Data Mining and Knowledge Discovery
https://doi.org/10.1007/s10618-020-00683-y

Guided sampling for large graphs

Muhammad Irfan Yousuf[1] · **Suhyun Kim**[1]

Received: 17 September 2019 / Accepted: 6 March 2020

Abstract
Large real-world graphs claim lots of resources in terms of memory and computational power to study them and this makes their full analysis extremely challenging. In order to understand the structure and properties of these graphs, we intend to extract a small representative subgraph from a big graph while preserving its topology and characteristics. In this work, we aim at producing good samples with sample size as low as 0.1% while maintaining the structure and some of the key properties of a network. We exploit the fact that average values of degree and clustering coefficient of a graph can be estimated accurately and efficiently. We use the estimated values to guide the sampling process and extract tiny samples that preserve the properties of the graph and closely approximate their distributions in the original graph. The distinguishing feature of our work is that we apply traversal based sampling that utilizes only the local information of nodes as opposed to the global information of the network and this makes our approach a practical choice for crawling online networks. We evaluate the effectiveness of our sampling technique using real-world datasets and show that it surpasses the existing methods.

Keywords Big graphs · Graph sampling · Social networks

1 Introduction

Large networks, formally represented as graphs, offer a tremendous amount of data to study but practical limitations of storage and computational power make it infeasible to fully analyze such massive graphs. In recent years, extracting a small representative

Responsible editor: Hanghang Tong

✉ Suhyun Kim
suhyun_kim@kist.re.kr

Muhammad Irfan Yousuf
irfan@kist.re.kr

[1] Imaging Media Research Center, Korea Institute of Science and Technology, Seoul, Republic of Korea

Published online: 18 March 2020

subgraph from a large original graph, known as graph sampling, has emerged as a solution to examine some important properties of the original graph with limited resources. Graph sampling helps us in saving time and resources needed to study massive graphs. In order to maintain the structure and properties of the original graph, the existing sampling algorithms could shrink a graph to 10% of its size but further reduction in sample size deteriorates the structure of the sample graph.

Given the fact that a real-world graph could have billions of nodes, the 10% sample with millions of nodes in it fades away the purpose of sampling. In this work, we are motivated to reduce the sample size below 1% while retaining its key properties, its degree, clustering coefficient, path length, diameter and a structural property, its assortativity. By extracting very small samples, we will be able to study massive graphs without needing any special resources. We exploit the fact that some properties of a graph, e.g. the average degree and average clustering coefficient, can be estimated efficiently without requiring lots of resources. The estimated values can guide us in extracting a very small but accurate sample that faithfully represents the actual graph.

In this paper, we propose a new sampling method that relies on the estimated values of degree and clustering coefficient of a graph. We let estimated values guide the sampling process, so that the extracted tiny samples maintain the structure and properties of the original graph. The main characteristics of our work are as follows:

1. We reduce the sample size below 1% while maintaining the key properties, that is degree, clustering coefficient, path length and diameter, of the original graph. We also retain the overall structure of the graph by preserving its degree mixing pattern (assortativity) and modularity.
2. We introduce a two step approach to graph sampling. In the first phase, we extract a sample graph that has the required number of nodes but extra edges in it. In the second phase, we remove some extra edges to closely match the original graph.
3. We present the idea of approximating the impact of removing an edge on the clustering coefficient of a graph and apply this idea to the selection of extra edges to remove.
4. Lastly, we visit a localized portion of a graph, as opposed to exposing the whole graph for uniform sampling, and still we sustain the overall structure of the graph being sampled. In other words, we utilize the local information of nodes and do not need any global information. For example, TIES (Ahmed et al. 2011) needs to know the whole graph in advance for uniform sampling. We, on the other side, traverse a small portion of a graph by exploring the neighborhood of sampled nodes and this makes our approach a practical choice for crawling online networks.

The rest of the paper is organized as follows. In Sect. 2, we provide some background knowledge and our intuition of guided sampling approach. In Sect. 3, we formally present our sampling technique. In Sect. 4, we present the evaluation criteria and datasets used in the experiments. In Sect. 5, we evaluate our approach and compare it with existing approaches. We present the related work in Sect. 6 and conclude the paper in Sect. 7.

2 Background and intuition

In this section, we describe the sampling problem, present research related to estimating the average degree and clustering coefficient of a graph, and give our basic intuition to exploit this information in sampling.

2.1 Sampling problem

Given a large graph G = (V,E), where V = $\{v_1, v_2, v_3, \ldots, v_n\}$ is the set of vertices (or nodes) and E = $\{e_1, e_2, e_3, \ldots, e_m\}$ is the set of edges (or links), we extract a sample graph $G_s = (V_s, E_s)$ from G such that $V_s \subset V$ and $E_s \subset E$. The resulting sample graph G_s has n_s number of vertices $V_s = \{v_{1_s}, v_{2_s}, v_{3_s}, \ldots, v_{n_s}\}$ and m_s number of edges $E_s = \{e_{1_s}, e_{2_s}, e_{3_s}, \ldots, e_{m_s}\}$ in it. We consider undirected graphs in this work and represent an edge between nodes v_i and v_j as a tuple $e(v_i, v_j)$. Given a sampling fraction ϕ such that $|V_s|/|V| = \phi$, the aim of sampling is to obtain a sample with a small value of ϕ.

2.2 Estimating degree and clustering coefficient

There has been a significant amount of research to estimate the different properties of a graph (Ahn et al. 2007; Ribeeiro and Towsley 2010; Gjoka et al. 2010). The goal of such a research work is to quickly explore and estimate some of the characteristics of a graph. The research community has proposed some good techniques to estimate the average degree and average clustering coefficient of a graph by mining a small portion of it.

The work in (Sethu and Chu 2012; Gjoka et al. 2010) shows that metropolis–hastings random walk (MHRW) is very efficient in capturing the degree distribution of a graph even if we visit a small portion of the graph. The work in (Hardiman and Katzir 2013) deploys Random Walk and mines a small percentage of nodes to estimate the average clustering coefficient of a graph with good accuracy. Once we have the estimated values of average degree and average clustering coefficient of the original graph, our idea is to feed them to the sampling algorithm and guide the sampling process to achieve a particular goal instead of blindly sampling the nodes and edges.

2.3 Our intuition

A good sample graph G_s of G closely approximates different properties and distributions of those properties of G. The well-studied properties of a sample graph are its degree and clustering coefficient along with their distributions while some other properties have also been explored by the research community (Hu and Lau 2014). The previous work on sampling could obtain good samples with a value of $\phi = 0.1$ and could preserve one or two properties of the original graph but if $\phi < 0.1$, we get a compromised sample. In this work, we aim at producing good samples with such small sampling fractions while preserving some key properties of the original graph.

We note that when ϕ is low, the size n_s of G_s is too small to capture the structure of G because the sampling is performed blindly without any prior knowledge of the underlying original graph. Intuitively, we believe that if we feed in some information to the sampling technique and guide our sampling towards a known goal, we can yield good samples with low values of ϕ. In order to lower the value of ϕ and still extracting a true sample graph G_s from G, we intuit to use the estimated values of average degree and average clustering coefficient and then guide the sampling process based on the estimated values.

3 Guided sampling

Our guided sampling (GS) algorithm has two phases. In the first phase, we deploy modified depth first sampling (ModDFS). In ModDFS, we modify the Depth First Sampling such that we always sample the highest degree neighbor of a node from its unsampled neighbors to favor high degree nodes in the sample graph. By applying ModDFS in the first phase, we collect a sample graph $G_s = (V_s, E_s)$ such that V_s has the required number of nodes in it but the edge set E_s has more than enough number of edges because of the biased nature of ModDFS, In the second phase, we remove the extra edges from the sample graph to achieve the required values of degree and clustering coefficient. Instead of removing the edges cluelessly in this phase, we calculate the weight of an edge to estimate its impact on the average clustering coefficient of a graph when it is removed. This weight guides the trimming of the graph to realize a good sample graph.

3.1 Edge weight

The neighborhood of a node is made up of all the nodes connected to it by an edge, not including the node itself and local clustering coefficient of a node measures the connectedness of its neighborhood. We measure it as the ratio of the existing edges between the neighbors of a node to the number of all possible edges between the neighbors. Consider a node v that has degree d_v and l_v number of edges exist between its neighbors, its clustering coefficient cc_v would be given by

$$cc_v = \frac{2 * l_v}{d_v * (d_v - 1)} \tag{1}$$

Now suppose that we remove an edge between its neighbors such that its degree remains d_v, this means the clustering coefficient of node v will decrease by an amount $\frac{2}{d_v*(d_v-1)}$. This provides the foundations of our idea of giving weight to all the edges in the sample graph and removing edges according to their weights to match the clustering coefficient of the sample graph to that of the original graph.

A node gives weight to an edge between a pair of its neighbors with a simple intuition of keeping its own degree constant. In a closed triplet of nodes v, u, and w, node v gives weight to edge $e(u, w)$ because removing this edge will decrease its clustering coefficient without affecting its degree. The weight it gives is calculated as

$$W_{e(u,w)} = \frac{2}{d_v * (d_v - 1)} \tag{2}$$

where d_v is the degree of node v. Similarly, node u gives weight to edge $e(v, w)$ and node w gives weight to edge $e(u, v)$ in the triplet. Intuitively, when we remove an edge between the neighbors of a node such that the degree of the node does not change, we can precisely measure the decrement in the local clustering coefficient of that node. However, it is very challenging to calculate the impact on the average clustering coefficient of the graph because removing a single edge could disturb many triplets in the graph.

A node repeats this process for all the closed triplets it is part of. Finally, the weight of an edge is the sum of weights given by all the nodes that make close triplets with nodes at the ends of this edge. Therefore, the weight of an edge $e(u, w)$, such that nodes u and w make closed triplets with nodes $\{v_1, v_2, v_3, \ldots, v_k\}$, is calculated as

$$W_{e(u,w)} = \sum_{i=1}^{k} \frac{2}{d_i * (d_i - 1)} \tag{3}$$

Intuitively, we assume that edges with high weight have high impact on the average clustering coefficient of a graph because (i) removing a high weight edge means that the local clustering coefficient of (probably) many nodes will decrease and therefore, the average clustering coefficient of the graph will decrease significantly compared to removing a low weight edge and (ii) low degree nodes contribute more to the weight of an edge and when we remove a high weight edge, the local clustering coefficient of low degree nodes decrease significantly contributing to sharp decrement in the average clustering coefficient of the graph.

Consider the small graph in Fig. 1 with $|V| = 9$, $|E| = 16$ and average clustering coefficient CC $= 0.66$. In the graph, node 1 has degree $d_1 = 3$ and the weight given by this node to edges e(2,3) and e(3,4) is 0.33 (using eq. 2). Similarly, node 6 has degree $d_6 = 5$ and it gives weight equal to 0.1 to edges e(2,3), e(2,5), e(3,7), and e(7,9) as removing these edges will not change its degree. Now the total weight of edge e(2,3) is $W_{e(2,3)} = 0.33 + 0.1 = 0.43$, the sum of the weights given by node 1 and node 6. This way, we calculate the weight of all the edges in the graph as shown in the figure. Now, if we remove the edge e(2,5) with $W_{e(2,5)} = 0.1$, the average CC of the graph drops to 0.63, a small decrement, and if we remove the edge e(4,7) with $W_{e(4,7)} = 1.1$, the average CC of the graph drops to 0.51, a significant drop. This supports our intuition that removing a high weight edge has a higher impact on the CC of a graph and the CC of a graph decreases sharply when we remove a high weight edge as compared to removing a low weight edge.

Supposedly, it is very strenuous to calculate the precise impact of removing an edge on the clustering coefficient of a graph, however, we apply heuristic and assume that removing a high weight edge reduces the clustering coefficient of a graph sharply as compared to removing a low weight edge. In the experimantal evaluation section, we perform an experiment to show that our heuristic is legitimate (see Fig. 3).

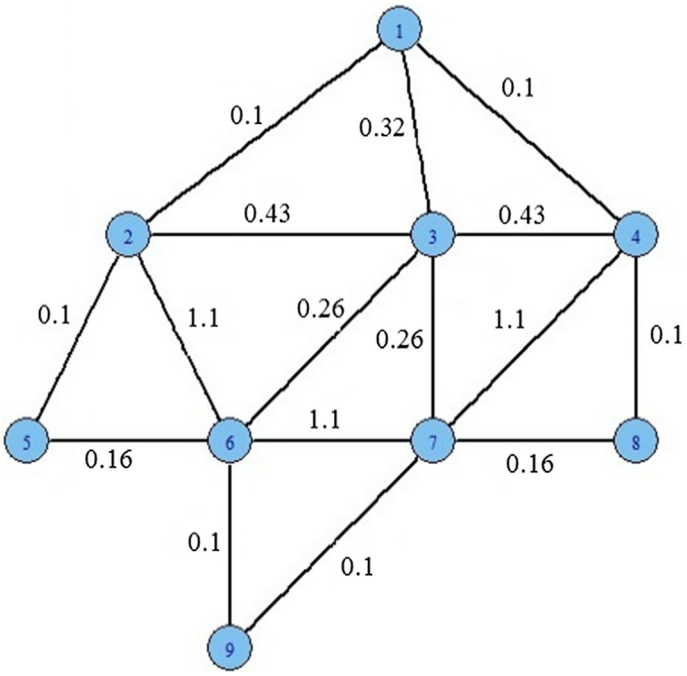

Fig. 1 A small network with $|V| = 9$ and $|E| = 16$. The labels on edges show their weight

3.2 Guided sampling algorithm

In the first phase of guided sampling (GS) algorithm, we perform ModDFS as explained above and populate the node set V_s. We then induce the sample graph G_s by adding all the existing edges between the nodes in V_s. We have an overestimated sample graph $G_s = (V_s, E_s)$ where the node set V_s contains the required number of nodes and we do not need to remove any node but the edge set E_s has extra edges in it and needs to be trimmed. Next, we calculate the weight of each edge in E_s as explained above, sort E_s in descending order of weights, count the number of extra edges in E_s and calculate the average clustering coefficient of G_s. At this stage, the sample graph G_s is an overestimated sample of G in terms of average degree and average clustering coefficient.

In the second phase, we remove extra edges from E_s, one by one, calculate the clustering coefficient of G_s after removing an edge and decide whether we should remove a high weight edge or a low weight edge in the next iteration. The algorithm runs till we have removed all the extra edges from E_s.

In order to decide whether to remove a low weight edge or a high weight edge, we deploy a simple idea of following the slope (line 11) of the hypothetical line drawn from the initial value (line 6) of clustering coefficient of sample graph to the target value inputted to the algorithm. We intend to guide the sampling process by following this hypothetical line heuristically. Moreover, assuming that the edges in E_s are available as a list, we sort the list in descending order of weights (line 4), mark the middle of the

list (line 14) and consider the edges on the left side (or upper portion) of this mid point as high weight edges and those on the right side (or lower portion) are taken as low weight edges. We delete (line 20) either a high weight edge (line 16) or a low weight edge (line 18) by comparing (line 15) the current value of clustering coefficient with the actual inputted value and the expected value (line 25) of clustering coefficient at that point on the hypothetical line. With this simple idea, we remove a high weight edge when we need a sharp decrement in clustering coefficient and a low weight if we need a minor adjustment to follow the hypothetical line and finally reach the target value when we have removed all the extra edges. It would be worth to mention that we delete an edge (line 20) only if the degree of both the nodes at its ends is greater than one; this makes sure that there is no isolated node and the sample graph is fully connected.

3.3 Time complexity

The run time complexity of the guided sampling technique is straight forward and could be calculated as follows. We first calculate the time to estimate the values of the degree and clustering coefficient of the original graph. Both the techniques discussed in Sect. 2.2 deploys Random Walk or its variation and the upper bound on the estimated time for a Random Walk to visit n_s vertices is $\frac{4}{27}{n_s}^3 + O({n_s}^3)$ (Feige 1995) where n_s is the number of nodes in the sample graph. For simplicity, we take it as $O({n_s}^3)$ and hence the time complexity to estimate the average degree and clustering coefficient of the original graph is $2 * O({n_s}^3)$, of the order of $O({n_s}^3)$. In the first phase of GS, we deploy ModDFS (line 1) with complexity $O(n_s)$ (we ignore the small overhead to find the highest degree neighbor of a node), calculate the weight of edges in E_s (line 3) with complexity $O({n_s}^3)$, sort E_s (line 4) with complexity $O(m_s log(m_s))$ and calculating the clustering coefficient (line 6) has complexity $O({n_s}^3)$. We represent the overall complexity of the first phase with a constant $\mathcal{C}$. In the second phase, we recalculate the average clustering coefficient (line 22) of the sample graph after deleting an extra edge and we do it e_{extra} number of times. When we delete an extra edge, the end nodes of the edge and the common friends of these nodes are affected. Basically, these common friends are the nodes that gave the weight to the edge being deleted. Assuming we have a complete graph for worst case analysis, the complexity to calculate the clustering coefficient (line 22) is $O({n_s}^2)$ and since e_{extra} is of the order of m_s where m_s is the number of edges in the sample graph, therefore, the complexity of the second phase is $O({n_s}^2) * m_s$ or simply $O({n_s}^2 m_s)$. To summarize, the expected time to realize a sample with GS is of the order of $O({n_s}^3) + O({n_s}^2 m_s) + \mathcal{C}$ where n_s and m_s are the number of nodes and edges in the sample graph respectively and $\mathcal{C}$ is a constant.

Guided Sampling (GS)

Input:
Original Graph G = (V,E) where V is the set of nodes and E is the set of edges. Target values of degree (d_{org}) and clustering coefficient (cc_{org}) of original graph and the sampling fraction ϕ.

Output:
Sample Graph G_s = (V_s,E_s) where V_s is the set of sampled nodes ($|V_s| = |V| * \phi$) and E_s is the set of sampled edges.

Initialization:
$V_s = 0$, $E_s = 0$

Sampling, Induction and Edge Weight:

$V_s \leftarrow ModDFS(G, \phi)$
Populate E_s by inducing node set V_s
Calculate the weight of edges in E_s
Sort E_s in descending order of weights

$e_{extra} = |E_s| - \frac{d_{org} * |V_s|}{2}$

$cc_{init} \leftarrow CalculateCC(G_s)$

Removing Extra Edges:

$e_{del} = 0$
$e_{ratio} = 1$
$cc_{curr} = cc_{init}$

$cc_{ratio} = \frac{cc_{org}}{cc_{curr}}$

$slope = \frac{\frac{cc_{curr}}{cc_{org}} - 1}{e_{extra}}$

$cc_{exp} = cc_{init} - (slope * e_{del} * cc_{org})$

while $e_{del} < e_{extra}$ **do**

    $mid = \frac{|E_s|}{2}$

    if $cc_{curr} > cc_{exp}$ **AND** $cc_{curr} > cc_{org}$ **then**
        $index = mid * cc_{ratio} * e_{ratio}$
    else
        $index = mid + mid * e_{ratio}$
    end if

    $DeleteEdge(E_s, index)$

    $e_{del} = e_{del} + 1$

    $cc_{curr} \leftarrow CalculateCC(G_s)$

    $cc_{ratio} = \frac{cc_{org}}{cc_{curr}}$

    $e_{ratio} = \frac{e_{extra} - e_{del}}{e_{extra}}$

    $cc_{exp} = cc_{init} - (slope * e_{del} * cc_{org})$

end while

4 Evaluation criteria and datasets

4.1 Evaluation criteria

There is no agreed-upon criteria to measure the goodness of a sample, however, the research community (Leskovec and Faloutsos 2006; Ahmed et al. 2011) has evaluated the sample quality by comparing some properties of sample graphs with the original graph. We compare the degree, clustering coefficient, path length, 90% effective diameter, assortativity and modularity of sample and original graphs and perform quantitative tests such as Jensen–Shannon Distance (JSD) and Root Mean Square Error (RMSE) for quantitative evaluation of sampling algorithms. All the results presented in this paper are averaged over five readings.

Point Statistics A point statistic is a single value statistic that shows the value of a property at a single point. We vary the sampling fraction ϕ from 0.001 to 0.01 and plot the scaling ratio of a property Θ as the ratio of the value of that property in the sampled graph Θ_S to the value of that property in the actual graph Θ_A:

$$Scaling\ Ratio = \frac{\Theta_S}{\Theta_A} \tag{4}$$

For example, we measure the average degree of the sample graph and the original graph and find the scaling ratio of degree by diving the average degree of the sample graph G_s at a sampling fraction ϕ with the average degree of the original graph G.

Distributions A distribution is a multivalued statistic and shows the distribution of a property in a graph. For example, the degree distribution shows the fraction of nodes that have degree greater than or less than a particular value. We find and plot the empirical cumulative distribution function (ECDF) of degree, clustering coefficient and path lengths of sample graphs at $\phi = 0.001$.

Root Mean Square Error Given the original graph G and sampled graph G_s, we want to measure how far is G_s from G. For scalar quantities such as the average degree, we use the common measure for the quality of estimation by root mean square error (RMSE), given as

$$RMSE = \sqrt{\frac{1}{n}\sum_{1}^{n}(\Theta_S - \Theta_A)^2} \tag{5}$$

where Θ_S and Θ_A are sampled and original values respectively.

Jensen–Shannon Distance For distributions of the properties, we measure Jensen–Shannon Distance. In probability theory, the Jensen–Shannon Divergence measures the similarity between two probability distributions, calculated as

$$D_{JS}(P||Q) = \frac{1}{2}D_{KL}(P||M) + \frac{1}{2}D_{KL}(Q||M) \tag{6}$$

where D_{JS} and D_{KL} are Jensen–Shannon and Kullback-Leibler Divergences respectively while P and Q are two Probability Distribution Functions (PDFs) and M =

$\frac{1}{2}$(P+Q). Its square root is a true metric often referred to as Jensen–Shannon Distance (JSD).

4.2 Datasets for analysis

In our experiments we use a total of 16 networks; 14 real graphs and 2 synthetic graphs. The 14 real networks include one autonomous network Skitter (Konect 2015), two ground truth community networks YouTube (Leskovec and Krevl 2014) and LiveJournal (Leskovec and Krevl 2014), four collaboration networks CiteSeer (Rossi and Ahmed 2015), GoogleScholar (Chen et al. 2017b), Actors (Konect 2015) and DBLP (Konect 2015) and seven social networks FourSquare (Rossi and Ahmed 2015), Twitter (Rossi and Ahmed 2015), Lastfm (Rossi and Ahmed 2015), Hyves (Zafarani and Liu 2009), Flicker (Konect 2015), Flixster (Rossi and Ahmed 2015) and Facebook (Rossi and Ahmed 2015).

In addition, we also use two synthetic networks as these networks have strong mathematical foundations. The first systhetic network, called FF, is generated using Forest Fire method presented in (Leskovec et al. 2007). We use those parameter values for generating the graph that generate the most realistic graphs as mentioned in (Leskovec et al. 2007). We use small-world model (Watts and Strogatz 1998) to generate our second synthetic network and call it SW. The parameter values are tuned such that the generated network has nearly the same average degree and clustering coefficient as that of FF synthetic graph. The reason to select Forest Fire generative model is that it generates graphs that follow many properties of real-world graphs (Leskovec et al. 2007) whereas small-world model generates random graphs with small-world properties that have been observed in real-world networks. We use sampling fractions $\phi = \{0.001, 0.0025, 0.005, 0.0075, 0.01\}$ to extract samples from these graphs. Table 1 summarizes the characteristics of these datasets.

5 Experimental evaluation

5.1 Accuracy in estimating degree and clustering coefficient

In this section, we estimate the degree and clustering coefficient (CC) of all datasets using the techniques discussed above. We deploy MHRW (Sethu and Chu 2012; Gjoka et al. 2010) to estimate the average degree and Random Walk based technique presented in (Hardiman and Katzir 2013) to estimate the average clustering coefficient of a graph. We estimate the values at different sampling fractions and plot the scaling ratio, i.e., estimated value over the true value, with 95% confidence intervals in Fig. 2. The graphs show that we can estimate the average degree and CC of a graph with good accuracy at different sampling fractions.

Table 1 Datasets used in the experiments

Dataset	Nodes	Edges	Average degree	Average clust. coeff.	Average path length	Assortativity	90% Effective diameter
CiteSeer	227,320	814,134	7.16	0.761	7.82	0.069	10.05
GoogleSchl.	277,074	1,234,019	8.90	0.361	5.21	0.312	7.47
Actors	382,219	15,038,083	78.68	0.784	3.56	0.227	4.31
Foursquare	639,014	3,214,986	10.06	0.108	3.65	− 0.288	3.85
Twitter	1,112,702	2,278,852	4.09	0.019	5.64	− 0.018	6.26
Lastfm	1,191,805	4,519,330	7.58	0.072	4.58	− 0.135	4.95
DBLP	1,314,050	5,362,414	8.16	0.734	6.09	0.103	7.14
Hyves	1,402,673	2,777,419	3.96	0.044	5.67	− 0.023	6.52
Skitter	1,696,415	11,095,298	13.08	0.258	5.04	− 0.081	5.94
Flicker	1,715,255	15,550,782	18.13	0.183	5.19	− 0.015	6.62
Flixster	2,523,386	7,918,801	6.27	0.083	4.86	− 0.321	5.51
Facebook	2,937,612	20,959,854	14.26	0.091	5.31	− 0.112	5.92
Youtube	3,223,589	9,375,374	5.81	0.085	4.99	− 0.063	6.28
Livejournal	3,997,962	34,681,189	17.34	0.283	6.00	0.045	6.47
FF	1,000,000	3,166,455	6.33	0.154	6.31	− 0.003	7.24
SW	1,000,000	3,000,000	6.00	0.164	9.12	0.000	9.85

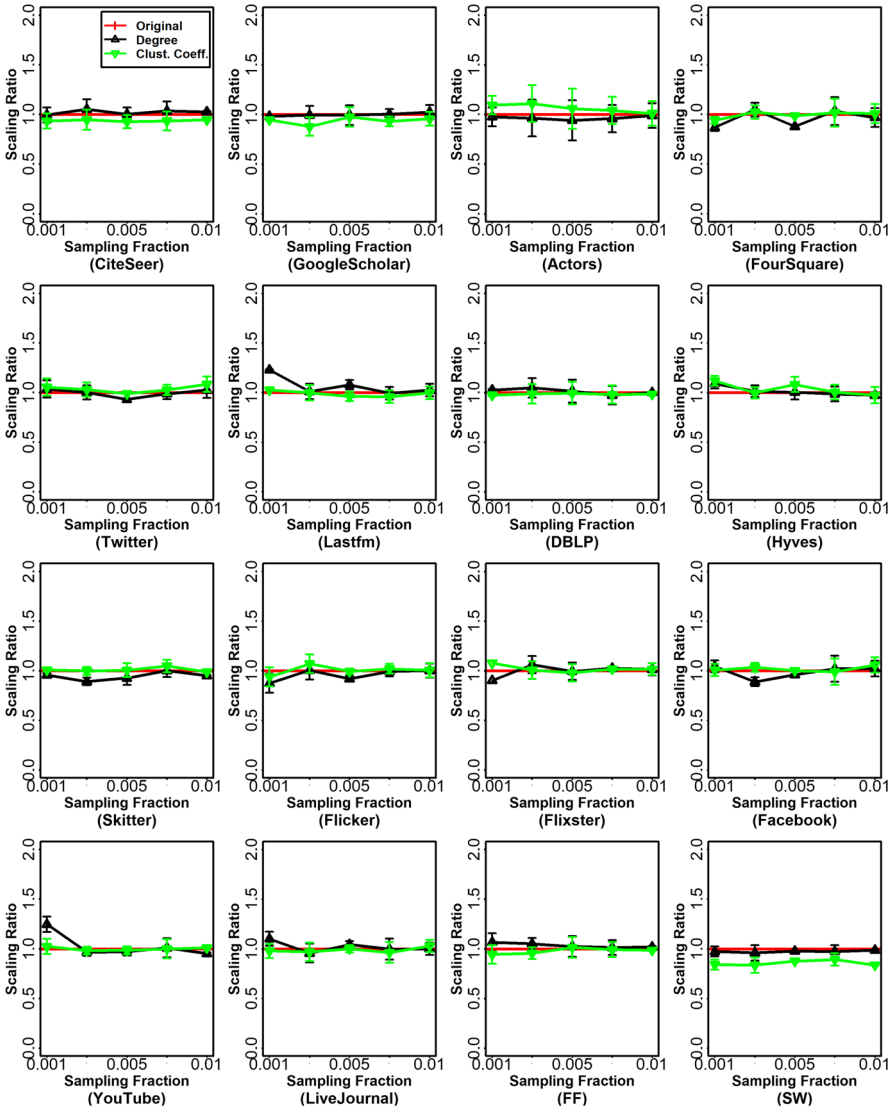

Fig. 2 Accuracy in estimating the degree and clustering coefficient of a graph

5.2 Impact of edge weight on clustering coefficient

In this section, we demonstrate the impact of removing a high weight edge versus a low weight edge on the average clustering coefficient of a graph. We perform an experiment in which we retrieve a sample from the original graph, induce it and calculate the edge weight (line 1 through 3 of GS) and then remove the extra edges using three different schemes. In the first scheme, we remove high weight edges (HWE), that means if the edges are sorted in descending order of their weights, we remove the top e_{extra}

number of edges where e_{extra} is calculated as per line 5 of GS. In the second scheme, we remove low weight edges (LWE), the same as HWE but the edges are sorted in ascending order of their weights. In the third scheme, we apply GS and remove a high weight or a low weight edge as discussed in the GS algorithm. We then plot the ratio of the current clustering coefficient to the original clustering coefficient against the percentage of extra edges removed. The results are shown in Fig. 3 for all datasets at $\phi = 0.001$.

First, we see that we collect overestimated samples in all datasets because we perform ModDFS in the first phase and these overestimated samples make sure that we have extra edges to remove in the second phase of our guided sampling approach. Second, in the case of HWE, the average clustering coefficient of the sample graph decreases sharply and this justifies our heuristic that high weight edges have higher impact on the clustering coefficient of a graph. In the case of LWE, even after removing all extra edges, the clustering coefficient is still very high in all datasets. We observe that removing a low weight edge induces a small decrement (or occasionally a negligibly small increment) in the value of the clustering coefficient of G_s. In the case of GS, we remove both high and low weight edges and hence reach the desired value after removing all the extra edges.

5.3 Impact of preserving degree and clustering coefficient

In this section, we perform two experiments to find the impact of preserving average degree and average clustering coefficient of a graph. Let d_{est} and cc_{est} be the estimated values of average degree and clustering coefficient of the graph being sampled.

In the first experiment, we perform undersampling and oversampling in terms of clustering coefficient. We input d_{est} as the target value of degree and $cc_{target} = \{0.5, 0.75, 1.0, 1.25, 1.5\} \times cc_{est}$ as the target values of clustering coefficients. In other words, we get five samples that have the same average degree, i.e., we preserve degree, but different clustering coefficients. We then find and plot the scaling ratios of average path length, 90% effective diameter and modularity and RMSE values of assortativity of four datasets (results are similar for the remaining datasets.) in Fig. 4. We see that clustering coefficient has a marginal effect on these properties. For example, we can see an improvement in Twitter and Lastfm datasets in path length but cannot observe such changes in other metrics. It seems that whether we oversample or undersample a graph in terms of clustering coefficient, the properties such as path length of the sampled graph does not change significantly. Regarding assortativity, we have mixed results. For two datasets, we get a smaller error when we oversample and a larger error when we undersample the graph.

In the second experiment, we perform undersampling and oversampling in terms of degree. We input cc_{est} as the target value of clustering coefficient and $d_{target} = \{0.5, 0.75, 1.0, 1.25, 1.5\} \times d_{est}$ as the target values of degree. In other words, we get five samples that have the same average clustering coefficient, i.e., we preserve clustering coefficient, but different average degrees. The results are shown in Fig. 5. In general, when a graph is undersampled in terms of degree it overestimates the average path length and 90% effective diameter, and when it is oversampled it underestimates

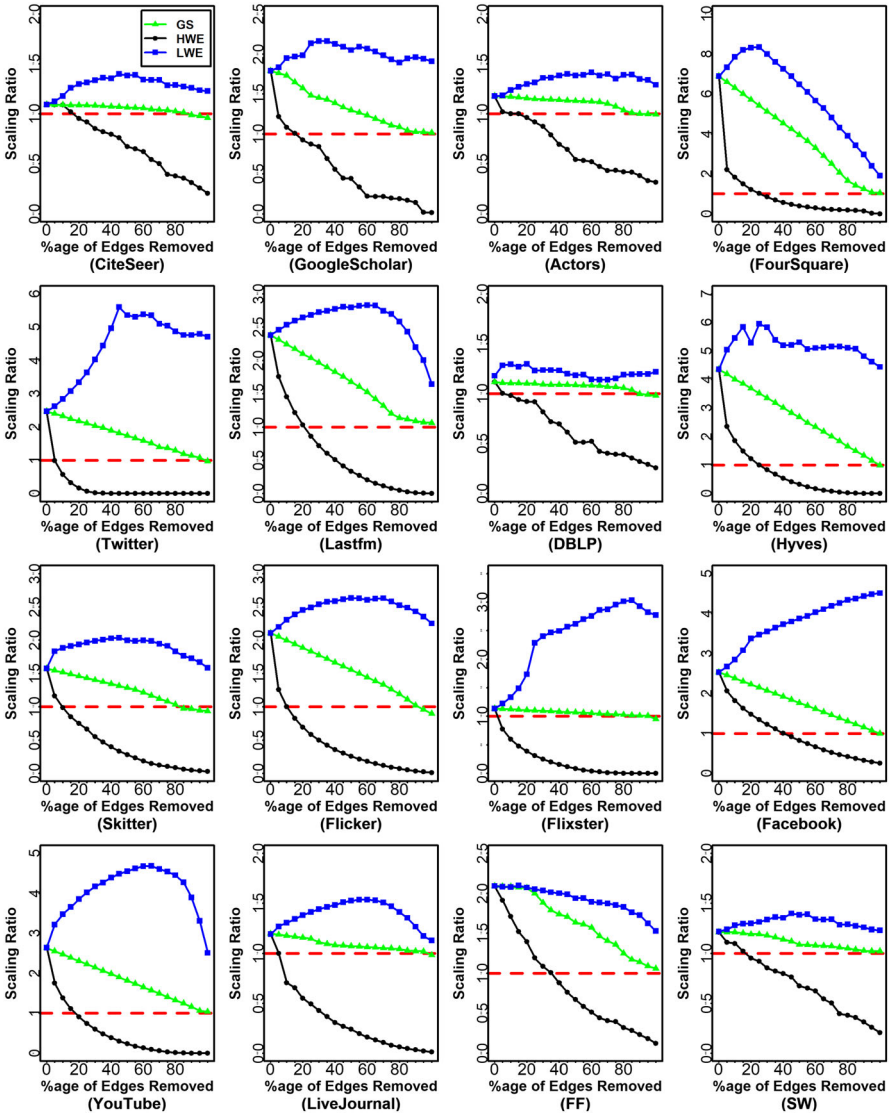

Fig. 3 Impact of edge weight on clustering coefficient

these values. In the case of modularity, an undersampled graph clearly overestimates it whereas an oversampled graph does not affect this metric significantly. Regarding assortativity, we cannot conclude clearly but it seems this property is less sensitive to undersampling or oversampling.

Even though degree and clustering coefficient are two of the key properties of a graph, we do not intend to claim that preserving degree and clustering coefficient will always result in better performance in other metrics. However, the experimental results in this section show that there is some correlation between the two properties

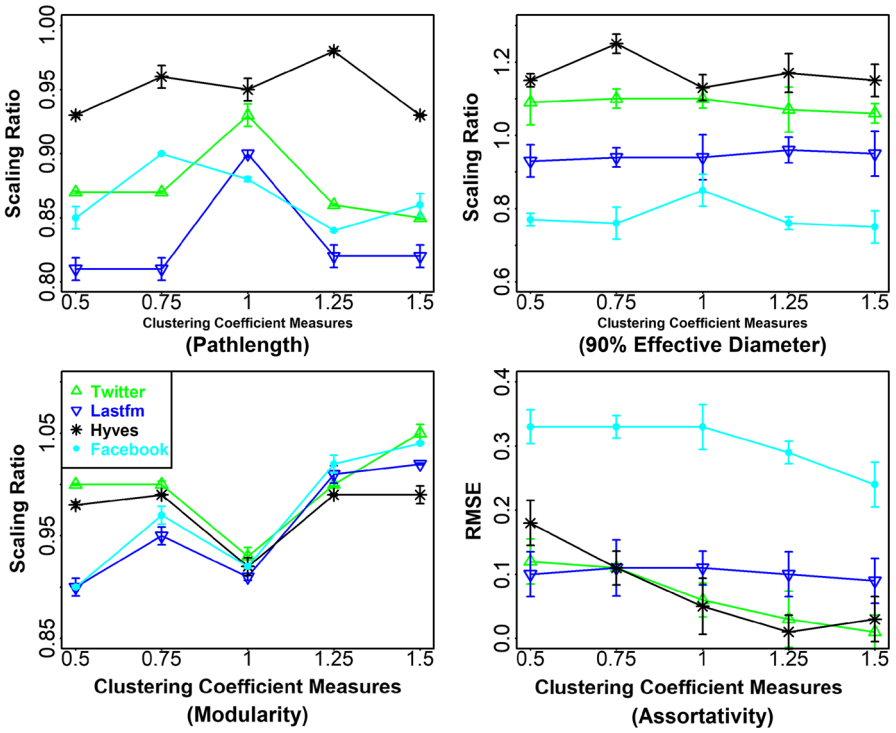

Fig. 4 Impact of preserving degree but undersampling or oversampling in terms of clustering coefficient

and other metrics, and preserving the average degree seem to have a bigger impact on other metrics than preserving the average clusteirng coefficient. In the coming sections, we will present the results of all datasets across many different metrics, and how GS compares to previous techniques in those metrics .

5.4 Before and after guidance

In order to understand the working of the GS, we perform an experiment and draw the distributions of the sampled graphs before and after the guidance. We find the distributions of degree, clustering coefficient and path lengths at $\phi = 0.001$ right after the first phase (before the guidance) when the sampled graph still has extra edges in it. We then remove extra edges as per GS algorithm and draw the distributions when the extra edges have been removed (after the guidance). The results of four datasets are shown in Fig. 6.

The first row shows the degree distribution plots and we see that we collect many high degree nodes in the first phase as expected. During guidance, we remove edges and hence the degree of many nodes decreases but still we have high degree nodes in the sampled graph. Although the average degrees of the sampled graphs are the same as that of their original counterparts, we have more high degree nodes than low degree nodes and hence the degree distributions do not follow closely. Similarly, the

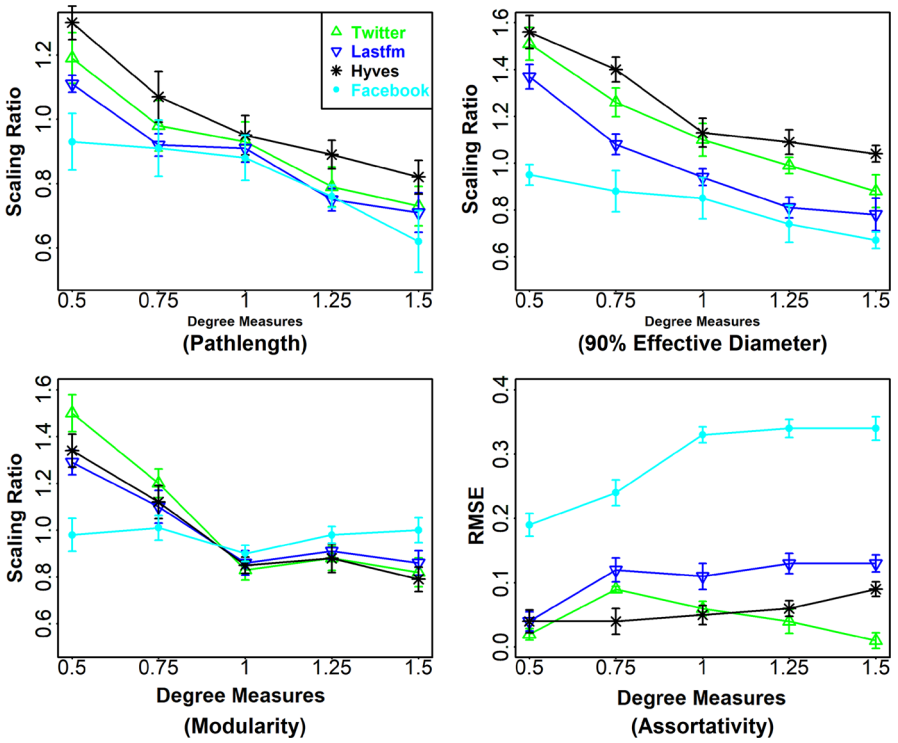

Fig. 5 Impact of preserving clustering coefficient but undersampling or oversampling in terms of degree

clustering coefficient distributions have higher values before the guidance and after removing the extra edges, we follow the clustering coefficient distributions better than the degree distributions. In the case of path length, we initially have shorter path lengths but when we remove extra edges, we drop many paths between the nodes and this results in good estimation of path lengths. In short, the second phase trims the graph and helps to make it a representative of the original graph.

A question may arise that why we did not apply simple breadth first sampling (BFS) or some of its variation to get samples in the first phase of GS as the well-studied (Doerr and Blenn 2013; Maiya and Berger-Wolf 2011) biased nature of BFS will make sure that we have extra edges to remove. The reason to prefer DFS over BFS is that, in BFS we explore graph in breadth dimension and this compromises the path length and effective diameter of the sampled graph because we do not explore the graph in depth. When we deploy DFS, we dig deep into the graph and as a result we get better results in terms of path length and effective diameter. The modified part of DFS makes sure that we have extra edges at the end of the first phase. To answer the question, we conduct a small experiment in which we extract samples in the first phase with BFS, DFS and ModDFS for a sampling fraction of $\phi = 0.005$. We then compute the ratio of induced edges to the expected edges to see if we always get extra edges in the first phase so that we could remove them in the guided phase. The results are presented in

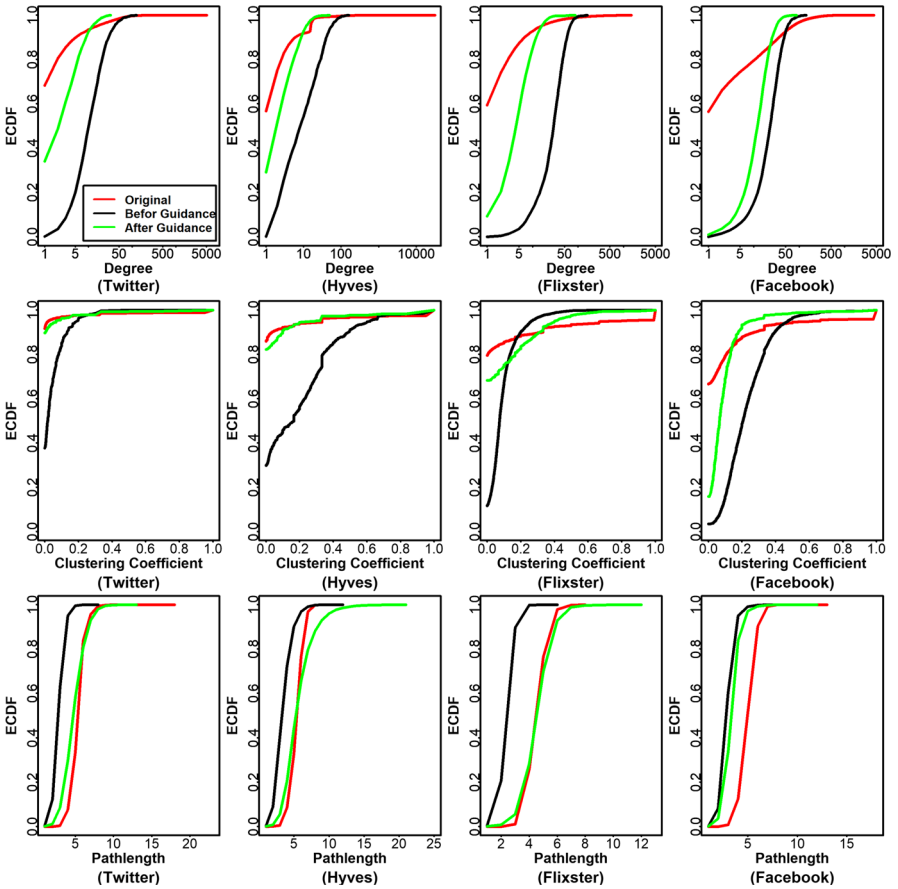

Fig. 6 Degree, clustering coefficient and path length distributions of four networks before and after the guidance. Results are similar for remaining networks. **(Best viewed in color)**

Table 2. It is clear that the biased nature of BFS does not always guarantee to sample extra edges whereas the ModDFS works as expected.

5.5 Overhead of recalculating the clustering coefficient

We remove extra edges one by one and recalculate the clustering coefficient of the graph at every step. This recalculation of the clustering coefficient may slow down the sampling process. However, we do not need to recalculate it for the whole graph but for a few nodes. When we delete an extra edge, the end nodes and the common friends of these nodes are affected and we recalculate the clustering coefficient of these nodes only. We perform an experiment in which we count the number of nodes for which the clustering coefficient is recalculated when sampling the graph at $\phi = 0.005$. We present the results in Table 3. We see that, on average, only a few nodes are affected and recalculating their clustering coefficient does not over burden the sampling process.

Table 2 The ratio of induced edges to the expected edges in the sample graphs extracted by ModDFS, BFS and DFS

Dataset	ModDFS	BFS	DFS
CiteSeer	2.41	1.14	0.76
GoogleScholar	2.53	1.27	0.87
Actors	4.66	1.38	1.15
Foursquare	8.62	3.42	1.34
Twitter	4.68	0.89	1.04
Lastfm	7.23	0.95	1.54
DBLP	4.15	0.93	0.87
Hyves	4.67	0.87	0.95
Skitter	5.79	1.66	0.97
Flicker	36.59	4.21	7.76
Flixster	9.04	2.32	0.88
Facebook	3.03	0.82	0.93
Youtube	15.35	4.49	0.83
Livejournal	7.92	0.93	0.86
FF	1.28	0.91	0.82
SW	1.36	0.87	0.73

Table 3 The average number of nodes for which the clustering coefficient is recalculated

Dataset	Average number of nodes
Citeseer	21.52
GoogleScholar	25.17
Actors	24.38
Foursquare	15.65
Twitter	2.32
Lastfm	6.84
DBLP	12.01
Hyves	5.93
Skitter	18.55
Flicker	12.35
Flixster	5.41
Facebook	6.48
Youtube	9.44
Livejournal	5.38
FF	4.26
SW	3.14

5.6 Evaluation against previous sampling methods

In this section, we evaluate GS against five previous sampling methods. We compare GS with the following methods.

- *Forest fire sampling (FFS)* In FFS (Leskovec and Faloutsos 2006), we start from a randomly picked seed node and burn a fraction of its outgoing edges along with the nodes on the other end of these edges. The fraction of nodes to be burned depends on the forward burning ratio p_f with a recommended value of $p_f = 0.7$ (Leskovec and Faloutsos 2006).
- *Totally induced edge sampling (TIES)* Nasreen et al. proposed totally induced edge sampling (TIES) (Ahmed et al. 2011) which is a variation of random edge sampling. The basic difference in TIES and random edge sampling is the graph induction step. In TIES, we augment all the edges among the sampled nodes by including other edges between the sampled nodes in addition to those sampled in the sampling step.
- *Expansion sampling (XS)* The XS strategy (Maiya and Berger-Wolf 2011) is based on the concept of expansion from work on expander graphs and seeks to greedily construct the sample with the maximal expansion. It was particularly designed to sample communities in networks (Maiya and Berger-Wolf 2010).
- *Frontier sampling (FS)* In FS (Ribeeiro and Towsley 2010), we deploy m-dimensional dependent random walks to sample a graph. FS firstly randomly chooses a set of nodes as seeds. Then FS samples a node from the set of seeds with the probability proportional to its degree.
- *Rank degree (RD)* RD (Voudigari et al. 2016) is a graph exploration sampling method based on edge selection. The core of this algorithm is the edge selection rule that is built on the ranking of nodes based on their degree values.

5.6.1 Degree statistics

We vary the sampling fraction from $\phi = 0.001$ to $\phi = 0.01$ and plot the scaling ratio of average degree as the ratio of the average degree in the sampled graph to that of in the original graph (see eq. 4). All values are shown with 95% confidence intervals. Figure 7 shows the scaling ratio of average degree on Y-axis against the sampling fraction on X-axis for all newtroks. We see that FFS and FS always underestimate the degree whereas TIES, XS and RD show a mix behavior. For some networks TIES and XS overestimate the average degree, e.g., FourSquare, Flicker and YouTube, while for other networks these methods underestimate the value, e.g., CiteSeer, Actors and Facebook. RD overestimates at higher sampling fractions in some datasets, e.g., Twitter and DBLP. We observe that TIES, XS and RD results vary with sampling fraction whereas FFS and FS, though undersample the graph, perform consistently at different sampling fractions. In the case of GS, we produce correct samples because we guide our sampling to a known target instead of blindly sampling the nodes and/or edges. For quantitative comparison, we give the values of RMSE averaged over all sampling fractions along with standard deviations in Table 4. We have highlighted the lowest value for each dataset. We see that GS induces less error than all other methods across

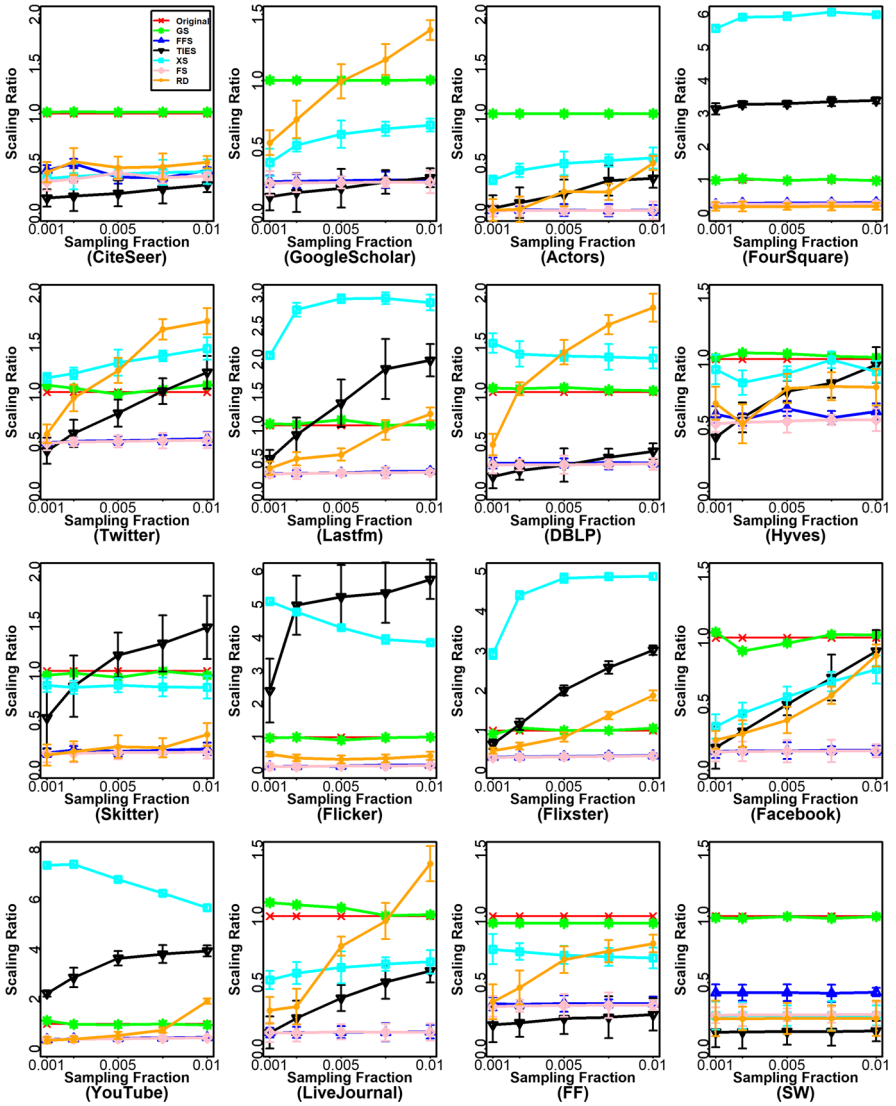

Fig. 7 Point statistics of average degree of all networks with 95% confidence intervals

all datasets; be they are real or synthetic. On average, RMSE value of GS is 16.48 times less than that of FFS which stands second to GS in this metric.

5.6.2 Clustering coefficient statistics

We vary the sampling fraction from $\phi = 0.001$ to $\phi = 0.01$ and plot the scaling ratio of average clustering coefficient with 95% confidence intervals in Fig. 8. We see that FFS and FS perform poorly and underestimate the values in all datasets. TIES and

Table 4 RMSE values and standard deviations of point statistics of average degree

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	**0.03** ± 0.09	6.12 ± 0.14	5.43 ± 0.07	5.01 ± 0.08	6.23 ± 0.07	3.05 ± 0.13
G.Scholar	**0.03** ± 0.09	6.72 ± 0.15	7.20 ± 0.06	4.04 ± 0.11	6.85 ± 0.09	2.75 ± 0.09
Actors	**0.01** ± 0.12	7.31 ± 0.12	6.29 ± 0.07	12.21 ± 0.11	76.64 ± 0.09	14.33 ± 0.09
Fr.Square	**0.67** ± 0.09	3.51 ± 0.07	11.59 ± 0.11	19.31 ± 0.09	7.36 ± 0.11	8.06 ± 0.11
Twitter	**0.08** ± 0.11	0.99 ± 0.07	0.70 ± 0.14	1.25 ± 0.08	2.03 ± 0.07	1.95 ± 0.09
Lastfm	**0.20** ± 0.07	2.70 ± 0.14	2.46 ± 0.06	13.09 ± 0.11	5.48 ± 0.11	3.23 ± 0.13
DBLP	**0.15** ± 0.07	3.81 ± 0.13	2.15 ± 0.14	3.20 ± 0.06	5.94 ± 0.11	4.65 ± 0.08
Hyves	**0.09** ± 0.06	0.83 ± 0.06	0.69 ± 0.13	0.43 ± 0.08	1.86 ± 0.11	1.24 ± 0.09
Skitter	**0.27** ± 0.09	5.23 ± 0.11	2.13 ± 0.07	2.04 ± 0.06	10.76 ± 0.07	10.07 ± 0.09
Flicker	**0.62** ± 0.06	7.88 ± 0.07	37.69 ± 0.12	12.22 ± 0.1	15.82 ± 0.12	10.99 ± 0.07
Flixster	**0.16** ± 0.12	2.03 ± 0.14	3.95 ± 0.07	21.75 ± 0.12	4.13 ± 0.08	3.27 ± 0.08
Facebook	**0.38** ± 0.07	6.06 ± 0.07	3.92 ± 0.07	6.81 ± 0.13	12.21 ± 0.11	8.34 ± 0.08
YouTube	**0.19** ± 0.11	1.67 ± 0.14	6.85 ± 0.15	33.46 ± 0.13	3.45 ± 0.08	3.61 ± 0.08
L.Journal	**0.90** ± 0.10	7.57 ± 0.08	5.69 ± 0.07	7.01 ± 0.11	15.21 ± 0.11	8.43 ± 0.07
FF	**0.33** ± 0.08	4.18 ± 0.06	4.94 ± 0.06	1.82 ± 0.12	4.27 ± 0.08	2.73 ± 0.07
SW	**0.27** ± 0.10	4.62 ± 0.09	6.96 ± 0.11	6.03 ± 0.06	5.95 ± 0.07	6.17 ± 0.11
Average	**0.27** ± 0.09	4.45 ± 0.11	6.79 ± 0.10	9.35 ± 0.09	11.51 ± 0.09	5.81 ± 0.09

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

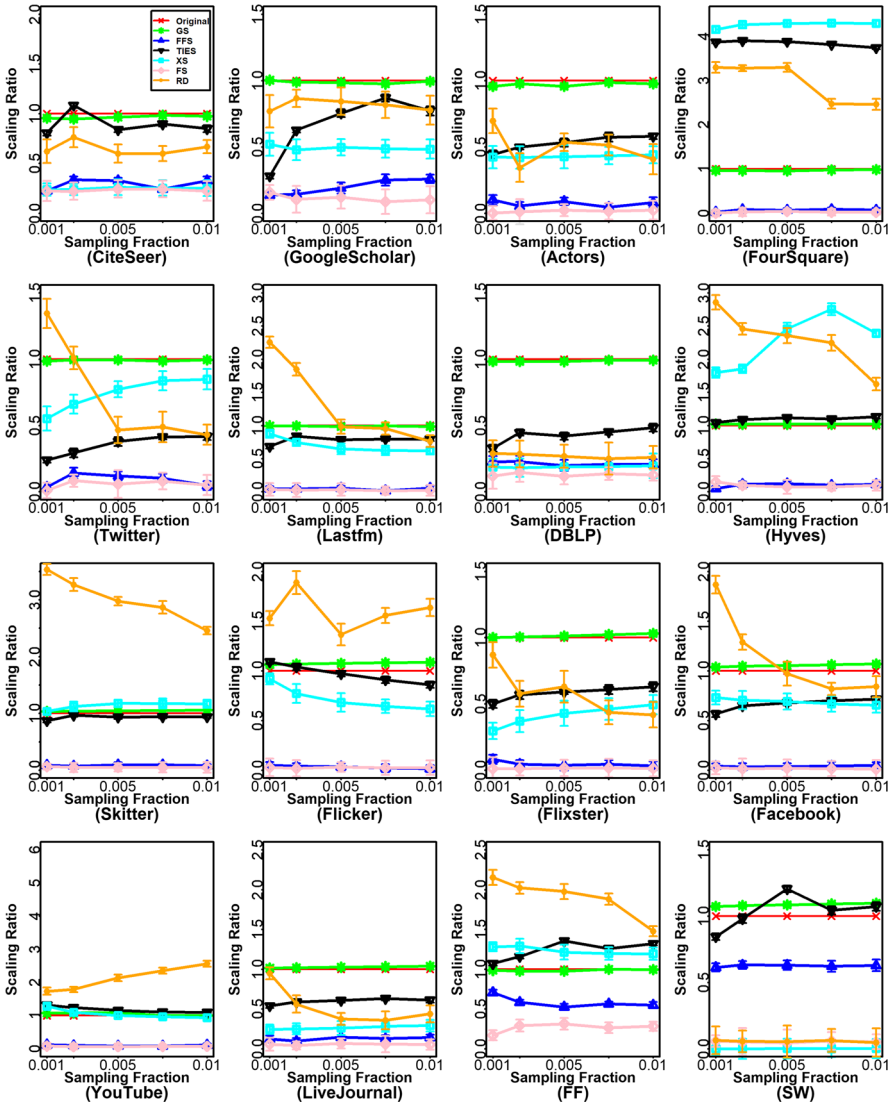

Fig. 8 Point statistics of average clustering coefficient of all networks with 95% confidence intervals

XS also gives inaccurate values in most of the networks but performs better in some datasets, e.g., Skitter, Youtube and FF. RD either oversamples, e.g., FourSquare and Hyves, or undersamples, e.g., DBLP and SW. GS leads to good results because we remove the extra edges depending on their weights that helps to reach the target value after we have removed all the extra edges. We give RMSE values in Table 5 with standard deviations. We see that GS wins against other methods in all the datasets. In some datasets, the RMSE value of GS is of the order of 10^{-4}. TIES is the second best method for this metric and the error of GS is 7.5 times less than that of TIES.

Table 5 RMSE values and standard deviations of point statistics of average clustering coefficient

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	**0.03** ± 0.08	0.31 ± 0.12	0.11 ± 0.08	0.33 ± 0.12	0.33 ± 0.07	0.20 ± 0.12
G.Scholar	**0.09** ± 0.08	0.29 ± 0.09	0.14 ± 0.07	0.18 ± 0.10	0.32 ± 0.07	0.07 ± 0.15
Actors	**0.05** ± 0.06	0.73 ± 0.11	0.37 ± 0.09	0.45 ± 0.07	0.77 ± 0.08	0.40 ± 0.11
Fr.Square	**0.01** ± 0.08	0.10 ± 0.11	0.57 ± 0.10	0.72 ± 0.15	0.22 ± 0.09	0.39 ± 0.15
Twitter	**0.00** ± 0.01	0.02 ± 0.07	0.01 ± 0.13	0.02 ± 0.13	0.06 ± 0.09	0.03 ± 0.11
Lastfm	**0.00** ± 0.00	0.07 ± 0.09	0.02 ± 0.12	0.04 ± 0.07	0.13 ± 0.11	0.09 ± 0.10
DBLP	**0.05** ± 0.11	0.58 ± 0.07	0.42 ± 0.07	0.60 ± 0.07	0.64 ± 0.10	0.54 ± 0.10
Hyves	**0.00** ± 0.00	0.04 ± 0.07	0.01 ± 0.12	0.13 ± 0.11	0.09 ± 0.12	0.13 ± 0.12
Skitter	**0.03** ± 0.10	0.25 ± 0.12	0.07 ± 0.11	0.04 ± 0.08	0.28 ± 0.06	0.43 ± 0.08
Flicker	**0.01** ± 0.07	0.18 ± 0.05	0.13 ± 0.12	0.11 ± 0.10	0.36 ± 0.13	0.23 ± 0.07
Flixster	**0.00** ± 0.01	0.08 ± 0.10	0.02 ± 0.11	0.12 ± 0.08	0.20 ± 0.08	0.10 ± 0.08
Facebook	**0.00** ± 0.00	0.09 ± 0.09	0.03 ± 0.12	0.07 ± 0.14	0.21 ± 0.08	0.09 ± 0.11
YouTube	**0.00** ± 0.06	0.08 ± 0.12	0.06 ± 0.08	0.02 ± 0.11	0.16 ± 0.11	0.43 ± 0.07
L.Journal	**0.06** ± 0.07	0.27 ± 0.10	0.17 ± 0.06	0.26 ± 0.13	0.33 ± 0.13	0.18 ± 0.07
FF	**0.01** ± 0.07	0.06 ± 0.07	0.04 ± 0.07	0.04 ± 0.11	0.11 ± 0.08	0.13 ± 0.12
SW	**0.03** ± 0.07	0.23 ± 0.12	0.29 ± 0.07	0.62 ± 0.12	0.59 ± 0.09	0.59 ± 0.13
Average	**0.02** ± 0.08	0.21 ± 0.09	0.15 ± 0.09	0.23 ± 0.10	0.30 ± 0.09	0.25 ± 0.10

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

Table 6 RMSE values and standard deviations of point statistics of average path length

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	1.15 ± 0.08	4.31 ± 0.14	3.15 ± 0.11	**0.74** ± 0.08	7.44 ± 0.11	1.02 ± 0.08
G.Scholar	0.50 ± 0.06	5.68 ± 0.06	4.23 ± 0.09	**0.28** ± 0.09	69.61 ± 0.07	0.93 ± 0.14
Actors	1.41 ± 0.06	5.44 ± 0.13	0.60 ± 0.08	0.70 ± 0.08	70.43 ± 0.13	**0.47** ± 0.12
Fr.Square	**0.31** ± 0.06	1.86 ± 0.16	1.24 ± 0.15	1.68 ± 0.07	2.04 ± 0.11	1.65 ± 0.12
Twitter	**0.41** ± 0.08	6.91 ± 0.16	0.43 ± 0.11	1.46 ± 0.07	40.31 ± 0.11	1.59 ± 0.13
Lastfm	**0.54** ± 0.09	6.73 ± 0.12	0.55 ± 0.08	1.46 ± 0.09	34.23 ± 0.14	0.60 ± 0.08
DBLP	1.68 ± 0.09	7.98 ± 0.11	**0.72** ± 0.16	1.88 ± 0.13	89.10 ± 0.12	1.68 ± 0.09
Hyves	0.78 ± 0.10	5.92 ± 0.16	**0.18** ± 0.15	0.41 ± 0.14	32.48 ± 0.14	2.21 ± 0.09
Skitter	**0.74** ± 0.10	4.96 ± 0.18	0.78 ± 0.08	1.16 ± 0.10	30.01 ± 0.11	2.02 ± 0.15
Flicker	**1.19** ± 0.09	4.25 ± 0.15	2.53 ± 0.10	2.49 ± 0.08	28.35 ± 0.11	2.41 ± 0.12
Flixster	**0.55** ± 0.09	7.02 ± 0.09	0.72 ± 0.07	1.59 ± 0.12	42.81 ± 0.11	0.66 ± 0.12
Facebook	0.97 ± 0.06	7.57 ± 0.12	1.23 ± 0.06	**0.38** ± 0.11	73.44 ± 0.09	0.54 ± 0.11
YouTube	**0.51** ± 0.06	4.16 ± 0.14	1.55 ± 0.09	2.13 ± 0.09	7.88 ± 0.09	2.44 ± 0.11
L.Journal	1.17 ± 0.06	8.51 ± 0.05	**0.47** ± 0.12	1.48 ± 0.09	83.64 ± 0.13	1.45 ± 0.11
FF	1.61 ± 0.08	9.09 ± 0.10	**1.33** ± 0.11	1.09 ± 0.09	74.9 ± 0.08	1.96 ± 0.08
SW	1.84 ± 0.06	1.67 ± 0.13	3.98 ± 0.15	**1.24** ± 0.09	82.64 ± 0.13	27.86 ± 0.08
Average	**0.96** ± 0.08	5.75 ± 0.12	1.48 ± 0.11	1.26 ± 0.10	48.08 ± 0.11	3.09 ± 0.11

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

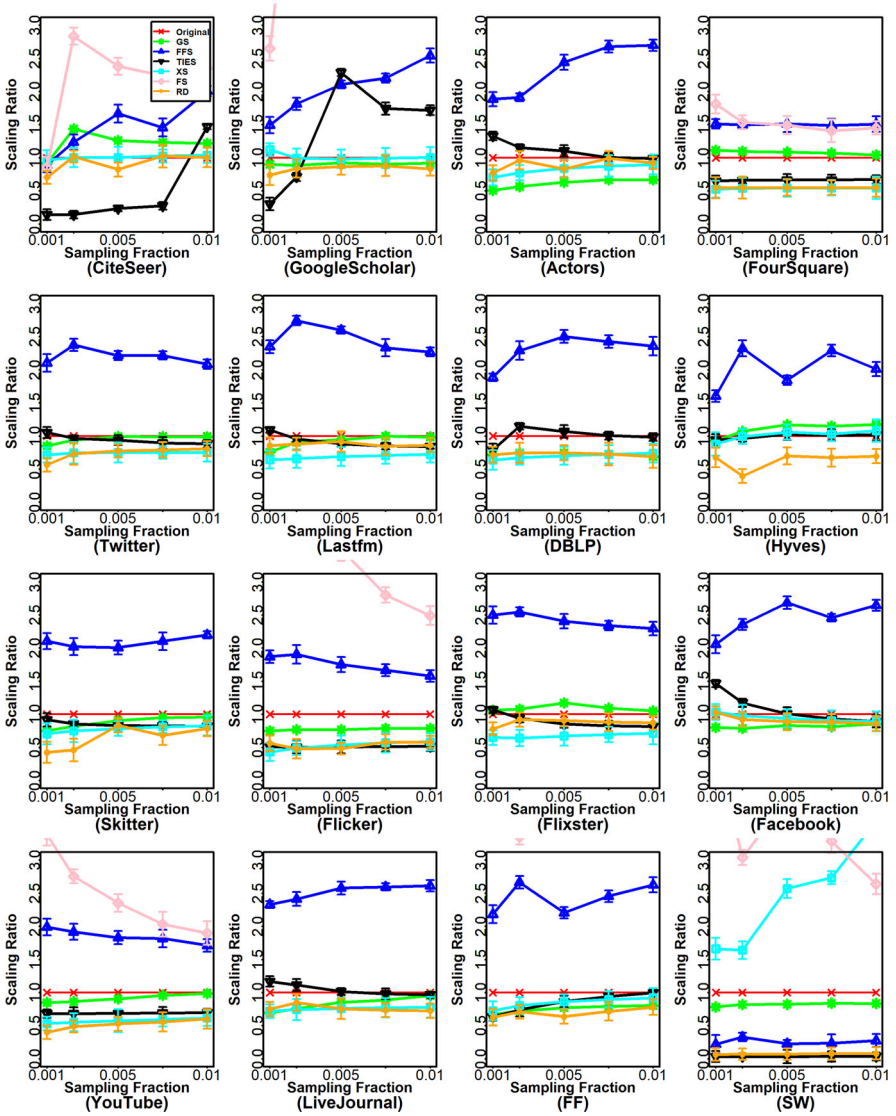

Fig. 9 Point statistics of average path length of all networks with 95% confidence intervals

5.6.3 Path length statistics

We show the results of scaling ratio of average path length at different sampling fractions in Fig. 9. FFS and FS remarkably estimate high values of path lengths in all datasets. FS values are many folds higher than the original values and fall outside the plotting area in many datasets. Other methods, i.e., TIES, XS and RD give good results in some datasets, e.g., Twitter, Lastfm and Facebook. GS outperforms in estimating the path lengths in most of the networks. Since TIES picks edges uniformly at random

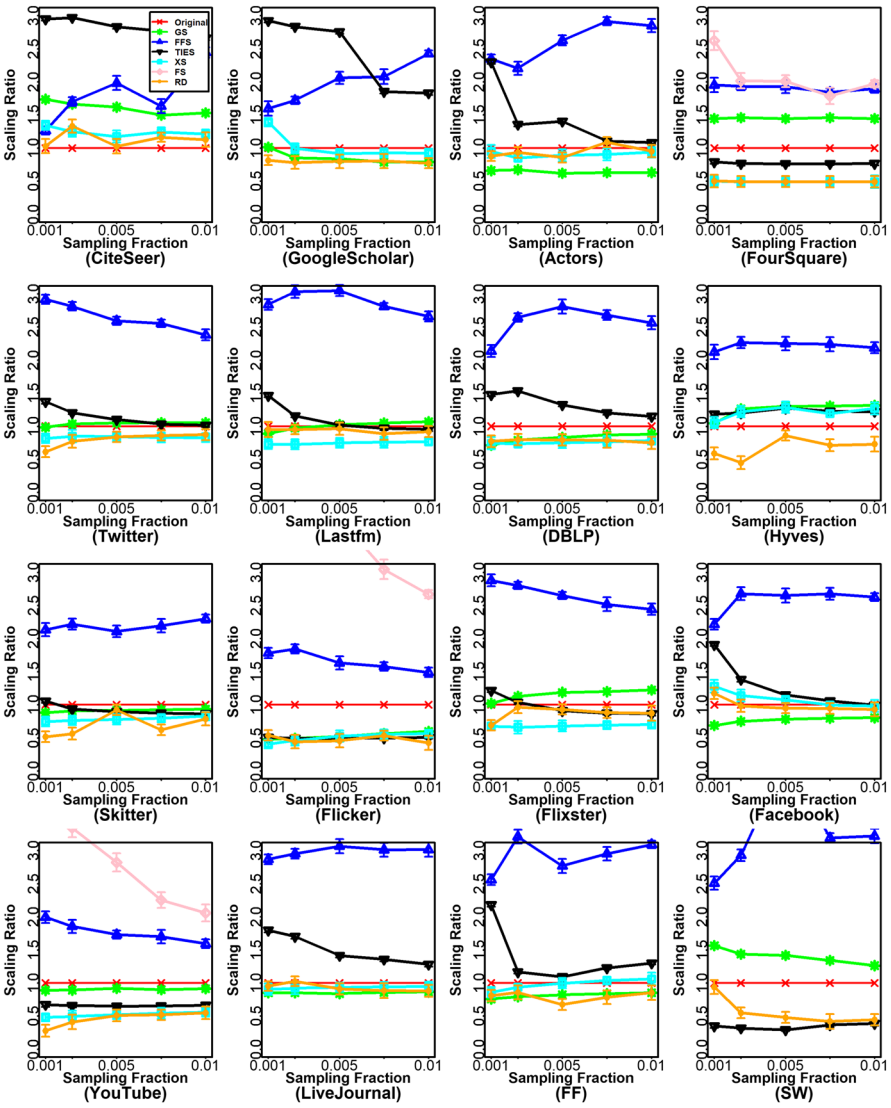

Fig. 10 Point statistics of 90% effective diameter of all networks with 95% confidence intervals

from the whole graph, so it has more chances of exploring the graph that helps it estimating the path lengths but still GS is better than TIES in most of the networks although GS mines a small portion of a graph. In the case of GS, we perform ModDFS and sample many nodes away from the seed node and it helps GS to measure the path lengths. XS also mines a small portion and it seems that the mechanism of selecting nodes in XS and augmenting all the edges between them is better than that of other methods and as a result XS yields good samples in terms of average path length. FFS and FS also sample a small fraction of neighboring nodes but do not augment all the

edges between the sampled nodes that leads them to deliver samples with high path lengths as they can drop many existing paths between the nodes. Table 6 gives the RMSE values along with standard deviations. We see that GS gives minimum error in seven datasets whereas TIES and XS introduce less error in four datasets each. On average, GS surpasses all the methods and generates less error than other methods whereas XS stands seconds in this metric. The GS error is 1.31 times less than that of XS.

5.6.4 90% effective diameter

The diameter of a graph is the length of the longest shortest path over all connected nodes. The 90% effective diameter or 90-percentile diameter is defined as the distance in which 90% of all node pairs are located and it is considered a more robust quantity than the diameter of a graph. We show the scaling ratios of this metric in Fig. 10. Similar to path length metric, FFS and FS overestimate the values of 90% effcetive diameter in all the datasets. TIES, XS and RD perform comparable to GS and give good results in some datasets, e.g., Twitter, Lastfm, Skitter and LiveJournal. For quantitative purposes, we show the RMSE values in Table 7. GS excels in nine datasets whereas TIES produces minimum error in four datasets. On average, GS performs the best by producing less error than other methods. XS and RD stands second and third respectively.

5.6.5 Assortativity

Assortativity quantifies the tendency of nodes in a graph to connect to others that are similar in some way. We use the degree of a node as a similarity measure and calculate the assortativity coefficient as the pearson correlation coefficient of degree between pairs of connected nodes. We use the definition given in (Hu and Lau 2014). Assortativity is also referred as degree mixing and ranges from $+1$ to -1, a network being assortative or disassortative respectively. We show the measured assortative values of G_s and the actual values of G in Fig. 11 for all networks. We see that RD does not match well with the original values in most of the networks. FFS, XS and FS perform good in some datasets whereas TIES work better than RD. GS gives mix results and prevails over other methods in some networks but not in all networks. TIES is the only method that perform uniform edge sampling by exposing the whole graph and hence it has more chances of picking nodes of varying degree from the graph but fails in maintaining their mixing patterns. Other methods explore a very limited region of a graph but the assortativity results show that they can still maintain the overall structure and mixing patterns of nodes well in some datasets. One possible explanation of GS results is that we perform ModDFS in the first phase of GS that helps us in preserving the degree mixing of nodes. In ModDFS, we sample the information that node v of degree d_v is connected to node u of degree d_u and hence GS gives good assortativity values. We show the RMSE values in Table 8. We see that GS, FFS, XS and FS beat one another in some of the datasets whereas RD always generates higher error than other methods. GS, XS and FS outperform in four datasets each. On average, GS gives the minimum error whereas XS stands second in assortativity

Table 7 RMSE values and standard deviations of point statistics of 90% effective diameter

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	**1.35** ± 0.06	9.01 ± 0.07	5.74 ± 0.08	2.51 ± 0.11	84.62 ± 0.11	1.76 ± 0.08
G.Scholar	**1.24** ± 0.11	7.62 ± 0.15	8.27 ± 0.13	1.38 ± 0.11	133.9 ± 0.14	1.55 ± 0.14
Actors	1.72 ± 0.08	7.77 ± 0.08	2.81 ± 0.11	**0.45** ± 0.07	139.4 ± 0.14	0.45 ± 0.09
Fr.Square	1.76 ± 0.11	3.14 ± 0.15	**1.01** ± 0.12	1.96 ± 0.11	4.32 ± 0.13	1.96 ± 0.14
Twitter	**0.52** ± 0.06	10.15 ± 0.12	0.84 ± 0.14	1.09 ± 0.07	84.64 ± 0.11	1.49 ± 0.12
Lastfm	**0.34** ± 0.07	9.34 ± 0.14	0.70 ± 0.11	1.29 ± 0.06	63.26 ± 0.15	0.36 ± 0.10
DBLP	**1.04** ± 0.06	8.76 ± 0.17	2.73 ± 0.14	1.81 ± 0.08	209.3 ± 0.12	1.62 ± 0.13
Hyves	1.91 ± 0.09	9.01 ± 0.16	**0.87** ± 0.14	1.46 ± 0.11	79.55 ± 0.16	2.40 ± 0.14
Skitter	**0.24** ± 0.07	6.74 ± 0.12	0.67 ± 0.15	1.37 ± 0.15	79.11 ± 0.1	2.23 ± 0.11
Flicker	**2.93** ± 0.09	5.17 ± 0.15	3.52 ± 0.13	3.49 ± 0.13	58.57 ± 0.08	3.64 ± 0.15
Flixster	1.06 ± 0.07	9.29 ± 0.14	**0.80** ± 0.15	1.79 ± 0.14	76.14 ± 0.18	0.92 ± 0.07
Facebook	1.36 ± 0.11	10.04 ± 0.05	2.19 ± 0.12	0.86 ± 0.09	225.3 ± 0.07	**0.54** ± 0.13
YouTube	**0.13** ± 0.08	5.23 ± 0.15	2.13 ± 0.09	3.28 ± 0.12	14.78 ± 0.07	3.79 ± 0.11
L.Journal	1.16 ± 0.10	11.73 ± 0.12	**0.09** ± 0.13	0.44 ± 0.07	260.2 ± 0.2	0.55 ± 0.12
FF	1.37 ± 0.12	12.53 ± 0.1	4.06 ± 0.08	**0.55** ± 0.11	165.4 ± 0.08	1.55 ± 0.12
SW	**1.43** ± 0.07	21.74 ± 0.11	6.47 ± 0.09	2.29 ± 0.09	385.8 ± 0.2	3.07 ± 0.09
Average	**1.22** ± 0.08	9.2 ± 0.12	2.68 ± 0.12	1.63 ± 0.10	128.9 ± 0.13	1.74 ± 0.11

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

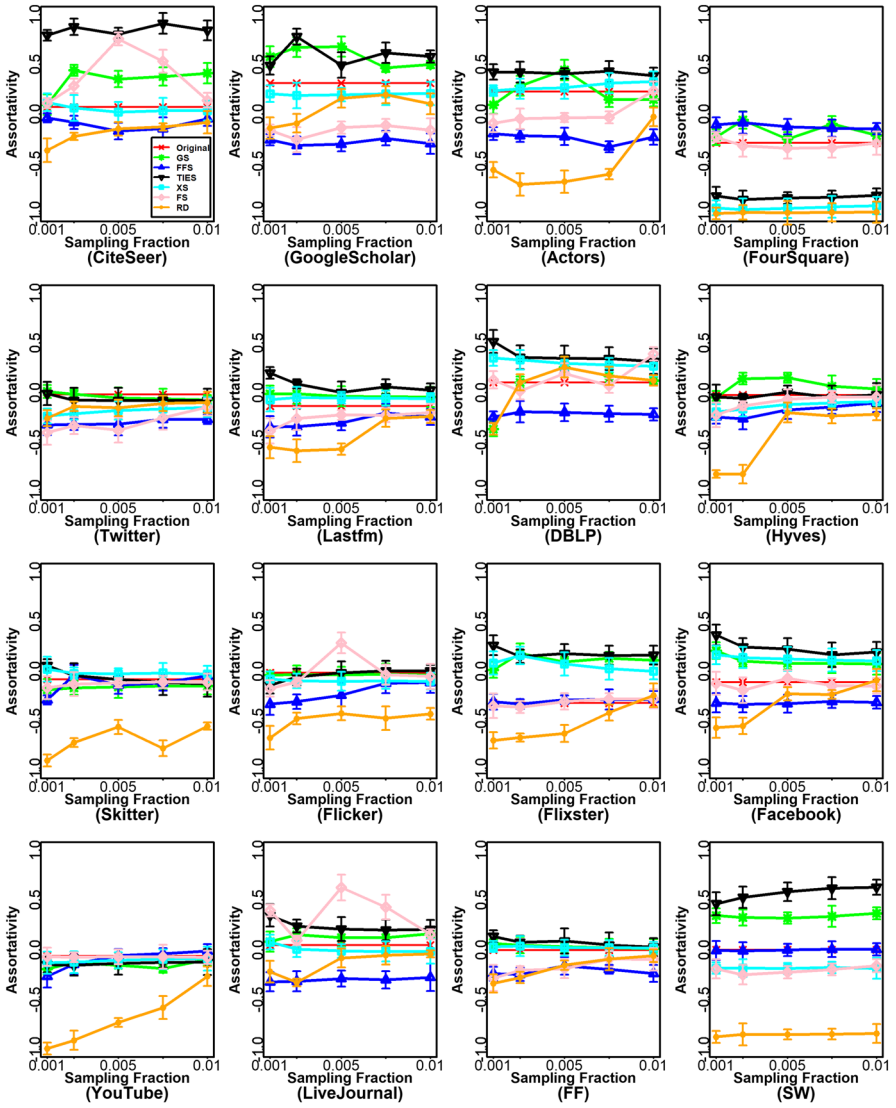

Fig. 11 Assortativity values of all networks with 95% confidence intervals

metric. Although FFS also maintains assortativity mixing for some networks, it does not perform well in retaining other properties. GS not only preserves node properties, e.g. degree, clustering coefficient and path length, but it also maintains the overall structure of the graph and sustains topological properties like effective diameter and degree assortativity.

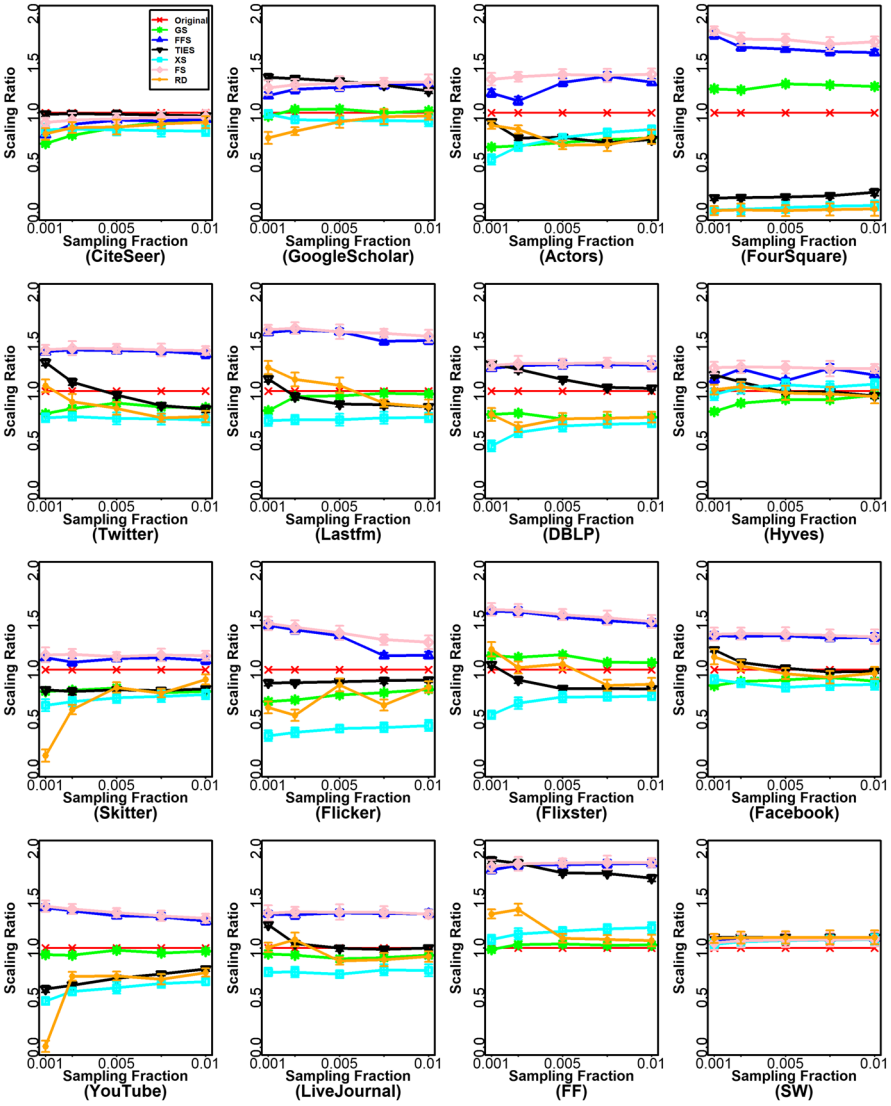

Fig. 12 Point statistics of modularity of all networks with 95% confidence intervals

5.6.6 Modularity

Modularity is one of the measures of the structure of networks or graphs. It measures the strength of division of a network into modules or communities. We calculate the modularity of the sampled and original graphs using Louvain method (Blondel et al. 2008). The purpose of this experiment is to see how well a sampling method can sample community structure in a graph. We calculate the scaling ratio of modularity and show the results in Fig. 12. We see that GS and TIES perform good in many

Table 8 RMSE values and standard deviations of point statistics of Assortativity

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	0.29 ± 0.06	0.39 ± 0.13	0.54 ± 0.08	$\mathbf{0.13} \pm 0.09$	0.43 ± 0.11	0.32 ± 0.07
G.Scholar	0.18 ± 0.07	0.59 ± 0.12	0.30 ± 0.11	$\mathbf{0.11} \pm 0.06$	0.48 ± 0.11	0.31 ± 0.11
Actors	0.13 ± 0.09	0.47 ± 0.14	0.19 ± 0.08	$\mathbf{0.06} \pm 0.07$	0.25 ± 0.08	0.78 ± 0.09
Fr.Square	0.07 ± 0.05	0.09 ± 0.12	0.28 ± 0.11	0.65 ± 0.07	$\mathbf{0.04} \pm 0.06$	0.71 ± 0.08
Twitter	$\mathbf{0.02} \pm 0.10$	0.14 ± 0.15	0.03 ± 0.09	0.17 ± 0.09	0.31 ± 0.08	0.14 ± 0.08
Lastfm	$\mathbf{0.05} \pm 0.08$	0.08 ± 0.08	0.10 ± 0.12	0.08 ± 0.07	0.14 ± 0.07	0.34 ± 0.11
DBLP	0.22 ± 0.05	0.32 ± 0.06	0.28 ± 0.12	0.21 ± 0.09	$\mathbf{0.14} \pm 0.09$	0.22 ± 0.09
Hyves	0.06 ± 0.10	0.09 ± 0.12	$\mathbf{0.01} \pm 0.07$	0.12 ± 0.12	0.11 ± 0.09	0.53 ± 0.11
Skitter	0.04 ± 0.07	$\mathbf{0.03} \pm 0.13$	0.04 ± 0.07	0.07 ± 0.07	0.05 ± 0.11	0.63 ± 0.12
Flicker	$\mathbf{0.01} \pm 0.06$	0.12 ± 0.13	0.02 ± 0.14	0.08 ± 0.09	0.16 ± 0.10	0.49 ± 0.11
Flixster	0.21 ± 0.11	$\mathbf{0.02} \pm 0.14$	0.26 ± 0.06	0.39 ± 0.08	0.04 ± 0.07	0.27 ± 0.10
Facebook	0.11 ± 0.05	0.10 ± 0.14	0.17 ± 0.15	0.24 ± 0.10	$\mathbf{0.05} \pm 0.07$	0.29 ± 0.08
YouTube	0.04 ± 0.05	0.05 ± 0.10	0.04 ± 0.14	0.05 ± 0.07	$\mathbf{0.01} \pm 0.11$	0.68 ± 0.11
L.Journal	$\mathbf{0.04} \pm 0.07$	0.16 ± 0.11	0.10 ± 0.12	0.05 ± 0.07	0.35 ± 0.06	0.22 ± 0.08
FF	0.04 ± 0.14	0.22 ± 0.08	0.09 ± 0.07	$\mathbf{0.03} \pm 0.09$	0.19 ± 0.11	0.21 ± 0.12
SW	0.63 ± 0.06	$\mathbf{0.01} \pm 0.13$	0.75 ± 0.13	0.18 ± 0.12	0.21 ± 0.11	0.85 ± 0.12
Average	$\mathbf{0.13} \pm 0.07$	0.18 ± 0.12	0.20 ± 0.10	0.16 ± 0.08	0.18 ± 0.09	0.44 ± 0.10

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

datasets whereas FFS and RD also sample the community structure better than FS and XS. Interestingly, one of the main focus of XS sampling is to seize community structures (Maiya and Berger-Wolf 2010), however, it seems that XS does not perform well at very small sampling fractions and it needs to explore a larger portion of the graph in order to capture communities in it. Table 9 shows that GS and TIES gives smaller error in five and four datasets respectively. On average, GS begets the least amount of error in this metric.

5.6.7 Degree distribution

We present the Empirical Cumulative Distribution Function (ECDF) of degree in Fig. 13 for all datasets at $\phi = 0.001$. FFS, FS and RD tend to pick low degree nodes at this sampling fraction whereas GS, TIES and XS select higher degree nodes that results in sub-optimal distributions. XS seems to perform good in Actors, DBLP and Skitter datasets whereas GS seems to capture distributions better in FF and SW. We show the JS distance in Table 10. We see that XS and FFS generate less error in six and five datasets respectively. On average, FFS performs the best in this metric. One possible reason of failure of GS is that when removing extra edges we are inclined to follow the clustering coefficient of a graph and hence we compromise the degree distribution.

5.6.8 Clustering coefficient distribution

We show the ECDF of clustering coefficient of all the datasets in Fig. 14. GS follows the original distributions in many networks whereas XS and RD also bring good results in a few datasets, e.g., Skitter and Facebooks. FFS and FS fail in all datasets and perform inadequately. The reason that GS fails sometimes is that the distribution is a complex metric and we need more knowledge of the original graph to extract the nodes in order to follow the distribution. We give the JS distances of this metric in Table 11. The table shows that GS samples achieve minimum distance in nine datasets whereas XS and RD generate less distance in three datasets each. On average, GS dominates by producing better distributions than other methods.

5.6.9 Path length distribution

We analyze the performance of sampling methods based on the match of the path length distributions between the original and sampled networks and plot the ECDF of path lengths of all datasets in Fig. 15. FFS and FS always give samples with higher path lengths. TIES, XS and RD seem to follow the distributions in some of the networks, e.g., CiteSeer, DBLP and LiveJournal. GS seems to approximate the path length distributions of original networks more accurately. For quantitative comparison, we give the JS distances in Table 12. We see that GS bears minimum error in eight datasets whereas TIES and XS have smaller errors in three datasets each. On average, GS achieves the best results in this metric.

Table 9 RMSE values and standard deviations of point statistics of modularity

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	0.03 ± 0.11	0.03 ± 0.07	$\mathbf{0.01} \pm 0.12$	0.25 ± 0.14	0.02 ± 0.11	0.03 ± 0.13
G.Scholar	$\mathbf{0.02} \pm 0.07$	0.18 ± 0.13	0.22 ± 0.14	0.05 ± 0.09	0.21 ± 0.11	0.11 ± 0.07
Actors	0.31 ± 0.08	0.19 ± 0.06	0.17 ± 0.07	0.21 ± 0.09	0.26 ± 0.12	$\mathbf{0.17} \pm 0.13$
Fr.Square	0.36 ± 0.11	$\mathbf{0.27} \pm 0.07$	0.34 ± 0.14	0.39 ± 0.09	0.31 ± 0.09	0.40 ± 0.07
Twitter	$\mathbf{0.11} \pm 0.07$	0.27 ± 0.09	0.11 ± 0.13	0.18 ± 0.13	0.28 ± 0.11	0.13 ± 0.12
Lastfm	0.26 ± 0.09	0.33 ± 0.05	$\mathbf{0.07} \pm 0.14$	0.16 ± 0.11	0.35 ± 0.06	0.09 ± 0.10
DBLP	0.22 ± 0.06	0.19 ± 0.10	$\mathbf{0.13} \pm 0.09$	0.31 ± 0.09	0.21 ± 0.12	0.22 ± 0.09
Hyves	0.09 ± 0.06	0.13 ± 0.12	0.07 ± 0.08	$\mathbf{0.04} \pm 0.07$	0.18 ± 0.15	0.26 ± 0.13
Skitter	0.17 ± 0.06	$\mathbf{0.09} \pm 0.06$	0.17 ± 0.12	0.24 ± 0.12	0.12 ± 0.07	0.36 ± 0.07
Flicker	0.16 ± 0.09	0.21 ± 0.11	$\mathbf{0.08} \pm 0.14$	0.39 ± 0.12	0.24 ± 0.11	0.21 ± 0.07
Flixster	$\mathbf{0.08} \pm 0.08$	0.32 ± 0.08	0.09 ± 0.12	0.19 ± 0.08	0.33 ± 0.11	0.08 ± 0.17
Facebook	0.13 ± 0.09	0.24 ± 0.17	0.07 ± 0.11	0.14 ± 0.12	0.25 ± 0.07	$\mathbf{0.05} \pm 0.09$
YouTube	$\mathbf{0.06} \pm 0.05$	0.22 ± 0.05	0.20 ± 0.06	0.27 ± 0.07	0.23 ± 0.06	0.32 ± 0.10
L.Journal	0.12 ± 0.08	0.25 ± 0.06	0.08 ± 0.08	0.17 ± 0.11	0.25 ± 0.11	$\mathbf{0.07} \pm 0.09$
FF	$\mathbf{0.02} \pm 0.09$	0.43 ± 0.11	0.41 ± 0.06	0.08 ± 0.12	0.44 ± 0.09	0.12 ± 0.07
SW	0.10 ± 0.14	0.11 ± 0.09	0.12 ± 0.09	0.09 ± 0.12	$\mathbf{0.09} \pm 0.07$	0.12 ± 0.12
Average	$\mathbf{0.14} \pm 0.08$	0.21 ± 0.08	0.15 ± 0.11	0.20 ± 0.10	0.23 ± 0.09	0.17 ± 0.10

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

Table 10 Jensen Shannon distance and standard deviations for degree distributions

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	0.37 ± 0.16	0.24 ± 0.18	0.40 ± 0.18	$\mathbf{0.24} \pm 0.06$	0.39 ± 0.3	0.43 ± 0.12
G.Scholar	0.39 ± 0.08	0.31 ± 0.11	0.52 ± 0.18	$\mathbf{0.17} \pm 0.09$	0.44 ± 0.15	0.25 ± 0.11
Actors	0.35 ± 0.19	0.48 ± 0.18	0.45 ± 0.21	$\mathbf{0.17} \pm 0.13$	0.56 ± 0.19	0.66 ± 0.11
Fr.Square	0.25 ± 0.12	$\mathbf{0.24} \pm 0.14$	0.54 ± 0.08	0.67 ± 0.13	0.51 ± 0.20	0.31 ± 0.12
Twitter	0.32 ± 0.14	0.34 ± 0.2	0.30 ± 0.18	0.32 ± 0.15	0.63 ± 0.27	$\mathbf{0.17} \pm 0.14$
Lastfm	0.46 ± 0.20	0.41 ± 0.17	$\mathbf{0.23} \pm 0.16$	0.51 ± 0.09	0.55 ± 0.29	0.29 ± 0.10
DBLP	0.51 ± 0.08	0.25 ± 0.11	0.47 ± 0.11	$\mathbf{0.17} \pm 0.06$	0.48 ± 0.27	0.34 ± 0.11
Hyves	0.35 ± 0.07	0.24 ± 0.13	$\mathbf{0.19} \pm 0.17$	0.19 ± 0.12	0.57 ± 0.09	0.27 ± 0.13
Skitter	0.20 ± 0.19	0.38 ± 0.08	0.25 ± 0.06	$\mathbf{0.15} \pm 0.12$	0.47 ± 0.24	0.51 ± 0.09
Flicker	0.49 ± 0.06	$\mathbf{0.25} \pm 0.12$	0.45 ± 0.14	0.56 ± 0.11	0.59 ± 0.09	0.32 ± 0.09
Flixster	0.39 ± 0.11	0.18 ± 0.15	0.37 ± 0.22	0.60 ± 0.09	0.58 ± 0.23	$\mathbf{0.17} \pm 0.06$
Facebook	0.57 ± 0.14	$\mathbf{0.28} \pm 0.12$	0.40 ± 0.09	0.35 ± 0.14	0.63 ± 0.21	0.29 ± 0.11
YouTube	0.19 ± 0.17	$\mathbf{0.13} \pm 0.09$	0.35 ± 0.17	0.55 ± 0.14	0.56 ± 0.12	0.48 ± 0.09
L.Journal	0.27 ± 0.18	0.44 ± 0.11	0.37 ± 0.09	$\mathbf{0.15} \pm 0.13$	0.53 ± 0.15	0.44 ± 0.15
FF	0.24 ± 0.21	$\mathbf{0.23} \pm 0.11$	0.51 ± 0.16	0.28 ± 0.14	0.48 ± 0.26	0.69 ± 0.11
SW	$\mathbf{0.21} \pm 0.15$	0.33 ± 0.18	0.61 ± 0.24	0.59 ± 0.12	0.58 ± 0.24	0.52 ± 0.16
Average	0.35 ± 0.14	$\mathbf{0.29} \pm 0.13$	0.41 ± 0.15	0.35 ± 0.11	0.53 ± 0.21	0.38 ± 0.11

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

Table 11 Jensen Shannon distance and standard deviations for clustering coefficient distributions

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	**0.22** ± 0.05	0.51 ± 0.08	0.63 ± 0.21	0.51 ± 0.09	0.50 ± 0.13	0.47 ± 0.11
G.Scholar	0.29 ± 0.08	0.48 ± 0.19	0.55 ± 0.24	**0.20** ± 0.19	0.48 ± 0.23	0.23 ± 0.07
Actors	**0.62** ± 0.12	0.74 ± 0.11	0.55 ± 0.12	0.56 ± 0.08	0.81 ± 0.18	0.70 ± 0.12
Fr.Square	**0.19** ± 0.12	0.29 ± 0.11	0.57 ± 0.21	0.77 ± 0.19	0.31 ± 0.14	0.31 ± 0.18
Twitter	**0.05** ± 0.12	0.15 ± 0.12	0.08 ± 0.16	0.12 ± 0.19	0.17 ± 0.24	0.11 ± 0.16
Lastfm	0.12 ± 0.04	0.29 ± 0.11	**0.11** ± 0.21	0.46 ± 0.15	0.31 ± 0.1	0.17 ± 0.08
DBLP	**0.42** ± 0.14	0.54 ± 0.22	0.62 ± 0.20	0.53 ± 0.14	0.55 ± 0.11	0.51 ± 0.07
Hyves	**0.07** ± 0.07	0.19 ± 0.26	0.12 ± 0.21	0.18 ± 0.10	0.18 ± 0.20	0.11 ± 0.16
Skitter	0.17 ± 0.12	0.53 ± 0.18	0.17 ± 0.06	**0.08** ± 0.08	0.54 ± 0.09	0.57 ± 0.17
Flicker	0.34 ± 0.06	0.35 ± 0.11	0.44 ± 0.07	0.56 ± 0.16	0.36 ± 0.2	**0.26** ± 0.12
Flixster	0.17 ± 0.08	0.22 ± 0.13	0.21 ± 0.11	0.48 ± 0.14	0.26 ± 0.22	**0.11** ± 0.11
Facebook	0.43 ± 0.11	0.33 ± 0.13	0.15 ± 0.09	**0.09** ± 0.11	0.34 ± 0.14	0.21 ± 0.17
YouTube	**0.15** ± 0.08	0.26 ± 0.13	0.25 ± 0.11	0.40 ± 0.16	0.42 ± 0.18	0.45 ± 0.17
L.Journal	**0.23** ± 0.09	0.52 ± 0.17	0.34 ± 0.16	0.33 ± 0.18	0.54 ± 0.22	0.38 ± 0.14
FF	**0.08** ± 0.13	0.47 ± 0.16	0.56 ± 0.13	0.18 ± 0.08	0.55 ± 0.14	0.53 ± 0.07
SW	0.60 ± 0.11	0.51 ± 0.16	0.67 ± 0.16	0.67 ± 0.09	0.55 ± 0.06	**0.51** ± 0.14
Average	**0.26** ± 0.09	0.40 ± 0.14	0.38 ± 0.15	0.38 ± 0.13	0.43 ± 0.16	0.35 ± 0.13

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

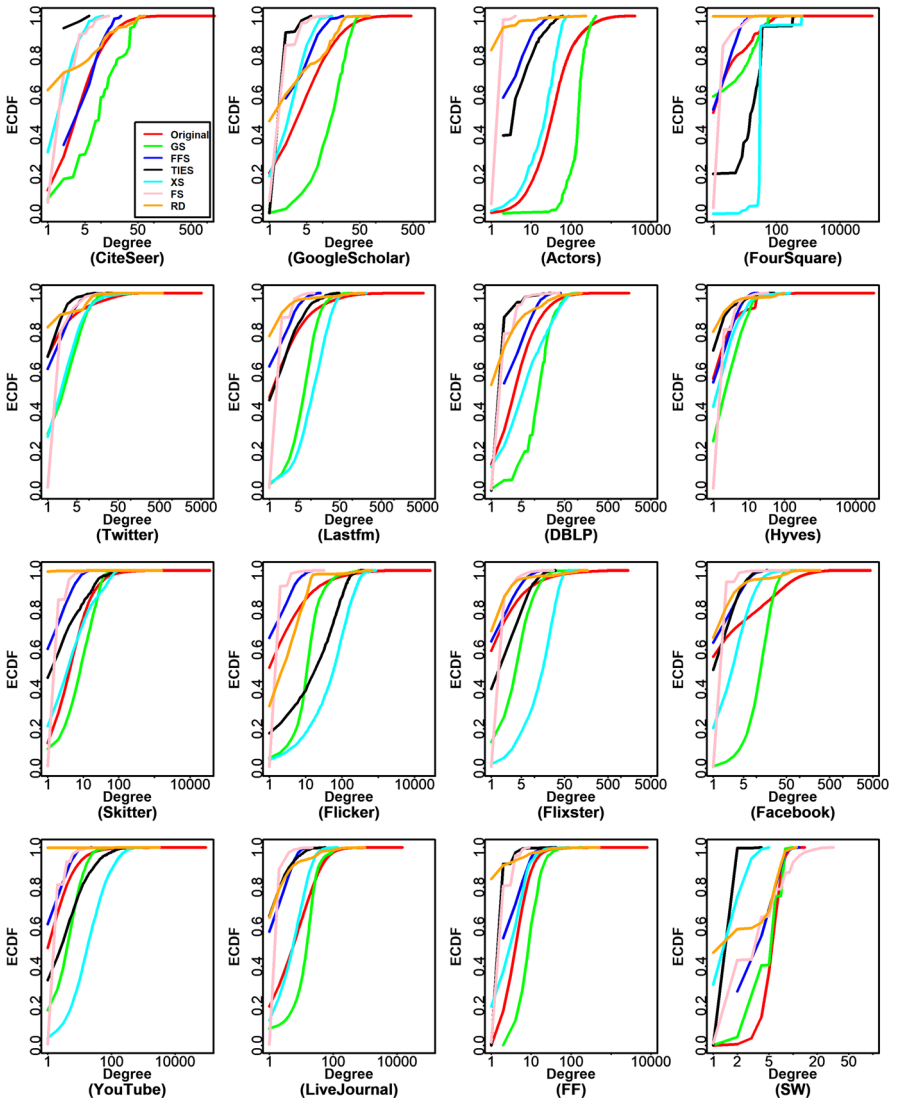

Fig. 13 Degree distributions of all networks at $\phi = 0.001$. **(Best viewed in color)**

5.6.10 Summary of results

For the purpose of discussion, the sampling methods can be combined into two groups. GS, TIES and XS are included in the first group, whereas FFS, FS and RD fall in the second group. The key difference between the two groups is the graph induction, or the inclusion of additional links existing in the original network between sampled nodes, performed by the first group methods.

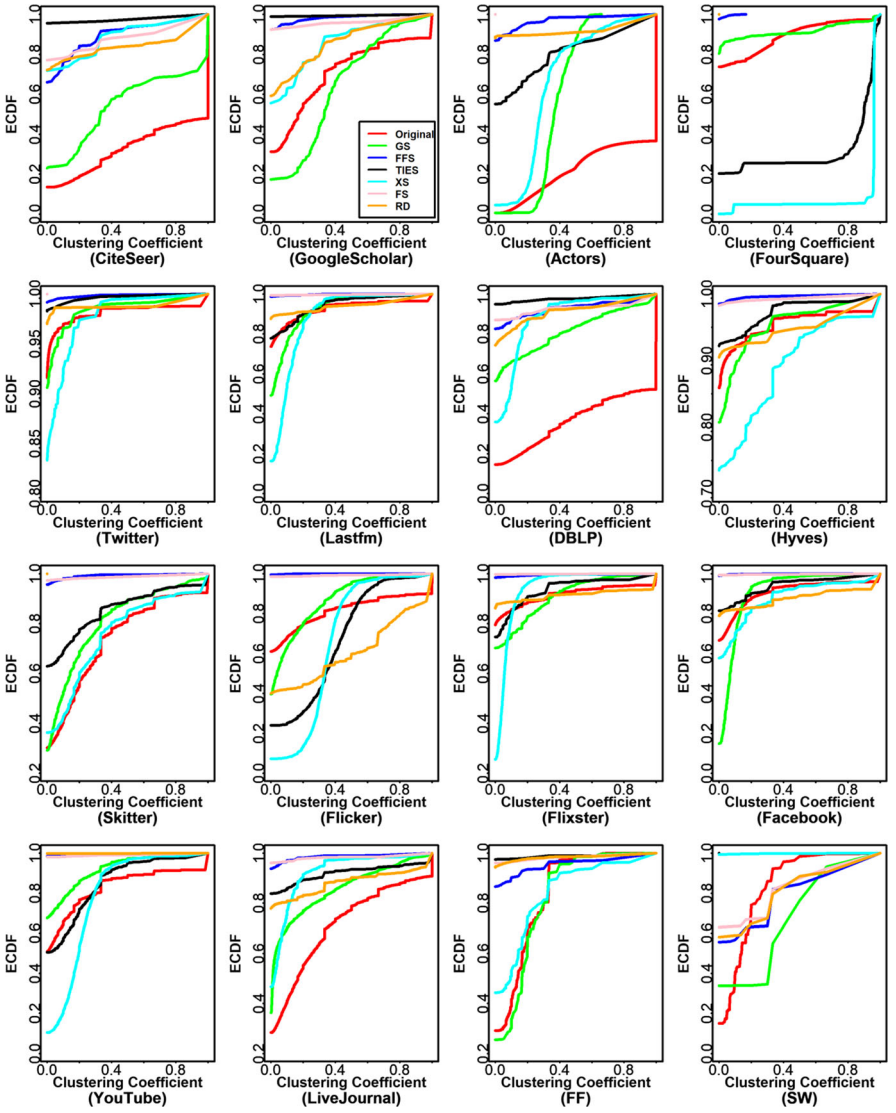

Fig. 14 Clustering coefficient distributions of all networks at $\phi = 0.001$. **(Best viewed in color)**

The methods in the first group dominate the point statistics like clustering coefficient, path length, 90% effective diameter and modularity. It seems that by selecting the nodes and then inducing the graph over these nodes the methods of the first group capture the structure of the original graph better. On the other side, the methods of group two prevail in degree and assortativity statistics. However, FFS and FS undersample the graph in terms of degree and create longer paths in the samples as seen in Figs. 9 and 15. This is in line with our observation in Fig. 5 when we preserve only the clustering coefficient and undersample in terms of degree in GS. In the case

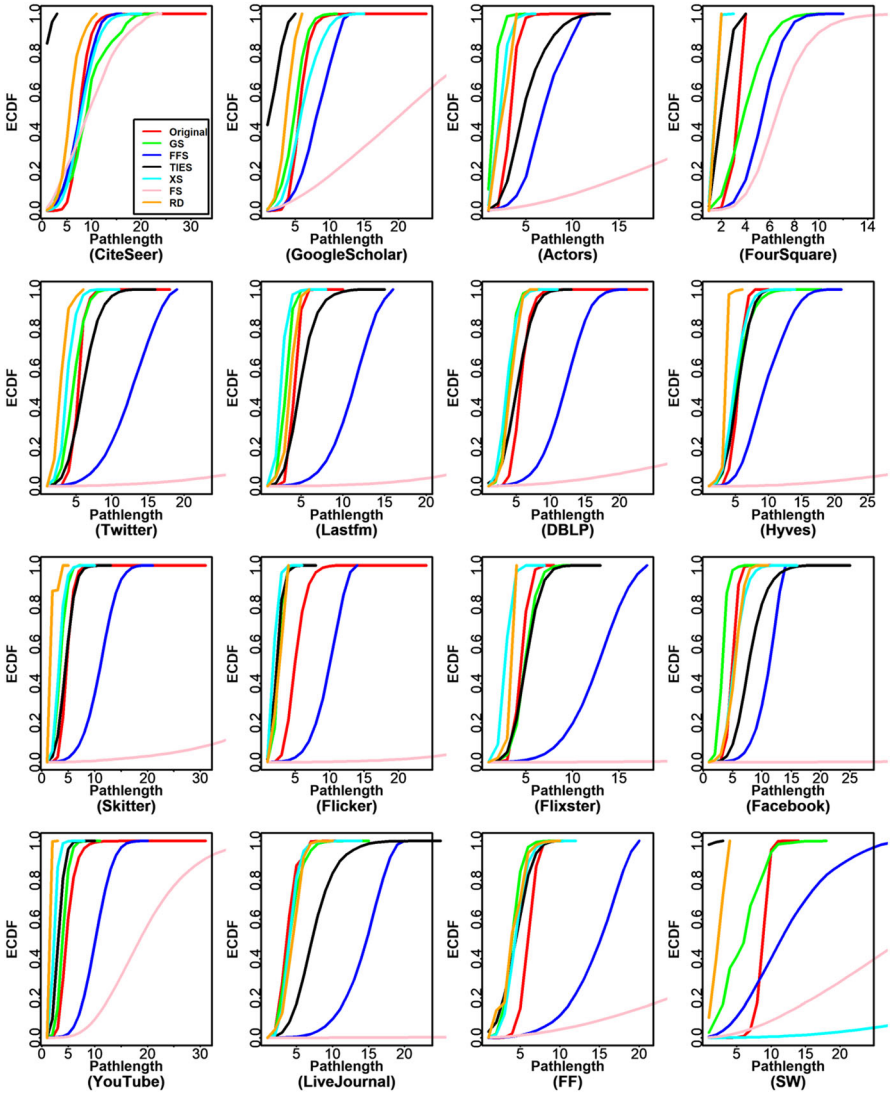

Fig. 15 Path length distributions of all networks at $\phi = 0.001$. **(Best viewed in color)**

of distributions, the first group captures clustering coefficient and path length distributions better than the second group whereas FFS of the second group give good results in extracting degree distributions of original networks. Upon close look, the results reveal that FFS picks low degree nodes more often and underestimate high degree nodes. FFS matches the degree distribution in those datasets that have many low degree nodes, e.g., FourSquare, Flicker and Flixster datasets. With the exception of degree distribution, GS surpasses all the methods in all the metrics on average and the samples obtained by GS exhibit the least deviation from the original graphs.

Table 12 Jensen Shannon distance and standard deviations for path length distributions

Dataset	GS	FFS	TIES	XS	FS	RD
CiteSeer	0.41 ± 0.12	0.30 ± 0.13	0.82 ± 0.16	**0.25** ± 0.11	0.50 ± 0.09	0.43 ± 0.08
G.Scholar	0.32 ± 0.12	0.42 ± 0.11	0.77 ± 0.12	**0.27** ± 0.15	0.59 ± 0.14	0.54 ± 0.12
Actors	0.71 ± 0.10	0.62 ± 0.22	**0.37** ± 0.16	0.52 ± 0.14	0.69 ± 0.15	0.44 ± 0.13
Fr.Square	**0.34** ± 0.16	0.61 ± 0.08	0.58 ± 0.15	0.78 ± 0.12	0.67 ± 0.17	0.78 ± 0.11
Twitter	**0.23** ± 0.13	0.69 ± 0.16	0.32 ± 0.08	0.51 ± 0.09	0.77 ± 0.10	0.68 ± 0.09
Lastfm	**0.21** ± 0.14	0.74 ± 0.06	0.34 ± 0.11	0.65 ± 0.11	0.79 ± 0.29	0.29 ± 0.15
DBLP	0.51 ± 0.11	0.66 ± 0.06	**0.35** ± 0.09	0.54 ± 0.12	0.73 ± 0.14	0.45 ± 0.11
Hyves	**0.29** ± 0.16	0.64 ± 0.17	0.28 ± 0.06	0.29 ± 0.11	0.78 ± 0.28	0.67 ± 0.07
Skitter	**0.26** ± 0.11	0.62 ± 0.17	0.27 ± 0.17	0.48 ± 0.12	0.78 ± 0.22	0.77 ± 0.14
Flicker	0.66 ± 0.12	**0.63** ± 0.12	0.68 ± 0.09	0.75 ± 0.13	0.79 ± 0.21	0.64 ± 0.14
Flixster	**0.22** ± 0.16	0.75 ± 0.05	0.26 ± 0.13	0.64 ± 0.08	0.81 ± 0.22	0.57 ± 0.14
Facebook	0.36 ± 0.14	0.75 ± 0.07	0.25 ± 0.09	**0.24** ± 0.15	0.81 ± 0.31	0.25 ± 0.15
YouTube	**0.16** ± 0.11	0.64 ± 0.19	0.46 ± 0.07	0.67 ± 0.15	0.73 ± 0.24	0.81 ± 0.17
L.Journal	**0.11** ± 0.15	0.76 ± 0.07	0.26 ± 0.11	0.13 ± 0.11	0.79 ± 0.11	0.18 ± 0.14
FF	0.58 ± 0.13	0.70 ± 0.13	**0.50** ± 0.13	0.48 ± 0.13	0.71 ± 0.12	0.56 ± 0.15
SW	0.64 ± 0.09	**0.59** ± 0.15	0.82 ± 0.12	0.77 ± 0.15	0.68 ± 0.12	0.82 ± 0.13
Average	**0.37** ± 0.13	0.63 ± 0.12	0.46 ± 0.12	0.50 ± 0.12	0.73 ± 0.18	0.55 ± 0.13

Bold values signify that they are the minimum distance for the dataset (row-wise) and the method used (column-wise). These are the best results

6 Related work

A variety of sampling techniques have been used to obtain a representative subgraph from a large graph. A very basic approach is to select nodes uniformly at random from the original graph and then induce the sample graph over this node set. By selecting nodes uniformly at random, we ensure that the selected nodes have a degree distribution similar to the original graph but this degree distribution is based on the degree of selected nodes in the original graph not in the sample graph. In addition, the induced subgraph is a disconnected graph with high probability even if the original graph is connected. A similar approach is to sample the edges uniformly at random and then add the nodes at the ends of those edges to the node set. However, in real networks that have to be crawled, it is very hard or nearly infeasible to generate a statistically valid node set or edge set where the nodes and/or edges are selected uniformly at random.

Due to the known limitations of randomly selecting nodes and/or edges for sampling real networks, the researchers have also considered topology based sampling methods including breadth first sampling (BFS) (Becchetti et al. 2006), random first sampling (RFS) (Doerr and Blenn 2013), snowball sampling (SS) (Lee et al. 2006) and different variations of random walk (RW) (Ribeeiro and Towsley 2010; Bar-Yossef and Gurevich 2008; Gkantsidis et al. 2006; Stutzbach et al. 2009; Rasti et al. 2009). All these approaches have their own pros and cons. For example, BFS produces overestimated samples in terms of degree and underestimates the path lengths in a graph but collects a well-connected graph like other topology based methods. It has been empirically observed that BFS samples are biased towards high-degree nodes (Lee et al. 2006; Najork and Wiener 2001; Becchetti et al. 2006; Ye et al. 2010). The authors in (Chiericetti et al. 2016) discuss three algorithms for sampling a node from the graph uniformly at random. In particular, they discussed Rejection Sampling, Maximum Degree Sampling and metropolis–hastings random walk (MHRW). The authors in (Lee et al. 2012) introduces two variations of random walk namely non-backtracking random walk with re-weighting and metropolis hasting algorithm with delayed acceptance for unbiased sampling of big graphs. These variations seem to improve over simple random walk and MHRW in sampling efficiency at higher sampling fractions. Similarly, the authors in (Wang et al. 2010) propose an unbiased sampling method for directed social graphs. Their method is based on MHRW and achieves a smaller error than classic uniform sampling of directed graphs. The authors in (Li et al. 2015) systematically analyze the drawbacks of the existing random walk based graph sampling algorithms and propose two algorithms to balance the tradeoff between the large deviation problem of random walk and sample rejection problem of MHRW.

The work in (Maiya and Berger-Wolf 2011) provides a detailed study on the nature of biases in network sampling. In particular, they studied the nature of biases in several sampling algorithms including Breadth First Sampling, Depth First Sampling, Random Walk, Forest Fire Sampling, Degree Sampling, Sample Edge Count and Expansion Sampling. The authors also described how these sampling biases can be exploited in several real-world applications including disease outbreak detection and market research. The work in (Wang et al. 2011) provides a good understanding of how sampling works in big graphs. The authors analyze several graph sampling algorithms and

evaluate their performance on some widely recognized graph properties on directed graphs using large-scale social network datasets. The sampling approach presented in (Kim et al. 2014) is a computational approach to predict RNA 3D topologies based on hierarchical sampling. This graph-based sampling approach for characterizing global helical arrangements in large RNAs is developed for biological networks and have applications in medicine and related technologies. The authors in (Chepuri and Leus 2017) focus on subsampling as well as reconstructing the second-order statistics of signals residing on nodes of arbitrary undirected graphs. The work in (Al Hasan and Zaki 2009) propose a generic sampling framework that is based on Metropolis–Hastings algorithm to sample the output space of frequent subgraphs. More recently, some researchers have deployed variations of Random Walk and Delay Sampling (Xu and Zhu 2016; Xu and Lee 2014; Xu et al. 2017; Chen et al. 2017a; Liu et al. 2019) to sample big graphs with good results. However, the sampling rates are as big as 0.3 and given the fact that a real network could have millions or even billions of nodes, this sampling fraction seems too big.

There has also been some attempts that rely on first estimating the different properties of the original graph and then extracting a representative subgraph based on the estimated values. In (Hubler et al. 2008), the authors propose Metropolis algorithms that refine the subgraph by replacing the sampled nodes with other potentially good nodes guided by simulated annealing to better match the properties of the original graph. The fundamental problem is that this method also relies on uniformly selecting samples from the entire network and it makes it infeasible for crawling real networks. The authors of report (Sethu and Chu 2012) apply Metropolized Random Walk guided by the degree exponent of the original graph. The fundamental problem is that the Biased Random Walk with Fly Back (BRW-FB), proposed in the report, converges slowly and makes it prohibitively time consuming when applied on big graphs. In addition, the work is presented as a technical report and is not evaluated over a real dataset but an instance of a Barabasi-Albert scale free network model that makes it less competitive for comparison. The interested readers can refer to (Rasti et al. 2009; Gjoka et al. 2010) for comparison between Random Walk and Metropolized Random Walk.

In comparison to the previous sampling approaches discussed in this section, a very clear difference between GS and previous approaches is that GS can work at very small sampling fractions. GS can capture really tiny samples while preserving the structure of the original graph. To the best of our knowledge, we are the first one to extract samples at sampling fractions as small as 0.1%. The previous approaches have reported a minimum of 1% samples and in most of the studies the methods are tested at higher sampling fractions. As mentioned in the introduction, samples obtained at higher sampling fractions fade away the purpose of sampling. By analyzing very small representative samples of big graphs, we can save time and resources needed to study massive graphs. Another significant difference between GS and previous approaches is its two-step recipe to obtain samples. We first generate a sample that has higher density and clustering coefficient, and then trim it by removing edges based on their contribution to clustering coefficient. This two-step approach is the key to extract tiny samples and it can lead to multi-stage sampling in future where at every stage we can apply a different technique to guide the sampling process to known targets.

7 Conclusion

Graph sampling makes it possible to study and analyze big networks with limited resources provided that the sample graph accurately represents the original graph. In order to realize tiny samples from a large graph, we propose a new two-step sampling technique that exploits the fact that the average degree and clustering coefficient of a graph could be estimated efficiently and these estimated values could guide us to yield good samples. In the first step of the proposed method, we collect samples with extra edges and then in the second step we remove extra edges to match with the estimated values of degree and clustering coefficient of the graph being sampled. Our guided sampling technique extracts good samples at very small sampling fractions and preserves key properties of a graph, its degree, clustering coefficient, path length, effective diameter and structural properties, its degree assortativity and modularity. Through experiments on real and synthetic networks and statistical tests, we show that our new approach surpasses the existing sampling methods in all the key metrics discussed in the paper. At very small sampling fractions, i.e., less than 1%, conventional sampling methods do not get enough chance to explore the graph, and either underestimate or overestimate its properties. Our two-step approach first ovesamples the graph and then trims it to extract a reasonable sample at a very small sampling fraction.

We believe that this two-step sampling approach can motivate the design of better multi-phase sampling techniques in the future. The idea of guiding the sampling process to the already estimated values can have far reaching implications. The idea can be generalized to use prior information in the sampling process. A guided sampler can extract representative samples by mining a very small portion of the graph and this can make the analysis of big graphs feasible even with limited resources.

Acknowledgements This work was supported by Korea Institute of Science and Technology (KIST) under the project "HERO Part 1: Development of core technology of ambient intelligence for proactive service in digital in-home care", and by the Technology Innovation Program (20006489, "Development of the embedded robot equipment control system equipped with an AI vision module for industrial environment") funded by the Ministry of Trade, Industry & Energy (MOTIE, Korea).

References

Ahmed N, Neville J, Kompella RR (2011) Network sampling via edge-based node selection with graph induction. Technical Report 11-016, Purdue Digital Library

Ahn Y, Han S, Kwak H, Moon S, Jeong H (2007) Analysis of topological characteristics of huge online social networking services. In: Proceedings of WWW, pp 835–844

Al Hasan M, Zaki MJ (2009) Output space sampling for graph patterns. Proc VLDB Endow 2(1):730–741

Bar-Yossef Z, Gurevich M (2008) Random sampling from a search engine's index. J ACM 55(5):24:1–24:74

Becchetti L, Castillo C, Donato D, Fazzone A (2006) A comparison of sampling techniques for web graph characterization. In: LinkKDD

Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech Theory Exp 10:P10008

Chen B, Liu L, Jia H, Zhang Y (2017a) Reducing repetition rate: unbiased delay sampling in online social networks. Recent Pat Comput Sci 10(4):308–314

Chen Y, Ding C, Hu J, Chen R, Hui P, Fu X (2017b) Building and analyzing a global co-authorship network using google scholar data. In: Proceedings of 26th international World Wide Web conference (WWW 2017) Companion

Chepuri SP, Leus G (2017) Graph sampling for covariance estimation. IEEE Trans Signal Inf Process Over Netw 3:451–466
Chiericetti F, Dasgupta A, Kumar R, Lattanzi S, Sarlós T (2016) On sampling nodes in a network. In: Proceedings of the 25th international conference on World Wide Web, WWW '16, pp 471–481
Doerr C, Blenn N (2013) Metric convergence in social network sampling. In: ACM Hotplanet
Feige U (1995) A tight upper bound on the cover time for random walks on graphs. Random Struct Algorithms 6:51–54
Gjoka M, Kurant M, Butts C, Markopoulou A (2010) Walking in facebook: a case study of unbiased sampling of OSNS. In: INFOCOM
Gkantsidis C, Mihail M, Saberi A (2006) Random walks in peer-to-peer networks: algorithms and evaluation. Perform Eval 63(3):241–263
Hardiman SJ, Katzir L (2013) Estimating clustering coefficient and size of social networks via random walk. In: ACM's WWW
Hubler C, Kriegel P, Borgwardt KM, Ghahramani Z (2008) Metropolis algorithms for representative subgraph sampling. In: ICDM
Hu P, Lau WC (2014) A survey and taxonomy of graph sampling. In: HONGKONG UNI
Kim N, Laing C, Elmetwaly S, Jung S, Curuksu J, Schlick T (2014) Graph-based sampling for approximating global helical topologies of RNA. Proc Natl Acad Sci 111(11):4079–4084
Konect (2015) Network dataset—KONECT. http://konect.uni-koblenz.de/networks/. Accessed Sept 2018
Lee CH, Xu X, Eun DY (2012) Beyond random walk and metropolis–hastings samplers: why you should not backtrack for unbiased graph sampling. In: Proceedings of the 12th ACM SIGMETRICS/PERFORMANCE joint international conference on measurement and modeling of computer systems, SIGMETRICS '12, pp 319–330
Lee S, Kim P, Jeong H (2006) Statistical properties of sampled networks. Phys Rev E 73:016102
Leskovec J, Kleinberg J, Faloutsos C (2007) Graph evolution: densification and shrinking diameters. ACM Trans Knowl Discov Data 1(1):2. https://doi.org/10.1145/1217299.1217301
Leskovec J, Faloutsos C (2006) Sampling from large graphs. In: SIGKDD, pp 631–636
Leskovec J, Krevl A (2014) SNAP datasets: Stanford large network dataset collection. http://snap.stanford.edu/data. Accessed Sept 2018
Liu L, Wang L, Wu W, Jia H, Zhang Y (2019) A novel hybrid-jump-based sampling method for complex social networks. IEEE Trans Comput Soc Syst 6(2):241–249
Li R, Yu JX, Qin L, Mao R, Jin T (2015) On random walk based graph sampling. In: 2015 IEEE 31st international conference on data engineering, pp 927–938
Maiya AS, Berger-Wolf TY (2010) Sampling community structure. In: Proceedings of the 19th international conference on World Wide Web, WWW '10, pp 701–710
Maiya AC, Berger-Wolf TY (2011) Benefits of bias: towards better characterization of network sampling. In: ACM KDD
Najork M, Wiener JL (2001) Breadth-first crawling yields high-quality pages. In: Proceedings of the 10th international conference on World Wide Web, WWW '01, pp 114–118
Rasti AH, Torkjazi M, Rejaie R, Duffield NG, Willinger W, Stutzbach D (2009) Respondent-driven sampling for characterizing unstructured overlays. In: INFOCOM 2009. 28th IEEE international conference on computer communications, 19–25 April 2009, Rio de Janeiro, Brazil, pp 2701–2705
Ribeeiro B, Towsley D (2010) Estimating and sampling graphs with multidimensional random walks. In: ACM internet measurement conference
Rossi RA, Ahmed NK (2015) The network data repository with interactive graph analytics and visualization. http://networkrepository.com. Accessed Sept 2018
Sethu H, Chu X (2012) A new algorithm for extracting a small representative subgraph from a very large graph. arXiv:1207.4825
Stutzbach D, Rejaie R, Duffield N, Sen S, Willinger W (2009) On unbiased sampling for unstructured peer-to-peer networks. IEEE/ACM Trans Netw 17(2):377–390
Voudigari E, Salamanos N, Papageorgiou T, Yannakoudakis EJ (2016) Rank degree: an efficient algorithm for graph sampling. In: 2016 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM), pp 120–129
Wang T, Chen Y, Zhang Z, Sun P, Deng B, Li X (2010) Unbiased sampling in directed social graph. In: Proceedings of the ACM SIGCOMM 2010 conference, SIGCOMM '10, pp 401–402

Wang T, Chen Y, Zhang Z, Xu T, Jin L, Hui P, Deng B, Li X (2011) Understanding graph sampling algorithms for social network analysis. In: Proceedings of the 2011 31st international conference on distributed computing systems workshops, ICDCSW '11, pp 123–128

Watts DJ, Strogatz SH (1998) Collective dynamics of 'small-world' networks. Nature 393:440–442

Xu X, Lee C (2014) A general framework of hybrid graph sampling for complex network analysis. In: Proceedings of INFOCOM

Xu XK, Zhu JJ (2016) Flexible sampling large-scale social networks by self-adjustable random walk. Phys A: Stat Mech Appl 463:356–365

Xu X, Lee CH et al (2017) Challenging the limits: sampling online social networks with cost constraints. In: IEEE INFOCOM 2017-IEEE conference on computer communications, pp 1–9

Ye S, Lang J, Wu F (2010) Crawling online social graphs. In: Proceedings of the 2010 12th international Asia-Pacific web conference, pp 236–242

Zafarani R, Liu H (2009) Social computing data repository at ASU. School of Computing, Informatics and Decision Systems Engineering