IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 4, 2021, accepted June 13, 2021, date of publication June 16, 2021, date of current version June 28, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3089699*

# Large Graph Sampling Algorithm for Frequent Subgraph Mining

**TIANYU ZHENG AND LI WANG**
School of Computer and Software Engineering, University of Science and Technology Liaoning, Anshan 114051, China
Corresponding author: Tianyu Zheng (ty_zheng@163.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 71472081.

**ABSTRACT** Large graph networks frequently appear in the latest applications. Their graph structures are very large, and the interaction among the vertices makes it difficult to split the structures into separate multiple structures, thus increasing the difficulty of frequent subgraph mining. The process of calculating subgraph isomorphism often requires many calculations. Reducing the unessential structure of the graph is an effective method to improve the efficiency. Therefore, we propose a large graph sampling algorithm (RASI) based on random areas selection sampling and incorporate graph induction techniques to reduce the structure of the original graph. In addition, we find that constraining the weight of the number of vertices in the entire graph is essential to reduce the calculation of subgraph isomorphisms. This parameter is constrained in the sampling process to improve the efficiency of frequent subgraph mining. Experimental results show that RASI has more stable performance and performs better than other sampling algorithms in non-connected graphs. Mining frequent subgraphs by graph sampling can significantly improve the efficiency of mining, and the number of subgraphs before and after sampling is very similar.

**INDEX TERMS** Graph sampling algorithm, frequent subgraph mining, random areas selection.

## I. INTRODUCTION

Graph mining is an important field of data mining. Graph structures perform better on complex and abstract data. In many practical applications, graph structures are used for modelling to express various complex logical relationships among problem objects. Frequent subgraph mining plays a vital role in the research of many data mining problems. The use of subgraph structure for graph classification, graph clustering, graph indexing, and graph network analysis has been extensively studied. For example, Deshpande *et al.* [1] classified compounds by frequent substructures; Acosta-Mendoza *et al.* [2] conducted frequent approximate subgraph mining on the atlas, and improved the image classification by identifying frequency patterns in the image collection; Yan *et al.* [3], [4] mined freqent patterns in closed graphs and further proposed to improve the query performance of the graph database by using the graph index based on the frequent subgraph structure; Rehman and Asghar [5] identified frequent patterns in social network data to extract interesting information to enrich the structure of its social platform and attract more users. When the scale of the graph data grows, it becomes more difficult to perform frequent subgraph mining tasks on large-scale graph data in memory. Mining tasks often cannot be performed due to insufficient memory when the frequency threshold is low. Therefore, many algorithms based on distributed platforms have been proposed, such as [6], [7], which can improve the memory performance through distributed computing. One of the bottlenecks of frequent subgraph mining is the calculation of the subgraph isomorphism, which has been proven to be an NP-complete problem [8]. Sampling the original graph or optimizing the original graph structure can reduce many subgraph isomorphism calculations, which improves the efficiency of frequent subgraph mining algorithms. Therefore, sampling technology is an effective method to solve the problem of frequent subgraph mining in a single large graph. We aim to perform frequent subgraph mining through sampled graphs to achieve a higher accuracy rate of the number of subgraphs while reducing the computational burden.

Graph sampling algorithms for large networks have been proposed, especially in social networks, such as [9]–[11]. However, most of the current sampling methods do not perform well on frequent subgraph mining tasks. By investigating various sampling methods, we found that by retaining

The associate editor coordinating the review of this manuscript and approving it for publication was Jerry Chun-Wei Lin.

as much of the regional substructure of the original graph as possible, the effect is better in the mining task. Based on this idea, we proposed a large graph sampling algorithm (Random Areas Selection and Graph Induction, RASI) based on random areas selection and subgraph induction technology, and combined it with the GraMi algorithm [12] for frequent subgraph mining tasks on a single large graph for experiments.

To compare the effects of different sampling algorithms in frequent subgraph mining tasks, we use different sampling graphs and original graphs to perform frequent subgraph mining. According to the experimental results, we evaluate the degree distribution, performance and accuracy of different sampling algorithms. The result shows that our proposed sampling algorithm has better robustness and more stable performance in connected graphs and non-connected graphs. Compared to other sampling algorithms, it also has a wider range of applicable scenarios.

Our main contributions are as follows:

1) A sampling method is used to better perform frequent subgraph mining tasks on a single large graph without using any distributed system.
2) The results of several sampling methods based on different sampling ideas on large graphs applied to frequent subgraph mining tasks are compared.
3) A new method to sample large graphs is proposed. It performs better and has more stable performance in frequent subgraph mining. In addition, we consider controlling the weight of the number of vertices in the graph to adjust the sampling effect. This approach has not been discussed in any sampling algorithms.
4) Multiple real public graph network datasets, including one connected graph and four non-connected graphs, are sampled.

## II. BACKGROUND

### A. RELEVANT CONCEPTS

#### 1) LABELLED GRAPH

Let graph

$$G = (V, E, l_V, l_E)$$

where $V$ is a set of vertices (or nodes), $E$ is a set of edges, they can be directed or undirected, $l_V$ is a set of vertices labels, and $l_E$ is a set of edges labels. $|V|$ is the number of vertices, and $|E|$ is the number of edges.

#### 2) THE WEIGHT OF THE NUMBER OF VERTICES

In graph $G$, the weight of the number of vertices is $W$, which is a dictionary set. It indicates the number of vertices $v$ in the entire graph.

$$W\{v_i\} = \sum_{k=0}^{|V|} (l_{v_k} = l_{v_i})$$

where $v \in V$, $i \in [0, |V|)$

#### 3) SUBGRAPH ISOMORPHISM [12]

Let $S = (V_S, E_S, L_S)$ be a subgraph of a graph $G$. A subgraph isomorphism of $S$ to $G$ is an injective function:

$$f : V_S \to V,$$

which satisfies

$$\begin{aligned} L_S(v) &= L(f(v)), v \in V_S \\ L_S(u, v) &= L(f(u), f(v)), (u, v) \in E_S \\ &\quad and\ (f(u), f(v)) \in E \end{aligned}$$

#### 4) FREQUENT SUBGRAPH [12]

Given a graph $G$ and a minimum support threshold $\tau$, the frequent subgraph mining problem is defined as finding all subgraphs $S$ in $G$ such that

$$s_G(S) \geq \tau$$

where $S$ is a frequent subgraph.

### B. RELATED SAMPLING ALGORITHMS

The current sampling algorithms can be roughly divided into three categories: random node selection, random edge selection and sampling by exploration. Most of the previous sampling methods focused on preserving the local or global attributes of the original graph, so that the basic properties of the original graph can be completely represented by sampling the graph as much as possible, and they are applied mostly to the sampling of connected graphs. However, in frequent subgraph mining, some large-scale network graphs are not fully connected, and include disconnected or loosely connected parts, such as wireless social networks [13]. Currently, many methods based on random walk sampling [9]–[11] fall into well-connected parts of the graph, and cannot complete the sampling of disconnected graphs.

**1. Random areas selection sampling** was proposed by Zou and Holder [14], and is different from the other three mainstream sampling methods: random vertex sampling, random edge sampling and random walk sampling. The random areas selection sampling method randomly selects a certain area in the graph each time.

Initially, the sample size $S$, number of areas $A$, and $N$ are provided as the set of nodes after sampling. First, a node is randomly selected as the starting node of a certain area and added it to $N$. The neighbouring nodes of all nodes in $N$ in the original graph $G$ are found and added to $N$. This process is repeated until the sample size $S$ is attained. Assuming that the number of areas $A = 3$, the sampling process of random areas selection is shown in Fig. 1.

The random areas selection algorithm selects a set of initial nodes, and the number of initial vertices is equal to the number of areas. Then, all neighbours of the nodes in the area are added to the vertex set ($V_S$). Then, it continuously expands the number of vertices according to the vertices in set ($V_S$). Even if the number of areas is very low, this process can

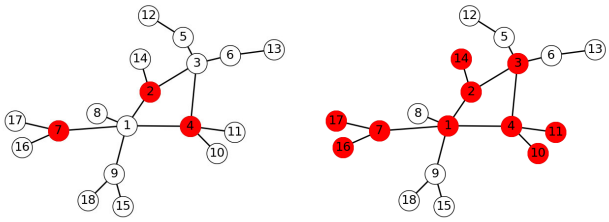


**FIGURE 1.** Demonstration of twice iteration processes of the random areas selection algorithm, where $A = 3$. (The value in the node in the graph represents the unique ID of the node, and does not represent the true value of the node.)

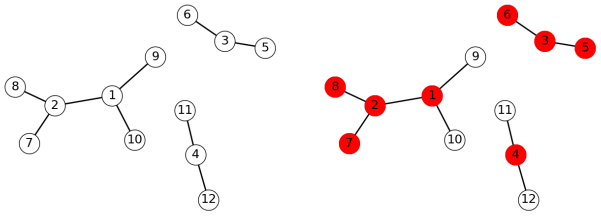


**FIGURE 2.** Demonstration of the random selection processes of the RASI algorithm (the value in the node in the figure is the unique ID of the node. Sampling rate $p = 0.7$).

successfully achieve sampling in the well-connected graph. However, this algorithm has many disadvantages:

1) When the number of vertices in the set ($V_S$) continues to increase, many nodes must be repeatedly traversed each time.
2) The result of the last traversal may exceed the sampling amount, or it may lose some vertices that have not reached the sampling amount.
3) In a disconnected graph, the number of regions is difficult to determine. The process of obtaining an accurate sampling amount is too complicated. If the number of areas is too small, sampling is not completed.

**2. Graph induction technology** was proposed by Ahmed *et al.* [15]. Since the connectivity of the sampled graph obtained based on random edge sampling is very low. Therefore, the graph induction is added to the algorithm to generate a representative subgraph taht matches the attributes of the original graph. This technology can improve the connectivity of the sampled subgraph and make the degree distribution similar to the original graph. Blagus *et al.* [16] proved that adding a graph induction step can improve the network sampling performance and be more similar to the original network density distribution. We have added it to the RASI sampling algorithm, which will be help us to obtain a sampling graph that is more similar to the original graph's degree distribution.

## III. RASI ALGORITHM IMPLEMENTATION

### A. RANDOM SELECTION PROCESS

In random areas selection sampling, the method of creating a sampling graph is to uniformly select a group of nodes $V_S$ at random and expand the areas through $V_S$ to obtain the sampling graph. It has achieved good results in connected graphs, but in frequent subgraph mining, there are objectively both connected graphs and non-connected graphs. We have changed the random selection method of random areas selection sampling, so that it can be better supported in disconnected graphs. In related work, we mentioned three different basic sampling methods. Because several new random walk methods retain the problem of getting stuck in non-connected graphs, they cannot produce a close match with the degree distribution attributes of the original graph. We did not consider it here. In addition, querying edges is much more difficult than querying vertices in many networks, such as Facebook [17] and MySpace [18], whose edges are not associated with the unique ID of the node. In contrast, random node selection sampling applies to a wider range and easily expands the area.

Random node selection sampling generates the induced graph by uniformly and randomly selecting nodes; thus, its sampling process does not fall into a certain local area, which gives us insight. Although it has been proven that it cannot maintain the power-law distribution [19], in practice, most of the original graphs are often not guaranteed to have an exact power-law [20]. Therefore, we do not consider its distribution here, and the key is to ensure a similar degree distribution before and after sampling. According to the conclusion, random node selection sampling tends to favour high-degree nodes, but this issue does not conflict with our ideas. For sampling in frequent subgraph mining, we usually select a higher sampling rate($p > 0.7$) to approach the degree distribution of the original graph, so as to achieve the purpose of mining more frequent subgraphs. Therefore, through these ideas, we use the idea of random node sampling to improve the original random selection process. We propose a new sampling algorithm, RASI, which is suitable for both types of graph networks. It changes the method of expanding the area in each iteration, to avoid the issue of getting stuck and to allow the sampling process to be satisfactorily completed in non-connected graphs.

The random selection process is shown in Fig. 2. Assuming that the RASI algorithm initially randomly selects 8 different nodes as the starting node based on the sampling rate, we do not need to worry about a node getting stuck in each iteration, because in the random selection process, each selection is independent of one another.

### B. OPTIMIZATION STRATEGY

In the previous section, The node selection process was reconsidered, which led to the obtaining of a sampled graph for the first time. In addition, for frequent subgraph mining, we further optimize the structure of the sampling graph. According to the concept, in frequent subgraph mining, the subgraph whose number greater than or equal to the

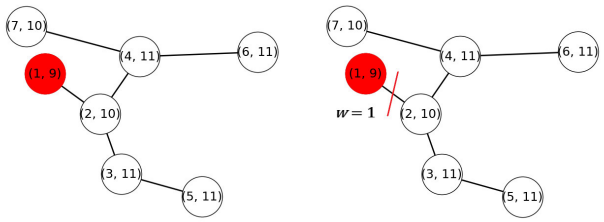



**FIGURE 3.** The vertices in the graph are represented by (a, b), where a is the unique ID of the vertex and b is the true value of the vertex.

frequency threshold $\tau$ is the frequent subgraph, which implies that the frequent appearance of sampling structure is more important. On the contrary, the part that rarely appears is redundant structure. However, it must also be calculated in the subgraph isomorphism. This requirement has a greater influence in a single large graph. We aim to reduce the redundant part as much as possible in the sampling method to improve the efficiency of the frequent subgraph mining algorithm.

During the node selection, the different nodes are visited easily. Such feature was used to consider the regional structure of the sampling nodes. We find that the weight of the number of vertices ($w$) affect the efficiency of frequent subgraph mining. A vertex with $w$ that is too small to become a branch in the frequent subgraph as usual and increases the time consumption of the algorithm. Therefore, we propose a novel optimization strategy: adding $w$ as a parameter to the sampling process and constraining its range, and using it to trim the vertex structure that does not meet the conditions to improve the sampling accuracy.

Fig. 3 shows a simplified process of frequent subgraph mining. We can see that when the frequency threshold $\tau = 3$, the frequent subgraphs are [(2,10), (3,11), (5,11)], [(2,10), (4,11), (6,11)] and [(7,10), (4,11), (6,11)], where the red vertex [(1,9)] is a redundant vertex. The weight of the number of all vertices are $W\{9\} = 1$, $W\{10\} = 2$, $W\{11\} = 4$. After the optimization strategy is introduced, when $w \leq 1$, the red vertex with too small $w$ must be trimmed. This effectively improve the execution efficiency of the algorithm in a single large picture. Therefore, the sampling method controls the weight of the number of vertices and plays a crucial role. In Section 4.5, we will evaluate $w$ in detail to verify the effectiveness of the optimization strategy.

### C. SPECIFIC SAMPLING PROCESS

Based on the idea of the random areas selection sampling algorithm, the RASI algorithm also uses random nodes as the vertices of different initial areas, but we do not specify the number of areas. This issue makes it difficult for the algorithm to execute smoothly in non-connected graphs. We aim to generate areas through random nodes and let these areas expand outward. This process is repeated until the number of nodes reaches the sampling amount.

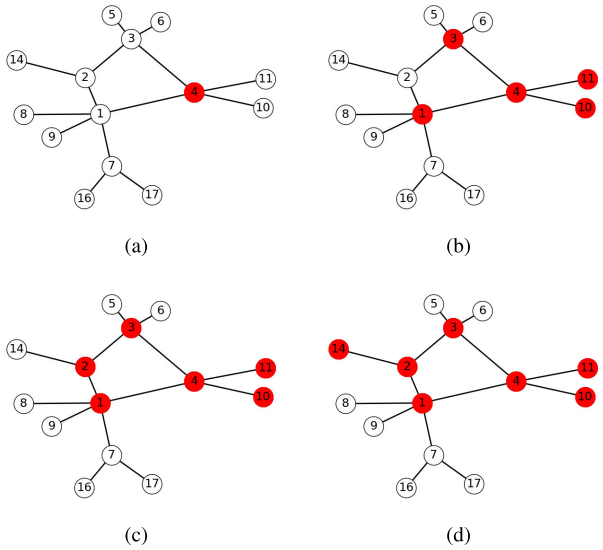



**FIGURE 4.** Demonstration of twice iteration processes of the RASI algorithm (the value in the node in the graph represents the unique ID of the node, and does not represent the true value of the node.).

Assuming that the starting node in the first iteration is 4, the neighbouring nodes are 1, 3, 10 and 11. In the second iteration, the starting node is 2, and the neighbouring nodes are 3 and 14 (Note: node 3 has been visited and will no longer be added to the node set of the sampling graph). The twice iteration processes of the RASI algorithm are shown in Fig. 4.

The pseudocode is shown in Algorithm 1. First, we randomly select a vertex as the starting node, add it to the sampling graph, and find all neighbouring vertices adjacent to the node. In the process of selecting the neighbouring vertices, we further consider the sampling according to the weight of the number of vertices in the original graph and try to exclude vertices from the sampling graph for which the weight of the number of vertices is too small. These vertices are redundant in frequent subgraph mining, which prolongs the search time of the subgraph isomorphism. The sampling process avoids such nodes as much as possible, and the filter parameter is $w$. After the conditions are satisfied, the connecting edge between the neighbouring vertex and the starting node is added to the sampling graph. This process is repeated, and the loop ends when the number of random vertices reaches the sample amount. Second, the graph induction step is performed. In the previous process, we extended the neighbouring areas through different starting vertices, but the neighbouring areas have low connectivity. The graph induction step can increase its connectivity and make its structure more complete. In line 20 of Algorithm 1, the difference between the edges of the original graph and the edges of the sampling graph is found. If edges are formed by the vertices of the sampling graph in the difference set, they are added to the sampling graph to improve the connectivity among the previous neighbouring areas and obtain the final sampling graph.

**Algorithm 1** RASI Algorithm

**Input:** original graph, $G$; sampling rate, $p$; the weight of the number of vertices, $w$
**Output:** sampled graph, $S_G$

$V = G.nodes$ ▷ original graph's vertices set
$W$ ▷ $W$ is generated when original graph data is read, it is the weight of the number of vertices dictionary set
$size = p \times |V|$
$V_R = random.sample(V, size)$ ▷ random vertices set
$i \leftarrow 0$
**while** $i < size$ **do**
  $v_i = V_R[i]$
  **if** $v_i \notin S_G.nodes$ **then**
    $S_G.addnode(v_i)$
  **end if**
  $Neighbors = G.neighbors(v_i)$
  **for** $n_i$ *in Neighbors* **do**
    **if** $W\{n_i\} > w$ and $(v_i, n_i) \notin S_G.edges$ **then**
      $S_G.edges \cup (v_i, n_i)$
    **end if**
  **end for**
  $i = i + 1$
**end while**
/*Graph induction step*/
$edges = G.edges - S_G.edges$ ▷ use difference sets to reduce the number of loops
**for** $(u, v)$ *in edges* **do**
  **if** $u \in S_G.nodes$ and $v \in S_G.nodes$ **then**
    $S_G.edges \cup (u, v)$
  **end if**
**end for**
**return** $S_G$

Two factors affect the sampling effect in the algorithm to control the generation of different sampling result graphs: the sampling rate $p$ and the weight of the number of vertices $w(> 0)$. These two factors satisfy a proportional relationship with $d$: $d = w/p$, where $d$ is the degree of similarity before and after sampling; a lower $d$ value, corresponds to more similarity. The lower weight of the number of vertices $w$ is and the higher sampling rate $p$ is, the lower the corresponding value of $d$ is. When the degree distributions are more similar before and after sampling, the number of frequent subgraphs found is also more similar, and this number can be adjusted according to the results of frequent subgraph mining. The settings of $w$ and $p$ in this paper are discussed in detail in the experimental evaluation section.

## IV. EXPERIMENTAL EVALUATION

We used a personal computer with an Intel Core i7-9700 CPU, 32 GB RAM and the Windows 10 operating system to perform all of the following experiments. All algorithms were implemented in Python version 3.7.9 with the networkx package [21].

**TABLE 1. Datasets information description.**

| Dataset | Nodes | Edges | Distinct node labels | Average degree | Connectivity |
|---|---|---|---|---|---|
| AIDS | 31,385 | 64,780 | 37 | 4 | False |
| Proteins-all | 43,471 | 162,088 | 3 | 7 | False |
| Deezer-HU | 47,538 | 222,887 | 20 | 9 | True |
| Tox21-AR-LBD | 152,273 | 312,288 | 47 | 4 | False |
| DD | 334,925 | 1,686,092 | 20 | 10 | False |

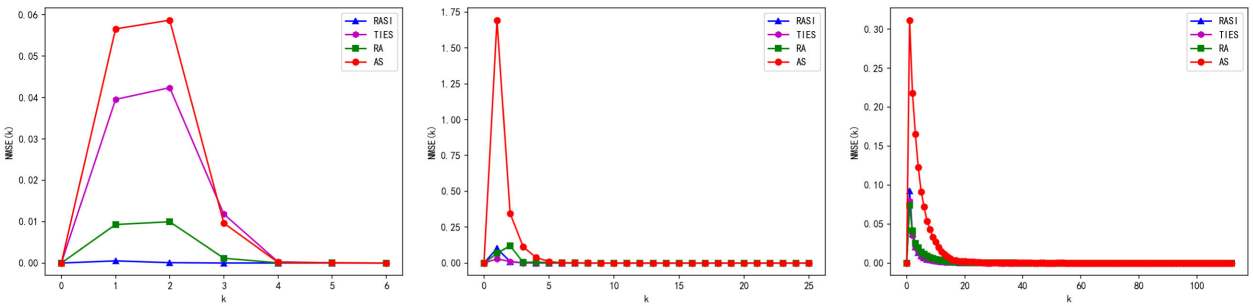
### A. DATASETS AND COMPARISON ALGORITHM

Many previous large graph networks contain only edge structures, and do not correspond to unique vertex IDs. Many frequent subgraph mining experiments [22] use randomly generated node labels. Here we also use this method in the Deezer-HU dataset, because it does not contain any node labels; other datasets contain node labels. The Deezer-HU comes from SNAP [23]. AIDS, Proteins-all, Tox21-AR-LBD and DD datasets are from the NETWORK REPOSITORY [24]. The information in the datasets is described in Table 1.

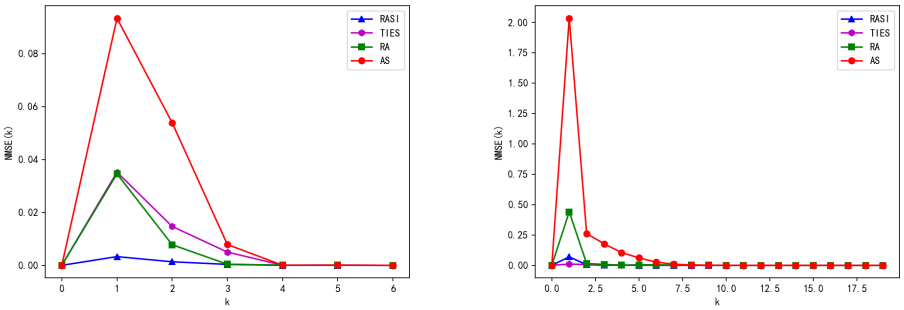
Due to different characteristics of different datasets, in the experiment, we must make the number of subgraphs in the sampling graph more similar to the number of subgraphs in the original graph. We must retain as many sampled graph structures as possible. Therefore, in the experiment, we set the sampling rate of all sampling algorithms to $p = 0.9$, because when the sampling rate is lower than 0.9, the number of subgraphs found is too small, which makes it impossible to effectively compare. In addition, in the RASI, RA and AS algorithms, we must adjust the weight of the number of vertices $w$, number of areas AreaNum($A$), random jump probability $p_{jump}$ and random jump steps $jump_{cost}$ according to different datasets, so that different algorithms simultaneously find more subgraphs with the highest efficiency under the setting of sampling rate $p = 0.9$. Therefore, we use different sampling algorithms to sample each dataset before the experiment, and use the generated sampling graph and original graph to perform frequent subgraph mining to obtain a set of parameter settings that are most similar to the number of subgraphs found in the original graph. The specific parameter settings of different datasets are shown in Table 2. The TIES algorithm only has one sampling rate parameter $p$, which is not listed in the table.

After obtaining the parameters of different datasets, we must also sample the same dataset multiple times to obtain a sampling graph with a relatively average sampling amount, and select the sampling graph with the most stable sampling performance and the highest accuracy. After obtaining the sampling graphs determined by different algorithms, we conduct frequent subgraph mining experiments for different frequency thresholds.

In addition to the two comparison algorithms in the related work, we select a sampling algorithm based on random walk,




(a) AIDS Dataset (b) Proteins-all Dataset (c) Deezer-HU Dataset




(d) Tox21-AR-LBD Dataset (e) DD Dataset

**FIGURE 5.** Distribution of *NMSE*(*k*) values at different degrees k. The lower the *NMSE* value, the more similar the degree distribution before and after sampling.

**TABLE 2.** Parameter settings of different algorithms.

| Dataset | RASI | RA | AS | All algorithms |
|---|---|---|---|---|
| AIDS | $w = 0$ | $A = 3500$ | | |
| Proteins-all | $w = 2$ | $A = 2050$ | | |
| Deezer-HU | $w = 0$ | $A = 2000$ | $p_{jump} = 0.01$<br>$jump_{cost} = 1$ | $p = 0.9$ |
| Tox21-AR-LBD | $w = 1$ | $A = 18000$ | | |
| DD | $w = 3$ | $A = 5500$ | | |

whose basic premise is different from that of all of the above algorithms, although it can also sample non-connected graphs. We added this algorithm to the experiment to enhance the effectiveness of the experiment. The introduction of the algorithm is as follows:

The AS [25] algorithm is a sampling algorithm based on random walk and was proposed by Jin *et al.*. This algorithm adds a random jump operation based on the basis of the MHRW algorithm [26], which avoids falling into a local area, unlike the original algorithm. It can be used for non-connected graphs.

### B. EVALUATION OF STRUCTURAL SIMILARITY

Regarding the normalized mean squared error (*NMSE*), *NMSE* is a commonly used method to evaluate the accuracy of the degree distribution of a graph. In this article, we use the *NMSE* standard to measure the robustness of the sampling method. This standard is used in many sampling methods [25], [27], [28], and defined as follows:

$$NMSE(k) = \frac{\sqrt{E[(\theta_k - \theta'_k)^2]}}{\theta'_k}$$

where, $\theta_k$ is the degree of the vertices in the original graph, and $\theta'_k$ is the degree of the vertices of the sampling graph. If the sampling algorithm exhibits a smaller *NMSE*, the degree distributions before and after sampling are more similar, which implies that the sampling method is more robust.

Fig. 5 shows the *NMSE* values of the sampled graphs generated by all algorithms under different datasets, we compare the structural similarity of the sampling graphs generated by all algorithms through *NMSE*. According to the results, at the same sampling rate, the *NMSE* of the RASI algorithm in the connected graph and non-connected graph is lower, and the distribution is relatively more stable, which indicates that RASI has a degree distribution more similar to that of the original graph, and it has a relatively complete original graph structure. The sampling amount of the RA algorithm and TIES algorithm are the most similar to that of the RASI algorithm. In the Proteins-all and Deezer-HU datasets, TIES performs even better. In the Deezer-HU dataset, the RA results are also similar to the RASI results. However, the gap

between them is certainly not large, and RASI is more robust in different types of datasets. The worst performance is the AS algorithm. It combines the ideas of the random jump and MHRW algorithm. Although the random jump algorithm can be added to handle non-connected graphs, the random walker visits only one of the neighbours of the vertex during the sampling process, which makes the AS to lose part of the neighbourhood structure.

### C. EVALUATION OF ACCURACY

For frequent subgraph mining, another evaluation index of the measured sampling algorithm is the accuracy. The formula to evaluate the accuracy is as follows:

$$Accuracy = |sG(S)|_{sampled} / |sG(S)|_{original}$$

where $|sG(S)|_{sampled}$ is the number of subgraphs whose appearance after sampling is exactly identical to that before sampling, $|sG(S)|_{original}$ is the number of subgraphs before sampling, and *Accuracy* is closer to 1, which implies that the sampling algorithm is more accurate and more suitable for frequent subgraph mining.

Table 3 shows the number of subgraphs found after performing frequent subgraph mining by different sampling algorithms, where the "No Sampling" column represents the number of subgraphs found in the original graph, and data closer to the original graph is shown in bold. The experimental results show that in all datasets, the samplimg graph of the RASI algorithm perform better than the other three algorithms in terms of the number of subgraphs and similarity to the number of original subgraphs. According to Fig. 5, in the Proteins-all and DD datasets, the *NMSE* value of the TIES algorithm performed better. However, beacuse it is based on random edge sampling so that it cannot maintain a large sampling diameter [28], it is not as accurate as the RASI algorithm. Therefore, under the same frequency threshold, RASI algorithm found more subgraphs. The RA algorithm did not consider the processing of non-connected graphs in the past, so it performed worse in the number of subgraphs in non-connected graph datasets. The AS algorithm performed the worst on the *NMSE* value of the low-degree interval, which caused it to find a small number of subgraphs in frequent subgraph mining.

Table 4 shows the accuracy of different algorithms. According to the results, we can see that the accuracy of the RASI algorithm is more accurate than the other three algorithms in all datasets. This shows that the improvement strategy we proposed is effective, and its performance in the two graph networks is more stable, and is more suitable for frequent subgraph mining.

### D. EVALUATION OF PERFORMANCE

In frequent subgraph mining, the process of subgraph isomorphism requires many calculations. Graph sampling reduces the structure of the graph and can effectively improve the calculation efficiency. We collect the calculation time of different sampling graphs under different frequency thresholds to further prove the effectiveness of the RASI algorithm. The experiment results are shown in Fig. 6, where the graph in the left column compares the calculation time of different sampling algorithms to perform frequent subgraph

**TABLE 3. Mining the number of subgraphs with different sampling algorithms.**

(a) AIDS dataset

| $\tau$ | No Sampling | RASI | RA | AS | TIES |
|---|---|---|---|---|---|
| 1000 | 54 | **53** | 40 | 36 | 49 |
| 1200 | 37 | **37** | 32 | 30 | 36 |
| 1400 | 32 | **32** | 26 | 24 | 30 |
| 1600 | 27 | **27** | 21 | 20 | **27** |
| 1800 | 24 | **24** | 19 | 17 | 21 |

(b) Proteins-all dataset

| $\tau$ | No Sampling | RASI | RA | AS | TIES |
|---|---|---|---|---|---|
| 9000 | 47 | **47** | 24 | 21 | 46 |
| 9100 | 43 | **43** | 24 | 21 | 42 |
| 9300 | 40 | **40** | 23 | 18 | **40** |
| 9500 | 38 | **38** | 22 | 18 | **38** |
| 9700 | 32 | **32** | 21 | 16 | 31 |

(c) Deezer-HU dataset

| $\tau$ | No Sampling | RASI | RA | AS | TIES |
|---|---|---|---|---|---|
| 220 | 8944 | **8936** | 8886 | 4696 | 8854 |
| 240 | 4199 | **4199** | 4197 | 4028 | 4191 |
| 280 | 4010 | **4010** | **4010** | 3999 | **4010** |
| 320 | 3831 | **3825** | 3821 | 2918 | **3825** |
| 340 | 3027 | **3009** | 3003 | 1425 | 3006 |

(d) Tox21-AR-LBD dataset

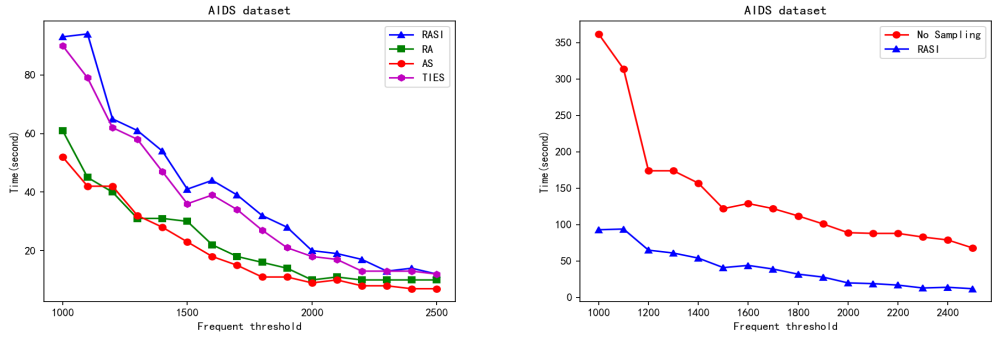
| $\tau$ | No Sampling | RASI | RA | AS | TIES |
|---|---|---|---|---|---|
| 13000 | 18 | **18** | 14 | 12 | 16 |
| 14000 | 14 | **14** | 12 | 12 | 13 |
| 15000 | 13 | **13** | 12 | 11 | 12 |
| 18000 | 12 | **12** | 10 | 8 | 11 |
| 19000 | 11 | **11** | 10 | 8 | 10 |

(e) DD dataset

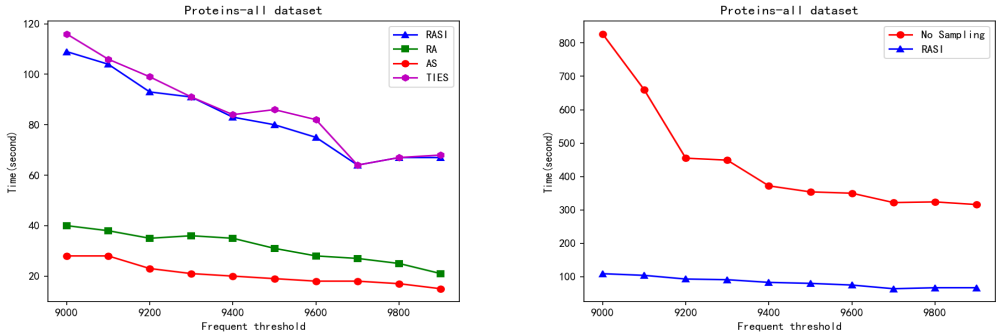
| $\tau$ | No Sampling | RASI | RA | AS | TIES |
|---|---|---|---|---|---|
| 2000 | 296 | **296** | 250 | 216 | 295 |
| 2100 | 272 | **272** | 221 | 198 | 271 |
| 2500 | 182 | **182** | 159 | 141 | 181 |
| 2800 | 149 | **149** | 126 | 118 | 148 |
| 3000 | 131 | **131** | 117 | 103 | **131** |

**TABLE 4. Accuracy of different sampling algorithms.**

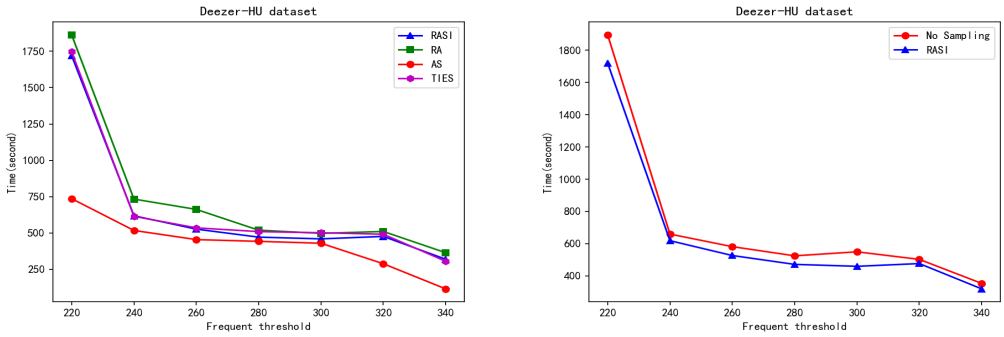
| Datasets | RASI | RA | AS | TIES |
|---|---|---|---|---|
| AIDS | **1** | 0.82 | 0.74 | 0.94 |
| Proteins-all | **1** | 0.59 | 0.47 | 0.99 |
| Deezer-HU | **0.99** | **0.99** | 0.81 | **0.99** |
| Tox21-AR-LBD | **1** | 0.89 | 0.76 | 0.94 |
| DD | **0.99** | 0.85 | 0.76 | **0.99** |



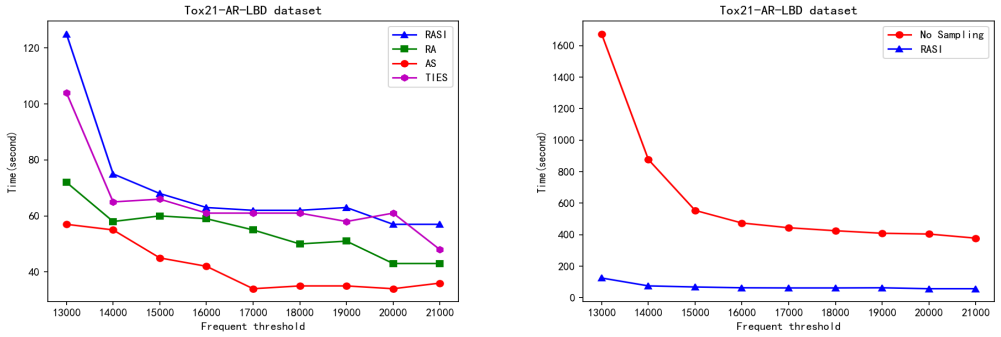

(a) AIDS datatset




(b) Proteins-all datatset

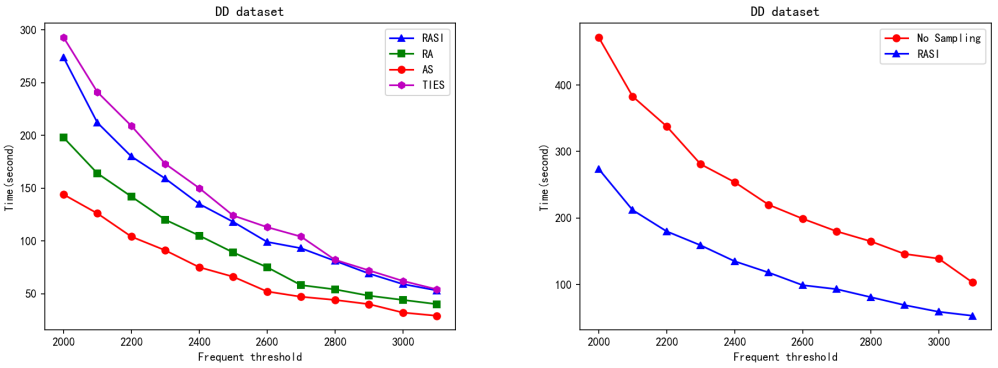



(c) Deezer-HU datatset




(d) Tox21-AR-LBD datatset

**FIGURE 6.** **(a) The left column compares the time for different algorithms to perform frequent subgraph mining. (b) The right column compares the sampling graph of the RASI algorithm and the time to perform frequent subgraph mining on the original graph.**




(e) DD datatset

**FIGURE 6.** *(Continued.)* **(a) The left column compares the time for different algorithms to perform frequent subgraph mining. (b) The right column compares the sampling graph of the RASI algorithm and the time to perform frequent subgraph mining on the original graph.**

mining, and the graph in the right column compares the calculation time of performing frequent subgraph mining between the sampling graph of the RASI algorithm and the original graph. The abscissa represents different frequency thresholds, and the ordinate is the calculation time in seconds.

We know that the more subgraphs are mined and the longer isomorphism detection process, the greater the corresponding algorithm execution time is increased. In Figs. 6(a) and 6(d), the mining time of the RASI algorithm is significantly longer. However, Table 3 shows that the RASI algorithm mines more subgraphs than the other algorithms and that the number of subgraphs mined by RASI is more similar to the original number of subgraphs. This is not in conflict with the purpose of setting the sampling rate $p = 0.9$ in the previous section. In Figs. 6(b), 6(c) and 6(e), the RASI algorithm finds more subgraphs consumes less time than the other two algorithms (RA and TIES), which proves that the RASI algorithm is more efficient.

In addition, in the right column graph of Fig. 6, we find that while ensuring high accuracy, we have effectively improved the efficiency of the frequent subgraph mining algorithm. This result is particularly obvious in the non-connected graph datasets. Due to the complexity of the vertex structure in the connected graph dataset Deezer-HU, in the frequency threshold range [220, 340], we found most of frequent subgraphs in this dataset while slightly improving the efficiency of the frequent subgraph mining algorithm.

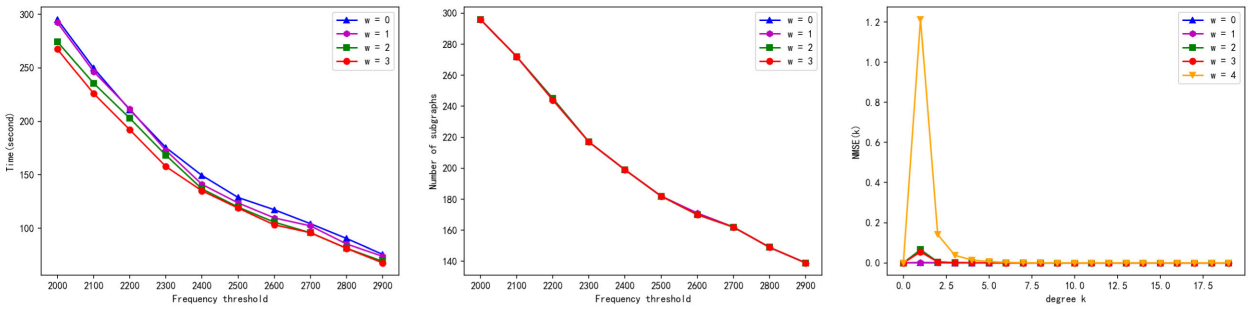
### E. EVALUATION OF THE IMPACT OF VERTEX NUMBER WEIGHT

In the previous section, we mentioned that different weights of the number of vertices affect the sampling effect of the sampled graph, which affects the performance of the frequent subgraph mining process. To the best of our knowledge, all previous sampling algorithms for frequent subgraph mining have not considered this factor. We take the DD dataset as an example and select the frequency threshold range of [2000, 2900] to further prove this result.

First, we control for the invariance of the number of the same subgraph, and set $w$ parameter as a variable. Here we set parameters $w$ to $w = [0, 1, 2, 3, 4]$ and generate different sampling graphs by different $w$ to perform frequent subgraph mining. This method is identical to that described in the performance evaluation section, and we do not repeat the details here. When $w = 4$, Fig. 7(c) shows that the degree distribution of the sampled graph had a large deviation from that of the original graph, which implies that when $w \leq 3$, the mining results of the sampling graph can be more similar to the original graph. Therefore, in 7(a) and 7(b), we did not further compare the case of $w = 4$.

The experimental results are shown in Fig. 7(a). We found that under the premise of identical numbers of subgraphs(7(b)), changing parameter $w$ can effectively shorten the time consumption of the frequent subgraph mining algorithm and the effect is more effective under the low-frequency threshold, which affirms our previous conclusion. A higher value of $w$ implies that the sampling process reduces a greater amount of structure. When one wants to mine more subgraphs, it is usually not easy to set the value of $w$ too high. Too much pruning also leads to the loss of more frequent structures. In our experiment, all datasets set the upper limit of $w$ to 3. When $w \geq 3$, the sampling graph mining results differs too much from the original graph mining result, which is inconsistent with the initial setting of our experiment.

However, when the frequency threshold $\tau = 2600$, according to Fig. 7(b), $|sG(S)_{w=1}| \neq |sG(S)_{w=2}|$, showing that the weight of the number of vertices $w$ does not always remain accurate. While reducing the graph structure by adjusting the $w$ parameter, it affects the number of some frequent subgraphs. Therefore, it is a better method to flexibly adjust parameter $w$ according to the frequency threshold. In other words, when the difference in the number of subgraphs before and after sampling is large (the number of frequent subgraphs found in the sampling graph is




(a) (b) (c)

**FIGURE 7.** DD dataset. (a) Comparison chart of frequent subgraph mining times under different frequency thresholds by setting different weights of the number of vertices. (b) Comparison chart of the number of frequent subgraphs found under different frequency thresholds by setting different number weight of vertices. (c) Comparision chart of *NMSE* values under different *w* values.

significantly less than that of the original graph), reduce the value of w to be more similar to the distribution properties of the original graph. For example, when $\tau = 2600$, setting $w = 1$ makes the number of subgraphs of the sampled graph closer to that of the original graph, and when setting $w = 3$, the mining speed becomes faster.

## V. CONCLUSION AND FUTURE WORKS

For frequent subgraph mining, we improve the random areas selection sampling method and propose a new sampling algorithm—RASI. During the sampling process, we used random vertices as the starting vertices and continuously expanded the vertex area, and we combined graph induction operations to increase the connectivity among the areas. At the same sampling rate, we evaluated the performance of different algorithms in many manners. In both connected graphs and non-connected graphs, the performance of the RASI algorithm was more robust than that of the other three algorithms. Especially in non-connected graphs, RASI demonstrated higher efficiency and greater accuracy in maintaining the number of subgraphs than the other three algorithms. In addition, we considered for the first time that the weight of the number of vertices was used to limit the structure of the sampled graph during the sampling process, and reduce unnecessary subgraph isomorphism calculations in frequent subgraph mining.

Inevitably, the RASI algorithm has limitations. After sampling connected graphs with a very complex graph structure, the efficiency of frequent subgraph mining algorithms is improved, although not significantly. Controlling the weight of the number of vertices may be ineffective in some special graphs, such as extremely sparse graphs.

Regarding this issue, to adjust the sampling method through the structural properties of the original graph, there may be various parameters to select, in addition to the weight of the number of vertices, and identifying these parameters is our next goal. We will continue to consider optimizing these issues.

## ACKNOWLEDGMENT

The authors are especially thankful to Mohammed Elseidy, who provided the GraMi source code.

## REFERENCES

[1] M. Deshpande, M. Kuramochi, and G. Karypis, "Frequent sub-structure-based approaches for classifying chemical compounds," in *Proc. 3rd IEEE Int. Conf. Data Mining*, Nov. 2003, pp. 35–42.
[2] N. Acosta-Mendoza, A. Gago-Alonso, and J. E. Medina-Pagola, "Frequent approximate subgraphs as features for graph-based image classification," *Knowl.-Based Syst.*, vol. 27, pp. 381–392, Mar. 2012.
[3] X. Yan and J. Han, "CloseGraph: Mining closed frequent graph patterns," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2003, pp. 286–295.
[4] X. Yan, P. S. Yu, and J. Han, "Graph indexing: A frequent structure-based approach," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, 2004, pp. 335–346.
[5] S. U. Rehman and S. Asghar, "Online social network trend discovery using frequent subgraph mining," *Social Netw. Anal. Mining*, vol. 10, no. 1, pp. 1–13, Dec. 2020.
[6] J. M.-T. Wu, G. Srivastava, M. Wei, U. Yun, and J. C.-W. Lin, "Fuzzy high-utility pattern mining in parallel and distributed Hadoop framework," *Inf. Sci.*, vol. 553, pp. 31–48, Apr. 2021.
[7] F. Qiao, X. Zhang, P. Li, Z. Ding, S. Jia, and H. Wang, "A parallel approach for frequent subgraph mining in a single large graph using spark," *Appl. Sci.*, vol. 8, no. 2, p. 230, Feb. 2018.
[8] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: Freeman, 1979.
[9] Y. Li, Z. Wu, S. Lin, H. Xie, M. Lv, Y. Xu, and J. C. S. Lui, "Walking with perception: Efficient random walk sampling via common neighbor awareness," in *Proc. IEEE 35th Int. Conf. Data Eng. (ICDE)*, Apr. 2019, pp. 962–973.
[10] Z. Zhou, N. Zhang, and G. Das, "Leveraging history for faster sampling of online social networks," 2015, *arXiv:1505.00079*. [Online]. Available: http://arxiv.org/abs/1505.00079
[11] R.-H. Li, J. X. Yu, L. Qin, R. Mao, and T. Jin, "On random walk based graph sampling," in *Proc. IEEE 31st Int. Conf. Data Eng.*, Apr. 2015, pp. 927–938.
[12] M. Elseidy, E. Abdelhamid, S. Skiadopoulos, and P. Kalnis, "GraMi: Frequent subgraph and pattern mining in a single large graph," *Proc. VLDB Endowment*, vol. 7, no. 7, pp. 517–528, Mar. 2014.
[13] N. Eagle, A. Pentland, and D. Lazer, "Inferring friendship network structure by using mobile phone data," *Proc. Nat. Acad. Sci. USA*, vol. 106, no. 36, pp. 15274–15278, Sep. 2009.
[14] R. Zou and L. B. Holder, "Frequent subgraph mining on a single large graph using sampling techniques," in *Proc. 8th Workshop Mining Learn. With Graphs (MLG)*, 2010, pp. 171–178.
[15] N. K. Ahmed, J. Neville, and R. Kompella, "Network sampling: From static to streaming graphs," *ACM Trans. Knowl. Discovery Data*, vol. 8, no. 2, pp. 1–56, Jun. 2014.

[16] N. Blagus, L. Šubelj, and M. Bajec, "Empirical comparison of network sampling: How to choose the most appropriate method?" *Phys. A, Stat. Mech. Appl.*, vol. 477, pp. 136–148, Jul. 2017.
[17] M. Gjoka, M. Kurant, C. T. Butts, and A. Markopoulou, "A walk in Facebook: Uniform sampling of users in online social networks," 2009, *arXiv:0906.0060*. [Online]. Available: http://arxiv.org/abs/0906.0060
[18] B. Ribeiro, W. Gauvin, B. Liu, and D. Towsley, "On MySpace account spans and double Pareto-like distribution of friends," in *Proc. INFOCOM IEEE Conf. Comput. Commun. Workshops*, Mar. 2010, pp. 1–6.
[19] M. P. H. Stumpf, C. Wiuf, and R. M. May, "Subnets of scale-free networks are not scale-free: Sampling properties of networks," *Proc. Nat. Acad. Sci. USA*, vol. 102, no. 12, pp. 4221–4224, Mar. 2005.
[20] J. Leskovec and C. Faloutsos, "Sampling from large graphs," in *Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2006, pp. 631–636.
[21] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proc. 7th Python Sci. Conf.*, G. Varoquaux, T. Vaught, and J. Millman, Eds. Pasadena, CA, USA, 2008, pp. 11–15.
[22] L. B. Q. Nguyen, B. Vo, N.-T. Le, V. Snasel, and I. Zelinka, "Fast and scalable algorithms for mining subgraphs in a single large graph," *Eng. Appl. Artif. Intell.*, vol. 90, Apr. 2020, Art. no. 103539.
[23] J. Leskovec and A. Krevl. (Jun. 2014). *SNAP Datasets: Stanford Large Network Dataset Collection*. [Online]. Available: http://snap.stanford.edu/data
[24] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. AAAI*, 2015, pp. 4292–4293. [Online]. Available: http://networkrepository.com
[25] L. Jin, Y. Chen, P. Hui, C. Ding, T. Wang, A. V. Vasilakos, B. Deng, and X. Li, "Albatross sampling: Robust and effective hybrid vertex sampling for social graphs," in *Proc. 3rd ACM Int. Workshop MobiArch (HotPlanet)*, 2011, pp. 11–16.
[26] C. Hübler, H.-P. Kriegel, K. Borgwardt, and Z. Ghahramani, "Metropolis algorithms for representative subgraph sampling," in *Proc. 8th IEEE Int. Conf. Data Mining*, Dec. 2008, pp. 283–292.
[27] B. Ribeiro and D. Towsley, "Estimating and sampling graphs with multidimensional random walks," in *Proc. 10th Annu. Conf. Internet Meas. (IMC)*, 2010, pp. 390–403.
[28] L. Longyang, D. Yihong, Y. Yuliang, C. Huahui, and Q. Jiangbo, "A sampling algorithm based on frequent edges in single large-scale graph under spark," *J. Comput. Res. Develop.*, vol. 54, no. 9, p. 1966, 2017.

**TIANYU ZHENG** received the B.S. degree in network engineering from the University of Science and Technology Liaoning, Anshan, China, in 2019, where he is currently pursuing the master's degree in computer application technology. His research interest includes graph mining.

**LI WANG** received the Ph.D. degree from Tianjin University, Tianjin, China, in 2004. She is currently a Professor with the Liaoning University of Science and Technology. She has authored or coauthored more than 50 publications in conferences and journals. Her research interests include data mining and big data analysis.