大規模グラフに対するノードの枝刈りを用いたRankClusの高速化

山崎耕太郎† 塩川 浩昭†† 北川 博之††

† 筑波大学大学院システム情報工学研究科 〒305-8573 つくば市天王台 1-1-1
†† 筑波大学計算科学研究センター 〒305–8573 つくば市天王台 1–1–1
E-mail: †k_yamazaki@kde.cs.tsukuba.ac.jp, ††{shiokawa, kitagawa}@cs.tsukuba.ac.jp

あらまし グラフ分析手法のひとつである RankClus はランキングとクラスタリングを統合し，相互に補完することで従来の手法より正確で効率的な分析を可能にした手法である．しかし，RankClus はグラフに含まれるクラスタの数だけ部分グラフを作成し，全ての部分グラフに対してランキングとクラスタリングを実行する必要があり，大規模グラフにおいて計算時間が膨大となってしまうという問題がある．そこで本稿では大規模グラフに対する RankClus を高速化するアルゴリズムを提案する．提案手法では，重要度が著しく低いノードを逐次的に計算対象から枝刈りすることで RankClus の高速化を図る．本稿では提案手法の概要と性能評価の結果について述べる．

キーワード ランキング，クラスタリング

1 はじめに

情報ネットワークは，ソーシャルメディアやモバイル機器の普及に伴い日々大量のデータが生成されており，データから新たに情報を獲得するデータ分析の需要が増加している．特に，データ間の関係性を表現するグラフというデータ構造はノード（頂点）群とノード間の連結関係を示すエッジ（枝）群で構成されるデータ構造である．

代表的なグラフ分析手法の一つに，ランキングとクラスタリングがある．ランキングはランキング関数に基づきグラフの各ノードの重要性を評価する手法である．代表的な手法として PageRank [3]，HITS [4] 及び ObjectRank [6] が挙げられる．一方，クラスタリングはある近接尺度に基づき，似ているノードを同じグループに，似ていないノードを異なるグループに分ける手法である．代表的なクラスタリング手法として modularity ベース [2, 7]，密度ベース [8, 11] のアルゴリズムが挙げられる．

そして，ランキングとクラスタリングを統合し，精度の高いグラフ分析を実現するフレームワークとして RankClus [10] がある．RankClus はランキングとクラスタリングを統合し，相互に補完することで従来の手法より正確で効率的な分析を可能にした手法である．実際に図 1 に示す学会と著者の関係性を表現する Bi-Type Information Network と呼ばれるネットワークに RankClus を適用した例を考える．この時，学会と著者間では「学会に著者による論文が出版された」，または「著者が学会に論文を出版した」関係がある場合にエッジを張り，著者間は共著関係があった場合にエッジを張るものとする．ここでは図 1 に示した学会ノード集合 X が X={SIGMOD, VLDB, PODS, ICDT, EDBT, SIGIR, TREC, ECIR, JCDL, ECDL} の 10 ノードから構成されるとする．ただし，SIGMOD, VLDB, PODS, ICDT, EDBT はデータベースに関連した国際学会，SIGIR, TREC, ECIR, JCDL, ECDL は情報検索に関連した国際学会である．このグラフに対して，ノード集合 X をクラスタリング対象して RankClus を適用した場合，RankClus はノード集合 X 内のクラスタと各クラスタに所属するノード（学会）のランキング結果を求めることができる．具体的な RankClus の出力を表 1 に示す．表 1 からわかるように，RankClus は研究分野に対応したクラスタをグラフから自動抽出するだけでなく，各分野内における重要性の高さを反映したランキング結果を同時に求めることができる．

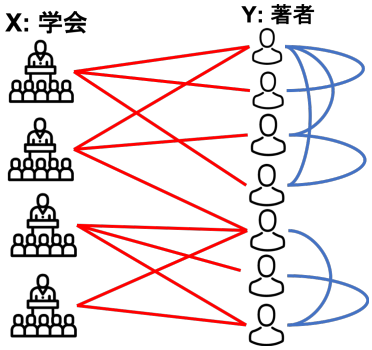

図 1 学会と著者の関係性を表現するグラフ．

しかし，RankClus はグラフに含まれるクラスタの数だけ部分グラフを作成し，全ての部分グラフに対してランキングとクラスタリングを実行する必要があり，大規模グラフにおいて計算時間が膨大となってしまうという問題がある．そこで本稿では大規模グラフに対する RankClus の高速化手法を提案する．提案手法では，計算対象となるノード数を削減することで RankClus の高速化を図る．具体的には，処理の過程において計算されるランク値の変化に着目し，ランク値の変化率が著しく低いノードを検出し逐次的に計算対象から枝刈りすることでノードの数を削減する．本稿では，人工データを用いた提案手法の有効性を検証し，ノード間のリンク接続密度が低いグラフに対して，提案手法が効果的であることを確認した．

表 1　RankClus を用いたランキング結果.

クラスタ 1

Rank	学会
1	SIGMOD
2	VLDB
3	EDBT
4	PODS
5	ICDT

クラスタ 2

Rank	学会
1	SIGIR
2	TREC
3	ECIR
4	JCDL
5	ECDL

表 2　記号の定義.

記号	定義
$\mathbb{G}$	グラフ.
$\mathbb{V}$	$\mathbb{G}$ のノード集合.
$\mathbb{E}$	$\mathbb{G}$ のエッジ集合.
$\mathbb{G}_i$	$\mathbb{X}_i$ による部分グラフ.
$\mathbb{V}_i$	$\mathbb{G}_i$ のノード集合.
$\mathbb{E}_i$	$\mathbb{G}_i$ のエッジ集合.
$\mathbb{P}_k$	$\mathbb{G}_k$ の枝刈り可能なノード集合.
X_k	クラスタ k 内の X の集合.
X	target type のノード集合.
Y	attribute type のノード集合.
m	X のノード数.
n	Y のノード数.
K	クラスタ数.
W	$(m+n)\times(m+n)$ 遷移行列.
$\vec{r}_X$	X の $m\times 1$ ランク値ベクトル.
$\vec{r}_Y$	Y の $n\times 1$ ランク値ベクトル.
$\pi_{i,k}$	x_i がクラスタ k に属する事後確率.
$D(i,k)$	x_i と k の距離.
$R^{(t)}(j,k)$	変化率.
λ_R	変化率パラメータ.
λ_I	安定性パラメータ.

2　前 提 知 識

本節では，ノード集合に対するランキングとクラスタリングを同時に実現するアルゴリズムである RankClus の前提知識について述べる．表 2 に本稿が用いる記号の定義を示す.

2.1　データモデル

2.1.1　Bi-Type Information Network:

RankClus は，入力として Bi-Type Information Network と呼ばれるデータモデルを対象とする．定義は以下の通りである．2 つのノード集合 $X=\{x_1, x_2, \cdots, x_m\}, Y=\{y_1, y_2, \cdots, y_n\}$ が与えられた時，対象グラフ $\mathbb{G}\langle\mathbb{V},\mathbb{E}\rangle$ を *Bi-Type Information Network* と呼ぶ．$\mathbb{V}$ と $\mathbb{E}$ はそれぞれグラフのノード集合とエッジ集合とし，$\mathbb{V}(\mathbb{G})=X\cup Y$，$\mathbb{E}(\mathbb{G})=\{\langle o_i, o_j\rangle\}$ (ただし $o_i, o_j\in X\cup Y$) で表される．各ノード間のリンクの重みを $W_{(m+n)\times(m+n)}=\{w_{o_i,\,o_j}\}$ となるリンクの隣接行列として表す．便宜上リンク行列を W_{XX}，W_{XY}，W_{YX}，W_{YY} の 4 つのブロックに分解する．各ブロックは添字のノード集合間のノードの部分ネットワークを示す．W は以下のように表現することができる.

$$W=\begin{pmatrix} W_{XX} & W_{XY} \\ W_{YX} & W_{YY} \end{pmatrix} \tag{1}$$

例として学会-著者の関係性を表す Bi-Type Information Network を図 1 に示す．このグラフは学会と著者の関係性を表すネットワークである．著者 y_i が学会 x_i で論文を出版した場合，学会 x_i と著者 y_i の間にエッジをつなぐ．同様に著者 y_i と著者 y_j に共著関係がある場合にエッジをつなぐ.

2.1.2　target type と attribute type:

RankClus では X，Y の中から target type と attribute type を定める．RankClus は target type であるノード集合に対してのみクラスタリングを行い，attribute type として指定されたノード集合はランキングとクラスタリング処理の補助情報として利用する.

2.2　RankClus アルゴリズム

RankClus はノード集合に対するランキングとクラスタリングを同時に実現する手法である．具体的に RankClus は以下の手順で処理を行う.

(1)　target type のノード集合を K 個のクラスタにランダムに分割し，K 個の部分グラフを構築する.

(2)　各部分グラフに対して X と Y のノードにランキング処理を行う.

(3)　(2) で求めたランク値を基に，各 target type のノードに対して K 次元の特徴ベクトルを計算する.

(4)　(3) で求めた各ノードの K 次元ベクトルを使い各 target type のノードのクラスタを更新する.

(5)　(2) から (4) の処理をクラスタが収束するまで繰り返す.

RankClus の処理は (2) のランキング処理部，(3) と (4) のクラスタリング処理部に分割できる．ランキング処理部とクラスタリング処理部の詳細をそれぞれ 2.2.1 節と 2.2.2 節に述べる.

2.2.1　ランキング処理部

RankClus でははじめに target type のノード集合 X を K クラスタに分割し，K 個の部分グラフを構築する．X_i を i 番目のクラスタ ($1\leqq i\leqq K$) に含まれる X の部分ノード集合とすると，部分グラフは $\mathbb{G}_i=\langle\mathbb{V}_i,\mathbb{E}_i\rangle$ と定義される．ただし $\mathbb{V}_i=\{X_i\cup Y\}$，$\mathbb{E}_i=\{\langle o_i,o_j\rangle | o_i,o_j\in X_i\cup Y\rangle\}$ である．部分グラフ構築後，RankClus は各部分グラフ $\mathbb{G}_i$ の各ノードに対して Authority に基づくランキング関数を計算する．$\vec{r}_X(x)$ と $\vec{r}_Y(y)$ をそれぞれノード $x\in X$ とノード $y\in Y$ のランク値とする．この時，各ランク値は次の定義に基づき計算される.

[定義 1]（Authority Ranking）　$\vec{r}_X(i,k)$ と $\vec{r}_Y(i,k)$ を各部分グラフ $\mathbb{G}_k$ に含まれる $x_i\in X_k$ の $y_i\in Y$ のランク値とする．この時 $\vec{r}_X(i,k)$ と $\vec{r}_Y(i,k)$ を以下の式で定義する.

$$\vec{r}_X(i,k)=\alpha\sum_{j=1}^{m}W_{XY}(i,j)\vec{r}_Y(j,k),$$

$$\vec{r}_Y(i,k)=\alpha\sum_{j=1}^{m}W_{XY}(i,j)\vec{r}_X(j,k)+(1-\alpha)\sum_{j=1}^{n}W_{YY}(i,j)\vec{r}_Y(j,k),$$

ただしパラメータ $\alpha\in[0,1]$ は任意に設定する．$\langle o_i,o_j\rangle\notin\mathbb{G}_i$ の時 $W_{XY}(i,j)=0$ となり，そうでない場合は $W_{XY}(i,j)=w_{o_i,o_j}$ となる.

定義 1 より $\vec{r_X}$,$\vec{r_Y}$ はお互いの値を入力として持つ再帰構造と

なっており，収束した $\vec{r_X}$,$\vec{r_Y}$ を獲得するまで，$\vec{r_X}$,$\vec{r_Y}$ を反復計算する必要がある．したがって，時間計算量は $O(t|\mathbb{E}|)$ で t は反復回数，ただし $|\mathbb{E}|$ はリンクの総数である．

2.2.2 クラスタリング処理部

クラスタリング処理部ではランキング処理部で求めたランク値 $\vec{r}_X(i,k)$ を基にクラスタの更新を行う．初めに target type ノード x_i がクラスタ X_k に属する事後確率 $\pi_{i,k}$ を求める．この時，各 $x_i \in X$ に対する K 次元ベクトル $s_{x_i} = \{\pi_{i,1}, \pi_{i,2}, \ldots, \pi_{i,K}\}$ を計算し，クラスタの中心を求める計算に用いる．

事後確率 $\pi_{i,k}$ の計算: RankClus は *mixture generative model* [5] に基づき事後確率 $\pi_{i.k}$ を計算する．$\pi_{i.k}$ はノード x_i がクラスタ X_k $(k = 1, \ldots, K)$ に属する確率である．$\pi_{i,k} = p(k|x_i) \propto p(x_i|k)p(k)$ より，RankClus は $p(x_i|k)$ と $p(k)$ を計算する必要がある．RankClus ははじめに $p(x_i|k)$ をランキング処理部で計算した $\mathbb{G}_k$ 内の x_i の条件付きランク値とみなす．すなわち，RankClus は $p(x_i|k) = \vec{r_X}(i,k)$ を得る．この時，エッジ $\langle x, y\rangle$ $(x \in X$ と $y \in Y)$ が生成する確率 $p(k)$ を EM アルゴリズム [1] を用いて計算する．結果的に RankClus は x_i とクラスタ k の組み合わせに対するの事後確率を以下の式で得る．

[定義 2]（事後確率 $\pi_{i,k}$） $\pi_{i,k}$ を x_i がクラスタ X_k に属する事後確率とする．この時 $\pi_{i,k}$ は以下の式で定義される:

$$\pi_{i,k} = p(k|x_i) = \frac{\vec{r}_X(i,k)p(k)}{\sum_{l=1}^{K} \vec{r}_X(i,l)p(l)},$$

ただし，$p(k)$ は EM アルゴリズムによって計算された確率である．

クラスタの更新: 事後確率 $\pi_{i,k}$ の計算後, RankClus はクラスタの更新をする．はじめに，それぞれの x_i に対して K 次元ベクトル $s_{x_i} = \{\pi_{i,1}, \pi_{i,2}, \ldots, \pi_{i,K}\}$ を計算する．この時，以下の定義に基づきクラスタの中心となるベクトルを計算する．

[定義 3]（クラスタの中心） $\vec{s}_{X_k}$ をクラスタ X_k の中心ベクトルを表現するベクトルとする時，$\vec{s}_{X_i}$ は以下の式で定義される．

$$\vec{s}_{Xk} = \frac{\sum_{x \in X_k} \vec{s}(x)}{|X_k|}.$$

次に各 target type ノード $x_i \in X$ について以下の指標を用いクラスタの中心との距離を計算する．

[定義 4]（距離） $D(i,k)$ をノード x_i とクラスタ X_k の距離とする，この時 $D(i,k)$ は以下の式で定義される．

$$D(i,k) = 1 - \frac{\sum_{l=1}^{K} \vec{s}_{x_i}(l)\vec{s}_{X_k}(l)}{\sqrt{\sum_{l=1}^{K} (\vec{s}_{x_i}(l))^2}\sqrt{\sum_{l=1}^{K} (\vec{s}_{X_k}(l))^2}}.$$

最終的に，RankClus は $x \in X$ を最も $D(i,k)$ が大きいクラスタに対して割当てクラスタを更新する．

3 提案手法

本稿では RankClus の高速化手法を提案する．提案手法の基本的なアイデアはランキング処理部の計算量を減らすことである．定義 1 より RankClus は $\vec{r}_X(i,k)$ と $\vec{r}_Y(i,k)$ を収束するまで反復計算する必要がある．したがって，ランク値の計算対象である $X \cup Y$ のノードの数を減らすことが RankClus の高速化に大きく寄与すると考えられる．

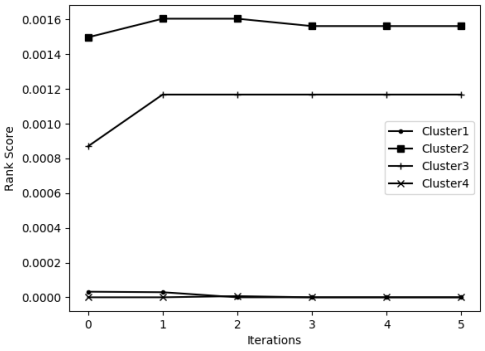

図 2 各反復におけるランク値の変化

3.1 予備実験

本研究では，ランキング処理部において削減可能なノードを特定するために，次の予備実験を行った．予備実験では DBLP データから Bi-Type Information Network を考えた．このデータセットは 20 の学会，5,639 の著者を持つ学会と著者のネットワークである．target type として学会を指定した．クラスタ数は $K = 4$ と設定し各反復ごとの著者のランク値の変化について調査した．結果を図 2 に示す．図 2 よりランク値は早い段階で収束し変化が小さく，各反復において適切なクラスタに対しては増加していき，そうでないクラスタに対して減少していくことがわかった．以上の性質より我々は各反復処理の中でランク値が減少していくノードはクラスタリング結果に与える影響が小さいと考え，そのようなノードを特定するための新たな指標を提案した．

3.2 ランク値の変化率

[定義 5]（変化率） t 回目の部分グラフ $\mathbb{G}_k$ 内のノード x_j のランク値の変化率を $R^{(t)}(j,k)$ とする時，変化率 $R^{(t)}(j,k)$ は以下の式で定義される．

$$R^{(t)}(j,k) = \frac{r^{(t)}(j,k)}{r^{(t-1)}(j,k)},$$

[定義 6]（枝刈り可能ノード） $\mathbb{P}_k$ を $\mathbb{G}_k$ から枝刈り可能なノード，$\bar{t}$ をランキング処理時の反復回数とする．この時 $\mathbb{P}_k$ は以下の式で定義される．

$$\mathbb{P}_k = \{u \in \mathbb{V}_k | \Theta(u,k) \geqq \lambda_I\},$$

ただし，$\Theta(u,k) = |\{R^{(t)}(u,k) | R^{(t)}(u,k) < \lambda_R \text{ for } t = 1, \ldots, \bar{t}\}|$ である．この時 $\Theta(u,k)$ は $\bar{t}$ 回目の反復までで変化率 $R^{(t)}(j,k)$ が λ_R 以下を満たす数である．

3.3 アルゴリズム

本手法の擬似コードをアルゴリズム 1 に示す．ランキング計算後にノードの変化率を求め，クラスタリング処理後に枝刈り可能なノード $\mathbb{P}_k$ を枝刈する．

Algorithm 1 提案手法

Input: $\mathbb{G} = \langle \mathbb{V}, \mathbb{E} \rangle$, クラスタ数 K, λ_R,λ_I
Output: クラスタ $X_i (i = 1, 2, \cdots, K)$, ランク値 $r\vec{X}$,$r\vec{Y}$

// Step 0: 初期化
$t = 0$.
初期 K クラスタ $X_1^{(t)}, X_2^{(t)}, \ldots, X_K^{(t)}$ の生成.
$X_i^{(t)}$ と Y からサブグラフ $\mathbb{G}_i^{(t)}$ の構築.
repeat
// Step 1: 各クラスタに対するランキング
for $i = 1$ to K **do**
for each $x_j \in X_i$ and $y_s \in Y$ **do**
定義 1 よりランク値 $\vec{r}_X(j,i)$ と $\vec{r}_Y(s,i)$ の計算.
5 より変化率 $R^{(t)}(s,k)$ の計算.
end for
end for
// Step 2: クラスタの中心の計算
for $i = 1$ to K **do**
for $x_j \in X_i$ **do**
定義 2 より $\pi_{j,i}$ の計算.
end for
定義 3 よりクラスタの中心 $\vec{s}_{X_i}$ の計算.
end for
// Step 3: クラスタの割当
for each $x_j \in X$ **do**
for $i = 1$ to K **do**
定義 4 より距離 $D(j,k)$ の計算.
end for
$k_0 = \arg\min_k D(j,k)$.
$X_{k_0}^{(t+1)} = X_{k_0}^{(t+1)} \cup \{x_j\}$.
end for
// Step 4: 枝刈り
for $i = 1$ to K **do**
$X_i^{(t+1)}$ と Y からサブグラフ $\mathbb{G}_i^{(t+1)}$ の構築.
定義 6 から $\mathbb{P}_i$ を導出.
$\mathbb{V}_i^{(t+1)} = \mathbb{V}_i^{(t+1)} \backslash \mathbb{P}_i$.
end for
$t = t + 1$.
until クラスタが収束.

4 評価実験

4.1 実験環境

提案手法と RankClus を C++11(gcc 4.8.5) で実装し，Intel(R) Xeon(R)E5-1620 3.50 GHz CPU，128 GB RAM を搭載した CentOS サーバで実験を行った．パラメータ α は 0.95 とし，枝刈りにおけるパラメータは $\lambda_R = 0.8$，$\lambda_I = 5$ とする．

4.2 データセット

本実験では RankClus [10] で用いられた 5 種類の人工 Bi-Type Information Network を利用した．各データセットは 45 個の target type ノード，2,000 個の attribute type ノードを持ち，クラスタ数 K=3 とする．3 つのクラスタ X_1, X_2 , 及び X_3 はそれぞれ 10，20，及び 15 個のノードを持つ．上記の設定に基づきエッジ密度を表現するパラメータ P とクラスタ間の分離度を表現するパラメータ T を変化させ 5 つのデータセットを作成する．P が大きいほどクラスタ内の接続が強く，T は大きいほどクラスタの独立性が高まる．Y を attribute type ノードとした時，$P = [P_1, P_2, P_3]$ はそれぞれ $X_1 \cup Y$, $X_2 \cup Y$，及び $X_3 \cup Y$ 内のエッジの数とする．$T = [T_{11}, T_{12}, T_{13}; T_{21}, T_{22}, T_{23}; T_{31}, T_{32}, T_{33}]$ は 2 つのクラスタ間のエッジの数を示し，T_{ij} はクラスタ i から j へのリンクの確率である．詳細の設定を以下に示す．

- Dataset1: 中程度に分離していて中密度なデータセット
  $P = [1000, 1500, 2000]$,
  $T = [0.8, 0.05, 0.15; 0.1, 0.8, 0.1; 0.1, 0.05, 0.85]$
- Dataset2: 中程度に分離していて低密度なデータセット
  $P = [800, 1300, 1200]$,
  $T = [0.8, 0.05, 0.15; 0.1, 0.8, 0.1; 0.1, 0.05, 0.85]$
- Dataset3: 中程度に分離していて高密度なデータセット
  $P = [2000, 3000, 4000]$,
  $T = [0.8, 0.05, 0.15; 0.1, 0.8, 0.1; 0.1, 0.05, 0.85]$
- Dataset4: 大きく分離していて中密度なデータセット
  $P = [1000, 1500, 2000]$,
  $T = [0.9, 0.05, 0.05; 0.05, 0.9, 0.05; 0.1, 0.05, 0.85]$
- Dataset5: 大きく分離していて低密度なデータセット
  $P = [800, 1300, 1200]$,
  $T = [0.9, 0.05, 0.05; 0.05, 0.9, 0.05; 0.1, 0.05, 0.85]$

4.3 クラスタリングの評価指標

クラスタリングの正確性を示すため，NMI [9] を用いる．NMI は結果を 0 から 1 の間の値を取り，1 に近いほどクラスタリング精度は高く，0 に近いほどクラスタリング精度が低いことを示す．

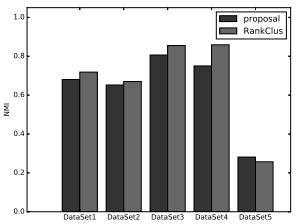

図 3 NMI

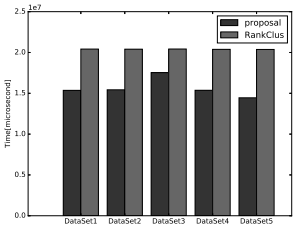

図 4 実行時間

4.4 効率性

4.2 節の 5 つのデータセットを用い提案手法と RankClus アルゴリズムの実行時間を比較した．図 4 に各人工データセットに対する実行時間の比較結果を示す．実験結果より，提案手法は RankClus より約 30%高速であることが確認できた．

4.5 正確性

提案手法のクラスタリング精度について確認した．精度を測る指標として，我々は 4.3 節に述べた NMI を用いた．評価では，各データセットに対する正解クラスタリング結果と提案手法と RankClus のクラスタリング結果を比較した．図 3 は各データセットに対する NMI の結果を示す．図 3 より提案手法は，RankClus と比較して高い近似精度であることが確認できる．

表 3　DBLP データに対するランキングとクラスタリング結果.

ランク	クラスタ 1	クラスタ 2	クラスタ 3	クラスタ 4
1	SIGMOD	KDD	SIGIR	NIPS
2	VLDB	ICDM	TREC	ICML
3	EDBT	SDM	ECIR	UAI
4	PODS	PAKDD	JCDL	COLT
5	ICDT	PKDD	ECDL	
6		ECML		

4.6　ケーススタディ

提案手法の実データ適用に対する影響を調査するためこの事例研究では、DBLP 書誌データベースから抽出された著者-学会型 Bi-Type Information Network を構築した．DBLP から 20 の代表的な国際学会の出版リストを収集し，20 の学会で少なくとも 2 つの論文を発表した 5,639 人の著者を抽出し，図 1 の例と同様に関係性を定義した．表 3 は提案手法による 20 の国際学会のランキングとクラスタリングの結果を示 す．表より提案手法は近似アルゴリズムであるにも関わらず適切な処理を行えることが確認できた．特に，クラスタ 1, 2, 3 および 4 はそれぞれデータベース，データマイニング，情報検索，そして機械学習の分野に分けられていることが分かる．興味深いことに，提案手法では ECML(European Conference on Machine Learning) という「機械学習」を名称に含んだ国際会議をクラスタ 4 (機械学習クラスタ) ではなくクラスタ 2 (データマイニング) に振り分けた．この理由は明らかで，ECML は 2008 年以降 PKDD と共に開催されていて，ECML と PKDD の主要な著者が重複しているからである．したがって提案手法では ECML をデータマイニングクラスタのメンバとみなした．このように，提案手法では大規模グラフにおける新たな研究コミュニティの発見をすることができた．

5　まとめと今後の課題

本稿では大規模グラフを対象としたときの RankClus の高速化手法を提案した．提案手法では，ノードのランク値の変化率が継続的に低いノードを特定し枝刈りすることによってノード数を削減する．提案手法は RankClus と比較して，高速かつ高精度で評価実験を行えることを確認した．今後の課題ではさらなる高速化，高精度化を目指す．予備実験では，クラスタの動向に着目し完全に収束しているクラスタの計算を省略しても情報の損失なく高速化することがわかった．今後，より多くのノードの枝刈りを実現するためクラスタの変化とランク値の調査を進め高速化を検討する予定である．

謝　　辞

本研究の一部は JST ACT-I ならびに JSPS 科研費 JP18K18057 による支援を受けたものである．

文　　献

[1] Jeff Bilmes. A gentle Tutorial of the EM Algorithm and its Application to Parameter Estimation for Gaussian Mixture and Hidden Markov Models. Technical report, 1998.

[2] V.D. Blondel, J.L. Guillaume, R. Lambiotte, and E.L.J.S. Mech. Fast Unfolding of Communities in Large Networks. *Journal of Statistical Mechanics: Theory and Experiment*, Vol. 2008, No. 10, p. P10008, 2008.

[3] Sergey Brin and Lawrence Page. The Anatomy of a Large-scale Hypertextual Web Search Engine. *Comput. Netw. ISDN Syst.*, pp. 107–117, 1998.

[4] Jon M. Kleinberg. Authoritative Sources in a Hyperlinked Environment. *J. ACM*, pp. 604–632, 1999.

[5] Andrew Y. Ng and Michael I. Jordan. On Discriminative vs. Generative Classifiers: A Comparison of Logistic Regression and Naive Bayes. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, pp. 841–848. MIT Press, 2002.

[6] Tomoki Sato, Hiroaki Shiokawa, Yuto Yamaguchi, and Hiroyuki Kitagawa. FORank: Fast ObjectRank for Large Heterogeneous Graphs. In *Companion Proceedings of the The Web Conference 2018*, pp. 103–104, 2018.

[7] Hiroaki Shiokawa, Yasuhiro Fujiwara, and Makoto Onizuka. Fast Algorithm for Modularity-based Graph Clustering. In *Proceedings of the 27th AAAI Conference on Artificial Intelligence*, pp. 1170–1176, 2013.

[8] Hiroaki Shiokawa, Yasuhiro Fujiwara, and Makoto Onizuka. SCAN++: Efficient Algorithm for Finding Clusters, Hubs and Outliers on Large-scale Graphs. *Proceedings of Very Large Data Bases Endowment*, Vol. 8, No. 11, pp. 1178–1189, 2015.

[9] Alexander Strehl and Joydeep Ghosh. Cluster Ensembles — a Knowledge Reuse Framework for Combining Multiple Partitions. *J. Mach. Learn. Res.*, pp. 583–617, 2003.

[10] Yizhou Sun, Jiawei Han, Peixiang Zhao, Zhijun Yin, Hong Cheng, and Tianyi Wu. RankClus: Integrating Clustering with Ranking for Heterogeneous Information Network Analysis. EDBT '09, pp. 565–576, 2009.

[11] Tomokatsu Takahashi, Hiroaki Shiokawa, and Hiroyuki Kitagawa. SCAN-XP: Parallel Structural Graph Clustering Algorithm on Intel Xeon Phi Coprocessors. In *Proceedings of the 2nd International Workshop on Network Data Analytics*, NDA, pp. 6:1–6:7, New York, NY, USA, 2017.