On Estimating the Average Degree

Anirban Dasgupta[*]
Indian Institute of Technology
Gandhinagar, Gujarat, India
anirbandg@iitgn.ac.in

Ravi Kumar
Google Inc.
Mountain View, CA, USA
ravi.k53@gmail.com

Tamás Sarlós
Google Inc.
Mountain View, CA, USA
stamas@gmail.com

ABSTRACT

Networks are characterized by nodes and edges. While there has been a spate of recent work on estimating the number of nodes in a network, the edge-estimation question appears to be largely unaddressed. In this work we consider the problem of estimating the average degree of a large network using efficient random sampling, where the number of nodes is not known to the algorithm. We propose a new estimator for this problem that relies on access to node samples under a prescribed distribution. Next, we show how to efficiently realize this ideal estimator in a random walk setting. Our estimator has a natural and simple implementation using random walks; we bound its performance in terms of the mixing time of the underlying graph. We then show that our estimators are both provably and practically better than many natural estimators for the problem. Our work contrasts with existing theoretical work on estimating average degree, which assume that a uniform random sample of nodes is available and the number of nodes is known.

Categories and Subject Descriptors

F.2.2 [**Theory of Computing**]: Analysis of Algorithms and Problem Complexity—*Nonnumerical Algorithms and Problems*; G.2.2 [**Mathematics of Computing**]: Graph Theory—*Graph Algorithms*; G.3 [**Mathematics of Computing**]: Probability and Statistics—*Probabilistic Algorithms*

Keywords

Graph sampling; Random walks; Average degree

1. INTRODUCTION

Estimating the size of an unknown population is a classical problem in statistics. This problem arises in a variety of fields—from estimating the number of German tanks in World War II to estimating animal population sizes—and has been studied for the past many decades. Several well-known statistical methods such as mark-and-capture, the Lincoln–Peterson estimator, the Chapman index, have been developed for this problem. In the last few years, such problems have also been actively considered in the World Wide Web setting. Typical problems studied in this context include estimating the size of the web, estimating the size of a web index, estimating certain online user populations, and estimating the parameters of online social networks.

In this work our focus is on estimating the key parameters of large networks. At a high-level, the motivation is clear: to understand networks in general, and in case of social networks, to gain business insights and competitive advantage. As usual, we assume the network is not available to us in its entirety. Instead, it can only be accessed by the following interface: one can query a node and obtain all its (publicly visible) neighbors. This interface is quite natural and is supported by the APIs of many online networks. Queries to the network are expensive in general (e.g., APIs are usually rate-limited) and therefore, any estimation method has to make only a small number of network access queries. In addition, the estimation method also cannot assume that it can access a uniformly random network node since generating random network nodes is a hard problem by itself!

There has been a spate of work on estimating the number of nodes in a network. Most of the algorithms work by using the "birthday paradox" to count collisions: the expected number of collisions in r samples from a universe of (unknown) size n is roughly r^2/n. In fact, it is easy to obtain the number of nodes in a social network; in most cases, Wikipedia has a reliable (and an almost up-to-date) answer. However, the number of edges—hence, the average degree—seems less reliably available for many networks.[*] This is even more so if each edge has a type, e.g., friend, family, coworker, etc. Our goal in this work is to develop algorithms to efficiently estimate the average degree of a network.

One easy way to solve our problem is to estimate the number n of nodes and the number m of edges separately and then combine them to yield an estimate for the average degree. However, this method is highly inefficient since we require roughly $O(\sqrt{n})$ and $O(\sqrt{m})$ samples if done naively; under some assumptions, these are tight bounds. These factors along with the others demand four desirable properties

[*]This work was done while the author was at Yahoo Labs, Sunnyvale, CA, USA.

[*]In fact, we became interested in the average degree problem when one of our friends from a social networking company refused to divulge the number of edges, terming it proprietary and not easily revealed!

WWW'14, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2744-2/14/04.
http://dx.doi.org/10.1145/2566486.2568019.

of a good sampling algorithm for estimating the average degree. The algorithm should:

(i) not need a uniform sample of nodes; as argued before, a uniform sample of nodes can be expensive and in some case, impossible to procure.

(ii) not assume that it knows the number of nodes in the network, since estimating the latter is a task in itself, often requiring a lot of samples from the network.

(iii) be conducive to be realized in practical and effective ways of accessing the graph such as by doing a random walk.

(iv) use asymptotically fewer number of samples than that required for separately estimating the number of edges and the number of nodes.

Our results. In this paper we obtain a highly efficient estimator for the average degree. This estimator uses roughly $O(\log U \cdot \log\log U)$ samples, where U is an upper bound on the maximum degree of the graph (typically, $|U| \ll n$). This is much lower than the sample complexity of some obvious methods. For example, sampling a few nodes and outputting the average of their degrees needs $\Theta(\sqrt{n})$ samples.

We describe two different estimators, namely, Smooth and Guess&Smooth. The estimator Smooth assumes a crude constant-factor approximation to the average degree and outputs an arbitrary approximation to the average degree. This estimator can be interpreted as doing a *smoothed random walk* on a graph, where at each step, a constant number of self-loops are added to the walk; the number of self-loops will depend on the crude average degree estimate. The estimator Guess&Smooth works by successively guessing estimates of the average degree and returning an estimate if it "looks right" in a statistical sense. The purpose of Guess&Smooth is to quickly create a constant factor approximation to the average degree that Smooth can then use. The final (theoretical) estimator Combined uses a combination of these two. We show several properties of our estimators, including their sample complexity and their bias. Note that while estimators Guess&Smooth and Combined are necessary to show the logarithmic sample size guarantees, in case an approximate estimate of average degree is readily available (e.g. if average degree is small, as in most real networks), the estimator Smooth suffices by itself.

We then conduct extensive experiments on four publicly available networks. We compare our estimators against several baselines, including ones based on previous work and ones based on collisions. We show that these new estimators outperform other estimators on all datasets.

2. RELATED WORK

The main topics related to this paper are the following: the theoretical work on estimating the average degree of a graph, the large body of work on estimating the number of nodes in a graph, and the mostly heuristic work on estimating graph parameters including the number of edges.

The problem of estimating the average degree of a graph was first addressed by Feige [7]. He used uniform sampling of the nodes to obtain a constant-factor approximation to the degree; the main point of this result is the use of a careful analysis that precludes the linear dependence of sample size on n, or on the maximum degree of the graph. Goldreich and Ron [9] presented a more involved estimator but with a simpler proof. They also show how to extend this estimator to achieve arbitrary approximations, provided they can access a random neighbor of each node. These estimators perform very well (and are in a sense optimal) when nodes can be sampled uniformly at random. When such a sampling is not feasible or is expensive, they are less effective. We use these estimators as baselines in our experiments.

There have been several recent papers on estimating the size (i.e., the number of nodes) of social networks. Katzir, Liberty, and Somekh [11] considered the size estimation problem. Their main idea was to use collisions in order to determine the size. We also consider collisions, but more of as a baseline since any collision-based approach requires $\Omega(\sqrt{n})$ many samples, which as we show, is an overkill for average degree estimation. Hardiman and Katzir [10] use collision among neighbors for network size estimation and show that it has a tighter confidence interval than a simple node collision estimator. They also consider the problem of estimating the clustering coefficients. Ye and Wu [20] also consider the social network size estimation problem; they assume the ability to uniformly sample a node. Gjoka et al. [8] modified the simple random walk to induce the uniform distribution on nodes as the stationary distribution. Such a method can be used to sample the nodes uniformly and hence estimate the network size. However, it is unclear if the mixing time of the modified random walk would be small, even if the original graph has a small mixing time. Cooper, Radzik, and Siantos [4] also used random walk methods for estimating network parameters, but they go beyond collisions by actually using the return times for estimation. However, their sample complexity is still bounded by that of collisions.

While many of the above techniques can be extended to estimating the number of edges, there has been very little work on estimating the average degree per se. Recently, Kurant, Butts, and Markopoulou [12] addressed average degree as part of their work on network size estimation. One of their estimators is the same as that of [7]; however, they do not provide any variance analysis of their estimators. There have been some recent work on estimating personal network size (i.e., degree) using techniques inspired by respondent-driven sampling [15, 16, 17]; however, many of these results are heuristic in nature and do not address the sample complexity in a principled manner.

The general question of size estimation has been addressed in the context of a web index. Several methods from the theory of random walks and sampling have been used for this purpose; see the work of Bharat and Broder [3] and the works of Bar-Yossef and Gurevich [1, 2].

3. PRELIMINARIES

Let $G = (V, E)$ be an undirected graph. Let $n = |V|$ be the number of nodes and $m = |E|$ be the number of edges. For a node $v \in V$, let $\Gamma(v) = \{w \mid (v, w) \in E\}$ be the set of its neighbors and let $\deg(v) = |\Gamma(v)|$ be its degree. Let $d_{\max} = \max_{v \in V} \deg(v)$ be the maximum degree, $d_{\text{avg}} = \sum_{v \in V} \deg(v)/n$ be the average degree, and $d_{\min} = \min_{v \in V} \deg(v)$ be the minimum degree.

We will focus on two models of accessing the graph. In the first model, called the *ideal* model, we assume that we can access the nodes of the graph according to a prescribed *sampling* distribution. A distribution of particular interest is the uniform distribution on the nodes, denoted $\mathcal{D}_{\text{u}}$. We will also consider distributions that are proportional to the degrees of the node. Let $\mathcal{D}_{\text{d,c}}$ denote the distribution where the node v is chosen with probability proportional to

$\deg(v) + c$, where c is some fixed quantity. For simplicity, we denote $\mathcal{D}_{\rm d} = \mathcal{D}_{{\rm d},0}$. We will use $X \sim \mathcal{D}$ to denote that X is chosen according to the distribution $\mathcal{D}$.

While the ideal model is a clean abstraction to express various algorithms and bounds, it is very expensive in practice and in some cases, may not be possible at all. Therefore, for practical purposes, we focus on a second model, called the *random walk* model, where we assume that the graph can be accessed by performing a random walk. Note that a uniform random walk will generate samples according to $\mathcal{D}_{\rm d}$. By adding c self-loops to each node, can generate samples according to $\mathcal{D}_{{\rm d},c}$; we call this the *smoothed random walk*. Note that in either case, these samples are not independent. Also note that the smoothed random walk is different from the usual lazy random walk [13]. In a lazy random walk, each self-loop acquires a constant fraction of the transition probability at each node; thus it does not alter the original stationary distribution.

In both ideal and random walk models, we will focus on estimating $d_{\rm avg}$ of a graph to within $(1 \pm \epsilon)$ accuracy, with probability at least $1 - \delta$. Our bounds will be expressed in terms of the accuracy parameter ϵ and the confidence parameter δ, in addition to the parameters of the graph. For simplicity, unless stated otherwise, we will drop the dependence of the sample complexity on δ, which will typically be $\log 1/\delta$, achieved by running the algorithm this many times and taking the median value to boost the confidence. Given a good approximation of $d_{\rm avg}$ and n, it is straightforward to estimate m, the number of edges in the graph.

We will use the following concentration inequalities:

Theorem 1 (Hoeffding's inequality). *[6] Let $X_1, \ldots, X_r$ be independent random variables with $|X_i| \le 1$ for all i. Set $\mu = \sum_{i=1}^{r} E[X_i]$. Then, for all $t > 0$, we have*

$$\Pr\left[\left|\sum_{i=1}^{r} X_i - \mu\right| > t\right] \le 2\exp\left(-\frac{2t^2}{r}\right).$$

Theorem 2 (Bernstein's inequality). *[6] Let $X_1, \ldots, X_r$ be independent random variables with $|X_i - E[X_i]| \le M$ for all i. Set $\mu = \sum_{i=1}^{r} E[X_i]$. Then, for all $t > 0$, we have*

$$\Pr\left[\left|\sum_{i=1}^{r} X_i - \mu\right| > t\right] \le 2\exp\left(-\frac{t^2/2}{\sum_{i=1}^{r} \mathrm{var}[X_i^2] + Mt/3}\right).$$

4. ESTIMATION IN THE IDEAL MODEL

First we present our average degree estimator. For ease of exposition, we describe it in the ideal model, i.e., we assume that we can sample the nodes of the graph according to a pre-specified distribution. We also present two other families of estimators: the first based on a uniform sampling of nodes and the second based on counting collisions.

4.1 Our estimators

Our estimators will sample nodes using $\mathcal{D}_{{\rm d},c}$ for some specified value c. Even though such a sampling looks contrived in the ideal model, as we will see later, it is very natural in a random walk model. Furthermore, our estimators also do not need to know the number n of nodes.

A priori, it is not obvious that estimating average degree is a much easier problem than estimating the number of nodes or edges. In fact, depending on the sampling model, it is possible to reduce the problem of estimating average degree to one of estimating the number of nodes to within a constant factor. Feige's [7] lower bound shows, under uniform sampling alone, with $o(n)$ samples cannot hope to do better than a 2-approximation to estimate the average degree; this is true even if we assume the number of nodes to be known—a star with n nodes is a simple example. On the other hand, if sampling proportional to degrees is the only sampling method available to the algorithm, it become hard to estimate the number of nodes that have low degree. As an example, consider a graph that has a clique of size $n/4$ and other $3n/4$ nodes of degree 1 each—using $o(n)$ degree proportional samples, we will not see a node of degree 1.

Our main intuition is to show that using a sampling distribution $\mathcal{D}_{{\rm d},c}$ that is a combination of the above two schemes, uniform and degree proportional, we can do exponentially better in theory, and also in practice.

Informally, our first estimator Guess&Smooth works by successively guessing estimates of the average degree and returning an estimate if it "looks right," i.e., if the number of nodes in the sample whose degree is more than the current estimate is a constant fraction of the sample itself. As we will show in Section 5.4 later, Guess&Smooth is able to give the estimate of average degree to a small constant factor.

Algorithm 1 Guess&Smooth(G, L, U, δ)

```
Input: Graph $G$, lower and upper bounds $L$ and $U$ for $d_{\rm avg}$,
  probability of failure $\delta > 0$
  $\epsilon_0 \leftarrow 1/12$
  for $c \in L \cdot \{1, 2, 2^2, \ldots, U/L\}$ do
    $N \leftarrow 0$
    for $r = \frac{\log((2/\delta)\log_2(U/L))}{2\epsilon_0^2}$ steps do
      $v \sim \mathcal{D}_{{\rm d},c}$          // sample with replacement
      if $\deg(v) \le c$ then
        $N \leftarrow N + 1$
    if $N/r \ge 1/2 - \epsilon_0$ then
      return $c$
  return $U$
```

We next present another estimator Smooth, which assumes that a crude estimate of the average degree is available, and outputs a more accurate estimate. In the following, the value c can be thought of as a crude estimate of the degree. The reason we present this alternate estimator are two-fold: it makes our analysis easier to describe and this simpler version works quite well in practice.

Algorithm 2 Smooth(G, r, c)

```
Input: Graph $G$, sample size $r$, $c > 0$
  $S \leftarrow \emptyset$
  for $r$ steps do
    $v \sim \mathcal{D}_{{\rm d},c}$          // sample with replacement
    $S \leftarrow S \cup \{v\}$
  return $(\sum_{u \in S} \frac{\deg(u)}{\deg(u)+c})/(\sum_{u \in S} \frac{1}{\deg(u)+c})$
```

Finally, we give a third, combined, estimator that is capable of efficiently estimating the average degree with arbitrary precision without the assumption that a crude estimate is already available. This estimator essentially functions by first obtaining a crude estimate using Guess&Smooth, which is then refined using Smooth.

The analysis of our estimators and some of their statistical properties will be presented in Section 5.

```
Algorithm 3 Combined($G, L, U, \epsilon, \delta$)
Input: Graph $G$, lower and upper bounds $L$ and $U$ for $d_{\rm avg}$,
  accuracy $0 < \epsilon \le 1/2$, probability of failure $\delta > 0$
  $\widetilde{d} \leftarrow$ Guess&Smooth($G, L, U, \delta/2$)
  $\widetilde{r} \leftarrow \frac{36\log(8/\delta)}{\epsilon^2}$
  return Smooth($G, \widetilde{r}, \widetilde{d}$)
```

4.2 Other sampling-based estimators

Here we present three other sampling-based estimators in the ideal model. All these estimators obtain the degree of a chosen node and use this information (sometimes, in less obvious ways) in order to estimate the average degree. All the estimators assume that n is known to the algorithm. These estimators are based on prior work (some not necessarily designed for the average degree) and will serve as baselines for experimental purposes.

The first estimator is straightforward. It is based on sampling nodes uniformly at random, i.e., according to $\mathcal{D}_{\rm u}$. Let $x_1, \ldots, x_r$ be the node samples. The estimator is given by

$$\hat{d}_{\sf Feige} = \frac{1}{r}\sum_{i=1}^{r}\deg(x_i).$$

Even though one can prove a weak bound on this estimator that will depend on $d_{\max}$, a much stronger sample complexity bound was proved by Feige [7].

Theorem 3. *[7] If $r = O(\sqrt{n/L}/\epsilon)$, where L is a lower bound on $d_{\rm avg}$, then $\hat{d}_{\sf Feige}$ is a $(2+\epsilon)$-approximation to $d_{\rm avg}$.*

Our second estimator is an improvement obtained by Goldreich and Ron [9] to $\hat{d}_{\sf Feige}$. The assumptions are the same as for $\hat{d}_{\sf Feige}$ except that an additional minor assumption is made: for any node v, we can obtain one of its random neighbors. This estimator (called $\hat{d}_{\sf GR}$) is a bit more involved: roughly, the idea is to bucket the nodes by their degrees into logarithmically many buckets and then discards buckets that are small; the latter step is done to reduce the variance. Using the nodes in the remaining buckets and obtaining a random neighbor for each such node, an estimate for the average degree is output. The readers are referred to [9] for more details.

Theorem 4. *[9] If $r = O(\sqrt{n/L}\cdot{\rm poly}(\log n, 1/\epsilon))$, where L is a known lower bound on $d_{\rm avg}$, then $\hat{d}_{\sf GR}$ is a $(1+\epsilon)$-approximation to $d_{\rm avg}$.*

Our third estimator is based on the work on estimating sums of n variables when we can sample variables proportional to their values (i.e., weighted sampling). Motwani, Panigrahy, and Xu [18] obtained an algorithm that uses a clever combination of weighted sampling and uniform sampling to approximate the sum; their algorithm uses an optimal bound of $O(n^{1/3})$ samples. The main idea in their algorithm is to bucket the variables depending on their weight and use weighted sampling to estimate the contribution of the heavy buckets and uniform sampling to estimate the contribution of the light buckets; the readers are referred to [18] for the precise details. Note that in our ideal setting, the weighted sampling is given by $\mathcal{D}_{\rm d}$ and the uniform sampling given by $\mathcal{D}_{\rm u}$ and hence we can apply their algorithm to estimate m and hence $d_{\rm avg}$ (since we assumed n is known to the algorithm); we call this estimate $\hat{d}_{\sf MPX}$.

Theorem 5. *[18] If $r = O(n^{1/3}/\epsilon^{9/2}\cdot{\rm poly}(\log n, \log 1/\epsilon))$, then $\hat{d}_{\sf MPX}$ is a $(1+\epsilon)$-approximation to $d_{\rm avg}$.*

4.3 Collision-based estimators

Here we present two estimators that utilize the collisions into account. Informally, collision-based estimators are based on the "birthday paradox" and work by counting the number of collisions: with r independent uniform samples, the expected number of collisions is roughly r^2/n. Hence, the number of samples needed is roughly the square root of the quantity to be estimated. Such an idea was used to estimate the number of nodes in a network [11]; we show how to estimate the average degree using similar ideas. These estimators will serve as baselines for our purposes.

The first estimator is based on counting node collisions: we assume the ideal model where nodes are sampled with probability proportional to degree, i.e., according to $\mathcal{D}_{\rm d,0}$ and we assume that n is known. Let $x_1, \ldots, x_r$ be the nodes sampled. Let X_{ij}^u be the indicator variable for the event "ith and the jth nodes are both node u", i.e., "$x_i = u = x_j$." Then, the estimator for average degree is

$$\hat{d}_{\sf nCol} = \frac{r^2}{2n\sum_{j>i} X_{ij}^u/\deg(u)}.$$

By a Hoeffding bound, the following is immediate.

Theorem 6. *If $r = O(\sqrt{n d_{\rm avg}/d_{\min}}/\epsilon^2)$, then $\hat{d}_{\sf nCol}$ is a $(1+\epsilon)$-approximation to $d_{\rm avg}$.*

The second estimator is based on counting edge collisions; we assume the ideal model where edges are sampled uniformly at random and the total number of nodes n is known. Let $e_1, \ldots, e_r$ be the edges sampled. Let Y_{ij} be the indicator variable for the event "$e_i = e_j$". Then, the estimator for average degree is

$$\hat{d}_{\sf eCol} = \frac{r^2}{2n\sum_{j>i} Y_{ij}}.$$

By a Hoeffding bound, the following is immediate.

Theorem 7. *If $r = O(\sqrt{m}/\epsilon^2)$, then $\hat{d}_{\sf eCol}$ is a $(1+\epsilon)$-approximation to $d_{\rm avg}$.*

5. PROPERTIES OF OUR ESTIMATORS

In this section we prove the correctness of our estimators Smooth and Guess&Smooth. We first show the estimator Smooth, when provided with a "reasonable" value for c, one that is close to the average degree $d_{\rm avg}$ itself, does an excellent job of estimating $d_{\rm avg}$ to within a very accurate multiplicative factor. Theorem 8 characterizes the number of samples from $\mathcal{D}_{\rm d,c}$ needed to obtain a $(1 \pm \epsilon)$ multiplicative approximate of $d_{\rm avg}$. Proving concentration bounds on the Smooth estimator is nontrivial since it is a ratio of two random variables. Hence the result of Theorem 8 does not provide a complete trade off between the number of samples used and the bias (or variance) of the Smooth estimator. Theorem 11 uses an alternate method, based on ratio estimators, to bound the bias and variance of the Smooth estimator—this result is more instructive when the number of samples used is smaller than what Theorem 8 requires.

5.1 Sample complexity of Smooth

We first show that if we already have a reasonable approximation of d_{avg}, then we need to use only a constant number of samples from $\mathcal{D}_{\text{d},c}$ in order to get a $(1 \pm \epsilon)$-approximation to d_{avg}. Since the Smooth estimator is a ratio of two random variables, proving that it gives a $(1 \pm \epsilon)$-approximation with high probability requires more care.

THEOREM 8. *Let $0 < \epsilon \le 1/2$, $\delta \in (0,1)$, and $\alpha > 0$ be constants. Let $f = \frac{d_{\min}}{d_{\text{avg}}}$ be the ratio of minimum to average degree in this graph. If Algorithm* Smooth *is provided with the value $c = \alpha d_{\text{avg}}$ and the target number of samples $r = \max\left(1+\alpha, \frac{1+\alpha}{f+\alpha}\right)\frac{3\log(4/\delta)}{\epsilon^2}$, then, with probability $1-\delta$, the estimate $\hat{d}$ returned satisfies $(1-2\epsilon)d_{\text{avg}} \le \hat{d} \le (1+4\epsilon)d_{\text{avg}}$. In particular, $r = \max\left(\alpha, \frac{1}{\alpha}\right)\frac{6\log(4/\delta)}{\epsilon^2}$ samples are sufficient to get this approximation with probability $1-\delta$.*

PROOF. Let $D = nd_{\text{avg}}$ denote the total degree of graph G. Let $D' = D + nc$. Define $N = \sum_{s\in[1,r]} N_s$, where N_s is a random variable that takes on value $\frac{d_u}{d_u+c}$ with probability $\frac{d_u+c}{D'}$. Also define $Q = \sum_{s\in[1,r]} Q_s$ where Q_s takes the value $\frac{1}{d_u+c}$ with probability $\frac{d_u+c}{D'}$. Recall that the estimator Smooth is $\hat{d} = \frac{N}{Q}$ and $d_{\text{avg}} = \frac{E[N]}{E[Q]}$. We will show that N and Q are individually concentrated, the concentration of $\hat{d}$ will follow from this. We will achieve the former by using Bernstein's inequality, stated in Theorem 2.

We first start with analyzing N. The N_s are i.i.d. random variables, with expectation $E[N_s] = \frac{D}{D'}$ and second moment

$$E(N_s^2) = \sum_u \frac{d_u^2}{(d_u+c)^2}\frac{d_u+c}{D'} \le \frac{D}{D'}. \tag{1}$$

Also, the maximum deviation of each N_s can be bounded as

$$|N_s - E[N_s]| = \max_u \left|\frac{d_u}{d_u+c} - \frac{D}{D'}\right| \le 1,$$

as both $\frac{d_u}{d_u+c}, \frac{D}{D'} \in (0,1)$. Since N is the sum of r i.i.d. random variables, we have that $E[N] = \frac{rD}{D'}$ and $\text{var}(N) = r \cdot \text{var}(N_s) \le r\text{E}[N_s^2] \le r\frac{D}{D'}$. Plugging these values into Theorem 2, and choosing $t = \frac{\epsilon r D}{D'}$, it follows that

$$\Pr\left[|N - E[N]| > \frac{\epsilon r D}{D'}\right] \le 2\exp\left(-\frac{0.5(\frac{\epsilon r D}{D'})^2}{r\frac{D}{D'} + \frac{\epsilon r D}{3D'}}\right).$$

Choosing $r = \frac{3D'}{\epsilon^2 D}\log(4/\delta)$ is thus enough for the above probability to be less than $\delta/2$.

Similarly, in order to bound Q, we first start with the Q_s i.i.d. random variables. The expectation is $E[Q_s] = \frac{n}{D'}$ while the second moment satisfies

$$E[Q_s^2] = \sum_u \frac{1}{(d_u+c)^2}\frac{d_u+c}{D'} \le \frac{n}{D'(d_{\min}+c)}. \tag{2}$$

Thus, $E[Q] = \frac{rn}{D'}$ and the variance is at most

$$\text{var}(Q) = r\cdot\text{var}(Q_s) \le rE[Q_s^2] \le \frac{rn}{D'(d_{\min}+c)}.$$

Since $E[Q_s] = n/D' = 1/(d_{\text{avg}}+c)$, it holds that

$$|Q_s - E[Q_s]| \le \max_u \left|\frac{1}{d_u+c} - \frac{1}{d_{\text{avg}}+c}\right| \le \frac{1}{d_{\min}+c}.$$

Thus, again using Theorem 2, this time for the Q_s random variables, and setting $t = \frac{\epsilon r n}{D'}$, we have that

$$\Pr\left[|Q - E[Q]| > \frac{\epsilon r n}{D'}\right] \le 2\exp\left(-\frac{0.5(\frac{\epsilon r n}{D'})^2}{\frac{rn}{D'(d_{\min}+c)} + \frac{\epsilon r n}{3D'(d_{\min}+c)}}\right).$$

Again, choosing $r = \frac{3D'}{\epsilon^2 n(d_{\min}+c)}\log(4/\delta)$ is sufficient to set the above probability to be less than $\delta/2$.

Combining the two results, with $r = \max(\frac{D'}{D}, \frac{D'}{n(d_{\min}+c)})\frac{3\log\frac{4}{\delta}}{\epsilon^2}$, both N and Q are close to their corresponding expectations with probability $1-\delta$. Hence the estimate $\hat{d}$ satisfies

$$\frac{1-\epsilon}{1+\epsilon}\frac{E[N]}{E[Q]} \le \hat{d} \le \frac{1+\epsilon}{1-\epsilon}\frac{E[N]}{E[Q]}.$$

Since $d_{\text{avg}} = \frac{E[N]}{E[Q]}$ and $\epsilon \le 1/2$, we have that $\hat{d}$ is a $1 \pm 4\epsilon$ approximation to d_{avg}.

In order to bound the number of samples r, note that choosing $c = \alpha d_{\text{avg}}$ sets $D' = D(1+\alpha)$. It is sufficient then to set $r = \max\left(1+\alpha, \frac{1+\alpha}{f+\alpha}\right)\frac{3\log(4/\delta)}{\epsilon^2}$. □

An interesting case for the sample size guarantee occurs when the average degree d_{avg} is small.

REMARK 9. *Consider setting $c = 1$ in Theorem 8, i.e., setting $\alpha = \frac{1}{d_{\text{avg}}}$. Then it is sufficient to draw*

$$r = \max\left(1 + \frac{1}{d_{\text{avg}}}, 1 + d_{\text{avg}}\right)\frac{3\log(4/\delta)}{\epsilon^2}$$

samples from the distribution $\mathcal{D}_{\text{d},c}$ in order to estimate d_{avg}. In virtually all real world networks $d_{\text{avg}} = \Theta(1)$, and hence the number of samples required in practice is a modest $r = \Theta(\frac{\log(1/\delta)}{\epsilon^2})$.

Relation to existing lower bounds. The result of Theorem 8 uses $O(\frac{\log(1/\delta)}{\epsilon^2})$ samples from the $\mathcal{D}_{\text{d},c}$ model if $c = \Theta(d_{\text{avg}})$. It does not make any assumption on the type of network or the degree distribution. Given this strong bound, it is useful to contrast it with the existing theoretical lower bounds on estimating average degree. Feige [7] and Goldreich and Ron [9] show a lower bound of $\Omega(\sqrt{n})$ to approximate average degree to any constant factor in the *uniform* sampling model, even when the total number of nodes n is known. We use samples from $\mathcal{D}_{\text{d},c}$ instead, which is a strictly stronger model (but efficiently implementable in practice, as we show in Section 5.3). Motwani, Panigrahy, and Xu [18] present an $\Omega(n^{1/3})$ lower bound on the number of samples for estimating the sum (and hence average, since they assume n is known) of n elements using a general weighted sampling scheme. In their model elements can be sampled with probability proportional to $f(x)$ where x is the weight of the element (which is the degree in our case) and $f(\cdot)$ is any function. However, their lower bound specifically applies only to the case when the function f satisfies the condition $\frac{f(1)}{f(0)} = \Theta(1)$. This condition is violated for the sampling probabilities generated by our $\mathcal{D}_{\text{d},\Theta(d_{\text{avg}})}$ distribution.

5.2 Bias and variance of Smooth

In this section we show that the bias and variance of the Smooth estimator is small provided that parameter c is not far from the true average degree d_{avg}. We will use the following result bounding the bias and variance of ratio estimators.

THEOREM 10 (BAR-YOSSEF, GUREVICH [2]). *Suppose N and Q are two estimators such that $\frac{E[N]}{E[Q]} = I$ and that $\mathrm{var}(N)$ and $\mathrm{var}(Q)$ are finite and $Q > 0$. Let $N_1, \ldots, N_r$ be independent copies of N and $Q_1, \ldots, Q_r$ be independent copies of Q. Define $N(r) = \sum_i N_i$, $Q(r) = \sum_i Q_i$, and $RE_r = \frac{N(r)}{Q(r)}$. Then, the bias of RE_r is*

$$E[RE_r] - I = \frac{1}{r}\left(I\frac{\mathrm{var}(Q)}{E^2[Q]} + \frac{\mathrm{cov}(N,Q)}{E^2[Q]}\right) + o(1/r),$$

and the variance of RE_r is

$$\mathrm{var}(RE_r) = \frac{I^2}{r}\left(\frac{\mathrm{var}(N)}{E^2[N]} + \frac{\mathrm{var}(Q)}{E^2[Q]} - \frac{\mathrm{cov}(N,Q)}{E[N]E[Q]}\right) + o(1/r).$$

THEOREM 11. *For the estimator* Smooth*, using r samples, and using $c = \alpha d_{\mathrm{avg}}$, the bias of the estimator* Smooth *is at most $\dfrac{\alpha + 1/\alpha}{r} d_{\mathrm{avg}} + o(1/r)$ and its variance does not exceed $\dfrac{1+\alpha+1/\alpha}{r} d_{\mathrm{avg}}^2 + o(1/r)$.*

PROOF. We use the same notation as, and calculations from Theorem 8. In particular, for a specific s, recall that $E[N_s] = D/D'$ and $E[Q_s] = n/D'$, where $D = nd_{\mathrm{avg}}$ and $D' = n(d_{\mathrm{avg}} + c)$. Now observe that $Q_s \geq 0$ and $0 \leq N_s \leq 1$ and hence $E[N_sQ_s] \leq E[Q_s]$ holds. By definition $\mathrm{cov}(N_s, Q_s) = E[N_sQ_s] - E[N_s]E[Q_s]$, and therefore it follows that

$$\frac{\mathrm{cov}(N_s,Q_s)}{E^2[Q_s]} \leq \frac{E[Q_s] - E[N_s]E[Q_s]}{E^2[Q_s]} = \frac{1 - D/D'}{n/D'} = c.$$

From inequality (2) we know that $E[Q_s^2] \leq \frac{n}{D'(d_{\min}+c)}$ and thus

$$\frac{\mathrm{var}(Q_s)}{E^2[Q_s]} \leq \frac{\frac{n}{D'(d_{\min}+c)}}{(n/D')^2} - 1 = \frac{d_{\mathrm{avg}} + c}{d_{\min} + c} - 1 \leq \frac{d_{\mathrm{avg}}}{c}. \quad (3)$$

Substituting the bounds above into Theorem 10, we obtain

$$E\left[\frac{N(r)}{Q(r)}\right] - d_{\mathrm{avg}} - o(1/r) \leq \frac{1}{r}\left(\frac{d_{\mathrm{avg}}^2}{c} + c\right) = \frac{1}{r} d_{\mathrm{avg}}\left(\frac{1}{\alpha} + \alpha\right),$$

establishing our first claim.

From inequality (1) we also know that $E[N_s^2] \leq D/D'$ and therefore we have that

$$\frac{\mathrm{var}(N_s)}{E^2[N_s]} = \frac{E[N_s^2]}{E^2[N_s]} - 1 \leq \frac{D'}{D} - 1 = \frac{c}{d_{\mathrm{avg}}}.$$

Lastly, $-\mathrm{cov}(N_s, Q_s) = E[N_s]E[Q_s] - E[N_sQ_s] \leq E[N_s]E[Q_s]$ as both N_s and Q_s are non-negative. We conclude the proof by combining the last two inequalities and inequality (3) with Theorem 10 and observing that

$$\mathrm{var}\left(\frac{N(r)}{Q(r)}\right) - o(1/r) \leq \frac{d_{\mathrm{avg}}^2}{r}\left(\frac{c}{d_{\mathrm{avg}}} + \frac{d_{\mathrm{avg}}}{c} + 1\right). \quad \square$$

5.3 Random walk version of Smooth

In the ideal setting, we assume that we can sample from the distribution $\mathcal{D}_{\mathrm{d},c}$ for any given c. In reality however, obtaining a sample from the set of nodes of a large network is computationally expensive. On the other hand, if the social network can be accessed through an API, it is often easier to conduct a random walk on the network. In this section, we show how to efficiently sample from the $\mathcal{D}_{\mathrm{d},c}$ distribution by using a random walk on the original graph G. We will assume that the graph G is connected. We show a different random-walk than the Metropolis–Hastings walk [13] in order to implement $\mathcal{D}_{\mathrm{d},c}$ as the stationary distribution on G. We will bound the mixing time of this new random walk in terms of the walk on the original graph.

Random walk for Smooth. Let A be the adjacency matrix of the graph G and D be the diagonal matrix such that $D_{uu} = d_u$, the degree of node u. Let $\tilde{A} = A + cI$ and $\tilde{D} = D + cI$. We first claim that $\tilde{P} = \tilde{A}\tilde{D}^{-1}$ is a transition matrix of a random walk whose stationary probability is the distribution $\mathcal{D}_{\mathrm{d},c}$. Furthermore, if the original transition matrix $P = AD^{-1}$ is irreducible and aperiodic, so is $\tilde{A}\tilde{D}^{-1}$. Furthermore, as outlined in [2], since the number of steps the random walk for $\tilde{P}$ stays at node is a geometric random variable, we can easily simulate it by sampling from the appropriate geometric distribution (without making any more queries to G).

We conjecture that for the above random walk it is possible to bound the number of queries needed to reach ε close to stationary distribution, in terms of the mixing time of the original walk. For now, we show that this random walk mixes fast if the original does so, and if c is small.

LEMMA 12. *If τ be the mixing time of the simple random walk using P. Then, the mixing time of $\tilde{P}$ is bounded by $O\left((1 + \frac{c}{d_{\min}})^2\tau^2 \log n\right)$.*

PROOF. Recall that conductance of a set is defined as $\phi(S) = \frac{e(S,\bar{S})}{d(S)}$, where $e(S,\bar{S})$ is the number of edges from S to $\bar{S}$, the complement of S, and $d(S)$ is the total degree of the set S. The conductance of the graph G, denoted by say ϕ, is the minimum conductance over all sets S. The following relation between conductance of G and the mixing time τ is well-known (e.g., from [19])

$$\Theta\left(\frac{1}{\phi}\right) \leq \tau \leq \Theta\left(\phi^{-2}\log n\right).$$

Our strategy is to show that conductance of any set of nodes in the graph G does not change by much, and hence the mixing time does not either. Consider the graph $\tilde{G}$ whose adjacency matrix is $\tilde{A}$ defined above. The conductance $\tilde{\phi}(S)$ of any set of nodes is

$$\tilde{\phi}(S) = \frac{e(S,\bar{S})}{d(S) + c|S|},$$

since in $\tilde{G}$, there are c self loops on every node. Since $c|S| \leq \frac{cd(S)}{d_{\min}}$, we have that

$$\tilde{\phi}(S) = \frac{e(S,\bar{S})}{d(S) + c|S|} \geq \frac{e(S,\bar{S})}{d(S)(1 + c/d_{\min})} = \phi(S)(1+\frac{c}{d_{\min}})^{-1}.$$

where $\phi(S)$ is the conductance in the original graph G. Hence, the above conductance bound on mixing time, if ϕ and $\tilde{\phi}$ are the conductances of G and $\tilde{G}$, and τ and $\tilde{\tau}$ the mixing times, then $\tau \geq \Theta(\frac{1}{\phi})$ whereas $\tilde{\tau} \leq \Theta\left(\tilde{\phi}^{-2}\log n\right)$. Plugging in the relation between $\tilde{\phi} \geq \phi(1 + \frac{c}{d_{\min}})^{-1}$, we get the above statement. $\square$

Sample complexity. In order to show concentration properties of the random walk based sampler of Smooth, we will use the following result.

THEOREM 13 (LEZAUD [14]). *Let P be a transition matrix of a irreducible and reversible Markov chain on a finite set V, having a stationary distribution π. Let (P, π) be a irreducible and reversible Markov chain on a finite set V. Let $f : V \to \Re$ be such that $E_\pi[f] = 0$, $\|f\|_\infty \le 1$ and $0 < E_\pi[f^2] \le b^2$. Then, for any initial distribution q, any positive integer r and all $0 < \gamma \le 1$,*

$$\Pr_q \left[r^{-1} \sum_{i=1}^{r} f(X_i) \ge \gamma \right] \le e^{-\varepsilon(P)/5} S_q \exp\left(- \frac{n\gamma^2 \varepsilon(P)}{4b^2(1 + h(5\gamma/b^2))} \right),$$

where $\varepsilon(P) = 1 - \lambda_1(P)$, $\lambda_1(P)$ being the second largest eigenvalue of P, $S_q = \|q/\pi\|_2$ and

$$h(x) = \frac{1}{2}(\sqrt{1+x} - (1 - x/2)).$$

If $\gamma \ll b^2$ and $\varepsilon(P) \ll 1$, the bound is

$$(1 + o(1)) S_q \exp\left(- \frac{r\gamma^2 \varepsilon(p)}{4b^2(1 + o(1))} \right).$$

Using the expectation and second moment calculations from Theorem 8, we can derive the following result about the random walk based Smooth estimator. The proof is omitted.

COROLLARY 14. *Let $\varepsilon(\tilde{P})$ denote the gap between the first and second eigenvalues of the transition matrix $\tilde{P}$ of the augmented graph $\widetilde{G}$. Suppose $c = \alpha d_{\rm avg}$. If the random walk is assumed to start from the stationary distribution itself, then by using $r = \Theta\left(\frac{1}{\varepsilon(\tilde{P})} \max\left(\alpha, \frac{1}{\alpha}\right) \frac{\log(1/\delta)}{\epsilon^2} \right)$ samples, the* Smooth *estimate is a $(1 \pm \epsilon)$-approximation of $d_{\rm avg}$ with probability $1 - \delta$.*

5.4 Sample complexity of Guess&Smooth

In this section we prove that Algorithm 1 (Guess&Smooth) computes a constant factor approximation $\hat{d}$. We begin with showing that the probability of sampling a low degree node in $\mathcal{D}_{\rm d,c}$ is closely related to the ratio of $c/d_{\rm avg}$.

LEMMA 15. *Let $\beta > 0$ and $c = \beta d_{\rm avg}$, then it holds that*

$$\frac{\beta - 1}{1 + \beta} \le \Pr_{u \sim \mathcal{D}_{\rm d,c}} [\deg(u) \le c] \le \frac{2\beta}{1 + \beta}.$$

PROOF. Observe that $\sum_{\deg(u) \le c} (\deg(u) + c) \le n(c + c)$. Therefore it holds that $\Pr[\deg(u) \le c] \le \frac{2nc}{n(d_{\rm avg}+c)} = \frac{2\beta}{1+\beta}$.

Also note that from Markov's inequality with the uniform measure it follows that

$$|\{u : \deg(u) \ge \beta d_{\rm avg}\}| \le n/\beta.$$

Therefore $\sum_{\deg(u) \le c} (\deg(u) + c) \ge (n - n/\beta)c$ and we have that $\Pr[\deg(u) \le c] \ge \frac{(n - n/\beta)c}{n(d_{\rm avg}+c)} = \frac{\beta - 1}{1+\beta}$. □

THEOREM 16. *For $\hat{d}$ returned by Algorithm 1 (*Guess&Smooth*) with probability at least $1 - \delta$ it holds that $d_{\rm avg}/3 \le \hat{d} < 6d_{\rm avg}$.*

PROOF. Let $p(c) = \Pr[\deg(v) \le c]$. From the Hoeffding bound, Theorem 1, combined with the choice of r and the union bound it follows that with probability at $1 - \delta$ in all $\log_2(U/L)$ iterations it holds that

$$|N/r - p(c)| \le \epsilon_0. \quad (4)$$

If $c < d_{\rm avg}/3$, then from Equation (4) and from the r.h.s. of Lemma 15 it follows that $N/r \le p(c) + \epsilon_0 < 1/2 + \epsilon_0$, i.e., the algorithm never returns in these iterations. On the other hand, if $c \ge 3d_{\rm avg}$, then from Equation (4) and from the l.h.s. of Lemma 15 it follows that $N/r \ge p(c) - \epsilon_0 \ge 1/2 - \epsilon_0$, i.e., the algorithm always returns in these iterations. Therefore the lowest $\hat{d}$ the algorithm returns is $d_{\rm avg}/3$ and in the worst case it doubles c from slightly below $3d_{\rm avg}$ to almost $6d_{\rm avg}$ in the last iteration. □

Network	n	m	$d_{\rm avg}$
SKITTER	1696415	11095298	13.1
DBLP	317080	1049866	6.62
LIVEJOURNAL	3997962	34681189	17.34
ORKUT	3072441	117185083	76.28

Table 1: Description of datasets.

5.5 Sample complexity of Combined

The following theorem demonstrates that by bootstrapping Algorithm 2, that requires a rough estimate of the average degree, with the constant factor approximation of Algorithm 1, we are able to estimate $d_{\rm avg}$ with high precision and few samples in any scenario.

THEOREM 17. *Let $0 < \epsilon \le 1/2$ and $0 < \delta < 1$ and $L \le d_{\rm avg} \le U$. Then the estimate $\hat{d}$ returned by Algorithm 3 (*Combined*) satisfies $(1 - 2\epsilon)d_{\rm avg} \le \hat{d} \le (1 + 4\epsilon)d_{\rm avg}$ with probability at least $1 - \delta$. Furthermore, the number of samples used is $O\left(\frac{\log(1/\delta)}{\epsilon^2} + \log(\frac{U}{L}) \left(\log(\frac{1}{\delta}) + \log\log(\frac{U}{L}) \right) \right)$.*

PROOF. From Theorem 16 it follows that $d_{\rm avg}/6 \le \widetilde{d} \le 6d_{\rm avg}$ holds with probability at least $1 - \delta/2$. Assuming the latter, from Theorem 8 and the choice of $\widetilde{r}$ by Combined, with probability at least $1 - \delta/2$ we have that $(1 - 2\epsilon)d_{\rm avg} \le \hat{d} = \mathsf{Smooth}(G, \widetilde{r}, \widetilde{d}) \le (1 + 4\epsilon)d_{\rm avg}$. Thus the first claim follows from the union bound.

To count the total number of samples, observe that in addition to the $\widetilde{r} = O(\frac{\log(1/\delta)}{\epsilon^2})$ samples used by Smooth, Guess&Smooth executes at most $\log_2(U/L)$ iterations, each with $\Theta(\log(1/\delta) + \log\log(U/L))$ samples. □

6. EXPERIMENTS

In this section we compare the performance of the different degree estimators empirically using four datasets of undirected networks. All the datasets were obtained from SNAP (`http://snap.stanford.edu`). Table 1 summarizes the basic statistics of the datasets. While the datasets LIVEJOURNAL and ORKUT are explicit social networks, the dataset DBLP is the co-authorship network between 3.1 million authors of computer science research papers. SKITTER, on the other hand, represents an autonomous system (AS) network, where the edges denote which AS exchanges traffic with whom using the border gateway protocol.

Algorithms and metrics. We test the following baseline algorithms in our experiments: Feige's algorithm ($\hat{d}_{\mathsf{Feige}}$) that relies on uniform sampling, the variant of it by Goldreich and Ron [9], denoted by $\hat{d}_{\mathsf{GR}}$, the algorithm by Motwani, Panigrahy, and Xu [18] denoted by $\hat{d}_{\mathsf{MPX}}$ that utilizes $n^{1/2}$ samples but has better behavior (than the theoretically best algorithm in [18]) in terms of ϵ and is suggested as the one suitable for practical implementation. We also test two collision-based estimators, referred to in Section 4. Finally, we also examine the variants of our Smooth algorithms. Rather than run the guessing version Guess&Smooth, we run our Smooth estimator using a small number of different values of the parameter c. Since the degree of the networks examined were small enough constants, we set $c \in \{0, 1, 5, 50\}$ ($c = 0$ represents the usual random walk on the network).

We present two different set of plots characterizing the performance of the estimators. The first is the normalized *mean absolute error (MAE)*, measured as $|\hat{d} - d_{\rm avg}|/d_{\rm avg}$. For each sample size, we compute 100 different experiments, and then compute the average mean absolute error, averaged over these 100 experiments. In order to characterize the variability of the estimates, we also compute the 10% and 90% estimates, normalized by the ground truth, for each sample size, also empirically computed over these 100 experiments.

6.1 Results

Ideal setting. Our first experiments are in the ideal setting, where we sample nodes from the corresponding distributions directly. Figure 1 presents the results for MAE in this setting for four of the algorithms first, on each of the four datasets. The names `ideal.feige` and `ideal.gr` are self explanatory, `ideal.sr.1` denotes the Smooth with $c = 1$ and `ideal.mpx` the $\hat{d}_{\sf MPX}$ estimator. The first observation is that the number of samples required is indeed small. In order to get average MAE of less than 0.1, it is enough to work with number of samples as only 0.1% of the total number of nodes. The maximum sample size used in all the experiments was 2048, and the minimum (averaged) error in each case drops to less than 2%. Beyond this sample size, the algorithms become virtually indistinguishable. The algorithms Smooth (denoted by `ideal.sr.1`), $\hat{d}_{\sf Feige}$ and $\hat{d}_{\sf MPX}$ perform essentially similarly for the datasets DBLP, LiveJournal, and Orkut, with possibly a very slight edge to Smooth. The $\hat{d}_{\sf GR}$ algorithm performs worse than the others for smaller sample sizes, but its performance improves rapidly with same size. Note that the performance of $\hat{d}_{\sf Feige}$ is indeed much better than what Theorem 3 predicts. Also, it is indeed pleasantly surprising that $\hat{d}_{\sf GR}$ does perform reasonably accurately, since the algorithm relies on an exponential degree bucketing scheme that seems tailored to theoretical bounds, not practical implementations. The Smooth estimator, with $c = 1$, is the best, with a clear edge over the others in the Skitter dataset. This is possibly because of the more heavy-tailed nature of the autonomous system degree distribution, where a larger fraction of the total volume is tied up in large degree nodes than in social networks, and so sampling from the combined distribution $\mathcal{D}_{{\rm d},c}$ is beneficial.

The confidence intervals plots in Figure 2 have two lines per algorithm, corresponding to the (normalized) 10% and 90% estimates over the multiple iterations. Again, the confidence interval for Smooth is almost as tight, or strictly tighter, than the intervals generated from the other algorithms. The comparative advantage of Smooth is again best observed in the Skitter dataset.

Random walk-based implementation. Next, in Figures 3 and 4 we observe the empirical behavior of the same set of algorithms where the samples were taken from an appropriate random walk. For uniform sampling, we used the Metropolis–Hastings method with corresponding stationary distribution. In order to sample from $\mathcal{D}_{{\rm d},1}$, we used the walk described in Section 5.3. In each case, the first 100 nodes of the walk were discarded as a "burn-in" period, and were not accounted for in the sampling cost. Since our aim is to actually calibrate the performance against the number of queries made to the graph, we added in samples from consecutive steps rather than choosing a node only after every "mixing-time" intervals. This introduces higher correlation among the samples, and is reflective of the setting that Theorem 13 formulates. Using this setting, the Smooth algorithm comes out as a more definitive winner in the MAE error metric. The confidence intervals of the different algorithms are more or less comparable, again with Smooth being marginally better than the rest.

6.2 Comparison with collision-based algorithms

In Figure 5 we compare the collision based estimators $\hat{d}_{\sf eCol}$, $\hat{d}_{\sf nCol}$, and $\hat{d}_{\sf hit}$ (described in Section 4), along with our candidate Smooth. Note that these collision estimators are really trying to estimate the number of edges, which is a harder problem, and we assume that they all know n, the number of nodes. Therefore, it is not a surprise that these estimators are unsuitable for the task of estimating average degree, since, at the range of sample sizes that Smooth already provides 1% error-rate, we rarely observe any collisions among the samples.

6.3 Comparison among variants of Smooth

Finally, in the random walk setting we compare among 4 different variants of the Smooth algorithm, for $c \in \{0, 1, 5, 50\}$ in Figures 6 and 7. The performance of the Smooth variants, both in terms of the normalized MAE, as well as the confidence intervals, are more or less similar for for this range of c. It is important to note that, as mentioned in Section 5, $c = 0$ itself produces a good estimate. Note that there is a inherent tension here between the mixing time of the walk, and the appropriate value of c—increasing c to make the stationary distribution closer to the $\mathcal{D}_{{\rm d},\Theta(d_{\rm avg})}$ ideal distribution might also potentially increases the mixing time, and the resulting effect on the required number of samples for a target accuracy is unclear. But based on the performances in Figures 6 and 7, we suggest using $\mathcal{D}_{{\rm d},c}$ with a small constant c as an practically viable algorithm with small mixing-time and theoretically guaranteed accuracy.

7. CONCLUSIONS

In this paper we considered the natural problem of efficiently estimating the average degree of a network. We obtain estimators that provably use very few samples despite producing an arbitrary approximation to the average degree, outperforming other natural estimators for this problem. The experimental results on large real-world social networks confirm our theoretical findings. It will be interesting to see if neighbor of neighbors can be used to improve the performance further as was observed in social sampling [5]. It will also be interesting to see whether for directed graphs the in and out-degrees can be estimated using sampling distributions that are efficiently implementable by random walks.

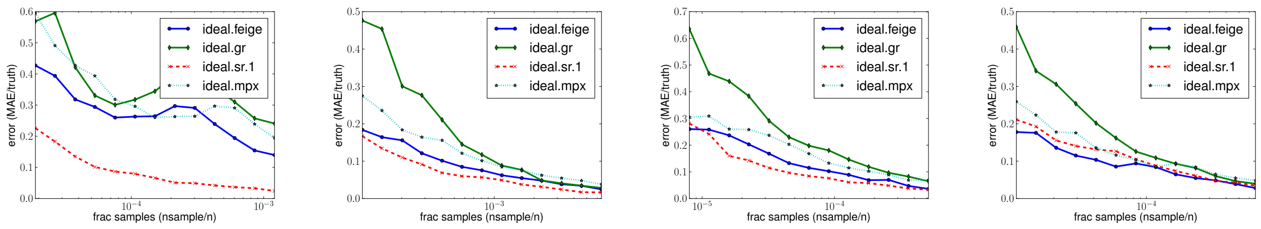

Figure 1: Normalized MAE in the ideal implementation of four algorithms for 1) Skitter **2)** DBLP **3)** LiveJournal **and 4)** Orkut **datasets. X-axis is sample size normalized by number of nodes.** `ideal.sr.1` **is** Smooth **with $c = 1$.**

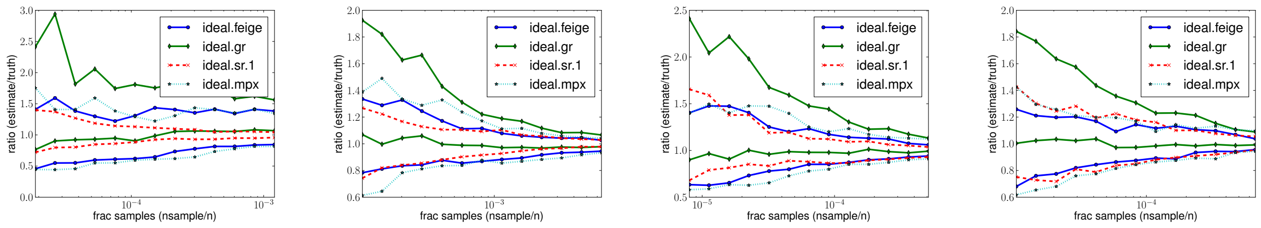

Figure 2: 10% and 90% confidence intervals in the ideal implementation of four algorithms for 1) Skitter **2)** DBLP **3)** LiveJournal **and 4)** Orkut **datasets. X-axis is sample size normalized by number of nodes.**

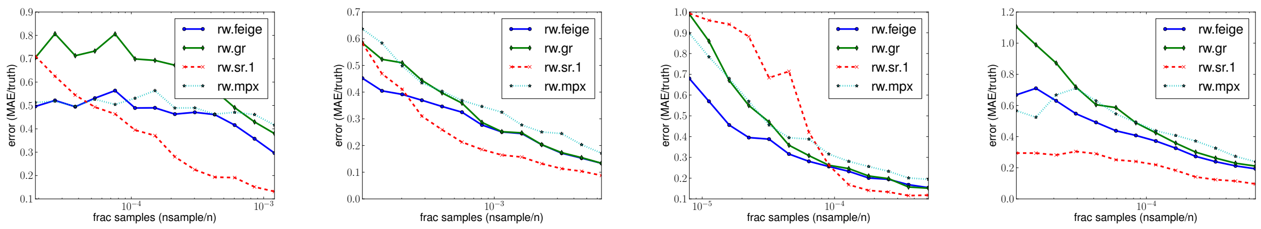

Figure 3: Normalized MAE in the random walk implementation of four algorithms for 1) Skitter **2)** DBLP **3)** LiveJournal **and 4)** Orkut **datasets. X-axis is sample size normalized by number of nodes.** `rw.sr.1` **is** Smooth **with random walk with $c = 1$.**

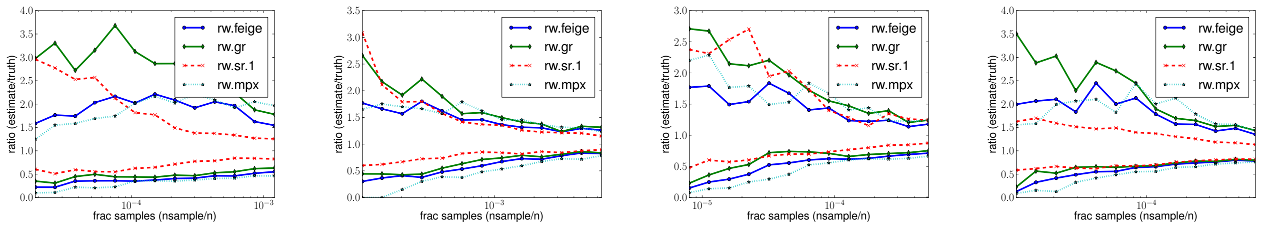

Figure 4: 10% and 90% confidence intervals in the random walk implementation of four algorithms for 1) Skitter **2)** DBLP **3)** LiveJournal **and 4)** Orkut **datasets. X-axis is sample size normalized by number of nodes.**

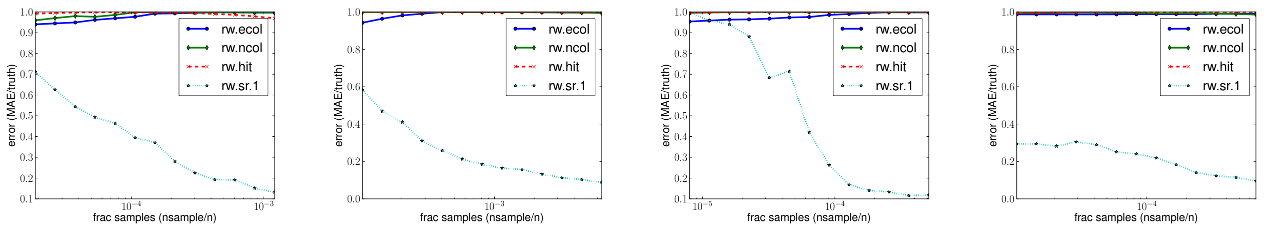

Figure 5: Normalized MAE in the random walk implementation of four collision based algorithms for 1) Skitter **2)** DBLP **3)** LiveJournal **and 4)** Orkut **datasets.** `rw.ecol` **is** $\hat{d}_{\mathsf{eCol}}$, `rw.ncol` **is** $\hat{d}_{\mathsf{nCol}}$ **and** `rw.hit` **is** $\hat{d}_{\mathsf{hit}}$. `rw.sr.1` **is** Smooth **with** $c = 1$**, using random walk.**

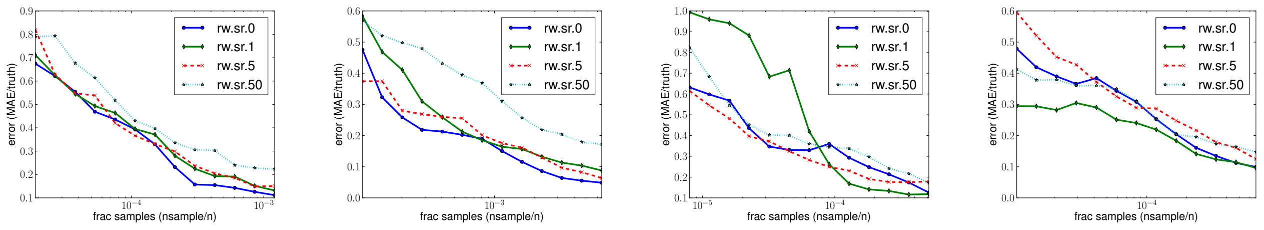

Figure 6: Normalized MAE in the random walk implementation of four Smooth **variants for 1)** Skitter **2)** DBLP **3)** LiveJournal **and 4)** Orkut **datasets.**

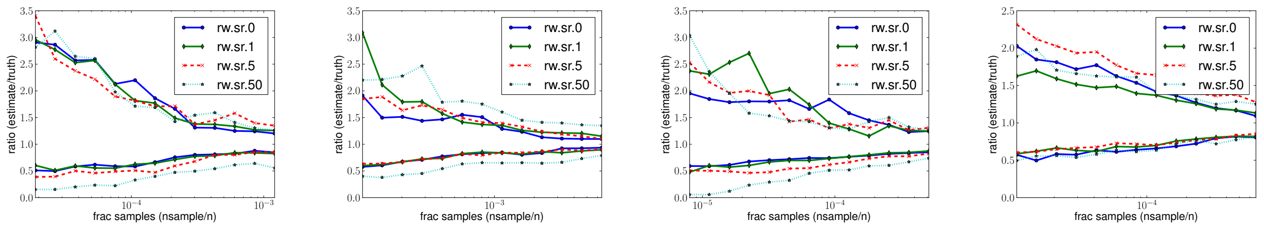

Figure 7: 10% and 90% confidence intervals of four random walk based Smooth **variants for 1)** Skitter **2)** DBLP **3)** LiveJournal **and 4)** Orkut **datasets.**

8. REFERENCES

[1] Z. Bar-Yossef and M. Gurevich. Random sampling from a search engine's index. *J. ACM*, 55(5), 2008.

[2] Z. Bar-Yossef and M. Gurevich. Efficient search engine measurements. *TWEB*, 5(4):18, 2011.

[3] K. Bharat and A. Broder. A technique for measuring the relative size and overlap of public web search engines. *Comput. Netw. ISDN Syst.*, 30(1-7):379–388, 1998.

[4] C. Cooper, T. Radzik, and Y. Siantos. Estimating network parameters using random walks. In *CASoN*, pages 33–40, 2012.

[5] A. Dasgupta, R. Kumar, and D. Sivakumar. Social sampling. In *KDD*, pages 235–243, 2012.

[6] D. P. Dubhash and A. Panconesi. *Concentration of Measure for the Analysis of Randomized Algorithms*. Cambridge University Press, 2009.

[7] U. Feige. On sums of independent random variables with unbounded variance and estimating the average degree in a graph. *SICOMP*, 35(4):964–984, 2006.

[8] M. Gjoka, M. Kurant, C. Butts, and A. Markopoulou. Walking in Facebook: A case study of unbiased sampling of OSNs. In *INFOCOM*, pages 1–9, 2010.

[9] O. Goldreich and D. Ron. Approximating average parameters of graphs. *RS&A*, 32(4):473–493, 2008.

[10] S. J. Hardiman and L. Katzir. Estimating clustering coefficients and size of social networks via random walk. In *WWW*, pages 539–550, 2013.

[11] L. Katzir, E. Liberty, and O. Somekh. Estimating sizes of social networks via biased sampling. In *WWW*, pages 597–606, 2011.

[12] M. Kurant, C. T. Butts, and A. Markopoulou. Graph size estimation. *CoRR*, abs/1210.0460, 2012.

[13] D. Levin, Y. Peres, and E. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Society, 2009.

[14] P. Lezaud. Chernoff-type bound for finite Markov chains. *AAP*, pages 849–867, 1998.

[15] T. H. McCormick, A. Moussa, J. Ruf, T. A. DiPrete, A. Gelman, J. Teitler, and T. Zheng. A practical guide to measuring social structure using indirectly observed network data. *Journal of Statistical Theory and Practice*, 7(1):120–132, 2013.

[16] T. H. McCormick, M. J. Salganik, and T. Zheng. How many people do you know?: Efficiently estimating personal network size. *JASA*, 105(489):59–70, 2010.

[17] T. H. McCormick and T. Zheng. A latent space representation of overdispersed relative propensity in "How many X's do you know" data. In *Conf. Proc. Joint Stat. Meet.*, 2010.

[18] R. Motwani, R. Panigrahy, and Y. Xu. Estimating sum by weighted sampling. In *ICALP*, pages 53–64, 2007.

[19] A. Sinclair. *Algorithms for Random Generation and Counting: A Markov Chain Approach*. Springer, 1993.

[20] S. Ye and F. Wu. Estimating the size of online social networks. In *SocialCom*, pages 169–176, 2010.