284

[5] McSherry F, Talwar K. Mechanism design via differential privacy//Proceedings of the IEEE Annual Symposium on Foundations of Computer Science (FOCS). Providence, USA, 2007: 94-103

[6] Nanavati N R, Jinwala D C. A novel privacy-preserving scheme for collaborative frequent itemset mining across vertically partitioned data. Security and Communication Networks, 2015, 8(18): 4407-4420

[7] Wong W K, Cheung D W, et al. Security in outsourcing of association rule mining//Proceedings of the International Conference on Very Large Data Bases (VLDB). University of Vienna, Austria, 2007: 111-122

[8] Wong W K, Cheung D W, et al. An audit environment for outsourcing of frequent itemset mining. Proceedings of the VLDB Endowment (PVLDB), 2009, 2(1): 1162-1172

[9] Atzori M, Bonchi F, et al. Anonymity preserving pattern discovery. Proceedings of the VLDB Endowment, 2008, 17(4): 703-727

[10] Loukides G, Gkoulalas-Divanis A, Shao Jian-Hua. Efficient and flexible anonymization of transaction data. Knowledge and Information Systems, 2013, 36(1): 153-210

[11] Wang Shyue-Liang, Tsai Yu-Chuan, et al. On anonymizing transactions with sensitive items. Applied Intelligence, 2014, 41(4): 1043-1058

[12] Lin Jerry Chun-Wei, Liu Qian-Kun, Fournier-Viger P, et al. PTA: An efficient system for transaction database anonymization. IEEE Access, 2016, 4: 6467-6479

[13] Bhaskar R, Laxman S, et al. Discovering frequent patterns in sensitive data//Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD). Washington, USA, 2010: 503-512

[14] Zeng Chen, Naughton Jeffrey F, Cai Jin-Yi. On differentially private frequent itemset mining. Proceedings of the VLDB Endowment (PVLDB). 2012, 6(1): 25-36

[15] Li Ning-Hui, Qardaji W H, Su Dong, Cao Jian-Neng. Privbasis: Frequent itemset mining with differential privacy. Proceedings of the VLDB Endowment (PVLDB), 2012, 5(11): 1340-1351

[16] Su Sen, Xu Sheng-Zhi, et al. Differentially private frequent itemset mining via transaction splitting//Proceedings of the IEEE International Conference on Data Engineering (ICDE). Helsinki, Finland, 2016: 1564-1565

[17] Wang Ning, Xiao Xiao-Kui, et al. PrivSuper: A superset-first approach to frequent itemset mining under differential privacy//Proceedings of the IEEE International Conference on Data Engineering (ICDE). San Diego, USA, 2017: 809-820

[18] Xu Jing-Xin, Han Kai, et al. PrivBUD-Wise: Differentially private frequent itemsets mining in high-dimensional databases//Proceedings of the Web and Big Data—Third International Joint Conference, APWeb-WAIM 2019. Chengdu, China, 2019: 110-124

[19] Xiong Xin-Yu, Chen Fei, et al. Frequent itemsets mining with differential privacy over large-scale data. IEEE Access, 2018, 6: 8877-8889

[20] Wang Tian-Hao, Li Ning-Hui, Jha Somesh. Locally differentially private frequent itemset mining. IEEE Symposium on Security and Privacy, 2018: 127-143

[21] Dwork C. Differential privacy in new settings//Proceedings of the 21st Annual ACM—SIAM Symposium on Discrete Algorithms (SODA). Austin, USA, 2010: 174-183

[22] Kellaris G, Papadopoulos S, et al. Differentially private event sequences over infinite streams. Proceedings of the VLDB Endowment, 2014, 7(12): 1155-1166

[23] Leung C K-S, Khan Q I, Hoque T. CanTree: A tree structure for efficient incremental mining of frequent patterns//Proceedings of the 5th IEEE International Conference on Data Mining (ICDM 2005). Houston, USA, 2005: 274-281

[24] Dwork C, Naor M, Pitassi T, Rothblum G N. Differential privacy under continual observation//Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC). Massachusetts, USA, 2010: 88-93

[25] Chan T-H H, Shi E, Song D. Private and continual release of statistics. ACM Transactions on Information and System Security, 2011, 14(3): 26: 1-24

[26] Chen Y, Machanavajjhala A, et al. PeGaSus: Data-adaptive differentially private stream processing//Proceedings of the ACM Conference on Computer and Communications Security CCS. TX, USA, 2017: 1375-1388

《计算机学报》

[27] Chen Rui, Shen Yi-Lin, Jin Hong-Xia. Private analysis of infinite data streams via retroactive grouping//Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM). VIC, Australia, 2015: 1061-1070

[28] Bolot J, Fawaz N, et al. Private decayed predicate sums on streams//Proceedings of the 16th International Conference on Database Theory (ICDT). Genoa, Italy, 2013: 284-295

[29] Dwork C, Naor M, et al. Pure differential privacy for rectangle queries via private partitions//Proceedings of the International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT). Auckland, New Zealand, 2015: 735-751

[30] Fan Li-Yue, Xiong Li, Sunderam V S. FAST: Differentially private real-time aggregate monitor with filtering and adaptive sampling//Proceedings of the ACM SIGMOD International Conference on Management of Data SIGMOD. New York, USA, 2013: 1065-1068

[31] Fan Li-Yue, Xiong Li. An adaptive approach to real-time aggregate monitoring with differential privacy. IEEE Transactions on Knowledge and Data Engineering, 2014, 26(9): 2094-2106

[32] Li Hao-Ran, Xiong Li, et al. Differentially private histogram publication for dynamic datasets: An adaptive sampling approach//Proceedings of the 24th ACM International Conference on Information and Knowledge Management. VIC, Australia, 2015: 1001-1010

[33] Zhang Xiao-Jian, Meng Xiao-Feng, Chen Rui. Differentially private set-valued data release against incremental updates//Proceedings of the International Conference on Database Systems for Advanced Applications (DASFAA). Wuhan, China, 2013: 392-406

[34] Wang Jin-Yan, Liu Chen, Fu Xing-Cheng, et al. Crucial patterns mining with differential privacy over data streams. Journal of Software, 2019, 30(3): 648-666(in Chinese)
(王金艳，刘陈，傅星珵等. 差分隐私的数据流关键模式挖掘方法. 软件学报，2019, 30(3): 648-666)

[35] Chi Yun, Wang Hai-Xun, et al. Catch the moment: Maintaining closed frequent itemsets over a data stream sliding window. Knowledge and Information Systems, 2006, 10(3): 265-294

[36] Yen Show-Jane, Wu Cheng-Wei, et al. A fast algorithm for mining frequent closed itemsets over stream sliding window//Proceedings of the IEEE International Conference on Fuzzy Systems. Taipei, China, 2011: 996-1002

[37] Nori F, Deypir M, et al. A sliding window based algorithm for frequent closed itemset mining over data streams. Journal of Systems and Software, 2013, 86(3): 615-623

[38] Manku G S, Motwani R. Approximate frequency counts over data streams//Proceedings of the 28th International Conference on Very Large DataBases. Hong Kong, China, 2002: 346-375

[39] Charikar M, Chen K C. Finding frequent items in data streams//Proceedings of the 2002 International Colloquium on Automata (ICALP). Malaga, Spain, 2002: 693-703

[40] Chang J H, Lee W S. Finding recent frequent itemsets adaptively over online data streams//Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. Washington, USA, 2003: 487-492

[41] Shin S J, Lee D S, Lee W S. CP-tree: An adaptive synopsis structure for compressing frequent itemsets over online data streams. Information Sciences, 2014, 278: 559-576

[42] Efraimidis P S, Spirakis P G. Weighted random sampling with a reservoir. Information Processing Letters, 2006, 97(5): 181-185

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

LIANG Wen-Juan, Ph. D. candidate, lecturer. Her research interests include privacy preserving, data mining and data management in IoTs.

CHEN Hong, Ph. D., professor. Her research interests include database, data privacy preserving, big data system.

ZHAO Su-Yun, Ph. D., associate professor. Her research interests include fuzzy systems, uncertainty data and privacy preserving.

LI Cui-Ping, Ph. D., professor. Her research interests include social network analysis and recommender systems.

Background

Frequent itemset mining (FIM) is a popular technique for analyzing transactional data. Currently, continually mining frequent patterns over data streams have important application value. However, if transactional data are sensitive, directly releasing frequent patterns and their true supports may disclose the privacy. How to protect the privacy of individuals while getting a continuously updated statistics is important.

Differential privacy, the current gold privacy model, has recently gained significant attention in FIM. Several

differentially private schemes for FIM[13-20] have been proposed. However, most of them are designed for static datasets, and cannot meet the requirement of real-time processing. The most related work to ours is Ref. [34]. They focus on crucial patterns mining in a sliding window under differential privacy, while we focus on top-k patterns mining against incremental update over data streams under differential privacy. Our work is the first attempt on this topic.

Compared to the static release, the following two challenges need to be solved in this paper: (1) The cumulative consumption of privacy budget results in low utility. The budget allocation strategies for the continual release in existing related works can be summarized to three types: event-level privacy, user-level privacy and w-event privacy. Event-level privacy hides any single event on time-series from attackers, while the others hide multiple events from attackers. Event level privacy can provide a higher utility but a lower level of privacy, while the others provide a higher privacy level but a lower utility. Our work aims to design an event-level privacy scheme for FIM over data streams to promote utility. (2) When mining frequent patterns over data streams, the enlargement of candidate itemsets results in a high error of the release result and a low efficiency. Therefore, how to efficiently reduce the candidate itemsets needs to be considered. In this paper, an efficient transaction splitting method based on count estimation is first proposed to reduce the candidate set, then based on the reduced candidate patterns set, an incremental update release scheme is designed combining weighted reservoir sampling with EM.

Although this paper provides a possible way to protect the privacy of FIM over data streams, there are also some improvements that could be explored. For example, how to design a privately continual release scheme for FIM over data streams that achieves both a high privacy level (e. g. user-level privacy) and a high utility needs to be explored. Many algorithms of FIM over data streams belong to approximation algorithms, how to design a privately continual release scheme based on approximation algorithms also needs to be explored.

This work is supported by the National Natural Science Foundation of China (Nos. 61532021, 61772537, 61772536, and 61702522).

《计算机学报》