A Novel Approach for Discovering Local Community Structure in Networks

Jinglian Liu[1,2], Daling Wang[1,3], Weiji Zhao[2], Shi Feng[1,3], Yifei Zhang[1,3],

[1]School of Computer Science and Engineering, Northeastern University, P. R. China
datamining@163.com
{wangdaling,fengshi,zhangyifei}@cse.neu.edu.cn
[2]School of Information Engineering, Suihua University, P. R. China
sdzhaoweiji@163.com
[3]Key Laboratory of Medical Image Computing of Ministry of Education,
Northeastern University, P. R. China

Abstract. The algorithms for discovering global community structure require the knowledge about entire network structures, which are still difficult and unrealistic to obtain from nowadays extremely large network. Several local algorithms that use local knowledge of networks to find the community for a given source node were proposed. However, these algorithms either require predefined thresholds which are hard to set manually or have lower precision rate. In this paper, we propose a novel method to discover local community for a given node. Firstly, we find the most similar node which is adjacent to the given node, and form the initial local community *D* together with the given node. Then, we calculate the connection degree of nodes belonging to *D*'s neighbors, and add the node whose connection degree is maximum to *D* if the local modularity measure will be increased. We evaluate our proposed method on well-known synthetic and real-world networks whose community structures are already given. The results of the experiment demonstrate that our algorithm is highly effective at discovering local community structure.

Keywords: Local Community Discovering; Community Structure; Connection Degree; Network Graph

1 Introduction

A wide variety of complex systems can be represented as networks, such as social networks [7, 8, 20], collaboration networks [17], the Internet [5], and E-mail networks [21]. Each of these networks consists of a set of nodes representing people in a social network, computers, or routers on the Internet. These nodes are connected together by edges, representing friendships between people, data connections between computers, and so forth [1].

Most of the networks display community structures that partition network nodes into groups within which the connections are dense but between which they are sparse

[15]. A number of community detection algorithms have been developed in recent years. The most popular algorithm is that proposed by Girvan and Newman [7, 14], which marked the beginning of a new era in the field of community detection. It detected community structure by removing the edges that connected nodes of different communities. In [14], the authors defined a modularity Q to test whether a particular division is meaningful, which was by far the most used and best known quality function [6]. Higher values of modularity indicate better partitions. Based on the modularity maximization, lots of algorithms [1, 15, 19] were proposed.

The global community detection algorithms have been well studied. But these algorithms are based on the entire network structures, which are still difficult and unrealistic to obtain from a large network nowadays. Therefore, community detection algorithms based on local network structure have been proposed. Clauset [4] proposed a local modularity measure R by only considering of nodes in the boundary of a sub-graph, and utilized a greedy maximization algorithm to find a sub-graph with a certain number of nodes. Bagrow et al. [2] explored the local structure of a given node by breadth-first search. A local module will be found until the change of the expansion falls below a predefined threshold. Luo et al. [11] proposed a local modularity measure M, which was the ratio of the number of internal edges to external edges. Based on this module definition, a locally optimized algorithm to identify local modules for a given node in a large network was given. Because the local modularity M was too strict for a community, it had low accuracy and recall in some cases. Ma et al. [13] proposed a seed-insensitive method called GMAC for local community detection. It revealed a local community by maximizing its internal similarity and minimizing its external similarity simultaneously.

However, most of the existing local community detection algorithms either require predefined thresholds which are hard to set manually or have lower precision rate. In this paper, we propose a novel method to discover the local community for a given node v. Firstly, we find the most similar node adjacent to v, which forms the initial local community D together with the given node v. Then, we calculate the connection degree of nodes belonging to D's neighbors, and add the node whose connection degree is maximum to D if that produces increase in local modularity. We evaluate our method on synthetic and real-world networks. The experimental results show that our algorithm is highly effective at discovering local community structure.

The rest of the paper is organized as follows. Section 2 gives related preliminary with our work. We describe our approach in Section 3 and report experimental results in Section 4, followed by conclusions in Section 5.

2 Preliminary

In this section, we first define the problem of discovering local community in networks, then review some existing algorithms for local community detection.

2.1 Definition of Local Community in Network

A network can be described by a graph $G=(V, E)$, where V is the set of nodes and E is set of edges. $|V|$ denotes the number of nodes in V. Nodes in V are denoted as v, w or V_i. $(v, w) \in E$ represents an edge connecting nodes v and w. For any node $v \in V$, function *neighbors*(G, v) returns a set of nodes that are adjacent to node v in G. Function *neighbors*(G, v) is the only way to gain additional information of G by visiting v's neighbor nodes.

Given a node v, our work is to discover the entire local community D that v belongs to. As shown in Fig.1, we have perfect knowledge of the connectivity of nodes in local community D. The shell node set of community D is $S=\{V_j|(V_i, V_j)\in E, V_i\in D, V_j\in V-D\}$, which contains the nodes that are adjacent to nodes in D but do not belong to D. Nodes of S have at least one neighbor in D. $U=V-D-S$ is the set of nodes in G except those in D or S, which we know nothing about. For any node $w \in S$, let $R=$*neighbors*(G, w), the nodes in R but not in D or S are explored from U. This is the only way to explore nodes in U during the process of detecting local community. Similar definitions of D, S can be found in [3, 10].

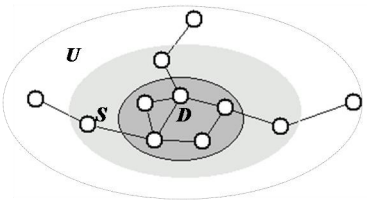


Fig.1: An Illustration of Division of a Network into Local Community D, D's Shell Node Set S and Unknown Node Set U

When local community detection algorithms begin, $D=\{v\}$ and $S=$*neighbors*(G, v). At each step, one or more nodes from S are chosen and agglomerated into D, then update S by adding their neighbor nodes of U. This process continues until an appropriate stopping criteria has been satisfied. D is the community discovered of v. D has two subsets: the core node set C and the boundary node set B. The nodes in C have no neighbor nodes belonging to S, but the nodes in B have at least one neighbor node belonging to S.

The basic quantity to consider is k_v, the degree of a node v, which is the number of nodes that are adjacent to node v. Radicchi et al. [18] extended the degree definition to the nodes in an undirected graph. For any node $v \in D$, $k_v^{in}(D)$ is the in-degree of v, which is the number of edges that connect v to nodes in D, $k_v^{out}(D)$ is the out-degree of v which is the number of edges that connect v to nodes that do not belong to D. So k_v is the sum of $k_v^{in}(D)$ and $k_v^{out}(D)$.

Generally, a network community is regarded as a group of nodes that are more densely connected inside the group than the outside of the network. Radicchi [18] proposed two community definitions. For a weak community, the sum of in-degree value of all nodes in it is greater than the sum of out-degree values of all nodes. For a strong module, each node in the community has higher in-degree than out-degree. But

no quantitative definition of community is universally accepted to decide whether D is a qualified local community or not [6].

2.2 Related Algorithms

Clauset [4] proposed a local modularity measure R by only considering boundary nodes in B.

$$R = \frac{B_{in}}{B_{in} + B_{out}} \tag{1}$$

where B_{in} is the number of inward edges that connect boundary nodes in B to other nodes in D, while B_{out} is the number of edges that connect boundary nodes in B to nodes in S. R measures the fraction of inward edges in all edges with one or more endpoints in B. At each step, the algorithm adds the node V_j in S which causes the largest increase in R to D, then updates S by adding V_j's neighbor nodes in U, until the community has reached a predefined size. However, fixing the community size does not allow the greedy algorithm to identify the locally optimal sub-graph from the given node [11]. Meanwhile, predefined parameters are hard to set manually when facing an unknown network.

Luo et al. [11] proposed another local modularity measure M for evaluating local community, which focuses on the ratio of the number of internal edges and external edges.

$$M = \frac{E_{in}}{E_{out}} \tag{2}$$

where E_{in} is the number of edges with two endpoints in D, while E_{out} is the number of edges with one endpoint in D and the other in S. At each step, nodes in S are agglomerated to D if they can cause an increase in M, then remove the nodes from D which can cause an increase in M, finally update S. This process is repeated until no nodes in S increase M if agglomerated in D.

Luo et al. [11] defined that D is considered to be a qualified local community if and only if $M>1$ and D contains v. The community definition is stricter than the strong definition by Radicchi et al. [18] sometimes. As shown in Fig.2, D is a strong community, but $M<1$. In this case, Luo's algorithm has low accuracy and recall.

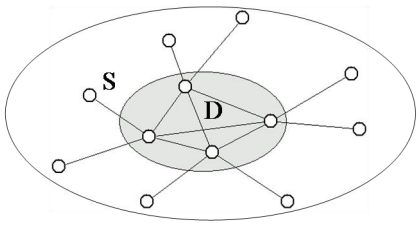


Fig.2: An Illustration of a Strong Community D

3 Our Algorithm

Before describing our approach, two improvements are given firstly based on strong community and local modularity measure M.

3.1 Problem Definition

For any node v in a strong community D, it requires $k_v^{in}(D)>k_v^{out}(D)$, which is equivalent to $k_v^{in}(D)>0.5\times k_v$. For node v in a community which satisfies $M>1$, even it requires $k_v^{in}(D)>2\times k_v^{out}(D)$ in average, which is equivalent to $k_v^{in}(D)>0.667\times k_v$. If there are only two communities in a network, for any node v in D, it's reasonable that half of the edges connected to v should be in D. However, for a large network G, there are maybe hundreds of communities in it. In such networks, for a node v belongs to community D, v only needs to have more edges with nodes in D than other communities of G. It's impossible to meet the demand that half of v's neighbor nodes fall in D. So in our algorithm, in order to measure the node v's connection degree with subgraph D, we propose *conn* as the criteria to choose candidate node in S.

$$conn(v,D)=\frac{k_v^{in}(D)}{k_v} \tag{3}$$

At each step, we choose the node in S whose *conn* is maximum as candidate nodes which will be agglomerated into D if an appropriate stopping criteria has been satisfied.

For local modularity metric M proposed by Luo et al. [11], there exists an exception when $E_{out}=0$. We propose an improved local modularity metric M', which avoids the exception raised by $E_{out}=0$.

$$M'=\frac{E_{in}}{E_{in}+E_{out}}=\frac{1}{1+\frac{E_{out}}{E_{in}}}=\frac{1}{1+\frac{1}{M}} \tag{4}$$

The monotonicity of the function M' is the same as M, which means that $\Delta M'>0$ is equivalent to $\Delta M>0$. We choose $\Delta M'>0$ as the stopping criteria to evaluate whether nodes in S can be added to D.

Suppose E_{in} denotes the number of internal edges in D, E_{out} denotes the number of external edges in D. For evaluating whether node V_j in S can be agglomerated in D, the increase of M' can be calculated as following.

$$\Delta M'=\frac{E_{in}+k_{V_j}^{in}(D)}{E_{in}+E_{out}+k_{V_j}-k_{V_j}^{in}(D)}-\frac{E_{in}}{E_{in}+E_{out}} \tag{5}$$

where $\Delta M'>0$ is equivalent to

$$\frac{k_{V_j}^{in}(D)}{k_{V_j}} > \frac{E_{in}}{2E_{in}+E_{out}} \tag{6}$$

which equivalents

$$conn(V_j,D) > \frac{E_{in}}{2E_{in}+E_{out}} \tag{7}$$

Formula (7) can be used as a rapid calculate method of $\Delta M'$.

3.2 Algorithm Description

Based on the above improvements, we propose a two-stage algorithm for discovering local community of a given node v without any manual parameters. In the first stage, find the most similar node adjacent to the given node v, form the initial local community D together with the given node v. In the second stage, calculate the connection degree of nodes belonging to D's neighbors, add the node whose connection degree is maximum to D if the local modularity measure will be increased.

Stage 1. Form the initial local community D. We calculate the given node v's similarity with every neighbor node according to Formula (8), and find the most similar node, denoted by w, then form initial local community D together with v.

$$similarity(v,x) = \frac{|neighbors(G,v) \cap neighbors(G,x)|}{min(|neighbors(G,v)|, |neighbors(G,x)|)} \tag{8}$$

Stage 1 is described in Algorithm 1 as follows.

Algorithm 1: Form the initial local community D

Input: a given node v, network $G=(V,E)$;
Output: initial local community D;
Describe:
1) $N=neighbors(G,v)$;
2) create a new list sim to store the similarities of nodes belonging to N with v;
3) for each node $x \in N$ do
4) $sim[x]=similarity(v,x)$
5) end for;
6) find w such that $sim[w]$ is maximum;
7) $D=\{v,w\}$;
8) return D

Stage 2. Expanding initial local community D. In the beginning, $D=\{v, w\}$, S is the shell nodes set of initial local community D. E_{in} is the number of edges in D. For there are only node v and its neighbor node w in D, so initialize $E_{in}=1$. E_{out} is the number of edges with one endpoint in D and the other in S. Choose the node in S whose connection degree is maximum as candidate node, denoted by c. If agglomerating

node c into D will cause an increase in M', which is equivalent to $conn(c,D) > \frac{E_{in}}{2E_{in} + E_{out}}$, add c to D, and update S, E_{in}, and E_{out}, repeat this step until S is empty; otherwise, return D as the local community of v.

Stage 2 is described in Algorithm 2 as follows.

Algorithm 2: Expanding initial local community D

Input: initial local community $D=\{v,w\}$, network $G=(V,E)$;
Output: local community D containing the given node v;
Describe:
1) $S=neighbors(G,v) \cup neighbors(G,w)-D$;
2) $E_{in}=1$;
3) E_{out}= the number of edges that connect v and w to nodes in S
4) while $S \neq$ empty do
5) for each node $x \in S$ do
6) calculate $conn(x, D)$;
7) end for;
8) find c such that $conn(c, D)$ is maximum;
9) if $conn(c,D) > \frac{E_{in}}{2 \times E_{in} + E_{out}}$ then
10) add c to D
11) update Ein, $Eout$, S
12) else
13) break
14) end if
15) end while
16) return D

4 Experiments

We compare our algorithm with Clauset's algorithm [4] and Luo et al.'s algorithm [11] (LWP for short) on LFR benchmark networks and four real-world networks for which the community structure are already known. The LFR benchmark networks are composed of 500 nodes and about forty communities [9], and the real-world networks are Zachary Karate Club Network [22], Dolphin Network [12], NCAA football network [7], and Books about US politics [16].

We test the performance of the three algorithms to detect local community by *Precision*, *Recall*, and *F-Score*, which are widely adopted by other community detection methods [3, 10]. The *precision* and *recall* are calculated as follows.

$$Precision = \frac{|C_F \cap C_R|}{|C_F|} \tag{9}$$

$$Recall=\frac{|C_F \cap C_R|}{|C_R|} \tag{10}$$

where C_R indicates the node set forming the real local community originating from a given node and C_F represents the node set which is the result of the local community detection algorithm.

Precision is the ratio of the correct nodes found in the detected local community. *Recall* is the ratio of the correct nodes to the real local community. *F-Score* is the harmonic mean of *Precision* and *Recall*. Its formula is as follows.

$$F-Score=2\times\frac{precision\times recall}{precision+recall} \tag{11}$$

In our experiments, every node in these networks has been taken as the start node to discover its local community. Based on the real community, the *precision*, *recall*, and *F-score* of every node is calculated. We average the score of *precision*, *recall*, and *F-Score* of all nodes in one network to evaluate the effectiveness of the algorithms to detect local community. A well-performed algorithm should have high *precision*, *recall*, and *F-score* at the same time.

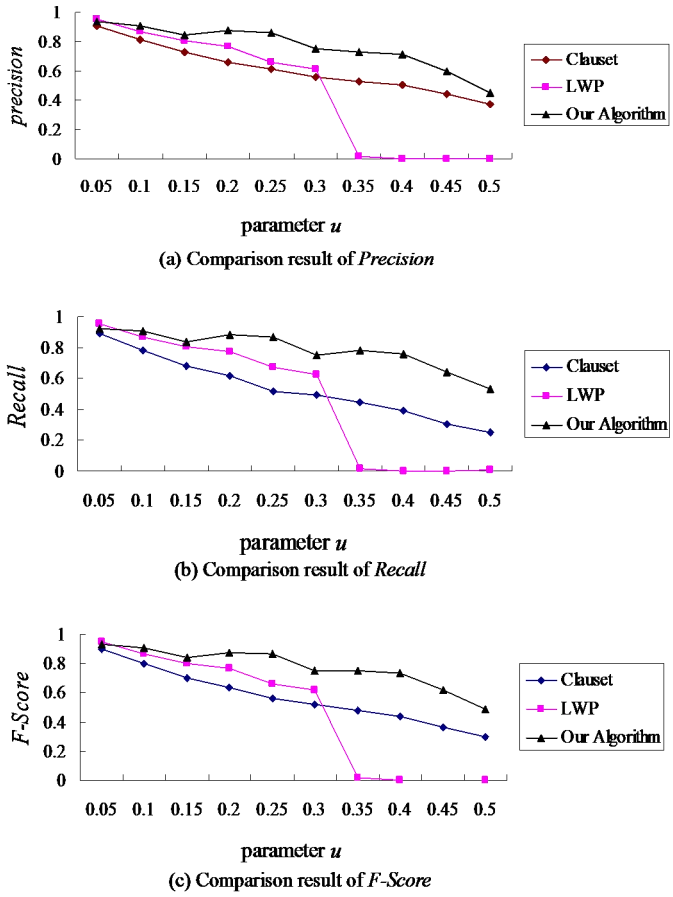
4.1 Experiment on LFR Benchmark Network

LFR benchmark network is given by Lancichinetti et al. [9]. We generate 10 networks with different mixing parameter u ranging from 0.05 to 0.5 with a span of 0.05. Mixing parameter u is the fraction of its links with nodes outside its community. These networks' important properties are presented as follows: the number of nodes n=500, the average degree of the nodes k=10, and the maximum degree k_{max}=50. For the others, such as minus exponent for the degree sequence t_1, minus exponent for the community size distribution t_2, number of overlapping nodes *on*, number of memberships of the overlapping nodes *om*, minimum for the community sizes *minc*, maximum for the community sizes *maxc* use default values. The community structures of these LFR benchmark networks are already known.

For any node v in these LFR networks, it has $(1-u)\times k_v$ neighbor nodes in its own community and $u\times k_v$ neighbors in other communities. The higher the mixing parameter u of a network is, the weaker community structure it has. So along with the community structures of the LFR networks become weaker, all the three algorithms suffer varying degrees of performance degradation and become ineffective to detect community structure. Fig.3 shows the comparison results of *precision*, *recall*, *F-score* for three algorithms on these networks, respectively.

To be more precisely, when $u\leq 0.3$, Clauset's algorithm has lower *precision*, *recall*, and *F-score* than other two algorithms. When $u\geq 0.35$, the *precision*, *recall*, and *F-score* of the LWP algorithm is zero or nearly zero. This is because all the local communities discovered by LWP algorithm satisfy $M>1$, which means the number of edges within the community should be more than the number of edges between nodes in the community and nodes outside it. However, almost no local community can

satisfy $M>1$ when $u\geq 0.35$, so LWP algorithm performs badly in this case. By comparison, our algorithm achieves much higher *precision*, *recall*, and *F-score* at the same time than other two algorithms except when $u=0.05$. When $u=0.05$, all three algorithms have good performance. Exactly, our algorithm is a little lower than LWP algorithm, but higher than Clauset's algorithm. In general, our algorithm achieves better performance to discovery local community against the other algorithms on LFR benchmark networks.



(a) Comparison result of *Precision*

(b) Comparison result of *Recall*

(c) Comparison result of *F-Score*

Fig.3: Comparison Results on LFR Benchmark Networks

4.2 Experiments on Real-World Networks

We evaluate the performance of the three algorithms on the four real-world networks. The real-world networks are Zachary Karate Club Network [22], Dolphin Network [12], NCAA football network [7], and Books about US politics [16].

(1) Zachary's karate club network (Karate for short)

Karate is a network of the friendships among 34 members of one karate club at a US university, which is observed by Zachary from 1970 to 1972, in which $|V|$=34 and $|E|$=78. The club was later divided into two smaller groups for the disputation between the supervisor and coach.

The comparison results on Karate is shown in Fig.4(a). Compared with LWP algorithm, Clauset's algorithm has the higher *precision*, but the lowest *recall* result in that it has the lowest *F-score* of the three algorithms. Our algorithm has the highest *precision*, *recall*, and *F-score* at the same time.

(2) Dolphins Network (Dolphins for short)

Dolphins is a network of frequent associations between 62 dolphins in a community living off Doubtful Sound, New Zealand, in which $|V|$=62 and $|E|$=159. Each node represents a dolphin, while each edge represents frequent relationships between two dolphins. The network is divided into two groups because of the migration of species.

The comparison results on Dolphins is shown in Fig.4(b). Compared with Clauset's algorithm, LWP algorithm has the higher *recall*, but the lowest *precision* result in that it has the lowest *F-score* of the three algorithms. And Our algorithm has the highest *recall* and *F-score,* and the *precision* is close to Clauset's.

(3) NCAA football network (NCAA for short)

NCAA is a network of American football games between Division IA colleges during regular season Fall 2000, in which $|V|$=115 and $|E|$=613. Each node represents a team and edge represents regular season games between two connected teams. Teams within the same conference play more games than teams from different conferences. 115 teams are divided into 11 conferences and five independent teams;

The comparison results on NCAA is shown in Fig.4(c). Compared with LWP algorithm, Clauset's algorithm has the higher *precision*, *recall*, and *F-score*. And our algorithm has the highest *precision*, *recall*, and *F-score* at the same time.

(4) Books about US politics (Polbooks for short)

Polbooks is a network of books about US politics published around the time of the 2004 presidential election and sold by the online bookseller Amazon.com, in which $|V|$=105 and $|E|$=441. Each node represents a book and edges represent frequent co-purchasing of books by the same buyers. 105 books are divided into 3 communities.

The comparison results on Polbooks is shown in Fig.4(d). Clauset's algorithm has the lowest *recall*, meanwhile it has the highest *precision*, result in that its *F-score* is higher that LPW's. And our algorithm has the highest *recall* and *F-score,* and the *precision* is close to Clauset's.

Fig.4 respectively shows the comparison results for the three algorithms on four real-world networks. We can observe that the *precision* of our algorithm is usually better than LWP algorithm, and is better or approximately equal to Clauset's

algorithm, and the *recall*, especially the *F-score* of our algorithm are usually higher than other two algorithms. The results of the experiment demonstrate that our algorithms are highly effective at discovering local community structure compared with the other strong baseline algorithms.

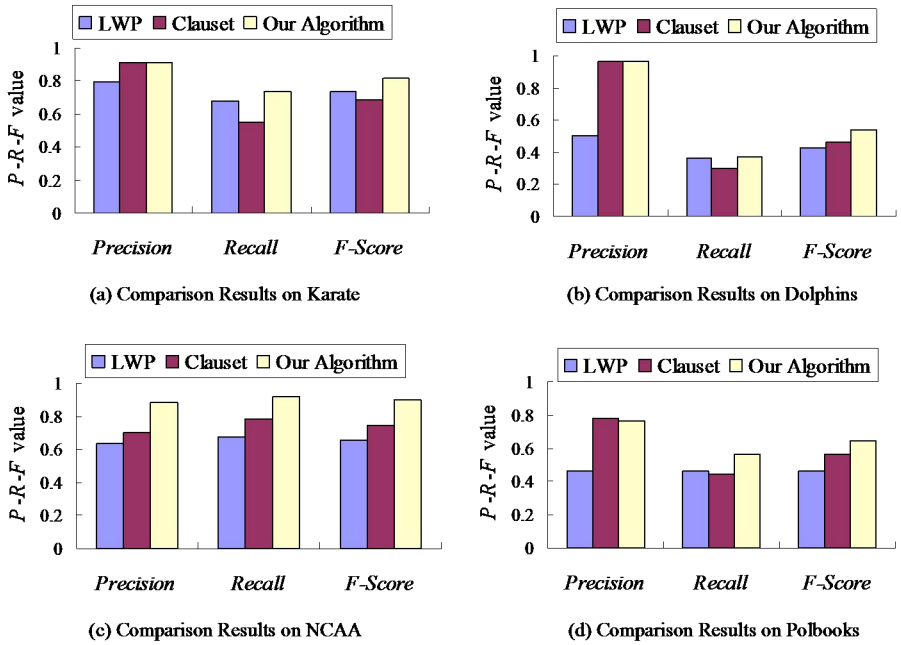


Fig.4: Comparison Results on Real-World Networks

5 Conclusion and Future Work

Currently, many local community detection algorithms have been proposed to identify community structure from the given starting node. However, these algorithms either require predefined thresholds which are hard to set manually or have lower precision rate. In this paper, we propose a novel approach for discovering local community. Comparing with other algorithms, our algorithm doesn't need any manual parameters, and achieves better performance on both the real world networks and synthetic networks. Future work can be done on the application of our algorithm on real networks for discovering local community.

Acknowledgments

The project is supported by National Natural Science Foundation of China (61370074, 61402091), the Fundamental Research Funds for the Central Universities of China under Grant N140404012.

References

1. Aaron C, Newman M, Cristopher M: Finding community structure in very large networks. Physical Review E Statistical Nonlinear & Soft Matter Physics, 2004, 70(6):264-277.
2. Bagrow J, Bolt E: A Local method for detecting communities. Physical Review E, 2005,72(4):046108-1-046108-10.
3. Chen Q, Wu T: A method for local community detection by finding maximal-degree nodes. ICMLC 2010:8-13.
4. Clauset A: Finding local community structure in networks, Physical Review E, 2005, 72(2):026132.
5. Faloutsos M, Faloutsos P, Faloutsos C: On Power-law Relationships of the Internet Topology. SIGCOMM 1999:251-262.
6. Fortunato S: Community detection in graphs. Physics Reports, 2009, 486(3/4/5):75-174.
7. Girvan M, Newman M: Community structure in social and biological networks. Proceedings of the National Academy of Sciences of the united States of America. 2002, 99(12): 7821-7826.
8. Jia G, Cai Z, Musolesi M, et al.: Community Detection in Social and Biological Networks Using Differential Evolution. LION 2012:71-85.
9. Lancichinetti A, Fortunato S, Radicchi F. Benchmark graphs for testing community detection algorithms. Physical Review E, 2008, 78(4): 046110-1-046110-5.
10. Liu Y, Ji X, Liu C, et al.: Detecting Local Community Structures in Networks Based on Boundary Identification. Mathematical Problems in Engineering, 2014:1-8. //http:// dx.doi.org/10.1155/2014/682015.
11. Luo F, Wang J, Promislow E: Exploring local community structures in large networks. Web Intelligence and Agent Systems (WIAS), 2008, 6(4):387-400.
12. Lusseau D: The emergent properties of a dolphin social network. Proceedings of the Royal Society of London. Series B Biological Sciences, 2003,270 suppl 2(1530):S186-188.
13. Ma L, Huang H, He Q, Chiew K, Wu J, Che Y: GMAC: A Seed-Insensitive Approach to Local Community Detection. DaWaK 2013:297-308.
14. Newman M, Girvan M: Finding and evaluating community structure in networks. Physical Review E Statistical Nonlinear & Soft Matter Physics, 2004, 69(2):026113-1-026113-15.
15. Newman M: Fast algorithm for detecting community structure in networks. Physical Review E Statistical Nonlinear & Soft Matter Physics, 2004,69(6):066133-1-066133-5.
16. Newman M: Modularity and Community Structure in Networks. Proceedings of the National Academy of Sciences. 2006, 103(23): 8577-8582. //http://www-personal.umich.edu/ ~mejn/netdata/.
17. Newman M: The Structure of Scientific Collaboration Networks. Working Papers, 2000, 98(2):404-409.
18. Radicchi F, Castellano C, Cecconi F, et al.: Defining and identifying communities in networks. Proceedings of the National Academy of Sciences of the United States of America, 2004, 101(9):2658-2663.
19. Shang R, Bai J, Jiao L, et al.: Community detection based on modularity and an improved genetic algorithm. Physical A, 2013, 392(5):1215-1231.
20. Takaffoli M: Community Evolution in Dynamic Social Networks - Challenges and Problems. ICDM Workshops 2011:1211-1214.
21. Tyler J, Wilkinson D, Huberman B: Email as Spectroscopy: Automated Discovery of Community Structure within Organizations. Information Society, 2005,21(2):143-153.
22. Zachary W: An information flow model for conflict and fission in small groups. Journal of Anthropological Research, 1977,33(4):452−473.