Vol.2014-AL-146 No.2
2014/1/31

データ研磨によるクリーク列挙クラスタリング

宇野 毅明[1,a)]　中原 孝信[1]　前川 浩基[1]　羽室 行信[2]

概要：近年の IT 技術の発達により，ビッグデータを用いたデータ解析はますますその重要性を増している．しかし，ビッグデータ解析には，データの大きさ以外にも多様性という大きな困難がある．多様なデータは，それぞれ異なる特徴を持つグループから構成されているため，全体的に解析することが困難であり，まずグループ構造の解明が重要である．既存のクラスタリング手法やパターンマイニングによってグループ構造の解明にアプローチすると，解が大量，少数のグループしか見つけられない，類似する大量の解を生成，見つかるグループの大きさに大きなばらつきがある，計算コストが大きすぎる，といった難点にぶつかることになる．本稿では，グラフクラスタリング問題に対して，そもそもデータがどのようになっていればグループ構造が抽出しやすいかを考え，ノイズの少ない明確なデータを定義し，ノイズ混じりの生データを，そのグループ構造を壊さないように明確なデータへと変換する，データ研磨という手法を紹介する．また，データ研磨アルゴリズムとデータ研磨を行ったグラフが持つ数理的な構造を紹介し，将来的に「明確なデータ」を研究するための礎とする．

キーワード：データクリーニング，データ解析，クリーク列挙，パターン発見，クラスタリング

Clustering by Clique Enumeration and Data Cleaning like Method

Abstract: Recent development on information technology has made bigdata analysis more familiar in research and industrial areas. However, bigdata has big difficulties on diversity, other than its huge size. Data with much diversity is usually composed of many groups each of those has its original feature, thus data analysis from the global structures usually fails to capture the details of the data. To analyze the data correctly, capturing the group structure is important. Existing clustring algorithms and pattern mining algorithms aim to extract the group structures from the data. However, they usually find huge number of solutions, too few groups, many similar groups, or groups with large biased sizes, and often take long computation time. In this paper, we address the graph clustering problem, and discuss what are good graphs in that we can easily capture the group structures. From the discussion, we define a graph class that is a model of noiseless clarified graph. We then propose a data cleaning like method that modifies the given data graph to a clarified graph without breaking the group structures. We also show some mathematical and algorithmic properties for the graph class and the modifying algorithm.

Keywords: data cleaning, data analysis, clique enumeration, pattern mining, clustering

1. Introduction

Web 上のテキスト，スマートフォンやナビゲーションの行動記録，位置情報，購買履歴や病歴など，近年は社会生活に関わる大型のデータがある程度簡単に手にはいるようになってきた．これらのデータを解析する際に問題になることの一つが，大きな多様性である．多様性の高いデータの場合，ある程度明かな特徴を共有する局所的なグループが存在することが多く，その特徴が個体の特徴や行動に深い関わりを持っていることが多い．そのグループを抽出することなく，単に全体的な解析を行うと，これらの特徴は薄まってしまい，発見することが非常に困難になる．そのため，多様性の大きなデータでは，これら局所なグループに共通する特徴を見つけ出したり，あるいはそのようなグ

[1] 国立情報学研究所
2-1-2, Hitotsubashi, Chiyoda, Tokyo 101–8430, Japan
[2] 関西学院大学経営戦略研究科
1-155, Ichiban-cho, Uegahara, Nishinomiya, Hyogo 622–0891, Japan
[a)] uno@nii.ac.jp

ループを見つけ出すという作業が重要である．

このようなデータ処理は，データマイニングやクラスタリングなどの分野で盛んに研究されてきた．その結果，例えば多様性がそれほど大きくなく，局所的なグループの数が数十程度であれば，クラスタリングや機械学習の技術で，それらグループを見つけることができるようになりつつある．また，多数のグループに対しては，クリークパーコレーション [11]，ニューマンクラスタリング [3] といった手法が開発されてきており，ある程度の成功を収めている．また，局所的に現れている特徴を列挙する手法としては，パターンマイニング [1], [12] が深く研究されており，多くの効率的な計算アルゴリズムが提案されている．HITS やブリーフプロパゲーション [4] といった，ランダムウォークに基づく，局所的に強いつながりを持つ構造を抽出するアルゴリズムも提案されている．しかし，どの手法も問題点を抱えており，決定打になっていないというのが現状である．

機械学習的な手法（サポートベクターマシン [2] など）を用いて，データを再帰的に分割する方法や，k-means[9] といった方法では，見つかるグループの数が小さい，多くの局所的な構造が破壊されてしまう，大域的な情報に基づいて行うため精度が粗い，といった問題点がある．また，１つの項目が２つ以上のグループに属するような，ソフトクラスタリングが難しいという欠点もある．クリークパーコレーション [11] やニューマンクラスタリング [3] などの手法は，全体の数十パーセントを占める非常に大きなグループ数個と非常に小さなグループを見つけてしまう，といった大きな偏りを持つことが多い．計算時間が非常に長くなるのも大きな弱点である．パターンマイニングは計算的には非常に効率が良く [6], [13]，短い時間で全ての解を列挙できることが多い．しかし，出力される解の数が膨大になることが多く，多くのパターンは意味を持たないか，あるいは非常に類似した意味を持つことが多い．これら大量の解の中から，意味を持たない物を捨て，代表的な物だけを取り出す作業は現実的に難しい．ランダムウォーク型のアルゴリズムは網羅性に乏しく，強いつながりを持つグループが存在すると，その周辺のグループは発見されにくくなる．また，初期解を変更して多数のトライアルを行うと，類似する解を大量に生成することとなり，パターンマイニングと同じ問題が生じる．しかも，１つ１つのグループ発見に比較的長い時間がかかるため，全体として非常に大きな計算コストを要することになる．

上記のような問題点から，現在大きな多様性を持つデータから多くの局所的なグループ構造を見つける問題に対しては，定石となりうるような手法は存在していない．そこで本稿では，既存手法とはまったく異なるアプローチにより，そのようなグループを見つけ出す手法，データ研磨について，グラフクラスタリングでの手法を解説する．我々

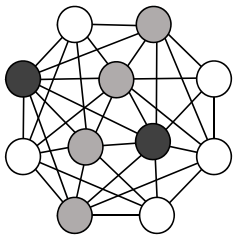

図 1 ある程度大きな密構造 (擬クリーク) に含まれる頂点の組には多くの頂点が共通して隣接する

はまず，グループ構造が見えやすいグラフとはどのようなものかを考える．グラフクラスタリングでは，通常，局所的に枝が密である部分をグループ（クラスタ）であると考える．しかし，実データの多くでは密な部分の境界が曖昧であり，取りようによっていくらでも類似するグループ構造が見つかってしまう．逆に，全てのグループがクリークになっていて，グループに含まれない部分は疎になっているのであれば，境界がはっきりしており，明確に構造を発見できる．つまり，密度の高い部分はクリークに，そうでない部分には枝がない，というグラフが，グループ構造が明確になっているグラフと考えることができる．

このような「明確」なグラフは，同じ密構造に含まれる頂点の間には必ず枝があり，同じ密構造に含まれない頂点の間には枝がないようなグラフ，と考えることができる．ただしこの条件は計算的に非常に困難であり，局所的な構造を持ちにくいため，ここでは頂点の対に関する条件付けで特徴付ける．ある頂点の対が共に密な構造に含まれるのであれば，それらの頂点両方に隣接する頂点がある程度，ある閾値 k 以上存在するはずである．この条件は，頂点対が同じ密構造に含まれるための必要条件としてモデル化できる．よって，この条件を満たす頂点対には枝があり，そうでない頂点対には枝がないようなグラフは，「明確」なグラフであると考えられる．

明確なグラフが導くグラフクラスは，興味深い構造を幾つか持っている．その一つが極大クリークの数が，頂点数 n に対して $O(n^k)$ 個で抑えられるということである．これは，明確なグラフは本質的に爆発的に多くのグループ構造を持つことはないということを暗示している．同時に，クリークに関する最適化問題が多項式時間で解けることも意味している．

データ研磨は，与えられたデータグラフを，そのグループ構造を壊さずに明確なグラフにする，あるいは明確なグラフに近づける物である．具体的には，上記の条件を満たす頂点対には枝を張り，満たさない頂点対からは枝を除去する．これを１回，あるいは複数回行うことで，明確化されたグラフを得る．データを変更しているため，多少気持ちの悪い部分もあるが，画像処理分野で著名な画像を明確化する手法であるギブスサンプリングと基本的な発想は同じであることを考えると，中規模な構造を見るためのアプ

ローチとしては自然な物であるとも考えられる．ギブスサンプリングは，画像の各画素を，その周辺の状況に従って確率的に変化させる．データ研磨も，共通して隣接する頂点の数という局所的な条件から，グラフの枝の追加削除を行う．両者共に，微細な情報は失うが，ある程度以上の大きさを持つ中規模な構造がはっきりと見えるようにする効果がある．実際，データ研磨の計算実験を行うと中規模のグループ構造を多数発見することができる上，その数はパターンマイニングのように爆発することはない．

本稿では，データ研磨の一反復により，ある程度の密度を持つ構造が必ずクリークになり，また，それより多少密度が薄い構造は密度が必ず増加することを証明する．これにより，データ研磨により密構造が失われてしまうことがないということが保証される．データ研磨の一反復は，各頂点対に対して共通する隣接頂点の数を計算することであり，これは直接的な方法では非常に時間がかかる．本稿では，疎なグラフに対応した計算方法を用いることで，次数分布がべき乗則に従うようなグラフに対する計算時間が入力グラフのほぼ線形で抑えられることを示す．

2. 記法

グラフの**クリーク**は，そのグラフの頂点部分集合で，全ての頂点間に枝がある物である．クリークは通常部分グラフとして定義されることが多いが，ここでは頂点集合で定義していることを注意しておく．他のクリークに含まれないクリークを**極大クリーク**という．枝で結ばれていない2つの頂点を**非枝**という．グラフの密度を，全ての頂点ついに対する枝の割合，つまり $\frac{|E|}{|V|(|V|-1)/2}$ とする．頂点集合の密度を，その頂点集合が誘導するグラフの密度とする．ただし，頂点集合 U が誘導する部分グラフとは，G の枝で U の頂点同士を結ぶ物を集めてできるグラフである．**擬クリーク**とは，ある程度以上の密度を持つ頂点集合のことである．密度が δ 以上のクリークのことを δ-**擬クリーク**とよぶ．

G の頂点 v に対し，v と枝で結ばれている頂点 u は，v に**隣接する**といい，そのような u を v の**近傍**という．v の近傍の集合を $N(v)$ と表記する．v の**次数** $d(v)$ は，v に隣接する頂点の数，つまり $|N(v)|$ である．$N[v]$ は $N(v)\cup\{v\}$ のことであり，**閉近傍**という．頂点 w が u と v の両方に隣接する頂点を**共通近傍**とよぶ．頂点集合 K に対し，$d_K(v)$ を K の頂点で v に隣接する物の数とする．K の**最小次数**を $\min_{v\in K} d_K(v)$ とする．

3. 研磨グラフ

2つの頂点 v と u が擬クリーク K に含まれるとき，K の頂点の多くは v と u の両者に隣接するはずである．そのため，u と v がある程度 (k 個) 以上多くの共通隣人を持つ，ということは，u と v が共にある擬クリークに含まれることを表す条件としてモデル化することができる（厳密には，v と u が同一の連結成分に含まれていない場合に成り立たないなど必要条件となっていないが，必要条件のモデルとしては良好であると考えられるだろう）．この条件，正確には $|N[v]\cap Nu|\geq k$ であり，これを k-**共通近傍条件**とよぶ．

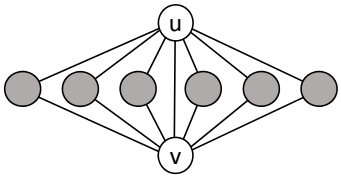

図 2 k-共通近傍条件の例: 2つの頂点は6つの共通近傍を持っており，$k<6$ であれば k-共通近傍条件を満たす．k-共通部分研磨グラフではこのような頂点の組に枝が張られている

k-共通近傍条件を満たす頂点対は，密度の濃い部分グラフ，つまり擬クリークに属していると考えられるため，その頂点対間には枝が張られているほうが，グラフのグループ構造が明確になるであろう．逆に条件を満たさないのであれば，擬クリークに含まれない可能性が高く，枝が張られていない方がグラフは明確となる．つまり，グループ構造が明確なグラフを，以下のようにモデル化することができる．

・$(u,v)\in E$ ならば，またそのときに限り $|N[v]\cap N[u]|\geq k$ が成り立つ．

このような性質を満たすグラフは，グラフクラスとして見なすことができる．そこで，この性質を満たすグラフを k-**共通部分研磨グラフ**とよぶことにする．直観的には，このグラフクラスのグラフは，ある程度の大きさ (k) 以上の複数のクリークを貼り合わせ，重なりが k 以上のクリークは併合したようなグラフになっている．実際，このグラフクラスのグラフに関しては，以下のように，極大クリークの数が少なく，ある種雑然としていないということが確認できる．

定理 1 n 頂点からなる任意の k-共通部分研磨グラフが含む極大クリークの数は高々 $O(n^k)$ である．

証明: まず最初に，大きさ k の頂点集合 S が2つの異なるクリーク K と K' に含まれる場合を考えよう．このとき，任意の K の頂点 u と K' の頂点 v に対して，$S\subseteq N[v],N[u]$ であるため，$|N[u]\cap N[v]|\geq k$ を満たす．これは，u と v の間には必ず枝があることを意味し，よって $K\cup K'$ はクリークになっている．このことから，S はただ一つの極大クリークに含まれることが分かる．反対に，大きさ k 以上の任意の極大クリークは少なくとも1

つは大きさ k の頂点集合を含むため，大きさ k 以上の極大クリークの数は ${}_nC_k$ で抑えられることが分かる．また，大きさ k 未満の極大クリークの数は高々 n^k であるので，グラフの極大クリークの数は高々 ${}_nC_k + n^k = O(n^k)$ となる．よって題意は成り立つ．■

一般のグラフでは極大クリークの数は最大で $3^{n/3}$ 個程度になることが知られている．これに比べると，$O(n^k)$ は非常に小さいと言えるであろう．この定理からただちに，いくつかの問題がこのグラフクラスにおいて，k を定数と見なした多項式時間で解けることが分かる．

系 1 n 頂点からなる k-共通部分研磨グラフの極大クリークは $O(n^{k+2.376})$ 時間で列挙できる．

証明: 牧野宇野アルゴリズム [10] は，グラフの極大クリークを 1 つあたり $O(n^{2.376})$ 時間で列挙する．これと上記の定理により題意を得る．■

系 2 n 頂点からなる k-共通部分研磨グラフでは，クリークに関する最適化問題は多項式時間で解ける．■

k が大きい，つまり $k = \Theta(n)$ である場合，k 共通部分研磨グラフのクリーク数は巨大になりうる．

定理 2 任意の 4 の倍数 n に対して，$n^2/4 + 3n/2$ 頂点の k-共通部分研磨グラフで，$2^{n/2}$ 個の極大クリークを持つ物が存在する．

証明: グラフ G を，頂点集合が $V = \{1, \ldots, n\}$ であり，頂点 i と j の間に，i が奇数かつ $j = i + 1$ であるときのみ (i, j) が枝でないようなグラフとする．G は完全グラフから完全マッチングを取り除いて得られるグラフである．このグラフの極大クリークは，各 $\{2i, 2i-1\}$ の頂点対から 1 つを選んできたものであり，その数は $2^{n/2}$ 個である．任意の頂点対 u と v は $|N[u] \cap N[v]| = n - 2$ を満たす．このグラフに頂点と枝を追加して，題意を満たすグラフ G' を作成する．

V を $V_1 = \{1, 3, 5, \ldots, n-1\}$ と $V_2 = \{2, 4, 6, \ldots, n\}$ に分割する．続いて，頂点集合 $U_0 = V_2$ と $U_i = V_1 \setminus \{2i-1\} \cup \{2i\}$，$1 \leq i \leq n/2$ を考える．各 U_j はクリークになっており，かつ G の枝は必ずどれか一つの U_j に含まれている．ここで各 U_j に新しく $n/2$ 個の頂点を加え，U_j がクリークになるように枝を加える．全ての U_j の和集合を取ったグラフを $G' = (V', E')$ とする．G' の頂点数は $|V'| = n + (n/2+1) \times n/2 = n^2/4 + 3n/2$ となる．各 U_j に対する頂点と枝の追加により，頂点対 (u, v) は，枝であれば共通近傍が増加し，枝でなければ共通近傍が増加していない．よって，G' の任意の頂点対 (u, v) に対して，$|N[u] \cap N[v]|$ は

(1) $(u, v) \in E$ であれば $n - 2 + n/2$ より大きい
(2) $u, v \in V$ かつ $(u, v) \notin E$ であれば $n - 2$
(3) $u \in U_j \setminus V$ かつ $v \in U_j$ であれば，n であり，かつ $(u, v) \in E'$
(3) $u \in U_j \setminus V$ かつ $v \notin U_j$ であれば，高々 $n/2$ であり，かつ $(u, v) \notin E'$

となる．よって，このグラフは n-共通部分研磨グラフであることがわかる．この G' は G の極大クリークを全て含む．また，G' を構築する際に V の頂点間に枝を追加しなかったことから，それらは G' の異なる極大クリークに含まれることがわかる．よって，G' は少なくとも $2^{n/2}$ 個の極大クリークを含む．■

4. マイクロクラスタリングに対するデータ研磨

通常，実データから導かれるグラフが k-共通部分研磨グラフであることは少ない．そこで，もとのグラフのグループ構造を壊さないようにグラフに枝を追加・削除し，k-共通部分研磨グラフにする，あるいは近づけるような操作を考える．この操作のことを**データ研磨**とよぶ．具体的には，与えられたグラフ $G = (V, E)$ に対して，G を以下のように作られるグラフで置き換える操作を k-**共通部分研磨**とよぶ．

・枝集合 $E' = \emptyset$ とし，各頂点対 (u, v) に対して $|N[v] \cap N[u]| \geq k$ なら u と v を枝で結び，そうでなければ u と v の間の枝を除去する

データ研磨は，与えられたグラフに対して，同じグループ構造を持つ明確なグラフを見つける作業である．つまり，ある種の最適化問題と捉えることもできるが，グループ構造が明確でない以上，陽に制約を与えることができない．局所的な必要条件に基づいて逐次的にデータの更新を行う，本稿のようなアプローチが適当であろう．

k-共通部分研磨を一回適用しただけでは，グラフは k-共通部分研磨グラフとなるとは限らないが，少なくとも以下の性質が成り立つ．

性質 1 大きさ γk の頂点集合 K は，もしその最小次数が $(\gamma + 1)k/2$ 以上であるならば，G 全体に対する k-共通部分研磨によりクリークとなる．

証明: 任意の K の 2 頂点 u と v に対し，両者に隣接しない頂点の数は最大でも

$$2 \times (\gamma k - \frac{(\gamma + 1)k}{2}) = \gamma k - k$$

である．そのため，$|N[u] \cap N[v]| \geq k$ となり，u と v の間には k-共通部分研磨により枝が張られる．全ての K の頂点間に枝が張られるため，K はクリークになる．∎

この場合，K の密度は最小でも $(\gamma+1)/2\gamma$ であることを注意しておく．

定理 3 K を大きさ γk で最小次数が $(\gamma-1)k/3$ 以上の δ-擬クリークであるとする．このとき，$\sqrt{3(1-\delta)} < (\frac{\gamma-1}{\gamma})^2$ が成り立つなら，k-共通部分研磨により K の密度は必ず増加する．

証明: M を K が含む非枝の数，M' を k-共通部分研磨後に K が含む非枝の数とする．$M = (1-\delta)\gamma k(\gamma k-1)/2$ である．まず，K の頂点 v のうち P を $d_K(v) \leq (\gamma+1)k/2$ を満たすもの，Q を $(\gamma+1)k/2 \leq d_K(v) \leq (2\gamma+1)k/3$ を満たすものの集合とする．$p = |P|$，$q = |Q|$ とする．すると，M' は $pq + p(p-1)/2$ で抑えられることが分かる．$q \leq \frac{2M-((\gamma-1)k/2)p}{(\gamma-1)k/3} = \frac{6M}{(\gamma-1)k} - 2p$ であるので，

$$\begin{aligned} M' &\leq p(\frac{6M}{(\gamma-1)k} - 2p) + \frac{p(p-1)}{2} \\ &= \frac{6pM}{(\gamma-1)k} - 2p^2 + \frac{p^2}{2} - \frac{p}{2} \\ &= \frac{-3}{2}p^2 + \frac{12M-(\gamma-1)k}{2(\gamma-1)k}p \end{aligned}$$

が成り立つ．$X = \frac{12M-(\gamma-1)k}{2(\gamma-1)k}$，つまり $p = X/3$ とおくと，M' は $-3p+X = 0$ が成り立つとき最大となり，$X^2/6$ になる．M と M' の比は

$$M'/M \leq \frac{X^2/6}{M} \leq \frac{(X^2/6}{(1-\delta)((\gamma-1)k)^2/2}$$

となる．もし $X^2 < 3(1-\delta)((\gamma-1)k)^2$，つまり $X < \sqrt{3(1-\delta)}(\gamma-1)k$ が成り立つなら，$M'/M < 1$ が成り立つ．$X < \sqrt{3(1-\delta)}(\gamma-1)k$ は

$$\frac{12M-(\gamma-1)k}{2(\gamma-1)k} < \sqrt{3(1-\delta)}(\gamma-1)k$$

と等価であり，これが成り立つ必要十分条件が

$$3(1-\delta)\gamma(\gamma k-1) - \frac{1}{2} < \sqrt{3(1-\delta)}(\gamma-1)k(\gamma-1)$$

である．この不等式が成り立つ条件は，

$$3(1-\delta)\gamma^2 k < \sqrt{3(1-\delta)}(\gamma-1)^2 k$$

であり，つまり

$$\sqrt{3(1-\delta)} < (\frac{\gamma-1}{\gamma})^2$$

である．∎

系 3 K を大きさ γk で，最小次数が $(\gamma-1)k/3$ 以上の $5/6$-擬クリークとする．もし，$\gamma > 6^{1/4}/(6^{1/4}-1) \simeq 2.7697$ が成り立つなら，k-共通部分研磨により K の密度は必ず増加する．

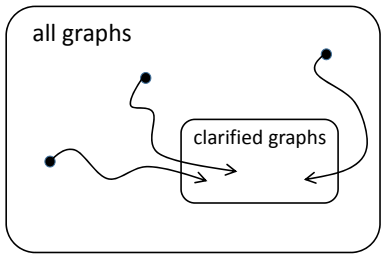

図 3 データ研磨は，与えられたグラフの構造を変えずに変更して研磨された明確なグラフを見つけるものである

証明: $\delta = 5/6$ に対し，上記の定理の言明は $1/6^{1/4} < (\gamma-1)/\gamma$ であるときに成り立つ．よって，$\gamma > 6^{1/4}/(6^{1/4}-1)$ のときに題意は成り立つ．∎

上記の定理と系から，十分大きく密度の濃いクリークは，次数分布に大きな偏りがなければ，k-共通部分研磨により必ず密度が増加することが分かる．現実的には，上記の定理の言明が成り立たずともそれなりに密であれば枝は追加される可能性が高いと考えられるため，k-共通部分研磨を適用することで，多くの擬クリークはクリークとなることが期待される．特に，適用後のグラフに変化がなくなるまで繰り返すことで，ある種の安定解・収束解を見つけるというアプローチが良いであろう．現在の所，k-共通部分研磨に関しては，なんら収束定理，つまりある程度の回数適用するとグラフに変化がなくなる，ということが証明できていないため，アルゴリズムの最悪計算時間を算定することができていない．一方で，現実問題に適用した場合は，多くとも十数回の適用後にはグラフが安定しており，このあたりの数理的な側面は，研究として非常に興味深い．

k-共通近傍条件は擬クリークが比較的独立して存在し，大きな重なりを持たないときにうまく働く．一方で，大きな次数の頂点が複数存在するとき，それらのいくつかに隣接する頂点が全て 1 つのクリークになってしまうことがあり，中規模のグループ構造を抽出することができない．このような構造はソーシャルネットワークなど多くの実グラフで見られる．これは，共通近傍のみに着目し，共通しない近傍がどの程度あるかを無視しているために起こることである．そこで，共通部分の代わりに他の類似度を用いることを考える．つまり，$N[u]$ と $N[v]$ がある程度類似しているとき 1 となり，そうでないときに 0 を取るような関数 $sim(u,v)$ を考える．そして，k-共通近傍条件の代わりに sim を使うことで，sim-**研磨グラフ**と，sim-**研磨**を定義する．sim としては，例えば Jaccard 係数を用いて $\frac{|N[u] \cap N[v]|}{|N[u] \cup N[v]|} \geq \theta$ のようにできる．これは，v と u の近傍集

合が類似度 θ 以上類似していることを示しており，両者が同一のグループに属するための必要条件として適当であろう．この類似度を用いると，ある程度以上次数の大きな頂点と次数の小さな頂点は類似しなくなる．そのため，前述の問題点を回避することができる．

以上の操作を記述した，グラフクラスタリングに対するデータ研磨アルゴリズムは以下のように記述される．

Algorithm DataPolishing_SimilarityGraph ($G = (V, E)$: graph, sim:similarity measure, τ:repeat number)
1. **for** $i := 1$ to τ
2. $\quad E' := \{(u, v) | sim(u, v) = 1\}$
3. $\quad E := E'$
4. **end for**
5. **output** G

5. Algorithm for Fast Data Polishing

本稿で紹介しているデータ研磨アルゴリズムの計算上のボトルネックは，全ての頂点対に対して $sim(u, v)$ を計算する部分であり，直接的な方法では $O(|V|^2)$ 時間かかる．これは $|V|$ が大きいときには非常な困難を伴い，$|V|$ が百万以上ともなれば，簡単に実用的な時間で計算することはできなくなる．もし，グラフ，あるいはデータ研磨後のグラフが密であるならば，この計算時間は根本的に回避できない．しかし，このようなグラフはそもそも多様性を持たず，大きな数個のグループで構成されていることが多く，データ研磨を用いる動機がない．多様性を持つ多くの実グラフデータは非常に疎であり，かつ研磨後のグラフも疎である．よって，疎な構造を利用し，枝が発生しうる枝候補部分の絞り込みを効率的に行うことができれば，計算時間を大幅に短縮することができる．

効率良い候補絞り込みのため，まず以下を観察する．

観察: 多くの集合類似尺度において，$sim(u, v) = 1$ が成り立つのは $|N[u] \cap N[v]| > 0$ である場合だけである．

この条件は，少なくとも一つは共通近傍がなければ類似しない，ということであり，自然であるといえるだろう．グラフが疎であるとき，ほぼ全ての頂点対に関して $|N[u] \cap N[v]| = 0$ が成り立つ．さらに，多くの集合類似度が，$|N[u] \cap N[v]|$ を使うことで定数時間で計算できる．これは，$|N[u] \cup N[v]| = |N[u]| + |N[v]| - |N[u] \cap N[v]|$ や $|N[u] \setminus N[v]| = |N[u]| - |N[u] \cap N[v]|$ のように，差集合や和集合の大きさが定数時間で計算できるからである．よって，ここでは共通近傍を持つ各頂点対に対して，共通近傍の数を高速に計算するアルゴリズムを紹介する．実のところこのアルゴリズムは文献には詳細がないが古くから用いられてきているものであり，自然言語解析やクラスタリングの分野では多用されている．ここでは，グラフの次数分布がべき乗則に従うとき，アルゴリズムが短時間で終了することを証明する．

性質 2 $N[u]$ が $N[v]$ と交わりを持つなら，またそのときに限り，u と v はある頂点 w の閉近傍 $N[w]$ に属している．∎

この性質より，次の性質を得る．

性質 3 $|N[u] \cap N[v]|$ は $N[u]$ の頂点 w で $v \in N[w]$ を満たす物の数である．∎

この性質から，ある u に対して，他の全ての v に対する $|N[u] \cap N[v]|$ を計算するには，全ての $w \in N[u]$ に対して $N[w]$ を走査すれば良いことがわかる．つまり，各 w に対して $N[w]$ を調べ，各 $v \in N[w]$ に対して v カウンタを 1 増やす，ということをすると，全ての w について調べ終わった後，各 v のカウンタは $|N[u] \cap N[v]|$ と等しくなる．このアイディアに基づきアルゴリズムを書くと，以下のようになる．

for each $w \in N[u]$ **do**
 for each $v \in N[w]$ s.t. $v < u$,
 $intersection[v] := intersection[v] + 1$
end for

最終的に $|N[u] \cap N[v]|$ が非ゼロの v についてのみ出力を行うためには，カウンタを増加させた v をどこかに記録しておく必要がある．そのため，リスト L を用意し，ある頂点 v のカウンタが 0 から 1 に増加されたとき，v を L に挿入するようにする．L はカウンタの再初期化を行う際にも効果的に使える．カウンタを全て 0 に戻すには，単純には $O(|V|)$ の作業が必要であるが，カウンタが非ゼロである v は全て L に入っていることを考えれば，L の要素についてのみカウンタの初期化を行えばよい．これにより，再初期化の時間は $O(|L|)$ となり，全体の計算のボトルネックとはならなくなる．これらの要素を詰め込んだアルゴリズムが，以下になる．

Algorithm NeighborIntersection ($G = (V, E)$)
1. $intersection[u] := 0$ for each $u \in V$
2. **for each** $u \in V$ **do**
3. $\quad L := \emptyset$
4. $\quad$ **for each** $w \in N[u]$ **do**
5. $\quad\quad$ **for each** $v \in N[w]$ s.t. $v < u$ **do**
6. $\quad\quad\quad$ **if** $intersection[v] = 0$ **then** insert v to L
7. $\quad\quad\quad$ $intersection[v] := intersection[v] + 1$
8. $\quad\quad$ **end for**
9. $\quad$ **end for**

10. **for each** $v \in L$ **output** $\{v, u\}$; $intersection[v] := 0$
11. **end for**

このアルゴリズムの計算時間は $O(\sum_{u\in V}\sum_{w\in N[u]}|\{v \in N[w] \mid v < u\}|)$ である．「$\sum_{u\in V}\sum_{w\in N[u]}$」の意味するところは「$u$ と u の近傍 w のペア全てについて」であるので，この式は $\sum_{w\in V}\sum_{u\in N[w]}$ と等価である．よって，計算時間は $\sum_{w\in V}\sum_{u\in N[w]}|\{v \in N[w] \mid v > u\}| = O(\sum_{w\in V}|N[w]|^2)$ と書き直せる．グラフの最大次数を Δ とおくと，この計算時間は $O(\sum_{w\in V}\Delta^2) = O(\Delta^2|V|)$ で抑えられるので，最大次数が小さい，例えば定数であれば，計算時間はグラフの大きさのほぼ線形となり，実用的に高速なアルゴリズムとなる．しかし，現実データは多くの場合，非常に大きな次数を持つ頂点を少数含んでいる．このようなグラフにおける計算時間を算定するため，グラフの次数がべき乗則に従う場合について考察する．ソーシャルネットワークや Web グラフなど，多くの実データから導かれるグラフの次数分布がべき乗則に従うことが知られている．

ここで，グラフ G の次数分布がべき乗則に従い，頂点 i の次数の期待値がある定数 α に対して α/i^k であるとしよう．このとき，定数 c と β に対して，任意の頂点 w の次数が $|N[w]| \leq c\alpha/w^k + \beta$ によって抑えられるとする．この制約を満たさない頂点がある場合でも，その数が小さければ，計算量の増加はない．そのため，この仮定は現実味を持っていると言って良いだろう．この制約から，以下が導かれる．

$$\begin{aligned}\sum_{w\in V}|N[w]|^2 &\leq \sum_{w\in V}(c\alpha/w^k+\beta)^2\\ &\leq \sum_{w\in V}3\times(c\alpha/w^k)^2+\beta^2\\ &\leq O(|E|)+\sum_{w\in V}3\times(c\alpha/w^k)^2\\ &= O(|E|+\alpha^2\sum_{w\in V}(1/w^{2k})\end{aligned}$$

よって，全体の計算時間は，$k = 1$ ならば $O(|E| + \alpha^2 \log |E|)$，$k > 1$ ならば $O(|E| + \alpha^2)$ となる．直観的には，α はグラフの最大次数であるので，それが $O(|V|^{1/2})$ であれば，全体の計算時間は $k = 1$ なら $O(|E| \log |E|)$，そうでなければ $O(|E|)$ となる．

定理 4 与えられたグラフ $G = (V, E)$ の i 番目の頂点の次数が定数 c と β を用いて $c\alpha/i^k + \beta$ で抑えられるとき，アルゴリズム NeighborIntersection は $k = 1$ のとき $O(|E| + \alpha^2 \log |E|)$ 時間で，$k > 1$ のとき $O(|E| + \alpha^2)$ 時間で終了する．■

実際には，このような条件を完全に満たすようなグラフは少ないと考えられる．しかし，少数，特に定数個の頂点の次数が大きい場合や，ある程度の頂点の次数が $c\alpha/i^k + \beta$ の定数倍で抑えられるような場合は，計算量は増加しない．このことから，実データでの計算時間は上記の理論的な上界と大きく変わることはないと考えられる．

6. まとめ

本稿では，大きな多様性を包含するデータからグループ構造を抽出する手法について提案と解説を行った．グラフクラスタリング問題に注目し，中規模なグループ構造が見えやすい明確なグラフはどのような物か考察し，その特徴付けを，頂点対の局所的な条件から行った．この条件はグラフクラスを導き，そのクラスのグラフは多項式個の極大クリークしか含まず，本質的にグループ構造が明確になっていることを証明した．また，グループ構造を壊さないようにこの条件を満たすようグラフを変更するグラフ研磨について解説し，グラフの次数分布がべき乗則に従うとき，グラフ研磨アルゴリズムの一反復がほぼ線形時間で終了することを示した．また，グラフ研磨の一反復により，ある程度の大きさを持ち，ある程度の密度を持つ部分グラフはクリークになり，また，それより少し密度が小さい部分グラフもその密度が増加することを示した．これにより，グラフ研磨が密構造を逃さないことが証明できる．

本稿で導入した，明確なグラフのクラスは，現実的な応用を直接的に持つ一方で，単純で局所的な特徴付けを持つ興味深いグラフクラスになっている．また，データ研磨の収束性や，他の類似性を用いた場合のグループ構造の保存など，非常に興味深い性質がまだ明らかになっていない．今後，このような点について深く研究していくことは，データ解析の実用と情報数理・アルゴリズムの理論の両面で非常に興味深い．

謝辞

大阪大学の鷲尾隆先生，情報学研究所所長喜連川優先生，産業総合研究所津田宏治先生，東京工業大学杉山将司先生に，研究のコメントに対する感謝の意を示したい．また，この研究は科学技術振興機構 CREST の補助を受けている．

参考文献

[1] R. Agrawal, H. Mannila, R. Srikant, H. Toivonen and A. I. Verkamo, “Fast discovery of association rules,” Advances in Knowledge Discovery and Data Mining, Chapter 12, AAAI Press / The MIT Press (1996).
[2] C. Cortes and V. N. Vapnik, “Support-Vector Networks”, Machine Learning **20** (1995).
[3] M. Girvan and M. E. J. Newman, “Community Structure in Social and Biological Networks,” Proc. Natl. Acad. Sci. USA **99**, pp. 7821–7826 (2002)
[4] P. Judea, “Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach”, Proceedings of the

Second National Conference on Artificial Intelligence, pp. 133136 (1982).
[5] C. D. Manning, P. Raghavan and H. Schütze, “Introduction to Information Retrieval”, Cambridge University Press, (2008).
[6] B. Goethals, “the FIMI repository,” `http://fimi.cs.helsinki.fi/` (2003).
[7] S. R. Kumar, P. Raghavan, S. Rajagopalan, and A. Tomkins, “Trawling the Web for Emerging Cyber-Communities,” Proceedings of the Eighth International World Wide Web Conference, Toronto, Canada, 1999.
[8] UCI Machine Learning Repository, http://archive.ics.uci.edu/ml/
[9] J. B. MacQueen, “Some Methods for Classification and Analysis of Multivariate Observations”, Berkeley Symposium on Mathematical Statistics and Probability, pp. 281-297 (1967).
[10] K. Makino and T. Uno, “New Algorithms for Enumerating All Maximal Cliques,” LNCS **3111** (Proc. SWAT 2004), pp. 260–272 (2004).
[11] G. Palla, I. Derènyi, I. Farkas, and T. Vicsek, “Uncovering the Overlapping Community Structure of Complex Networks in Nature and Society”, Nature **435**, pp. 814-818 (2005)
[12] T. Uno, T. Asai, Y. Uchida, H. Arimura, “An Efficient Algorithm for Enumerating Closed Patterns in Transaction Databases,” LNAI **3245**, pp. 16–31 (2004).
[13] T. Uno, M. Kiyomi, H. Arimura, “LCM ver. 2: Efficient Mining Algorithms for Frequent/Closed/Maximal Itemsets,” Proc. IEEE ICDM’04 Workshop FIMI’04 (2004).
[14] T. Uno, “An Efficient Algorithm for Solving Pseudo Clique Enumeration Problem” Algorithmica 56, pp. 3–16 (2010).