文章编号: 1000-5641(2019)06-0073-15

单图中的近似频繁子图挖掘算法

窦建凯[1], 林 欣[1], 胡文心[2]

(1. 华东师范大学 计算机科学与技术系, 上海 200062;
2. 华东师范大学 计算中心, 上海 200062)

摘要: 图数据的挖掘工作是数据挖掘工作中的重要组成部分, 已经有许多人在这个领域进行了深入的研究. 由于数据获取不可避免噪音数据, 故在挖掘频繁图时考虑近似十分重要. 然而许多此前的工作只考虑了子图间编辑距离(Graph Edit Distance, GED)的绝对值, 而没有考虑子图间编辑距离与子图大小的相对关系. 提出了一种在单图中进行近似频繁子图挖掘的新算法, 并在计算近似程度时考虑当前子图的大小. 该算法通过对近似频繁子图的大小上限进行预测, 并通过局部反单调性进行剪枝, 提高了算法的效率. 实验表明, 该算法能够挖掘出传统算法无法发现的近似频繁子图, 且相比对比算法具有更好的时间性能.

关键词: 近似; 图; 频繁子图挖掘; 剪枝

中图分类号: TP391.4 **文献标志码**: A **DOI:** 10.3969/j.issn.1000-5641.2019.06.008

Algorithm for mining approximate frequent subgraphs in a single graph

DOU Jian-kai[1], LIN Xin[1], HU Wen-xin[2]

(1. *Department of Computer Science and Technology*, *East China Normal University*, *Shanghai* 200062, *China*;
2. *Computer Center*, *East China Normal University*, *Shanghai* 200062, *China*)

Abstract: Graph mining is an important part of data mining, and significant research has been dedicated to the field. Due to the inevitability of noise during data acquisition, it is crucial to address the issue of approximation in mining frequent subgraphs. Previous work has only considered the absolute value of the graph edit distance (GED); however, the relative value between the GED and the size of the subgraph should also be considered. Hence, in this paper, we propose a novel algorithm to mine approximate frequent subgraphs in a single graph; this algortihm takes into account the size of current subgraphs for the calculation of approximations. We increase the efficiency of this algorithm by estimating the upper bound of frequent subgraphs, and pruning according to local anti-monotonicity. Experimental results show that this algorithm can find subgraphs that are missed by traditional mining algorithms, and our proposed approach is relatively efficent compared to other algorithms.

收稿日期: 2018-12-21

第一作者: 窦建凯, 男, 硕士研究生, 研究方向为频繁图挖掘. E-mail: yirandjk@163.com.

通信作者: 胡文心, 女, 高级工程师, 硕士生导师, 研究方向为计算机科学.E-mail:wxhu@cc.ecnu.edu.cn.

Keywords: approximate; graph; frequent subgraph mining; pruning

0 引 言

图是用来表示数据的一种特殊数据结构, 它不仅可以用来表示实体本身的性质, 同时还可以用来表示实体之间的关系. 图的这种特点使得图在多种领域具有广泛应用, 如生物信息学、网络分析等. 随着社会和科学的发展, 实体之间的关系越来越多样, 实体的数量越来越多, 使用图结构来表示实体间的关系显得尤为高效, 从图数据中挖掘实用的模式亦显得越来越重要.

然而图结构虽然能高效地表示实体间的关系, 但其结构的复杂性使得从图中识别频繁的子图也变得困难. 如在挖掘有用子图的过程中, 同构图的识别问题, 就被认为是一个NP(Non-deterministic Polynomia)完全问题. 因此需要高效的频繁子图挖掘的方法.

现有的从图结构数据中挖掘频繁子图的方法主要分为两种: 准确频繁图挖掘和近似频繁图挖掘. 准确频繁图挖掘指从给定图或图数据库中挖掘频繁子图, 当记录每个子图的支持度时, 只计算与当前子图完全一致的同构图的数量, 如 gSpan[1]和 Grami[2]; 而近似频繁图挖掘与准确频图挖掘的主要区别为, 除了完全一致的同构图, 还同时记录与当前子图有一定区别的同构图的数量, 如 gApprox[3]与 AGraP[4].

由于获取数据的过程中, 不可避免数据丢失或噪音, 因此在实际应用中, 近似频繁子图挖掘更加符合用户需求. 已有的近似频繁子图挖掘算法在计算两个给定图的近似程度时, 只关注其近似程度的绝对值, 即从一个图转化为另一个图需要的操作数的绝对值; 而在实际应用中, 在衡量两个图的近似程度时, 图本身的大小也是非常重要的衡量指标.

针对以上现有的近似频繁子图挖掘算法的问题, 本文提出了子图大小相关的近似频繁子图算法. 该算法改进了子图近似程度的计算方式, 使得近似程度的结果与当前候选子图的大小有关. 算法首先通过一个基于深度优先搜索策略的算法, 以及基于给定频繁程度和近似程度的子图大小上限算法, 生成所有符合大小要求的候选子图; 然后对每个候选子图生成一个最小的、需要搜索的近似子图集合, 并对其中的每个近似子图进行图同构搜索; 最后计算所有匹配的同构图中, 互斥的同构图的数量并将其作为子图的近似支持度.

本文算法的主要难点有: ①随着给定图的大小增加, 候选子图的数量呈指数增加, 因此需要一个有效的搜索方式及 early-stop 的条件; ②对任意一个候选子图, 其近似子图的数量随候选子图的大小呈指数式增加, 且由于图同构问题是一个 NP 完全问题, 因此高效的识别同构图十分困难; ③计算所有近似匹配中互斥匹配的数量, 及候选子图的近似支持度的问题, 可以转化为 MIS(Maximum Independent Set)问题[5], 而 MIS 问题是一个公认的 NP 完全问题, 因此计算子图的近似支持度是一个重要且困难的问题.

基于以上算法的主要难点, 本文的主要贡献如下.

(1) 基于深度优先搜索策略的候选子图生成算法、候选子图大小上限的计算方法.

(2) 提出了一个基于点和边的删除尝试策略的近似子图生成算法、高效的图匹配算法.

(3) 根据近似频繁子图的特点, 利用候选子图的支持度上限, 来代替候选子图的准确支持度, 极大地提高了算法的效率.

(4) 实验证明, 与对比算法相比, 本文算法不仅高效, 同时能发现对比算法无法发现的近似频繁子图.

本文的结构如下: 第 1 节介绍一些与本文提出算法及目标相关的工作; 第 2 节介绍一些相关概念; 第 3 节到第 5 节详细介绍本文算法的 3 大模块; 第 6 节通过实验展现本文算法的有效性; 第 7 节给出本文的总结与展望.

1　相关工作

频繁子图挖掘指的是, 给定一个频繁程度下限, 从给定的图或图集合中挖掘出全部的、符合频繁程度要求的子图. 因此频繁子图挖掘问题主要分为两种: 单图中的频繁子图挖掘和图集合上的频繁子图挖掘. 图集合上的频繁子图挖掘已经有许多成熟的工作, 如, gSpan 算法[1], 在图集合中挖掘频繁子图, 避免了候选子图的生成; Gaston 算法[6]修改了搜索策略, 从而使得算法速度加快. 类似的算法还有 gRed[7]和 GraphSig[8]. 这些算法的不同及其优缺点在 Cheng 等的文章[9]和 Krishna 等的文章[10]中有详述的论述.

近年来, 随着数据规模的增大和需求的不断变化, 一些特殊的频繁子图挖掘算法也不断被提出. 如, Choudhury 等研究了在动态不断变化的图中挖掘可扩展的子图[11], 他们提出的算法可以随着给定图的变化, 动态地判定某个子图是否频繁; Ingalalli 等研究了在多重图中的频繁子图挖掘[12], 其所提出的算法的目的是适应不断丰富且复杂化的数据, 除此之外, 该算法在应用于简单图上时, 也能取得良好的效果.

虽然单图中的频繁子图挖掘工作已经在多种应用中被提及, 如 Alguliev 等的工作[13]和 Lima 等的工作[14], 以及 Elseidy 等人提出的 Grami 算法[2], 都能有效地从单图中挖掘频繁子图. 但对单图中的频繁子图挖掘的研究, 仍然少于图集合中的频繁子图挖掘的研究. 研究表明, 单图中的频繁子图挖掘算法可以通过简单修改, 应用于图集合上; 然而图集合上的算法并不能应用于单图中[5].

近似频繁子图挖掘是在频繁子图挖掘的基础上, 当计算一个子图的支持度时, 除了与该子图完全一致的匹配, 同时计算满足一定近似程度的匹配. 近似程度的计算通常有图的编辑距离[15]、结构区别[16]、特性区别[17]等几种方式. 已经有一些近似频繁子图挖掘方面的工作, 如 Holder 等在 SUBDUE 上设计了一种变形函数[18], 使计算支持度时可以计算有一定区别的近似匹配; Chen 等的 gApprox 在无标签的图上, 设计了函数计算图之间的编辑距离[18]; Jia 等的 APGM 通过相容性矩阵计算图之间的距离, 从而允许子图的近似匹配与子图之间可以有点的区别[19]. AGraP[4]算法在此基础上进行了改进, 从而允许近似匹配与子图之间拥有点和边上的区别, 包括缺失和替换等. Flores-Garrido 等在 AGraP 上进行了修改, 使算法挖掘全部近似频繁子图的子集 CloseAFG[20].

除此之外, Acosta-Mendoza 等研究了在多重图上的近似频繁子图挖掘[21], 并利用范式的思想, 使算法在时间成本和可扩展性上都有良好的效果. 同时, Acosta-Mendoza 等利用近似频繁子图挖掘算法, 设计了新的图像分类算法[22], 他们利用近似频繁子图挖掘算法的结果, 作为图像分类中图像的特征, 从而实现了在过程中自动计算和获取需求的替换矩阵, 并获得了令人满意的结果, 再次证明了近似频繁子图算法具有实际的意义. 同时, Acosta-Mendoza 等也对近似频繁子图挖掘算法的结果, 在图像聚类方面的应用做了研究[23], 并通过实验表明, 以近似频繁子图作为聚类的特征, 同样可以提升聚类的效果.

然而已有的近似频繁子图的挖掘算法, 在计算近似程度时, 只考虑了编辑距离的绝对值, 而忽略了图大小对近似程度的影响. 因此, 本文提出了子图大小相关的近似程度计算方式, 并在此基础上设计了算法挖掘符合要求的近似频繁子图.

2 基本概念

定义 1 标签图 一个标签图 G 是一个四元组 $G=(V,E,L,f)$, V 表示图中所有点的集合, E 表示边的集合, L 是所有标签的集合, $f:V,E\rightarrow L$是一个映射函数, 将 L 中的标签映射到每个点和边上.

这是一个被广泛应用的定义, 图中的点表示实体, 边表示实体之间有一定关系, 点上的标签表示实体的属性或类别, 边上的标签表示的是实体之间关系的类别. 如在蛋白质相互作用图(PPI: Protein-Protein Interaction)中, 点表示蛋白质, 点上的标签可以表示蛋白质种类. 因此有许多点具有相同的标签, 边表示蛋白质之间可以发生相互作用, 边的标签表示相互作用的种类.

定义 2 图同构 给定两个图 $g=(V,E,L,f)$ 和 $g'=(V',E',L',f')$, 图同构是一个映射$F:V\rightarrow V'$, 并且满足以下条件: ①$\forall u\in V,F(u)\in V'$且$f(u)=f'(F(u))$; ②$\forall(u,v)\in E,(F(u),F(v))\in E'$且$f(u,v)=f'(F(u),F(v))$. 如果图 g' 中包含图 g, 则 g 是 g' 的子图.

定义 3 近似程度θ 给定一个图 g, g 的子图 g' 与 g 的近似程度 θ 为: g' 与 g‘相比缺失的个数或边的条数与 g' 的大小的比值, 即

$$\theta=\frac{|g|-|g'|}{|g|},\tag{1}$$

其中 $|g|$ 和 $|g'|$ 分别表示两个图的大小, 即图中包含的点和边的数量之和.

定义 4 近似匹配 给定两个图 $g=(V,E,L,f)$ 和 $G=(V',E',L',f')$, 图的近似程度阈值 θ', 当且仅当 g' 是 g 的子图, 并且 g' 与 g 的近似程度 $\theta\geqslant\theta'$, 且 g' 是 G 的子图, 则认为 g' 是 g 在图 G 中的一个近似匹配.

如图 1 所示, 给定图 g (图1(a)) 和图 G (图 1(b)), $\theta=0.8$, 图 1(c) 中所有子图是图g在图G中的近似匹配.

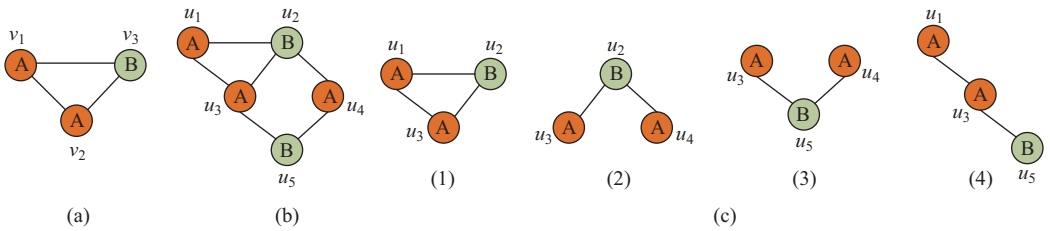


图 1 近似匹配例子

Fig. 1 Example of approximate matches

定义 5 子图的近似支持度δ 给定图 G 和近似程度下限 θ, 图 g 在图 G 中的两个近似匹配如果不包含任何相同的点, 则说这两个近似匹配是互斥的. 图 g 在图 G 中的支持度 δ 指图 g 能在图 G 中找到的互斥的近似匹配的最大数.

通过以上定义, 可以对单图中的近似频繁子图挖掘问题给出形式化定义: 给定单图 G, 支持度下限 δ 和近似程度下限 θ, 单图中的近似频繁子图挖掘的目的是, 找出图 G 的所有近似支持度 $\delta'\geqslant\delta$ 的子图.

定义 6 置信度ρ 给定图 G 和近似程度下限 θ, 图 g 和图 g' 是图 G 中的子图, 其近似支持度分别为 θ_1 和 θ_2, 且 g 是 g' 的子图, 则置信度 ρ 是指在子图 g 表示的实体间的关系成

立的前提下, 子图 g' 表示的实体间关系成立的概率, 即 $\rho = \frac{\theta_2}{\theta_1}$.

例如图 1 中图 G (图1(b)), 若设 $\delta = 2$ 且 $\theta = 0.8$, 图 G 的所有近似频繁子图如图 2 所示.

图 2　图 G 的近似频繁子图

Fig. 2　Approximate frequent subgraphs from graph G

本文算法的框架如图 3 所示, 算法主要分为 3 个部分: ① 第一部分遍历给定图的所有候选子图, 由于子图数量过多, 此处通过预测来确定可能的频繁子图的大小上限, 提高遍历速度. ② 算法第二部分对每个子图生成所有的近似匹配, 由于当近似程度下限为1时, 寻找近似匹配的问题实际上是子图同构问题, 二子图同构问题被广泛认为是一个 NP-complete 的问题, 为了降低计算成本, 算法利用生成和搜索过程, 在生成阶段, 对候选子图的每个结点和边进行尝试删除, 生成所有符合近似要求, 且不互相包含的近似子图; 在搜索过程, 对每个近似子图进行查询. ③ 第三部分主要对子图的近似支持度进行计算. 本文算法的原理和详细过程将在下面几节中详细阐述.

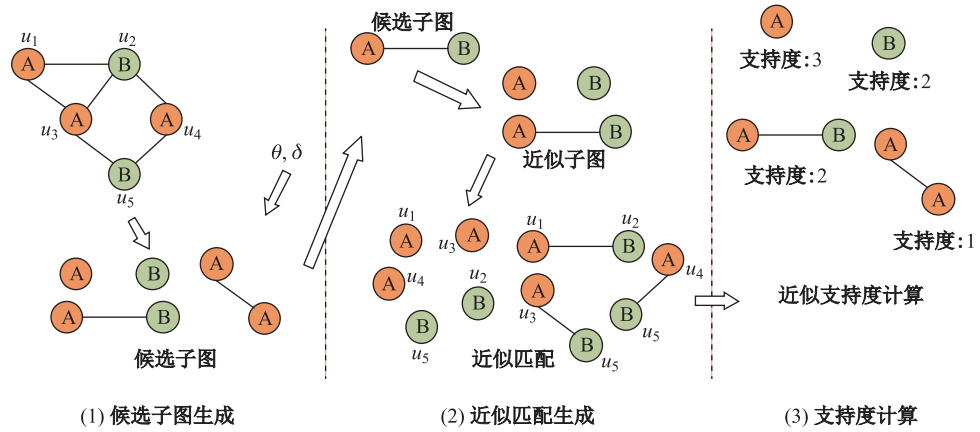


图 3　算法主要框架

Fig. 3　Framework of our algorithm

3　候选子图生成

定理 1　候选子图大小上限　对给定图 G, 支持度下限 δ 和近似程度下限 θ, 有可能成为近似频繁子图的子图 g 的大小上限为

$$|g| \leqslant \frac{|G|}{\theta \times \delta} \tag{2}$$

证　明　由近似程度和近似匹配的定义可得, 子图 g 的最小近似匹配 g' 的大小为: (a) $|g'| \geqslant |g| \times \theta$, 通过近似支持度的定义可知, 当且仅当子图 g 拥有至少 δ 个互斥的近似匹配, 才可以被称作近似频繁子图, 因此: (b) $|g'| \times \delta \leqslant |G|$, 结合(a)(b)两式, 可得$|g| \times \theta \leqslant \frac{(|G|)}{\delta}$, 即$|g| \leqslant \frac{|G|}{\theta \times \delta}$.

通过定理 1, 本文设计了一个基于广度优先搜索的策略. 具体策略步骤如下.

(1) 从当前图 G 中任选一个点 1, 找出所有包含该点, 且大小符合要求的子图.

(2) 从图 G 中移除点 1, 然后回到步骤(1), 任选另一个点开始, 直到图 G 中没有点.

为了实现策略中的步骤(1), 本文设计了以下算法, 具体步骤如下.

(1) 如图 4 所示, 从点 1 开始, 将点 1 标记为“必须的”.

(2) 以当前子图为起点, 标记其所有邻居结点为被标记的, 如图 4 中的点 1,2,3.

(3) 使用邻居结点依次扩展当前子图, 获得的子图依次为 01,02,03.

(4) 每获得一个新子图, 判断其大小是否超出候选子图大小上限. 若不超出, 以当前子图为新起点, 重复步骤(2).

(5) 若获得的子图大小超出候选子图大小上限, 则终止当前扩展策略, 尝试扩展下一个被标记的邻居结点.

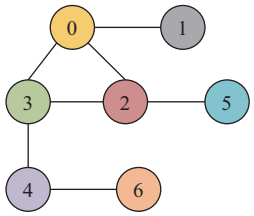
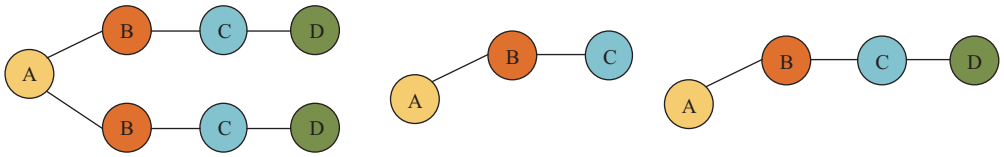


图 4 候选子图扩展例图

Fig. 4 Example graph of candidate graph generation

图 4 中, 不考虑最小近似程度和支持度, 所有获得的子图按顺序应为: 0, 01, 012, 0123, 01234, 012345, 0123456, 012346, 01235, 0125·····

反单调性是提高挖掘效率的重要条件. 在频繁图挖掘中, 反单调性是指, 给定图 G 中, 若图 g' 是图 g 的子图, 且图 g' 与图 g 这两图都是图 G 的子图, 则图 g 在 G 中的支持度必然不超过 g' 在 G 中的支持度. 在利用近似程度的绝对值, 即点或边的缺失数量作为衡量指标时, 反单调性是满足的. 如图 5 中, 若设最小支持度为 3, 最多缺失点或边的数量为 2, 则图 5(a) 和图 5(b) 在图 G 中都是近似频繁的. 但本文中, 反单调性并不能全局保持. 如图 5 中, 图 5(a) 是图 5(b) 的子图, 但若设 $\delta = 2, \theta = 0.7$, 由于图 5(a) 的近似匹配可以允许最多 1 个点或边的缺失(由于任何点或边的缺失都会造成图的不连通, 因此(a)没有任何近似子图), 而图 5(b) 可以允许最多两个点或边的缺失, 因此图 5(b) 拥有更多的近似匹配(如删除点 A 及相连的边), 图 5(b) 是一个近似频繁子图, 而图 5(a) 不是.



G (a) (b)

图 5 反单调性无法保持示意图

Fig. 5 Example when anti-monotonicity does not hold

本文提出部分反单调性的概念, 定理如下.

定理 2 局部反单调性 给定图 g 和近似程度 θ, 图 g' 是图 g 的子图, 在满足近似程度的前提下, 若图 g 可允许的点和边的缺失数量, 等于图 g' 可允许的点和边的缺失数量, 即 $0 \leqslant (|g| - |g'|) \times (1 - \theta) < 1$, 则 g' 的支持度不超过 g 的支持度.

证　明　当子图 g' 和图 g 允许缺失的点或边的数量相同时, 图 g 的任意近似匹配必然是图 g' 的某个近似匹配的超图, 而 g' 的互斥近似匹配的数量, 即 g' 的支持度, 必然不超过其互斥的超图的数量, 即 g 的支持度.

根据局部反单调性, 可以在前述算法上继续剪枝, 即在使用当前子图做新输入时, 若当前子图的支持度不符合最小支持度的要求, 则可以直接扩展当前子图, 而不需要判断其是否频繁, 直到可允许的点或边的数量发生改变. 因此, 不需要对每个候选子图生成近似匹配和支持度的计算, 从而可以节省大量时间成本. 同时, 由局部反单调性的证明可知, 当两个子图之间的关系符合局部反单调性的要求, 则其中较大的图的所有近似匹配, 必然是其子图的某个近似匹配的超图, 因此可以通过记录子图的所有近似匹配, 减少超图的近似匹配的搜索范围, 从而节省大量时间成本. 改进后算法流程如图 6 所示.

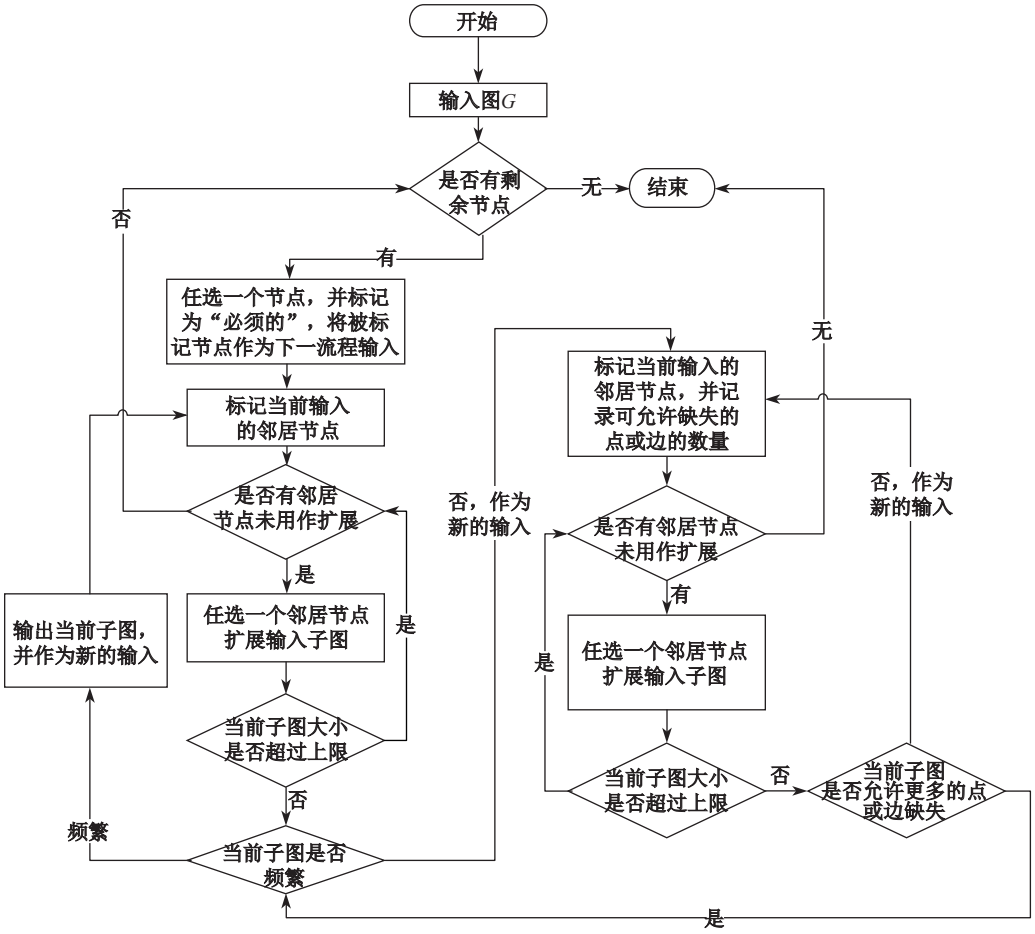


图 6　候选子图生成流程示意

Fig. 6　Process of candidate subgraph generation

4　近似匹配生成

4.1　近似子图生成

定义 7　近似子图　给定图 g, 近似程度 θ, 当且仅当 g 的子图 g' 与 g 的相似度超过 θ, 即 $\theta \leqslant \frac{|g|-|g'|}{|g|}$, 则称 g' 是 g 的一个近似子图.

为了查询每个候选子图的近似支持度, 需要针对每个候选子图, 找出其在图中的所有近似匹配. 为此提出了一个定理, 并在此基础上设计了一个基于点和边的删除的生成算法, 来找出候选子图的所有符合近似条件的子图.

定理 3 给定图 G 和近似程度 θ, 图 g 是 G 的子图, 图 a 和图 b 是 g 的子图, a 是 b 的子图, 则任意 b 在 G 中的匹配, 都与至少一个 a 在 G 中的匹配有重复点.

证 明 任意 b 在 G 中的匹配, 是 b 的一个同构图, 则 a 是此匹配的一个子图, 因此此匹配包含 a, 即包含一个 a 在 G 中的匹配, 因此任意 b 在 G 中的匹配, 都与至少一个 a 在 G 中的匹配有重复点.

根据定理 3, 可以确定对任意候选子图, 其最小需要进行匹配的近似子图集合中, 应不含两个图具有包含关系.

基于以上定理, 本文设计了一个基于点和边的删除的子图生成算法, 生成给定候选子图的所有符合近似条件的子图, 其基本思路如下, 具体算法见算法 1.

(1) 对所有点和边进行编号, 并按编号排序, 计算给定候选子图 g 在给定的近似条件 θ 下, 可以允许的缺失的点或边的最大数量, 即$|g| \times (1-\theta)$.

(2) 遍历所有编号, 尝试删除编号对应的点或边, 并判断结果是否连通. 若连通, 则将编号对应的点或边标记为已删除, 并更新当前已删除的点或边的数量. 若不连通, 则跳过当前编号, 继续尝试.

(3) 以步骤(2)的结果为新的起点, 继续遍历剩余编号, 直至已删除的点或边的数量到达最大数量, 或剩余任意点或边的删除都会导致数量超出最大值, 则当前结果即为一个近似子图.

(4) 跳过最后一个被删除的点或边, 继续步骤(2), 直到所有点或边的组合删除都被尝试过.

步骤(3)的所有结果, 即为最终需要进行匹配的候选子图的近似图, 由于每个子图被删除的点或边的数量不超出最大值, 保证了所有结果都是候选子图的近似图; 同时, 由于任意一个子图都删除了允许删除的最多的点或边, 保证了任意两个结果不具有包含关系.

Algorithm 1 SimilarGraphGene

Input: $V_g, E_g, T, d = |g| \times (1-\theta)$

Output: All Similar Graph of g

1: $S \leftarrow V_g, E_g,$
2: **for** each vertex or edge x in S **do**
3: Deleted $\leftarrow x$
4: **if** x is a vertex **then**
5: Deleted $\leftarrow$ every edge connected to x
6: **end if**
7: m=isconnected(Deleted, V_g, E_g)
8: **if** $m =$ true **then**
9: $k = |T|$
10: $V_{g'} = V_g$ Deleted$_V$

11:　　$E_{g'} = E_g$ $\ominus$ Deleted$_E$
12:　　**if** |Deleted| $< d$ **then**
13:　　SimilarGraphGene($V_{g'}$, $E_{g'}$, $T, d -$ |Deleted|)
14:　　**if** $k = |T|$ **then**
15:　　$T \leftarrow g$
16:　　**end if**
17:　　**end if**
18:　　**if** |Deleted| $= d$ **then**
19:　　$T \leftarrow g'$
20:　　**end if**
21:　**end if**
22:　undo line 3-5
23: **end for**
24: **return** T

4.2　近似子图匹配

为了对每个候选子图, 生成对应的近似匹配, 本文设计了一种在单图中查询给定子图的所有同构图的算法. 给定图$G = (V, E, L, f)$, 目的是找到一个合适的点的顺序, 使得查找过程的时间花费最小. 即在第 i 步, 需要从未找到匹配的点中, 选择一个点, 使得该点一旦找到匹配, 则已被匹配的图的大小增大最多, 或使得可能匹配的数量减小到最小.

为了完成上述要求, 需要对所有的可能匹配进行最大的约束, 使得算法能尽早过滤掉错误匹配. 为此, 本文设计了一种贪心算法, 为给定图中的所有点, 生成一个匹配顺序.

算法首先初始化点的匹配顺序 τ 为空, 从拥有最多邻居结点的点开始, 若有多个点拥有相同的邻居数量, 则根据点的标签, 在目标图中拥有最少对应标签的点的点, 作为起始点, 插入匹配顺序 τ. 对剩余未插入匹配顺序的点, 按照以下条件插入匹配顺序.

(1) 对任意一个未插入匹配顺序的结点, 其邻居结点中, 已插入匹配顺序的结点数量最多的结点. 即令 $V_{v_1} = u | u \in \tau, (u, v_1) \in E$, 使得 $|V_{v_1}|$ 最大的点 v_1.

(2) 若符合条件(1)的有多个点, 则对任意一个符合条件(1)的结点, 选择其邻居结点与最多个已在匹配顺序中的结点, 同时也是邻居的结点. 即令 $V_{v_2} = u | u \in \tau, \exists p \in V, (u, p) \in E$ 且 $(p, v_2) \in E$, 使得 $|V_{v_2}|$ 最大的点 v_2.

(3) 若符合条件(2)的点仍有多个, 则对任意一个符合条件(2)的点, 选择存在最多个满足条件的邻居结点的点, 即邻居结点与任意一个已插入匹配顺序的点都不相邻. 即令 $V_{v_3} = u | u \notin \tau, (u, v_3) \in E$ 且 $\forall p \in \tau, (u, p) \notin E$, 使得 $|V_{v_3}|$ 最大的点 $v3$.

(4) 若符合条件(3)的点仍有多个, 则选择其中在目标图中拥有最少对应标签的点, 即设目标图为$G' = (V', E', L', f')$, 令$V_{v_4} = v' | v' \in V', f'(v) = f(v_4)$, 使得$|V_{v_4}|$最小的点. 若满足要求的还有多个, 则任选一个点.

获得匹配顺序后, 算法根据匹配顺序, 对任意一个未匹配的点, 遍历目标图中的点, 检测是否符合匹配要求, 若符合要求, 则记录该匹配, 并按顺序寻找下一个未匹配点的匹配; 若不符合, 则跳过该点, 尝试目标图中的下一个点进行匹配. 算法通过匹配顺序, 对任意一个未匹配的点, 仅计算其前一个已匹配的点的所有对应点的邻居结点中, 未被匹配到任何一个点上

的点, 具体见图 7.

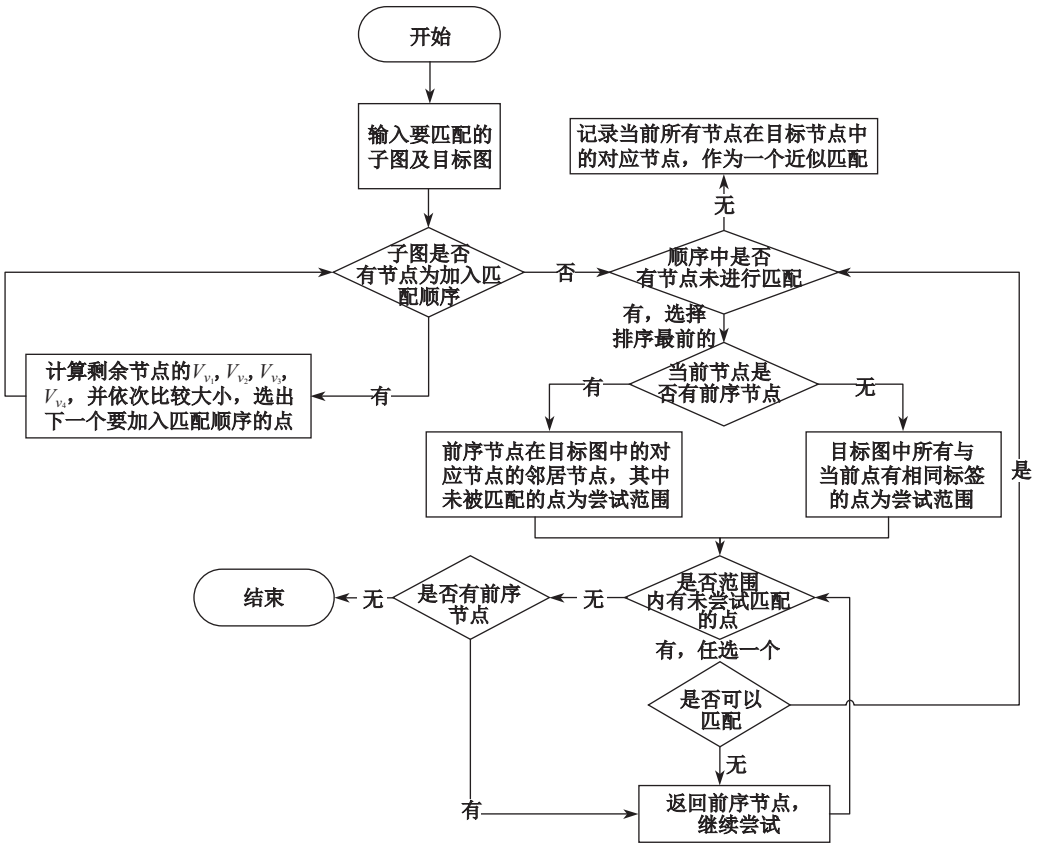


图 7 近似子图匹配流程示意图

Fig. 7 Process of approximate subgraph matching

5 支持度计算及结果展示

5.1 支持度计算

在获得了每个候选子图在给定图中的近似匹配后, 需要判断一个候选子图是否是近似频繁图. 根据子图的近似支持度的定义(定义 5), 一个候选子图的支持度应该是候选子图所有近似匹配中, 可能的互斥的近似匹配的最大数量. 然而, 确定这个最大数量可以被转化为最大独立集(Maximum Independent Set, MIS)问题, 这是一个 NP-complete 的问题, 因此计算一个候选子图的准确近似支持度是十分耗时和困难的.

然而, 在本文的情况下, 在计算子图的匹配时, 已经利用的是子图的近似匹配, 因此一个候选子图的准确近似支持度其实不是非常重要. 此处本文参考了 gApprox 算法中支持度的计算方式, 为每个候选子图的近似支持度计算了一个上限阈值, 本文相信这个阈值在本文问题中已经足够可以作为参考, 来判断一个候选子图是否近似频繁. 计算方式如下.

(1) 将候选子图的一个匹配图的顶点作为一个集合, 初始化近似支持度上限为 0.

(2) 计算候选子图的所有匹配图的点集中的点的出现次数, 即每个点出现在多少个匹配图中, 并按数字从大到小排序.

(3) 选择出现次数最高的点, 若有多个, 则任选一个, 将近似支持度上限加 1, 删除所有

包含该点的匹配图的点集, 并更新每个点的出现次数.

(4) 不断迭代第(3)步, 直到所有的点集都被删除.

假设有一个候选子图的所有近似匹配有 $\{v_1, v_2\}$, $\{v_1, v_3\}$, $\{v_2, v_3\}$, $\{v_1, v_4\}$, $\{v_4, v_5\}$, 根据上述算法, 初始化支持度上限为 0, 点 v_1 出现在 3 个近似匹配中, 是所有点中最多的, 则删除所有包含 v_1 的近似匹配; 支持度上限加 1, 剩余的近似匹配为$\{v_2, v_3\}$, $\{v_4, v_5\}$, 继续迭代, 删除所有包含点 v_2 的; 上限加 1, 然后删除包含 v_4 的匹配, 上限加 1; 此时所有匹配已经被删除, 停止迭代, 得到最终上限为 3.

定理 4　所有候选子图的真实近似支持度都不超过上述算法中的支持度上限.

证　明　由于对于任意一个点, 只记录一次近似匹配, 即只将支持度上限加 1, 相当于记录了一个包含该点的近似匹配, 则保证了记录的近似匹配至少在该点上是互斥的. 但由于并不保证在所有点上都是互斥的, 因此支持度上限必然大于或等于真实的近似支持度.

5.2　结果展示

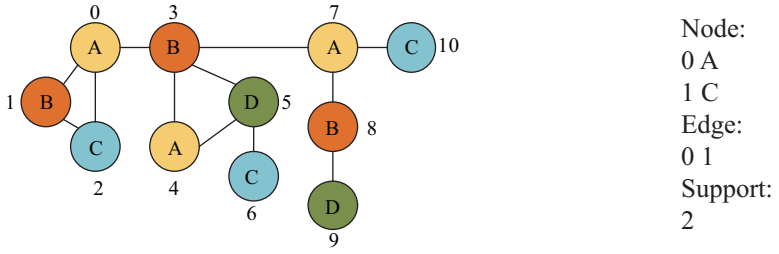
通过上节算法计算, 可以获得每一个候选近似频繁子图的近似支持度; 在与给定的支持度阈值进行比较之后, 可以最终确定一个候选近似频繁子图是否频繁. 为了提高算法结果的展示效果, 以及提高算法效率, 对算法挖掘结果的展示作以下约束.

第一, 对具有相同结构, 以及相同标签的结果图, 只在结果中展示一次.

第二, 对于每个结果图, 需展示其点的 ID 和标签, 以及边的左右两点的 ID 和标签, 其中 ID 是重新编辑的.

第三, 对于每个结果图, 展示其近似频繁程度和全部对应.

以上约束中, 第一条的原因是在候选子图的生成中, 可能有多个具有相同标签和结构的子图, 由于其在给定单图中的位置不同, 可能出现多次. 如图8(a)中, 点 0 和点 1 组成的图, 与点 4 和点 3 组成的图结构和标签都相同, 但在图中的位置不同, 因此会在候选子图生成的过程中出现两次. 为了提高算法效率, 在记录候选子图时, 仅记录其中一个, 对生成的每一个候选子图, 首先与已经被记录的候选子图作比较, 若果有相同结构和标签的记录, 则不再记录此子图. 对于第二条与第三条约束, 主要目的是为了更好地理解近似频繁子图的结构, 以及更好地展示其近似匹配. 如图 8(a)中的点 0 和点 2 组成的子图, 若其为近似频繁子图, 则显示结果为图 8(b).



(a)　　(b)

图 8　结果展示例图

Fig. 8　Sample result illustration

6　实　　验

本文设计了两种实验来证明本文算法的有效性, 以及发现更多近似频繁子图的能力. 第一种实验为有效性试验, 实验分别在真实数据及人工数据上进行. 第二种实

验仅基于人工数据, 用于证明本文算法与使用近似程度绝对值算法相比, 本文算法可以发现更多的近似频繁子图. 所有实验的真实数据抽取自 PPI 网络, PPI 网络获取自 DIP (Database of Interacting Proteins)数据库(http://dip.doe-mbi.ucla.edu), 人工数据由 NetworkX 工具(https://networkx.github.io/ documentation/stable/) 构建. 所有实验都在 1 台 Windows 设备上完成, 设备配置为 CPU i5-4690, 8 GB MEM. 实验程序由 Java 完成. 据本文所知, 目前已有的与本文算法目的最接近的算法为 AGraP, 因此本文所有实验中的对比实验为 AGraP 算法, 本文仅修改了 AGraP 中的近似度计算公式与支持度计算方式, 使对比实验基于同一种相似度公式, 从而降低不同相似度公式造成的影响.

由于对比算法 AGraP 在使用子图大小相关的相似度时效率较低, 算法时间和空间消耗随给定图大小的增长而增长过快, 受条件限制, 无法完成对普通大小的图的挖掘. 因此本文通过 NetworkX 工具分别生成了 7 个、9 个、11 个、13 个点的样本图($G7$、$G9$、$G11$、$G13$), 并在样本图上进行挖掘, 取 $\theta = 0.9, \delta = 2$, 结果表 1 所示.

表 1 样本图上挖掘的时间消耗对比

Tab. 1 Comparison of time required for mining of sample graph

算法	t/ms			
	$G7$	$G9$	$G11$	$G13$
本文算法	53	554	760	1 347
AGraP	1 041	10 226	346 918	46 301 253

通过表 1 可知, 本文算法相对于 AGraP 算法有较大优势, 且由于 AGraP 算法时间消耗随图大小增长过快, 因此不适于对较大的图的挖掘, 而本文算法随着图大小的增长, 虽然时间消耗有所增加, 但增长速度相对较为合理.

除了在样本图上的对比实验, 本文利用本文算法, 对规模更大的图进行了挖掘, 从而展示本文算法除了可以应用于十分小的样本图, 同样可以应用于较大规模的图. 本文在人工数据与真实数据上都进行了实验. 人工数据同样由 NetworkX工具生成, 为了显示算法效率随图大小的变化, 分别生成了 100 个、125 个、150 个、175 个、200 个点的不同大小的图.

实验真实数据来自于PPI网络. PPI 网络为蛋白质交互网络, 其中包含多种已被发现的蛋白质及蛋白质交互反应. 其中有多条蛋白质对拥有高于 10^{-7}的 BLAST(Basic Local Alignment Search Tool)相似度, 本文认为这些蛋白质对具有极其相似的的性质, 因此可以用相同的标签做标记, 不同的蛋白质之间有不同种类的反应, 每种反应可以用不同的标签进行标记, 因此 PPI 网络可以转化为一个标签图. 本文从中分别抽取了 5 个不同的 250 个点($G100$、$G125$、$G150$、$G175$、$G200$、$G250$ (真)), 作为实验数据, 并将其结果取平均数作为真实数据上的结果. 实验结果如图 9、图 10 所示.

从图 9、图 10 中可以发现, 算法的时间消耗随近似支持度要求的提升不断降低, 原因是随着近似支持度要求的提升, 更多的候选子图在早期被过滤掉. 根据局部反单调性的特点, 其特定大小范围内的超图都不可能是频繁的, 因此减少了需要生成近似子图及近似子图匹配的候选子图数量, 大大节省了时间消耗. 同样地, 随着近似程度要求的降低, 每个候选子图会拥有更多的近似子图, 因此需要进行匹配的近似子图增加. 同时, 更多的候选子图被判定为频繁, 因此时间消耗上升. 除此之外, 给定单图的大小对时间消耗也具有较大的影响, 原因是随给定单图大小增加, 其候选子图的数量呈指数上升, 因此时间消耗上升.

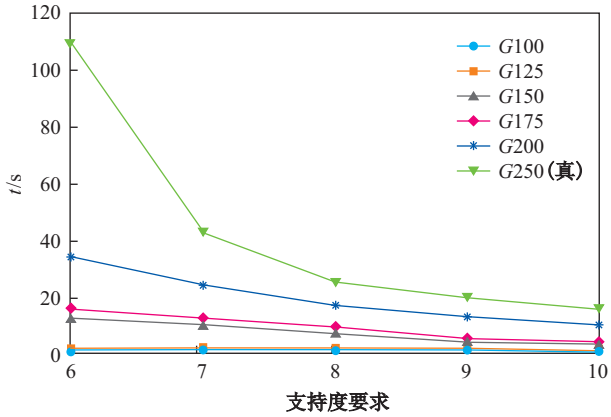


图 9　算法时间消耗随支持度要求的变化

Fig. 9　Time required varies with support requirements

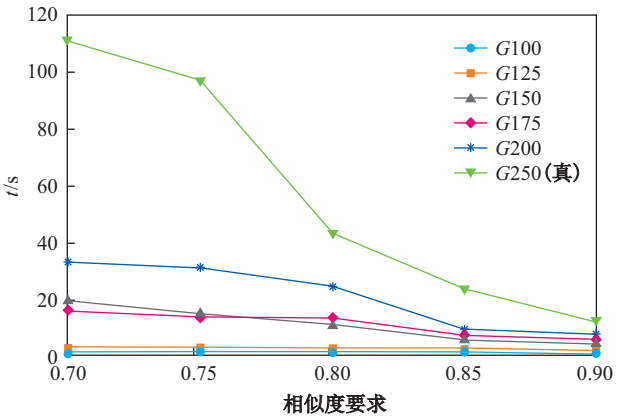


图 10　算法时间消耗随近似程度要求的变化

Fig. 10　Time required varies with approximation

除上述实验, 本文还在样本图上进行了对比实验, 实验主要比较本文算法发现的近似频繁子图数量, 与利用近似程度绝对大小的算法发现的近似频繁子图的数量的区别. 实验利用的样本图为图 8(a). 实验对比算法为本文算法的修改版, 即近似程度计算公式为缺失的点或边的绝对值. 实验设置近似程度相对值 θ 为 0.6, 对比算法中近似程度绝对值为 1, 即仅允许一个点或一条边的缺失, 近似支持度 δ 都为 3. 实验结果如表 2.

表 2　发现子图能力结果

Tab. 2　Experimental results showing the ability to discover subgraphs

图	近似匹配	
	本文算法	对比算法
A	点0,4,7	点 0,4,7
B	点1,3,8	点 1,3,8
C	点 2,6,10	点 2,6,10
A—B	(0,1),(3-4),(7,8)	(0,1),(3-4),(7,8)
B—A—C	(1,0,2),(4,3),(8,7,10)	(1,0,2), (8,7,10)

如表 2 所示, 本文算法利用候选子图大小相关的近似度相对值计算公式, 在近似支持度设置为 3 时, 发现候选子图 B-A-C 为近似频繁子图, 而对比算法利用缺失点或边的绝对值, 则忽略了该候选子图.

7 结 论

随着数据规模的增加和数据中可能产生的噪音数据量的增长, 近似频繁子图挖掘算法将受到越来越广泛的关注. 因此, 近似频繁子图挖掘算法需要根据候选子图大小, 来判断可容忍的噪音数据量的多少, 即可容忍的数据缺失数量是多少. 本文设计了一种利用候选子图大小相关的近似程度计算公式, 从而允许不同大小子图缺失不同数量的点或边的近似频繁子图挖掘算法. 该算法利用缺失点或边的数量与子图大小的百分比作为近似程度计算的标准, 通过候选子图生成, 近似子图生成及匹配, 近似支持度的计算几个步骤, 实现了候选子图大小相关的近似频繁子图挖掘算法. 人工数据和真实数据上的实验都表明, 本算法效率更高, 且能发现利用缺失点或边数量绝对值作为近似程度的算法不能发现的候选子图. 在未来的工作中, 希望能进一步提高算法效率, 使得算法能在大规模数据上应用.

[参 考 文 献]

[1] YAN X F, HAN J W. gSpan: Graph-based substructure pattern mining [C]// Proceedings of the 2002 IEEE International Conference on Data Mining. 2002:721-724.

[2] ELSEIDY M, ABDELHAMID E, SKIADOPOULOS S, et al. GraMi: Frequent subgraph and pattern mining in a single large graph [J]. Proceedings of the VLDB Endowment, 2014, 7: 517-528.

[3] CHEN C, YAN X F, ZHU F D, et al. gApprox: Mining frequent approximate patterns from a massive network [C]// 7th IEEE International Conference on Data Mining. 2007:445-450.

[4] FLORES-GARRIDO M, CARRASCO-OCHOA J A, MARTÍNEZ-TRINIDAD J F. AGraP: An algorithm for mining frequent patterns in a single graph using inexact matching [J]. Knowledge and Information Systems, 2015,2(44): 385-406.

[5] KURAMOCHI M, KARYPIS G. Finding frequent patterns in a large sparse graph [J]. Data Mining and Knowledge Discovery, 2005, 11(3): 243-271.

[6] NIJSSEN S, KOK J N. A quickstart in frequent structure mining can make a difference [C]// Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2004: 647-652.

[7] ALONSO A G, PAGOLA J E M, CARRASCO-OCHOA J A, et al. Mining frequent connected subgraphs reducing the number of candidates [C]// Joint European Conference on Machine Learning and Knowledge Discovery in Databases. 2008: 365-376.

[8] RANU S, SINGH A K. Graphsig: A scalable approach to mining significant subgraphs in large graph databases [C]// 2009 IEEE 25th International Conference on Data Engineering. 2009: 844-855.

[9] CHENG H, YAN X F, HAN J W. Mining graph patterns [C]// Frequent Pattern Mining, Berlin: Springer, 2014: 307-338.

[10] KRISHNA V, SURI N R, ATHITHAN G. A comparative survey of algorithms for frequent subgraph discovery [J]. Current Science, 2011, : 190-198.

[11] CHOUDHURY S, PUROHIT S, LIN P, et al. Percolator: Scalable pattern discovery in dynamic graphs [C]// Proceedings of the 11th ACM International Conference on Web Search and Data Mining. 2018: 759-762.

[12] INGALALLI V, IENCO D, PONCELET P. Mining frequent subgraphs in multigraphs [J]. Information Sciences, 2018, 451/452: 50-66.

[13] ALGULIEV R M, ALIGULIYEV R M, GANJALIYEV F S. Extracting a heterogeneous social network of academic researchers on the Web based on information retrieved from multiple sources [J]. American Journal of Operations Research, 2011, 1(2): 33.

[14] LIMA JR D P, GIACOMINI H C, TAKEMOTO R M, et al. Patterns of interactions of a large fish–parasite network in a tropical floodplain [J]. Journal of Animal Ecology, 2012, 81(4): 905-913..

[15] GAO X B, XIAO B, TAO D C, et al. A survey of graph edit distance [J]. Pattern Analysis and Applications, 2010, 13(1): 113-129.

[16] HIDOVIĆ D, PELILLO M. Metrics for attributed graphs based on the maximal similarity common subgraph [J]. International Journal of Pattern Recognition and Artificial Intelligence, 2004, 18(3): 299-313.

[17] DEHMER M, EMMERT-STREIB F. Comparing large graphs efficiently by margins of feature vectors [J]. Applied Mathematics and Computation, 2007, 188(2): 1699-1710.

[18] HOLDER L B, COOK D J, DJOKO S. Substucture discovery in the SUBDUE system [C]// KDD Workshop, 1994: 169-180.

[19] JIA Y, ZHANG J T, HUAN J. An efficient graph-mining method for complicated and noisy data with real-world applications [J]. Knowledge and Information Systems, 2011, 28(2): 423-447.

[20] FLORES-GARRIDO M, CARRASCO-OCHOA J A, MARTíNEZ-TRINIDAD J F. Extensions to AGraP algorithm for finding a reduced set of inexact graph patterns [J]. International Journal of Pattern Recognition and Artificial Intelligence, 2018, 32(1): 1860012.

[21] ACOSTA-MENDOZA N, GAGO-ALONSO A, CARRASCO-OCHOA J A, et al. Extension of canonical adjacency matrices for frequent approximate subgraph mining on multi-graph collections [J]. International Journal of Pattern Recognition and Artificial Intelligence, 2017, 31(8): 1750025.

[22] ACOSTA-MENDOZA N, MORALES-GONZáLEZ A, GAGO-ALONSO A, et al. Image classification using frequent approximate subgraphs [C]// Iberoamerican Congress on Pattern Recognition. 2012: 292-299.

[23] ACOSTA-MENDOZA N, CARRASCO-OCHOA J A, MARTíNEZ-TRINIDAD J F, et al. Image clustering based on frequent approximate subgraph mining [C]// Mexican Conference on Pattern Recognition. 2018: 189-198.

(责任编辑: 李　艺)

(上接第 41 页)

[参　考　文　献]

[1] 马如云. 非线性常微分方程非局部问题 [M]. 北京: 科学出版社, 2004.

[2] 王伟, 史希福. 三阶常微分方程两点边值问题解的存在性及单调迭代方法 [J]. 数学学报(中文版), 1992, 35: 213-219.

[3] 姚庆六. 一类非线性三阶两点边值问题的单调迭代方法 [J]. 云南大学学报(自然科学版), 2011, 33(1): 1-5.

[4] 姚庆六. 三阶常微分方程的某些非线性特征值问题的正解 [J]. 数学物理学报, 2003, 23A: 513-519.

[5] 姚庆六. 一类非线性三阶两点边值问题的三重正解 [A]. 滨州学院学报, 2014, 30(3): 1673-2618.

[6] YAO Q L. Solution and positive solution for a semilinear third-order two-point boundary value problems [J]. Appl Math Letters, 2004, 17: 1171-1175.

[7] MA R Y, LU Y Q. Disconjugacy and extremal solutions of nonlinear third-order equations [J]. Commun Pure Appl Anal, 2014, 13(3): 1223-1236.

[8] SUN Y P. Existence and iteration of monotone positive solutions for a third-order two-point boundary value problem [J]. Appl Math J Chinese Univ (Ser B), 2008, 23(4): 413-419.

[9] GUO L J, SUN J P, ZHAO Y H. Existence of positive solutions for nonlinear third-order three-point boundary value problems [J]. Nonlinear Analysis, 2008, 68(10): 3151-3158.

(责任编辑: 林　磊)