数据流中一种基于滑动窗口的前 K 个频繁项集挖掘算法

张文煜, 周满元
(桂林电子科技大学 计算机科学与工程学院, 广西 桂林 541004)

摘 要: 数据流频繁项集挖掘是当今数据挖掘和知识学习领域重要的研究课题之一。数据流高速性、连续性、无界性、实时性对挖掘算法在时间和空间方面提出了更高的要求。传统的数据挖掘算法由于其存储结构需要频繁地维护,其挖掘方式的精度和速度较低,空间、时间效率不高。在基于粒计算和 ECLAT 算法的基础上提出一种挖掘数据流滑动窗口中 top-K 频繁项集算法,采用二进制方式存储项,利用位移运算实现增量更新,实施与运算计算项集支持度,同时利用二分查找法插入到项目序表中,输出前 K 个频繁项。实验结果表明,该算法在 K 取值不太高时具有较好的时空高效性。

关键词: 数据挖掘; 数据流; 频繁项集; 滑动窗口; 二进制; 二分法
中图分类号: TP311.13;TP301.6 **文献标志码**: A **文章编号**: 1001-3695(2011)07-2519-03
doi:10.3969/j.issn.1001-3695.2011.07.033

Alogrithm for mining top-*K* frequent itemsets over sliding window in data streams

ZHANG Wen-yu, ZHOU Man-yuan
(*School of Computer Science & Engineering, Guilin University of Electronic Technology, Guilin Guangxi* 541004, *China*)

Abstract: Frequent itemset mining over data streams is a hot topic in data mining and knowledge discovery. The features of data streams ,such as consecution, unboundedness and real-time,raise requirements for higher time and space performance of mining algorithms. The storage structure of the present algorithms need to be revised continually. Otherwise,the accuracy and rate of the present algorithms are low. The above two reasons lead up to disadvantage impact on both time and space efficiency. This paper designed a frequent itemset minming algorithm based on algonithm ECLAT and granular computing. It adopted the binary system to mean the item. Carried out displacement operation to update binary data. Meanwhile,used AND-operation for figuring up support threshold of itemset. The frequent itemset was put in OLL (orderd itemset list) by dichotomy,and output the top-*K* itemsets of the OLL. At last,experiments are performed to evaluate the excellent time and space efficiency of the algorithm.

Key words: data mining; data stream; frequent itemset; sliding window; binary; dichotomy

0 引言

数据流[13]是一种连续、高速、无限、时变的有序序列,这种数据广泛存在于现实世界中,如金融分析、传感器网络、天气和环境监控数据、网络监控等。与传统的静态数据库不同,数据流中的数据是无限增长的,无法全部保存下来。对数据流的实时查询,传统的数据挖掘算法显然难以胜任。由于数据流的无限性和流动性等特点,对数据流的挖掘算法设计必须满足一次扫描和较低的时空复杂度要求。近来,针对数据流的频繁项集挖掘出现了许多相关的算法[4~6],而 FP-stream[7] 和 Lossy-Counting[8] 是两个比较有影响的频繁项集挖掘算法。Giannella 等人提出的 FP-stream 结构,用于解决数据流频繁项集的挖掘,采用倾斜时间窗口技术来维护频繁项集以解决时间敏感问题。FU 等人[9]提出的 Itemset-Loop 算法和 Cheung 等人[10]提出的 BOMO 和 Loop-Back 算法是在 Apriori 和 FP-Growth 算法的基础上加以改进,实现了静态数据库 top-*K* 频繁项集的挖掘,但不支持数据的动态增量更新。Babcock 等人[11]研究了分布式数据流上的最大 *K* 个值的查询算法。Metwally 等人[12]研究了数据流最频繁的 *K* 个元素的挖掘算法,但是对于多项集则没有涉猎。

本文提出了一种基于数据流滑动窗口中的 top-*K* 频繁项集动态更新挖掘算法。采用二进制向量表示法将数据按照项集存储,并构建相关的项目序表,以实现数据流中前 *K* 个频繁项集的有效挖掘。

1 问题相关描述及定义

令 *DS* 事务数据流代表一系列连续到来的事务的集合,即 $DS=\{T_1,T_2,T_3,\cdots,T_n\}$。其中:$T_1$ 表示最先到达的事务,T_n

收稿日期: 2010-12-10; **修回日期**: 2011-01-17

作者简介:张文煜(1981-),男,河南开封人,硕士研究生,主要研究方向为数据挖掘(93683604@163.com);周满元(1971-),男,湖南衡阳人,教授,硕导,博士,主要研究方向为 CAD/CAM、网络与信息安全、数据挖掘等.

是最近到达的事务。令 $I=\{I_1,I_2,\cdots,I_m\}$ 代表所有项的集合，I_i 为第 i 个项，T_i 由 I 中的 n 个项组成，即 $T_i=\{\text{tid},k_1,k_2,\cdots,k_n\}$。其中：$k_i\in I$ $(1\leqslant i\leqslant n)$；$n$ 表示该事务中项的数量即事务的大小；tid 是事务的标志符。

定义 1　top-K 频繁项集。将事务数据流中所有的项集按支持度降序排序，设 sp 是排在第 K 位的项集的支持度，则 top-K 频繁项集是数据流中所有支持度不小于 sp 的项集的集合。

性质 1　若项集 IS 属于 top-K 频繁项集，则项集 IS 的所有非空子项集均属于 top-K 频繁项集。

定义 2　项集序表 OIL(ordered item list)是一个长度不小于 K 的有序表，记录当前数据流中的项集及其支持度。OIL 中的每一项是一个二元组〈item，isp〉。其中，item 表示项目名称，isp 表示该项目在当前数据流中的支持度。

2　算法描述

2.1　二进制表示法

表示方法规定：每个项用二进制向量表示，若某个事务包含该项，则该项的二进制向量的相应位置为 1，否则为 0，向量的长度为滑动窗口的大小。

假设滑动窗口大小为 6，当前到来的事务为 T_1、T_2、T_3、T_4、T_5、T_6，设项目表如表 1 所示。二进制向量表示法如表 2 所示。

表 1　事务表

事务	项	事务	项
T_1	a b d	T_4	a e
T_2	c d e f	T_5	b c e
T_3	a d f	T_6	c e f

表 2　二进制向量表示法

项	列表	二进制向量
a	$\{T_1,T_3,T_4\}$	101100
b	$\{T_1,T_5\}$	100010
c	$\{T_2,T_5,T_6\}$	010011
d	$\{T_1,T_2,T_3\}$	111000
e	$\{T_2,T_4,T_5,T_6\}$	010111
f	$\{T_2,T_3,T_6\}$	011001

使用二进制向量表示法的优点是：由于滑动窗口大小固定，事务流源源不断地到来，因此需要不断地移除最老的事务。由于二进制向量表示的特点，可以通过位运算不断地移除最老的事务。在频繁项集 top-K 产生阶段，通过位运算得到获取项集的支持度，可以节约运算的时间开销。此外，以项为单位的存储方式可以大大节约空间开销，有效地降低空间复杂度。

2.2　算法描述 FKIM-SW

FKIM-SW(frequent K itemsets mining over sliding window)算法将 top-K 项集的挖掘过程分为三个阶段：窗口初始化阶段、窗口滑动即增量更新阶段和前 K 项频繁项集产生阶段。

1)窗口初始化阶段

当事务流未将滑动窗口填满时，也就是事务数据流中的事务数量不大于用户预先定义的滑动窗口大小时，进行窗口的初始化。在此阶段，新到事务中的每一项都要被表示成二进制向量形式。例如用户定义滑动窗口大小为 6，当前到来的事务分别为 T_1、T_2、T_3、T_4、T_5、T_6，则窗口的初始化就是将其表示为表 3 的形式。

表 3　二进制向量表示

项	二进制向量表示	项	二进制向量表示
a	101100	d	111000
b	100010	e	010111
c	010011	f	011001

2)滑动窗口增量更新阶段

如果 SW 已满，新事务的到来就使得窗口滑动，此时就需要移除 SW 中最老的事务，即窗口中最先到达的事务。由于二进制向量表示的特点，可以通过将项的二进制向量按位向左移动的方法以实现在当前 SW 中移除最老的事务。当一个新到事务包含对应的项，就在对应的项左移，左移之后低位补 1；若新事务不包含该项，左移之后低位补 0。例如，当 SW 已满，T_7 $\{b,d\}$ 事务到来时，需要先把二进制向量按位左移，把高位覆盖，然后低位相应项补 1 或 0，如表 4 所示。

表 4　SW 增量更新表

项	二进制向量(更新后)	项	二进制向量(更新后)
a	011000	d	110001
b	000101	e	101110
c	100110	f	110010

3)top-K 频繁项集产生阶段

FKIM-SW 算法的挖掘原理用到了 Apriori 的性质，以便于产生频繁项集，此性质为频繁项集的所有非空子集也都必须是频繁的。算法还利用了项目序表，按照项集的支持度降序存储项集的名称和支持度。

以下为算法的具体过程。

```
算法：FKIM-SW
输入：一个事务数据流 TS，用户指定的 K 值和滑动窗口大小|W|。
输出：前 K 个频繁项集。
SW = NULL;
for 滑动窗口中的每个事务 T_i{
//窗口初始化
if(SW! = FULL)
用二进制向量来表示项 I_x;
else
//窗口增量更新
for 事务 T_i 中每一项
按位左移滑动窗口中所有项的高位
并将低位相应位补 0 或 1;
}
//top-K 频繁项集产生阶段
FIS_1 = {频繁 1-项集};
C_k = FIS_{K-1} ∞ FIS_{K-1}//连接操作;
for C_k 中每个候选项 c{
对 C_k 进行按位与操作以得到其支持度;
将每个候选项利用二分查找法插入到项目序表 OIL 中;}
FKIS-Output = {项目序表 OIL 中前 K 项};
```

算法步骤描述：

a)将每个项作为候选 1-项集的 C_1 的成员，并使得 $\text{FIS}_1=C_1$。

b)进行连接操作 $\text{FIS}_{K-1}\infty\text{FIS}_{K-1}$ 产生候选 k-项集 C_k。计算 C_k 中每个候选项的支持度，并利用二分查找法插入到有序项目表 OIL 中。

c)循环步骤 b)，直到没有新的候选项($k+1$)-项集产生

时,算法结束。

d)输出 OIL 中前 K 项项集。

3　实验研究及仿真结果分析

在由 IBM 合成数据发生器产生的数据集上进行了实验研究。实验环境为联想 PC 机,CPU 为 Pentium® 4 3.00 GHz,内存为 512 MB,操作系统为 Microsoft Windows XP 系统。算法由 Microsoft VC++6.0 编码。实验研究以两套 IBM 合成数据发生器[13]产生的顾客购物数据流作为实验的合成数据集 T10I4D1000K 和 T20I81000K 研究了数据流滑动窗口中的增量更新和挖掘时间消耗以及运行中的内存消耗。其中:T 表示数据集中事务的平均长度,I 表示潜在的前 k 项频繁项集的平均长度,D 表示事务流中总的事务数目;数据发生器的其他参数设置为缺省值。数据集的不同项目个数为 10×10^3,前后两套数据集由 100 万个事务组成,频繁项集的平均长度分别为 4 和 8,设定窗口大小为 10×10^4。

3.1　不同数据集和 K 值对算法性能的影响

在两个数据集上测试了算法的性能,如图 1、2 所示。随着要处理的数据流的不断增大,时间花费和内存消耗呈缓慢增长趋势,在 T20I8D1000K 数据集上所用的时间和内存消耗增长较快,这是因为项集相对较长,在二进制转换中需要的时间和内存较多,而且在更新和挖掘中也需要较长的时间。

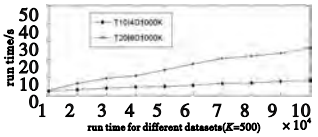

图1　FKIM-SW在不同数据集上的运行时间

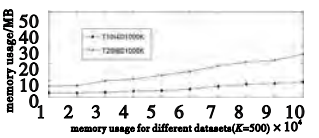

图2　FKIM-SW在不同数据集上的内存开销

通过设置不同的 K 值,测试了 K 值对算法的性能影响。如图 3、4 所示,在数据集 T10I4D1000K 上分别测试了四个不同的 K 值,以检测 K 值不同对算法的性能影响。实验表明,在当前数据集下每种 K 值的时间消耗和内存开销都近似平稳地线性增长,且随着 K 值的不断增大,所花费的资源呈稳定性增长。这是因为转换、更新、查找和插入更多的频繁项集需要较多的时间和内存。

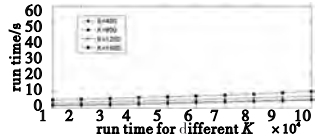

图3　FKIM-SW算法对不同的 K 值运行时间

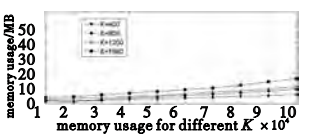

图4　FKIM-SW算法对不同的 K 值内存消耗

3.2　基于滑动窗口的前 K 个频繁项集挖掘算法与其他挖掘算法的性能比较分析

将算法 FKIM-SW 与一种类似的前 K 项挖掘算法(TOPSIL-Miner[14])在时间消耗和内存开销方面作了比较,在数据集 T10I4D1000K 上取 K 值为 2 000,由于算法 TOPSIL-Miner 还需要初始的剪枝阈值,在此次实验中设定为 3.5,实验仿真结果如图 5、6 所示。由于 TOPSIL-Miner 需要不断地进行剪枝操作,且操作的过程中剪枝阈值和挖掘阈值会进行动态的变化,另外该算法中概要结构树的维护更是一个较复杂的过程,因此 FKIM-SW 算法采用二进制方式表示项为支持度的运算大大节省了时间。但由于对项集采用 OIL 的存储方式,随着数据流的不断增大,需要存储的频繁项集增多,就不可避免地增大了内存开销,但整体性能评估 FKIM-SW 算法还是优于 TOPSIL-Miner 的。

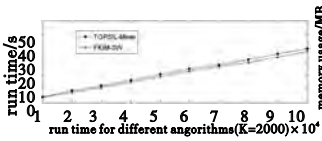

图5　两种算法的运行时间对比图

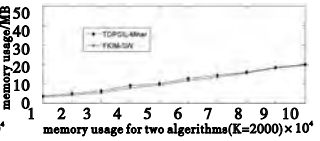

图6　两种算法的内存开销对比图

4　结束语

以数据流中的滑动窗口为研究角度,量化项的存储方式为二进制方式存储,并利用二进制中的位运算实现滑动窗口中事务的增量更新,通过与运算快速计算项集的支持度,同时通过二分查找的方法快速插入到项目序表中,最后输出项目序表的前 K 个频繁项集。通过与经典的算法比较证明,该算法在运行的时间和内存开销方面都具有更加优越的表现。

参考文献:

[1] 敖富江,颜跃进,黄健,等.数据流频繁模式挖掘算法设计[J].计算机科学,2008,35(3):675-678.

[2] GABER M M,ZASLAVSKY A, KRISHNASWAMY S. Mining data streams:a review[J]. ACM SIGMOD Record,2005,34(2):18-26.

[3] 潘云鹤,王金龙,徐从富.数据流频繁模式挖掘研究进展[J].自动化学报,2006,32(4): 594-602.

[4] 王伟平,李建中,张冬冬,等.一种有效的挖掘数据流近似频繁项算法[J].软件学报,2007,18(4):884-892.

[5] CHANG J K,LEE W S. Finding recently frequent itemsets adaptively over online transactional data streams[J]. Information Systems, 2006,31(8):849-869.

[6] 刘学军,徐宏炳,董逸声,等.基于滑动窗口的数据流闭合频繁模式的挖掘[J].计算机研究与发展,2006,43(10):1738-1743.

[7] GIANNELLA C,HAN Jia-wei,PEI Jian, *et al*. Mining frequent patterns in data streams at mutiple time granularities[C]//Proc of Next Generation Data Mining. Menlo Park, CA:AAAI/ MIT, 2003:191-212.

[8] MANKU G S, MOTWANI R. Approximate frequency counts over streaming data[C]//Proc of the 28th International Conference on Very Large Data Bases. San Franciso, CA:Morgan Kanufmann, 2002:346-357.

[9] FU A W C,KWONG R W W,TANG Jian. Mining N-most interesting itemsets[C]//Proc of International Symposium on Methodologies for Intelligent Systems. Berlin:Springer,2009:59-67.

[10] CHEUNG Y L,FU A W C. Mining frequent itemsets without support threshold:with and without item constraints[J]. IEEE Trans on Knowledge and Data Engineering, 2004,16(9):1052-1069.

[11] BABCOCK B,OLSTON C. Distributed top-K monitoing[C]//Proc of ACM SIGMOD International Conference on Management of Data. New York: ACM Press, 2003:8-39.

[12] METWALLY A,AGRAWAL D, ABBADI A E. Efficient computation of frequent and top-K elements in data streams[C]//Proc of International Conference on Data Theory. Berlin:Springer,2005:398-412.

[13] IBM Almaden Research Center. Synthetic data generation code for associations and sequential patterns[EB/OL]. (2010)[2010-05-06]. http://www.almaden.ibm.com/software/quest/.

[14] 杨蓓,黄厚宽.挖掘数据流界标窗口 TOP-K 频繁项集[J].计算机研究与发展[J].2010,47(3):463-473.

[15] 邵峰晶,于忠清,王金龙,等.数据挖掘原理与算法[M].北京:科学出版社,2009:99-106.