9(1):99 - 103.

金融时间序列频繁模式挖掘算法①

Mining Sequential Pattern in Financial Time Series Data

樊　伟[1]　黄　斌[1]　朱　冲[2,3]　王大为[1]

(1.桂林电子科技大学 信息与通信学院 广西 桂林 541004; 2.中国科学技术大学 自动化系 安徽 合肥 230026; 3.中国科学院 合肥智能机械研究所 安徽 合肥 230031)

摘　要：针对金融时间序列数据库信息，提出一种时间序列频繁模式自动发现算法，该算法首先构造投影树，然后采用深度优先策略遍历投影树，挖掘出所有最长频繁模式，实验结果表明，该算法成功地挖掘出满足约束的频繁序列，在相同条件、不同支持度情况下,取得了与传统 AprioriAll 方法相同的规则集，而运行效率优于 AprioriAll 方法。

关键词：金融时间序列 频繁模式 数据挖掘 模式发现

1　引言

时间序列是一组有序的随着时间改变的序列值或事件，从数学意义上看，如果对某一过程中的某个变量或一组变量 $X(t)$ 进行观察测量，在一系列时刻 t_1, t_2,… t_n,(t 为自变量，且 $t_1 < t_2 < \cdots < t_n$)得到的离散有序集合 $X_{t1}, X_{t2}, \cdots X_{tn}$ 称为离散数字时间序列，即随机过程的一次样本实现，而时序数据库是由时间序列组成的数据库。时序数据库广泛运用于各种领域，例如：科学实验的数据分析、股票市场的波动分析等等，为了进行预测，首先必须建立一个适当的预测模型，因此，如何从时序数据中挖掘出时序模式，就成为一个重要的研究课题。

金融时间序列就是将价格在不同时间上的不同数值，按照时间的先后顺序排列而成的数列。在金融市场中,信息连续地影响着市场价格变化，快速作出价格趋势判断是金融时间序列分析中的重点和难点，高频金融数据不但可以用来比较不同交易系统在价格发现方面的有效性,还可以用来研究某只特定股票买卖报价的动态性等等,目前,国内外对时间序列的数据挖掘方面的研究已经有了不少报道。文献[1]提出了一个事物型数据库中频繁项目集的启发式搜索策略－－Apriori 算法；文献[2]将 Apriori 算法引入事件序列频繁模式的分析中,提出事件序列中的频繁模式发现算法；文献[3]又进一步将该策略发展到时间序列中,提出了时间序列关联规则的分析,并提出一种固定窗口分割时间序列的方法,其不足在于计算复杂。

本文针对金融时间序列数据库信息,提出一种时间序列频繁模式自动发现算法，文中给出了算法的伪代码描述，并对算法的时间及空间复杂度进行了必要的分析，实验结果表明,在相同条件、不同支持度情况下,本文方法取得了与传统AprioriAll方法相同的规则集，而运行效率高于 AprioriAll 方法，满足金融时间序列实时性要求。

2　数据预处理[4]

非平稳性是金融时间序列其中最为显著的特征之一。随机过程的平稳性是指其统计特性不随时间而变化，一般是指一种广义平稳性，即随机过程的期望值和协方差与时间起点无关。平稳性条件是许多时间序列建模的基础条件。但由于影响市场的政治、经济、文化环境等随时间的变迁，时间序列的一阶矩，协方差不可能维持不变，因而通常表现为明显的非平稳性。

金融时间序列数据为非平稳序列,但其发展变化有其内在的发展规律,比较常见的方法就是通过确定其变化模式来做预测，具体而言，农产品价格时间序列的波动性一般比较小，但因为政策或特殊气候如暴

① 基金项目:广西研究生教育创新计划(2008105950810M420)
收稿时间:2009-02-25

雪、连续干旱等,也表现出随机波动特性，这种波动可以看作噪声信号。数据挖掘的目的是发现序列中隐含的一些本质规律，噪声的存在一方面会淡化规则，即降低规则的显著性，另一方面又可能提供一些“假规则”，从而严重影响挖掘的效果。因此在时间序列进行挖掘之前，有必要先对其进行平滑[5]、聚类[6,7]等预处理，尽可能消去一些随机噪声。

3 算法设计

定义 1. 称S ={$A_{t1}, A_{t2}, A_{t3}, \cdots, A_{tn}$}($t1 < t2 < t3 \cdots < tn$)，为时间序列(time sequence),其中 A 为观测的属性，ti ($1 \le i \le n$)为观测时的时间。

定义 2. 称一个序列S ={$A_{t1}, A_{t2}, A_{t3}, \cdots, A_{tm}$}($t1 < t2 < t3 \cdots < tn$)包含项的个数，为序列长度 l，长度为的序列记为 l-序列。

定义 3. 称序列S_1 ={$A_{t1}, A_{t2}, A_{t3}, \cdots, A_{tm}$}($t1 < t2 < t3 \cdots < tm$)在序列数据库$S$ ={$A_{t1}, A_{t2}, A_{t3}, \cdots, A_{tn}$} ($t1 < t2 < t3 \cdots < tn$)内出现的次数，为支持数，记为$\xi$，其中($1 \le m \le n$)。

定义 4. 称在序列数据库S ={$A_{t1}, A_{t2}, A_{t3}, \cdots, A_{tn}$}($t1 < t2 < t3 \cdots < tn$)中的支持数不低于给定支持度阈值$\xi$的序列$S_1$ ={$A_{t1}, A_{t2}, A_{t3}, \cdots, A_{tm}$},为频繁序列，记为$Support(S_1)$。

问题描述：给定序列数据库和最小支持度阈值，时间序列频繁模式发现就是要找出序列数据库中满足最小支持度阈值的频繁序列中的最长序列。每一个这样的最长序列就是一个最大频繁模式。

3.1 构造投影树

举例说明，假设有时间序列S ={$A_{t1}, A_{t2}, A_{t3}, \cdots, A_{tn}$}($t1 < t2 < t3 \cdots < tn$),其中$A_{ti} \in \{I_1, I_2\}$,序列{ $I_1, I_2, I_1, I_2, I_1, I_1, I_2, I_2, I_1, I_1$ }是这个时间序列的实例,设置时间窗口长度w =3,最小支持度计数ξ=2。

构造序列的投影树：首先，创建树的根节点，用”null”标记。从序列头开始顺序读取w 个记录，创建一个分支，<(I_1:1), (I_2:1), (I_1:1)>,该分支具有三个节点，其中第一个(I_1:1)作为树根的子女节点，接下来(I_2:1)作为它的子女，(I_2:1)还有个子女节点(I_1:1)。每个节点都包含两项:值和计数。时间序列挖掘时必须考虑到时间的先后关系，虽然值I_1出现了两次，也不能将它合并。接下来将时间窗口向后移动一位,读入一个序列{I_2, I_1, I_2}。搜索根节点的子女节点，没有发现I_2，为树根创建一个子女节点(I_2:1)，创建第二个分支<(I_2:1),(I_1:1), (I_2:1)>。再将时间窗口向后移动一位，读入一个序列{I_1, I_2, I_1}。搜索根节点的子女节点，发现I_1，将它的计数加 1，接着搜索它的子女节点，发现I_2，相应的计数加 1，将I_2的子女节点I_1计数加 1，以此类推，直至完成最后一个序列{I_1}的投影，即完成投影树的构造，如图 1：

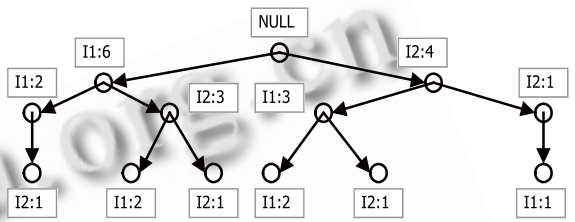

图 1　投影树

3.2 频繁模式搜索

投影树构造好以后，就要开始搜索频繁模式。由树根开始搜索，采用深度优先策略遍历树。首先，读入树根的第一个子节点(I_1:6)，节点(I_1:6)的计数大于最小支持度计数ξ，搜索节点(I_1:6)的子节点。读入节点(I_1:6)的第一个子节点(I_2:3)，同理，节点(I_2:3)大于最小支持度计数，搜索节点(I_2:3)的子节点。节点(I_2:3)的第一个子节点(I_1:2)计数为 2，且其没有子节点。因此，将{$I_1\ I_2\ I_1$} 添加到频繁序列集合中。回溯到节点(I_2:3)，搜索它的第二个节点(I_2:1)，小于最小支持度计数。回溯到节点(I_1:6)，搜索(I_1:6)的第二个节点(I_1:2)，满足最小支持度计数，往下搜索，找到节点(I_1:2)的子节点(I_2:1)，不满足最小支持度计数，回溯。将{$I_1\ I_1$}添加到频繁序列集合中。回溯到根节点，结束对根的第一棵子树的搜索。同理，搜索根的第二根子树，找到频繁序列{$I_2\ I_1\ I_1$}，获得的频繁序列集合{{$I_1\ I_2\ I_1$}，{$I_1\ I_1$}，{$I_2\ I_1\ I_1$}}。分析这个集合，它包含的最长序列长度为 3，最短为 2。对频繁序列集合中的每个 2-序列在 3-序列中搜索，可以发现序列{$I_1\ I_1$}是序列{$I_2\ I_1\ I_1$}的子序列，从频繁序列集合中将其移去。因为窗口长度为 3，而已发现的最长频繁模式的长度也为 3，有可能没有挖掘出所有最长频繁模式。利用已经发现的频繁 3-序列连接，生成频繁 4-序列候选集{{$I_1\ I_2\ I_1\ I_1$}},在原始序列中搜索其出现次数，找到 1 次，因此其不是频繁序列。

至此，挖掘过程结束，得到最大频繁序列集合{{ I_1 I_2 I_1}，{ I_2 I_1 I_1}}。

4 算法描述及分析

输入：时间序列S ={A_{t1}, A_{t2}, A_{t3}, ⋯, A_{tn}} ($t1 < t2 < t3 \cdots < tn$)，A 为观测属性，ti ($1 \le i \le n$)为观测时的时间属性；最小支持度计数阈值ξ；时间窗口w 。

输出：最大频繁序列集。

(1) 构造投影树

① built_tree：

创建投影树的根节点，以 ” null ” 标记。

a) 顺序读入序列 m 项到 Sub_list；

b) if 读入项小于 2，break；

c) else insert_tree(Sub_list)

② insert_tree(Sub_list)：

设 now_node 为树根,X[i]为 Sub_list 的第 i 项，i 初始值为 0；

a) If now_node 有子节点的值为 X[i]，设这个子节点为 node,node 计数加 1，now_node=node ；

b) else 为 now_node 创建子节点 node，值为 X[i]，计数为 1，now_node=node ；3) i++,if(i<Sub_list.length) 返回 1)；else 结束。

(2)频繁模式挖掘

① search(node):

if node 计数不小于最小支持度计数阈值ξ，then

for node 的所有子节点 sub_node，调用 search(sub_node)

if node 没有子节点或所有子节点计数小于最小支持度计数阈值ξ，将当前路径添加到频繁序列集中，return；

Return;

② 调用 search(root),得到初始频繁序列集。分析集合，去掉其中非最大频繁序列。如果集合中最长频繁序列长度等于窗口长度，且数量大于 1。这时需要连接最长频繁序列，生产候选集，搜索原始序列，这部分与 ApprioriAll 算法一致。如果集合中最长频繁序列长度小于窗口长度，或只有一条等于窗口长度，算法结束。

算法复杂度分析：投影树构造过程中，窗口需要逐位后移，只有一层循环，故该算法在投影树构造期间时间复杂度与样本数量 n 成正比，而在频繁模式挖掘过程中，树的的深度等于窗口长度，即为常数，不随样本数量 n 增长，则算法总的时间复杂度 T(n)=O(n),即线性时间复杂度。算法的空间复杂度则主要消耗在投影树的存储，由前面分析可知，投影树节点数量与窗口长度w 、属性数量、最小支持度计数ξ有关，而实际应用中，窗口长度、属性数量、最小支持度计数均为有限的常量数值，故该算法空间复杂度 S(n)=O(1),即算法为原地工作。

5 实验结果

从 www.sounong.net 抽取安徽省从 2004 年 2 月到 2008 年 1 月包菜价格数据，价格按 0.01 元进行离散。这段时间的包菜价格在 0.7~2.1 元每/公斤波动。

运行环境：PC 机，Pentium®D915 CPU，1024M RAM，Win2k 操作系统，Java 实现算法。样本数量为 1144，支持度计数阈值设定为 3，窗口宽度设为 7。耗用时间为 78ms,获得 112 条满足约束的频繁序列，如表 1 所示。

表 1　频繁序列发现结果

频繁序列长度	数 量
L=5	1 条，（0.91，0.92，0.93，0.94，0.94）
L=4	3 条，(1.09,1.07,1.06,1.06)
L=3	17 条，例如(1.06,1.05,1.05)
L=2	91 条，例如(0.85,0.83)

在相同条件下在不同支持度计数情况下比较了传统 ApprioriAll 与本文方法，其结果是两者获得的规则集相同，但计算时间差别较大，如图 2 所示。

由图 2 可知，本文方法的效率明显高于传统 ApprioriAll 算法,最大耗时仅仅需要 107ms，完全满足金融时间序列预测实时性要求。

6 结论

在本文中，以金融时间序列数据库为基础，设计并实现了一种序列内时序模式发现算法，实验表明，该算法成功地挖掘出满足约束的频繁序列，在相同条件、不同支持度情况下,取得了与传统 ApprioriAll 方法

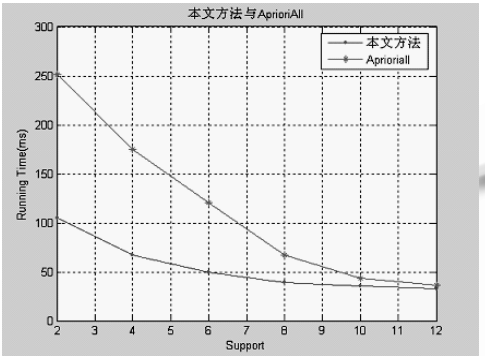

图 2 本文方法与 AprioriAll 算法在不同支持度情况下耗时

相同的规则集，而运行效率优于 AprioriAll 方法，满足金融时间序列预测的实时性要求，下一步的研究工作应该考虑对多个时间序列进行分析[4,8-10]，发现不同时间序列间可能存在的关联关系，发现这种多时间序列中的频繁结构模式对于人们更彻底的认识各个时间序列的相互影响并据此做出合理的决策具有重要的参考价值。

参考文献

1 Agrawal R, Ramakrishnan S. Fast algorithms for mining association rules in large databases. Proceedings of the Twentieth International Conference on Very Large Databases, Santiago: ACM Press, 1994. 487 - 499.

2 Manila H, Toivonen H, Verkamo AI. Discovery frequent episodes in sequences. Proc. of KDD,95.

3 Das G, Lin K, Mannila H, et al. Rule discovery from time series. Proceedings of Fourth Annual Conference on Knowledge Discovery and Data Mining. New York: AAAI Press,1998.16 - 22.Montreal: AAAI Press, 1995. 210 - 215.

4 朱冲,朱贤贵,张向利.金融时间序列挖掘综合模型.计算机系统应用, 2009,18(2):46 - 48.

5 Kwok CO, Etzioni O,Weld DS. Scaling question answering to the Web. ACM Trans. Information Systems, 2001,19(3):242 - 262.

6 史忠植.知识发现.北京:清华大学出版社, 2002.

7 Whitehead SD. Auto-FAQ: An experiment in cyberspace leveraging. Proc. of the Second International WWW Conference.1995.25 - 38.

8 王晓晔.时间序列数据挖掘中相似性和趋势预测的研究[博士学位论文].天津:天津大学, 2003.

9 黄河,黄轲,杭小树,熊范纶.时间序列中快速模式发现算法的研究.计算机工程与应用, 2003,39(21):192 - 194.

10 Oyama S, Kokubo T, Ishida T. Domain-Specific Web search with keyword spices. IEEE Trans. Knowledge and Data l