頂点併合によるグラフ系列クラスタリングの効率化

Efficient Graph Sequence Clustering by Merging Vertices

竹内 弦希 [1]　猪口 明博 [1] *
Genki Takeuchi[1]　Akihiro Inokuchi[1]

[1] 関西学院大学 理工学部
[1] Science and Technology, Kwansei Gakuin University

Abstract: In this paper, we tackle a problem for clustering a graph sequence to discover changing clusters. Recently, we have proposed a method for clustering graph sequences based on the spectral clustering. Although the method can detect changes of clusters e.g. division and/or mergence of clusters from graph sequences, it requires much computation time to cluster long graph sequences consisting of big graphs. In this paper, we propose a method for improving its efficiency by merging vertices in its preprocess.

1　はじめに

情報技術の発展により，膨大な量のデータを蓄積することが可能となった．しかし，日々肥大化するデータは人間の理解力を超えたため，有益な情報が含まれていてもそのままでは理解できなくなっている．そこで，そのような膨大なデータから有益な情報を発見するため，近年データマイニングに関する研究が注目され，盛んに研究されている．そのなかでもクラスタリングは，教師無し学習手法であるため，カテゴリが未知のデータを分類するのに有用である [2, 3].

本研究が対象とするグラフ系列マイニングでは，グラフ系列を複数のクラスタ系列に分割する．ここでグラフ系列とは，辺の重みが時間変化するグラフを時刻順に並べたものである．またクラスタとは，各時刻のグラフの頂点を辺の結び付きが強いもの同士がまとまるように分割したものである．さらに，クラスタ系列とは，各時刻のグラフの頂点集合を分割してできた複数のクラスタから一つを選び，時刻順に並べたものであり，クラスタの時間変化を表したものである．例えば，人間関係ネットワークにおいて，人をグラフの頂点，人と人の関係をグラフの辺で表し，その親密度に応じて辺を重み付けすると，ある時点の人間関係ネットワークを重み付きグラフにより表現することができる．さらに時刻の経過と共にその構造が変化する人間関係ネットワークは，重み付きグラフの系列として表すことが可能である．また，人間関係ネットワークでは親密度が大きい人同士が集まってクラスタが形成され，クラスタの時間変化を表すクラスタ系列は，クラスタに含まれる人の人間関係の変化を表す．人間関係ネットワークを表すグラフ系列をクラスタ系列に分割することにより，人間関係ネットワークに隠れたクラスタの変化を発見することが期待され，マーケティング戦略などへ役立てることができると考えられる．

奥井らが提案した手法 [2] は，グラフ系列を入力として受け取り，グラフ系列をクラスタリングする．この手法で最も計算時間を要するのが，その手法の内部で用いられる固有値計算であり，グラフ系列の長さを T で，各時刻のグラフの頂点数を n とすると，その時間複雑度は $O((Tn)^3)$ である．このため，グラフ系列が長くなったり，各時刻のグラフの頂点数が増えると，計算時間が大きくなる．本稿では奥井らの手法がもつ計算時間の課題を克服することを目的とする．

2　グラフ系列のクラスタリング

2.1　問題定義

時刻 t のグラフを $G^{(t)} = \left(V^{(t)}, E^{(t)}, w^{(t)}\right)$ で表す．ただし，$V^{(t)}$ は頂点の集合であり，その要素には個別の ID が振られており一意に識別可能であるとする．[1] また，$E^{(t)}$ は全ての辺の集合 $E^{(t)} = V^{(t)} \times V^{(t)}$ であり，$w^{(t)}$ は時刻 t において辺に非負の実数値を割り当てる関数である．このような T 個のグラフを時系列順に並べたものを，T ステップ重み付きグラフ系列とし，$\left\langle G^{(1)}, \ldots, G^{(T)} \right\rangle$ で表す．以下では重み付きグラフ系列を単にグラフ系列と呼ぶ．

図 1 はグラフ系列の例である．同図では辺の重みが辺の太さで表されており，太い辺ほど大きな重み付け

*連絡先：関西学院大学理工学部情報科学科
〒 669-1337 兵庫県三田市学園 2-1
E-mail: inokuchi@kwansei.ac.jp

[1] 本稿では，各頂点が ID を持つデータを対象とする．各頂点が ID を持たずに，時間の経過とともに頂点が増加するデータを対象としたクラスタリング手法 [4, 5, 6, 7, 8, 9] があるが，それらの手法では，本稿で扱う問題を解くことはできない．

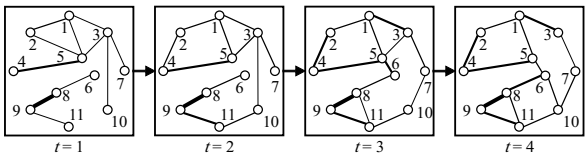

図 1: グラフ系列の例 (頂点に付随する数字は頂点 ID)

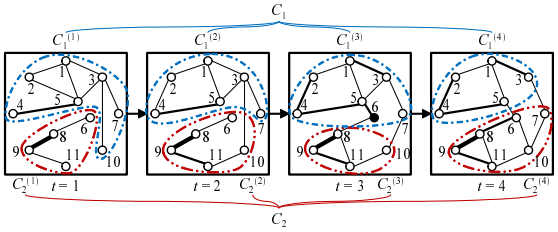

図 2: 図 1 のグラフ系列と $k = 2$ に対する，クラスタ系列 (1)

がされていることを表す．なお，簡略化のため重みが 0 の辺は図に示しておらず，以降でも重みが 0 の辺は図に描画されない．

時刻 t のグラフの頂点集合は，$\bigcup_{j=1}^{k} C_j^{(t)} = V^{(t)}$ を満たす k 個の互いに素な部分集合 $P^{(t)} = \left\{C_1^{(t)}, \cdots, C_k^{(t)}\right\}$ に分割される．これらの表記を用いて，$1 \leq j \leq k$ に対してクラスタ系列は $C_j = \left\langle C_j^{(1)}, \cdots, C_j^{(T)} \right\rangle$ と表される．ここで，ある t と j に対して $|C_j^{(t)}| = 0$ は許容するが，$\sum_{t=1}^{T} |C_j^{(t)}| = 0$ は許容しない．

本稿で扱う問題は，グラフ系列 $\left\langle G^{(1)}, \ldots, G^{(T)} \right\rangle$ と分割数 k が入力として与えられたとき，k 個のクラスタ系列 $\left\{ \left\langle C_j^{(1)}, \cdots, C_j^{(T)} \right\rangle \mid 1 \leq j \leq k \right\}$ を出力する問題である．この問題の応用例の 1 つは，1 節で述べた人間関係ネットワークの時系列から互いに結び付きの強いクラスタ（コミュニティ）を検出することである．人間関係ネットワークでは，連続する 2 時刻においてコミュニティが大きく変化することはほとんどないと考えられる．そこで，本稿が扱う問題では，以下の 2 つの要請を満たすものとする．

1. 各時刻のグラフでは，結び付きが強い頂点同士が同一のクラスタに属す．
2. $C_j^{(t)}$ と $C_j^{(t+1)}$ でクラスタに属する頂点は大きく変わらない．これをクラスタの連続性と呼ぶ．

図 2 と図 3 は図 1 からクラスタ系列を得た結果である．要請 2 を考慮しなければ，図 2 に示されるように，時刻 3 の辺 $(5, 6)$ は大きな重みをもつので，頂点 5 と 6 は同じクラスタに分割される．一方，要請 2 を考慮すると，時刻 3 の前後の分割と同様に頂点 6 は $C_2^{(3)}$ に分割され，$C_2^{(2)} = C_2^{(3)}$ となり，2 つのクラスタに属する頂点は変わらない．

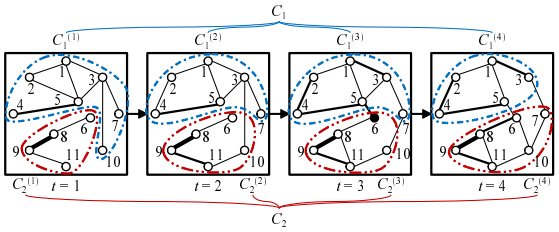

図 3: 図 1 のグラフ系列と $k = 2$ に対する，クラスタ系列 (2)

2.2 スペクトラルクラスタリング

$G = (V, E, w)$ の k 分割問題は，以下で定義される．これは RatioCut と呼ばれる関数の最少化問題である．

$$\min \sum_{j=1}^{k} \frac{1}{|C_j|} \sum_{e \in E(C_j, V \setminus C_j)} w(e)$$

ただし，$E(S, V \setminus S)$ は，一方の頂点が集合 S に含まれ，他方の頂点が集合 $V \setminus S$ に含まれる辺の集合である．また，スペクトラルクラスタリングを解説した文献 [1] では，上記の最少化問題は以下と等価であることが述べられている．

$$\min_{X \in \mathcal{R}^{|V| \times k}} \mathrm{Tr}(X^T L X) \quad s.t.\ X^T X = I^2 \qquad (1)$$

ここで，X の (i, j) 要素 x_{ij} は以下の式で与えられる．

$$x_{ij} = \begin{cases} \frac{1}{\sqrt{|C_j|}} & if\ v_i \in C_j \\ 0 & otherwise \end{cases}$$

各頂点は，k 個のクラスタのうち 1 つのクラスタに属するので，I を $|V|$ 次の単位行列とすると，$X^T X = I$ となる．また，L は G のラプラシアン行列であり，以下で定義される．行列の (i, j) 要素 a_{ij} が G の辺の重み $w(i, j)$ である G の隣接行列を A とする．また，$D = diag(\sum_{i=1}^{|V|} a_{i1}, \cdots, \sum_{i=1}^{|V|} a_{in})$ とする．このとき，$L = D - A$ である．

2.3 グラフ系列クラスタリング

本節では，文献 [3, 2] において奥井らが提案したグラフ系列クラスタリング手法を説明する．その手法では以下を最少化する $X_t \in \mathcal{R}^{|V^{(t)}| \times k}$ を求める問題を解く．

$$\sum_{t=1}^{T} \mathrm{Tr}(X_t^T L_t X_t) + \alpha \sum_{t=1}^{T-1} |X_t - X_{t+1}|^2 \qquad (2)$$

ここで，$\alpha > 0$ である．式 (2) の第 1 項を最少化することは，各時刻のグラフ $G^{(t)}$ を要請 1 に従いクラスタ

[2] 以下では $X^T X = I$ などの制約条件の表記を省略する．

$$\mathrm{Tr}\left[\begin{pmatrix}X_1\\X_2\\\vdots\\\vdots\\X_T\end{pmatrix}^{\top}\left\{\underline{\begin{pmatrix}D_1+\alpha I & 0 & \cdots & \cdots & 0\\0 & D_2+2\alpha I & \ddots & \ddots & \vdots\\\vdots & \ddots & \ddots & \ddots & 0\\\vdots & \ddots & \ddots & D_{T-1}+2\alpha I & 0\\0 & \cdots & \cdots & 0 & D_T+\alpha I\end{pmatrix}}-\underline{\underline{\begin{pmatrix}W_1 & \alpha I & 0 & \cdots & 0\\\alpha I & W_2 & \ddots & \ddots & \vdots\\0 & \ddots & \ddots & \ddots & 0\\\vdots & \ddots & \ddots & W_{T-1} & \alpha I\\0 & \cdots & 0 & \alpha I & W_T\end{pmatrix}}}\right\}\begin{pmatrix}X_1\\X_2\\\vdots\\\vdots\\X_T\end{pmatrix}\right]$$

図 4: T ステップグラフ系列のクラスタリングの目的関数

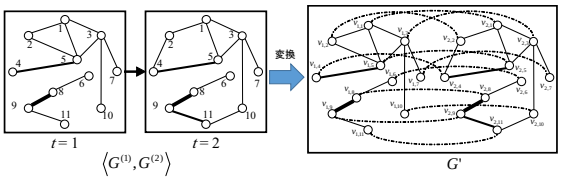

図 5: 2 ステップのグラフ系列のグラフ G' への変換

リングすることに相当する．式 (2) より図 4 に示される式を導出できる．この導出の過程は，PCM [10] と呼ばれるグラフ系列のクラスタリング手法と同様である．図 4 の下線部の行列を D'，2 重下線部の行列を W' とし，$L' = D' - W'$ とすると，L' は以下を満たすグラフ G' のラプラシアン行列である．

- G' の頂点数は $\sum_{t=1}^{T}|V^{(t)}|$ である．これ以降，時刻 t における i 番目の頂点を $v_{t,i}$ で表す．
- $G^{(t)}$ に重みが $w(i,j)$ の辺 (i,j) が存在するならば，G' に重みが $w(i,j)$ の辺 $(v_{t,i}, v_{t,j})$ が存在する．
- $1 \leq t \leq T-1, 1 \leq i \leq \max\{|V^{(t)}|, |V^{(t+1)}|\}$ に対して頂点 $v_{t,i}$ と $v_{t+1,i}$ の間にも辺が存在し，その重みは α である．

したがって，式 (2) の最少化問題は G' の k 分割問題に帰着される．頂点 $v_{t,i}$ と $v_{t+1,i}$ の間には，重みが α の辺が存在する．この辺がカットされるように，グラフ G' を k 分割すると，目的関数のとる値が大きくなるため，できるだけこの辺をカットしないように，すなわち $v_{t,i}$ と $v_{t+1,i}$ が同じクラスタに属するように分割がなされ，クラスタの連続性の検出が可能となる．以上より，要請 2 が満たされていることが分かる．図 5 に 2 ステップからなるグラフ系列 $\langle G^{(1)}, G^{(2)}\rangle$ をグラフ G' に変換した例を示す．この図において，破線で描かれた辺の重みは α である．

α を小さくすると，式 (2) の第 1 項を重視してクラスタリングされるため，要請 2 よりも要請 1 が重視され，各時刻のグラフが別々にクラスタリングされる．すなわち図 2 に示される結果となる．一方，α を大きくすると，式 (2) の第 2 項を重視してクラスタリングされるため，要請 1 よりも要請 2 が重視されて，クラスタの連続性を検出でき，図 3 に示される結果となる．

Algorithm 1: グラフ系列クラスタリング

Data: $\langle G^{(1)}, \cdots, G^{(T)}\rangle, k, \alpha$
Result: $\{\langle C_j^{(1)}, \cdots, C_j^{(T)}\rangle\}$
1 $\langle G^{(1)}, \cdots, G^{(T)}\rangle$ から G' を生成する．；
2 G' のラプラシアン行列 L' を求める．；
3 L' の固有ベクトルを固有値が小さいもの順に k 個 $\boldsymbol{\gamma}_1, \cdots, \boldsymbol{\gamma}_k$ を求める．；
4 固有ベクトル $\boldsymbol{\gamma}_q$ を q 列目に持つ行列 $\Gamma \in \mathcal{R}^{Tn\times k}$ を生成する．；
5 Γ について i 番目の行を k 次元空間上の点として扱い，Tn 個の点を k-means を用いてクラスタリングする．その結果を $P_1, \cdots, P_k$ とする．；
6 クラスタ P_j の各点 $x_{t,i}$ から $\langle C_j^{(1)}, \cdots, C_j^{(T)}\rangle$ を得る．；
7 **return** $\{\langle C_j^{(1)}, \cdots, C_j^{(T)}\rangle\}$;

Algorithm 1 に示される擬似コードの 2-5 行目で文献 [1] のスペクトラルクラスタリングのアルゴリズムを含んでいる．

しかしこの既存手法には計算時間に関する課題がある．この既存手法ではラプラシアン行列から固有ベクトルを得るための計算過程がある．大きさ m のある行列の固有値計算の時間複雑度は $O(m^3)$ であるが，既存手法では各時刻のグラフの頂点数が n，グラフ系列の長さが T のとき，G' のラプラシアン行列の大きさは Tn であるのでこの既存手法の固有値計算にかかる時間は $O((Tn)^3)$ となる．ここで Algorithm 1 をみると，$\langle G^{(1)}, \cdots, G^{(T)}\rangle$ から G' の生成は隣接するグラフで対応する頂点に α の辺を結ぶことであるので，$O(Tn)$ で解くことが出来る．また，k-means は頂点数が Tn 個なので $O((Tn)^2)$ で解くことが出来る．これより，既存手法で最も計算時間を要するのが固有値計算過程であることが分かる．そのため，グラフ系列が長くなったり，各時刻のグラフの頂点数が増えると，既存手法の固有値計算の時間は非常に大きくなる．この点から，既存手法を頂点数が数十万を超える実データに適応する際

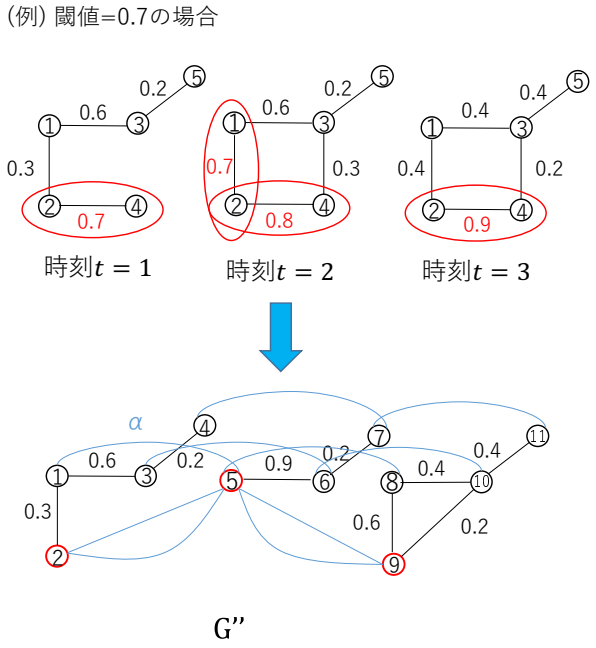

図 6: グラフ系列をグラフ G'' へ変換

に計算時間が大きな課題となる．

3 提案手法

本研究では，既存手法がもつ計算時間の課題を克服することを目的としており，スペクトラルクラスタリングでの固有値計算の前に各時刻のグラフの頂点数を減らし，ラプラシアン行列を小さくすることで計算時間を短縮させる手法を提案する．

3.1 提案手法 1

固有値計算の前に頂点数を減らすが，具体的には，図 6 に示される上側のグラフ系列のように，大きい重みの辺で結ばれた 2 頂点は同じクラスタに属す可能性が大きいので，ある閾値以上の重みで結ばれた頂点 v と u を併合し頂点 v' とし，各時刻のグラフの頂点数を減らす．そして頂点数を減らしたグラフ系列を隣接するグラフで対応する頂点を重み α の辺で結びグラフ G'' に変換し，変換したグラフ G'' に対してスペクトラルクラスタリングを行う．その後，併合された頂点をもとに戻す．ここで，頂点 v' は頂点 v と u を併合して出来た頂点であるため，v がクラスタ $C_j^{(t)}$ に属している場合，u も $C_j^{(t)}$ に属す．

提案手法 1 の擬似コードを Algorithm 2 に示す．入力で与えられたグラフ系列 $\langle G^{(1)}, \cdots, G^{(T)} \rangle$ を閾値 μ 以上の重みで結ばれた頂点を併合し，頂点数を減らす．この処理は 1-9 行目で行われ，時刻 t において併合された頂点の集合は $\Lambda_{v,t}$ で表され，rep_v はその代表となる頂点である．このようにして併合された頂点集合を $\mathcal{V}$ とし，$\mathcal{V}$ の要素に対して 1 からはじまる ID を割り振る．ここで $\mathcal{V}$ の要素は互い素である．続いて，頂点集合を $\{1, \cdots, |\mathcal{V}|\}$ とするグラフ G'' を生成する．このグラフにスペクトラルクラスタリングを適用するが，図 6 の G'' をクラスタリングした際に切られる辺の重みが G' と同じになるように，G'' の辺にも重みを付与する．その処理が 14-20 行目である．その後，21 行目において G'' にスペクトラルクラスタリングを適用して，22 行目以降で，併合した頂点を復元し，クラスタ系列の集合 $\{\langle C_j^{(1)}, \cdots, C_j^{(T)} \rangle\}$ を得る．

Algorithm 2: 提案手法

Data: $\langle G^{(1)}, \cdots, G^{(T)} \rangle$，$k$，$\alpha$，閾値 μ
Result: $\{\langle C_j^{(1)}, \cdots, C_j^{(T)} \rangle\}$

1 **for** $t \in [1, T]$ **do**
2 **for** $v \in V(G^{(t)})$ **do**
3 $\Lambda_{v,t} \leftarrow \{v\}$; $rep_v \leftarrow v$;
4 **for** $(u, v) \in E(G^{(t)})$ *s.t.* $w^{(t)}(u, v) \geq \mu$ **do**
5 $v' \leftarrow rep_v$; $u' \leftarrow rep_u$;
6 **if** $v' \neq u'$ **then**
7 $\Lambda_{v',t} \leftarrow \Lambda_{v',t} \cup \Lambda_{u',t}$; $\Lambda_{u',t} \leftarrow \emptyset$;
8 **for** s *s.t.* $rep_s = u'$ **do**
9 $rep_s \leftarrow v'$;
10 $\mathcal{V} = \{\Lambda_{v,t} \mid \Lambda_{v,t} \neq \emptyset\}$ とし，$\mathcal{V}$ の要素に対して，1 から $|\mathcal{V}|$ までの整数値を割り当てる．その割り当て関数を h とする;
11 $V'' \leftarrow \{1, \cdots, |\mathcal{V}|\}$;
12 $G'' \leftarrow (V'', V'' \times V'', w'')$;
13 **for** $t \in [1, T]$ **do**
14 **for** $\Lambda_{v,t}, \Lambda_{u,t'} \in \mathcal{V}$ *s.t.* $t = t'$ **do**
15 $n \leftarrow h(\Lambda_{v,t})$; $n' \leftarrow h(\Lambda_{u,t'})$;
16 $w''(n, n') \leftarrow \sum_{(v',u') \in \Lambda_{v,t} \times \Lambda_{u,t'}} w^{(t)}(v', u')$;
17 **for** $t \in [1, T-1]$ **do**
18 **for** $\Lambda_{v,t}, \Lambda_{u,t+1} \in \mathcal{V}$ **do**
19 $n \leftarrow h(\Lambda_{v,t})$; $n' \leftarrow h(\Lambda_{u,t+1})$;
20 $w''(n, n'') \leftarrow |\Lambda_{v,t} \cap \Lambda(u, t+1)| \times \alpha$;
21 G'' にスペクトラルクラスタリングを実行して，クラスタ $P_1, \cdots, P_k$ を得る．;
22 **for** $j \in [1, k]$ **do**
23 $P'_j \leftarrow \bigcup_{b \in P_j} h^{-1}(b)$;
24 クラスタ P'_j の各点 $x_{t,i}$ から $\langle C_j^{(1)}, \cdots, C_j^{(T)} \rangle$ を得る．;
25 **return** $\{\langle C_j^{(1)}, \cdots, C_j^{(T)} \rangle\}$;

1-9 行目の頂点の併合は，各時刻で μ 以上の重みで結ばれた頂点 v と u を探して併合し，これを μ 以上の重みで結ばれた頂点が見つからなくなるまで繰り返す．つまり，併合すべき頂点は各時刻のグラフの隣接行列を一度みることで探すことが出来るので，頂点を併合するための計算の時間複雑度は $O(Tn^2)$ である．10 行目は $\mathcal{V}$ の要素と自然数を一対一に対応付ける関数なので $O(|\mathcal{V}|)$ で $(|\mathcal{V}| \leq Tn)$ 解ける．13-16 行目は，頂点併合により $O(Tn^2)$ で解け，また，17-20 行目は，隣接するグラフの頂点集合 Λ をみれば良いので $O(Tn)$ で解ける．22-23 行目は，各クラスタに属している頂点を探すので $O(|\mathcal{V}|)$ で解け，また，24 行目は $O(Tn)$ で解ける．すべて $O((Tn)^3)$ よりも小さいオーダで解けるので，既存手法の固有値計算の時間を上回ることはない．

図 6 の時刻 $t = 1$ では $\Lambda_{1,1} = \{1\}$, $\Lambda_{2,1} = \{2, 4\}$, $\Lambda_{3,1} = \{3\}$, $\Lambda_{4,1} = \emptyset$ $\Lambda_{5,1} = \{5\}$ であり，他の時刻においても同様に求めることができる．そしてこのようにして併合された頂点集合 $\mathcal{V}$ に対して，関数 h によって図 6 の下側の G'' のように 1 からはじまる ID を割り振る．また図 6 の下側の G'' の頂点 5 と頂点 6 の重み 0.9 は $w^{(2)}(1, 3)$ と $w^{(2)}(3, 4)$ の和であり，従来法が $G^{(2)}$ の頂点と {1,2,4} と {3, 4} に分割したときに切られる辺の重みに一致する．Algorithm 2 の 17-20 行目により青い辺の 1 つの重みは α になる．G'' の頂点 2 と 5 の間には，青い辺が 2 つあることに注意されたい．つまり，辺 1 つの重みは α であるが，これらの頂点間には，重み 2α の辺があることを表している．これは，これら 2 頂点の併合前の頂点集合 $\Lambda_{2,1} = \{2, 4\}$ と $\Lambda_{1,2} = \{1, 2, 4\}$ は，共通の要素を 2 つもつからである．そしてこのようにして完成した G'' に対してスペクトラルクラスタリングを行いクラスタリングする．

しかし，この提案手法 1 には課題があり，過度に頂点を併合してしまいクラスタリングの精度が低下する可能性がある．そこで，次の提案手法 2 ではこの課題を改善する．

3.2 提案手法 2

提案手法 2 では，提案手法 1 と同様に辺の重みが閾値以上であり，かつその前後のステップでも辺の重みが閾値以上である両端の頂点を前処理で 1 つに併合する．具体的には図 7 に示される上側のグラフ系列のように，時刻 t=2，かつその前後のステップである時刻 t=1，t=3 でも頂点 2 と頂点 4 の重みが閾値以上なので，時刻 t=2 の頂点 2 と頂点 4 を 1 つの頂点に併合する．そして，同様にこのグラフ系列を図 7 の下側の 1 つのグラフ G''' に変換し，スペクトラルクラスタリングによってクラスタリングする．

この手法の擬似コードは Algorithm 2 の 4 行目の頂点併合条件を

$$\textbf{for } (u, v) \in E(G^{(t)}) \ s.t. \ w(u, v)^{(t)} \geq \mu \\ \wedge \ w(u, v)^{(t-1)} \geq \mu \ \wedge \ w(u, v)^{(t+1)} \geq \mu$$

に変更し，他の手続きは Algorithm 2 と同様である．

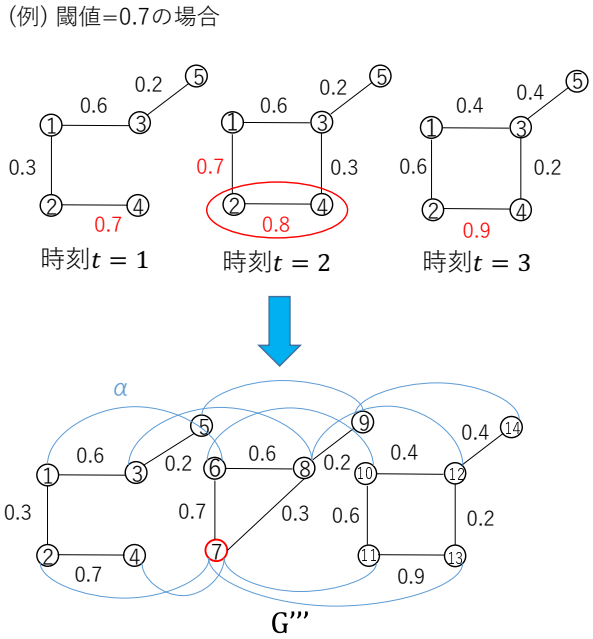

図 7: グラフ系列をグラフ G''' へ変換

この提案手法では，前後のステップでも閾値以上の辺の重みを持っている場合のみ頂点を併合するため，提案手法 1 よりもクラスタリングの精度が良いことが期待できる．また，頂点併合により G''' のラプラシアン行列に対する固有値計算時間を既存手法の場合よりも短縮できるため，既存手法よりも計算時間が速いことが期待できる．

4 評価実験

4.1 実験設定

本稿では評価のために Adjusted Rand Index (ARI) を用いる．ARI とはデータ生成時のクラスタをどの程度再現できるかを計る指標であり，比較するクラスタに共通して含まれる頂点数を用いて次のように計算される．n 個の頂点集合が互いに素な r 個の部分集合 $\mathcal{U} = \{U_1, \cdots, U_r\}$ と c 個の部分集合 $\mathcal{V} = \{V_1, \cdots, V_c\}$ に分割されているとする．ただし，$\sum_{i=1}^{r} |U_i| = \sum_{j=1}^{c} |V_j| = n$ である．次に n_{ij} を U_i と V_j に共通して含まれる頂点数 $|U_i \cap V_j|$ とすると，n_{ij} はクラスタの組み合わせによって表 1 に示す通りになる．$n_{i\cdot}$ はクラスタ U_i の頂点数，$n_{\cdot j}$ はクラスタ V_j の頂点数を表す．このとき，

表 1: 分割 $\mathcal{U}$，$\mathcal{V}$ を比較する分割表

$\mathcal{U} \setminus \mathcal{V}$	V_1	V_2	$\cdots$	V_c	合計
U_1	n_{11}	n_{12}	$\cdots$	n_{1c}	$n_{1.}$
U_2	n_{21}	n_{22}	$\cdots$	n_{2c}	$n_{2.}$
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
U_r	n_{r1}	n_{r2}	$\cdots$	n_{rc}	$n_{r.}$
合計	$n_{.1}$	$n_{.2}$	$\cdots$	$n_{.c}$	$n_{..} = n$

表 2: 人工データ生成のパラメータ

パラメータ	既定値
頂点数 n	1100 (=600+300+200)
クラスタ系列数 k	3
半径 r	3
分散 var	1.5
ステップ数 T	10
振幅 A	1.0
初期位相 φ	0
周期 ω	$\frac{\pi}{4}$
移動する頂点数 m	5
連結度 κ	50

U_i と V_j に共通して含まれる頂点数の個数は $\binom{n_{ij}}{2}$ で計算される．したがって，ARI は次のように計算され，ARI は 0 以上 1 以下の値を取る．

$$\frac{\sum_{i,j}\binom{n_{ij}}{2} - \left[\sum_i \binom{n_{i.}}{2}\sum_j \binom{n_{.j}}{2}\right]/\binom{n}{2}}{\frac{1}{2}\left[\sum_i \binom{n_{i.}}{2} + \sum_j \binom{n_{.j}}{2}\right] - \left[\sum_i \binom{n_{i.}}{2}\sum_j \binom{n_{.j}}{2}\right]/\binom{n}{2}}$$

実験結果では分割 $\mathcal{U}$ は生成時に設定した本来の分割に，$\mathcal{V}$ はクラスタリングを適用して得られた分割に対応するため，ARI が 1 に近づくほど良い結果が得られたことを表す．また，本稿で示す ARI は 20 回試行した結果の平均である．

本稿では，以下の手順に従い人工的に実験データの生成する．2 次元平面上の原点を中心とする半径 r の円周上に k 個の点を等間隔に配置し，これらの点の座標を k 個のガウス分布の平均とする．そして，各平均に対して，分散 var のガウス分布に従う点集合を生成する．この実験での 3 つの点集合内の点の数はそれぞれ $600, 300, 200$ とする．生成されたそれぞれの点集合が 1 つのクラスタに相当する．ただし，時刻を経るごとに各ガウス分布の平均が原点に近づいたり離れたりするようにして，点集合を生成する．そこで，ガウス分布の平均は初期位相 φ，振幅 A で，半径 r の円周上を中心に周期 ω で正弦的に振動させて，ガウス分布に従う点集合を生成する．すなわち，時刻 t で i 番目 $(1 \leq i \leq k)$

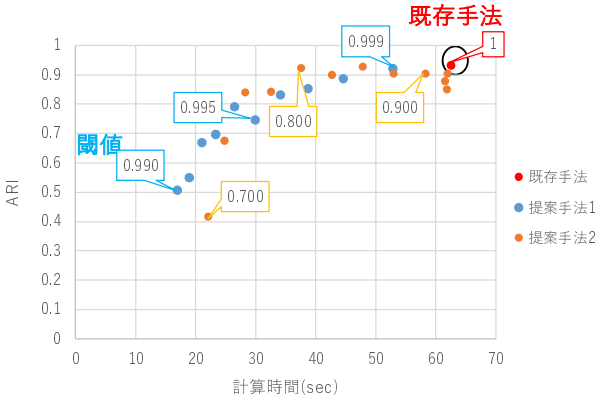

図 8: 頂点数 1100 個，ステップ数 10 のグラフ系列に対して閾値を変化させたときの計算時間と ARI の変化

のクラスタに相当するガウス分布の平均座標 (x, y) は，$R(\theta)$ を回転角 θ の回転行列として，次の通りである．

$$\begin{pmatrix} x \\ y \end{pmatrix} = R\left(\frac{2\pi(i-1)}{k}\right)\left[A\begin{pmatrix} \sin\left(\omega(t-1)+\varphi\right) \\ 0 \end{pmatrix} + \begin{pmatrix} r \\ 0 \end{pmatrix}\right]$$

このままでは，α に大きな値を設定することで高い ARI を得ることができるため，過大に評価されることになる．これを避けるために，各時刻において，最もの大きい点集合からランダムに m 個を選び，2 番目に大きい点集合に属する点となるように点を移動させる．以上で生成された各時刻の点集合をグラフの頂点集合，2 頂点間のユークリッド距離を d とするとき，辺の重みを $\exp\left(-\frac{d^2}{2}\right)$ とする κ 最近傍グラフを各時刻のグラフとした．さらに，ステップ数が T のグラフ系列を生成した．本実験では，表 2 に示されるパラメータで生成されたグラフ系列に対して，既存手法の ARI が最も高くなる $\alpha = 90$ で実験を行った．

4.2 実験結果

表 2 のパラメータを用いて作成された人工データに対して，既存手法と提案手法を適用した実験結果を図 8 から図 10 に示す．図の青の点は提案手法 1, オレンジの点は提案手法 2 の実験結果である．また，$\exp\left(-\frac{d^2}{2}\right)$ で計算される辺の重みの最大値は 1 であり，閾値が 1 より大きいときに併合される頂点はないので，閾値が 1 のときは既存手法と同じ結果となる．閾値が 1 より小さいときの計算時間，または ARI の値は提案手法 1, 提案手法 2 を用いたときのものである．各図に吹き出しに書かれてある数字は閾値 μ の値である．

図 8 は頂点数 n が 1100，ステップ数 T が 10 であるグラフ系列に対して，提案手法 1 と提案手法 2 を用い

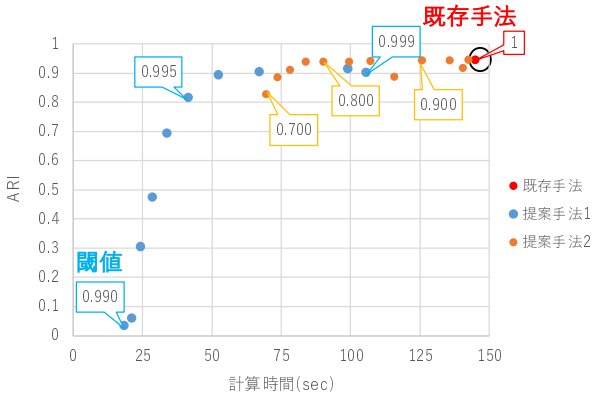

図 9: 頂点数 2200 個，ステップ数 10 のグラフ系列に対して閾値を変化させたのときの計算時間と ARI の変化

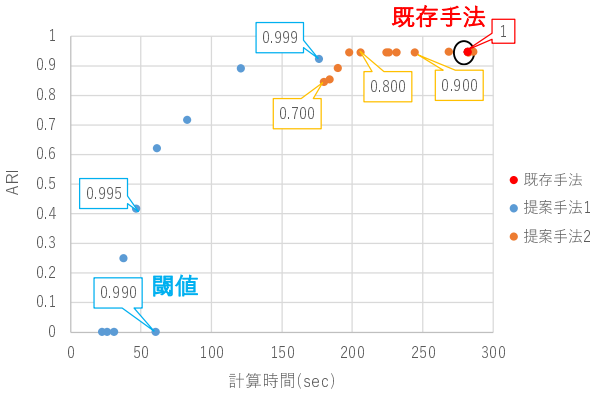

図 10: 頂点数 3300 個，ステップ数 10 のグラフ系列に対して閾値を変化させたのときの計算時間と ARI の変化

たときの計算時間と ARI である．提案手法 1 に対しては閾値 μ を 0.990 から 0.999 まで変化させ，提案手法 2 に対しては閾値を 0.975 から 0.700 まで変化させた．計算時間の変化を見ると，閾値を小さくするほど計算時間が短くなっている．既存手法と比べても，提案手法 1 と提案手法 2 を用いた場合のほうが計算時間が短い．これは，固有値計算をする前に頂点を併合することで頂点数を減らし，ラプラシアン行列の大きさを小さくしているからである．次に ARI の結果を見ると，提案手法 1 は閾値を小さくするほど ARI の値が減少している．この要因として，閾値を小さくすればするほど多くの頂点を併合することとなり，過度に頂点を併合してしまったことがあげられる． しかし，提案手法 2 は閾値を 0.800 まで下げても高い ARI(ここでは ARI が 0.900 以上の場合を指す) を保持したまま，計算時間を短縮できている．

図 9 は頂点数 n が 2200 であるグラフ系列に対して，提案手法 1 と提案手法 2 を用いて実験を行ったときの計算時間と ARI の変化を表したグラフである．その他の設定は，上記の実験と同じである．計算時間の変化を見ると，頂点数 1100 個のときと同様に，閾値を小さくするほど計算時間が短くなっている．既存手法と比べても，固有値計算をする前に頂点を併合することで頂点数を減らし，ラプラシアン行列の大きさを小さくしている提案手法 1 と提案手法 2 を用いた場合のほうが計算時間が短い．しかし，図 8 では既存手法を用いた場合の計算時間と比べて提案手法 1 を用いた場合の計算時間が約 15 秒短くなっているのに対し，頂点数 2200 個で実験した場合は約 80 秒も短くなっている．また，提案手法 2 を用いた場合の計算時間は図 8 では約 25 秒短くなっているのに対し，頂点数 2200 個で実験した場合は約 70 秒も短くなっている．これは提案手法 1 と提案手法 2 ともに ARI が 0.900 の時の計算時間を参考にしている．この要因として，頂点数が多くなると人工データ生成時のガウス分布に従う点集合の密度が大きくなるため，頂点数 1100 個での実験と同様の値で閾値を小さくすると，頂点数 2200 個での実験の場合の方が併合された頂点数が多くなることがあげられる．次に ARI の結果を見ると，提案手法 1 は頂点数 1100 個のときと同様に閾値を小さくするほど ARI の値が減少している．要因も頂点数 1100 個のときと同様に，閾値を小さくすればするほど多くの頂点を併合することとなり，過度に頂点を併合してしまったことがあげられる．しかし，この場合も提案手法 2 は頂点数 1100 個のときと同様に，閾値を 0.800 まで下げても高い ARI を保持したまま，計算時間を短縮できている．

図 10 は頂点数 n が 3300 であるグラフ系列に対して実験を行ったときの計算時間と ARI の変化を表したグラフである．計算時間の変化を見ると，これも頂点数 1100 個と 2200 個のときと同様に，閾値を小さくするほど計算時間が短くなっている．既存手法と比べても，固有値計算をする前に頂点を併合することで頂点数を減らし，ラプラシアン行列の大きさを小さくしている提案手法 1 と提案手法 2 を用いた場合のほうが計算時間が短い．さらに，既存手法を用いた場合の計算時間と比べて提案手法 1 を用いた場合の計算時間は約 160 秒短くなっており，提案手法 2 を用いた場合の計算時間は 90 秒短くなっている．この場合も提案手法 1 と提案手法 2 ともに ARI が 0.900 の時の計算時間を参考にしている．次に ARI の結果を見ると，提案手法 1 は頂点数 1100 個と頂点数 2200 個のときと同様に閾値を小さくすればするほど ARI の値が減少している．要因も頂点数 1100 個,2200 個のときと同様に，閾値を下げれば下げるほど多くの頂点を併合することとなり，過度に頂点を併合してしまったことがあげられる．しかし，この場合も提案手法 2 は頂点数 1100 個のときと同様

に，閾値を 0.800 まで下げても高い ARI を保持したまま，計算時間を短縮できている．

図 8，図 9，図 10 の結果から，提案手法 1 は閾値を小さくするほど計算時間を短縮することができるが，小さすぎると ARI の値が大幅に減少してしまう．このことから，閾値は小さければ小さいほど良いとは限らないことが考えられ，適切な閾値を設定することによって，既存手法に近い ARI の値を保ちつつ，計算時間を短縮できることが確認できた．また，提案手法 2 は閾値を 0.700 まで小さくしても高い ARI を保持しながら既存手法よりも計算時間を短縮することが確認でき，同じ計算時間でグラフ系列クラスタリングを行う場合，提案手法 2 は提案手法 1 よりも高い ARI でクラスタリングを行うことができることが確認できた．これらの結果によって，提案手法 1 と提案手法 2 は奥井らの手法に比べ，頂点数が非常に大きいと考えられる実データに適応した手法であり，提案手法 2 の方が提案手法 1 よりも実用的であることが言える．

5 まとめ

本稿では，奥井らの手法が持つ計算時間の課題を改善するグラフ系列のクラスタリング手法を提案した．また，提案手法 1 と提案手法 2 が，実際に奥井らの手法よりも計算時間を短縮することが出来るのか，比較実験を行いその性能を評価した．

その結果，提案手法 1 と提案手法 2 では閾値を小さくするほど既存手法よりも計算時間を短縮できる．これは固有値計算をする前に頂点を併合することで頂点数を減らし，ラプラシアン行列の大きさを小さくしているからである．しかし，提案手法 1 では閾値を小さくするほど過度に頂点を併合してしまい，ARI の値が減少してしまう．それに対して提案手法 2 では，閾値を 0.800 まで小さくしても，高い ARI を保持しながら既存手法よりも計算時間を短縮でき，同じ計算時間でグラフ系列クラスタリングを行う場合，提案手法 2 は提案手法 1 よりも高い ARI でクラスタリングを行うことができる．また, 提案手法 2 は閾値 μ を 0.975 から 0.800 まで変化させても高い ARI を保持したままクラスタリングができるため，提案手法 1 よりも ARI が閾値に対して感度が高くないことが分かった．以上より，閾値が 0.800 以上という現実的な値において，提案手法 2 は提案手法 1 よりも高い ARI でクラスタリングができるため，より実用性の高いクラスタリング手法と考えられる．

本稿の評価実験により，提案手法 1 と提案手法 2 は奥井らの手法と同等の精度を持っており，現実社会に存在するような人工データに対しても有効であることが明らかとなった．したがって，ソーシャルネットワークにおけるコミュニティの変化などの大規模なグラフ系列に対して，より計算時間を短縮させてクラスタリングを行うことが可能である．

参考文献

[1] U. von Luxburg. A Tutorial on Spectral Clustering. *Statistics and Computing*, Vol. 17, No. 4, pp. 395–416, 2007.

[2] S. Okui, K. Osamura, and A. Inokuchi. Detecting Smooth Cluster Changes in Evolving Graphs. *Proc. of International Conference on Machine Learning and Applications*, pp. 369–374, 2016.

[3] 長村 佳歩，奥井 颯平，猪口 明博．滑らかな変化を検出するためのグラフ系列クラスタリング．情報処理学会論文誌, Vol. 58. No. 1, pp. 278–287, 2016.

[4] C. C. Aggarwal, J. Han, J. Wang, and P. Yu. A framework for clustering evolving data streams. *Proc. of International Conference on Very Large Data Bases*, pp. 81–92, 2003.

[5] C. C. Aggarwal, J. Han, J. Wang, and P. Yu. A framework for projected clustering of high dimensional data streams. *Proc. of International Conference on Very Large Data Bases*, pp. 852–863, 2004.

[6] C. C. Aggarwal, J. Han, J. Wang, and P. Yu. On demand classification of data streams. *Proc. of ACM International Conference on Knowledge Discovery and Data Mining*, pp. 503–508, 2004.

[7] J. Beringer and E. Aullermeier. Online clustering of parallel data streams. *Data and Knowledge Engineering*, pp. 180–204, 2006.

[8] F. Cao, M. Estery, W. Qian, and A. Zhou. Density-based clustering over an evolving data stream with noise. *Proc. of SIAM Conference on Data Mining*, pp. 328–339, 2006.

[9] M. Charikar, L. O'Callaghan, and R. Panigrahy. Better streaming algorithms for clustering problems. *Proc. of ACM Symposium on Theory of Computing*, pp. 30–39, 2003.

[10] D. Chakrabarti, R. Kumar, and A. Tomkins. Evolutionary clustering. *Proc. of ACM International Conference on Knowledge Discovery and Data Mining*, pp. 554–560, 2006.