软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2020,31(7):2169–2183 [doi: 10.13328/j.cnki.jos.005851]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

融合多种支持度定义的频繁情节挖掘算法*

朱辉生[1], 陈 琳[2], 倪艺洋[1], 汪 卫[3], 施伯乐[3]

[1](江苏第二师范学院 数学与信息技术学院,江苏 南京 211200)

[2](泰州学院 计算机科学与技术学院,江苏 泰州 225300)

[3](复旦大学 计算机科学技术学院,上海 200433)

通讯作者: 朱辉生, E-mail: zhs@fudan.edu.cn

摘 要: 事件序列中蕴藏的频繁情节刻画了用户或系统的行为规律.现有的频繁情节挖掘算法在各自支持度定义下具有较好的挖掘效果,但在支持度定义发生变化时却很难甚至无法直接挖掘频繁情节.针对用户多变的支持度定义需求,提出了一种频繁情节挖掘算法 FEM-DFS(frequent episode mining-depth first search).该算法通过单遍扫描事件序列,以深度优先搜索方式来发现频繁情节,以共享前/后缀树来存储频繁情节,以单调性、前缀单调性或后缀单调性来压缩频繁情节的搜索空间.实验评估证实了所提出算法的有效性.

关键词: 事件序列;频繁情节;挖掘;支持度;深度优先遍历

中图法分类号: TP311

中文引用格式: 朱辉生,陈琳,倪艺洋,汪卫,施伯乐.融合多种支持度定义的频繁情节挖掘算法.软件学报,2020,31(7):2169–2183. http://www.jos.org.cn/1000-9825/5851.htm

英文引用格式: Zhu HS, Chen L, Ni YY, Wang W, Shi BL. Frequent episode mining algorithm compatible with various support definitions. Ruan Jian Xue Bao/Journal of Software, 2020,31(7):2169–2183 (in Chinese). http://www.jos.org.cn/1000-9825/5851.htm

Frequent Episode Mining Algorithm Compatible with Various Support Definitions

ZHU Hui-Sheng[1], CHEN Lin[2], NI Yi-Yang[1], WANG Wei[3], SHI Bai-Le[3]

[1](School of Mathematics and InformationTechnology, Jiangsu Second Normal University, Nanjing 211200, China)

[2](School of Computer Science and Technology, Taizhou University, Taizhou 225300, China)

[3](School of Computer Science, Fudan University, Shanghai 200433, China)

Abstract: Frequent episodes hidden in an event sequence describe the behavioral regularities of users or systems. Existing algorithms yield good results for mining frequent episodes under their respective definitions of support, but each of them is difficult or impossible to directly mine frequent episodes when the definition of support is changed. To meet the needs of changeable support definitions of users, an algorithm called FEM-DFS (frequent episode mining-depth first search) is proposed to mine frequent episodes in this paper. After scanning the event sequence one pass, FEM-DFS finds frequent episodes in a depth first search fashion, stores frequent episodes in a shared prefix/suffix tree and compresses the search space of frequent episodes by utilizing monotonicity, prefix monotonicity or suffix monotonicity. Experimental evaluation demonstrates the effectiveness of the proposed algorithm.

Key words: event sequence; frequent episode; mining; support; depth first search

* 基金项目: 国家自然科学基金(61802274, 61701201, U1509213); 教育部“云数融合科教创新”基金(2017B06109); 江苏省自然科学基金(BK20141307, BK20170758); 江苏省“333 工程”基金(BRA2015212); 江苏省无线通信重点实验室开放研究基金(2017WICOM02)

Foundation item: National Natural Science Foundation of China (61802274, 61701201, U1509213); “Integration of Cloud Computing and Big Data, Innovation of Science and Education” Foundation of Ministry of Education of China (2017B06109); Natural Science Foundation of Jiangsu Province of China (BK20141307, BK20170758); “333 Engineering” Foundation of Jiangsu Province of China (BRA2015212); Open Project Foundation of Key Laboratory of Wireless Communications of Jiangsu Province of China (2017WICOM02)

收稿时间: 2018-03-27; 修改时间: 2018-12-28; 采用时间: 2019-04-19

随着物联网、云计算和大数据等新一代信息技术的发展,在数字图书、城市交通、工业制造、通信服务、网络安全、金融证券、生命科学、医疗健康等众多领域产生了由若干值对(事件类型,发生时间)组成的事件序列.事件序列中蕴藏着丰富的语义模式,作为最常见的语义模式,频繁情节揭示了事件序列中用户或系统的行为规律,因此,频繁情节挖掘有着广阔的应用前景,典型场景如文献推荐阅读、城市智慧交通、工业智能制造、通信故障诊断、网络入侵检测、股市态势预测、基因序列分析、疾病预防控制等.

自 Manilla 等人[1]引入事件序列上的频繁情节挖掘问题以来,众多学者对此展开了研究,提出了许多具有代表性的频繁情节挖掘算法,见表 1.

Table 1 Existing algorithms for frequent episode discovery

表 1 现有的频繁情节挖掘算法

支持度定义	算法	单调性
窗口发生	WINEPI[1]、WinMiner[2]、HUEM-GAO[3]	满足
最小发生	MINEPI[1]、EPT[4]、PPS[4]、Clo_episode[5]、Ap-epi[6]、UP-Span[7]、DMinEpi[8]、MELLO[9]	不满足
头发生	MINEPI+[10]、EMMA[10]	不满足
总发生	T-freq[11]、DiscoveryTotal[12]	满足
非交错发生	Non-interleaved[13]	不满足
非重叠发生	Non-WinMiner[6]、DiscoveryNonOver[12]、NONOVERLAPPING[14]、Non-overlapped[15]	满足
最小且非重叠发生	MANEPI[16]、FCEMiner[17]、2PEM[18]	满足

从表 1 可以看出,支持度定义是挖掘频繁情节时必须考虑的一个尺度.一方面,支持度定义决定着挖掘的结果,针对同一事件序列根据不同支持度定义挖掘的频繁情节不尽相同;另一方面,支持度定义影响着挖掘的过程,有些支持度定义满足单调性,有些则不然,而单调性是加快频繁情节搜索的重要依据.

上述算法在各自支持度定义下具有较好的挖掘效果,但在支持度定义发生变化时却很难甚至无法直接挖掘频繁情节.为此,Achar 等人[19]提出了基于 Apriori 思想逐层发现频繁情节的统一算法.该算法采用广度优先搜索策略,首先由 $k(k>0)$层的频繁情节产生 $k+1$ 层的候选频繁情节,然后通过扫描事件序列来跟踪各个候选频繁情节状态机的状态变化,跟踪过程由参数 TRANSIT、COPY、JOIN、INCREMENT 和 RETIRE 来分别控制当前状态机是否发生状态转移、当前状态机转移至下一状态前是否要复制一个副本、两个状态机转移至同一状态时是否要删除较早的状态机、当前状态机转移至终止状态时情节发生次数是否增 1、情节发生次数增 1 后是否删除该情节的所有状态机,从而计算每个候选频繁情节在不同支持度定义下的发生次数,进而发现 $k+1$ 层的频繁情节.该算法虽然兼顾了多种支持度定义,但挖掘过程中需要多遍扫描事件序列,且每遍扫描前都要存储大量的候选频繁情节,这势必导致较为昂贵的时间和空间代价.

如何在单遍扫描事件序列和不产生候选频繁情节的前提下,融合多种支持度定义来挖掘频繁情节,即为本文的研究动机.

1 预备知识

1.1 基本概念

定义 1(事件,事件序列). 给定事件类型集$\varepsilon=\{E_1,E_2,\ldots,E_n\}$,一个事件就是一个二元组$(E,t)$,其中,$E\in\varepsilon$,$t$ 表示该事件的发生时间.定义在ε上的一个事件序列 ES 是按发生时间先后排列的若干事件,表示为 $ES=\langle(E_1,t_1),(E_2,t_2),\ldots,(E_m,t_m)\rangle$,其中,$t_i<t_j(1\leqslant i<j\leqslant m)$.为讨论方便,本文假设在每个时间点上至多只发生一个事件.

例如,$ES1=\langle(A,1),(A,2),(A,4),(B,5),(A,6),(A,7),(C,8),(B,9),(D,11),(C,12),(A,13),(B,14),(C,15),(D,16),(A,17)\rangle$ 就是一个事件序列.如图 1 所示.

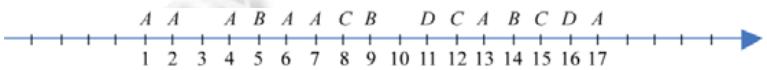


Fig.1 A sequence of events

图 1 事件序列示例

定义 2(情节,子情节). 情节α是ε中由若干事件类型组成的序列,记为$\alpha=\langle E_1E_2\ldots E_k\rangle$,其中,$E_i(1\leqslant i\leqslant k)\in\varepsilon$且对于所有的 i 和 $j(1\leqslant i<j\leqslant k)$满足 E_i 排列在 E_j 之前.情节α中事件类型的个数称为α的长度(记为$|\alpha|$),长度为 k 的情节称为 k-情节.若情节β中的事件类型均来自情节α,且与α中这些事件类型的先后顺序一致,则称β是α的子情节,记作$\beta\sqsubseteq\alpha$.

例如,$\langle ABC\rangle$是一个 3-情节,$\langle AB\rangle$是$\langle ABC\rangle$的子情节,而$\langle BA\rangle$不是.

定义 3(窗口). 给定事件序列 $ES=\langle(E_1,t_1),(E_2,t_2),\ldots,(E_m,t_m)\rangle$,设 $t_1\leqslant t_s\leqslant t_e\leqslant t_m$,则$[t_s,t_e]$是 ES 上的一个窗口,该窗口包含了 t_s 到 t_e 的所有事件,t_e-t_s 称为该窗口的宽度,t_s 和 t_e 分别称为该窗口的起始和终止时间.

例如,[2,8]是事件序列 $ES1$ 上一个宽度为 6 的窗口,其起始时间为 2,终止时间为 8.

定义 4(前缀,后缀,串接). 给定情节$\alpha=\langle E_1E_2\ldots E_k\rangle$,则$\langle E_1E_2\ldots E_{k-1}\rangle$称为$\alpha$的前缀,记为 $prefix(\alpha)$;$\langle E_2\ldots E_k\rangle$称为$\alpha$的后缀,记为 $suffix(\alpha)$.给定情节 $\alpha=\langle E_1E_2\ldots E_n\rangle$和 $\beta=\langle E'_1E'_2\ldots E'_m\rangle$, 则 $\langle E_1E_2\ldots E_nE'_1E'_2\ldots E'_m\rangle$ 称为 α 和β的串接,记为 $concat(\alpha,\beta)$.

例如,$\langle AB\rangle$和$\langle BC\rangle$分别是情节$\langle ABC\rangle$的前缀和后缀,$concat(\langle AB\rangle,\langle BC\rangle)=\langle ABBC\rangle$.

定义 5(发生,最早转移发生). 给定事件序列 ES 和情节$\alpha=\langle E_1E_2\ldots E_k\rangle$,若$\langle(E_1,t_1),(E_2,t_2),\ldots,(E_k,t_k)\rangle$是从 ES 中删除若干事件后得到,其中,$t_i<t_{i+1}(1\leqslant i\leqslant k-1)$,则称$[t_1,t_2,\ldots,t_k]$为$\alpha$在 ES 上的一次发生,$[t_1,t_k]$为发生区间,t_k-t_1 为区间长度.若$[t_1,t_2,\ldots,t_k]$是情节$\alpha=\langle E_1E_2\ldots E_k\rangle$在事件序列 ES 上的一次发生,且 $t_i(2\leqslant i\leqslant k)$是继 t_{i-1} 之后事件类型 E_i 的首次发生时间,则称$[t_1,t_2,\ldots,t_k]$是α在 ES 上的一次最早转移发生,情节α在 ES 上所有最早转移发生组成的集合记为 $eto(\alpha)$.

例如,在事件序列 $ES1$ 上,[4,9,12]是情节$\langle ABC\rangle$的一次发生(但不是最早转移发生),该发生区间为[4,12],区间长度为 8,$eto(\langle ABC\rangle)=\{[1,5,8],[2,5,8],[4,5,8],[6,9,12],[7,9,12],[13,14,15]\}$.

性质 1(单调性). 若β是α的任一子情节,则β的发生次数不小于α的发生次数.

性质 2(前缀单调性). $prefix(\alpha)$的发生次数不小于 α 的发生次数.显然,性质 2 是性质 1 的一种情形.

性质 3(后缀单调性). $suffix(\alpha)$的发生次数不小于 α 的发生次数.显然,性质 3 也是性质 1 的一种情形.

定义 6(基于窗口发生的支持度). 给定事件序列 $ES=\langle(E_1,t_1),(E_2,t_2),\ldots,(E_n,t_n)\rangle$和窗口宽度 w,则 ES 上共包含(t_n-t_1+w+1)个宽度为 w 的窗口,其中,第 1 个窗口仅包含事件(E_1,t_1),最后一个窗口仅包含事件(E_n,t_n).宽度为 w 且至少包含情节 α 一次发生的窗口的集合记为 $wo(\alpha)$,集合 $wo(\alpha)$的基数$|wo(\alpha)|$称为情节 α 基于窗口发生的支持度.基于窗口发生的支持度定义满足性质 1.

例如,设窗口宽度 $w=6$,情节 $\alpha=\langle ABC\rangle$,则在事件序列 $ES1$ 上,$wo(\alpha)=\{[2,8],[3,9],[4,10],[6,12],[7,13],[9,15],[10,16],[11,17],[12,18],[13,19]\}$,$|wo(\alpha)|=10$.

定义 7(基于最小发生的支持度). 设$[t_1,t_2,\ldots,t_k]$是情节 $\alpha=\langle E_1E_2\ldots E_k\rangle$在事件序列 ES 上的一次发生,若 ES 上不存在 α 的另一次发生$[t'_1,t'_2,\ldots,t'_k]$,使得 $t_1<t'_1$ 且 $t'_k\leqslant t_k$, 或 $t_1\leqslant t'_1$ 且 $t'_k<t_k$, 则称$[t_1,t_2,\ldots,t_k]$是 α 的一次最小发生.

情节 α 在 ES 上所有最小发生组成的集合记为 $mo(\alpha)$,集合 $mo(\alpha)$的基数$|mo(\alpha)|$称为情节 α 基于最小发生的支持度.

例如,设情节 $\alpha=\langle ABC\rangle$,$\beta=\langle AC\rangle$,则在事件序列 $ES1$ 上 $mo(\alpha)=\{[4,5,8],[7,9,12],[13,14,15]\}$,$|mo(\alpha)|=3$,$mo(\beta)=\{[7,8],[13,15]\}$,$|mo(\beta)|=2$.因 β 是 α 的子情节,且$|mo(\beta)|<|mo(\alpha)|$,所以基于最小发生的支持度定义不满足性质 1,但满足性质 2 和性质 3.

定义 8(基于头发生的支持度). 给定事件序列 $ES=\langle(E_1,t_1),(E_2,t_2),\ldots,(E_n,t_n)\rangle$和窗口宽度 w,若宽度为 w 的窗口$[t_s,t_e]$至少包含了情节 $\alpha=\langle E_1E_2\ldots E_k\rangle$的一次发生,且 t_s 正好是 E_1 的发生时间,则称$[t_s,t_e]$是 α 的一次头发生.情节 α 在 ES 上所有头发生组成的集合记为 $ho(\alpha)$,集合 $ho(\alpha)$的基数$|ho(\alpha)|$称为情节 α 基于头发生的支持度.

例如,设窗口宽度 $w=6$,情节 $\alpha=\langle ABC\rangle$,$\beta=\langle BC\rangle$,则在事件序列 $ES1$ 上,$ho(\alpha)=\{[2,8],[4,10],[6,12],[7,13],[13,19]\}$,$|ho(\alpha)|=5$,$ho(\beta)=\{[5,11],[9,15],[14,20]\}$,$|ho(\beta)|=3$.因为 β 是 α 的后缀,且$|ho(\beta)|<|ho(\alpha)|$,所以基于头发生的支持度定义不满足性质 1 和性质 3,但满足性质 2.

定义 9(基于总发生的支持度). 设 β 是 α 的一个子情节,若不存在 α 的另一个子情节γ,使得$|ho(\gamma)|<|ho(\beta)|$,

则称$|ho(\beta)|$为情节 α 基于总发生的支持度,记为$|to(\alpha)|$.基于总发生的支持度定义满足性质 1.

例如,设窗口宽度 w=6,在事件序列 $ES1$ 上,$ho(\langle AC\rangle)$={[2,8],[4,10],[6,12],[7,13],[13,19]},$ho(\langle AB\rangle)$={[1,7],[2,8],[4,10],[6,12],[7,13],[13,19]},$ho(\langle BC\rangle)$={[5,11],[9,15],[14,20]},所以,$|to(\langle ABC\rangle)|=|ho(\langle BC\rangle)|$=3.

定义 10(基于非交错发生的支持度). 设$[t_1,t_2,\ldots,t_k]$和$[t'_1,t'_2,\ldots,t'_k]$是情节 $\alpha=\langle E_1E_2\ldots E_k\rangle$在事件序列 ES 上的两次发生,若对于所有的 $i(1\leqslant i\leqslant k-1)$,满足 $t_i\geqslant t'_{i+1}$ 或 $t'_1\geqslant t_{i+1}$,则称$[t_1,t_2,\ldots,t_k]$和$[t'_1,t'_2,\ldots,t'_k]$是 α 在 ES 上的非交错发生.情节 α 在 ES 上非交错发生组成的最大集合记为 $ni(\alpha)$,集合 $ni(\alpha)$的基数$|ni(\alpha)|$称为情节α基于非交错发生的支持度.

例如,设情节 $\alpha=\langle ABC\rangle$,$\beta=\langle AC\rangle$,在事件序列 $ES1$ 上 $ni(\alpha)$={[1,5,8],[6,9,12],[13,14,15]},$|ni(\alpha)|$=3,$ni(\beta)$={[1,8],[13,15]},$|ni(\beta)|$=2.因为 β 是α的子情节,且$|ni(\beta)|<|ni(\alpha)|$,所以基于非交错发生的支持度定义不满足性质 1,但满足性质 2 和性质 3.

定义 11(基于非重叠发生的支持度). 设$[t_1,t_2,\ldots,t_k]$和$[t'_1,t'_2,\ldots,t'_k]$是情节 $\alpha=\langle E_1E_2\ldots E_k\rangle$在事件序列 ES 上的两次发生,若 $t_k<t'_1$ 或 $t'_k<t_1$,则称$[t_1,t_2,\ldots,t_k]$和$[t'_1,t'_2,\ldots,t'_k]$是 α 在 ES 上的非重叠发生.情节 α 在 ES 上非重叠发生组成的最大集合记为 $no(\alpha)$,集合 $no(\alpha)$的基数$|no(\alpha)|$称为 α 基于非重叠发生的支持度.基于非重叠发生的支持度定义满足性质 1.

例如,在事件序列 $ES1$ 上,$no(\langle ABC\rangle)$={[1,5,8],[13,14,15]},$|no(ABC\rangle)|$=2.

定义 12(基于最小且非重叠发生的支持度). 设$[t_1,t_2,\ldots,t_k]$和$[t'_1,t'_2,\ldots,t'_k]$是情节 $\alpha=\langle E_1E_2\ldots E_k\rangle$在事件序列 ES 上的两次最小发生,若 $t_k<t'_1$ 或 $t'_k<t_1$,则称$[t_1,t_2,\ldots,t_k]$和$[t'_1,t'_2,\ldots,t'_k]$是 α 在 ES 上的最小且非重叠发生.情节 α 在 ES 上最小且非重叠发生组成的最大集合记为 $mn(\alpha)$,集合 $mn(\alpha)$的基数$|mn(\alpha)|$称为 α 基于最小且非重叠发生的支持度.基于最小且非重叠发生的支持度定义满足性质 1.

例如,在事件序列 $ES1$ 上,$mn(\langle ABC\rangle)$={[4,5,8],[13,14,15]},$|mn(ABC\rangle)|$=2.

定义 13(频繁情节). 给定支持度阈值 min_sup,若情节 α 的支持度不小于 min_sup,则称 α 是一个频繁情节.

1.2 问题描述

给定事件序列 ES、窗口宽度 w、支持度阈值 min_sup 和支持度定义 def_sup,则问题描述为:设计一个能够挖掘 ES 上所有频繁情节的算法,要求:(1) 单遍扫描 ES;(2) 挖掘过程中不产生候选频繁情节;(3) 融合基于窗口发生、最小发生、头发生、总发生、非交错发生、非重叠发生、最小且非重叠发生的支持度定义.

2 频繁情节挖掘算法 FEM-DFS

2.1 算法思想

因为许多频繁情节具有相同的前缀或后缀,所以采用共享前/后缀树来存储发现的频繁情节,可以节省存储空间[4,12,16].共享前/后缀树是一棵有向根树,除根节点外,树中的每个节点是一个三元组($Name$,Occ,$Child$),其中,$Name$ 表示该节点对应的情节,Occ 表示该节点对应情节的发生集,$Child$ 表示该节点的孩子指针集.树中父节点的 $Name$,要么是子节点 $Name$ 的前缀,要么是子节点 $Name$ 的后缀,故取名共享前/后缀树.

为了在频繁情节发现过程中只扫描事件序列 1 遍,且不产生候选频繁情节,本文采用深度优先搜索方式来发现频繁情节,并利用性质 1、性质 2 或性质 3 来压缩频繁情节的搜索空间.共享前/后缀树的构建过程如下.

第 1 步,生成一个只有根节点的树 T.

第 2 步,扫描事件序列 ES 一遍,依据支持度定义和支持度阈值,发现所有的频繁 1-情节并按字典序排列.

第 3 步,对于每个频繁 1-情节 α,在树 T 中生成根节点的孩子节点 N_α.

第 4 步,分别对每个频繁 1-情节 α,进行如下递归处理.

依次取出每个频繁 1-情节 β 对 α 进行情节增长,令增长后的情节为$\gamma=concat(\alpha,\beta)$或$\gamma=concat(\beta,\alpha)$,即$\gamma$是以 α 为前缀或后缀进行增长,每次增长后依据支持度定义计算γ的发生集,若γ是频繁情节,则在树 T 中添加 N_α 的孩子节点 N_γ,并对γ进行类似于 α 的情节增长处理,如此不断迭代,直至没有发现更长的频繁情节为止.

显然,构建共享前/后缀树时需要解决两个关键问题:一是存储策略,即每个节点存储情节的何种发生集才能保证不丢失情节支持度的计算依据;二是增长策略,即情节增长时是选择前缀增长还是后缀增长才能保证挖掘过程不会丢失频繁情节.为此,需要依据不同的支持度定义作如下处理.

情形 1. 窗口发生.

若节点 N_α 存储情节 α 的窗口发生集 $wo(\alpha)$,则无法计算 α 增长后情节的窗口发生集,这是因为 $wo(\alpha)$的每个窗口并未记载 α 中各事件类型的发生时间.为此,节点 N_α 需要存储情节 α 的最早转移发生集 $eto(\alpha)$,这样既可以计算情节 α 的窗口发生集 $wo(\alpha)$,也便于与其他支持度定义采取同一存储策略.例如,设窗口宽度 w=6,情节 α=⟨ABC⟩,则在事件序列 $ES1$ 上,$eto(\alpha)$={[1,5,8],[2,5,8],[4,5,8],[6,9,12],[7,9,12],[13,14,15]},由 $eto(\alpha)$中每个区间长度≤w 的发生[t_1,t_2,…,t_k],可以得到 w–(t_k–t_1)+1 个宽度为 w 的连续窗口.从以上窗口集中删除重复的窗口后,得到 $wo(\alpha)$={[2,8],[3,9],[4,10],[6,12],[7,13],[9,15],[10,16],[11,17],[12,18],[13,19]}.

该支持度定义满足性质 1,情节增长时可以选用前缀增长或后缀增长.

情形 2. 最小发生.

最小发生不受窗口宽度的约束.若节点 N_α 存储情节 α 的最早转移发生集 $eto(\alpha)$,则可以由此计算情节 α 的最小发生集 $mo(\alpha)$.例如,设情节 α=⟨ABC⟩,则事件序列 $ES1$ 上,$eto(\alpha)$={[1,5,8],[2,5,8],[4,5,8],[6,9,12],[7,9,12],[13,14,15]},对于 $eto(\alpha)$中相邻的两个发生,若结束时间相同,则前者一定不是最小发生,所以 $mo(\alpha)$={[4,5,8],[7,9,12],[13,14,15]}.

该支持度定义满足性质 2 和性质 3,情节增长时可以选用前缀增长或后缀增长.

情形 3. 头发生.

头发生是一种窗口发生,也受窗口宽度的约束.若节点 N_α 存储情节 α 的最早转移发生集 $eto(\alpha)$,则可以由此计算情节 α 的头发生集 $ho(\alpha)$.例如,设窗口宽度 w=6,情节 α=⟨ABC⟩,则在事件序列 $ES1$ 上,$eto(\alpha)$={[1,5,8],[2,5,8],[4,5,8],[6,9,12],[7,9,12],[13,14,15]},由 $eto(\alpha)$中每个区间长度≤w 的发生[t_1,t_2,…,t_k],可以得到 1 个长度为 w 的窗口[t_1,t_{1+w}],所有这些窗口组成的集合即为 $ho(\alpha)$={[2,8],[4,10],[6,12],[7,13],[13,19]}.

该支持度定义只满足性质 2,情节增长时只能进行前缀增长,否则会丢失频繁情节.

情形 4. 总发生.

总发生是一种头发生,同样受窗口宽度的约束.若节点 N_α 存储情节α的最早转移发生集 $eto(\alpha)$,则可以由此计算情节 α 的总发生集 $to(\alpha)$.由定义 9 可知,只有知道 α 的所有子情节基于头发生的支持度,才能计算 α 基于总发生的支持度.然而,在共享前/后缀树构建过程中若 α 由其前缀增长得到,则此时 α 的部分子情节支持度可能是未知的.

文献[11]提出了另一种基于总发生的支持度计算方法|$to(\alpha)$|=min(|$ho(\alpha)$|,|to($suffix(\alpha)$)|),这说明情节增长时只能采用后缀增长.设增长后情节为 α,增长前|to($suffix(\alpha)$)|已知,增长后|$ho(\alpha)$|可由 $eto(\alpha)$计算得到.例如,设窗口宽度 w=6,情节 α=⟨ABC⟩,则在事件序列 $ES1$ 上,$eto(\alpha)$={[1,5,8],[2,5,8],[4,5,8],[6,9,12],[7,9,12],[13,14,15]},|to($suffix(\alpha)$)|=|to(⟨BC⟩)|=3,故有|$ho(\alpha)$|=|{[2,8],[4,10],[6,12],[7,13],[13,19]}|,|$to(\alpha)$|=min(|$ho(\alpha)$|,|to($suffix(\alpha)$)|)=min(5,3)=3.

情形 5. 非交错发生.

非交错发生也不受窗口宽度的约束.若节点 N_α 存储情节 α 的最早转移发生集 $eto(\alpha)$,则可以由此计算情节 α 的非交错发生集 $ni(\alpha)$.例如,设情节 α=⟨ABC⟩,则在事件序列 $ES1$ 上,$eto(\alpha)$={[1,5,8],[2,5,8],[4,5,8],[6,9,12],[7,9,12],[13,14,15]}.初始令 $ni(\alpha)$={[1,5,8]},然后将 $eto(\alpha)$中首先与[1,5,8]非交错的发生[6,9,12]添加至 $ni(\alpha)$中,再将 $eto(\alpha)$中首先与[6,9,12]非交错的发生[13,14,15]添加至 $ni(\alpha)$中,最后有 $ni(\alpha)$={[1,5,8],[6,9,12],[13,14,15]}.

该支持度定义满足性质 2 和性质 3,情节增长时可以选用前缀增长或后缀增长.

情形 6. 非重叠发生.

非重叠发生(未必是最小发生)也不受窗口宽度的约束.若节点 N_α 存储 α 的非重叠发生集 $no(\alpha)$,则深度优先搜索时可能会丢失频繁情节.例如,在事件序列 $ES1$ 上,no(⟨AAB⟩)={[1,2,5],[6,7,9]},no(⟨A⟩)={[1,1],[2,2],[4,4],

[6,6],[7,7],[13,13],[17,17]},从而有 $no(\langle AABA\rangle)$={[1,2,5,6]}.若支持度阈值 min_*sup*=2,则认为$\langle AABA\rangle$是非频繁的.事实上,$\langle AABA\rangle$在 *ES*1 上有非重叠发生[1,2,5,6]和[7,13,14,17],$\langle AABA\rangle$是频繁情节.

若节点 N_α存储情节 α 的最早转移发生集 $eto(\alpha)$,则不仅可以计算情节 α 的非重叠发生集 $no(\alpha)$,而且也保证挖掘过程中不会丢失频繁情节.例如,在事件序列 *ES*1 上,$eto(\langle AAB\rangle)$={[1,2,5],[2,4,5],[4,6,9],[6,7,9],[7,13,14]}.初始令 $no(\langle AAB\rangle)$={[1,2,5]},然后将 $eto(\langle AAB\rangle)$中首先与[1,2,5]非重叠的发生[6,7,9]添加至 $no(\langle AAB\rangle)$中,从而有 $no(\langle AAB\rangle)$={[1,2,5],[6,7,9]}.因 $eto(\langle A\rangle)$={[1,1],[2,2],[4,4],[6,6],[7,7],[13,13],[17,17]},故 $eto(\langle AABA\rangle)$= {[1,2,5,6],[2,4,5,6],[4,6,9,13],[6,7,9,13],[7,13,14,17]},从而有 $no(\langle AABA\rangle)$={[1,2,5,6],[7,13,14,17]}.

该支持度定义满足性质 1,情节增长时可以选用前缀增长或后缀增长.

情形 7. 最小且非重叠发生.

最小且非重叠发生也不受窗口宽度的约束.若节点 N_α存储 α 的最小且非重叠发生集 $mn(\alpha)$,则深度优先搜索时也可能会丢失频繁情节.例如,在事件序列 *ES*1 上,$mn(\langle AAB\rangle)$={[2,4,5],[6,7,9]},$mn(\langle A\rangle)$={[1,1],[2,2],[4,4],[6,6],[7,7],[13,13],[17,17]},从而有 $mn(\langle AABA\rangle)$={[2,4,5,6]}.若支持度阈值 min_*sup*=2,则认为$\langle AABA\rangle$是非频繁的.事实上,$\langle AABA\rangle$在 *ES*1 上有最小且非重叠发生[2,4,5,6]和[7,13,14,17],$\langle AABA\rangle$是频繁情节.

若节点 N_α存储情节 α 的最早转移发生集 $eto(\alpha)$,则不仅可以计算情节 α 的最小且非重叠发生集 $mn(\alpha)$,而且也保证挖掘过程中不会丢失频繁情节.例如,在事件序列 *ES*1 上,$eto(\langle AAB\rangle)$={[1,2,5],[2,4,5],[4,6,9],[6,7,9],[7,13,14]},$mo(\langle AAB\rangle)$={[2,4,5],[6,7,9],[7,13,14]}.初始令 $mn(\langle AAB\rangle)$={[2,4,5]},然后将 $mo(\langle AAB\rangle)$中首先与[2,4,5]非重叠的发生[6,7,9]添加至$mn(\langle AAB\rangle)$中,从而有$mn(\langle AAB\rangle)$={[2,4,5],[6,7,9]}.由$eto(\langle AABA\rangle)$={[1,2,5,6],[2,4,5,6],[4,6,9,13],[6,7,9,13],[7,13,14,17]}可得 $mo(\langle AABA\rangle)$={[2,4,5,6],[6,7,9,13],[7,13,14,17]},从而有 $mn(\langle AABA\rangle)$={[2,4,5,6],[7,13,14,17]}.

该支持度定义满足性质 1,情节增长时可以选用前缀增长或后缀增长.

2.2 算法描述

基于上述思想,我们设计了一种融合多种支持度定义的频繁情节挖掘算法 FEM-DFS(frequent episode mining-depth first search).该算法在进行情节增长时,基于窗口发生、头发生、最小发生、非交错发生、非重叠发生、最小且非重叠发生的支持度定义都采用前缀增长,而基于总发生的支持度定义采用后缀增长.

算法 FEM-DFS 的伪代码如下.

Algorithm FEM-DFS(ES,w,min_sup,def_sup).

Input: *ES*: an event sequence=$\langle(E_1,t_1),(E_2,t_2),...,(E_n,t_n)\rangle$;

w: a window width;

min_*sup*: a support threshold;

def_sup: one of support definitions. *wo*, *mo*, *ho*, *to*, *ni*, *no*, *mn* stand for window occurrence-based, minimal occurrence-based, head occurrence-based, total occurrence-based, non-interleaved occurrence-based, non-overlapped occurrence-based, minimal and non-overlapped occurrence-based respectively.

Output: *T*: a tree which stores all frequent episodes found from *ES*

1. Create a tree *T* with simply a root node
2. Scan *ES* once to find all frequent 1-episodes and sort them in lexicographical order
3. FOR each frequent 1-episode α do
4. Create node N_α as a child of the root
5. FOR each frequent 1-episode α do
6. *EpisodeGrow*(α)
7. Return *T*

Procedure *EpisodeGrow*(α).

Input: α: an episode to be grown

Objective: grow the existing frequent episode α

1. FOR each frequent 1-episode β do
2. IF *def_sup*≠'*to*'
3. Let γ=*concat*(α,β)
4. ELSE
5. Let γ=*concat*(β,α)
6. Let *eto*(γ)=*ComputeETO*(*eto*(α),*eto*(β),*def_sup*)
7. Let *sup*(γ)=*ComputeSUP*(*eto*(γ),*def_sup*)
8. IF *sup*(γ)≥min_*sup*
9. Create node N_γ as a child of node N_α
10. *EpisodeGrow*(γ)

Procedure *ComputeETO*(*eto*(α),*eto*(β),*def_sup*).

Input: *eto*(α): the earliest transiting occurrences of α listed in increasing order of start time;
eto(β): the earliest transiting occurrences of β listed in increasing order of start time;
def_sup: one of support definitions

Output: *eto*(γ): the earliest transiting occurrences of concat(α,β)

1. Let *eto*(γ)=∅
2. IF *def_sup*≠'*to*'
3. FOR i=1 to |*eto*(α)| do //i points to the ith occurrence [$t_{i1},t_{i2},\ldots,t_{i|\alpha|}$] of *eto*($\alpha$)
4. FOR j=1 to |*eto*(β)| do //j points to the jth occurrence [t_j,t_j] of *eto*(β)
5. IF $t_j>t_{i|\alpha|}$
6. Add [$t_{i1},t_{i2},\ldots,t_{i|\alpha|},t_j$] to *eto*($\gamma$)
7. break
8. ELSE
9. FOR j=1 to |*eto*(β)| do //j points to the jth occurrence [t_j,t_j] of *eto*(β)
10. FOR i=1 to |*eto*(α)| do //i points to the ith occurrence [$t_{i1},t_{i2},\ldots,t_{i|\alpha|}$] of *eto*($\alpha$)
11. IF $t_{i1}>t_j$
12. Add [$t_j,t_{i1},t_{i2},\ldots,t_{i|\alpha|}$] to *eto*($\gamma$)
13. break
14. RETURN *eto*(γ)

Procedure *ComputeSUP*(*eto*(α),*def_sup*).

Input: *eto*(α): the earliest transiting occurrences of α listed in increasing order of start time;
def_sup: one of support definitions

Output: *sup*(α): the support of α

1. switch(*def_sup*)
2. case '*wo*':
3. Let *wo*(α)=∅
4. FOR i=1 to |*eto*(α)| do //i points to the ith occurrence [$t_{i1},t_{i2},\ldots,t_{i|\alpha|}$] of *eto*($\alpha$)
5. IF $t_{i|\alpha|}-t_{i1}\leqslant w$
6. FOR j=0 to $w-(t_{i|\alpha|}-t_1)$ do
7. IF [$t_{i|\alpha|}-w+j,t_{i|\alpha|}+j$]∉*wo*($\alpha$)
8. Add [$t_{i|\alpha|}-w+j,t_{i|\alpha|}+j$] to *wo*($\alpha$)

```
9.      RETURN |wo(α)|
10.   case 'mo':
11.     Let mo(α)=∅
12.     FOR i=1 to |eto(α)| do    //i points to the ith occurrence [t_i1,t_i2,…,t_i|α|] of eto(α)
13.       IF t_i|α|<t_(i+1)|α|
14.           Add [t_i1,t_i2,…,t_i|α|] to mo(α)
15.     RETURN |mo(α)|
16.   case 'ho':
17.     Let ho(α)=∅
18.     FOR i=1 to |eto(α)| do    //i points to the ith occurrence [t_i1,t_i2,…,t_i|α|] of eto(α)
19.       IF t_i|α|–t_i1≤w
20.           Add [t_i1,t_i1+w] to ho(α)
21.     RETURN |ho(α)|
22.   case 'to':
23.     IF |α|=1
24.       RETURN |eto(α)|
25.     ELSE
26.       RETURN min(|eto(α)|,ComputeSUP(eto(suffix(α)),'to'))
27.   case 'ni':
28.    Let ni(α)={[t_11,t_12,…,t_1|α|]}    //Add the first occurrence of eto(α) to ni(α)
29.    Let j=1    //j points to the jth occurrence [t_j1,t_j2,…,t_j|α|] of ni(α)
30.     FOR i=2 to |eto(α)| do    //i points to the ith occurrence [t_i1,t_i2,…,t_i|α|] of eto(α)
31.       FOR k=1 to |α|–1 do
32.         IF t_ik<t_j(k+1)
33.           break
34.       IF k>|α|–1
35.         Add [t_i1,t_i2,…,t_i|α|] to ni(α)
36.         j=j+1
37.     RETURN |ni(α)|
38.   case 'no':
39.    Let no(α)={[t_11,t_12,…,t_1|α|]}    //Add the first occurrence of eto(α) to no(α)
40.    Let j=1    //j points to the jth occurrence [t_j1,t_j2,…,t_j|α|] of no(α)
41.     FOR i=2 to |eto(α)| do    //i points to the ith occurrence [t_i1,t_i2,…,t_i|α|] of eto(α)
42.       IF t_i1>t_j|α|
43.         Add [t_i1,t_i2,…,t_i|α|] to no(α)
44.         j=j+1
45.     RETURN |no(α)|
46.   case 'mn':
47.     Let mn(α)=∅
48.     FOR i=1 to |eto(α)| do    //i points to the ith occurrence [t_i1,t_i2,…,t_i|α|] of eto(α)
49.       IF t_i|α|≠t_(i+1)|α|∧t_i|α|<t_(i+1)1
50.         Add [t_i1,t_i2,…,t_i|α|] to mn(α)
```

51. RETURN $|mn(\alpha)|$

2.3 算法正确性证明

算法 FEM_DFS 的关键步骤是以发现的频繁情节为前缀或后缀进行情节增长,并由过程 ComputeETO 和 ComputeSUP 来分别计算增长后情节的最早转移发生集和支持度.为此,需要证明过程 ComputeETO 和 ComputeSUP 的正确性.不失一般性,对于过程 ComputeETO 而言,只证明前缀增长情形下的正确性.

定理 1. 给定频繁情节 α 和频繁 1-情节 β 的最早转移发生集 eto(α)和 eto(β),过程 ComputeETO 计算得到的是情节$\gamma=concat(\alpha,\beta)$的最早转移发生集 $eto(\gamma)$.

证明:

(1) 证明 $eto(\gamma)$中的每个发生都是情节$\gamma=concat(\alpha,\beta)$的最早转移发生.

设$[t_1,t_2,\ldots,t_{|\alpha|},t_\beta]$是由$[t_1,t_2,\ldots,t_{|\alpha|}]\in eto(\alpha)$和$[t_\beta,t_\beta]\in eto(\beta)$根据时序关系连接得到(记为条件 1),这表示$[t_1,t_2,\ldots,t_{|\alpha|}]$是 t_β前α的最近一次最早转移发生,t_β是 $t_{|\alpha|}$后β的最早发生时间.

若$[t_1,t_2,\ldots,t_{|\alpha|},t_\beta]$不是$\gamma$的一次最早转移发生,则必然存在$\gamma$的一次最早转移发生$[t_1,t_2,\ldots,t_{|\alpha|},t_{\beta'}]$,此时,$t_{\beta'}$有两种可能:$t_{\beta'}<t_{|\alpha|}$或 $t_{|\alpha|}<t_{\beta'}<t_\beta$.若 $t_{\beta'}<t_{|\alpha|}$,说明$[t_1,t_2,\ldots,t_{|\alpha|}]$不是$\alpha$的一次最早转移发生,这与条件 1 矛盾;若 $t_{|\alpha|}<t_{\beta'}<t_\beta$,说明 t_β不是 $t_{|\alpha|}$后β的最早发生时间,这也与条件 1 矛盾.

(2) 证明 $eto(\gamma)$包含了情节$\gamma=concat(\alpha,\beta)$的所有最早转移发生.

γ的任何一次最早转移发生$[t_1,t_2,\ldots,t_{|\alpha|},t_\beta]$都蕴含着$[t_1,t_2,\ldots,t_{|\alpha|}]$是 t_β前α的最近一次最早转移发生,t_β是 $t_{|\alpha|}$后β的最早发生时间.根据时序关系,算法 ComputeETO 都能将$[t_1,t_2,\ldots,t_{|\alpha|}]$和$[t_\beta,t_\beta]$连接成$[t_1,t_2,\ldots,t_{|\alpha|},t_\beta]$作为$\gamma$的一次最早转移发生.

综合上述(1)和(2),定理 1 得证. □

性质 4. 设 h 是情节α的一次最早转移发生,h'是情节α的一次发生,若 $h(t_1)\leqslant h'(t_1)$,则 $h(t_i)\leqslant h'(t_i)$,其中,$1\leqslant i\leqslant|\alpha|$,$h(t_i)$和 $h'(t_i)$分别表示 h 和 h'中α的第 i 个事件类型发生的时间.

推论 1. 记 h_i 是情节α的第 i 次最早转移发生.若 $h_i(t_1)<h_j(t_1)$,则 $h_i(t_k)\leqslant h_j(t_k)$,其中,$i<j$,$2\leqslant k\leqslant|\alpha|$.

推论 2. 设 h_i 是情节α的第 i 次最早转移发生,当且仅当 $h_i(t_{|\alpha|})=h_{i+1}(t_{|\alpha|})$时,$h_i$ 不是情节α的一次最小发生.

证明:

(a) 证明充分性

因为 $h_i(t_{|\alpha|})=h_{i+1}(t_{|\alpha|})$,且由推论 1 可知 $h_i(t_1)<h_{i+1}(t_1)$,所以 h_i 不是情节α的一次最小发生,充分性得证.

(b) 证明必要性

假设 $h_i=[h_i(t_1),h_i(t_2),\ldots,h_i(t_{|\alpha|})]$不是$\alpha$的一次最小发生,则窗口$[h_i(t_1),h_i(t_{|\alpha|})]$中必然包含$\alpha$的另一次发生 $h=[h(t_1),h(t_2),\ldots,h(t_{|\alpha|})]$,此时有两种可能:$h(t_1)=h_i(t_1)$或 $h(t_1)>h_i(t_1)$.若 $h(t_1)=h_i(t_1)$,则 $h(t_{|\alpha|})<h_i(t_{|\alpha|})$,然而,由于 h_i 是α的最早转移发生,任何由 $h_i(t_1)$开始的α的发生不可能在 $h(t_{|\alpha|})$之前结束,所以,$h(t_{|\alpha|})<h_i(t_{|\alpha|})$是不可能的,这表示 $h(t_1)=h_i(t_1)$也是不可能的.若 $h(t_1)>h_i(t_1)$,则根据性质 4,有 $h_i(t_{|\alpha|})\leqslant h(t_{|\alpha|})$.因为 h 是窗口$[h_i(t_1),h_i(t_{|\alpha|})]$中的一次发生,所以 $h_i(t_{|\alpha|})=h(t_{|\alpha|})$.由于 h_{i+1} 是继 h_i 之后α的最近最早转移发生,且存在由 $h(t_1)$开始的α的发生 h,所以 $h_{i+1}(t_1)\leqslant h(t_1)$,$h_{i+1}(t_{|\alpha|})\leqslant h(t_{|\alpha|})$.再由性质 4,有 $h_i(t_{|\alpha|})\leqslant h_{i+1}(t_{|\alpha|})$.综合 $h_i(t_{|\alpha|})=h(t_{|\alpha|})$,$h_{i+1}(t_{|\alpha|})\leqslant h(t_{|\alpha|})$和 $h_i(t_{|\alpha|})\leqslant h_{i+1}(t_{|\alpha|})$,有 $h_i(t_{|\alpha|})=h_{i+1}(t_{|\alpha|})$,必要性得证.

综合上述(a)和(b),推论 2 得证. □

推论 3. 由推论 2 可知,设 h_i 是情节α的第 i 次最早转移发生,若 $h_i(t_{|\alpha|})<h_{i+1}(t_{|\alpha|})$,则 h_i 是情节α的一次最小发生,反之亦然.为此,要计算情节α的最小发生集 $mo(\alpha)$,可以从情节α的最早转移发生集 $eto(\alpha)$中,选择满足 $h_i(t_{|\alpha|})<h_{i+1}(t_{|\alpha|})$的最早发生 h_i.

性质 5. $no(\alpha)$中 h_1 是情节α的第 1 次最早转移发生,h_{i+1} 是 $h_i(t_{|\alpha|})$之后与 h_i 首次非重叠的发生,$h_n(t_{|\alpha|})$之后不存在与 h_n 非重叠的发生.

推论 4. 情节α至多存在$|no(\alpha)|=n$ 个非重叠的发生.

证明:设 $no'(\alpha)=\{h_1',h_2',\ldots,h_m'\}$是$\alpha$的任一非重叠发生集,并令 $k=\min(m,n)$.现用归纳法证明 $h_i(t_{|\alpha|})\leqslant h_i'(t_{|\alpha|})$,

其中,$1\leqslant i\leqslant k$.证明过程如下.

当 i=1 时,假设 $h_1'(t_{|\alpha|})<h_1(t_{|\alpha|})$,则存在$\alpha$的一个开始于 $h'(t_1)$的最早转移发生 g.由性质 4,有 $g(t_{|\alpha|})\leqslant h_1'(t_{|\alpha|})$,这表示找到$\alpha$的一个比 h_1 还早的最早转移发生,这与性质 5 矛盾.因此,$h_1(t_{|\alpha|})\leqslant h_1'(t_{|\alpha|})$成立.

当 $1\leqslant i<k$ 时,假设 $h_i(t_{|\alpha|})\leqslant h_i'(t_{|\alpha|})$成立,需要证明 $h_{i+1}(t_{|\alpha|})\leqslant h_{i+1}'(t_{|\alpha|})$也成立.假设 $h_{i+1}'(t_{|\alpha|})<h_{i+1}(t_{|\alpha|})$,则与 i=1 时类似,在 $h_i(t_{|\alpha|})$之后可以找到一个比 h_{i+1} 还早且与 h_i 非重叠的发生,这与性质 5 矛盾.因此,$h_{i+1}(t_{|\alpha|})\leqslant h_{i+1}'(t_{|\alpha|})$成立.

综上,$h_i(t_{|\alpha|})\leqslant h_i'(t_{|\alpha|})$,其中,$1\leqslant i\leqslant k$,从而有 $m\leqslant n$.因为 $h_n(t_{|\alpha|})$之后不存在与 h_n 非重叠的发生(性质 5),所以情节α至多存在$|no(\alpha)|=n$ 个非重叠的发生.推论 4 得证. □

定理 2. 给定情节α的最早转移发生集 $eto(\alpha)$和支持度定义,过程 ComputeSUP 计算得到的是α的支持度.

证明:给定事件序列和窗口宽度,则有如下事实:情节α的窗口发生集 $wo(\alpha)$、头发生集 $ho(\alpha)$和最小发生集 $mo(\alpha)$都是唯一的;由α的所有子情节的头发生集可以计算α的总发生支持度$|to(\alpha)|$;情节α的非重叠发生集 $no(\alpha)$、非交错发生集 $ni(\alpha)$、最小且非重叠发生集 $mn(\alpha)$可能是不唯一的.以下分情形证明定理 2 的正确性.

(1) 窗口发生、头发生和总发生

对于窗口发生,过程 ComputeSUP 依据定义 6,首先计算情节α的窗口发生集 $wo(\alpha)$,然后求得α的支持度$|wo(\alpha)|$;对于头发生,过程 ComputeSUP 依据定义 8,首先计算情节α的头发生集 $ho(\alpha)$,然后求得α的支持度$|ho(\alpha)|$;对于总发生,过程 ComputeSUP 依据文献[11]提出的总发生支持度计算方法,首先计算情节α的$|ho(\alpha)|$,然后由 $\min(|ho(\alpha)|,|to(suffic(\alpha))|)$求得情节$\alpha$的支持度$|to(\alpha)|$.

(2) 最小发生

依据推论 3,过程 ComputeSUP 首先计算情节α的头发生集 $mo(\alpha)$,然后求得α的支持度$|mo(\alpha)|$.

(3) 非重叠发生

过程 ComputeSUP 依据定义 11,首先计算情节α的非重叠发生集 $no(\alpha)=\{h_1,h_2,\ldots,h_n\}$,其中,每个 h_i 都是α的最早转移发生,然后求得α的支持度$|no(\alpha)|$.依据推论 4,$no(\alpha)$是α的一个非重叠发生最大集.

(4) 非交错发生、最小且非重叠发生

过程 ComputeSUP 首先依据定义 10 和定义 12,分别计算情节α的非交错发生集 $ni(\alpha)$、最小且非重叠发生集 $mn(\alpha)$,然后分别求得α的支持度$|ni(\alpha)|$、$|mn(\alpha)|$.$ni(\alpha)$和 $mn(\alpha)$是最大集的证明同 $no(\alpha)$.

综合上述(1)~(4),定理 2 得证. □

2.4 算法复杂度分析

设$|ES|$为事件序列 ES 的长度,ε为 ES 上的事件类型集,FE 为 ES 上的频繁情节集,则有:

定理 3. FEM-DFS 的时间复杂度为$O(|FE|\cdot|\varepsilon|\cdot|ES|)$.

证明:FEM-DFS 的时间代价主要在情节增长处理.设待增长的频繁情节为α,与之串接的频繁 1-情节为β,增长后的情节为γ,完成一次情节增长,需要扫描 $eto(\alpha)$和 $eto(\beta)$以计算 $eto(\gamma)$和 $sup(\gamma)$,其时间复杂度为$O(|ES|)$.FEM-DFS 需以 FE 中的每个情节为前缀或后缀进行情节增长,增长次数至多为频繁 1-情节的个数(上界为$|\varepsilon|$).因此,FEM-DFS 的时间复杂度为$O(|FE|\cdot|\varepsilon|\cdot|ES|)$. □

定理 4. FEM-DFS 的空间复杂度为$O(|FE|\cdot|ES|)$.

证明:FEM-DFS 需要维护所有的频繁情节,每个频繁情节需要存储其最早转移发生集,至多需要空间$|ES|$,所以 FEM-DFS 的空间复杂度为$O(|FE|\cdot|ES|)$. □

3 各种支持度比较

定理 5. 情节α在给定事件序列上的支持度满足:$|wo(\alpha)|\geqslant|ho(\alpha)|\geqslant|to(\alpha)|$,$|ni(\alpha)|\geqslant|mo(\alpha)|\geqslant|no(\alpha)|=|mn(\alpha)|$.

证明:

(1) 证明$|wo(\alpha)|\geqslant|ho(\alpha)|\geqslant|to(\alpha)|$.

由定义 6、定义 8、定义 9 得证.

(2) 证明$|ni(\alpha)|\geqslant|mo(\alpha)|\geqslant|mn(\alpha)|=|no(\alpha)|$.

首先证明$|ni(\alpha)|\geq|mo(\alpha)|$,即证明:对于情节$\alpha$相邻的最早转移发生$h_i$和$h_{i+1}$,若$h_i$是最小发生,则$h_i$和$h_{i+1}$一定是非交错发生.我们使用反证法证明.因为$h_i$是最小发生,所以有$h_i(t_k)<h_{i+1}(t_k)$,其中,$1\leq k\leq|\alpha|$.假设$h_i$和$h_{i+1}$不是非交错发生,则必然存在一个小于$|\alpha|$的$k$,使得$h_{i+1}(t_k)<h_i(t_{k+1})$,则有$h_i(t_k)<h_{i+1}(t_k)<h_i(t_{k+1})<h_{i+1}(t_{k+1})$.因$h_i(t_k)<h_{i+1}(t_k)<h_i(t_{k+1})$,且$h_i$和$h_{i+1}$是相邻的最早转移发生,所以有$h_i(t_{k+1})=h_{i+1}(t_{k+1})$,这与$h_i(t_{k+1})<h_{i+1}(t_{k+1})$矛盾,说明若$h_i$是最小发生,则$h_i$和$h_{i+1}$一定是非交错发生,从而证明了$|ni(\alpha)|\geq|mo(\alpha)|$.

其次证明$|mn(\alpha)|=|no(\alpha)|$.对于$no(\alpha)$中的任何一个发生$h$,若$h$是最小发生,则$h\in mn(\alpha)$,否则,根据推论2,一定存在一个发生$h'$,满足$h'(t_{|\alpha|})=h(t_{|\alpha|})$.显然,$h'$与$no(\alpha)$中的其他发生都是非重叠的.这表明,最小且非重叠发生的次数至少等于非重叠发生的次数,即$|mn(\alpha)|\geq|no(\alpha)|$.假设$|mn(\alpha)|>|no(\alpha)|$,则说明$no(\alpha)$不是$\alpha$的非重叠发生的最大集,这与推论4矛盾.因此,$|mn(\alpha)|=|no(\alpha)|$.

最后证明$|mo(\alpha)|\geq|mn(\alpha)|$.因为$mn(\alpha)$中的发生都是最小发生,所以$mn(\alpha)\subseteq mo(\alpha)$,即$|mo(\alpha)|\geq|mn(\alpha)|$. □

4 实验评估

我们使用文献[20,21]的合成数据集与真实数据集来进行实验评估.对于合成数据集,首先使用IBM合成数据生成器Quest Market-Basket的修改版生成了每个交易为单个项的交易序列,通过设置D=0.001,C=300000,N=20,S=300000,其中,参数D表示交易序列的个数(单位为1 000),C表示每个交易序列中交易的平均个数,N表示所有交易项的类型种数(单位为1 000),S为最长交易序列中交易的平均个数,这样就得到了一个20 000种交易项类型上的由300 000个交易组成的交易序列.然后,为该交易序列中的每个交易依次赋上一个连续的正整数以作为每个交易发生的时间戳,这样,就构造了一个20K种事件类型上的由300K个事件组成的事件序列.

对于真实数据集,考虑到作为国内最具影响力的知识传播与数字化学习平台,中国知网为全社会提供了最丰富、最全面的文献资源,为了能够发现中国知网相关文献之间的引用关系,并为广大学者展开相关研究提供个性化的推荐服务,我们选用了中国知网的一个Web服务器上从2010年11月1日~2010年11月30日的日志数据,该日志数据包括了相关读者对132 885种不同文献的211 665个阅读序列.

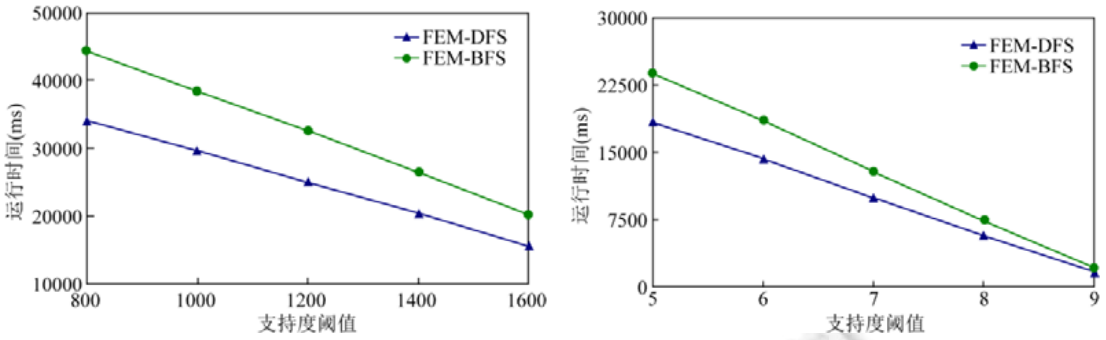
通过7组实验对比算法FEM-DFS和文献[19]提出的算法(为描述方便,简称为FEM-BFS)的时空性能,并分析FEM-DFS在不同支持度定义下的挖掘结果.实验的硬件环境为3.6GHz Intel(R) Core(TM) i7-4790 CPU,内存为8GB,操作系统为Windows 8,程序采用Java实现.

实验1. 运行时间vs.支持度阈值.合成数据集和真实数据集的窗口宽度分别设定为200和5天,通过改变支持度阈值,得到如图2所示的两种算法各自在7个支持度定义下的平均运行时间.

可以看出,随着支持度阈值的增加,两种算法的运行时间都在线性减少,且FEM-DFS要优于FEM-BFS,主要原因是:支持度阈值越大,频繁情节越少;FEM-BFS采用广度优先搜索策略,需要多遍扫描事件序列,而FEM-DFS采用深度优先搜索策略,只需单遍扫描事件序列.

尽管算法FEM-DFS和FEM-BFS在运行时间上存在差异,但它们的挖掘结果相同.例如,〈基于隐马尔可夫模型的多步攻击预测研究,面向分布数据安全的误用检测算法和入侵检测系统的研究,隐马尔可夫模型在入侵检测中的应用,基于入侵响应的入侵警报关联性的攻击预测算法,基于因果网络的攻击计划认别与预测,面向网络安全的基于入侵事件的早期预警方法〉是两种算法在真实数据集上都能挖掘得到的一个频繁6-情节,该情节刻画了一种行为模式:读者们旨在研究一个基于隐马尔可夫模型的攻击预测方法(见第1篇阅读文档),他们首先研究了两种常用的入侵检测模型,即误用检测模型(见第2篇阅读文档)和异常检测模型(见第3篇阅读文档),然后分析了现有攻击方法的不足(见后3篇阅读文档).算法FEM-DFS和FEM-BFS的挖掘结果有助于CNKI向读者提供个性化的文献阅读推荐服务.

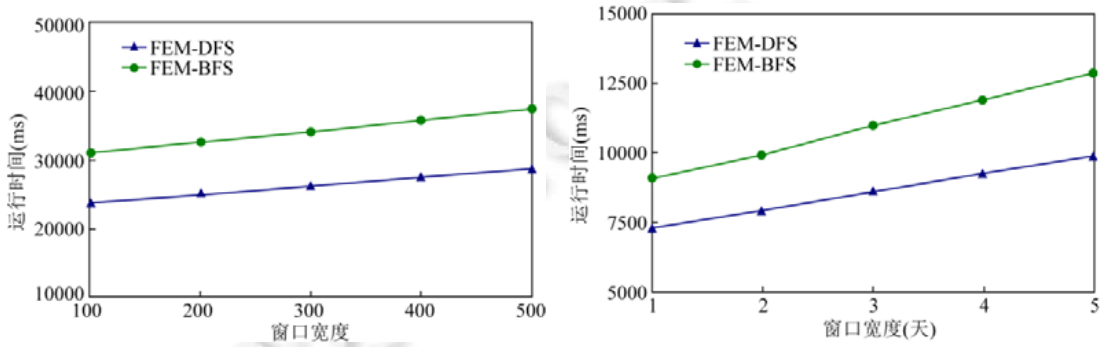
实验2. 运行时间vs.窗口宽度.合成数据集和真实数据集的支持度阈值分别设定为1 200和7,通过改变窗口宽度,得到如图3所示的两种算法各自在窗口发生、头发生和总发生支持度定义下的平均运行时间.



(a) 合成数据集 (b) 真实数据集

Fig.2 Average runtime vs. support threshold

图 2 运行时间 vs.支持度阈值



(a) 合成数据集 (b) 真实数据集

Fig.3 Average runtime vs. window width

图 3 运行时间 vs.窗口宽度

可以看出,随着窗口宽度的增加,两种算法的运行时间也在线性增加,这是因为窗口宽度越大,频繁情节越多.另外,FEM-DFS 要优于 FEM-BFS,这是因为两者采用了不同的搜索策略.

实验 3. 运行时间 vs.序列长度.设定合成数据集的支持度阈值和窗口宽度分别为 800 和 200,真实数据集的支持度阈值和窗口宽度分别为 7 和 5 天,并选择前 100K、前 150K、前 200K、前 250K 个和所有 300K 个合成数据作为 5 个合成子序列,选择前 6 天、前 12 天、前 18 天、前 24 天和所有 30 天真实数据作为 5 个真实子序列,通过改变序列长度,得到如图 4 所示的两种算法各自在 7 个支持度定义下的平均运行时间.

可以看出,两种算法的运行时间都随着序列长度的增加而线性增加,这是因为序列长度越大,频繁情节越多.另外,FEM-DFS 优于 FEM-BFS,这也源于两者不同的搜索策略.

实验 4. 内存开销 vs.支持度阈值.与实验 1 的设置相同,通过改变支持度阈值,得到如图 5 所示的两种算法各自在 7 个支持度定义下的平均内存开销.

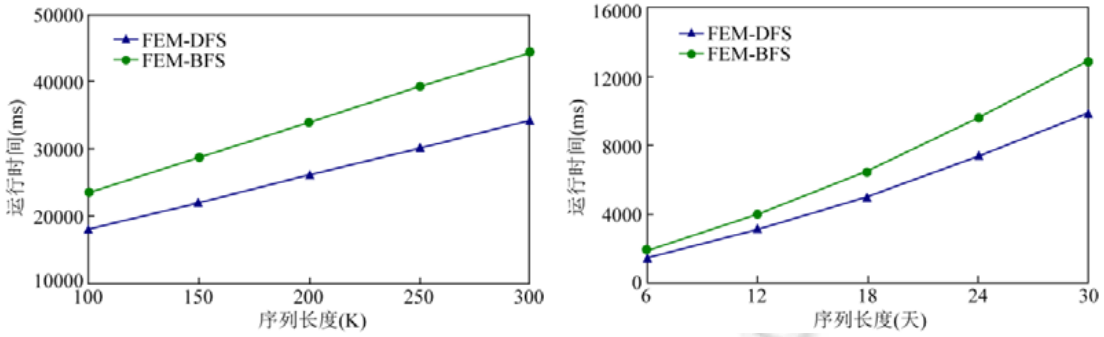
可以看出,随着支持度阈值的增加,两种算法的内存开销都在线性减少,且 FEM-DFS 要优于 FEM-BFS,原因与实验 1 相同.

实验 5. 内存开销 vs.窗口宽度.与实验 2 的设置相同,通过改变窗口宽度,得到如图 6 所示的两种算法各自在窗口发生、头发生和总发生支持度定义下的平均内存开销.

可以看出,随着窗口宽度的增加,两种算法的内存开销都在线性增加,但 FEM-DFS 要优于 FEM-BFS,原因与实验 2 相同.

实验 6. 内存开销 vs.序列长度.与实验 3 的设置相同,通过改变序列长度,得到如图 7 所示的两种算法各自在 7 个支持度定义下的平均内存开销.

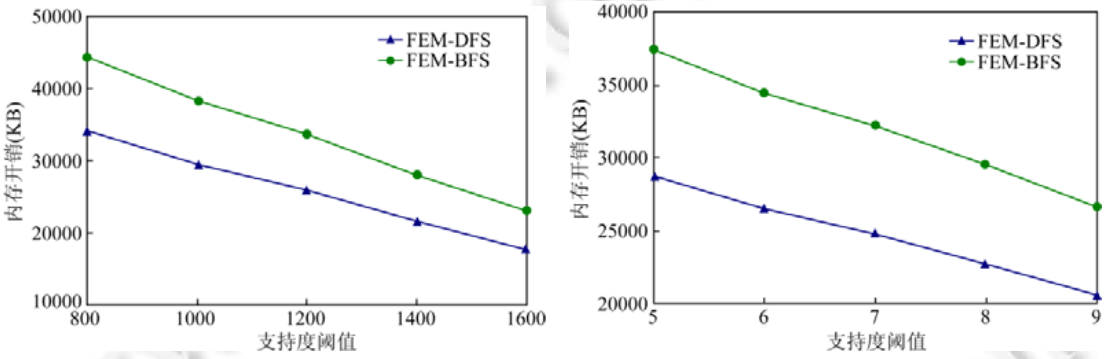
可以看出,两种算法的内存开销都随着序列长度的增加而呈线性增加,但 FEM-DFS 要优于 FEM-BFS,原因与实验 3 相同.



(a) 合成数据集 (b) 真实数据集

Fig.4 Average runtime vs. sequence length

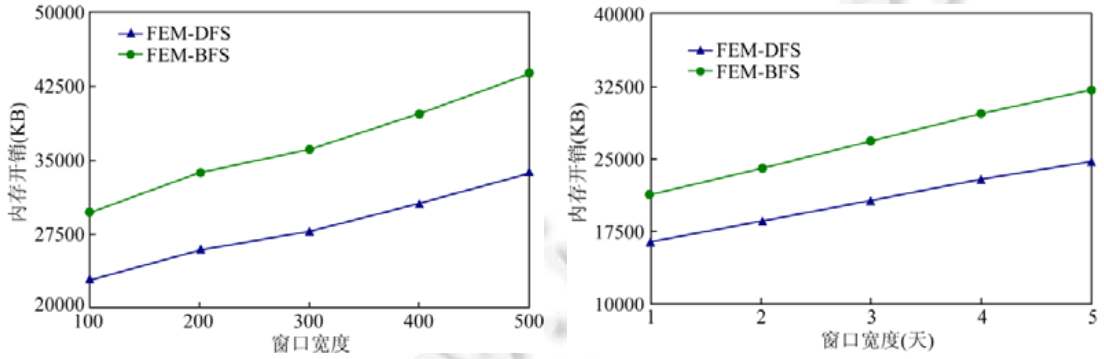
图 4 运行时间 vs.序列长度



(a) 合成数据集 (b) 真实数据集

Fig.5 Average memory vs. support threshold

图 5 内存开销 vs.支持度阈值

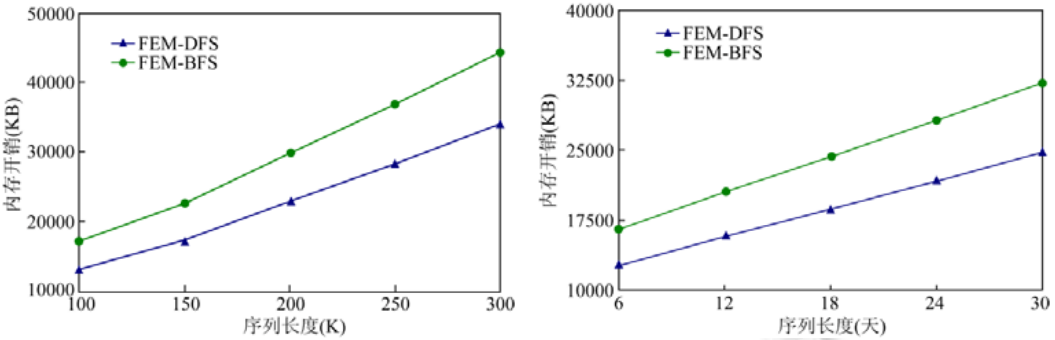


(a) 合成数据集 (b) 真实数据集

Fig.6 Average memory vs. window width

图 6 内存开销 vs.窗口宽度

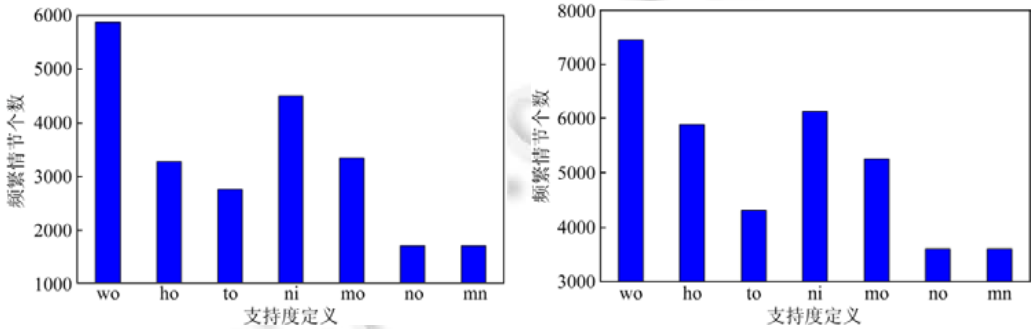
实验 7. 频繁情节个数 vs.支持度定义.选择 300K 合成数据集和 30 天真实数据集,两个数据集的支持度阈值分别设定为 800 和 7,窗口宽度分别设定为 200 和 5 天,通过改变支持度定义,得到如图 8 所示的算法 FEM-DFS 在不同支持度定义下的频繁情节个数.



(a) 合成数据集 (b) 真实数据集

Fig.7 Average memory vs. sequence length

图 7 内存开销 vs.序列长度



(a) 合成数据集 (b) 真实数据集

Fig.8 Episode number vs. support definition

图 8 频繁情节个数 vs.支持度定义

可以看出,基于窗口发生、头发生、总发生的 3 种支持度定义,发现的频繁情节个数依次递减;基于非交错发生、最小发生、非重叠发生的 3 种支持度定义,发现的频繁情节个数也在依次递减;基于非重叠发生、最小且非重叠发生的两种支持度定义,发现的频繁情节个数相同.主要原因:同一情节的发生次数在不同支持度定义下有着固定的比较关系,这与定理 5 一致.

5 总结与展望

针对现有频繁情节挖掘算法存在的不足,本文提出了一个采用深度优先搜索方式和共享前/后缀树存储结构的频繁情节挖掘算法 FEM-DFS,满足了实际情况下用户多变的支持度定义需求,实验评估证实了算法 FEM-DFS 能够有效地发现事件序列上的频繁情节.

未来将基于事件流环境,研究一种能够融合多种支持度定义的频繁情节挖掘算法.

References:

[1] Mannila H, Toivonen H, Verkamo AI. Discovery of frequent episodes in event sequences. Data Mining and Knowledge Discovery, 1997,1(3):259–289.

[2] Méger N, Rigotti C. Constraint-based mining of episode rules and optimal window sizes. In: Proc. of PKDD. 2004. 313–324.

[3] Lin YF, Huang CF, Tseng VS. A novel methodology for stock investment using high utility episode mining and genetic algorithm. Applied Soft Computing, 2017,59:303–315.

[4] Ma X, Pang HH, Tan KL. Finding constrained frequent episodes using minimal occurrences. In: Proc. of the ICDM. 2004. 471–474.

[5] Zhou WZ, Liu HY, Cheng H. Mining closed episodes from event sequences efficiently. In: Proc. of the PAKDD. 2010. 310–318.

[6] Wu JJ, Wan L, Xu ZR. Algorithms to discover complete frequent episodes in sequences. In: Proc. of the PAKDD. 2011. 267–278.

[7] Wu CW, Lin YF, Yu PS, Tseng VS. Mining high utility episodes in complex event sequences. In: Proc. of the SIGKDD. 2013. 536–544.
[8] Lin SK, Qiao JZ. An episode mining method based on episode matrix and frequent episode tree. Control and Decision, 2013,28(3): 339–344 (in Chinese with English abstract).
[9] Ao X, Luo P, Li CK, Zhuang FZ, He Q, Shi ZZ. Discovering and learning sensational episodes of news events. In: Proc. of the WWW. 2014. 217–218.
[10] Huang KY, Chang CH. Efficient mining of frequent episodes from complex sequences. Information Systems, 2008,33(1):96–114.
[11] Iwanuma K, Ishihara R, Takano Y, Nabeshima H. Extracting frequent subsequences from a single long data sequence. In: Proc. of the ICDM. 2005. 186–193.
[12] Avinash A, Ibrahim A, Sastry PS. Pattern-growth based frequent serial episode discovery. Data & Knowledge Engineering, 2013,87:91–108.
[13] Laxman S. Discovering frequent episodes: Fast algorithms, connections with HMMs and generalizations [Ph.D. Thesis]. Bangalore, 2006.
[14] Laxman S, Sastry PS, Unnikrishnan K. Discovering frequent episodes and learning hidden Markov models: A formal connection. IEEE Trans. on Knowledge and Data Engineering, 2005,17(11):1505–1517.
[15] Laxman S, Sastry PS, Unnikrishnan KP. A fast algorithm for finding frequent episodes in event streams. In: Proc. of the SIGKDD. 2007. 410–419.
[16] Zhu HS, Wang P, He XM, Li YJ, Wang W, Shi BL. Effcient episode mining with minimal and non-overlapping occurrences. In: Proc. of the ICDM. 2010. 1211–1216.
[17] Zhu HS, Wang P, Wang W, Shi BL. Discovering frequent closed episodes from an event sequence. In: Proc. of the IJCNN. 2012. 2292–2299.
[18] Liao GQ, Yang XT, Xie S, Yu PS, Wan CX. Two phase mining for frequent closed episodes. In: Proc. of the WAIM. 2016. 55–66.
[19] Avinash A, Laxman S, Sastry PS. A unified view of the apriori-based algorithms for frequent episode discovery. Knowledge and Information Systems, 2012,31:223–250.
[20] Zhu HS, Wang W, Shi BL. Extracting non-redundant episode rules based on frequent closed episodes and their generators. Chinese Journal of Computers, 2012,35(1):53–64 (in Chinese with English abstract).
[21] Zhu HS. Research on stream prediction based on episode rule matching [Ph.D. Thesis]. Shanghai: Fudan University, 2011 (in Chinese with English abstract).

附中文参考文献:

[8] 林树宽,乔建忠.一种基于情节矩阵和频繁情节树的情节挖掘方法.控制与决策,2013,28(3):339–344.
[20] 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.计算机学报,2012,35(1):53–64.
[21] 朱辉生.基于情节规则匹配的数据流预测研究[博士学位论文].上海:复旦大学,2011.

朱辉生(1968－),男,博士,教授,CCF 高级会员,主要研究领域为数据库,数据挖掘,大数据.

汪卫(1970－),男,博士,教授,博士生导师,CCF 杰出会员,主要研究领域为数据库,数据挖掘,大数据.

陈琳(1981－),女,副教授,CCF 专业会员,主要研究领域为图像处理,模式识别.

施伯乐(1935－),男,教授,博士生导师,CCF 高级会员,主要研究领域为数据库,数据挖掘.

倪艺洋(1986－),女,博士,副教授,主要研究领域为移动通信,机器学习.