计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2022,31(5):157−164 [doi: 10.15888/j.cnki.csa.008466]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

# 改进的 *k* 度匿名图构造算法①

曾　滔

(华南师范大学 计算机学院, 广州 510631)
通信作者: 曾　滔, E-mail: 2019022624@m.scnu.edu.cn

**摘　要**: 在社交网络中, 为防范用户隐私泄漏, 在用户数据发布前需要做匿名化处理. 针对以节点度数为背景知识的隐私攻击, 将社交网络匿名化问题建模为图的 *k* 度匿名化问题; 其主要方法是对图添加尽可能少的边或点来满足度匿名化要求, 其中要求添加边或点较少是期望尽可能保持原图结构特性. 目前, 加边类算法并不能很好地保留平均路径长度等结构特性; 加边且可加点类算法尽管能更好地保留原图结构特性, 但添加的边或点较多. 本文融合两类算法的策略提出改进算法. 新算法利用贪心法生成匿名度序列, 然后基于社区结构加边, 并且优先满足其匿名代价高于平均匿名代价的节点的匿名化要求; 若加边不能完成匿名化, 则通过加点实现图匿名化. 真实数据集上的实验结果表明新算法能更好地保留图的几种典型的结构特性, 并且添加的边或点更少.

**关键词**: 社交网络; 隐私保护; *k* 度匿名化; 度序列; 加边; 加点; 复杂网络

引用格式: 曾滔.改进的 *k* 度匿名图构造算法.计算机系统应用,2022,31(5):157–164. http://www.c-s-a.org.cn/1003-3254/8466.html

## Improved Algorithm for Constructing *k*-Degree Anonymous Graphs

ZENG Tao

(School of Computer Science, South China Normal University, Guangzhou 510631, China)

**Abstract**: In social networks, anonymization processing is required to prevent the leakage of user privacy before user data is released. In this study, the social network anonymization is modeled as the *k*-degree anonymization of graphs given the privacy attack with the background of node degrees. The major method is to add as few edges or nodes as possible to the graph to meet the degree-anonymization requirements, and by doing this, the structural characteristics of the original graph are expected to be maintained to a large extent. At present, the edge-adding algorithms cannot well retain the basic structural characteristics such as the average path length, while the edge-node adding algorithms can better retain the structural characteristics of the original graph after adding many edges or nodes. Considering this situation, this study proposes an improved algorithm combining the two algorithms. Firstly, the greedy algorithm is used to generate the anonymity sequence, and then the edge-adding operation is performed on the basis of the community structure. The anonymization requirements of nodes with higher anonymity costs than the average anonymity costs are satisfied first, and node adding can be applied to the graph when edge adding cannot complete the anonymization. The experimental results of actual datasets show that the new algorithm can better retain several typical structural characteristics of the graph, and the number of added edges or nodes is less.

**Key words**: social network; privacy protection; *k*-degree anonymization; degree sequence; edge adding; node adding; complex network

① 收稿时间: 2021-07-23; 修改时间: 2021-08-18; 采用时间: 2021-08-31; csa 在线出版时间: 2022-04-11

## 1 引言

随着互联网的快速发展, 在线社交网络在全世界普及, 社交网络的运营商收集了大量用户数据, 如果通过简单的匿名化处理将用户数据提供给第三方 (研究人员、市场营销等) 使用, 就会出现隐私泄漏的风险[1,2]. 因此, 社交网络的隐私安全越来越受到人们的关注, 运营商需要在发布数据之前, 对用户数据进行匿名化处理, 确保用户隐私不会泄漏[3]; 同时尽可能地保留社交网络的结构特性, 从而保证数据的实用性[4,5].

基于图结构变换的匿名方法[6], 能够有效地解决社交网络中用户的隐私保护问题. Backstrom 等人[7] 指出使用无意义的唯一标识符替换用户的身份作匿名化处理, 并不能防止用户隐私泄漏. Hay 等人[8] 发现利用节点周围的结构信息仍然能够在匿名图中重新识别该节点, 这种结构信息与节点及其邻居节点的度有关. Liu 等人[9] 基于 $k$ 匿名性, 提出了 $k$ 度匿名化模型, 这种匿名保护能够有效抵御以节点度数为背景知识的隐私攻击, 使得每个节点被重新识别的概率不大于$1/k$.

针对 $k$ 度匿名化问题, Liu 等人[9] 基于动态规划思想生成匿名度序列, 根据该序列通过添加边的方式去构建图, 由于构图操作并不能保证一定成图, 所以需要构图测试, 这会增加算法运行时间; 而且构图操作的随机过程导致原图结构信息的损失. Lu 等人[10] 提出了一种贪心匿名化算法, 在给原图添加边的同时进行度序列的匿名化, 直接在原图上进行加边能够避免度序列的构图测试, 但是同步匿名化需要添加较多的边来实现图匿名化.

Chester 等人[11,12] 提出了加点的匿名化策略, 基于动态规划思想生成匿名度序列, 通过新节点与原图节点连边满足节点匿名要求, 然而加点太多导致原图结构信息的损失. Ma 等人[13] 结合加边且可加点的匿名化策略提出匿名化算法, 利用贪心法生成匿名度序列, 并基于社区结构进行加边, 这种匿名化方式能较好地保留图的某些结构特性, 但添加的边或节点数较多. 吴童[14] 结合加边且可加点的匿名化策略提出了改进算法, 利用 Liu 等人[9] 提出的动态规划匿名分组算法生成匿名度序列, 根据该序列贪心加边; 若加边不能完成匿名化, 则进行加点完成图匿名化. 由于未充分利用原图中的点进行加边, 导致添加的节点数较多. Rajabzadeh 等人[15] 利用 Ma 等人[13] 提出的贪心分组算法和基于模块度的社区发现算法[16] 确定每个社区内节点的匿名化代价, 然后通过遗传算法决定每个社区内的节点之间如何连边, 从而实现图的度匿名化. 本文的主要贡献包括:

(1) 结合加边且可加点的策略, 改进了匿名化方式, 分两个阶段进行加边, 这种匿名方式能有效减少添加的边或节点.

(2) 在加边操作中, 考虑节点度、社区结构和节点距离等因素选择目标节点进行连边, 这种操作能够有效保留原图的某些结构特性.

(3) 在多个真实数据集上进行了大量实验, 实验结果验证了新算法的有效性.

## 2 预备知识

本文将社交网络建模为无向图$G(V, E)$, 其中, $V$是节点集, $E$是边集. 令节点个数$n = |V|$, $V = \{v_1, v_2, \cdots, v_n\}$, 则称图$G$所有节点的点度序列$\boldsymbol{d}_G = \{\boldsymbol{d}_G(v_1), \boldsymbol{d}_G(v_2), \cdots, \boldsymbol{d}_G(v_n)\}$为图 $G$ 的度序列, 其中, $\boldsymbol{d}_G(v_i)$为图 $G$中节点$v_i$的度数. 为了方便讨论, 不妨假设图$G$的度序列为非递增序列, 由此,图$G$最小点度为$\delta_G = \boldsymbol{d}_G(v_n)$,最大点度为$\Delta_G = \boldsymbol{d}_G(v_1)$.

给定无向图$G$和正整数$k$ ($\delta_G \leqslant k \leqslant \Delta_G$), 当且仅当图中每个节点至少与$k-1$ 个其他节点具有相同的度, 则称图$G$是$k$度匿名图, 其度序列称为$k$度匿名度序列.

设图$G$不是 $k$ 度匿名图, 通过在图$G$中添加边得到新图$G'$, 使得$G'$是 $k$ 度匿名图, 称序列$\boldsymbol{d}_{G'} = \{\boldsymbol{d}_{G'}(v_1), \cdots, \boldsymbol{d}_{G'}(v_n)\}$为图$G'$的 $k$ 度匿名度序列. 令$\boldsymbol{d}_{\text{Cost}} = \{\boldsymbol{d}_{G'}(v_1) - \boldsymbol{d}_G(v_1), \cdots, \boldsymbol{d}_{G'}(v_n) - \boldsymbol{d}_G(v_n)\}$为匿名代价序列. 为了方便操作, 不妨假设匿名代价序列为非递增序列, 其中, $\boldsymbol{d}_{G'}(v_i) - \boldsymbol{d}_G(v_i)$表示节点$v_i$的匿名化代价.

图匿名问题可描述如下: 给定无向图$G$和正整数$k$ ($\delta_G \leqslant k \leqslant \Delta_G$), 构造图$G$的 $k$ 度匿名图$G'$满足$G \subseteq G'$, 且使得$diff(M(G), M(G'))$可能小. 其中, $M(G)$表示图$G$的某个结构特性 (平均路径长度等), 而$diff(M(G), M(G'))$表示两个图在某个结构特性上的差异程度. 即要求图$G$的 $k$ 度匿名图$G'$满足$\boldsymbol{min}_{G \subseteq G'} diff(M(G), M(G'))$. 由于图的结构特性的多样性, 本文将图的 3 种典型的结构特性 (平均路径长度、平均聚类系数、传递系数) 作为匿名效果的度量指标, 使得匿名图与原图在这 3 种结构特性上的差异尽可能小.

## 3 算法模型

新算法包括如下子算法: 匿名分组算法、加边算法、加点算法、层次社区发现算法. 新算法思想是利用贪心匿名分组算法生成匿名度序列, 基于层次社区结构加边, 优先满足匿名代价大于平均匿名代价的节点的匿名要求; 然后再次利用贪心匿名分组算法得到匿名度序

列, 通过同样的加边操作满足未匿名节点的匿名要求; 若加边不能完成匿名化, 则通过加点实现图匿名化.

### 3.1 匿名分组算法

本文采用 Liu 等人[9] 提出的贪心匿名分组算法, 首先形成一个包含前$k$个最高度节点的组, 并把组内节点的度赋值为$\boldsymbol{d}_G(v_1)$, 然后考虑两个匿名代价$C_{\text{merge}}$和$C_{\text{new}}$, 计算公式如下:

$$C_{\text{merge}} = (\boldsymbol{d}_G(v_1) - \boldsymbol{d}_G(v_{k+1})) + Cost(\boldsymbol{d}_G[v_{k+2}, v_{2k+1}]) \quad (1)$$

$$C_{\text{new}} = Cost(\boldsymbol{d}_G[v_{k+1}, v_{2k}]) \quad (2)$$

当$C_{\text{merge}} > C_{\text{new}}$时, 把第$k+1$个节点至第$2k$个节点合并成一个新组, 组内所有节点度为$\boldsymbol{d}(v_{k+1})$; 否则将第$k+1$个节点合并到前一个组中, 并将第$k+2$个节点视为合并或作为新组的起点, 算法在遍历所有节点后停止. 其中$Cost(\boldsymbol{d}_G[v_i, v_j])$表示第$i$至$j$个节点划分到同个匿名组的匿名代价, 计算公式如下:

$$Cost(\boldsymbol{d}_G[v_i, v_j]) = \sum_{l=i}^{j} (\boldsymbol{d}_G(v_i) - \boldsymbol{d}_G(v_l)) \quad (3)$$

通过该算法得到匿名度序列, 每个组中至少有$k$个度相同的节点, 满足 $k$ 匿名性原则.

### 3.2 加边算法

加边算法包括两个阶段: 优先加边和普通加边. 优先加边阶段尽量满足匿名代价大于平均匿名代价的节点的匿名要求, 按照匿名代价降序遍历节点, 基于层次社区结构选择目标节点进行连边, 每次加边操作更新匿名代价序列. 在加边操作中, 从当前节点所在的社区开始遍历, 考虑两类节点: 1) 节点度比当前节点度小; 2) 节点度比当前节点度大且匿名代价大于 0. 将符合条件的节点标识为目标节点, 并优先选择与当前节点距离近的目标节点进行连边. 若标记的目标节点数仍不能满足当前节点的匿名要求, 则向高级别社区遍历, 找到足够多的目标节点进行连边.

算法 1. 优先加边算法 (pre_edges_addition)

输入: 图$G$, 匿名代价序列$\boldsymbol{d}_{\text{Cost}}$, 层次社区 $L$
输出: 图 $G'$

```
1. initialize T=∅//目标节点集合
2. FOR vertex v in {v|d_Cost(v)>avg_cost,v∈V}DO
3.   C←find_community(L,v)
4.   FOR community c∈C DO
5.     FOR node n∈V DO//选择目标节点
6.       IF n∈c∧n∉T ∧ n≠v ∧ (n,v)∉E THEN
7.         IF d_G(n)<d_G(v) THEN
8.           T←T∪{n}
9.         ELSE IF d_G(n)>d_G(v)∧d_Cost(n)>0 THEN
10.          T←T∪{n}
11.        END IF
12.      END IF
13.    END FOR
14.    IF |T| ≥d_Cost(v) THEN
15.      BREAK//目标节点数满足当前节点匿名要求
16.    END IF
17.  END FOR
18.  sort T according to the shortestpath (T_i,v),i∈(0, |T|) in the ascending order//若节点不可达, 则距离为∞
19.  WHILE d_Cost(v)>0 ∧ |T|>0 DO//加边操作
20.    E∪{(T_i,v)},i∈(0, |T|)
21.    T←T\T_i
22.    d_Cost(v)←d_Cost(v)−1
23.  END WHILE
24. END FOR
25. RETURN G′
```

再次通过贪心匿名分组得到匿名度序列, 普通加边阶段根据该序列尽量满足所有节点的匿名要求. 采用与优先加边阶段同样的加边操作, 仅考虑匿名代价大于 0 的节点, 并优先选择与当前节点距离近的目标节点进行连边.

算法 2. 普通加边算法 (edges_addition)

输入: 图$G'$, 匿名代价序列$\boldsymbol{d}_{\text{Cost}}$, 层次社区$L$
输出: 图 $G''$, 匿名代价序列$\boldsymbol{d}'_{\text{Cost}}$

```
1. initialize T=∅//目标节点集合
2. FOR vertex v in {v|d_Cost(v)>0,v∈V} DO
3.   C←find_community(L,v)
4.   FOR community c∈C DO
5.     FOR node n in {n| d_Cost(n)>0,n∈V}DO
6.       IF n∈c∧n∉T ∧ n≠v ∧(n,v)∉E THEN
7.         T←T∪{n}
8.       END IF
9.     END FOR
10.    ...//省略部分参考算法 1
11. END FOR
12. RETURN G″,d′_Cost
```

### 3.3 层次社区发现算法

在层次化社区树中, 社区级别越高则社区规模越大, 整个图为最高级别的社区, 如图 1 所示. 采用 Louvain 算法[16] 获得层次化社区结构, 这是一种基于模块度优化[17] 的启发式算法. 第 1 步, 每个节点都属于一个独立的社区, 通过社区之间的节点移动, 使得社区划分的模块度最大化; 第 2 步, 根据当前划分的每个社区聚合成单个节点, 节点之间的边权重为其代表的两个社区之间的边权重之和; 重复这两个步骤, 直到模块度大小不再变化.

igraph 软件包[18] 提供了该算法的实现函数 *community_multilevel*. 当传入参数*return_levels*为 True, 返回包含每层社区结构的节点集合$L=\{\{L_1^2,\cdots,L_{n_2}^2\},\cdots,\{L_1^l,\cdots,L_{n_l}^l\}\}$; 否则返回模块度最优的社区结构$\{L_1^2,\cdots,L_{n_2}^2\}$.

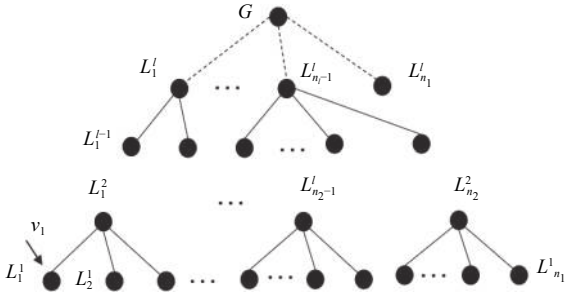


图 1 层次化社区树

算法 3. 层次社区发现算法 (find_community)

输入: 层次社区$L$, 当前节点$v$
输出: 包含$v$的层次社区$C$

1. initialize $C=\emptyset$
2. FOR *hierarchy_community* $hc\in L$ DO
3. FOR *community* $c\in hc$ DO
4. IF $v\in c$ THEN
5. $C\cup\{c\}$
6. BREAK
7. END IF
8. END FOR
9. END FOR
10. $C\cup\{V\}$//添加最高级别社区
11. RETURN $C$

### 3.4 图匿名化算法

图匿名化算法整合如下算法: 匿名分组算法、层次社区发现算法、加边算法和加点算法, 原图经过加边和加点操作生成满足 $k$ 度匿名化的图.

算法 4. 图匿名化算法 (graph_anonymization)

输入: 图$G(V,E)$, 参数$k$
输出: 匿名图$G'''$

1. $L\leftarrow G$.community_multilevel(*return_level*$\leftarrow T$)
2. $\boldsymbol{d}_{\text{Cost}}\leftarrow$anonymize_partition($G,k$)
3. $G'\leftarrow$pre_edges_addition($G,\boldsymbol{d}_{\text{Cost}},L$)
4. $\boldsymbol{d}_{\text{Cost}}\leftarrow$anonymize_partition($G',k$)
5. $G'',\boldsymbol{d}'_{\text{Cost}}\leftarrow$edges_addition($G',\boldsymbol{d}_{\text{Cost}},L$)
6. IF $\{v|\boldsymbol{d}'_{\text{Cost}}(v)>0,v\in V\}\neq\emptyset$ THEN//是否加点
7. $G'''\leftarrow$nodes_addition($G'',\boldsymbol{d}'_{\text{Cost}},k$)
8. END IF
9. $G'''\leftarrow G''$
10. RETURN $G'''$

对于一些特殊情况, 新算法仅通过加边无法满足节点匿名化要求. 所以, 通过 Chester 等人[11,12] 提出的加点算法, 将原图中未匿名的节点依次与新节点相连来满足节点的匿名化要求; 同时, 新节点也需要进行匿名化处理, 并最终实现图的匿名化. 新节点个数$m$满足如下关系, $mc$表示节点的最大匿名代价.

$$m=(1+\max(mc,k)\bmod 2)+\max(mc,k) \tag{4}$$

### 3.5 算法复杂性分析

Liu 等人[9] 提出的贪心匿名分组算法的时间复杂度为O($nk$). 由匿名分组算法确定的未匿名节点个数不会超过($n-n/k$), $n/k$表示匿名分组数. 优先加边算法的时间复杂度为O($lmn$), $l$表示层次化社区树的高度, $m$表示匿名代价大于平均匿名代价的节点个数; 当节点匿名代价很大时, 可能需要遍历全图寻找目标节点. 普通加边算法的时间复杂度为O($(n-n/k)\times l\times mc$), 虽然此阶段需要匿名处理的节点数较多, 但是节点最大匿名代价$mc$较小, $mc\ll n$. Louvain 算法[16] 的时间复杂度是线性阶的, 层次社区发现算法的时间复杂度也是线性阶的. 通过 Floyed 算法[19] 计算所有节点的路径长度, 将其输出作为算法输入, 该时间复杂度为O($n^3$). 因此, 新算法的时间复杂度为O($n^2$).

## 4 实验结果与分析

实验环境为 Windows 10 专业版, 运行环境采用 PyCharm 2020 (Python 语言), PC 机主频 3.6 GHz, 内存 8 GB. 新算法命名为 KDVEM21, 在 5 个真实数据集上进行实验, 同 5 个经典算法比较, 对比算法仅包含加边或加点的多项式时间复杂度算法: 包括 Priority 算法[9]、FKDA 算法[10]、V_ADD 算法[11,12]、KDVEM15 算法[13] 和 KDVEM18 算法[14], 分别从匿名效果和匿名代价两个方面评估算法的性能, 并对实验结果进行分析.

### 4.1 数据集描述

本文实验数据采用斯坦福大学提供的公开数据集 (SNAP), 表 1 给出了数据集的详细介绍, 实验数据集都是无向的、简单的、无标记的.

### 4.2 评价方法

徐恪等人[20] 概述了典型的社交网络拓扑参数, Ji 等人[21] 讨论了图数据匿名化的研究演变和匿名度量的有效性, Zhao 等人[22] 指出组合多个度量指标作为隐私度量更具可比性, 赵蕙等人[23] 认为社交网络匿名化度量方法的多样和复杂, 匿名算法研究者难以找到合适

的方法评估自己的创新研究. 因此, 实验采用典型的 3 个图的结构特性作为匿名效果的度量指标.

(1) 平均路径长度 (average path length)

平均路径长度计算公式如下, $SPL$是节点$u$和$v$之间的最短距离, $CP$表示所有可连接的节点对.

$$\overline{P}=\frac{\sum_{(u,v)\in CP} SPL(u,v)}{|CP|} \tag{5}$$

(2) 传递系数 (transitivity)

传递系数, 也称为全局聚类系数, 计算公式如下, $\Delta$表示闭合三联体数目, $\wedge$表示三联体数目.

$$T=\frac{\Delta}{\wedge} \tag{6}$$

(3) 平均聚类系数 (average clustering coeffiient)

在社交网络中, 聚类系数的体现是一个人的两个好友彼此也是好友的概率, 计算公式如下:

$$C_i=\frac{2|E_{xy}|}{d_i(d_i-1)},0\leqslant C_i\leqslant 1 \tag{7}$$

其中, $E_{xy}$表示节点$i$的相邻节点$x$和$y$之间的边数, $d_i$表示节点$i$的度, $n$为节点数. 平均聚类系数是所有节点的聚类系数的平均值, 计算公式如下:

$$\overline{C}=\frac{1}{n}\sum_{i=1}^{n}C_i,0\leqslant\overline{C}\leqslant 1 \tag{8}$$

表 1 数据集描述

| 名称 | 节点数 | 边数 | 平均路径长度 | 传递系数 | 平均聚类系数 |
|---|---|---|---|---|---|
| ca-GrQc | 5 242 | 14 496 | 6.047 | 0.630 | 0.530 |
| ca-HepTh | 9 877 | 25 998 | 5.945 | 0.284 | 0.471 |
| ca-HepPh | 12 008 | 118 521 | 4.672 | 0.659 | 0.611 |
| ca-AstroPh | 18 772 | 198 110 | 4.194 | 0.318 | 0.631 |
| ca-CondMat | 23 133 | 93 497 | 5.352 | 0.264 | 0.633 |

### 4.3 结果分析

实验参数$k$的取值范围{5, 10, 15, 20, 25, 50}, 在复现 Priority 算法时, 构图操作中采用了随机选点的策略, 因而在每个参数上运算 10 次取平均值作为实验结果. V_ADD 算法和 KDVEM15 算法由作者提供源代码. 将实验结果绘制成点线图, 其中, 横轴表示参数$k$, 纵轴表示度量指标的取值范围. 水平线表示原图的度量指标, 其余分别表示对比算法在对应参数下输出匿名图结构特性的度量指标. 图 2–图 6 分别表示各个真实数据集在对应参数 $k$ 生成匿名图的结构特性.

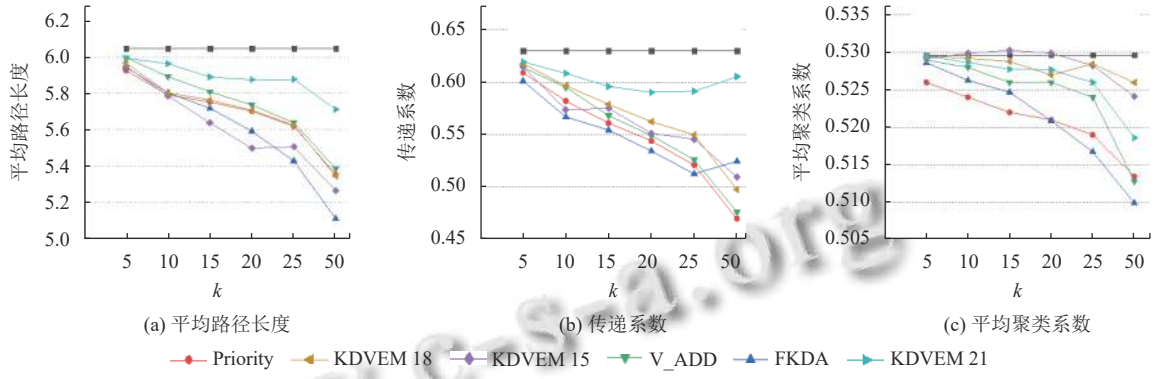


图 2 ca-GrQc 数据集上匿名图的结构特性

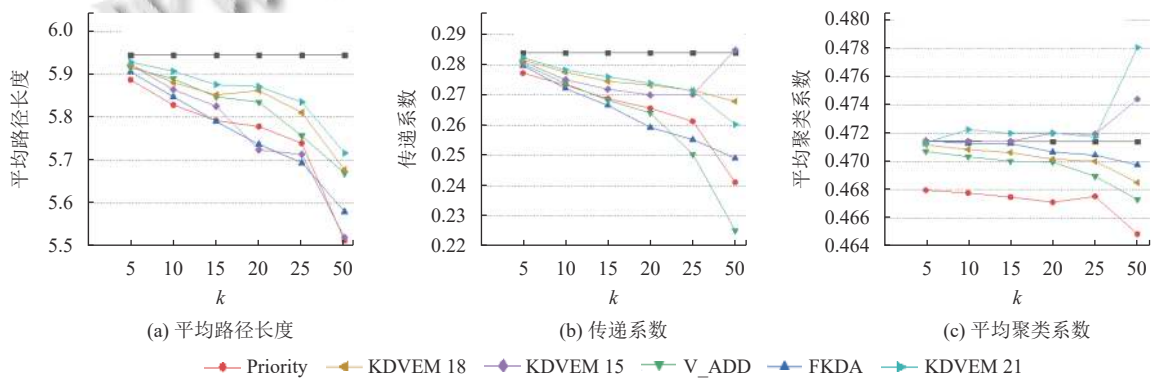


图 3 ca-HepTh 数据集上匿名图的结构特性

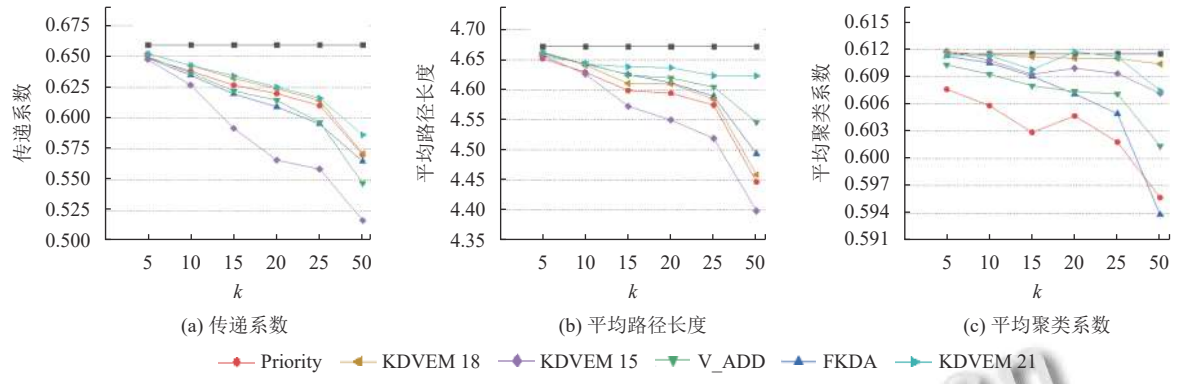


图 4 ca-HepPh 数据集上匿名图的结构特性

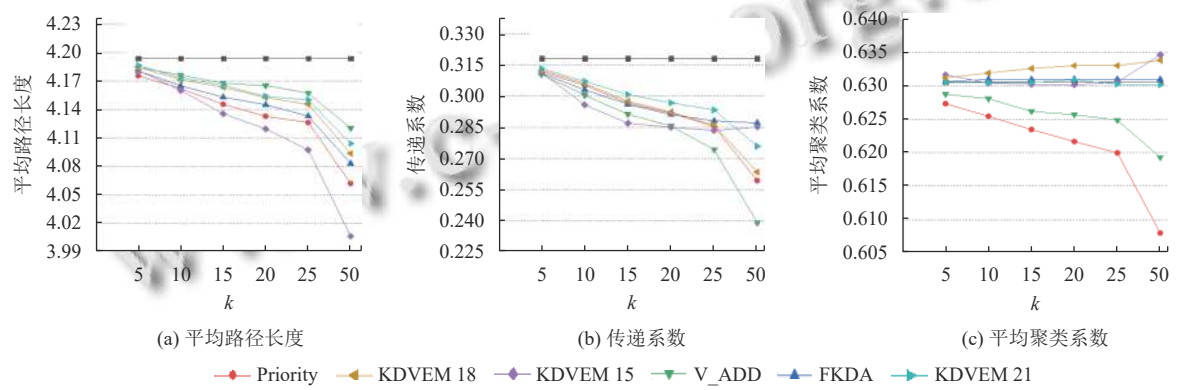


图 5 ca-AstroPh 数据集上匿名图的结构特性

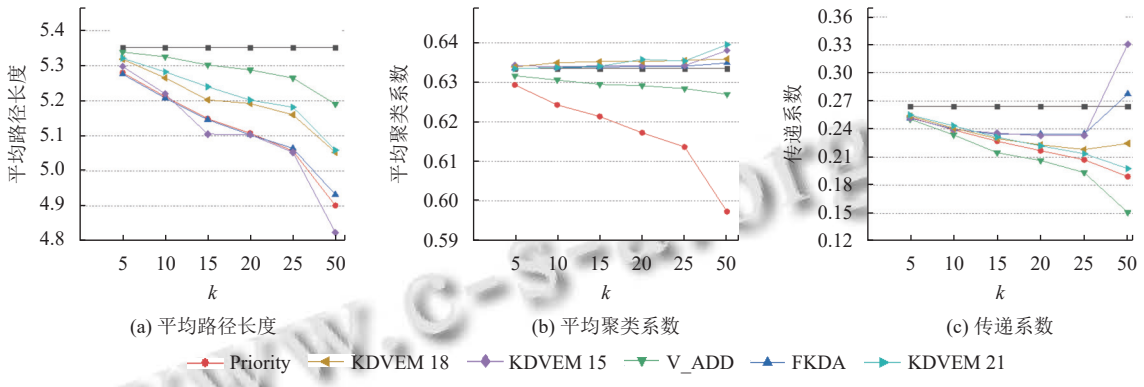


图 6 ca-CondMat 数据集上匿名图的结构特性

为了直观地对比算法性能, 采用平均变化量的百分比形式作为实验结果, 如表 2 所示, 数值越小说明匿名图与原图的结构特性差异越小. 计算公式如式 (9), 其中, $m$表示图的结构特性的度量指标.

$$R=\frac{1}{|k|}\sum_{i\in k}\left|\frac{\Delta m}{m_0}\right|_i\times 100 \tag{9}$$

Rajabzadeh 等人[15] 认为 3 个度量指标具有同等的重要性, 取 3 个度量指标的平均值作为算法的综合评价. 随着参数$k$的增大, 需要添加的边就越多, 若将相近节点相连, 则路径长度会变小; 简单的加边使得原图的社区结构变得交错, 从而传递系数和平均聚类系数变化较大, 而基于原图社区结构加边, 一定程度保留了社区结构, 使得传递系数和平均聚类系数变化较小. KDVEM21 算法基于层次化社区结构, 优先与距离近的目标节点连边, 使得平均路径长度的变化较小, 从表 2 可以看出, KDVEM21 算法在平均路径长度的相对变化较小; 正是考虑社区结构的原因, 传递系数和平均聚类系数的变化较小, 在 ca-GrQc、ca-

HepTh、ca-HepPh 和 ca-AstroPh 数据集上的综合评价最好. 但对于节点较多且最大度较大的数据集, 采用优先加边的操作时, 可能需要遍历高级别社区, 这一定程度上破坏了小规模社区的结构, 加剧了传递系数的变化.

Zhang 等人[24] 通过对比添加边和节点来评估算法的匿名代价, 如表 3、表 4 所示. KDVEM21 算法采用两阶段加边操作, 充分利用原图的节点进行加边, 减少了添加的节点. 对比加边且可加点类算法, KDVEM21 算法添加的边或节点更少.

表 2 匿名图的结构特性

| 算法 | ca-GrQc | | | | ca-HepTh | | | | ca-HepPh | | | | ca-AstroPh | | | | ca-CondMat | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | $\overline{P}$ | $T$ | $\overline{C}$ | $avg$ | $\overline{P}$ | $T$ | $\overline{C}$ | $avg$ | $\overline{P}$ | $T$ | $\overline{C}$ | $avg$ | $\overline{P}$ | $T$ | $\overline{C}$ | $avg$ | $\overline{P}$ | $T$ | $\overline{C}$ | $avg$ |
| Priority | 5.89 | 13.03 | 1.65 | 6.85 | 3.17 | 6.86 | 0.91 | 3.65 | 1.92 | 6.13 | 1.38 | 3.14 | 1.42 | 8.31 | 1.52 | 3.75 | 4.38 | 16.02 | 2.56 | 7.65 |
| FKDA | 7.40 | 12.85 | 1.60 | 7.28 | 3.13 | 7.17 | **0.13** | 3.47 | 1.46 | 7.19 | 0.88 | 3.18 | 1.20 | 6.96 | **0.02** | 2.73 | 4.31 | **8.78** | **0.09** | **4.39** |
| V_ADD | 5.03 | 11.93 | 1.01 | 5.99 | 2.12 | 8.35 | 0.39 | 3.62 | 1.20 | 7.32 | 0.70 | 3.08 | **0.76** | 10.88 | 0.81 | 4.15 | **1.26** | 21.20 | 0.64 | 7.70 |
| KDVEM15 | 7.27 | 10.85 | **0.27** | 6.13 | 3.08 | **3.10** | 0.15 | 2.11 | 2.53 | 11.35 | 0.31 | 4.37 | 1.84 | 8.44 | 0.16 | 3.48 | 4.70 | 12.29 | 0.21 | 5.73 |
| KDVEM18 | 5.72 | 10.02 | 0.28 | 5.34 | 1.89 | 3.99 | 0.26 | 2.05 | 1.66 | 5.56 | **0.08** | 2.43 | 1.00 | 7.90 | 0.32 | 3.07 | 2.87 | 12.31 | 0.26 | 5.15 |
| KDVEM21 | **2.67** | **4.46** | 0.62 | **2.58** | **1.42** | 3.50 | 0.34 | **1.75** | **0.71** | **5.03** | 0.18 | **1.98** | 0.91 | **6.37** | 0.03 | **2.44** | 2.58 | 14.04 | 0.30 | 5.64 |

表 3 ca-HepPh 数据集上边和节点数的变化量

| $k$ | 类型 | Priority | | FKDA | | V_ADD | | KDVEM15 | | KDVEM18 | | KDVEM21 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 数量 | 百分比 (%) | 数量 | 百分比 (%) | 数量 | 百分比 (%) | 数量 | 百分比 (%) | 数量 | 百分比 (%) | 数量 | 百分比 (%) |
| 5 | 节点 | 0 | 0 | 0 | 0 | 49 | 0.41 | 49 | 0.41 | 11 | 0.09 | 0 | 0 |
| | 边 | 358 | 0.30 | 862 | 0.73 | 845 | 0.71 | 1 065 | 0.90 | 488 | 0.41 | 612 | 0.52 |
| 10 | 节点 | 0 | 0 | 0 | 0 | 67 | 0.56 | 89 | 0.74 | 11 | 0.09 | 11 | 0.09 |
| | 边 | 964 | 0.82 | 2 253 | 1.90 | 2 061 | 1.74 | 3 866 | 3.26 | 1 185 | 1.00 | 1 355 | 1.14 |
| 15 | 节点 | 0 | 0 | 0 | 0 | 83 | 0.69 | 119 | 0.99 | 17 | 0.14 | 15 | 0.12 |
| | 边 | 1 614 | 1.36 | 4 143 | 3.50 | 3 599 | 3.04 | 10 192 | 8.60 | 2 022 | 1.71 | 2 712 | 2.29 |
| 20 | 节点 | 0 | 0 | 0 | 0 | 111 | 0.92 | 149 | 1.24 | 21 | 0.17 | 21 | 0.17 |
| | 边 | 2 214 | 1.87 | 6 106 | 5.15 | 4 290 | 3.62 | 15 246 | 12.86 | 2 658 | 2.24 | 3 529 | 2.98 |
| 25 | 节点 | 0 | 0 | 0 | 0 | 119 | 0.99 | 165 | 1.37 | 33 | 0.27 | 25 | 0.21 |
| | 边 | 2 792 | 2.36 | 8 992 | 7.59 | 6 476 | 5.46 | 22 737 | 19.18 | 6 485 | 2.94 | 4 891 | 4.13 |
| 50 | 节点 | 0 | 0 | 0 | 0 | 159 | 1.32 | 195 | 1.62 | 79 | 0.66 | 51 | 0.42 |
| | 边 | 6 186 | 5.22 | 22 873 | 19.30 | 14 848 | 12.53 | 57 711 | 48.69 | 7 719 | 6.51 | 10 525 | 8.88 |

表 4 ca-AstroPh 数据集上边和节点数的变化量

| $k$ | 类型 | Priority | | FKDA | | V_ADD | | KDVEM15 | | KDVEM18 | | KDVEM21 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 数量 | 百分比 (%) | 数量 | 百分比 (%) | 数量 | 百分比 (%) | 数量 | 百分比 (%) | 数量 | 百分比 (%) | 数量 | 百分比 (%) |
| 5 | 节点 | 0 | 0 | 0 | 0 | 117 | 0.62 | 69 | 0.37 | 65 | 0.35 | 5 | 0.03 |
| | 边 | 425 | 0.22 | 885 | 0.45 | 1 044 | 0.53 | 905 | 0.46 | 620 | 0.31 | 717 | 0.36 |
| 10 | 节点 | 0 | 0 | 0 | 0 | 155 | 0.83 | 29 | 0.15 | 93 | 0.50 | 11 | 0.06 |
| | 边 | 1 029 | 0.52 | 2 297 | 1.16 | 2 607 | 1.32 | 4 220 | 2.13 | 1 347 | 0.68 | 1 788 | 0.90 |
| 15 | 节点 | 0 | 0 | 0 | 0 | 181 | 0.96 | 61 | 0.32 | 125 | 0.67 | 15 | 0.08 |
| | 边 | 1 869 | 0.94 | 4 077 | 2.06 | 4 174 | 2.11 | 8 873 | 4.48 | 2 619 | 1.32 | 3 365 | 1.70 |
| 20 | 节点 | 0 | 0 | 0 | 0 | 223 | 1.19 | 85 | 0.45 | 151 | 0.80 | 0 | 0.00 |
| | 边 | 2 434 | 1.23 | 5 205 | 2.63 | 5 123 | 2.59 | 11 586 | 5.85 | 3 257 | 1.64 | 4 294 | 2.17 |
| 25 | 节点 | 0 | 0 | 0 | 0 | 235 | 1.25 | 95 | 0.51 | 153 | 0.82 | 25 | 0.13 |
| | 边 | 3 345 | 1.69 | 7 303 | 3.69 | 7 773 | 3.92 | 16 494 | 8.33 | 4 485 | 2.26 | 5 720 | 2.89 |
| 50 | 节点 | 0 | 0 | 0 | 0 | 291 | 1.56 | 109 | 0.58 | 203 | 1.08 | 51 | 0.27 |
| | 边 | 7 750 | 3.91 | 15 138 | 7.64 | 17 510 | 8.84 | 56 392 | 28.46 | 10 690 | 5.40 | 13 596 | 6.86 |

## 5 总结与展望

社交网络匿名化是复杂网络领域中一个重要的研究方向. 解决社交网络匿名化问题需要平衡用户隐私保护和图数据实用性二者的关系. 本文利用经典算法的优势提出了改进算法, KDVEM21 算法采用加边且可加点的匿名化策略, 分两个阶段进行加边, 并结合节

点特性和社区结构对加边操作进行了优化. 在 5 个真实网络上测试 KDVEM21 算法的有效性, 实验结果表明, 对比 5 个经典算法, 大多数情况下, KDVEM21 算法能更好地保留图的几种典型的结构特性 (平均路径长度、平均聚类系数和传递系数), 并且对比加边且可加点类算法, 添加边或节点更少; 但新算法的时间复杂度并不具备优势. 未来工作: 对算法时间复杂度进行优化; 探索算法在其它图结构特性度量下的匿名效果; 减少图匿名化的代价.

## 参考文献


1 Sahoo SR, Gupta BB. Security issues and challenges in online social networks (OSNs) based on user perspective. In: Gupta BB, Agrawal DP, Wang HX, eds. Computer and Cyber Security. Boca Raton: Auerbach Publications, 2018. 591–606.

2 Beigi G, Liu H. A survey on privacy in social media: Identification, mitigation, and applications. ACM/IMS Transactions on Data Science, 2020, 1(1): 7.

3 Kiranmayi M, Maheswari N. A review on privacy preservation of social networks using graphs. Journal of Applied Security Research, 2021, 16(2): 190–223.

4 Casas-Roma J, Herrera-Joancomartí J, Torra V. *k*-Degree anonymity and edge selection: Improving data utility in large networks. Knowledge and Information Systems, 2017, 50(2): 447–474. [doi: 10.1007/s10115-016-0947-7]

5 Casas-Roma J. DUEF-GA: Data utility and privacy evaluation framework for graph anonymization. International Journal of Information Security, 2020, 19(4): 465–478. [doi: 10.1007/s10207-019-00469-4]

6 Casas-Roma J, Herrera-Joancomartí J, Torra V. A survey of graph-modification techniques for privacy-preserving on networks. Artificial Intelligence Review, 2017, 47(3): 341–366.

7 Backstrom L, Dwork C, Kleinberg J. Wherefore art thou R3579X?: Anonymized social networks, hidden patterns, and structural steganography. Proceedings of the 16th International Conference on World Wide Web. Alberta: ACM, 2007. 181–190.

8 Hay M, Miklau G, Jensen D, *et al*. Resisting structural re-identification in anonymized social networks. Proceedings of the VLDB Endowment, 2008, 1(1): 102–114.

9 Liu K, Terzi E. Towards identity anonymization on graphs. Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. Vancouver: ACM, 2008. 93–106.

10 Lu XS, Song Y, Bressan S. Fast identity anonymization on graphs. Proceedings of the 23rd International Conference on Database and Expert Systems Applications. Vienna: Springer, 2012. 281–295.

11 Chester S, Kapron BM, Ramesh G, *et al*. *k*-Anonymization of social networks by vertex addition. ADBIS, 2011, 789(2): 107–116.

12 Chester S, Kapron BM, Ramesh G, *et al*. Why Waldo befriended the dummy? k-Anonymization of social networks with pseudo-nodes. Social Network Analysis and Mining, 2013, 3(3): 381–399. [doi: 10.1007/s13278-012-0084-6]

13 Ma TH, Zhang YL, Cao J, *et al*. *KDVEM*: A *k*-degree anonymity with vertex and edge modification algorithm. Computing, 2015, 97(12): 1165–1184.

14 吴童. k 度匿名图构造算法的研究 [ 硕士学位论文 ]. 上海: 华东师范大学, 2018.

15 Rajabzadeh S, Shahsafi P, Khoramnejadi M. A graph modification approach for *k*-anonymity in social networks using the genetic algorithm. Social Network Analysis and Mining, 2020, 10(1): 38. [doi: 10.1007/s13278-020-00655-6]

16 Blondel VD, Guillaume JL, Lambiotte R, *et al*. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, 2008, 2008: P10008.

17 Newman MEJ, Girvan M. Finding and evaluating community structure in networks. Physical Review E, 2004, 69(2): 026113. [doi: 10.1103/PhysRevE.69.026113]

18 Csárdi G, Nepusz T. The igraph software package for complex network research. InterJournal Complex Systems, 2006, 1695(5): 1–9.

19 Kleene SC. Representation of events in nerve nets and finite automata. Automata Studies, 1956, 34: 3–41.

20 徐恪, 张赛, 陈昊, 等. 在线社会网络的测量与分析. 计算机学报, 2014, 37(1): 165–188.

21 Ji SL, Mittal P, Beyah R. Graph data anonymization, de-anonymization attacks, and de-anonymizability quantification: A survey. IEEE Communications Surveys & Tutorials, 2017, 19(2): 1305–1326.

22 Zhao YC, Wagner I. Using metrics suites to improve the measurement of privacy in graphs. IEEE Transactions on Dependable and Secure Computing, 2020, 19(1): 259–274.

23 赵蕙, 王良民, 申屠浩, 等. 网络匿名度量研究综述. 软件学报, 2021, 32(1): 218–245. [doi: 10.13328/j.cnki.jos.006103]

24 Zhang YL, Ma TH, Cao J, *et al*. *K*-anonymisation of social network by vertex and edge modification. International Journal of Embedded Systems, 2016, 8(2–3): 206–216.