RF-001

ピボットを用いた K-medoids クラスタリング高速化による大規模機能コミュニティ抽出

Extracting Large-Scale Functional Communities by Fast K-medoids Clustering Using Pivot Selection

伏見 卓恭 † 斉藤 和巳 ‡ 池田 哲夫 ‡ 風間 一洋 §
Takayasu Fushimi Kazumi Saito Tetsuo Ikeda Kazuhiro Kazama

1. はじめに

近年，ハイパーリンクネットワークや SNS 内でのユーザ関係ネットワークだけでなく，電力網や道路網の分析に対しても複雑ネットワーク分析のアプローチが取られるようになってきている [1, 2]．こうした実ネットワークを対象にコミュニティを抽出する手法として，Clauset らによる Modularity というネットワークの分割指標を用いたコミュニティ抽出手法が高速で大規模ネットワークに対しても有効であり注目を浴びている [3]．スペクトラルグラフ分析の手法である Normalized Cut 法 [4] も有名な手法である．これらは，クラスタ内リンクを多く，クラスタ間リンクを少なくする，すなわち，ノード同士の結合が疎な部分を切断し，いくつかのノード集合に分割する方法である．一方，ネットワーク上のノード同士が密結合したような緻密な部分をコミュニティと見なして，完全部分グラフであるクリークやその条件を緩和したサブネットワークを抽出するために様々な手法が提案されている [5, 6, 7]．

著者らは，数多くある従来型のコミュニティ抽出法とは一線を画した機能コミュニティ抽出法を提案した [8]．ここで，機能コミュニティとは，社内ネットワークにおける「部長」という機能をもつノード群のように，ネットワークに対して類似の機能を有するノードからなる，連結性を仮定しないコミュニティを意味する．この手法では，ネットワーク上でのランダムウォークに着目して，各ノードの PageRank スコアが収束する過程を機能ベクトルとよぶ特徴ベクトルで表現し，そのコサイン類似度でノード間の機能類似度を定義している．そして，機能類似度に基づき K-medoids 法によりノードをクラスタリングすることで，機能的に類似するノード群を同定する．図 1 に，人工的な階層ネットワークに対する機能コミュニティ抽出結果を示す．同階層のノード，すなわち，同質の機能を有するノードを同定している．

大規模なネットワークにおいては，全ノードペアの機能類似度を計算するために膨大な時間が掛かることに加えて，メモリ上に保持することも困難になる．この問題に対して，サンプリングが最も簡単な解決策であるが，サンプリングで選ばれたノード群による解の精度は保証されない．他の解決策として，機能類似度をメモリ上に保持せず，逐次計算する方法があげられる．しかし，各ノードの機能ベクトルは高次元ベクトルであり，類似度計算のコストが大幅に増加してしまう．多種多様な次元圧縮アルゴリズム [9, 10, 11, 12, 13, 14, 15] が提案されており，利用することも考えられるが，機能類似度の差は比較的小さく，類似度計算の精度低下は計算結果に大きく影響してしまう．

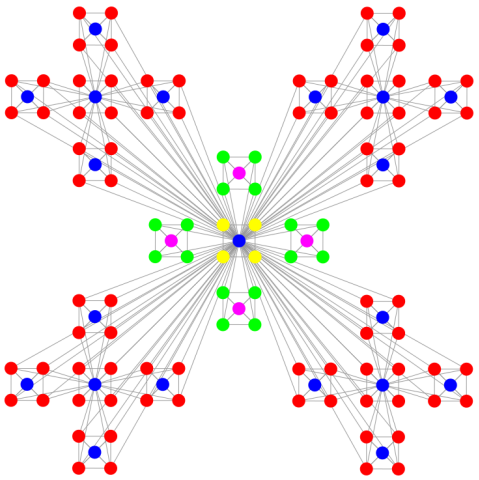


図 1: 階層ネットワークの機能コミュニティ

本稿では，機能コミュニティ抽出法における K-medoids 法の貪欲解法を高速化するアルゴリズムを提案する．具体的には，幾つかのノードをピボットとして抽出し，ピボットとの距離から距離の下界値を求めることで，ノードペアの機能類似度計算回数を削減する．本稿で対象とする機能ベクトルは高次元で表現されるため，計算回数削減の効果は大きい．さらに，類似度が非常に高いノードがある程度存在するため，下界値による枝刈りができるノードペアが多いと期待できる．

2. 関連研究

この章では，本稿の提案アルゴリズムと関連のあるピボットを用いた研究とクラスタリングの高速化に関する研究について述べる．

2.1. ピボットによる枝刈りに関する既存研究

類似検索の研究分野では，ピボットを用いて不要な距離計算を枝刈りする手法が多く提案されている．Bustos ら [16] は，選択するピボットの枝刈り率を表す目的関数を最大にするようなピボットをインクリメンタルに選択する手法を提案した．彼らの手法は，比較した幾つかの手法の中で最も距離計算回数を削減でき，次元数やオブジェクト数が多いほど大きな性能差を示して

†筑波大学図書館情報メディア系，日本学術振興会特別研究員 (PD)
‡静岡県立大学経営情報学部
§和歌山大学システム工学部

いる．

Jagadish ら [17] は，クラスタのセンターをピボット（reference point）として選択し，各オブジェクトと最近傍のピボットとの距離にスケーリング値を足した値を iDistance と呼び，全オブジェクトを 1 次元にマッピングする．この iDistance の値に対して B+木による Index を適用することで，探索中に各クラスタへのアクセス回数を削減する技術を提案している．特に現実データに多く見られる歪んだ分布のデータに適した技術であることを示した．

Kurasawa ら [18] は，Partitioning Capacity Tree(PCTree) と呼ぶ，ピボットによって分割された領域のバランスとオブジェクト枝刈り率の両方の点で最適なピボットを選択する手法を提案している．領域（Indexed Tree）のバランスを考えない既存手法と比較して，高い性能を得られることを示した．

Oscar ら [19] は，Sparse Spatial Selection (SSS) と呼ぶ，既に選定したピボットから離れて位置するオブジェクトを新たなピボットとして選択する手法を提案している．この手法は，オブジェクト集合に新たなオブジェクトが追加されるような動的なデータコレクションに対して適用でき，かつ，あらかじめピボット数を決定する必要がない点でも優れている．

Kobayashi ら [20] は，上述した手法とは異なり，ピボットをオブジェクトの中から選択するのではなく，オブジェクト空間の任意の点をピボットとして生成する手法を提案し，ピボットがオブジェクト集合に限定された既存手法と比較して，高い性能を得られることを示した．

2.2. クラスタリングの高速化に関する既存研究

本稿の高速化のベースとなる，ピボットによる距離下界値を用いた高速化アルゴリズムに関する関連研究について述べる．Elkan [21] は，距離の三角不等式を利用し，各オブジェクトの最近傍セントロイドを割り当てる時の距離計算回数削減を図ることで，K-means 法を高速化している．この手法では，K 個のセントロイドとの距離下界値をメモリ上に保持しなければならない．Hamerly [22] は，[21] をベースにして，1 つの距離下界値のみをメモリ上に保持するように拡張することで，空間計算量を悪化せず K-means 法を高速化する手法を提案している．さらに Drake ら [23] は，[21] と [22] それぞれの長所をいかし，1 から K 個の最適な数の下界値を保持する手法を提案している．

Paterlini ら [24] は，K-medoids 法における初期メドイド候補を効果的に選択することで，反復解法の収束を加速する手法を提案している．機能コミュニティ抽出法では，アルゴリズムの効率性と解の一意性，最悪ケースの解品質が保証されるなどの望ましい性質を有する貪欲法を採用しているため，文献 [24] とは問題の性質が異なる．

このほかにも，大量のデータからフラクタル次元が変わらない部分集合を抽出するいわゆるサンプリングにより対象データを絞り込む手法 [25] も提案されているが，全データでのクラスタリング結果との比較評価はなされていない．本稿では，上述したように類似検索において多大な成果を上げているピボット枝刈り技術を，機能コミュニティ抽出法における K-medoids クラスタリングに適用し高速化を図る．

3. 機能コミュニティ抽出法

機能コミュニティ抽出法は，N 個のノードの集合 V，リンク集合 E からなるネットワーク $G = (V, E)$ とコミュニティ数 K を入力とし，以下のようなアルゴリズムにより機能コミュニティを抽出する．

1. 各ステップでの PageRank スコアベクトル $\{\mathbf{y}_1, \ldots, \mathbf{y}_S\}$ を計算;
2. 各ノード u に対して，機能ベクトル $\mathbf{x}_v = (y_1(v), \ldots, y_S(v))^T$ を構築;
3. 各ノードペア u, v に対して，機能類似度 $\rho(u, v)$ を計算;
4. K-medoids 法により，全ノードを K 個の機能コミュニティ $\{V_1, \cdots, V_K\}$ に分割;

以下の節で，各ステップの詳細を説明する．

3.1. 機能ベクトルと機能類似度

機能コミュニティ抽出法 [8] において，各ノードの機能・役割はネットワーク構造に埋め込まれていると仮定している．ノードの機能として，ネットワーク内での階層的地位や相対的な位置，次数や周辺ノードの次数，周辺ノードとのつながり方などを意図しているが，これらが類似するノードどうしは，PageRank スコアの収束過程も類似すると推測できる．従って，ネットワーク構造上でのランダムウォークのモデルである PageRank [26] を用いて，各ノードの機能を表す特徴ベクトルを計算する．

以下に，ネットワーク構造から窺い知ることのできるノードの機能を表す機能ベクトル計算法を示す．機能ベクトルは，大域ジャンプなしの PageRank を用いて計算する．無向ネットワーク $G = (V, E)$ の各ノードに 1 から N までの整数値を一意に割り振る．ここで，$(u, v) \in E$ のとき $a(u, v) = 1$，それ以外のとき $a(u, v) = 0$ とし隣接行列 $\mathbf{A} \in \{0, 1\}^{N \times N}$ を定義する．各ノード $u \in V$ に対して，$\Gamma(u)$ をノード u の隣接ノード集合とする．すなわち，$\Gamma(u) = \{v \in V;\ (u, v) \in E\}$ となる．ここで，行推移確率行列 $\mathbf{P}$ の各要素を $p(u, v) = a(u, v)/|\Gamma(u)|$ とする．各ノードの PageRank スコアを要素としたベクトル $\mathbf{y}$ は，$y(v) \geq 0$ で $\sum_{v \in V} y(v) = 1$ となる．初期ベクトルを $\mathbf{y}_0 = (1/N, \ldots, 1/N)^T$ とし，繰り返しステップ数 s を用い，PageRank スコアベクトル $\mathbf{y}$ は以下の更新式の極限分布として定義される:

$$\mathbf{y}_s^T = \mathbf{y}_{s-1}^T \mathbf{P} \tag{1}$$

ここで $\mathbf{b}^T$ は $\mathbf{b}$ ベクトルの転置を表わす．式 1 は，推移確率行列 $\mathbf{P}$ の左固有ベクトルをパワー法により求めていることに等しい．また，ノード u に注目すると，

$$y_s(u) = \sum_{v\in\Gamma(u)} y_{s-1}(v)\cdot p(v,u) = \sum_{v\in\Gamma(u)} \frac{y_{s-1}(v)}{|\Gamma(v)|}$$

で計算される．パワー法の反復計算を所定の回数 S まで繰り返し，各反復回数でのノード u のスコアを要素としたベクトルを

$$\mathbf{x}_u = (y_1(u), y_2(u), \ldots, y_S(u))^T$$

と定義する．このベクトル $\mathbf{x}_u$ をノード u の機能ベクトルと呼ぶ．各ノードの収束する値は，各ノードの次数のみで決まるが，一般に収束曲線は次数のみでは決まらない．周辺ノードの影響や周辺ノードとの相対的な位置関係，ネットワーク構造の影響を受ける．機能ベクトルの次元 S が小さいと，各ノードのローカルな構造的性質が色濃く反映される．一方，S を大きくするにしたがい，ネットワークにおける相対的な位置関係など大域的な性質をベクトルとして表現できる．したがって，機能コミュニティ抽出法では，次元数の高い機能ベクトルが要求される．

以上のようにして得られた各ノードの機能ベクトルに対し，コサイン類似度によりノードペア間の機能類似度を計算する．

$$\rho(u,v) = \left\langle \frac{\mathbf{x}_u}{\|\mathbf{x}_u\|}, \frac{\mathbf{x}_v}{\|\mathbf{x}_v\|} \right\rangle$$

コサイン類似度を用いることで，大域ジャンプ無し PageRank の収束値（次数）に依存しない収束過程の類似度を求めることができる．あらかじめ L2 ノルムが 1 になるように正規化しておけば，あるノードペアの機能類似度の時間計算量は $O(S)$ である．

3.2. K-medoids クラスタリング

K-medoids 法は，非階層クラスタリングで有名な K-means 法と同様に，ノード集合 V が与えられたとき，全ノードを K 個のクラスタに分割する手法である．任意のノードペア $u, v \in V$ 間に，適切な類似度 $\rho(u,v)$ が定義されていると仮定し，他ノードとの類似度の和が高いノードを代表ノードとして選定する．代表ノード以外のノードは，自身と最も類似度の高い（距離の小さい）代表ノードのクラスタに割り当てる．K-medoids 法の近似解法には反復法や貪欲法があるが，機能コミュニティ抽出法ではアルゴリズムの効率性と解の一意性，最悪ケースの解品質が保証されるなどの望ましい性質を有する貪欲法を採用する [27]．貪欲法では，既に選定した代表ノード集合 R を固定し，以下に示す目的関数を増加させるノード w を代表ノードとして求める．

$$f(R\cup\{w\}) = \sum_{v\in V\setminus R} \max\{\mu(v\,;R), \rho(v,w)\}. \quad (2)$$

ここで，$\mu(v\,;R)$ は既に選定された代表ノードとの類似度の最大値を表し，$\mu(v\,;R) = \max_{r\in R}\{\rho(v,r)\}$ で定義される．以下に貪欲法による K-medoids 法のアルゴリズムを説明する．

1. $k \leftarrow 1$, $R_0 \leftarrow \emptyset$，各 $v \in V$ に対し，$\mu(v\,;\emptyset) \leftarrow 0$ と初期化する；
2. 式 (2) で $\hat{r}_k = \arg\max_{w\in V\setminus R_{k-1}} \{f(R_{k-1}\cup\{w\})\}$ を求め，$R_k \leftarrow R_{k-1}\cup\{\hat{r}_k\}$ とする；
3. $k = K$ ならば $R_K = \{\hat{r}_1, \ldots, \hat{r}_K\}$ を出力し終了する；
4. 各ノード $v \in V$ に対し，$\mu(v\,;R_k)$ を求め，$k \leftarrow k+1$ としステップ 2. へ戻る．

そして，各ノード u を最類似代表ノード $r_k = \arg\max_{r_k\in R_K} \rho(u, r_k)$ のクラスタ V_k に割り当てる．

4. 高速化アルゴリズム

1 章でも説明したように，全ノードペアの機能類似度がメモリ上に乗らないような大規模なネットワークを対象とする．この章では，機能コミュニティ抽出法における，K-medoids クラスタリングの貪欲解法を高速化する方法について説明する．貪欲法では，目的関数の差分値（marginal gain）が最大となるノードを 1 つずつ求め，最終的に K 個の代表ノードを出力する．

1. 第 1 代表ノード r_1 の抽出
2. ピボットノード $P = \{p_1, \ldots, p_H\}$ の選択
3. 第 2 代表ノード以降を抽出（以下を $K-1$ 回繰り返す）
   (a) 遅延評価による判定
   (b) 代表ノードによる判定
   (c) ピボットノードによる判定
   (d) 類似度計算（距離計算）
   (e) 目的関数の差分値計算
   (f) 第 k 代表ノード r_k 抽出

ここで，本稿の類似度は機能ベクトル間のコサイン類似度であり，任意のペアワイズ類似度 $\rho(u,v)$ に対して，$d(u,v) = \sqrt{1-\rho(u,v)}$ により三角不等式（公理）を満たす距離に変換することに注意されたい．

次節以降で，類似度計算回数削減のための要素技術である遅延評価，ピボットノード選択，ピボットによる枝刈りについて説明する．

4.1. 遅延評価

K-medoids クラスタリングにおける貪欲解法の目的関数は，サブモジュラ最大化問題として定式化できるため，遅延評価という技術を導入することにより高速化できる [28]．目的関数 (2) の差分値を以下のように定める：

$$g(w;R) = f(R\cup\{w\}) - f(R) = \sum_{v\in V\setminus R} \max\{\mu(v;R), \rho(v,w)\} - \mu(v;R).$$

ただし，$f(\emptyset) = 0$ とする．K-medoids 法の各ステップ $1 \leq k \leq K$ で得られる代表ノード集合 $R_1, R_2, \ldots, R_k$ に対して，目的関数のサブモジュラ性より差分値 $g(w; R_k)$ は非増加関数であるため，$\kappa < k$ に対して $g(w; R_\kappa) > g(w; R_k)$ となる．すなわち，$g(w; R_\kappa)$ は $g(w; R_k)$ の上限値となる．代表ノード候補 $w \in V$ に対する差分上限値を $\xi(w)$ で表し，差分上限値でノードを降順ソートしたリストの上位 n 番目を l_n とする．具体的には，K-medoids 法の第 k ステップにおいて，r_k の候補となるノード群 $\forall w \in V \backslash R_{k-1}$ に対して，$\xi(w) \leftarrow g(w; R_{k-2})$ とする．ただし，$g(w; R_0) \leftarrow \infty$ とする．差分上限値 $\xi(w)$ が最良差分値 g^* より小さいノード w は，$g(w; R_{k-1}) < \xi(w) = g(w; R_{k-2}) < g^*$ から差分値 $g(w; R_{k-1})$ も g^* より大きくなることはできないため，目的関数を最大にする代表ノードには成り得ない．ここで最良差分値 g^* とは，走査の過程で出現したノードの差分値の最大を表す．したがって，リストのノードを上位から順に走査し，$\xi(l_n) < g^*$ となるノード l_n より下位のノードは類似度計算を省略できる．

4.2. ピボット選択

Bustos らの文献 [16] などで比較に用いられている Outlier 法によりピボットを選択する．Outlier 法では，ノード集合の中から最も外れた機能ベクトルをもつノードを選択する．1 番目のピボットは，機能ベクトルから算出する重心ベクトルを $\bar{\mathbf{x}}$ とし，以下のように求める．

$$\hat{p}_1 = \arg\max_{u \in V} d(\mathbf{x}_u, \bar{\mathbf{x}})$$

そして，選択したノードをピボット集合に含める $P_1 \leftarrow \{p_1\}$．次いで，$h > 1$ 番目のピボットは以下のように，最近傍ピボットへの距離が最大のノードを選択する：

$$\hat{p}_h = \arg\max_{u \in V \backslash P_{h-1}} \min_{p \in P_{h-1}} d(\mathbf{x}_u, p).$$

4.3. ピボットと代表ノードによる枝刈り

この節では簡単のため，ピボット数，選定済み代表ノード数をそれぞれ 1 として説明する．ピボットを p，選定済み代表ノードを r とし，2 つ目の代表ノード候補 w に関して他のノードとの類似度 $\rho(w, v)$ を計算し，目的関数差分値を算出する必要がある．三角不等式から距離の下界値を計算することで，差分値を増加できないノードとの類似度（距離）計算を省略する．図 2 に，代表ノード候補 w に関して他のノードとの類似度を計算する必要があるか否かについて判定するイメージ図を示す．例えば，ノード w と u の距離を計算するかどうかを判定する．ノード u, w 間の距離 $d(w, u)$ のピボット p による下界値

$$LB(w, u; p) = |d(w, p) - d(u, p)| \leq d(w, u)$$

と既に選定済みの代表ノード r との距離 $d(r, u)$ について，

$$LB(w, u; p) > d(r, u)$$

であるため，$d(w, u) \geq LB(w, u; p) > d(r, u)$ が明らかであり，

$$\mu(u, \{r, w\}) = \max\{\rho(r, u), \rho(w, u)\} = \rho(r, u)$$

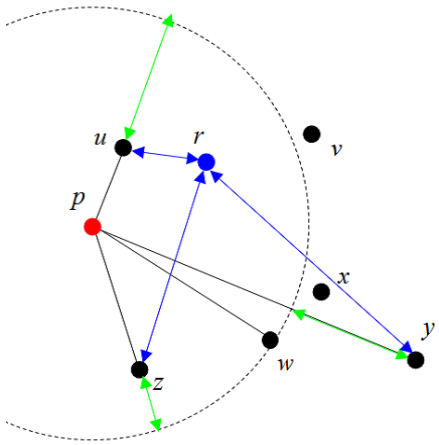


図 2: ピボットおよび代表ノードとの距離関係の例

となる．すなわち，ノード u は w が代表ノードに選ばれたとしても，既にある代表ノード r との類似度 $\rho(r, u)$ の方が高いため，目的関数増加に寄与しない．したがって，あえて距離 $d(w, u)$ を計算する必要がないと判断でき，枝刈りできる．

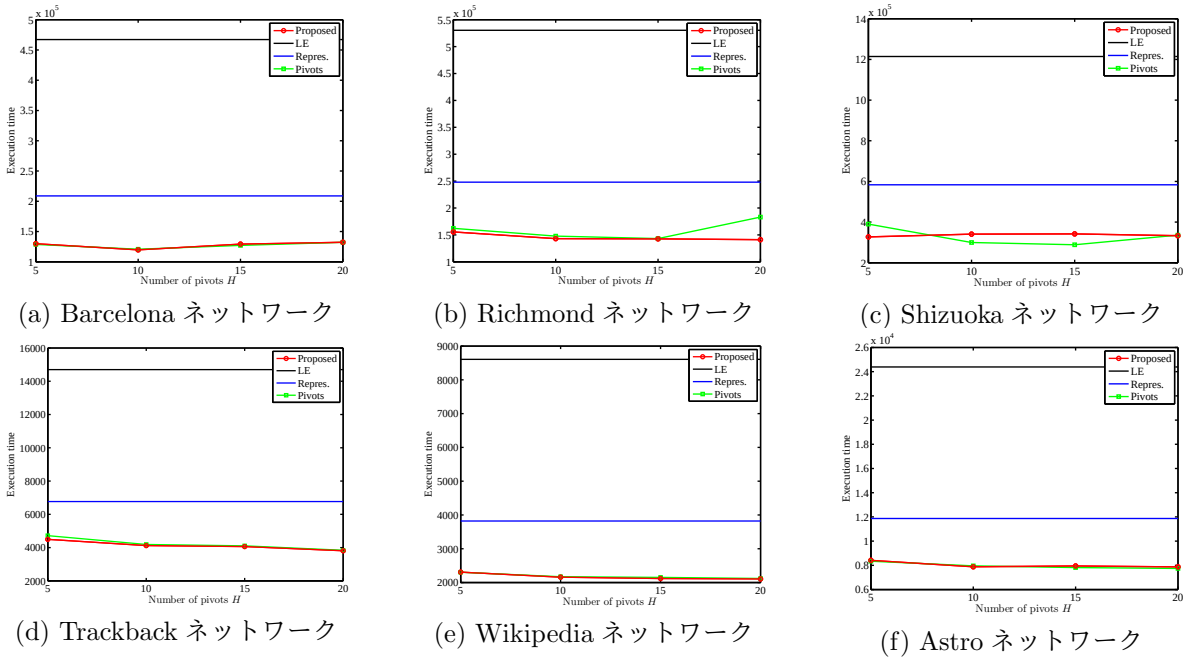
次の例として，ノード w と z の距離を計算するかどうかを判定する．ノード z, w 間の距離 $d(w, z)$ のピボット p による下界値 $LB(w, z; p) = |d(w, p) - d(z, p)| \leq d(w, z)$ と代表ノード r との距離 $d(r, z)$ について，$LB(w, z; p) < d(r, z)$ となり，$d(w, z)$ と $d(r, z)$ の大小関係が明らかでないため，枝刈りできない．したがって，$d(w, z)$ を計算しなければならない．

この節では簡単のため，ピボット数を 1 として説明したが，ピボット数が多いほど枝刈り可能なノードペアの割合は増加する．さらに本稿の提案法では，既に選定済みの代表ノードとの類似度（距離）も保持しているため，ピボットと同様に類似度計算回数削減のための枝刈りに利用する．また，遅延評価による類似度計算回数削減はノードに対して適用されるが，この節で説明したピボットおよび代表ノードに関する距離下界値を利用した枝刈りは，ノードペアに対して適用される点に注意されたい．また，提案アルゴリズムにより得られる K 個の代表ノードは，遅延評価，代表ノードおよびピボットによる類似度計算省略による高速化を導入しない単純な貪欲法の解と明らかに一致する．さらに，各ノードとピボットノードおよび選定済み代表ノードとの類似度および距離を保持するだけで済むため，空間計算量は最大で，$O(2 \times N \times (H + K))$ である．

5. 評価実験

5.1. データセット

ネットワーク構造が異なれば，各ノードの機能ベクトルの類似性も異なるため，ピボットよる効率的な枝刈りができるか否かに差が出ると考えられる．したがって，異なる種類の大規模なネットワークを用いて，提案手法の性能を評価する．道路網として，Open Street Map より以下の 2 つの都市における交差点をノード，交差点間の道路をリンクとしたネットワークを用いる．



(a) Barcelona ネットワーク　(b) Richmond ネットワーク　(c) Shizuoka ネットワーク

(d) Trackback ネットワーク　(e) Wikipedia ネットワーク　(f) Astro ネットワーク

図 3: 各ネットワークにおける計算時間の比較

表 1: 評価に用いるネットワークの統計量

ネットワーク	ノード数	リンク数	種別
Barcelona	66,790	198,774	道路網
Richmond	66,739	175,140	道路網
Shizuoka	110,925	162,322	道路網
Trackback	12,047	39,960	SNW
Wikipedia	9,481	122,522	SNW
Astro	17,903	196,972	SNW

スペイン・バルセロナ市およびその周辺を Barcelona ネットワーク，カナダ・リッチモンドおよびその周辺を Richmond ネットワーク．加えて，デジタル道路地図より静岡県全域の交差点と道路をノード，リンクとし，Shizuoka ネットワークを構築する．

ソーシャルネットワークとして次の 3 つを用いる．1 つ目はブログ記事をノード，関連するブログへのトラックバックをリンクとし無向化した Trackback ネットワークである．2 つ目はウィキペディアにおける人名記事をノード，6 回以上共起するノード間にリンクを張った Wikipedia ネットワークである．3 つ目は Arxiv Astro Physics における著者をノード，1 回以上共著関係にあるノード間にリンクを張り，最大連結成分を抽出した Astro ネットワークである [29]．以上のネットワークの基本統計量を表 1 に示す．

5.2. 比較に用いる手法

4 章で説明した提案アルゴリズムにおいて，(a) 遅延評価，(b) 代表ノード，(c) ピボットの 3 つの類似度計算回数削減のための判定をしている．サブモジュラ性を有する目的関数最大化における最先端技術である (a) 遅延評価をベースラインとし，(b) 代表ノードおよび (c) ピボットによる枝刈りが，どの程度効果的に働いているのかを評価するために，(a) のみ，(b) のみ，(c) のみにより判定する手法を比較に用いる．

遅延評価法
: 遅延評価（Lazy Evaluation）により類似度計算計算回数を削減する手法．この手法の最悪ケース（遅延評価が効かない場合）の計算量は $O(K \times N^2 \times S)$ であるが，効率的な枝刈りができた場合は，これより少なくなる．結果の図では，LE と表記し，本稿のベースラインとする．

代表ノード法
: この手法では逐次的に求まる代表ノードのみを用いて枝刈りする．代表ノード数が増えるにつれて，枝刈りできる割合が増加すると見込まれる．結果の図では，Repres. と表記する．

ピボット法
: この手法ではあらかじめ選定したピボットのみを用いて枝刈りする．選定したピボット数により，枝刈りできる割合が変化する．結果の図では，Pivots と表記する．

5.3. 計算時間と枝刈り率に関する結果

本稿では，各ノードの機能ベクトルの次元を，PageRank スコアが十分に収束する $S = 10000$ とした．クラスタ数（代表ノード数）は $K = 10$ とした．ピボットを用いる手法では，ピボット数を $5, 10, 15, 20$ と変化させて実験した．CPU は Intel Xeon 2.3GHz を使用した．

図 3 に，横軸にピボット数，縦軸に各種法によるク

ラスタリング実行時間をプロットした¶．図 3 より，いずれのネットワークに対しても，提案手法（Proposed）およびピボット法（Pivots）が圧倒的に高速であることが確認できる．具体的には，3 から 5 倍程度の速度で計算が完了している．さらに，代表ノード法（Repres.）よりも提案手法やピボット法の方が高速であることも確認できる．一般的に代表ノードは，特徴空間（本稿では，機能ベクトルの空間）において多くのノードの中心に存在するようなノード（medoid）が抽出される．一方，本稿で選択したピボットは重心から最も遠いノードや最近傍ピボットとの距離が最大のノード（outlier）を選択しているため，代表ノード法とは性質が大きく異なる．類似度計算回数を省略するためには，outlier ノードの方が適していることが知られているため，代表ノード法より提案手法やピボット法の方が高速であるという結果が得られたと考えれる．提案手法において，ピボット数の違いは大きな影響はないが，ピボット数が多いほど少ない計算時間で解を得られることも確認できた．さらに，ネットワーク構造の違いにより機能ベクトル間の類似の度合や傾向が異なると想定できるが，いずれのネットワークにおいても提案手法の有効性が確認できた．

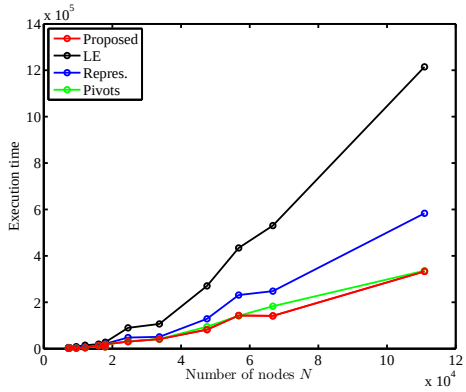


図 4: ノード数の違いと計算時間

図 4 に，横軸にノード数，縦軸に比較手法とピボット数 $H = 20$ における提案手法によるクラスタリング実行時間をプロットした．図 4 より，ノード数の増加に伴い，遅延評価法（LE）は 2 乗オーダーで実行時間が増加するが，提案手法ではそれほどの増加はみられない．さらに，ノード数が増えるにつれ，提案手法と比較に用いた手法の差は広がる傾向にある．今回の実験で用いた Shizuoka ネットワークでは，4 日以内に計算を終えている．一方，メモリ上に全ノードペアの類似度を倍精度で保持するには，98×10^9 バイト以上のメモリ空間が必要になる．

次に枝刈り率について評価する．$LE(k)$ を第 k ステップ目に遅延評価により枝刈りできたノード数とする．遅延評価による枝刈りは代表ノード候補の枝刈りであるため，類似度計算回数としては $|LE(k)| \times N$ 回省略できたことになる．$R(k)$，$P(k)$ をそれぞれ，代表ノード，

¶機能ベクトル計算時間は含まない．

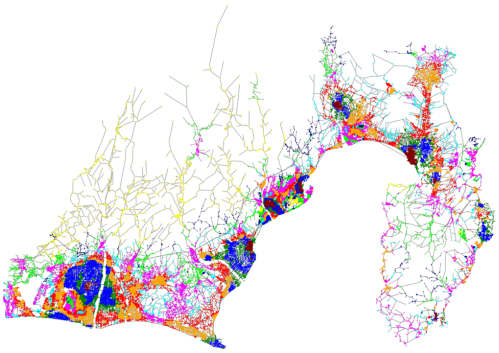


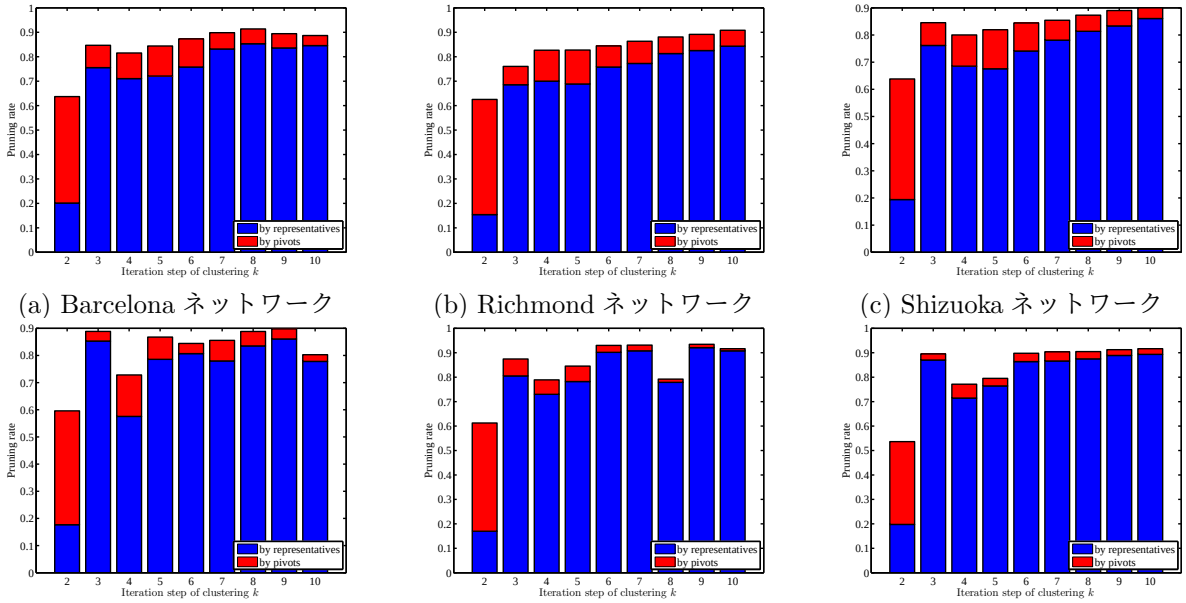
図 6: Shizuoka ネットワークの機能コミュニティ

ピボットにより枝刈りできたノードペア数とする．すなわち，遅延評価で枝刈りできなかった類似度計算回数のうちどの程度を代表ノードおよびピボットで枝刈りできたかを評価する．図 5 に，横軸に K-medoids クラスタリングのステップ数 $2 \leq k \leq K = 10$，縦軸に枝刈り率 $prune(k) = (|R(k)| + |P(k)|)/(N - |LE(k)|) \times N$ をプロットした‖．図 5 より，遅延評価により枝刈りできなかった部分の多くを枝刈りできることを確認できた．特に $k = 2$ では，選定済みの代表ノード数が 1 であるため枝刈り率が低いが，それを補うようにピボットによる枝刈り率が高くなっている．これらの図で，代表ノード（青）の方がピボット（赤）より割合が多いが，これは提案アルゴリズムにおいて代表ノードの枝刈りを先に実行し，それでも枝刈りできなかった部分をピボットにより枝刈りするからであり，代表ノードによる枝刈りの方が勝っているという意味ではない．

5.4. 機能コミュニティ抽出結果に関する考察

図 6 に，Shizuoka ネットワークに対する機能コミュニティ抽出結果を示す．ノードの色は，各機能コミュニティを表す．西（図中左）から浜松駅，焼津駅，静岡駅，清水駅，富士宮駅，沼津駅，三島駅など主要な市町の東海道線の駅周辺地域が抽出された（茶）．これらの抽出された駅周辺の繁華街は，規模は異なるものの，類似した道路構造を持つことから同一コミュニティとして抽出できたと考えられる．その他にも，学園地域・民家が多く並ぶ地域（赤，橙），山岳地帯（黄），農村地帯（桃）のように，道路構造に基づいて，交差点ノードを分類できていることがわかる．また，各機能コミュニティが，駅近隣コミュニティ（茶）から同心円状に広がっていることも見て取れる．これは，歴史的な経緯などにより，駅や城下町などの中心部からの距離に伴って地域の機能が変化し，それを反映した道路構造になっているからと考えられる．図示はしていないが，対象範囲を静岡市に限定した場合でもほぼ同様の結果が得られた．すなわち，道路ネットワークの機能コミュニティには，スケール不変性がある．これは，対象範囲を限定しても各交差点ノードの機能は大きく変化しないという直感と合致している．また，紙面の

‖$k = 1$ では枝刈りしないため，プロットしない．



(a) Barcelona ネットワーク　(b) Richmond ネットワーク　(c) Shizuoka ネットワーク

(d) Trackback ネットワーク　(e) Wikipedia ネットワーク　(f) Astro ネットワーク

図 5: 各ネットワークにおける枝刈り率の比較

都合上表示していないが，他の道路ネットワークにおいても，類似の機能を果たす領域をある程度カバーできていることを確認した．

6. おわりに

本稿では，従来のコミュニティとは異なる性質の機能コミュニティを抽出する手法において，高速にクラスタリングするためのアルゴリズムを提案した．複数のネットワークに対する評価実験から，高コストな類似度計算回数を大幅に削減でき，大規模ネットワークにも適用できることを確認した．また一般に，次元数が大きくなるにつれ Intrinsic Dimension [30] も大きくなる傾向にあり，ピボットによる枝刈りが困難になることが知られている．本稿で対象とする各ノードの機能ベクトルは，次元数は大きいが Intrinsic Dimension が小さいため，ピボットによる効率的な枝刈りができたと予測できるが，詳細な分析が必要である．

今後は，より効果的な枝刈りを実現できるピボット選択法の導入を検討する．さらに，機能ベクトルの次元数を変化させた際のクラスタリング性能および計算時間に関する検証をする予定である．

謝辞 本研究は，JSPS 特別研究員奨励費 15J00735 の助成を受けたものである．

参考文献

[1] Kalapala, V., Sanwalani, V., Clauset, A. and Moore, C.: Scale Invariance in Road Networks, *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, Vol. 73, No. 2 Pt 2, p. 6 (2006).

[2] Crucitti, P., Latora, V. and Porta, S.: Centrality Measures in Spatial Networks of Urban Streets, *Physical Review E*, Vol. 73, No. 3, pp. 036125+ (2006).

[3] Clauset, A., Newman, M. E. J. and Moore, C.: Finding community structure in very large networks, *Physical Review E*, Vol. 70, No. 6, pp. 066111+ (2004).

[4] Shi, J. and Malik, J.: Normalized Cuts and Image Segmentation, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 22, No. 8, pp. 888–905 (2000).

[5] Seidman, S. B.: Network structure and minimum degree, *Social Networks*, Vol. 5, No. 3, pp. 269 – 287 (1983).

[6] Palla, G., Derényi, I., Farkas, I. and Vicsek, T.: Uncovering the Overlapping Community Structure of Complex Networks in Nature and Society, *Nature*, Vol. 435, pp. 814–818 (2005).

[7] Saito, K., Yamada, T. and Kazama, K.: Extracting Communities from Complex Networks by the k-Dense Method, *IEICE Trans. Fundam. Electron. Commun. Comput. Sci.*, Vol. E91-A, No. 11, pp. 3304–3311 (2008).

[8] Fushimi, T., Saito, K. and Kazama, K.: Extracting Communities in Networks based on Functional Properties of Nodes, *Proceedings of the 12th Pacific Rim Knowledge Acquisition Workshop (PKAW2012)* (Richards, D. and Kang,

B. H., eds.), Berlin, Heidelberg, Springer-Verlag, pp. 328–334 (2012).

[9] Torgerson, W.: Multidimensional scaling: I. Theory and method, *Psychometrika*, Vol. 17, pp. 401–419 (1952).

[10] Jolliffe, I.: *Principal Component Analysis*, Springer Verlag (1986).

[11] Landauer, T. and Dumais, S.: A solution to Plato's problem: The Latent Semantic Analysis theory of the acquisition, induction, and representation of knowledge, *Psychological Review*, Vol. 104, pp. 211 – 240 (1997).

[12] Lee, D. D. and Seung, H. S.: Learning the parts of objects by nonnegative matrix factorization, *Nature*, Vol. 401, pp. 788–791 (1999).

[13] Tenenbaum, J. B., Silva, V. and Langford, J. C.: A Global Geometric Framework for Nonlinear Dimensionality Reduction, *Science*, Vol. 290, No. 5500, pp. 2319–2323 (2000).

[14] Blei, D. M., Ng, A. Y. and Jordan, M. I.: Latent Dirichlet Allocation, *J. Mach. Learn. Res.*, Vol. 3, pp. 993–1022 (2003).

[15] van der Maaten, L. and Hinton, G. E.: Visualizing High-Dimensional Data Using t-SNE, *Journal of Machine Learning Research*, Vol. 9, pp. 2579–2605 (2008).

[16] Bustos, B., Navarro, G. and Chavez, E.: Pivot selection techniques for proximity searching in metric spaces, *PATTERN RECOGNITION LETTERS*, Vol. 24, No. 14, pp. 2357–2366 (2003).

[17] Jagadish, H. V., Ooi, B. C., Tan, K.-L., Yu, C. and Zhang, R.: iDistance: An Adaptive B+-tree Based Indexing Method for Nearest Neighbor Search, *ACM Trans. Database Syst.*, Vol. 30, No. 2, pp. 364–397 (2005).

[18] Kurasawa, H., Fukagawa, D., Takasu, A. and Adachi, J.: Optimal Pivot Selection Method Based on the Partition and the Pruning Effect for Metric Space Indexes, *IEICE Transactions*, Vol. 94-D, No. 3, pp. 504–514 (2011).

[19] Brisaboa, N. R., Farina, A., Pedreira, O. and Reyes, N.: Spatial selection of sparse pivots for similarity search in metric spaces, *Proceedings of the 33rd conference on Current Trends in Theory and Practice of Computer Science, SOFSEM 2007*, pp. 434–445 (2007).

[20] Kobayashi, E., Fushimi, T., Saito, K. and Ikeda, T.: Similarity Search by Generating Pivots Based on Manhattan Distance, *Proceedings of the 13th Pacific Rim International Conference on Artificial Intelligence(PRICAI2014)*, Springer International Publishing, pp. 435–446 (2014).

[21] Elkan, C.: Using the Triangle Inequality to Accelerate k-Means., *Machine Learning, Proceedings of the 12th International Conference (ICML 2003)* (Fawcett, T. and Mishra, N., eds.), AAAI Press, pp. 147–153 (2003).

[22] Hamerly, G.: Making k-means Even Faster, *SIAM International Conference on Data Mining*, pp. 130–140 (2010).

[23] Drake, J. and Hamerly, G.: Accelerated k-means with adaptive distance bounds, *Proceedings of the 5th NIPS Workshop on Optimization for Machine Learning* (2012).

[24] Paterlini, A. A., Nascimento, M. A. and Jr., C. T.: Using Pivots to Speed-Up k-Medoids Clustering., *Journal of Information and Data Management*, Vol. 2, No. 2, pp. 221–236 (2011).

[25] Jiang, C., Li, Y., Shao, M. and Jia, P.: Accelerating Clustering Methods through Fractal Based Analysis, *the 1st Workshop on application of self-similarity and fractals in data mining(KDD2002 Workshop)* (2002).

[26] Langville, A. N. and Meyer, C. D.: Deeper Inside PageRank, *Internet Mathematics*, Vol. 1, No. 3, pp. 335–380 (2004).

[27] Nemhauser, G. L., Wolsey, L. A. and Fisher, M. L.: An Analysis of Approximations for Maximizing Submodular Set Functions, *Mathematical Programming*, Vol. 14, pp. 265–294 (1978).

[28] Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J. and Glance, N.: Cost-effective Outbreak Detection in Networks, *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, New York, NY, USA, ACM, pp. 420–429 (2007).

[29] Leskovec, J., Kleinberg, J. and Faloutsos, C.: Graph Evolution: Densification and Shrinking Diameters, *ACM Trans. Knowl. Discov. Data*, Vol. 1, No. 1 (2007).

[30] Levina, E. and Bickel, P. J.: Maximum Likelihood Estimation of Intrinsic Dimension., *NIPS*, pp. 777–784 (2004).