An improved algorithm for finding community structure in networks with an application to IPv6 backbone network

GUO Yingxin XU Ke
State key Lab of Software Development Environment, Beihang University, Beijing 100083, China
{gyx, kexu}@ nlsde.buaa.edu.cn

Abstract The discovery of community structure in a large number of complex networks has attracted lots of interest in recent years. One category of algorithms for detecting community structure, the divisive algorithms, has been proposed and improved impressively. In this paper we propose an improved divisive algorithm, the basic idea of which is to take more than one parameters into consideration to describe the networks from different points of view. Although its basic idea appears to be a little simple, it is shown experimentally that it outperforms some other algorithms when applied to the networks with a relatively obscure community structure. We also demonstrate its effectiveness by applying it to IPv6 backbone network. The communities detected by our algorithm indicate that although underdeveloped compared with IPv4 network, IPv6 network has already exhibited a preliminary community structure. Moreover, our algorithm can be further extended and adapted in the future. In fact, it suggests a simple yet possibly efficient way to improve algorithms.

Keywords community, divisive algorithm, IPv6 network

1 Introduction

Evidence found in recent years reveals that some common properties exist among numerous real-world networks ranging from biological systems [1] to scientific collaboration systems [2]. Among the discovered important topological properties is the *community structure* which has attracted a great deal of interest in recent years.

Community structure is defined to describe the clustering or grouping characteristic of networks. A Network with community structure can be easily divided into several communities (groups) in which the nodes connect to each other more densely than with the rest of the network. However, to date there has been no strict definition on community although some well-accepted measurable parameters have been proposed to judge whether this type of structure exists and how typical it is in certain networks.

Algorithms focused on finding community structures have been discussed and have experienced remarkable improvements. Most of them may roughly fall into two categories. The first category of algorithms, agglomerative algorithms, starts with a graph with all the original nodes but no edges. Edges are added into the graph progressively based on their weights. In this way, communities, if exist in the networks, are gradually formed and ultimately all the edges in the networks are added into the graph. The other category of algorithms, divisive algorithms, is implemented in the opposite manner. It begins with the graph representing the whole network, and then removes the edges from the initial graph step by step based on the weights of the edges. During the process the networks are divided into several groups and communities are found.

The very famous GN algorithm is a divisive algorithm [1]. It was first discussed by Girvan and Newman in 2001 [1] and later improved [2]. Later, Tyler *et.al.* proposed a new algorithm based on the GN algorithm and applied the faster algorithm to compute large email network in 2003 [3]. But the increased efficiency comes at the expense of accuracy. Radicchi *et.al.* introduced another divisive algorithm based on edge clustering coefficient in 2004 [4]. The implementation of the algorithm involves only local variables and thus it runs much faster than the GN algorithm. In the same year, Fortunato *et.al.* proposed their algorithm based on information centrality [5] and demonstrated its effectiveness especially when the communities were very mixed and hardly detectable. In 2005, another new algorithm using extreme optimization was introduced by Duch *et.al.*[10]. This algorithm outperforms the existing algorithms in finding a higher modularity Q [2]. More recently in 2006, Newman proposed a spectral algorithm using the modularity matrix for detecting community structure in networks [11]. This newly proposed algorithm is capable of generating results with better quality than some other algorithms in much shorter running times.

In this paper we propose an improved divisive algorithm based on the GN algorithm and a clustering algorithm. This paper is organized as follows. Section 2 discusses two of the divisive algorithms, based on edge betweenness and edge clustering coefficient respectively. In Section 3 we propose our algorithm and demonstrate how it works through a specific and simple case in which it can give a more reasonable result than the two algorithms discussed in Section 2. In Section 4 the experimental results on computer generated networks are given from which we can see that our algorithm performs better than its predecessors when communities are hard to detect. In Section 5 we apply our algorithm to real-world networks, Zachary's Karate club network and the IPv6 network. Finally, we conclude the paper in Section 6.

2 Related work

In this section we review two relatively widely accepted algorithms. Both are divisive algorithms and thus are implemented in the following manner:

1) Start with a graph with all the nodes and edges in the original network.
2) Calculate the weight for each edge.
3) Remove the edge with the maximal (or minimal) weight from the graph.
4) If there are any edges in the graph, go back to step 2).

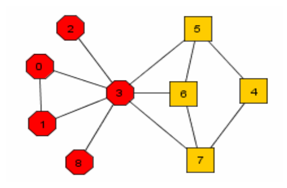
The two algorithms differ in that they take different parameters as the weight of the edges. Each will be discussed in turn.

2.1 Algorithm Based On Edge Betweenness

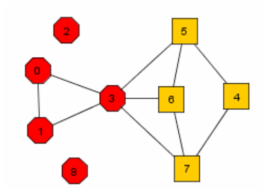
The GN algorithm, which takes the edge betweenness as the weight of edges, is the pioneer of divisive algorithms. The betweenness of an edge is defined as the number of shortest paths between pairs of nodes passing it. The algorithm to calculate edge betweenness is discussed in detail elsewhere [2].

Further studies illustrate that the performance of the GN algorithm is quite satisfactory when it is applied to the networks in which the community structure can be easily recognized. However, in the cases where communities are difficult to identify, it sometimes fails to generate a satisfactory result. One important factor is its tendency to split nodes connecting to the networks loosely in the early stages, leading to the final less reasonable community division.

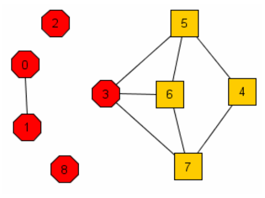
A detailed example is depicted in Fig. 1. In Fig. 1 (a), node 8 connects to the network loosely via a single edge, edge <3,8>. Thus the edge <3,8> has the maximal betweenness of all the edges. The same can be said of the edge <2,3>. Therefore, when the GN algorithm is applied to the network, as shown in Fig. 1 (b), node 2 and node 8 are split from the network in the first and second steps respectively, resulting in two communities with a single node each. After the removal of the two edges, node 3 becomes more densely connected with the right part of the network and is incorrectly classified. The final result is that 4 communities shown in Fig.1(c) are found, namely (2), (8), (0,1) and (3,4,5,6). This result is not as reasonable as expected.



(a) Original network



(b) After fist two steps



(c) Detected communities

Fig. 1 Application of the GN algorithm to a simple network. The shapes of nodes represent that they are expected to be in two different communities. The sub-figuration (b) and (c) demonstrate how the GN algorithm works in this case.

2.2 Algorithm Based On Edge Cluster Coefficient

The algorithm proposed by Radicchi *et.al.* [4] was initially aimed at solving the efficiency problem of the GN algorithm. It introduced the concept that the edge clustering coefficient could also be used as the weight of the edges. Formally, for the edge connecting node *i* to node *j*, the clustering coefficient is

$$C_{i,j}^{(3)} = \frac{Z_{i,j}^{(3)} + 1}{\min[k_i - 1, k_j - 1]}$$

where $Z_{i,j}^{(3)}$ is the number of triangles containing the edge, $\min[k_i\text{-}1, k_j\text{-}1]$ is the maximal possible number of triangles and k_i (k_j) is the degree of node *i* (node *j*). It is worth noticing that when a node connects to the networks via a single edge, $C_{i,j}^{(3)}$ is infinite because the denominator is zero. This algorithm is based on the assumption that within communities where connections are relatively dense, there are more triangles than those constructed by the edges between communities and thus the clustering coefficients of the edges connecting nodes within the same community is likely to be larger than that of the edges linking the nodes of different communities.

Considering the existence of high order cycles in the networks, the definition of the edge clustering coefficient in a wider sense can be stated as

$$C_{i,j}^{(g)} = \frac{Z_{i,j}^{(g)} + 1}{S_{i,j}^{(g)}}$$

where $Z_{i,j}^{(g)}$ is the number of cyclic structures of *g* order containing that edge and $S_{i,j}^{(g)}$ is the maximal possible number of cyclic structures.

Experiments show that this algorithm runs much faster than the GN algorithm, making the computation of some large networks possible. But its performance depends largely on the number of *g* order cycles in a given network. Also, when the edge clustering coefficients of several edges are equal, a common occurrence in the networks, the algorithm may choose one edge randomly, consequently leading to the less reasonable division.

Again, we use the network depicted in Fig. 1 (a) as an example. In this graph, when 3 is assigned to *g*, the edges <3,5>, <3,7>, <5,6>, <6.7>, <4,5> and <4,7> have exactly the same clustering coefficient, so it is difficult for the algorithm to choose the proper edge and reach a reasonable division.

Radicchi *et.al.* also introduced the quantitative definitions of community, on which we won't have a detailed discussion here. Details can be seen in [4].

3 Our algorithm for finding communities

In this section we propose our improved algorithm based on the algorithms discussed in Section II. What we learn from Section 2 is that the complexity of network structure indicates that taking only one parameter into consideration is occasionally insufficient because when the communities are difficult to detect, a single parameter sometimes fails to describe the information needed for further division. Thus, we improved an algorithm with the goal of using more than one parameter to describe the networks from more than one points of view. The fundamental assumption of our algorithm is that the edges within communities, while exhibiting a tendency towards lower betweenness, are likely to form a greater number of triangles and thus have higher clustering coefficients. On the other hand, the edges linking two different communities are likely to have higher betweenness as well as lower clustering coefficients. Therefore, the weight we set for each edge is

$$W = (B - \min B) * a_1 - (C - \min C) * a_2$$

where *B* and *C* are the betweenness and the clustering coefficient, respectively, of a given edge, and min*B* and min*C* are the minimal betweenness and the minimal clustering coefficient of all edges. Specifically, we calculate shortest-path edge betweenness by the algorithm proposed in [2] and use the definition of the edge clustering coefficient in [4] based on triangles (*g*=3).

The parameter a_1 is defined as

$$a_1 = \begin{cases} \dfrac{a}{\max B - \min B} & \text{if} \quad \max B \neq \min B \\ a & \text{if} \quad \max B = \min B \end{cases}$$

where max*B* is the maximal betweenness of all the edges while min*B* is the minimal betweenness.

Correspondingly, a_2 is defined as

$$a_2 = \begin{cases} \dfrac{1-a}{\max C - \min C} & \text{if} \quad \max C \neq \min C \\ 1-a & \text{if} \quad \max C = \min C \end{cases}$$

where max*C* is the maximal clustering coefficient of all the edges while min$C$ is the minimal clustering coefficient. The parameter a is a decimal variable ranging from 0 to 1. By adjusting the value of a we may alter the contributions of betweenness and the clustering coefficient to the weight of edges. In extreme cases, assigning 0 to a means that only the edge clustering coefficient is taken into consideration during the division process, and edge betweenness is the only parameter taken into consideration when a is set to 1. The influence of a in different networks will be shown in our experiments in Section 4.

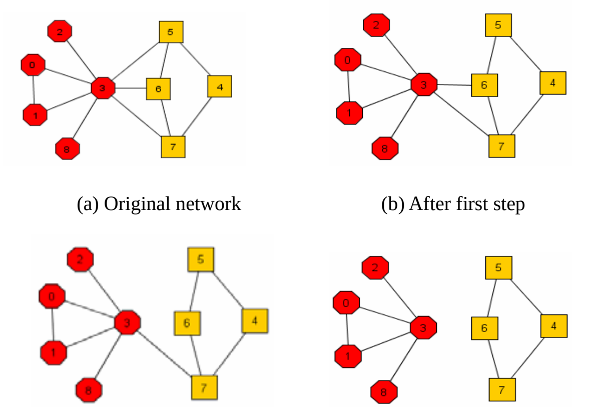
Now we give the working process of our algorithm:

1) Set a to a specific decimal ranging from 0 to 1.
2) Begin the graph with all the nodes and edges in the network
3) Compute the edge betweenness and clustering coefficient for all edges.
4) Compute a_1 and a_2.
5) For every edge calculate W according to a_1, a_2, B and C.
6) Remove the edge with the highest W.
7) If there are any edges in the graph, go back to step 3).

We can see how our algorithm manages in giving better results in certain networks. First, it performs better than the GN algorithm by avoiding circumstances in which a single node is separated at an early stage and classified as a community with only one node, which will affect the reasonable classification of other nodes in the later stages. The underlying reason for this avoidance is that if a specific node connects to the networks via a single edge, it must have the maximal clustering coefficient and thus a relatively small W. Therefore, it is less likely to be separated at an early stage. Secondly, compared with the algorithm that only takes edge clustering coefficients into consideration, our algorithm performs better because there will be fewer cases in which a random choice has to be made. To make it clear, when several edges have the same clustering coefficient, it is not very likely that they possess the same betweenness and W at the same time. Therefore, by choosing the edge with the greatest value of W, we choose an edge that is more likely to be an edge between different communities and thus are more likely to reach a satisfactory result. In summary, we account for the limitations of the algorithms discussed in Section 2 by taking a more complete view of each edge from the graph and then choosing the edge which is more likely to be an edge linking different communities based on this more complete view.

Now we look at how our algorithm gives a more reasonable result for the networks depicted in Fig. 1 (a) and Fig. 2 (a). Section 2 demonstrated that the GN algorithm and the algorithm based only on the edge clustering coefficient fail to give satisfactory results. In the following demonstration, it will be shown that for the same network, our algorithm is capable of reaching a more reasonable result. In this case we assign 0.5 to the parameter a. In the first step, as depicted in Fig. 2 (b), it chooses edge < 3, 5> with the highest W because of its high rank in edge betweenness and low edge clustering coefficient at the same time. Then, after recalculation, though edges <3,6> <3,7>, <5,6>, <6,7>, <4,5>, <4,7> and <3,8> have the same clustering coefficient, the betweenness of edges <3,6> and <3,5> is much higher than that of the other edges. Therefore, the algorithm removes edge <3,5> from the graph, resulting in the network shown in Fig. 2 (c). Finally, it removes edge <3,7>, which has the maximal betweenness and the minimal clustering coefficient, and the whole network is split into two communities shown in Fig. 2 (d), namely (0, 1, 2, 3, 8) and (4, 5, 6, 7) respectively.

The effectiveness of the improved algorithm on larger


(a) Original network (b) After first step

(c)After second step (d) Detected communities

Fig. 2 Application of our improved algorithm to a simple network. The shapes of nodes represent that they are expected to be divided into two different communities. The sub-figurations demonstrate how the algorithm works.

networks will be shown in further experiments in Section 4 and Section 5.

Every time an edge is removed from a network, W is recalculated for each edge, involving calculating the shortest-path betweenness and the clustering coefficient at the same time. Therefore, the total time complexity of our algorithm is O (m^2n) where m is the number of the nodes in the network and n is the number of edges, which is equal to that of the GN algorithm.

4 Testing on computer generated networks

We now apply our algorithm to computer generated networks. Random graphs are constructed in such a manner that they have a pre-defined community structure. All graphs have 128 nodes with an average degree expectation of 16. The nodes are divided into 4 communities, which respectively contain nodes 0-31, 32-63, 64-95 and 96-127. By carefully adjusting the parameter P_{out} with which pairs of nodes belonging to different communities connect to each other, we alter how strong the community structure is in generated graphs. This is the same way to generate graph previously used by Newman [1]. To avoid randomness, for every P_{out} we generate 100 graphs and apply our algorithm (with the parameter a set to 0.5) as well as the GN algorithm and the clustering algorithm to them.

The experimental results are displayed in Fig. 3. The plots in Fig.3 (a) indicate the fractions of correct divisions, the divisions which are exactly the same as pre-defined, in every 100 graphs with the same P_{out}. The plots in Fig.3 (b) illustrate for each fixed P_{out}, the average proportions of nodes which are correctly classified in the pre-defined communities in every 100 generated graphs.

We can see from the results demonstrated in Fig. 3 (a) and (b) that when P_{out} is small, indicating a strong community structure in networks, our algorithm performs as well as the other two algorithms. But as P_{out} increases and communities become difficult to detect, our algorithm outperforms the other two algorithms.

Finally, we used modularity Q [2] to measure the divisions. The modularity is defined as

$$Q = \sum_i (e_{ii} - a_i^2)$$

where e_{ii} denotes the number of edges that fall in the community i, and a_i denotes the number of edges that link to the nodes in community i. If the edges in a network lie no better than randomly, the value of Q is close to 0. On the other hand, values close to 1, which is the maximum, indicate a strong community structure in networks. For a given network, the higher the resultant Q is, the more reasonable the division is in some sense. In fact, this measurable parameter Q provides such a sensible and also well-accepted way to qualify divisions that some algorithms, such as algorithm using extreme optimization [10] and modularity matrix [11], are particularly focused on looking for the divisions with higher resultant Q values.

The resultant values of Q in our experiment are shown in Fig. 3 (c). As P_{out} increases and the community structure becomes obscure and fuzzy, our algorithm is capable of giving divisions with higher values of Q, indicating the divisions given by our algorithms are more reasonable, though the divisions may not be exactly the same as expected.

In the process of division, the parameter a plays a key role, deciding the weight value of each edge and the order of edges to be removed. Therefore, we also have performed some preliminary experiments to determine how the parameter a influences the performance of our algorithm. The result shown in Fig. 4 demonstrates that the difference between the performances given by the three algorithms becomes more apparent as P_{out} increases. Especially when a is adjusted properly, our algorithm performs better than its antecedents, demonstrating both a higher possibility of accurately matching the expected divisions and a greater proportion of correctly classified nodes. More specifically, peaks indicating the maximal fractions of correct divisions in Fig. 4 (a) usually take place where a is set between 0.3 and 0.7, and the peaks denoting the maximal factions of nodes classified correctly in Fig. 4 (b) and the maximal average Q values in Fig. 4 (c) usually occur when a is set to 0.3. This indicates that betweenness and clustering coefficient play a comparatively important role in describing the structure of networks, at least for computer-generated networks.

Finally, some experiments have been done to evaluate the effect of network size on the performance of our algorithm. We changed the computer generated networks from 64 nodes to 128 nodes, then to 256 nodes, and the average degree expectation from 8 to 16, and finally 32. The results of these experiments are shown in Fig. 5, indicating that while the networks are larger, our algorithm exhibits even better performance in higher possibility of ever finding correct divisions compared to the other two algorithms.

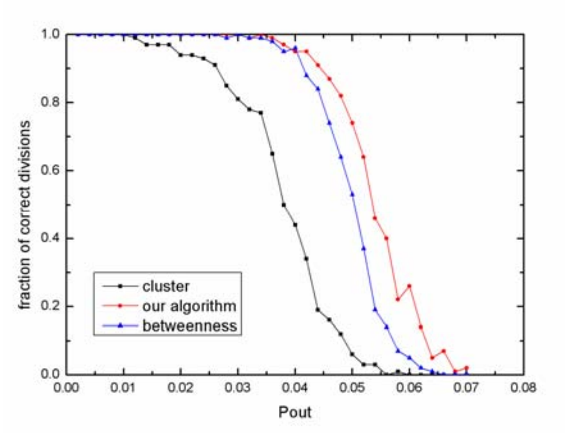
In summary, the experiments show that in computer generated graphs, as the network becomes larger and the community structure becomes fuzzy, the relative performance of our algorithm compared to the other two algorithms increases. The reason for this is, as discussed in Section 2, obtaining a reasonable community division in a more complex network requires describing the network in a more complete way, which is exactly the aim of our algorithm. It is also shown experimentally that in order to reach a satisfactory result, the parameter a should be set between 0.3 to 0.7.

5 Application to real networks

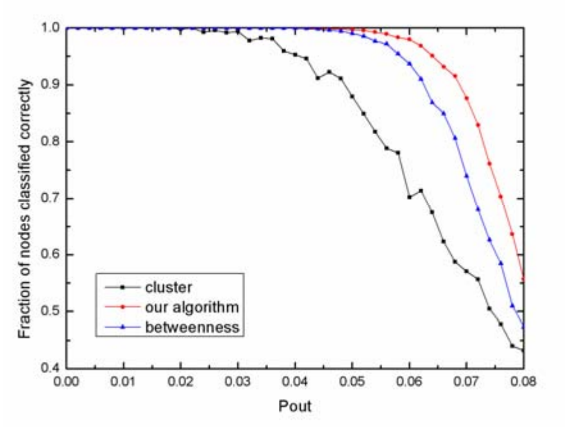
In this section we apply our algorithm to two real-world networks. One is the classic Zachary club network and the other is the IPv6 backbone network.

5.1 Zachary's Karate club network

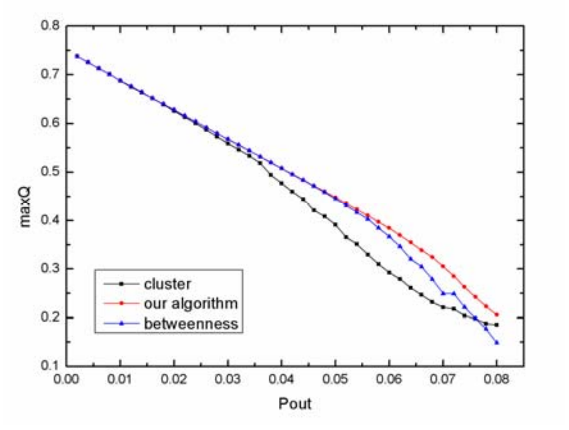
The Zachary's Karate club network is drawn from the well-known Karate club study of Zachary W W [9]. Zachary observed 34 members of a karate club over a period of 2 years, which was later separated for a reason. Zachary constructed a network of friendships between members of the



(a)

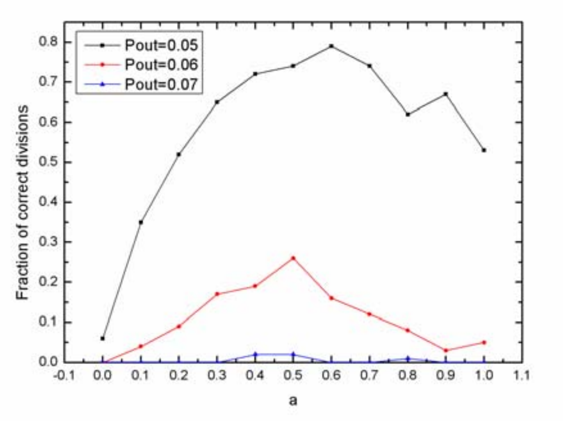


(b)

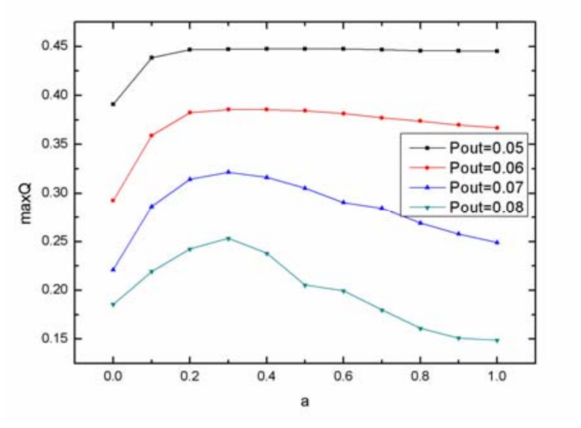


(c)

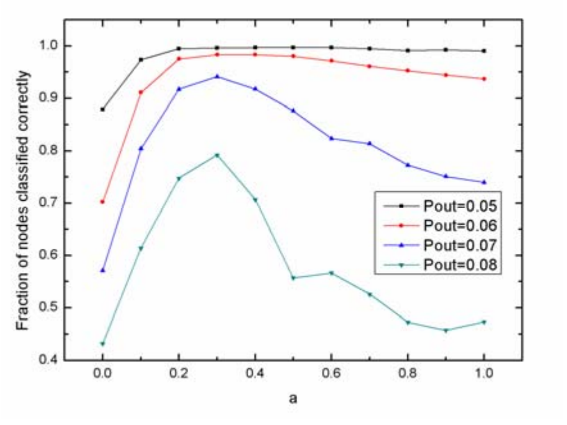
Fig. 3 Test of three algorithms on computer generated graphs with pre-defined community structure. The fractions of correct divisions and of nodes classified correctly are shown and compared in (a) and (b) respectively. The resultant maximal values of Q in the test are shown and compared in (c).



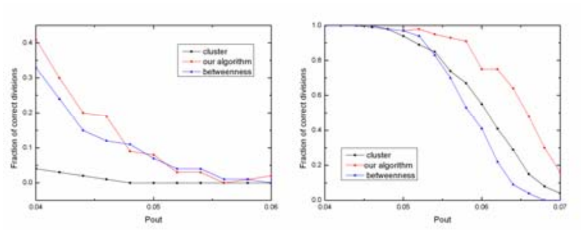
(a)



(c)



(b)



(a) 64 nodes (b) 256 nodes

Fig. 5 Test of three algorithms on computer generated graphs with different size. The sub figuration (a) shows the comparison of the factions of correct divisions when applying the three algorithms to the computer generated graphs with 64 nodes. The sub figuration (b) illustrates the similar comparison on graphs with 256 nodes

Fig. 4 Test of influence of the parameter *a* on computer generated graphs with a pre-defined community structure. Each plot in (a) represents the fractions of correct divisions with the variation of parameter *a* when P_{out} is set constantly. Correspondingly, each plot in (b) represents the fractions of nodes classified correctly and each plot in (c) represents the resultant average values of Q during the course of division with the variation of parameter *a* when P_{out} is fixed.

club and this network has been used for algorithm testing in many papers. A consensus community structure of Zachary's club is shown in Fig. 6.

We apply our algorithm to the network and the change in *Q* in the process of the division is shown in Fig 7.

In the first step, the network splits into two groups with a local maximum of *Q* close to 0.4, indicating a strong natural division at this point. The result given by our algorithm only classifies one node, which is node 10, incorrectly. This inaccurate classification is primarily due to the amphiboly of the node (in that it connects to each of the two groups via a

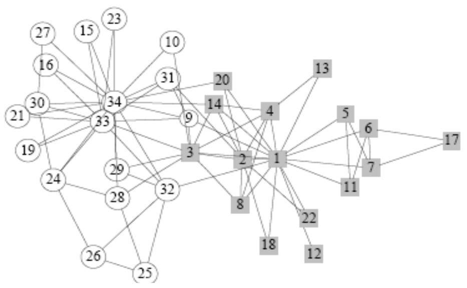


Fig.6 Taken from Girvan and Newman [1]. A node in the network represents an individual and a link means a connection between the two individuals. A consensus community structure is shown.

single respective edge). Therefore, in this case our algorithm performs well, giving a reasonable division.

5.2 IPv6 backbone network

We perform an experiment of our algorithm using the data from the IPv6 network. Beginning in 2004 we used our topology discovery system, named *Dolphin* [6] and based on distributed network probing through IPv6-in-IPv4 tunnels, to discover the IPv6 backbone topology. Dolphin has collected a large amount of data at both the *AS* (Autonomous System) level and router level. Until Oct 2006, 6325 routers, 51115 links and 506 ASes have been discovered by Dolphin.

CAIDA [7] has also run a project entitled "Macroscopic IPv6 Topology Measurements", which performs a study similar to that of our Dolphin system. According to the published data at present (2005/3/16) [8], they collected 2913 addresses and 7905 links from 333 ASes. Compared to the CAIDA project, Dolphin succeeds in making a more complete description of the IPv6 topology.

Based on the data collected by Dolphin, we created a graph in which each node represents an AS and an edge linking two nodes indicates a link between the two ASes. This graph is power law in degree with an exponent of 1.273.

In order to obtain a more reasonable division, firstly we

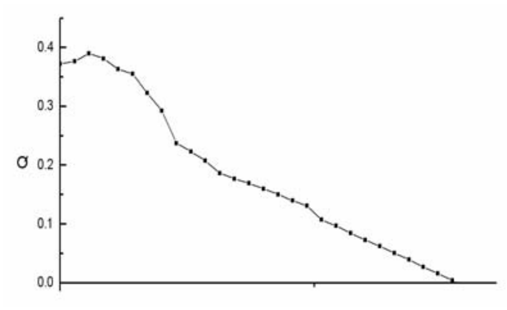


Fig. 7 The resultant values of Q in the process of division when applying our algorithm to the Zachary's Karate club network.

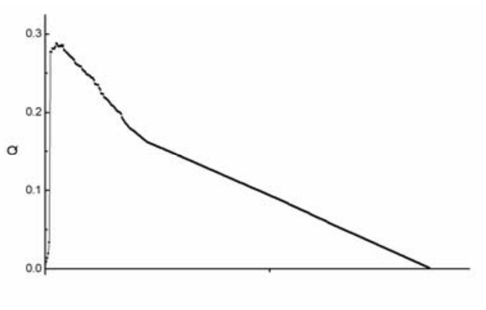


Fig. 9 The resultant values of Q in the process of division when applying our algorithm to the IPv6 network.

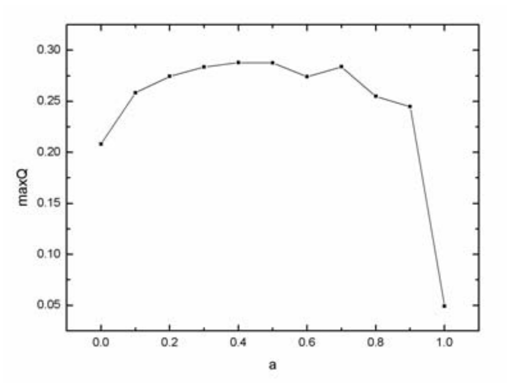


Fig. 8 The resultant value of Q with the variation of a when applying our algorithm to IPv6 network

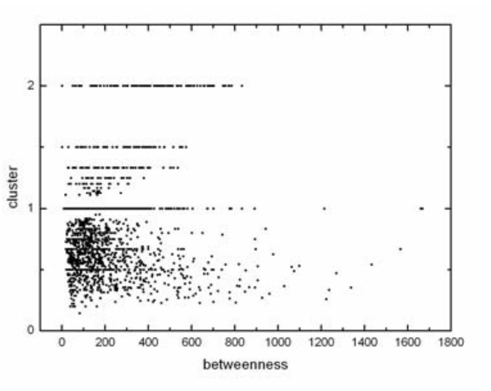


Fig. 10 The resultant value of Q when applying our algorithm with different value of a to IPv6 network

Table 1 Comparison of ASes, addresses and links discovered by Dolphin and CAIDA

	Dolphin (2006/10)	**Dolphin** (2005/4)	**CAIDA** (2005/4)
ASes	506*	346	333
Addresses	7401	3022	2913
Links	51115	6825	7905

test our algorithm with different values of *a*. The resultant *Q* with the variation of *a* is shown in Fig. 8. We can see that when *a* is set to 0.4, the maximal value of *Q* is reached. Moreover, when we apply the clustering algorithm and the GN algorithm to the network by assigning 0.0 and 1.0 to the parameter *a* respectively, the resultant values of *Q* are relatively low, indicating that the two algorithms perform comparatively poorly in the case of IPv6 network.

For the IPv6 network, 13 communities were detected by our algorithm, with *a* set to 0.4. The change of *Q* in the process of the division is shown in Fig. 9. We can see from Fig. 9 that there is an obvious peak close to 0.3, indicating a reasonable division at this point, where 13 communities are detected. The largest one consists of 127 ASes, while the smallest one only consists of 4 ASes. We demonstrate the division in Table 2 in such a way that ASes in each community are classified according to their geography information.

5.3 Discussion

What we can see from the result is that although IPv6 network is relatively underdeveloped when compared with IPv4 network, the community structure has already emerged from it. The algorithms for detecting community structure perform differently in the case of IPv6 network primarily due to its special structure. As a newly developed network still in an experimental stage, IPv6 network consists of a large number of small ASes connected to the network loosely via one or two large ASes, and a small number of ASes connected to each other as the borders of communities (i.e., some large experimental platforms are connected to many IPv6 ASes while many small ASes are only linked to one or two of these large experimental platforms). The structure is similar to the situation discussed in Section 2 in which the GN algorithm sometimes fails to give a reasonable division due to the tendency to split loosely connected ASes in the early stages of division. This tendency further affects the way other ASes are connected to the networks. The algorithm based only on edge clustering coefficients does not perform as well as our algorithm because it takes too much random actions due to the lack of other information. From Fig. 10, showing the distribution of edges according to their clustering coefficients and betweenness in the original network, we can see that in the IPv6 network there are many edges with the same clustering coefficient and therefore there is some difficulty in deciding which edges should be removed only based on clustering coefficients.

6 Conclusion

In this paper we presented an improved algorithm based on the GN algorithm and a clustering algorithm for finding community structure in complex networks. It calculates the weight for each edge according to both the edge betweenness and the clustering coefficient, and adjusts the contribution of these two parameters via a parameter *a*. This major improvement made by our algorithm is the accuracy in reaching a more satisfactory result in the cases where the

* Among the 506 detected ASes, there are 98 isolated due to certain detecting reasons.

Table 2 The communities found by our algorithm in the IPv6 network. As: Asia Eu:Europe SA:South America Af :Africa NA:North American Au:Australia Un:unknown

Com	**As**	**Eu**	**SA**	**NA**	**Af**	**Au**	**Un**	**total**
A	0	0	0	2	0	0	2	4
B	0	4	0	0	0	0	0	4
C	0	2	0	0	0	0	2	4
D	0	4	0	0	0	0	1	5
E	0	4	0	0	0	0	1	5
F	2	11	0	0	0	0	0	13
G	9	0	1	7	0	0	0	17
H	21	0	0	0	0	1	0	22
I	2	9	0	8	0	2	3	24
J	0	23	0	1	0	0	5	29
K	0	54	0	9	1	0	11	75
L	4	54	0	12	1	0	10	81
M	59	17	4	39	0	4	4	127
total	97	182	5	78	2	7	37	408

structure of a network is complex and communities are not obvious, though its time complexity remains the same as the GN algorithm. Therefore it is suitable for the cases where the accuracy rather than the time consuming is the major consideration.

We have demonstrated the experimental results of our algorithm and used the modularity Q to evaluate the performance of different algorithms. It is shown that our algorithm performs as well as and sometimes even better than the other two algorithms in computer-generated graphs when a is properly set.

Finally, we demonstrated the effectiveness of our algorithm by applying it to real-world networks especially IPv6 network. By assigning a proper value to a, 13 communities were detected with an obvious peak close to 0.3 in the change of modularity Q, indicating a natural division at this point. Though relatively underdeveloped when compared with IPv4 network, IPv6 network has already exhibited a preliminary community structure.

Although the basic idea of our algorithm, to take more than one parameters into consideration, seems to be a little simple, it can lead to satisfying results. And our algorithm motivated by this idea can be further extended and adapted in the future. Moreover, the idea itself has a wider significance in that it suggests a simple yet possibly efficient way to improve algorithms.

In future work, we hope that our algorithm can be used to find community structure for more complex networks. Besides, we will keep an eye on the development of community structure in the IPv6 network as it progresses in the future.

7 Acknowledgement

This work is supported by National 973 Program of China (Grant No. 2005CB321901) and Beijing Nova Program (Grant No. 2005B12).We would like to thank Prof. Newman for useful information about network visualization.

References

1. Girvan M, Newman M E J. Community structure in social and biological networks. Proc. Natl. Acad. Sci.,2001, 99, 7821-7826
2. Newman M E J, Girvan M. Finding and evaluating community structure in networks. Phys. Rev. E ,2004,69,026113
3. Tyler J, Wilkinson D, Huberman B. Email as spectroscopy: Auto discovery of community structure within organizations. International Conference on Communities and Technologies, 2003, 81-96
4. Radicchi F, Castellano C, Cecconi F, Loreto V, Parisi D. Defining and identifying communities in networks. Proc. Natl. Acad. Sci. 2004,101,2658-2663
5. Fortunato S, Latora V, Marchiori M. A method to find community structures based on information centrality. Phys. Rev. E. 2004, 70, 056104
6. http://nlsde.buaa.edu.cn/dolphin/
7. CAIDA, Visualizing IPv6 AS-level Internet Topology, http://www.caida.org
8. http://www.caida.org/analysis/topology/as_core_network/ipv6.xml
9. Zachary W W. An information flow model for conflict and fission in small groups. Journal of Anthropological Research,1997, 33, 452-473
10. Duch J, Arenas A. Community detection in complex networks using extreme optimization. Physical Review E 2005,72,027104
11. Newman M E J. Modularity and community structure in networks. Proc. Natl. Acad. Sci. 2006,103,8577-8582