2E5-OS-25b-3

PageRank のための高速な検索手法

藤原 靖宏[*1] 中辻 真[*2] 塩川 浩昭[*1] 三島 健[*1] 鬼塚 真[*1]
Yasuhiro Fujiwara Makoto Nakatsuji Hiroaki Shiokawa Takeshi Mishima Makoto Onizuka

[*1]NTT ソフトウェアイノベーションセンタ NTT Software Innovation Center
[*2]NTT サービスエボリューション研究所 NTT Service Evolution Laboratories

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 by using 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ために用いられる最も有名な手法である [Page 99]．しかし PageRank の問題点として計算コストが高いことが挙げられる．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返し計算を行わなければならないためである．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．

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．提案手法と特徴として以下のものが挙げられる．

- **高速:** 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:** 提案手法は正確に上位 k 個のノードを検索可能．
- **高い柔軟性:** 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:** 提案手法に必要となる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．

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．また各ステ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 u 番目の要素 $p[u]$ がノード u の PageRank のスコアに対応する列ベクトルとする．また N をグラフのノード数としたときに，$\mathbf{e}$ を全ての要素の値が $1/N$ 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，$\mathbf{W}$ を列要素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 $\mathbf{e}$ に設定される．この繰返し計算を行うオリジナル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．

連絡先: 藤原靖宏，日本電信電話株式会社，〒 180-8585 東京都武蔵野市緑町 3-9-11，fujiwara.yasuhiro@lab.ntt.co.jp

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 のスコアの下限値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，部分グラフ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部分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部分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ができる．また提案手法に必要となる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 $(i = 0, 1, 2, \ldots)$ 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 u 番目の要素がエッジの最大の重みから $\overline{W}[u] = \max\{W[u, v] : v \in \mathbb{V}\}$ となる $N \times 1$ の列ベクトルを $\overline{\mathbf{W}}$ 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 $\mathbf{W}$ 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 番目の繰り返し計算における下限値 $\underline{\mathbf{p}}_i$ と上限値 $\overline{\mathbf{p}}_i$ を以下のように定義する．

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

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

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

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

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

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

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

補助定理 1 (下限値) i 番目の繰り返し計算においてベクトル $\mathbf{p}$ と $\underline{\mathbf{p}}_i$ の u 番目の要素において $\underline{p}_i[u] \leq p[u]$ が成り立つ．

証明 式 (1) から

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

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 $\mathbf{W}^\infty$ の要素は 0 から 1 であるため

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

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．

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

補助定理 2 (上限値) i 番目の繰り返し計算において $\overline{p}_i[u] \geq p[u]$ がベクトル $\mathbf{p}$ と $\overline{\mathbf{p}}_i$ に対して成り立つ．

証明 上記の証明にあるとおり

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

となる．まず $r_{i+j}[u] \leq r_i[u] + j\Delta_i\overline{W}[u]$ 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

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

となる．$\mathbf{W}^{j-1}$ 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

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

となる．よって

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

となる．この性質を用いて

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

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

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

となる．そのため式 (3) より

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

となる．よって成り立つ． $\square$

補助定理 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 $\underline{p}_\infty[u] = \overline{p}_\infty[u] = p[u]$ となる．

証明 紙幅の都合により省略． $\square$

補助定理 3 は提案手法が収束することを示す．

3.3 部分グラフの構築

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部分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部分グラフの定義とその性質を示す．

閾値 ϵ_{i-1} を $i-1$ 番目の繰り返し計算おける k 番目に高い下限値とすると，i 番目の繰り返し計算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 番目の繰り返し計算おける候補ノードの集合を以下のように計算する．

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

集合 $\mathbb{C}_i$ の理論的性質は以下の通りである．

補助定理 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，補助定理 1 から明らかに $\epsilon_{i-1} \leq \epsilon$．また補助定理 2 から $\overline{p}_{i-1}[u] \geq p[u]$．$\mathbb{A}$ を解ノードの集合とするともし $i \neq 0$ であれば

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

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 $u \notin \mathbb{C}_i$ となるノードは存在しない．結果としてもし $u \notin \mathbb{C}_i$ であればノード u は解ノードになり得ない． $\square$

補助定理 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できる．

定義 4 (候補ノードの更新) もし $i \neq 0$ であれば各繰り返し計算において集合 $\mathbb{C}_i$ を以下のように逐次的に計算する．

$$\mathbb{C}_i = \{u \in \mathbb{C}_{i-1} : \overline{p}_{i-1}[u] \geq \epsilon_{i-1}\} \tag{6}$$

補助定理 5 (候補ノードの更新) 各繰り返し計算において候補ノードの集合は単調減少する．すなわち $\mathbb{C}_i \subseteq \mathbb{C}_{i-1}$ である．

証明 式 (6) において集合 $\mathbb{C}_i$ は集合 $\mathbb{C}_{i-1}$ の部分集合として得られるため，$\mathbb{C}_i \subseteq \mathbb{C}_{i-1}$ であることは明らかである． □

部分グラフより候補ノードに対して推定値を計算する．i 番目の繰り返し計算おける部分グラフは以下のように定義する．

定義 5 (部分グラフ) $\mathbb{G}_i = \{\mathbb{V}_i, \mathbb{E}_i\}$ を i 番目の繰り返し計算おける部分グラフとする．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．

部分グラフについての以下の補助定理を示す．

補助定理 6 (部分グラフ) i 番目の繰り返し計算おける候補ノードに対する推定値は部分グラフ $\mathbb{G}_i$ から計算できる．

証明 もし $i = 0$ であれば部分グラフ $\mathbb{G}_i$ はグラフ $\mathbb{G}$ 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 $\mathbb{R}_i$ 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． □

補助定理 7 (部分グラフ $\mathbb{G}_i$ の単調減少) 繰返し計算において部分グラフは $\mathbb{G}_i \subseteq \mathbb{G}_{i-1}$ となる性質がある．

証明 （1）集合 $\mathbb{R}_i$ は集合 $\mathbb{C}_i$ に到達可能なノードの集合であり，（2）補助定理 5 から $\mathbb{C}_i \subseteq \mathbb{C}_{i-1}$ であるため，明らかに $\mathbb{R}_i \subseteq \mathbb{R}_{i-1}$ である．よって定義 5 から $\mathbb{G}_i \subseteq \mathbb{G}_{i-1}$ となる． □

補助定理 7 に基づき部分グラフを構築する方法は後に示す．

繰返し計算において部分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 6 (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．

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

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

ここでもしもし $i \neq 0$ であれば確率 $r_i[u]$ は部分グラフ $\mathbb{G}_i$ から $r_i[u] = \sum_{v \in \mathbb{V}_i} W[u, v] r_{i-1}[v]$ と計算し，そうでなければ $\mathbf{r}_0 = \mathbf{e}$ とする．

補助定理 8 (逐次的な推定値の計算) もし $v \in \mathbb{V}_i$ 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，$u \in \mathbb{C}_i$ であるノード u に対して定義 6 から推定値は $O(1)$ の計算コストで計算できる．

証明 紙幅の都合により省略． □

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 5 より集合 $\mathbb{C}_0$ とグラフ $\mathbb{G}_0$ をそれぞれ $\mathbb{C}_0 = \mathbb{V}$ と $\mathbb{G}_0 = \mathbb{G}$ として初期化する（2 ～ 3 行目）．そうでなければグラフ $\mathbb{G}_{i-1}$ に幅優先探索を用いて集合 $\mathbb{C}_i$ から集合 $\mathbb{R}_i$ を計算する（7 行目）．これは補助定理 7 からグラフ $\mathbb{G}_i$ に対して $\mathbb{G}_i \subseteq \mathbb{G}_{i-1}$ という性質があるからである．そして定義 5 から集合 $\mathbb{R}_i$ を用いて部分グラフ $\mathbb{G}_i$ を計算する（8 行目）．部分グラフ $\mathbb{G}_i$ 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），これは補助定理 6 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 $\mathbb{C}_i$ に対して推定値を計算し（13 ～ 15 行目），候補ノード $\mathbb{C}_i$ 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 $\mathbb{C}_{i+1}$ を計算する（17 行目）．もし集合 $\mathbb{C}_{i+1}$ の大きさが k であれば（すなわち $|\mathbb{C}_{i+1}| = k$ であれば），補助定理 4 から候補ノードの集合 $\mathbb{C}_{i+1}$ 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 $\mathbb{C}_{i+1}$ を解ノードとして出力する（19 行目）．

Algorithm 1 F-Rank

Input: $\mathbb{G}$, 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: $\mathbb{C}_0 := \mathbb{V}$;
3: $\mathbb{G}_0 := \mathbb{G}$;
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 $\mathbb{G}_{i-1}$ のノード集合 $\mathbb{C}_i$ に対するノード集合 $\mathbb{R}_i$ を計算;
8: 部分グラフ $\mathbb{G}_i$ をノード集合 $\mathbb{R}_i$ から計算;
9: **end if**
10: **for** $u \in \mathbb{V}_i$ となるノードに対して **do**
11: 部分グラフ $\mathbb{G}_i$ から確率 $r_i[u]$ を計算;
12: **end for**
13: **for** $u \in \mathbb{C}_i$ となるノードに対して **do**
14: 式 (7) と (8) から推定値 $\underline{p}_i[u]$ と $\overline{p}_i[u]$ を計算;
15: **end for**
16: 候補ノード $\mathbb{C}_i$ から閾値 ϵ_i を計算;
17: 式 (6) を用いて ϵ_i と $\mathbb{C}_i$ から $\mathbb{C}_{i+1}$ を計算;
18: **until** $|\mathbb{C}_{i+1}| = k$
19: **return** $\mathbb{C}_{i+1}$;

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易に 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し $\overline{p}_i[u] < \epsilon_i$ であればノード u を枝刈りする．補助定理 1 より $\epsilon_i \leq \epsilon$ であり，また補助定理 2 より $\overline{p}_i[u] \geq p_i[u]$ 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，補助定理 2 からすくなくともある繰返し計算において $\overline{p}_i[u] < \epsilon$ 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 をそれぞれ繰返し計算における部分グラフの平均のノード数とエッジ数とする．また c と t を繰返し計算における候補ノードの平均個数と繰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 $O((n + m + \log c \log k)t)$ である．

証明 提案手法はまず幅優先探索を用いて $O((n+m)t)$ の計算コストで部分グラフを構築する．そして部分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．補助定理 8 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 $O(\log c \log k)$ 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

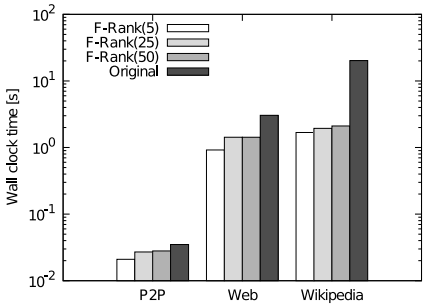

図 1: 検索時間

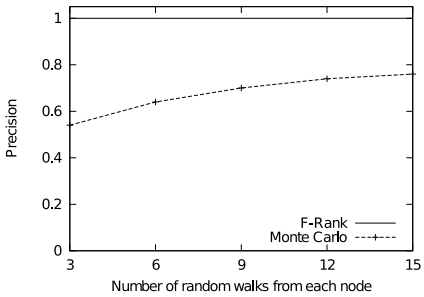

図 2: ランダムウォーク回数と適合率

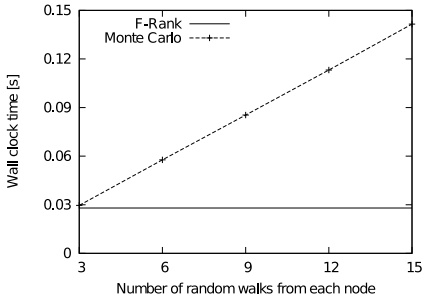

図 3: ランダムウォーク回数と検索時間

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

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 $\mathbb{C}_i$ から更新後の候補ノード $\mathbb{C}_{i+1}$ 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 $O((n+m+\log c \log k)t)$ となる． □

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 06]．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 $O((n+m+\log c \log k)t)$ であり（定理 2），大幅に低減されているからである．表 1 に示すとおり，提案手法における部分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 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うことが分かる．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，動的に部分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* (2007)

[Fujiwara 13] Fujiwara, Y., Nakatsuji, M., Shiokawa, H., Mishima, T., and Onizuka, M.: Fast and Exact Top-k Algorithm for PageRank, in *AAAI* (2013)

[Langville 06] Langville, A. N. and Meyer, C. D.: Updating Markov Chains with an Eye on Google's PageRank, *SIAM J. Matrix Analysis Applications* (2006)

[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)

*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