PageRank のための高速な Top-k 検索

Efficient Top-k Search for PageRank

藤原 靖宏
Yasuhiro Fujiwara

NTT ソフトウェアイノベーションセンタ
NTT Software Innovation Center
fujiwara.yasuhiro@lab.ntt.co.jp

中辻 真
Makoto Nakatsuji

NTT サービスエボリューション研究所
NTT Service Evolution Laboratories
nakatsuji.makoto@lab.ntt.co.jp

塩川 浩昭
Hiroaki Shiokawa

NTT ソフトウェアイノベーションセンタ
NTT Software Innovation Center
shiokawa.hiroaki@lab.ntt.co.jp

三島 健
Takeshi Mishima

(同 上)
mishima.takeshi@lab.ntt.co.jp

鬼塚 真
Makoto Onizuka

大阪大学
Osaka University
oni@acm.org

keywords: graph mining, efficient, top-k search

Summary

In AI communities, many applications utilize *PageRank*. To obtain high PageRank score nodes, the original approach iteratively computes the PageRank score of each node until convergence from the whole graph. If the graph is large, this approach is infeasible due to its high computational cost. The goal of this study is to find top-k PageRank score nodes efficiently for a given graph without sacrificing accuracy. Our solution, *F-Rank*, is based on two ideas: (1) It iteratively estimates lower/upper bounds of PageRank scores, and (2) It constructs subgraphs in each iteration by pruning unnecessary nodes and edges to identify top-k nodes. Experiments show that F-Rank finds top-k nodes much faster than the original approach.

1. はじめに

PageRank は人工知能の分野において、グラフにおけるノードの重要性を計算する手法のひとつである [Page 99]。しかし PageRank の問題点として高い計算コストがある。それは PageRank の計算ではグラフ全体を用いて全てのノードのスコアが収束するまで繰り返し計算を行うからである。本論文では PageRank が上位 k 個のノードを高速に検索する問題に取り組む。なお本論文においては PageRank の値が同じノードは同様に扱うものとする。そのため k 番目の PageRank の値を複数のノードが持つとき、それらのノードをすべて求めることとする。

本論文では *F-Rank* を提案する [Fujiwara 13]。提案手法は繰り返し計算の中で再帰的に類似度の下限値と上限値を推定し、動的に解ノードになり得ないノードを枝刈りする。PageRank を高速に計算する方法について様々な手法が提案されてきた。例えば Kamvar らは PageRank の計算において値が収束したノードを更新しない手法を提案した [Kamvar 04]。またグラフが動的に変化するという性質を用いて PageRank を高速に計算する手法も提案されている [Bahmani 12]。またモンテカルロ法を用い

て計算する手法も提案されている [Avrachenkov 07]。しかしこれらの手法は PageRank の値を正確に計算できないという問題がある。提案手法の特徴として以下がある。

- 高速性: 従来の繰り返し計算に基づくオリジナルの手法と比較して提案手法はより高速に検索が可能。
- 正確性: 正確に上位 k 個のノードを検索可能。
- 柔軟性: 提案手法は事前計算を必要としないため任意のグラフに対してアドホックに検索可能。
- パラメータフリー: 提案手法に必要な内部パラメータの設定はない。そのためユーザは PageRank による検索を簡易に行うことができる。

本論文は国際会議で発表した論文 [Fujiwara 13] の完全版で、国際会議の論文では紙幅の関係で省略した証明を記載し、論文として完全性を向上する修正を行った。

2. 前 準 備

まず本論文で用いる記号を定義し、背景技術の詳細を説明する。PageRank ではランダムなノードからランダムウォークを開始し、各ステップにおいて再帰的にランダムウォークを確率 s ($0 < s < 1$) で繰り返す。また各ス

トップにおいて一定の確率 $1-s$ でランダムなノードにジャンプする．集合 $\mathbb{V}$ と $\mathbb{E}$ をそれぞれグラフ全体のノードとエッジの集合とすると，問い合わせ対象のグラフは $\mathbb{G} = (\mathbb{V}, \mathbb{E})$ と表現できる． $\mathbf{p}$ を第 u 成分 $p[u]$ がノード u の PageRank のスコアに対応するような列ベクトルとする．また N をグラフのノード数としたときに， $\mathbf{e}$ を全ての成分の値が $1/N$ である列ベクトルとする．また $W[u, v]$ をノード v からノード u へ移動する確率としたときに， W を列成分が正規化されたグラフの隣接行列とする．各ノードの PageRank のスコアは以下の式を再帰的に収束するまで繰返し計算を行うことで得られる．

$$\mathbf{p}_i = s\mathbf{W}\mathbf{p}_{i-1} + (1-s)\mathbf{e} \quad (1)$$

ここでもし $i=0$ であれば $\mathbf{p}_i$ は $\mathbf{e}$ になる．この手法は各ノードにおける PageRank のスコアが収束するまで行う． M をグラフのエッジ数とし T を収束するまでの計算回数とすると，この計算には $O((N+M)T)$ の計算コストが必要となる．そのため大規模なグラフに対して高速に検索が行えないという問題がある．

3. 提案手法

ここではまず手法の概要を述べてから具体的に上位 k 個のノードを検索する方法について述べる．

3.1 手法概要

提案手法は高速に検索するために PageRank のスコアの下限值と上限値を推定する．オリジナルの手法のようにグラフ全体を用いず，推定値により不要なノードとエッジを繰返し計算において枝刈りし，グラフの一部を用いて検索を行う．

提案手法には様々な利点がある．まず k 個のノードが検索結果として特定されればスコアの収束を待つことなく繰返し計算を打ち切ることができる．そのためオリジナルの手法と比較して少ない繰返し計算回数で検索を行うことができる．また提案手法は推定値を用いて検索を行うが，検索結果は理論的に正確であることが保証されている．これは推定値により検索結果に影響を与えないことが保証されているノードのみを枝刈りすることができるからである．また推定値を用いることにより任意に与えられたグラフに対して高速に計算対象のグラフを構築することができる．推定値から検索結果の計算に不要なノードとエッジを特定することができる．そのため提案手法は検索に必要なノードとエッジのみを有する計算対象グラフを動的に構築することができる．結果としてグラフ全体を用いるオリジナルの手法と比較して，提案手法は高速な検索を行うことができる．また提案手法に必要な内部パラメータの設定はない．そのためユーザは PageRank による検索を簡易に行うことができる．

3.2 下限値と上限値の推定

ここでは下限値と上限値の推定方法を述べ，またそれらの性質を示す． i ($i=0, 1, 2, \dots$) 番目の繰返し計算において候補ノードの集合に含まれるノードの推定値を計算する．候補ノードの集合を求める方法については後に述べる．上限値を計算するために候補ノードの集合 $\mathbb{C}_i$ に到達可能なノードの集合 $\mathbb{R}_i$ を用いる．ここでノード u がノード v に到達可能とは，ノード u からノード v にグラフ上でパスが存在するということである．また第 u 成分がエッジの最大の重み $\overline{W}[u] = \max\{W[u, v] : v \in \mathbb{V}\}$ となる $N \times 1$ の列ベクトルを $\overline{W}$ とする．また長さが i のランダムウォークの確率を $N \times 1$ の列ベクトル $\mathbf{r}_i$ とする．ここでグラフの隣接行列 W の i 乗を用いて $\mathbf{r}_i$ は $\mathbf{r}_i = \mathbf{W}^i \mathbf{e}$ と計算できる．なおもし $i=0$ であれば $\mathbf{W}^i = \mathbf{I}$ とする ($\mathbf{I}$ は単位行列)． i 番目の繰返し計算における下限値 $\underline{p}_i$ と上限値 $\overline{p}_i$ を以下のように定義する．

【定義 1】(下限値) i 番目の繰返し計算における下限値 $\underline{p}_i$ は以下のように計算する．

$$\underline{p}_i = (1-s) \sum_{j=0}^i s^j \mathbf{r}_j \quad (2)$$

【定義 2】(上限値) i 番目の繰返し計算における上限値 $\overline{p}_i$ は以下のように計算する．

$$\overline{p}_i = (1-s) \sum_{j=0}^i s^j \mathbf{r}_j + s^{i+1} \mathbf{r}_i + \Delta_i \sigma_i \overline{W} \quad (3)$$

式 (3) において $\sigma_i = s^{i+1}(1-s)^{-1}$ であり Δ_i はベクトル $\mathbf{r}^i$ の成分を用いて以下のように計算する．

$$\Delta_i = \begin{cases} 1 & (i=0) \\ \sum_{u \in \mathbb{R}_i} \Delta_i[u] & (i \neq 0) \end{cases} \quad (4)$$

ここで $\Delta_i[u] = \max\{r_i[u] - r_{i-1}[u], 0\}$ である．

【補題 1】(下限値) i 番目の繰返し計算においてベクトル $\mathbf{p}$ と $\underline{\mathbf{p}}_i$ の第 u 成分において $\underline{p}_i[u] \leq p[u]$ が成り立つ．

《証明》 式 (1) から

$$\begin{aligned} \mathbf{p}_i &= s\mathbf{W}\mathbf{p}_{i-1} + (1-s)\mathbf{e} = s^2\mathbf{W}^2\mathbf{p}_{i-2} + (1-s)(s\mathbf{W}\mathbf{e} + \mathbf{e}) \\ &= s^i\mathbf{W}^i\mathbf{p}_0 + (1-s)(s^{i-1}\mathbf{W}^{i-1}\mathbf{e} + s^{i-2}\mathbf{W}^{i-2}\mathbf{e} + \dots + \mathbf{e}) \\ &= s^i\mathbf{W}^i\mathbf{e} + (1-s) \sum_{j=0}^{i-1} (s^j\mathbf{W}^j\mathbf{e}) \end{aligned}$$

となる．ページランクの各ノードのスコアは式 (1) の収束値であるため $\mathbf{p} = \mathbf{p}_\infty$ となる．そのため $0 < s < 1$ であり行列 $\mathbf{W}^\infty$ の成分の値は 0 から 1 であるため

$$\mathbf{p} = s^\infty \mathbf{W}^\infty \mathbf{e} + (1-s) \sum_{j=0}^\infty (s^j \mathbf{W}^j \mathbf{e}) = (1-s) \sum_{j=0}^\infty s^j \mathbf{r}_j$$

となる．この式からノード u において以下の不等式が成り立つ．

$$p[u] = (1-s) \sum_{j=0}^\infty s^j r_j[u] \geq (1-s) \sum_{j=0}^i s^j r_j[u] = \underline{p}_i[u]$$

□

[補題 2] (上限値) i 番目の繰返し計算において $\bar{p}_i[u] \geq p[u]$ がベクトル $\mathbf{p}$ と $\bar{\mathbf{p}}_i$ に対して成り立つ。

《証明》 上記の証明にあるとおり

$$\begin{aligned} p[u] &= (1-s) \sum_{j=0}^{\infty} s^j r_j[u] \\ &= (1-s) \sum_{j=0}^i s^j r_j[u] + (1-s) \sum_{j=1}^{\infty} s^{i+j} r_{i+j}[u] \end{aligned}$$

となる。まず $r_{i+j}[u] \leq r_i[u] + j\Delta_i \bar{W}[u]$ が成り立つことを示す。 $\mathbb{H}_j[u]$ を j ホップでノード u へ到達できるノードの集合とする。なおここで $\mathbb{H}_j[u] \subseteq \mathbb{R}_i \subseteq \mathbb{V}$ となる。 $\mathbf{r}_{i+j} - \mathbf{r}_{i+j-1} = \mathbf{W}^{i+j} \mathbf{e} - \mathbf{W}^{i+j-1} \mathbf{e} = \mathbf{W} \mathbf{W}^{j-1} (\mathbf{r}_i - \mathbf{r}_{i-1})$, であるため

$$\begin{aligned} r_{i+j}[u] - r_{i+j-1}[u] &= \sum_{v \in \mathbb{H}_1[u]} \sum_{w \in \mathbb{H}_{j-1}[v]} W[u, v] W^{j-1}[v, w] (r_i[w] - r_{i-1}[w]) \\ &\leq \sum_{w \in \mathbb{H}_{j-1}[v]} \sum_{v \in \mathbb{H}_1[u]} \bar{W}[u] W^{j-1}[v, w] \Delta_i[w] \\ &\leq \bar{W}[u] \sum_{w \in \mathbb{R}_i} \Delta_i[w] \left(\sum_{v \in \mathbb{H}_1[u]} W^{j-1}[v, w] \right) \end{aligned}$$

となる。ここで $\mathbf{W}^{j-1}$ は列が正規化された行列であるため $\sum_{v \in \mathbb{H}_1[u]} W^{j-1}[v, w] \leq 1$ となる。そのため

$$r_{i+j}[u] - r_{i+j-1}[u] \leq \bar{W}[u] \sum_{w \in \mathbb{R}_i} \Delta_i[w] = \Delta_i \bar{W}[u]$$

となる。よって

$$r_{i+j}[u] \leq r_{i+j-1}[u] + \Delta_i \bar{W}[u] \leq \dots \leq r_i[u] + j\Delta_i \bar{W}[u]$$

となる。この性質を用いて

$$(1-s) \sum_{j=1}^{\infty} s^{i+j} r_{i+j}[u] \leq (1-s) \sum_{j=1}^{\infty} (s^{i+j} r_i[u] + j s^{i+j} \Delta_i \bar{W}[u])$$

となる。 $\sum_{j=1}^{\infty} s^{i+j} \leq \frac{s^{i+1}}{1-s}$ と $\sum_{j=1}^{\infty} j s^{i+j} \leq \frac{\sigma[i]}{1-s}$ から

$$(1-s) \sum_{j=1}^{\infty} s^{i+j} r_{i+j}[u] \leq s^{i+1} r_i[u] + \Delta_i \sigma[i] \bar{W}[u]$$

となる。そのため式 (3) より

$$p[u] \leq (1-s) \sum_{j=0}^i s^j r_j[u] + s^{i+1} r_i[u] + \Delta_i \sigma[i] \bar{W}[u] = \bar{p}_i[u]$$

となる。よって成り立つ。□

なお PageRank の値が同じノードに対しては繰返し計算において下限値と上限値は同じ値になる性質がある。

[補題 3] (推定値の収束) 推定値は PageRank の正確なスコアに収束する。すなわち $\underline{p}_{\infty}[u] = \bar{p}_{\infty}[u] = p[u]$ となる。

《証明》 まず $\underline{p}_{\infty}[u] = p[u]$ であることを示す。これは $\underline{p}_{\infty}[u] = (1-s) \sum_{j=0}^{\infty} s^j r_j[u] = p[u]$ であり明らかである。次に $\bar{p}_{\infty}[u] = p[u]$ であることを示す。 $\bar{p}_{\infty}[u] = (1-s) \sum_{j=0}^{\infty} s^j r_j[u] + s^{\infty} r_{\infty}[u] + \Delta_{\infty} \sigma[\infty] \bar{W}[u]$ であるが、ここで $0 \leq s, r_{\infty}[u] \leq 1$ であるため $s^{\infty} r_{\infty}[u] = 0$ となる。また $0 \leq \Delta_{\infty}, \bar{W}[u] \leq 1$ であり $\sigma[\infty] = s^{\infty} (1-s)^{-1} = 0$ であるため $\Delta_{\infty} \sigma[\infty] \bar{W}[u] = 0$ となる。よって $\bar{p}_{\infty}[u] = (1-s) \sum_{j=0}^{\infty} s^j r_j[u] = p_{\infty}[u]$ となる。よって成り立つ。□

補題 3 は提案手法が収束することを示す。

3.3 計算対象グラフの構築

提案手法は再帰的に上位 k 個のノードを検索するために候補ノードを計算し、もし候補ノードの数が k 個になれば計算を打ち切る。推定値は候補ノードの集合に含まれるノードに対して計算対象となるグラフを計算するが、候補ノードは繰返し計算の中で動的に更新する。ここでは候補ノードと計算対象グラフの定義とその性質を示す。

閾値 ϵ_{i-1} を $i-1$ 番目の繰返し計算における k 番目に高い下限値とすると、 i 番目の繰返し計算における候補ノードの集合 $\mathbb{C}_i$ は以下のように定義される。

【定義 3】 (候補ノード) i 番目の繰返し計算における候補ノードの集合を以下のように計算する。

$$\mathbb{C}_i = \begin{cases} \mathbb{V} & (i=0) \\ \{u \in \mathbb{V} : \bar{p}_{i-1}[u] \geq \epsilon_{i-1}\} & (i \neq 0) \end{cases} \quad (5)$$

定義 3 から k 番目の PageRank の値を複数のノードが持つ場合、それらのノードはすべて集合 $\mathbb{C}_i$ に含まれることがわかる。集合 $\mathbb{C}_i$ の理論的性質は以下の通りである。

[補題 4] (候補ノード) もしあるノード u が集合 $\mathbb{C}_i$ に含まれなければ (すなわち $u \notin \mathbb{C}_i$ であれば), ノード u は解ノードになり得ない。

《証明》 ϵ を PageRank の k 番目に高いスコアとすると、補題 1 から明らかに $\epsilon_{i-1} \leq \epsilon$ 。また補題 2 から $\bar{p}_{i-1}[u] \geq p[u]$ 。 $\mathbb{A}$ を解ノードの集合とするともし $i \neq 0$ であれば

$$\mathbb{A} = \{u \in \mathbb{V} : p[u] \geq \epsilon\} \subseteq \{u \in \mathbb{V} : \bar{p}_{i-1}[u] \geq \epsilon_{i-1}\} = \mathbb{C}_i$$

である。また $i=0$ であれば $\mathbb{A} \subseteq \mathbb{V} = \mathbb{C}_i$ である。そのため $u \notin \mathbb{C}_i$ となるノード $u \in \mathbb{A}$ は存在しない。結果としても $u \notin \mathbb{C}_i$ であればノード u は解ノードになり得ない。□

補題 4 から $\mathbb{A} \subseteq \mathbb{C}_i$ であるため、各繰返し計算において集合 $\mathbb{C}_{i-1}$ から集合 $\mathbb{C}_i$ を以下のように逐次的に計算する。

$$\mathbb{C}_i = \{u \in \mathbb{C}_{i-1} : \bar{p}_{i-1}[u] \geq \epsilon_{i-1}\} \quad (6)$$

この式から明らかに各繰返し計算において候補ノードの集合は単調減少することがわかる。すなわち $\mathbb{C}_i \subseteq \mathbb{C}_{i-1}$ となる。

計算対象グラフより候補ノードに対して推定値を計算する。 i 番目の繰返し計算における計算対象グラフは以下のように定義する。

【定義 4】 (計算対象グラフ) $\mathbb{G}_i = (\mathbb{V}_i, \mathbb{E}_i)$ を i 番目の繰返し計算における計算対象グラフとする。もし $i=0$ であれば $\mathbb{V}_0$ と $\mathbb{E}_0$ はそれぞれ $\mathbb{V}$ と $\mathbb{E}$ とする。もし $i \neq 0$ であれば $\mathbb{V}_i$ と $\mathbb{E}_i$ はそれぞれ $\mathbb{V}_i = \mathbb{R}_i$ と $\mathbb{E}_i = \{(u, v) \in \mathbb{E} : u \in \mathbb{R}_i, v \in \mathbb{R}_i\}$ とする。ここで (u, v) はノード u からノード v へのエッジである。

計算対象グラフについての以下の補題を示す。

[補題 5] (計算対象グラフ) i 番目の繰返し計算における候補ノードに対する推定値は計算対象グラフ $\mathbb{G}_i$ から計算できる。

《証明》もし $i = 0$ であれば計算対象グラフ G_i はグラフ G と等しいため成り立つ．そうでなければ定義 1 と 2 からもしノード u のランダムウォークの確率からノード u の推定値は計算できる．もしノード v がノード u へ到達できなければ、ノード v のランダムウォークの確率はノード u のランダムウォークの確率に影響しない．そのためノード集合 R_i とその集合におけるエッジの集合が推定値を求めるために必要となる．□

[補題 6] (計算対象グラフ G_i の単調減少) 繰返し計算において計算対象グラフは $G_i \subseteq G_{i-1}$ となる性質がある．すなわちグラフ G_i はグラフ G_{i-1} の部分グラフとなる．

《証明》(1) 集合 R_i は集合 C_i に到達可能なノードの集合であり、(2) $C_i \subseteq C_{i-1}$ であるため、明らかに $R_i \subseteq R_{i-1}$ である．よって定義 4 から $G_i \subseteq G_{i-1}$ となる．□

補題 6 に基づき計算対象グラフを構築する方法は後に示す．

繰返し計算において計算対象グラフを用いて逐次的に推定値を以下のように計算する．

【定義 5】(逐次的な推定値の計算) 下限値と上限値を逐次的に以下のように計算する．

$$p_i[u] = \begin{cases} (1-s)/N & (i=0) \\ p_{i-1}[u] + (1-s)s^i r_i[u] & (i \neq 0) \end{cases} \quad (7)$$

$$\bar{p}_i[u] = \begin{cases} 1/N + s(1-s)^{-1}\bar{W}[u] & (i=0) \\ p_{i-1}[u] + s^i r_i[u] + \Delta_i \sigma_i \bar{W}[u] & (i \neq 0) \end{cases} \quad (8)$$

ここでもし $i \neq 0$ であれば確率 $r_i[u]$ は計算対象グラフ G_i から $r_i[u] = \sum_{v \in V_i} W[u, v] r_{i-1}[v]$ と計算し、そうでなければ $r_0 = e$ とする．

[補題 7] (逐次的な推定値の計算) もし $v \in V_i$ であるノード v に対してランダムウォークの確率が得られていれば、 $u \in C_i$ であるノード u に対して定義 5 から推定値は $O(1)$ の計算コストで計算できる．

《証明》 $i = 0$ であれば定義 1 と 2 から $p_i[u] = (1-s) \sum_{j=0}^0 s^j r_j[u] = (1-s)/N$ となる．また $i = 0$ のとき $\bar{p}_i[u] = (1-s) \sum_{j=0}^0 s^j r_j[u] + s r_0[u] + \Delta_0 \sigma_0 \bar{W}[u]$ であるため、 $\bar{p}_i[u] = 1/N + s(1-s)^{-1}\bar{W}[u]$ である．ここで s と N と $\bar{W}[u]$ は定数であるため、もし $i = 0$ であれば $p_i[u]$ と $\bar{p}_i[u]$ は $O(1)$ の計算コストで計算できる．また $i \neq 0$ であれば定義 1 と 2 から $p_i[u] - p_{i-1}[u] = (1-s) \sum_{j=0}^i s^j r_j[u] - (1-s) \sum_{j=0}^{i-1} s^j r_j[u] = (1-s)s^i r_i[u]$ である．またこのとき $\bar{p}_i[u] - \bar{p}_{i-1}[u] = (1-s) \sum_{j=0}^i s^j r_j[u] + s^{i+1} r_i[u] + \Delta_i \sigma_i \bar{W}[u] - (1-s) \sum_{j=0}^{i-1} s^j r_j[u] = s^i r_i[u] + \Delta_i \sigma_i \bar{W}[u]$ となる．ここで $p_{i-1}[u]$ は $p_i[u]$ と $\bar{p}_i[u]$ を計算する前にすでに計算されており、また s と $r_i[u]$ と σ_i と $\bar{W}[u]$ は定数であるため、もし $i \neq 0$ であれば $p_i[u]$ と $\bar{p}_i[u]$ は $O(1)$ の計算コストで計算できる．よって成り立つ．□

Algorithm 1 F-Rank

Input: G , オリジナルのグラフ; k , 解ノードの数

Output: 解ノードの集合

```

1:  $i := 0$ ;
2:  $C_0 := V$ ;
3:  $G_0 := G$ ;
4: repeat
5:   if  $i \neq 0$  then
6:      $i := i + 1$ ;
7:     幅優先探索を用いてグラフ  $G_{i-1}$  のノード集合  $C_i$  に対するノード集合  $R_i$  を計算;
8:     計算対象グラフ  $G_i$  をノード集合  $R_i$  から計算;
9:   end if
10:  for  $u \in V_i$  となるノードに対して do
11:    計算対象グラフ  $G_i$  から確率  $r_i[u]$  を計算;
12:  end for
13:  for  $u \in C_i$  となるノードに対して do
14:    式 (7) と (8) から推定値  $p_i[u]$  と  $\bar{p}_i[u]$  を計算;
15:  end for
16:  候補ノード  $C_i$  から閾値  $\epsilon_i$  を計算;
17:  式 (6) を用いて  $\epsilon_i$  と  $C_i$  から  $C_{i+1}$  を計算;
18: until  $|C_{i+1}| = k$ 
19: return  $C_{i+1}$ ;

```

3.4 検索アルゴリズム

Algorithm 1 に上位 k 個のノードを検索するアルゴリズムを示す．もし $i = 0$ であれば定義 3 と 4 より集合 C_0 とグラフ G_0 をそれぞれ $C_0 = V$ と $G_0 = G$ として初期化する (2 ~ 3 行目)．そうでなければグラフ G_{i-1} に幅優先探索を用いて集合 C_i から集合 R_i を計算する (7 行目)．これは $G_0 = G$ に初期化しているため、 $i = 1$ において R_i を正しく計算でき、またそのため $i > 1$ において補題 6 とその証明から $G_i \subseteq G_{i-1}$ 及び $R_i \subseteq R_{i-1}$ という性質があるため、 G_{i-1} から R_i を正しく計算できるからである．そして定義 4 から集合 R_i を用いて計算対象グラフ G_i を計算する (8 行目)．計算対象グラフ G_i における各ノードに対してランダムウォークの確率を計算するが (10 ~ 12 行目)、これは補題 5 からこのランダムウォークの確率が推定値を計算するために必要だからである．そして候補ノード C_i に対して推定値を計算し (13 ~ 15 行目)、候補ノード C_i から閾値 ϵ_i を計算する (16 行目)．また候補ノードを更新し C_{i+1} を計算する (17 行目)．もし集合 C_{i+1} の大きさが k であれば (すなわち $|C_{i+1}| = k$ であれば)、補題 4 から候補ノードの集合 C_{i+1} に含まれるノードは全て解ノードである．そのため繰返し計算を打ち切り (18 行目)、候補ノードの集合 C_{i+1} を解ノードとして出力する (19 行目)．

3.2 節で述べたとおり、PageRank の値が同じノードにおいて下限値と上限値は同じ値になる．そのため Algorithm 1 において下限値が閾値 ϵ_i と同じ値をもつノードが複数あり、かつそれら上限値がすべて同じ値であればそれらのノードをすべて解ノードとして出力する．

Algorithm 1 にあるとおり、提案手法は検索における事前計算を必要としない．すなわち提案手法はアドホックに検索を行うことができる．また提案手法はユーザに内

図1 検索時間

図2 ランダムウォーク回数と適合率

図3 ランダムウォーク回数と検索時間

部パラメータの設定を求めることはない．そのためユーザは簡単に PageRank による検索を行うことができる．

提案手法の理論的解析を示す．以下の定理は提案手法が正確に検索を行うことを示す．

[定理1] (検索の正確性) 提案手法は PageRank のスコアが上位 k 個のノードを正確に計算する．

《証明》 i 番目の繰り返し計算においても $\bar{p}_i[u] < \epsilon_i$ であればノード u を枝刈りする．補題1より $\epsilon_i \leq \epsilon$ であり，また補題2より $\bar{p}_i[u] \geq p_i[u]$ であるため，提案手法により解ノードが枝刈りされることはない．もしノード u が解ノードでなければ，補題2からすくなくともある繰り返し計算において $\bar{p}_i[u] < \epsilon$ となる．そのためノード u はその上限値から枝刈りされる．そのため提案手法における検索結果はオリジナルの手法による検索結果と等しくなる．□

次に提案手法における計算コストを示す． n と m をそれぞれ繰り返し計算における計算対象グラフの平均のノード数とエッジ数とする．また c と t を繰り返し計算における候補ノードの平均個数と繰り返し計算回数とする．ここで明らかに $c \leq n$ である．なおオリジナルの手法の計算コストは $O((N+M)T)$ である．

[定理2] (計算コスト) 提案手法で検索を行うのに必要となる計算コストは $O((n+m+\log c \cdot \log k)t)$ である．

《証明》 提案手法はまず幅優先探索を用いて $O((n+m)t)$ の計算コストで計算対象グラフを構築する．そして計算対象グラフの各ノードに対してランダムウォークの確率を $O((n+m)t)$ の計算コストで計算する．補題7から各ノードの推定値は $O(1)$ の計算コストで得られるため，候補ノードの推定値は $O(ct)$ の計算コストで計算できる．各繰り返し計算において下限値を用いて候補ノードから閾値 ϵ_i を計算するが，これには $O(\log c \cdot \log k)$ の計算コストが必要となる．これは(1)もし候補ノードから新たに k 番目のノードが得られればフィボナッチヒープを用いて k 番目の下限値を $O(\log k)$ の計算コストで更新でき(2)候補ノードにランダムにアクセスすることで更新の平均回数は $O(\log c)$ となるからである．閾値 ϵ_i と下限値を用いて候補ノード C_i から更新後の候補ノード C_{i+1} を $O(ct)$ の計算コストで得られる．結果として提案手法に必要な計算コストは $O((n+m+\log c \cdot \log k)t)$ となる．□

表1 それぞれのパラメータの値

パラメータ	データセット		
	P2P	Web	Wikipedia
N	6.26×10^4	3.26×10^5	2.39×10^6
c	3.16×10^4	1.49×10^5	4.00×10^5
n	4.69×10^4	2.70×10^5	6.29×10^5
M	1.48×10^5	3.22×10^6	5.02×10^6
m	1.20×10^5	3.06×10^6	2.44×10^6
T	18	116	97
t	9	33	21

4. 評価実験

提案手法の有効性を確認するために評価実験を行った．実験では P2P^{*1}，Web^{*2}，Wikipedia^{*3} の3つのデータを用いた．P2P は Gunutella におけるネットワークでありノード数は 62,586 でありエッジ数は 147,892 である．Web はイタリアにおける CNR ドメインにおけるウェブのネットワークであり，ノード数とエッジ数はそれぞれ 325,557 と 3,216,152 である．Wikipedia は Wikipedia に登録されたユーザ間のネットワークでありノード数は 2,394,385 でありエッジ数は 5,021,410 である．PageRank におけるパラメータは過去の論文 [Page 99] と同様に $s = 0.85$ とした．実験は CPU が Intel Xeon 3.33GHz の Linux サーバで行った．

4.1 検索時間

提案手法とオリジナルの手法の検索時間を調べた．図1に結果を示す．この図において“F-Rank(k)”は F-Rank において解の個数を k としたときの結果を示す．オリジナルの手法では更新後におけるページランクの差分が 10^{-10} 以下になるまで繰り返し計算を行った [Langville 05]．なおオリジナルの手法はすべてのノードに対してページランクを計算するため，解の個数 k の値は計算時間に影響ない．また表1に $k = 50$ のときの各手法におけるそれぞれのパラメータの値を示す．なおこれらの値は与えられたグラフに対して自動的に決定される．

図1から提案手法はオリジナルの手法より大幅に高速であることが分かる．これはオリジナルの手法の計算量が $O((N+M)T)$ であるのに対して，提案手法の計算量が $O((n+m+\log c \cdot \log k)t)$ であり(定理2)，大幅に

*1 <http://snap.stanford.edu/data/p2p-Gnutella31.html>
 *2 <http://law.di.unimi.it/webdata/cnr-2000/>
 *3 <http://snap.stanford.edu/data/wiki-Talk.html>

低減されているからである．表 1 に示すとおり，提案手法における計算対象グラフは与えられたグラフより小さく，また繰り返し計算回数もオリジナルの手法より少ない．

4.2 正 確 性

提案手法の大きな利点の一つとして，オリジナルの手法と同じ検索結果を得られることが挙げられる．この利点を示すために，提案手法を PageRank の近似計算手法の一つである“MC complete path stopping in dangling nodes”と比較を行った．この手法は Avrachenkov らによって提案されたものである [Avrachenkov 07]．この手法はモンテカルロ法を用いて PageRank を近似するもので，具体的にはランダムウォークを複数回試行し，各ノードごとのランダムウォークがたどった回数に基づき PageRank を近似する．この手法はランダムウォークの回数が増えるほど近似の精度が向上するため，実験ではランダムウォークの回数を変えて精度と速度を調べた．図 2 および図 3 に精度および速度の結果を示す．なおこの実験においてデータセットは P2P とし， $k = 50$ とした．図 2 において，精度の評価にはオリジナルの手法による解に対するそれぞれの手法による解の適合率を用いた．

図 2 から提案手法の適合率は 1 であることが分かる．これは提案手法が理論的に正確に検索できるからである (定理 1)．また図 2 から従来の近似手法はランダムウォークの回数が増えるほど精度が向上することが分かる．しかし図 3 からランダムウォークの回数が増えるほど検索時間が長くなってしまふことが分かる．これらの図から提案手法は既存の近似手法に対して速度においても精度においても優れていることが分かる．

5. ま と め

本論文では PageRank に対して高速かつ正確に上位 k 個のノードを検索する手法を提案した．提案手法は PageRank の下限値と上限値を推定し，動的に計算対象となるグラフを構築することで高速な検索を行う．実データを用いて提案手法と既存手法を比較したところ，提案手法はより高速に上位 k 個のノードを検索できることを確認した．

◇ 参 考 文 献 ◇

- [Avrachenkov 07] Avrachenkov, K., Litvak, N., Nemirovsky, D., and Osipova, N.: Monte Carlo Methods in PageRank Computation: When One Iteration is Sufficient, *SIAM J. Numerical Analysis*, Vol. 45, No. 2, pp. 890–904 (2007)
- [Bahmani 12] Bahmani, B., Kumar, R., Mahdian, M., and Upfal, E.: PageRank on an evolving graph, in *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12–16, 2012*, pp. 24–32 (2012)
- [Fujiwara 13] Fujiwara, Y., Nakatsuji, M., Shiokawa, H., Mishima, T., and Onizuka, M.: Fast and Exact Top-k Algorithm for PageRank, in *Proceedings of the Twenty-Seventh AAAI Conference on Artificial*

- Intelligence, July 14–18, 2013, Bellevue, Washington, USA*. (2013)
- [Kamvar 04] Kamvar, S., Haveliwala, T., and Golub, G.: Adaptive methods for the computation of PageRank, *Linear Algebra and its Applications*, Vol. 386, No. 0, pp. 51 – 65 (2004), Special Issue on the Conference on the Numerical Solution of Markov Chains 2003
- [Langville 05] Langville, A. N. and Meyer, C. D.: Updating Markov Chains with an Eye on Google's PageRank, *SIAM J. Matrix Anal. Appl.*, Vol. 27, No. 4, pp. 968–987 (2005)
- [Page 99] Page, L., Brin, S., Motwani, R., and Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web., Technical report, Stanford InfoLab (1999)

〔担当委員：瀧川 一学〕

2014 年 8 月 28 日 受理

著 者 紹 介

藤原 靖宏

2003 年早稲田大学大学院理工学研究科電気工学専攻修士課程修了．同年，日本電信電話株式会社入社．2011 年東京大学大学院情報理工学系研究科電子情報学専攻博士課程修了，2014 年ニューヨーク大学客員研究員．現在，日本電信電話株式会社 NTT ソフトウェアイノベーションセンタ特別研究員・博士（情報理工学）．グラフマイニングの研究開発に従事．第 27 回テレコムシステム技術賞，第 9 回上林奨励賞など受賞．情報処理学会，電子情報通信学会，日本データベース学会各会員．

中辻 真 (正会員)

2001 年京大工学部数理工学科卒業．2003 年京大大学院情報学研究科システム科学専攻修士課程修了．2003 年日本電信電話株式会社入社．2010 年京大大学院情報学研究科社会情報学専攻博士課程修了（情報学）．2013 年から米国レンセラー工科大客員研究員．日本電信電話株式会社サービスエボリューション研究所所属．Semantic Web，情報推薦の研究に従事．電子情報通信学会会員．

塩川 浩昭

2009 年筑波大学第三学群情報学類卒業．2011 年同大学院システム情報工学研究科博士前期課程修了．同年，日本電信電話株式会社入社．現在，日本電信電話株式会社 NTT ソフトウェアイノベーションセンタ研究員，および筑波大学大学院システム情報工学研究科博士後期課程在籍．大規模データ分析，分散並列処理の研究開発に従事．2013 年 DEIMForum 2013 最優秀論文賞および優秀論文賞，2014 年 DEIM Forum 2014 優秀論文賞，日本データベース学会平成 25 年度論文賞受賞．日本データベース学会会員．

三島 健

1994 筑波大・第三学群・情報学類卒．1996 同大学院工学研究科修士課程修了．同年，NTT 入社．電話交換システムの開発に従事．2010 東大工学研究科博士課程修了．博士（工学）．2010 年上林奨励賞受賞．日本データベース学会，情報処理学会，各会員．現在，データベースシステム，クラウドコンピューティングの研究に従事．

鬼塚 真

1991 年東京工業大学工学部情報工学科卒業．同年，日本電信電話株式会社入社．2000 ～ 01 年ワシントン大学客員研究員，2010 ～ 14 年日本電信電話株式会社特別研究員，2012 ～ 14 年電気通信大学客員教授，現在，大阪大学大学院情報科学研究科教授．博士（工学）．大規模グラフデータの分散データ処理に関する研究開発に取り組んでいる．2004 年情報処理学会 山下記念賞，2008 年データベース学会上林奨励賞など受賞．情報処理学会，電子情報通信学会，日本データベース学会，ACM 各会員．