2011

Network Sampling via Edge-based Node Selection with Graph Induction

Nesreen Ahmed
Purdue University, nkahmed@cs.purdue.edu

Jennifer Neville
Purdue University, neville@cs.purdue.edu

Ramana Rao Kompella
Purdue University, kompella@cs.purdue.edu

Report Number:
11-016

Network Sampling via Edge-based Node Selection with Graph Induction

Nesreen Ahmed, Jennifer Neville, and Ramana Kompella
Computer Science Department
Purdue University
West Lafayette, IN 47907
{nkahmed, neville, kompella }@cs.purdue.edu

ABSTRACT

In order to efficiently study the characteristics of network domains and support development of network systems (*e.g.* algorithms, protocols that operate on networks), it is often necessary to *sample* a representative subgraph from a large complex network. While prior research has shown that topological (*e.g.* random-walk based) sampling methods produce more accurate samples than approaches based on node or edge sampling, they still do not produce samples that closely match the distributions of graph properties (*e.g.*, degree) found in the original graph. In this paper, we observe that part of the problem is that any sampling process fundamentally biases the structure of the sampled subgraph, since all neighbors of a sample node may not be included in the sampled subgraph. We address this problem using a novel sampling algorithm called TIES that (1) aims to offset this bias by using edge-based node selection, which favors selection of high-degree nodes, and (2) uses a graph induction step to select additional edges between sampled nodes to restore connectivity and bring the structure closer to that of the original graph. To understand the properties of TIES we compare it analytically to random node and edge sampling. We also evaluate the efficacy of TIES empirically using several real-world data sets. Across all datasets, we found that TIES produces samples that better match the original distributions. In terms of two distributional distance metrics, KS distance and skew divergence, we found that samples produced by TIES consistently outperform other sampling algorithms—with up to 2× reduction in KS distance and up to 3-7× reduction in skew divergence, compared to the current state-of-the-art algorithms.

1. INTRODUCTION

Many real-world complex systems can be represented as graphs and networks—from information networks, to communication networks, to biological networks. Recently there has been a surge of interest in studying the characteristics of these networks, modeling their structure, as well as developing algorithms and systems that operate on the networks. However, many of the real-world networks are too large to efficiently acquire, store and/or analyze (*e.g.* there are 3 billion emails per day worldwide). Although the data mining community focuses on developing scalable analytic methods for very large datasets, in order to facilitate the development and testing of systems for network domains it is often necessary to *sample* smaller subgraphs from the larger network structure. A sampled subgraph can be used to drive realistic simulations and experimentation before deploying new protocols and systems in the field—for example, new Internet protocols, social/viral marketing schemes, and/or fraud detection algorithms. In order to make accurate assessments about the performance of such systems, it is important to have sampling methods that can select a *representative* subgraph from the larger network. In addition, since it can be costly or difficult to acquire the full network structure (*i.e.* due to temporal evolution or restricted access), many datasets naturally comprise a subset of the network data. In this case, it is important to understand the effects of data collection mechanisms on the properties of the sampled subgraph.

The standard graph sampling formulation is as follows: Assume an input graph $G = (V, E)$ from which the sampling algorithm selects a subset of the nodes ($V_s \subset V$) and/or edges ($E_s \subset E$). Within this framework, past work on network sampling has focused on two sampling objectives: (1) To use the nodes V_s and/or edges E_s to accurately *estimate* network parameters in the original graph G (*e.g.*, degree, diameter), (2) to select a *representative* subgraph $G_s = (V_s, E_s)$ from the original graph G (*i.e.*, such that G_s has structure similar to G).

Several sampling approaches (*e.g.*, [25, 5, 9, 23]) focus primarily on the first goal. Sampling, in these works, is used to quickly explore and estimate characteristics of network topology in domains that are either hard to explore completely or that have significant amounts of churn in the structure. Other sampling methods focus on the second goal with an aim of sampling subgraphs from network domains where the structure is both known and accessible, but where it is costly to acquire the sample (*e.g.*, crawling large social networks) and/or costly to operate on the full network structure (*e.g.*, when testing network protocols). For example, to collect data from large online social networks, researchers often use snowball sampling (*e.g.*, [5]), random walk sampling (*e.g.*, [18]), or enhanced versions of node sampling that improve the subgraph properties through direct optimization [11].

In this work, we focus on objective (2), *i.e.*, *how to sample a representative subgraph G_s given the original graph G*. Since the representativeness of the graph structure is difficult to evaluate directly, our aim is to select a subgraph G_s such that it simultaneously preserves many properties of G (*e.g.*, degree distribution, path length distribution). While recent work in this direction, notably [18, 11], have produced sampling algorithms that are more accurate than simple random node or edge sampling algorithms (NS and ES respectively), they still do not produce samples that closely match the distributions of various properties in the original graph G.

We observe that the problem is intrinsic to the process of *subgraph formation* itself (used to construct the sample), since sampling fundamentally biases the structure of the sampled subgraph. For instance, when a node is selected for inclusion in the sample, it is unlikely that *all* of its neighbors will be included in the subgraph, and thus, sampled degrees of nodes tend to be smaller than original degrees. Thus, conventional wisdom of selecting nodes in an unbiased manner (*e.g.*, uniformly at random) may not yield representative *subgraphs* that match the properties of the original

graph.

To address this problem, we propose a new sampling approach that effectively offsets the bias of subgraph formation process, thus enabling a closer match between the samples and the original graph, compared to previous sampling algorithms. Specifically, we propose a sampling algorithm called *totally-induced edge sampling* (TIES) that is based on two key ideas: First, in order to mitigate the effects of the downward bias, we use edge sampling (ES) that naturally exhibits an upward bias, *i.e.*, it selects high degree nodes with higher probability [23], for node selection. Second, while ES does a good job selecting the high degree nodes, the connectivity in its resulting sampled subgraph is quite sparse since each edge is sampled independently. To counter this, we use a *graph induction* step to add all edges that exist between the sampled nodes. This step improves the connectivity of the sampled subgraph and brings the distributions close(r) to those of the original graph.

We evaluate TIES over a number of real world (*e.g.*, Facebook, Twitter, arXiv, Enron) datasets collected by other researchers ([2, 26]), and an email network constructed from two weeks of Purdue email traffic. Across all datasets, we observed that TIES produces samples that better match the distributions of degree, path length and clustering compared to other existing algorithms. In terms of two distributional distance metrics, KS distance and skew divergence [15], we found that samples produced by TIES consistently outperform other sampling algorithms—with up to 2× reduction in KS distance and up to 3-7× reduction in skew, compared to the current state-of-the-art Forest Fire sampling algorithm [18].

Another major strength of TIES is that it is amenable to a *streaming* implementation. In domains where the network graph is constructed from a sequence of edges over time (*e.g.* email, social collaboration applications), it is important for the sampling algorithm to incrementally compute the sampled graph as the edges are streamed in. TIES by itself cannot be efficiently implemented in a streaming fashion, but we develop a simple modification to TIES, that induces the graph in the forward direction, *i.e.*, includes an edge among a pair of sampled nodes if it occurs *after* both nodes have been added to the sample. We refer to this algorithm as partially-induced edge sampling (PIES). PIES essentially retains the core strengths of TIES, and thus, outperforms other algorithms similar to TIES, and yet can be implemented in a streaming fashion.

The rest of the paper is organized as follows. We first present a background on sampling methods in Section 2. Next, we outline our proposed sampling algorithm, TIES, explore its properties analytically, and discuss the streaming implementation of TIES in Section 3. We compare TIES with other state-of-the-art sampling algorithms in Section 4. We briefly review other prior work in Section 5 before we conclude in Section 6.

2. BACKGROUND

In this section, we formally state the sampling problem and outline a few state-of-the-art sampling mechanisms briefly.

2.1 Problem definition

Let $G = (V, E)$ represent the graph, where V is the set of nodes and E is the set of edges in the graph. Each edge $e \in E$ can be described as a tuple of the form (v_i, v_j) where $v_i, v_j \in V$. Given a *sampling fraction* ϕ, the goal is to create a sample graph $G_s = (V_s, E_s)$ such that $|V_s|/|V| = \phi$, that preserves the structure of the original network. Note that we assume that we have access to the full graph G to begin with, *i.e.*, the sampling algorithm can access all the nodes and edges in the full graph to create the sampled graph.

In order to assess the representativeness of G_s, we evaluate whether the sampled graph is able to simultaneously preserve *the distributions* of several characteristic measures of G such as degree, path length, clustering coefficients, and size of connected components. The degree and clustering coefficient distributions capture the local properties of nodes in the graph, while path length and connected component distributions capture its global topological features. We consider distributions mainly since they capture intrinsic graph structure and connectivity better than point statistics such as average degree.

Most graph sampling algorithms have two basic components: (1) node selection, and (2) induced graph formation. The node selection step identifies a sample set of nodes (V_s), while the graph induction step selects the set of edges (E_s) to be included in the sampled graph. We distinguish between two different approaches to graph induction—total and partial graph induction—which differ by whether *all* or *some* of the edges incident on the sampled nodes are included in the sampled graph. The resulting sampled graphs are referred to as the *induced subgraph* and *partially induced subgraph* respectively.

2.2 Current sampling methods

State-of-the-art sampling techniques can be broadly classified as node-based, edge-based, and topology-based techniques.

Node sampling (NS). In classic node sampling, nodes are chosen independently and uniformly at random from the original graph for inclusion in the sampled graph. For a target fraction ϕ of nodes required, each node is simply sampled with a probability of ϕ. Once the nodes are selected, the sampled graph consists of the *induced subgraph* over the selected nodes, *i.e.*, all edges among the sampled nodes are added to form the sampled graph.

Sampled subgraphs produced by node sampling can be further refined using the Metropolis algorithms proposed in [11]. The key idea is to replace sampled nodes with other potential nodes that will better match the original degree distribution (or other metrics). Of course, this assumes that we have computed the desired distributions from the original graph, which may be quite difficult on very large graphs. In addition, since the Metropolis algorithm searches in the space of possible node sets, the search is computationally intensive for large sampled graphs (*e.g.*, >1000 nodes). In our work we found the optimization process of [11] was ineffective for larger samples— producing graphs with characteristics comparable to NS. We conjecture that this is due to the increased search space of possible candidate swaps, which significantly increases the time needed for the Markov chain to converge. Since we found that [11] produced graphs with characteristics comparable to NS, but with runtimes significantly higher, we only report NS result in this paper.

Edge sampling (ES). Edge sampling focuses on the selection of edges rather than nodes to populate the sample. Thus, the node selection step in edge sampling algorithm proceeds by just sampling edges, and including both nodes when a particular edge is sampled. The partially induced graph is created just out of the sampled edges; which means no extra edges are added in addition to those chosen during the random edge selection process.

Topology-based sampling. Due to the known limitations of NS [24, 16] and ES (bias toward high degree nodes), researchers have also considered many other topology-based sampling methods. One example is snowball sampling, which selects nodes using breadth-first search from a randomly selected seed node. Snowball sampling accurately maintains the network connectivity within the snowball, however it suffers from a *boundary bias* in that many peripheral nodes (*i.e.*, those sampled on the last round) will be missing a large

number of neighbors [16].

In [18], Leskovec *et al.* analyze various sampling algorithms for sampling large graphs, and propose a Forest Fire Sampling (FFS) method. FFS is a hybrid combination of snowball sampling and random-walk sampling that has been shown to produce quite accurate samples in practice. It starts by picking a node uniformly at random and adding it to the sample. It then 'burns' a fraction of its outgoing links with the nodes attached to them. The fraction is a random number drawn from a geometric distribution with mean $p_f/(1-p_f)$). (The authors recommend $p_f = 0.7$, which means on average each selected node burns 2.33 nodes from its neighbors). This process is recursively repeated for each burnt neighbor until no new node is selected, and a new random node is chosen to start the process until we obtain the desired sample size.

In general, topology-based sampling approaches such as FFS are considered the state-of-the-art sampling algorithms. However, while they do better than NS and ES, they still do not match the original distributions precisely. In addition, FFS is difficult to implement easily for time-varying graphs (as we shall argue in Section 3.4).

3. EDGE-BASED NODE SAMPLING WITH GRAPH INDUCTION

In this section, we propose a novel graph sampling approach based on edge-based node selection with graph induction.

3.1 Key intuition

Our approach exploits two key observations: First, we observe that edge sampling is inherently biased towards selection of nodes with higher degrees, resulting in an *upward bias* in the degree distributions of sampled nodes compared to nodes in the original graph [23]. However, in all sampled subgraphs, degrees are naturally underestimated since only a fraction of neighbors may be selected. This results in a *downward bias*, regardless of the actual sampling algorithm used. While the upward bias of edge sampling can help offset this downward bias to some extent, it alone is not sufficient to fully offset the bias. Because ES samples each edge independently, it is unlikely that the structure of the graph *surrounding* the high degree nodes will be preserved. Thus, the second observation we make is that a simple *graph induction* step over the edge-sampled node set (where we sample all the edges between any sampled nodes in the graph) can recover much of the connectivity around the high degree nodes—offsetting the downward degree bias as well as improving local clustering in the sampled graph.

These observations, while simple, makes the sampled graphs approximate the characteristics of the original graph much more accurately, even better than topology-based sampling algorithms. We use both theoretical analysis (Section 3.3) as well as empirical evaluation (Section 4) to validate the efficacy of our approach. Further, as we shall show in Section 3.4, our approach lends itself to a scalable streaming implementation that makes it even more attractive for sampling large-scale temporally-varying activity graphs.

3.2 TIES algorithm

In our approach, we select nodes in pairs by sampling edges in the same manner as the classic edge sampling approach. The key difference between our approach and ES is in the induced graph step; we augment the edges selected by the edge sampling step by including other edges between the set of sampled nodes. For example, suppose edges $e_1 = (v_1, v_2)$ and $e_2 = (v_3, v_4)$ are sampled, that leads to the addition of the vertices $v_1, ..., v_4$ into the sampled graph. In conventional edge sampling only these two edges e_1 and e_2 will be added to the sampled graph. In our approach, however, we add any other edges that exist in the original graph between any of these sampled nodes (*e.g.*, edge $e_3 = (v_1, v_3)$, edge $e_4 = (v_2, v_4)$, or any other such combinations). We refer to this algorithm as totally-induced edge sampling (TIES) and specify it formally in Algorithm 1.

Algorithm 1 TIES (sample fraction ϕ, edge set E)

```
Assume edges in E are stored in an array
▷ V_s = ∅, E_s = ∅
// Edge-based node sampling step
while |V_s| < φ × |V| do
    r = random (1, |E|) // uniformly random
    ▷ (u, v) = e_r
    V_s = V_s ∪ {u, v}
end while
// Graph induction step
for k = 1 : |E| do
    ▷ (u, v) = e_k,
    if u ∈ V_s AND v ∈ V_s then
        E_s = E_s ∪ {e_k}
    end if
end for
Output G_s = (V_s, E_s)
```

The algorithm runs in an iterative fashion, picking an edge at random from the original graph and adding both the nodes to the sampled node set in each iteration. It stops adding nodes once a target fraction ϕ of nodes are collected. After this, the algorithm proceeds to the graph induction step where it walks through all the edges in the graph and forms the induced graph by adding all edges which have both end-points already in the sampled node set.

3.3 Analytical comparison with ES and NS

In this section, we compare TIES analytically with ES and NS in order to illustrate the characteristics of TIES that lead to improved sampling accuracy. As noted before, there are two components to graph sampling procedures: (1) node selection, and (2) induced graph formation. TIES shares some similarity with NS and ES along each of these dimensions.

3.3.1 Node selection

First, consider the node selection process. Let V and E be the number of nodes and edges in the original graph. Let $f_D(k)$ be the number of nodes of degree k in the original graph. Let V_s be the target number of nodes in the sample graph (*i.e.*., $\phi = \frac{V_s}{V}$). Let $p_v = \frac{V_s}{V}$ be the probability of sampling a node in NS. Let E_s be the number of sampled edges in ES and TIES such that the sample will have V_s nodes. Then, $p_e = \frac{E_s}{E}$ is the probability of sampling a particular edge in ES or TIES (before graph induction). Let $E_*[|d_k|]$ refer to the expected number of sampled nodes that have degree k in the original graph, where * refers to any sampling method. Then:

$$\begin{aligned} E_{NS}[|d_k|] &= f_D(k) \cdot p_v \\ E_{ES}[|d_k|] &= f_D(k) \cdot [1 - (1 - p_e)^k] \\ E_{TIES}[|d_k|] &= f_D(k) \cdot [1 - (1 - p_e)^k] \end{aligned}$$

The first result is easy to see because for node sampling, each node has a uniform probability of being sampled. For edge sampling and TIES, the probability of selection is proportional to a node's degree. More specifically, the likelihood of selection corresponds to the complement of the probability that none of the node's k edges is sampled. Now we can show that ES (and by extension, TIES) selects high degree nodes with greater probability than NS.

LEMMA 3.3.1. *For degrees $k > log(1-p_v)/log(1-p_e)$, ES will sample degree k nodes at a higher rate than NS (i.e., $E_{ES}[|d_k|] > E_{NS}[|d_k|]$).*

Proof: Consider the threshold k at which the expected number of sampled nodes is greater for ES:

$$\begin{aligned} E_{NS}[|d_k|] &\leq E_{ES}[|d_k|] \\ 0 &\leq E_{ES}[|d_k|] - E_{NS}[|d_k|] \\ &= f_D(k) \cdot [1-(1-p_e)^k] - f_D(k) \cdot p_v \\ &= (1-p_v) - (1-p_e)^k \\ (1-p_e)^k &\leq (1-p_v) \\ k &\geq log(1-p_v)/log(1-p_e) \end{aligned}$$

□

For example, when $p_v = 0.20$ and $p_e = 0.05$, then $log(1-p_v)/log(1-p_e) = 4.35$, thus nodes with degree greater than 4 will have higher probability of selection in ES compared to NS. Since TIES samples nodes in the same manner as ES, the same result holds for TIES.

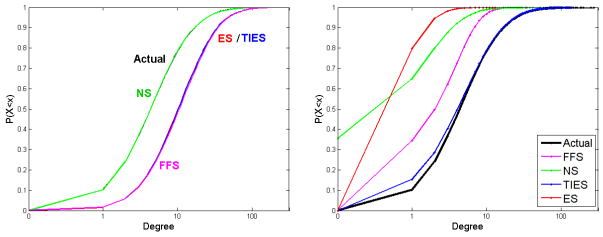
3.3.2 Induced graph formation

Now consider the graph induction process. Here instead of focusing on the degrees in the original graph d_k, we need to consider the *sampled* degrees in the induced (or partially-induced) graph G_s. Let d_k^s represent the sampled degree (in G_s) of a node that had degree k in the original graph G. Then, letting d_i refer to the degree of a neighboring node i:

$$\begin{aligned} E_{NS}[d_k^s] &= \textstyle\sum_k p_v = k \cdot p_v \\ E_{ES}[d_k^s] &= \textstyle\sum_{k-1} p_e \ + \ 1 = (k-1)p_e + 1 \\ E_{TIES}[d_k^s] &= \textstyle\sum_{i=1}^{k-1} \left[1-(1-p_e)^{d_i}\right] + 1 \end{aligned}$$

The sampled degree d_k^s depends on the manner in which the induced graph is formed. For NS, the graph is fully induced so the sampled degree depends on the probability that each neighbor is sampled. For ES, the induced graph consists of only the edges that were originally sampled in E_s. This means that sampled degree will be determined by the edge selection process. Note that the expectation is over $k-1$ neighbors since we know that in ES a minimum of one neighbor exists for each sampled node (i.e., the edge that added the node to the sample). For TIES, the induced graph consists of *all* edges that occur between the sampled nodes. In this case, the expected degree will be a function of the likelihood of the neighboring nodes' selection. Clearly the expected sampled degrees will be greater in TIES than in ES.

Note that all the expectations above are less than k, so this shows how the sampled degrees will underestimate original degrees for all the algorithms. TIES however, is less affected by this downward bias, due to its use of edge-based selection and induction process. We illustrate the difference between the sampled and original degrees, *i.e.*, degrees of the sampled nodes in the sample and original graphs, in Figure 1. The example shows that NS selects nodes in an unbiased manner with respect to their degrees in the original graph (see Figure 1a), but then, those degrees are underestimated in the sampled graph (Figure 1b) (*i.e.* NS curve shifts to left). In contrast, ES, FFS, and TIES overestimate the degrees in the original graph (1a). However, when the overestimation is combined with the FFS or TIES subgraph formation process, it results in a more accurate distribution of degrees in the sampled graph (1b) compared to NS. Because of its induction step, however, TIES compensates for the downward bias more than all other algorithms and thus, comes close to the original distribution.



(a) d_k (b) d_k^s

Figure 1: Illustration of original and sampled degrees for nodes selected in NS, ES, TIES, and FFS, on CondMAT network.

We can show how the downward bias in sampled degree has a larger impact on NS, by considering the expected number of nodes with sampled degree of 0.

LEMMA 3.3.2. *Let $E_*[|d_*^s = d|]$ refer to the expected number of nodes with degree d in the sampled graph $G_S = (V_s, E_s)$ Then $E_{NS}[|d_*^s = 0|] > E_{ES}[|d_*^s = 0|] = 0$.*

Proof: In ES, each node is added to the sample through the selection of one of its incident edges. Thus every node will have a minimum degree of 1 in the induced graph. In NS, the expected number of degree 0 nodes can be calculated by considering, for each selected node, the likelihood that *none* of its neighbors are chosen to be added to the sample: $E_{NS}[|d_*^s = 0|] = \sum_k f_D(k) \cdot p_v \cdot (1-p_v)^k > 0$. □

As p_v decreases, the likelihood that NS selects none of a node's neighbors increases, which will result in both an increase in nodes with degree 0, as well as an increase in the number of disconnected components. In contrast, TIES selects nodes through their incident edges, thus they will have the same property as ES and have a minimum sampled degree of 1 for each node in V_s.

Next, we consider the expected sample degrees of ES and NS, and show that ES will more accurately estimate lower degree nodes due to its selection of at least one neighbor for each sampled node.

LEMMA 3.3.3. *For degrees $k < \frac{1-p_e}{p_v-p_e}$ in the original graph G, the expected sample degree d_k^s will be larger for ES compared to NS: $E_{ES}[d_k^s] > E_{NS}[d_k^s]$.*

Proof: Consider the threshold k at which the expected sampled degree is greater for ES:

$$\begin{aligned} E_{NS}[d_k^s] &\leq E_{ES}[d_k^s] \\ 0 &\leq E_{ES}[d_k^s] - E_{NS}[d_k^s] \\ &= [(k-1)p_e + 1] - [k \cdot p_v] \\ k(p_v - p_e) &\leq (1-p_e) \\ k &\leq \frac{1-p_e}{p_v-p_e} \end{aligned}$$

Thus for values of $k < \frac{1-p_e}{p_v-p_e}$ the expected sample degree of nodes with degree k in G will be greater for ES compared to NS. □

For example, when $p_v = 0.20$ and $p_e = 0.05$, then $\frac{1-p_e}{p_v-p_e} = 6.33$, thus nodes with degree less than 6 will have larger induced degree in ES samples compared to NS samples. Since the induced degrees of TIES will be larger than the induced degrees of ES, this bound also applies to TIES.

Summary. The analysis above illustrates the reasons for the accuracy of TIES sampling. Based on its uniform sampling, NS will select nodes that accurately represent the original degree distribution (Lemma 3.3.1). However, since the nodes are sampled independently, the *sample* degrees in the NS induced graph will be

much lower than the original degrees (Lemma 3.3.2-3.3.3). Consequently, many of the low degree nodes will be disconnected in the sample due to expected degrees less than one. ES, on the other hand, samples high degree nodes more frequently than NS (Lemma 3.3.1), but since it only includes the selected edges in the sample graph the sampled degrees of those nodes will be even lower (since p_e is typically less than p_v). However, sampling of nodes via edges implies that the ES graphs are likely to be more connected than NS graphs (Lemma 3.3.3). In TIES, we add induction to the ES process, thus increasing the expected degrees of the sampled nodes. In addition, since high degree neighbors are also more likely to be included in the sample (and connected through induction), the clustering and connectivity of the sampled graphs will increase even further coming closer to the original graph.

3.4 Algorithm implementation

While so far our goal has been devising a sampling algorithm that preserves various graph characteristics, another dimension of importance is the implementation complexity. In particular, many real world networks are quite large and naturally evolve over time in a streaming fashion as edges are added over time, especially in the context of collaborative and sharing applications. In these environments, it is important that the sampling algorithm be amenable to a streaming implementation where the edge is either sampled or not and is not visited again in the future. To the best of our knowledge, the notion of streaming graph sampling algorithms has not been discussed before in literature, although streaming algorithms are generally quite popular among both database and networking communities. The following formulation captures our notion of *streaming graph sampling*.

We let $G(V, E_{[0,T)})$ represent the temporally-varying network graph, where V is the set of nodes and $E_{[0,T)}$ is the set of edges $e = (v_i, v_j, t)$, $v_i, v_j \in V$ and $t \in [0, T)$ is the timestamp of the edge. As each edge e arrives, the sampling algorithm π needs to decide whether to include the edge or not as the edge is *streamed* in. The sampling algorithm π may also maintain state Ψ, and consult the state to determine whether to sample a subsequent edge or not, but the total storage associated with Ψ should be of the order the size of the output sampled graph G_s, *i.e.*, $|\Psi| = O(|G_s|)$. Note that this requirement is potentially larger than the $o(N, t)$ (preferably, $polylog(N, t)$) that streaming algorithms typically require [21]. But, since the algorithm cannot require less space than the output, we relax this requirement in our definition as follows.

Definition 3.1. *We define a* streaming graph sampling algorithm *as any sampling algorithm π that produces a sampled graph G_s such that $|V_s|/|V| = \phi$, which (1) samples edges of the original graph $G(V, E_{(0,T)})$ in a sequential order (*i.e.*, not random access) in* one pass*; and, (2) maintains state Ψ that is of the order of the size of the sampled graph G_s,* i.e.*, $|\Psi| = O(|G_s|)$.*

From Algorithm 1, we can observe that TIES takes at least two passes through the data—first to select nodes by sampling edges at random, and second to form the induced graph by adding all edges among the sampled nodes. So, while the amount of storage maintained is not more than $O(G_s)$, the fact that it requires two passes through the data, violates our one-pass requirement through the data according to the criteria (1).

We can also observe that NS and FFS are not streaming algorithms either. NS requires at least two passes, one to select nodes probabilistically and another for graph induction, thus violating constraint (1). Implementing FFS as described in [18] requires maintaining the graph in more sophisticated data structures (to represent connectivity across nodes) than a simple sequential list of nodes, and also requires at least two passes through the data. This is since FFS can only determine the neighbor by looking at edges, which means, it needs to wait till *all* edges ((u, v_1), (u, v_2), etc.), are streamed in, the last of which can be as late as the last edge.

Although TIES is not a *strict* streaming algorithm, we note that we can transform into a streaming algorithm that requires only one pass through the data with a simple modification. Instead of full induction, we can utilize *partial* induction by combining the edge-based node sampling with the graph induction in Algorithm 1 into a single step. Specifically, the algorithm will simply run over the edges in a single pass, selecting each edge in a single pass with some probability p_e (assumed given for the moment) and if selected, add the incident nodes to the sampled graph. At each step, it will also add the edge if its two incident nodes are already in the sampled node set (to produce a partial induction effect). By combining node selection and induction, we can achieve single-pass streaming algorithm, we call partially-induced edge sampling (PIES) as shown in Algorithm 2.

Algorithm 2 PIES (probability p_e, edge set E)

```
Assume edges in E are sorted by time
▷ V_s = ∅, E_s = ∅
for k = 1 : |E| do
    ▷ (u, v) = e_k,
    if u ∈ V_s AND v ∈ V_s then
        E_s = E_s ∪ {e_k}
    else
        Sample e_k with probability p_e.
        if e_k is sampled then
            V_s = V_s ∪ {u, v}
            E_s = E_s ∪ {e_k}
        end if
    end if
end for
Output G_s = (V_s, E_s)
```

Proposition 1: *The expected sample degree of nodes in PIES will be bounded from below by the expected sample degree in ES and bounded from above by the expected sample degree in TIES.*

We omit the proof for brevity, but intuitively, we can see that PIES lies between the full induction of TIES and no induction in ES. This illustrates the relationship between ES, TIES, and PIES. The longer PIES collects edges among the sampled nodes, the more its sample graph will converge to that of TIES.

A key parameter in PIES is the edge sampling probability p_e with which edges are sampled at random. In this paper, we consider the online estimation and calibration of p_e as part of our future work. (For evaluation purposes, we assume we know the right value of p_e that gives us the target fraction of nodes.) One way to set the parameter in practice would be to calibrate p_e on a small portion of the initial sequence of edges to observe the relationship between p_e and ϕ, and then, generalize to the larger stream.

4. EXPERIMENTAL EVALUATION

In this section, we evaluate the efficacy of both our sampling algorithms, TIES and PIES, on several real data sets ranging from about 10,000 - 200,000 nodes, with from 30,000 - 1.3 million edges.

Data sets for analysis. In our experiments, we consider six real networks: a citation network, a collaboration network, two email communication networks, and two online social networks. For our

Graph Metric	HepPH	Twitter	Facebook (NO)	email-Enron	CondMAT	Email PU
Nodes	34,546	8,581	46,952	36,692	23,133	214,893
Edges	420,877	27,889	183,412	183,831	93,439	1,270,285
No. of components	61	162	842	1,065	567	24
Giant component	34,401	8,214	43,953	33,696	21,363	212,622
Average path length	4.33	4.17	5.6	4.03	5.35	3.91
Density	0.0007	0.0007	0.0002	0.0003	0.0004	0.000055
Clustering coefficient	0.146	0.061	0.085	0.085	0.264	0.0018

Table 1: Characteristics of Networks

evaluations, we focus mainly on simplified, undirected graphs, with only one edge between any pair of nodes, and without self-loops to facilitate fair comparison with prior work (FFS), but our results generally hold for unsimplified graphs. Table 1 summarizes the characteristics of the (simplified) real networks.

The three data sets titled HepPH, CondMAT, and Enron correspond to a citation graph, collaboration graph, and email communication graph respectively, provided by Leskovec *e*t al. [2]. The Facebook data corresponds to Wall communications among users that belong to a city collected by Mislove *et al.* [26]. The Twitter dataset contains tweets of users in discussion surrounding the United Nations climate change conference in Dec. 2009. Finally, the Purdue University email data corresponds to two weeks of data we collected from the email logs on the Purdue mailserver(s), where we considered Purdue accounts that had at least one incoming and outgoing edge in the trace.

Note that while the main focus of our paper is on the activity graphs in social networks (*e.g.*, Facebook, email, and Twitter graphs fit this category), we also examined other types of data sets (*e.g.*, citation and collaboration graphs) to demonstrate the generality and wider applicability of our algorithms and approach.

Evaluation measures. Our evaluation is primarily along four main properties—degree, path length, clustering coefficient, and size of weakly connected components. We measure the performance of a sampling algorithm by how well the sampled subgraphs preserve the probability density function (PDF) and cumulative distribution function (CDF) of each of these four properties. Unlike other measure based on aggregate statistics (*e.g.*, density, reciprocity), these four measures represent the distribution of properties across the nodes and edges in the sample, which facilitates detailed comparison and evaluation of sample representativeness.

In addition to visually comparing the similarity of the distributions on the sampled subgraphs to those of the original networks, we also compute two statistics to compare the distributions quantitatively. First, we use the Kolmogorov-Smirnov (KS) statistic to assess the distance between two CDFs. The KS-statistic is a widely used measure of the agreement between two distributions; the authors of [18] also have used the KS distance to illustrate the accuracy of FFS samples in the past. It is computed as the maximum vertical distance between the two distributions, where x represents the range of the random variable and F_1 and F_2 represent two CDFs: $KS(F_1, F_2) = max_x|F_1(x) - F_2(x)|$. Second, we use the skew divergence [15] (SD) to assess the difference between two PDFs. Skew divergence is used to measure the Kullback-Leibler (KL) divergence between two distribution that do not have continuous support over the range of values (*e.g.* skewed degree). KL measures the average number of extra bits required to represent samples from the original distribution when using the sampled distribution. However, since KL divergence is not defined for distributions that have some values with zero probabilities, skew divergence *smoothes* the PDFs before computing the KL divergence: $SD(P_1, P_2, \alpha) = KL[\alpha P_1 + (1-\alpha)P_2 \,||\, \alpha P_2 + (1-\alpha)P_1]$. The results shown in [15] indicate that using SD yields better results than other methods to approximate KL divergence on non-smoothed distributions. In this work, as in [15], we use $\alpha = 0.99$.

4.1 Results

In our experiments, we focus on obtaining a sample between 5–50% ($\phi = 0.05$ to 0.50) of the original graph. We picked this sampling range to illustrate how the different sampled graphs (produced by different sampling algorithms) converge to match the properties of the original graph as we increase the sampling fraction. For each sample fraction, we experiment with ten different runs, and in each run, we generate a sample from a new random seed. For the case of PIES, we randomly sort the edges of the graph in each run to simulate the streaming aspect of time-evolving graphs.

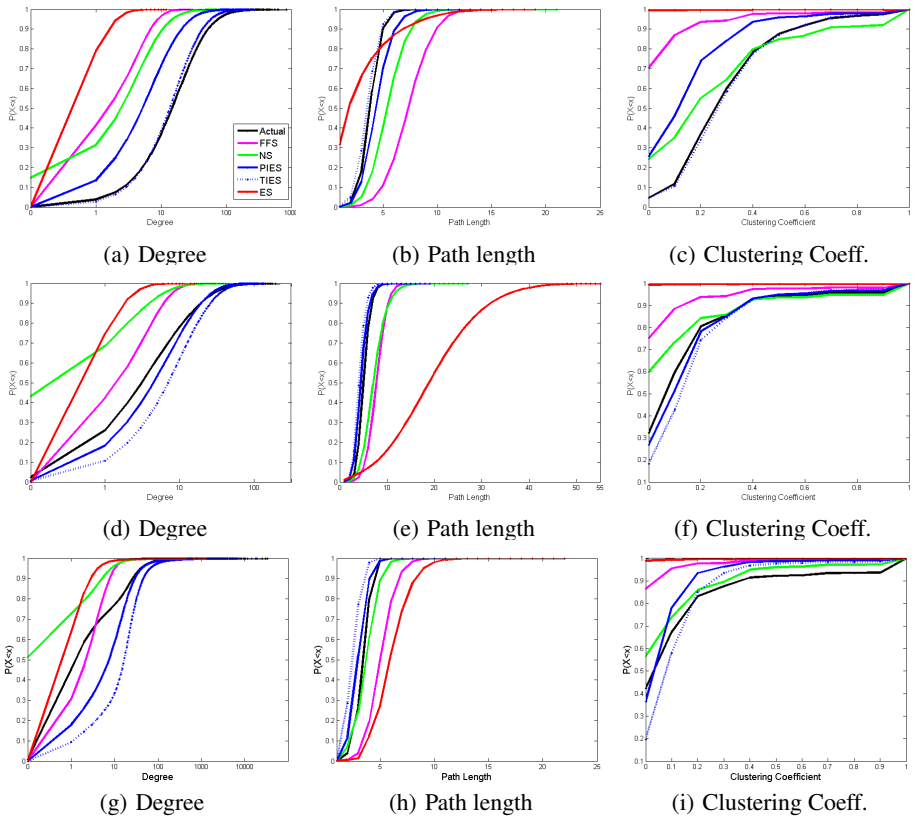
We first compare these algorithms visually based on their cumulative distributions—for degree, path length, and clustering coefficient distributions. We then compute the average KS and SD distances, across the ten different runs and the six networks. We plot both the averages and the standard errors.

Distributions. We plot the distributions of the three metrics in Figure 2 for HepPH (a-c), Facebook (d-f), and Email PU (g-i) at 20% sampling fraction. We picked the 20% sampling fraction as a reasonable sample size to show the difference between the distribution of the different sampling algorithms. However, other sampling proportions show similar relative behavior among the algorithms. Note that, due to the space limitations, we don't show the plots for the other three datasets, but we include their results when we compute the average KS and SD statistics.

Degree distribution. Figures 2(a), 2(d), and 2(g) show the degree distribution for the three networks. From the figures, we can observe that NS under-estimates the degree of the nodes, resulting in a large fraction of zero-degree (low-degree) nodes in its sample across the three networks. FFS often exhibits a similar characteristic, although it is better than NS on both Facebook and Email PU. However, NS performs better than FFS in the case of HepPH.

For two of the three networks, TIES and PIES are clearly more accurate at preserving degree distributions than either NS or FFS. As expected (and proved in Lemma 3.3.1), both PIES and TIES capture higher degree nodes than NS and FFS. However, with the induced graph formation step, the expected sampled degree of the nodes is higher in both PIES and TIES than ES—which allows them to match the degree distributions more accurately. However, the results for the Email PU network are less clear. We note that the Email PU network has a high proportion (>50%) of degree 1 nodes. While FFS is able to estimate the amount of low degree nodes better than PIES and TIES, at the same time, FFS underestimates the amount of high degree nodes compared to PIES and TIES.

Although PIES seems to perform slightly better than TIES for Facebook and Email PU, it is opposite for the HepPH network. This is likely because both Facebook and Email PU are less dense compared to HepPH (see Table 1). Since TIES uses total induction,



(a) Degree (b) Path length (c) Clustering Coeff.

(d) Degree (e) Path length (f) Clustering Coeff.

(g) Degree (h) Path length (i) Clustering Coeff.

Figure 2: Results at 20% sampling fraction: (a-c) HepPH, (d-f) Facebook, (g-i) Email Purdue.

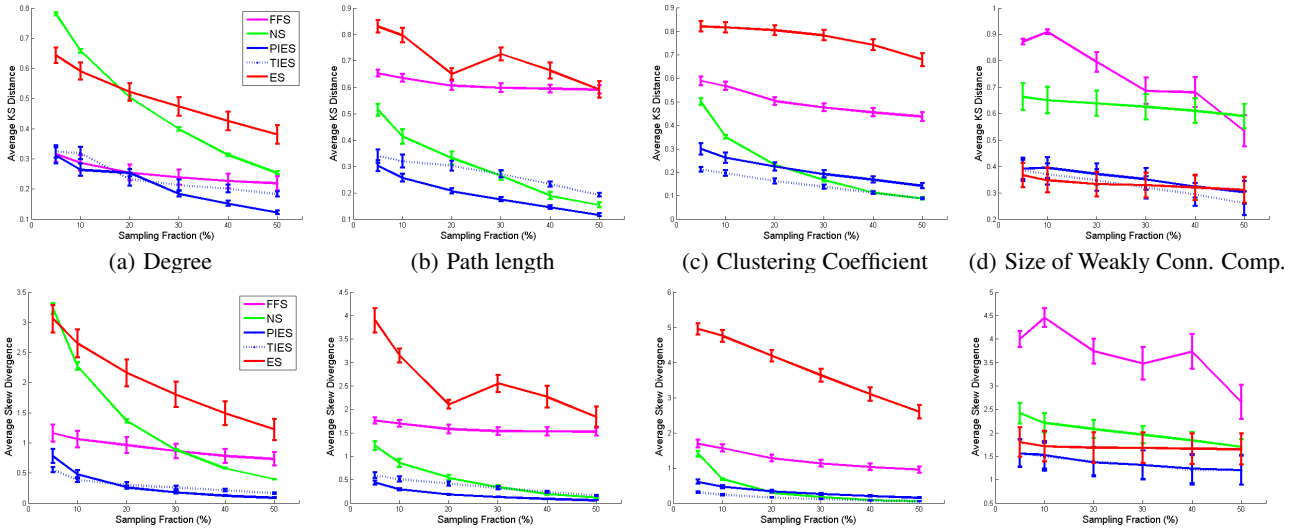
it captures more edges in the sample than PIES, therefore, which indicates that it will estimate the degree distribution more accurately than PIES for more dense graphs, but it will overestimates the degree distribution for the less dense graphs. This is an interesting result, since it appears that total induction may not always be the best sampling approach, and that partial induction may be desirable not just for its lower implementation complexity, but even for its ability to better match the degree distributions of the original graph than TIES for certain data sets. In general, it appears that we might be able to devise a 'tuning knob' to control the amount of induction to better match the properties of the original graph; we however leave such exploration as part of our future work.

Path length distribution. Figures 2(b), 2(e), and 2(h) show the path length distribution for the three networks. From the figures, we observe that FFS and NS samples have a high fraction of long path lengths compared to TIES, and PIES. This illustrates the effect of the induced graph formation which enhances the overall connectivity of the sampled graph, and hence produces shorter (and more accurately matching) path lengths. This explains why NS also typically performs better than FFS (due to its increased graph induction). But, graph induction alone is not sufficient, as we can observe both TIES and PIES perform better than NS, because of the ability of these algorithms to select high degree nodes.

Clustering coefficient. In the case of clustering coefficient (as shown in Figures 2(c), 2(f), and 2(i)), FFS shows a high fraction of low (zero) clustered nodes since it explores on average only 2.3 nodes from the neighbors of the burned node. FFS also tends to miss several edges among the sampled nodes. In order to further emphasize the effect of the induced graph step, we observe that NS, TIES and PIES perform better than FFS. We also observe that TIES performs better than NS and PIES in HepPH. However, PIES performs slightly better than TIES and NS on Facebook. The possible reason behind this observation is that Facebook exhibits less clustering and thus, the partial induction process of PIES would produce samples that match the original distribution better. On the other hand, the total induction process of TIES could overestimate the clustering distribution. Also notice that NS tends to produce a high fraction of low clustered nodes since the nodes are selected uniformly independent from the graph.

Summary. With a few exception, both TIES and PIES outperform NS, FFS and ES in the distributions of degree, path length, and clustering coefficient, across the three datasets. The edge-based node selection feature helps TIES and PIES to capture the high degree nodes, while the induced graph formation feature enhances the overall connectivity of the sampled graph. NS underestimates the degree of the nodes since it selects the nodes uniformly from the graph. However, the induced graph formation step helps NS to capture the clustering coefficient better than FFS. FFS matches the degree distribution better than NS, but it tend to miss several edges based on its burning process. Therefore, FFS should perform better if it is combined with the induced graph formation step.

In general, TIES performs slightly better than PIES, however, we conjecture that their performance is based on the properties of the original graph. If the graph is dense and highly clustered, then PIES will produce samples that underestimate the properties of the original graph based on its partial induction process. Therefore, TIES will perform better. On the other hand, if the graph is less dense and less clustered, then TIES will produce samples that overestimate the properties of the original graph. Thus, PIES will be better in this case. Note that, PIES is also amenable to a streaming implementation while TIES is not as, we discussed before. We aim to study the full induction versus the partial induction with a parameterized version of PIES in the future.



(a) Degree (b) Path length (c) Clustering Coefficient (d) Size of Weakly Conn. Comp.

(e) Degree (f) Path length (g) Clustering Coefficient (h) Size of Weakly Conn. Comp.

Figure 3: (a-d) Average KS distance and (e-h) average skew divergence, across 5 datasets.

KS-statistic. We compute the average of both of these measures across the five datasets and ten runs for each metric. Figures 3(a)–3(d) show the average KS-statistic as well as the standard errors for degree, path length, clustering coefficient and size of connected components respectively. Across the four metrics, we observe that both TIES and PIES outperform NS, FFS, and ES (by up to 2× lower than FFS). Overall, all sampling algorithms that include an induced graph step in their process perform well for the cases of path length and clustering coefficient as they capture more edges between the sampled nodes. ES, TIES and PIES perform better for the size of connected components due to their edge-based node selection feature. However, for the degree metric both the edge-based node selection and the induced graph formation are effective to help match the degrees of the original graph. Moreover, both TIES and PIES produce better quality samples than NS and FFS on small sampling fractions (5% – 20%).

Skew divergence. While KS-statistic observes the maximum distance between two distributions, the skew divergence shown in Figures 3(e)–3(h) captures the divergence across the entire range of values. We can observe that both TIES and PIES exhibit much lesser skew compared to other sampling algorithms including FFS. Specifically, we can observe upto 3-7× lesser skew than FFS in degree, path length and clustering. Among all metrics, component sizes are not as well preserved, but even here, the divergence is the least among all sampling algorithms.

5. RELATED WORK

The problem of sampling graphs has been of interest in many different fields of research. The work in [16, 29, 24] studies the statistical properties of samples of complex networks produced by traditional sampling algorithms like node sampling, edge sampling and random-walk based sampling and discusses the biases in estimates of graph metrics due to sampling. In addition, there have been a number of sampling algorithms in other communities such as in peer-to-peer networks [25, 10, 8], Internet modeling research community [13, 7, 4] and the WWW information retrieval community has focussed on random walk based sampling algorithms like PageRank [22, 12]. In social networks context, recent work [23] uses random walks to estimate node properties in G (e.g., degree distributions in online social networks). These different sampling algorithms large focused on estimating either the local or global properties of the original graph, but *not* to sample a representative subgraph of the original graph which is our goal. In the literature, the most closely related efforts are that of Leskovec *et al.* in [18] and Hubler *et al.* in [11], which were both discussed in Section 2.

Due to the popularity of online social networks such as Facebook [1] and Twitter [3], there has been a lot of work [20, 19, 17, 14, 5, 6] studying the growth and evolution of these online social networks. While most of them have been on static graphs, recent works [28, 27] have started focusing on interactions in social networks. These efforts focus more on characterizing social networks and thus are orthogonal to our research.

6. CONCLUSIONS

Much of the past efforts on sampling networks have focused on accurately estimating properties of the original graph. However, it is also important to have sampling mechanisms to select a *representative* subgraph for study and evaluation of real protocols and systems. Although there are recent algorithms for sampling subgraphs, these methods still fail to accurately capture many distributional properties of the original graph. We make the key observation that there is an inherent bias resulting from the subgraph formation process, leading to an underestimation of degrees and thereby, connectivity in the sampled subgraph. We propose a novel sampling approach based on edge-based node selection and graph induction that offsets this natural downward bias due to subgraph sampling, yielding samples that better match the distributions of graph properties in the sampled graphs with those of the original graph. Moreover, our method is simple and efficient to implement in a streaming fashion for large time-varying communication and activity graphs where edges accumulate over time (*e.g.*, email).

7. REFERENCES

[1] Facebook. http://www.facebook.com/.

[2] Stanford large network dataset collection. http://snap.stanford.edu/data/index.html.

[3] Twitter. http://www.twitter.com/.

[4] D. Achlioptas, A. Clauset, D. Kempe, and C. Moore. On the bias of traceroute sampling: or, power-law degree distributions in regular graphs. In *ACM STOC*, pages 694–703, 2005.

[5] Y. Ahn, S. Han, H. Kwak, S. Moon, and H. Jeong. Analysis of topological characteristics of huge online social networking services. In *WWW*, pages 835–844, 2007.

[6] H. Chun, H. Kwak, Y. Eom, Y. Ahn, S. Moon, and H. Jeong. Comparison of online social relations in volume vs interaction: a case study of cyworld. In *ACM/USENIX IMC*, pages 57–70, 2008.

[7] L. Dall 'Asta, I. Alvarez-Hamelin, A. Barrat, A. Vázquez, and A. Vespignani. Exploring networks with traceroute-like probes: Theory and simulations. *Theoretical Computer Science*, 355(1):6–24, 2006.

[8] S. Datta and H. Kargupta. Uniform data sampling from a peer-to-peer network. In *Proceedings of ICDCS'02*, page 50, 2007.

[9] M. Gjoka, M. Kurant, C. Butts, and A. Markopoulou. Walking in facebook: A case study of unbiased sampling of osns. In *Proceedings of IEEE INFOCOM '10*, 2010.

[10] C. Gkantsidis, M. Mihail, and A. Saberi. Random walks in peer-to-peer networks. In *IEEE INFOCOM*, 2004.

[11] C. Hubler, H.-P. Kriegel, K. M. Borgwardt, and Z. Ghahramani. Metropolis algorithms for representative subgraph sampling. In *ICDM*, 2008.

[12] J. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM*, 46(5):604–632, 1999.

[13] V. Krishnamurthy, M. Faloutsos, M. Chrobak, J. Cui, L. Lao, and A. Percus. Sampling large Internet topologies for simulation purposes. *Computer Networks*, 51(15):4284–4302, 2007.

[14] R. Kumar, J. Novak, and A. Tomkins. Structure and evolution of online social networks. In *SIGKDD*, pages 611–617, 2006.

[15] L. Lee. On the effectiveness of the skew divergence for statistical language analysis. In *Artificial Intelligence and Statistics*, 2001.

[16] S. Lee, P. Kim, and H. Jeong. Statistical properties of sampled networks. *Physical Review E*, 73:016102, 2006.

[17] J. Leskovec, L. Backstrom, R. Kumar, and A. Tomkins. Microscopic evolution of social networks. In *SIGKDD*, 2008.

[18] J. Leskovec and C. Faloutsos. Sampling from large graphs. In *SIGKDD*, pages 631–636, 2006.

[19] J. Leskovec and E. Horvitz. Worldwide Buzz: Planetary-Scale Views on an Instant-Messaging Network. In *WWW*, 2008.

[20] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel, and B. Bhattacharjee. Measurement and analysis of online social networks. In *ACM/USENIX IMC*, 2007.

[21] S. Muthukrishnan. Data streams: algorithms and applications. *Foundations and Trends in Theoretical Computer Science*, 1(2), Aug. 2005.

[22] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. 1998.

[23] B. Ribeiro and D. Towsley. Estimating and sampling graphs with multidimensional random walks. In *ACM SIGCOMM Internet Measurement Conference*, Nov. 2010.

[24] M. Stumpf, C. Wiuf, and R. May. Subnets of scale-free networks are not scale-free: Sampling properties of networks. *Proceedings of the National Academy of Sciences*, 102(12):4221–4224, 2005.

[25] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger. On unbiased sampling for unstructured peer-to-peer networks. In *IMC*, pages 27–40, 2006.

[26] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Social Networks (WOSN'09)*, August 2009.

[27] B. Viswanath, A. Mislove, M. Cha, and K. P. Gummadi. On the evolution of user interaction in facebook. In *WOSN*, August 2009.

[28] C. Wilson, B. Boe, A. Sala, K. P. Puttaswamy, and B. Y. Zhao. User interactions in social networks and their implications. In *EuroSys*, pages 205–218, 2009.

[29] S. Yoon, S. Lee, S.-H. Yook, and Y. Kim. Statistical properties of sampled networks by random walks. *Phys. Rev. E*, 75(4):046114, Apr 2007.