Sampling Directed Graphs with Random Walks

Bruno Ribeiro[1], Pinghui Wang[2], Fabricio Murai[1], and Don Towsley[1]

[1]Computer Science Department
University of Massachusetts
Amherst, MA, 01003
{ribeiro, fabricio, towsley}@cs.umass.edu

[2]State Key Lab for Manufacturing Systems
Xi'an Jiaotong University
Xi'an P.R.China
phwang@sei.xjtu.edu.cn

***Abstract*—Despite recent efforts to characterize complex networks such as citation graphs or online social networks (OSNs), little attention has been given to developing tools that can be used to characterize *directed graphs* in the wild, where no pre-processed data is available. The presence of hidden incoming edges but observable outgoing edges poses a challenge to characterize large directed graphs through crawling. Unless we can crawl the entire graph or the directed graph edges are highly symmetrical, hidden incoming edges induce unknown biases in the sampled nodes. In this work we propose a random walk sampling algorithm that is less prone to these biases. The driving principle behind our random walk is to construct, in real-time, an undirected graph from the directed graph in a way that is consistent with the sample path followed by the algorithm walking on either graph. We also study out-degree and in-degree distribution estimation. Outdegrees are visible to the walker while in-degrees are hidden (latent). This makes for strikingly different estimation accuracies of in- and out-degree distributions. We show that our algorithm can accurately estimate out-degree distributions and show that no algorithm can accurately estimate unbiased in-degree distributions unless the directed graph is highly symmetrical.**

I. Introduction

Despite recent efforts to characterize complex networks such as citation graphs or online social networks (OSNs), little attention has been given to developing tools that can be used to characterize *directed graphs* in the wild, where no pre-processed data is available. A network is said to be directed when the relationships between its agents (users or profiles) may not be reciprocated. For instance, a Wikipedia [17] entry about Columbia Records cites Thomas Edison but Thomas Edison's entry makes no reference to Columbia Records.

The presence of hidden incoming edges but observable outgoing edges makes characterizing large directed graphs through crawling a challenge. An edge $b \to a$ is a hidden incoming edge of node a if $b \to a$ can only be observed from node b. For instance, in our earlier Wikipedia example about Columbia Records and Thomas Edison we cannot observe the edge "Columbia Records" $\to$ "Thomas Edison" from Thomas Edison's wiki entry (but this edge is observable if we access Columbia Records's wiki entry).

Unless we can crawl the entire graph, hidden incoming edges induce unknown biases in the sampled nodes. Moreover, there may not even be a directed path from a given node to all other nodes. Graphs with hidden outgoing edges but observable incoming edges exhibit essentially the same problem. For instance, in some countries individuals, politicians, and NGOs may be required to publicaly disclose their contributors but contributors are not required to publicaly list their contributions.

In this work we propose a random walk sampling algorithm that does not suffer from unknown sampling biases when partially crawling directed graphs with hidden incoming edges. More importantly, we present a method to unbias the samples. Our random walk algorithm resorts to two main principles to achieve unbiased samples:

- In real-time we construct an undirected graph using the directed nodes that are sampled by the random walker on the directed graph. The undirected graph role is to guarantee that at the end of the sampling process we can approximate the probability of sampling a node, even though incoming edges are not observed. The random walk proceeds in a way that the sample path of walking on the directed graph is consistent with the sample path followed by the algorithm when walking on the constructed undirected graph. Knowing the sampling probability of a node allows us to unbias the samples.
- A very limited amount of uniformly sampled nodes (less than 0.01 of all sampled nodes) to guarantee that different parts of the directed graph are explored.

Contributions

Our work makes two main contributions:

- Directed Unbiased Random Walk (DURW): Our random walk algorithm accurately estimates characteristics of large directed graphs through sampling.
- In-degree Distribution Estimation: We show that no unbiased estimator can accurately obtain the indegree distribution (recall indegrees are latent variables in the directed graph) of the datasets used in this work from sampled edges unless a large fraction of the graph is sampled or the graph is highly symmetric. This result is surprising as the average indegree and average outdegree are identical and the outdegree distribution can be accurately characterized.

Outline

The rest of the paper is organized as follows. Section II presents the graph model and some definitions used throughout this work. Section III presents our *DURW* algorithm and estimators. Section IV presents our outdegree distribution estimation simulation results on real world graphs. Section V presents an application of *DURW* on the Wikipedia network. Section VI shows that indegree distribution estimation is

inaccurate unless most of the graph is sampled (even with side information). Section VII reviews the related work. Finally, Section VIII presents our conclusions and future work.

II. Definitions and Problem Formulation

Let $G_d = (V, E_d)$ be a directed graph, where V is the set of nodes and E_d is the set of edges. Let $o(v)$ denote the number of edges out of node $v \in V$ (outdegree) and $i(v)$ denote the number of edges into node $v \in V$ (indegree). We seek to obtain both the outdegree distribution $\phi = (\phi_0, \phi_1, ..., \phi_R)$ and the indegree distribution $\theta = (\theta_0, \theta_1, ..., \theta_W)$, where ϕ_l is the fraction of nodes with outdegree l, θ_j is the fraction of nodes with indegree j, R is the largest outdegree, and W is the largest indegree.

The degree distribution of a large undirected graph can be estimated using random walks (RW) [7], [11], [13]. But these RW methods cannot be readily applied to directed graphs with hidden incoming edges, which is the case of a number of interesting directed networks, e.g., the WWW, Wikipedia, and Flickr.

To address these problems, we build a random walk with jumps under the assumption that nodes can be sampled uniformly at random from G_d (something not feasible for the WWW graph but possible for Wikipedia and Flickr). But why perform a random walk if we can sample nodes uniformly? There are two reasons for that: (1) Random walk is more efficient in networks where uniform node sampling is costly (e.g., Flickr). We denote the cost of random node sampling c. In networks where users have numeric IDs, the cost of uniformly sampling comes from the fact that the ID space is sparsely populated [5], [6], [12] and a number of uniformly generated ID values are invalid. In these networks c is the average number of IDs queried until one valid ID is obtained. For instance, in the case of MySpace and Flickr, we estimate these costs to be $c = 10$ [12] and $c = 77$ (refer to our technical report [14]), respectively. We consider the cost of sampling an unsampled neighoring node to be one; the total sampling budget is B. (2) A random walk can better characterize highly connected nodes than uniform sampling as random walks are biased to sample highly connected nodes. This bias can be later corrected, giving us smaller estimation errors for the characteristics of highly connected nodes.

III. Sampling Directed Graphs with DURWs

Estimating characteristics of *undirected* graphs with random walks (RWs) is the subject of a number of recent works [11], [13], [15]. RW estimation methods presented in the literature require that $\forall u, v \in V$, the probability of eventually reaching u given that the walker is in v be non-zero. However, over a directed graph with hidden incoming edges this may not be true. For instance, consider a node $v \in V$ that has one outgoing edge but no incoming edges. If the random walker does not start at v then v is not visited by the walker (as the outgoing edge of v is a hidden incoming edge if some other node). On the other hand, a node $u \in V$ with no outgoing edges becomes a sink to the random walker.

A natural way to deal with the unreachability of nodes is to perform random jumps within the random walk, just like the PageRank algorithm [4]. The PageRank walker at node v jumps to a uniformly chosen node in the graph with probability α; and with probability $(1-\alpha)$ the walker performs a RW step (i.e., follows an edge chosen uniformly at random from the set of outgoing edges of v). Unfortunately, next we see that PageRank is not well suited to characterize directed graphs.

A. The case against PageRank sampling

Unfortunately, PageRank does not allow us to accurately estimate graph characteristics, such as the outdegree distribution, from a sampled subset of the graph. Estimating these characteristics requires obtaining the steady state distribution of the RW without exploring the entire graph [13].

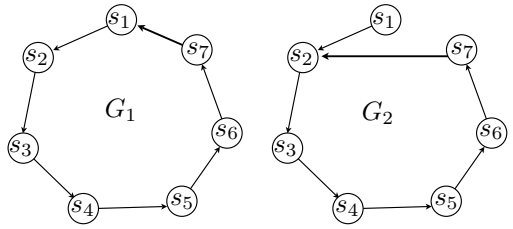


Figure 1: PageRank dependence on graph structure: Without sampling s_7 one cannot tell the PageRank sampling probability of node s_1.

In the example of Figure 1 we see that the steady state distribution of PageRank requires knowing the graph structure. Consider the two directed graphs, G_1 and G_2, with 7 nodes each, as shown in Figure 1. Incoming edges are hidden. Let s_1 be the starting node of PageRank. Let $\pi(v)$ denote the steady state probability that PageRank visits node v, $\forall v \in V$. For graph G_1 $\pi(s_1) = 1/7$ and for graph G_2 $\pi(s_1) = \alpha/(7\alpha+6)$. Thus, the sampling probability of s_1 depends on the edges of the unsampled node s_7. The above example shows why PageRank is not suited to sample large graphs.

B. Directed Unbiased Random Walk (DURW)

Our Directed Unbiased Random Walk (DURW) algorithm has two parts:

- **Backward edge traversals** (detailed in Section III-C): We allow the random walker to traverse known outgoing edges backwards under certain conditions. For instance, if at the i-th step the RW is at node s_i, we allow the random walker to traverse the edge $s_{i-1} \rightarrow s_i$ backwards. However, in order to avoid large transients the algorithm places some restriction on which edges can be made undirected.
- **Degree-proportional jumps** (detailed in Section III-D): The algorithm performs a jump from node v to an uniformly chosen node, $\forall t \in V$, with probability $w/(w + \deg(v))$, where $\deg(v)$ is the degree of v in the undirected graph G_u. Our jumping algorithm is subtle but fundamentally different than other random jump algorithms such as PageRank.

C. Backward edge traversals

We allow the walker to traverse some outgoing edges backwards. In general, if we apply this "backward walking" principle to all outgoing edges in G_d, we can construct an

undirected version of G_d. The undirected version of G_d allows us to apply the techniques described in Ribeiro and Towsley [13] to estimate the characteristics of G_d such as the outdegree distribution. However, the degree of a node v in the final undirected version of G_d is only known after exploring all edges of G_d. Thus, the above sampling algorithm is not practical as unbiasing the sampled would require access to the complete underlying graph (as the probability is a function of v's degree [13]).

To avoid this problem our RW interactively builds an undirected graph G_u. This building process is such that once a node is visited at the i-th step no additional edges are added to that node in subsequent steps. Such a restriction fixes the degree of the nodes visited by the random walker, thus ensuring that nodes will not keep changing their degrees as we walk the graph. This is an important feature to reduce the unknown bias of the random walk transient and thus reducing estimation errors. Note that the final undirected graph G_u depends on the sample path taken by the random walker. Further details of the algorithm can be found in Section III-E.

The above solution addresses the problem of knowing the degree of a node as soon as the node is sampled. However, we still do not know the steady state distribution of the RW when we add random jumps. In what follows we present an algorithm that allows us to obtain a simple closed-form solution to the steady state distribution.

D. Degree-proportional jumps

Let $G_u = (V, E_u)$ be an undirected graph. In DURW, the probability of randomly jumping out of a node v, $\forall v \in V$, is $w/(w + \deg(v))$, $w > 0$. This modification is based on a simple observation: let G' be a weighted undirected graph formed by adding a *virtual* node σ to G_u such that σ is connected to all nodes in V with edges having weight w. All remaining edges have unitary weight. In a weighted graph a walker transverses a given edge with probability proportional to the weight of this edge. The steady state probability of visiting a node v on G' is $(w+\deg(v))/(\text{vol}(V)+w|V|)$, where $\text{vol}(V) = \sum_{\forall u \in V} \deg(u)$. Thus, except for the unknown constant normalization term $(\text{vol}(V)+w|V|)$, the steady state distribution of v is known as we know the degree of v and the value of w when v is visited by the random walker.

By combining *backward edge traversal* (Section III-C) and *degree-proportional jumps* (Section III-D) we obtain the DURW algorithm.

E. The DURW algorithm

DURW is a random walk over a weighted undirected connected graph $G_u = (V, E_u)$, which is built on-the-fly. The algorithm works as follows. We build an undirected graph using the underlying directed graph G_d and the ability to perform random jumps. Let $G^{(i)} = (V^{(i)}, E^{(i)})$ be the constructed undirected "graph" at DURW step i, where $V^{(i)}$ is the node set and $E^{(i)}$ is the edge set. We call $G^{(i)}$ a "graph" because we allow $E^{(i)}$ to have edges of nodes that are not in $V^{(i)}$. Denote $G_u \equiv \lim_{i\to\infty} G^{(i)}$. In what follows we describe the construction of $G^{(i)}$.

Let $v \in V$ be the initial node in the random walk. Let $\mathcal{N}(v)$ denote the outgoing edges of v in G_d and let node σ denote a virtual node that represents a random jump. We initialize $G^{(1)} = (\{s_1\}, E^{(1)})$, where $E^{(1)} = \mathcal{N}(s_1) \cup \{(u, \sigma) : \forall u \in V\}$, where $\{(u, \sigma) : \forall u \in V\}$ is the set of all undirected virtual edges to virtual node σ (this construct of adding edges to σ is introduced to simplify our exposition, in practice we do not need to add virtual edges to σ). Note that we allow self loops created when $\sigma = s_1$. The random walker proceeds as follows.

We start with $i = 1$; at step i the random walker is at node s_i. Let

$$W(u, v) = \begin{cases} w & \text{if } u = \sigma \text{ or } v = \sigma \\ 1 & \text{otherwise} \end{cases}$$

denote the weight of edge (u, v), $\forall (u, v) \in E^{(i)}$, $i = 1, 2, \ldots$. The next node, s_{i+1}, is selected from $E^{(i)}$ with probability $W(s_i, s_{i+1})/\sum_{\forall (s_i,v)\in E^{(i)}} W(s_i, v)$. Upon selecting s_{i+1} we update $G^{(i+1)} = (V^{(i)} \cup \{s_{i+1}\}, E^{(i+1)})$, where

$$E^{(i+1)} = E^{(i)} \cup \mathcal{N}'(s_{i+1}), \tag{1}$$

and

$$\mathcal{N}'(s_{i+1}) = \{(s_{i+1}, v) : \forall (s_{i+1}, v) \in \mathcal{N}(s_{i+1}) \text{ s.t. } v \notin V^{(i)}\}$$

is the set of all nodes (u, v) in $\mathcal{N}(s_{i+1})$ where node v is not already in $V^{(i)}$. Note that $\mathcal{N}'(s_{i+1}) \subseteq \mathcal{N}(s_{i+1})$. By using $\mathcal{N}'(s_{i+1})$ instead of $\mathcal{N}(s_{i+1})$ in equation (1) we guarantee that no nodes in $V^{(i)}$ change their degrees, i.e., $\forall v \in V^{(i)}$ the degree of v in $G^{(i)}$ is also the degree of v in G_u. Thus, we comply with the requirement presented in Section III-C that once a node v, $\forall v \in V$, is visited by the RW no edges can be added to the graph with v as an endpoint.

The edges in $G^{(i)}, i = 1, 2, \ldots$, that connect all nodes to the virtual node σ can be easily emulated with uniform node sampling.

Space complexity: The space required to store $G^{(i)}$ is $O(|E|)$, where $|E|$ is the number of edges in the graph.

F. Outdegree Distribution Estimator

In this section we use the nodes visited (sampled) by our *DURW* algorithm to estimate the outdegree distribution. The estimator presented in this section can be easily extended to obtain the distribution of node labels, as detailed in Section III-G.

Let s_i denote the i-th node visited by DURW, $i = 1, \ldots, n$, $n \geq B$. Let ϕ_j be the fraction of nodes with outdegree j in G_d. Let $\pi(v)$ be the steady state probability of sampling node v in G_u, $\forall v \in V$. The outdegree distribution can be estimated as

$$\hat{\phi}_j = \frac{1}{n} \sum_{i=1}^{n} \frac{h_j(s_i)}{\hat{\pi}(s_i)}, j = 0, 1, \ldots \tag{2}$$

where $h_j(v)$ is the indicator function

$$h_j(v) = \begin{cases} 1 & \text{if the outdegree of } v \text{ in } G_d \text{ is } j \\ 0 & \text{otherwise} \end{cases},$$

and $\hat{\pi}(s_i)$ is an estimate of $\pi(s_i)$: $\hat{\pi}(s_i) = (w + \deg(s_i))S$. Here $\deg(v)$ is the degree of v in $G^{(\infty)}$ and

$$S = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{w + \deg(s_i)}.$$

The following theorem shows that $\hat{\pi}(s_i)$ is asymptotically unbiased.

Theorem 3.1: $\hat{\pi}(s_i)$ is an asymptotically unbiased estimator of $\pi(s_i)$.

Proof: To show that $\hat{\pi}(s_i)$ an asymptotically unbiased we invoke Theorem 4.1 of Ribeiro and Towsley [13], yielding $\lim_{B\to\infty} S = |V|/(|E^{(\infty)}| + |V|w)$ almost surely. Thus, $\lim_{B\to\infty} \hat{\pi}(s_i) = \pi(s_i)$ almost surely. Taking the expectation of Equation (2) in the limit $B \to \infty$ yields $E[\lim_{B\to\infty} \hat{\phi}_j] = \phi_j$, which concludes our proof. ∎

G. Estimating other metrics

In a more general setting we seek to estimate the distribution obtained by the function

$$h_j(v) = \begin{cases} 1 & \text{node } v \text{ is labeled } j, \\ 0 & \text{otherwise.} \end{cases} \quad (3)$$

where labels can indicate any characteristics of the nodes.

In order to estimate the fraction of nodes with label j, we plug the values $h_j(s_i)$, $s_i = 1, \ldots, n$, $n \geq B$ into equation (2). Here $\hat{\pi}$ is computed in the same way as before and $E[\lim_{B\to\infty} \hat{\phi}_j] = \phi_j$, still holds.

Now that we have an asymptotically unbiased estimators it is left to test the accuracy of DURW in a variety of real world graphs.

IV. Experimental Results

This section compares the outdegree distribution estimates obtained by our algorithm (DURW) against the estimates obtained by the random walk algorithm of Bar-Yossef et al. [3] (presented in Section VII) and independent uniform node sampling (UNI). Our experiments are performed on a variety of real world graph datasets. The statistics of these datasets are summarized in Table I.

We now describe each dataset. Flickr, LiverJournal, and YouTube are popular photosharing, blog, and video sharing websites, respectively. In these websites a user (node) can subscribe to other user (nodes) updates forming a directed edge. Wikipedia is a free encyclopedia written collaboratively by volunteers. Each registered user has a talk page, that she and other users can edit in order to communicate and discuss updates to various articles on Wikipedia. Nodes in the Wiki-Talk dataset represent Wikipedia users and a directed edge from node u to node v represents that user u edited a talk page of user v at least once. The Web-Google dataset was released in 2002 by Google as a part of Google Programming Contest, where nodes represent web pages and directed edges represent hyperlinks between them [1]. Further details of the Flickr, LiverJournal, and YouTube datasets can be found in Mislove et al. [10].

Table I: Overview of directed graph datasets used in our simulations.

Graph	# nodes	# edges	E[out-deg]	symmetry	Type
Flickr [10]	1,715,255	22,613,981	18.1	0.38	OSN
YouTube [10]	1,138,499	4,945,382.	5.3	0.21	OSN
LiveJournal [10]	5,204,176	77,402,652	18.7	0.27	OSN
Wiki-Talk [2]	2,394,385	5,021,410	3.9	0.86	usr talk
Web-Google [1]	875,713	5,105,039	9.87	0.69	Web

Error Metric

Before we proceed with our results we need to introduce the error metric used to compare the different sampling methods we want to test. Our primary metric of interest is the outdegree distribution. We chose this metric because the outdegree distribution is a metric that is present in all of our datasets. Let

$$\text{NMSE}(\hat{\phi}_j) = \frac{\sqrt{E[(\hat{\phi}_j - \phi_j)^2]}}{\phi_j}, j = 1, 2, \ldots,$$

be a metric that measures the relative error of the estimate $\hat{\phi}_j$ with respect to its true value ϕ_j. **Note that our metric uses the relative error. Thus, when ϕ_j is small, we consider values as large as NMSE$(\hat{\phi}_j) = 1$ to be acceptable.** Let c denote the cost of UNI which is also the cost of a random jump (the average number of IDs queried until one valid ID is obtained). For instance, Flickr has a random node sampling cost of $c = 77$ (as observed in the experiments presented in [14]). Let B denote the sampling budget (when $c = 1$, B is the number of distinct sampled nodes). Because we create an undirected graph on the side, multiple visits to the same node counts as just one unit of the sampling budget. Also, before proceeding to our results, it is important to note that in DURW the probability of performing a random jump increases with w (such that in the limit $w \to \infty$ DURW is equivalent to UNI).

Results

The first simulation results are presented in Figures 2 to 4d. Figure 2 shows three sample paths with estimates of ϕ_{1000} (y-axis) using DURW (using random jump weights $w \in \{0.1, 1, 10\}$) and UNI. The x-axis shows the number of samples (in log scale). These results were obtained using the Youtube dataset with independent sampling cost $c = 1$. The choice of ϕ_{1000} is arbitrary; later we investigate estimation errors of all degrees. Note that none of the three runs of UNI can find a single node with outdegree 1000 until almost $200,000$ nodes have been sampled. On the other hand, our DURW algorithm quickly finds at least one node with outdegree 1000, an useful property in scenarios where we are interested in finding high degree nodes. Moreover, DURW has consistently smaller estimation errors than UNI throughout most of the sample paths.

Figures 3 to 4d show estimates of the NMSE **in log scale** (over 1000 runs) of DURW for all outdegrees in the graph over different datasets. The NMSE error is obtained over $10,000$ runs. In these simulations we compare the NMSE of DURW

with the NMSE of the Metropolis-Hastings algorithm of Bar-Yossef et al. [3] and the NMSE of uniform random sampling (UNI). In all first five scenarios we sample 10% of the graph. The DURW random jump weight and cost are $w = 10$ and $c = 10$, respectively. Figures 3, 4a, 4b, 4c, 4d show the NMSE for the Youtube, Wiki-Talk, Flickr, Livejournal, and Web-Google datasets, respectively.

We find that DURW obtains accurate estimates over all datasets, recalling that we consider NMSE$(\hat{\phi}_j) \leq 1$ to be an accurate estimator if ϕ_j is small, as the NMSE measures **the relative error** of $\hat{\phi}_j$ when compared to the true value ϕ_j. *DURW outperforms all other methods w.r.t. the accuracy when estimating the fraction of nodes with large outdegrees, often by a factor of one order of magnitude.* Moreover, DURW is significantly more accurate than Bar-Yossef et al. [3] over almost all outdegrees values. In the Youtube, Wiki-Talk, Flickr, and Livejournal datasets the value of $\hat{\phi}_j$ obtained with DURW for large outdegrees is up to two orders of magnitude more accurate than Bar-Yossef et al. [3]. DURW is also consistently more accurate over most outdegrees than UNI for the Youtube, Wiki-Talk and Flickr datasets. DURW is also more accurate than UNI in the Livejournal and Web-Google datasets for large outdegrees (outdegrees larger than 50 in Livejournal and outdegrees larger than 30 in Web-Google).

In Figures 3 to 4d we note that UNI is sometimes slightly better than DURW for low degrees, but not by much. This is because DURW is biased towards sampling nodes with high outdegrees and indegrees (these nodes tend to have high degrees in our constructed undirected graph). Thus, DURW tends to sample nodes with few outgoing edges less frequently than UNI, sometimes causing larger estimation errors for these small outdegree nodes. As we see later, increasing the DURW jump weight parameter w reduces the estimation error for small outdegrees at the cost of increasing the error of larger outdegrees.

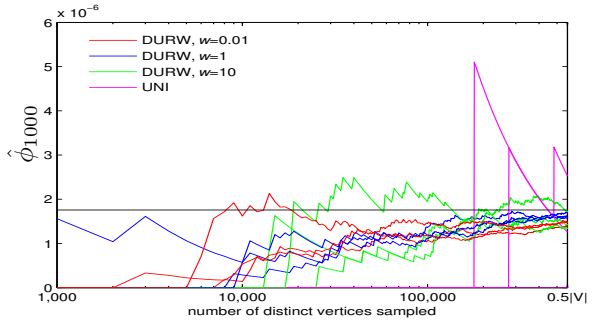


Figure 2: (YouTube) True value of $\phi_{1000} = 1.8 \times 10^{-6}$ shown in the black solid line. Y-axis: Estimated ϕ_{1000} (DURW with $w \in 0.1, 1, 10$ and UNI). X-axis: Number of samples in log-scale. DURW is consistently better than UNI.

A. Varying DURW Parameters

In what follows we study the impact of the DURW parameters on the accuracy of the estimates. Figure 5 shows the results of varying the random jump weight, w, while keeping the random jump cost $c = 10$ and the sampling budget $B = 0.1|V|$ fixed. The DURW parameter w controls the trade-off between

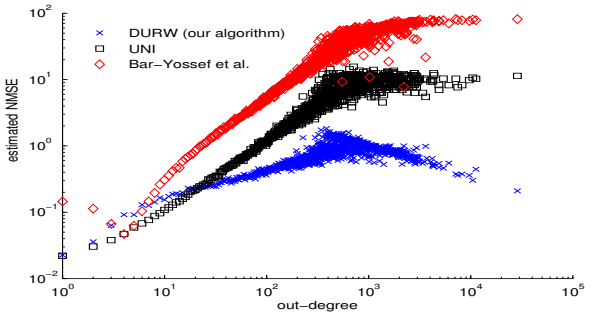
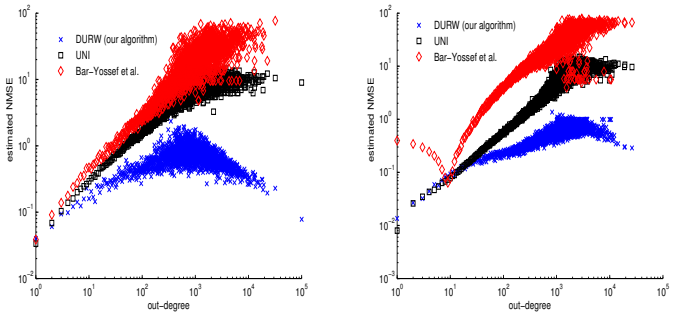
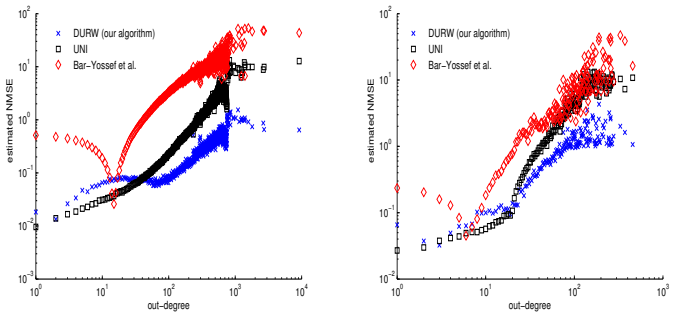


Figure 3: (YouTube) NMSE of DURW with $B0.1|V|$, $w = 10$, and $c = 10$ compared with UNI.



(a) Wiki-Talk (b) Flickr



(c) LiveJournal (d) Web-Google

Figure 4: NMSE of DURW with $B = 0.1|V|$, $w = 10$, and $c = 10$ compared with UNI.

the error for estimating small and large outdegrees. As w increases, the estimation error for large outdegrees increases while the error of small outdegrees decreases. Conversely, as w decreases the opposite happens, the estimation error of large outdegrees decreases while the error for small degrees increase. This is expected, as when w increases the DURW algorithm performs random jumps often and, thus, mimics uniform sampling (UNI), sampling small outdegrees nodes more often.

In the next set of simulations we look at the impact of the sampling budget, B, on the NMSE. Figure 6 presents the NMSE of Youtube for sampling budgets $B \in \{0.01|V|, 0.1|V|, 0.2|V|\}$ with $c = 10$ and $w = 10$. We observe that the error of sampling B nodes is roughly proportional to $1/\sqrt{B}$. For instance, in Figure 6 we see that a one order of magnitude increase in B roughly decreases the

error by $1/\sqrt{10}$.

As DURW can perform random jumps, we also study the impact of the cost of these jumps over the accuracy of the estimates. The cost of a jump is measured by the amount of "sampling budget" (queries) required to perform the jump. On some social networks, such as MySpace and Flickr, a number of queries is needed to sample a node uniformly at random. For instance, on Flickr random jumps are performed by querying randomly generating user IDs. The average number of random IDs queried until one valid ID is obtained is 77 to 1 (see our technical report [14]). The cost of jumps effectively reduces the number of total nodes that can be sampled, and, thus, increases the NMSE. Figure 7a shows the NMSE of Youtube with $c \in \{1, 10, 77\}$ and constant values $B = 0.1|V|$ and $w = 10$. Unsurprisingly, we observe that the estimation error of the outdegree distribution tail increases with c. As decreasing w decreases the frequency of jumps, which reduces the impact of parameter c the NMSE error. To better understand this relationship, we repeat the above experiment (Figure 7a) with less frequent jumps $w = 1$. In Figure 7b we plot the NMSE of the DURW algorithm with different values of $c \in \{1, 10, 77\}$ and keeping $w = 1$ constant. Comparing Figures 7a and 7b we see that a smaller w can significantly lessen the negative impact of higher jump costs on the accuracy of the estimator.

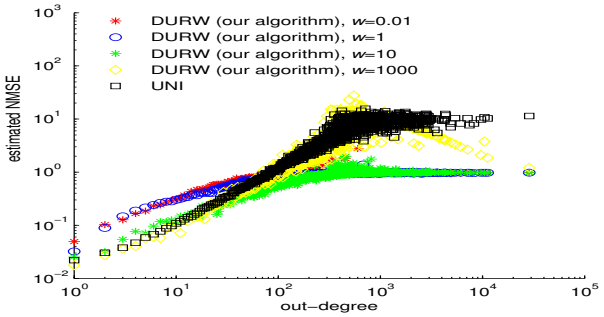


Figure 5: (Youtube) NMSE of DURW with $w \in \{0.01, 1, 10, 1000\}$ against the NMSE of UNI, $c = 10$ and $B = 0.1|V|$.

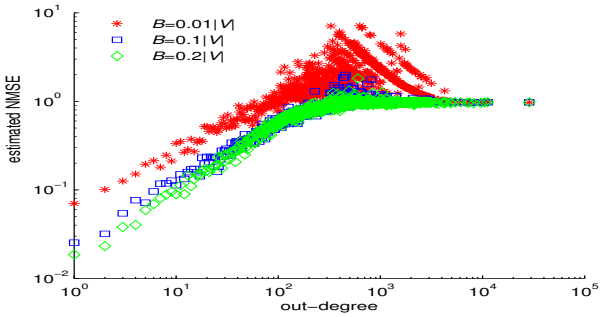
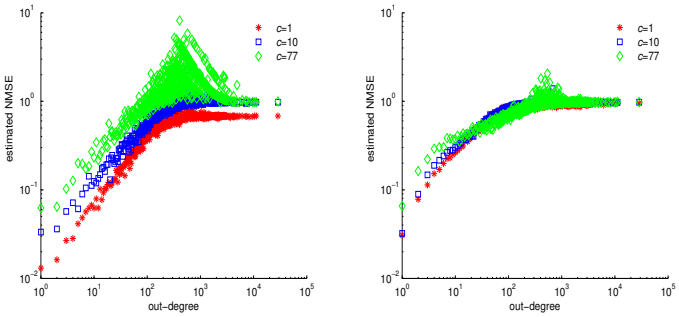


Figure 6: (Youtube) NMSE of DURW with $B \in \{0.01|V|, 0.1|V|, 0.2|V|\}$, $c = 10$ and $w = 10$.

V. Applications

In this section we describe some experiments performed to estimate other metrics evaluated at the Wikipedia network. As opposed to the results presented in the previous section, which



(a) $w = 10$ (b) $w = 1$

Figure 7: (Youtube) NMSE of DURW with $c \in \{1, 10, 77\}$, $B = 0.1|V|$ and $w = \{1, 10\}$.

are based on datasets, we now use a DURW to crawl the graph in an online fashion.

These metrics are interesting if measured online: (1) the time between two consecutive article (node) revisions and (2) the distribution of the number of revisions in the last 7 days, classified by the outdegree of the node (article). But why do we want to sample Wikipedia revisions using DURW? *Articles that point to several other articles are likely to suffer more revisions than nodes with almost no references.* As DURW can accurately estimate metrics of nodes with high outdegrees (see Section IV), DURW is well equipped to estimate revision-related metrics.

Wikipedia provides a query API that can be used to obtain information from an node such as the categories to which it belongs, its revision timestamps (timestamps marking when the node was changed) and the content itself, which includes the text describing the node and links to other Wikipedia nodes. We implemented a crawler that uses this query API to perform a DURW random walk on the Wikipedia graph. For the DURW uniform random jumps we use the handy "show me a random article" function of the Wikipedia API. We set the DURW jump weight to $w = 0.1$, allowing the walker to perform some random jumps while preserving most of the random walk characteristic of sampling high degree nodes. We are interested in sampling high degree nodes well because these nodes, although rare, are likely to be the ones that are edited more frequently. Our DURW crawler collected 60,000 Wikipedia nodes (approximately 2% of the entire Wikipedia database) over the course of two days (from 07/27/2011 5pm until 07/29/2011 5pm).

When an node is visited, the crawler also gathers measures of interest such as the outdegree and the times of the 500 most recent revisions. For conciseness, we omit the results regarding the outdegree (see Section III-G for a description of how to estimate the outdegree distribution). In particular, the metrics we obtain include:

- $X(v)$: number of revisions of v created in date $d \in [R(v) - 7 \text{ days}, R(v)]$, and
- $Y(v)$: time (in days) that has passed between the last revision of v and $R(v)$,

where $R(v)$ is the time when the node v was first retrieved.

Following the estimator presented in Section III-G, we adapt equation (3) to estimate the distribution of the number of revisions in the last 7 days (denoted *recent revision*) since the node was first accessed binned by node outdegree. We group the node outdegrees in bins such that the fraction of nodes (in the entire Wikipedia) is roughly the same in each bin. Note that we use the estimated outdegree distribution to specify the bins. Hence, we have

$$h_{(b,j)}(v) = \begin{cases} 1 & \text{node } v \text{ in bin } b \text{ has } j \text{ recent revisions,} \\ 0 & \text{otherwise.} \end{cases}$$

Our bins are chosen such that each bin has approximately $10,000$ sampled nodes (articles). Figure 8 plots the average value of R per bin. Observe that the nodes with the highest outdegrees are also those which are update more often. High outdegree nodes are clearly important in Wikipedia.

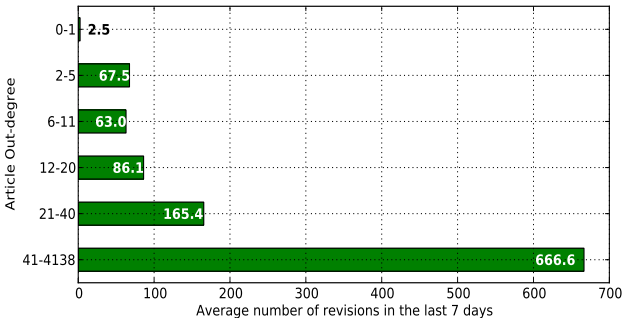


Figure 8: (Wikipedia) Average number of revisions in the last 7 days binned by node outdegree.

Figure 9 depicts the estimated CCDF of Y as a solid red line. Note that the majority of the nodes were updated in the last 100 days. Now suppose that we want to obtain an estimate of the time between revisions at the instant we are sampling Wikipedia. The difference between the average time between revisions in a long period of time and the instantaneous time between revisions is parallel to the difference between average speed of a car over, say, 100 miles and its instantaneous velocity.

Note that Y is not the time between revisions due to the *inspection paradox*, described as follows. Let Z be the time between revisions (in days) at the time we sample Wikipedia. Let $\zeta_z = P[Z = z]$ and $\zeta_{\geq z} = P[Z \geq z]$. Assume that we arrive to sample an node uniformly between two revisions (a reasonable assumption). The probability of landing on an inter-revision time of size $Z = z$ is $(z\zeta_z) \,/\, (\sum_{k=1}^{\infty} k\zeta_z)$. Now we can relate Y (the residual time) to Z

$$P[Y = y] = \sum_{z=y}^{\infty} \frac{1}{z} \frac{\zeta_{\geq z}}{E[Z]}$$

The maximum log-likelihood estimator of ζ_z is

$$\underset{\{\hat{\zeta}_z\}_{\forall z}}{\operatorname{argmax}} \sum_{\forall y} f(y) \frac{1 - \sum_{z=1}^{y-1} \hat{\zeta}_z}{\sum_{k=1}^{\infty} k\hat{\zeta}_k},$$

where $f(y)$ is the number of samples of Y with value y. We also need to enforce the constraints $0 \leq \hat{\zeta}_z \leq 1$, $z = 1, 2, \ldots$ and $\sum_{\forall z} \hat{\zeta}_z = 1$. Figure 9 shows the estimated distribution $\hat{\zeta}_z$. Observe that more than 10% of the nodes have time between revisions greater than 100 days.

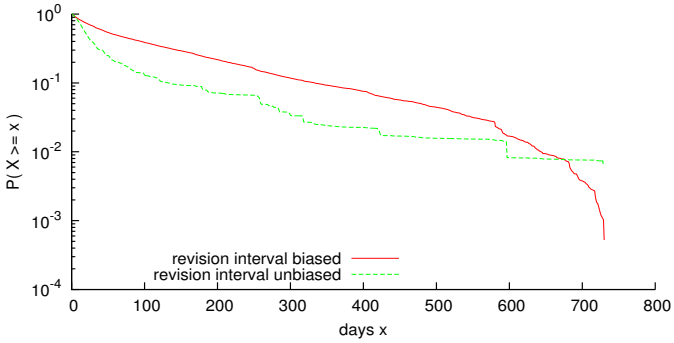


Figure 9: (Wikipedia) Estimated distribution of time between revisions. The red solid curve shows the raw results obtained by the DURW algorithm and the green dashed curve shows the results with their inspection paradox bias removed.

VI. Estimating Latent In-degree Distributions

The approach used above to estimate the outdegree distribution can also be used to estimate the indegree distribution if indegrees are visible to the random walker. However, in this section we consider a much harder problem: estimating the indegree (outdegree) distribution when indegrees (outdegrees) are hidden. Unfortunately, our results are negative. We show that in the presence of hidden incoming edges one needs to sample most of the edges of the graph in order to obtain an accurate indegree distribution estimate. Here the indegree distribution is an example of a latent graph characteristic. A latent graph characteristic is one that cannot be directly observed but is rather inferred (through a mathematical model) from other observable variables.

A random walk on the undirected graph G_u samples its edges uniformly at random. In what follows we simplify our analysis by assuming edges sampled by our DURW algorithm are sampled independently. In reality we can achieve near independence using large random jump weights, w. But even without large w, independent edge sampling has been successfully used before to model random walk-based sampling [13] and inspired our independence assumption.

First let's obtain the likelihood function of the indegree distribution. An edge incident to u, $v \to u$, can be observed by sampling node v. After sampling a fraction p of the graph, an average fraction of p of the edges incident to u are sampled. Using the partially reconstructed indegree of u (and later the estimated outdegree distribution) we can reconstruct the original indegree distribution. Making this statement more formal: Let i be the indegree of a given node u and let X be a random variable that denotes the number of sampled incoming edges of u if edges are sampled independently and with probability p.

The above model is a simplification of our original model. Independent edge sampling is different than sampling a fraction p of the nodes and then getting their outgoing edges. In

the former a node can have multiple outgoing edges, making edge samples dependent. Nevertheless, having worked with both models in the past, we notice little practical difference in large networks. This is because we seek an error lower bound on the estimation error and, in practice, a model with greater independence also provides a *lower bound* for the model with greater dependence when the dependence in the data is unknown. It is easy to see that

$$P[X = j] = b(j, i) = \binom{i}{j} p^j (1-p)^{i-j}\ ,\ j = 1, 2, \dots \quad (4)$$

where $b(j, i) = 0$, $\forall j > i$.

Now we can estimate the indegree distribution. The details of our maximum likelihood estimator used in this work can be found in our technical report [14]. We limit our estimator to a maximum indegree of 50 (i.e., we remove vertices with more than 50 incoming edges from the graph) to simplify the estimation procedure, which is computationally intensive. Figure 10 shows the true indegree distribution of the Flickr network (black line with asterisk) and the indegree distribution estimates for different sampling probabilities $p \in \{0.1, 0.5, 0.9\}$. Note that while the estimates with an average of 90% of the edges sampled ($p = 0.9$) are reasonable for small indegrees (but still not accurate for higher degrees), the estimates with sampling probabilities $p = 0.1$ and $p = 0.5$ are unstable.

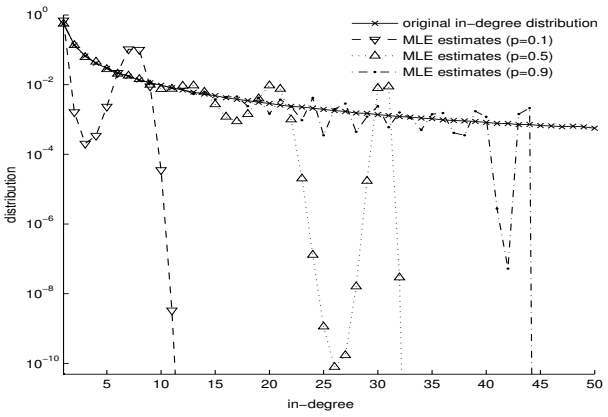


Figure 10: (Flickr) MLE indegree distribution estimates for $p = 0.1, 0.5,$ and 0.9.

The experiment in Figure 10 is a good indication that indegree distribution estimates are inaccurate. In our technical report [14] we provide a lower bound of the mean squared error of any unbiased indegree distribution estimator. Our theoretic results agree with our experiment shown in Figure 10. Even when adding additional side information, for instance, the fact that the average indegree is equal to the average outdegree, little improvement is seen.

A more promising property of many directed graphs is the presence of symmetric edges. For example, in the Web-Google dataset 69% of the edges are symmetric (the symmetry of other datasets is in Table I). In a graph where all edges are symmetric (i.e., every edge $(u, v) \in E_d$ has a corresponding edge $(v, u) \in E_d$) the indegree and the outdegree distributions are the same. In what follows we consider a model that adds such symmetric edge information to the estimation and show that while moderate edge symmetry increases estimation accuracy, it is still insufficient to obtain accurate estimates.

A. Side Information: Edge Symmetry

Consider a directed graph $G_d = (V, E_d)$. Let s denote the fraction of symmetric edges in E_d, where $s = 1$ when all edges in G_d are symmetric. Edge symmetry can convey information about the indegree distribution. For instance, if $s = 1$ the indegree distribution is equivalent to the outdegree distribution. To assess the increase in estimation accuracy that comes from the presence of symmetric edges, consider the following model.

Let v be a sampled vertex. Consider the following random variables of v:

- Z: indegree of v.
- Z_s: number of symmetric incoming edges.
- Z_a: number of incoming asymmetric edges.
- Y: observed outdegree.
- X_s observed number of symmetric incoming edges.
- X_a observed number of asymmetric incoming edges.

Also, let $\rho(y, z) = P[Y = y, Z = z]$ be the joint indegree and outdegree distribution of v, p be the sampling rate, and α be the fraction of symmetric edges. We assume that the number of outgoing edges of v that are symmetric is a Binomial random variable with parameter α and has distribution

$$P[Z_s = z_s | Y = y, Z = z] = \begin{cases} \binom{\min(y,z)}{z_s} \alpha^{z_s} (1-\alpha)^{\min(y,z)-z_s} & \text{if } z_s \le \min(y, z), \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

We seek to find a likelihood function of the observed random variables Y, X_s, and X_a with respect to ρ, $P[Y = y, X_s = x_s, X_a = x_a | \rho]$. Note that

$$\begin{aligned} &P[Y = y, X_s = x_s, X_a = x_a | \rho] \\ &= \sum_{\forall z} P[X_s = x_s, X_a = x_a | Y = y, Z = z] \rho_{y,z} \\ &= \sum_{\forall z} \rho_{y,z} \sum_{z_s=0}^{z} P[X_s = x_s, X_a = x_a | Z_s = z_s, Y = y, Z = z] \\ &\qquad \times P[Z_s = z_s | Y = y, Z = z], \end{aligned}$$

where

$$\begin{aligned} &P[X_s = x_s, X_a = x_a | Z_s = z_s, Y = y, Z = z] \\ &= P[X_s = x_s, X_a = x_a | Z_s = z_s, Y = y, Z_a = z - z_s] \\ &= \binom{z_s}{x_s} p^{x_s} (1-p)^{z_s - x_s} \binom{z - z_s}{x_a} p^{x_a} (1-p)^{z - z_s - x_a} \\ &= \binom{z_s}{x_s} \binom{z - z_s}{x_a} p^{x_s + x_a} (1-p)^{z - x_s - x_a} \end{aligned}$$

with $P[Z_s = z_s | Y = y, Z = z]$ as defined in equation (5).

The indegree distribution Fisher information associated with the symmetric edge information can be computed from the Fisher information of $P[Y = y, X_s = x_s, X_a = x_a | \rho]$ with respect to ρ by noting that θ, the indegree distribution, can be

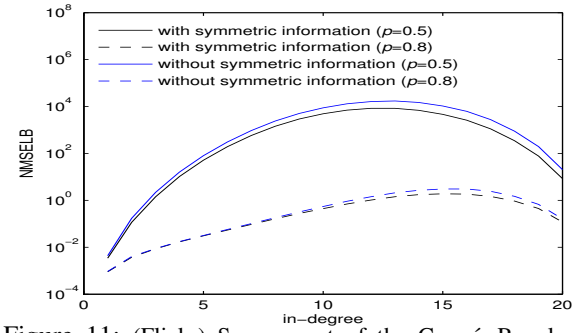


Figure 11: (Flickr) Square root of the Cramér-Rao lower bound normalized by the true value of θ. The curves show the error lower bound with and without symmetric edge information. Edge symmetry at $\alpha = 62\%$. Lower is better.

defined as $\theta_z = \sum_{\forall y} \rho(y, z)\,, \forall z$, or in matrix form $\theta = H\rho^{\mathsf{T}}$, where $\rho = (\rho(1,1), \rho(2,1), \dots)$ and

$$H = \begin{bmatrix} 1 & \dots & 1 & & & & & \\ & & & \dots & & & & \\ & & & & & 1 & \dots & 1 \end{bmatrix}.$$

Let J_ρ denote the Fisher information with respect to the joint distribution ρ. Computing J_ρ from $P[Y = y, X_s = x_s, X_a = x_a|\rho]$ is trivial. Let J_θ denote the Fisher information with respect to the indegree distribution θ. Then [16, pages 83–84]

$$J_\theta = H J_\rho H^{\mathsf{T}}.$$

Matrix J_θ encodes the information obtained from the observed incoming edges plus the information that the graph is symmetric. To obtain the Cramér-Rao bound we need to invert J_θ. We do this inversion numerically in Section VI-B and observe that adding symmetric information does not significantly improve the estimation error unless most edges in the graph are symmetric.

B. Numerical Results

In the following experiment we include symmetry information in the Cramér-Rao lower bound computed by inverting J_θ (which is a bound on the mean squared error of any unbiased estimator of θ). Figure 11 shows the square root of the Cramér-Rao lower bound divided by the true value of θ (NMSELB) of Flickr for maximum indegree $W = 20$, with and without Flickr's symmetric information. In Flickr the fraction of edges that are symmetric is $\alpha = 0.62$. Observe that while symmetry reduces the Cramér-Rao lower bound, it is not enough to significantly increases the estimation accuracy to acceptable levels. Moreover, other experiments (not shown here) indicate that increasing W significantly increases the estimation error (to the point that even estimating θ_1 can be made inaccurate).

VII. Related Work

Estimating observable characteristics by sampling a directed graph (in this case, the Web graph) has been the subject of Bar-Yossef et al. [3] and Henzinger et al. [8], which transform the directed graph of web-links into an undirected graph by adding reverse links, and then use a Metropolis-Hastings RW to sample webpages uniformly. Our "backward edge traversal" is an adaptation of the method of Bar-Yossef et al. [3] to work with a pure random walk and random jumps. Both of these Metropolis-Hastings RWs are designed to sample directed graph that do not allow random jumps. However, in the presence of random jumps (even if jumps are rare), the Metropolis-Hastings RW algorithm is not as efficient and as accurate as our DURW algorithm. Random walks with PageRank-style jumps are used in Leskovec and Faloutsos [9] to sample large graphs. In Leskovec and Faloutsos [9], however, there is no technique to remove the large biases induced by the random walk and the random jumps, which makes this method unfit to estimate graph characteristics. In contrast, our distribution estimates are asymptotically unbiased.

VIII. Conclusions & Future Work

In this work we provide the first random walk method to accurately estimate characteristics of directed graphs that allow random jumps. Also, to the best of our knowledge our work is the first to study and provide a sound theoretical analysis of the problem of estimating latent indegree distributions. Our DURW has a single parameter, w, which controls how often the random walk needs to perform an uniform random jump. From our experience we observe that a value of w that is no less than half and no more than twice the average outdegree yields good results. Our future work includes reducing the transient and exploring the parameter space of the DURW algorithm.

References

[1] Google Programming Contest. http://www.google.com/programming-contest/, 2002.
[2] *Predicting Positive and Negative Links in Online Social Networks*, 2010.
[3] Ziv Bar-Yossef and Maxim Gurevich. Random sampling from a search engine's index. *J. ACM*, 55(5):1–74, 2008.
[4] S. Brin and L. Page. The anatomy of a large-scale hypertextual Web search engine. In *Proc. of the WWW*, 1998.
[5] Facebook. http://www.facebook.com, 2010.
[6] Flickr. http://www.flickr.com, July 2010.
[7] Douglas D. Heckathorn. Respondent-driven sampling: A new approach to the study of hidden populations. *Social Problems*, 1997.
[8] Monika R. Henzinger, Allan Heydon, Michael Mitzenmacher, and Marc Najork. On near-uniform url sampling. In *Proceedings of the WWW*, pages 295–308, 2000.
[9] Jure Leskovec and Christos Faloutsos. Sampling from large graphs. In *Proc. of the KDD*, pages 631–636, 2006.
[10] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and Analysis of Online Social Networks. In *Proc. of the IMC*, October 2007.
[11] Amir H. Rasti, Mojtaba Torkjazi, Reza Rejaie, Nick Duffield, Walter Willinger, and Daniel Stutzbach. Respondent-driven sampling for characterizing unstructured overlays. In *Proc. of the IEEE Infocom*, pages 2701–2705, April 2009.
[12] Bruno Ribeiro, William Gauvin, Benyuan Liu, and Don Towsley. On MySpace account spans and double Pareto-like distribution of friends. In *Proceedings of the IEEE Infocom NetSciCom Workshop*, 2010.
[13] Bruno Ribeiro and Don Towsley. Estimating and sampling graphs with multidimensional random walks. In *Proc. of the IMC*, 2010.
[14] Bruno Ribeiro, Pinghui Wang, Fabricio Murai, and Don Towsley. Sampling directed graphs with random walks. Technical Report UM-CS-2011-031, UMass Amherst, 2011.
[15] D. Stutzbach, R. Rejaie, N. Duffield, S. Sen, and W. Willinger. On unbiased sampling for unstructured peer-to-peer networks. *IEEE/ACM Trans. Netw.*, 17(2):377–390, 2009.
[16] Hary L. van Trees. *Estimation and Modulation Theory, Part 1*. Wiley, New York, 2001.
[17] Wikipedia website. http://www.wikipedia.org, 2010.