GraphSDH: A General Graph Sampling Framework with Distribution and Hierarchy

Jingbo Hu, Guohao Dai, Yu Wang, Huazhong Yang
Department of Electronic Engineering, BNRist, Tsinghua University, Beijing, China
{hjb19}@mails.tsinghua.edu.cn, {daiguohao, yu-wang, yanghz}@tsinghua.edu.cn

Abstract—**Large-scale graphs play a vital role in various applications, but it is limited by the long processing time. Graph sampling is an effective way to reduce the amount of graph data and accelerate the algorithm. However, previous work usually lacks theoretical analysis related to graph algorithm models. In this study, GraphSDH (Graph Sampling with Distribution and Hierarchy), a general large-scale graph sampling framework is established based on the vertex-centric graph model. According to four common sampling techniques, we derive the sampling probability to minimize the variance, and optimize the design according to whether there is a pre-estimation process for the intermediate value. In order to further improve the accuracy of the graph algorithm, we propose a stratified sampling method based on vertex degree and a hierarchical optimization scheme based on sampling position analysis. Extensive experiments on large graphs show that GraphSDH can achieve over 95% accuracy for PageRank by sampling only 10% edges of the original graph, and speed up PageRank by several times than that of the non-sampling case. Compared with random neighbor sampling, GraphSDH can reduce the mean relative error of PageRank by about 17% at a sampling neighbor ratio (sampling fraction) of 20%. Furthermore, GraphSDH can be applied to various graph algorithms, such as Breadth-First Search (BFS), Alternating Least Squares (ALS) and Label Propagation Algorithm (LPA).**

I. Introduction

Graph, as a data structure which can represent the relationship between data, is widely used in many fields such as social network analysis [1], intelligent recommendation system [2], and biological network [3], etc. With the advent of the era of big data, the scale of graphs is also expanding (e.g. a typical social network has billions of vertices and tens of billions of edges). Even for simple graph algorithms, this will lead to expensive calculation, which brings severe challenges to data analysis.

Graph sampling is a technique for picking a subset of vertices and/or edges from the original graph. It can accelerate the processing effectively under the premise of ensuring the algorithm accuracy (e.g. GraphSAGE [12] and FastGCN [13] can improve the processing speed of graph neural network algorithm by 1-2 orders of magnitudes). However, most of the previous work focuses on the design of sampling method of single graph algorithm, which is lack of generality [7], [8], [10], [11]. On the other hand, some work based on experimental research lacks theoretical analysis for different sampling characteristics to ensure accuracy [5], [19]. Therefore, it is necessary to design a unified mapping scheme between graph algorithm and sampling approach.

In this work, we present GraphSDH, a general graph sampling framework, to accelerate different graph algorithms. Our main contributions are as follows:

- **We propose sampling approaches for variance reduction**. According to four common graph sampling techniques, we strictly derive the optimal sampling probability in theory. Our sampling approach can achieve over 95% accuracy for PageRank [11] by sampling only 10% edges of the original graph.
- **We propose a stratified sampling strategy to further improve the algorithm accuracy**. We classify the vertices based on their degrees, and sample their neighbors in different scales. Compared with random neighbor sampling, stratified sampling can improve the accuracy of PageRank by 5% to 8%.
- **We propose a hierarchical sampling optimization scheme**. We apply sampling techniques to the stage of fast updating vertex values or the stage with a fair amount of redundant information. The scheme can increase the updated ratio of vertices in BFS [20] by 20%, and reduce the mean relative error of PageRank to about 1%.
- **We have carried out extensive experiments to prove the effectiveness and generality of GraphSDH**. Experimental results show that compared with random neighbor sampling, GraphSDH can improve the accuracy of PageRank by about 17%, BFS by about 75%, ALS [25] by about 95%, and LPA [26] by about 8%.

The rest of the paper is organized as follows. Section II presents the related work about different sampling methods on graph algorithms. In Section III, a unified graph algorithm model is established and a general graph sampling mapping framework is given based on variance analysis. The experimental process and results analysis are shown in Section IV. Section V concludes the paper.

II. Related Work

The commonly studied graph sampling techniques can be divided into four categories: Vertex Sampling (VS), Edge Sampling (ES), Vertex Sampling with Neighbourhood (VSN) and Traversal Based Sampling (TBS) [4].

- **Vertex Sampling (VS)**. It selects vertices from the original graph uniformly [14] or based on different probabilities [6], [9], [13]. Then a subgraph can be created by connecting edges between these sampled vertices.
- **Edge Sampling (ES)**. In this method, edges are selected to form a subgraph [11]. However, ES often leads to

sparse connectivity when it only contains the sampled edges and the vertices at both ends of them [5].

- **Vertex Sampling with Neighbourhood (VSN)**. This approach takes each vertex as a center and samples their neighbors which have direct connections with them [4]. VSN is more intuitive than other sampling methods because graph algorithms need to aggregate the information of neighbor vertices [12], [15].
- **Traversal Based Sampling (TBS)**. TBS is a multi-stage sampling process, which selects a set of initial vertices and edges first, and then expands the set according to the current observation results [16], [17], [18].

In recent years, the research based on the above sampling techniques has been carried out. [7], [8], [10], [11] design sampling strategies for a single graph algorithm or attribute, but they cannot be applied to other graph algorithms and lacks generality. By experimenting with various sampling approaches, [5], [19] find the relatively optimal sampling method to match different graph attributes, but lack of corresponding theoretical guidance. [12], [13] sum up the aggregation model of neighbor information and derived the sampling probability which can minimize the variance, but they can only be used in the graph neural network (GNN). On the basis of them, our work use a more general algorithm model to design a sampling framework called GraphSDH, which further covers the traditional graph algorithm, such as PageRank, BFS, LPA, and ALS.

III. Methodology

In this section, the notations and the vertex-centric graph model are defined first. Based on the model, we derive the theoretical minimum variance under the four sampling algorithms. In order to further reduce the accuracy loss caused by sampling, a novel hierarchical optimization scheme is proposed in Section III. C. Finally, we introduce the workflow of our general sampling framework GraphSDH.

A. Terms and Notations

A graph $G = (V, E)$ consists of its vertices V and edges E. Define $E \subseteq \{(u, v)|u \in V, v \in V\}$, where (u, v) is an unordered pair for the undirected graph, or an order pair from u to v for the directed graph. The neighbour of vertex v is denoted as $N(v) = \{u|(u, v) \in E, u \in V\}$. Denote $n = |V|$, and $m = |E|$. For a directed graph, the out-degree of vertex v is defined as $outD(v)$, and the in-degree of v is defined as $inD(v)$. For an undirected graph, $D(v)$ represents the degree of vertex v. We denote the sampled graph by $G_s = (V_s, E_s)$, where $V_s \in V$, and $E_s \in E$.

The vertex-centric model is one of the most popular abstractions. It is general enough to express a variety of algorithms whose computation for a vertex is to update its value by aggregating the information of its neighbors. An iterative graph algorithm map easily to the vertex-centric model, and for the $(k + 1)th$ iteration, the update of vertex v can be expressed as follows:

$$I'_{(k+1)}(v) = \sum_{u \in V} A(u, v) I_{(k)}(u) f(u, v) \tag{1}$$

$$I_{(k+1)}(v) = g(I'_{(k+1)}(v)) \tag{2}$$

where $I_{(k)}(u)$ is the value of vertex u in the $k-th$ iteration, A is the graph adjacency matrix, $A(u, v) = 1$ if $(u, v) \in E$, and 0 otherwise. $f(u, v)$ is a mapping function related to a specific algorithm, for example, for PageRank, $f(u, v) = 1/outD(u)$, while for BFS, if u has the minimum value in v's neighbors, $f(u, v) = 1$, otherwise, $f(u, v) = 0$. $I'_{(k+1)}(v)$ is an intermediate variable, which can obtain the updated value in the $(k + 1)th$ iteration through a transformation g. Function g may have different mathematical expressions, but it is uniquely determined by the graph algorithm.

B. Variance Reduction

When designing a sampling method, one of the most important aspects is to improve the variance with accurate value. Since the form of function g is unknown, it is difficult to discuss the influence of sampling on (2). Therefore, we aim to analyze how to reduce the variance of (1) under four sampling techniques in detail. For convenience, we only consider one iteration of the graph algorithm and simplify the symbols as: $M(v)$ instead of $I'_{(k+1)}(v)$, and $I(u)$ instead of $I_{(k)}(u)$.

a) Vertex Sampling with Neighbourhood (VSN):

We assume that the number of sampling neighbors has been determined and treated it as a constant, which is included in function g. Then the result of (1) after VSN can be approximately evaluated as

$$M(v) = \sum_{u \in N_s(v)} \frac{I(u) f(u, v)}{\alpha(u, v)} \tag{3}$$

where $N_s(v)$ represents the neighbor vertices of v in the sampling set, and $\alpha(u, v)$ is a Aggregation parameter. When the aggregate function cannot be directly represented by the sampling probability, $\alpha(u, v)$ equals to 1, otherwise, its value is the probability of sampling u under the condition of v. Since the former case is not directly related to the sampling probability, we use importance sampling to analyze the second case in detail, and (3) can be modified into (4).

$$M(v) = \sum_{u \in N_s(v)} I(u) f(u, v) \frac{dS(u)}{dS_m(u)} \tag{4}$$

where $S(u)$ is the sampling probability of u, and $S_m(u)$ is the probability measure. In VSN, we consider each iteration separately, and each target vertex can be regarded as independent, that is, the variance of graph G can be calculated as the sum of the variance of all vertices. Therefore, we only need to analyze the variance of one vertex. In order to facilitate the analysis, the expectation and the variance of (4) are transformed into integral form, as Chen did in FastGCN [13]. Conditioned on v, we can derive that

$$E(M(v)) = \int I(u) f(u, v) dS(u) = E \tag{5}$$

$$Var(M(v)) = \frac{\int I(u)^2 f(u,v)^2 dS(u)^2}{dS_m(u)} - E^2 \tag{6}$$

It is notable that the optimal $dS_m(u)$ must be proportional to $|I(u)f(u,v)|dS(u)$, and meet the condition that it integrate to unity. Therefore, we can get:

$$dS_m(u) = \frac{|I(u)f(u,v)|dS(u)}{\int |I(u)f(u,v)|dS(u)} \tag{7}$$

For the target vertex v, the sampling probability of its neighbor u is:

$$\frac{dS(u)}{dS_m(u)} = \frac{|I(u)f(u,v)|}{\sum\limits_{u' \in N(v)} |I(u')f(u',v)|} \tag{8}$$

The disadvantage of defining $dS_m(u)$ directly according to (8) is that it contains $I(u)$, which is constantly changing in the iterative process and has high computational complexity. Therefore, $I(u)$ can be estimated by appropriate pre-processing. If the algorithm does not have prior knowledge, i.e. it does not include pre-processing steps, then (8) will be simplified. The detailed optimization schemes of four sampling techniques are shown in Section III. D.

b) Vertex Sampling (VS):

The difference between VS and VSN is that the adjacency matrix A needs to be involved. The result of vertex v after VS is shown in (9).

$$M(v) = \sum_{u \in V_s} A(u,v)I(u)f(u,v)\frac{dS(u)}{dS_m(u)} \tag{9}$$

Based on the variance analysis of sampled graph, similar to the process in VSN, we weigh the accuracy and cost which is inspired by Chen et al. [13], and finally define $dS(u)/dS_m(u)$:

$$\frac{dS(u)}{dS_m(u)} = \frac{||A(:,u)||^2||f(:,u)||^2}{\sum\limits_{u' \in V} ||A(:,u')||^2||f(:,u')||^2} \tag{10}$$

c) Edge Sampling (ES):

The intermediate result $M(v)$ based on ES is

$$M(v) = \sum_{e=(u,v) \in E_s} \frac{I(u)f(u,v)}{S_e} \tag{11}$$

where S_e is the probability of sampling (u,v). Then we derive the variance aiming at an undirected graph (for a directed graph, only simple transformation needs to be made), and $R_e = I(u)f(u,v) + I(v)f(v,u)$.

$$Var(G_{sm}) = \sum_e \frac{R_e^2}{S_e} - (\sum_e R_e)^2 \tag{12}$$

To minimize the variance, we get S_e by Cauchy-Schwarz inequality.

$$S_e = \frac{|R_e|}{\sum\limits_{e' \in E} |R_{e'}|} \tag{13}$$

d) Traversal Based Sampling (TBS):

In TBS, we assume that only one subgraph is sampled before the algorithm starts, that is, the sampled subgraph is used for calculation in each subsequent iteration. In this paper, We mainly consider two sampling approaches in TBS, including vertex-based subgraph sampling and edge-based subgraph sampling. The details are shown in Algorithm 1.

Algorithm 1 Traversal Based Sampling in GraphSDH.

```
Input: Graph $G = (V,E)$; Number of sampled vertices $|V_s|$;
    Number of sampled edges $|E_s|$; Probability of sampling
    vertices $P_v$; Probability of sampling edges $P_e$;
Output: Sampled subgraph $G_s$;
function VERTEX($|V_s|, P_v$)
    $V_s \leftarrow$ Sampling vertices according to $|V_s|$ and $P_v$
    $E_s \leftarrow \{(u,v)|(u,v) \in E, u \in V_s, v \in V_s\}$
    $G_s \leftarrow (V_s, E_s)$.
end function
function EDGE($|E_s|, P_e$)
    $E_s \leftarrow$ Sampling edges according to $|E_s|$ and $P_e$
    $V_s \leftarrow$ Set of vertices that are end-points of $E_s$
    $G_s \leftarrow (V_s, E_s)$.
end function
```

C. Hierarchical Optimization Scheme

In order to further improve the accuracy of the above sampling approaches, we explore the impact of the sampling position. On this basis, we propose a hierarchical optimization scheme. For an iterative algorithm, the sampling strategy is more suitable for two situations. One is the stage of quickly updating the vertex value, that is, when the difference between the vertex values in the two successive iterations is greater than a threshold, the sampling technique is applied. Otherwise, the original graph is used for calculation instead of a sampled graph. The other is the stage with too much redundant information, that is, some edges and vertices are independent of the graph algorithm. The latter situation is easy to understand intuitively, so we mainly analyze the former one.

In order to prove the effectiveness of the above strategy in theory, we take VSN as an example, and sample the neighbor vertices randomly. The preset sampling ratio is set to $1/n$, then for vertex v, the number of neighbors after sampling is $t = \left\lceil \frac{|N(v)|}{n} \right\rceil$. The total original value of graph G in the $(k+1)th$ iteration is:

$$G^{(k+1)} = \sum_{v \in V} g(\sum_{u \in N(v)} I_{(k)}(u)f(u,v)) \tag{14}$$

and the total value after sampling is:

$${G_s}^{(k+1)} = \sum_{v \in V} g(\frac{|N(v)|}{t} \sum_{u \in N_s(v)} I_{(k)}(u)f(u,v)) \tag{15}$$

The difference between the original value and the sampled value is shown in (16). It is limited to a certain range according to the specific algorithm and data distribution, and the maximum value is expressed as $\max(DIF_s)$.

$${DIF_s}^{(k+1)} = |G_s^{(k+1)} - G^{(k+1)}| \in [0, \max(DIF_s)] \tag{16}$$

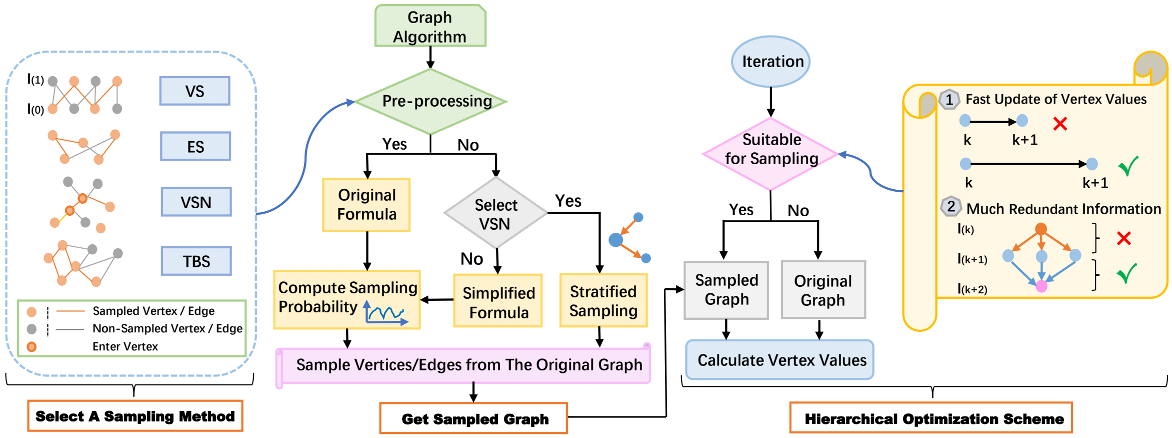


Fig. 1. The workflow of GraphSDH. It is mainly divided into three parts. The first part selects one of the four common sampling methods according to the algorithm requirements. The second part describes the process of obtaining the sampled graph, which is related to the pre-processing and sampling methods. In the last part, the hierarchical optimization scheme is applied, which needs to judge whether each iteration is suitable for sampling.

The difference of vertex values between the $k-th$ iteration and the $(k+1)th$ iteration is:

$$DIF^{(k,k+1)} = |G^{(k+1)} - G^{(k)}| \tag{17}$$

When $\max(DIF_s) \ll DIF^{(k,k+1)}$, the sampling process has little effect on the accuracy of the graph algorithm. On the contrary, sampling may cause the trend of changing vertex values to be opposite to that of the algorithm. In practical application, this strategy can be formulated to stop using the sampling approach when the relative error in the two iterations is no longer reduced. It does not increase the computational complexity, because it is consistent with the convergence condition of the iterative algorithm.

D. The workflow of GraphSDH

Based on the above analysis of variance and sampling position, we propose GraphSDH, whose workflow is shown in Fig. 1. The steps of GraphSDH for accelerating large-scale graph algorithms are as follows:

a. **Selection of sampling technique**. According to the graph algorithm and its metric, select an appropriate sampling approach.

b. **Generation of sampled graph**. Get the sampled graph according to its corresponding sampling probability or optimization strategy.Due to TBS is based on the sampling probability of VS and ES (see algorithm 1 for details), we only need to consider the other three sampling approaches except for TBS. According to whether data pre-processing is carried out, which is about estimating the vertex value I in the previous iteration, the situation can be divided into the following two types:

   i. The above pre-processing is implemented based on the prior information of graph algorithm. Then the approximate value of $I(u)f(u,v)$ can be obtained, and the sampling probability P_n of VSN, P_v of VS, P_e of ES can be calculated according to (8) (11) (15) respectively.

   ii. The above pre-processing is not implemented, due to the lack of prior information or the pre-processing is time-consuming. As a reasonable simplification, the sampling probability of ES is modified to $P_e \propto \frac{f(u,v)}{D(u)} + \frac{f(v,u)}{D(v)}$. When the f of the graph algorithm is also uncertain, it can be omitted to further simplify the calculation of sampling probability in VS and ES. For VSN, the importance sampling based on the simplified probability of neighbor vertices often leads to the inaccuracy of the algorithm, so it can be transformed into the stratified sampling. Considering a directed graph, according to the law of large numbers, the vertex with a large in-degree is more suitable for VSN. On the other hand, when a vertex transmits information, it is related to the proportion of its in-degree and out-degree r_d. That is, the larger the r_d of a vertex, the more important the information it transmits to its neighbors. Therefore, we consider both in-degree and r_d in the stratified sampling. The specific experimental results and analysis are shown in Section IV. B.

c. **Application of hierarchical optimization scheme**. Adopt the hierarchical optimization scheme to further improve the accuracy of a graph algorithm. It is determined in advance which stage in the iterative process is suitable for sampling, that is, considering the update speed of vertex values or the amount of redundant information. Finally, use the sampled graph at the appropriate stage, otherwise, use the original graph for calculation.

IV. Experimental Results

In this section, we present experimental results to evaluate the validity of our theoretical results in Section III when applying our sampling framework GraphSDH.

A. Experimental Setup

Experiments are performed on a 64-bit Ubuntu 16.04.4 server, with 64 GB of RAM Memory and a 3.60GHz Intel(R) Core(TM) i7-9700K CPU. The implementations of different graph algorithms are created using GraphChi, which is a disk-based single-machine system following the vertex-centric programming model. We conduct our evaluation using real-world datasets obtained from the Stanford Large Network Dataset Collection [21]. The statistics of the datasets are shown in Table I. In addition, we use the MovieLens dataset [23] to evaluate ALS algorithm, which contains 943 users and 1682 movies.

TABLE I
LARGE GRAPH DATASETS USED IN EXPERIMENTATION

Graph	Type	#Vertices	#Edges
soc-LiveJournal1	Directed	4,847,571	68,993,773
amazon0302	Directed	262,111	1,234,877
soc-Pokec	Directed	1,632,803	30,622,564
web-Google	Directed	875,713	5,105,039
web-BerkStan	Directed	685,230	7,600,595
amazon0601	Directed	403,394	3,387,388
com-Amazon	Undirected	334,863	925,872
ego-Facebook	Undirected	4,039	88,234

B. A Case Study of PageRank for Sampling Approaches

We first consider the case that we can rely on the prior information of the graph algorithm to sample. Take PageRank as an example, and perform TBS based on vertex sampling distribution in (11). PageRank is a popular algorithm, which is often used to measure the importance of vertices in a graph. The PageRank value $PR(v)$ of vertex v is defined as:

$$PR(v) = \alpha \sum_{u \in N(v)} \frac{PR(u)}{outD(u)} + \frac{1-\alpha}{|V|} \qquad (18)$$

α is the damping factor, which is usually taken as 0.85. Therefore, the vertex sampling probability in (11) is transformed into $P(u) \propto \frac{inD(u)}{outD(u)}$. Here, we use Mean Average Precision (MAP) as the metric for TBS [22], and regard the first 1000 vertices as important ones. The MAP results at different sampling scales for amazon0601 dataset is shown in Fig. 2. Observe that MAP has reached more than 95% even at 10% sampling ratio (sampling 10% edges of the original graph), which shows the effectiveness of this sampling method.

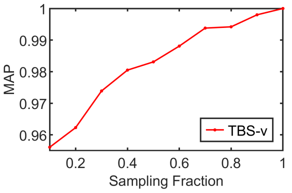
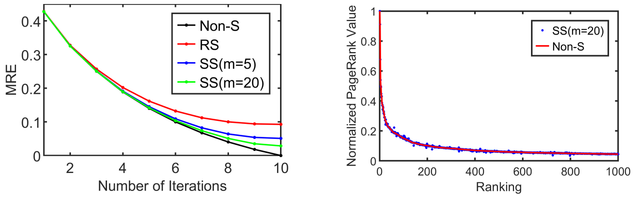


Fig. 2. The MAP results based on TBS at different sampling fractions for amazon0601 dataset.

Next, we evaluate the influence of the simplified VSN, that is, stratified sampling, on the accuracy of PageRank. We define the metric of algorithm accuracy as the mean relative error (MRE) between the sampled graph and the original graph, as shown in (19).

$$MRE(G) = \frac{1}{|V|}\left(\sum_{u \in V} \frac{|I_s(u) - I(u)|}{I(u)}\right) \qquad (19)$$

where $I_s(u)$ is the value of vertex u calculated after sampling, and $I(u)$ is the value of u calculated without sampling. In the experiment of PageRank, the number of iterations is set to 10. First, running the algorithm without sampling, and the vertex value in each iteration is calculated. In order to reduce the influence of randomness, the MRE value is obtained by average more than 100 different runs in each experiment.

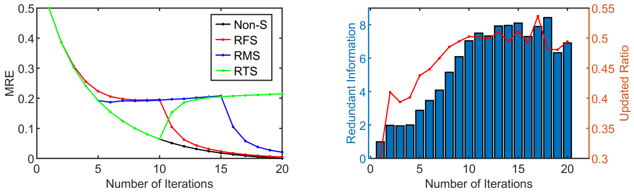


(a) MRE (b) Top 1000 Ranking

Fig. 3. The PageRank accuracy of different sampling approaches for soc-LiveJournal1 dataset (under the sampling fraction of 30%).

In Fig. 3 (a), we show the MRE results of soc-LiveJournal1 dataset under different sampling approaches, in which the sampling fraction is 30%. The non-sampling method can be regarded as the ground truth. We compare random sampling with two stratified sampling techniques. Random sampling refers to select 30% of neighbors for each vertex (if there is only one neighbor, no sampling).

For stratified sampling, we first set two thresholds m and n to divide vertices. When the in-degree of a vertex is greater than m and the ratio of in-degree to out-degree r_d is greater than n, the sampling is conducted according to r_d. Otherwise, it is converted to equal proportion sampling, ensuring that the total sampling ratio is still kept at 50%. In this experiment, we set n to 5. For two different stratified sampling methods, the parameter settings are m = 5, the equal proportion sampling rate is 50%, and m = 20, the equal proportion sampling rate is one third. It can be found that stratified sampling is better than random sampling, and the accuracy is related to m. In Fig. 3 (b), we show the normalized PageRank values of vertices in the case of non-sampling (Non-S) and stratified sampling (SS, m=20), which are very close under the sampling fraction of 30%. It indicates that the sampled graph can be used to get approximately accurate ranking results of vertex importance.



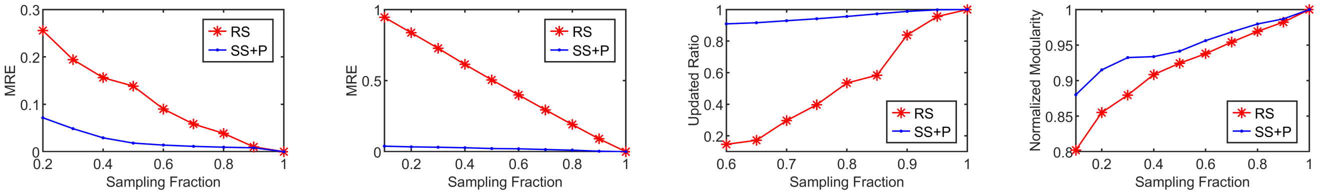
(a) PageRank (b) BFS

Fig. 4. The influence of different sampling positions on the algorithm accuracy under the sampling fraction of 30%. (The datasets used by PageRank is amazon0302, and BFS is com-Amazon.)

C. Results of the Hierarchical Optimization Scheme

In this subsection, in order to verify the effectiveness of a hierarchical optimization scheme, we evaluate the influence of sampling position on the accuracy of graph algorithms. We take PageRank and BFS as examples to illustrate two suitable sampling situations in Section III. C.

For PageRank algorithm, we set the number of iterations to 20. 10 iterations use the sampled graph, and the remaining



(a) PageRank (b) ALS (c) BFS (d) LPA

Fig. 5. The accuracy comparison of four algorithms (PageRank, ALS, BFS, LPA) under different sampling methods and sampling fractions. (The datasets used by the four algorithms are amazon0302, MovieLens, com-Amazon and ego-Facebook in order.)

10 use the original graph. We carry out experiments under three sampling positions, respectively, sampling for the first 10 iterations (RFS), sampling for the $11th$-$20th$ iterations (RMS) and sampling for the last 10 iterations (RTS). The experimental results are shown in Fig. 4 (a), from which we can find that RFS is the best one. As can be seen from Fig. 3 (a), PageRank updates the vertex value quickly at the beginning, and the speed gradually slows down. Therefore, it is more suitable for sampling in the previous iterations. In addition, when converting the sampled graph to the original graph, the convergence speed is obviously faster, and the error is rapidly reduced. The reason we analyze is that the sampling process gives each vertex a good initial value. Once the sampling is not carried out, it will quickly converge to the accurate value.

For BFS algorithm, we also set the number of iterations to 20, and do 20 experiments. Each experiment only samples in one iteration, and select the first to the $20th$ iterations in turn. We define the metric as the proportion P_v of the updated vertex numbers between the sampled graph and the original graph in an iteration. The accuracy becomes higher with the increase of P_v. The redundant information RI is defined as the ratio of the updated vertex numbers to the number of edges used in the original graph. For com-Amazon dataset, when the sampling fraction is 30%, the corresponding relationship between P_v and RI is shown in Fig. 4 (b). It can be found that the relationship between them is approximately positive, that is, the larger RI is, the larger P_v is. It shows that the application of sampling techniques to the iterative process with more redundant information can effectively improve the accuracy of the graph algorithm.

D. Generality Evaluation

We use four graph algorithms to evaluate the impact of different sampling approaches on the accuracy. As shown in Fig. 5, the combination of stratified sampling and hierarchical optimization strategy (SS+P) can significantly improve the accuracy of graph algorithms, especially in the case of small sampling proportion. Specifically, compared with random sampling (RS), SS+P can reduce the MRE of PageRank by about 17% (sampling ratio of 20%), and ALS by about 95% (sampling ratio of 10%). SS+P can increase the correctly updated values ratio of BFS by about 75% (sampling ratio of 60%), and the normalized modularity value [24] of LPA by about 8% (sampling ratio of 10%).

The results of using various sampling approaches in PageRank are shown in Table II. The second column in the Table II is the calculation time of the original graph. The total time of each sampling method includes the pre-processing time to obtain the sampled graph (considering that each sampling can be executed in parallel, we only count the time of sampling once) and the algorithm runtime. Compared with the three sampling approaches, random sampling (RS) can not guarantee the accuracy of the algorithm after reducing the calculation time. The accuracy of stratified sampling (SS) is better than random sampling, and the MRE of SS is about 5% in different datasets. In the case of stratified sampling combined with the hierarchical optimization scheme (SS+P), the sampling fraction can continue to decline, which results in similar or less calculation time than the other two methods. The MRE of SS+P can be reduced to less than 2% in different datasets.

TABLE II
ACCURACY COMPARISON OF PAGERANK ALGORITHM BASED ON THREE DIFFERENT SAMPLING METHODS. (TIME IS IN SECONDS, AND A TO F CORRESPOND TO THE FIRST TO SIXTH DATA SETS IN TABLE I.)

Dataset	Original	RS		SS		SS+P	
	Time	Time	MRE	Time	MRE	Time	MRE
A	16.817	9.687	0.1058	7.721	0.0471	7.272	**0.0179**
B	0.168	0.093	0.1376	0.078	0.0501	0.081	**0.0198**
C	9.792	5.377	0.0957	4.826	0.0426	4.498	**0.0172**
D	0.509	0.222	0.1518	0.206	0.0478	0.207	**0.0189**
E	0.237	0.146	0.1290	0.102	0.0457	0.091	**0.0136**
F	0.215	0.138	0.1109	0.093	0.0516	0.097	**0.0135**

[a] “RS” stands for random sampling, and “SS” for stratified sampling. ”SS+P” stands for stratified sampling combined with hierarchical optimization scheme.

V. CONCLUSION

In summary, this article introduces a general large-scale graph sampling framework: GraphSDH. By analyzing the variance of four common graph sampling techniques, We propose different sampling approaches. In order to further simplify the calculation and improve the accuracy of graph algorithms, we propose stratified sampling and hierarchical optimization scheme. Extensive experiments show that compared with the non-sampling case, GraphSDH can speed up PageRank by more than two times when the MRE of PageRank is less than 2%. In addition, GraphSDH can effectively improve the sampling accuracy of various graph algorithms, such as BFS, ALS and LPA. In our future work, we will build GraphSDH in a distributed system.

Acknowledgment

This work was supported by National Key Research and Development Program of China (No. 2018YFB0105000, 2017YFA0207600), National Natural Science Foundation of China (No. U19B2019, 61832007, 61622403, 61621091), China Postdoctoral Science Foundation (No. 2019M660641), Beijing National Research Center for Information Science and Technology (BNRist), and Beijing Innovation Center for Future Chips.

References

[1] Leão, J. C., Brandão, M. A., de Melo, P. O. V., & Laender, A. H. .“Who is really in my social circle?” in Journal of Internet Services and Applications, vol.9, no.1, pp. 20, 2018.

[2] Wang, Jizhe, et al. “Billion-scale commodity embedding for e-commerce recommendation in alibaba.” InSIGKDD, pp. 839-848, 2018.

[3] Eckhardt, Manon, et al. “Multiple routes to oncogenesis are promoted by the human papillomavirus–host protein network.” in Cancer Discovery, vol.8, no.11, pp. 1474-1489, 2018.

[4] P. Hu and W. C. Lau. “A survey and taxonomy of graph sampling.” CoRR, abs/1308.5865, 2013.

[5] Leskovec, Jure, and Christos Faloutsos. “Sampling from large graphs.” Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 631-636, 2006.

[6] Voudigari, E., Salamanos, N., Papageorgiou, T., & Yannakoudakis, E. J. “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, 2016.

[7] Riondato, Matteo, and Evgenios M. Kornaropoulos. “Fast approximation of betweenness centrality through sampling.” Data Mining and Knowledge Discovery 30(2), pp. 438-475, 2016.

[8] Mahmoody, Ahmad, Charalampos E. Tsourakakis, and Eli Upfal. “Scalable betweenness centrality maximization via sampling.” Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1765-1773, 2016.

[9] Wang, Tianyi, et al. “Understanding graph sampling algorithms for social network analysis.” 2011 31st international conference on distributed computing systems workshops. IEEE, pp. 123-128, 2011.

[10] Maiya, Arun S., and Tanya Y. Berger-Wolf. “Sampling community structure.” Proceedings of the 19th international conference on World wide web. pp. 701-710, 2010.

[11] Gao, R., Xu, H., Hu, P., & Lau, W. C.. “Accelerating graph mining algorithms via uniform random edge sampling.” 2016 IEEE International Conference on Communications (ICC). IEEE, pp. 1-6, 2016.

[12] Hamilton, Will, Zhitao Ying, and Jure Leskovec. “Inductive representation learning on large graphs.” Advances in neural information processing systems. pp. 1024-1034, 2017.

[13] Jie Chen, Tengfei Ma, and Cao Xiao. “Fastgcn: Fast learning with graph convolutional networks via importance sampling.” In International Conference on Learning Representations (ICLR), 2018b.

[14] Antunes, N., Bhamidi, S., Guo, T., Pipiras, V., & Wang, B . “Sampling-based estimation of in-degree distribution with applications to directed complex networks.” arXiv preprint arXiv:1810.01300 (2018).

[15] De Choudhury, Munmun, et al. “How does the data sampling strategy impact the discovery of information diffusion in social media?” Fourth International AAAI Conference on Weblogs and Social Media. 2010.

[16] B. Ribeiro and D. Towsley. “Estimating and sampling graphs with multidimensional random walks.” In Proceedings of the 10th ACM SIGCOMM conference on Internet measurement, pages 390–403. ACM, 2010.

[17] Li, R. H., Yu, J. X., Qin, L., Mao, R., & Jin, T. “On random walk based graph sampling.” 2015 IEEE 31st International Conference on Data Engineering. IEEE, pp. 927-938, 2015.

[18] Ribeiro, B., Wang, P., Murai, F., & Towsley, D. “Sampling directed graphs with random walks.” 2012 Proceedings IEEE INFOCOM. IEEE, pp. 1692-1700, 2012.

[19] Ahmed, Nesreen K., Jennifer Neville, and Ramana Kompella. "Network sampling designs for relational classification.” Sixth International AAAI Conference on Weblogs and Social Media. 2012.

[20] M. Kurant, A. Markopoulou and P. Thiran, “On the bias of BFS (Breadth First Search).” 2010 22nd International Teletraffic Congress (lTC 22), Amsterdam, 2010, pp. 1-8.

[21] https://snap.stanford.edu/data/

[22] C. Buckley and E. M. Voorhees. “Evaluating evaluation measure stability.” In Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval, pages 33–40. ACM, 2000.

[23] http://files.grouplens.org/datasets/movielens/

[24] G. Agarwal and D. Kempe. Modularity-maximizing graph communities via mathematical programming. The European Physical Journal BCondensed Matter and Complex Systems, vol. 66, no. 3, pp. 409–418, 2008.

[25] R. Wan and J. Cai, “Community Detection Using an Optimized Label Propagation Algorithm.” 2013 International Conference on Cloud Computing and Big Data, Fuzhou, 2013, pp. 360-365.

[26] D. Meira, J. Viterbo and F. Bernardini, “An Experimental Analysis on Scalable Implementations of the Alternating Least Squares Algorithm.” 2018 Federated Conference on Computer Science and Information Systems (FedCSIS), Poznan, 2018, pp. 351-359.