序列模式挖掘研究①

吴孔玲，缪裕青，苏 杰，张晓华

(桂林电子科技大学 计算机科学与工程学院，桂林 541004)

摘 要：为了更好地分析购物篮数据，挖掘出潜在客户，序列模式挖掘应运而生。序列模式挖掘是数据挖掘一个重要研究内容，近年来在很多领域得到广泛运用。概述序列模式挖掘的发展现状，研究基本挖掘框架的经典挖掘算法与扩展模型挖掘算法，特别针对近年来出现的新数据形式序列模式挖掘，以及基于零压缩二叉决策图(ZBDD)结构的挖掘算法做了阐述，最后对序列模式挖掘发展趋势进行了展望。

关键词：序列模式挖掘；模式增长；投影数据库；零压缩二叉决策图

Sequential Pattern Mining Research

WU Kong-Ling, MIAO Yu-Qing, SU Jie, ZHANG Xiao-Hua

(School of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: In order to analysis shopping basket data better, mine potential customers, sequential pattern minging emerged.Sequential pattern mining is an important research content of data mining and has been widely used in many fields in recent years. We summary sequential pattern mining development situation at present, research classical algorithm based on basic mining framework and algorithm based on extension model, Especially describe the new data form sequential pattern mining appeared in recent years and the algorithm based on zero compression binary decision figure (ZBDD), Finally prospect the sequential pattern mining development trend.

Key words: sequential pattern mining; pattern-growth; projected-database; ZBDD

序列模式挖掘最早由 R. Agrawal 和 R. Srikant 在 1995 年提出[1]，是数据挖掘一个重要的研究领域。随着信息技术的发展，新数据形式不断产生，存储的数据量越来越大，序列模式挖掘的重要性和必要性也更加呈现在人们面前。例如通过对客户购买行为进行分析可以发现顾客购买规律、对 WEB 日志进行分析可以发现访问规律、对 DNA 进行分析可以发现疾病机理、对医学数据进行分析可以进行疾病诊断等，这些发现可以为商业决策提供支持，为疾病诊治提供依据，从而获得良好的经济效益与社会效益。

1 基本术语与问题描述

设 $I=\{i_1,i_2,\ldots,i_n\}$是序列数据库中所有出现过的单项（Item）集合。项集（Itemset）是单项组成的非空集合，是 I 的子集，表示为 $X=\{x_1,x_2,\ldots,x_m\}$，其中 x_l（$1<=l<=m$）是单项。

序列（Sequence）是项集的有序表，记作 $S=<s_1,s_2,\ldots,s_n>$，其中 s_k ($1<=k<=n$)是非空项集，称为序列的一个元素（Element），元素内的单项是无序的，默认按字典序排序。

序列的长度一般有两种定义，一种定义为序列中包含的单项总个数，另一种定义为序列中包含的元素总个数，长度为 k 的序列称为 k-序列。

给定两个序列 $A=<a_1,a_2,\ldots,a_m>$和 $B=<b_1,b_2,\ldots,b_n>$，如果存在一组整数 $j_1<j_2<\ldots j_m$，使得 a_1 包含于 b_{j1},a_2 包含于 $b_{j2}\ldots$，a_m 包含于 b_{jm}，则称序列 A 包含于序列 B，A 是 B 的子序列，记为 $A\subseteq B$。

序列数据库 D 是元组(Sid,S)的集合，Sid 是序列号，

① 收稿时间:2011-09-14;收到修改稿时间:2011-11-16

S 是序列。如果序列 T 是 S 的子序列（即 $T \subseteq S$），称元组(Sid,S)包含序列 T。序列 T 在序列数据库 D 中的支持度是数据库中包含 T 的元组个数，即 $supportD(T) = \{(Sid,S) | (Sid,S) \in D \wedge T \subseteq S\}$,记作 $support(T)$。给定支持度阈值 ξ，如果序列 T 在序列数据库中的支持度不低于 ξ，则称序列 T 为序列模式，长度为 l 的序列模式记为 l-模式。

序列模式挖掘就是要找出序列数据库中所有的序列模式，即支持度不小于用户给定的支持度阈值的所有序列。

2　经典序列模式挖掘算法

经典序列模式挖掘算法针对传统交易数据库，主要有两种基本挖掘框架。一种是候选码生成—测试框架，它基于 Apriori 理论，即序列模式的任一子序列也是序列模式。通过多次扫描数据库，根据较短的序列模式生成较长的候选序列模式，然后计算候选序列模式的支持度，从而获得所有序列模式。另一种是模式增长框架，基于分而治之的思想，迭代的将原始数据库进行划分，同时在划分的过程中动态的挖掘序列模式，并将新发现的序列模式作为新的划分元，进行下一次的挖掘过程，从而获得长度不断增长的序列模式。

2.1 候选码生成—测试框架挖掘算法

R. Agrawal 和 R. Srikant 在提出序列模式挖掘的同时，给出了三个候选码生成—测试框架挖掘算法 AprioriAll[1]、AprioriSome[1]、DynamicSome[1]。它们基于 R. Agrawal 和 R. Srikant 早前挖掘关联规则时提出的 Apriori 特性，可以称为类 Apriori 算法。随后研究人员提出了一系列基于 Apriori 性质算法，主要有 R. Agrawal 和 R. Srikant 提出的 GSP[2]，F. Masseglia 等提出的 PSP[3]，M. Zhang、B. Kao 等提出的 MFS[4]，Zaki 提出的 SPADE[5]，J. Ayres、J. Flannick 等提出的 SPAM[6]，Z. Yang、Y. Wang、M. Kitsuregawa 提出的 LAPIN-SPAM 等。它们的演化关系如图 1 所示。根据数据的不同分布方式，又可以将算法分为水平格式算法和垂直格式算法。垂直分布是数据集由一系列项集和序列标志符组成，如表 1 所示。数据的水平分布是数据集由一系列序列标识符和序列组成，如表 2 所示。

(1) 水平格式挖掘算法

类 Apriori 算法将挖掘过程分为五个阶段：排序阶段、大项目集阶段、转换阶段、序列阶段、最大序列阶段。基本思想是：在每一次遍历数据库时，利用前一次遍历所获得的序列模式产生长度增一的候选序列，在完成遍历时测试它们的支持度，从而获得长度加一的序列模式。其中 AprioriSome[1]、DynamicSome[1]只求最大序列模式。AprioriSome[1]将挖掘过程分为前半部分与后半部分，前半部分通过函数确定需要计数的序列长度，后半部分先删除被包含的非最大序列模式，然后跳过已计数序列。DynamicSome 与 Apriori Some 类似，由函数确定要计数的序列长度，分为四个部分，但与 AprioriAll、AprioriSome 不同，它的剪枝在后半部分，并不像 AprioriAll、AprioriSome 那样计数后就进行剪枝，因此在支持度较小时，会产生大量的候选序列。一般情况下，AprioriSome 优于其它两种算法。

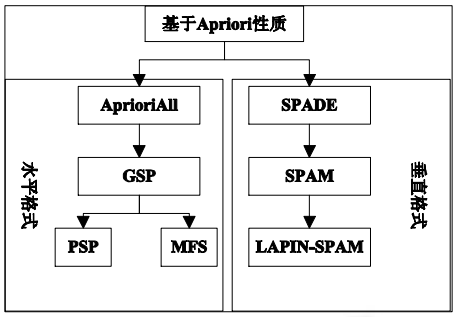

图 1 基于 Apriori 性质算法演化关系

表 1　垂直数据库

Sid	Items
10	(ac)
10	(b)
10	(ef)
20	(cef)
20	(b)
20	(a)
30	(ef)
30	(a)

表 2　水平数据库

Sid	Sequences
10	(ac)(b)(ef)
20	(cef)(b)(a)
30	(ef)(a)

GSP[2]在类 Apriori 算法基础上，引进时间约束、

滑动窗口与分层机制，更符合人们的目标需求，同时采用哈希树作为存储结构，因此候选序列与扫描数据库次数比类 Apriori 算法大大减少，算法效率比类 Apriori 算法将近提高了 20 倍。PSP[3]基于 GSP 基本原理，采用前缀树作为存储结构，只存储一次公共前缀，不需重复存储，提高了空间效率。MFS[4]对 GSP 的改进主要在 I/O 消耗上。GSP 每次扫描数据库都是通过连接相同长度的已知序列模式产生相同长度的序列模式。而 MFS 把挖掘过程分为两个阶段，先通过挖掘一个样本数据库，获得对序列模式的粗糙评价，然后根据这些评价进行挖掘。它改进 GSP 中候选序列产生的过程，从而可以连接不同长度的序列模式，产生不同长度的序列模式，它较早得到较长序列模式，并优化剪枝过程。

(2) 垂直格式挖掘算法

SPADE[5]基于数据垂直分布，把数据库映射成表，同时记录项的位置信息，通过连接表而得到序列模式。只需要 3 次扫描数据库：首先把数据库转化为垂直表示，第一次扫描产生 1-序列模式；将垂直格式转化为水平格式，第二次扫描生成 2-序列模式，同时构建格，使有相同前缀项的序列在同一格内；第三次扫描，动态连接产生所有的序列模式。在挖掘过程中，随着序列模式长度的增长，表越来越小，表连接的次数和候选序列的产生大大减小，比水平格式算法效率大大提高。SPAM[6]同样基于数据垂直分布，是第一个使用深度优先搜索策略挖掘序列模式的算法。它首次提出项扩展与序列扩展以及基于这两种方式的剪枝策略，使用位图表示数据库，提高计算支持度的速度，节省了时间开销；对序列的最大长度作了规定，并假设数据能全部装入内存，能一次输出不同长度的序列模式，较适用于挖掘长模式。LAPIN-SPAM 避免连接操作和比较操作，是 SPAM 的改进算法。

候选码生成—测试框架挖掘算法需多次扫描数据库，产生大量候选序列，大量的支持度计算，时间和空间开销大。这也激励研究人员另辟蹊径，研究不同挖掘框架算法，促使模式增长框架挖掘算法的诞生。

2.2 模式增长框架挖掘算法

模式增长框架挖掘算法在挖掘过程中不产生候选序列，通过分而治之思想把搜索空间划分成更小的空间，通过连接实现序列模式的增长。J. Han 在 2000 年首先提出不产生候选集的 FP-growth 算法，随后提出 FreeSpan[7]算法以及后来 J.Pei 提出的 PrefixSpan[8]算法都基于这一思想，如图 3 所示。

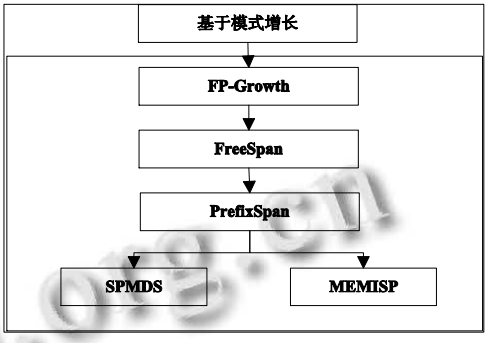

图 3 模式增长算法演化关系

首先给出几个相关术语。

前缀：设每个元素中的所有项目按照字典序排列。给定序列 α = <e_1,e_2,…,e_n>，b = <e_1',e_2',…, e_m'> (m<= n)，如果 e_i' = e_i (i <= m - 1)，e_m' $\subseteq$ e_m，并且(e_m-e_m')中的项目均在 e_m'中项目的后面，则称 b 是 a 的前缀。如序列<(ab)> 是序列<(abd)(acd)> 的一个前缀，而序列<(ad)>则不是。

投影：给定序列 α 和 β，如果 β 是 α 的子序列，则 α 关于 β 的投影 a'必须满足：β 是 α '的前缀，α ' 是 α 的满足上述条件的最大子序列。如序列 α =<(ab)(acd)>，其子序列 β = <(b)>的投影是 α ' = <(b)(acd)>，<(ab)>的投影是原序列<(ab)(acd)>。

后缀：序列 α 关于子序列 β = <e_1,e_2,…,e_{m-1},e_m'> 的投影为 α ' = <e_1,e_2,…,e_n> (n >= m)，则序列 α 关于子序列 β 的后缀为<e_m",e_{m+1},…, e_n>，其中 e_m" = (e_m-e_m')，如对于序列<(ab)(acd)>，其子序列<(b)>的投影是<(b)(acd)>，则<(ab)(acd)>对于<(b)>的后缀为<(acd)>。

投影数据库：设 α 为序列数据库 S 中的一个序列模式，则 α 的投影数据库为 S 中所有以 α 为前缀的序列相对于 α 的后缀，记为 S|α 。

投影数据库中的支持度：设 α 为序列数据库 S 中的一个序列，序列 β 以 α 为前缀，则 β 在 α 的投影数据库 S|α 中的支持度为 S|α 中满足条件 $\beta \subseteq \alpha . \gamma$ 的序列 γ 的个数。

(1) 基于 FP-tree 挖掘算法

FP-growth 算法使用 FP-tree 作为存储结构，将原始数据库压缩表示为 FP-tree，这比前缀树有更好的压

缩存储效果。它利用 FP-tree 自顶向下挖掘，在挖掘过程中不产生候选集，通过划分，缩小搜索空间，同时将单路径和多路径分开处理，大大提高了挖掘效率。但是 FP-tree 是根据共享序列前缀而构造的，当序列无相同前缀或是很少时，压缩效果不明显，所以比较适合挖掘共享序列前缀多的数据库。

(2) 基于投影数据库挖掘算法

FreeSpan[7]结合序列模式挖掘与频繁模式挖掘，它利用已产生的频繁集递归产生投影数据库，然后在投影数据库中增长子序列。算法不仅可以挖掘所有序列模式，还减少了产生候选序列所需开销，提高了效率。但由于该算法是基于序列的任何位置增长，组合数多，当某一模式出现频率高，投影数据库缩减不明显，同时也会产生大量投影数据。PrefixSpan[8]克服 FreeSpan 序列模式在任何位置增长问题，它只基于频繁前缀投影，确保序列向后增长，也确保了投影数据库的缩减。同时采用隔层投影和伪投影优化技术减少投影数据库的产生。PrefixSpan 的主要开销在于要构造大量投影数据库、当序列同源性较高时，存在大量重复投影数据库。SPMDS 算法通过对投影数据库的伪投影作单项杂凑函数，检测是否存在重复投影 ,避免大量重复扫描数据库。但是它只对简单项（序列元素只含一项）序列进行处理。

(3) 基于内存索引挖掘算法

MEMISP[9]基于内存索引，由 M. Y. Lin 和 S. Y. Lee 提出。它只需扫描数据库一次，将数据读入内存，不生成候选序列和投影数据库，通过递归使用查找-索引技术，获得长度不断增长的序列模式。算法对 CPU 和内存使用率高，在支持度较低时，也具有良好收缩性。对不能全部装入内存的数据库，则划分成可以装入内存的多个小数据库，再在每个小数据库中利用 MEMISP 挖掘局部序列模式，最后再一次扫描数据库，形成全局序列模式。算法最多扫描数据库两次，性能高于 PrefixSpan。

3 经典算法比较分析

在两类挖掘框架的经典算法中，最具代表性的是 AprioriAll、GSP、SPADE、PrefixSpan，其中前三个算法属于候选码生成—测试框架算法，而 PrefixSpan 属于模式增长框架算法。他们在数据结构，扫描数据库次数的异同如下表 3 所示

表 3　算法比较

	候选序列	存储结构	数据库缩减	原数据库扫描次数	算法执行
AprioriAll	产生	Hashtree	否	最长模式长度	循环
GSP	产生	Hashtree	否	最长模式长度	循环
SPADE	产生	序列格	是	3	递归
PrefixSpan	不产生	前缀树	是	2	递归

(1) GSP 与 AprioriAll

从表中我们可以看出 AprioriAll 与 GSP 算法有惊人的相似性。这是因为他们都是基于相同的 Apriori 原理，即任何频繁集的子集都是频繁的。基本思想是：在每一次遍历数据库时，利用上一次遍历时产生的频繁序列生成候选序列，并在遍历的同时计算支持度，满足支持度的候选序列作为下次遍历的序列。AprioriAll 是最早提出的算法之一，是 Apriori 思想最原始扩展，存在着以下的不足：1)生成数量庞大的候选序列。2）需要多次遍历原始数据库。当序列长度增加一，就要遍历一次数据库，遍历数据库的次数为最长序列模式的长度，遍历开销大。3）不容易发现长序列模式。随着挖掘序列长度的增加，候选序列呈指数增长。4）挖掘过程中数据转换开销大。

GSP 是 AprioriAll 的扩展改进算法。它引入时间约束、滑动时间窗口、分类层次技术，增加扫描约束条件，有效控制候选序列，减少无用模式的产生。它改进 AprioriAll 候选序列产生连接方式。只有当序列 S1 去掉第一个项目后与序列 S2 去掉最后一个项目的所得到的序列相同， S1，S2 才可以相连，S2 的最后一个项目附加到 S1 上所得的结果即为 S1 与 S2 相连所生成的候选序列，且在数据转化过程中不需要事先计算频繁集。基于以上原因，GSP 在执行效率上有了很大提高。J.Han[2]等在合成数据库与客户序列数据库进行对比，都有相似的结果。 在合成数据库中，当最小支持度逐渐增大时，GSP 效率提高在 30%到 5 倍之间。在客户数据库中，GSP 比 AprioriAll 快 2 倍到 20 倍。时间复杂度与序列元素个数成线性关系。但由于 GSP 与 AprioriAll 基于相同的原理，也存在着与 AprioriAll 相同的不足。

(2) GSP 与 SPADE

虽然 SPADE 算法也是基于 Apriori 原理，但从表中我们可以看出 SPADE 算法更优于 GSP 算法，这是由于它采用数据的垂直表示。利用数据垂直表示，把原始数据库转换为垂直数据库，之后对垂直数据库进行交集操作，从而产生频繁序列。具体步骤为：1)将原始数据库转化为数据垂直存储方式，扫描一次，产生长度为 1 的频繁项；2)将垂直格式转换为水平格式进行计算产生出长度为 2 的频繁序列；3)将具有相同等价类的序列进行自连接和交叉连接产生出新的等价类，在通过支持度计算，产生新的频繁序列。

实验表明，即使在支持度很低的情况下，SPADE[5] 的执行速度比 GSP 快将近 2 倍。这归结于以下几点：1）SPADE 的操作对象是 id-list，随着序列长度的增加，id-list 大小缩减，能快速连接。避免 GSP 中连接的相等比较。2）不采用哈希树结构，只对线性结构 id-list 进行搜索连接。3）只扫描数据库三次，避免 GSP 中多次对数据库的遍历。4）采用等价类进行挖掘，对原始数据库进行了缩减。SPADE 被认为是候选码生成—测试框架算法中运行最快的算法，但由于算法基于 Apriori 原理，当支持度很低，数据库很大，项集很少时，产生的垂直数据库大，id-list 表交叉操作时间效率与空间效率都不是很理想。

(3) 类 Apriori 与 PrefixSpan

与前三种算法相比，PrefixSpan 算法采用完全不同的挖掘策略，通过对频繁前缀投影，产生频繁后缀项，进而连接产生新的模式。不产生候选序列，最多两次扫描原始数据库 2 次，并且通过分而治之的思想，把数据库投影缩减成更小的数据库。每次迭代挖掘，都限定在投影后的更小的数据库中，节省了运行时间。同时为了减少投影数据数量以及重复投影数据库，采用隔层投影与伪投影技术。PrefixSpan 算法是公认的，目前挖掘算法中效率最好之一。

PrefixSpan 算法主要分为 4 步骤：1)找出所有长度为 1 的频繁序列。2)根据频繁前缀(prefix) 分割检索空间。3)找频繁序列的子集。4)返回步骤 3 进行递归挖掘，挖掘过程中产生的模式集合就是序列模式集。算法的主要开销集中在投影数据库的构建过程。经过测试实验比较，PrefixSpan 算法性能优于基于 Apriori 原理算法 GSP 和 SPADE。原因在于：1)模式增长方式不产生候选序列。传统的候选集生成-测试算法挖掘一个较小的数据库，生成和测试大量的候选序列模式开销巨大。2) 基于投影的分治是数据缩减(reduction) 的有效方法。投影数据库包含且仅包含用来挖掘那些由扩展得到的模式的必需信息，投影数据库的大小随着挖掘更长的序列模式进行而迅速缩减。3) PrefixSpan 需要的内存空间相对稳定。因为它采用分治的方法，不生成候选集。而 GSP 和 SPADE，当支持度降低时，由于需要容纳大量候选序列，需要相当数量的内存。同时基于模式扩展的方法，可以应用到多层次、多维数的序列模式中，也可以挖掘其他结构化的模式。

(4) 适用范围

一般而言，候选码生成—测试框架算法实现简单，比较适合稀疏型数据集，如有约束条件，则 GSP 更适用。因为在密集型数据集中，会产生大量的候选序列，运行时间消耗大，降低算法效率。SPADE 则比较适用于数据库中项集比较多，但各个项的出现并不是特别频繁的情况。模式增长框架算法在秘密型和稀疏型数据集中都适用，但实现过程比候选码生成—测试框架算法复杂，因此选择何种算法，要综合考虑挖掘对象与挖掘系统的各方面因素。

4 基于2-序列矩阵序列模式挖掘

C. Y. Hsieh、D. L. Yang 在 FSE[10]的基础上提出基于 2-序列矩阵 ESPE[10]算法。它提出 2-序列和 2-序列矩阵思想，优点在于：它只需扫描数据库一次，不需提前设定支持度，通过 2-序列矩阵对序列进行编码提高比较和查找效率。它把序列分割成两 X 与 Y 两部分，即 S=<X：Y>，X 是 2-序列，Y 是剩余序列。如 S=<abcdef>，X=<ab>为 2-序列，Y=<cdef>为剩余序列。在挖掘过程中，首先把序列分割，不产生候选序列，计算 2-序列索引值，用 2-序列矩阵对 2-序列进行编码，再把 Y 部分再一次进行分解，用 1-序列和 2-序列表示。因为任何的序列模式都可以分解为 2-序列与 1-序列的组合，从而实现序列模式的挖掘。使用者还可以设定需要挖掘的序列长度或特定序列。之后 W. C. Liao，D. L. Yang 等提出的 FEGC，J. J. Ru 提出的 FSPE 都是基于 2-序列，并在支持度的计算与候选序列的产生上进行了相应的改进。图 4 为 2-序列矩阵示例。

5 基于ZBDD序列模式挖掘

零压缩二叉决策图(ZBDD)[11]是一种特殊的有序

二叉决策图(OBDD) [11]。它不存储不相关结点，具有唯一性，PD 删除规则和合并规则，使 ZBDD 有很好的压缩存储功能。S. Minato 和 H. Arimura 首先把 ZBDD 用于数据挖掘领域，提出挖掘频繁模式 ZBDD-growth 算法和挖掘最大模式 ZBDD-growth-max 算法。E. Loekito and J. Bailey 在 S. Minato 基础上，提出 FIMiner，CFI-Miner，MFI-Miner 算法。第一个序列模式挖掘算法则是由 E. Loekito，J. Bailey，J.Pei 提出 SeqBDDMiner[11]。

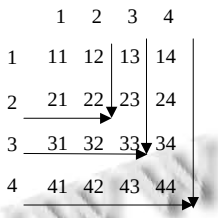

图 4　2-序列矩阵示例

SeqBDDMiner 算法首先根据输入的序列，构造初始 Sequence ZBDD，删除非频繁项。然后根据加权数对整个数据库进行剪枝，同时检查缓冲，是否存在的相同数据库挖掘。然后对 f_list 表中的每个频繁 X 项，找出后缀树，得出 f|x_list，在后缀树删除未出现在 f|x_list 中的项，得出投影数据库，挖掘出所有的频繁子序列与 X 相连接，然后以 Sequence ZBDD 输出结果。实验证明，该算法在挖掘支持度低，存在大量长序列和相似度高的序列数据库时，有很好的结果。这归结于 Sequence ZBDD 本身的结构特性。Sequence ZBDD 是 ZBDD 的一种变种，用以表示原始序列数据库，中间结果，以及挖掘结果。它不存储冗余结点，共享大量子图，每个内部节点的扇入边和扇出边，使得可以共享前缀及后缀，比之前的存储结构，有更好的压缩能力。从根结点到 1 叶子结点的一条路径表示一个序列，0-分支的变量必须按规定的变量序出现，1-分支则不受变量序约束。其目的主要是能共享更多子图。加权 Sequence ZBDD 则是为每个结点输入边赋予一个正数，代表结点所表示的所有序列数，同时采用缓冲机制，存储中间计算结果，不需重复计算。算法采用与 PrefixSpan[8]相似的模式增长挖掘框架，定义计算支持度和后缀树操作。

6　扩展模型序列模式挖掘算法

数据量的爆炸式增长，数据形式的多样化，数据库的不断更新，使传统挖掘模型经典算法面临巨大挑战。研究人员纷纷开始研究各种扩展模型序列模式挖掘算法。如下图 5 所示。

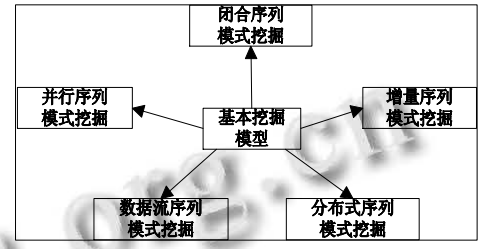

图 5　扩展模式序列模式挖掘

6.1 闭合序列模式挖掘

闭合序列模式挖掘是针对传统序列模式挖掘结果集庞大而提出的。它能完整表达结果集，但却更加精简。对于序列模式 S，如果不存在 S 的真超序列 S’，并且 S’和 S 有相同的支持度，则称序列模式 S 是闭合的。X.F.Yan 提出第一个挖掘闭合序列模式算法 CloSpan[12]。CloSpan 基于 Prefixspan，采用两种剪枝技术，用哈希算法优化搜索空间，运行效率高，但需要维护已挖掘的闭合序列，当结果集大时，维护代价高。BIDE[13]克服需要维护中间结果的不足，采用双向扩展闭合检测方法，通过向后扫描和跳跃扫描优化技术，缩小剪枝搜索空间，比 CloSpan 有更好的算法性能。沙金等提出的 PosD[21]算法克服 CloSpan 中未充分利用项的位置信息，利用支持度、约束策略和位置信息来减少搜索空间。而林颖提出的 PosD*在原 PosD 算法中加入了时间约束，进一步减小搜索空间。

6.2 分布式序列模式挖掘

跨地区合作与交流，使很多的数据信息分布式存在各地区之间，如大型购物中心数据库分布存储于各地子数据库中。如何在分布式环境中挖掘序列模式受到研究人员的重视，并设计了一系列算法。V. Guralnik 提出基于内存分布存储的并行挖掘算法 STPF[17][23]和 DTPF[17]；S. Parkj 提出 PDM[17]；R. Agrawal 提出对 Apriori 算法的并行化 CD[17]算法；J. P. Lu 提出挖掘全局最大项目集，而 D. W. Cheung 提出 FDM[18]算法则是在分布式环境下挖掘关联规则。常鹏等提出的 DSPM 基于 PrefixSpan[8]，采用抽样检测技术，将任务分配到

多台计算机上，以多线程多进程执行，并通过伪投影减少投影数据库的生成。皱翔等提出的 FDMSP 算法采用频繁前缀投影划分空间，用频繁前缀指定选举站点统计序列的全局支持数，采用减少候选序列策略减小候选集，并用 3 个异步运行的子程序实现算法，以减少开销。但分站点传输到选举站的序列数多，合并后被删减的候选序列数多。胡孔发等提出一种以语法序列树存储局部序列模式，采用项扩展剪枝和序列扩展剪枝策略的算法 FMGSP，较适合挖掘长模式。

6.3 并行序列模式挖掘

当前并行算法大多是串行算法的并行化，主要有 Shintani 和 Kitsuregawa 提出的 MPSPM[22]，SPSPM[22] 和 HPSPM[22]。MPSPM 中，每个处理器处理全局候选序列，存在内存溢出问题；SPSPM 中，处理器处理全局候选序列的一个子集，HPSPM 采用 Hash 机制对候选序列进行处理，优于前两种算法。Zaki 提出的 pSPADE[19]，它利用垂直数据格式，格理论，异步机制，动态负载平衡机制取得了较好的效率，但由于有线带宽问题，算法的可扩展性受到限制。Valerie 等人提出的 STPF，可分为 DPF 和 TPF 两种。J. Han[8]把并行技术引入闭合序列模式挖掘，提出 Par-CSP，通过将伪投影分配给不同的处理器以实现并行化；马传香等人提出的基于改进的 DSG 图的 PMSP 算法，通过剪枝候选集及数据的合理分布，有效地解决了 I/O 代价问题；李庆华等也都提出了一系列有效的并行挖掘序列模式的算法。

6.4 数据流序列模式挖掘

数据流是大量的、连续的、以特定次序传输的数据序列，通常由实时监测系统、通信网络、Internet 传输信息、金融市场或是零售业的联机事务处理、电力供应网、工业生产过程、科学和工程实验、传感器和其他动态环境产生。当前基于数据流的序列模式挖掘研究较少，处在起步阶段。MFSDS-1[20]和 MFSDS-2[20] 针对数据流数据量大无法存储的特点，通过基于窗口模型来解决序列模式挖掘问题。但算法实验准确度依赖于入选度，当入选度过大，实验准确度下降。黄崇争等提出 CFSM[20]算法，该算法结合滑动窗口技术、近似序列集以及 SP-Tree 结构挖掘当前滑动窗口内频繁序列。C. I. Ezeife etc.提出 SSM 算法，他将数据流分解到大小可变的数据块中，利用一个 D-List 结构，来保存数据单项的频繁计数。当一个数据块形成后，利用基于静态数据库的序列模式挖掘算法对该块进行挖掘，然后将结果保存在一个全局前缀树中。Luiz F. Mendes, J. Han etc.提出了 SS-BE[21]算法和 SS-MB[21]算法。两种算法挖掘过程大致相同，都是把数据流分解成数据块，然后在每一个数据块中挖掘序列模式，并保存在字典树中，它们主要区别在剪枝策略上。

6.5 增量序列模式挖掘

现实世界中，数据库随着时间不断变动，要是每次数据库更新，都从新对整个数据库进行一次挖掘，不仅效率低而且浪费大量资源。于是研究如何在更新的数据库挖掘序列模式显得非常的重要。第一个增量序列模式挖掘算法是由 S. Parthasarathy, M. J. Zaki 等提出的 ISM[15]算法。ISM 算法基于 SPADE，仅考虑数据库增加新序列的情况，并提供与用户的交互功能。它构建一个增量序列格记录原有数据库序列的支持度及反向边界信息，通过扫描新增加部分数据库一次，把得出的信息与序列格中的信息连接，以决定需重新扫描的原数据库的大小，而不是重新扫描整个原数据库。它能挖掘出所有的序列模式，但当反向边界很大时，耗内存。

GSP+[14]和 MFS+[14]基于 GSP 和 MFS，只是采用的剪枝策略不同，它只遍历更新数据库计算支持度。ISE[15]扩展 ISM 只增加新序列的情况，还考虑在序列中增加后缀的情况。通过记录原数据库序列信息，以及连接已挖掘的序列模式与新增序列产生候选序列，不对原数据库已产生的候选序列重新扫描，不存储反向边界，但须多次扫描数据库，也未对序列中增加前缀的情况进行考虑。IUS[15]对 ISM 和 ISE 都进行了改进，扩展增加序列前缀情况，定义反向边界最小约束，减小内存消耗。之后的 IncSpan[16]和 IncSpan+基于 PrefixSpan[8]，提出半频繁模式概念，通过一个缓冲因子得到介于频繁序列和非频繁序列之间的半频繁序列集，当更新数据库时，只要维护半频繁序列集。PBIncSM 则是基于前缀树的增量挖掘算法，而 FS-Miner，RePL4UP (Revised PLWAP For UPdate), PL4UP (PLWAP For UPdate)则是 WEB 日志挖掘方面的增量算法。

7 结语

序列模式挖掘从提出发展至今，不管在应用方面，还是在算法效率方面，都取得了很大进步。序列模式

挖掘未来发展方向将集中在：1)传统挖掘算法研究。应用数学领域，如超图、粗糙集等理论，设计适合存储序列数据的数据结构，结合两类传统算法的优势，进一步提高算法的空间和时间效率。2)扩展模型算法研究。结合具体的应用领域，扩展传统模型，设计适合特殊领域挖掘算法。如分布式与并行算法是为了解决数据量大与异地存储问题，而进行的扩展。多维序列模式则是对实际情况进行了更客观考虑，更符合用户需求。还有周期模式挖掘，序列模式图挖掘等。3)负序列模式挖掘研究。研究序列数据库中缺失数据所隐含的有用知识。 4) 复杂数据挖掘研究。研究对象型数据库，关系型与对象型混合数据库中的序列模式挖掘。5) 挖掘结果可视化处理。使用户更好的理解，应用挖掘结果，是执行数据挖掘的目的所在。6）数据量控制研究。数据源与挖掘结果数据量是庞大的，而其中有些数据是噪声数据，无用数据，如何对数据进行处理，控制源数据与结果规模，对于算法效率的提高起着重要作用。

参考文献

1 Agrawal R, Srikant R. Mining Sequential Patterns. Proc. of the 11th Int. Conf. on Data Engineering. Taipei, 1995: 3–14.

2 Srikant R, Agrawal R. Mining Sequential Patterns:Generalizations and Performance Improvements. Proc. of the 5th Int. Conf. on Extending Database Technology. Avignon, France, 1996:3–17.

3 Masseglia F, Cathala F, Poncelet P. The PSP Approach for Mining Sequential Patterns. Proc. of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery. Berlin: Springer–Verlag, 1998:174–184.

4 Zhang Ming hua, Kao B, Yip Chi Lap, et al. A GSP-based Efficient Algorithm for Mining Frequent Sequences. Proc. of International Conference on Artificial Intelligence. Las Vegas, Nevada, 2001.

5 Zaki MJ. SPADE:An Efficient Algorithm for Mining Frequent Sequences. Machine Learning Journal, 2001,42(1):31–60.

6 Han Jia-wei, Pei J, Mortazavi-Asl B, et al. FreeSpan: Frequent Pattern–Projected Sequential Pattern Mining. Proc. of 2000 Int. Conf. on Knowledge Discovery and Data Mining. Boston, MA, 2000:355–359.

7 Pei J, Han Jia-wei, Mortazavi-Asl B, et al. PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. Proc. of 2001 Int. Conf. on Data Engineering. Heidelberg, Germany, 2001:215–224.

8 Lin Ming-Yen, Lee Suh-Yin. Fast Discovery of Sequential Patterns by Memory Indexing. Proc. of the 4th International Conference of Data Warehousing and Knowledge Discovery. London, Springer-Verlag, 2002:150–160.

9 Hsieh Chia-ying, Yang Don-lin, Wu Jung-pin. An Efficient Sequential Pattern Mining Algorithm Based on the 2-Sequence Matrix. Proc. of 2008 IEEE International Conference on Data Mining. Pisa, Italy, 2008:583–591.

10 Loekito E, Bailey J, Pei J. A binary decision diagram based approach for mining frequent subsequences. Knowledge and Information Systems, 2010,24(2):235–268.

11 Yan Xi-feng, Han Jia-wei, Afshar R. CloSpan: Mining Closed Sequential Patterns in Large Datasets. Proc. of the 3rd SIAM International Conference on Data Mining. San Francisco, 2003:166–177.

12 Wang Jian-yong, J. Han W. BIDE: Efficient Mining of Frequent Closed Sequences. Proc. of the 20th International Conference on Data Engineering. Washington: IEEE Computer Society, 2004:79–90.

13 Zhang Ming-hua, Kao B, Cheung D, et al. Efficient algorithms for incremental update of frequent sequences. Proc. of the Pacific-Asia Conference of Knowledge Discovery and Data Mining. London, Springer-Verlag, 2002:186–197.

14 Zheng Qin-guo, Xu Ke, Ma Shi-long, et al. The algorithms of updating sequential patterns. Proc. of the 5th International Workshop on Hign Performance Data Mining. Washington DC, 2002.

15 Cheng Hong, Yan Xi-feng, Han Jia-wei. IncSpan: Incremental Mining of Sequential Patterns in Large Database. Proc. of 2004 Int. Conf. on Knowledge Discovery and Data Mining. Seattle, 2004:527–532.

16 常鹏,陈耿,朱玉全.一种分布式序列模式挖掘算法.计算机应用, 2008,28(11):2964–2967.

17 Cheung DW, Han Jia-wei, Ng V T, et al. A fast distributed algorithm for mining association rules. Proc. of the 4th International Conference on Parallel and Distributed Information Systems. Los Alamitos, Cal, USA: IEEE Computer

Society Press, 1996:31−44.

18 Zaki MJ. Parallel sequence mining on shared-memory machines. Journal of Parallel and Distributed Computing, 2001,61:401−426.

19 黄崇争,吴元锡,陈红.数据流中一种有效的当前频繁序列挖掘方法.山东大学学报:理学版, 2007,42(11):37−39.

20 Luiz F. Mendes, Ding Bo-lin, Han Jia-wei. Stream sequential pattern mining with precise error bounds. Proc. of Int. Conf. on Data Mining (ICDM). Piscataway, NJ: IEEE, 2008:941−946.

21 陈卓,杨炳儒,宋威,宋泽锋.序列模式挖掘综述.计算机应用研究, 2008,25(7):1961−1963.

22 姜海辉.并行序列模式挖掘关键问题研究.合肥:合肥工业大学, 2009.

23 缪裕青.频繁闭合项目集的并行挖掘算法研究.计算机科学, 2004,31(5):166−168.

24 Patel Ashish, Patel Aisha. Graph based Approach and Clustering of Patterns (GACP) for Sequential Pattern Ming. International Journal on Computer Science and Engineering, 2011,3(4):1501−1504.

25 Gouda K, Hassaan M. Ming Sequential Patterns in Dense Databases. International Journal of Database Management Systems, 2011,3(1):179−185.

26 Laurent A, Dong-Li, Teisseire M. On Transversal Hypergraph Enumeration in Mining Sequential Patterns. IDEAS, 11thInternational, Banff, Alta, 2007,303−307.

27 Vijayalakshmi S, Mohan V, Sasirekha M.S et al. Extracting Sequential Access Pattern from Pre-Processed Web Logs. Proc. of 2011 International Conference on PACC. Coimbatore, Indai, 2011:1−6.

28 Tang Pei-yi, Turkia Markus P, Gallivan Kyle A. Ming Web access patterns with first-occurrence linked WAP-trees. Proc. of the 16th International Conference on Software Engineer- ing and Data Engineering. 2007,247−252.

29 童咏昕,张媛媛,袁玫等.一种挖掘压缩序列模式的有效算法.计算机研究与发展, 2010,47(1):72−80.

30 Liu Li-zhi, Liu Jun. Mining Web Log Sequential Patterns with Layer Coded Breadth-First Linked WAP-Tree.International Colloquium on CCCM, 2009.447−451.

31 Calders T. Giinther CW. Pechenizkiy M. and Rozinat A. Using minimum description length for process mining. in SAC'09. Proc. of the 2009 ACM Symposium on Applied Computing. ACM, 2009.1451−1452.