DEIM Forum 2019 D2-4

プライベートなノードを含むソーシャルネットワークの統計量推定

中嶋　一貴†　首藤　一幸†

† 東京工業大学 情報理工学院 数理・計算科学系

あらまし　本研究では，プライベートなノードを含むソーシャルネットワークに対する，ランダムウォークを用いた統計量推定アルゴリズムを議論する．近年では，ランダムウォークを用いた，ソーシャルネットワークの統計量の効率的な推定アルゴリズムが多く提案されている．現実のソーシャルネットワークは隣接ノードが非公開であるプライベートなノードが一定の割合で存在するが，既存の推定アルゴリズムはこれらの存在を考慮していない．特に，平均次数とノード数に対する既存の推定アルゴリズムは，グラフ内のプライベートなノードの割合 p が増加するにつれて，推定値の誤差が増加する問題点がある．本論文では p にほぼ依存しない，グラフの平均次数とノード数の推定アルゴリズムをそれぞれ提案する．我々はサンプルされた各ノードに既存の推定アルゴリズムと異なる重みをかけることで，p に対して頑健な推定アルゴリズムを実現する．実験では，提案アルゴリズムが p にほぼ依存せずに，高い精度で平均次数とノード数を推定できることを示す．

キーワード　ソーシャルネットワーク，グラフサンプリング，ランダムウォーク，プライバシ

1 はじめに

近年，Online Social Networks (OSNs) の解析を目的とする研究が盛んに行われている．OSNs の解析は平均次数やノード数などのグラフの統計量や，中心性などのノードの統計量を推定することである [1–4]．しかし，ほとんどの OSN はそのグラフデータへのアクセスを制限しており，推定することは容易ではない．巨大な OSN の全てのグラフデータを取得して統計量を厳密計算することはほぼ不可能であり，また事前にノードの分布を知ることもできないためランダムサンプリングに基づく推定アルゴリズムを適用することも難しい [5,6]．

この問題を打開するために，ランダムウォークを用いた統計量推定アルゴリズムが多く提案されてきた [7–14]．多くの OSN のアプリケーション・プログラミング・インターフェース (API) は，クエリを実行したノードの隣接ノード情報を提供する．API から提供された隣接ノードのインデックスをランダムに 1 つ選び遷移することを繰り返すことで，OSN 上でランダムウォークを実行してインデックスなどのグラフデータのサンプリングが可能である．また，ランダムウォークのマルコフ性から各ノードのサンプルの分布を計算できるため，サンプルノードに適切な重み付けを行うことで不偏な推定量を求めることができる [6,4,15]．既存の統計量推定アルゴリズムは，ごく限られた API へのクエリ回数で統計量を高精度に推定することを目標として設計されてきた．

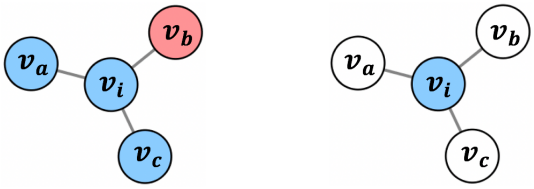
OSN に既存の統計量推定アルゴリズムを適用する上で，ユーザのプライバシ保護によるランダムウォークの制限は無視できない問題である．ほとんどの OSN では，各ユーザは友人リストの公開・非公開を設定することができる．Dey らは，ニューヨークの Facebook ユーザの 52.6%が友人リストを非公開にしていたと報告した [16]．本論文では隣接ノード集合を公開・非公開にするノードをそれぞれパブリック・プライベートなノードと呼ぶ．ランダムウォークでプライベートなノードに遷移した場合は，API がその隣接ノードを提供しないためアルゴリズムを続行できない．

Gjoka らは，実際に Facebook 上でランダムウォークを実行したとき，パブリックな隣接ノードに遷移する対処をとった [4]．このランダムウォークは，グラフ全体ではなく，グラフ内のパブリックなノードのみから成る連結な部分グラフ上で実行されている．このため，Gjoka らのランダムウォークを既存の統計量推定アルゴリズムに適用すると，その部分グラフの統計量を推定することになる．

平均次数とノード数に対する既存の推定アルゴリズムは，プライベートなノードの割合 p が増加するにつれて推定値の誤差が大きくなる問題点がある．これは，ランダムウォークが実行される部分グラフの平均次数とノード数が，p が増加するにつれて真値との誤差が大きくなるためである．

本研究の貢献は，p に対して頑健な，平均次数とノード数の推定アルゴリズムを提案することである．まず，Gjoka らのランダムウォーク [4] の定常分布を導出し，各ノードがそのパブリックな隣接ノード数に比例した分布でサンプルされることを示す．各ノードのサンプルの分布を計算することで，各サンプルノードへの適切な重み付けを議論することが可能になる．そして我々は，各サンプルノードに既存の推定アルゴリズムと異なる重みをかけることで，p に対して頑健な推定を可能にする．本研究では，OSN の API が提供する隣接ノード情報の 2 つのモデルを考え，それぞれのモデルでクエリ回数を考慮した推定アルゴリズムを設計する．実際の OSN のグラフに対する実験では，いずれのモデルに対しても，提案アルゴリズムが p にほぼ依存せずに，ごく限られたクエリ回数で平均次数とノード数を高精度に推定できることを示す．

This is an unrefereed paper.



(a) API が隣接ノードのプライバシ情報を提供するモデル.

(b) API が隣接ノードのプライバシ情報を提供しないモデル.

図 1　パブリックなノード v_i にクエリを実行したとき，API が提供する隣接ノード情報の，2 つのモデル: 青色のノードはパブリック，赤色のノードはプライベート，白色のノードはプライバシ情報が不明なノードを示す.

2　準　　備

本研究では無向で連結なグラフ $G = (V, E)$ を扱う．ソーシャルネットワークでは，ノードはユーザ，エッジはユーザ間の関係を表す．グラフ G はセルフループと多重辺を持たず辺に重みはないとする．ノード数 $n = |V|$, エッジ数 $m = |E|$ とし，グラフ G のノード集合を $V = \{v_1, v_2, ..., v_n\}$, エッジ集合を E，ノード v_i の次数を d_i と表記する．グラフ G の平均次数は以下のように定義される:

定義 1. $d_{avg} = \frac{\sum_{v_i \in V} d_i}{n} = \frac{2m}{n}$.

研究対象とするネットワークはランダムウォークモデル [9] を満たすとする．事前にグラフの情報は持たず，OSN の API が提供する隣接ノード情報を辿ってサンプリングを行う．また，推定アルゴリズムを実行している間，グラフは静的であるとする.

グラフ G の各ノード v_i はパブリックまたはプライベートのいずれかのラベルを持つ．ノード v_i に対してクエリを実行したとき，v_i がパブリックであれば，その隣接ノード情報を取得でき，v_i がプライベートであれば，その隣接ノード情報は取得できないとする [1]．

本研究では，図 1 のように，OSN の API が提供する隣接ノード情報の 2 つのモデルを考える．Gjoka らが Facebook 上でランダムウォークを実行した事例は，図 1(a) のモデルに該当する [4]．しかし，図 1(b) のように，隣接ノードのインデックスのみ提供される場合も考えられる.

ランダムウォークを用いた推定アルゴリズムの目標は，少ないクエリ回数で OSN のグラフの統計量を推定することである [5,17]．一度クエリを実行したノードの隣接ノードの情報は保存する.

$\boldsymbol{P} = \{p_{ij}\}_{i,j \in S}$ を有限な状態空間 S におけるマルコフ連鎖の遷移確率行列とする．$\boldsymbol{P}$ の定常分布について以下の定理が成り立つ.

定理 1. *[18]*$\boldsymbol{P}$ がエルゴード的であるとき，その定常分布 $\boldsymbol{\pi}$ が一意に存在する.

1：出力はエラーメッセージや空集合など，OSN の API に依存する.

3　プライベートなノードを考慮したランダムウォーク

プライベートなノードを含むグラフ上のランダムウォークを議論する．ランダムウォークは，あるノードの隣接ノードからランダムに 1 つのノードを選択して遷移することを繰り返し，遷移したノードのインデックスをはじめとする情報をサンプリングする．プライベートなノードに遷移した場合はその隣接ノードの情報が取得できないため，ランダムウォークを続行することができない.

このため，Gjoka らが提案したパブリックな隣接ノードをランダムに選び遷移するランダムウォークが適切である [4]: ランダムウォークの i ステップ目のパブリックなサンプルノード v_{x_i} の隣接ノード集合からランダムに 1 つ選んだノード $v_{x_{i+1}}$ のプライバシ情報を確認し，$v_{x_{i+1}}$ がパブリックなノードであればそのノードに遷移し，プライベートなノードであれば v_{x_i} の隣接ノードを選択し直す．API が隣接ノードのプライバシ情報を提供しないモデルでは，$v_{x_{i+1}}$ にクエリを実行してその出力からプライバシ情報を判別する.

Gjoka らのランダムウォークは，元のグラフ G からプライベートなノードを削除してできるいくつか部分グラフのうち，初期ノードが属する部分グラフ上で実行される．グラフ G から全てのプライベートなノードとそれに接続するエッジを削除したあとに分割される k 個の連結な部分グラフを C_i とする．それぞれの部分グラフ C_i をクラスタと呼ぶ [19,20]．この中でノード数が最大のクラスタを $C^* = (V^*, E^*)$ とする．C^* に対して $n^* = |V^*|, m^* = |E^*|$ とおく．最大クラスタ C^* の平均次数を以下のように定義する:

定義 2. $d^*_{avg} = \frac{\sum_{v_i \in V^*} d^*_i}{n^*} = \frac{2m^*}{n^*}$.

本研究ではランダムウォークの初期ノードは最大クラスタに属するノードを選ぶとする．ほとんどの OSN はスケールフリーネットワークに分類され [1–4]，スケールフリーネットワークはランダムなノードの除去に対して頑健であることが知られている [19,20]. すなわち，グラフにプライベートなノードがランダムに分布していると仮定すると，ほとんどのパブリックなノードは最大クラスタに属される．このため，既存研究でも最大クラスタでサンプリングが可能であった [1–4].

3.1　最大クラスタ上のランダムウォーク

最大クラスタ上のランダムウォークでは，各ノードはそのパブリックな隣接ノード数に比例した分布でサンプルされる．これは最大クラスタ上のランダムウォークの定常分布を導出することでわかる．C^* 上のランダムウォークは有限な状態空間 V^* におけるマルコフ連鎖でありエルゴード性を満たす．よって，定理 1 より定常分布 π^* が一意に存在する．C^* における各ノード $v_i \in V^*$ の次数を d^*_i と表す．C^* の各ノードの次数は，そのノードの G におけるパブリックな隣接ノード数である．ここで V^* の各ノードの添字集合を $I = \{i_1, i_2, ..., i_{n^*}\}$ とおく．C^* 上のランダムウォークの定常分布を $\boldsymbol{\pi^*} = \{\pi^*_j\}_{1 \leqq j \leqq n^*}$ とする

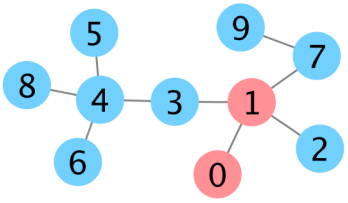


図 2　プライベートなノードを含むグラフの例．青色のノードはパブリック，赤色のノードはプライベートなノードである．

と，各ノード $v_{i_j} \in V^*$ に対して以下が成り立つ:

$$\pi^*_j = \frac{d^*_{i_j}}{2m^*}.$$

例えば図 2 のグラフで $v_i = i$ $(0 \leqq i \leqq 9)$ とする．以下の 3 つのクラスタ C_1, C_2, C_3 に分割され，最大クラスタは $C^* = C_1$ である:

- $C_1 = (\{3,4,5,6,8\}, \{(3,4),(4,5),(4,6),(4,8)\})$
- $C_2 = (\{7,9\}, \{(7,9)\})$
- $C_3 = (\{2\}, \{\})$.

添字集合 $I = \{i_1, i_2, i_3, i_4, i_5\}$ を $i_1 = 3, i_2 = 4, i_3 = 5, i_4 = 6, i_5 = 8$ とすると，$d^*_3 = 1, d^*_4 = 4, d^*_5 = 1, d^*_6 = 1, d^*_8 = 1$ である．よって C^* 上のランダムウォークの定常分布は $\boldsymbol{\pi}^* = (\frac{1}{8}, \frac{1}{2}, \frac{1}{8}, \frac{1}{8}, \frac{1}{8})$ である．

4　パブリックな隣接ノード数の計算

平均次数とノード数の推定において，各サンプルノード v_{x_i} に重み付けをするために，そのパブリックな隣接ノード数 $d^*_{x_i}$ を計算する必要がある．API が提供する隣接ノード情報のそれぞれのモデル (図 1) に対して，クエリ数を考慮した計算方法を述べる．

4.1　API が隣接ノードのプライバシ情報が提供するモデル

API が v_{x_i} の隣接ノードのプライバシ情報を提供する場合，$d^*_{x_i}$ を厳密に計算できる．実際に，Gjoka らが Facebook 上でランダムウォークを実行した事例では，このモデルに該当する [4].

4.2　API が隣接ノードのプライバシ情報が提供するモデル

この場合，$d^*_{x_i}$ の厳密値を計算するには v_{x_i} の全ての隣接ノードに対してクエリを実行する必要があり，現実的ではない．このため，ランダムウォークでサンプルした各ノードに対して，パブリックな隣接ノードを選択できた割合から近似計算する方法を提案する．各サンプルノード v_{x_i} に対して，パブリックな隣接ノードを選択した合計回数を a_{x_i}，隣接ノードを選択した合計回数を b_{x_i} とする．各サンプルノード v_{x_i} のパブリックな隣接ノード数 $d^*_{x_i}$ の近似値 $\hat{d}^*_{x_i}$ を以下のように定義する:

$$\hat{d}^*_{x_i} = \begin{cases} d_{x_i} \frac{a_{x_i}}{b_{x_i}} & (b_{x_i} \neq 0) \\ d_{x_i} & (otherwise) \end{cases}$$

定理 2. 近似値 $\hat{d}^*_{x_i}$ の期待値は真値 $d^*_{x_i}$ に等しい．

$$E[\hat{d}^*_{x_i}] = d^*_{x_i}$$

証明. i ステップ目のサンプルノードの隣接ノードを 1 つランダムに選んだとき，確率 $\frac{d^*_{x_i}}{d_{x_i}}$ でパブリックな隣接ノードを選択する．よって，$E[\hat{d}^*_{x_i}] = d_{x_i} \frac{d^*_{x_i}}{d_{x_i}} = d^*_{x_i}$ が成り立つ．□

5　プライベートなノードを考慮した統計量推定

本論文ではグラフ G の平均次数 d_{avg} とノード数 n を推定するアルゴリズムを議論する．グラフのノード数とエッジ数は最も基本的なグラフの統計量である．定義 1 よりノード数と平均次数の推定値からエッジ数の推定値を求めることができる．

既存の推定アルゴリズムは C^* の平均次数とノード数を推定しており，p が増加するにつれて推定値の誤差が増加する問題点がある．提案アルゴリズムは，既存アルゴリズムのサンプルノードへの重みを変更することで，p の増加に対する推定値の誤差の増加を大幅に小さくする．

5.1　平均次数の推定

Dasgupta らはグラフの平均次数を推定するアルゴリズム **Smooth** を提案した [9]．**Smooth** は，入力として平均次数の大まかな推定値 c を与え，それより正確な推定値 d^{Smooth}_{avg} を出力する．ランダムウォークで遷移した各ノードにセルフループを c 本追加して次の遷移先のノードを選択する．

本研究ではグラフの事前知識が無いことを仮定するため $c = 0$ とする．入力として与える定数 c は平均次数に対する大まかな近似値とされるが，グラフに対する事前知識が無い場合はその近似値を求めることにクエリの追加実行を必要とする．Dasgupta らは定数 c=0, 1, 5, 50 に対して推定精度の評価実験をしており，0 や 1 の小さい値が望ましいとしている [9]．特に $c = 0$ の場合は Gjoka らが Facebook の平均次数を推定したアルゴリズムと一致する [4].

C^* 上で *Smooth* を適用すれば推定値は以下のように計算される:

$$d^{Smooth}_{avg} = \frac{r}{\sum_{i=1}^{r} \frac{1}{d^*_{x_i}}}$$

定理 3. 推定値 d^{Smooth}_{avg} は d^*_{avg} に収束する．

証明. R は有限な状態空間 V^* におけるエルゴード的なマルコフ連鎖であるから，定常分布 $\boldsymbol{\pi}^*$ を持つ．$\Phi^{Smooth}_{avg} = \frac{1}{r} \sum_{i=1}^{r} \frac{1}{d^*_{x_i}}$ とおく．$E[\Phi^{Smooth}_{avg}] = \frac{1}{d^*_{avg}}$ を示す．

$$\begin{aligned} E[\Phi^{Smooth}_{avg}] &= E[\frac{1}{d^*_{x_k}}] = \sum_{v_i \in V^*} \pi^*_i E[\frac{1}{d^*_{x_k}} | x_k = i] \\ &= \sum_{v_i \in V^*} \frac{d^*_i}{2m^*} \frac{1}{d^*_i} = \frac{1}{d^*_{avg}} \end{aligned}$$

1 つ目の等号は期待値の線型性より，2 つ目の等号は繰り返し期待値の法則より成り立つ．$d^*_{avg} = \frac{1}{E[\Phi^{Smooth}_{avg}]}$ である．

Φ_{avg}^{Smooth} は期待値に収束するので d_{avg}^{Smooth} は d_{avg}^* に収束する．□

本論文では各サンプルノード v_i にかける重みを $\frac{1}{d_i}$ に変更した推定アルゴリズムを提案する．提案アルゴリズムによる d_{avg} の推定値 $\hat{d}_{avg}$ を以下のように定義する:

$$\hat{d}_{avg} = \frac{r}{\sum_{i=1}^{r} \frac{1}{d_{x_i}}}$$

定理 4. $\hat{d}_{avg}$ は $\tilde{d}_{avg} = \frac{2m^*}{\sum_{v_i \in V^*} \frac{d_i^*}{d_i}}$ に収束する．

証明. $\hat{\Phi}_{avg} = \frac{1}{r}\sum_{i=1}^{r} \frac{1}{d_{x_i}}$ とおく．定理 3 の証明と同様にして，

$$E[\hat{\Phi}_{avg}] = \sum_{v_i \in V^*} \frac{d_i^*}{2m^*}\frac{1}{d_i} = \frac{1}{\tilde{d}_{avg}}$$

$\tilde{d}_{avg} = \frac{1}{E[\hat{\Phi}_{avg}]}$ である．$\hat{\Phi}_{avg}$ は期待値に収束するので $\hat{d}_{avg}$ は $\tilde{d}_{avg}$ に収束する．□

5.2 ノード数の推定

Katzir と Hardiman はサンプルノードの衝突に基づく，グラフのノード数を推定するアルゴリズムを提案した [7]．この推定アルゴリズムは，ランダムウォークでサンプルしたノード集合 $R = \{v_1, v_2, ..., v_r\}$ に対してある定数 m 以上離れたインデックスのペアの集合 I を調べる:

$$I = \{(k,l) \mid m \leqq |k-l| \wedge 1 \leqq k,l \leqq r\}$$

インデックスを m だけ離すことで，それぞれのノードが定常分布 π^* から独立にサンプルしたとみなすことができる [7]．

各ノードペア (v_{x_k}, v_{x_l}) に対して $v_{x_k} = v_{x_l}$ のとき 1 を返しそうでなければ 0 を返す変数 $\phi_{k,l}$ を定義する:

$$\phi_{k,l} = 1_{\{x_k = x_l\}}$$

ここで各ノードペア $(v_{x_k}, v_{x_l}) \in I$ に対して重み付けした $d_{x_k}^*\phi_{k,l}$ と $\frac{(d_{x_k}^*)^2}{d_{x_l}^*}$ のそれぞれの平均値 Φ_n^{KH}, Ψ_n^{KH} を以下のように定義する:

$$\Phi_n^{KH} = \frac{1}{|I|}\sum_{(k,l)\in I} d_{x_k}^*\phi_{k,l}$$

$$\Psi_n^{KH} = \frac{1}{|I|}\sum_{(k,l)\in I} \frac{(d_{x_k}^*)^2}{d_{x_l}^*}$$

C^* 上で Katzir と Hardiman の推定アルゴリズムを適用すれば，その推定値 n^{KH} は以下のように定義される:

$$n^{KH} = \frac{\Psi_n^{KH}}{\Phi_n^{KH}}$$

定理 5. n^{KH} は n^* に収束する．

証明. まず $E[\Phi_n^{KH}] = \sum_{v_j \in V^*} d_j^* \left(\frac{d_j^*}{2|E^*|}\right)^2$ を示す．

$$\Phi_n^{KH} = E[d_{x_k}^*\phi_{k,l}] = \sum_{v_j \in V^*} d_j^* \left(\frac{d_j^*}{2|E^*|}\right)^2$$

一つ目の等号は期待値の線型性より成り立つ．2 つ目の等号は $x_k = j$，$x_l = j$ である確率がともに $\frac{d_j^*}{2|E^*|}$ であり，インデックスが m だけ離れていることでそれぞれの事象は独立に起こることより成り立つ．

次に $E[\Psi_n^{KH}] = n^* \sum_{v_j \in V^*} d_j^* \left(\frac{d_j^*}{2|E^*|}\right)^2$ を示す．

$$E[\Psi_n^{KH}] = E[\frac{(d_{x_k}^*)^2}{d_{x_l}^*}] = \sum_{v_i \in V^*}\sum_{v_j \in V^*} \frac{(d_j^*)^2}{d_i^*}\frac{d_i^*}{2|E^*|}\frac{d_j^*}{2|E^*|}$$

$$= n^* \sum_{v_j \in V^*} d_j^* \left(\frac{d_j^*}{2|E^*|}\right)^2$$

$n^* = \frac{E[\Psi_n^{KH}]}{E[\Phi_n^{KH}]}$ である．Φ_n^{KH}, Ψ_n^{KH} は期待値に収束するので n^{KH} は n^* に収束する．□

我々は Φ_n^{KH} と Ψ_n^{KH} の重み付けを変更した $\hat{\Phi}_n$ と $\hat{\Psi}_n$ から推定値を求める．

$$\hat{\Phi}_n = \frac{1}{|I|}\sum_{(k,l)\in I} d_{x_k}\phi_{k,l}$$

$$\hat{\Psi}_n = \frac{1}{|I|}\sum_{(k,l)\in I} \frac{(d_{x_k})^2}{d_{x_l}^*}$$

ノード数の推定値 $\hat{n}$ を以下のように定義する．

$$\hat{n} = \frac{\hat{\Psi}_n}{\hat{\Phi}_n}$$

定理 6. $\hat{n}$ は $\tilde{n} = \frac{n^* \sum_{v_j \in V^*} d_j^* (d_j)^2}{\sum_{v_j \in V^*} d_j (d_j^*)^2}$ に収束する．

証明. 定理 5 の証明と同様にして以下が示される．

$$E[\hat{\Phi}_n] = \sum_{v_j \in V^*} d_j \left(\frac{d_j^*}{2|E^*|}\right)^2$$

$$E[\hat{\Psi}_n] = n^* \sum_{v_j \in V^*} d_j^* \left(\frac{d_j}{2|E^*|}\right)^2$$

$\tilde{n} = \frac{E[\hat{\Psi}_n]}{E[\hat{\Phi}_n]}$ である．$\hat{\Phi}_n, \hat{\Psi}_n$ は期待値に収束するので $\hat{n}$ は $\tilde{n}$ に収束する．□

API が隣接ノードのプライバシ情報を提供しないモデルでは，それぞれの推定アルゴリズムにおける各サンプルノード v_{x_i} の重みに用いる厳密値 $d_{x_i}^*$ の代わりに近似値 $\hat{d}_{x_i}^*$ を用いる．$\hat{d}_{x_i}^*$ は 4 章に基づいて近似計算する．このモデルでも，定理 2 より，定理 4 と定理 6 が成り立つ．

6 実　　験

実際の OSN のグラフを用いて提案アルゴリズムを評価する．対象とするグラフが有向グラフであれば反対向きのエッジを追加することで無向グラフとし，非連結なグラフはそのグラフの最大連結成分を扱った．また，セルフループと多重辺は削除した．本論文で扱う各グラフのノード数，平均次数を表 6 に示す．

API が提供する隣接ノード情報のそれぞれのモデル (図 1)

Network	n	d_{avg}
LiveMocha [21]	104103	42.13
douban [21]	154908	4.22
gowalla [21]	196591	9.67
academia [21]	200167	10.22
Pokec [22]	1632803	27.32

表 1　データセット

	収束値	相対誤差
Smooth	19.49	0.287
Ours	28.31	**0.0362**
KH	1080278	0.338
Ours	1627218	**0.00342**

表 2　Pokec におけるそれぞれの推定アルゴリズムの収束値

において，提案アルゴリズムの頑健性の評価と，実際にプライベートなノードを含む Pokec グラフ [24] における推定精度の評価を行う．推定精度の評価に正規化平方二乗誤差 (NRMSE) を用いる．NRMSE は推定値の誤差と分散を評価できる指標であり，関連する多くの研究で用いられている [15, 7, 11, 23].

6.1　提案アルゴリズムの頑健性

提案アルゴリズムの頑健性を評価する．グラフ内のプライベートなノードの割合が増加したとしても，NRMSE の増加が小さいことが望ましい．グラフ内のプライベートなノードの割合を増加させた時のそれぞれの推定アルゴリズムの NRMSE を比較する．一方で，プライバシ保護などの観点からプライベートなノードを含む公開データセットは我々の知る限り Pokec のみであった [24]．このため本論文では 4 つのグラフ LiveMocha, douban, gowalla, academia に対してパラメータ p を与えて，割合 p のノードをランダムに選びそれらをプライベートなノードとする前処理を行った．そして，その最大クラスタからランダムに選んだ初期ノードとクエリ数を与えてランダムウォークを実行し，それぞれの推定値を計算する．上記のシミュレーションを独立に 1000 回行い，それぞれのアルゴリズムの NRMSE を比較する．パラメータ p は 0.0 から 0.30 まで 0.01 ごとに変化させた．$p = 0.0$ のときそれぞれのアルゴリズムは等しい NRMSE を持つ．また，ノード数推定アルゴリズムにおける定数 m は，サンプル列の長さの 2.5%と設定した．

まず，API が隣接ノードのプライバシ情報を提供するモデルにおける提案アルゴリズムの頑健性を議論する．平均次数の推定アルゴリズムの頑健性を評価する．図 3 と図 4 は各グラフにおけるそれぞれクエリ数をノード数の 1%，5%とした時の Dasgupta らのアルゴリズム (**Smooth**) [9] と提案アルゴリズム (**Ours**) のパラメータ p に対する NRMSE を示す．いずれのクエリ数に対しても，全てのグラフにおいて **Smooth** はパラメータ p が増加するにつれて NRMSE が増加している．一方で，**Ours** の NRMSE は p が増加してもほとんど増加していない．

ノード数の推定アルゴリズムの頑健性を評価する．図 5 と図 6 はそれぞれクエリ数をノード数の 1%，5%とした時の Katzir と Hardiman のアルゴリズム (**KH**) [7] と提案アルゴリズム (**Ours**) のパラメータ p に対する NRMSE を示す．ノード数衝突に基づくアルゴリズムはクエリ数が極端に少ない場合，サンプル内のノードの衝突がほとんど生じないため推定精度が低くなる．このため，クエリ数が 1%のとき両方のアルゴリズムの NRMSE が高くなっている．また，p が 0.0 から 0.2 まででは **Ours** の NRMSE は **KH** より高いが，p が 0.2 から 0.3 まででは **Ours** の NRMSE は **KH** より低い．一方で，クエリ数を 5%としたとき NRMSE に顕著な差が見られる．平均次数推定と同様に，提案アルゴリズムの NRMSE は p の増加に対してほとんど増加していない．

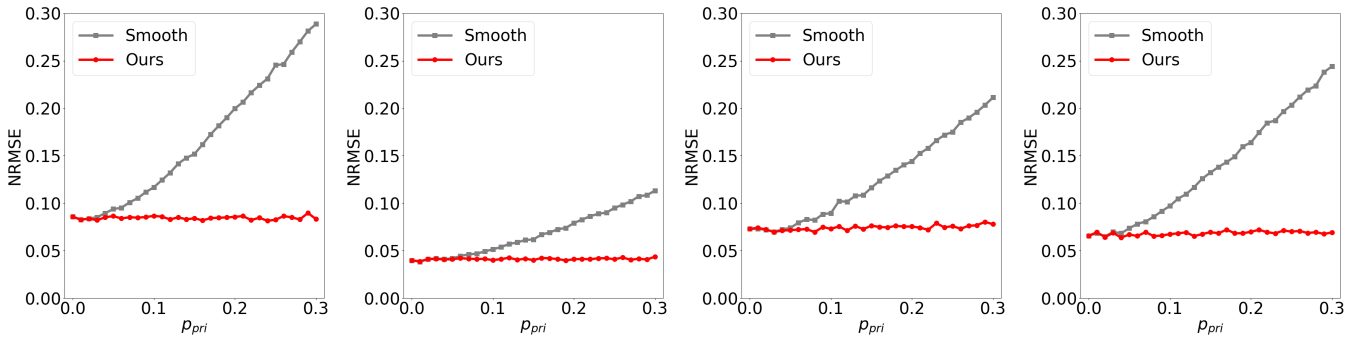
図 7 は LiveMocha における，API が隣接ノードのプライバシ情報を提供しないモデルにおける結果を示す．この場合も，提案アルゴリズムは既存アルゴリズムと比較して p が増加しても NRMSE はほとんど増加していない．Douban, gowalla, academia でも同様の結果が得られた．

6.2　Pokec グラフにおける提案アルゴリズムの推定精度

実際にプライベートなノードを含むソーシャルネットワークである Pokec グラフにおいて，提案アルゴリズムの推定精度を評価する．Pokec グラフは Takac らがプライベートなノードを含む当時の全グラフデータを取得し，各ノードのプライバシ情報も付与されている [22, 24]．プライベートなノードは 552525 ノード (約 33.8%) 存在する．

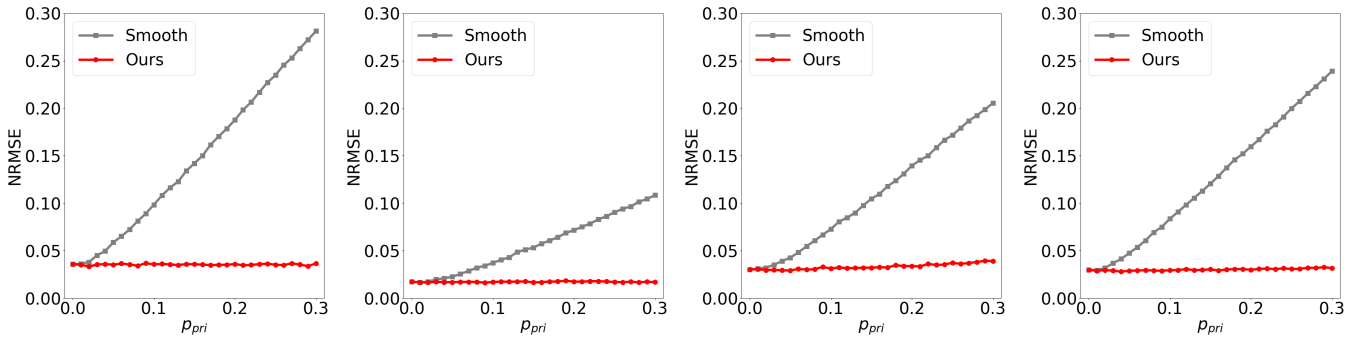
Pokec グラフに対して，その最大クラスタからランダムに選んだ初期ノードとクエリ数を与えてランダムウォークを実行し，それぞれの推定値を計算する．このシミュレーションを独立に 1000 回行い，それぞれのアルゴリズムの NRMSE を比較する．クエリ数はノード数の 0.5%から 5%まで 0.25%ごとに変化させた．

図 8 は，API が隣接ノードのプライバシ情報を提供するモデルにおける，Pokec の平均次数とノード数推定における既存アルゴリズム (**Smooth**,**KH**) と提案アルゴリズム (**Ours**) のクエリ数に対する NRMSE を示す．既存アルゴリズムの NRMSE は，クエリ数が増加してもほとんど減少していない．一方で提案アルゴリズムはクエリ数が増加するにつれて NRMSE が減少している．表 2 は定理 3,4,5,6 を用いてそれぞれの推定アルゴリズムの収束値と真値との相対誤差を計算したものである．提案アルゴリズムの収束値の相対誤差は既存アルゴリズムと比較して非常に小さいことがわかる．いずれの推定においても，提案アルゴリズムによる推定値は既存アルゴリズムより総じて低い NRMSE を示している．1%未満のクエリ数でのノード数推定では提案アルゴリズムの推定値の誤差が大きくなっており，この改善は今後の課題である．図 9 は API が隣接ノードのプライバシ情報を提供しないモデルにおける結果を示す．この場合も，提案アルゴリズムは総じて低い NRMSE を示している．



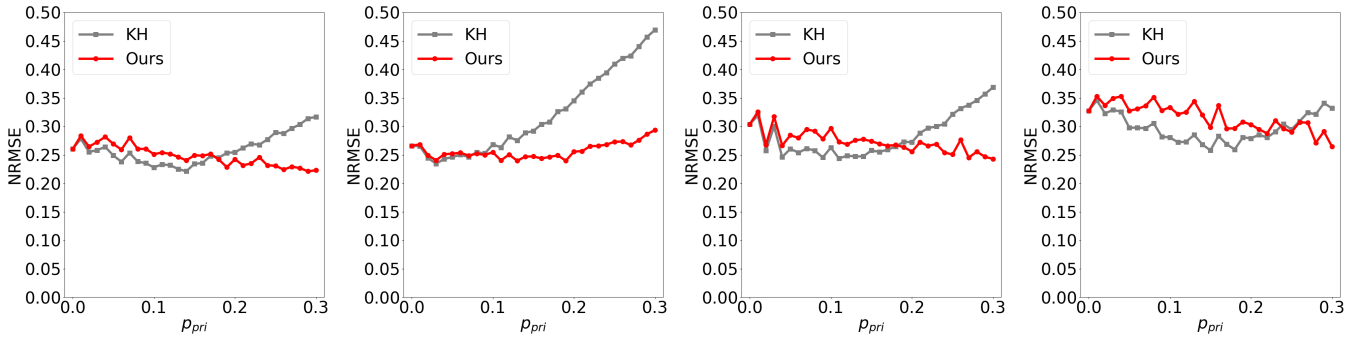
(a) LiveMocha (b) douban (c) gowalla (d) academia

図 3　p に対する平均次数の推定値の NRMSE (API が隣接ノードのプライバシ情報を提供するモデル，クエリ数: $0.01n$)



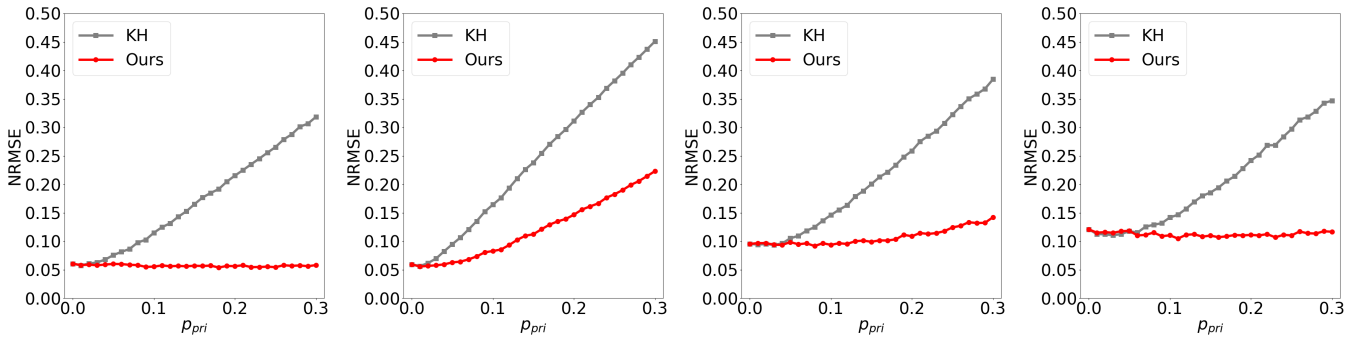
(a) LiveMocha (b) douban (c) gowalla (d) academia

図 4　p に対する平均次数の推定値の NRMSE (API が隣接ノードのプライバシ情報を提供するモデル，クエリ数: $0.05n$)



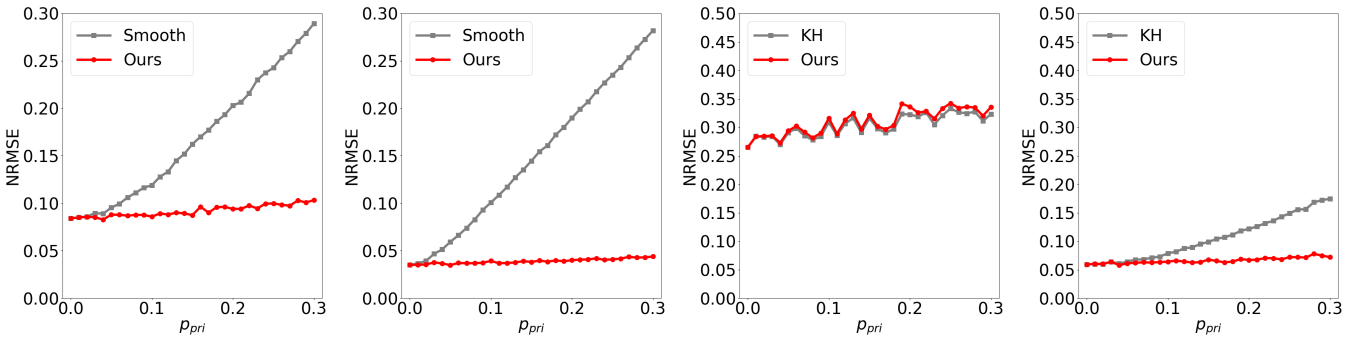
(a) LiveMocha (b) douban (c) gowalla (d) academia

図 5　p に対するノード数の推定値の NRMSE (API が隣接ノードのプライバシ情報を提供するモデル，クエリ数: $0.01n$)



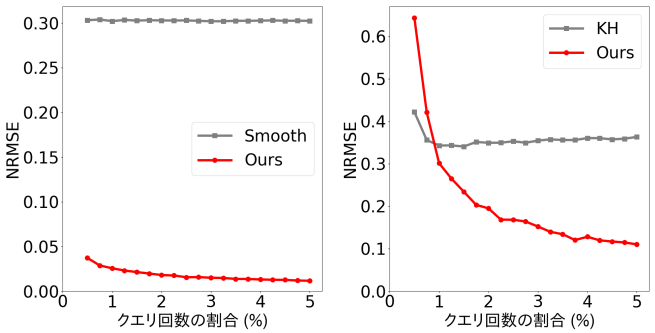
(a) LiveMocha (b) douban (c) gowalla (d) academia

図 6　p に対するノード数の推定値の NRMSE (API が隣接ノードのプライバシ情報を提供するモデル，クエリ数: $0.05n$)



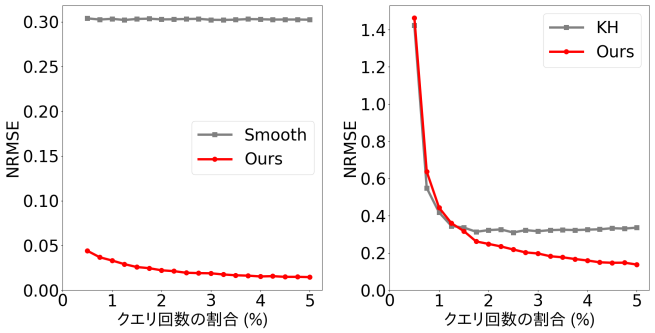
(a) 平均次数推定, クエリ数: $0.01n$ (b) 平均次数推定, クエリ数: $0.05n$ (c) ノード数推定, クエリ数: $0.01n$ (d) ノード数推定, クエリ数: $0.05n$

図 7 LiveMocha における p に対する推定値の NRMSE (API が隣接ノードのプライバシ情報を提供しないモデル)



(a) 平均次数推定 (b) ノード数推定

図 8 Pokec におけるクエリ数に対する推定値の NRMSE (API が隣接ノードのプライバシ情報を提供するモデル)



(a) 平均次数推定 (b) ノード数推定

図 9 Pokec におけるクエリ数に対する推定値の NRMSE (API が隣接ノードのプライバシ情報を提供しないモデル)

7 関 連 研 究

オンラインソーシャルネットワークのプライバシを考慮した研究はいくつかある [25–27]. Bonneau らは Facebook の公開されている隣接関係に基づくグラフ統計量の近似を議論した [25]. 各ユーザの友人のうちわずか 8 ユーザが公開されているだけで, その部分グラフ上のノードの次数や媒介中心性, コミュニティ検出などの統計量を十分に近似できることを実験的に示した. しかし, ランダムウォークを用いて統計量を推定する本研究とは異なる. Ye らは OSN のプライベートなノードがクローリングに与える影響を言及している [26]. 実際の OSNs のデータセットに対して 10 万ノードをランダムにプライベートなノードとしてクローリングで取得できるノード数とエッジ数の減少率を調べており, 大きな減少は見られないとしている. しかし, この研究は統計量推定におけるプライベートなノードの影響は論じていない. Chierichetti らはソーシャルネットワークにおける public-private model を提案した [27]. このモデルでは全体のグラフ G をパブリックなグラフと呼び, パブリックなグラフに属する各ノード u はプライベートなグラフ G_u を持つ. パブリックなグラフはどのノードからでも見えるグラフであり, プライベートなグラフ G_u はノード u のみが見えるグラフである. 各ノード u に対して $G \cup G_u$ の統計量を効率よく計算するアルゴリズムを提案している. しかし, この研究はグラフの全データにアクセスできることを前提にしているため本研究とは異なる.

8 ま と め

本研究ではプライベートなノードを含むソーシャルネットワークに対する, ランダムウォークを用いた統計量推定アルゴリズムを議論した. まず, Gjoka らのパブリックなノードのみを遷移するランダムウォークの定常分布を導出し, 各サンプルノードはそのパブリックな隣接ノード数に比例した分布でサンプルされることを示した. 平均次数とノード数に対する既存の推定アルゴリズムは, プライベートなノードの割合が増加するにつれて推定値の誤差が大きくなる問題点があった. 我々は, 既存アルゴリズムのサンプルノードへの重みを変更することで, プライベートなノードの割合に対して頑健な推定を可能にした. 本研究では, OSN の API が提供する隣接ノード情報の 2 つのモデルを考え, それぞれのモデルにおける実験で, 提案アルゴリズムがプライベートなノードの割合にほぼ依存せず,

高精度に平均次数とノード数を推定できることを示した．今後は，統計量の推定精度の向上だけでなく，プライベートなノードの割合に対して頑健な推定を達成するアルゴリズムを設計するべきである．

謝　　辞

本研究の一部は、国立研究開発法人新エネルギー・産業技術総合開発機構（NEDO）の委託業務として行われました．本研究は JSPS 科研費 16K12406 の助成を受けたものです．

文　　献

[1] Yong-Yeol Ahn, Seungyeop Han, Haewoon Kwak, Sue Moon, and Hawoong Jeong. Analysis of topological characteristics of huge online social networking services. In *Proceedings of the 16th international conference on World Wide Web*, pp. 835–844. ACM, 2007.

[2] Alan Mislove, Massimiliano Marcon, Krishna P Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pp. 29–42. ACM, 2007.

[3] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pp. 591–600. AcM, 2010.

[4] Minas Gjoka, Maciej Kurant, Carter T Butts, and Athina Markopoulou. Practical recommendations on crawling online social networks. *IEEE Journal on Selected Areas in Communications*, Vol. 29, No. 9, pp. 1872–1892, 2011.

[5] Flavio Chiericetti, Anirban Dasgupta, Ravi Kumar, Silvio Lattanzi, and Tamás Sarlós. On sampling nodes in a network. In *Proceedings of the 25th International Conference on World Wide Web*, pp. 471–481. International World Wide Web Conferences Steering Committee, 2016.

[6] Rong-Hua Li, Jeffrey Xu Yu, Lu Qin, Rui Mao, and Tan Jin. On random walk based graph sampling. In *2015 IEEE 31st International Conference on Data Engineering (ICDE)*, pp. 927–938. IEEE, 2015.

[7] Liran Katzir and Stephen J Hardiman. Estimating clustering coefficients and size of social networks via random walk. *ACM Transactions on the Web (TWEB)*, Vol. 9, No. 4, p. 19, 2015.

[8] Liran Katzir, Edo Liberty, Oren Somekh, and Ioana A Cosma. Estimating sizes of social networks via biased sampling. *Internet Mathematics*, Vol. 10, No. 3-4, pp. 335–359, 2014.

[9] Anirban Dasgupta, Ravi Kumar, and Tamas Sarlos. On estimating the average degree. In *Proceedings of the 23rd international conference on World wide web*, pp. 795–806. ACM, 2014.

[10] Bruno Ribeiro and Don Towsley. On the estimation accuracy of degree distributions from graph sampling. In *Decision and Control (CDC), 2012 IEEE 51st Annual Conference on*, pp. 5240–5247. IEEE, 2012.

[11] Xiaowei Chen, Yongkun Li, Pinghui Wang, and John Lui. A general framework for estimating graphlet statistics via random walk. *Proceedings of the VLDB Endowment*, Vol. 10, No. 3, pp. 253–264, 2016.

[12] Kenta Iwasaki and Kazuyuki Shudo. Estimating the clustering coefficient of a social network by a non-backtracking random walk. In *Big Data and Smart Computing (BigComp), 2018 IEEE International Conference on*, pp. 114–118. IEEE, 2018.

[13] Toshiki Matsumura, Kenta Iwasaki, and Kazuyuki Shudo. Average path length estimation of social networks by random walk. In *Big Data and Smart Computing (BigComp), 2018 IEEE International Conference on*, pp. 611–614. IEEE, 2018.

[14] Kazuki Nakajima, Kenta Iwasaki, Toshiki Matsumura, and Kazuyuki Shudo. Estimating top-k betweenness centrality nodes in online social networks. In *Proceedings of the IEEE ISPA-IUCC-BDCloud-SocialCom-SustainCom 2018*, pp. 1128–1135. IEEE, 2018.

[15] Chul-Ho Lee, Xin Xu, and Do Young Eun. Beyond random walk and metropolis-hastings samplers: why you should not backtrack for unbiased graph sampling. In *ACM SIGMETRICS Performance evaluation review*, Vol. 40, pp. 319–330. ACM, 2012.

[16] Ratan Dey, Zubin Jelveh, and Keith Ross. Facebook users have become much more private: A large-scale study. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2012 IEEE International Conference on*, pp. 346–352. IEEE, 2012.

[17] Kenta Iwasaki and Kazuyuki Shudo. Comparing graph sampling methods based on the number of queries. In *Proceedings of the IEEE ISPA-IUCC-BDCloud-SocialCom-SustainCom 2018*, pp. 1136–1143. IEEE, 2018.

[18] David A Levin and Yuval Peres. *Markov chains and mixing times*, Vol. 107. American Mathematical Soc., 2017.

[19] Réka Albert, Hawoong Jeong, and Albert-László Barabási. Error and attack tolerance of complex networks. *nature*, Vol. 406, No. 6794, p. 378, 2000.

[20] Reuven Cohen, Keren Erez, Daniel Ben-Avraham, and Shlomo Havlin. Resilience of the internet to random breakdowns. *Physical review letters*, Vol. 85, No. 21, p. 4626, 2000.

[21] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.

[22] Jure Leskovec and Andrej Krevl. SNAP Datasets: Stanford large network dataset collection. `http://snap.stanford.edu/data`, June 2014.

[23] Pinghui Wang, John Lui, Bruno Ribeiro, Don Towsley, Junzhou Zhao, and Xiaohong Guan. Efficiently estimating motif statistics of large networks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, Vol. 9, No. 2, p. 8, 2014.

[24] Lubos Takac and Michal Zabovsky. Data analysis in public social networks. In *International Scientific Conference and International Workshop Present Day Trends of Innovations*, 2012.

[25] Joseph Bonneau, Jonathan Anderson, Ross Anderson, and Frank Stajano. Eight friends are enough: social graph approximation via public listings. In *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems*, pp. 13–18. ACM, 2009.

[26] Shaozhi Ye, Juan Lang, and Felix Wu. Crawling online social graphs. In *Web Conference (APWEB), 2010 12th International Asia-Pacific*, pp. 236–242. IEEE, 2010.

[27] Flavio Chierichetti, Alessandro Epasto, Ravi Kumar, Silvio Lattanzi, and Vahab Mirrokni. Efficient algorithms for public-private social networks. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 139–148. ACM, 2015.