第36卷 第19期 计 算 机 工 程 2010年10月
Vol.36 No.19 Computer Engineering October 2010
·软件技术与数据库· 文章编号：1000—3428(2010)19—0110—03 文献标识码：A 中图分类号：TP391.4

基于时序向量聚类的周期关联规则发现算法

罗 兰[1,2]，曾 斌[2]
(1. 浙江大学计算机学院，杭州 310027；2. 浙江林学院信息工程学院，浙江 临安 311300)

摘 要：针对目前周期关联规则难以划分时间区域和基础算法效率低等问题，提出一种基于周期关联规则的发现算法(CARDSATSV)。采用由项目支持度组成的时序向量作为时域数据特征点进行聚类，用DB Index准则控制聚类个数以达到最佳的聚类效果。给出CFP-tree算法来发现周期关联规则，利用基于条件FP-tree的周期性剪裁技术提高算法效率。实验表明，和目前周期关联规则发现算法相比，CARDSATSV可以发现更多有用的周期关联规则，时空效率有一定的提高。

关键词：时序向量；强周期关联规则；差异序列法；周期FP-tree算法；差异序列聚类算法

Discovering Arithmetic of Cyclic Association Rules Based on Time Series Vector Clustering

LUO Lan[1,2], ZENG Bin[2]
(1. College of Computer, Zhejiang University, Hangzhou 310027, China;
2. School of Information Engineering, Zhejiang Forestry University, Lin'an 311300, China)

【Abstract】 The existing cyclic association rules have disadvantage to compartmentalize a cycle into several time segments and the base arithmetic disadvantage is low-level efficiency etc. This paper presents CARDSATSV. It chooses the time sequence vector which consists of the support of item to cluster, and uses DB Index to determine the optimal class number of cluster. It brings forward Cyclic FP-tree(CFP-tree) to discover cyclic association rules. CFP-tree handle cycle clipping technology is based on conditional FP-tree to improve efficiency. Experiments show that CARDSATSV can discover more useful cyclic association rules and can improve efficiency, compared with the existing cyclic association rules.

【Key words】 time series vector; lusty cyclic association rules; difference sequence arithmetic; Cyclic FP-tree(CFP-treee) algorithm; clustering arithmetic based on difference sequence

1 概述

关于时间区域内周期关联规则的研究目前在国内外处于初期起步阶段。目前已有的周期关联规则发现算法存在的问题主要有：

(1)时域数据特征点的选择。文献[1]中提出的周期关联规则时间是人为确定的。针对文献[1]的问题，文献[2]提出一种周期模型，通过聚类将周期分为长度不同的时间段，可以更准确地发现周期关联规则。但这种周期关联规则发现模型选择每一时刻发生的事务数目为时域数据特征点进行聚类，其聚类是针对事务进行的，并不能反映单个项目的规律。

(2)周期时间区域分段数目的确定。文献[1-2]将一个周期分成多少个时间段是人为确定的，聚类个数的选择要根据具体数据的实际情况来判定，而不是人为规定。

(3)发现周期关联规则基础算法的选择。文献[1-2]的周期关联规则算法都基于Apriori[3]算法，他们都存在着处理的候选项集十分大、资源耗用高、运行效率不高等问题。

本文提出一种基于周期关联规则的发现算法(CARDSATSV)解决上述问题，CARDSATSV由两部分组成：CMDSA和CFP-tree。

2 时序向量的聚类算法(CMDSA)

2.1 时间区域划分

在第一周期进行聚类，确定各个时间区域，以后其他周期的时间区域也按此划分。

2.2 聚类的时域数据特征点选取

本文选择每一个项目在每一时刻发生项目的支持度形成的时序向量为时域数据特征点。

2.3 CMDSA算法步骤

CMDSA[4-5]算法的具体步骤如下：

(1)计算时序向量序列 E 的 m-1 个差异度 $g[i]$和 m-2 个二次差异度 $h[i]$，形成相应的差异度和次差异度数组。

(2)：序列 E 聚类个数 c 取值由 $2\sim\sqrt{m}$，在二次差异度序列数组中先找到最大的二次差异度值 $h[i]$，确定相应分割点，然后在剩余 $h[i]$中找最大的 $h[i]$，再确定相应分割点，如此类推。

(3)由于排序使每个 $h[i]$的初始位置被打乱，要想找到 $h[i]$后确定相应的分割点位置，记录 $h[i]$的初始位置，让 $h[i]$包含2个分量的结构体变量，$h[i].data$ 是二次差异度大小值，$h[i]$的位置 $h[i].place$ 存放了初始位置 i。

(4)步骤(2)的分步骤1：在步骤(2)中每找到一个 $h[i]$，就根据 $h[i]$确定分割点。

(5)步骤(2)的分步骤 2：在步骤(4)中每确定一个新分割

基金项目：浙江省自然科学基金资助项目(Y1090603)；浙江省科技厅科技计划基金资助项目(2009C35012)
作者简介：罗 兰(1979－)，女，硕士研究生，主研方向：数据挖掘；曾 斌，讲师
收稿日期：2010-04-13 **E-mail：**luolan_313@yahoo.cn

点，根据新变化的 *Fen* 利用过程 *INDEX*(*c*, *Fen*)计算与新分割点对应的 DB_c 值。DB_c 值和 *DB**比较，若 DB_c 值小于 *DB**，将 DB_c 覆盖 *DB**，同时将与 DB_c 值对应链表 *Fen* 覆盖 *Fen**。当步骤(2)中执行结束就找到了最小的 DB_c 值。

3 周期关联规则发现算法(CFP- tree)

3.1 发现周期频繁项目集的思路

思路 1：在每个周期的各个时间段构造 FP-tree 树，产生频繁项目集，所有周期对应时间段之间进行频繁项目集比对，产生周期频繁项目集，但是时间开销巨大不可取。

思路 2：在第一个周期的各个时间段构造 FP-tree 树，产生第一个周期各个时间段的频繁项目集，将这些频繁项目集和其他所有周期对应时间段的事务库进行比对，产生所有周期频繁项目集。

思路 3：进一步优化思路 2 的方法，先在第一个周期各时间段构造 FP-tree 树，从周期频繁项目列表 *L* 出发，采用基于条件 FP-tree 的周期性剪裁技术生成所有周期频繁项目集，进而发现所有强周期关联规则。思路 3 也就是本文的 CFP-tree 算法。

3.2 CFP-tree 算法步骤

3.2.1 FP-tree 树的构造过程

根据前面 CMDSA 在第一个周期得到的 *c* 个最合理时间分段，每一个时间分段一个事务数据库，就有 *c* 个事务数据库 $D_1, D_2,\cdots,D_C$(D_i 对应第 *i* 个时间分段)，一个事务数据库一棵 FP-tree 树就需要构造 *c* 棵 FP-tree 树[6]。

3.2.2 基于条件 FP-tree 的周期性剪裁

基于条件 FP-tree 的周期性剪裁技术的运用使得思路 3 优于思路 2。下面分析该技术的有效性。

以图 1 中的生成树结点 *ma* 为例，设 *M* 是 *ma* 的条件模式基中所有涉及到的项目形成的项目集，在 *ma* 的子结点 *mac* 和 *maf* 中有 $\{c\}\subseteq M$; $\{f\}\subseteq M$ ，所以，在图 1 的 FP-tree 中，*ma* 的周期性支持度大于其所有子结点 *mac*、*maf* 的周期性支持度，同理 *mac*、*maf* 的周期性支持度也大于各自的子结点的周期性支持度。以 *ma* 为根结点的子树中，根结点 *ma* 的周期性支持度最大。

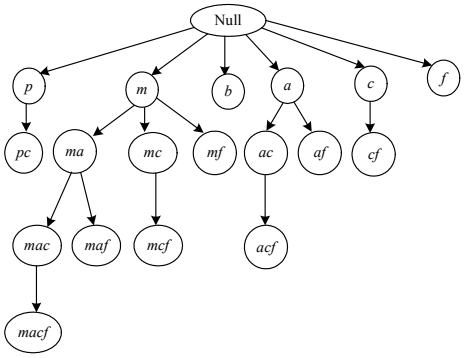

图 1 在第一周期时间段$[s_j,e_j]$中频繁项目集生成树

如果采用思路 3 的基于条件 FP-tree 的周期性剪裁技术，只有根结点 *ma* 需要和其他周期对应时间段事务数据库进行比对一次，而除了 *ma* 的其他所有结点都不用和其他周期对应时间段事务数据库进行比对。所以，在周期频繁项目集生成树中，以 *ma* 为根结点的子树在图 2 中不会生成。这样和思路 2 相比就节约了生成以 *ma* 为根结点的子树的代价，同时节约了该子树进行周期性比对的代价。在 *ma* 不是周期性频繁集的条件下，思路 2 先在第一周期生成频繁项目集生成树(如图 1 所示)，经过周期性剪裁后生成图 2 的频繁项目集生成树，思路 3 在第一周期直接生成图 2。所以，思路 3 中基于条件 FP-tree 的周期性剪裁技术能有效控制周期频繁集生成树的规模，减少搜索空间。

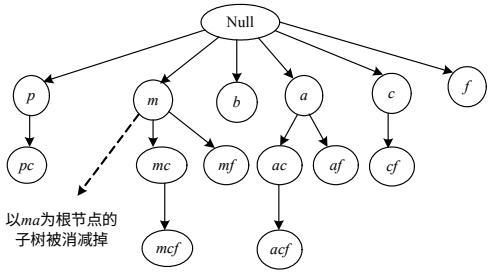

图 2 时间段$[s_j,e_j]$中周期频繁项目集生成树

基于条件 FP-tree 的周期性剪裁技术的具体思路如下：主要是对文献[6]的 FP-growth 的改进，增加 FP-growth 发现周期频繁项目集的功能，改进算法是后面的 CFP-growth。CFP-growth 对 FP-growth 中的模式 β 的条件 FP-tree 进行周期性剪裁。也就是将模式 β 和 β 的条件 FP-tree 形成的集合(例如 β 为$\{kma\}$，β 的条件 FP-tree 为$\{(f{:}3,b{:}3)\}|kma$，则形成的集合$\{\{fkam\},\{bkam\}\}$)和其他周期的对应时间分段$[s_j,e_j]$中所有事务进行比对，剪裁掉集合中周期性支持度小于 $s_{\min}$ 的项目集，生成模式 β 的周期性条件 FP-tree。在第一周期每一个时间分段$[s_j,e_j]$的 FP-tree 遍历完成时，即生成所有周期频繁项目集。

3.2.3 周期频繁项目集的发现过程

输入 第一个周期的 *c* 棵 FP-tree:$FP\text{-}tree_1$, $FP\text{-}tree_2,\cdots$, $FP\text{-}tree_c$，其中，$FP\text{-}tree_i$ 是第 *i* 个时间段$[s_i,e_i]$产生的 FP 树

输出 所有周期频繁项目集

Procedure CFP-growth (FP-tree$_i$,Null) //对模式 β 的条件模式 //FP-tree 进行周期性剪裁生成周期频繁项目集的过程

if Tree 包含一条单一路径 P then

for each 路径 P 中结点组合(记为 β)//β 是路径 P 中的结点元素 //的合并

产生 α ∪β 模式，使其支持度等于 β 中各结点元素的最低周期性支持度;

Else For each Tree 中所有头元素 a_i do

begin

产生模式 β= a_i∪ α 使其支持度等于 a_i 的周期性支持度;

构造 β 的条件模式基和 β 的条件 FP-tree β_c;

Treeβ_c 和 β 构成的集合 C;

CFP-Pruning(C)

If Treeβ<>φ then CFP-growth (Treeβ,β);

End//结束 5)

Procedure CFP-Pruning(C)

for(j=2;j<=n;j++)do begin//2 到 n 个周期

C_t=φ ;//清空 C_t

for all c ∈ C do begin //计算集合 C 中所有项目集 c 和第 j 个周 //期第 i 个时间段$[s_i,e_i]$的事务数据库 D_{ji} 中支持度，剪裁掉

支持度 $support_c$ 小于 s_{min} 的项目集

计算项目集 c 在 D_{ji} 中支持度 $support_c$;

If $support_c >= s_{min}$ then $c\cup C_t$;

$C=C_t$;

If $C=\phi$ then exit; //若所有项目集都被剪裁掉，则退出所有 //循环

end

If $C \neq \phi$ then begin

根据 C 生成 Treeβ; //例如集合 C={{fkam}, {bkam}}, β=kam, //Treeβ 为{(f:3,b:3)}|kma

Return Treeβ; //输出 Treeβ

end

else Return;

End

3.2.4 强周期关联规则的发现过程

每个周期的每一个时间段$[s_i,e_i]$中的强周期关联规则可用该时间段中找到的所有周期频繁项目集直接产生。具体为对每一个周期频繁项目集 L 的所有非空的子集合 a，若 $support(L)/support(a) \geq c_{\min}$，则有强周期关联规则 $a \rightarrow (L\text{-}a)$ $[C,s_i,e_i,s_{a\rightarrow(L-a)},c_{a\rightarrow(L-a)}]$。

4 算法分析与实验结果

测试的软/硬件平台和参数如下：(1)硬件主控系统 MICROGRID ZMG08-9 型；(2)编程语言 VxWorks C 环境，嵌入式操作系统 VxWorks 6.7；(3)一天取 60 万条预处理数据(包含线电容、偏心、外直径、发泡度及椭圆度等工艺参数)。周期为一天，那么时序向量序列 E 经历的时间跨度 $Time(E)$=1 天。采样时间粒度 $Granularity(E)$为 1 s，即 1 s 采样一次，所以一个周期采样数目 $Lend(E)$=86 400 个采样值。实验数据共有 300 个项目，所以每个采样值有 300 个属性指标(项目)，每一个采样值是 300 维的时序向量。因为一个周期采样数目 $Lend(E)$=86 400 个，所以 $C_{\max}=\sqrt{Lend(E)} \approx 294$，经过 CMDSA 算法聚类后得到 C_{opt}为 95，在同样条件下实际生产中将一天分成 94 段左右是合理的，因此，经有效性函数 DB Index 准则判断 C_{opt}为 95 是实际可行的。

如图 3 所示，在相同的最小支持度和相同的时间总长度 T 情况下，基于 CMDSA 的 CARDSATSV 算法发现周期频繁项目集数量多于文献[2]中基于 Fisher 的周期性关联规则模型发现的周期频繁项目数量，说明了本文 CARDSATSV 算法优于文献[2]中周期性关联规则模型。

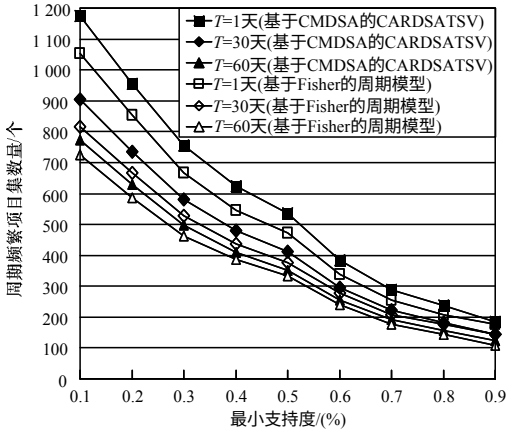

图 3 发现有效周期频繁项目集数量的比较

由于文献[2]中 Fisher 算法的周期性关联规则模型比文献[1]《Cyclic Association Rules》的算法发现更多有用周期关联规则，因此 CARDSATSV 算法也优于文献[1]算法。

由图 4 有：(1)2 种算法的时间开销随着最小支持度的增加而减少。即最小支持度越高，淘汰的项目越多。(2)CARDSATSV 算法远低于基于 Apriori 的周期关联规则算法时间开销。

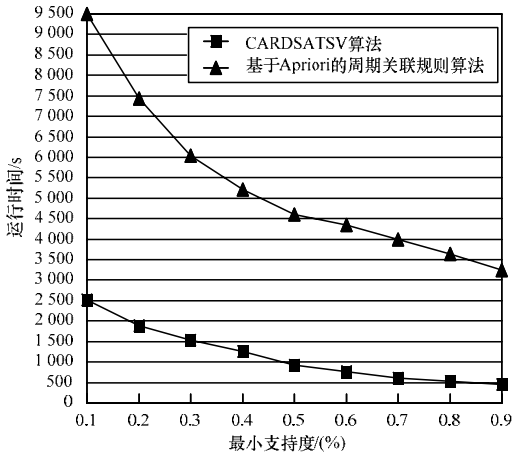

图 4 T=30 天时在不同最小支持度下运行时间的比较

5 结束语

本文提出基于时序向量差异序列法聚类的周期关联规则发现算法(CARDSATSV)，各项实验和应用表明本文的 CARDSATSV 算法适用于工业生产的周期性海量数据的分析和预处理。CARDSATSV 算法的实际研究成果——线缆在线时频分析与工艺缺陷的关联系统已投入前期运行阶段，为工业生产中在线工艺参数的变化频谱中的分析各种周期性工艺缺陷提供有效解决办法，帮助解决各种工艺缺陷和在线工艺优化等问题。

在以后发展中需要考虑每一个周期中事务数据库的事务数目变化、周期性支持度阈值的变化等情况下周期关联规则的更新维护问题。

参考文献

[1] Ozden B, Ramaswamy S, Silberschatz A. Cyclic Association Rules[J]. IEEE Trans. on Data Engineering, 1998, 8(9): 412-421.

[2] 徐 敏, 金远平. 一种新的周期性关联规则模型[J]. 计算机工程与科学, 2000, 22(4): 78-81.

[3] Agrawal R. Fast Algorithms for Mining Association Rules[C]//Proc. of the 20th International Conference on Very Large Databases. Santiago, Chile: [s. n.], 1994: 487-499.

[4] 程乾生. 一种新的样品聚类方法——差异序列法[J]. 科学通报, 1994, 39(2): 8-12.

[5] Theodoridis S. 模式识别[M]. 李晶皎, 译. 2 版. 北京: 电子工业出版社, 2004: 163-205.

[6] Han Jiawei. Mining Frequent Patterns Without Candidate Generation[C]//Proc. of ACMSIGMOD Conference on Management of Data. Dallas, TX, USA: [s. n.], 2000: 1-12.

编辑 任吉慧