跨时间序列关联规则分析的高效处理算法

董泽坤[1] 史忠植[2] 李 辉[2]
[1](中国科技大学研究生院计算机学部,北京 100039)
[2](中科院计算技术研究所智能信息重点实验室,北京 100080)
E-mail:dongzk@ics.ict.ac.cn

摘 要 多元金融时间序列之间是互相影响的。该文就跨时间序列的关联规则挖掘提出一种新方法:ES-Apriori,此方法通过减少数据库扫描次数,优化内存分配,能够高效地分析多元时间序列之间的关联规则。试验表明,用此方法分析中国证券市场的股票时间序列非常有效。

关键词 关联规则 跨时间序列 ES-Apriori

文章编号 1002-8331-(2003)25-0196-03 **文献标识码** A **中图分类号** TP301.6

An Efficient Algorithm for Mining Inter-Time Series Association Rules

Dong Zekun[1] Shi Zhongzhi[2] Li Hui[2]
[1](Graduate School, University of Science and Technology of China, Beijing 100039)
[2](Key Laboratory of Intelligent Information Processing, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100080)

Abstract: There are many association rules among multiple financial time series.In this paper,a new algorithm named ES-Apriori will be presented to mine inter-time series association rules.This algorithm needs to search the whole database only time,with the rational arrangement of memory usage it can successfully mine the association rules among time series.Experiments have shown that this method can efficiently analyse the time series of Chinese Stock Market.

Keywords: association rule,inter-time series,ES-Apriori

1 引言

关联规则是数据挖掘领域的一项重要研究内容。关联规则的挖掘算法首先由 Agrawal 等在文[1]提出。随后关联规则挖掘有多种扩充,如多层关联规则挖掘[3][4],多维关联规则挖掘[5],多表间关联规则挖掘[7]等等。在传统的关联规则(Intra-transaction association rules)挖掘算法中,挖掘的关联项都出现在同一事务或同一序列中,用这些方法无法挖掘存在于不同时间序列中的关联关系[6],而"跨事务"的关联分析(Inter-transaction association rules)可以解决这一问题[6]。

跨事务性可以帮你们找到多笔交易记录中项目与项目之间的关联规则。如果将这些交易纪录加上时间维度,那么这些纪录发生在不同时间,彼此就有了先后的关系。跨时间序列关联规则分析可以用来分析属于同系列的金融时间序列,比如各种外币汇率的时间序列、银行利率与债券利率时间序列、证券市场各股票时间序列等。它可以发现类似如下规则:

*R*1:深发展涨(0),浦发银行涨(1)->民生银行涨(2)(20%,80%)

由于 *R*1 中加入了时间因子,各项之间体现的关系已经不再是事件是否发生在同一时间,而是不同时间序列中各事件之间的前后关系。很明显,规则 *R*1 对时间序列的预测很有贡献。

如何运用特殊数据结构来节省对数据库扫描的次数以提升"跨事务性"关联规则算法效能,仍然是一个需要解决的问题。针对这一问题,该文在如下两个方面对多元时间序列的跨时间序列关联分析进行了改进:(1) 减少数据库扫描次数,(2) 对读入内存的数据采用"分而治之"的策略。经过改进的算法称为:ES-Apriori。

用 ES-Apriori 算法设计的程序对中国证券市场五年近 500 支股票的时间序列分析,表明 ES-Apriori 是一种行之有效的跨时间序列关联规则分析方法。此算法只扫描一次数据库就可以产生所有频繁项集,同时降低了一次调入内存的数据量,提高了系统的效能,有效地缓解了数据爆炸耗尽内存的问题。

该文在第二部分介绍跨时间序列关联规则分析,第三部分给出 ES-Apriori 算法。试验在第四部分给出,最后是结束语。

2 跨时间序列关联规则 (Inter-time series association rules)

跨时间序列集:设时间序列的集合 $S=\{s_1,s_2,\ldots,s_u\}$ 是多元时间序列,T_i 是在时刻 i 对 S 的观察值集合,$T_i=\{s_1(i),s_2(i),\ldots,s_u(i)\}(1\leqslant i\leqslant n)$,跨时间序列集 D 定义为:$D=\{T_1,T_2,\ldots,Tn\}$。D 中每组观察值各分配一个识别号 *TID*。

跨时间序列关联规则:设 $\sum=\{e_1(0),\ldots,e_1(w-1),e_2(0),\ldots,e_2(w-1),\ldots,e_u(0),\ldots,e_u(w-1)\}$ 是项的集合,w 是多元时间序列

基金项目:北京市自然科学基金"源于信息获取知识的知识挖掘理论与技术研究"(编号:4011003)

作者简介:董泽坤,男,硕士,研究方向:数据挖掘,机器学习。史忠植,男,博士生导师,研究员,研究方向:人工智能、智能主体、机器学习。李辉,男,博士后,研究方向:人工智能、数据挖掘、模式识别。

集 D 的滑动窗口。以时刻 $t(1\leqslant t\leqslant n-w+1)$ 为 D 的参考时间基准点，如果此时事件 $e_i(1\leqslant i\leqslant u)$ 出现，则标记为 $e_i(0)$；时刻 $t+x$ $(0<x\leqslant w-1)$ 此事件又出现，则标记为 $ei(x)$。每一个 $ei(x)$ 分配一个识别号 *IID*。跨时间序列关联规则是形如 $X\Rightarrow Y$ 的蕴涵式，并且满足以下条件：

①$X\subset\sum, Y\subset\sum$，

②$\exists e_i(0)\in X, 1\leqslant i\leqslant u$，

③$\exists e_j\ (q)\in X, 1\leqslant j\leqslant u, ((i=j)\wedge(0<q<w-1))\vee((i\neq j)\wedge(0\leqslant q<w-1))$，

④$\exists e_i(p)\in Y, 1\leqslant i\leqslant u, \max(q)<p\leqslant w-1$，

⑤$X\cap Y=\Phi$

跨时间序列关联规则的支持度和置信度与传统关联规则的定义相似。

3 跨时间序列关联规则算法

和传统关联规则算法比较，跨时间序列关联规则算法要更复杂：

(1)要处理的数据超过算法能承受的范围后，频繁项集的数目将变得巨大而无法处理；

(2)候选集数目的增加导致更频繁的数据库扫描动作。

“跨事务性”关联规则算法 E-Apriori 和 EH-Apriori 在文献[6]中提出。算法的思想源于 Apriori。为了满足跨时间序列关联规则定义，所有的 C_2 都包括 $e_i(0)|(1\leqslant i\leqslant u)$ 项。EH-Apriori 算法在产生 L_1 时采用了哈希表技术，它的效率要比 E-Apriori 算法高。

E(H)-Apriori 算法找每个 L_k 需要一次数据库扫描，当得到所有频繁项集后再生成规则。此算法不但频繁访问数据库，而且将其应用到实际股票数据的预测分析中，当加大涨跌幅度分段数、加大滑动时间窗口时，效果并不理想。

3.1 ES-Apriori 算法

为了减少数据库扫描次数，减少程序运行消耗的内存，该文在 Apriori 算法的基础上，按照“分而治之”的思想提出了一种新算法。并把它命名为 ES-Apriori（扩展的分步 Apriori 算法）。算法见图 1。

```
Step 1
1  C1={{ei(x)}|(ei(x)∈Σ)∧(0≤x≤w-1)}
2  for each inter-time series transaction T, do
3    for each candidate c:ei(x)∈C1(ei∈T,+x)
4      {c.TID}∪=T, .TID
5  L1={<c.IID,{c. TID}>,c:{ei(x)}|(c∈C1)∧c.IID=ei(x).IID∧(|c.TID|≥support)}

Step 2
6  for each item:ei(0)∈L1{
7    for each item:ek(x)∈L1((x≠0)∨(x=0∧i<k))){
8      {c.IID}={ei(0).IID}∪{ek(x).IID};
9      for each candidate c:{ei(0),ek(x)}
10       {c.TID}={ei(0).TID}∩{ek(x).TID}
11   }
12   L'2={<{c.IID},{c.TID}>,c:{ei(0),ek(x)}|(c∈C2')∧(|c.TID|≥support)}
13   for(k=3;L'k-1≠Φ;k++){
14     C'k=ES-Apriori-Gen(L'k-1);//join L'k-1 with L'k-1 without prune step
15     for each candidate c:{ei(0),...,ek(x)}{
16       {c.IID}={ei(0).IID}∪...∪{ek(x).IID};
17       {c.TID}={ei(0).TID}∩...∩{ek(x).TID};
18     }
19     L'k={<c.IID,{c.TID}>,c:{ei(0),...,ek(x)}|c∈C'k∧(|c.TID|≥support)}
20   }
21   L'=∪k L'k
22   R'=GenerateRule(L');
23 }
24 L=∪k L'
25 R=∪R'
```

图 1 ES-Apriori 算法

ES-Apriori 算法在第一步遍历数据库并构造 C_1 频繁候选集（图 1 行 1-5），同时记录每个 C_1 的 *IID*，包含此 *IID* 的 *TID* 集合和 *TID* 集合的计数值。计数值满足最小支持度阈值的 C_1 将加入到 L_1 中。在第二步中挑出所有的 $e_i(0)$ 项，并分成 u 步，每步构造所有包含 $e_i(0)$ 项的频繁项集（图 1 行 6-23）。新产生的频繁 k-项集的{*IID*}由两个 k-1-项集的{*IID*}的并集组成，{*TID*}由两个 k-1-项集的{*TID*}的交集组成。将这 u 步得到的频繁项集合并（图 1 行 24），就是所有频繁项集的集合。

3.2 ES-Apriori 算法特性

ES-Apriri 算法从两个方面提高了系统的性能。

3.2.1 仅扫描一次数据库

在文献[2]中，一种命名为 ApririTid 的算法被提出。AprioriTid 通过构造一个不断收缩的临时数据库，减少了扫描数据库的开销。但是，AprioriTid 算法只降低了单次的数据库扫描空间，而总的扫描次数没有降低。ES-Apriori 算法继承并拓展了 AprioriTid 的思想。通过改变它的数据结构，将<*TID*,{*IID*}>改进为<{*IID*},{*TID*}>使算法扫描数据库的开销进一步减少。

ES-Apriori 算法仅在产生 C_1 时扫描一次数据库。对于每一个项 e_i，扫描跨时间序列集 D 的第 T_i 个观察值集合，如果 T_{i+x} 集合中包含项 e_i，则在记录所有包含 $ei(x)$ 项的 *TID* 链表中加入 T_i 的 *TID*。如果 *TID* 的计数值超过最小支持度计数值，将 $ei(x)$ 项加入到 L_1 频繁项集。由于所有数据库信息保存在 L_1{<*IID*,{*TID*}>}中，以后的所有操作都可以脱离数据库。由 L_1 构造 C_2{<{$IID_{l_1}^1, IID_{l_1}^2$},{$TID_{l_1}^1\cap TID_{l_1}^2$}>}时直接连接两个 L_1：$l_1\infty l_2$，并求 l_1 和 l_2 的{*TID*}的交集，计算交集中 *TID* 的个数就可以得到 $l_1\infty l_2$ 的支持度计数。这样，不用再次扫描数据库就可以直接生成候选集 C_k，近而得到频繁项集 L_k。

在图 2 所示的数据库中，假设有 5 项 Item，4 笔交易，最小支持度计数为 2。如图中所示，L_1 由满足支持度计数的 C_1 构成，C_2 由 L_1 连接构成。L_1 中的 *IID*=1 和 *IID*=2,3,5 连接可以构成{1,2}、{1,3}、{1,5}三项，它们的 *TID* 分别为{100,300}∩{200,300,400}={300}，{100,300}∩{100,200,300}={100,300}，{100,300}∩{200,300,400}={300}。由于{1,2}、{1,5}两项的 *TID* 计数值小于最小支持度计数值 2，所以不再加入到 L_2 中。如此类推，L_2 由 4 项组成，即：{1,3}、{2,3}、{2,5}、{3,5}。L_2 连接形成 C_3，C_3 只有一项，且满足最小支持度计数，所以<{2,3,5},{200,300}>加入到 L_3，算法结束。

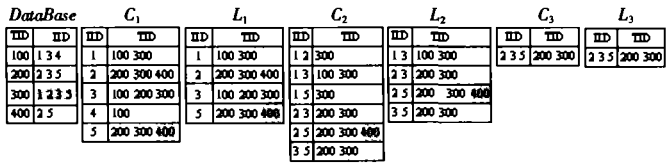

图 2 ES-Apriori 算法中候选集和频繁项集的产生

3.2.2 减少了单次调入内存的数据量

ES-Apriori 的所有大项集保留了各 item 的数据库信息，从而减少了数据库扫描次数，但是它的代价是使用大量内存。所以，如何合理分配内存，是 ES-Apriori 方法的另一重点。

利用跨时间序列关联规则性质（如下），可以分步构造 Lk，使每一步需要扫描的空间大幅缩减，从而降低内存的开销。

跨时间序列关联规则性质：跨时间序列关联规则的所有频繁项集都可以在各时间序列观察值的参考起点 $e_i(0)$ 的基础上产生。

此性质可以由跨时间序列关联规则定义中的条件2(频繁项集的 X 子集必定存在相对地址值为0的元素)推出。这一性质为 ES-Apriori 的分步策略提供了理论依据。

如图3所示,假设有2个基本项 A、B,滑动时间窗口=3,它的扩展1-项集为 A0、A1、A2、B0、B1、B2。考察6-项集{A0,A1,A2,B0,B1,B2},它包含的规则有且仅有:(1)A0,B0->A1,A2,B1,B2;(2)A0,A1,B0,B1->A2,B2。在计算这2条规则的置信度时,只需要搜索由 A0 构造的频繁项集空间,并不需要搜索由 B0 构造的频繁项集空间。因为这个6-项集的符合跨时间序列关联规则条件的所有 X 子集只有{A0,B0}、{A0,A1,B0,B1},这两项都是在 A0 的基础上构造产生的。同理,5项集{B0,A1,A2,B1,B2}的 X 子集{B0}、{B0,A1,B1}只须搜索由 B0 构造的频繁项集空间。

基本项	1-项集	2-项集	3-项集	4-项集	5-项集	6-项集
A	A0	A0A1	A0A1A2	A0A1A2B0	A0A1A2B0B1	A0A1A2B0B1B2
			A0A1B0	A0A1A2B1	A0A1A2B0B2	
			A0A1B1	A0A1A2B2	A0A1A2B1B2	
			A0A1B2	A0A1B0B1	A0A1B0B1B2	
				A0A1B0B2		
				A0A1B1B2		
		A0A2	A0A2B0	A0A2B0B1	A0A2B0B1B2	
			A0A2B1	A0A2B0B2		
			A0A2B2	A0A2B1B2		
		A0B0	A0B0B1	A0B0B1B2		
			A0B0B2			
		A0B1	A0B1B2			
	A1,A2	A0B2				
B	B0	B0A1	B0A1A2	B0A1A2B1	B0A1A2B1B2	
		B0A2	B0A1B1	B0A1A2B2		
		B0B1	B0A1B2	B0A1B1B2		
		B0B2	B0A2B1	B0A2B1B2		
			B0A2B2			
	B1,B2		B0B1B2			

图3 跨时间序列关联规则性质

从上面的分析得出,挖掘所有规则可以分成 u 步运行:每步只挖掘包含 $e_i(0)|(1\leqslant i\leqslant u)$ 的关联规则。这样,一次调入内存的数据可降低为全部调入的 $1/u$,当有大量数据项参与运算时,此方法也能顺利运行。

在 ES-Apriori 算法中,针对每一个 $e_i(0)$ 执行一次算法(图1第6-24行),每次得到的规则的集合即是全部的关联规则。

ES-Apriori 算法分割了频繁项集空间,降低了一次调入内存的数据量,从而缓解了因数据爆炸而耗尽内存的问题。

4 试验

该文使用中国证券市场1997-2001五年共1125个交易日近500支股票的收盘价时间序列作为测试集,比较了 E-Apriori 和 ES-Apriori 算法的性能。

试验使用 PIII667,512M 内存,操作系统是 Win2000 Server 的计算机。每只股票的涨跌幅度分成两段,频繁项最大长度为6,时间窗口为3,最小支持度为1%,分别使用含有5k-30k个数据项的数据测试,结果如图4所示。

由图4可知,当总的 item 项小于20k时,E-Apriori 和 ES-Apriori 的执行效率都很高。但是随着数据的增加,E-Apriori 的内存使用量将急速增加,导致运算时间骤然变长;而 ES-Apriori 无论在内存上还是在时间上都呈现平稳增加的态势。在试验中,当总的 item 项大于30k后,E-Apriori 会耗尽计算机内存而无法继续运行;而 ES-Apriori 却可以顺利运行。试验结论证明,分析较大数据量的跨时间序列关联规则时,ES-Apriori 算法在时间/空间性能上要优于 E-Apriori 算法。

5 结论

为了解决 E(H)-Apriori 算法在加大数据量后运算不理想的问题,该文提出了一种新的跨时间序列关联规则分析的方法

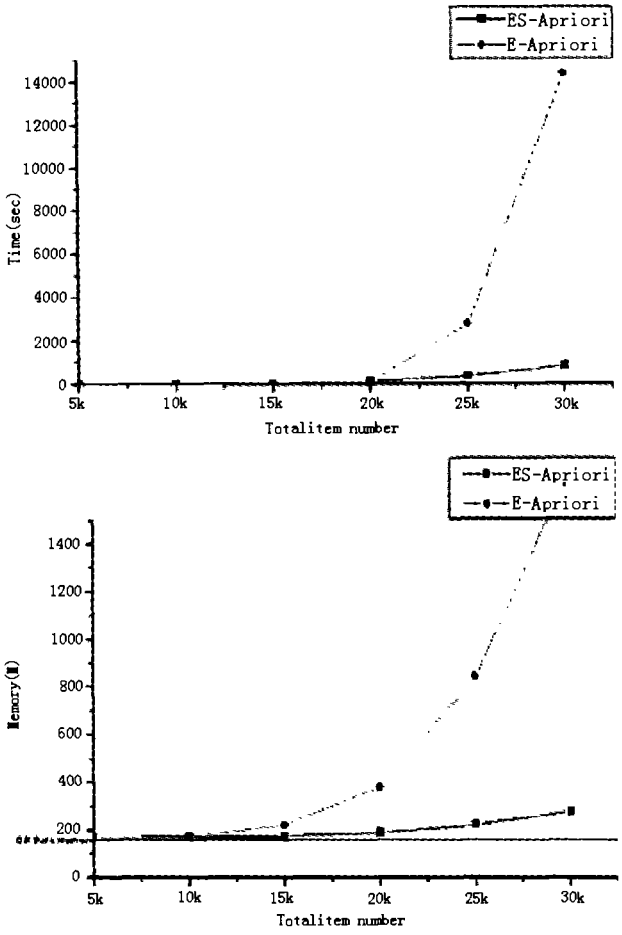

图4 ES-Apriori 和 E-Apriori 在时间/空间性能对比

ES-Apriori。由于 ES-Apriori 在计算中使用了分步策略,使得每步计算的搜索空间急剧下降,内存的开销也较 EH-Apriori 小很多。所以,在增大数据、增大时间窗口、加大涨跌幅度分段数时,算法仍然能够顺利运行。该文使用中国证券市场1997-2001五年间1125个交易日,近500只股票的时间序列作为试验数据,证明 ES-Apriori 非常有效。(收稿日期:2003年5月)

参考文献

1.R Agrawal,T Imielinski,A Swami.Mining assocation rules between sets of items in large databases[C].In:Proc of the ACM SIGMOD Conference on Management of Data,1993

2.R Agrawal,R Srikant.Fast algorithms for mining association rules[C].In:Proc of the 20th Conference on Very Large Data Bases,1994

3.J Han,Y Fu.Discovery of multiple-level association rules from large databases[C].In:Proc of the 21th Conference on Very Large Data Bases,1995

4.R Srikant,R Agrawal.Mining generalized association rules[C].In:Proc of the 21th Conference on Very Large Data Bases,1995

5.M Kamber,J Han,Y Chiang.Metarule-guided mining of multi-dimensional association rules using data cubes[C].In:Proc of the Knowledge Discovery and Data Mining,1997

6.H Lu,J Han,L Feng.Stock movement and n-dimensional inter-transaction association rules[C].In:Proc of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery,1998

7.史忠植.知识发现[M].北京:清华大学出版社,2002