Community Detection of Complex Networks Based on the Spectrum Optimization Algorithm

Yueheng Sun, Shuo Zhang, Xingmao Ruan
School of Computer Science and Technology, Tianjin University
Tianjin, China
yhs@tju.edu.cn, yhstju@163.com, ruanhero@yahoo.cn

***Abstract*—This paper presents a spectrum optimization algorithm for community detection of the complex networks. An edge cutting model is proposed for selecting edges to be removed from candidates by minimizing algebraic connectivity function. In this model a greedy heuristic method is used to get the lower bound of optimal value, which makes it applicable to large-scale networks. Additionally, by weighting every edge based on Fielder vector, this model can effectively reduce the disadvantage influence of periphery edges of a graph on the result. The experiments on the simulated and the real complex networks show that this algorithm can reduce the time complexity of Newman-Girvan algorithm while preserving the performance of community detection.**

Keywords-community detection; complex networks; spectrum optimization algorithm; edge cutting model

I. INTRODUCTION

Facebook, Twitter, Google+ and other social networks promote the development of researches on the complex social networks, among which an important issue is to find communities from a network graph [1]. Usually, in a complex network the nodes and edges with the same properties tend to gather into a group and form a community structure, while the edges in different communities are relatively sparse. Finding such communities may have many potential applications. For example, we can take different marketing strategies for consumer groups in different communities to improve the delivery accuracy of advertising.

One of the most important algorithms for community detection is proposed by Newman and Girvan (NG algorithm) [2]. According to the centrality of each edge, this algorithm selects the edge to be deleted, and iteratively processes a network graph until the formation of expected community structures. However, the algorithm deletes only one edge each time, so if the mixing degree between communities is higher, the time complexity of this algorithm will be relatively high.

This paper presents an effective spectrum optimization community detection (SOCD) algorithm. It uses a heuristic method to measure the centrality of an edge, chooses those with higher centralities as candidates, and then proceeds to the optimization of network connectivity. This can effectively reduce the time complexity of traditional algorithms. In order to avoid the over-cutting to a network graph, we use the community modularity [3] as the algorithm's termination condition, which ensures our algorithm has the same performance as the NG algorithm.

The rest of this paper is organized as follows. Section 2 describes an edge cutting model. In Section 3, we present a community detection algorithm based on spectrum optimization. The experimental results are shown in Section 4. Finally, we give concluding remarks in Section 5.

II. CUTTING EDGE MODEL

Suppose that $G=(V,E)$ is an undirected graph with n nodes and m edges. For an edge $l \square (i,j)$ connecting node i and j, its corresponding vector is $a_l \in R^n$, here $a_{li}=1$, $a_{lj}=-1$, and the other elements of this vector are equal to zero. A is an $n \times m$ incidence matrix of graph G, and its l^{th} column is vector a_l. The Laplace matrix of G can be defined as equation (1).

$$L = AA^T = \sum\nolimits_{i=1}^{m} a_l a_l^T \tag{1}$$

The diagonal element L_{ii} is the degree of node i. if node i is connected with node j, then L_{ij} = -1. The other elements of L are equal to zero. The regularization eigenvector corresponding to the second smallest eigenvalue λ_2 is called Fiedler vector.

For two graphs $G_1=(V,E_1)$ and $G_2=(V,E_2)$, if $E_1 \subset E_2$, then $\lambda_2(L_1) \le \lambda_2(L_1)$. This means that the less the edge number is, the lower the algebraic connectivity is. It can be proved that $\lambda_2(L)$ is a monotone decreasing function of the number of edges. Therefore, we can define λ_2 as the network connectivity function $\lambda_2(L(x))$. It is an optimization problem to select the edge set resulting in the fastest decline of network connectivity, which can be defined as equation (2).

$$\min\nolimits_x \lambda(L(x)) \tag{2}$$

Given an original network graph $G=(V,E)$, it is necessary to find out the top-k edge set E_{cut} that has a higher influence on the network connectivity, where $E_{cut} \subseteq E$. Each edge in the graph G can be represented as a Boolean variable $x \in \{0,1\}^m$. If $l \in E_{cut}$, then x_l = 1, otherwise zero. If the number of candidate edges is m, then the search space is C_m^k. For a large-scale complex network, such search space is too huge to be solved. Considering the target edges usually connect two different communities and their betweenness are also higher, we can select top-m_c edges with the biggest betweenness as candidates, where $m_c \ll m$. With this greedy strategy, the search space is reduced to $C_{m_c}^k$. Another problem is that the algebraic connectivity function is sensitive to argument *x*. When the edges connecting periphery nodes between communities are cut, the algebraic connectivity reaches the lower bound, just affecting the model to optimize the results. Therefore, we set a weight $w_l \in [0,1]$ for each edge, which will make the connectivity function smoother. It is demonstrated that the partial derivative of function to argument *x* is $(v_i - v_j)^2$, where v_i is one element in Fielder vector. This value of an edge connecting two different communities is higher, otherwise lower. So, the weight of an edge is defined as equation (3).

$$w_l = \frac{(v_i - v_j)^2}{\max((v_A - v_B)^2) + \min((v_A - v_B)^2)} \tag{3}$$

Now, an edge cutting model can be defined as equation (4).

$$\begin{aligned} &\text{minimize} && \lambda_2(L - \textstyle\sum_{l=1}^{m_c} x_l w_l a_l a_l^T) \\ &\textit{subject to} && 1^T x \le k, \\ &&& x \in [0,1]^{m_c} \end{aligned} \tag{4}$$

Using the ultra-gradient optimization method [4], this model can process the graph cutting of large-scale complex network. Additionally, we can get a local optimal solution for this model by the greedy strategy.

III. SPECTRUM OPTIMIZATION COMMUNITY DETECTION ALGORITHM

Spectrum optimization community detection is a recursive partition process. At each step a network is split into two non-connected sub-graphs. If the global modularity of partition result is increased, then the algorithm is recursively executed in each sub-community until the modularity doesn't continue to rise.

Given a complex network $G_0 = (V, E)$, *V* is the node set and *E* is the edge set. The edge cutting model removes *k* edges at each iteration. Based on this model, the steps of SOCD algorithm are as follows:

(1) Compute the betweenness of each edge in G_0, and select m_c edges to be cut as candidates;

(2) Run the edge cutting model and delete *k* edges; compute the second smallest eigenvalue $\lambda_2(L(G_1))$ of the new network G_1, and if this value equals to zero, then go to step 3, otherwise return to step 1;

(3) Compute the global modularity of the partition graph, and if this value is increased, then execute step 1 and 2 in the non-connected sub-graph recursively, otherwise terminate the algorithm in the sub-graph.

The algorithm computes the centrality of each edge for selecting the candidates to be cut, which solves the problem of deleting only one edge each time in NG algorithm.

IV. EXPERIMENTS ON THE SIMULATED AND REAL NETWORKS

The experimental data includes the simulated networks generated by LFR Benchmark model [5] and the real network PolBlogs [6]. The correct results of community partition for the simulated networks have been given. We compare the performance of NG algorithm, NF algorithm [7] and SOCD algorithm on the two types of complex networks.

A. Experiments on the Simulated Networks

Two indicators, standard mutual information (SMI) and Jaccard coefficient, are used to measure the results of community detection of the simulated networks.

The SMI is used to compare the similarity between the partition result and the correct one, which is defined as equation (5).

$$I_{norm}(X,Y) = \frac{2I(X,Y)}{H(X) + H(Y)} \tag{5}$$

Where $0 \le I_{norm} \le 1$. If $I_{norm} = 1$, then the results of algorithms are identical to the standard result. On the contrary, if $I_{norm} = 0$, then the results of algorithms are completely irrelevant to the standard result. The SMI results on two networks with 200 and 500 nodes, respectively, are shown as Figure 1.

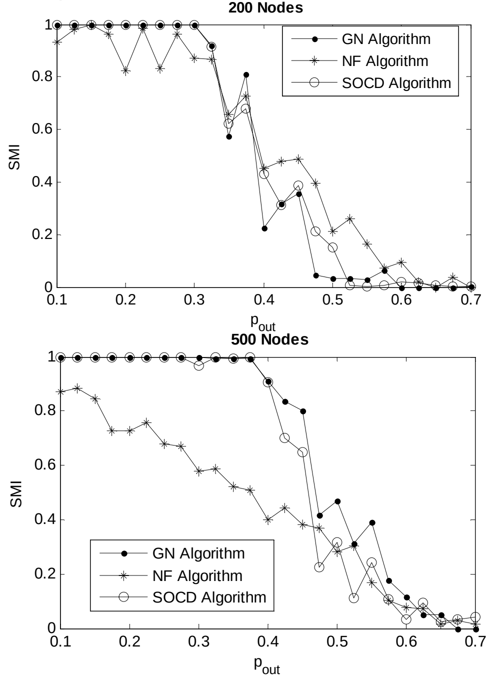

Figure 1. The comparison of SMI on two networks with 200 and 500 nodes.

As can be seen, when p_{out} is less than 0.3, the SOCD algorithm and NG algorithm achieve the same performance, and are better than NF algorithm. When p_{out} is greater than 0.3, the performance of three algorithms are decreased. When the network size is increased, the SOCD algorithm keeps the same partition performance with the NG algorithm, while the performance of NF algorithm is degraded quickly, which proves that this algorithm seems unfit to process large-scale networks.

The Jaccard coefficient is used to measure the correctness of community partition, which is defined as equation (6).

$$J(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|} \tag{6}$$

Where S_1 is the node pair set of correct result, and S_2 is the set of partition result. $J(S_1, S_2) = 1$ indicates that the most accurate result is achieved. The results on the above two networks are shown as Figure 2. It shows that the Jaccard coefficients of SOCD algorithm have an identical trend to SMI in Figure 1 but with a slower pace of declines, which indicates that the algorithm is capable of finding these intensive edge sets.

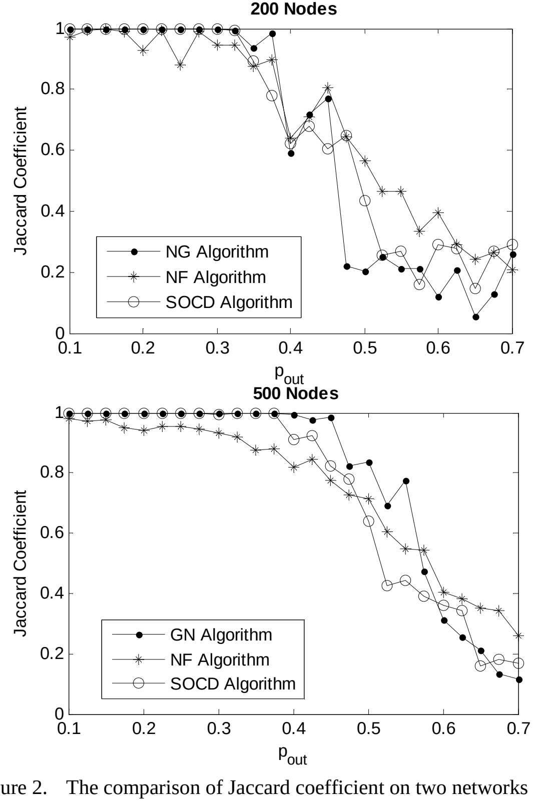

Figure 2. The comparison of Jaccard coefficient on two networks with 200 and 500 nodes.

B. *Experiments on the Real Network*

The real network involves the American statesman blog data in 2005, where each node represents a politician's blog, and each edge represents a link between two blogs. The data consists of 1490 nodes and 19090 edges. The partition result of SOCD algorithm is shown as Figure 3.

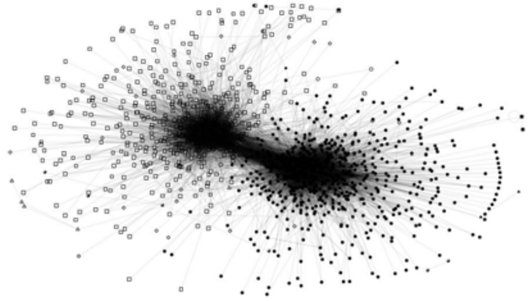

Figure 3. The partition of American statesman blog network.

It can be seen that the network is divided into two chief communities, respectively corresponding to the Republicans and Democrats in the USA, which indicates that the partition result is consistent with the reality.

C. *Algorithm Complexity Analysis*

The traditional NG algorithm removes an edge with the highest centrality in each iteration, and its time complexity is $O(n^2m)$, where n is the number of nodes and m is the number of edges. On the contrary, the SOCD algorithm cuts k edges in each iteration. Lanczos [8] algorithm can be used to calculate the Fielder vector corresponding to the second smallest eigenvalue. This algorithm also involves similar iterative process and the time complexity of each iteration is $O(n)$. It stops running when m_1 iterations are reached, where $m_1 < 100$. The m_c candidate edges with the higher values of $(v_i - v_j)^2$ are selected from all the edges, the time complexity of which is $O(n \log m_c)$. The algebraic connectivity $\lambda_2(L(x))$ is resolved by Gossip algorithm [4] whose time complexity is $O(m_2 \log n)$, where m_2 is the average iteration times on using the gradient descent method to optimize this algorithm. Therefore, the time complexity of SOCD algorithm is $O(m_1 + n \log m_c + m_2 \log n) \to O(n)$.

V. CONCLUSION AND FUTURE WORK

This paper presents an efficient algorithm for the community detection of complex networks. This algorithm uses a novel edge cutting model to select edges to be removed, which reduces the time complexity of traditional NG algorithm. The experiments on the simulated and the real networks show that our algorithm can also maintain the high accuracy of NG algorithm. In the future, we will combine the edge cutting model with the clustering algorithms to deal with these complex networks in which community partition may not be quite as obvious.

ACKNOWLEDGMENT

This work is sponsored by the Ministry of Education Humanities and Social Sciences Research Fund (13YJC870023) and Independent Innovation Fund of Tianjin University (06060306004).

REFERENCES

[1] Charu C. Aggarwal, Social Network Data Analytics. Springer Science + Business Media, LLC, 2011), pp. 1-15.

[2] M. E. J. Newman, M. Girvan, "Finding and evaluating community structure in networks", Phys. Rev. E, vol. 69, 2004, pp. 3-9. doi:10.1103/PhysRevE.69.026113.

[3] G. Agarwal, D. Kempe, "Modularity-maximizing network communities via mathematical programming", Eur. Phys. J. B, vol. 66, 2008, pp. 409-418. doi: 10.1140/epjb/e2008-00425-1.

[4] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Gossip algorithms: Design, analysis and applications," Proceedings of IEEE Infocom 2005, Miami, vol. 3, pp. 1653-1664, March 2005.

[5] A. Lancichinetti, S. Fortunato, F. Radicchi, "Benchmark graphs for testing community detection algorithms", Phys. Rev. E, vol. 78, 2008, pp. 3-5, doi: 10.1103/PhysRevE.78.046110.

[6] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 u.s. election: divided they blog", Proceedings of the 3rd international workshop on Link discovery, Chicago (Illinois), pp 36-43, August 2005.

[7] A Clauset, M. E. J Newman, "Finding community structure in very large networks", Phys. Rev. E, vol. 70, 2004, pp, 4-10, 10.1103/PhysRevE.70.066111.

[8] Cullum J. K., Willoughby R. A., "Lanczos algorithms for large symmetric eigenvalue computations", Classics in Applied Mathematics, (41) , Cambridge: Cambridge University Press, 2002.