Wu YJ, Huang H, Hao ZF *et al.* Local community detection using link similarity. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 27(6): 1261–1268 Nov. 2012. DOI 10.1007/s11390-012-1302-4

Local Community Detection Using Link Similarity

Ying-Jun Wu[1] (吴英骏), Han Huang[1,2,*] (黄　翰), *Member*, *CCF*, *ACM*, *IEEE*, Zhi-Feng Hao[3] (郝志峰), and Feng Chen[4] (陈　丰)

[1] *School of Software Engineering, South China University of Technology, Guangzhou 510000, China*

[2] *Department of Management Sciences, College of Business, City University of Hong Kong, Hong Kong, China*

[3] *Faculty of Computer Science, Guangdong University of Technology, Guangzhou 510000, China*

[4] *School of Journalism and Communication, South China University of Technology, Guangzhou 510000, China*

E-mail: wu.yj0616@gmail.com; {hhan, mazfhao}@scut.edu.cn; c.feng04@mail.scut.edu.cn

Received September 9, 2011; revised March 9, 2012.

Abstract Exploring local community structure is an appealing problem that has drawn much recent attention in the area of social network analysis. As the complete information of network is often difficult to obtain, such as networks of web pages, research papers and Facebook users, people can only detect community structure from a certain source vertex with limited knowledge of the entire graph. The existing approaches do well in measuring the community quality, but they are largely dependent on source vertex and putting too strict policy in agglomerating new vertices. Moreover, they have predefined parameters which are difficult to obtain. This paper proposes a method to find local community structure by analyzing link similarity between the community and the vertex. Inspired by the fact that elements in the same community are more likely to share common links, we explore community structure heuristically by giving priority to vertices which have a high link similarity with the community. A three-phase process is also used for the sake of improving quality of community structure. Experimental results prove that our method performs effectively not only in computer-generated graphs but also in real-world graphs.

Keywords social network analysis, community detection, link similarity

1 Introduction

Researchers have paid much attention to analyzing the large complex networks, such as Facebook networks[1-2], web page networks[3], and academic networks[4]. These networks are commonly modeled as a graph containing m vertices and n edges. Finding community structure can be regarded as exploring subgraphs where vertices are densely connected among themselves and loosely connected to other vertices[5]. With the knowledge of the whole graph, subsets can be detected by using techniques such as hierarchical clustering[6], spectral clustering[7-8], partitioned clustering[9], correlation-based clustering[10-11] and ranking-based algorithms[12-13].

However, many real-world networks, i.e., social network, World Wide Web, share the common feature that the complete structure of the network is often unavailable since the entire network is too large and too dynamic[14]. Restricted by this confinement, we should try to explore community structure from limited accessible region of a graph. Many researchers have proposed several local methods that use partial knowledge of the network to discover the local community with a certain source vertex[15-20]. Several metrics for local community structure have also been presented in some work[16-17,19]. Combined with the advantage of these quality metrics, the existing algorithms have made great improvement in producing a more precise and complete community. Meanwhile, some work has also been issued to reduce the complexity of this task[18]. Nevertheless, these existing methods suffer in one or more ways. For instance, methods proposed in [16, 18] depend much on predefined parameters which are hard to obtain; methods in [16-18] are sensitive to the source vertex's position and their results may

Short Paper

This work was supported by the National Natural Science Foundation of China under Grant No. 61170193, the Doctoral Program of the Ministry of Education of China under Grant No. 20090172120035, the Natural Science Foundation of Guangdong Province of China under Grant No. S2012010010613, the Fundamental Research Funds for the Central Universities of South China University of Technology of China under Grant No. 2012ZM0087, the Pearl River Science & Technology Start Project of China under Grant No. 2012J2200007.

*Corresponding Author

contain many outliers; method in [19] puts a strict limitation in agglomerating new vertices so that an integrated community is hard to be explored.

In this paper, we present a method that mainly depends on link similarity between the vertex and community to explore local community. This method searches the potential vertices in a specific sequence so as to help improve the accuracy. Meanwhile, our method is a self-adaptive three-phase algorithm that includes adding suitable vertices into the community and removing unqualified vertices from the community without any help of predefined parameters. Moreover, the third phase of our algorithm removes possible outliers so as to guarantee that we can finally get a strong community[21-22]. Experimental results convince that our three-phase method using link similarity performs effectively not only in computer-generated graphs but also in real-world graphs.

This paper is organized as follows. Section 2 defines the local community detection problem and reviews the existing methods. In Section 3, we describe the idea of link similarity and analyze progresses of our algorithm. Section 4 shows our experimental results and Section 5 concludes the paper.

2 Preliminaries

2.1 Problem Definition

Problem definition of local community detection is first and formally proposed by Clauset in [16]. Generally, we define local community detection problem as follows: Given an undirected network $G = (V, E)$ where V represents the set of vertices in the graph and E the set of edges, we have perfect knowledge of the connectivity of some set of vertices, i.e., the known local community of the graph, which is denoted as C. Necessarily, a set of vertices N that are adjacent to vertices in C but do not belong to C can be partially known (note "partially" means that the complete connectivity of any vertex in N is unknown). Moreover, we define the shell vertex set as region S, where any vertex $v \in S$ has at least one neighbor in N. See Fig.1(a). Let assume that the only way to gain further knowledge about G is to choose one vertex from N and merge it into C, thus the additional unknown vertices may be added to N. The task of this problem is to constitute a local community C from a single source vertex by this one-vertex-at-a-time step.

2.2 Related Work

We have reviewed several effective approaches to explore local community structure. These methods are presented below in the sequence of publication date.

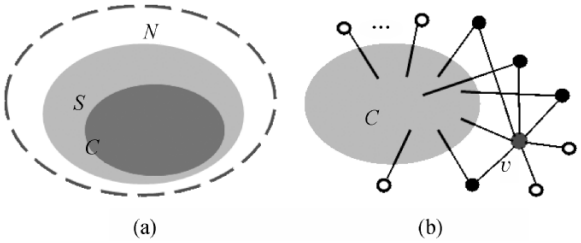


Fig.1. (a) Local community structure. C is the community that has been explored; S is the region composed of shell vertices; N is the set of known vertices which are in the neighborhood of community C. Vertices in N are agglomerated into community C one by one. (b) Vertex v and community C share some same links. In this figure, $LS_C(v)$ (link similarity of v defined in Subsection 3.1) equals to 0.714.

Clauset has defined the local modularity R in order to solve the local community detection problem[16]. This metric mainly focuses on the connectivity of vertices in S:

$$R = \frac{\sum_{ij} S_{ij}\delta(i,j)}{\sum_{ij} S_{ij}}, \tag{1}$$

where $\delta(i, j)$ is 1 when either $v_i \in s$ and $v_j \in C$ or $v_i \in C$ and $v_j \in S$, and is 0 otherwise; S_{ij} is 1 when vertices i and j are connected and is 0 otherwise. Clauset's algorithm always chooses the vertex which results in the largest ΔR as the one that is inserted into C. However, this approach asks for predefined parameter that is used to determine the size of C. Meanwhile, its result is influenced by the source vertex.

LWP algorithm is an improved method which mainly promotes the stopping criteria[17]. It defines another form of local modularity M which uses the idea of weak community. The local modularity M is:

$$M = \frac{\frac{1}{2}\sum_{ij} A_{ij}\theta(i,j)}{\sum_{ij} A_{ij}\lambda(i,j)}, \tag{2}$$

where $\theta(i, j)$ is 1 if both vertices i and j are in subgraph C, and 0 otherwise; $\lambda(i, j)$ is 1 if only one vertex is in subgraph C, and 0 otherwise; A_{ij} is 1 when vertices i and j are connected, and 0 otherwise. This algorithm has both an addition step and a deletion step. Vertices will be added or removed from C if and only if it can cause an increase in M. This algorithm turns out to result in high recall but low accuracy.

Bagrow used M_{out} proposed in LWP as a measure of community quality[18]. He also defined the "outwardness" of a vertex to determine which vertex should be agglomerated into the community. However, the stopping criteria of this algorithm also depend on an arbitrary parameter which controls how "strong" the community should be.

Chen *et al.* have presented an alternative method to discover local communities, which aims at reducing outliers and improving detection accuracy[19]. A new measure of local community structure called L was also proposed to help optimize the community hierarchy. Due to its strict criteria in agglomerating vertices, this algorithm can hardly obtain a comparatively integrated community structure.

Some variations of this problem have also been discussed in recent years. The studies presented in [21] and [22] mainly discuss how to explore local community in bipartite networks, while work in [23] and [24] focuses on how to detect community structure from a set of source vertices.

In the topic of global community structure detection, some other novel methods have also emerged recently. Su *et al.*[10] and Kriegel *et al.*[11] discussed the correlation-based clustering method for community detection. Liu *et al.*[12] and Hotho *et al.*[13] solved this problem using ranking-based algorithms. In [25], an information theoretic approach was presented.

In this paper, we only discuss the problem of detecting local community structures from a certain source vertex in a partially-known network.

3 Improved Local Community Detection Method

Local community detection approaches are restricted by two limitations[16-20]. One is that only a small part of the vertices can be known by us, and the other is that only one vertex can be agglomerated into the community at one step. Limited by these two confinements, we propose an algorithm focusing on the feature of link similarity. Notations appeared in this section were defined in Subsection 2.1.

3.1 Link Similarity

The algorithm we propose is quite easy to implement. Firstly, we define the link similarity $LS_C(v)$ of vertex $v \in N$ from community C:

$$LS_C(v) = \frac{|(\Gamma(C) \bigcup C) \bigcap \Gamma(v)|}{|\Gamma(v)|} = \frac{|(N \bigcup C) \bigcap \Gamma(v)|}{|\Gamma(v)|}, \tag{3}$$

where $\Gamma(C)$ is the set of adjacent vertices of C and $\Gamma(v)$ is the set of neighbors of v. $\Gamma(C)$ actually equals to N. Intuitively, we can regard the link similarity of a vertex as the ratio of the number of common links shared by the vertex and the community to the number of neighbors the vertex owns. As a result, $LS_C(v)$ gets the maximum value of 1 when its neighbors are either in C or N. The minimum value of $LS_C(v)$ is $1/(|\Gamma(v)|)$ since any $v \in N$ has at least one adjacent vertex that belongs to C. It is easy to understand that the vertex sharing more common neighbors with community C is more likely to be a member of C. See Fig.1(b). In our algorithm, the vertex having the largest value of $LSC(v)$ will be agglomerated into community C at each step.

Link Similarity method works due to the reason that it can help to find out a stable community structure as soon as possible. Actually, this method can be regarded as a heuristic method. Instead of searching potential vertices in an arbitrary manner, our approach is capable of locating the position of the community by giving priority to vertices having a high link similarity with the community. Consequently, a more integrated and precise community can be obtained since it is extended from an initially stable community structure. Without this stable structure, an algorithm is more likely to miss its direction in merging new vertices, hence a comparatively bad result may be returned.

This method also has actual meanings in real-world networks. Take citation network for example. Two different papers focusing on the same field will definitely cite some same work. So we can predict that two papers sharing a lot of same references are more likely to be in the same cluster. Similarly, we can also derive the conclusion that a paper belongs to a certain field if it cites a lot of common papers with the work in the field. This conclusion still holds in the case of social network. Intuitively, two persons belonging to the same community will certainly share a lot of friends, even if the two guys do not know each other. After discover a small community containing only two intimate-related guys, we can gradually use this conclusion to detect a larger community.

3.2 Measure of Quality and Stopping Criteria

To evaluate the quality of a community structure, several metrics have been introduced, including local modularity R proposed in [16] and M proposed in [17]. R focuses on the shell vertex set S to evaluate the quality of the discovered local community C, while M depends on the "internal edges" inside and "external edges" outside community C to evaluate quality. The experiments in Section 4 show that local modularity M proposed in LWP algorithm is good enough for our algorithm. Some other metrics may also work well with our algorithm such as local modularity L presented in [19], but L may be too strict. Meanwhile, we prefer to choose a simpler one. Detailed experimental analysis will be shown in Section 4.

Our algorithm keeps on agglomerating new vertices into the community until no vertex's agglomeration into the community can result in increment of M. If the

source vertex is removed from community in the optimization phase presented in Subsection 3.3 below, our algorithm will stop immediately and return a result that no community is found for the source vertex.

3.3 Three-Phase Algorithm

Our algorithm consists of three phases: the greedy agglomeration phase, the optimization phase, and the trimming phase. In the agglomeration phase, we sort the vertices in N according to their link similarity in descending order, and then determine whether to agglomerate them into the community according to the stopping criteria. In the optimization phase, all the vertices in S will be judged to determine whether we should remove them from the community. These two phases continue until no vertex can be agglomerated into the community. If the source vertex is removed from the community in the optimization phase, then we stop the algorithm and determine that no community exists for this algorithm.

After these two phases, a trimming phase is performed in order to remove the outliers. The trimming phase examines whether each vertex i in region S has more neighbors inside C than outside:

$$k_{\text{in}}(i) > k_{\text{out}}(i), \tag{4}$$

where k_{in} denotes the number of edges inside community C and k_{out} denotes the number of edges between community C and region N. Vertices that do not satisfy (4) will be removed from C. This process guarantees that a strong community[26-27] can be finally explored.

The summary of our algorithm is shown in Fig.2. Local modularity M implemented in Fig.2 is computed as (2).

The computation of ΔM for each vertex v in the network G in either agglomeration phase or optimization phase can be done quickly according to (5):

$$\Delta M = \begin{cases} \dfrac{xk_{\text{out}} + (x-y)k_{\text{in}}}{(k_{\text{out}} - x + y)k_{\text{out}}}, \\ \qquad\qquad \text{in agglomeration phase}, \\ \dfrac{-xk_{\text{out}} - (x-y)k_{\text{out}}}{(k_{\text{out}} + x - y)k_{\text{out}}}, \\ \qquad\qquad \text{in optimization phase}, \end{cases} \tag{5}$$

where x represents the number of edges that will be added or removed from C because of the agglomeration or removal of vertex v, and y is the number of edges which connect v and a vertex outside C. It is easy to find out that the degree of v equals to $x + y$.

For the purpose of achieving a better efficiency, we built a max-heap of vertices in N sorted by their link similarity. Hence, extracting the vertex with maximum link similarity will only cost $O(\log |N|)$. And it also takes $O(\log |N|)$ to insert a new vertex into the heap. Note that the heap should be rebuilt after agglomeration or removal.

Input: An undirected network G and a source vertex p_0
Output: A local community containing source vertex v_0
Add source vertex v_0 to C and S, add v_0's neighbors to N
Local modularity $M = 0$
Build heap H for vertices in N by link simularity $LS_c(v)$ in descending order
Do
  While H is not empty **Do**
    Find vertex p such that $LS_c(v)$ is maximum
    Compute ΔM for v
    If $\Delta M > 0$ **Then**
      Add v to C and remove v from N
      Update S, M
    Do
      For each $v \in S$ **Do**
        Compute ΔM for v
        If $\Delta M > 0$
          Remove v from S and C
          Add v to N
          Update M
        If $v = v_0$ **return** "no communitry"
    Until no vertex is removed from S and C
  Update S, N and H
Until no new vertex is inerged into C
For each $v \in S$ **Do**
  If $k_{\text{in}}(t) > k_{\text{out}}(i)$
    Remove v from C
If source vertex v_0 is in C
  Return C_0
Else Return "no community"

Fig.2. Local community detection algorithm using link similarity.

It is worth noticing that some existing approaches also have multi-phase process[17,19-20]. However, our algorithm is stricter and simpler in the optimization and trimming phases. Firstly, we only check vertices in region S instead of those in community C to optimize the community since vertices in the boundary of a community is more likely to be removed from the community. Secondly, we stop our algorithm as soon as the source vertex is removed from the community. Thirdly, we ensure that our result is a strong community by performing the trimming phase.

4 Experiments and Results

In our experiments, we compare the results of different methods. Following is the description of these labels we use to denote each of these algorithms:

• *Clauset.* This is a basic algorithm of [16]. Note that we improve its stopping criteria by detecting changes in local modularity R.

• *LWP*. This is a two-phase algorithm proposed in [17] using local modularity M.

• *Bagrow-TLS*. This algorithm is presented in [18]. It uses trailing least-squares stopping criterion.

• *Chen*. This method is also a two-phase algorithm using local modularity L proposed in [19].

• *Local Core*. This algorithm is the most recently proposed method we can find. It is a three-phase algorithm proposed in [20].

• *LS with M*. This method is the version of our algorithm using link similarity with local modularity M.

• *LS with R*. This method is the other version of our algorithm using link similarity with local modularity R.

4.1 Datasets

We perform experiments using one computer-generated graph and one real-world dataset.

Computer-Generated Graph. We use the computer-generated GN benchmark graph to validate our algorithm[28]. This graph is composed by 128 vertices, which are equally divided into four groups consisting of 32 vertices. Each vertex has a constant degree of 16. The possibilities of links between vertices in the same group are larger than those between vertices in different groups. Assume each vertex has z_{in} neighbors in the same group and z_{out} neighbors in the different groups. Then we have $z_{\text{in}} + z_{\text{out}} = 16$. This benchmark graph is tested in [16, 19-20].

NCAA Football Network. This dataset describes a network of National Collegiate Athletic Association (NCAA) Football Bowl Subdivision during regular season 2006[29]. This network contains 787 football games between 115 university football teams from 11 communities. Notably, 61 school teams are from lower divisions and do not belong to any community. Each university in a community plays more games with those in the same community. Lower division teams play only a few games. We transform this dataset into a graph containing 115 vertices and 787 edges. In this graph, 61 vertices can be regarded as outliers. Similar datasets have also been used in some of the previous work[19-20].

Cora Citation Network. This is a large-scale network about computer science research papers and their citations[30]. It consists of more than 17 000 vertices with about 77 000 edges representing the citation relationship. Papers in this network can be classified into 10 different research fields with 70 sub-categories.

4.2 Evaluation

The evaluation method we use is quite simple: metrics precision, recall and F-score, that are quite frequently used in many areas such as statistics, information retrieval and machine learning[31-33]. Some other papers focusing on community detection problem also adopt these metrics[19-20]. Precision is the fraction of correctly classified vertices in the community and recall is the ratio of the number of correctly classified vertices to the total number of vertices that should be agglomerated into the community. F-score is the harmonic mean of precision and recall:

$$F = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{6}$$

A well-performed algorithm should get high precision, recall and F-score at the same time.

4.3 Experimental Result

4.3.1 Results on Computer-Generated Graph

Results pertaining to precision, recall and F-score as a function of z_{out} on the computer-generated benchmark graph are shown in Fig.3. We perform over 500 realizations of the computer-generated graph. For the purpose of clarity, only data series with $z_{\text{out}} \leqslant 7.0$ are shown in the figure.

As the result shows, our algorithm with local modularity M achieves best in both precision and recall. The precision and recall of our algorithm are almost 1 when

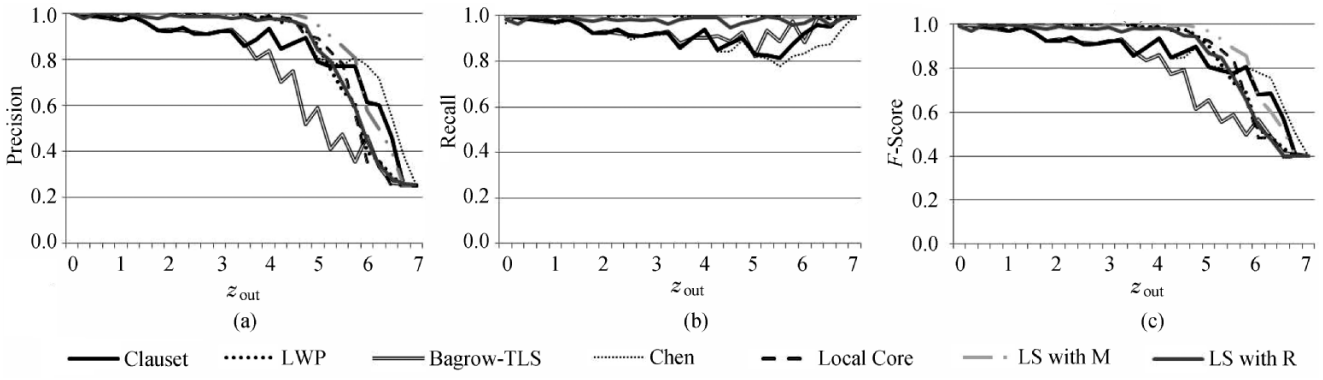


Fig.3. (a) Precision as a function of z_{out}. (b) Recall as a function of z_{out}. (c) F-score as a function of z_{out}.

z_{out} is less than 5. In other words, our algorithm can extract exactly the full community structure when z_{out} is no larger than 5. Meanwhile, our algorithm with local modularity R works relatively well but its performance is not quite stable. We also notice that all these algorithms are ineffective to detect community structure when z_{out} is larger than 6.

4.3.2 Results on NCAA Dataset

We next focus on results obtained on NCAA network dataset. Seven different methods listed in the beginning of this section are all performed on this dataset. Table 1 shows the precision, recall, F-score and number of outliers detected for all comparison methods. Fig.4 shows the chart comparing the results of different algorithms according to each of three evaluation metrics respectively.

Table 1. Results of Precision, Recall, F-Score and Number of Outliers Detected for NCAA Dataset

Method	Precision	Recall	F-Score	No. Outlier
Clauset	0.5123	0.8366	0.6267	0
LWP	0.6182	1.0000	0.7623	0
Bagrow-TLS	0.2632	0.8557	0.3756	0
Chen	0.7213	0.8175	0.7633	1
Local Core	0.4982	1.0000	0.6349	49
LS with M	0.9696	0.9924	0.9794	54
LS with R	0.9752	1.0000	0.9853	54

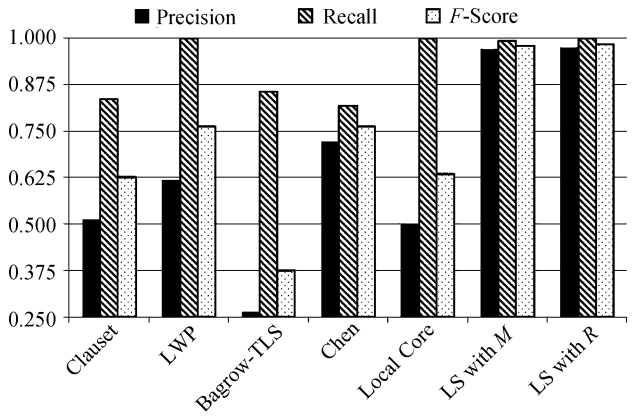


Fig.4. Comparison of precision and recall together with F-score for different methods on NCAA dataset.

When detecting the community structure in this dataset, our algorithm will give priority to the vertices sharing more links with the explored community. This process can help to find out a stable community structure as soon as possible. Consequently, a more integrated and precise community can be obtained. The trimming phase in our algorithm also guarantees that outliers in the community can be removed. As shown in Fig.4, our algorithm with either local modularity M or local modularity R has comparatively high value of precision, recall and F-score; meanwhile, 54 out of 61 outliers are detected by our algorithms. This experiment also points out the disadvantage of other existing methods. On the one hand, although Local Core method can effectively detect 49 out of 61 outliers, its precision is not ideal, leading to an unsatisfactory F-score. On the other hand, although LWP method can achieve best in recall value, its precision rate is not very good. Meanwhile, no outlier can be detected by this method. This result clearly demonstrates that LWP method ignores the confusion of outliers. The performance of Chen algorithm is relatively good; however, its strict policy in controlling community density does not result in a very high F-score.

4.3.3 Results on Cora Dataset

Experiments on Cora help convince the scalability of our algorithm. Although Cora provides us with sub-category labels, we cannot arbitrarily adopt these labels to evaluate the algorithms' effectiveness. This is because the clustering granularity using sub-category labels are still too large for the local community. An intuitive example is that papers on the topic about social network analysis are not likely to cite those focusing on sequential data analysis, although both topics are under the data mining sub-category.

To solve this problem, we try to manually check keywords of every paper in the discovered community since the keywords of a paper can directly represent its topic.

Due to lack of space, we present discovered local communities for only one paper (ID 66563), which is chosen as the source vertex. The keywords of this paper are *genetic algorithms* and *genetic programming*. Experimental results are shown in Table 2. The second column in the table shows the number of community members with keywords of *genetic algorithms* or *genetic programming*, while the third column presents the number of community members with other kinds of keywords.

Table 2. Algorithm Comparison on Cora Dataset

Method	"genetic algorithms" & "genetic programming"	Others
Clauset	7	0
LWP	71	4
Bagrow-TLS	23	0
Chen	7	0
Local Core	132	17
LS with M	75	1
LS with R	53	1

While all the algorithms detect useful local communities, our algorithm with local modularity M returns a relatively better community, since the size of the community is comparatively large and its members are generally focusing on the topic about *genetic algorithms* and *genetic programming*, which are the keywords of the source vertex.

5 Conclusions

Exploring local community structure is becoming an appealing problem since perfect knowledge of real-world network is usually inaccessible. In this paper, we proposed an improved method to detect local community structure. Our algorithm mainly takes advantage of link similarity between the vertex and the community. A greedy agglomeration phase, an optimization phase and a trimming phase are included in our algorithm. Compared with other multi-phase algorithms, our algorithm implements comparatively easier and stricter stopping criteria for the purpose of simplifying and optimizing our algorithm. Experimental results show that our algorithm can discover local community better than other existing methods both in computer-generated benchmark graphs and in real-world networks.

One possible direction for future research is to detect local community structure in directed graphs since this kind of graph contains potentially more useful information. To the best of our knowledge, most of existing approaches to exploring local community structure simply ignore edge direction so as to transform directed graphs to undirected graphs. For future work, we are planning to analyze local community structure in directed graph using similar approaches based on link similarity.

References

[1] Parthasarathy S, Ruan Y, Satuluri V. Community discovery in social networks: Applications, methods and emerging trends. In *Social Network Data Analytics*, Aggarwal C (ed.), 2011, pp.79-113.

[2] Tang W, Zhuang H, Tang J. Learning to infer social ties in large networks. In *Proc. the 2011 European Conf. Machine Learning and Knowledge Discovery in Databases*, Sept. 2011, Part 3, pp.381-397.

[3] Kumar R, Raghavan P, Rajagopalan S, Sivakumar D, Tompkins A, Upfal E. The web as a graph. In *Proc. the 19th ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, May 2000, pp.1-10.

[4] Tang J, Zhang J, Yao L, Li J, Zhang L, Su Z. Arnetminer: Extraction and mining of academic social networks. In *Proc. the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Aug. 2008, pp.990-998.

[5] Clauset A, Newman M, Moore C. Finding community structure in very large networks. *Physical review E*, 2004, 70(6): 066111.

[6] Rattigan M, Maier M, Jensen D. Graph clustering with network structure indices. In *Proc. the 24th International Conference on Machine Learning*, June 2007, pp.783-790.

[7] Von Luxburg U. A tutorial on spectral clustering. *Statistics and Computing*, 2007, 17(4): 395-416.

[8] Pan J J, Yang Q. Co-localization from labeled and unlabeled data using graph laplacian. In *Proc. the 20th Int. Joint Conf. Artificial Intelligence*, Jan. 2007, pp.2166-2171.

[9] Fortunato S. Community detection in graphs. *Physics Reports*, 2010, 486(3/5): 75-174.

[10] Su Z, Yang Q, Zhang H, Xu X, Hu Y. Correlation-based document clustering using web logs. In *Proc. the 34th Annual Hawaii Int. Conf. System Sciences*, 2001, Vol.5,, p.5022.

[11] Kriegel H, Kröger P, Zimek A. Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, 2009, 3(1), Article No.1.

[12] Liu N, Yang Q. Eigenrank: A ranking-oriented approach to collaborative filtering. In *Proc. the 31st Int. Conf. Research and Develop. Inform. Retrieval*, July 2008, pp.83-90.

[13] Hotho A, Jäschke R, Schmitz C, Stumme G. Information retrieval in folksonomies: Search and ranking. In *Proc. the 3rd European Conf. The Semantic Web: Research and Applications*, June 2006, pp.411-426.

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

[15] Bagrow J, Bollt E. Local method for detecting communities. *Physical Review E*, 2005, 72(4): 046108.

[16] Clauset A. Finding local community structure in networks. *Physical Review E*, 2005, 72(2): 026132.

[17] Luo F, Wang J, Promislow E. Exploring local community structures in large networks. *Web Intelligence and Agent Systems*, 2008, 6(4): 387-400.

[18] Bagrow J. Evaluating local community methods in networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008, 2008: P05001.

[19] Chen J, Zaïane O, Goebel R. Local community identification in social networks. In *Proc. the 2009 Int. Conf. Advances in Social Network Analysis and Mining*, July 2009, pp.237-242.

[20] Zhang X, Wang L, Li Y, Liang W. Extracting local community structure from local cores. In *Proc. the 16th Int. Conf. Database Systems for Advanced Applications*, April 2011, pp.287-298.

[21] Rosvall M, Bergstrom C. Maps of random walks on complex networks reveal community structure. *Proc. the National Academy of Sciences of the United States of America*, 2008, 105(4): 1118-1123.

[22] Radicchi F, Castellano C, Cecconi F, Loreto V, Parisi D. Defining and identifying communities in networks. *Proc. the National Academy of Sciences of the United States of America*, 2004, 101(9): 2658-2663.

[23] Andersen R. A local algorithm for finding dense subgraphs. *ACM Transactions on Algorithms*, 2010, 6(4), Article No. 60.

[24] Andersen R, Lang K. An algorithm for improving graph partitions. In *Proc. the 19th Annual ACM-SIAM Symposium on Discrete Algorithms*, Jan. 2008, pp.651-660.

[25] Andersen R, Lang K. Communities from seed sets. In *Proc. the 15th International Conference on World Wide Web*, May 2006, pp.223-232.

[26] Riedy J, Bader D, Jiang K, Pande P, Sharma R. Detecting communities from given seeds in social networks. Technical Report GT-CSE-11-01, Georgia Institute of Technology, Feb. 2011.

[27] Flake G, Lawrence S, Giles C. Efficient identification of Web communities. In *Proc. the 6th Int. Conf. Knowledge Discovery and Data Mining*, Aug. 2000, pp.150-160.

[28] Girvan M, Newman M. Community structure in social and biological networks. *Proc. the National Academy of Sciences of the United States of America*, 2002, 99(12): 7821-7826.

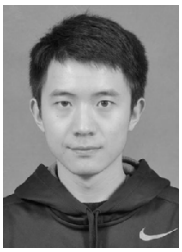
[29] Xu X, Yuruk N, Feng Z, Schweiger T. Scan: A structural clustering algorithm for networks. In *Proc. the 13th Int. Conf. Knowledge Discovery and Data Mining*, Aug. 2007, pp.824-833.

[30] McCallum A, Nigam K, Rennie J, Seymore K. Automating the construction of internet portals with machine learning. *Information Retrieval*, 2000, 3(2): 127-163.

[31] Lu Y, Zhang L, Liu J, Tian Q. Constructing concept lexica with small semantic gaps. *IEEE Transactions on Multimedia*, 2010, 12(4): 288-299.

[32] Yang J, Cai R, Wang Y, Zhu J, Zhang L, Ma W. Incorporating site-level knowledge to extract structured data from web forums. In *Proc. the 18th International Conference on World Wide Web*, April 2009, pp.181-190.

[33] Li X, Wang Y, Acero A. Learning query intent from regularized click graphs. In *Proc. the 31st Int. Conf. Research and Development in Information Retrieval*, July 2008, pp.339-346.



Ying-Jun Wu is an undergraduate student at School of Software Engineering, South China University of Technology, China. His research interests include machine learning, data mining, statistics and distributed computing.

Han Huang was born in 1980. He got his Ph.D. degree in computer application technology in 2008. He is an associate professor of the School of Software Engineering, South China University of Technology. He has over 60 publications in journals and conference proceedings. His research interests include theoretical foundation of evolutionary computation methods, evolutionary optimization, application of evolutionary algorithms, data mining, machine learning and intelligence computation, etc.

Zhi-Feng Hao was born in 1968. He got his Ph.D. degree in mathematics in 1995. He is a professor in the Faculty of Computer Science, Guangdong University of Technology. He has over 80 publications in journals and conference proceedings. His research interests are mainly in algebra, machine learning, bioinformatics and intelligence computation.

Feng Chen is an undergraduate student at School of Journalism and Communication, South China University of Technology. She is interested in the area of social network analysis.