Vol.2013-AL-144 No.8
2013/5/17

マッチングアルゴリズムを用いた大規模データk-匿名化の解法

村上 啓介[1,a)]　宇野 毅明[2,b)]

概要：医療データなどの個人情報はデータ解析を行う上で非常に役に立つ．ところが，個人情報はプライバシー保護の観点から簡単に公開することはできない．そこで，データの匿名化が必要になる．一般的な匿名化手法の一つに k-匿名化がある．k-匿名化とは，個人に対応する項目を k 個以下に絞り込まれないようにする手法である．すなわち，個人が特定される確率が $1/k$ 以下になるようにデータを部分的に加工したり削除する手法である．その際，元のデータをより多く残した上での匿名化の達成が望まれる．k-匿名化に関する先行研究は多く存在するが，大規模データの k-匿名化を行っている研究はあまり存在しない．そこで，本研究では大規模なデータテーブルに対して実時間で匿名化を行うアルゴリズムを提案する．最後に数値実験により，本提案手法が従来研究の手法より元のデータを多く残した上で匿名化を行うことおよび，本提案手法が大規模なデータテーブルに対しても実用的な時間内に匿名化を完了することを確認する．

1. はじめに

個人情報を含むデータを，プライバシーを保護しつつ利用する方法の一つにデータの匿名化がある．匿名化は知られているある個人の情報を突き合わせても，データのどの部分がその個人に対応するかわからなくするようにデータをぼかすという手法である．表 1 では，例え「Name」を全て隠したとしても，「Zip」「Gender」など，他者が入手可能な属性（準識別子と呼ぶ）を見て候補が 1 つに絞られてしまうような項目があるならば，それは個人情報を保護したと言えないだろう．

k-匿名化 [1]-[4] は，データの一部を隠すことで，準識別子を見ただけでは特定の個人に対応する項目を k 個以下に絞り込めないようにする手法である．表 2 のデータは，いくつかの部分を「*」で隠すことにより，元データのどの人にとっても，自身と対応しうる準識別子の組を持った項目が少なくとも 2 つ以上存在するため，2 匿名性を持つという．実際，Alan に対応するのは，1 番上と下から 3 番目の項目である．k-匿名性の条件は様々存在するが，近年 Wong らによって提案された条件は，データ損失量を小さく抑えることができる上に高い秘匿性を持つ [5]．ここでの情報損失量とは，ぼかされたデータ量を意味する．本研究では，表 2 のように部分的にデータを消す（「*」にする）ことで匿名化を行い，情報損失量は「*」の数で定義する．

k-匿名化問題とは，情報損失量の小さな k 匿名化されたデータテーブルを求める問題であり，「k-匿名性の条件を満たす下でのデータ損失量最小化問題」と表現される．Wong らの k-匿名性の条件を用いた場合，上述の k-匿名化問題を解くことが大変困難になる．特に，規模の大きな問題（データテーブル）において良い解（「*」の数が少ない匿名化されたデータテーブル）を求めることは難しい．この問題に対して，Wong らは，クラスタリングとランダム探索を基本とした解法を提案している．Wong らの解法は大規模な問題に対しても速く実行可能解を求めることができる．ところが，Wong らの解法はランダムなグループ分けによって解を得ているため、良質な解が求められるとは考えにくい．そこで，本研究では，最適化手法の 1 つである最小費用流アルゴリズムを応用した解法を提案して，大規模な問題においても Wong らの解法よりも良い解を求めることを目的とする．

2. モデル

識別子が付いたデータテーブル (表 3) の項目集合を X，そのデータテーブルに対応している，匿名化すべきデータテーブル (表 2) の項目集合を Y とおく．レコード集合 X, Y をそれぞれ頂点集合とした場合，図 1 のような 2 部グラフ $G = (X, Y, E)$ で表す．X は 2 部グラフの片側の頂点

[1] 青山学院大学
Aoyama Gakuin University, Fuchinobe, Chuo-ku, Sagamihara, Kanagawa, Japan
[2] 国立情報学研究所
NII, Chiyoda-ku, Tokyo, Japan
[a)] murakami@ise.aoyama.ac.jp
[b)] uno@nii.jp

表 1 データテーブル

Name	Zip	Gender	Country	Income
Alan	94221	M	US	10,000
Betiina	94112	F	US	5,000
Christina	94121	F	US	1,500
Devola	94111	M	Canada	3,000
Edmond	94222	M	Canada	30,000
Flora	94122	F	UK	20,000
Georgia	93111	M	Canada	40,000

表 2 匿名化されるデータテーブル

項目 ID	Zip	Gender	Country	Income
r'_1	9422*	M	*	10,000
r'_2	941**	F	*	5,000
r'_3	941**	F	*	1,500
r'_4	9*111	M	Canada	3,000
r'_5	9422*	M	*	30,000
r'_6	941**	F	*	20,000
r'_7	9*111	M	Canada	40,000

表 3 識別子付きデータテーブル

項目 ID	Name	Zip	Gender	Country
r_1	Alan	94221	M	US
r_2	Betiina	94112	F	US
r_3	Christina	94121	F	US
r_4	Devola	94111	M	Canada
r_5	Edmond	94222	M	Canada
r_6	Flora	94122	F	UK
r_7	Georgia	93111	M	Canada

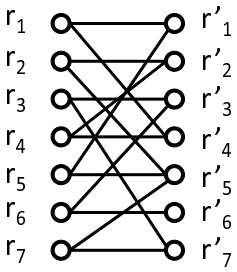

図 1 2 部グラフ

集合を，Y はもう一方の頂点集合を意味していて，E は頂点 $r_i \in X$ と頂点 $r'_j \in Y$ に隣接する枝の集合を意味している．また，レコード $r'_j \in Y$ が $r_i \in X$ のデータになりうるとき，頂点 r_i と r'_j は隣接する．逆の視点から見ると，頂点 r_i と r'_j が隣接するためには，項目 r'_j のデータに付ける「*」の場所を決める必要がある．k-匿名化問題は，このように「*」を付ける場所を決める問題である．

2.1 k-匿名性の条件

先行研究のいくつかは，以下のように k-匿名性の条件を定義している．

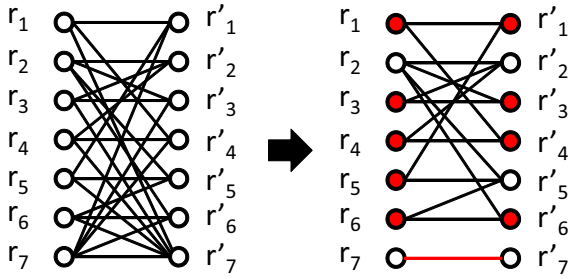

図 2 偏ったグラフ

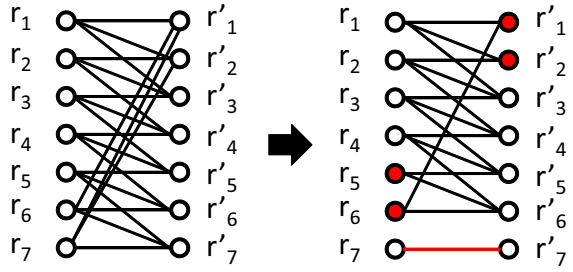

図 3 バランスの良いグラフ

定義 1 2 部グラフ $G = (X, Y, E)$ において，全ての頂点の次数が k 以上のときに k-匿名性を満たしている．

各個人が，k 個以上の項目に対応すれば k-匿名性を満たしていると言えるので，定義 1 は，この条件を満たしている．ここで，データテーブルが匿名化されたことによって図 2 の左側のような 2 部グラフができたとする．この 2 部グラフは $k = 3$ において定義 1 を満たしているので，データテーブルは 3-匿名化されていると言える．ところが，図 2 において r'_7 が r_7 に対応することが攻撃者に推測されたとする．このとき，$r_1, r_3, r_4, r_5, r_6, r'_1, r'_3, r'_4, r'_6$ で頂点の次数が 3 未満になり，その後の推測を行いやすくしてしまう (図 2 の右側)．これは，r_7, r'_7 に隣接する枝が多いことが原因である．例えば，図 3 の左側のように各頂点に均等に枝が隣接した場合において，先程と同じように r'_7 が r_7 に対応することが攻撃者に推測されたとする．このとき，r_5, r_6, r'_1, r'_2 で頂点の次数が 3 未満になるが，図 2 の例ほど多くの頂点の次数は減らない．したがって，攻撃者が推測しなければならない範囲は依然広く，推測だけで個人を特定するのは難しい．すなわち，図 2 よりも図 3 のデータテーブルの方が秘匿性は高いと言える．

以上の事を踏まえて，Wong らは秘匿性のより高い以下のような k-匿名性の条件を提案している．

定義 2 2 部グラフ $G = (X, Y, E)$ に k 個の辺素な完全マッチングが存在する場合，k-匿名性を満たしている．

定義 2 を満たす場合，攻撃者による推測が行われても簡単には個人を特定することはできない．また，「2 部グラフ内に k 個の辺素な完全マッチングが存在する」ことと，「2

部グラフが k 正則グラフを含んでいる」ことは同値である（Hall の定理より導かれる）．すなわち，k 正則グラフも含む 2 部グラフを作れば，定義 2 の条件を満たすことができる．本研究では，定義 2 の条件を用いて k-匿名化を行う．

2.2 k-匿名化問題

定義 2 の k-匿名性の条件を満たすためには，以下の式を満たす必要がある．ただし，k は「k-匿名化」の k を意味している．

$$\sum_{r'_i \in Y} x_{i,j} = k-1, \quad \forall r_i \in X \tag{1}$$

$$\sum_{r_i \in X} x_{i,j} = k-1, \quad \forall r'_j \in Y \tag{2}$$

$$x_{i,j} = \{0,1\}, \quad \forall r_i \in X, \ \forall r'_j \in Y \tag{3}$$

式 (1)-(3) は，2 部グラフ $G=(X,Y,E)$ が k 正則グラフを含むための条件であり，$x_{i,j}$ は 0-1 変数を表していて，頂点 r_i が頂点 r'_j に隣接する場合には $x_{i,j}=1$，そうでない場合には $x_{i,j}=0$ となる．また，式 (1),(2) は各頂点の次数が $k-1$ であることを意味している．k 匿名性を満たすためには，本来次数は k である必要があるが，自分同士を結ぶ自明なマッチングが必ず 1 本存在するので，実質次数は $k-1$ になる．このとき，k-匿名化問題は，条件 (1)-(3) を満たしつつ，必要な「*」の数を最小にする隣接 ($x_{i,j}$ の値) を決める問題となる．

この問題を言い換えると，枝集合 E の中から実際にマッチングに使用する枝集合 $E' \subset E$ を求める問題となる．厳密解を求めるためには，E はすべての枝を含む必要があり，枝の数 $|E|$ は $|X| \times |Y|$ となる．本研究では，大規模なデータを想定しているので $|E|$ はかなり大きな値になる．その結果，枝の重みの計算時間や枝を保持するためのメモリ使用量なども膨大になってしまい，現実的に解くことのできない問題を作ってしまうことになる．そこで，本研究では E を重みが小さな枝に限定する．ここでの各枝の重みは，1 つもマッチングが存在していない状況，すなわち表 2 のようなデータテーブルに 1 つも「*」が付いてない状況を想定して計算される．（3 章で述べるが，本来枝の重みは「*」の追加状況に応じて動的に変化する．）枝の限定により最適性は失われるが，大規模データにおいても現実的に解くことのできる 2 部グラフ $G=(X,Y,E)$ を作成できる．

3. アルゴリズム

k-匿名化問題は，「*」が付いている場所が変化すると，枝の重み（必要な「*」の数）が変化するために解くことが難しい．例えば，「*」が一つも付いてない状況で図 4 のように r_2 と r'_1, r'_3 間の枝の重みを計算する．r'_1 が r_2 に対応するためには，3 つの「*」が必要なので，r_2, r'_1 間の枝の重みは 3 になる．同じように r_2, r'_3 間の枝の重みは 2 になる．一方で図 5 のように，まず r'_1 と r'_3 がすでに r_4 に対応していて「*」が付いている状況を考える．このとき，r_2, r'_1 間の枝の重みは 2，r_2, r'_3 間の枝の重みは 1 となる．このように，同じ枝であっても状況に応じてその重みは変化する．ただし，$k=2$ のときは各頂点の次数は 1 になるので，1 つも「*」が付いていない状況で計算した枝の重みが厳密な値となる．

本章では，$k>2$ のときに最小費用流アルゴリズムを用いた 2 段階の発見的解法を提案する．第 1 段階で貪欲法を基本とした手法で初期の実行可能解を求める．第 2 段階では，負閉路消去法を用いて第 1 段階で求めた解を改善する．

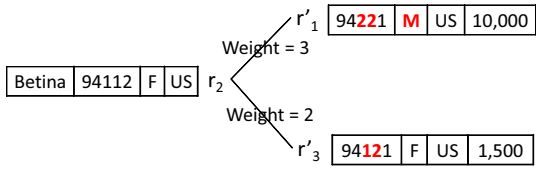

図 4 枝の重み計算

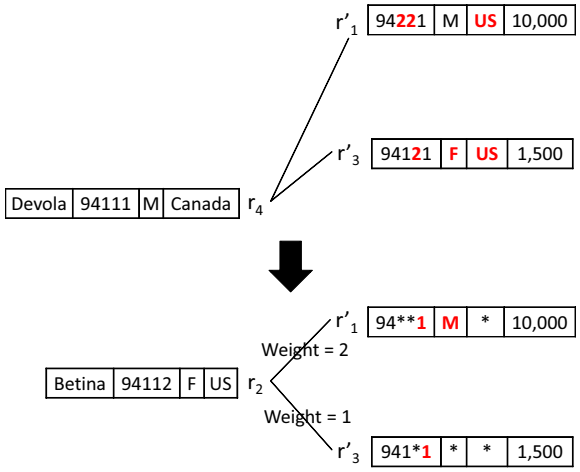

図 5 枝の重みの動的な変化

3.1 貪欲法を用いた初期の実行可能解の導出方法

第 1 段階において，初期の実行可能解を求める方法を説明する．まず，1 つも「*」が付いていない状況で各枝の重みを求めて，最小重み完全マッチング問題 (式 (1) と (2) において $k=2$ とした問題) を解く．次に，その解に応じてデータテーブルに「*」を追加して，各枝の重みも「*」の追加に応じて更新する．この更新によって，枝の重みの動的な変化に対応する．そして，新たに求めた枝の重みを用いて，再び最小重み完全マッチング問題を解く．これを $k-1$ 回繰り返す．ここで，表 3 と表 2 からなる図 1 のような 2 部グラフを想定して，実行可能解を求める手順を以下に示す．

Step 0　2 部グラフの右側のデータベースに 1 つも「*」が付いていない状況で，最小重み完全マッチング問題を作る．

Step 1　最小重み完全マッチング問題を解く．

Step 2　Step 1 の最小重み完全マッチング問題の解 (マッチング) によって，2 部グラフの右側のデータベースに「*」を追加する．

Step 3　最小重み完全マッチング問題を $k-1$ 回解いたなら，2 部グラフの右側のデータベースを k-匿名化問題の初期解として出力する．そうでないならば，Step 4 に進む．

Step 4　Step 2 で更新した 2 部グラフの右側のデータベースを用いて，新たに最小重み完全マッチング問題を作って (枝の重みを更新して)Step 1 へ戻る．

Step 1 の最小重み完全マッチング問題は，最小費用流問題の特殊ケースと見なすことができるので，本研究ではプリフロー・プッシュ法 [6] を用いて解く．また，Step 1 で，枝が足りず完全マッチングが存在しないこともあり得る．そのときは，完全マッチングができるように適当に枝を選ぶものとする．この第 1 段階のアルゴリズムは厳密解法ではないので，解を改善する余地はある．そこで，第 2 段階で解の改善を行う．

3.2　負閉路消去法による実行可能解の改善

第 2 段階では，実行可能解を改善する．ここでは，最小費用流問題において実行可能解を改善する方法である負閉路消去法を用いる．まず，2 部グラフ $G=(X,Y,E)$ を次のような有向グラフにする．現在の実行可能解に使用されている枝を右の頂点から左の頂点方向と定義する．それ以外の枝は，左の頂点から右の頂点方向と定義する (図 6 の左側)．次に，枝の向きに沿ってグラフ中の負閉路を発見して，その負閉路に沿って，枝の向きを入れ替える (図 6 の右側)．すなわち，解に使用する枝を入れ替える．これにより，実行可能性を維持しつつ解を改善する (「*」の数を減らす)．ただし，これまでの 2 部グラフ $G=(X,Y,E)$ はそのまま使用できない．なぜなら，2 部グラフ G の枝 1 本だけでは，「*」の増減数を計算できないからである．「*」の増減数は，2 部グラフ G の右側の頂点 (匿名化されるデータテーブル) に入る枝と出る枝の組合せで決まる．そこで，新たなグラフを $G'=(X',E')$ と定義すると，$X'=X$ となり，E' は頂点 $\forall r'_j \in Y$ に入る枝と出る枝の組合せの集合で表される．

負閉路消去法では，いくつも負閉路を発見しなければならないので，1 つの負閉路を速く発見する必要がある．本研究では，大規模データテーブルを想定しており，データテーブルの規模の増大化に伴って，2 部グラフ G のサイズも大きくなるので，効率的な負閉路発見が必要である．そ

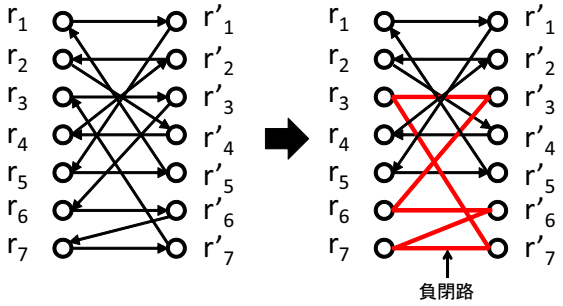

図 6　負閉路消去法

こで，ダイクストラ法のように各頂点への距離を更新していく，ラベル更新法を用いた負閉路発見法を提案する．ラベル更新法は，各頂点のラベルを順次更新していく解法であるが，もし負閉路が存在する場合は，同じ頂点が何度も更新されることになる．本手法では，各頂点の更新回数に着目して，更新回数が多い頂点が存在した場合にパスをさかのぼることをする．そして，パスをさかのぼっていき同じ頂点に戻ってくれば，負閉路を発見できる．

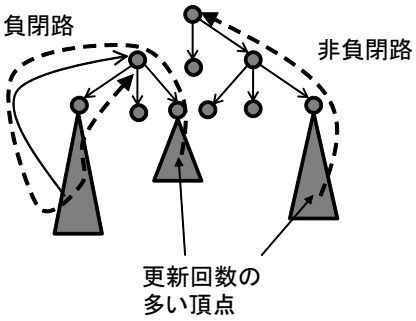

図 7　経路木

図 7 のようにラベル更新法の経路木を保持しておけば，更新回数の多い頂点から木をさかのぼることで負閉路を発見することができる．ただし，閉路ができた場合は木構造ではなくなる．また，木をさかのぼる起点となる頂点が負閉路に含まれていなくても，さかのぼった先の頂点が負閉路に含まれていれば，負閉路を発見できるので比較的高い確率で負閉路を発見することができる．この手法は，正確にすべての負閉路を発見することはできないが，大規模なグラフにおいて，効率的に負閉路を発見できると考える．そして，終了条件を満たした時点で負閉路消去法を終了する．以下にラベル更新法を用いた負閉路消去法を示す．

Step 0　2 部グラフ $G=(X,Y,E)$ をグラフ $G'=(X',E')$ に変換し，任意の 1 頂点をびその頂点のラベルを 0 に，その他の頂点のラベルを ∞ 設定する．

Step 1　ラベル更新法の手順に従い頂点のラベルを更新して，経路木も更新する．

Step 2　更新された頂点のラベル更新回数が t 以上になっ

た場合は，その頂点から経路木をさかのぼる．そうでない場合は，Step 1 へ戻る．

Step 3 もし負閉路が発見された場合は Step 4 へ，そうでなければ，Step5 へ．

Step 4 データテーブルの「*」の位置を変更して，負閉路に沿ってグラフ G の枝の向きを逆にする．それに伴って，グラフ G' の枝の向きも逆にする．さらに，その負閉路内の頂点と，負閉路内の頂点の子孫のラベルを全て ∞ に設定して，それら全ての頂点間の親子関係を解消する．その後，Step 1 へ戻る．

Step 5 終了条件を満たしていれば Step 6 へ，そうでなければ Step 1 へ戻る．

Step 6 匿名化されたデータテーブルを解として出力する．

Step 2 における t は 5 から 10 ぐらいに設定する．Step 4 において負閉路が発見された場合は枝の向きが逆になり，そのまま親子関係を保つことはできないので，負閉路の頂点とその負閉路内の頂点の子孫を全て初期状態に戻している．Step 5 における終了条件は，計算時間やラベル更新法の繰り返し数などが考えられる．

3.3 問題の分割

3.1 章と 3.2 章の手法で，匿名化されたデータベースを求めることができるが，データベースの規模が大きくなるにつれて，良い解を求めることが難しくなる．大規模な問題に対するアプローチの 1 つで，問題を分割する方法がある．そこで，規模の大きな問題に対しては，元のデータベースをあらかじめ分割しておいて，分割された各データベースにおいて，3.1 章と 3.2 章の手法を用いる．

4. 数値実験

本研究で提案した手法 (Two-Phase Algorithm) と Wong らによって提案された手法を比較する．以下に Wong らの手法を簡単に説明する．

Step 1 データテーブルを辞書順に並び替える．

Step 2 データテーブルを上から k から $2k$ のサイズにクラスタリングする．

Step 3 各クラスタ内において，k-匿名性の条件である定義 2 を満たすように個人と項目の対応を決める (2 部グラフにおいて頂点間に枝を張る)．

Step 4 Step 3 で求めた解に対してランダムに閉路を見つけて解を作りかえる．

Step 3 での定義 2 を満たすような対応の見つけ方は，図 1 の 2 部グラフで $k = 2$ を例に説明すると以下のようになる．まず，r_1 と r'_1, r'_2 間に枝を張る．次に，r_2 と r'_2, r'_3 間に枝を張る．これを繰り返していき，最後は r_7 と r'_7, r'_1 間に枝を張る．こうしてできた解は明らかに定義 2 の条件を満たしている．Step 4 でランダムに閉路を見つけて入れ替えている理由は，Step 3 での実行可能解の導出方法が攻撃者に見破られてしまった場合のリスクを回避するためである．

また，インスタンスは以下の 11 個を準備する．

Occ_100k（10 万人データ）
Occ_200k（20 万人データ）
Occ_300k（30 万人データ）
Occ_400k（40 万人データ）
Occ_500k（50 万人データ）
Random_100k，Random_100k2（10 万人データ）
Random_1m，Random_1m2（100 万人データ）
Random_10m，Random_10m2（1000 万人データ）

Occ_は現実のデータセット (http://www.ipums.org)[8], [9] で，全てのインスタンスとも属性数は 8 である．Random_は，乱数を発生させて自作したインスタンスで，属性数は 10 である．また，Random_には同じデータ数で乱数のばらつきを変えた 2 つのインスタンスを作る．インスタンス名の最後に “2”が付いている方がばらつきが大きく設定されている．また，k の範囲は 3 から 10 までとした．

Wong の手法および提案手法によって匿名化されたデータテーブルにおける「*」の数の割合 (Information loss)[7], [10] と計算時間を以下に示す．「*」の数の割合は，「*」の数 ÷(属性数 × データの人数) で計算される．属性数とデータの人数は，インスタンスごとには固定値なので，「*」の数を最小化することと「*」の数の割合 (Information loss) を最小化することは同じ意味である．

図 8–17 に「*」の数の割合 (Information loss) の結果を示す．

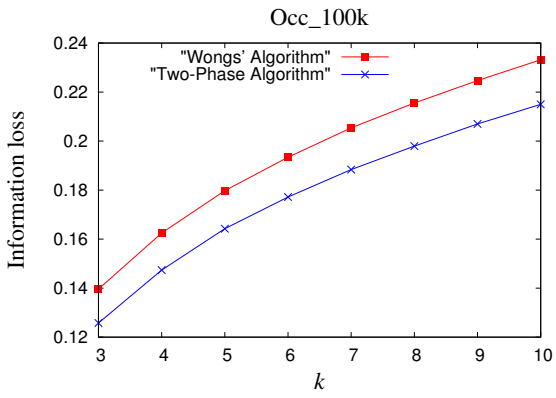

図 8 Occ_100k

図 8-17 より，全てのインスタンスにおいて提案手法の方が Information loss が小さくなった．Occ_では全てのインスタンスにおいて k が変化しても，2 つの手法の Information loss の差は 0.01 から 0.03 の間になった．また，データ

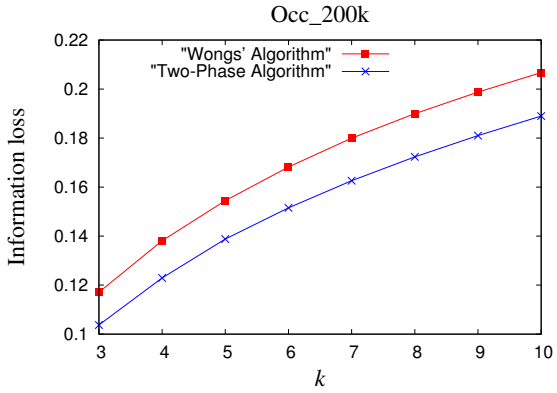

図 9 Occ_200k

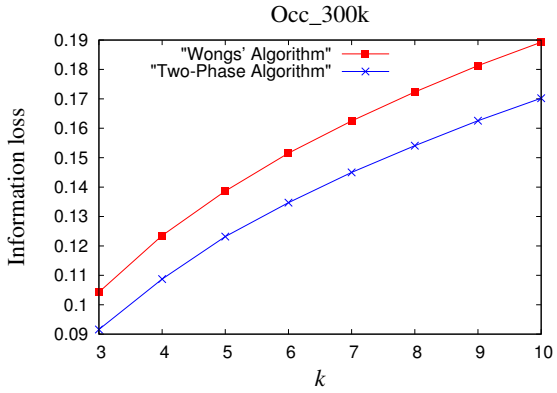

図 10 Occ_300k

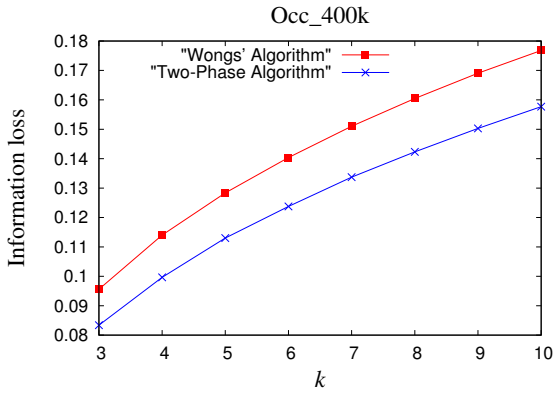

図 11 Occ_400k

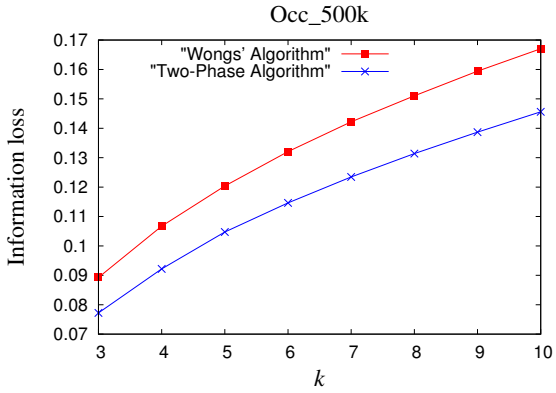

図 12 Occ_500k

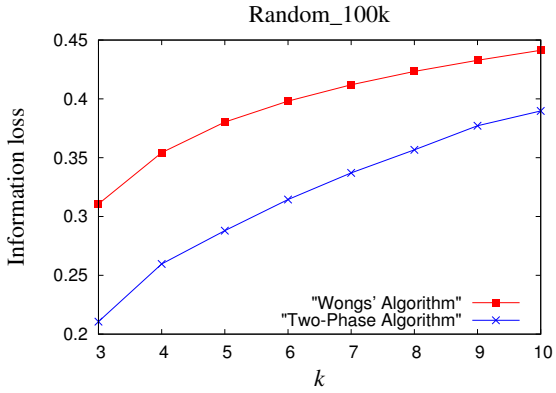

図 13 Random_100k

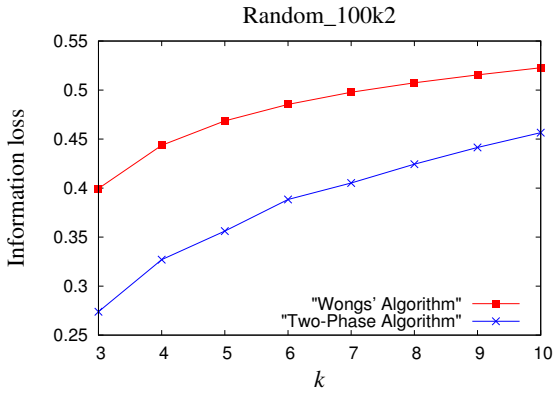

図 14 Random_100k2

数が大きくなるにつれて 2 つの手法とも Information loss が小さくなった．Random_では，ばらつきの大きなインスタンスの方が，2 つの手法の差が小さくなった．これは，ばらつきの大きなインスタンスを匿名化するためには多くの「*」が必要になるので，ランダムに探索を行う Wong らの手法でもあまり悪い結果にならなかったためと考えられる．また，Random_1m と 10m 以外は，k が大きくなるにつれて，2 つの手法の Information loss の差が小さくなった．これも，上述の理由と同じで k が大きくなると，匿名化するために多くの「*」が必要になるからであると考えられる．ただ，Random_1m と 10m においては k が大きくなると 2 つの手法の Information loss の差も大きくなった．この結果は，Occ_の結果と類似している．これらの共通点としては，Information loss が小さいことが挙げられる．したがって，Information loss が小さくなるインスタンスにおいて，k を大きくすると 2 つの手法の Information loss の差が大きくなる傾向があることが推測される．

表 4-13 より，計算時間に関しては提案手法よりも Wong の手法の方がかなり小さくなった．ただ，提案手法の方も，1000 万人データの匿名化を最大でも 1 日と数時間で行っているので実用的と言える．

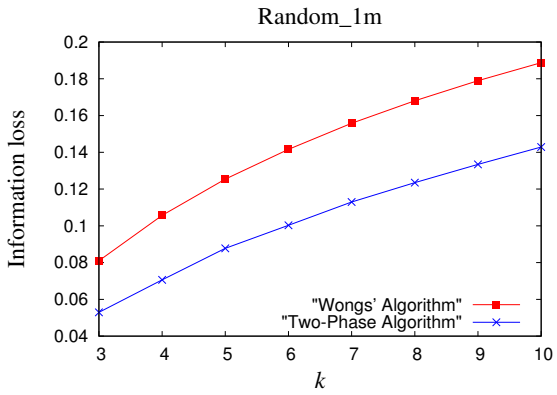

図 15 Random_1m

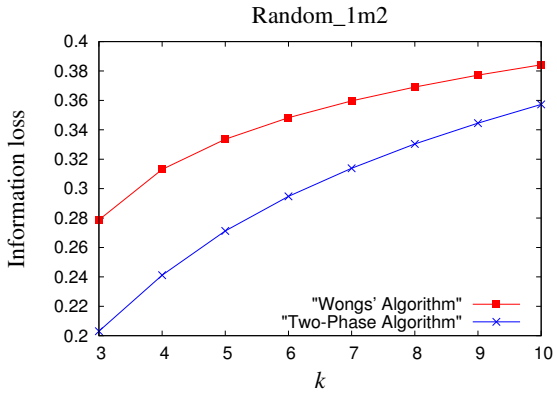

図 16 Random_1m2

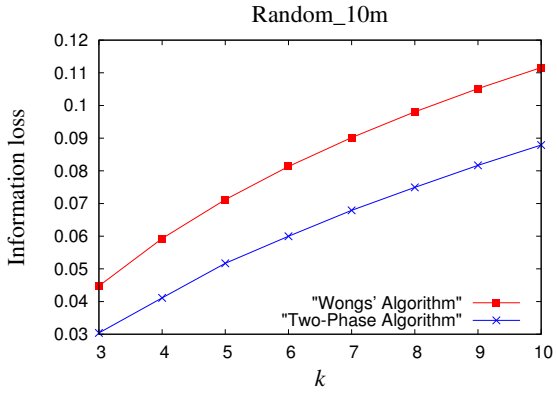

図 17 Random_10m

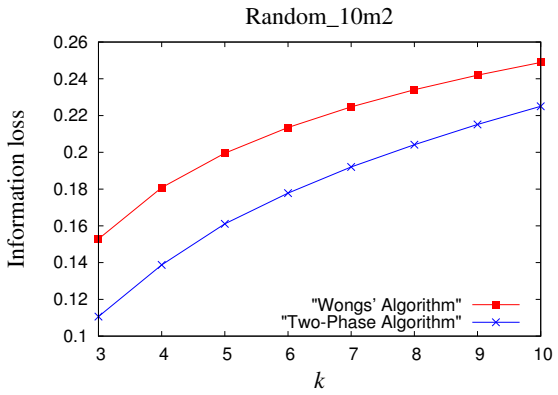

図 18 Random_10m2

5. おわりに

本研究では，2 段階から成る大規模データの k-匿名化手法の提案を行った．まず第 1 段階において，貪欲法をベースにして最小費用流のアルゴリズムを用いることで初期の実行可能解を出した．次に，第 2 段階で負閉路を消去することで解の改善を行った．また，数値実験では従来法である Wong らの手法に比べて情報損失に関して優れた結果を出した．計算時間に関しては，Wong らの手法の方がかなり速かったが，提案手法も実時間内に匿名化を完了しており十分有用であることが確認された．

参考文献

[1] L. Sweeney: k-anonymity: A model for protecting privacy, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systemsw*, Vol. 10, pp. 557–570 (2002).

[2] P. Samarati and L. Sweeney: Protecting Privacy when Disclosing Information: k- Anonymity and its Enforcement through Generalization and Suppression, *Technical Report SRI-CSL-98-04,* SRI Computer Science Laboratory (1998).

[3] R. J. Bayardo and R. Agrawal, Data Privacy through Optimal k-Anonymization, *Proceedings 0f 21st International Conference on Data Engineering*, pp.217–228 (2005).

[4] K. LeFevre, D. J. DeWitt and R. Ramakrishnan: Incognito: Efficient Full-Domain k-Anonymity, *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data,* Vol. 21, pp. 49–60 (2005).

[5] W. K. Wong, N. Mamoulis and D. W. L. Cheung: Non-homogeneous generalization in privacy preserving data publishing, *Proceedings of the the 2010 ACM SIGMOD International Conference on Management of Data*, pp.747–758 (2010).

[6] A. V. Goldberg: An Efficient Implementation of a Scaling Minimum-Cost Flow Algorithm, *Journal of Algorithms* , Vol. 22, pp. 1–29 (1997).

[7] G. Ghinita, P. Karras, P. Kalnis, and N. Mamoulis: Fast data anonymization with low information loss, *Proceedings of the 33rd international conference on Very large data bases*, pp. 758–769 (2007).

[8] X. Xiao and Y. Tao: Anatomy: Simple and effective privacy preservation, *VLDB 2006. Proceedings of the 32nd international conference on Very large data bases*, pp. 139–150 (2006).

[9] X. Xiao and Y. Tao: M-invariance: Towards privacypreserving re-publication of dynamic datasets, *Proceedings of the the 2007 ACM SIGMOD International Conference on Management of Data*, pp. 689–700 (2007).

[10] J. Xu, W. Wang, J. Pei, X. Wang, B. Shi, and A. W.-C. Fu: Utility-based anonymization using local recoding, *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 785–790 (2006).

表 4 計算時間 (Occ_100k)

k	3	4	5	6	7	8	9	10
Wong	0.23s	0.24s	0.23s	0.25s	0.28s	0.26s	0.26s	0.28s
Two-Phase	5m55s	7m53s	9m56s	12m33s	13m54s	16m3s	17m50s	20m4s

表 5 計算時間 (Occ_200k)

k	3	4	5	6	7	8	9	10
Wong	0.45s	0.49s	0.48s	0.48s	0.51s	0.49s	0.52s	0.51s
Two-Phase	12m40s	14m58s	21m19s	22m52s	30m2s	32m56s	37m59s	42m1s

表 6 計算時間 (Occ_300k)

k	3	4	5	6	7	8	9	10
Wong	0.65s	0.67s	0.67s	0.70s	0.73s	0.76s	0.73s	0.76s
Two-Phase	20m4s	25m50s	31m9s	40m37s	46m1s	53sm53	59sm23s	62m43s

表 7 計算時間 (Occ_400k)

k	3	4	5	6	7	8	9	10
Wong	0.93s	0.88s	0.93s	0.90s	0.93s	0.96s	1.01s	1.07s
Two-Phase	22m15s	31m7s	37m43s	46m44s	58m37s	63m3s	72m56s	67m39s

表 8 計算時間 (Occ_500k)

k	3	4	5	6	7	8	9	10
Wong	1.12s	1.15s	1.13s	1.16s	1.18s	1.21s	1.26s	1.27s
Two-Phase	26m18s	34m46s	42m45s	55m42s	63m28s	66m31s	69m32s	92m29s

表 9 計算時間 (Random_100k)

k	3	4	5	6	7	8	9	10
Wong	0.20s	0.18s	0.26s	0.24s	0.23s	0.28s	0.28s	0.34s
Two-Phase	7m53s	7m59s	12m59s	15m16s	18m14s	19m53s	19m46s	26m33s

表 10 計算時間 (Random_100k2)

k	3	4	5	6	7	8	9	10
Wong	0.20s	0.20s	0.20s	0.26s	0.24s	0.28s	0.29s	0.35s
Two-Phase	8m11s	8m6s	13m22s	12m52s	18m53s	21m37s	24m32s	26m4s

表 11 計算時間 (Random_1m)

k	3	4	5	6	7	8	9	10
Wong	1.74s	1.82s	1.99s	1.99s	2.21s	2.27s	2.43s	2.49s
Two-Phase	41m40s	56m22s	78sm41	96sm28s	112m46s	130m6s	155m38s	154m22s

表 12 計算時間 (Random_1m2)

k	3	4	5	6	7	8	9	10
Wong	1.90s	1.90s	2.058s	2.33s	2.38s	2.55s	2.74s	2.83s
Two-Phase	65m55s	90m43s	107m4s	131m24s	154m55s	172m52s	200m36s	225m53s

表 13 計算時間 (Random_10m)

k	3	4	5	6	7	8	9	10
Wong	20.43s	20.85s	21.47s	22.14s	22.97s	23.86s	24.81s	25.91s
Two-Phase	420m22s	574m22s	735m11s	890m27s	960m24s	1214m50s	1427m	1604m19s

表 14 計算時間 (Random_10m2)

k	3	4	5	6	7	8	9	10
Wong	21.35s	22.21s	23.13s	24.35s	25.7s	27.21s	28.74s	30.06s
Two-Phase	516m11s	747m3s	907m41s	949m23s	1173m1s	1321m34s	1503m47s	1761m58s