文章编号:1000-582x(2001)01-0092-03

序列模式的一种挖掘算法*

陈金玉,樊兴华,曹长修
(重庆大学 自动化学院,重庆 400044)

摘 要:序列模式挖掘是数据挖掘中最重要的研究课题之一。基于记录数据库频繁集中各元素 Ctid 表的基础上,提出了序列模式挖掘的一种算法 ISP。该算法考虑了项目集与序列之间的关系,利用时序连接法,采用不同的构造法,构造出相对应的候选集,从而计算出频繁集。由于算法 ISP 能够利用中间的挖掘结果,故提高了挖掘过程的效率。

关键词:序列模式;挖掘算法 ISP;频繁集;候选集;数据挖掘
中图分类号:TP 18 **文献标识码**:A

在数据库中发现知识(Knowledge discovery in databases,简称 KDD),亦称为数据挖掘(Data Mining),已是当今国际上人工智能和数据库研究方面最富活力的新兴领域。其目标是为了满足用户目标,自动处理大量的原始数据,从中识别出重要的和有意义的模式,并将其作为知识加以表达。由于其强大的应用潜力以及可广泛用于存在于各种数据库中的大量数据上,因此,KDD 成为一个具有迫切现实需要的热点研究课题。

序列模式(Sequential Pattern)是由 R. Agrawal 首先提出的。目前的绝大多数序列模式挖掘算法都采用一种宽度优先的搜索模式,如 AprioriAll[1], GSP[2]等。算法一般分为两个阶段:①频繁序列的发现;②规则的产生。算法的计算量主要集中在第一阶段上。笔者基于记录频繁集各元素 ctid 表的基础上,提出了序列模式挖掘的一种算法 ISP。算法 ISP 考虑了项目集与序列之间的关系,利用时序连接法,采用不同的构造法,构造出相对应的候选集,从而计算出频繁集。由于算法 ISP 能够利用中间的挖掘结果,故提高了挖掘过程的效率。本文与文[4]相比能开采出更多的有效规则。

1 定义及相关结果

为使本文能在概念上自包,现给出所有可能涉及到的定义[4]。它们将体现在文中例子的运算及算法 ISP 的推导上。

定义 1 非空集合 $I=\{i_1,i_2,\cdots,i_m\}$ 称为项集,其中 i_k 称为项。

定义 2 序列是项集的有序表,记为 $\alpha=\alpha_1\rightarrow\alpha_2\rightarrow\cdots\rightarrow\alpha_n$,其中 $\alpha_k\subset I(k=1,2,\cdots,n)$。含有 k 个项的序列的长度为 k,称为 k 序列($k=\sum|\alpha_i|$)。

定义 3 令序列 $\alpha=\alpha_1\rightarrow\alpha_2\rightarrow\cdots\rightarrow\alpha_n$,序列 $\beta=\beta_1\rightarrow\beta_2\rightarrow\cdots\rightarrow\beta_m$。若存在整数 $i_1<i_2<\cdots<i_n$,使得 $\alpha_1\subset\beta_{i1},\alpha_2\subset\beta_{i2},\cdots,\alpha_n\subset\beta_{in}$,则称序列 α 是序列 β 的子序列,或序列 β 包含序列 α。在一组序列中,如果某序列 α 不包含在其它任何序列中,则称 α 是该组中最长序列(Maximal sequence)。一组序列中可能有多个最长序列。

定义 4 函数 $c:I\rightarrow I^+$(正整数集),称为项集 I 的标识函数。函数 $t:I\rightarrow I^+$,称为项集 I 的时间函数,它表示项集 I 对应的时刻。事务 T 是由标识函数值及时间函数值标识的项集,标识函数 c 称为事务的标识号,记为 $c(T)$。时间函数值称为事务发生的时刻,记为 $t(T)$。数据以($c(T),t(T)$)表的形式存储,简称为 ctid 表。事务序列的集合称为事务数据库。

定义 5 给定序列 α,事务数据库 DB,若 D 为 DB 中所有事务标识号的集合模,即 $D=|\{c(T)|T\in DB\}|$,则 $\sigma(\alpha,DB)=|\{c(s)|$ 序列 s 包含 α 且 $s\in$

* 收稿日期:2000-06-29
基金项目:国家教育部博士生基金资助项目(98061117)
作者简介:陈金玉(1970-),男,福建省仙游县人,重庆大学自动化学院博士研究生,福建省泉州师范学院助教。主要研究方向为数据挖掘与企业互联网。

DB| / D，称为 α 在 DB 上的支持度。支持度大于最小支持度(min-sup) 的 k- 序列，称为 DB 上的频繁 k- 序列，记为 F_k。

定义 6　序列模式是形如 $\alpha \rightarrow \beta[\sigma, \delta]$ 规则的，其中 $\alpha \rightarrow \beta$ 为最长频繁序列，$\sigma(\alpha \rightarrow \beta, \mathrm{DB})$ 称为规则的支持度。$\delta = \sigma(\alpha \rightarrow \beta, \mathrm{DB}) / \sigma(\alpha, \mathrm{DB})$ 称为规则的置信度。置信度大于最小可信度(用户给定阈值，记作 min-conf) 的规则被认为是用户感兴趣的规则。序列模式挖掘就是从事务数据中找出满足用户指定的序列模式的过程。

为算法讨论方便，给出以下两个例子。如下图所示，(b) 中 DB 为时刻 20 时挖掘的数据库。(a) 中 DB' 为时刻 25 时挖掘的数据库。其中 A-H 为商品(条码数据)，T 为顾客交易事务，$c(T)$ 为顾客号，$t(T)$ 为事务号(图 1 中用 T, c, t 表示)。

c	t	T
1	15	$\{A\ B\ C\}$
1	20	$\{A\ B\ C\ E\ F\}$
1	25	$\{A\ C\ D\ F\}$
2	15	$\{A\ B\ F\}$
2	20	$\{C\ E\}$
3	15	$\{A\ B\ F\}$
3	20	$\{C\ E\}$
4	20	$\{B\ C\ E\ F\}$
4	25	$\{A\ C\ G\ H\}$

(a) DB′

c	t	T
1	15	$\{C\ D\}$
1	20	$\{A\ B\ C\}$
2	15	$\{D\}$
2	20	$\{A\ B\ F\}$
3	20	$\{A\ B\ F\}$
4	15	$\{D\ G\ H\}$

(b) DB

图 1　事务数据库

在讨论中，取 min-sup = 25%．为讨论的方便，只给出上图(b)DB 中的 F_k 中各元素及其 Ctid 表。相应的序列模式依定义 6 可简单得到。

$$
\begin{aligned}
F_1 = \{&(A)[(1,20)(2,20)(3,20)],\\
&(B)[(1,20)(2,20)(3,20)],\\
&(D)[(1,15)(2,15)(4,15)],\\
&(F)[(2,20)(3,20)]\}\\
F_2 = \{&(AB)[(1,20)(2,20)(3,20)],\\
&(AF)[(2,20)(3,20)],\\
&(BF)[(2,20)(3,20)],\\
&(D \rightarrow B)[(1,20)(2,20)],\\
&(D \rightarrow A)[(1,20)(2,20)]\}\\
F_3 = \{&(ABF)[(2,20)(3,20)],\\
&(D \rightarrow AB)[(1,20)(2,20)]\}\\
F_4 = &\Phi
\end{aligned}
$$

2　挖掘算法 ISP 的提出

算法 ISP 的基本框架和其它算法一样，可分为两个阶段：① 频繁序列的发现；② 规则的产生。不同的是，当记录下频繁集各元素的 Ctid 表后，已不需要对交易数据库进行多趟扫描了。而只要利用已得到的 Ctid 表以及利用时序连接法，即可得到所有的频繁序列集。现把频繁序列集 F_k 分为项集 L_k 与序列集 P_k 两个部分加以讨论。设所有候选 k 项目集与所有候选 k 序列集分别为 LC_k，PC_k。

仿照 Apriori-gen 函数[4]，提出一个新的候选 k 项目集生成函数 Ctid-gen(L_{k-1}) 来生成 LC_k。假设 L_{k-1} 项目集不空，其中 time-cons 函数检查两事务号是否满足时序连接条件，如考虑两项集 $(A)[(1,20)]$，$(B)[(1,20)]$，由于 $t_A = t_B = 20$，$c(A) = c(B) = 1$，则 $(AB) \in LC_k$。Ctid-gen(L_{k-1}) 函数分为两步：

① 拼接

Insert into LC_k

Select $p.\mathrm{item}_1, p.\mathrm{item}_2, \cdots, p.\mathrm{item}_{k-1}, q.\mathrm{item}_{k-1}$，并记录 (cid, t_b)；

From $L_{k-1}\ p, L_{k-1}\ q$ and for all tid $t_b \in q.\mathrm{ctid}$

Where $p.\mathrm{item}_1 = q.\mathrm{item}_1, \cdots, p.\mathrm{item}_{k-2} = q.\mathrm{item}_{k-2}, p.\mathrm{item}_{k-1} < q.\mathrm{item}_{k-1}$ and

If $\exists$ tid $t_a \in p.\mathrm{ctid}$ such that $t_a = t_a$ and time-cons(t_a, t_b) = true

② 修剪

For all itemsets $c \in LC_k$ do

for all $(k-1)$-subsets s of c do

if $(s \notin L_{k-1})$ then

delete c from LC_k

函数 Ctid-gen 与函数 Apriori-gen 最大的不同是，函数记录下每个序列的 Ctid 表，以便于计算其支持度。同时，通过扫描上一次的 Ctid 表，直接得到新的候选集，而不必重新扫描数据库，从而有效地节省了开采时间。同理，可以构造时序连接函数 Temp-join(L_j, L_{k-j})，来得到 k 序列候选集 PC_k。

① 拼接

for $(j = 1, j \leqslant k-1, j++)$

if $\alpha \in L_j, \beta \in L_{k-j}$ and $\alpha \cap \beta = \phi$ then

for all tid $t_b \in \beta.\mathrm{ctid}$ do

if $\exists$ tid $t_a \in \alpha.\mathrm{ctid}$ such that $t_b > t_a$ and time-cons(t_a, t_b) = true then

添加$(\alpha \rightarrow \beta)$，并记录$(\mathrm{cid}, t_b)$

② 修剪

for all sequence $c \in PC_k$ do

for all $(k-1)$-subsequence s of c do

if $(s \notin P_{k-1})$ then

delete c from PC_k

现在来说明候选 k 项目与序列集的生成函数

Ctid-gen, Temp-gen 的正确性,也就是 $PC_k \supseteq P_k$, $LC_k \supseteq L_k$ 成立。正如前面所定义的,P_k 中的每一频繁 k 序列集都是由两个不相交的非空频繁项集 α 与 β 组成,而 Temp-gen 中的拼接步就相当于将每一个频繁集 L_j 中的项分别与 L_{k-j} 中的项进行并操作。通过 j 的变化,考虑了将频繁序列集划分为两个非空子集的所有组合,修剪步只是删除那些根本不可能出现在 P_k 中的序列集。因此有 $PC_k \supseteq P_k$。同理 $LC_k \supseteq L_k$ 也是成立的。

C_k 生成之后,紧接着就扫描 Ctid 表,最终生成 F_k。上述过程一直重复到不再有新的频繁集生成为止。在下面的算法描述中,只简单地以判断是否为空来决定 ISP 算法是否还需进入 $(k+1)$ 趟。ISP 算法的基本框架描述如下所示:

1) F_1 = {Large 1-itemsets}, 并记录 Ctid 表;
2) For ($k = 2, F_{k-1} \neq \Phi, k++$) do begin {
3) LC_k = Ctid-gen(L_{k-1});
4) PC_k = Temp-join(L_j, L_{k-j});
5) $C_k = LC_k \cup PC_k$;
6) For each $t \in C_k$ do begin
7) t.sup = |{$c(t) \mid t \in C_k$}|; /* 利用 Ctid 表计算各元素的支持度 */
8) End
9) $L_k = \{c \in LC_k \mid c.\text{sup} > \text{min-sup} \times D\}$;
10) $P_k = \{c \in PC_k \mid c.\text{sup} > \text{min-sup} \times D\}$;
11) $F_k = L_k \cup P_k$;}
12) End
13) Answer: = Maximal Sequence in $\cup_k F_k$

图 2 ISP 算法

以图 1 中数据库 DB′ 为例,在最小支持度为 25% 的条件下,利用本算法,可挖掘出较文[4]更多的频繁项目。如多得到序列集 $(AB \rightarrow CE)[(1.20), (2.20), (3.20)]$, $(BE \rightarrow AC)[(1.25), (4.25)]$, $(BF \rightarrow AC)[(1.25), (4\ 25)]$, $(EF \rightarrow AC)[(2\ 20), (3.20)]$, $(AF \rightarrow CE)[(2.20), (3.20)]$, $(BF \rightarrow CE)[(2.20), (3.20)]$, $(BEF \rightarrow AC)[(1.25), (4.25)]$ 与 $(ABF \rightarrow CE)[(2.20), (3.20)]$。于是由定义 6 即可得到更多的有效规则,从而改进了文[4]的结果。

3 结束语

基于记录数据库频繁集各元素的 Ctid 表的基础上,笔者提出了序列模式挖掘的一种算法 ISP。算法 ISP 由于能够利用中间的挖掘结果,故提高了挖掘过程的效率,且其与文[4]相比能开采出更多的有效规则。当数据库或支持度发生变化时数据库模式的维护问题,将是下一步的研究工作之一。

参考文献:

[1] AGRAWAL R, SRIKANT R. Mining sequential patterns [A]. Proc International Conference on Data Engineering [C]. Taipei Taiwan, 1995. 3-14

[2] SRIKANT R, AGRAWAL R. Mining sequential patterns: generalizations and performance Improvements [A] Proc International Conference on Extending Database Technology [C]. Avignon France, 1996. 3-17.

[3] AGRAWAL R, SRIKANT R. Fast algorithms for mining association rules[A] proceedings of the 20th International conference on Very Large Database [C] Santiago, Chile. 1994. 487-499.

[4] 周斌,吴泉源 序列模式挖掘的一种渐进算法[J]. 计算机学报,1999,22(8):882-887.

Algorithm for Mining Sequential Pattern

CHEN Jin-yu, FAN Xing-hua, CAO Chang-xiu

(College of Automation, Chongqing University, Chongqing 400044, China)

Abstract: Mining sequential pattern is an important topic in the data mining research. In this paper, on the basis of recording the Ctid scheme of the set in every frequent set, the authors propose an algorithm named ISP for mining sequential pattern. In the algorithm the items and the sequence are discussed respectively, and the time-join method is used to introduce the candidate sets, so the frequent sets can be gotten. The ISP algorithm takes full use of the existing and updated Ctid scheme, therefore the efficiency of the process is increased besides guaranteeing the validity of the algorithm. Comparing with the algorithm named IMSP, more efficient rules are obtained.

Key words: sequential pattern; mining algorithms ISP; frequent sets; candidate sets; data mining

(责任编辑 吕赛英)