Detecting Community Structure for Undirected Big Graphs Based on Random Walks

Xiaoming Liu[1], Yadong Zhou[1], Chengchen Hu[1], Xiaohong Guan[1,2], Junyuan Leng[1]

[1] MOE KLNNIS Lab, Xi'an Jiaotong University, P.R.China

[2] Center for Intelligent and Networked Systems and TNLIST Lab, TsinghuaUniversity, P.R.China

{xmliu, ydzhou, xhguan}@sei.xjtu.edu.cn, huc@ieee.org, junyuanleng@gmail.com

ABSTRACT

Community detection is a common problem in various types of big graphs. It is meaningful to understand the functions and dynamics of networks. The challenges of detecting community for big graphs include high computational cost, no prior information, etc.. In this work, we analyze the process of random walking in graphs, and find out that the weight of an edge gotten by processing the vertices visited by the walker could be an indicator to measure the closeness of vertex connection. Based on this idea, we propose a community detection algorithm for undirected big graphs which consists of three steps, including random walking using a single walker, weight calculating for edges and community detecting. Our algorithm is running in $O(n^2)$ without prior information. Experimental results show that our algorithm is capable of detecting the community structure and the overlapping parts of graphs in real-world effectively, and handling the challenges of community detection in big graph era.

Categories and Subject Descriptors

G.2.2 [Mathematics of Computing]: Graph Theory - Network problems; G.3 [Mathematics of Computing]: Probability and Statistics - Markov processes;

General Terms

Algorithms, Experimentation, Theory.

Keywords

Community detection; big graphs; random walking; overlapping parts.

1. INTRODUCTION

The theory of complex networks is employed for exploring the structure and function of social networks, the Internet and other complex systems in recent years. It is found that some common topological features occur widely in various types of complex networks, such as scale-free network, small world, and community structure, etc. [1]. Among these topological features, a community refers to the groups whose vertices are densely connected [2]. It reveals the internal organization of the vertices in networks [3]. Based on community detection results, recommendation systems [4], information diffusion models [5] and other researches on social networks can be further improved. However, it is challengeable to detect community structure for such a big graph effectively constrained by an acceptable time consumption and other demands. The main challenges include the high computational cost, no prior information of community number, etc..

Many outstanding works have been done in this field [1] but with some limitations in big graphs. In order to attain the best community detection results in quality, most of the existing algorithms need to measure modularity Q [6] in the process of each clustering or partitioning, which exacerbates the poor performance in computational cost. Even a few ones need prior information, such as the number of communities, etc., which is difficult to be obtained in graphs of real-world. To the best of our knowledge, the lowest time computational complexity of existing community detection algorithms [6, 8, 13] with good performance in quality and no need of prior information is larger than $O(n^2)$, where n is the number of vertices in networks.

To address these challenges, we propose an efficient community detection algorithm for undirected graphs based on random walks [7], with lower computational cost and no need of prior information. By analyzing the process of random walking in networks, we find that the weight of an edge obtained by processing the walks could detect the closeness of vertices. According to this feature, we use a random walker for gaining the information of the graph structure. The information can be helpful to compute the weight of each edge in networks in order to obtain the importance and closeness of the two vertices. Then we can merge the vertices connected by the higher weight edges into one community, and get the communities and analysis of the overlapping parts. During the community detecting, our algorithm does not need to measure the modularity Q any more. The computational complexity of the whole algorithm is reduce to $O(n^2)$ in worst case. Our algorithm surpasses previously proposed ones in running time and stands among the best ones concerning the quality of the community results, which are shown in comparison experiments.

The contributions of our work are listed below.

➢ We propose a novel algorithm which can discover the communities and overlapping parts based on a new measure which indicates the closeness of vertices in the framework of *Markov chain*.

➢ We reduce the computational complexity to $O(n^2)$ in worst case which is lower than the previously proposed ones [6, 8, 13]. Our algorithm can deal with the big graphs.

➢ We propose a new method to analyze the overlapping parts quantitatively and give the probability that which community the vertices belong to.

WWW'14 Companion, April 7–11, 2014, Seoul, Korea.
ACM 978-1-4503-2745-9/14/04.
http://dx.doi.org/10.1145/2567948.2580060

The rest of this paper is organized as follows. Section 2 gives some preliminaries and mathematical analyses of our idea. Section 3 presents the community detection algorithm based on random walks. Section 4 shows the experimental results. Section 5 summarizes some related work. Section 6 is about the conclusions and future work. Section 7 shows the acknowledgments

2. PRELIMINARIES AND NOTATIONS

Due to the heterogeneous structural feature and scale-free degree property [9] of complex networks, some vertices have large degree and other ones have relatively small degree. Based on preferential attachment [10], those vertices with large degree are connected closely. Thus, the structures of communities tend to be that some closely connected vertices with large degree are connected by the common vertices with small degree. For illustrating this, an example with three communities is given in Figure 1.

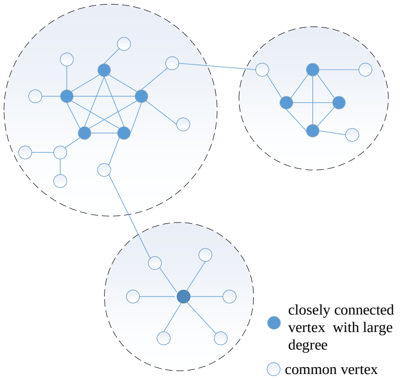

Figure 1. An example for community structure.

Random walks is a widely used method in sampling and estimating networks [7]. In this paper, it is also employed to gather the structure information of social networks. Based on random walks, we propose a solution for discovering community by detecting closely connected vertices with large degree in communities, and figuring out the connection relationships between them and other vertices. The preliminaries and mathematical analyses of our idea are introduced below.

A network is donated by an undirected and unweighted graph G = (V, E), where $V = \{v_1, v_2, \ldots, v_n\}$ is the vertex set and $E = \{e_{ij}, \ldots, e_{lf}\}$ is the set of edges. e_{ij} and e_{ji} present the same edge, i.e. $e_{ij} = e_{ji} = (v_i, v_j) = (v_j, v_i)$. Let $n = |V|$ and $m=|E|$. The degree of the vertex v_i is denoted by $d(i)$, where $1 \le i \le n$.

The *adjacency matrix* of graph G is denoted by a n×n matrix A. Namely, if v_i and v_j are connected, $A_{ij} = A_{\mathrm{ji}} = 1$, and $A_{ij} = A_{ji} = 0$ otherwise.

If the walker chooses a vertex v_i among the neighbors of the current vertex v_j randomly and uniformly, the process of random walks on graph G is a *Markov process* [11], and the *state space* is the vertex set of graph G. The *transition matrix* is defined as P, so

$$P = D^{-1}A \tag{1}$$

where D is the *diagonal matrix* of the vertex degrees and

$$D_{ij} = \begin{cases} d(i), & i = j \\ 0, & i \ne j \end{cases} \tag{2}$$

At each step, the *transition probability* from v_i to v_j is

$$P_{ij} = \frac{A_{ij}}{d(i)} \tag{3}$$

The *transition probability* from v_i to v_j through walking t steps randomly is denoted by $P_{ij}^{(t)}$. When the number of steps t tends towards infinity, the probability is [8]

$$\lim_{t\to\infty} P_{ij}^{(t)} = \frac{d(j)}{\sum_f d(f)} \tag{4}$$

The *transition probability* is independent with the start vertex v_i, and just depends on the degree of end vertex v_j.

The sequence of vertices visited by the walker is a *Markov chain* [11], and the stationary probability distributions of *Markov chain* follows

$$\ldots\ldots \pi P = \pi \tag{5}$$

where $\pi = \{\pi_1, \pi_2, \ldots, \pi_n\}$ is the *stationary probability distribution*. For the connected undirected graph G with finite vertices, the *stationary probability distribution* [11] of v_j follows

$$\pi_j = \lim_{t\to\infty} P_{ij}^{(t)} \tag{6}$$

When the probability reaches the *stationary distribution*, from (4), (5) and (6), the probability of traversing any edge $e = (v_i, v_j)$ of G is

$$p(e) = \pi_j \bullet \frac{1}{d(j)} + \pi_i \bullet \frac{1}{d(i)} = \frac{2}{\sum_f d(f)} \tag{7}$$

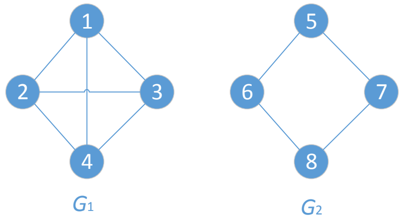

Figure 2. Two subgraphs are constructed for analysis, and the vertices in complete subgraph G_1 are more closely connected than the vertices in cycle subgraph G_2.

Suppose that G_1 and G_2 in Figure 2 are two subgraphs of graph G. In order to visit each vertex at least once, the number of the steps is large enough to make random walks reach *stationary distribution*. When the random walks reach the s*tationary distribution*, from (7), it is shown that any edge is traversed in equal probability, so the number of times of each edge traversed by the walker in G_1 and G_2 would be nearly the same. Thus, the numbers of times of the edges traversed by the walker are not capable of indicating the closeness of vertex connection.

Faced with the problems above, we analyze the process of random walking and propose a new measure to indicate the closeness between vertices below.

After random walks reaching *stationary distribution*, the probability that a path consisting of s edges is traversed by a walker is $(p(e))^s$, known from (7). In a specific situation in G_1 and G_2, the paths consisting of three edges and four different vertices are considered. There are 24 different such special paths in G_1, but 8 different ones in G_2. By making use of the different numbers of paths in subgraphs, we propose a new measure to indicate the closeness of vertices.

The main idea is that both the edges traversed by the walker and other edges existing between each two vertices of the path are computed. E.g. a path is $\{v_1, v_2, v_3, v_4\}$ in G_1, and then not only the edges e_{12}, e_{23}, e_{34} traversed by the walker but also the other edges e_{14}, e_{13}, e_{24} which exist between each two vertices of the path are computed. In this case, if one of such special paths is traversed by the walker, all of the edges in subgroups are computed. In another word, the probability of computing an edge is linear with the number of such special paths in the subgraphs.

The analysis shows that, in such special paths described above, the probability of computing an edge in G_1 is $24(p(e))^3$, but that in G_2 is $8(p(e))^3$. The probability of computing an edge obtained in G_1 is larger than that in G_2, because there exist more different such special paths in G_1 whose vertices are more closely connected.

Generally, the number of such special paths in subgraphs is computed, i.e. the paths consists of n_0-1 edges and all the n_0 different vertices in subgraphs. This computation is given in two cases. The number of such special paths in complete subgraphs (W_1) whose vertices are most closely connected, and cycle subgraphs (W_2) whose vertices are connected nearly most sparsely are given. The examples of these two types of subgraphs are shown in Figure 2. Here $W_1 = n_0!$ and $W_2 = 2n_0$.

$$W_1/W_2 = (n_0 - 1)!/2 \tag{8}$$

From (8), it is shown that the number of such paths in the subgraphs with closely connected vertices is much larger than that of the subgraphs with loosely connected vertices. After random walks reaching *stationary distribution*, the probability that an edge in complete subgraphs is computed is $W_1(p(e))^{n_0-1}$ and that in cycle subgraphs is $W_2(p(e))^{n_0-1}$. When the number of steps t is large enough, the number of times of each edge traversed by the walker is about $tW_1(p(e))^{n_0-1}$ in complete subgraphs, and it is about $tW_2(p(e))^{n_0-1}$ in cycle subgraphs. Known from (8), the huge gap of the times of the edges computed between closely connected subgraphs and loosely connected subgraphs are able to distinguish the closeness of vertex connection in graph G.

Thus, the times of the edge computed could be a new measure to indicate the closeness of vertex connection. We transform the graph into weighted graph based on the times of the edge computed. According to the nature of community, the weights of edges connecting the vertices from a same community should be larger than edges connecting vertices belonging to different communities. Therefore, through sorting the edges by their weights in descending order, the vertices connected by the edges in the front part of the sorted list are more likely to be within a community. Based on the sorted edges, the vertices are assinged to commnuties.

3. THE ALGORITHM

The framework of our algorithm is shown in Figure 3. In the algorithm, with the input of the graph data, random walks with a single walker is employed firstly, and a sequence of vertices visited by the walker is obtained. Then we calculate the weights for edges with k-vertices splitting strategy and sort them in descending order. It is the key point to detect the closely connected vertices with large degree in communities, and figure out the connection relationships between them and other common vertices. The vertex merging algorithm and overlapping analyzing algorithm are applied to the sorted weighted edges, and the detecting results of communities and overlapping parts are output respectively.

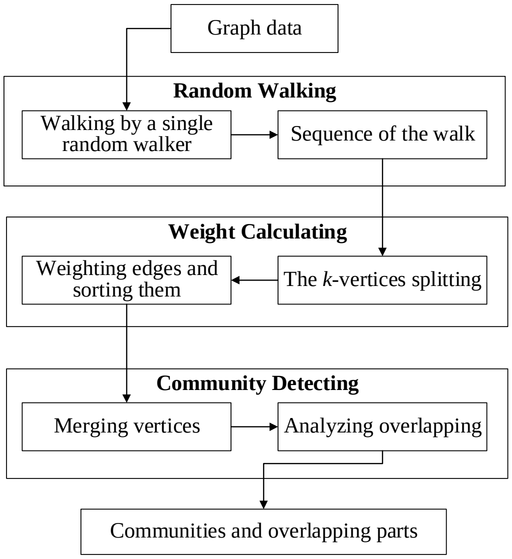

Figure 3. The framework of our community detection algorithm.

3.1 Random Walks with a Single Random Walker

Random walks employed in this paper begins with a single walker. The walker starts from an initial vertex v_0 selected randomly and uniformly from vertex set V. At each step, the probability of selecting any neighbor of the current vertex v_c as the destination vertex is $1/d(v_c)$, as shown in (3). The walker stops when the number of steps reaches the budget. Since the budget can be taken as n^2 according to empirical study, the time complexity of this part of the algorithm is $O(n^2)$.

3.2 Weight Calculation for Edges

The sequence of vertices visited by the walker is devoted by VS, and

$VS = \{v_a, v_b, \ldots, v_q | v_a \in V \wedge v_b \in V \wedge \ldots \wedge v_q \in V\}$.

The length of sequence VS is n^2. To get the subgraphs of graph G, VS is processed by k-vertices splitting strategy as the example in Table 1, and a new sequence which is denoted by VSS is gotten $VSS = \{\{v_a,...,v_f\},...,\{v_c,...v_q\}\}$

and $\{v_a,...,v_f\} \subset VS,...,\{v_c,...,v_q\} \subset VS$.

Thus, every adjacent k vertices in sequence VS are gathered into a sequence, and the vertices and the edges between the vertices form a subgraph of graph G. The edges between each pair of the k vertices are recorded and the number of records of the edge is denoted by Ne. An example of the process of edge recording is shown in Figure 4. From the analyses mentioned in section 2, it is concluded that the edges between closely connected vertices would be recorded much more times than those between loosely connected vertices. Finally, the edges are weighted by the number of times of being recorded, and then sorted by the weights. The sequence of the sorted edges are defined as

$SE = \{e_{df},...,e_{sh} \,|\, e_{df} \subset E \wedge,..., \wedge e_{df} \subset E\}$

Table 1. Examples for *k*-vertices splitting strategy

Sequence	Examples
VS	{1,2,4,3,2,1,4,5,6,7,5,6}
VSS with 3-vertices splitting strategy	{{1,2,4},{3,2,1},{4,5,6},{7,5,6}}

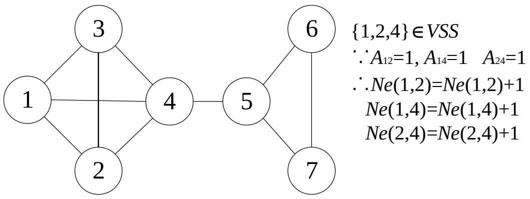
3
6
1
4
5
2
7
$\{1,2,4\} \in VSS$
$\because A_{12}=1, A_{14}=1 \quad A_{24}=1$
$\therefore Ne(1,2)=Ne(1,2)+1$
$Ne(1,4)=Ne(1,4)+1$
$Ne(2,4)=Ne(2,4)+1$

Figure 4. An example of edge recording.

The complexity of weight calculation for edges is $O(kn^2 + m\log m)$. Since the graphs of social networks are sparse graph [1], m and n are in the same order of magnitude. k is very small compared to n. So the time complexity is similar to $O(n^2)$.

3.3 Community Detection and Overlapping Analysis

Since the connections in communities are dense but between them are sparse, the edges in community would get larger weights. Thus, the pair of vertices of the edge with larger weight are more likely to belong to the same community. Especially, in the sorted edge sequence *SE*, the high-ranking edges and the related vertices compose the subgraphs with closely connected vertices in communities. Besides, the different communities should be connected by the low-ranking edges.

Based on the analysis above, we propose a community detection algorithm, in which the vertices are assigned into communities according to the sorted edge sequence *SE*. During the algorithm, the two vertices of the edge with the largest weight are taken as vertices of the initial community C_0. For the next edge in the sequence, if one of the two vertices is in C_0, another vertex would be assigned into C_0; if any one of the two vertices is not in C_0, both of them are assigned to a new community C_1. In accordance with the rules described above, the edges in *SE* would be processed one by one. Particularly, if the two vertices of an edge have been assigned to different communities, e.g. $e_{gh}=(v_g, v_h)$, $v_g \in C_i$ and $v_h \in C_j$, the edge and the two vertices are taken as the overlapping parts of the two communities.

In social networks, overlapping phenomenon exists among most communities, for the reason that people usually plays different roles, or has different hobbies etc.. Therefore, it is important to detect and analyze the overlapping parts. In this work, the probability that the vertex of overlapping parts belongs to certain community is given below.

The number of neighbors of vertex v in community C is donated by $Nb(v,C)$. If v_d belongs to the overlapping part of C_r and C_u, the probability that v_d belongs to C_r is defined as

$$p(v_d \in Cr) = \frac{Nb(v_d, C_r)}{Nb(v_d, C_r) + Nb(v_d, C_u)} \qquad (9)$$

The time complexity of this part is $O(mn - n^2/2)$ in the worst situation.

In this section, we present our algorithm whose time complexity is nearly $O(n^2)$ in details. Our results do not depend on maximizing Q but the nature of community, which is benefit for reducing the time complexity.

4. EXPERIMENTAL RESULTS

For evaluating the performance of our algorithm, the experiments are based on the data of five real social networks, including Zachary's karate club network, the bottlenose dolphins of Doubtful Sound network, the college football match network, the collaboration network of scientists, and friend relationship of members in Facebook. The algorithms compared with our algorithm are two classical algorithms, including Spectral bisection [12], Girvan Newman [13], and another algorithm using random walks with different idea, i.e. Walktrap [8]. Our experiments are implemented on a PC with Intel i7-3370 CPU and 16G DDR3 memory.

4.1 The Zachary's karate club network

The Zachary's karate club network is a well labeled data and widely used in the test of community detection algorithms. Based on the data, our algorithm is compared with the three algorithms in time complexity, accuracy (the percent of vertices which are classified correctly to the community they belong to in real-world), and the need of prior information. The comparison results are listed in Table 2.

Since the initial vertex is selected randomly and uniformly, and the process of random walks is stochastic, the experimental results of our algorithm may be different even based on the same network data. In this case, the average accuracy of 100 experimental results is employed to evaluate the performance. Known from Table 2, the time complexity of our algorithm is lower than those of other algorithms, and even the accuracy is improved. The higher computational cost of other ones makes them powerless to be used for large scale networks in real-world, which will be shown below. Besides, the spectral bisection algorithm needs the prior information of the number of communities in networks.

Table 2. Algorithm comparison

Algorithm	*Prior information*	*Time complexity*	*Accuracy*
Spectral bisection	YES	$O(n^3)$	0.971
Girvan Newman	NO	$O(m^2n)$	0.971
Walktrap	NO	$O(mn^2)$	0.971
Our algorithm	NO	$O(n^2)$	0.974

In networks, sometimes vertices between communities are difficult to be classified into one certain community. Our algorithm also analyzes the overlapping parts, which is an advantage compared with the other algorithms. There are two communities in the Zachary's karate club, as shown in Figure 7. The community of the square vertices is denoted by C_1, and the community of the circular vertices is denoted by C_2. The probability that the vertex in overlapping parts belongs to certain community is shown in Table 3. The vertices in the overlapping parts have higher probabilities to belong to the communities which they belong to in the real-world.

Table 3. Algorithm comparison

Edges between communities	*The left vertex*		*The right vertex*	
	C_1	C_2	C_1	C_2
(3,33)	0.500	0.500	0.917	0.083
(20,34)	0.333	0.667	0.882	0.117
(1,32)	0.125	0.875	0.833	0.167
(3,9)	0.500	0.500	0.600	0.400
(14,34)	0.200	0.800	0.882	0.117
(3,10)	0.500	0.500	0.500	0.500
(2,31)	0.111	0.889	0.750	0.250
(1,9)	0.125	0.875	0.600	0.400
(3,29)	0.500	0.500	0.667	0.333
(3,28)	0.500	0.500	0.250	0.750

In order to reach the goal that most of the vertices in the graph are visited at least once with low computational cost, we analyze the relationships between the number of steps and the coverage rate of the vertices in the network. The distribution of the step number for visiting each vertex at least once in the network with 10000 experiments. In the process of our experiments, it is found the possibility of walker visiting all vertices reaches more than 98% when the number of steps is taken as n^2, where n is the number of vertices of the network.

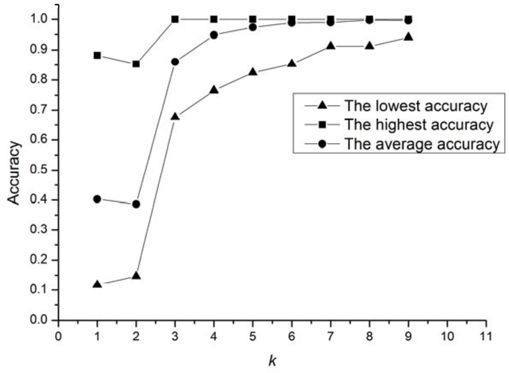

Figure 5． The accuracy of the results with 100 experiments for different value of *k* of *k*-vertices splitting strategy.

For the different value of *k*, we analyze the accuracy of the results with 100 experiments, including the best, worst and average cases, as shown in Figure 5. In order to compare the experimental results, we let *k*=1 represent the community detection results without the process of *k*-vertices splitting strategy. When *k*>2, the accuracy of results is significantly improved and the accuracy of the best results is 1.0, which indicates the effectiveness of the *k*-vertices splitting strategy. When *k*=8, the average accuracy of results is 0.997. Seen from Figure 6, when $k \geq 5$, the accuracy of results tends to be steady. Consider the computational cost, we set *k*=5 for the Zachary's karate club network.

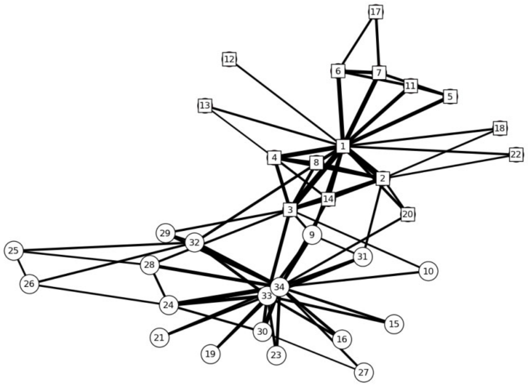

Figure 6. Weighted edges of the Zachary's karate club are the results of edge recording with 5-vertices splitting strategy. Different shapes present different communities gotten by our algorithm.

4.2 The other four networks

In this section, we briefly compare the performance of our algorithm with the mentioned ones above in the quality of results (*Q*) and running time in four further networks, i.e. bottlenose dolphins; football matches; collaboration network of scientists and the friendship of members in Facebook. Because of the limitations of prior information in the four networks, the spectral bisection algorithm cannot be tested. The results (running time/ modularity) can be seen in Table 4, and s denotes seconds. For the four networks, the values of *k* of *k*-vertices splitting strategy are taken as 5, 5, 10, 20, separately.

Table 4. Algorithm comparison

Network	*Dolphins*	*Football matches*	*Scientists*	*Facebook*
The number of vertices/mean degree	(62/5.12)	(115/ 10.66)	**(**22016/ 5.32**)**	**(**100000/ 25.72**)**
Our algorithm	0.15s/ 0.50	0.55s/0.56	21028.34s /0.45	206851.21 s/0.36
Walktrap	0.63s/ 0.50	2.74s/0.60	755873.87 s/0.41	>10days
Girvan Newman	56.02s /0.52	3429.34s/ 0.57	>10days	>10days

The experimental results show that our algorithm decreases the modularity slightly but has a huge advantage in running time. Even in the large scale network of collaboration of scientists, we have a better performance both in running time and quality. Particularly, when the number of vertices in networks reaches tens of thousands, Girvan Newman algorithm and Walktrap algorithm are gradually becoming powerless faced with the big graph data.

5. RELATED WORK

In recent years, many outstanding community detection algorithms have been proposed. Most of the algorithms [14] are graph partitioning or clustering based on distance, which leads to high computational cost. Meanwhile, many algorithms need to know the prior information, such as the community number, etc., which is difficult to be obtained in real networks.

Graph partitioning method. The Spectral bisection algorithm [12] is a graph partitioning method with good results and fast operating speed in practical applications. The algorithm runs in $O(n^3)$. However, it is required to define the number of the subgraphs before the graph partitioning. Newman and Girvan [13] propose an algorithm to get the communities based on edge removed. Its time complexity is $O(m^2n)$.

Hierarchical clustering method. The similarity of vertices determines the relationship between the vertices. Newman [6] proposes a greedy algorithm, which takes the modularity Q as a basis for whether to merge the two communities. The complexity of the algorithm is $O(n^2 \log n)$. A multi-level method for graph clustering is shown by Satuluri and Parthasarathy using stochastic flows [15].

Random walks. Pascal and Matthieu [8] propose a community detection algorithm based on random walks called "Walktrap". The algorithm obtains information of various vertices by the transition matrix, and defines three distances, and does hierarchical clustering by distance. The complexity of the algorithm for the worst case is $O(mn^2)$. A novel clustering method based on random walks for weighted graph is proposed by Harel and Koren [16].

Although these algorithms have good performance in quality, the high computational cost makes them hard to deal with the big graph data. With the development of web 2.0, the number of members in online social networks is increasing sharply. The challenges of big graphs are confronting us today. Faced with the challenges, we propose a n efficient algorithm just based on the topology of the graph and without prior information.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we propose an effective algorithm to solve the problem of community detection for big graphs and reduce the time complexity to $O(n^2)$ using random walks. The key point is how to find the closely connected vertices by random walks. Some new ideas including *k*-vertices splitting strategy, edge recording method and community detecting strategy are proposed. The experimental results show that our method provides good results in various networks. The comparison with other algorithms shows that our method has a clear advantage in running time with slightly reducing in quality. Even in the large scale networks, our algorithm also has advantage in quality.

Certain works are carrying on, such as the algorithm and theory based on multi-random walks to support parallel computing, the comparison with the latest algorithms, and the bigger graph data, the relationship between the value of *k* and the edge weights, etc.. The preliminary results show that the running time is reduced significantly by parallel computing.

There are also some problems left to the future work. How many steps can the walker visit each vertex of the networks at least once? How to calculate the value of *k* of the *k*-splitting strategy for different scales of networks? When the walker reaches stationary distribution is also a difficult problem.

ACKNOWLEDGMENTS

The research presented in this paper is supported in part by the National Natural Science Foundation (61221063, 61202392, 61172124, 61375040, 61272459), 863 High Tech Development Plan (2012AA011003), Key Projects in the National Science & Technology Pillar Program (2011BAK08B00) and Specialized Research Fund for the Doctoral Program of Higher Education (20120201120023).

7. REFERENCES

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

[2] Boccaletti, S., Latora, V., Moreno, Y., Chavez, M., & Hwang, D. U. (2006). Complex networks: Structure and dynamics. *Physics reports*, *424*(4), 175-308.

[3] Kelley, S., Goldberg, M., Magdon-Ismail, M., Mertsalov, K., & Wallace, A. (2012). Defining and Discovering Communities in Social Networks. In *Handbook of Optimization in Complex Networks* (pp. 139-168). Springer US Guy Shani, and Asela Gunawardana, "Evaluating Recommendation Systems", Recommender Systems Handbook, 2011, pp 257-297.

[4] Shani, G., & Gunawardana, A. (2011). Evaluating recommendation systems. In *Recommender systems handbook* (pp. 257-297). Springer US.

[5] Myers, S. A., Zhu, C., & Leskovec, J. (2012, August). Information diffusion and external influence in networks. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 33-41). ACM.)

[6] Newman, M. E. (2004). Detecting community structure in networks. *The European Physical Journal B-Condensed Matter and Complex Systems*, *38*(2), 321-330.

[7] Ribeiro, B., & Towsley, D. (2010, November). Estimating and sampling graphs with multidimensional random walks. In *Proceedings of the 10th ACM SIGCOMM conference on Internet measurement* (pp. 390-403). ACM.

[8] Pons, P., & Latapy, M. (2005). Computing communities in large networks using random walks. In *Computer and Information Sciences-ISCIS 2005* (pp. 284-293). Springer Berlin Heidelberg.

[9] Pastor-Satorras, R., & Vespignani, A. (2001). Epidemic spreading in scale-free networks. *Physical review letters*, *86*(14), 3200.

[10] Newman, M. E. (2001). Clustering and preferential attachment in growing networks. *Physical Review E*, *64*(2), 025102.

[11] Meyn, S. S. P., & Tweedie, R. L. (2009). *Markov chains and stochastic stability*. Cambridge University Press.

[12] Pothen, A., Simon, H. D., & Liou, K. P. (1990). Partitioning sparse matrices with eigenvectors of graphs. *SIAM Journal on Matrix Analysis and Applications*, *11*(3), 430-452..

[13] Newman, M. E., & Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical review E*, *69*(2), 026113.

[14] Lancichinetti, A., & Fortunato, S. (2009). Community detection algorithms: a comparative analysis. *Physical review E*, *80*(5), 056117.

[15] Satuluri, V., & Parthasarathy, S. (2009, June). Scalable graph clustering using stochastic flows: applications to community discovery. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 737-746). ACM.

[16] Harel, D., & Koren, Y. (2001). On clustering using random walks. In *FST TCS 2001: Foundations of Software Technology and Theoretical Computer Science* (pp. 18-41). Springer Berlin Heidelberg.