DEIM Forum 2014 E5-4

# ソーシャルネットワークデータの距離関係の変化を抑制する k匿名化アルゴリズム

岡田　莉奈†　　渡辺知恵美††　　北川　博之††

† 筑波大学情報学群情報科学類　〒 305-8573 つくば市天王台 1-1-1
†† 筑波大学システム情報系　〒 305-8573 つくば市天王台 1-1-1
E-mail: †okarina@kde.cs.tsukuba.ac.jp, ††{chiemi,kitagawa}@cs.tsukuba.ac.jp

**あらまし**　ソーシャルネットワークデータ (SN データ) を研究やデータ分析の目的で一般的に公開するためには，利用者のプライバシー保護のための匿名化が必要である．SN データ匿名化の一既存手法として任意のノードの隣接ノードから成るサブグラフに着目し，同型のサブグラフが少なくとも $k$ 個存在するようにノイズエッジを追加する $k$-neighbor という匿名化がある．しかし，$k$-neighbor を実現するアルゴリズムではノイズエッジの追加によりノード間の距離関係が大きく変化する．そこで，本研究では距離関係の変化を抑制する $k$-匿名化のアルゴリズムを提案する．

**キーワード**　Privacy，Social network，k-Anonymity

# The k-neighbor anonymization algorithm of social network data with keeping distances between nodes

Rina OKADA†, Chiemi WATANABE††, and Hiroyuki KITAGAWA††

† College of Information Sciences University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan
†† Faculty of Engineering, Information and Systems University of Tsukuba
1-1-1 Tennodai, Tsukuba, Ibaraki 305-8573, Japan
E-mail: †okarina@kde.cs.tsukuba.ac.jp, ††{chiemi,kitagawa}@cs.tsukuba.ac.jp

## 1. はじめに

近年，ソーシャルネットワークサービス (SNS) の利用者が急増している．SNS での友人関係やメッセージのやり取りはユーザの生活スタイルや社会的関係に密接にかかわっており，ソーシャルネットワークデータ（以降，SN データと呼ぶ.）の分析と利活用に注目が集まっている．そこで，SN データは研究やデータ分析を行なうために公開され，利用できるようになってきている．このように研究者や企業に SN データを公開するためには，ソーシャルネットワークサービス (SNS) ユーザのプライバシーを保護する必要がある．なぜならば，何かしらの背景知識を持っている人が特定のユーザを見つけ出し，攻撃をする恐れがあるからである．そこで，SN データの匿名化方法が注目されている．

これまで，既に SN データに対する匿名化方法 [1], [2], [3], [4], [5] などが提案されている．SN データではユーザをノード，ユーザ間の関係をエッジで表現したグラフ構造とすることが一般的であり，SN データの匿名化にはグラフデータの構造を考慮する必要がある．SN データの匿名化の一指標である $k$-neighbor [1] では，攻撃者が攻撃対象となるユーザの友人関係を知っていることを想定し，隣接ノードから成るサブグラフに着目する．そして，任意のノードが少なくとも他の $k-1$ 個のノードとサブグラフが同型であれば，$k$-neighbor 匿名であると定義している．SN データに対する匿名化指標はこの他に，ノードの次数に着目した $k$-degree [3] やサブグラフの同型に着目した $k$-isomorphism [5] などがある．これらの手法と比較して我々は $k$-neighbor が SN データに対して現実的な匿名化指標であると注目している．その理由は，SNS では多くのユーザがその公開範囲を “友人まで公開する” と設定することに対し，$k$-neighbor では任意のノードの隣接ノードを考慮対象とするからである．

また，SN データの匿名化ではいくつかのグラフ構造を同型にするために，エッジやノードの追加・削除，ノードに含まれるラベルの一般化などを行なう．このような操作によって元の SN データに含まれる特徴をできるだけ失わないように工夫

する必要がある，以降，匿名化後もできるだけ維持すべき SN データの特徴をユーティリティと呼ぶ．$k$-neighbor を実現するアルゴリズムでは SN データの特徴のうちスケールフリー性とスモールワールド性に着目し，これらを崩さないように工夫している．スケールフリー性とは，ユーザとそのユーザの友人の数の分布が冪乗則分布に従う性質のことであり，スモールワールド性とは，任意の二名のユーザが中間にいくつかの数の友人を介するだけで繋がっているという性質のことである．

しかし，いくら SN データの性質が保存できるような匿名化が行えたとしても，分析結果が有用なものにならなければデータの価値が無くなる．SN データは，しばしばクラスタリングやリンク予測などを用いて分析される．文献 [1] で提案されているアルゴリズムでは，ユーザ間の距離関係が考慮されていない．ユーザ間の距離関係はクラスタリングやリンク予測に使用されるため，匿名化前と匿名化後において大きく変化することは望ましくない．そこで，本研究では SN データの性質であるスケールフリー性とスモールワールド性を保存しつつ，ユーザ間の距離関係の変化を抑制するアルゴリズムを提案する．

本稿は次のように構成される．まず，2. 節で SN データに対する既存の匿名化の一手法である $k$-neighbor の匿名化指標と $k$-neighbor を実現するアルゴリズム，及びそのアルゴリズムの問題点について述べる．次に，3. 節で本研究の提案アルゴリズムについて述べ，4. 節で評価実験について述べる．更に，5. 節で提案アルゴリズムの拡張について議論する．また，6. 節で他の関連研究との比較をする．最後に，7. 節で本稿のまとめと本研究の今後の課題について言及する．

## 2. $k$-neighbor

本節では，2.1 節で本研究に関係する SN データに対する既存の匿名化の一手法である $k$-neighbor の匿名化指標とその指標が適応できる攻撃者の背景知識の仮定について述べ，その後，2.2 節で $k$-neighbor を達成するアルゴリズムについて述べる．

### 2.1 $k$-neighbor の匿名化指標

$k$-neighbor では，SN データを重複するエッジのないシンプルな無向グラフ $G = (V, E, L, \mathcal{L})$ で表し，このグラフデータを匿名化するための指標として定義している．グラフ $G$ において各ユーザのプロパティはノードのラベルとして簡潔に表現されている．このラベル階層は半順序集合である．$V$ はノードの集合，$E \subseteq V \times V$ はエッジの集合，$L$ はラベルの集合，$\mathcal{L} : V \to L$ は各ノードからラベルを結びつけるラベル関数である．$V(G)$, $E(G)$, $L_G$, $\mathcal{L}_G$ は，それぞれグラフ $G$ のノード集合，エッジ集合，ラベル集合，ラベル関数のことである．

攻撃者は，攻撃対象となるユーザのラベルとそのユーザの友人のネットワークトポロジーに関する背景知識を持ち，攻撃対象ユーザのノードを特定したいと想定する．これは攻撃者が対象ノードの隣接ノードからなるグラフを知っており，そのグラフにマッチするグラフ G 内のサブグラフを探すことと対応付けられる．例として図 1 のグラフを用いて説明する．図 1(a) は SN データであり，ここから個人が特定できる属性を取り除いたグラフ図 1(b) を公開することを想定する．また，攻撃者は図

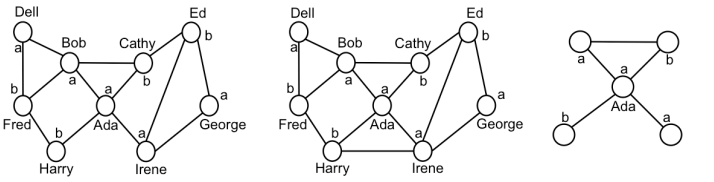


図 1　$k$-neighbor の例

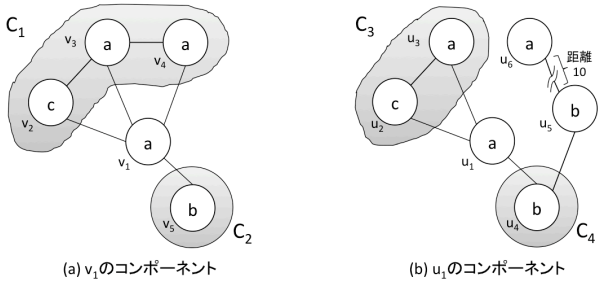


図 2　コンポーネントの例

1(b) から Ada のノードを特定したい．その際，背景知識として Ada はこの SN データで Bob，Cathy，Harry，Irene と友人関係にあり，そのうち Bob と Cathy はこの SNS で友人関係にあることを知っているとする．つまり，攻撃者は図 1(c) のグラフにマッチするサブグラフを発見しようとする．図 1(b) にはこれにマッチするサブグラフは一つしかないため，中心にあるノードが Ada であることが分かってしまう．そこで，$k$-neighbor では任意のノードが少なくとも $k-1$ 個のノードとサブグラフが同型であることを匿名化指標とする．図 1(b) の例では下の二つのノードの間にエッジを追加することによって，図 1(c) と同型となるサブグラフは図 1(b) に 3ヶ所登場する．これによって攻撃者は背景知識を用いても Ada を 1/3 以上の確率で特定することができなくなる．

### 2.2 $k$-neighbor を実現するアルゴリズム

Bin Zhou らは文献 [1] にて，$k$-neighbor の定義に加えて $k$-neighbor のためのグラフ編集アルゴリズムを提案している．このアルゴリズムでは，ノード $v$ の誘導部分グラフを隣接コンポーネント集合 $Neighbor_G(v)$ として表し，ある 2 ノード $u, v$ の隣接コンポーネント集合 $Neighbor_G(u)$,$Neighbor_G(v)$ を匿名化するためのコスト $Cost(u, v)$ を定義している．これをもとに，匿名化コストが小さな $k$ 個のノードをグラフ $G$ から見つけて同型にするグリーディアルゴリズムによってグラフ全体の $k$-neighbor を実現している．匿名化コスト関数は式 (1) の通りである．例えば図 2(a) における $Neighbor_G(v_1)$ は $C_1$，$C_2$ となる．二つのノード $u, v$ の隣接コンポーネント集合 $Neighbor_G(u)$,$Neighbor_G(v)$ の匿名化コストは，互いのコンポーネント集合を同型にするためにノードのラベルの一般化コスト，追加するエッジ数，コンポーネントに新たに含めるノード数によって求められる．

ここでは，図 2 の $u_1$ と $v_1$ の隣接コンポーネント集合を同型にすることを考える．$C_2$ と $C_4$ は同型であるため，$C_1$ と $C_3$ のみ同型にすればよい．$v_2$ と $u_2$, $v_3$ と $u_3$ はそれぞれ対応する

が，$C_1$ の $v_4$ に対応するノードが $C_3$ には含まれていないため，$v_4$ に対応するノードをコンポーネント外から選択し，$C_3$ に含める．図 2 の例では $u_6$ を選択し，$u_3$ 及び $u_1$ と接続するエッジを追加することで $C_1$ と $C_3$ を同型にすることができる．

$$
\begin{aligned}
&Cost(u, v) \\
&= \alpha \cdot \sum_{v' \in H'} NCP(v') \\
&\quad + \beta \cdot |(u, v | (u, v) \notin E(H), (A(u, v)) \in E(H'))| \\
&\quad + \gamma \cdot (|V(H')| - |V(H)|) \qquad (1)
\end{aligned}
$$

式 (**??**) 内の $\alpha$,$\beta$,$\gamma$ は重みパラメータである．$Neighbor_G(u)$ がノード $u \in V(G)$ の 1-neighbor のノード集合である．さらに，$u, v \in V(G)$ に対して $Neighbor_G(u)$ と $Neighbor_G(v)$ 内のノードのラベルを一般化したものが $Neighbor_{G'}(A(u))$ と $Neighbor_{G'}(A(v))$ である．また，$H = Neighbor_G(u) \cup Neighbor_G(v)$，$H' = Neighbor_{G'}(u) \cup Neighbor_{G'}(v)$ である．式 (1) の第一項目では，はノードのラベルを同値にするためのコストである．ラベルを同値にするためにはラベル階層を用いて一般化を行なうので，その際に発生する情報損失をコストの一部としている．ラベルの一般化コストの計算は式 (2) を用いる．第二項目では，匿名化を行なう際に追加するノイズエッジの数で情報損失を表しており，第三項目では，ノイズエッジによって接続されてコンポーネント内に導入されるノードの数で情報損失を表している．

$$
NCP(l) = \frac{size(l)}{size(*)} \qquad (2)
$$

### 2.3 既存アルゴリズムの問題点

文献 [1] のアルゴリズムでは，匿名化による情報損失を抑制するためにコスト関数を導入し，コストの低い $k$ 個以上のノードの隣接コンポーネントを同型にする．しかしながら，隣接コンポーネントを同型化する処理の中で，コスト関数では考慮されていないにもかかわらず，グラフのユーティリティに大きく影響を与える箇所がある．それはコンポーネント外からコンポーネント内に新たに含めるノードの選択である．この処理で不適切なノードを選択した場合，グラフ構造が大きく変化する可能性がある．

文献 [1] において，このノードを選択する際の優先順位は次の通りである．これらはグラフ全体が SN データとしての特徴であるスケールフリー性とスモールワールド性を保つための基準である．ただし全ての場合において，匿名化するノード $v$ と $u$ において対応するコンポーネントがそれぞれ $C_i(v), C_i(u)$ のとき，選択できるノード $w$ の条件は，$w \notin C_i(v), C_i(u)$ である．

(1)　まだ匿名化されていないノードのうち最少次数を持つもの

(2)　(1) のようなノードが複数ある場合は，まだ匿名化されてないノードのうち最少次数を持ち，最もラベルが類似しているもの

(3)　(1) や (2) のようなノードがない場合は，既に匿名化されているノードのうち最少次数を持つもの

(4)　(3) のようなノードが複数ある場合は，既に匿名化されているノードのうち最少次数を持ち，最もラベルが類似しているもの

(3) や (4) の場合が起こる理由は，選択できるノード $w$ の条件が $C_i(v)$ にも $C_i(u)$ 含まれないものなので，匿名化されていないノードが全て注目しているコンポーネント $C_i(v), C_i(u)$ に含まれている場合はそのようなノードが選択できなくなるからである．ただし (3) と (4) の場合，その選択されたノードと同時に匿名化された他の $k-1$ 個のノードをまだ匿名化していないノードの集合 $VertexList$ に入れる．これによって $k$ 匿名が保証される．(1) と (2) の場合のコンポーネントに追加するノードを選択するためのアルゴリズムを Algorithm1 に示す．(3) と (4) の場合のコンポーネントにノードを追加するノードを選択するためのアルゴリズムは，Algorithm1 のまだ匿名化されていないノードの集合 $VertexList$ を既に匿名化されたノードの集合 $AnonymizedVertexList$ に変更すれば良い．

しかしながら，この優先順位ではグラフ $G$ 内のノード間の距離関係が大きく変化してしまう可能性がある．例えば，図 2 の場合，ノード $u_3$ とノード $u_6$ の間にノイズエッジを追加すると $C_1$ と $C_3$ が同型になるが，同型化前のノード $u_3$ とノード $u_6$ の距離は 13 であり，同型化後の距離は 1 である．

**Algorithm 1**：コンポーネントに追加するノードの選択

**Input:** 接続元のノード $u$，接続先のラベル $l$;
**Output:** ノード $u$ と接続するノード $v$;

```
1: V を VertexList 内で最少次数を持つノードの集合とする；
2: if |V| >= 2 then
3:     return ラベルが l と最も類似する v ∈ V；
4: else
5:     return v ∈ V；
6: end if
```

## 3. 提 案 手 法

本節では，2.3 節に上げた問題点を解決する方法として，任意の 2 ノード間の距離関係をできるだけ変化させないようなノードの選択方法を提案する．Algorithm1 では，まだ匿名化されていないノードの集合 $VertexList$ に含まれるノードのうち次数の小さいノード集合を求め，その中でラベルが $l$ に類似するノードを選択している．このアルゴリズムでは，ノイズエッジの接続元のノード $v$ との距離関係は考慮されていないことが分かる．

### 3.1 提案アルゴリズム

提案アルゴリズムでは，コンポーネントに追加するノードを選択する順序として，接続元のノード $v$ と距離の近いものを優先的に選択する．具体的な提案アルゴリズムのノード選択は，既存のアルゴリズムに対して優先順位が次のようになっている．

(1)　まだ匿名化されていないノードのうち最短距離のもの

(2)　(1) のようなノードが複数ある場合は，まだ匿名化されてないノードのうち最短距離であり，最少次数を持つもの

(3)　(1) や (2) のようなノードが複数ある場合は，まだ匿名化されてないノードのうち最短距離であり，最少次数を持ち，かつ，ラベルが最も類似しているもの
(4)　(1)，(2)，(3) のようなノードがない場合は，既に匿名化されているノードのうち最短距離のもの
(5)　(4) のようなノードが複数ある場合は，既に匿名化されているノードのうち最短距離であり，最少次数を持つもの
(6)　(4) や (5) のようなノードが複数ある場合は，既に匿名化されているノードのうち最短距離であり，最少次数を持ち，かつ，ラベルが最も類似しているもの

(1)，(2)，(3) の場合のコンポーネントに追加するノードを選択するための提案アルゴリズムを Algorithm3 に示す．Algorithm3 はダイクストラ法を応用する．まず，ノイズエッジの接続元のノードの隣接ノード集合を考え，それらの距離を 1 し，探索ノード集合 $Q$ へ入れる (Algorithm3:1 から 5 行目)．次に，Algorithm4 に示したノード集合からノードを選択する際に用いる優先順位関数 *select* を用いて，$Q$ 内からノードを選択する (Algorithm3:6 行目)．ここで選択されたノード $v$ が同型化できるノードの条件を満たしていればそのノードを選択し，終了となる．同型化できるノードとなる条件は以下のとおりである．

- 匿名化対象となっているノード
- 匿名化対象となっているノードのコンポーネント集合である Neighbor 内のノード
- Neighbor 内のノードの隣接ノード
- 匿名化の際に Neighbor に新たに加わったノード
- 匿名化の際に Neighbor に新たに加わったノードの隣接ノード

同型化できるノードの条件を満たすノードが $Q$ 内になければ，$v$ を $Q$ から取り除き，そのノードのまだ探索していない隣接ノードを $Q$ に入れる (Algorithm3:10 から 19 行目)．Algorithm3 内の $distance(v)$ は Input である接続元のノード $u$ とノード $v$ の距離を表している．この操作を探索ノード集合 $Q$ が空集合になるまで繰り返す (Algorithm3:7 行目)．

本提案アルゴリズムを適応した例を図**??**に示す．図**??**(a) は図**??**の (b) と同等のサブグラフであり，ノード $u_3$ を接続元のノードとする．図**??**での接続先のノードは $u_6$ であったのに対し，提案アルゴリズムにでは接続先のノードとして $u_8$ を選択する．ノード $u_8$ を選択するまでの過程を説明する．まず，ノード $u_3$ から距離が近い順に幅優先探索を開始し，同型化できるノードであるか確認する．探索の順番は，ノード $u_2$，ノード $u_1$，ノード $u_4$，ノード $u_7$，ノード $u_5$ となるが，ここまでいずれも同型化できるノードでは無い．次にノード $u_8$ へ探索が進み，このノードは同型化できるノードであるので，$u_8$ が選択されて探索が終了する．

## 4. 評価実験

我々は，提案手法 (Algorithm2) の有効性を示すために評価実験を行った．本節では，4.1 節で実験環境とデータセットについて述べる．4.2 節で評価実験内容について述べ，4.3 節で実験結果を示す．そして，4.4 節で結果について考察する．

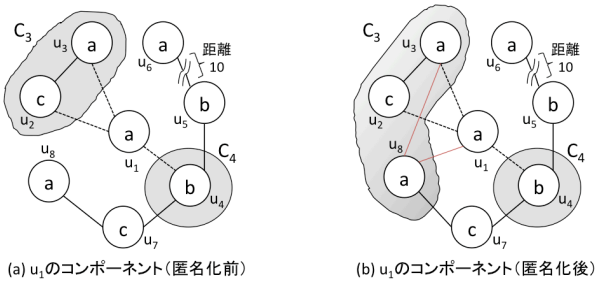


図 3　提案アルゴリズムによる匿名化例

**Algorithm 2**：距離関係を考慮したコンポーネントに追加するノードの選択

**Input:** グラフ $G$ 内のノード集合 $V(G)$，接続元のノード $u$，接続先のノードのラベル $l$;
**Output:** ノード $u$ と接続するノード $v$;

1: **for** each $v \in V(G)$ **do**
2: $distance(v) = \infty$
3: **end for**
4: $distance(u) = 0$
5: 探索ノード集合 $Q \leftarrow u$；
6: **while** $|Q| > 0$ **do**
7: $v = select(Q, l)$
8: **if** $v$ が同型化できるノードの条件を満たす **then**
9: return $v$；
10: **else**
11: $Q$ から $v$ を取り除く；
12: **for** each $v$ の隣接ノード $w$ **do**
13: **if** $distance(w) > distance(v) + 1$ **then**
14: $distance(w) = distance(v) + 1$；
15: $w$ を $Q$ に入れる．
16: **end if**
17: **end for**
18: **end if**
19: **end while**

**Algorithm 3**：関数：ノード集合からノードを選択する優先順位 *select*

**Input:** ノード集合 $Q$，接続先のラベル $l$;
**Output:** 選択するノード $v$;

1: $v \in Q$ において，$distance(v)$ が最小のノード集合を $V$ とする；
2: **if** $|V| \geqq 2$ **then**
3: $S$ を $V$ 内で最少次数を持つノードの集合とする；
4: **if** $|S| \geqq 2$ **then**
5: return ラベルが $l$ と最も類似する $v \in S$；
6: **else**
7: return $v \in S$；
8: **end if**
9: **else**
10: return $v \in V$；
11: **end if**

### 4.1 実験環境とデータセット

全ての実験は，オペレーションシステムが 64bit, Windows 8 Pro, プロセッサが Intel Core i7-3930K CPU 3.20GHz，実

装メモリが 64.0 GB の PC を用いて進めた．プログラム言語は Java を使用し，JRE 64bit version 1.6.0_65 上で実行した．

評価実験には人工のデータセットを使用した．このデータセットは，Python のライブラリである networkX を用いてスモールワールド性を持ったグラフを作成した．使用した人工グラフのノードの数，エッジの数，ラベル数はそれぞれ $(300, 600, 3)$ である．また，全ての評価実験において 5 つの異なるデータセットを用いて平均値を計測した．また，今回はグラフトポロジーについて視点を置いているため，全実験においてコスト関数のパラメータは，$\alpha = 0, \beta = 1.0, \gamma = 1.0$ とした．

### 4.2 評価指標

本研究では，分析の精度を落とさないためにユーティリティとして 2 ノード間の距離関係の変化に焦点を当てている．我々は，既存手法 (Algorithm1) と提案手法 (Algorithm2) の比較をするために次の二項目の評価実験を行った．一つ目は，ノイズエッジの追加による接続先のノードとの距離の計測である．二つ目は，グラフ $G$ における平均パス長の計測である．

#### 4.2.1 接続先のノードの選択

提案手法が 2 ノード間の距離関係を崩さないようになっているかを確かめるために，匿名化パラメータ $k$ の値に従ってノイズエッジの追加によって選択される接続先のノードが元のグラフにおいてどのような距離であるかについて計測する必要がある．つまり，短い距離にあるノード間にエッジを追加するほど良い．そこで，平均してどのような距離にあるノードが選ばれているのか確認する．

#### 4.2.2 平均パス長の計測

任意の 2 ノード間の距離が変化してデータ分析の精度を低減させていないか確認するために，$k$ の値に従って平均パス長 (APL(Average Path Length)) を計測する必要がある．平均パス長とは，グラフ $G$ 内の全てのノードの組についてのパス長の平均のことである．パス長とは，最短パスの長さのことを指す．グラフ $G$ の平均パス長は式 (3) で測ることができる．式 (3) 中の $N$ はグラフ $G$ 内の全てのノード数で，$n_i$ はグラフ $G$ 内に含まれるノードのことである．平均パス長は匿名化前の SN データでの値に近いほどユーティリティが保存されていると言える．

$$APL_G = \frac{2}{N(N-1)} \sum_{\forall n_i, n_j \in G} d(n_i, n_j) \qquad (3)$$

### 4.3 実験結果

4.2 節で述べた二項目の評価指標に則って実験を行った．本節ではその結果を示し，考察する．ノイズエッジによるノード間の距離の平均の計測結果を図 4 に，平均パス長の計測結果を図 5 に示す．

### 4.4 考　　察

まず，ノイズエッジによる接続先のノードとの距離に関する計測結果について述べる．図 4 の横軸は匿名化パラメータ $k$ であり，縦軸は接続元と先のノードの距離の平均値である．図 4 より，提案手法を用いた場合は既存手法を用いた場合と比較すると，$k$ の値に寄らず短い距離のノードを選択することができている．また，提案手法ではもともと距離が 1 であるノードを

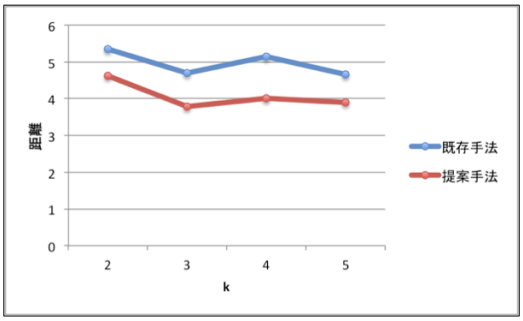


図 4　選択するノードの距離の平均の計測結果

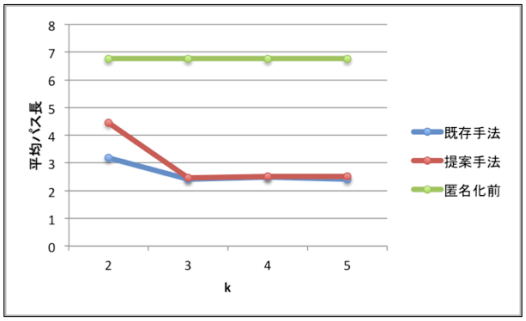


図 5　平均パス長の計測結果

選択することがあり，これによって追加するノイズエッジの数も減らすことができた．ノイズエッジの追加が多くなるほど情報損失につながるので，追加する本数が減ることは良い．

次に，平均パス長の計測結果について述べる．図 5 は，横軸が匿名化パラメータ $k$ であり，縦軸は平均パス長の値である．これは，匿名化前の平均パス長の値に近いほど，ユーティリティが保存されている．図 5 より，$k = 2$ のときは，既存手法を用いた場合よりも提案手法を用いた場合のほうが匿名化前の値に近いので，ユーティリティが保存されている．しかし $k$ の値が 3 以上になると，既存手法を用いた場合と提案手法を用いた場合でほぼ変わりない結果となった．このような結果になった原因として次の三点が考えられる．一つ目は，$k$ の値が大きくなるにつれ，追加するエッジの数が増えるためと考えた．これは，$k$ の値が大きくなるということは匿名化する対象のノードが増えるということなので，同型にする際によりエッジを貼りやすくなることがあるからである．二つ目は，グラフの性質に依存してしまったためと考えた．グラフの性質とは，ノード数に対するエッジの密度によって異なる．今回は，スモールワールド性を満たすような密度にしているため，既存手法と提案手法で変化を見ることができなかったのかもしれないからである．三つ目は，データセットのノード数が少ないためと考えた．ノード数が少ない場合，処理の後半で匿名化しきれない場合が発生するからである．

## 5. 提案手法の拡張

3. 節で述べた提案手法の拡張として，2 ノード間の距離関係の変化の更なる抑制をするためにノイズノードの追加を行なうことが考えられる．今回の実験で，距離が近いものを選択して

**Algorithm 4**：ノイズノードを用いたコンポーネントに追加するノードの選択

**Input:** グラフ $G$ 内のノード集合 $V(G)$，接続元のノード $u$，接続先のノードのラベル $l$, 距離の閾値 $d$;
**Output:** ノード $u$ と接続するノード $v$;

```
1: for each v ∈ V(G) do
2:    distance(v) = ∞
3: end for
4: distance(u) = 0
5: 探索ノード集合 Q ← u；
6: while |Q| > 0 do
7:    v = select(Q, l)
8:    if distance(v) ≦ d then
9:       if v が同型化できるノードの条件を満たす then
10:         return v；
11:      else
12:         Q から v を取り除く；
13:         for each v の隣接ノード w do
14:            if distance(w) > distance(v) + 1 then
15:               distance(w) = distance(v) + 1；
16:               w を Q に入れる.
17:            end if
18:         end for
19:      end if
20:   else
21:      ノイズノード v_n を V(G) に加える；
22:      ノイズノード v_n を VertexList に加える；
23:      return v_n；
24:   end if
25: end while
```

いるにも関わらず，同型化ができるノードを探索しているうちに遠くのものを選択している場合が発生していることが分かった．このことを踏まえて，距離の閾値を設けて閾値以上の距離をもつノードを選択しようとする際には，ノイズノードを選択先のノードとするという方法である．この拡張アルゴリズムをAlgorithm3 に示す．追加するノイズノードも匿名化対象ノードの一つとなる．(Algorithm3:22 行目)

## 6. 関 連 研 究

本節では，これまでに行われてきた本研究と関連する SN データにおける $k$ 匿名化の研究について言及する．

### 6.1 次数に着目した $k$ 匿名化

$k$-degree [3] と $k$-degree-$l$-diversity [4] の指標は，次数が同じノードが少なくとも $k$ 個保証する．$k$-degree では，ラベルを一切考慮していないため $k$-neighbor と比べて攻撃のリスクが高いと考えられる．また，$k$-degree-$l$-diversity では，$l$-多様性まで考慮している．文献 [4] では，匿名操作の際のノイズエッジを加えるときににノード間の距離の変化が大きくならないようなノイズエッジの加え方を議論していた．さらに，どのようにしても距離の変化が大きくなってしまう場合はノイズノードを加えることによってデータ分析の精度が低減しないように工夫していた．本研究でも距離の変化をさらに抑制する場合にノイズノードを加えることも今後の視野に入れている．

### 6.2 サブグラフに着目した $k$ 匿名化

$k$-isomorphism [5] の指標は，任意のノードと隣接するノードを含めたサブグラフに着目し，他に $k-1$ 個の同型のサブグラフが存在することを保証する．このサブグラフにはポップ数に制限が無い．しかし，現実的な SN データの攻撃者はそれほど多くの情報は使い切ることができないため，$k$-neighbor で考慮するコンポーネント単位での匿名化のほうがより現実的である．

### 6.3 $k$-neighbor の $l$ 多様性

本研究では，$k$-neighbor [1] のアルゴリズムの改良を行った．しかし，$k$-degree-$l$-diversity [2] では $k$-匿名化だけでなく，ラベルに関して $l$-多様性も考慮している．本研究のアルゴリズムは，$k$-neighbor-$l$-diversity の $l$-多様性を保証するアルゴリズムもそのまま適応できる．

## 7. まとめと今後の課題

本稿では，$k$-neighbor の指標を達成するためのアルゴリズムとしてノード間の距離を考慮するものを提案した．そして，いくつかのデータセットに対して実際にアルゴリズムを適応し，匿名化を行った．評価実験の結果より，提案アルゴリズムは SN データのユーティリティを保存しつつ，ノード間の距離の変化が抑制された匿名化が行えた．このことからデータ分析にも有用であることが示された．しかし，すでにスモールワールド性を持つグラフに対しては，距離の変化があまり無いようにも見える．そこで，エッジの疎密関係を考慮して分析を行う必要がある．

今後の課題として，二点考えている．一つ目は，ノイズエッジの追加の抑制である．実験結果より匿名化パラメータ $k$ が大きくなるにつれ，ノイズエッジの追加が膨大になるということがわかったので，適切なコンポーネントやノードのマッチング方法を考えたい．二つ目は，ラベルの一般化を抑制するアルゴリズムを考えたい．Bin Zhou らのアルゴリズムや本研究で提案したアルゴリズムでは，コンポーネントにノードを含める場合に接続先のノードの次数や距離を最優先にしたため，ラベルの一般化が頻繁に起こる．これを防ぐためにはラベルの一般化は極力避けつつ，他のユーティリティを考慮することが必要となる．これを達成するためのアイデアがある．それは，ノイズノードの追加によって実現できる．仮にラベルの一般化を行わないとすると距離が最小のものには目的のラベルが含まれないことがある．よって，距離が小さいノードの集合の中から目的のラベルを探索することになる．ここで，距離が小さいといえる範囲を閾値として与える．そして，この閾値の範囲内で探索を行なう．しかし，閾値内に目的のラベルが含まれない場合がある．その時にノイズノードを追加する．これによって遠くのノード同士が接続されるのを防ぎつつ，ラベルの一般化による情報損失を避けることができる．しかしながらここで注意しなければならないのが，ノイズノードも情報損失に繋がるということである．これは提案手法の拡張アルゴリズムでも同様のことが言える．ノイズノードが大量に追加されるようであれば，ラベルの一般化における制約を緩めたり，距離の閾値を大きく

するなどの工夫が必要になる．すなわち，この課題に取り組む際には，追加するノイズノードの個数，ラベルの一般化，距離に関する閾値に注目して情報損失を測り，議論する必要がある．

## 文　献


[1] Bin Zhou and Jian Pei. Preserving privacy in social networks against neighborhood attacks. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, ICDE '08, pages 506–515, Washington, DC, USA, 2008. IEEE Computer Society.

[2] Bin Zhou and Jian Pei. The k-anonymity and l-diversity approaches for privacy preservation in social networks against neighborhood attacks. *Knowl. Inf. Syst.*, 28(1):47–77, 2011.

[3] Kun Liu and Evimaria Terzi. Towards identity anonymization on graphs. In *In Proceedings of ACM SIGMOD*, pages 93–106, 2008.

[4] Mingxuan Yuan, Lei Chen, Philip S. Yu, and Ting Yu. Protecting sensitive labels in social network data anonymization. *IEEE Transactions on Knowledge and Data Engineering*, 25(3):633–647, 2013.

[5] James Cheng, Ada Wai-chee Fu, and Jia Liu. K-isomorphism: Privacy preserving network publication against structural attacks. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, SIGMOD '10, pages 459–470, New York, NY, USA, 2010. ACM.