第 62 卷　第 1 期
2026 年 1 月

南京大学学报(自然科学)
JOURNAL OF NANJING UNIVERSITY
(NATURAL SCIENCE)

Vol. 62, No. 1
Jan., 2026

DOI:10.13232/j.cnki.jnju.2026.01.010

面向大规模图数据的参数化模式挖掘技术及应用

蒋　星,王　欣*,潘海洋,胡　滨
(西南石油大学计算机与软件学院,成都,610500)

摘　要:大规模图上的频繁模式挖掘(Frequent Pattern Mining,FPM)因其在社交分析等各类应用中的广泛应用,受到高度关注.受模式语义的约束,传统的FPM技术难以满足多样化的数据分析需求,基于此,提出有参模式(Parameterized Patterns,p-模式)及其挖掘技术.通过在模式中引入参数来拓展匹配语义,实现图中复杂关系的有效捕获;设计并实现了高效的挖掘算法(PMiner),用于从大图中发现频繁p-模式;提出基于p-模式的图关联规则(Graph Association Rule,GAR),并设计GAR挖掘算法GARGen,以发现图中节点间的潜在关联.在真实图数据上的实验不仅验证了上述算法的运行效率,还揭示了p-模式与传统模式的差异以及定义在p-模式上的GAR在链接预测等任务上的有效性.

关键词:有参模式,频繁模式挖掘,关联规则挖掘,图关联规则
中图分类号:TP301　　**文献标志码**:A

Parameterized pattern mining techniques and applications for large-scale graph data

Jiang Xing, Wang Xin*, Pan Haiyang, Hu Bin
(School of Computer Science and Software Engineering, Southwest Petroleum University, Chengdu, 610500, China)

Abstract: Frequent Pattern Mining (FPM) on large-scale graphs has garnered significant attention due to its broad applications in areas such as social network analysis. However, constrained by traditional pattern semantics, conventional FPM techniques struggle to meet the diverse demands of data analysis. To address this challenge, this paper introduces Parameterized Patterns (p-patterns) and their mining framework. By incorporating parameters into patterns, the matching semantics are extended, enabling the effective capture of complex relationships within graphs. An efficient mining algorithm, PMiner, is designed and implemented to discover frequent p-patterns from large graphs. Furthermore, this paper proposes Graph Association Rules (GAR) based on p-patterns and designs the GARGen algorithm to uncover latent associations between nodes. Experiments on real-world graph datasets not only validate the computational efficiency of the proposed algorithms but also highlight the distinctions between p-patterns and traditional patterns, as well as the effectiveness of GAR in tasks such as link prediction.

Keywords: parameterized pattern, frequent pattern mining, association rules mining, graph association rules

频繁模式挖掘(Frequent Pattern Mining,FPM)在图数据分析等领域发挥着至关重要的作用.一般地,FPM的目标是识别数据集中出现频率超过给定阈值的子图模式.根据应用背景,FPM的研究可分为两个主要分支:基于图数据库的频繁模式挖掘和基于单个大图的频繁模式挖

基金项目:国家自然科学基金(62172102),四川省科技创新人才基金(2022JDRC0009)
收稿日期:2025－10－21
* 通信联系人,E-mail:xinwang@swpu.edu.cn

掘.近年来,基于单个大图的频繁模式挖掘因其在社交网络分析[1]和关联规则挖掘[2]等领域的深度应用而获得了更多的关注.

传统FPM算法聚焦于发现具有确定标签的子图模式,然而,这些方法往往忽略了模式中的潜在变量或参数化需求.在现实场景中,一是图数据往往蕴含不确定信息,如节点的语义模糊、属性缺失等;二是缺乏参数的模式,其结构匹配语义严格有余,灵活不足,难以捕获图中潜在的复杂频繁结构.因此,有必要在模式中引入变量节点,通过参数化机制拓展匹配语义,进而更有效地捕捉图中深层次的节点间关联.图1展示了有参模式的有效性.

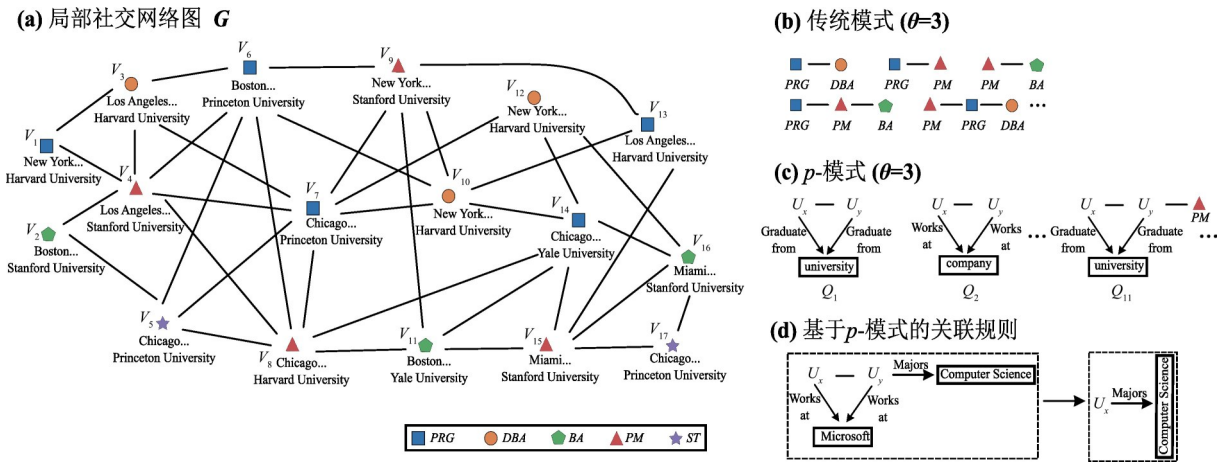


图1　(a)局部社交网络图G;(b)传统模式;(c)p-模式;(d)基于p-模式的关联规则

Fig. 1　(a) Snapshot of a social network G, (b) traditional patterns, (c) p-patterns, (d) a p-pattern-based association rule

例1　图1a展示了社交网络Google+的部分视图,其中每个节点代表一个用户,具有唯一的标识符和特定的职位(例如数据库管理员(DBA)、程序员(PRG)、业务分析师(BA)、软件测试员(ST)和项目经理(PM)),连接节点的边表示用户之间的友谊关系.图1b展示了传统频繁模式挖掘算法在图G上发现的一些频繁模式,支持度阈值设为3.图1c展示了PMiner在图G上挖掘的一些频繁p-模式(支持度阈值同样设为3),其中参数代表职位未知的用户,参数之间的关系通过布尔表达式进行约束($U_x.Attribute = U_y.Attribute$).这些$p$-模式可以反映社交网络中实体之间的规律[2].例如,如图1d所示,如果用户U_x和用户U_y是朋友,U_x和U_y都在微软工作,U_y之前的专业是计算机科学,那么U_x的专业也可能是计算机科学.

显然,与传统模式相比,有参模式(简称p-模式)的语义更丰富,能表达更复杂的拓扑关系,捕获具有特定意义的频繁结构.研究表明,p-模式在众多领域具有重要的应用价值,尤其是在知识补全和社交推荐中.p-模式的引入为大图分析,如关系推断提供了新的方法.然而,从大规模图数据中挖掘p-模式并非易事,需要算法层面的创新.本文的主要贡献如下.

(1)对传统模式进行语义扩展,提出了有参模式,用以捕捉节点间的复杂关系.

(2)设计了面向单一大图的p-模式挖掘算法PMiner.

(3)依托p-模式,拓展图关联规则语义,设计规则生成算法GARGen,实现图中节点关系的预测.

(4)在真实图数据上的实验结果表明PMiner在时间和空间开销方面表现出色.p-模式蕴含的特殊语义,使PMiner的挖掘结果排除了大量结构简单、使用价值低的单边模式.此外,GARGen能够捕获特殊语义的图关联规则,这类规则可以有效预测图中的缺失关系,平均准确率为63.27%,显著优于现有的链接预测方法.

1 相关工作

1.1 频繁模式及其变体的挖掘 近年来,单一大图上的FPM引起了广泛关注,催生出众多先进的挖掘算法. 值得注意的是,Elseidy et al[3]将FPM重新定义为约束满足问题,设计了GraMi算法,特别地,该算法利用具有反单调性的最小图像作为支持度度量来评估模式的频率. 由于GraMi专为单一大图上的挖掘任务设计,因而面临计算资源消耗大的问题. 为了提高挖掘效率,ScaleMine[4]采用了两阶段挖掘方法,首先通过近似处理快速识别频繁模式,随后通过并行挖掘对剩余模式进行挖掘,高效地解决FPM问题. 为了进一步提升FPM的挖掘效率,引入并行处理技术,如Graph-INC[5],TopKWY[6]和Peregrine[7]等算法通过与分布式系统结合显著提高了挖掘效率. 然而,分布式系统自身也带来了复杂和高成本的问题. 为此,Yuan et al[8]的T-FSM算法通过加载任务驱动的执行引擎以及多种优化技术的结合,不仅实现了较低的运行时间和内存消耗,也实现了高效的负载平衡. 汤小春等[9]提出基于数据流的FPM算法,通过"微批"模式和优化的正规编码计算提高了并行性,且能够处理大规模图数据,然而,大量的子图实例的存储给内存管理带来了挑战. 针对此问题,Cheng et al[10]的可扩展频繁模式挖掘算法通过引入数据间的关联性及内存管理优化策略,有效解决了大数据探索中的可扩展性问题,使算法能在内存有限的情况下高效运行. 为了适应不同的应用场景对算法进行扩展,挖掘目标也从通用模式转变成特定模式. 例如郭方方等[11]针对动态污点分析中存在的效率低下的问题,设计了SPIN-MGM算法来挖掘最大频繁子图,将生成树与最大频繁子图挖掘结合,显著提升了识别效率. 随着图数据规模的不断增长,传统的挖掘算法已经难以满足需求. 为了应对多重图上的挖掘挑战,Ingalalli et al[12]提出了MuGram算法,不仅能在多重图上发现频繁多重图模式,也能满足普通图上的挖掘需求. 针对加权图上的挖掘问题,Islam et al[13]的WFSM-MaxPWS框架引入了MaxPWS剪枝技术,实现了在动、静态权重图中加权频繁子图的高效挖掘. Min et al[14]提出一个新的频繁模式发现算法,将字母表分为强、中、弱三个组件来识别一种新型的模式,即tri-模式. Wu et al[15]提出基于序列模式的高平均效用不重叠序列模式(HANP),实验证明,和现有的频繁序列模式相比,HANP具有更高的挖掘价值. 这些特定模式大大增强了频繁模式挖掘的语义丰富性,能够更好地捕捉数据中的潜在关系. 截至目前,尚未发现可直接用于挖掘p-模式的算法,但前述的频繁模式挖掘算法为研究p-模式挖掘问题提供了宝贵的启示和思路.

1.2 关联规则挖掘及应用 关联规则挖掘旨在识别数据集中项集间的关联关系,定义了面向交易数据的关联规则,其中,最具代表性的关联规则挖掘算法是Apriori算法[10]. 随着图数据的广泛使用,面向图数据的关联分析受到越来越多的关注. 在早期的社交网络关联分析中[16],主要是基于社交图中提取的属性元组来挖掘传统规则和霍恩规则,没有充分考虑规则的结构特性. Fan et al[2]将关联规则从项集拓展到图模式,引入了特殊类型的图关联规则(Graph Pattern Association Rule, GPAR)用于开展社交媒体营销等,而GPAR中的后继被定义为包含单边的特殊模式图. 在此基础上,Wang et al[16]进一步提出代表性的GPAR,形式化了GPAR挖掘问题并将其分为两个子问题,即频繁模式挖掘和规则生成,高效地解决了关联规则挖掘中的挑战. 此外,Liu et al[17]拓展了传统GPAR的语义,提出GROs (Graph Rule with Oracles),通过引入内部计算与外部知识集成机制以及基于枢轴双模拟的匹配方式,解决了传统图规则在表达能力与计算效率上的瓶颈. 受益于GPAR挖掘问题的有效突破,大图上的链接预测问题也得到了有效解决. 链接预测旨在推断网络结构中节点间存在关联的可能性,在社交推荐等场景中至关重要,传统方法大多通过节点特征相似性度量来实现[18]. 近年来提出了一系列链接预测技术,其中,Benhidour et al[19]提出面向有向网络的链接预测法,包括相似度-流行度算法、路径探索算法以及混合算法,填补了有向网络上链接预测问题的研究空白. 然而,这类算法依然面临计算复杂度高的挑战. 此外,焦鹏飞等[18]提出一种基于多视图对比学习的动态网络链

接预测方法,实现了动态网络的表示学习和链接预测.

本文通过引入p-模式扩展了传统模式的语义,并进一步将p-模式融入图关联规则(Graph Association Rule,GAR),丰富了GAR的表达能力,强化了其在链接预测上的作用.

2　基本定义

定义1　图[3]　图可以定义为$G=(V,E,L)$,其中,(1)V是节点的集合;(2)$E\subseteq V\times V$是边的集合;(3)每个节点$v\in V$对应一个元组$L(v)=(A_1=a_1,A_2=a_2,\cdots,A_n=a_n)$,其中,$A_i=a_i\ (i\in[1,n])$表示节点$v$在属性$A_i$上的值,记作$v.A_i=a_i$.

给定图$G'=(V',E',L')$,如果$V'\subseteq V$,$E'\subseteq E$,并且对于任何节点$v\in V'$,有$L'(v)=L(v)$,则称G'是G的子图.

定义2　模式[20]　模式可以定义为$Q=(V_p,E_p,f_v)$,其中,V_p和E_p分别是节点和边的集合,而f_v是定义在V_p上的函数.对于任何节点$u\in V_p$,它的标签$f_v(u)$是原子公式的合取,每个公式的形式为"$A=a$",其中,A表示节点u的属性,a是A的值.通常,$f_v(u)$指节点u的查询条件.

定义3　图模式匹配[21]　对于图$G=(V,E,L)$和模式$Q=(V_p,E_p,f_v)$,如果图G中的节点v满足模式Q中节点u的搜索条件,即对于$f_v(u)$中的每个原子公式"$A=a$",在$L(v)$中存在一个属性A,使得$v.A=a$,那么称v匹配u,记作$v\sim u$.模式Q在图G中的匹配是由一个双射函数$h:V_s\Rightarrow V_p$形成的子图$G_s=(V_s,E_s,L_s)$,满足以下条件:

(1)$\forall v\in V_s,v\sim h(v)$;

(2)$\forall(v_i,v_j)\in E_s,(h(v_i),h(v_j))\in E_p$.

定义4　支持度[3]　模式Q在图G中的支持度,记作$Sup(Q,G)$,表示Q在G中出现的频率.基于图像的最小支持度MNI是一种广泛使用的支持度指标,具有反单调特性,如下所示:

$$Sup(Q,G)=\min\{|img(u)|\,|u\in V_p\}\tag{1}$$

其中,$img(u)$表示模式Q中节点u在图G上的匹配去重后的节点集合.本文使用MNI作为支持度指标.

定义5　有参模式(简称"p-模式")　p-模式可以定义为$Q[X]=(V_p,E_p,f_v,f_x)$,其中,

(1)V_p,E_p和f_v分别表示节点集、边集以及节点标签映射函数(同定义2);

(2)$X=\{U_x,U_y,\cdots\}$为一个包含不同变量的集合,其基数被限制为$|X|\in\{1,2\}$,且允许通配符"_"作为特殊变量出现;

(3)f_x是从V_p到X的双射函数,它为V_p中的节点指定一个唯一的变量.

对于节点$u\in V_p$,

(1)若$f_x(u)\notin X$,则节点u的查询条件由$f_v(u)$指定;

(2)若$f_x(u)\in X$且$|X|=1$,则节点u的查询条件由$f_x(u)$指定,且$f_v(u)=$"_";

(3)当$f_x(u)\in X$时,存在u的邻接节点u'使得$u.A=u'.A$,其中,邻接节点u'既可能是变量节点也可能是常规节点,A是u,u'的共同属性.

例2　对于模式Q_{11}(见图1c),其节点集$V_p=\{v_1,v_3,v_4,v_5,v_6,v_8,\cdots\}$,边集$E_p=\{(v_1,v_3),(v_3,v_4),(v_5,v_6),\cdots\}$.变量集合$X=\{U_x,U_y\}$,满足$|X|\in\{1,2\}$且$U_x.university=U_y.university$.标签映射函数$f_v$将$V_p$中的节点映射到模式中常规节点,如$f_v(v_4)=PM$;变量映射函数$f_x$将$V_p$中的节点映射到模式中变量节点,如$f_x(v_1)=U_x$,$f_x(v_3)=U_y$.

与定义3类似,p-模式在图G中的匹配是由一个双射函数$h':V_s\Rightarrow V_p$形成的子图$G_s=(V_s,E_s,L_s)$,满足以下条件:

(1)$\forall v\in V_s$,若$f_x(h'(v))\in X$,则存在一个节点$v'((v,v')\in E_s)$使得$h'(v).A=h'(v').A$;

(2)$\forall v\in V_s$,若$f_x(h'(v))\notin X$,$v\sim h'(v)$;

(3)$\forall(v_i,v_j)\in E_s,(h'(v_i),h'(v_j))\in E_p$;

(4)$\forall v\in V_s$,若$f_v(h'(v))=$"_"则$f_v(h'(v))=L(v)$,即通配符可以匹配任意标签,p-模式的支持度定义与定义4一致.

变量的引入丰富了p-模式的语义表达能力,

使其能够捕获传统模式(参见定义 2,又称“无参模式”)难以捕获的特殊结构,同时也带来因结果集过于庞大而导致的可用性低的新问题.

从实际应用出发,本文对 p-模式进行了约束,即:

(1)p-模式 $Q[X]$ 仅包含不超过两个变量节点;

(2)p-模式中变量节点 u 与其邻居节点 u' 通过布尔表达式 $u.A=u'.A$ 形成 $Q[X]$ 的变量约束,该约束在 $Q[X]$ 后续的扩展过程中将始终存在,即使变量节点被实例化.

例 3 如图 1a 所示,对于图 G 的子图 $G_1=\{(v_1,v_3),(v_3,v_4)\}$,不难发现 $v_1.university=v_3.university$,$v_1$ 映射到 U_x,v_3 映射到 U_y,v_4 映射到 PM,因此 G_1 构成 Q_{11} 在 G 中的一个有效匹配.此外,模式 Q_{11} 在 G 中的支持度 $Sup(Q_{11},G)=4$,因为,

$$Img(U_x)=\{v_1,v_5,v_{10},v_{11}\}$$
$$Img(U_y)=\{v_3,v_6,v_7,v_{13},v_{14}\}$$
$$Img(PM)=\{v_4,v_8,v_9,v_{15}\}$$

故,

$$|Img(U_x)|=|Img(PM)|=4<|Img(U_y)|=5$$

定义 6 模式域[20] 给定一个图 G 和一个节点集合为 V_p 的模式 Q,Q 的定义域记为 $D(Q,G)$. 将 Q 在 G 中的所有匹配 $M(Q,G)$ 组织为一个表格,该表格的列标题和主体分别对应模式节点 u_i(其中 u_i 是 V_p 的元素)及其对应的映射 $Img(u_i)$. $D(Q,G)$ 的第 j 个域被表示为 $D_j(Q,G)$,对应表格的第 j 列.

定义 7 前向扩展和后向扩展[20] 给定模式 Q,通过从其节点 u 对 Q 进行深度优先搜索来构建其 DFS 树 T_q,称 T_q 中的边为前向边,称 Q 中的其余边为后向边.因此前向扩展通过引入一条从 Q 中的现有节点到新引入节点的新边来扩大 Q,扩展出来的模式形如树状结构称为树模式.后向扩展从 Q 的两个现有节点引入新边,扩展出来的模式,其结构中包含回边,不再是树状结构,被称为非树模式.

定义 8 图关联规则 图关联规则(Graph Association Rule,GAR)定义为 $R=Q_l\rightarrow Q_r$,其中,Q_l 和 Q_r 分别是 R 的先导和后继,且满足以下条件:(1)Q_l 和 Q_r 都包含至少一条边;(2)Q_l 和 Q_r 都是连通的;(3)Q_l 和 Q_r 之间没有共享的边.

规则 R 表明,在图 G 中,如果存在一个同构映射 h_l,可以将模式 Q_l 映射到子图 G_1,那么很可能存在另一个同构映射 h_r,可以将模式 Q_r 映射到子图 G_2. 注意,Q_l 和 Q_r 中所有公共节点 u(u 属于 V_l 和 V_r 的交集,其中,V_l 和 V_r 分别为 Q_l 和 Q_r 的节点集)必须映射到图 G 中的同一个节点 v. 最后,将 Q_l 与 Q_r 的边集合扩展形成一个新模式 Q_R.

通过规则 R,可以开展好友推荐等任务.例如在图 1a 中,节点 v_3 被推荐给 v_2,节点 v_9 被推荐给 v_5. 除了社交推荐,图关联规则还能广泛应用于链路预测[22]、事件检测[23]和主题检索[23]等各种场景.

传统关联规则挖掘通过支持度实现搜索空间的剪枝.沿用定义 4 对模式支持度的定义,图关联规则 R 的支持度定义为:

$$Sup(R,G)=Sup(Q_R,G) \tag{2}$$

即将 R 视为模式 Q_R 并计算 Q_R 的支持度.

为了确定在 Q_l 成立的情况下 Q_r 成立的可能性,本文采用传统的置信度指标并定义图关联规则 R 在图 G 中的置信度($conf$)为模式 Q_R 在 G 中的支持度与模式 Q_l 在 G 中的支持度之比,如下所示:

$$conf(R,G)=\frac{Sup(Q_R,G)}{Sup(Q_l,G)} \tag{3}$$

例 4 如图 2 所示,图关联规则 R 可以表示为 $Q_l\rightarrow Q_r$,其中,Q_l 和 Q_r 分别是先导和后继.使用 Q_r 的边集扩展 Q_l,R 可以表示为模式图 Q_R.

进一步,Q_l 在图 G 中有四个匹配,因为 $\min(Img(u))=3$,其支持度为 $Sup(Q_l,G)=3$. 模式图 Q_R 在图 G 中仅有一个匹配,其支持度为 $Sup(Q_R,G)=1$.

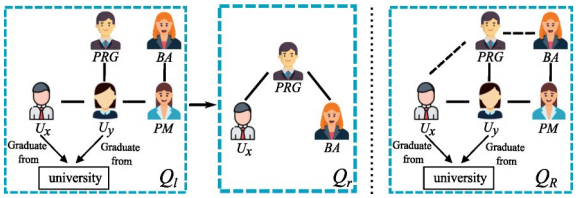


图 2 图关联规则 R

Fig. 2 The GAR R

因此,R的置信度为$conf(R,G)=\frac{1}{3}$.

3　p-模式挖掘算法

PMiner接受图G和支持度阈值θ作为输入,返回频繁p-模式集合S_t. 详细流程如算法1所示.

算法1　PMiner

输入:图G,阈值θ

输出:频繁p-模式集合S_t

Step 1. initialize $S_t=\varnothing$, T as an empty tree, $S^{[1]}$

Step 2. initialize *fsip*, remove $Q[X]$ from *fsip* if $Sup(Q[X],G)<\theta$; update S_t; update T

Step 3. $T=FwExtension(G,T,S^{[1]},\theta)$

Step 4. $T=BwExtension(G,T,S^{[1]},\theta)$

Step 5. update S_t with T

Step 6. return S_t

Step 7. function $FwExtension(G,T,S^{[1]},\theta)$

Step 8.　initialize $flag=$ true, $l_f=\varnothing$

Step 9.　while $flag\neq$ false do

Step 10.　　$l_f=PatTreeGen(G,T,S^{[1]})$

Step 11.　　for each unseen pattern Q_c in l_f do

Step 12.　　if $Sup(Q_c,G)\geqslant\theta$ then

Step 13.　　update T with Q_c

Step 14.　　end for

Step 15.　　if T is not updated then

Step 16.　　$flag=$ false

Step 17.　end while

Step 18. return T

Step 19.　function $BwExtension(G,T,S^{[1]},\theta)$

Step 20.　　initialize h as the height of T;

Step 21.　for each pattern Q at level h do

Step 22.　　$l_b=BackExt(G,Q,S^{[1]})$;

Step 23.　　for each unseen pattern Q_c in l_b do

Step 24.　　if $Sup(Q_c,G)\geqslant\theta$ then

Step 25.　　update T with Q_c

Step 26.　　end for

Step 27.　　update h

Step 28.　end for

Step 29. return T

PMiner首先初始化三个参数:一个集合S_t用于记录频繁p-模式,一棵模式树T用于将频繁p-模式根据生成关系进行组织,一个集合$S^{[1]}$用于存储所有支持度不低于阈值θ的单边无参模式(第1行),该集合将作为后续辅助模式扩展的核心结构. 随后,PMiner在图G中挖掘所有的单边p-模式,即仅包含一条边的p-模式,如图1c中的Q_1,Q_2,并将其存储在*fsip*中. 接着,PMiner移除所有支持度低于阈值θ的p-模式. 最终,这些频繁的单边p-模式被更新到T和S_t中,为后续扩展奠定基础(第2行). 初始化完成后,PMiner通过*FwExtension*进行p-模式的前向扩展(第7~18行)以及通过*BwExtension*进行p-模式的后向扩展(第19~29行). 最后,PMiner通过模式树T更新集合S_t,并返回更新后的S_t作为最终结果.

3.1　*FwExtension*函数　该函数接受图G、p-模式树T、单边无参模式集合$S^{[1]}$以及支持度阈值θ作为输入,返回经前向扩展形成的树结构的p-模式集合,这些p-模式以层次化的形式被组织在树T上.

首先,它初始化一个布尔变量*flag*,用于控制while循环的执行;它还创建集合l_f用于记录每一层扩展中生成的候选p-模式(第8行). 接下来,*FwExtension*反复生成候选模式并验证它们的支持度. 在每次迭代中,*FwExtension*调用*PatTreeGen*(见算法2)生成一组候选模式(第10行). 实际上,对于不同类型的待扩展模式,*PatTreeGen*采用与之相对应的扩展策略,以确保扩展过程的完整性,具体的扩展策略参见算法2. 对于l_f中的每个未生成过的候选模式Q_c,*FwExtension*会验证其支持度. 如果Q_c的支持度超过阈值θ,则将Q_c更新到模式树T中(第11~14行). 若处理完所有候选模式后,模式树T未被更新,*FwExtension*将把循环控制变量*flag*设置为false,以终止循环,返回模式树T(第15~18行).

3.2　*PatTreeGen*函数　*PatTreeGen*函数利用单边无参模式集合$S^{[1]}$对模式树T进行逐层扩展并返回一个候选模式集合,伪代码如算法2所示.

算法2　*PatTreeGen*

输入:图G,模式树T,单边模式集合$S^{[1]}$

输出:候选模式集合L

Step 1. initialize $L=\varnothing$, $L_{leaf}=\varnothing$, $L_c=\varnothing$

Step 2. $L_{leaf}=T.getLeaf(\ )$

Step 3. for each pattern Q in L_{leaf} do
Step 4. $L_c = PatGen(Q, S^{[1]})$
Step 5. if $Q.VertexNum = 2$ then
Step 6. $L_c = L_c \cup PatInst(Q, S^{[1]})$
Step 7. if $Q.ParameterNum = 1$ then
Step 8. $L_c = L_c \cup ParGen(Q, S^{[1]})$
Step 9. for each candidate pattern Q_{L_c} in L_c do
Step 10. $D(Q_{L_c}, G) = UpdateDomain(Q_{L_c}, S^{[1]}, G)$
Step 11. end for
Step 12. update L with L_c
Step 13. end for
Step 14. return L
Step 15. function $UpdateDomain(Q_{L_c}, S^{[1]}, G)$
Step 16. $Q_p = getParent(Q_{L_c}, T)$
Step 17. initialize $D(Q_{L_c}, G)$ with $D(Q_p, G)$
Step 18. for each $D_j(Q_{L_c}, G)$ in $D(Q_{L_c}, G)$ do
Step 19. for each invalid node v in $D_j(Q_{L_c}, G)$ do
Step 20. identify a match G_s including v
Step 21. if G_s is not found then
Step 22. remove v from $D_j(Q_{L_c}, G)$
Step 23. end for
Step 24. end for
Step 25. return $D(Q_{L_c}, G)$

算法 2 首先初始化三个集合:集合 L 用于存储所有候选模式,集合 L_{leaf} 用于记录叶子模式(即树 T 中叶子层的模式),集合 L_c 用于维护 L_{leaf} 中的每个模式经前向扩展所生成的候选模式(第 1 行). 接下来,*PatTreeGen* 调用 *getLeaf* 函数(未展示)来获取模式树 T 的所有叶节点(第 2 行). 基于这些叶节点,算法执行前向扩展(首轮扩展中叶节点对应着最简单的频繁单边 p-模式).

PatTreeGen 针对不同的待扩展模式采用不同的扩展策略:

(1)对于每个模式 Q,*PatTreeGen* 调用 *PatGen* 执行前向扩展,通过 $S^{[1]}$ 中记录的单边无参模式,引入一条从 Q 中的现有节点到新引入的节点的新边来扩大 Q(第 4 行).

(2)如果模式 Q 为单边 p-模式(即首次扩展阶段),*PatTreeGen* 调用 *PatInst* 对 Q 进行实例化扩展,在扩展过程中实例化一个参数,从而生成单参 p-模式,即仅包含一个参数的 p-模式(第 5~6 行).

(3)如果模式 Q 为单参 p-模式,*PatTreeGen* 调用 *ParGen* 函数进行参数扩展,引入一个新的参数节点并与原模式连接(新引入的参数节点与扩展点之间需满足属性约束),生成包含两个参数的新模式(第 7~8 行). 之后,*PatTreeGen* 调用 *UpdateDomain* 函数更新所有由 Q 扩展出的候选模式的模式域(第 9~11 行).

处理完 L_c 中的所有候选模式后,*PatTreeGen* 将 L_c 更新到 L 中(第 12 行). 在 L_{leaf} 中的模式扩展完成后,*PatTreeGen* 将候选模式集合 L 返回给 *FwExtension*(第 14 行).

例 5 待扩展模式为模式 Q_1: $U_x - U_y$ $(U_x.university = U_y.university)$(见图 1c)时,*PatTreeGen* 的扩展流程如下:

(1)调用 *PatGen* 对 Q_1 进行前向扩展,生成一系列 p-模式(例如 $U_x - U_y - PM$, $U_x - U_y - DBA$ 等)并将其放入候选模式集合 L_c 中.

(2)调用 *PatInst* 对 Q_1 进行实例化扩展,生成一系列单参 p-模式,例如 $U_x - PRG - PM$(注意节点 U_x 与实例化节点间仍满足属性约束,即 U_x. university = PRG. university),之后将其加入到 L_c 中.

(3)由于 Q_1 已包含两个参数,算法不再调用 *ParGen* 进行参数扩展(若待扩展模式为 $U_x - PRG - PM$ 时,便可通过参数扩展产生形如 $U_x - PRG - PM - U_z$ 的新模式,其中,U_z 为新引入的参数节点且 PM 与 U_z 之间需满足属性约束).

所有候选模式存入 L_c 后,再依次执行更新模式域、支持度检测.

3.3 *UpdateDomain* 函数 首先,*UpdateDomain* 函数调用 *getParent* 函数在 p-模式树 T 中查找 Q_{L_c} 的父模式 Q_p(第 16 行). 接着,该函数初始化 $D(Q_{L_c}, G)$(第 17 行),具体地,首先将 $D(Q_{L_c}, G)$ 设置为 $D(Q_p, G)$,然后向 $D(Q_{L_c}, G)$ 中扩展一个空列,存储上一轮前向扩展中新引入节点 v' 的匹配点集,并用 $S^{[1]}$ 中单边模式 (v, v') 的匹配情况填充该列,列中的节点是可能与新节点匹配的节点.

之后,*UpdateDomain*函数逐列检查$D(Q_{L_c}, G)$中节点的有效性(第18~24行).在每次迭代中,函数选择一个尚未标记为有效的节点v,并尝试在G中找到一个匹配G_s,使v属于G_s并与第j列对应的模式节点u_j相匹配(第19~20行).值得一提的是,G_s的识别是通过引导搜索完成的.在验证节点v的有效性时,算法实际上是在检查v是否能够与其他列中的节点组合成模式Q_{L_c}在G中的完整匹配.在此过程中,算法引入了一些剪枝策略:如果验证发现v可以与来自其他列的节点匹配,则这些节点可以直接在各自的列中标记为有效,从而避免对这些节点的重复验证.如果无法找到匹配G_s,*UpdateDomain*函数将从$D_j(Q_{L_c}, G)$中移除v(第21~22行).最后,当$D(Q_{L_c}, G)$的所有列都处理完毕后,*UpdateDomain*将更新后的模式域返回给*PatTreeGen*(第25行).

例6　以Q_{11}(图1c)为例,

$$D(Q_{11}, G)=\left[U_x:[v_1, v_5, v_{10}, v_{11}], U_y:[v_3, v_6, v_7, v_{13}, v_{14}], PM:[v_4, v_8, v_9, v_{15}]\right]$$

对于模式Q_0:$PM-BA$,

$$D(Q_0, G)=\left[PM:[v_4, v_8, v_9, v_{15}], BA:[v_2, v_{11}, v_{16}]\right]$$

当通过单边模式Q_0对Q_{11}执行前向扩展时,可得新模式Q_{111}:$U_x-U_y-PM-BA$.在模式域更新过程中,首先将$D(Q_{111}, G)$初始化为$D(Q_{11}, G)$,并新增一列(初始为空)对应新引入节点BA,随后利用$PM-BA$模式域中BA节点的匹配$[v_2, v_{11}, v_{16}]$填充该新列.接着逐列验证各节点的有效性:以第一列中的节点v_1为例,已知图G中存在子图$G_s=\{(v_1, v_3), (v_3, v_4), (v_4, v_2)\}$与模式$Q_{111}$匹配,则将$v_1, v_3, v_4, v_2$分别在其对应列中标记为有效.后续的检查将跳过已标记为有效的节点,从而避免重复验证.

3.4　*BwExtension*函数

经过前向扩展构建模式树T后,PMiner调用*BwExtension*函数挖掘非树模式.具体地,*BwExtension*首先初始化一个整数h(初始为p-模式树T的高度),用以跟踪当前的扩展层次(第20行).之后,遍历当前层的所有模式,并使用*BackExt*函数(未展示)逐一扩展这些模式.每次扩展过程中生成的候选模式存储在l_b中(第22行).*BackExt*的工作原理与*PatTreeGen*类似,不同之处在于*BackExt*通过引入新的边来扩展模式,不会引入新的节点.随后,所有满足支持度阈值的模式会被更新到p-模式树T中(第23~26行).处理完当前层的所有模式后,h值减1,进入下一次迭代(第27行).最后,*BwExtension*返回p-模式树T,其中包含了所有频繁p-模式.

例7　图3展示了频繁单边p-模式Q_1:U_x-U_y的完整扩展过程.首先,PMiner利用*FwExtension*对Q_1(Level 1)进行前向扩展,生成一系列的候选模式.经过支持度检测后,得到频繁p-模式Q_{11},Q_{12}和Q_{13}(level 2),其中,Q_{13}是经实例化扩展(*PatInst*)得到的单参p-模式.随后,前向扩展过程继续进行.Q_{11}进一步扩展为Q_{111}和Q_{112}(level 3)以及Q_{1121}(level 4).类似地,Q_{12}被扩展

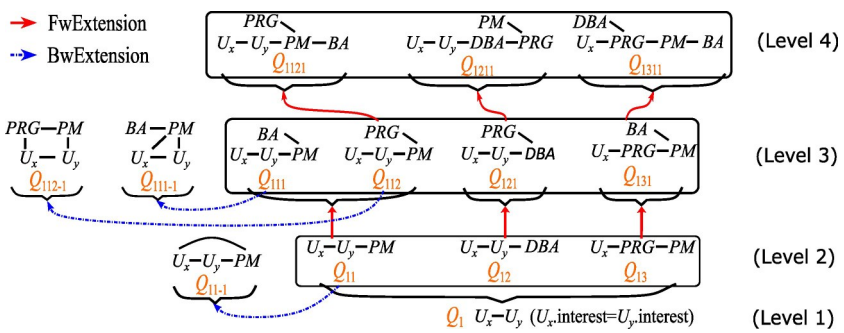


图3　模式Q_1的扩展过程

Fig. 3　**The expansion process of pattern Q_1**

为 Q_{121} 和 Q_{1211}，而 Q_{13} 被扩展为 Q_{131} 和 Q_{1311}. 当第四层模式生成的候选模式不再满足阈值要求，无法更新模式树时，前向扩展终止. 随后，PMiner利用 *BwExtension* 逐层识别非树模式，并将其更新至 T. 例如模式 Q_{11}，Q_{111} 和 Q_{112} 分别生成 Q_{11-1}，Q_{111-1} 和 Q_{112-1}.

4 图关联规则生成算法

规则生成算法GARGen基于频繁模式树框架构建，能够生成高度关联的图模式规则. GARGen接受图 G，模式树 T，置信度阈值 η 作为输入，返回一个GARs集合 S. 在树 T 中，节点 $v(v\in V_T)$ 唯一地索引一个频繁 p-模式，该模式记为 $Q(v)$. 算法伪代码如算法3所示.

算法3 GARGen

输入：图 G，p-模式树 $T=(V_T,E_T)$，置信度阈值 η

输出：GARs集合 S

Step 1. initialize a set $S=\varnothing$, a queue $q=\varnothing$

Step 2. for each node v in T do

Step 3. $q=\varnothing$, push v in q

Step 4. While $q\neq\varnothing$ do

Step 5. $v'=q.pop(\ )$

Step 6. for each $(v'',v'\in E_T)$ do

Step 7. push v'' in q, generate Q_r

Step 8. if Q_r is connected and $\dfrac{Sup(Q(v),G)}{Sup(Q(v''),G)}\geqslant\eta$

Step 9. $S=S\cup\{(Q(v'')\rightarrow Q_r)\}$

Step 10. return S

GARGen首先初始化一个空集合 S 和一个空队列 q(第1行). 随后，GARGen从树中每一个节点 v 开始，通过反向广度优先搜索遍历 T 并生成GAR(第2~9行). 具体地，对于遍历过程遇到的每一个祖先节点 v''，通过从模式 $Q(v)$ 排除与 $Q(v'')$ 对应的边来生成新模式 Q_r，若 Q_r 连通，且 $\dfrac{Sup(Q(v),G)}{Sup(Q(v''),G)}\geqslant\eta$ 成立，则生成一个新的GAR $(Q(v'')\rightarrow Q_r)$ 并将其加入集合 S. 处理完 T 的所有节点后，返回集合 S.

共进行 $|V_T|$ 次反向广度优先搜索，每次反向广度优先搜索最多需要 $O(|V_T|+|E_T|)$ 的时间，则算法的时间复杂度为 $O(|V_T|(|V_T|+|E_T|))$.

例8 回顾例7. 在频繁 p-模式树 T 构建完成后，GARGen遍历模式树中的节点以生成GAR. 以 $\eta=0.5$ 为例，假设遍历从对应模式 $Q_{111\text{-}1}$ 的节点开始，其边集为：

$$\{(U_x,U_y),(U_y,PM),(PM,BA),(U_x,PM)\}$$

且 $Sup(Q_{111\text{-}1},G)=3$. 随后，可以从模式 $Q_{111\text{-}1}$ 索引到其父模式 Q_{111}，其边集为：

$$\{(U_x,U_y),(U_y,PM),(PM,BA)\}$$

且 $Sup(Q_{111\text{-}1},G)=3$. 然后，生成一个先导为 $Q_l=\{(U_x,U_y),(U_y,PM),(PM,BA)\}$，后继为 $Q_r=(U_x,PM)$ 的GAR，并验证该GAR的置信度为1. 因此这是一个有效的GAR.

5 实验结果与分析

使用真实图数据开展实验并与基准算法进行比较以评估PMiner的性能，主要包括两个方面：(1) PMiner在模式挖掘效率上的表现；(2) p-模式在规则生成中的有效性. 实验环境为一台配备2.4 GHz CPU，16 GB RAM的Windows 11主机，所有代码均由Java编写.

5.1 实验设置

5.1.1 实验数据 实验共使用六个真实数据集：(1)谷歌用户共享圈网络图Google+[24]；(2)Twitter网站社交网络图[25]；(3)Pokec在线社交网络图[26]；(4)Facebook社交网络[27]；(5)YouTube共享网络[27]；(6)照片分享网络Flickr[27]. 相关统计信息如表1所示.

表1 数据统计信息

Table 1 Data statistics information

数据集	节点集大小	边集大小	平均度数
Google+	107614	13673453	127.06
Twitter	81306	1768149	21.75
Pokec	1632803	30622564	18.75
Facebook	112640	1713152	15.20
YouTube	1048576	3355443	3.20
Flickr	182944	6291456	34.39

5.1.2　基线算法

(1)T-FSM[9]:一个面向单一大图的频繁模式挖掘的并行系统,采用基于任务的执行引擎,确保高并发、有限的内存消耗和高效的负载均衡.

(2)ScaleMine[4]:一个面向单一大图的频繁子图挖掘的并行系统,其工作分两个阶段.第一阶段,快速识别高概率的频繁子图;第二阶段,使用近似结果计算精确解.

(3)GraMi[3]:一个面向单一大图的频繁子图挖掘框架,通过仅搜索满足阈值的最小实例集,避免了传统方法的高成本.

(4)RuleGen[21]:一个面向单一大图的图模式关联规则挖掘算法,通过频繁模式并行挖掘(无参模式)显著提升运算效率.

(5)Jaccard[28]和SimRank[29]:两种方法都基于节点相似性进行链路预测,但使用了不同的相似性度量.具体地,给定两个节点v_1和v_2,Jaccard通过式(4)来定义节点相似性:

$$Sim_j(v_1,v_2)=\frac{N(v_1)\cap N(v_2)}{N(v_1)\cup N(v_2)} \tag{4}$$

其中,$N(v)$表示节点v的邻居集合.SimRank通过式(5)来定义v_1和v_2的相似性:

$$Sim_s(v_1,v_2)=C\cdot\frac{\sum v_1'\in N(v_1)\sum v_2'\in N(v_2)Sim_s(v_1',v_2')}{k_{v_1}\cdot k_{v_2}} \tag{5}$$

其中,$C\in[0,1]$为衰减因子,k_v为节点v的度数.

5.2　实验结果　在相同条件下评估PMiner与T-FSM,ScaleMine和GraMi的时间和空间性能.由于PMiner与传统FPM算法在挖掘目标上存在差异,进一步对比了挖掘结果,并评估了PMiner所挖掘的p-模式在链接预测方面的表现.

5.2.1　PMiner的时间与内存开销　测试所有挖掘算法在六个真实数据集上的时间和空间开销,如图4和图5所示.在Google+,Twitter,Pokec,Facebook,YouTube和Flickr数据集上变化支持度阈值θ,具体为{0.17K,0.2K,0.01K}{0.56K,0.62K,0.02K}{20K,23K,1K}{0.5K,0.8K,0.1K}{1.3K,1.6K,0.1K}和{1.1K,1.4K,0.1K},其中,阈值设置由三元组{最小值,最大值,增量}表示.实验结果表明了以下结论.

(1)随着支持度阈值θ的增大,所有算法的时间和空间开销都会减小,这是因为候选模式的大幅减少显著降低了子图同构检测和支持度计算的开销.

(2)时间性能,PMiner在Twitter,YouTube和Flickr数据集上表现突出,在Pokec和Facebook数据集上表现适中.尽管PMiner因引入参数需要验证的候选模式数量通常超过传统算法,但其时间性能仍然保持在可接受的水平.

(3)空间性能,PMiner在Google+和Facebook数据集上表现较好,但在其他数据集上的表现不如传统算法.同样是因为PMiner需要处理更多的候选模式,此外PMiner还需考虑节点属性,导致空间开销增加.尽管如此,算法的整体性能仍然保持在合理范围内.

5.2.2　PMiner挖掘模式数量　由于PMiner与传统算法在挖掘目标上的差异,本文在评估了时间和空间开销之后,进一步分析在θ相同时各算法生成的模式结果集,结果如表2所示.在六个数据集上分别将支持度阈值θ设置为(170,200)(560,620)(20000,23000)(500,800)(1300,1600)(1100,1400)进行测试,得出以下结果.

(1)T-FSM和GraMi挖掘的模式数量相同,因为两者都是精确挖掘算法,而ScaleMine在第一阶段采用近似解法,导致其在部分数据集上挖掘的模式数量低于其他两个算法.

(2)在大多数数据集中,PMiner识别出的p-模式数量高于传统算法挖掘出的无参模式数量,但两者的差异不显著.这是因为本文提出的布尔表达式$(U_x.Attribute=U_y.Attribute)$对参数的有效约束,有效避免了引入参数后可能出现的模式数量爆炸问题.此外,Pokec数据集的稀疏结构也导致被发现的p-模式较少.

(3)在YouTube数据集上,PMiner挖掘出的p-模式明显少于传统算法.进一步分析传统算法的挖掘结果,发现其中90%的模式为单边模式,而且在其他数据集的挖掘结果中至少包含50%的单边模式.相比之下,PMiner采用了新颖的扩展策略,其结果中单边模式的比例不到10%.

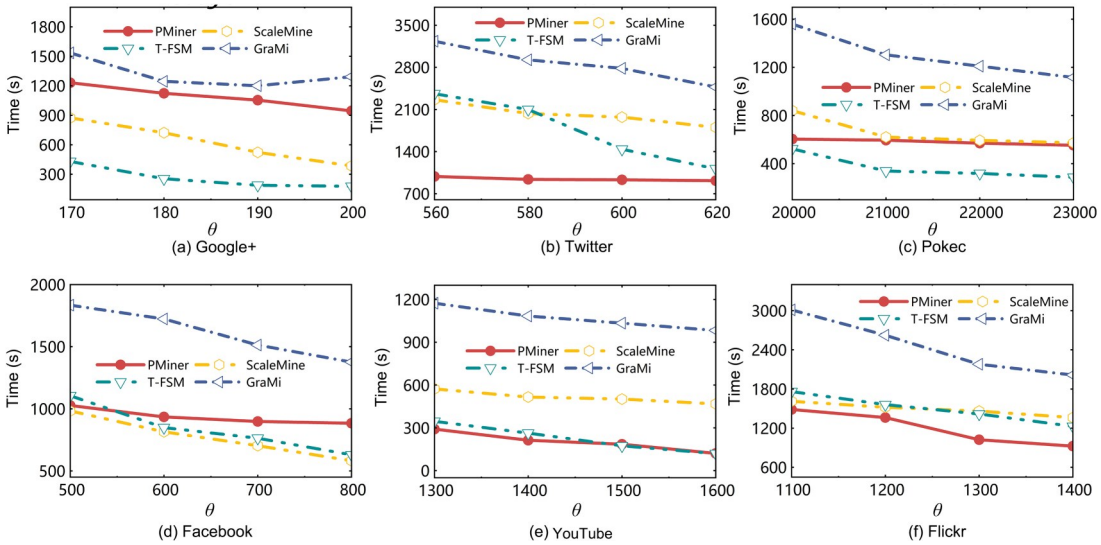


图 4 PMiner 的时间开销

Fig. 4 The time cost of PMiner

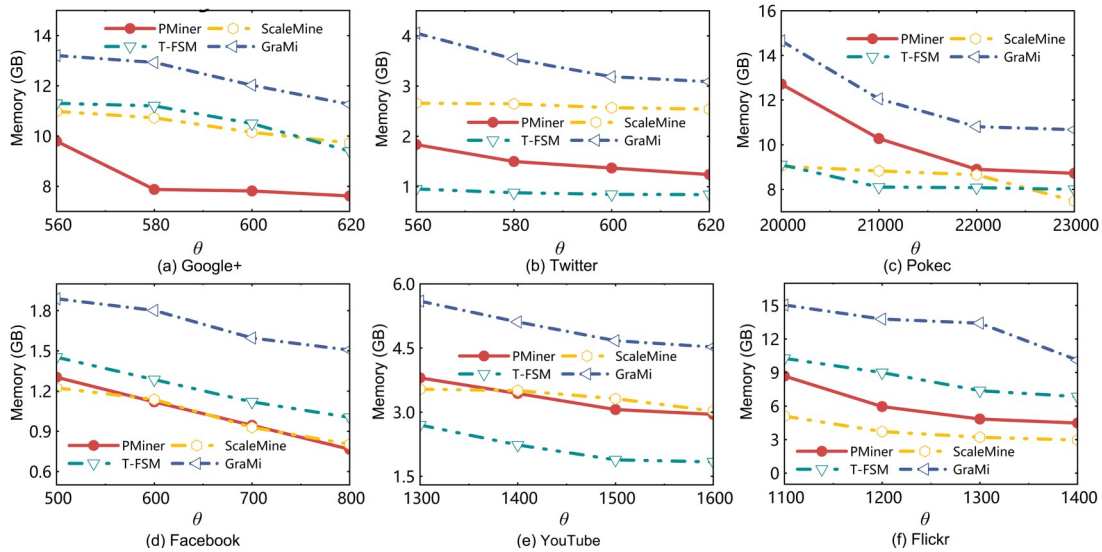


图 5 PMiner 的内存开销

Fig. 5 The memory cost of PMiner

表 2 不同算法挖掘的模式数量

Table 2 The number of patterns mined by different algorithms

	Google+		Twitter		Pokec		Facebook		YouTube		Flickr	
	170	200	560	620	20000	23000	500	800	1300	1600	1100	1400
T-FSM	92	12	275	171	212	98	748	336	1214	1129	248	197
GraMi	92	12	275	171	212	98	748	336	1214	1129	248	197
ScaleMine	92	12	275	171	212	98	646	309	1194	1063	230	139
PMiner	747	413	581	483	77	53	1401	471	703	337	293	144

总体上,PMiner在时间和空间效率上的表现令人满意,并且有效解决了传统算法结果中单边模式冗余的问题. 在θ相同时,PMiner能够挖掘出更多结构多样的模式.

5.2.3　预测准确率　为了研究p-模式在链接预测中的性能,采用GARGen算法基于p-模式生成图关联规则(GAR). 将GARGen与RuleGen,Jaccard和SimRank算法进行比较,以评估它们的预测准确率. 具体设置如下. GARGen与RuleGen的预测准确率通过交叉验证[29]进行评估:给定图G,将其分为两个片段F_1和F_2,从F_1中挖掘并选择前$k(k=10,20,30,40,50)$个GAR,在F_2上评估预测准确率. 准确率的定义为:

$$Acc(R)=\frac{Sup(Q_R,F_2)}{Sup(Q_l,F_2)} \tag{6}$$

此过程需要使用度量指标来评估GAR的"质量"以对其进行排序. 因此,选取Gain[30]和Interest[31]两个经典度量指标来衡量GAR的有趣性.

Jaccard和SimRank的评估使用四重交叉验证. 具体操作如下.

(1)从Google+, Twitter, Pokec, Facebook, YouTube和Flickr中分别提取(5000, 32329)(5000, 22508)(5000, 21301)(5000, 20974)(5000, 21125)和(5000, 21853)的子图G_F(由于SimRank无法在整个图上运行,因此提取子图进行实验).

(2)对于每个G_F,边集被随机分为四个部分,其中一部分作为测试集,其余三部分用于构建训练图. 交叉验证过程重复四次,每个子集仅使用一次作为测试集.

(3)对于每个训练图,运行Jaccard和SimRank算法,基于相似度生成一个节点对的排名列表. 选择前L个节点对作为预测边,预测准确率定义为:

$$Acc=\frac{L_r}{L} \tag{7}$$

其中,L_r为正确预测的边数.

(4)SimRank的参数C设置为0.8,迭代次数设置为5. Gain参数θ设置为0.4.

表3展示了GAR的预测准确率,并与RuleGen,Jaccard和SimRank进行比较. 由表可以得出以下结论.

(1)使用Gain和Interest度量选择前10个GAR进行链接预测时,GARGen在六个真实图上的平均预测准确率分别为65.7%和68.3%. 当设置$k=50$时,平均预测准确率降至59.7%和56.6%. 随着k的增大,准确率$Acc(R)$呈下降趋势. 因为在k较小时,所选的GAR通常具有较高的"有趣性",随着k的增加,更多低"有趣性"模式被包含进来,导致预测准确率的降低.

(2)在选用Gain度量和Interest度量时,GARGen的平均预测准确率分别为64.4%和62.13%,相较于基线算法RuleGen(49.7%和50.27%)平均提升幅度达13.3%. GARGen在链接预测任务上性能明显优于RuleGen,这归因于p-模式和无参模式相比,具备更强的表达能力,能够捕捉实体间更加复杂的关系.

(3)在预测准确率方面,GARGen算法明显优于Jaccard和SimRank($L=200$). 以Pokec数据集为例,当使用Interest度量并且设置$k=50$时,GARGen的预测准确率分别为Jaccard和SimRank的360%和645%.

如图6所示,在Google+数据集中选择一些具有代表性的GAR进行案例分析,从图中可以得出以下结论.

(1)R_1表示如果未知用户U_x与用户U_1是朋友且两者都居住在芝加哥,那么U_1的另一个朋友U_2很可能与U_x是朋友.

(2)R_2表示如果未知用户U_x与用户U_1是朋友且两者都毕业于斯坦福大学,那么U_1的另一个朋友U_2也可能毕业于斯坦福大学.

(3)R_3表示如果未知用户U_x与用户U_1是朋友且两者都毕业于耶鲁大学,且U_1与U_2是朋友,那么U_2也有可能是另一位毕业于耶鲁大学的用户U_y的朋友.

这些规则表明,可以利用GAR进行用户属性预测和朋友推荐.

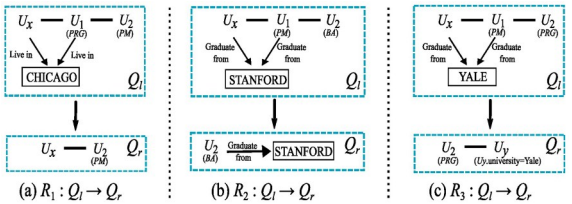
6　结论

本文提出了有参模式,和传统模式相比,它提供了更加丰富的语义表达能力. 还设计了高效的挖掘算法PMiner来发现有参模式. 该算法采用

表 3 预测准确率:GARGen 对 Jaccard 和 SimRank 的预测

Table 3 The prediction accuracy of GARGen vs. Jaccard and SimRank

		Google+					Twitter				
		10	20	30	40	50	10	20	30	40	50
GARGen	Gain	69.2%	58.8%	56.1%	54.2%	53.1%	66.2%	66.0%	64.0%	62.7%	62.0%
	Interest	61.2%	59.4%	58.5%	56.7%	55.1%	66.2%	64.0%	62.0%	60.0%	59.8%
RuleGen	Gain	53.7%	52.8%	48.5%	44.5%	42.4%	54.3%	53.2%	50.1%	48.3%	48.0%
	Interest	51.3%	49.3%	46.0%	44.8%	42.4%	56.1%	54.9%	54.7%	52.4%	51.9%
		10	50	100	150	200	10	50	100	150	200
Jaccard		27.5%	25.0%	36.0%	28.8%	36.6%	12.5%	14.5%	15.3%	23.5%	30.3%
SimRank		17.5%	29.0%	21.0%	27.0%	30.8%	15.0%	20.5%	17.8%	20.2%	19.1%
		Pokec					Facebook				
		10	20	30	40	50	10	20	30	40	50
GARGen	Gain	68.9%	68.8%	68.5%	67.3%	67.1%	67.4%	55.7%	49.8%	44.7%	41.2%
	Interest	83.3%	80.5%	79.2%	74.6%	73.5%	67.9%	55.4%	49.0%	44.7%	41.6%
RuleGen	Gain	55.3%	54.8%	54.1%	53.5%	52.9%	49.5%	48.5%	47.7%	46.0%	44.3%
	Interest	60.4%	58.9%	58.2%	56.7%	56.0%	49.3%	47.4%	46.6%	44.0%	43.5%
		10	50	100	150	200	10	50	100	150	200
Jaccard		12.5%	21.5%	23.7%	21.3%	20.4%	10.0%	21.5%	19.5%	24.7%	24.5%
SimRank		20.0%	13.5%	12.3%	8.5%	11.4%	7.5%	15.0%	14.5%	18.3%	16.3%
		YouTube					Flickr				
		10	20	30	40	50	10	20	30	40	50
GARGen	Gain	63.6%	62.6%	62.0%	61.6%	61.2%	58.8%	58.7%	58.6%	58.4%	56.9%
	Interest	63.5%	62.6%	62.0%	61.6%	61.2%	67.6%	61.0%	55.3%	51.9%	48.6%
RuleGen	Gain	53.2%	52.5%	50.3%	49.2%	48.9%	48.6%	48.0%	46.8%	45.7%	45.0%
	Interest	55.4%	52.1%	50.4%	49.6%	47.8%	49.2%	48.4%	47.5%	46.6%	46.2%
		10	50	100	150	200	10	50	100	150	200
Jaccard		27.5%	25.0%	36.0%	28.8%	36.6%	27.5%	23.5%	19.8%	13.8%	16.2%
SimRank		17.5%	29.0%	21.0%	27.0%	30.8%	7.5%	16.5%	17.0%	14.5%	17.9%



(a) $R_1: Q_l \to Q_r$ (b) $R_2: Q_l \to Q_r$ (c) $R_3: Q_l \to Q_r$

图 6 真实场景中的 GARs:Google+

Fig. 6 Real-life GARs:Google+

基于树的分层扩展策略,确保算法具备提前终止的优良性质. 此外,基于 p-模式扩展图关联规则(GAR),实现了缺失结构的预测. 实验结果验证了以上算法性能和有效性的优越性,为大规模图数据分析提供了一种有效工具.

未来将进一步探讨挖掘技术,如设计更加有效的剪枝方法以降低运算开支. 此外,模式重要性的度量问题也是一个值得深入研究的方向.

参考文献

[1] Daud N N, Ab Hamid S H, Saadoon M, et al. Applications of link prediction in social networks: A review. Journal of Network and Computer Applications, 2020, 166: 102716.

[2] Fan W F, Wang X, Wu Y H, et al. Association rules with graph patterns. Proceedings of the VLDB Endowment, 2015, 8(12): 1502－1513.

[3] Elseidy M, Abdelhamid E, Skiadopoulos S, et al. GraMi: Frequent subgraph and pattern mining in a single large graph. Proceedings of the VLDB Endowment, 2014, 7(7): 517－528.

[4] Abdelhamid E, Abdelaziz I, Kalnis P, et al. ScaleMine: Scalable parallel frequent subgraph mining in a single large graph//Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis. Salt Lake City, UT, USA: IEEE, 2016: 716－727.

[5] Hussein R, Lerner A, Ryser A, et al. GraphINC: Graph pattern mining at network speed. Proceedings of the ACM on Management of Data, 2023, 1(2): 1－28.

[6] Jyoti, Kailasam S, Buzmakov A. A scalable, distributed framework for significant subgroup discovery. Knowledge-Based Systems, 2024, 284: 111335.

[7] Jamshidi K, Mahadasa R, Vora K. Peregrine: A pattern-aware graph mining system//Proceedings of the 15th European Conference on Computer Systems. Heraklion, Greece: Association for Computing Machinery, 2020: 1－16.

[8] Yuan L H, Yan D, Qu W W, et al. T-FSM: A task-based system for massively parallel frequent subgraph pattern mining from a big graph. Proceedings of the ACM on Management of Data, 2023, 1(1): 1－26.

[9] 汤小春,樊雪枫,周佳文,等. 基于数据流的大图中频繁模式挖掘算法研究. 计算机学报, 2020, 43(7): 1293－1311.

[10] Cheng W S, Lin Y T, Huang P Y, et al. A fast and highly scalable frequent pattern mining algorithm. Future Generation Computer Systems, 2024, 160: 854－868.

[11] 郭方方,王欣悦,王慧强,等. 基于最大频繁子图挖掘的动态污点分析方法. 计算机研究与发展, 2020, 57(3): 631－638.

[12] Ingalalli V, Ienco D, Poncelet P. Mining frequent subgraphs in multigraphs. Information Sciences, 2018, 451/452: 50－66.

[13] Islam M A, Ahmed C F, Alam M T, et al. Graph-based substructure pattern mining with edge-weight. Applied Intelligence, 2024, 54(5): 3756－3785.

[14] Min F, Zhang Z H, Zhai W J, et al. Frequent pattern discovery with tri-partition alphabets. Information Sciences, 2020, 507: 715－732.

[15] Wu Y X, Geng M, Li Y, et al. HANP-Miner: High average utility nonoverlapping sequential pattern mining. Knowledge-Based Systems, 2021, 229: 107361.

[16] Wang X, Xu Y, Zhan H Y. Extending association rules with graph patterns. Expert Systems with Applications, 2020, 141: 112897.

[17] Liu X L, Dong B W, Fu W Z, et al. Extending graph rules with oracles. Proceedings of the VLDB Endowment, 2024, 17(7): 1775－1787.

[18] 焦鹏飞,吴子安,刘欢,等. 基于多视图对比学习的动态图链接预测方法. 南京大学学报(自然科学), 2024, 60(3): 383－395.

[19] Benhidour H, Almeshkhas L, Kerrache S. Link prediction in directed complex networks: Combining similarity-popularity and path patterns mining. Applied Intelligence, 2024, 54(17): 8634－8665.

[20] Wang X, Lan Z, He Y A, et al. A cost-effective approach for mining near-optimal top-k patterns. Expert Systems with Applications, 2022, 202: 117262.

[21] Lin P, Song Q, Shen J L, et al. Discovering graph patterns for fact checking in knowledge graphs// Database Systems for Advanced Applications. Cham: Springer, 2018: 783－801.

[22] Diaz-Garcia J A, Ruiz M D, Martin-Bautista M J. A survey on the use of association rules mining techniques in textual social media. Artificial Intelligence Review, 2023, 56(2): 1175－1200.

[23] Gong N Z, Xu W C, Huang L, et al. Evolution of social-attribute networks: Measurements, modeling, and implications using Google+//Proceedings of the 2012 Internet Measurement Conference. New York, NY, USA: Association for Computing Machinery, 2012: 131－144.

[24] Leskovec J, Mcauley J. Learning to discover social circles in ego networks. Advances in Neural Information Processing Systems, 2012, 1: 539－547.

[25] Takac L, Zabovsky M. Data analysis in public social networks//International Scientific Conference and

International Workshop Present Day Trends of Innovations 2012. Łomża:[s.n],2012:1－6.

[26] Rossi R A, Ahmed N K. The network data repository with interactive graph analytics and visualization//Proceedings of the AAAI Conference on Artificial Intelligence. Austin, TX, USA: AAAI Press,2015:4292－4293.

[27] Lü L Y, Zhou T. Link prediction in complex networks: a survey. Physica A: Statistical Mechanics and its Applications,2011,390(6):1150－1170.

[28] Jeh G, Widom J. SimRank: A measure of structural-context similarity//Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, NY, USA: Association for Computing Machinery, 2002: 538－543.

[29] Galárraga L A, Teflioudi C, Hose K, et al. AMIE: Association rule mining under incomplete evidence in ontological knowledge bases//Proceedings of the 22nd International Conference on World Wide Web. New York, NY, USA: Association for Computing Machinery,2013:413－422.

[30] Bayardo R J, Agrawal R. Mining the most interesting rules//Proceedings of the 5th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. New York, NY, USA: Association for Computing Machinery,1999:145－154.

[31] Brin S, Motwani R, Silverstein C. Beyond market baskets: Generalizing association rules to correlations//Proceedings of the 1997 ACM SIGMOD International Conference on Management of data. New York, NY, USA: Association for Computing Machinery,1997:265－276.

(责任编辑 杨可盛)