Provable and Practical Approximations for the Degree Distribution using Sublinear Graph Samples*†

Talya Eden
School of Computer Science, Tel Aviv University
Tel Aviv, Israel
talyaa01@gmail.com

Shweta Jain
University of California, Santa Cruz
Santa Cruz, CA, USA
sjain12@ucsc.edu

Ali Pinar
Sandia National Laboratories
Livermore, CA
apinar@sandia.gov

Dana Ron
School of Computer Science, Tel Aviv University
Tel Aviv, Israel
danaron@tau.ac.il

C. Seshadhri
University of California, Santa Cruz
Santa Cruz, CA
sesh@ucsc.edu

ABSTRACT

The degree distribution is one of the most fundamental properties used in the analysis of massive graphs. There is a large literature on *graph sampling*, where the goal is to estimate properties (especially the degree distribution) of a large graph through a small, random sample. Estimating the degree distribution of real-world graphs poses a significant challenge, due to their heavy-tailed nature and the large variance in degrees.

We design a new algorithm, SADDLES, for this problem, using recent mathematical techniques from the field of *sublinear algorithms*. The SADDLES algorithm gives provably accurate outputs for all values of the degree distribution. For the analysis, we define two fatness measures of the degree distribution, called the *h-index* and the *z-index*. We prove that SADDLES is sublinear in the graph size when these indices are large. A corollary of this result is a provably sublinear algorithm for any degree distribution bounded below by a power law.

We deploy our new algorithm on a variety of real datasets and demonstrate its excellent empirical behavior. In all instances, we get extremely accurate approximations for all values in the degree distribution by observing at most 1% of the vertices. This is a major improvement over the state-of-the-art sampling algorithms, which typically sample more than 10% of the vertices to give comparable results. We also observe that the h and z-indices of real graphs are large, validating our theoretical analysis.

ACM Reference Format:
Talya Eden, Shweta Jain, Ali Pinar, Dana Ron, and C. Seshadhri. 2018. Provable and Practical Approximations for the Degree Distribution using Sublinear Graph Samples. In *WWW 2018: The 2018 Web Conference, April 23–27, 2018, Lyon, France.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3178876.3186111

*Work funded by Sandia LDRD program, Israel Science Foundation grant No. 671/13, Azrieli Fellowship, and NSF TRIPODS grant CCF-1740850. Part of this work was initiated at the Simons Institute Semester on Algorithms and Uncertainty.
†Both Talya Eden and Shweta Jain contributed equally to this work, and are joint first authors of this work.

WWW 2018, April 23–27, 2018, Lyon, France

ACM ISBN 978-1-4503-5639-8/18/04.
https://doi.org/10.1145/3178876.3186111

1 INTRODUCTION

In domains as diverse as social sciences, biology, physics, cybersecurity, graphs are used to represent entities and the relationships between them. This has led to the explosive growth of network science as a discipline over the past decade. One of the hallmarks of network science is the occurrence of specific graph properties that are common to varying domains, such as heavy tailed degree distributions, large clustering coefficients, and small-world behavior. The degree distribution is especially significant, since the early days of modern network science [7, 8, 21].

Given an undirected graph G, the degree distribution (or technically, histogram) is the sequence of numbers $n(1), n(2), \ldots$, where $n(d)$ is the number of vertices of degree d. In almost all real-world scenarios, the average degree is small, but the variance (and higher moments) is large. Even for relatively large d, $n(d)$ is still non-zero, and $n(d)$ typically has a smooth non-increasing behavior. In Fig. 1, we see the typical degree distribution behavior. The average degree in a Google web network is less than 10, but the maximum degree is more than 5000. There are also numerous vertices with all intermediate degrees. This is referred to as a "heavy tailed" distribution. The degree distribution, especially the tail, is of significant relevance to modeling networks, determining their resilience, spread of information, and for algorithmics [6, 9, 13, 16, 34–37, 43].

With full access to G, the degree distribution can be computed in linear time, by simply determining the degree of each vertex. Yet in many scenarios, we only have *partial* access to the graph, provided through some graph samples. A naive extrapolation of the degree distribution can result in biased results. The seminal research paper of Faloutsos et al. claimed a power law in the degree distribution on the Internet [21]. This degree distribution was deduced by measuring a power law distribution in the graph sample generated by a collection of traceroute queries on a set of routers. Unfortunately, it was mathematically and empirically proven that traceroute responses can have a power law *even if the true network does not* [1, 11, 28, 38]. In general, a direct extrapolation of the

degree distribution from a graph subsample is not valid for the underlying graph. This leads to the primary question behind our work.

How can we provably and practically estimate the degree distribution without seeing the entire graph?

There is a rich literature in statistics, data mining, and physics on estimating graph properties (especially the degree distribution) using a small subsample [2, 3, 5, 17, 29, 31, 32, 40, 45, 46]. Nonetheless, there is no provable algorithm for the entire degree distribution, with a formal analysis on when it is sublinear in the number of vertices. Furthermore, most empirical studies typically sample 10-30% of the vertices for reasonable estimates.

1.1 Problem description

We focus on the *complementary cumulative degree histogram* (often called the cumulative degree distribution) or *ccdh* of G. This is the sequence $\{N(d)\}$, where $N(d) = \sum_{r \geq d} n(r)$ is the number of vertices of degree at least d. The ccdh is typically used for fitting distributions, since it averages out noise and is monotonic [12]. Our aim is to get an accurate bicriteria approximation to the ccdh of G, at all values of d.

Definition 1.1. The sequence $\{\widetilde{N}(d)\}$ is an $(\varepsilon, \varepsilon)$-estimate of the ccdh if $\forall d$, $(1-\varepsilon)N((1+\varepsilon)d) \leq \widetilde{N}(d) \leq (1+\varepsilon)N((1-\varepsilon)d)$.

Computing an $(\varepsilon, \varepsilon)$-estimate is significantly harder than approximating the ccdh using standard distribution measures. Statistical measures, such as the KS-distance, χ^2, ℓ_p-norms, etc. tend to ignore the tail, since (in terms of probability mass) it is a negligible portion of the distribution. An $(\varepsilon, \varepsilon)$-estimate is accurate for all d.

The query model: A formal approach requires specifying a *query model* for accessing G. We look to the subfields of property testing and sublinear algorithms within theoretical computer science for such models [23, 24]. Consider the following three kinds of *queries*.

- Vertex queries: acquire a uniform random vertex $v \in V$.
- Neighbor queries: given $v \in V$, acquire a uniform random neighbor u of V.
- Degree queries: given $v \in V$, acquire the degree d_v.

An algorithm is only allowed to make these queries to process the input. It has to make some number of queries, and finally produce an output. We discuss two query models, and give results for both.

The Standard Model (SM) All queries allowed: This is the standard model in numerous sublinear algorithms results [19, 20, 23–25]. Furthermore, most papers on graph sampling implicitly use this model for generating subsamples. Indeed, any method involving crawling from a random set of vertices and collecting degrees is in the SM. This model is the primary setting for our work, and allows for comparison with rich body of graph sampling algorithms. It is worth noting that in the SM, one can determine the entire degree distribution in $O(n \log n)$ queries (the extra $\log n$ factor comes from the coupon collector bound of finding all the vertices through uniform sampling). Thus, it makes sense to express the number of queries made by an algorithm as a fraction of n. Alternately, the number of queries is basically the number of vertices encountered by the algorithm. Thus, a sublinear algorithm makes $o(n)$ queries.

The Hidden Degrees Model (HDM) Vertex and neighbor queries allowed, not degree queries: This is a substantially weaker model. In numerous cybersecurity and network monitoring settings, an algorithm cannot query for degrees, and has to infer them indirectly. Observe that this model is significantly harder than the SM. It takes $O((m+n)\log n)$ to determine all the degrees, since one has to at least visit all the edges to find degrees exactly. In this model, we express the number of queries as a fraction of m.

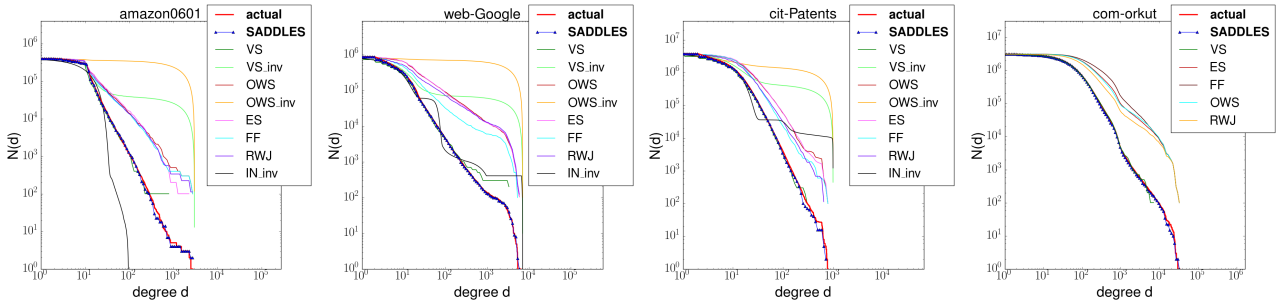
Regarding uniform random vertex queries: This is a fairly powerful query, that may not be realizable in all situations. Indeed, Chierichetti et al. explicitly study this problem in social networks and design (non-trivial) algorithms for sampling uniform random vertices [10]. In a previous work, Dasgupta, Kumar, and Sarlos study algorithms for estimating average degree when only random walks are possible [14]. Despite this power, we believe that SM is a good testbed for understanding *when* a small sample of a graph provably gives properties of the whole. Furthermore, in the context of graph sampling, access to uniform random vertices is commonly (implicitly) assumed [5, 17, 29, 31, 39, 40, 46]. The vast majority of experiments conducted often use uniform random vertices.

As a future direction, we believe it is important to investigate sampling models without random vertex queries.

1.2 Our contributions

Our main theoretical result is a new sampling algorithm, the Sublinear Approximations for Degree Distributions Leveraging Edge Samples, or SADDLES. This algorithm provably provides $(\varepsilon, \varepsilon)$-approximations for the ccdh. We show how to design SADDLES under both the SM and the HDM. We apply SADDLES on a variety of real datasets and demonstrate its ability to accurately approximate the ccdh with a tiny sample of the graph.

- **Sampling algorithm for estimating ccdh:** Our algorithm combines a number of techniques in random sampling to get $(\varepsilon, \varepsilon)$-estimates for the ccdh. A crucial component is an application of an edge simulation technique, first devised by Eden et al. in the context of triangle counting [19, 20]. This (theoretical) technique shows how to get a collection of weakly correlated uniform random edges from independent uniform vertices. SADDLES employs a weighting scheme on top of this method to estimate the ccdh.
- **Heavy tails leads to sublinear algorithms:** The challenge in analyzing SADDLES is in finding parameters of the ccdh that allow for sublinear query complexity. To that end, we discuss two parameters that measure "heaviness" of the distribution tail: the classic h-index and a newly defined z-index. We prove that the query complexity of SADDLES is sublinear (for both models) whenever these indices are large.
- **Excellent empirical behavior:** We deploy an implementation of SADDLES on a collection of large real-world graphs. In all instances, we achieve extremely accurate estimates for the entire ccdh by sampling at most 1% of the vertices of the graph. Refer to Fig. 1. Observe how SADDLES tracks various jumps in the ccdh, for all graphs in Fig. 1.
- **Comparison with existing sampling methods:** A number of graph sampling methods have been proposed in practice, such as vertex sampling (VS), snowball sampling (OWS), forest-fire sampling (FF), induced graph sampling (IN), random walk (RWJ),



(a) amazon0601 copurchase network (b) web-Google web network (c) cit-Patents citation network (d) com-orkut social network

Figure 1: The output of SADDLES **on a collection of networks:** amazon0601 **(403K vertices, 4.9M edges),** web-Google **(870K vertices, 4.3M edges),** cit-Patents **(3.8M vertices, 16M edges),** com-orkut **social network (3M vertices, 117M edges).** SADDLES **samples** 1% **of the vertices and gives accurate results for the entire (cumulative) degree distribution. For comparison, we show the output of a number of sampling algorithms from past work, each run with the same number of samples. (Because of the size of** com-Orkut**, methods involving optimization [46] fail to produce an estimate in reasonable time.)**

edge sampling (ES) [5, 17, 29, 31, 39, 40, 46]. A recent work of Zhang et al. explicitly addresses biases in these sampling methods, and fixes them using optimization techniques [46]. We run head-to-head comparisons with all these sampling methods, and demonstrate the SADDLES gives significantly better practical performance. Fig. 1 shows the output of all these sampling methods with a total sample size of 1% of the vertices. Observe how across the board, the methods make erroneous estimates for most of the degree distribution. The errors are also very large, for all the methods. This is consistent with previous work, where methods sample more than 10% of the number of vertices.

1.3 Theoretical results in detail

Our main theoretical result is a new sampling algorithm, the Sublinear Approximations for Degree Distributions Leveraging Edge Samples, or SADDLES.

We first demonstrate our results for power law degree distributions [7, 8, 21]. Statistical fitting procedures suggest they occur to some extent in the real-world, albeit with much noise [12]. The classic power law degree distribution sets $n(d) \propto 1/d^{\gamma}$, where γ is typically in $[2, 3]$. We build on this to define a power law lower bound.

Definition 1.2. Fix $\gamma > 2$. A degree distribution is bounded below by a power law with exponent γ, if the ccdh satisfies the following property. There exists a constant $\tau > 0$ such that for all d, $N(d) \geq \lfloor \tau n/d^{\gamma-1} \rfloor$.

The following is a corollary of our main result. For convenience, we will suppress query complexity dependencies on ε and $\log n$ factors, using $\widetilde{O}(\cdot)$.

Theorem 1.3. *Suppose the degree distribution of G is bounded below by a power law with exponent γ. Let the average degree be denoted by $\overline{d}$. For any $\varepsilon > 0$, the* SADDLES *algorithm outputs (with high probability) an $(\varepsilon, \varepsilon)$-approximation to the ccdh and makes the following number of queries. For SM: $\widetilde{O}(n^{1-\frac{1}{\gamma}} + n^{1-\frac{1}{\gamma-1}}\overline{d})$. For HDM: $\widetilde{O}(n^{1-\frac{1}{2(\gamma-1)}}\overline{d})$*

In most real-world instances, the average degree $\overline{d}$ is typically constant. Thus, the complexities above are strongly sublinear. For example, when $\gamma = 2$, we get $\widetilde{O}(n^{1/2})$ for both models. When $\gamma = 3$, we get $\widetilde{O}(n^{2/3})$ and $\widetilde{O}(n^{3/4})$.

Our main result is more nuanced, and holds for all degree distributions. If the ccdh has a heavy tail, we expect $N(d)$ to be reasonably large even for large values of d. We describe two formalisms of this notion, through *fatness indices*.

Definition 1.4. The *h-index* of the degree distribution is the largest d such that there are at least d vertices of degree at least d.

This is the exact analogy of the bibliometric h-index [27]. As we show in the §2.1, h can be approximated by $\min_d (d + N(d))/2$. A more stringent index is obtained by replacing the arithmetic mean by the (smaller) geometric mean.

Definition 1.5. The *z-index* of the degree distribution is $z = \min_{d:N(d)>0} \sqrt{d \cdot N(d)}$.

Our main theorem asserts that large h and z indices lead to a sublinear algorithm for degree distribution estimation. Theorem 1.3 is a direct corollary obtained by plugging in values of the indices for power laws.

Theorem 1.6. *For any $\varepsilon > 0$, the* SADDLES *algorithm outputs (with high probability) an $(\varepsilon, \varepsilon)$-approximation to the ccdh, and makes the following number of queries. For SM: $\widetilde{O}(n/h + m/z^2)$. For HDM: $\widetilde{O}(m/z)$.*

1.4 Challenges and Main Idea

The heavy-tailed behavior of the real degree distribution poses the primary challenge to computing $(\varepsilon, \varepsilon)$-estimates to the ccdh. Sampling uniform random vertices is inefficient when $N(d)$ is small. A random neighbor of a random vertex is more likely to be a high degree vertex. This is the idea behind methods like OWS, FF, RWJ graph sample-and-hold, etc. [5, 17, 29, 31, 39, 40, 46]. But these lead to biased samples, since vertices with the same degree may be picked with differing probabilities.

A direct extrapolation/scaling of the degrees in the observed graph does not provide an accurate estimate. Our experiments show that existing methods always miss the head or the tail. From a mathematical standpoint, the vast majority of existing results tend to analyze the KS-statistic, or some ℓ_p-norm. As we mentioned earlier, this does not work well for measuring the quality of the estimate at all scales. As shown by our experiments, none of these methods give accurate estimate for the entire ccdh with less than 5% of the vertices.

The main innovation in SADDLES comes through the use of a recent theoretical technique to simulate edge samples through vertex samples [19, 20]. The sampling of edges occurs through two stages. In the first stage, the algorithm samples a set of r vertices and sets up a distribution over the sampled vertices such that any edge adjacent to a sampled vertex may be sampled with uniform probability. In the second stage, it samples q edges from this distribution. While a single edge is uniform random, the set of edges are correlated.

For a given d, we define a weight function on the edges, such that the total weight is exactly $N(d)$. SADDLES estimates the total weight by scaling up the average weight on a random sample of edges, generated as discussed above. The difficulty in the analysis is the correlation between the edges. Our main insight is that if the degree distribution has a fat tail, this correlation can be contained even for sublinear r and q. Formally, this is achieved by relating the concentration behavior of the average weight of the sample to the h and z-indices. The final algorithm combines this idea with vertex sampling to get accurate estimates for all d.

The HDM is dealt with using birthday paradox techniques formalized by Ron and Tsur [42]. It is possible to estimate the degree d_v using $O(\sqrt{d_v})$ neighbor queries. But this adds overhead to the algorithm, especially for estimating the ccdh at the tail. As discussed earlier, we need methods that bias towards higher degrees, but this significantly adds to the query cost of actually estimating the degrees.

1.5 Related Work

There is a rich body of literature on generating a graph sample that reveals graph properties of the larger "true" graph. We do not attempt to fully survey this literature, and only refer to results directly related to our work. The works of Leskovec & Faloutsos [31], Maiya & Berger-Wolf [32], and Ahmed, Neville, & Kompella [2, 5] provide excellent surveys of multiple sampling methods.

There are a number of sampling methods based on random crawls: forest-fire [31], snowball sampling [32], and expansion sampling [31]. As has been detailed in previous work, these methods tend to bias certain parts of the network, which can be exploited for more accurate estimates of various properties [31, 32, 40]. A series of papers by Ahmed, Neville, and Kompella [2–5] have proposed alternate sampling methods that combine random vertices and edges to get better representative samples.

All these results aim to capture numerous properties of the graph, using a single graph sample. Nonetheless, there is much previous work focused on the degree distribution. Ribiero and Towsley [40] and Stumpf and Wiuf [45] specifically study degree distributions. Ribiero and Towsley [40] do detailed analysis on degree distribution estimates (they also look at the ccdh) for a variety of these sampling methods. Their empirical results show significant errors either at the head or the tail. We note that almost all these results end up sampling up to 20% of the graph to estimate the degree distribution.

Zhang et al. observe that the degree distribution of numerous sampling methods is a random linear projection of the true distribution [46]. They attempt to invert this (ill-conditioned) linear problem, to correct the biases. This leads to improvement in the estimate, but the empirical studies typically sample more than 10% of the vertices for good estimates.

Some methods try to match the shape/family of the distribution, rather than estimate it as a whole [45]. Thus, statistical methods can be used to estimate parameters of the distribution. But it is reasonably well-established that real-world degree distributions are rarely pure power laws in most instances [12]. Indeed, fitting a power law is rather challenging and naive regression fits on log-log plots are erroneous, as results of Clauset-Shalizi-Newman showed [12].

The subfield of *property testing and sublinear algorithms for sparse graphs* within theoretical computer science can be thought of as a formalization of graph sampling to estimate properties. Indeed, our description of the main problem follows this language. There is a very rich body of mathematical work in this area (refer to Ron's survey [41]). Practical applications of graph property testing are quite rare, and we are only aware of one previous work on applications for finding dense cores in router networks [26]. The specific problem of estimating the average degree (or the total number of edges) was studied by Feige [22] and Goldreich-Ron [24]. Gonen et al. and Eden et al. focus on the problem of estimating higher moments of the degree distribution [20, 25]. One of the main techniques we use of simulating edge queries was developed in sublinear algorithms results of Eden et al. [19, 20] in the context of triangle counting and degree moment estimation. We stress that all these results are purely theoretical, and their practicality is by no means obvious.

On the practical side, Dasgupta, Kumar, and Sarlos study average degree estimation in real graphs, and develop alternate algorithms [14]. They require the graph to have low mixing time and demonstrate that the algorithm has excellent behavior in practice (compared to implementations of Feige's and the Goldreich-Ron algorithm [22, 24]). Dasgupta et al. note that sampling uniform random vertices is not possible in many settings, and thus they consider a significantly weaker setting than SM or HDM. Chierichetti et al. focus on sampling uniform random vertices, using only a small set of seed vertices and neighbor queries [10].

We note that there is a large body of work on sampling graphs from a stream [33]. This is quite different from our setting, since a streaming algorithm observes every edge at least once. The specific problem of estimating the degree distribution at all scales was considered by Simpson et al. [44]. They observe many of the challenges we mentioned earlier: the difficulty of estimating the tail accurately, finding vertices at all degree scales, and combining estimates from the head and the tail.

2 PRELIMINARIES

We say that the input graph G has n vertices and m edges and $m \geq n$ (since isolated vertices are not relevant here). For any vertex

v, let $\Gamma(v)$ be the neighborhood of v, and d_v be the degree. As mentioned earlier, $n(d)$ is the number of vertices of degree d and $N(d) = \sum_{r \geq d} n(r)$ is the ccdh at d. We use "u.a.r." as a shorthand for "uniform at random". We stress that the all mention of probability and error is with respect to the randomness of the sampling algorithm. There is no stochastic assumption on the input graph G. We use the shorthand $A \in (1 \pm \alpha)B$ for $A \in [(1-\alpha)B, (1+\alpha)B]$. We will apply the following (rescaled) Chernoff bound.

Theorem 2.1. *[Theorem 1 in [15]] Let $X_1, X_2, \ldots, X_k$ be a sequence of iid random variables with expectation μ. Furthermore, $X_i \in [0, B]$.*

- *For $\varepsilon < 1$, $\Pr[|\sum_{i=1}^{k} X_i - \mu k| \geq \varepsilon \mu k] \leq 2\exp(-\varepsilon^2 \mu k/3B)$.*
- *For $t \geq 2e\mu$, $\Pr[\sum_{i=1}^{k} X_i \geq tk] \leq 2^{-tk/B}$.*

2.1 More on Fatness indices

(All proofs in this section are fairly straightforward calculations, and are hence omitted in this version.) The following characterization of the h-index will be useful for analysis. Since $(d + N(d))/2 \leq \max(d, N(d)) \leq d + N(d)$, this proves that $\min_d (d + N(d))/2$ is a 2-factor approximation to the h-index.

Lemma 2.2. $\min_d \max(d, N(d)) \in \{h, h+1\}$

The h-index does not measure d vs $N(d)$ at different scales, and a large h-index only ensures that there are "enough" high-degree vertices. For instance, the h-index does not distinguish between two different distributions whose ccdh N_1 and N_2 are such that $N_1(100) = 100$ and $N_1(d) = 0$ for $d > 100$, and $N_2(100,000) = 100$ and $N_2(d) = 100$ for all other values of $d \geq 100$. The h-index in both these cases is 100.

The h and z-indices are related to each other.

Claim 2.3. $\sqrt{h} \leq z \leq h$.

To give some intuition about these indices, we compute the h and z indices for power laws. The classic power law degree distribution sets $n(d) \propto 1/d^{\gamma}$, where γ is typically in $[2, 3]$.

Claim 2.4. *If a degree distribution is bounded below by a power law with exponent γ, then $h = \Omega(n^{\frac{1}{\gamma}})$ and $z = \Omega(n^{\frac{1}{2(\gamma-1)}})$.*

Plugging in values, for $\gamma = 2$, both h and z are $\Omega(\sqrt{n})$. For $\gamma = 3$, $h = \Theta(n^{1/3})$ and $z = \Theta(n^{1/4})$.

2.2 Simulating degree queries for HDM

The HDM does not allow for querying the degree d_v of a vertex v. Nonetheless, it is possible to get accurate estimates of d_v using the birthday paradox argument, as formalized by Ron and Tsur [42], by sampling u.a.r. neighbors (with replacement) of v until the same vertex is seen twice. If this happens after t samples, t^2 is a constant factor approximation for d_v. The following can be obtained directly from Theorem 3.1 of [42]. (Details omitted in this version.)

Corollary 2.5. *There is an algorithm* DEG *that takes as input a vertex v, and has the following properties:*

- *For all v: with probability $> 1 - 1/n^3$, the output* DEG(v) *is in $(1 \pm \varepsilon/10)d_v$.*
- *The expected running time and query complexity of* DEG(v) *is $O(\varepsilon^{-2}\sqrt{d_v}\log n)$.*

We will assume that invocations of DEG with the same arguments use the same sequence of random bits. Alternately, imagine that a call to DEG(v) stores the output, so subsequent calls output the same value. For the sake of analysis, it is convenient to imagine that DEG(v) is called once for all vertices v, and these results are stored.

Definition 2.6. The output DEG(v) is denoted by $\hat{d_v}$. The random bits used in all calls to DEG is collectively denoted Λ. (Thus, Λ completely specifies all the values $\{\hat{d_v}\}$.) We say Λ is *good* if $\forall v \in V$, $\hat{d_v} \in (1 \pm \varepsilon/10)d_v$.

The following is a simple consequence of conditional probabilities (proof omitted in this version).

Claim 2.7. *Consider any event $\mathcal{A}$, such that for any good Λ, $\Pr[\mathcal{A}|\Lambda] \geq p$. Then $\Pr[\mathcal{A}] \geq p - 1/n^2$.*

For any fixed Λ, we set $\widehat{N_\Lambda}(d)$ to be $|\{v|\hat{d_v} \geq d\}|$. We will perform the analysis of SADDLES with respect to the $\widehat{N_\Lambda}$-values. (Proof is a straightforward calculation, and omitted in this version.)

Claim 2.8. *Suppose Λ is good. For all v, $\widehat{N_\Lambda}(v) \in [N((1+\varepsilon/9)d), N((1-\varepsilon/9)d)]$.*

3 THE MAIN RESULT AND SADDLES

We begin by stating the main result on the SADDLES procedure. Note that D refers to a set of degrees, for which we desire an approximation to $N(d)$.

Theorem 3.1. *There exists an algorithm* SADDLES *with the following properties. Let c be a sufficiently large constant. Fix any $\varepsilon > 0, \delta > 0$. Suppose that the parameters of* SADDLES *satisfy the following conditions: $r \geq c\varepsilon^{-2}n/h$, $q \geq c\varepsilon^{-2}m/z^2$, $\ell \geq c\log(n/\delta)$, $\tau \geq c\varepsilon^{-2}$.*

Then with probability at least $1 - \delta$, for all $d \in D$, SADDLES *outputs an $(\varepsilon, \varepsilon)$-approximation of $N(d)$.*

The expected number of queries made depends on the model, and is independent of the size of D.

- *SM: $O((n/h + m/z^2)(\varepsilon^{-2}\log(n/\delta)))$.*
- *HDM: $O((m/z)(\varepsilon^{-4}\log^2(n/\delta)))$.*

Ignoring constant factors and assuming $m = O(n)$, asymptotically increasing h and z-indices lead to an algorithm with sublinear query complexity.

The same algorithmic structure is used for the SM and the HDM. The only difference is the use the algorithm of Corollary 2.5 to estimate degrees in the HDM, while the degrees are directly available in SM.

The core theoretical bound: The central technical bound deals with the properties of each individual estimate $\widetilde{N}(d)[t]$.

Theorem 3.2. *Suppose $r \geq c\varepsilon^{-2}n/h$, $q \geq c\varepsilon^{-2}m/z^2$, $\tau = c\varepsilon^{-2}$. Then, for all $d \in D$, with probability $\geq 5/6$, $\widetilde{N}(d)[t] \in [(1-\varepsilon/2)N((1+\varepsilon/2)d), (1+\varepsilon/2)N((1-\varepsilon/2)d]$.*

The proof of this theorem is the main part of our analysis, which appears in the next section. The error/accuracy bound of Theorem 3.1 can be proved through a straightforward application of "boosting through medians". For space constraints, we leave the proof of error bound and query complexity for a longer version of this paper available on arxiv [18].

Algorithm 1: SADDLES(D, r, q, ℓ, τ)
Inputs:
D: set of degrees for which N(d) is to be computed
r: budget for vertex samples
q: budget for edge samples
ℓ: boosting parameter
τ: cutoff for vertex sampling
Output:
$\{N'(d)\}$: estimated $\{N(d)\}$

1 For $t = 1, \ldots, \ell$:
2 For $i = 1, \ldots, r$:
3 Select u.a.r. vertex v and add it to multiset R.
4 In HDM, call DEG(v) to get $\hat{d_v}$. In SM, set $\hat{d_v}$ to d_v.
5 For $d \in D$:
6 If $\hat{d_v} \geq d$, set $X_{id} = 1$. Else, $X_{id} = 0$.
7 Let $\hat{d_R} = \sum_{v \in R} \hat{d_v}$ and $\mathcal{D}$ denote the distribution over R where $v \in R$ is selected with probability $\hat{d_v}/\hat{d_R}$.
8 For $i = 1, \ldots, q$:
9 Sample $v \sim \mathcal{D}$.
10 Pick u.a.r. neighbor u of v.
11 In HDM, call DEG(u) to get $\hat{d_u}$. In SM, set $\hat{d_u}$ to d_u.
12 For $d \in D$:
13 If $\hat{d_u} \geq d$, set $Y_{id} = 1/\hat{d_u}$. Else, set $Y_{id} = 0$.
14 For $d \in D$:
15 If $\sum_{i \leq r} X_{id} \geq \tau$:
16 $\widetilde{N}(d)[t] = \frac{n}{r} \sum_{i \leq r} X_{id}$.
17 else $\widetilde{N}(d)[t] = \frac{n}{r} \cdot \frac{\hat{d}_R}{q} \sum_{i \leq q} Y_{id}$.
18 For $d \in D$:
19 $N'(d) = median\{\widetilde{N}(d)\}$
20 Return $\{N'(d)\}$

4 ANALYSIS OF SADDLES

We now prove Theorem 3.2. There are a number of intermediate claims towards that. We will fix $d \in D$ and a choice of t. Abusing notation, we use $\widetilde{N}(d)$ to refer to $\widetilde{N}(d)[t]$. The estimate of Step 16 can be analyzed with a direct Chernoff bound.

Claim 4.1. *The following holds with probability* $> 9/10$. *If* SADDLES(r, q) *outputs an estimate in Step 16 for a given d, then* $\widetilde{N}(d) \in (1 \pm \varepsilon/10)\widehat{N_\Lambda}(d)$. *If it does not output in Step 16, then* $\widehat{N_\Lambda}(d) < (2c/\varepsilon^2)(n/r)$.

Proof. Each X_i is an iid Bernoulli random variable, with success probability precisely $\widehat{N_\Lambda}(d)/n$. We split into two cases.

Case 1: $\widehat{N_\Lambda}(d) \geq (c/10\varepsilon^2)(n/r)$. By the Chernoff bound of Theorem 2.1, $\Pr[|\sum_{i \leq r} X_i - r\widehat{N_\Lambda}(d)/n| \geq (\varepsilon/10)(r\widehat{N_\Lambda}(d)/n)] \leq 2\exp(-(\varepsilon^2/100)(r\widehat{N_\Lambda}(d)/n) \leq 1/100$.

Case 2: $\widehat{N_\Lambda}(d) \leq (c/10\varepsilon^2)(n/r)$. Note that $\mathbf{E}[\sum_{i \leq r} X_i] \leq c/10\varepsilon^2 \leq (c/\varepsilon^2)/2e$. By the upper tail bound of Theorem 2.1, $\Pr[\sum_{i \leq r} X_i \geq c/\varepsilon^2] < 1/100$.

Thus, with probability at least 99/100, if an estimate is output in Step 16, $\widehat{N_\Lambda}(d) > (c/10\varepsilon^2)(n/r)$. By the first case, with probability at least 99/100, $\widetilde{N}(d)$ is a $(1 + \varepsilon/10)$-estimate for $\widehat{N_\Lambda}(d)$. A union bound completes the first part.

Furthermore, if $\widehat{N_\Lambda}(d) \geq (2c/\varepsilon^2)(n/r)$, then with probability at least 99/100, $\sum_{i \leq r} X_i \geq (1 - \varepsilon/10)r\widehat{N_\Lambda}(d)/n \geq c/\varepsilon^2 = \tau$. A union bound proves (the contrapositive of) the second part. □

We define weights of *ordered* edges. The weight only depends on the second member in the pair, but allows for a more convenient analysis. The weight of $\langle v, u \rangle$ is the random variable Y_i of Step 13.

Definition 4.2. The d-weight of an ordered edge $\langle v, u \rangle$ for a given Λ (the randomness of DEG) is defined as follows. We set $\mathrm{wt}_{\Lambda,d}(\langle v, u \rangle)$ to be $1/\hat{d_u}$ if $\hat{d_u} \geq d$, and zero otherwise. For vertex v, $\mathrm{wt}_{\Lambda,d}(v) = \sum_{u \in \Gamma(v)} \mathrm{wt}_{\Lambda,d}(\langle v, u \rangle)$.

The utility of the weight definition is captured by the following claim. The total weight is an approximation of $\widetilde{N}(d)$, and thus, we can analyze how well SADDLES approximates the total weight.

Claim 4.3. *If* Λ *is good,* $\sum_{v \in V} \mathrm{wt}_{\Lambda,d}(v) \in (1 \pm \varepsilon/9)\widehat{N_\Lambda}(d)$.

Proof.

$$\sum_{v \in V} \mathrm{wt}_{\Lambda,d}(v) = \sum_{v \in V} \sum_{u \in \Gamma(v)} \mathbf{1}_{\hat{d_u} \geq d}/\hat{d_u}$$
$$= \sum_{u : \hat{d_u} \geq d} \sum_{v \in \Gamma(u)} 1/\hat{d_u} = \sum_{u : \hat{d_u} \geq d} d_u/\hat{d_u} \quad (1)$$

Since Λ is good, $\forall u, \hat{d_u} \in (1 \pm \varepsilon/10)d_u$, and $d_u/\hat{d_u} \in (1 \pm \varepsilon/9)$. Applying in (1), $\sum_{v \in V} \mathrm{wt}_{\Lambda,d}(v) \in (1 \pm \varepsilon/9)\widehat{N_\Lambda}(d)$. □

We come to an important lemma, that shows that the weight of the random subset R (chosen in Step 3) is well-concentrated. This is proven using a Chernoff bound, but we need to bound the maximum possible weight to get a good bound on $r = |R|$.

Lemma 4.4. *Fix any good* Λ *and d. Suppose* $r \geq c\varepsilon^{-2}n/d$. *With probability at least* 9/10, $\sum_{v \in R} \mathrm{wt}_{\Lambda,d}(v) \in (1 \pm \varepsilon/8)(r/n)\widehat{N_\Lambda}(d)$.

Proof. Let $\mathrm{wt}(R)$ denote $\sum_{v \in R} \mathrm{wt}_{\Lambda,d}(v)$. By linearity of expectation, $\mathbf{E}[\mathrm{wt}(R)] = (r/n) \cdot \sum_{v \in V} \mathrm{wt}_{\Lambda,d}(v) \geq (r/2n)\widehat{N_\Lambda}(d)$. To apply the Chernoff bound, we need to bound the maximum weight of a vertex. For good Λ, the weight $\mathrm{wt}_{\Lambda,d}$ of any ordered pair is at most $1/(1 - \varepsilon/10)d \leq 2/d$. The number of neighbors of v such that $\hat{d_u} \geq d$ is at most $\widehat{N_\Lambda}(d)$. Thus, $\mathrm{wt}_{\Lambda,d}(v) \leq 2\widehat{N_\Lambda}(d)/d$.

By the Chernoff bound of Theorem 2.1 and setting $r \geq c\varepsilon^{-2}n/d$,

$$Pr\left[|\mathrm{wt}(R) - \mathbf{E}[\mathrm{wt}(R)]| > (\varepsilon/20)\mathbf{E}[\mathrm{wt}(R)]\right]$$
$$< 2\exp\left(-\frac{\varepsilon^2 \cdot (c\varepsilon^{-2}n/d) \cdot (\widehat{N_\Lambda}(d)/2n)}{3 \cdot 20^2 \cdot 2\widehat{N_\Lambda}(d)/d}\right) \leq 1/10$$

With probability at least 9/10, $\mathrm{wt}(R) \in (1 \pm \varepsilon/20)\mathbf{E}[\mathrm{wt}(R)]$. By the arguments given above, $\mathbf{E}[\mathrm{wt}(R)] \in (1 \pm \varepsilon/9)(r/n)\widehat{N_\Lambda}(d)$. We combine to complete the proof. □

Now, we determine the number of edge samples required to estimate the weight $\mathrm{wt}_{\Lambda,d}(R)$.

Lemma 4.5. *Let* $\widetilde{N}(d)$ *be as defined in Step 17 of* SADDLES. *Assume* Λ *is good,* $r \geq c\varepsilon^{-2}n/d$, *and* $q \geq c\varepsilon^{-2}m/(d\widehat{N_\Lambda}(d))$. *Then, with probability* $> 7/8$, $\widetilde{N}(d) \in (1 \pm \varepsilon/4)\widehat{N_\Lambda}(d)$.

Proof. We define the random set R selected in Step 3 to be *sound* if the following hold. (1) $\mathrm{wt}(R) = \sum_{v\in R} \mathrm{wt}_{\Lambda,d}(v) \in (1 \pm \varepsilon/8)(r/n)\widehat{N_\Lambda}(d)$ and (2) $\sum_{v\in R} d_v \le 100r(2m/n)$. By Lemma 4.4, the first holds with probability $> 9/10$. Observe that $\mathbf{E}[\sum_{v\in R} d_v] = r(2m/n)$, since $2m/n$ is the average degree. By the Markov bound, the second holds with probability $> 99/100$. By the union bound, R is sound with probability at least $1 - (1/10 + 1/100) > 8/9$.

Fix a sound R. Recall Y_i from Step 13. The expectation of $Y_i|R$ is $\sum_{v\in R} \Pr[v \text{ is selected}] \cdot \sum_{u\in\Gamma(v)} \Pr[u \text{ is selected}]\mathrm{wt}_{\Lambda,d}(\langle v,u\rangle)$. We plug in the probability values, and observe that for good Λ, for all v, $\hat{d}_v/d_v \in (1 \pm \varepsilon/10)$.

$$\begin{aligned}
\mathbf{E}[Y_i|R] &= \sum_{v\in R} (\hat{d}_v/\hat{d}_R) \sum_{u\in\Gamma(v)} (1/d_v)\mathrm{wt}_{\Lambda,d}(\langle v,u\rangle) \\
&= (1/\hat{d}_R) \sum_{v\in R} (\hat{d}_v/d_v) \sum_{u\in\Gamma(v)} \mathrm{wt}_{\Lambda,d}(\langle v,u\rangle) \\
&\in (1 \pm \varepsilon/10)(1/\hat{d}_R) \sum_{v\in R} \sum_{u\in\Gamma(v)} \mathrm{wt}_{\Lambda,d}(\langle v,u\rangle) \\
&\in (1 \pm \varepsilon/10)(\mathrm{wt}(R)/\hat{d}_R) \qquad (2)
\end{aligned}$$

Note that $\widetilde{N}(d) = (n/r)(\hat{d}_R/q)\sum_{i\le q} Y_i$ and $(n/r)(\hat{d}_R/q)\mathbf{E}[\sum_{i\le q} Y_i|R] \in (1 \pm \varepsilon/10)(n/r)\mathrm{wt}(R)$. Since R is sound, the latter is in $(1 \pm \varepsilon/4)\widehat{N_\Lambda}(d)$. Also, note that

$$\mathbf{E}[Y_i|R] = \mathbf{E}[Y_1|R] \ge \frac{q\mathrm{wt}(R)}{2\hat{d}_R} \ge \frac{(r/n)\widehat{N_\Lambda}(d)}{4(100r(2m/n)} = \frac{\widehat{N_\Lambda}(d)}{800m} \qquad (3)$$

By linearity of expectation, $\mathbf{E}[\sum_{i\le q} Y_i|R] = q\mathbf{E}[Y_1|R]$. Observe that $Y_i \le 1/d$. We can apply the Chernoff bound of Theorem 2.1 to the iid random variables $(Y_i|R)$.

$$\begin{aligned}
&\Pr[|\sum_i Y_i - \mathbf{E}[\sum_i Y_i]| > (\varepsilon/100)\mathbf{E}[\sum_i Y_i]|R] \\
&\le 2\exp\Big(-\frac{\varepsilon^2}{3\cdot 100^2}\cdot d\cdot q\mathbf{E}[Y_1|R]\Big) \qquad (4)
\end{aligned}$$

We use (3) to bound the (positive) term in the exponent is at least

$$\frac{\varepsilon^2}{3\cdot 100^2}\cdot\frac{c\varepsilon^{-2}m}{\widehat{N_\Lambda}(d)}\cdot\frac{\widehat{N_\Lambda}(d)}{800m} \ge 10.$$

Thus, if R is sound, the following bound holds with probability at least 0.99. We also apply (2).

$$\begin{aligned}
\widehat{N_\Lambda}(d) &= (n/r)(\hat{d}_R/q)\sum_{i=1}^{q} Y_i \\
&\in (1 \pm \varepsilon/100)(n/r)(\hat{d}_R/q)q\mathbf{E}[Y_i|R] \\
&\in (1 \pm \varepsilon/100)(1 \pm \varepsilon/10)(n/r)\mathrm{wt}(R) \in (1 \pm \varepsilon/4)\widetilde{N}(d)
\end{aligned}$$

The probability that R is sound is at least 8/9. A union bound completes the proof. □

The bounds on r and q in Lemma 4.5 depend on the degree d. We now bring in the h and z-indices to derive bounds that hold for all d. We also remove the conditioning over a good Λ.

Proof. (of Theorem 3.2) We will first assume that Λ is good. By Claim 2.8, $\widehat{N_\Lambda}(d) \in [N((1+\varepsilon/9)d, N((1-\varepsilon/9)d)]$.

Suppose $\widehat{N_\Lambda}(d) = 0$, so there are no vertices with $\hat{d}_v \ge d$. By the bound above, $N((1+\varepsilon/9)d) = 0$, implying that $N((1+\varepsilon/2)d) = 0$. Furthermore $\widetilde{N}(d) = 0$, since the random variables X_i and Y_i in SADDLES can never be non-zero. Thus, $\widetilde{N}(d) = N((1+\varepsilon/2)d)$, completing the proof.

We now assume that $\widehat{N_\Lambda}(d) > 0$. We split into two cases, depending on whether Step 16 outputs or not. By Claim 4.1, with probability $> 9/10$, if Step 16 outputs, then $\widetilde{N}(d) \in (1 \pm \varepsilon/9)\widehat{N_\Lambda}(d)$. By combining these bounds, the desired bound on $\widetilde{N}(d)$ holds with probability $> 9/10$, conditioned on a good Λ.

Henceforth, we focus on the case that Step 16 does not output. By Claim 4.1, $\widehat{N_\Lambda}(d) < 2c\varepsilon^{-2}(n/r)$. By the choice of r and Claim 2.8, $\widehat{N_\Lambda}((1+\varepsilon/9)d) < h$. By the characterization of h of Lemma 2.2, $z^2 \le \max(\widehat{N_\Lambda}((1+\varepsilon/9)d), (1+\varepsilon/9)d) = (1+\varepsilon/9)d$. This implies that $r \ge c\varepsilon^{-2}n/d$. By the definition of z, $z^2 \le N(\min(d_{max}, (1+\varepsilon/9)d))\cdot \min(d_{max}, (1+\varepsilon/9)d)$. By the Claim 2.8 bound in the first paragraph, $\widehat{N_\Lambda}(d) \ge N((1+\varepsilon/9)d)$. Since $\widehat{N_\Lambda}(d) > 0$, $\widehat{N_\Lambda}(d) \ge \widehat{N_\Lambda}(d_{max})$. Thus, $z^2 \le \widehat{N_\Lambda}(d)\cdot(1+\varepsilon/9)d$. and hence, $m \le c\varepsilon^{-2}m/(d\widehat{N_\Lambda}(d))$. The parameters satisfy the conditions in Lemma 4.5. With probability $> 7/8$, $\widetilde{N}(d) \in (1 \pm \varepsilon/4)\widehat{N_\Lambda}(d)$, and by Claim 2.8, $\widetilde{N}(d)$ has the desired accuracy.

All in all, assuming Λ is good, with probability at least 7/8, $\widetilde{N}(d)$ has the desired accuracy. The conditioning on a good Λ is removed by Claim 2.7 to complete the proof. □

5 EXPERIMENTAL RESULTS

We implemented our algorithm in C++ and performed our experiments on a MacBook Pro laptop with 2.7 GHz Intel Core i5 with 8 GB RAM. We performed our experiments on a collection of graphs from SNAP [30], including social networks, web networks, and infrastructure networks. The graphs typically have millions of edges, with the largest having more than 100M edges. Basic properties of these graphs are presented in Table 1. We ignore direction and treat all edges as undirected edges.

5.1 Implementation Details

For the HDM, we explicitly describe the procedure DEG(v), which estimates the degree of a given vertex (v).

Algorithm 2: DEG(v)

1 (Initialize $S = \emptyset$.) Repeatedly add u.a.r. vertex to S, until the number of pair-wise collisions is at least $k = 25$.
2 Output $\binom{|S|}{2}/k$ as estimate $\hat{d}_v$.

In the algorithm DEG, a "pair-wise collision" refers to a pair of neighbor samples that yield the same vertex. The expected number of pair-wise collisions is $\binom{|S|}{2}/d_v$. We simply reverse engineer that inequality to get the estimate $\hat{d}_v$. Ron and Tsur essentially prove that this estimate has low variance [42].

Setting the parameter values. The boosting parameter ℓ is simply set to 1. (In some sense, we only introduced the median boosting for the theoretical union bound. In practice, convergence is much more rapid that predicted by the Chernoff bound.)

The threshold τ is set to 100. The parameters r and q are chosen to be typically around $0.005n$. These are not "sublinear" per se, but are an order of magnitude smaller than the queries made in existing graph sampling results (more discussion in next section).

We set $D = \{\lfloor 1.1^i \rfloor\}$, since that gives a sufficiently fine-grained approximation at all scales of the degree distribution.

Code for all experiments is available here[1].

5.2 Evaluation of SADDLES

Accuracy over all graphs: We run SADDLES with the parameters discussed above for a variety of graphs. Because of space considerations, we do not show results for all graphs in this version. (We discovered the results to be consistent among all our experiments.) Fig. 1 show the results for the SM for some graphs in Tab. 1. For all these runs, we set $r + q$ to be 1% of the number of vertices in the graph. Note that the sample size of SADDLES in the SM is exactly $r + q$. For the HDM, we show results in Fig. 2. Again, we set $r + q$ to be 1%, though the number of edges sampled (due to invocations of DEG(v)) varies quite a bit. The required number of samples are provided in Tab. 1. Note that the number of edges sampled is well within 10% of the total, except for the `com-youtube` graph.

Visually, we can see that the estimates are accurate for all degrees, in all graphs, for both models. This is despite there being sufficient irregular behavior in $N(d)$. Note that the shape of the various ccdhs are different and none of them form an obvious straight line. Nonetheless, SADDLES captures the distribution almost perfectly in all cases by observing 1% of the vertices.

Convergence: To demonstrate convergence, we fix the graph `com-orkut`, and run SADDLES only for the degrees 10, 100, and 1000. For each choice of degree, we vary the total number of samples $r + q$. (We set $r = q$ in all runs.) Finally, for each setting of $r + q$, we perform 100 independent runs of SADDLES.

For each such run, we compute an error parameter α. Suppose the output of a run is M, for degree d. The value of α is the smallest value of ϵ, such that $M \in [(1-\epsilon)N((1+\epsilon)d), (1+\epsilon)N((1-\epsilon)d)]$. (It is the smallest ϵ such that M is an (ϵ, ϵ)-approximation of $N(d)$.)

Fig. 3 shows the spread of α, for the 100 runs, for each choice of $r + q$. Observe how the spread decreases as $r + q$ goes to 10%. In all cases, the values of α decay to less than 0.05. We notice that convergence is much faster for $d = 10$. This is because $N(10)$ is quite large, and SADDLES is using vertex sampling to estimate the value.

Large value of h and z-index on real graphs: The h and z-index of all graphs is given in Tab. 1. Observe how they are typically in the hundreds. Note that the average degree is typically an order of magnitude smaller than these indices. Thus, a sample size of $n/h + m/z^2$ (as given by Theorem 3.1, ignoring constants) is significantly sublinear. This is consistent with our choice of $r + q = n/100$ leading to accurate estimates for the ccdh.

5.3 Comparison with previous work

There are several graph sampling algorithms that have been discussed in [2, 17, 29, 31, 39, 40, 46]. In all of these methods we collect the vertices and scale their counts appropriately to get the estimated ccdh. We describe these methods below in more detail, and discuss our implementation of the method.

- Vertex Sampling (VS, also called egocentric sampling) [5, 17, 29, 31, 39, 40]: In this algorithm, we sample vertices u.a.r. and scale the ccdh obtained appropriately, to get an estimate for the ccdh of the entire graph.
- Edge Sampling (ES) [5, 17, 29, 31, 39, 40]: This algorithm samples edges u.a.r. and includes one or both end points in the sampled network. Note that this does *not* fall into the SM. In our implementation we pick a random end point.
- Random walk with jump (RWJ) [5, 17, 31, 39, 40]: We start a random walk at a vertex selected u.a.r. and collect all vertices encountered on the path in our sampled network. At any point, with a constant probability (0.15, based on previous results) we jump to another u.a.r. vertex.
- One Wave Snowball (OWS) [5, 17, 29]: Snowball sampling starts with some vertices selected u.a.r. and crawls the network until a network of the desired size is sampled. In our implementation, we usually stop at the first level since that accumulates enough vertices.
- Forest fire (FF) [5, 17, 31]: This method generates random sub-crawls of the network. A vertex is picked u.a.r. and randomly selects a subset of its neighbors (according to a geometric distribution). The process is repeated from every selected vertex until it ends. It is then repeated from another u.a.r. vertex.

We run all these algorithms on the `amazon0601`, `web-Google`, `cit-Patents`, and `com-orkut` networks. To make fair comparisons, we run each method until it selects 1% of the vertices. The comparisons are shown in Fig. 1. Observe how none of the methods come close to accurately measuring the ccdh. (This is consistent with previous work, where typically 10-20% of the vertices are sampled for results.) Naive vertex sampling is accurate at the head of the distribution, but completely misses the tail. Except for vertex sampling, all other algorithms are biased towards the tail. Crawls find high degree vertices with disproportionately higher probability, and overestimate the tail.

Note that our implementations of FF, OWS, RWJ assume access to u.a.r. vertices. Variants of these algorithms can be used in situations where we only have access to seed vertices, however, one would typically have to sample many more edges to deal with larger correlation among the vertices obtained through the random walks. Despite this extra capability to sample u.a.r. vertices in our implementation of these algorithms, they show significant errors, particularly in the tail of the distribution.

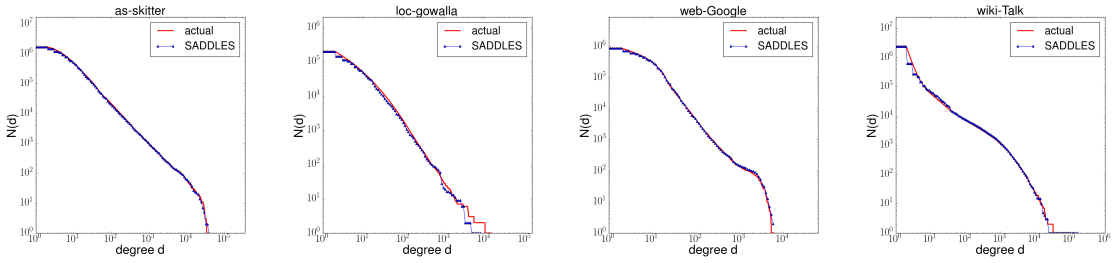
Inverse method of Zhang et al [46]: An important result of estimating degree distributions is that of Zhang et al [46], that explicitly points out the bias problems in various sampling methods. They propose a bias correction method by solving a constrained, penalized weighted least-squares problem on the sampled degree distribution. We apply this method for the sampling methods demonstrated in their paper, namely VS, OWS, and IN (sample vertices u.a.r. and only retain edges between sampled vertices). We show results in Fig. 1, again with a sample size of 1% of the vertices. Observe that no method gets even close to estimating the ccdh accurately, even after debiasing. Fundamentally, these methods require significantly more samples to generate accurate estimates.

The running time and memory requirements of this method grow superlinearly with the maximum degree in the graph. The largest graph processed by [46] has a few hundred thousand edges, which is on the smaller side of graphs in Tab. 1. SADDLES processes a graph with more than 100M edges in less than a minute, while our

[1] https://sjain12@bitbucket.org/sjain12/saddles.git

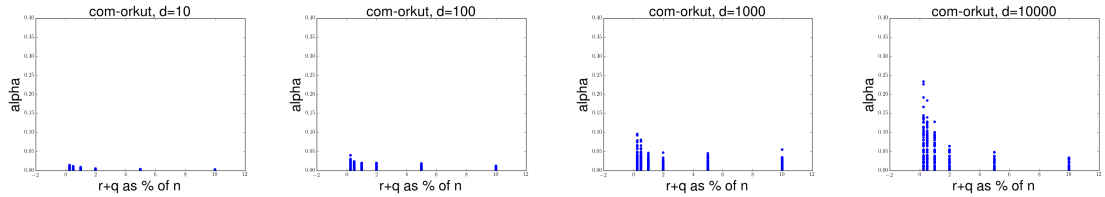
Table 1: Graph properties: #vertices (n), #edges (m), maximum degree, h-index and z-index. The last column indicates the median number of samples over 100 runs (as a percentage of m) required by SADDLES under HDM, with $r + q = 0.01n$.

graph	#vertices	#edges	max. degree	avg. degree	H-index	Z-index	Perc. edge samples for HDM
loc-gowalla	1.97E+05	9.50E+05	14730	4.8	275	101	7.0
web-Stanford	2.82E+05	1.99E+06	38625	7.0	427	148	6.4
com-youtube	1.13E+06	2.99E+06	28754	2.6	547	121	11.7
web-Google	8.76E+05	4.32E+06	6332	4.9	419	73	6.2
web-BerkStan	6.85E+05	6.65E+06	84230	9.7	707	220	5.5
wiki-Talk	2.39E+06	9.32E+06	100029	3.9	1055	180	8.5
as-skitter	1.70E+06	1.11E+07	35455	6.5	982	184	6.7
cit-Patents	3.77E+06	1.65E+07	793	4.3	237	28	5.6
com-lj	4.00E+06	3.47E+07	14815	8.6	810	114	4.7
soc-LiveJournal1	4.85E+06	8.57E+07	20333	17.7	989	124	2.4
com-orkut	3.07E+06	1.17E+08	33313	38.1	1638	172	2.0



(a) as-skitter (b) loc-gowalla (c) web-Google (d) wiki-Talk

Figure 2: The result of runs of SADDLES on a variety of graphs, for the HDM. We set $r + q$ to be 1% of the number of vertices, for all graphs. The actual number of edges sampled varies, and is given in Tab. 1.



(a) $d = 10$ (b) $d = 100$ (c) $d = 1000$ (d) $d = 10000$

Figure 3: Convergence of SADDLES: We plot the values of the error parameter α (as defined in §5.2) for 100 runs at increasing values of $r + q$. We have a different plot for $d = 10, 100, 1000, 10000$ to show the convergence at varying portions of the ccdh.

attempts to run the [46] algorithm on this graph did not terminate in hours.

6 ACKNOWLEDGEMENTS

Ali Pinar's work is supported by the Laboratory Directed Research and Development program at Sandia National Laboratories. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

Both Shweta Jain and C. Seshadhri are grateful to the support of the Sandia National Laboratories LDRD program for funding this research. C. Seshadhri also acknowledges the support of NSF TRIPODS grant, CCF-1740850.

This research was partially supported by the Israel Science Foundation grant No. 671/13 and by a grant from the Blavatnik fund. Talya Eden is grateful to the Azrieli Foundation for the award of an Azrieli Fellowship.

Both Talya Eden and C. Seshadhri are grateful to the support of the Simons Institute, where this work was initiated during the Algorithms and Uncertainty Semester.

REFERENCES

[1] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore. 2009. On the bias of traceroute sampling: Or, power-law degree distributions in regular graphs. *J. ACM* 56, 4 (2009).

[2] N.K. Ahmed, J. Neville, and R. Kompella. 2010. Reconsidering the Foundations of Network Sampling. In *WIN 10*.

[3] N. Ahmed, J. Neville, and R. Kompella. 2012. Space-Efficient Sampling from Social Activity Streams. In *SIGKDD BigMine*. 1–8.

[4] Nesreen K Ahmed, Nick Duffield, Jennifer Neville, and Ramana Kompella. 2014. Graph sample and hold: A framework for big-graph analytics. In *SIGKDD*. ACM, ACM, 1446–1455.

[5] Nesreen K Ahmed, Jennifer Neville, and Ramana Kompella. 2014. Network sampling: From static to streaming graphs. *TKDD* 8, 2 (2014), 7.

[6] Sinan G. Aksoy, Tamara G. Kolda, and Ali Pinar. 2017. Measuring and modeling bipartite graphs with community structure. *Journal of Complex Networks* (2017). to appear.

[7] Albert-László Barabási and Réka Albert. 1999. Emergence of Scaling in Random Networks. *Science* 286 (Oct. 1999), 509–512.

[8] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. 2000. Graph structure in the web. *Computer Networks* 33 (2000), 309–320.

[9] Deepayan Chakrabarti and Christos Faloutsos. 2006. Graph Mining: Laws, Generators, and Algorithms. *Comput. Surveys* 38, 1 (2006). https://doi.org/10.1145/1132952.1132954

[10] F. Chierichetti, A. Dasgupta, R. Kumar, S. Lattanzi, and T. Sarlos. 2016. On Sampling Nodes in a Network. In *Conference on the World Wide Web (WWW)*.

[11] A. Clauset and C. Moore. 2005. Accuracy and scaling phenomena in internet mapping. *Phys. Rev. Lett.* 94 (2005), 018701.

[12] A. Clauset, C. R. Shalizi, and M. E. J. Newman. 2009. Power-Law Distributions in Empirical Data. *SIAM Rev.* 51, 4 (2009), 661–703. https://doi.org/10.1137/070710111

[13] R. Cohen, K. Erez, D. ben Avraham, and S. Havlin. 2000. Resilience of the Internet to Random Breakdowns. *Phys. Rev. Lett.* 85, 4626âĂŞ8 (2000).

[14] A. Dasgupta, R. Kumar, and T. Sarlos. 2014. On estimating the average degree. In *Conference on the World Wide Web (WWW)*. 795–806.

[15] D. Dubhashi and A. Panconesi. 2012. *Concentration of Measure for the Analysis of Randomised Algorithms*. Cambridge University Press.

[16] N. Durak, T.G. Kolda, A. Pinar, and C. Seshadhri. 2013. A scalable null model for directed graphs matching all degree distributions: In, out, and reciprocal. In *Network Science Workshop (NSW), 2013 IEEE 2nd*. 23–30. https://doi.org/10.1109/NSW.2013.6609190

[17] Peter Ebbes, Zan Huang, Arvind Rangaswamy, Hari P Thadakamalla, and ORGB Unit. 2008. Sampling large-scale social networks: Insights from simulated networks. In *18th Annual Workshop on Information Technologies and Systems, Paris, France*.

[18] Talya Eden, Shweta Jain, Ali Pinar, Dana Ron, and C. Seshadhri. 2017. Provable and practical approximations for the degree distribution using sublinear graph samples. *CoRR* abs/1710.08607 (2017). arXiv:1710.08607 http://arxiv.org/abs/1710.08607

[19] T. Eden, A. Levi, D. Ron, and C. Seshadhri. 2015. Approximately Counting Triangles in Sublinear Time. In *Foundations of Computer Science (FOCS)*, GRS11 (Ed.). 614–633.

[20] T. Eden, D. Ron, and C. Seshadhri. 2017. Sublinear Time Estimation of Degree Distribution Moments: The Degeneracy Connection. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, GRS11 (Ed.). 614–633.

[21] M. Faloutsos, P. Faloutsos, and C. Faloutsos. 1999. On power-law relationships of the internet topology. In *SIGCOMM*. 251–262.

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

[23] O. Goldreich and D. Ron. 2002. Property Testing in Bounded Degree Graphs. *Algorithmica* (2002), 302–343.

[24] O. Goldreich and D. Ron. 2008. Approximating average parameters of graphs. *Random Structures and Algorithms* 32, 4 (2008), 473–493.

[25] M. Gonen, D. Ron, and Y. Shavitt. 2011. Counting stars and other small subgraphs in sublinear-time. *SIAM Journal on Discrete Math* 25, 3 (2011), 1365–1411.

[26] Mira Gonen, Dana Ron, Udi Weinsberg, and Avishai Wool. 2008. Finding a dense-core in Jellyfish graphs. *Computer Networks* 52, 15 (2008), 2831–2841. https://doi.org/10.1016/j.comnet.2008.06.005

[27] J. E. Hirsch. 2005. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences* 102, 46 (2005), 16569âĂŞ16572.

[28] A. Lakhina, J. Byers, M. Crovella, and P. Xie. 2003. Sampling biases in IP topology measurements. In *Proceedings of INFOCOMM*, Vol. 1. 332–341.

[29] Sang Hoon Lee, Pan-Jun Kim, and Hawoong Jeong. 2006. Statistical properties of sampled networks. *Physical Review E* 73, 1 (2006), 016102.

[30] Jure Leskovec. 2015. SNAP Stanford Network Analysis Project. http://snap.standord.edu. (2015).

[31] Jure Leskovec and Christos Faloutsos. 2006. Sampling from large graphs. In *Knowledge Data and Discovery (KDD)*. ACM, 631–636.

[32] A. S. Maiya and T. Y. Berger-Wolf. 2011. Benefits of Bias: Towards Better Characterization of Network Sampling, In Knowledge Data and Discovery (KDD). *ArXiv e-prints*, 105–113. arXiv:1109.3911

[33] Andrew McGregor. 2014. Graph stream algorithms: A survey. *SIGMOD* 43, 1 (2014), 9–20.

[34] M. Mitzenmacher. 2003. A Brief History of Generative Models for Power Law and Lognormal Distributions. *Internet Mathematics* 1, 2 (2003), 226–251.

[35] M. E. J. Newman. 2003. The Structure and Function of Complex Networks. *SIAM Rev.* 45, 2 (2003), 167–256. https://doi.org/10.1137/S003614450342480

[36] M. E. J. Newman, S. Strogatz, and D. Watts. 2001. Random graphs with arbitrary degree distributions and their applications. *Physical Review E* 64 (2001), 026118.

[37] D. Pennock, G. Flake, S. Lawrence, E. Glover, and C. L. Giles. 2002. Winners don't take all: Characterizing the competition for links on the web. *Proceedings of the National Academy of Sciences* 99, 8 (2002), 5207–5211. https://doi.org/10.1073/pnas.032085699

[38] T. Petermann and P. Rios. 2004. Exploration of scale-free networks. *European Physical Journal B* 38 (2004), 201–204.

[39] Ali Pinar, Sucheta Soundarajan, Tina Eliassi-Rad, and Brian Gallagher. 2015. *MaxOutProbe: An Algorithm for Increasing the Size of Partially Observed Networks*. Technical Report. Sandia National Laboratories (SNL-CA), Livermore, CA (United States).

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

[41] Dana Ron. 2010. Algorithmic and Analysis Techniques in Property Testing. *Foundations and Trends in Theoretical Computer Science* 5, 2 (2010), 73–205.

[42] Dana Ron and Gilad Tsur. 2016. The Power of an Example: Hidden Set Size Approximation Using Group Queries and Conditional Sampling. *ACM Transactions on Computation Theory* 8, 4 (2016), 15:1–15:19.

[43] C. Seshadhri, Tamara G. Kolda, and Ali Pinar. 2012. Community structure and scale-free collections of Erdös-Rényi graphs. *Physical Review E* 85, 5 (May 2012), 056109. https://doi.org/10.1103/PhysRevE.85.056109

[44] Olivia Simpson, C Seshadhri, and Andrew McGregor. 2015. Catching the head, tail, and everything in between: a streaming algorithm for the degree distribution. In *International Conference on Data Mining (ICDM)*. IEEE, 979–984.

[45] Michael PH Stumpf and Carsten Wiuf. 2005. Sampling properties of random graphs: the degree distribution. *Physical Review E* 72, 3 (2005), 036118.

[46] Yaonan Zhang, Eric D Kolaczyk, and Bruce D Spencer. 2015. Estimating network degree distributions under sampling: An inverse problem, with applications to monitoring social media networks. *The Annals of Applied Statistics* 9, 1 (2015), 166–199.