广义无冗余情节规则抽取方法研究

尤 涛,徐 伟,杨 凯,杜承烈,钟 冬
(西北工业大学计算机学院,陕西西安 710129)

摘 要: 情节规则挖掘旨在发现频繁情节之间的因果关联,现有无损情节规则挖掘方法没有考虑多规则间的关联关系,故而存在大量冗余.利用演绎推导特性对情节规则间的关联关系进行建模,引入无冗余情节迹规则的概念,分析了情节迹冗余的原因,通过最大重叠项冗余性检查给出广义无冗余情节规则抽取算法;证明了广义无冗余情节规则对情节规则的等价表达能力.理论分析和实验评估表明该算法在处理效率基本不变的前提下,提高了情节规则的生成质量.

关键词: 事件序列; 演绎; 情节迹; 最大重叠项; 情节规则
中图分类号: TP311 **文献标识码:** A **文章编号:** 0372-2112 (2015)02-0269-07
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2015.02.010

Research on Extracting Generalized Non-Redundant Episode Rules

YOU Tao, XU Wei, YANG Kai, DU Cheng-lie, ZHONG Dong
(*Department of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an, Shaanxi* 710129, *China*)

Abstract: Aiming at the problem that current nondestructive episode rule mining algorithms don't consider the relationship between episode rules and generate redundancy, we model the relationship among the episode rules by using deduction characteristic, and introduce the concept of non-redundant episode trace rules. We also analyze reasons for episode trace redundancy, and present the generalized non-redundant episode rules mining algorithm based on the redundant checking on maximum overlap items. Then we prove that generalized non-redundant episode rules keep the equivalent expression ability to episode rules. Theoretical analysis and experiments demonstrate this algorithm improved the quality of generatedepisode rules with almost the same efficiency.

Key words: event sequence; deduction; episode trace; maximum overlap items; episode rule

1 引言

情节刻画了事件间的紧随关系,而情节规则描述了情节内的因果关联.自 Mannila 等[1]引入情节规则的概念以来,情节规则挖掘问题[2,3]一直是数据挖掘领域研究的热点之一.基于情节规则匹配的预测方法被广泛应用在传感器数据处理、网络安全监控、金融证券管理、软件规范挖掘、城市交通管理、事务日志分析等众多领域[4].在典型预测方法中,无论是 Laxman 等人[5]提出的生成模型预测算法,还是 Cho 等人[6]提出 ToFel 算法、基于后向检索规则前件策略的数据流预测算法 CBS-Tree、CBS-Tree 改进算法[7],它们的核心都是通过查找规则前件的发生来进行预测.而从历史数据中挖掘出的规则数量,直接影响各预测算法的预测效率.因此,针对规则精简集的研究一直比较活跃.

已有情节规则挖掘的各类算法,虽然在数据组织、处理流程等方面各有不同,但主要分为三类,如表 1 所示.

产生情节规则全集的典型算法为 TASA[8]、WinMiner[9],该类算法以频繁情节为规则基,通过投影的方式产生情节规则全集.

产生最小前件情节规则全集的典型算法为 GenMiner[10],其规则基为频繁情节与生成子.算法首先采用深度优先的搜索策略来创建存储所有情节的前缀搜索树 PSL,然后通过遍历 PSL 得到包含所有情节模式生成子的超集,据此可以得到最小前件情节规则.

产生无冗余情节规则集的典型算法为 Extractor[11],其规则基为频繁闭情节与生成子.算法采用最小且非重

收稿日期:2013-12-13;修回日期:2014-08-07;责任编辑:马兰英
基金项目:国家自然科学基金(No.61303225);航空科学基金(No.20135553034);中央高校基本科研业务费专项资金(No.3102014JSJ0008)

叠发生的支持度定义和深度优先的搜索策略来发现频繁闭情节及其生成子;直接由频繁闭情节及其生成子产生情节规则.Extractor 算法的规则基——闭情节及其生成子已被证明可产生具有最小前件和最大后件的无冗余情节规则[10,11].

表 1 典型情节规则挖掘算法分类比较

类别	规则基	产生规则
1	频繁情节	情节规则全集
2	频繁情节与生成子	最小前件情节规则全集
3	频繁闭情节与生成子	无冗余情节规则集

从上述情节规则挖掘算法的发展不难看出,规则的产生方式经历了频繁情节投影、频繁情节及其生成子投影、频繁闭情节及其生成子投影等阶段;算法的效率、精确程度、精简粒度都在逐步提高;都保持了情节规则的完备性,属于无损规则挖掘方法.

近年来,伴随不同应用的需求变更,诞生了许多对规则集的有损挖掘方法.文献[12]提出了挖掘 top-K 无冗余情节规则的方法,算法根据规则的支持度大小,选择前 K 个情节规则.算法可保证规则代表信息的有用性,达到对规则进行约减的目的,但却损失了支持度较小的情节规则信息.文献[13]提出信息情节规则的概念,在给定预测目标的前提下,采用模式倒置树生成对应预测目标的信息情节规则,保证了参与预测的情节规则数量最少.这样虽然不会对预测结果产生影响,但是得到的情节规则未包含完整的情节规则集信息.文献[14]为了使挖掘的情节规则更具预测性,提出了严格支持度阈值的概念,同时在挖掘过程中不断调整置信度.虽然挖掘到的情节规则蕴含较高的预测信息,但却损失了较低的支持度与置信度的情节规则,在某些情况下会影响到最终预测的效果.

综上所述,现有的情节规则挖掘算法存在以下不足:(1)无损规则挖掘方法仅考虑了两规则间的包含冗余,忽略了多规则间的关联关系.以当前最优的 Extractor 算法为例,频繁闭情节与生成子的投影只能避免规则间的包含冗余,其挖掘结果仍是存在冗余的;(2)有损规则挖掘方法虽然考虑了多规则间的关联关系(如 top-K 利用多规则间排序关系、信息情节规则利用多规则的表达关系等),但得到的是不完备规则集.

能否得到一种利用多规则间关联关系的无损情节规则挖掘方法?众所周知,频繁情节的挖掘是在频繁项挖掘的基础上发展而来的;基于频繁闭情节及其生成子产生无冗余情节规则的方法也是从频繁闭项集及其生成子产生精简关联规则的思想发展而来的.延续这一思路,从比频繁闭项集更加精简的项集挖掘过程(这些项集挖掘考虑了多项集间的关联)中是否可以发现更加精简的情节规则挖掘方法?这正是本文研究的主要动机.

非可导频繁项集[15]是在 2002 年根据多项集间的演绎关系提出来的,文献[15]将非可导项集应用到关联规则挖掘上,取得了良好的效果.但一方面该算法没有考虑演绎推导对规则集的约简作用,另一方面它针对的是关联规则挖掘而非情节规则挖掘.

为此,我们利用演绎推导特性对情节规则间的关系进行建模,引入了无冗余情节迹规则的概念,提出了广义无冗余情节规则,并在闭情节及其生成子挖掘的基础上给出了广义无冗余情节规则抽取方法(Generalized Non-Redundant Miner,GNRMiner).

2 广义无冗余情节规则抽取方法

2.1 相关定义[7]

定义 1 (事件,事件序列).事件是给定事件类型集 $\varepsilon=\{E_1,E_2,\cdots,E_n\}$ 中的事件 E 和事件发生时间 t 的二元组 (E,t).事件序列是由若干 ε 中的事件按发生时间先后排列的序列,表示为 $ES=<(E_1,t_1),(E_2,t_2),\cdots,(E_s,t_s)>$.

定义 2 (情节).一个情节是由若干事件组成的序列,表示为 $\alpha=<(E_1,t_1),(E_2,t_2),\cdots,(E_k,t_k)>$,简记为 $\alpha=<E_1E_2\cdots E_k>$.

定义 3 (串接,投影).给定情节 $\alpha=<E_1E_2\cdots E_m>$ 和 $\beta=<E'_1E'_2\cdots E'_k>$,则 $<E_1E_2\cdots E_mE'_1E'_2\cdots E'_k>$ 称为 α 和 β 的串接,记为 $concat(\alpha,\beta)$.设 $\beta\subseteq\alpha$,j 是 β 在 α 中首次出现的结束位置,则从 α 中删除第 1 至第 j 个事件后剩余的情节称为 β 在 α 上的投影,记为 $project(\alpha,\beta)$.

定义 4 (发生).给定事件序列 ES 和情节 $\alpha=<E_1E_2\cdots E_k>$,若 ES 在时间区间 $[t_1,t_k]$ 上按 α 的事件排列顺序出现了 α 所代表的所有事件,则称 ES 上发生了情节 α,时间区间 $[t_1,t_k]$ 称为 α 在 ES 上的一次发生.

定义 5 (支持度).情节 α 在事件序列 ES 上所有发生的数目称为 α 的支持度,记为 $\alpha.sup$.

定义 6 (频繁情节,频繁闭情节,情节生成子).给定支持度阈值 min_sup,若情节 α 的支持度大于等于 min_sup,则 α 是一个频繁情节.若情节 α 是频繁的,且 α 的支持度不等于 α 的任何一个真超情节的支持度,则 α 是一个频繁闭情节.设 f 是一个闭情节,$g\subseteq f$,若 g 的支持度等于 f 的支持度,且 g 不存在与其支持度相同的任何一个真子情节,则 g 称为闭情节 f 的一个情节生成子.

定义 7 (情节规则).一个情节规则 γ 是一个五元组 (l,r,s,c,ω).其中 γ 的前件、后件、支持度、置信度和窗口宽度分别记为 l、r、s、c、ω.

定义 8　(无冗余情节规则[16]).给定情节规则 $\gamma(l,r,s,c,w)$,若不存在情节规则 $\gamma'(l,r,s,c,w)$,使得 $\gamma'.s=\gamma.s,\gamma'.c\subseteq\gamma.c,\gamma'.l\subseteq\gamma.l,\gamma'.r\supseteq\gamma.r$,则称 γ 是一个无冗余情节规则,否则是一个冗余情节规则.

2.2　无冗余情节迹规则

由定义 8 可知,无冗余情节规则给出了两情节规则间的包含关系,相关算法据此对规则集合进行约简.下面我们应用演绎推理给出多情节规则间的演绎关系.

定理 1　规则演绎.如果规则 γ、γ'、γ''之间,满足(1) $\gamma.l=\gamma'.l$ 或者 $\gamma.l=concat(\gamma'.l,\gamma'.r)$;(2) $concat(\gamma.l,\gamma.r)=\gamma''.l$ 或者 $concat(\gamma.l,\gamma.r)=concat(\gamma''.l,\gamma''.r)$,则 $\gamma'\cap\gamma''\Rightarrow\gamma$.

证明　$\gamma'.l$、$concat(\gamma'.l,\gamma'.r)$的支持度可以由规则$\gamma'$得到,而 $\gamma.l$ 满足条件(1),即 $\gamma.l$ 的支持度可知;$\gamma''.l$、$concat(\gamma''.l,\gamma''.r)$的支持度可以由规则$\gamma''$得到,而 $concat(\gamma.l,\gamma.r)$满足条件(2),即 $concat(\gamma.l,\gamma.r)$的支持度可知;进而可以计算规则 γ 的支持度、置信度,确定其窗口宽度.因此 $\gamma'n\gamma''\Rightarrow\gamma$.定理 1 得证.

由规则间的演绎定理,我们可以得到无冗余情节迹规则的概念描述如下.

定义 9　(无冗余情节迹规则).给定情节规则 $\gamma(l,r,s,c,w)$,若不存在情节规则 $\gamma'(l,r,s,c,w)$、$\gamma''(l,r,s,c,w)$,使得(1) $\gamma.l=\gamma'.l$ 或者 $\gamma.l=concat(\gamma'.l,\gamma'.r)$;(2) $concat(\gamma.l,\gamma.r)=\gamma''.l$ 或者 $concat(\gamma.l,\gamma.r)=concat(\gamma''.l,\gamma''.r)$同时成立.即 γ 不可以从其他规则的产生轨迹上推导得出,则称 γ 是一个无冗余情节迹规则,否则是一个冗余情节迹规则.

定义 10　(广义无冗余情节规则).满足无冗余情节特征和无冗余情节迹特征的规则称为广义无冗余情节规则.

文献[11]依据频繁闭情节和生成子,按照生成子在闭情节投影的方式,产生了无冗余情节规则,但这些规则中蕴含了冗余的情节迹规则.如何在无冗余情节规则中过滤掉冗余的情节迹规则呢?直观的看,可以依据定义通过事后检查的方式进行过滤.但是这种事后过滤的方法需要遍历整个规则集合,增加了处理的时间.下面我们从规则产生的过程中,分析冗余情节迹规则的产生原因,为过滤算法的研究提供依据.

定理 2　生成子向频繁闭情节投影时,对互相重叠的生成子和闭情节,他们既可作为闭情节被其他生成子投影,又可作为生成子向其他闭情节投影,这造成了冗余情节迹规则.

证明　反证法.设冗余情节迹规则不是由重叠生成子和闭情节的多重投影造成,即规则 γ、γ'、γ''之间满足 $\gamma'\cap\gamma''\Rightarrow\gamma$,则(1) $\gamma.l=\gamma'.l$ 或者 $\gamma.l=concat(\gamma'.l,\gamma'.r)$;(2) $concat(\gamma.l,\gamma.r)=\gamma''.l$ 或者 $concat(\gamma.l,\gamma.r)=concat(\gamma''.l,\gamma''.r)$.

分四种情况讨论:

(1) $\gamma.l=concat(\gamma'.l,\gamma'.r)$且 $concat(\gamma.l,\gamma.r)=concat(\gamma''.l,\gamma''.r)$,此时,$\gamma'$的产生是由于 $\gamma.l$ 被 $\gamma'.l$ 投影,γ 的产生是由于 $\gamma.l$ 的在 $concat(\gamma''.l,\gamma''.r)$上投影.即 $\gamma.l$ 既是生成子又是闭情节,充当了投影和被投影的角色,这与假设矛盾,情况 1 得证.

同理可以证明其他三种情况,定理 2 得证.

2.3　GNRMiner 算法流程

依据定理 2,生成子和闭情节的重叠集既可作为闭情节被其他生成子投影,又可作为生成子向其他闭情节投影,这造成了冗余情节迹规则.因此在生成规则的过程中,通过检查、过滤机制就可以有效避免冗余情节迹规则的产生.但即便如此,由于投影关系的传递性,进行情节迹规则的过滤仍然是复杂的.为了提高过滤效率,我们给出如下定理.

定理 3　生成子向频繁闭情节投影时,对互相重叠的生成子和闭情节,重叠集内部会存在互相包含关系,这些关系中,只需考虑最大的重叠项进行情节迹冗余检查即可.

证明　设生成子 g_0、g_1、g_2、g_1、g_2 是重叠集中的元素,并且 g_0 可以投影到 g_1,g_1 可以投影到 g_2.由于投影规则是可传递的,g_0 也可以投影到 g_2,记为 $g_0-g_1-g_2$.设有闭情节 e,满足 g_2-e,则有 $g_0-g_1-g_2-e$.根据定理 1,对于 g_0-g_1-e 而言,g_0-e、g_1-e 两条规则蕴含 g_0-g_1;对于 g_0-g_2-e 而言,g_0-e、g_2-e 两条规则蕴含 g_0-g_2;对于 g_1-g_2-e 而言,g_1-e、g_2-e 两条规则蕴含 g_1-g_2.可以看出,由 g_0-g_1-e 所产生的规则 g_0-e、g_1-e 完全包含在 g_1-g_2-e、g_0-g_2-e 所产生的规则中,而 g_2 为重叠集中较大的元素.以此类推,可证明在进行情节迹冗余检查时,只需要检查重叠集最大元素的投影和被投影情况即可.得证.

依据定理 3,基于频繁闭情节及其生成子的广义无冗余规则抽取流程如算法 1 所示.其中,1 ~ 8 步为找出重叠集最大的元素;9 ~ 12 步为找出最大元素的所有投影和被投影元素;13 ~ 22 步为冗余情节迹规则的过滤过程,即在最大重叠元素的一次投影和被投影过程中,最多只产生两个规则;23 ~ 29 步为生成子和闭情节无重叠情况的规则产生.可见,算法依据重叠集的最大元素过滤掉了冗余情节迹规则,而算法本身的投影过程产生的就是无冗余规则[16],最终产生了广义无冗余情节规则.

算法 1　*ruleExtractor* (*Set ee*, *Set ge*)

输入:*ee* 为蕴含情节生成子的频繁闭情节集合,*ge* 为对应的生成子集合

输出:*result* 为结果规则集合

```
1. result = empty
2. Find ge′ in ge which ge′ has the same item in ee
3. For each g1 and g2 in ge′
4.    If g1 can project to g2
5.       Delete g1 in ge′
6.    If g2 can project to g1
7.       Delete g2 in ge′
8. Delete ge′ in ge//找到闭情节和生成子的公共集合 ge 并去除 ge 中的重叠元素
9. Find gee in ge which gee can project to ge′//找出 ge 中的可被投影集合
10. Delete gee in ge
11. Find ee′ in ee which ge′ can project to ee//找出 ee 中的可被 ge 中元素投影集合
12. Delete ee′ in ee
13. For each g1 in gee and g2 in ge′ and e1 in ee′
14.    If g1 can project to g2 and g2 can project to e1
15.       r = project(g1, g2)
16.       a = contact(g1, r)
17.       If a.sup/g.sup >= minconf
18.          result.add(g1, r, a.sup, a.sup/g1.sup, a.w)
19.       r = project(g2, e1)
20.       a = contact(g2, r)
21.       If a.sup/g.sup >= minconf
22.          result.add(g2, r, a.sup, a.sup/g2.sup, a.w)
23. For each f in ee and g in ge
24.    If g can project to f
25.       r = project(g, f)
26.       a = contact(g, r)
27.       If a.sup/g.sup >= minconf
28.          result.add(g, r, a.sup, a.sup/g.sup, a.w)
29. Return result
```

定理 4 广义无冗余情节规则蕴含了所有情节规则的信息.

证明 广义无冗余情节规则是在无冗余情节规则基础之上过滤而来的.因此证明分为两步,首先证明广义无冗余情节规则覆盖了无冗余情节规则,再证明无冗余情节规则包含了所有情节规则的信息.

广义无冗余情节规则覆盖了无冗余情节规则.反证法:假设广义无冗余情节规则没有覆盖所有无冗余情节规则,即假设 γ 为一无冗余情节规则,在构建广义无冗余情节规则集时被删除,且不可由广义无冗余情节规则集推导.按照广义无冗余情节规则的情节迹检测过程可知,γ 要么可被推导、被删除,要么不可被推导、保留.这与假设矛盾,假设不成立,得证.

无冗余情节规则包含了所有情节规则的信息.该证明在文献[11]中有详细说明,在此不再赘述.

综上,定理 4 得证.

图 1 给出了各类情节规则间的关联关系,以事件序列上蕴含的所有情节规则为全集,无冗余情节规则和无冗余情节迹规则都是它的子集.其中无冗余情节规则对同置信度、支持度规则间的互相包含关系进行了化简;无冗余情节迹规则对多规则间的推导关系进行了约简.对无冗余情节规则进行进一步的情节迹约简后即可得到广义无冗余情节规则(图中方格部分区域).

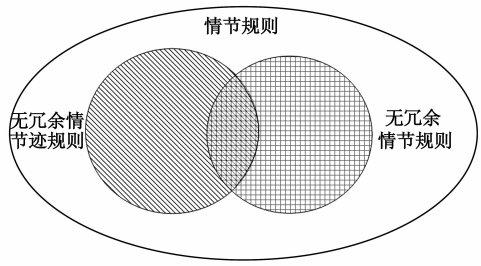


图1 各类情节规则关联关系示意图

2.4 算法性能分析

设 L 为事件序列 ES 的长度,ε 为 ES 中的事件类型集,FE 为 ES 上所有频繁情节组成的集合,则算法 GNRMiner 的复杂度分析如下:

GNRMiner 算法的时间复杂度:算法首先需要找到生成子集中具有和 FE 中相同元素的且之间不含前缀包含关系的生成子集 gee,其复杂度为 $O(|FE| * |FE|)$,然后需要在 FE 中分别找到 FE 可以对 gee 中元素进行投影的元素集合以及 gee 可以对 FE 中元素进行投影的元素集合,其复杂度皆为 $O(|FE| * |FE|)$.此外,规则产生时需要将情节生成子一一投影到所有的频繁闭情节中进行规则产生,所以规则产生过程所需的时间复杂度为 $O(|FE| * |FE|)$.

GNRMiner 算法的空间复杂度:由于需要维护所有的规则信息,最坏情况下空间复杂度为 $O(|FE| * |FE|)$.因此空间复杂度为 $O(|FE| * |FE|)$.

表 2 GNRMiner 与 WinMiner * 和 Extractor * 规则抽取阶段时间复杂度比较

算法	时间复杂度
WinMiner 规则抽取[9]	$O(\|FE\| * \|FE\|)$
Extractor 规则抽取[11]	$O(\|FE\| * \|FE\|)$
GNRMiner	$O(\|FE\| * \|FE\|)$

表 2 为 GNRMiner 与 WinMiner 规则抽取阶段算法、Extractor 规则抽取阶段算法(分别记为 WinMiner * 和 Extractor *)的时间复杂度比较.可见在最坏情况下,三个算法的时间复杂度均为 $O(|FE| * |FE|)$.

3 实验

我们通过实验对比了算法 GNRMiner、Extractor * 和

WinMiner * 处理效果和速度.为了保障对比的公平性,首先保障 WinMiner * 已完成频繁情节挖掘、GNRMiner 和 Extractor * 已完成闭情节及生成子挖掘,其次各算法均采取标准的发生作为情节支持度的定义.实验所在计算机的配置为 CPU INTEL E8500 2.93GHz,RAM 4GB,Windows XP Professional,程序采用 C + + 实现.

3.1　数据集

实验采用两种类型的数据集:(1)合成数据集,使用 IBM 合成数据生成器[17]生成数据序列,通过设置序列个数为 1,事件类型数为 2000,产生 30000 个事件组成的事件序列;(2)公交数据集,选取浦东地区 400 条公交线路车辆运行信息,每个事件包含车辆编号、线路编号、时间戳、位置、速度等属性.本文将速度属性值分为 10 个区间形成了 4000 个事件类型.选取了 2013 年 5 月 8 日至 14 日的传感数据组成事件序列.

3.2　实验结果

实验 1　规则个数与置信度阈值的关系.在设定两个数据集的支持度阈值分别为 160 和 10 的情况下,通过改变置信度阈值,得到了如图 2 所示的算法在两个数据集上发现的情节规则个数.从图中可以看出,随着置信度阈值的减少,3 个算法均发现了更多的情节规则,这是由于置信度阈值越小,将会有更多的规则满足阈值约束.同时,GNRMiner 发现的规则个数少于 Extractor *,Extractor * 又少于 WinMiner *.这是因为 GNRMiner 产生的是最为精简的广义无冗余情节规则,Extractor * 产生的是无冗余情节规则,WinMiner * 产生的是所有规则.当置信度为 30% 时,存在的规则迹冗余较大,平均下来 GNRMiner 得到的规则与 Extractor * 得到的规则相比,下降了近 70%.当置信度为 70% 时,存在的规则迹冗余小,平均下来 GNRMiner 得到的规则与 Extractor * 得到的规则相比,下降了 30% 左右.

实验 2　规则个数与支持度阈值的关系,在两个数据集置信度阈值设置为 60% 的情况下,通过改变支持度阈值,得到如图 3 所示的 3 个算法在两数据集上发现的情节规则个数.可以看出,随着支持度阈值的减少,3 个算法均发现了更多的情节规则,同时可以发现 GNRMiner 发现的规则个数少于 Extractor *,Extractor * 又小于 WinMiner *.原因同实验 1.

实验 3　规则个数与序列长度的关系.通过设定两数据集的支持度阈值分别为 160 和 10,置信度阈值均为 60%,得到了如图 4 所示的序列长度对算法所发现的情节规则个数影响.从图中可以看出,随着序列长度的增加(合成数据从 10k 到 30k,公交数据从 1 天到 5 天),3 个算法均发现了更多的情节规则,但 GNRMiner 发现的规则个数少于 Extractor *,Extractor * 又小于 WinMiner *.原因同实验 1.

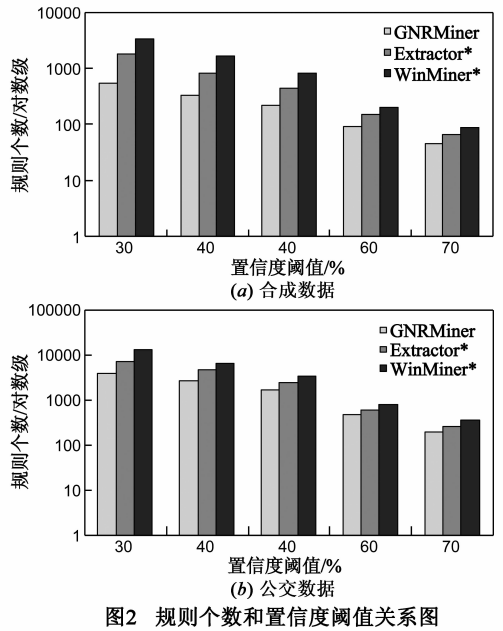


图2　规则个数和置信度阈值关系图

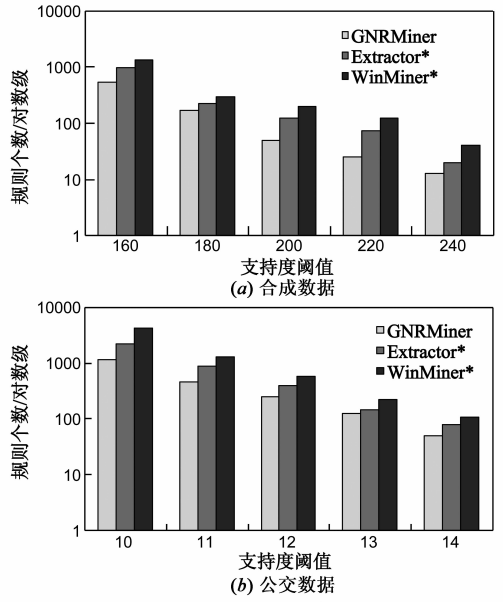
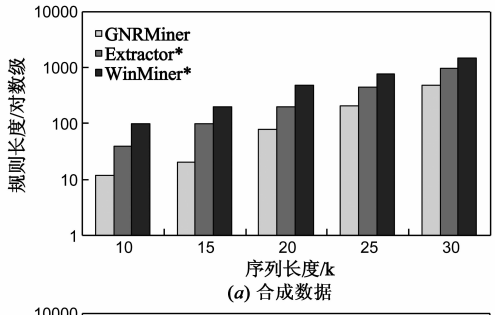
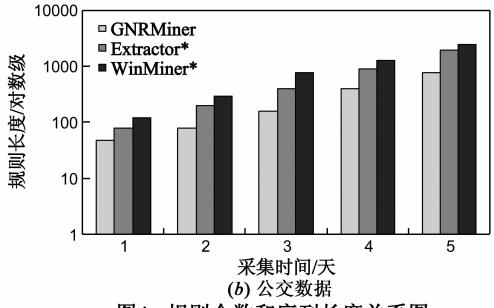


图3　规则个数和支持度阈值关系图

实验 4　运行时间与置信度阈值的关系.在设定两数据集的支持度阈值为 160 和 10 的前提下,通过改变置信度的阈值,得到如图 5 所示的 3 个算法在两个数据集上的运行时间(单位 ms).从图中可以看出,随着支持度阈值的减少,3 个算法的运行时间都线性增加,但是 Extractor * 和 GNRMiner 要优于 WinMiner *.这是由于他们都是基于闭情节和生成子完成规则生成的;而 WinMiner * 需要基于频繁情节产生规则,要处理的情节数量多.GNRMiner 执行时间较 Extractor * 略高,这是由于两算法规则抽取的时间复杂度均为 $O(|FE| * |FE|)$,



(*a*) 合成数据

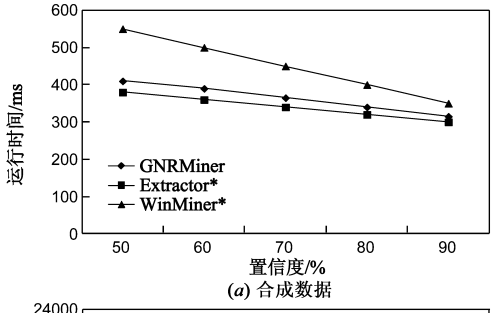


(*b*) 公交数据

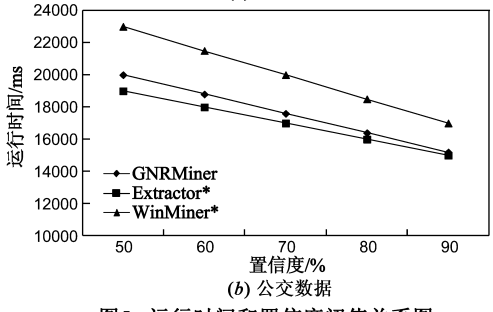
图4 规则个数和序列长度关系图

但 GNRMiner 还需要进行情节迹检查.

运行时间与支持度阈值、序列长度的关系与实验 4 类似,这里就不再赘述.与实验 1 联合分析,GNRMiner 虽然在处理时间上略有增加,但却将规则个数减少了许多.



(*a*) 合成数据



(*b*) 公交数据

图5 运行时间和置信度阈值关系图

4 结论

针对现有无损情节规则挖掘方法没有考虑多规则间的关联关系因素,故而存在大量冗余的问题,本文引入了无冗余情节迹规则的概念,提出了广义无冗余情节规则,并在闭情节及其生成子挖掘的基础上给出了广义无冗余情节规则抽取方法.理论分析和实验评估表明该算法在保障规则完备性的前提下,得到了更为精简的情节规则.当然,情节规则挖掘只是情节预测的第一步,后续工作我们将研究基于广义无冗余情节规则的情节预测方法.

参考文献

[1] Mannila H, Toivonen H, Verkamo A I. Discovering frequent episodesin sequences extended abstract[A]. 1st Conference on Knowledge Discovery and Data Mining[C]. Montreal: CA. 1995.210 – 215.

[2] Koh Y S, Pears R. Efficient negative association rule mining based on chance thresholds[J]. Intelligent Data Analysis, 2014, 18(2):243 – 260.

[3] Lam H T, Calders T, Yang J, et al. Zips: mining compressing sequential patterns in streams[A]. Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics[C]. New York : ACM, 2013.54 – 62.

[4] Van Der Aalst W. Process mining: Overview and opportunities [J]. ACM Transactions on Management Information Systems, 2012, 3(2):1 – 17.

[5] Laxman S, Tankasali V, White R W. Stream prediction using a generative model based on frequent episodes in event sequences [A]. Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining[C]. Las Vegas, NV: ACM, 2008.453 – 461.

[6] Cho C W, Zheng Y, Chen A L P. Continuously Matching Episode Rules for Predicting Future Events over Event Streams [M]. Advances in Data and Web Management. 2007. 884 – 891.

[7] Cho C W, Wu Y H, Yen S J, et al. On-line rule matching for event prediction[J]. The VLDB Journal, 2011, 20(3): 303 – 334.

[8] Hatonen K, Klemettinen M, Mannila H, et al. Knowledge discovery from telecommunication network alarm databases[A]. Proceedings of the Twelfth IEEE International Conference[C]. Trois-Rivières: IEEE, 1996.115 – 122.

[9] Méger N, Rigotti C. Constraint-based mining of episode rules and optimal window sizes [A]. Knowledge Discovery in Databases: PKDD 2004[C]. Berlin Heidelberg: PKDD, 2004. 313 – 324.

[10] Lo D, Khoo S C, Li J. Mining and ranking generators of sequential patterns[A]. SIAM International Conference on Data Mining[C]. Atlanta Georgia: SIAM, 2008.553 – 564.

[11] 朱辉生,汪卫,施伯乐.基于频繁闭情节及其生成子的无冗余情节规则抽取[J].计算机学报,2012,35(1):53 – 64.

Zhu Huisheng, Wang wei, Shi Bole. Extracting non-redundant

episode rules based onfrequent closed episodes and their generators[J]. Journal of Computers, 2012, 35(1): 53 – 64. (in Chinese)

[12] Fournier-Viger P, Tseng V S. Tns: mining top-k non-redundantsequential rules[A]. Proceedings of the 28th Annual ACM Symposium on Applied Computing[C]. Coimbra, Portugal: ACM, 2013. 164 – 166.

[13] Zhang S, Wu X. Fundamentals of association rules in data mining and knowledge discovery[J]. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2011, 1(2): 97 – 116.

[14] Rudin C, Letham B, Madigan D. Learning theory analysis for association rules and sequential event prediction[J]. The Journal of Machine Learning Research, 2013, 14(1): 3441 – 3492.

[15] Toon Calders, Bart Goethals. Non-derivable itemset mining [J]. Data Mining and Knowledge Discovery, 2007, 14(1): 171 – 206.

[16] Ao F, Du J, Yan Y, et al. An efficient algorithm for mining closed frequent itemsets in data Streams[A]. Computer and Information Technology Workshops of IEEE 8th International Conference 2008[C]. Aizu-Wakamatsu: IEEE, 2008. 37 – 42.

[17] Agrawal R, Srikant R. Fast algorithms for mining association rules[A]. Proceedings of 20th International Conference on Very Large Data Bases[C]. Santiago de Chile: ACM, 1994. 487 – 499.

作者简介

尤　涛　男,1983 年生于河南三门峡.西北工业大学计算机学院讲师,研究方向为分布式数据流处理.

E-mail: youtao@nwpu.edu.cn

徐　伟　男,1989 年生于江苏南京.西北工业大学计算机学院硕士研究生,研究方向为数据流处理.