在数据库中自动发现广义序贯模式*

*#欧阳为民 #蔡庆生

*(安徽大学计算中心 合肥 230039)

#(中国科学技术大学计算机系 合肥 230027)

摘要 本文将序贯模式的发现从单层(Single Level)概念扩展到多层(Multiple Level)概念,即既允许在同层概念之间,也允许在不同层概念之间发现序贯模式,提出了发现广义序贯模式的自顶向下逐层递进的方法.

关键词 知识发现,广义序贯模式,概念层次.

中图法分类号 TP391

在数据库中发现知识 KDD(knowledge discovery in databases),亦称为数据发掘(Data Mining),是当今国际人工智能和数据库研究中十分活跃的新兴领域.[1]在信息爆炸的今天,各种数据库中的数据迅速增长,没有计算机和强大的分析工具,要想对这些数据进行分析,并从中获得有意义的模式几乎是不可能的. KDD 的主要目标是为了满足用户目标,自动处理大量的原始数据,识别重要和有意义的模式,并将其作为知识加以表达.[2]

序贯模式(Sequential Pattern)是 R. Agrawal[3]首先提出的. 设有一个交易数据库 D,每个顾客可在不同时间购买不同物品,每次购买活动称为交易(Transaction). 这里,顾客、交易时间和所购物品分别以 Customer-ID, Transaction-Time 和 Itemset 标识. 如果我们以 Customer-ID 为第 1 关键字,Transaction-Time 为第 2 关键字对数据库 D 排序,那么,对每位顾客而言,他进行的所有交易是以交易时间的升序排列的,从而构成了一个序列. 我们称这种序列为顾客序列 CS(customer-sequence). 一般地,令某顾客的各次交易时间为 $t_1, t_2, t_3, \ldots, t_n$,该顾客在交易时间 t_i 购买的物品集记为 $itemset(t_i)$. 于是,该顾客的 CS 序列为 $itemset(t_1), itemset(t_2), itemset(t_3), \ldots, itemset(t_n)$. 相应地,我们也可以认为上述交易数据库 D 已转换为顾客序列数据库.

如果某序列 s 包含在某顾客的 CS 序列中,那么我们称该顾客支持(Support)该序列 s. 某序列的支持度为支持该序列的顾客数与顾客序列数据库中顾客总数之比. 因为数据库中的顾客总数是一定的,所以,为方便计,我们一般只考虑支持某序列的顾客数,以此来代表序

* 本文研究得到国家自然科学基金和国家教委博士点基金资助. 作者欧阳为民,1964 年生,副教授,在职博士生,主要研究领域为 KDD,机器学习,人工智能及其应用. 蔡庆生,1938 年生,教授,博士导师,主要研究领域为机器学习,知识发现,人工智能.

本文通讯联系人:欧阳为民,合肥 230039,安徽大学计算中心 E-mail:oywm@mars.ahu.edu.cn

本文 1997-06-25 收到修改稿

列的支持度. 序贯模式(Sequential Pattern)就是在上述顾客序列数据库中满足用户指定最低支持的最长的序列.[3]每个这样的最长序列代表一个序贯模式(Sequential Pattern).

例如,我们假设有一个如表 1 所示的某交易数据库相应的顾客序列数据库.

表 1

Customer-ID	*Customer-Sequence*
1	(*D*) (*I*)
2	(*A*) (*D H*)
3	(*C E*) (*F G*)
4	(*D E*) (*F I G*)
5	(*B*) (*E H*)
6	(*K*) (*E*) (*G*)
7	(*L M*)

假定最低支持为 2. 按序贯模式的定义可知,序列(*E F G*)和(*D I*)为不低于最低支持的最长的序列,即为序贯模式. 序列(*E F G*)得到顾客 3 和顾客 4 的支持. 顾客 4 在购买 *F* 和 *G* 之间,还购买了 *I*,但仍然支持序列(*E F G*),因为我们并不要求模式必须是连续的,只要满足时间上的先后偏序关系即可. 顾客 3 支持序列(*E F G*)是显然的. 序列(*D I*)的支持为 2. 尽管序列(*D*),(*E*),(*F*),(*G*),(*E F*),(*E G*)和(*F G*)均不低于最低支持,但它们不是最长的,因为它们要么包含在序列(*E F G*)中,要么包含在序列(*D I*)中,从而不是序贯模式.

显然,上述序贯模式是建立在原始概念级上的. 然而,在大多数情况下,物品之间的概念层次关系是可以获得的. 图 1 所示的就是一种概念层次关系. 第 0 层为食品总类,第 1 层为食品类别(Category),如 *milk* 和 *bread* 等;第 2 层为食品含量成分(Content),如含 *sugar*,...,*chocolate* 等;第 3 层为品牌名(Brand),如 *A*,*B*,*C* 等等.

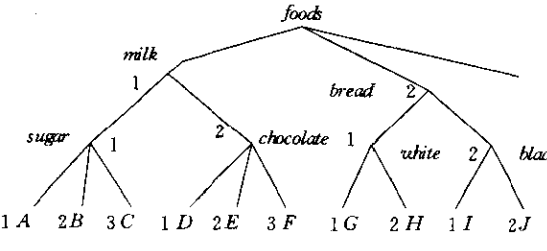

图1

按照图 1 所示的编码方式对表 1 所示的顾客序列表中的物品进行编码. 例如,品牌为 *B* 的加糖牛奶的编码为 112,其第 1 位数字 1 代表第 1 层概念,即 *milk*,第 2 位数字 1 代表第 2 层概念,即 *sugar*,第 3 位数字 2 代表第 3 层概念,即品牌 *B*. 而编码 11 * 则代表各种加糖牛奶,其中'*'为通配符,在这里代表各种品牌. 注意,在任何概念层,编码相同的物品视为同一物品,我们不要求序贯模式在顾客序列中连续.

假定我们在第 2 概念级上考虑问题,同时假定该层最低支持为 3. 这样,*A*,*B*,*C* 的编码均为 11 *,因而视为同一物品. 同样,*D*,*E*,*F* 和 *G*,*H* 以及 *I*,*J* 分别视为相同的物品. 于是,(*A H*),(*C G*)和(*B H*)的编码表示均为(11 * 21 *). 显然,序列(11 * 21 *)满足该层最低支持为 3,且是最长的,因而是该层的序贯模式. 其含义为购买加糖牛奶(11 *)后通常还会购买白面包(21 *).

如果我们不要求所考察的对象必须是同一概念层的,即允许在不同概念层的对象之间发现序贯模式. 例如,"购买 *D* 品牌的巧克力牛奶(121)后常常会购买黑面包(22 *)"就是不

同概念层的模式.

目前,国际学术界关于序贯模式的研究均未考虑概念的层次问题,或者说是均限于单一的概念层次[1,3],尚未研究多层次概念的序贯模式.然而,在不同概念层次发现序贯模式是十分有价值的:(1)某个序列在较低概念层不是序贯模式,并不意味着在较高概念层也不是序贯模式.上面的例子已清楚地表明了这一点.这就是说,如果将发现限制在最低概念层,即原始概念层上,就可能会遗漏一些有价值的信息.(2)不同的概念层提供了不同的抽象级别.这样在不同概念层发现的序贯模式就在不同的抽象级别表达了不同的解释.(3)概念层次关系还可以用来修剪不令人感兴趣的或冗余的序贯模式.本文的工作就是研究如何在大型数据库中发现广义序贯模式.其基本思想是利用概念层次关系,自顶向下逐层递进地在不同概念层发现相应的序贯模式.

本文第1节提出与广义序贯模式有关的概念;第2节提出在大型数据库中发现多层序贯模式的基本算法;第3节总结并指出进一步的工作.

1 广义序贯模式的有关概念

为研究如何在大型数据库中发现广义序贯模式,假设我们有

(1) 交易数据库 D:($Customer\_ID$, $Customer\text{-}Time$, $Itemset$),其中 $Customer\_ID$, $Customer\text{-}Time$ 和 $Itemset$ 分别为顾客标识、交易时间和物品集. $Itemset=(A_p,\ldots,A_q)$, $A_i\in T$ $(i=p,\ldots,q)$,其中 T 为所有物品的集合.

(2) 物品描述集,其中包括物品集 T 中每个物品的描述,形式为(A_i, $description_i$), $A_i\in T$.

定义 1.1. 某顾客的顾客序列(Customer Sequence)为序列 $itemset(t_1)$, $itemset(t_2)$, $itemset(t_3)$,...,$itemset(t_n)$,其中 $t_i(i=1,2,\ldots,n)$为该顾客的交易时间,且 $t_1<t_2<t_3<\ldots<t_n$, $itemset(T_i)$为该顾客在时间 t_i 所购物品的集合.

定义 1.2. 序列的长度为该序列中所含有的物品(Item)的个数.

定义 1.3. 长度为 K 的序列称为 K-序列.

定义 1.4. 某顾客支持序列 s,如果 s 包含在该顾客的顾客序列中.

定义 1.5. 某序列在某概念层的支持为交易数据库中在该层包含该序列的顾客数.

定义 1.6. 在某概念层次,满足该层最低支持的序列为该层的常见序列(Frequent Sequence).

引理 1. 若某 K-序列为常见序列,则其任一长度为(K-1)的子序列必是常见序列.

证明:设有一 K-序列 s 为常见序列.如果存在 s 的一个长度为(K-1)的子序列 $x=(I_1, I_2,\ldots,I_{k-1})$不是常见序列,即序列 x 的出现次数低于最低支持.这样,不管在序列 x 中的什么位置增加一个元素,所构成的 K-序列在交易数据库中的出现次数必然低于最低支持,从而 K-序列 s 在交易数据库中的出现次数必然低于最低支持,即不是常见序列.于是,引理1得证. □

定义 1.7. 在某概念层次,最长的常见序列称为该层的广义序贯模式(Generalized Sequential Pattern).

2 广义序贯模式的发现算法

本节提出的广义序贯模式的发现算法所使用的交易数据库不是普通的原始交易数据库，而是包含了描述库中物品的某种概念层次关系的交易数据库. 这是基于下述考虑. 第 1，只有具备物品的概念层次关系，才有可能在多层次上发现序贯模式. 第 2，数据发掘(Data Mining)常常仅与交易数据库的某一部分有关，比如说食品类 *food*，而不必考察全部物品. 这有利于收集相关数据集，继而在此与任务相关的数据集上工作. 第 3，物品编码工作可在收集与任务相关的数据集的同时进行，从而不必再次遍历数据库. 第 4，表达物品在概念层次关系中位置的代码串可以较短，而且，在每个概念层，编码相同的物品即行合并，从而视为同一物品. 这又进一步地削减了编码交易数据库的规模. 下面，我们首先给出一个例子来说明广义序贯模式的发现方法，然后再提出相应的算法描述.

例. 设表 1 所示的顾客序列表为我们的相关数据集，图 1 为其中物品的概念层次关系. 按照图 1 所示的编码方式对表 1 所示的顾客序列表中的物品进行编码，结果为表 $T[1]$，如表 2 所示. 例如，品牌为 B 的加糖牛奶的编码为 112，其第 1 位数字 1 代表第 1 层概念，即 *milk*，第 2 位数字 1 代表第 2 层概念，即 *sugar*，第 3 位数字 2 代表第 3 层概念，即品牌 B. 而编码 11 * 代表各种加糖牛奶，其中'*'为通配符，在这里代表各种品牌. 编码 1 * * 代表各种牛奶，其中'*'为通配符，但第 1 个'*'代表各种含量成分，第 2 个'*'仍代表各种品牌. 我们称其编码中含有'*'的为广义物品(Genaralized Item)，而物品 111 为物品 11 * 后代，11 * 又是 1 * * 的后代. 自然，物品 11 * 为物品 111 的前辈，1 * * 为 11 * 的前辈. 注意：(1)我们不考虑象 1 * 1 这种中间夹有'*'编码的物品；(2)任何概念层，编码相同的物品视为同一物品，我们不要求序贯模式在顾客序列中是连续的，只要能保持时间上的偏序关系即可；(3)假定不同概念层具有不同的最低支持，记为 min$sup[L]$，其中 L 代表层次.

表 2 $T[1]$

Customer-ID	*Customer-Sequence*
1	(121) (221)
2	(111) (121 212)
3	(113 122) (123 211)
4	(121 122) (123 221 211)
5	(112) (122 212)
6	(311) (122) (211)
7	(312 322)

表 3 $T[2]$

Customer-ID	*Customer-Sequence*
1	(121) (221)
2	(111) (121 212)
3	(113 122) (123 211)
4	(121 122) (123 221 211)
5	(112) (122 212)
6	(122) (211)

第 1 概念层的序贯模式推导如下. 设该层最低支持为 4，即 min$sup[1]=4$. 扫描表 $T[1]$，对其中所有 1-序列，即 1 * *，2 * * 和 3 * *，分别计算各自的支持，选取支持不低于该层最低支持的常见 1-序列 $Fre[1,1]$. 接着，利用 $Fre[1,1]$ 过滤表 $T[1]$，即根据引理 1，删除 $T[1]$ 中不与 $Fre[1,1]$ 中任何 1-序列匹配的 1-序列. 如果某顾客序列不包含任何 $Fre[1,1]$ 中 1-序列，就删除该顾客序列. 由此得表 $T[2]$，如表 3 所示. 以后就用 $T[2]$ 来代替 $T[1]$，以便降低库扫描过程中无关检查的次数. 根据常见 1-序列 $Fre[1,1]$ 推导常见 2-序列 $Fre[1,2]$. 首先从常见 1-序列 $Fre[1,1]$ 派生出候选 2-序列 $C[1,2]$，并扫描表 $T[2]$，计算各候选 2-

序列的支持. 然后,从 $C[1,2]$ 删除支持低于 4 的候选,得到第 1 层的常见 2-序列 $Fre[1,2]$. 各结果如图 2 所示. 因为 $Fre[1,2]$ 中只有 1 个 2-序列,无法进一步组合,所以不可能有更长的常见序列. 于是,第 1 级的常见序列生成就此结束.

sequence	support
1**	6

Fre[1,1]

sequence	support
1** 2**	6
2** 1**	0

C[1,2]

sequence	support
1** 2**	6

Fre[1,2]

sequence	support
11*	3
12*	6
21*	5

Fre[2,1]

sequence	support	sequence	support
11* 12*	3	11* 2**	3
11* 21*	3	12* 2**	6
12* 11*	0	21* 1**	0
12* 21*	5	1** 21*	5
21* 11*	0	2** 11*	0
21* 12*	0	2** 1**	0

C[2,2]

sequence	support
11* 12*	3
11* 21*	3
12* 21*	5
11* 2**	3
12* 2**	6
1** 21*	5

Fre[2,2]

sequence	support
11* 12* 21*	3
11* 12* 2**	3

Fre[2,3]

sequence	support
121	3
122	4
211	3

Fre[3,1]

sequence	support
122 211	3
122 21*	4
12* 211	3
1** 211	3
112 2**	4
121 2**	3

sequence	support	sequence	support
11* 12* 21*	3	12* 2** 21*	1
11* 12* 2**	3	12* 1** 21*	1
11* 2** 12*	0	11* 12* 2**	2
11* 21* 2**	0	12* 11* 2**	0
11* 2** 21*	1		
12* 21* 2*	0		

C[2,3]

图2

假设第 2 概念级的最低支持为 3,即 minsup[2]=3. 扫描表 $T[2]$,对所有第 2 概念级的 1-序列,计算各自的支持,选取支持不低于 3 的 1-序列,从而得第 2 级的常见 1-序列 $Fre[2,1]$. 第 2 级的候选常见 2-序列 $C[2,2]$不仅由 $Fre[2,1]$与其自身组合生成,而且还可与 $Fre[1,1]$共同生成. 但是,应该注意的是,由于象(111 11 *)和(11 * 111)这种相互之间具有后代或前辈关系的序列没有什么意义(这不同于关联规则,111→11 * 没什么意义,但 11 * →111却是有意义的),我们在构造候选时就不加考虑了. 扫描表 $T[2]$,计算各候选的支持,选取支持不低于 3 的 2-序列,从而得第 2 级的常见 2-序列 $Fre[2,2]$,如图 2 所示.

第 2 级的常见 3-序列 $Fre[2,3]$可这样推出:由 $Fre[2,2]$生成候选常见 3-序列 $C[2,3]$,再通过扫描 $T[2]$计算各个候选的支持,从 $C[2,3]$中选取那些支持不低于 minsup[2]的候选常见 3-序列构成第 2 级的常见 3-序列 $Fre[2,3]$,如图 2 所示.

由于 $Fre[2,3]$仅有 1 个序列,不可能产生更长的序列,所以第 2 概念层的常见序列推导过程结束,开始第 3 概念层的常见序列推导. 设第 3 概念层的最低支持为 3,即 minsup[3]=3. 该层各常见序列推导方法与前类似. 即扫描表 $T[2]$,对所有第 3 概念级的 1-序列,计算各自的支持,选取支持不低于 minsup[3]的 1-序列,从而得第 3 级的常见 1-序列 $Fre[3,1]$. 但要注意,在生成第 3 概念层的候选常见 2-序列 $C[3,2]$时,$Fre[3,1]$不仅要自身构造候选,还要与其第 1 层和第 2 层的常见 1-序列,即 $Fre[2,1]$和 $Fre[1,1]$进行链接,以构造候选. 至

于第 3 层的其它常见 K-序列($K>2$)则均根据 $Fre[3,K\text{-}1]$生成.

至此，已没有更深的概念层次待处理，于是结束整个算法. 每一层 $Fre[L,K]$的终止条件为 $Fre[L,K\text{-}1]$中的只有 1 个序列. 基于上述描述，我们有如下 MLSP(multiple level sequential pattern)算法.

Algorithm Discovery of Generalized Sequential Pattern

Input: (1) A transaction database D, a hierarchy information encoded and task-relevant set of transaction, in the format of (Customer-ID, Transaction-Time, Itemset), in which each Item in the Itemset contains encoded concept hierachy information, and

(2) the mininum support threshod (min$sup[L]$) for each concept level L.

Ouput: Generalized Sequential Pattern

Begin

(1) Sort transaction databases D with Customer-ID as major key and Transaction-Time as minor key. This step result in $T[1]$;
(2) for (L=1; L<max_*level*; L++) {
(3)　　if (L= = 1) {
(4)　　$Fre[1,1]$=get_frequent_sequence($T[1]$,L);
(5)　　$T[2]$=get_filtered_table($T[1]$,$Fre[1,1]$);
(6)　　}
(7)　　else $Fre[L,1]$=get_frequent_sequence($T[2]$,L);
(8)　　for (K=2; *number_of* ($Fre[L,K\text{-}1]$) >1 ; K++) {
(9)　　if (K= =2) $C[L,2]$=candidates_gen($Fre[L,K-1]$, $L-1$);
(10)　　else $C[L,K]$=get_candidate_set($Fre[L,K-1]$);
(11)　　for each candidate $c \in C[L,K]$ do
(12)　　for each Customer_ID do
(13)　　if ($c \in$ *Itemset*) c.*support*++;
(14)　　$Fre[L,K]=\{c|c(C[L,K],c.support($ min$sup[L]\}$;
(15)　　}
(16)　　$FP[L]$=Maximal sequences in (k $Fre[L,K]$;
(17) }

End

说明：(1) 算法的第 1 步是将交易数据库 D 转化为顾客序列数据库，即表 $T[1]$.

(2) 在第 1 概念层，常见 1-序列 $Fre[1,1]$是由函数 get_frequent_sequence 直接处理 $T[1]$得到的；而在其它层，$Fre[L,1]$ ($L>1$) 则是处理 $T[2]$得到的.

(3) 表 $T[2]$由函数 get_filtered_table 根据 $Fre[1,1]$对 $T[1]$处理而得，即以 $Fre[1,1]$为过滤条件，从 $T[1]$中去除：(a) 不包含在 $Fre[1,1]$中的序列；(b) 不含 $Fre[1,1]$中任何序列的顾客序列.

(4) 在第 L 概念层，常见 2-序列分如下两步得到：(a) 由函数 candidates_gen 生成候选常见 2-序列 $C[L,2]$，方法是由 $Fre[L,1]$再由 $Fre[L,1]$与 $Fre[i,1](i=1,2,\ldots,L)$进行链接生成候选常见 2-序列. (b) 对 $C[L,2]$中的每个候选序列 c，扫描 $T[2]$中的每一个顾客序列 CS，如果某个 CS 包含候选序列 c，那么该候选序列 c 的支持增 1. 这样计算出所有候选序列的支持后，选取其中支持不低于 min$sup[L]$的候选序列，从而得 $Fre[L,2]$.

(5) 在第 L 概念层，常见 k-序列($k>2$)分如下两步得到：(a) 仿照文献[4]中的 aprior_gen 候选生成算法，根据 $Fre[L,K\text{-}1]$生成候选 K-序列 $C[L,K]$，即连接 $Fre[L,K\text{-}1]$中任何两个含有相同(K-2)-序列的(K-1)-序列. 然后，再删除 $C[L,K]$中那些含有不在 $Fre[L,K\text{-}1]$中的(K-1)-序列的候选. (b) 对 $C[L,K]$中的每个候选序列 c，扫描 $T[2]$中的每一个顾客序列 CS，如果某个 CS 包含候选序列 c，那么该候选序列 c 的支持增 1. 这样计算出所有候

选序列的支持后，选取其中支持不低于 min$sup[L]$的候选序列，从而得 $Fre[L,K]$.

(6) 每一概念层的序贯模式为该层最长的常见序列，记为 $FP[L]$.

3 结论与进一步工作

本文将序贯模式的发现从单层(Single Level)概念扩展到多层(Multiple Level)概念，即既允许在同层概念之间，也允许在不同层概念之间发现序贯模式，提出了自顶向下逐层递进的方法发现广义序贯模式.

本文所述工作的不足之处在于我们未考虑噪音数据和不确定性的处理问题. 为此，我们的下一步研究工作将是处理知识发现过程中噪音数据和不确定性.

致谢 本文工作是在国际 KDD 研究权威加拿大 Simon Fraster 大学 Jiawei Han 教授的大力帮助下进行的. Han 教授不仅指出了研究方向，而且提供了相关的资料. 特此深表感谢.

参考文献

1 Agrawal R, Srikant R. Mining sequential patterns. In the Proc. 1995 Int. Conf. Data Engineering, Taibei, Taiwan, March 1995.

2 Piatetsky-Shapiro G. Discovery, analysis, and presentation of strong rules. In: Piatetsky-Shapiro G, Frawley W editors, Knowledge Discovery in Databases, AAAI/MIT Press, Menlo Park, CA, 1991. 229～248.

3 Han J, Fu Y. Discovery of multiple-level association rules from large databases. In the Proc. 21th VLDB Conf. Zurich, Swizerland, 1995.

4 Agrawal R, Srikant R. Fast algorithm for mining association rules. In the Proc. 1994 Int. Conf. Very Large Data Bases, Santiago, Chile, Sept., 1994. 487～499.

AUTOMATIC DISCOVERY OF GENERALIZED SEQUENTIAL PATTERN IN DATABASES

*#OU-YANG Weimin #CAI Qingsheng

*(*The Computing Center Anhui University Hefei 230039*)

#(*Department of Computer Science University of Science and Technology of China Hefei 230027*)

Abstract In this paper, the scope of discovery of sequential pattern has been extended from single level to multiple level, that is, discover sequential pattern among the objects in the same concept level or in different level. And a top-down and progressively deepening method for discovering sequential pattern in each different level has been put forward.

Key words Knowledge discovery, generalized sequential pattern, concept hierarchy.

Class number TP391