时间约束序列模式的有效生成候选项的方法

尹莉莉,郑 诚,郑小波
(安徽大学 计算智能与信号处理教育部重点实验室,安徽 合肥 230039)

摘 要:针对序列模式的几个经典的算法的缺点,提出了一种基于时间约束序列模式的快速产生候选项的方法(TFEGC)。此算法不但避免了频繁的扫描数据库,还考虑了时间限制因素,避免了无用的候选序列的产生,提高了算法运行的时间效率。

关键词:序列模式挖掘;时间约束;候选项;快速产生
中图分类号:TP311 **文献标识码**:A **文章编号**:1674-7720(2011)10-0069-04

Fast generation of candidate-sequences for time constraint sequential pattern mining

Yin Lili, Zheng Cheng, Zheng Xiaobo
(Educational Department Key Laboratory of Intelligent Computing & Signal Processing, AnHui University, Hefei 230039, China)

Abstract: The paper has several key algorithms reviewed briefly, puts forward their faults, and propose a new approach, called Fast Generation of Candidate-sequences for Time constraint Sequential Pattern Mining(TFEGC), this algorithm doesn't need to scan the database frequently, and adds the time limit factor to the algorithm, avoids generating the candidate sequences which are not useful, improves the efficiency of the algorithm's running time.

Key words: sequential pattern; time constraint; candidate; fast generation

序列模式挖掘在很多领域都具有十分重要的意义,比如它可以根据分析顾客购买行为来决定商品的摆放位置,从而制定商场的营销策划。所以,近年来出现了很多序列模式挖掘的改进算法,目前提出算法中,有两类比较典型:GSP[1]算法和采用分治策略来进行模式增长的 PrefixSpan[2]算法。但是这两种算法都存在一定的缺点。参考文献[3]中提出的快速有效的产生候选项的 FEGC 算法,不需要多次扫描数据库,且不需要在前一次迭代的基础上来产生候选项,也不需对非频繁项进行剪枝或修剪,能够达到快速产生候选项的效果。但是,FEGC 算法是针对数据库总体的序列来产生候选项的,有些并不是有效的和用户感兴趣的序列,这在实际应用中就耗费了大量的时间和空间,如分析顾客的购买行为,就不需要将其一月份购买的产品和十二月份购买的产品放在一起进行研究比较。所以本文在 FEGC 算法的基础上将时间限制因素加了进去,可称之为 TFEGC 算法,本算法继承了 FEGC 算法的优点,而且避免了不必要的、无用的一些候选项的产生,提高了算法的运行效率,且在序列结合的过程中,只需检查 uid、$fid(t)$以及 $s(t)$的值,便可知道与哪些项进行结合,无须再进行检验。

1 相关算法介绍

GSP 算法,即广义序列模式算法,使用序列模式的向下封闭性,并采用多次扫描的候选产生-测试方法,它是由 Srikant 和 Agrawal 于 1996 年提出的。它的主要思想是利用序列模式的种子集,即前次扫描得来的序列模式来产生潜在的频繁序列,即候选序列,每个候选序列都会比产生它的种子序列模式多包含一个项。直到一遍扫描不能产生新的序列模式或者新的候选序列时,算法终止。

PrefixSpan 算法是前缀投影序列模式增长算法[4],此算法的特点是不需要产生候选项,算法思想是先找出各个频繁项,然后分别产生它们所对应的投影数据库,接着对各个投影数据库进行挖掘,最后将前缀模式与后缀模式相连得到频繁序列。此算法的缺点就在于可能会产生大量的投影数据库,而且要扫描数据库多次[3]。

FEGC 算法的特点有:(1)Head、Body、Tail 的引入,在一个交易序列中,处于 Head 的项在形成二序列时,只能作为二序列中的 Head,处于 Body 的项在形成二序列

时，既可以作为 Head，也可以作为 Tail；(2)Nid 的引入，Nid 的作用是用来规定在形成候选序列的过程中的 2-序列的结合顺序[1]。

2 TFEGC 算法的理论和相关概念

对于给定的某个顾客的交易数据序列 $B=\{(Item, TID), \cdots\}$，其中 *Item* 表示交易的项，*TID* 表示交易的时间，设 $I=\{(i_1,t_1),(i_2,t_2),\cdots(i_n,t_n)\}$ 是由 n 个不同的项组成的序列，它记录了某个顾客在不同的时间购买的产品，然后根据时间限制将这个序列划分成不同的子序列，再将这些子序列转化为 2-序列[5]，最后以这些 2-序列为基础，找出所有的候选项。

2.1 相关定义

定义 1 已知一个序列 $x=\{(s_1,t_1),(s_2,t_2),\cdots(s_n,t_n)\}$，假设根据时间限制得到序列 $y=\{s_1s_2\cdots s_k\}$，其中 $k\leqslant n$，则称序列 y 是序列 x 的子序列。

定义 2 对于子序列 $s=\{t_1t_2\cdots t_n\}$，如果序列 $s_1=\{t_1t_2\}$，因为 s_1 的长度为 2，则称序列 s_1 为子序列 s 的 2-序列。

定义 3 在 2-序列形成 n-序列的过程中，用函数 *uid* 来限制可以产生与其结合的 2-序列的子序列的个数，具体公式如式(1)所示：

$$uid=l-2 \tag{1}$$

其中，l 是子序列的长度。

定义 4 函数 $fid(t)$ 是用来限制已知 2-序列应该与一个子序列中的哪些 2-序列进行结合，具体公式如式(2)所示：

$$fid(t)=l-T \tag{2}$$

其中，t 是子序列中的项（不包括首尾两项），T 是指项 t 在其子序列中的顺序值。

定义 5 当 $uid>1$ 时，说明已知 2-序列可以和很多个子序列中的 2-序列进行结合，在结合的过程中，因为求得了 $fid(t)$ 的值，就需要明白 $fid(t)$ 指的是哪个子序列产生的 2-序列。这就需要用到函数 $s(t)$ 的值，具体公式如式(3)所示：

$$s(t)=T-1 \tag{3}$$

其中，t 和 T 的意思同上。

2.2 公式的应用及 *td* 的引入

上面的三个公式都是为了解决利用 TFEGC 算法来求候选项时遇到的问题，它们所发挥的作用可通过下面的具体例子来进行详细描述。

由图 1 可知所有的 2-序列过程，下面以子序列<bcde>产生的 2-序列<bc>为例，具体过程如下：

(1)计算出 $uid=4-2=2$，$fid(c)=4-2=2$，$fid(d)=4-3=1$，$s(c)=2-1=1$，$s(d)=3-1=2$；

(2)对于 2-序列<bc>，因为 $s(c)=1$，说明它只能与排在子序列<bcde>之后的第一个子序列产生的 2-序列相结合，由图 1 可知是子序列<cdef>产生的 2-序列；又因为 $fid(c)=2$，说明它只能与<cdef>产生的前两个 2-序列<cd>、<ce>相结合，生成<bcd>、<bce>；又因为 $s(d)=2$、$fid(d)=1$，说明<bcd>还可以与排在子序列<bcde>之后的第二个子序列<def>产生的第一个 2-序列<de>进行结合，生成<bcde>，e 为子序列<bcde>的最后一项，不作考虑。

由上面的步骤(2)可知，虽然求得了 $s(c)$ 的值，但是在算法的实现过程中，并不知道子序列<cdef>就是排在子序列<bcde>之后的第一个子序列，为了解决这个问题，引入了 *td* 的值，为节省存储空间，可以用矩阵的形式来存储 *td* 的值，如图 2 所示。

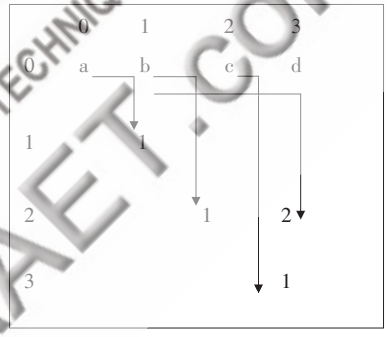

图 2 *td* 的矩阵图

矩阵中没有出现项 e 和项 f，因为项 f 为序列的最后一项，所以它不会与任何二序列进行结合，由图 1 知并没有产生以 e 为首项的子序列，所以不会产生以 e 为首项的 2-序列。

其中，[1,1]=1-0=1，表示排在以 a 为首字母的子序列之后的第一个子序列，是以 b 为首字母的子序列；[2,2]=2-1=1，表示排在以 b 为首字母的子序列之后的第一个序列，是以 c 为首字母的子序列，依此类推。

所以，当算出 $s(t)$ 的值时，只需比较其与 *td* 的值，当 $s(t)=td$ 的值时，则此序列就是所要结合的 2-序列。

3 TFEGC 算法

3.1 算法思想

本算法一共有四大步，第一步是根据时间限制将原

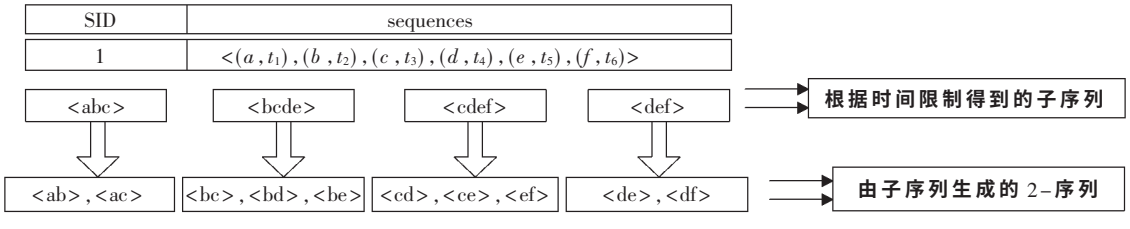

图 1 子序列及产生的 2-序列

序列划分为多个子序列；第二步是将得到的子序列转化为 2-序列；第三步是根据 uid、$fid(t)$和 $s(t)$的值将 2-序列进行结合，从而得到所有的候选项；第四步根据最小支持度 minsup 得到频繁项集。

从上所述可以看出，此算法的主要特点有：(1)加入了时间限制因素；(2)只需要扫描数据库一次，节省了算法的运行时间；(3)引入了 uid、$fid(t)$、$s(t)$，使之在形成 n-序列的过程中，能够准确地找到要结合的 2-序列，不需要扫描所有子序列产生的二序列，提高了算法的时间效率。

3.2 TFEGC 核心算法描述

2-序列结合的算法如下：

```
for all I ∈ DB{                    //I 为满足时间限制的子序列
    l=strlen(I);
    uid=l-2;
  for(n is smaller thanl ){
      if(in ∈ I and n≠1 and n≠1)
                                   { //in 不能为首尾两项
        s(in)=n-1;
        fid(in)=l-n;}
  }
for(scan the Matrix Chart of td){          //扫描矩阵 td
  if(td==s(in)){
    record the item r and
  record those 2-sequence se whose first item is r;}
                                  //r 用来存放首项，se 用来
                                //存放以其为首项形成的 2-序列
  }
for all tn ∈ se{
  if(tn.index==fid(in)){     //找到所要结合的 2-序列
    in+tn;}        //2-序列 in 和 2-序列 tn 相结合
  }
}
```

3.3 算法举例

为了进一步描述 TFEGC 算法，以图 1 中的子序列<abc>产生的第一个 2-序列<ab>为例，求以这个 2-序列为基础所形成的候选项过程，如图 3 所示。

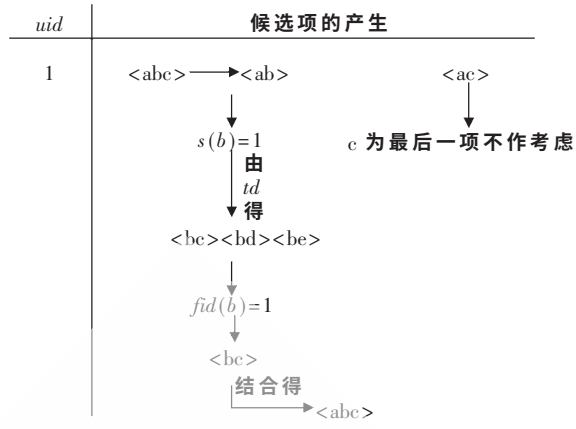

图 3 候选项的产生

4 实验结果

算法实验环境为 Pentium Ⅳ 3.0 GHz 双核 PC 机器，内存为 1 GB，Windows XP 操作系统，采用 C++语言在 VC 编程环境下实现。实验测试数据采用 IBM 数据生成器，该生成器是目前数据挖掘相关研究中使用广泛和权威的生成器，可以通过设置参数生成不同特性的数据集，具体参数含义如表 1 所示。

表 1 IBM 数据生成器命令具体含义

名字	命令	意思
D	-ncust	序列的数目
C	-slen	每个序列中包含的平均项集的个数
T	-tlen	每个项集中包含的平均项的数目
N	-nitems	不同项的个数
Ns	-seq.npats	序列模式的数目
S	-seq.patlen	最大序列的平均长度
	-seq.corr	序列间的相关度
	-seq.conf	序列的置信度

PrefixSpan 算法和 TFEGC 算法在不同的支持度下的运行时间如图 4 所示，从结果可以看出文中提出的方法优于 PrefixSpan 算法，因为 TFEGC 算法只需要扫描数据库一次，而且由于 uid、$fid(t)$以及 $s(t)$的引入，避免了一些无用序列的产生，大大减少了运行时间。

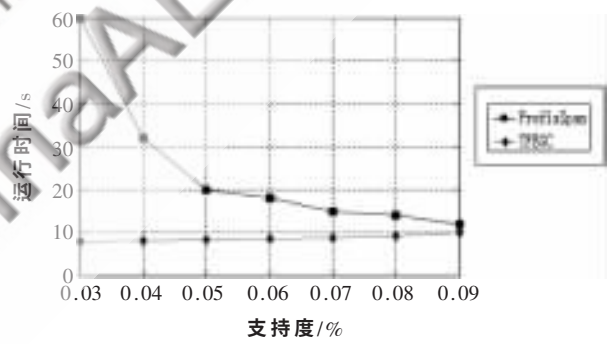

图 4 在不同支持度下的运行时间

图 5 所示的是 FEGC 和 TFEGC 在不同数量的顾客下算法的运行时间，从中也可以看出 TFEGC 算法的运

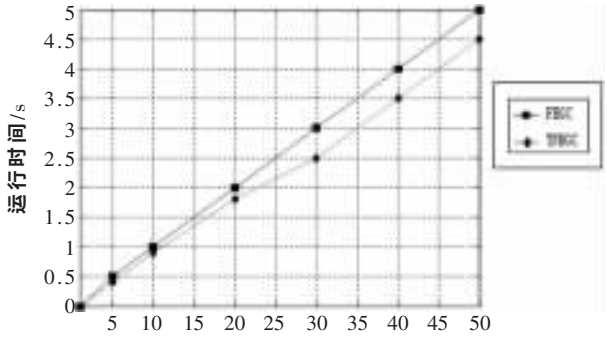

图 5 在不同数量客户下的运行时

行时间少于 FEGC 算法的运行时间。这是因为在二序列的结合过程中,TFEGC 不需要逐个扫描其他的二序列,只需查看 uid、$fid(t)$以及 $s(t)$的值就可以判断应该与哪些二序列进行结合,而且生成的用户不感兴趣的序列也大大减少了。因为加入了时间的限制,所以根据支持度来提取频繁序列的时间也就相应地减少了。

本文提出的算法,可由用户自己设定时间限制来查找频繁序列,也可以用来处理不同数量的客户群,所以具有一定的灵活性。又因为其只扫描数据库一次,而且二序列的结合也具有一定的明确性,所以在时间效率上具有一定的优越性[6]。

参考文献

[1] SRIKANT R, AGRAWAL R. Mining sequential patterns: generalizations and performance improvements[C]. Proc. of Int'l Conf. on Extending Database Technology, 1996: 3-17.

[2] PEI J, HAN J, Mortazavi-Asl B, et al. Prefixspan: mining sequential patterns efficiently by prefix-projected pattern growth[C]. Proc. of Int. Conf. on Data Eng. (ICDE'01), 2001: 215-224.

[3] Liao Weicheng, Yang Donlin, Wu Jungpin, et al. Fast and effective generation of candidate-sequences for sequential pattern mining[C]. 2009 Fifth International Joint Conference on INC, IMS and IDC, 2009: 2006-2009.

[4] 赵畅,杨冬青,唐世渭. Web 日志序列模式挖掘[J]. 计算机应用, 2000, 9(20): 13-16.

[5] 连一峰,戴英侠,王航. 基于模式挖掘的用户行为异常检测[J]. 计算机学报, 2002, 25(3): 325-329.

[6] 吉根林,帅克,孙志挥. 数据挖掘技术及其应用[J]. 南京师范大学学报(自然科学版), 2000, 23(2): 25-27.

(收稿日期:2010-07-16)

作者简介:

尹莉莉,女,1984年生,硕士研究生,主要研究方向:数据挖掘和数据库技术。

郑诚,男,1964年生,博士后,副教授,主要研究方向:数据挖掘、数据库技术与语义 Web。

郑小波,男,1983年生,硕士研究生,主要研究方向:语义 Web 和数据挖掘。