# A New Social Network Sampling Algorithm Based on Temperature Conduction Model

Xiaolin Du and Yunming Ye
Shenzhen Graduate School, Harbin Institute of Technology Shenzhen 518055, P.R. China
Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen 518055, P.R. China
Email: duxiaolinhitsz@gmail.com, yeyunming@hit.edu.cn

Yueping Li
Shenzhen Polytech, Shenzhen 518055, P.R.China
leeyueping@gmail.com

Xiaohui Huang
Shenzhen Graduate School, Harbin Institute of Technology Shenzhen 518055, P.R. China
Shenzhen Key Laboratory of Internet Information Collaboration, Shenzhen 518055, P.R. China
Email: XiaohuiHuang.hxh016@gmail.com

***Abstract*—A popular solution to dealing with large-scale social networks is to derive a representative sample from a social network. This sample is expected to represent the original social network well such that the sampled network can be used for simulations and analysis. In this paper, we propose a new social network sampling algorithm based on the Temperature Conduction model. Our sampling approach is able to effectively maintain the topological similarity between the sampled network and its original network. We have evaluated our algorithm on several well-known data sets. The experimental results show that our algorithm outperforms the state-of-the-art methods.**

***Index Terms*—sampling algorithm, temperature conduction, conduction boundary, topology structure**

## I. INTRODUCTION

Social networks, such as twitter, micro-blog, MSN, Facebook, co-citation relation, credit network, etc., appear everywhere in our modern lives. The modern science of networks has brought significant advances in our understanding of complex systems [1]. In research, social networks are usually represented by different types of graphs. Vertices represent entities, and edges represent interactions between pairs of entities. Some graph mining techniques, such as graph visualization techniques, graph structure analyzing techniques, etc., are then employed to assist social networks analysis. However, given a large graph with millions of vertices, it is very difficult to use typical graph mining approaches to handle the entire graph directly. An essential issue is to find certain methods to accelerate the graph mining process. A popular solution is to accomplish a sub-graph, which can represent the original graph effectively such that we are able to use this sub-graph for simulations and analysis. The accomplishment of a sub-graph relies on a graph sampling process. This process aims at selecting a set of vertices and edges in a way that the resulting sub-graph obeys some general characteristics of the original graph. In this paper, we focus on developing new methods in the context of graph sampling techniques.

Sampling in a large-scale graph usually encounters three questions [4]. What is a good sampling method? What is a good sample size? How do we measure the goodness of a single sample as well as the goodness of a whole sampling method? At present, the state-of-the-art sampling algorithms include: Random Node (RN) sampling, Random PageRank Node (RPN) sampling, Random Degree Node (RDN) sampling, Random Edge (RE) sampling, Random Walk (RW) sampling, Random Jump (RJ) sampling, Forest Fire (FF) sampling [2], and other sampling strategies, which will be briefly introduced in section II. In these algorithms, sample size is usually predefined by users so that they can get their expected sampled graphs. In a sampling process, maintaining similar properties between a sampled graph and its original graph is essential, because we can study the sampled graph, instead of its original graph, only when a sampled graph represents its original graph effectively. Another important issue is to evaluate whether a sampled graph and its original graph have similar properties. Likewise, the existing techniques that measure the between-graph similarity will be introduced in section II.

The rest of the paper is organized as follows: Section II presents the related works. Section III describes the proposed Temperature Conduction sampling algorithm. The experiment process and the results are presented in Section IV. Finally, Section V concludes the paper.

## II. RELATED WORKS

Currently, there have been several state-of-the-art graph sampling algorithms. Conceptually, we can split these existing algorithms into three groups [4]: methods


doi:10.4304/jsw.8.11.2688-2696

based on randomly selecting vertices, methods relying on randomly selecting edges, and exploration techniques that simulate random walks or virus propagation to find a representative sample of the vertices.

As a typical approach based on randomly selecting vertices, Random Node sampling (RN) algorithm starts by selecting a set of vertices randomly, and then a sampled graph is induced by the selected vertices. The process of Random PageRank Node sampling (RPN) lies in setting the probability of a vertex, which is selected into the sampled graph, to be proportional to its PageRank weight. The idea of Random Degree Node sampling (RDN) is that the probability of a vertex being selected is proportional to its degree.

Similarly to RN sampling, one can also select edges randomly. This process is called Random Edge (RE) sampling. We present three methods based on exploration techniques. Random Walk (RW) sampling starts at randomly picking a vertex, and then it simulates a random walk on the original graph. Random Jump (RJ) sampling is very similar to RW sampling. The only difference is that, under RJ sampling, we randomly jump to any vertex in a graph with probability c = 0.15. Forest Fire (FF) sampling [2] is a recursive process. First, randomly pick a seed vertex, and begin "burning" outgoing links and the corresponding vertex. If a link gets burned, the vertex at the other endpoint has a chance to burn its own links, and so on recursively.

Apart from above-mentioned methods, there are other simple sampling strategies. In particular, Krishnamurthy et al. [6] explored contraction-based methods and graph traversal based on depth and breadth first search. But none of them performed well over all.

The sampling algorithms enable us to utilize sub-graphs with a small- scale of vertices and edges. But, how can we evaluate the performances of these algorithms? In other words, how can we evaluate the similarity between a sampled graph and its original graph? At present, researchers have designed several evaluation measures. One strategy is to compute the similarity of the distributions of the sampled graph and its original graph to indicate their similarity. The following are representatives of existing evaluation techniques:

- The degree distribution: for every degree $d$, we count the number of vertices with degree $d$ [9];
- The distribution of sizes of weakly connected components: we count the number of weakly connected components with the same size;
- The distribution of the clustering coefficient: let vertex $v$ have $k$ neighbors, then at most $k*(k-1)/2$ edges can exist between them; let $C_v$ denote the fraction of these allowable edges that actually exist, the clustering coefficient is then defined as the average $C_v$ over all the vertices of degree $d$ [7];
- Hop-plot: the number $P(h)$ of reachable pairs of nodes at distance $h$ or less, where $h$ is the number of hops [10];
- The distribution of the first left singular vector of the graph adjacency matrix versus the rank ;
- The distribution of singular values of the graph adjacency matrix versus the rank: spectral properties of graphs often follow a heavy-tailed distribution [11].

Among these sampling algorithms and evaluation techniques, one important character of a graph, topological structure, is overlooked. Topological structure is capable of revealing the real topology and social relation of networks. A promising sample of a network should maintain the similar topological structure to its original network. A sampling algorithm should consider the topological structure maintenance between the original network and its sampled one. Our proposed algorithm is just this.

In this paper, we propose a sampling algorithm, which can formulize a sampled network with similar topological structure to its original network. We evaluate our algorithm with respects to some existing evaluation techniques on several well-known data sets. The experimental results demonstrate that our algorithm outperforms other competitive methods.

## III. Graph Sampling based on Temperature Conduction Model

We firstly introduce the terminologies that are frequently used in this paper. Given an initial relational graph $G=(V_G,E_G)$, $V_G$ represents the vertex set of $G$, and $E_G$ represents the edge set of $G$. Let $G_S=(V_S,E_S)$ be a sample of graph $G$, where $V_S$ represents the vertex set of $G_S$, and $E_S$ represents the edge set of $G_S$.

Our motivation is that, given an initial graph $G$, we are expected to sample the vertices and edges distributing globally in $G$ in order to maintain the topology of $G$. That is, here are some vertices embedding in almost every part of $G$. At the same time, the sampled graph $G_S$ also performs well on the existing evaluation techniques mentioned in section II.

### A. Temperature Conduction Model

In this section, we will present the Temperature Conduction (TC) sampling model, which is able to assure the similar topology structure between the original graph and its sampled graph.

First, we will introduce two important concepts in our model: "Hot Vertex" and "Temperature Conduction". Given a relational graph $G$, we pick a vertex $v$ in $G$, then add $v$ to the sampled graph $G_S$. Here, we denote this vertex $v$ as a "Hot Vertex". Once a vertex becomes a "Hot Vertex", a process of heat emission will start. The hot vertex will deliver its temperature to the nearby vertices connected directly or indirectly to this hot vertex. The temperature of a vertex around the hot one is measured by the distance from this vertex to the hot vertex and the total counts of paths connecting this vertex with the hot vertex. The shorter distance to the hot vertex, the higher "Temperature" value the vertex has. If there

are several hot vertices around a "not-hot" vertex, the temperature value of this vertex is a temperature value summation from all hot vertices. In addition, the temperature value is also affected by the counts of paths to the hot vertex. More paths can conduct more temperature value. In this paper we only consider the paths less than three steps, and we suppose if the distance from a vertex to the hot vertex is larger than three, this vertex will not take the temperature into account. Subsequently, we present the method to calculate the temperature value.

Once a vertex is a "Hot Vertex", which is denoted as $v_{hot}$, we firstly set the temperature value of the hot vertex at ten, and then the vertices around $v_{hot}$ can be conducted "Temperature" from $v_{hot}$. For a vertex $v$ with current temperature $T_v$, the conduction temperature denoted by $\Delta T_v$ can be given by:

$$\Delta T_v = \sum_{i=1}^{3} \frac{1}{i} Cp_i * \frac{T_{hot} - T_v}{Dis_{hot\_v}} \quad (1)$$

where $T_{hot}$ denotes the temperature value of $v_{hot}$, and $Cp_i$ denotes the number of paths with $i$ steps size between $v$ and $v_{hot}$. Here we only consider the paths within three steps for simplicity. $Dis_{hot\_v}$ is the distance between $v_{hot}$ and $v$. In graph theory, the distance between two vertices in a graph is the number of edges in a shortest path connecting them.

From (1), we can obtain some properties of $\Delta T_v$. First, $\Delta T_v$ is inversely proportional to $Dis_{hot\_v}$. The shorter distance to the hot vertex, the higher "Temperature" the vertex has. Second, $\Delta T_v$ is proportional to the difference of $T_{hot}$ and $T_v$. Third, $\Delta T_v$ is related to the counts of paths to $v_{hot}$ within three steps.

After computing the $\Delta T_v$ of each vertex around a hot vertex, we add $\Delta T_v$ to every $T_v$. The whole graph is then in the state of heat balance, that is, there is no any temperature conduction between vertices. Subsequently, we will heat another vertex, which will become a new "Hot Vertex".

To design the strategy of heating a vertex, we must firstly introduce another important concept: "Conduction Boundary". Conduction Boundary is a set of vertices. The vertices in the conduction boundary set must meet two conditions: first, vertices in the conduction boundary set are not hot; second, vertices in the conduction boundary set have at least one edge to some hot vertices.

The initial conduction boundary set is an empty set. A conduction boundary set maintains above-mentioned properties, when vertices in this set become hot vertices. Once one vertex becomes a hot vertex, we will perform two operations to the conduction boundary set: first, delete the hot vertex from the conduction boundary set; second, add the neighbors to this hot vertex (the neighbor vertices are not in the conduction boundary set, and they are not hot vertex). The heating strategy is to randomly choose a vertex with lower temperature in temperature conduction boundary. Fig. 1 shows an example of the temperature conduction boundary. In Fig. 1, the star-shaped vertices are hot vertices, and the triangle vertices are all in the conduction boundary set, because they all satisfy with the conditions of the conduction boundary set: not being a "Hot Vertex" and having at least one edge connected to some hot vertices.

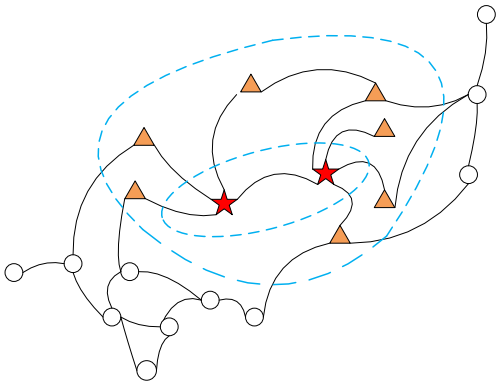

Figure 1. An Example of Conduction Boundary.

Our sampling model starts heating vertices in the original graph according to the proposed heating strategy. We then add the hot vertices to the sampled graph repeatedly until the number of vertices in the sampled graph reaches to certain threshold that we set at the beginning of the algorithm.

Specifically, our model is described as follows. The initial conduction boundary set is an empty set, and the initial temperature value of every vertex is 0. After setting the size of the sampled graph or the sampling percentage of the original graph, we randomly select a vertex as the starting vertex and heat it to be a hot vertex. Meanwhile, we add this hot vertex to the sampled graph. Then we update temperature values of the vertices around the hot vertex. We add neighbors of the hot vertex to the conduct boundary set. Our vertex heating strategy relies on choosing some vertices in the conduction boundary set to be hot vertices. Subsequently, we randomly select a vertex with the lowest temperature in the conduction boundary set and heat it. We update the temperature values and maintain the conduction boundary set. This is an iterative process until the number of vertices in the sampling graph is up to a user's requirement. After the sampling process of selecting vertices to the sampled graph, we add the induced edges to the sampled graph.

### *B. Algorithm Details*

For clarity, we summarize the entire algorithm as follows. Initially, we must set two parameters: sampling size $N$ (or sampling percentage $P$) and random percentage $r$. Their roles will be described below. Given a sampling size $N$, our algorithm starts at choosing a vertex $v_s$ randomly, then heat vertices according to the following process:

1. Heat $v_s$ to be a hot vertex and add $v_s$ to $G_S$. Update temperature values of vertices around $v_s$. Update the conduction boundary set $S_{cb}$;
2. Randomly choose a vertex $v$ in $S_{cb}$ with lower temperature value at random percentage $r$. Heat $v$ and update temperature values of vertices around $v$. Then update the conduction boundary set $S_{cb}$;
3. Execute step 2 recursively. As the process continues, the number of vertices in $G_S$ can be up to the predefined vertices amount. The recursively process stops;
4. Induce edges and add these edges to $G_S$.

Input: original graph $G$, sampling size $N$ (or sampling percentage $P$) and random percentage $r$.
Output: sampled graph $G_S$
T1: Input the sampling size: $N$;
T2: Randomly choose a vertex $v_s$ from $G$ as start vertex and heat $v_s$;
T3: Add $v_s$ to $G_S$;
T4: Compute conduction temperature value and update the conduction boundary set;
T5: While {the number of vertices in $G_S$ < $N$ }
T6: Randomly choose a vertex $v$ with low temperature value in conduction boundary set;
T7: Add $v$ to $G_S$;
T8: Compute conduction temperature values and update the conduction boundary set;
T9: End
T10: Add edges whose endpoints are in the vertex set of $G_S$ to $G_S$.

Figure 2. Sampling Algorithm Based on Temperature Conduction Model.

Thus, as is shown in Fig. 2, the heating process in the Temperature Conduction model begins with choosing a vertex $v_s$ randomly, spreads to the vertices in the conduction boundary set, and proceeds recursively until the number of vertices in sampled graph $G_S$ is up to our predefined amount. In this process, two important steps are temperature conduction and updating the conduction boundary set. The essential property of this model is that we randomly choose the vertex with low temperature value in the conduction boundary set.

We next explain the reason of this choosing strategy. Higher temperature value of one vertex indicates more hot vertices around the vertex or shorter distance to the hot vertices or even both. Our objective is to sample the vertices and edges distributing globally over $G$ in order to maintain the topology of $G$. Hence, choosing vertices with low temperature values can make the "heating" process not lie in a local part of graph $G$ but disperse all over graph $G$. The sampling process can be performed globally. That is, here are some vertices that are embedded in almost every part of $G$.

### C. Extensions

Our basic version of the Temperature Conduction model requires that the original graph is a connected graph. But real-life networks may not be fully connected. By extending this model to real-life networks, we introduce an extension method: we can perform the "heating" process in every connected component. That is, before "heating", we must add an extra step, which is to get the connected components of the original graph. Then we can run our algorithm in every connected component. Fig. 3 shows a graph with four connected components, and we do the "heating" process in four connected components.

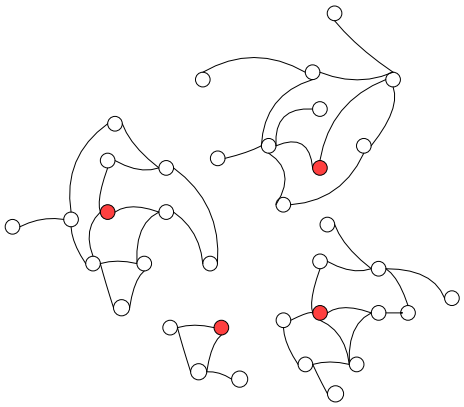

Figure 3. Temperature Conduction in Every Connected Component.

## IV. EXPERIMENT

In this section, we evaluate our proposed model on several real-life graphs. We have considered five common used data sets collected from the homepage of Newman [15]. As is shown in Fig. 4 and Table I, these data sets are email, power, hep-th, astro-ph and cond-mat. Data sets: hep-th, astro-ph and cond-mat, are not fully connected. So we firstly get their biggest weakly connected component and denote them hep-th_conect, astro-ph_connect and cond-mat_connect, respectively. Table I shows the detailed description of these five data sets and Fig. 4 shows the visualization layouts of five data sets.

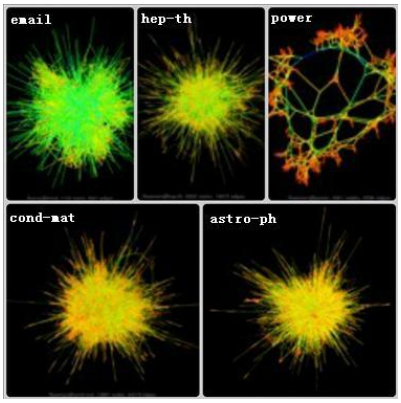


Figure 4. Original Data Sets Visualization Layout.

TABLE I.
DATA SETS DESCRIPTION

| Data set Name | Edge Count | Vertex Count | Diameter | Description |
|---|---|---|---|---|
| email | 1134 | 5452 | 8 | List of edges of the network of e-mail interchanges between members of the University Rovira i Virgili (Tarragona) [12]. |
| hep-th_conect | 13815 | 5835 | 19 | Weighted network of coauthor ships between scientists posting preprints on the High-Energy Theory E-Print Archive between Jan 1, 1995 and December 31, 1999[13]. |
| power | 6594 | 4941 | 46 | An undirected, unweighted network representing the topology of the Western States Power Grid of the United States [7]. |
| cond-mat_connect | 44619 | 36458 | 18 | Weighted network of coauthor ships between scientists posting preprints on the Condensed Matter E-Print Archive between Jan 1, 1995 and December 31, 1999[13]. |
| astro-ph_connect | 119652 | 14845 | 14 | Weighted network of coauthor ships between scientists posting preprints on the Astrophysics E-Print Archive between Jan 1, 1995 and December 31, 1999[13]. |

In statistics, the Kolmogorov-Smirnov test (K-S test) is a non-parametric test for the equality of continuous, one-dimensional probability distributions that can be used to compare a sample with a reference probability distribution (one-sample K-S test), or to compare two samples (two-sample K-S test) [14]. The smaller the testing value is, the larger the probability that two samples obey the same distribution is. Thus, we employ K-S test to measure the similarity of two distributions in our paper.

We summarize the results in Table II, Table III, Table IV and Table V. The results are obtained by averaging the K-S Test over 20 runs on each dataset. These four tables show the experimental results by different sampling percentages (P). In each column, we bold the best test value. In general, we can observe that our algorithm delivers most of the best test values.

TABLE II.
STATISTIC RESULTS ON 3 EVALUATION CRITERIA (P=0.05)

| P=.05 | Data Sets Name | email | hep-th_connect | power | cond-mat_connect | astro-ph_connect |
|---|---|---|---|---|---|---|
| RN | CD | 0.9856 | 0.9636 | 0.9922 | 0.8327 | 0.7573 |
| | Degree | 0.5314 | **0.5155** | 0.8674 | 0.4718 | 0.3783 |
| | Hop_plot | 0.0541 | 0.0706 | 0.0226 | 0.0392 | 0.0888 |
| RDN | CD | 0.9535 | 0.9611 | **0.9815** | 0.9888 | 0.8084 |
| | Degree | 0.4339 | 0.6769 | 0.8031 | 0.7690 | 0.3471 |
| | Hop_plot | 0.0431 | 0.0444 | 0.0275 | 0.0600 | 0.0615 |
| RPN | CD | 0.9384 | 0.9729 | 0.9862 | 0.9611 | 0.9875 |
| | Degree | 0.5728 | 0.6482 | 0.7949 | 0.4611 | 0.3366 |
| | Hop_plot | 0.0375 | 0.0311 | 0.0242 | 0.0277 | 0.0742 |
| RE | CD | 0.9887 | 0.8957 | 0.9852 | 0.8774 | 0.5427 |
| | Degree | **0.4457** | 0.6741 | 0.8532 | 0.7924 | 0.4920 |
| | Hop_plot | 0.0325 | 0.0676 | 0.0222 | 0.0353 | 0.0570 |
| RW | CD | 0.9989 | 0.9999 | 0.9996 | 0.9413 | 0.9707 |
| | Degree | 0.7068 | 0.6601 | **0.4480** | 0.4487 | 0.1068 |
| | Hop_plot | 0.0174 | 0.0018 | 0.0018 | 0.0026 | 0.0028 |
| RJ | CD | 0.9993 | 0.9882 | 0.9974 | 0.9956 | 0.9715 |
| | Degree | 0.7603 | 0.7879 | 0.4732 | 0.6383 | 0.2156 |
| | Hop_plot | 0.0043 | 3.70E-05 | 3.66E-09 | 1.79E-04 | 9.33E-04 |
| TC | CD | **0.8694** | **0.7619** | 1.0000 | **0.2712** | **0.1200** |
| | Degree | 0.6579 | 0.6094 | 0.6034 | **0.1203** | **0.0865** |
| | Hop_plot | **2.49E-04** | **8.62E-06** | **3.40E-11** | **8.94E-06** | **4.37E-04** |

TABLE III.
STATISTIC RESULTS ON 3 EVALUATION CRITERIA (P=0.1)

| P=0.1 | DataSets Name | email | hep-th_connect | power | cond-mat_connect | astro-ph_connect |
|---|---|---|---|---|---|---|
| RN | CD | 0.9615 | 0.9068 | 0.9835 | 0.8637 | 0.6091 |
| | Degree | 0.5267 | 0.7143 | 0.7814 | 0.6224 | 0.3441 |
| | Hop_plot | 0.0315 | 0.0682 | 0.0234 | 0.0453 | 0.0817 |
| RDN | CD | 0.9336 | 0.9735 | **0.9090** | 0.9864 | 0.8371 |
| | Degree | 0.4788 | 0.7131 | 0.8552 | 0.7284 | 0.2248 |
| | Hop_plot | 0.0178 | 0.0348 | 0.0454 | 0.0320 | 0.1058 |
| RPN | CD | 0.9742 | 0.9786 | 0.9924 | 0.9967 | 0.9990 |
| | Degree | 0.5149 | 0.6806 | 0.8197 | 0.4443 | 0.1665 |
| | Hop_plot | 0.0282 | 0.0535 | 0.0222 | 0.0291 | 0.0481 |
| RE | CD | 0.9855 | 0.9124 | 0.9615 | 0.9872 | 0.4808 |
| | Degree | **0.4131** | 0.7566 | 0.7431 | 0.7745 | 0.5942 |
| | Hop_plot | 0.0305 | 0.0823 | 0.0251 | 0.0231 | 0.0758 |
| RW | CD | 0.9999 | 0.9992 | 0.9985 | 0.9847 | 0.9627 |
| | Degree | 0.7638 | 0.9220 | **0.6290** | 0.3550 | 0.0412 |
| | Hop_plot | 0.0037 | **1.35E-05** | 0.0001 | 0.0020 | 1.40E-04 |
| RJ | CD | 0.9997 | 0.9877 | 0.9999 | 0.9971 | 0.9710 |
| | Degree | 0.7651 | 0.8642 | 0.6341 | 0.5107 | 0.0510 |
| | Hop_plot | 0.0037 | 1.88E-04 | **1.33E-08** | 0.0017 | 0.0021 |
| TC | CD | **0.8491** | **0.7654** | 0.9772 | **0.2844** | **0.0112** |
| | Degree | 0.5045 | **0.5475** | 0.7220 | **0.0970** | **8.44E-04** |
| | Hop_plot | **0.0018** | 1.27E-04 | 2.43E-07 | **2.19E-04** | **0.0012** |

TABLE IV.
STATISTIC RESULTS ON 3 EVALUATION CRITERIA (P=0.15)

| P=.15 | DataSets Name | email | hep-th_connect | power | cond-mat_connect | astro-ph_connect |
|---|---|---|---|---|---|---|
| RN | CD | 0.9563 | 0.9670 | 0.9687 | 0.7908 | 0.7010 |
| | Degree | **0.4138** | 0.5939 | 0.7566 | 0.5017 | 0.4176 |
| | Hop_plot | 0.0297 | 0.0776 | 0.2792 | 0.0559 | 0.0767 |
| RDN | CD | 0.9949 | 0.9514 | 0.9325 | 0.9960 | 0.9732 |
| | Degree | 0.4928 | 0.6905 | 0.8303 | 0.7593 | 0.3604 |
| | Hop_plot | 0.0202 | 0.0530 | 0.0300 | 0.0813 | 0.1012 |
| RPN | CD | 0.9840 | 0.9879 | 0.9347 | 0.9966 | 0.9997 |
| | Degree | 0.5263 | 0.6499 | **0.7368** | 0.6171 | 0.2342 |
| | Hop_plot | 0.0116 | 0.0283 | 0.0298 | 0.0415 | 0.0563 |
| RE | CD | 0.9997 | 0.8962 | **0.9131** | 0.9887 | 0.6302 |
| | Degree | 0.5198 | 0.7081 | 0.8531 | 0.8618 | 0.6071 |
| | Hop_plot | 0.0743 | 0.0404 | 0.0271 | 0.0628 | 0.0797 |
| RW | CD | 1.0000 | 0.9999 | 1.0000 | 0.9471 | 0.9412 |
| | Degree | 0.7606 | 0.8341 | 0.8050 | 0.3845 | 0.0146 |
| | Hop_plot | 0.0059 | 0.0035 | 5.07E-04 | 0.0013 | 0.0056 |
| RJ | CD | 0.9928 | 0.9991 | 0.9990 | 0.9951 | 0.9338 |
| | Degree | 0.8108 | 0.7642 | 0.8239 | 0.5879 | 0.1234 |
| | Hop_plot | 0.0011 | 0.0007 | 1.31E-07 | 8.77E-04 | **0.0055** |
| TC | CD | **0.8325** | **0.8917** | 0.9955 | **0.1588** | **0.0020** |
| | Degree | 0.6523 | **0.3446** | 0.8497 | **0.0934** | **0.0018** |
| | Hop_plot | **0.0017** | **3.31E-04** | **1.65E-10** | **1.91E-04** | 0.0058 |

TABLE V.
STATISTIC RESULTS ON 3 EVALUATION CRITERIA (P=0.2)

| P=0.2 | DataSets Name | email | hep-th_connect | power | cond-mat_connect | astro-ph_connect |
|---|---|---|---|---|---|---|
| RN | CD | 0.9809 | 0.9228 | 0.9483 | 0.9723 | 0.8506 |
| | Degree | 0.6842 | 0.5592 | 0.8098 | 0.5543 | 0.5392 |
| | Hop_plot | 0.0305 | 0.0563 | 0.0224 | 0.0360 | 0.0505 |
| RDN | CD | 0.9995 | 0.9903 | **0.9338** | 0.9689 | 0.9812 |
| | Degree | 0.3939 | 0.6979 | 0.7591 | 0.8980 | 0.5023 |
| | Hop_plot | 0.0347 | 0.0398 | 0.0474 | 0.0459 | 0.0958 |
| RPN | CD | 0.9693 | 0.9996 | 0.9993 | 0.9990 | 0.9995 |
| | Degree | 0.4279 | 0.7184 | 0.6325 | 0.4580 | 0.2375 |
| | Hop_plot | 0.0627 | 0.0524 | 0.0400 | 0.0407 | 0.1133 |
| RE | CD | 0.9999 | 0.9206 | 0.9548 | 0.9629 | 0.6886 |
| | Degree | 0.3416 | 0.5734 | 0.8738 | 0.8004 | 0.6884 |
| | Hop_plot | 0.6884 | 0.0587 | 0.0185 | 0.0430 | 0.0742 |
| RW | CD | 1.0000 | 1.0000 | 1.0000 | 0.9918 | 0.9222 |
| | Degree | 0.6150 | 0.6227 | **0.0716** | 0.3973 | 0.0133 |
| | Hop_plot | 0.0286 | 0.0055 | 2.59E-05 | 0.0040 | 0.0089 |
| RJ | CD | 1.0000 | 0.9999 | 0.9999 | 0.9980 | 0.9980 |
| | Degree | 0.6489 | 0.7884 | 0.7276 | 0.4850 | 0.1784 |
| | Hop_plot | **0.0014** | 0.0029 | 0.0021 | **0.0034** | 0.0078 |
| TC | CD | **0.5995** | **0.7277** | 0.9815 | **0.1163** | **0.0021** |
| | Degree | **0.3342** | **0.4857** | 0.8776 | **0.0316** | **0.0013** |
| | Hop_plot | 0.0017 | **0.0016** | **1.72E-09** | 0.0037 | **0.0058** |

For data sets "cond-mat_connect" and "astro-ph_connect", our algorithm produces almost the best test values for all sampling percentages. The sizes of these two data sets are larger than the other three data sets. It implies that our algorithm performs better when the scale of networks increases. For data set "power", our method cannot gain the best performance. Fig. 4 (power) shows the layout of "power". From Fig. 4 (power), we observe that the distribution of "power" differs from other data sets. The diameter of "power" data set is 46, which is larger than those of others. Also, vertices in "power" are not distributed radially around some centroids, but they are dispersed irregularly. This suggests that our algorithm may not work well for this type of data set. With the sampling percentage increases, the test values of our method in all five datasets tend to decrease, as more samples of original graph can represent the original structure better. From the analysis above, we can conclude our method is better than the others.

Fig. 5, Fig. 6, Fig. 7, Fig. 8 and Fig. 9 show the comparative layout results of 7 sampling algorithms on 5 data sets. The results suggest that algorithms base on randomly choosing vertices or edges can produce many isolated vertices in the sampled graphs and fail in maintaining similar topological structure between the original graph and its sampled graph, while algorithms based on the exploration strategy can maintain this similarity better. After comparing these visualization results to the origin graphs, we can see that our algorithm (TC) performs better than the algorithms (RW and RJ) based on exploration.

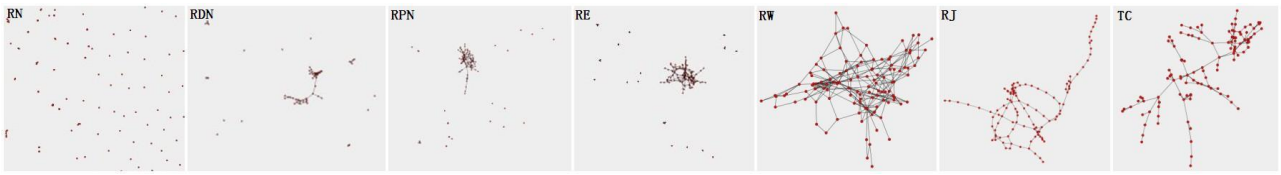


Figure 5. Visualization results of "email" data set (P=0.1).

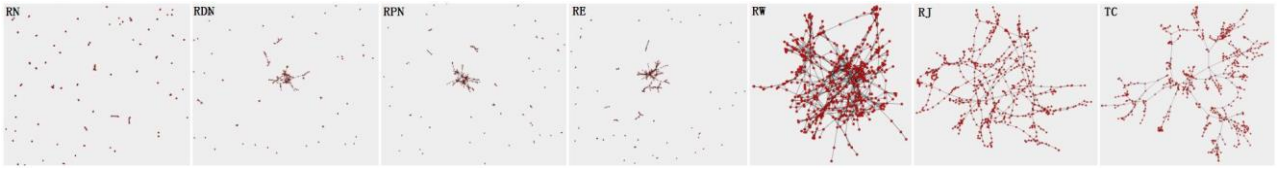


Figure 6. Visualization results of "hep-th" data set (P=0.1).

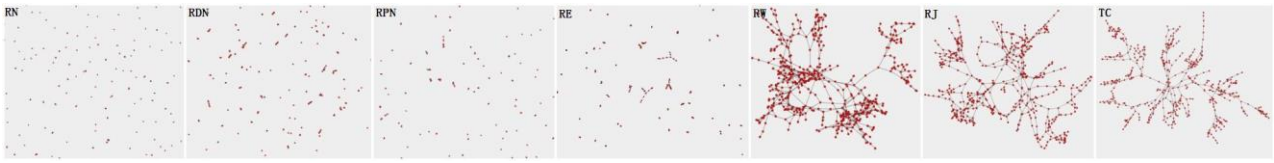


Figure 7. Visualization results of "power" data set (P=0.1).

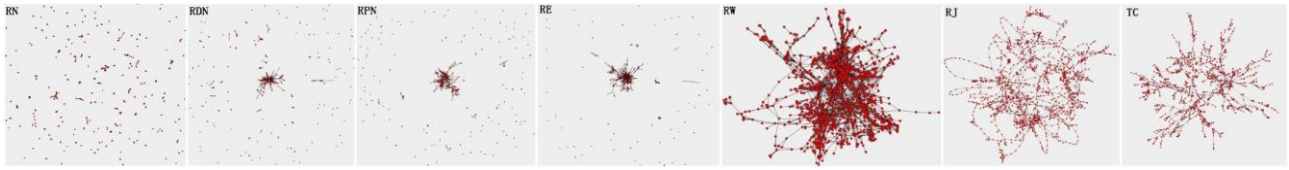


Figure 8. Visualization results of "cond-mat_connect" data set (P=0.1).

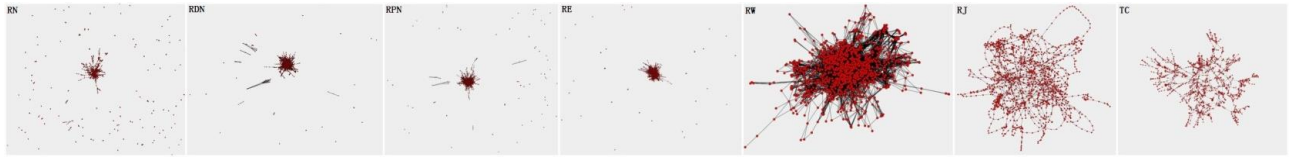


Figure 9. Visualization results of "astro-ph_connect" data set (P=0.1).

## V. CONCLUTION

It is important to generate a representative sampled graph, which enables us to accelerate the large-scale graph mining process. Despite many existing evaluations and algorithms with respect to graph sampling, only few studies work on the properties of topological similarity between the original graph and its sampled graph. This is exactly the focus of this work. In this paper, we propose a Temperature Conduction sampling algorithm. We provide extensive analysis and comparisons with the state-of-the-art methods. In particular, we perform a systematic evaluation of sampling algorithms by non-trivial statistical evaluation methods (the Kolmogorov-Smirnov Test). The comparative results suggest that our algorithm can effectively maintain the topological similarity between the sampled graph and its original graph.

## ACKNOWLEDGMENT

This research was supported in part by NSFC under Grant No.61272538, National Commonweal Technology R&D Program of AQSIQ China under Grant No.201310087, Shenzhen Science and Technology Program under Grant No.CXY201107010163A, and Shenzhen Strategic Emerging Industries Program under Grants No.JCYJ20120613135329670 and No. ZDSY20120613125016389.

## REFERENCES

[1] Fortunato S. "Community detection in graphs," *Physics Reports*, 2010, 486(3): 75-174.

[2] Leskovec J, Kleinberg J, Faloutsos C. "Graphs over time: densification laws, shrinking diameters and possible explanations," *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*. ACM, 2005: 177-187.

[3] Stumpf M P H, Wiuf C, May R M. "Subnets of scale-free networks are not scale-free: sampling properties of networks," *Proceedings of the National Academy of Sciences of the United States of America*, 2005, 102(12): 4221-4224.

[4] Leskovec J, Faloutsos C. "Sampling from large graphs," *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2006: 631-636.

[5] Zou R, Holder L B. "Frequent subgraph mining on a single large graph using sampling techniques," *Proceedings of the Eighth Workshop on Mining and Learning with Graphs*. ACM, 2010: 171-178.

[6] Krishnamurthy V, Faloutsos M, Chrobak M, et al. "Reducing large internet topologies for faster simulations," *Networking Technologies, Services, and Protocols; Performance of Computer and Communication Networks; Mobile and Wireless Communications Systems*. Springer Berlin Heidelberg, 2005: 328-341.

[7] Watts D J, Strogatz S H. "Collective dynamics of 'small-world'networks," *Nature*, 1998, 393(6684): 440-442.

[8] Bouttier J, Di Francesco P, Guitter E. "Geodesic distance

in planar graphs," *Nuclear Physics* B, 2003, 663(3): 535-567.
[9] Faloutsos M, Faloutsos P, Faloutsos C. "On power-law relationships of the internet topology," *ACM SIGCOMM Computer Communication Review*. ACM, 1999, 29(4): 251-262.
[10] Palmer C R, Gibbons P B, Faloutsos C. "ANF: A fast and scalable tool for data mining in massive graphs," *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2002: 81-90.
[11] Chakrabarti D, Zhan Y, Faloutsos C. "R-MAT: A recursive model for graph mining," *Computer Science Department*, 2004: 541.
[12] Guimera R, Danon L, Diaz-Guilera A, et al. "Self-similar community structure in a network of human interactions," *Physical review E*, 2003, 68(6): 065103.
[13] Newman M E J. "The structure of scientific collaboration networks," *Proceedings of the National Academy of Sciences*, 2001, 98(2): 404-409.
[14] http://en.wikipedia.org/wiki/KS_Test#cite_note-0 2012
[15] http://www-personal.umich.edu/~mejn/netdata/ 2012
[16] Yu W, Li S, Zhang Y, et al. "Mining Users Similarity of Interests in Web Community," *Journal of Computers*, 2011, 6(11): 2357-2364.
[17] Ma R, Deng G, Wang X. "A Cooperative and Heuristic Community Detecting Algorithm," *Journal of Computers*, 2012, 7(1): 135-140.
[18] Zhu S, Lin K, Zeng Z, et al. "A Sampling Method Based on Gauss Kernel Learning and the Expanding Research," *Journal of Computers*, 2012, 7(2): 547-554.
[19] Li Y, Du X, Ye Y, et al. "Stratified Sampling Large Relational Networks Using Topologically Divided Stratums," *Procedia Engineering*, 2011, 15: 3774-3779.

**Xiaolin Du** was born in HeiLongjiang Province, China in Jan. 1983, and received her Master Degree in Computer Science from Harbin Institute of Technology in 2009.

Currently, she is a PhD candidate in Shenzhen Graduate School, Harbin Institute of Technology. Her research interests involve data mining, data visualization and social network discovering.

**Yunming Ye** was born in China in Sep. 1976, and received his PhD degree in Computer Science from Shanghai Jiao Tong University in 2004.

Currently, he is a professor in Shenzhen Graduate School, Harbin Institute of Technology. His research interests include Web mining, Web Search, and social computing.

**Yueping Li** was born in Guangdong Province, China in Sep. 1980, and received his PhD in Computer Science from Sun Yat-sen University in 2008.

Currently, he is a post doctor in Shenzhen Graduate School, Harbin Institute of Technology. His research interests involve web mining, graph algorithm and optimization.

**Xiaohui Huang** is a PhD candidate in the Shenzhen Graduate School, Harbin Institute of Technology, China. His research interests are in the areas of data mining, topic detection and clustering algorithm.