DBSJ Japanese Journal, Vol.13-J, No.1
October 2014

部分グラフに基づく効率的な PageRank 推定

An Efficient Method for PageRank Estimation based on Subgraphs

坂倉 悠太[1♡] 山口 祐人[♢]
天笠 俊之[♠] 北川 博之[♣]

Yuta SAKAKURA Yuto YAMAGUCHI
Toshiyuki AMAGASA Hiroyuki KITAGAWA

PageRank はグラフ上のノードの重要度を評価するリンク構造解析手法であり，様々な分野で用いられている．しかし，**PageRank** には計算コストが大きいという問題がある．一方で，一部のノードの重要度だけが必要とされる場合が多い．そのため，グラフ全体の情報を用いずに特定のノードの **PageRank** 値を推定する手法がいくつか提案されている．具体的には，**Chen** らの手法は特定のノードを含む部分グラフを作成し，その **PageRank** 値を推定する．しかし，部分グラフを作成するために反復計算を何度も繰り返すため，その計算コストはまだ大きい．本稿では，反復計算を行わずに再帰的な計算を用いることで，効率的に部分グラフを作成する手法を提案する．評価実験により，提案手法は推定精度を維持しながら効率的に **PageRank** 値を推定できることを示した．

In this paper, we propose an improved method for estimating PageRank scores efficiently. PageRank is known to be computationally expensive particularly when dealing with large graphs. However, it is often the case that we require the scores of a small subset of nodes. To tackle this issue, some researchers have proposed methods to estimate the scores of specific nodes without accessing the whole graph. Specifically, the method by Chen et al. constructs a subgraph containing the target node to estimate the target's score. However, its computational cost is still expensive because it requires repeated calculations to construct a subgraph. Meanwhile, our proposed method requires solving linear systems without conducting costly iterative processes, leading to efficient PageRank estimation. Experimental results show that our method can significantly reduce the computational cost while maintaining the estimation accuracy at the same level of the method by Chen et al.

[1]2014 年 4 月より日本テラインターナショナル株式会社．
♡ 非会員　筑波大学大学院システム情報工学研究科
yuuta@kde.cs.tsukuba.ac.jp
♢ 正会員　筑波大学大学院システム情報工学研究科
yuto_ymgc@kde.cs.tsukuba.ac.jp
♠ 正会員　筑波大学システム情報系
amagasa@cs.tsukuba.ac.jp
♣ 正会員　筑波大学システム情報系
kitagawa@cs.tsukuba.ac.jp

1 はじめに

PageRank [13] はリンク構造解析手法の一つであり，グラフ上のノードの重要度を評価する手法である．PageRank は，Web 検索 [6] やソーシャルネットワーク分析 [15]，バイオインフォマティクス [12] などの様々な分野に用いられている．しかし，PageRank は様々な分野に応用されている反面，計算コストが大きいという問題がある．そのため，巨大なグラフに対して PageRank を計算すると，計算コストが膨大になってしまう．この問題に対して，PageRank の計算の高速化についての研究がされている [11, 2, 10]．

さらに，PageRank の計算にはグラフ全体の情報が必要であるが，この点が問題になることがある．すなわち，グラフ全体の情報を取得することが困難な場合がある．例えば，検索エンジンが Web ページをクロールする場合，クロールされたデータは複数のデータセンターに分割して保存される．それゆえ，膨大なデータ量とデータの転送コストのため，データ全体についての分析を行うことは困難である．

また，ユーザは全てのノードではなく一部のノードの重要度だけを必要とすることが多い．例えば，特定のノードの重要度の時間的な変化を調べたい場合や，特定のコミュニティ内における重要なノードを調べたい場合などが挙げられる．このような場合，注目しているノード（群）以外の重要度は必要ではない．そのため，必要なノードの重要度だけを高速に計算できると理想的である．

この問題に対して，グラフ全体の情報を用いずに特定のノード（対象ノード）の PageRank 値を計算する手法 [8, 4, 5] が提案されている．Chen ら [8] は初めてこの問題に取り組んだ．Chen らは，ユーザがノード v の ID を与えると，ノード v の子ノードと親ノードを取得できる fetch 操作によってノードの情報を取得する環境を想定した．fetch 操作の回数が小さいほど，取得するノードの情報が少ないため，fetch 操作の回数を最小にすることを目指した．Chen らの手法では，対象ノードを含む部分グラフを作成し，その部分グラフを用いて対象ノードの PageRank 値を推定する．Chen らは，あるノードが与える対象ノードの PageRank 値への影響（影響力）を考慮しながら部分グラフを作成した．部分グラフの作成後，部分グラフ内で PageRank を計算することで，対象ノードの PageRank 値を推定する．しかし，部分グラフの作成する過程において，各ノードの影響力を計算するために何度も反復計算を行うため，Chen らの手法はまだ時間がかかる．そのため，部分グラフサイズは PageRank 値の推定精度に影響を与える一方で，部分グラフの作成時間にも影響を与える．

本稿では，対象ノードの PageRank 値をより効率的に推定する手法を提案する．具体的には，Chen らの手法を参考にしながら，より効率的に部分グラフを作成する．あるノードの影響力はそのノードがエッジを張っているノードの影響力を用いて再帰的に計算することができる．この特徴を用いることで，反復計算を行わずに部分グラフを作成することができる．これにより，提案手法は PageRank 値の推定精度を維持したまま，効率的に対象ノードの PageRank 値を推定することができる．

提案手法の性能を評価するために Twitter[2] と Web グラフのデータセットを用いて評価実験を行った．実験結果より，提案手法は推定精度を維持しながら，高速に対象ノードの PageRank 値を推定できることが分かった．具体的には，Chen らの手法と比較して，提案手法は精度を維持したまま，最大で約 3.2 倍高速に

[2]http://twitter.com/

PageRank 値を推定することができた．

本稿の構成は以下の通りである．2 節で関連研究について説明し，3 節で Chen らの手法について説明する．4 節で影響力を定義し，5 節で提案手法について説明する．6 節で評価実験を行い，7 節でまとめを述べる．

2 関連研究

Chen ら [8] は，部分グラフを用いて対象ノードの PageRank 値を推定する手法を提案した．Chen らは，対象ノードの PageRank 値の推定に必要なノードを部分グラフに追加していくことで，部分グラフを作成した．作成した部分グラフ内で PageRank を計算することで，対象ノードの PageRank 値を推定した．詳細は 3 節で説明する．本研究とは，部分グラフの作成方法が異なる．

Bar-Yossef ら [4] は，部分グラフを作成せずに対象ノードの PageRank 値を推定する手法を提案した．Bar-Yossef らは，対象ノードからエッジを逆方向にたどりながら，そのノードが対象ノードに与える PageRank 値を計算した．計算した PageRank 値を加算していくことで，対象ノードの PageRank 値を推定した．しかし，Bar-Yossef らは推定精度に関する評価を行っていなかった．また，Bressan [5] らは，推定した PageRank 値に基づいて k 個のノードをランキングするという問題について調査した．Bressan らは Bar-Yossef ら [4] の手法を改善した手法を用いて評価実験を行った．しかし，Bressan らは他の関連手法との比較を行わなかった．本研究は部分グラフを用いて PageRank 値を推定している点が，これらの研究とは異なる．

また，特定の部分グラフ（対象部分グラフ）内のノードの PageRank 値に注目した研究も行われている [9, 16, 17]．Davis ら [9] は，対象部分グラフにエッジを張っているノードを追加することで，対象部分グラフを拡大した．拡大した対象部分グラフ内で PageRank を計算することで，対象部分グラフ内のノードの PageRank 値を推定した．Wu ら [16] は，対象部分グラフに含まれていないノード群を一つのノードに凝集した．対象部分グラフと凝集されたノードから構成されるグラフを用いて，PageRank 値を推定した．Xue ら [17] は，対象部分グラフとユーザのアクセスパターンを用いて新しい部分グラフを作成し，その部分グラフを用いて PageRank 値を推定した．これらの研究は部分グラフについて注目しているが，本研究は特定のノードに注目している点が異なる．

グラフ中の全てのノードの PageRank 値を推定する手法も提案されている [7]．Broder ら [7] は，グラフ凝集を用いた PageRank 値の推定手法を提案した．Broder らは，ホストが同じ Web ページを一つのノードに凝集することで，ホストグラフと呼ばれる，グラフ全体の概略のグラフを作成した．そして，ホストグラフを用いて，全てのノードの PageRank 値を推定した．

3 Chen らの PageRank 推定手法

3.1 概要

Chen らの手法 [8] は，特定のノードの PageRank 値の推定を目的とした手法である．Chen らの手法では，PageRank 値を推定したいノード（対象ノード）と対象ノードの周囲のノード，これらのノードと繋がっているエッジから構成される部分グラフ（局所グラフ）を構成する．その後，構成した局所グラフを用いて対象ノードの PageRank 値を推定する．

Chen らは，*fetch* 操作によってノードの情報を取得する環境を想定した．fetch 操作では，ノード v の ID が与えられると，以下の情報を取得することができる．

- ノード v から出ているエッジとその接続先のノード ID
- ノード v に入っているエッジとその接続元のノード ID

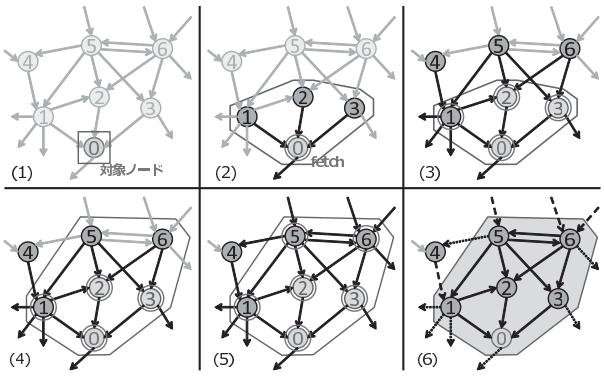

図 1: Chen らの手法を用いた局所グラフ作成の例

また，ノード v と繋がっているエッジを取得することができるため，ノード v の入次数と出次数を計算することもできる．Chen らは，fetch 操作の回数を計算コストの指標として用いた．

Chen らの手法は，“局所グラフの作成” と “PageRank 値の推定” の二つの処理に分かれている．それぞれの処理について，3.2 節と 3.3 節で説明する．

3.2 局所グラフの作成

まず，対象ノードからエッジを逆方向にたどっていくことで部分グラフを拡大し，局所グラフを作成する．初期部分グラフは対象ノードだけを含む部分グラフである．まず，対象ノードに対して fetch 操作を行い，対象ノードの情報を取得する．そして，対象ノードに入っているエッジの接続元のノードを部分グラフに追加する．この操作をノードの展開と呼ぶ．次に，新しく部分グラフに追加されたノードに対して fetch 操作を行い，それらのノードが展開の基準を満たしているかを判定する．そして，展開の基準を満たしているノードだけを展開する．この時，各ノードにおいて展開の判定は一度しか行われない．そのため，一度展開をしないと判定されたノードは，以降の部分グラフを拡大する過程において，再び展開の判定をされることはない．展開の基準を満たすノードが無くなるまで，再帰的に追加されたノードを展開していくことで部分グラフを拡大し，局所グラフを作成する．

局所グラフ作成の例を図 1 に示す．この例では，対象ノード 0 から部分グラフを拡大することで局所グラフを作成する．初期の部分グラフは対象ノード 0 だけを含む（図 1 の (1)）．まず，対象ノード 0 に fetch 操作を行い，これを展開することで，ノード 1, 2, 3 が部分グラフに追加される．(図 1 の (2))．次に，ノード 1, 2, 3 に対して fetch 操作を行い，展開の基準を満たしているかどうかを判定する．ここでは，ノード 2, 3 だけが展開の基準を満たしたとする（図 1 の (3)）．ノード 2, 3 を展開することで，ノード 5, 6 が新しく部分グラフに追加される（図 1 の (4)）．そして，ノード 5, 6 が展開の基準を満たしているかどうかを判定する．ここでは，ノード 5, 6 の両方が展開の基準を満たさなかったとする (図 1 の (5))．展開の基準を満たすノードが無くなったため，部分グラフ部分グラフの拡大を終了する．結果として，図 1 の (6) の局所グラフが作成される．

局所グラフは三種類のノードとエッジを含んでいる．一つ目のエッジは入力エッジと呼ばれる局所グラフ外から局所グラフ内へ入ってくるエッジ，二つ目のエッジは出力エッジと呼ばれる局所グラフ内から局所グラフ外へ出ていくエッジ，三つ目のエッジは内部エッジと呼ばれる局所グラフ内から局所グラフ内へのエッジである．また，入力エッジを張られているエッジを境界ノードと呼ぶ．対象ノードと境界ノードを除く，局所グラフ内のノードを

内部ノードと呼ぶ．図 1 の（6）の局所グラフは，7 本の出力エッジ（点線）と 4 本の入力エッジ（破線），11 本の内部エッジ（実線）を持つ．また，対象ノードは 1 個，内部ノードは 2 個，境界ノードは 3 個である．

展開の基準．Chen らは，効率的に部分グラフを拡大するために，ノードの対象ノードに対する影響力を考慮した．あるノードの対象ノードに対する影響力とは，そのノードが持つ PageRank 値のうち，ランダムジャンプを除いて，最終的に対象ノードに到達する PageRank 値の割合のことである．影響力が大きいノードは対象ノードの PageRank 値の推定精度への影響が大きい．また，影響力が大きいノードに対してエッジを張っているノードも大きな影響力を持つと考えられる．さらに，Chen らは fetch 操作の回数を計算コストの指標として用いた．そのため，入次数が大きなノードを展開すると，多くのノードが新たに追加されるため，計算コストが大きくなる．そのため，Chen らは，計算コストと推定精度の両方を考慮するために，影響力を入次数で割った値（影響力/入次数）を展開の基準として用いた．影響力/入次数が閾値以上であるノードを展開した．

影響力の正確な計算は計算コストが大きいため，Chen らは OPIC アプローチ [1] を用いて，以下の手順で影響力を推定した．

1. 影響力を推定したいノード v にスコア 1 を与える．
2. ノード v のスコアを，ノード v から出ているエッジによって遷移させる．この時，出力エッジによって局所グラフの外部へ遷移するスコアは無視する．
3. 対象ノードを除いた，最大のスコアを持つノードをランダムに 1 つ選択し，そのノードのスコアを操作 2 と同様に遷移させる．
4. 対象ノード以外のノードが持つスコアの合計が閾値未満になるまで操作 3 を繰り返す．
5. 終了時の対象ノードのスコアをノード v の対象ノードに対する影響力とする．

OPIC アプローチでは，終了条件である閾値が変化すると，影響力の推定精度と計算時間が変化する．閾値が小さくなるほど，影響力の推定精度が良くなるが，計算時間が長くなる．

OPIC アプローチを用いた影響力推定の例を図 2 に示す．図 1 の（5）のグラフにおいて，ノード 6 の影響力を推定する場合を考える．また，OPIC アプローチの閾値を 0.6 とする．まず，ノード 6 に対してスコア 1 を与える（図 2 の（1））．次に，ノード 6 のスコアを遷移させると，スコアは図 2 の（2）のように遷移する．この時，ノード 6 のスコアはノード 2, 3, 5 だけに遷移し，出力エッジによって遷移するスコアは無視する．そして，対象ノードを除く，最大のスコアを持つノードを選択して，そのスコアを遷移させる．ここでは，ノード 2, 3, 5 のスコアが同じ（0.25）であるため，その中の一つをランダムに選択する．ここでは，ノード 3 が選択されたとする（図 2 の（3））．ノード 3 のスコアを遷移させると，スコアは図 2 の（4）のように遷移する．ここで，対象ノード以外のノードのスコアの合計（$0.5 = 0.25 + 0.25$）が閾値（0.6）未満であるため，OPIC アプローチの処理を中止する．対象ノードのスコアが 0.125 であるため，ノード 6 の対象ノードに対する影響力は 0.125 とする．

3.3 PageRank 値の推定

対象ノードの PageRank 値を推定する前に，境界ノードの PageRank 値を推定する．境界ノードは局所グラフ内からだけではなく，局所グラフ外からもエッジ（入力エッジ）を張られている．そのため，境界ノードの PageRank 値を直接計算することはできず，その PageRank 値を推定する必要がある．また，対象ノードの

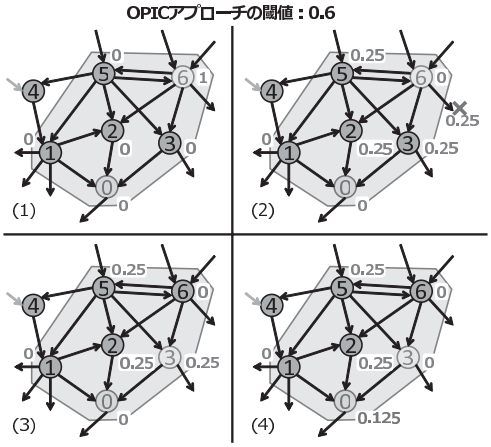

図 2: OPIC アプローチを用いた影響力推定の例

PageRank 値の推定精度は，境界ノードの影響力と境界ノードの PageRank 値の推定精度に影響される．そのため，境界ノードの PageRank 値を精度良く推定することが重要である．Chen らは，ヒューリスティックな方法を用いて，境界ノードの PageRank 値を推定した．具体的には，境界ノードに入っている入力エッジから，PageRank 値 d/E が遷移してくると仮定した．ここで，d は PageRank におけるダンピングファクター，E はグラフ全体におけるエッジの総数である．ランダムジャンプと局所グラフ内のノードから境界ノードに遷移する PageRank 値は計算することができる．これにより，境界ノードの PageRank 値を計算することができる．

最後に，局所グラフ内で PageRank を計算することで対象ノードの PageRank 値を推定する．PageRank の計算の繰り返しごとに，以下の操作を行う．

- 推定した PageRank 値（d/E）が入力エッジから境界ノードに遷移する．
- ランダムジャンプと出力エッジによって局所グラフ外に遷移する PageRank 値を無視する．

PageRank の計算終了後の対象ノードの PageRank 値を，推定した対象ノードの PageRank 値とする．

3.4 考察

Chen らの手法では，局所グラフの作成における計算コストの方が PageRank 値の推定における計算コストよりも大きい．なぜなら，局所グラフを作成する過程において，各境界ノードを展開するかどうかを判定するために，各境界ノードの影響力を一つずつ OPIC アプローチを用いて計算するためである．そのため，局所グラフのサイズが大きくなるほど，計算コストは大きくなる．さらに，局所グラフのサイズは対象ノードの PageRank 値の推定精度に影響を与える．それゆえ，局所グラフの作成における計算コストを小さくすることで，より効率的に PageRank 値を推定することができると考えられる．

また，OPIC アプローチでは，ノードの影響力を推定する時に，すでに推定した影響力を用いていない．展開の判定を行ったノードの影響力はすでに推定されている．しかし，OPIC アプローチでは，すでに推定した影響力を用いずに，反復計算によって境界ノードの影響力を推定する．そのため，すでに推定した影響力を用いることで，より効率的に影響力を推定することができると考えられる．

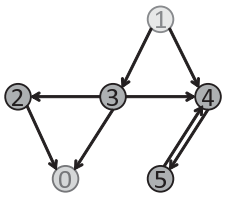

図 3: 影響力の計算例

4 影響力の定義

ノード v のノード u に対する影響力とは，ノード v の PageRank 値のうち，ランダムジャンプを除いて，ノード u に到達する PageRank 値の割合である．本稿では，ノード v のノード u に対する影響力 $inf(v,u)$ を以下のように定義する．

$$inf(v,u) = \begin{cases} \dfrac{1}{outdeg(v)} \displaystyle\sum_{w \in N_v} inf(w,u) & v \text{ から } u \text{ へのパスがある} \\ 0 & v \text{ から } u \text{ へのパスが無い} \\ 1 & v = u \end{cases} \quad (1)$$

ここで，$outdeg(v)$ はノード v の出次数，N_v はノード v がエッジを張っているノードの集合である．すなわち，ノード v のノード u に対する影響力は，ノード v がエッジを張っているノード w のノード u に対する影響力の平均である．ただし，ノード v からノード u へのパスが無い場合，ノード v の PageRank 値はノード u には到達できないため，ノード v のノード u に対する影響力は 0 である．

図 3 を用いて，影響力の計算例を示す．ノード 1 のノード 0 に対する影響力 $inf(1,0)$ を計算すると，

$$\begin{aligned} inf(1,0) &= \frac{1}{2}(inf(3,0) + inf(4,0)) \\ &= \frac{1}{2}\cdot\frac{1}{3}(inf(2,0) + inf(0,0) + inf(4,0)) + \frac{1}{2}\cdot 0 \\ &= \frac{1}{6}\cdot\frac{1}{1}inf(0,0) + \frac{1}{6}\cdot 1 + \frac{1}{6}\cdot 0 = \frac{1}{3} \end{aligned}$$

5 提案手法

5.1 アイデア

4 節の定義より，あるノードの影響力はそのノードがエッジを張っているノードの影響力を用いて計算することができる．しかし、グラフにループが存在する場合は再帰的な計算が必要となってしまう。また、局所グラフを作成する過程において、あるノードがエッジを張っている全てのノードが局所グラフに追加されているとは限らない。そのため提案手法では、局所グラフに追加されているノードの影響力だけを用いてノードの影響力を推定する。このように影響力を近似することにより再帰計算を避ける事が可能である。例えば，図 1 の（5）を考える．この時，ノード 1, 2, 3 の影響力はすでに推定されている．そのため，ノード 5，6 の影響力は，ノード 1，2，3 の影響力を用いて計算することができる．

しかし，影響力を推定したい境界ノードが複数あり，それらの境界ノード間にエッジが張られている場合に問題がある．なぜなら，それらの境界ノードの影響力はまだ推定されていないためである．例えば，図 1 の（5）では，ノード 5（6）の影響力を計算するために，ノード 6（5）の影響力が必要である．このような場合，未知の影響力を変数とする連立一次方程式を用いて，ノードの影響力を表現する．そして，この連立一次方程式を解くことで，ノードの影響力を推定する．

5.2 影響力の推定手法

提案手法では，あるノードがエッジを張っている，局所グラフ内のノードの影響力を用いて，そのノードの影響力を推定する．グラフ全体を $G = (V, E)$，対象ノードを $v_t \in V$ とする．また，現在の局所グラフを $G_S = (V_S, E_S)$．さらに，$C = \{v_i |$ これから影響力を推定する境界ノード $v_i \in V_S\}$，すでに影響力を推定したノードの集合を $W = V_S - C$ とする．ノード $v_i \in C$ の対象ノード $v_t \in W$ に対する影響力 α_i は以下のように定義される．

$$\alpha_i = \frac{1}{outdeg(v_i)}\left(\sum_{v_j \in C_i} \alpha_j + \sum_{v_k \in W_i} \beta_k\right) \quad (2)$$

$$\beta_t = 1.0 \quad (3)$$

ここで，$outdeg(v_i)$ はノード v_i の出次数，C_i（W_i）はノード v_i がエッジを張っている集合 C（W）に含まれるノード集合，α_j（β_k）はノード $v_j \in C$（$v_k \in W$）の影響力を表している．

式（2）では，ノード v_i がエッジを張っている，局所グラフ内のノード（C_i, W_i）の影響力の合計をノード v_i の出次数で割ることで，ノード v_i の対象ノード v_t に対する影響力を計算している．

ノード v_i が他の境界ノード $v_j \in C_i$ にエッジを張っている場合，ノード v_i の影響力 α_i についての式は未知数 α_j を含む．そのため，影響力 α_i を計算するために未知数 α_j を計算する必要がある．ここで，影響力についての式の数は未知数の数と同じ（$|C|$ 個）であり，影響力についての式はグラフの位相を反映しているため線形独立である．すなわち，影響力についての式が線型従属になるような，グラフの接続関係のパターンは存在しない．そのため，各境界ノード $v_i \in C$ の影響力 α_i についての式を連立一次方程式として解くことで，影響力 α_i（$i = 1, 2, ..., |C|$）を計算することができる．

提案手法では，以下の手順で局所グラフを作成する．

1. 対象ノード v_t の影響力を 1 として，境界ノード集合 C を作成する．
2. 集合 C 内の境界ノードの影響力についての式を連立一次方程式として解き，各境界ノードの影響力を計算する．
3. 各境界ノードの影響力と閾値を比較し，閾値以上の影響力を持つ境界ノードを展開する．新しく局所グラフに追加されたノードを次の境界ノードとする．
4. 展開されるノードが無くなるまで，操作 2，3 を繰り返す．

ここで，3 節で説明した Chen らの手法とは異なり，展開の基準として “影響力” を用いている．これは，評価実験において，“影響力/入次数” を用いたところ，全ての比較手法（詳細は 6.1 節）の性能が非常に悪かったためである．6 節の評価実験においても，展開の基準として “影響力” を用いる．また，提案手法と Chen らの手法では，影響力の推定方法が異なっているため，最終的に作成される局所グラフは異なる．

5.3 提案手法を用いた局所グラフの作成例

図 4 に提案手法を用いた局所グラフの作成例を示す．この例では，ノード 0 を対象ノード，影響力の閾値を 0.5 とする（図 4 の（1））．

まず，対象ノード 0 に fetch 操作を行い，対象ノード 0 にエッジを張っているノード 1，2，3 を局所グラフに追加する（図 4 の（2））．そして，ノード 1，2，3 に fetch 操作を行ってノードの情報を取

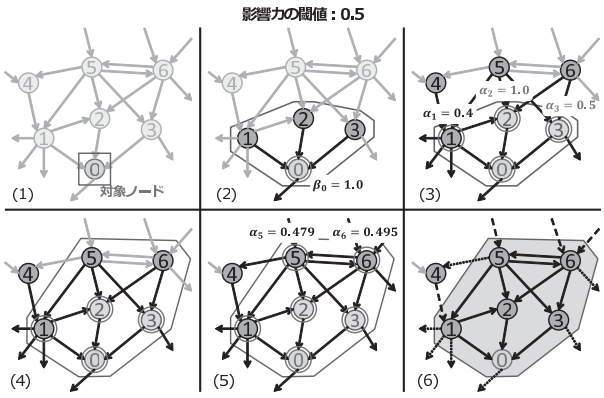

図 4: 提案手法を用いた局所グラフの作成例

得し，これらのノードの影響力を計算する．ここで，$C = \{1, 2, 3\}$, $W = \{0\}$ であり，各ノードについて以下のことが分かる．

ノード	*outdeg*(i)	C_i	W_i
1	5	{2}	{0}
2	1	∅	{0}
3	2	∅	{0}

この表より，以下の連立一次方程式が得られる．

$$\alpha_1 = \frac{1.0}{5}(\alpha_2 + 1.0) \quad (4)$$

$$\alpha_2 = 1.0 \cdot 1.0 \quad (5)$$

$$\alpha_3 = \frac{1.0}{2} \cdot 1.0 \quad (6)$$

この連立一次方程式を解くことで，各ノードの影響力 $\alpha_1 = 0.4$ $(= \beta_1)$，$\alpha_2 = 1.0$ $(= \beta_2)$，$\alpha_3 = 0.5$ $(= \beta_3)$ が得られる．影響力の閾値は 0.5 であるため，閾値以上の影響力を持つノード 2，3 を展開する（図 4 の (3)）．

次に，ノード 2，3 を展開することで，ノード 5，6 が局所グラフに追加される（図 4 の (4)）．そして，ノード 5，6 に fetch 操作を行ってノードの情報を取得し，これらのノードの影響力を計算する．ここで，$C = \{5, 6\}$, $W = \{0, 1, 2, 3\}$ であり，各ノードについて以下のことが分かる．

ノード	*outdeg*(i)	C_i	W_i
5	5	{6}	{1, 2, 3}
6	4	{5}	{2, 3}

この表より，以下の連立一次方程式が得られる．

$$\alpha_5 = \frac{1.0}{5}(\alpha_6 + 0.4 + 1.0 + 0.5) \quad (7)$$

$$\alpha_6 = \frac{1.0}{4}(\alpha_5 + 1.0 + 0.5) \quad (8)$$

この連立一次方程式を解くことで，各ノードの影響力 $\alpha_5 = 0.479$ $(= \beta_5)$，$\alpha_6 = 0.495$ $(= \beta_6)$ が得られる（図 4 の (5)）．閾値 (0.5) 以上の影響力を持つノードが無いため，局所グラフの作成を終了する．結果として，図 4 の (6) の局所グラフが得られる．

5.4 考察

一般に連立一次方程式を解くための計算量は大きいため，提案手法の計算コストも大きいと考えるかもしれない．連立一次方程式をシンプルに解く場合の計算量は $O(N^3)$（N は方程式の数）である．しかし，展開の各ステップにおいて追加される境界ノードの数はあまり多くない．そのため，提案手法は OPIC アプローチよりも効率的に影響力を推定できると考えられる．

6 評価実験

本実験では，提案手法と Chen らの手法を比較し，提案手法が精度を維持しながら高速に PageRank を推定できることを検証する．

6.1 実験設定

データセット．本実験では，以下の二つのデータセットを用いる．

- **Twitter** ソーシャルグラフ（**TSG**）：*TSG* は，我々が Twitter をクローリングして作成したデータセットである．一人のユーザをシードユーザとして選択し，シードユーザがフォローしているユーザを幅優先探索で収集した．収集したユーザ数が 2,500,000 になったら収集を終了した．その後，収集されていないユーザへのエッジを削除した．*TSG* はノード数が 2,500,000，エッジ数が 28,810,947 である．
- **Web** グラフ（**WG**）：*WG* は Stanford Network Analysis Platform (SNAP) が提供している Google の Web グラフ[3]のデータセットである．*WG* には，Web ページの ID と Web ページ間に張られているリンクの情報が含まれている．*WG* はノード数が 875,713，エッジ数が 5,105,039 である．

実験環境．本実験では，比較手法（6.1 節）を Python で実装した．連立一次方程式を解く時には，Python のライブラリ Scipy[4]を用いた．また，実験に用いたコンピュータは，OS が Ubuntu 13.04, CPU が Intel Core i7-3770 (3.40GHz)，メモリが 32GB であった．さらに，各手法においてグラフデータをメモリ上に読み込むように実装した．

比較手法．本実験では，以下の三つの手法を比較した．

- 提案手法：5 節で説明した提案手法．
- **Chen** らの手法：3 節で説明した Chen らの手法．ただし，展開の基準として “影響力” を用いる．
- **Naive** な手法：対象ノードまでの距離が s 以下であるノードを全て含む局所グラフを作成する手法．すなわち，全ての境界ノードを展開する．

また，本実験では，Naive な手法と比較するために，提案手法と Chen らの手法に対して，展開の最大ステップ数 s の制限を設定した．具体的には，提案手法と Chen らの手法において，以下の二つの条件を満たすノードを展開する．

- 対象ノードに対する影響力が閾値以上である．
- 対象ノードまでの距離が s 未満である．

対象ノードまでの距離が s 未満であるノードを展開することで，対象ノードまでの距離が最大で s 以下であるノードを含む局所グラフが作成される．これにより，Naive な手法と比較して，提案手法と Chen らの手法は，精度を維持しながら，どのくらい局所グラフサイズを小さくすることができるかを評価することができる．

実験方法．まず，グラフ全体に対して PageRank を計算し，各ノードの正確な PageRank 値を求めた．次に，PageRank 値を推定する対象ノードをサンプリングした．本実験では，PageRank 値の上位 1,000 ノードから 100 ノードを対象ノードとしてランダムサンプリングした．そして，各比較手法を用いて，サンプリングされた各対象ノードの PageRank 値を推定した．本実験では，Chen らの手法を参考にして，入力エッジから評価値 d/E が遷移してくるとした．また，ダンピングファクター d の値を 0.85 に設定した．

[3] https://snap.stanford.edu/data/web-Google.html
[4] http://www.scipy.org/

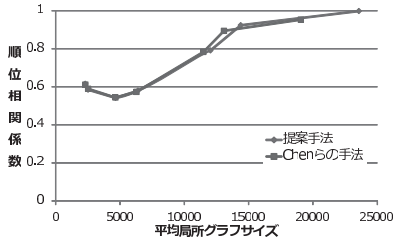

図 5: 提案手法，Chen らの手法：順位相関係数と平均局所グラフサイズの比較（*TSG*）

評価指標．本実験では，推定精度と計算コストの観点から比較手法を評価した．ノードの PageRank 値そのものよりもノードのランキングの方に興味があるため，推定した PageRank 値に基づくノードのランキングについて評価する．

推定精度の評価指標として，スピアマンの順位相関係数 [14] を用いた．スピアマンの順位相関係数 ρ は以下のように定義される．

$$\rho = 1 - \frac{6\Sigma D^2}{N^3 - N} \tag{9}$$

ここで，D は正確な PageRank 値に基づくサンプリングした 100 個の対象ノードのランキングと推定した PageRank 値に基づくサンプリングした 100 個の対象ノードのランキングにおける同一ノードの順位差，N はノード数（本実験では 100）である．順位相関係数 ρ の値が 1 に近いほど，推定した PageRank 値に基づくランキングは精度良く推定できていることを示している．

また，計算コストを評価するために，局所グラフサイズ，局所グラフの作成時間，PageRank 値の推定時間を用いた．局所グラフに追加されたノードだけに fetch 操作を行うので，局所グラフサイズは fetch 操作の回数と同じである．また，各比較手法における局所グラフの作成にかかる時間と PageRank 値の推定にかかる時間を比較する．

6.2 提案手法と Chen らの手法の比較

予備実験によって，ステップ数 s と Chen らの手法における OPIC アプローチの閾値を決定した．具体的には，*TSG* では，ステップ数は 4，OPIC アプローチの閾値は 0.5 を用いた．*WG* では，ステップ数は 3，OPIC アプローチの閾値は 0.9 を用いた．

推定精度．図 5，6 は，*TSG* と *WG* を用いた実験における，提案手法と Chen らの手法の順位相関係数と平均局所グラフサイズの比較結果を示している．縦軸は順位相関係数，横軸は平均局所グラフサイズを示している．グラフ上の各点は，各手法の各閾値における結果に対応している．

図 5，6 より，平均局所グラフサイズが同程度の時，提案手法と Chen らの手法の順位相関係数が同程度であることが分かった．図 5 より，*TSG* を用いた実験では，平均局所グラフサイズが大きくなるほど，順位相関係数が向上した．また，図 6 より，*WG* を用いた実験では，順位相関係数は平均局所グラフサイズにかかわらずほぼ同じであった．

計算コスト．図 7 – 12 は，*TSG* と *WG* を用いた実験における，提案手法と Chen らの手法の計算時間と平均局所グラフサイズの比較結果を示している．縦軸は，局所グラフの平均作成時間，評価値の平均推定時間，合計平均時間のいずれか，横軸は平均局所グラフサイズを表している．合計平均時間は局所グラフの平均作成時間と評価値の平均推定時間の合計である．グラフ上の各点は，各手法の各閾値における結果に対応している．

図 7，10 より，平均局所グラフサイズが同程度の時，提案手法の方が効率的に PageRank 値を推定できることが分かった．具体的には，図 8，11 より，平均局所グラフサイズが同程度の時，提案手法の方が局所グラフの平均作成時間が非常に小さかった．これは，提案手法は，反復計算ではなく再帰的な計算によって影響力を計算しているため，より高速に局所グラフを作成することができたためである．また，図 9，12 より，平均局所グラフサイズが同程度の時，提案手法と Chen らの手法の評価値の平均推定時間は同程度であった．

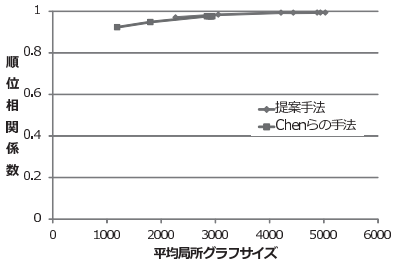

図 6: 提案手法，Chen らの手法：順位相関係数と平均局所グラフサイズの比較（*WG*）

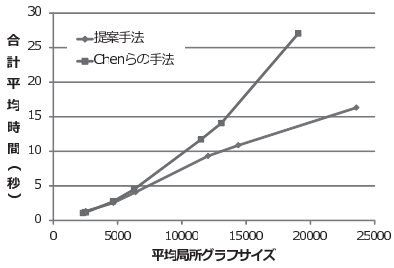

図 7: 提案手法，Chen らの手法：合計平均時間と平均局所グラフサイズの比較（*TSG*）

6.3 提案手法と Naive な手法の比較

表 1 は，提案手法と Naive な手法の順位相関係数と平均局所グラフサイズの比較結果を示している．提案手法の影響力の閾値は，性能が最も良かった時の閾値を用いた．具体的には，*TSG* では 0.0001，*WG* では 0.05 である．表 1 より，Naive な手法と比較して，提案手法は順位相関係数は同程度であるが，平均局所グラフサイズが非常に小さいことが分かった．これは，提案手法は影響力を考慮しながら展開するノードを選択しているため，PageRank 値の推定に不要なノードを局所グラフに追加していないためである．

表 1: 提案手法と Naive な手法の比較

	TSG		*WG*	
	Naive	提案手法	Naive	提案手法
順位相関係数	1.000	0.998	0.993	0.983
平均局所グラフサイズ	57,247	23,578	5,029	3,055

7 まとめと今後の課題

本稿では，局所グラフを用いて特定のノードの PageRank 値を効率的に推定する手法を提案した．提案手法では，境界ノードとその隣接ノードの接続関係に基づいて作られる連立一次方程式を解くことで，ノードの影響力を推定した．これにより，反復計算を行わないため，提案手法は高速に影響力を推定することができた．評価実験により，Chen らの手法と比較して，提案手法は推定精度を維持しながら，効率的に PageRank 値を推定できる

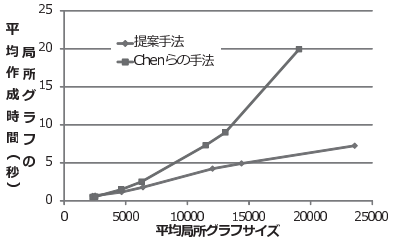

図 8: 提案手法，Chen らの手法：局所グラフの平均作成時間と平均局所グラフサイズの比較（*TSG*）

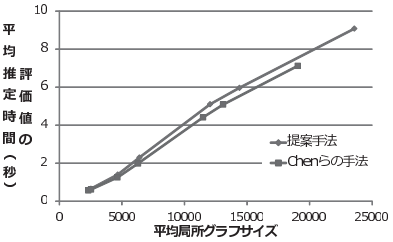

図 9: 提案手法，Chen らの手法：評価値の平均推定時間と平均局所グラフサイズの比較（*TSG*）

ことを示した．具体的には，提案手法は，最大で約 3.2 倍高速に PageRank 値を推定することができた．

今後の課題として，ObjectRank [3] などの他のランキング手法に提案手法を適用することが挙げられる．また，影響力の閾値は推定精度にとって重要な要素であり，適切な閾値はデータセットの性質によって異なる．そのため，適切な閾値を自動的に決定することができれば，推定精度を向上させることができると考えられる．さらに，提案手法について理論的な分析を行いたいと考えている．

【謝辞】

共同研究（富士通研究所 CPE25149），JSPS 科研費（25330124），文部科学省委託事業「ビッグデータ利活用のためのデータ連携技術に関するフィージビリティスタディ及び予備研究」による．

[文献]

[1] Serge Abiteboul, Mihai Preda, and Gregory Cobena. Adaptive on-line page importance computation. In *World Wide Web Conference Series*, pages 280–290, 2003.

[2] Arvind Arasu, Jasmine Novak, Andrew Tomkins, and John Tomlin. PageRank Computation and the Structure of the Web: Experiments and Algorithms. In *World Wide Web Conference Series*, 2002.

[3] Andrey Balmin, Vagelis Hristidis, and Yannis Papakonstantinou. ObjectRank: Authority-Based Keyword Search in Databases. In *Very Large Data Bases*, pages 564–575, 2004.

[4] Ziv Bar-yossef and Li tal Mashiach. Local approximation of pagerank and reverse pagerank. In *International Conference on Information and Knowledge Management*, pages 279–288, 2008.

[5] Marco Bressan and Luca Pretto. Local computation of PageRank: the ranking side. In *International Conference on Information and Knowledge Management*, pages 631–640, 2011.

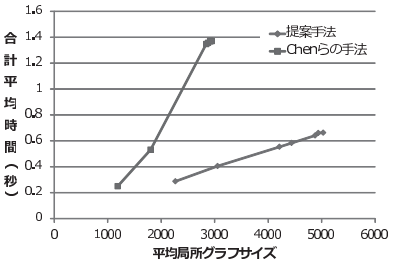

図 10: 提案手法，Chen らの手法：合計平均時間と平均局所グラフサイズの比較（*WG*）

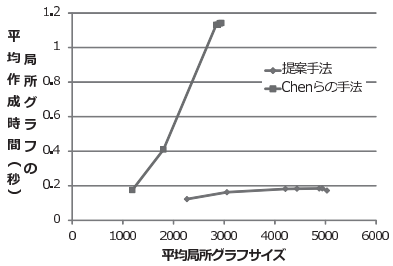

図 11: 提案手法，Chen らの手法：局所グラフの平均作成時間と平均局所グラフサイズの比較（*WG*）

[6] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and Isdn Systems*, 30:107–117, 1998.

[7] Andrei Z. Broder, Ronny Lempel, Farzin Maghoul, and Jan O. Pedersen. Efficient PageRank approximation via graph aggregation. *Information Retrieval*, 9:123–138, 2006.

[8] Yen-Yu Chen, Qingqing Gan, and Torsten Suel. Local methods for estimating pagerank values. In *International Conference on Information and Knowledge Management*, pages 381–389, 2004.

[9] Jason V. Davis and Inderjit S. Dhillon. Estimating the global pagerank of web communities. In *Knowledge Discovery and Data Mining*, pages 116–125, 2006.

[10] T.H. Haveliwala. Efficient computation of pagerank. Technical report, Stanford InfoLab, October 1999.

[11] Sepandar D. Kamvar, Taher H. Haveliwala, Christopher D. Manning, and Gene H. Golub. Extrapolation methods for accelerating PageRank computations. In *World Wide Web Conference Series*, pages 261–270, 2003.

[12] Julie L. Morrison, Rainer Breitling, Desmond J. Higham, and David R. Gilbert. GeneRank: Using search engine technology for the analysis of microarray experiments. *BMC Bioinformatics*, 6, 2005.

[13] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, Stanford InfoLab, November 1999.

[14] Charles Spearman. FOOTRULE FOR MEASURING CORRELATION. *British Journal of Psychology*, 2:89–108, 1906.

[15] Jianshu Weng, Ee-Peng Lim, Jing Jiang, and Qi He. TwitterRank: finding topic-sensitive influential twitterers. In *Web Search and Data Mining*, pages 261–270, 2010.

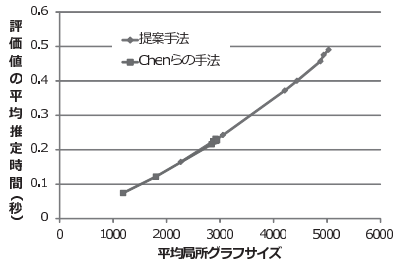

図 12: 提案手法，Chen らの手法：評価値の平均推定時間と平均局所グラフサイズの比較（*WG*）

[16] Yao Wu and Louiqa Raschid. ApproxRank: Estimating Rank for a Subgraph,. In *International Conference on Data Engineering*, pages 54–65, 2009.

[17] Gui-Rong Xue, Hua-Jun Zeng, Zheng Chen, Wei-Ying Ma, Hong-Jiang Zhang, and Chao-Jun Lu. Implicit link analysis for small web search. In *Research and Development in Information Retrieval*, pages 56–63, 2003.

坂倉 悠太　Yuta SAKAKURA

平成 24 年筑波大学情報学群情報科学類を卒業．平成 26 年筑波大学大学院システム情報工学研究科博士前期課程を修了．

山口 祐人　Yuto YAMAGUCHI

2014 年筑波大学大学院システム情報工学研究科博士後期課程修了．博士（工学）．現在，筑波大学大学院システム情報工学研究科博士研究員，日本学術振興会特別研究員（PD）．データマイニングに関する研究に従事．情報処理学会正会員．

天笠 俊之　Toshiyuki AMAGASA

筑波大学システム情報系准教授．データ工学，データベース，Web マイニング等の研究に従事．日本データベース学会，情報処理学会，ACM 各会員．電子情報通信学会，IEEE 各シニア会員．

北川 博之　Hiroyuki KITAGAWA

1978 年東京大学理学部物理学科卒業．1980 年同大学理学系研究科修士課程修了．日本電気（株）勤務の後，筑波大学講師，助教授を経て，現在，筑波大学システム情報系教授，同計算科学研究センター教授併任．理学博士（東京大学）．データベース，データマイニング，情報検索等の研究に従事．本会会長，情報処理学会フェロー，電子情報通信学会フェロー，ACM，IEEE-CS，日本ソフトウェア科学会，各会員．