Identification and Evaluation of Weak Community Structures in Networks

Jianhua Ruan and **Weixiong Zhang**
Department of Computer Science and Engineering
Washington University
St. Louis, MO 63130
Email: {jruan, zhang}@cse.wustl.edu

Abstract

Identifying intrinsic structures in large networks is a fundamental problem in many fields, such as engineering, social science and biology. In this paper, we are concerned with communities, which are densely connected sub-graphs in a network, and address two critical issues for finding community structures from large experimental data. First, most existing network clustering methods assume sparse networks and networks with strong community structures. In contrast, we consider sparse and dense networks with weak community structures. We introduce a set of simple operations that capture local neighborhood information of a node to identify weak communities. Second, we consider the issue of automatically determining the most appropriate number of communities, a crucial problem for all clustering methods. This requires to properly evaluate the quality of community structures. Built atop a function for network cluster evaluation by Newman and Girvan, we extend their work to weighted graphs. We have evaluated our methods on many networks of known structures, and applied them to analyze a collaboration network and a genetic network. The results showed that our methods can find superb community structures and correct numbers of communities. Comparing to the existing approaches, our methods performed significantly better on networks with weak community structures and equally well on networks with strong community structures.

Introduction and Overview

Complex networks have drawn much interest lately in many different domains, ranging from engineering, social science to biological studies (Newman 2003). In a framework of network analysis, a system is modeled as a graph, in which the nodes are the elements of the system (e.g. the individuals in a society), and the edges represent the interactions, links, or similarities between pairs of elements. One of the key problems that attracted a great deal of interest recently is the identification of the so-called community structure, a relatively densely connected sub-graph. Identification of such structures is fundamentally important for understanding the dynamics and design principles of complex systems.

Identifying community structures in a network amounts to clustering nodes into groups. Clustering algorithms have been proposed in diverse areas, including data mining, VLSI design, social networks, and bioinformatics, as surveyed in (Jain, Murty, & Flynn 1999). Many of these algorithms are not designed specifically for clustering networks, and make strong assumptions of the statistical or topological properties of the clusters (e.g., Gaussian distribution and spherical shapes). When experimental data do not agree well with these assumptions, these methods often fail. In addition, determining the right number of clusters is difficult in general and requires deep insight to the network of interest. A few ideas have been proposed for this problem with limited success (Chan, Schlag, & Zien 1993).

Recently, Newman & Girvan (2004) proposed a network clustering algorithm that considers network topologies explicitly. Their method is based on the concept of edge betweenness centrality, which measures how likely that an edge connects two nodes in two communities rather than within the same community. The algorithm is a divisive hierarchy clustering, which iteratively removes the edge with the highest betweenness and then adjusts the betweenness scores of the remaining edges. Furthermore, they proposed a modularity function, Q, to quantify the strength of community structures (Newman & Girvan 2004). They empirically showed that high Q values are often correlated with high-quality clusters for both computer-generated and real-world networks. Therefore, their method can potentially be used to automatically determine the number of clusters.

Even though the Newman & Girvan method has been successful on a variety of networks, it requires a substantial amount of computation, due to the recalculation of betweenness scores after each edge removal. The algorithm runs in $O(m^2n)$ time on arbitrary networks and $O(n^3)$ time on sparse networks with m edges and n nodes. Newman (2004) proposed a faster algorithm based on a greedy optimization of the modularity function Q that runs in $O((m+n)n)$ time for arbitrary networks and $O(n^2)$ time on sparse graphs. White & Smyth (2005) related a relaxed optimization of Q function with spectral clustering, and proposed an algorithm that works almost as well as the Newman & Girvan method but with a smaller running time of $O(hn)$ for sparse graphs, with h being the number of iterations.

We make two contributions in this paper. First, most existing network clustering methods assume sparse networks and networks with strong community structures, i.e., there are more intra-community edges than inter-community edges. In contrast, we consider both sparse and dense networks that have weak community structures, which are often resulted from noise and errors in experiments (e.g. in genetic interaction network). We introduce a set of simple operations to capture local neighborhood information of a node. Combined with spectral clustering, our method can identify weak community structures with significantly improved accuracies. Furthermore, our method is very efficient, having the same running time as the White & Smyth method.

Second, we address the issue of automatically determining the most appropriate number of communities in weighted dense graphs. Under a good community structure in weighted networks, intra-community edges tend to have higher weights (or shorter distances) than inter-community ones. Applying the modularity function Q to such community structures, however, often results in very low Q values. Although it is still possible to select the number of clusters using the Q measure, it does not shed light on the quality of community structures. We propose a simple extension to estimate the Q function on weighted graphs by a rank-based transformation of edge weights that can produce much meaningful results.

The paper is organized as follows. We first investigate local neighborhoods in real-world networks and propose two local operations, and discuss a modularity function for choosing the number of clusters. We then present the overall algorithm and extensively evaluate our methods on various networks of known structures, and apply them to a real-world collaboration network and a genetic network. We conclude with some final remarks.

Local Structures

Real-world networks often possess intrinsic properties that are lacked in random graphs, such as power-law distributions of node degrees (Newman 2003). In particular, real-world networks often have surprisingly higher clustering coefficients than random graphs (Newman 2003). The clustering coefficient c is defined as $c = \frac{3\times(\text{number of triangles in the graph})}{(\text{number of connected triples})}$, where a "connected triple" means a path of three nodes. This coefficient is related to node transitivity, i.e., two nodes connecting to a third node are likely to be directly connected. In terms of social networks, this means that a friend of your friend is likely to be your friend as well. In fact, c is the probability that your friend's friend is also your friend. For most real-world networks, $0.1 \leq c \leq 0.5$, while for random networks of n nodes, $\lim_{n\to\infty} c = 0$ (Newman 2003).

Let $G = (V, E)$ be a network or graph , where V is the set of n nodes and E the set of m edges. Let $A = (A_{ij})$ be the adjacency matrix of G, i.e., $A_{ij} = 1$ if $(v_i, v_j) \in E$, or 0 otherwise. Let D be the diagonal degree matrix of A, where $D_{ii} = \sum_j A_{ij}$. We then define a matrix $\hat{B} = A \times A \cdot (1 - I)$, and $\hat{C} = A \times A \cdot A$, where I is an identity matrix, "$\times$" represents ordinary matrix multiplication and "$\cdot$" means entry-wise multiplication. It is evident that $\hat{B}_{ij} = \sum_k A_{ik}A_{kj}$ if $i \neq j$, or 0 otherwise, and $\hat{C}_{ij} = A_{ij}\hat{B}_{ij}$. Therefore, $\hat{B}_{ij}$ is the number of common neighbors shared by nodes v_i and v_j, which is also the number of paths of length two between them. So the sum of all entries in $\hat{B}$, $||\hat{B}|| = 2 \times$ (number of connected triplets). Similarly, $\hat{C}_{ij}$ is the number of common neighbors of nodes v_i and v_j if they are directly connected, and 0 otherwise. In other words, $\hat{C}_{ij}$ is the number of triangles that contain edge (v_i, v_j). Therefore, $||\hat{C}|| = 6 \times$ (number of triangles in the network), and the clustering coefficient can be calculated as $c = ||\hat{C}||/||\hat{B}||$.

The above transitivity property indicates that two nodes with many common neighbors tend to be in the same community. Therefore, the number of triangles passing along an edge, normalized by the probability that this may happen by chance, can be used to weight the edge. On the other hand, if two nodes are both connected to many common neighbors, regardless whether there is a direct edge between them, they have a higher chance to be in the same community than random. Therefore, we define two normalized matrices:

$$B = D^{-\frac{1}{2}} \times \hat{B} \times D^{-\frac{1}{2}}/S_b; \quad (1)$$

$$C = D^{-\frac{1}{2}} \times \hat{C} \times D^{-\frac{1}{2}}/S_c, \quad (2)$$

where S_b and S_c are scaling factors such that the values in B and C are within [0, 1]. Note that $B_{ij} = \hat{B}_{ij}/S_b\sqrt{D_{ii}D_{jj}}$ and $C_{ij} = \hat{C}_{ij}/S_c\sqrt{D_{ii}D_{jj}}$. The square root in the denominators gives relatively higher weights to node pairs that share more common neighbors.

Both B and C can be considered as adjacency matrices of some weighted graphs transformed from the original graph A. C is a weighted subgraph of A, where the edges belonging to more triangles gain higher weights. Thus, comparing to A, intra-community edges in C have increased weights while inter-community edges have reduced weights. The edges not in any triangle are simply removed. This may cause some problem if the intra-community edges are sparse, since in this case a community may be broken into several disconnected components. As a remedy we can combine A and C, which brings the edges in A back to the graph. The relationship between A and B is more complex, in that B contains all the edges in C, as well as some additional edges that may or may not be in A. In general, B is much denser than A or C. If the original graph is sparse, the added edges due to similar local neighborhood structures may provide additional information of communities. Therefore, we consider a combination of the three:

$$H = (\alpha \times A) + (\beta \times B) + C, \quad (3)$$

where α and β are scalars. Ideally, small α and β are preferred for dense graphs or graphs with high clustering coefficients, and large α and β should be used for sparse graphs. As to be seen in the experiment section, we find that simply taking $\alpha = \beta = 1$ is sufficient for most cases that we studied.

Although the above discussion is for unweighted networks, Equations (1) - (3) can be directly applied to weighted graphs, where A_{ij} is a positive weight for an edge between nodes v_i and v_j. In the special case where a network is a weighted complete graph, we choose $\alpha = \beta = 0$ since C would not remove any edge from A, and empirically it turns out to be better than any other combination in our study (see Experimental Results).

Modularity and Number of Clusters

Given a clustering Γ_k of a graph that partitions its nodes into k groups, the modularity Q of Γ_k is defined as

$$Q(\Gamma_k) = \sum_{i=1}^{k}(e_{ii} - a_i^2), \tag{4}$$

where e_{ii} is the fraction of the edges that fall within cluster i, and a_i the fraction of edges each of which has at least one end connected to a node in cluster i (Newman & Girvan 2004). The Q function is conceptually intuitive: It measures the edge density within a cluster, subtracted by the density that one would expect by chance, and sums all such differences over all clusters. If a particular partitioning gives no more intra-community edges than would be expected by chance, the modularity $Q = 0$. For a trivial clustering with a single community, Q is always equal to zero.

A nice feature of modularity Q is that it provides a global quality measurement of community structures for a network. It has been found that most real-world networks have $Q > 0.3$, which was suggested as a threshold for good community structures (Newman & Girvan 2004). We have also found that networks with $Q > 0.3$ are relatively easy to cluster in that most existing algorithms will return good clustering results; while the clustering quality of most clustering algorithms decreases dramatically when the Q value is below 0.3 (see Experimental Results).

The definition of Q can be generalized to weighted networks by extending e_{ii} and a_i to corresponding fractions of edge weights. However, the generalization is only meaningful for sparse networks, while weighted networks in real applications are often dense or even complete graphs. A weighted network is usually derived from similarity scores between pairs of nodes, which can be computed for any pair of nodes, resulting in a dense or complete graph. On such networks, the Q function often fails to produce meaningful results as we will see in Experimental Results.

A simple way to fix this problem is to use a weight threshold to remove some edges. However, without knowing the community structure of a network, it is difficult to choose the right threshold. Furthermore, it is always possible to use a high threshold to break a network into small disconnected components, so as to obtain a high Q value, whereas the resulting clustering may not be meaningful for the original network. Therefore, maximizing Q is not a good criterion.

Here we propose a method to determine a weight threshold for edge removal so that meaningful community structures can be revealed, and the corresponding Q value can be used to unbiasedly compare different clustering results. The intuition of our method is as following. Since a community is a set of nodes that are highly connected among themselves but only loosely to the rest of the network, we can choose a threshold to remove low-weight edges so that, in the ideal case, the number of remaining ones would be just enough to form completely connected communities with no inter-community connections. Therefore, for a perfect clustering, the weights of the intra-community edges should be all higher than that of inter-community edges.

Based on this insight, the edge-weight threshold can be uniquely determined as follows. Suppose a clustering method returns a partition $\Gamma_k = \{P_1, P_2, \cdots, P_k\}$ on a weighted graph. The number of edges needed to completely connect the intra-cluster node pairs is $s = \sum_i(|P_i| \times (|P_i| - 1)/2)$. We sort all the edges in the network in a non-increasing order of their weights. We then set the weights of the first s edges to 1, and discard the other edges. In case that the s-th edge is tied with other edges on weights, we remove all or none of them to keep the number of remaining edges as close to s as possible. The Q value of the resulting graph can then be computed according to Equation (4).

It is evident that the above method also works for sparse and unweighted graphs. We call a Q value computed as above *the thresholded Q value*, or Q^t in short. Note that certain variants of our method are also possible. For example, we could use a threshold to determine which edges should be removed, without changing the weights of the remaining edges. The other variant is that, instead of keeping the top s edges, we may keep only a fraction p of s, where $0 < p \leq 1$, if we require the communities to be sparse. From our experience, however, we have found that the results of these variants are very similar, given that p is not too small.

The Algorithm

Based on our method for constructing a new graph from a network and the method for measuring modularity, our overall algorithm is generic in that it can be combined with any clustering algorithm. In our study, we consider spectral clustering, since it has been extensively studied for graph partitioning problems (Chan, Schlag, & Zien 1993; Ng, Jordan, & Weiss 2001; White & Smyth 2005). In general, a spectral clustering algorithm uses eigenvectors of a matrix to map the original data to vectors in the spectral space, which are then clustered by standard algorithms such as k-means. In this research, we adopt the widely used spectral clustering algorithm in (Ng, Jordan, & Weiss 2001). It has been shown that this algorithm is equivalent to optimizing Q in a relaxed sense that ignores the discreteness constraints (White & Smyth 2005).

Given a graph $G = (V, E)$ and its adjacency matrix $A = (A_{ij})$, our algorithm executes the following steps:

1. Compute matrices B and C by Equations (1) and (2).
2. Compute $H = \alpha A + \beta B + C$. Default values of α and β are 1 for sparse graphs and 0 for dense graphs.
3. Let D be a diagonal matrix with $D_{ii} = \sum_j H_{ij}$ and construct a matrix $L = D^{-1/2} H D^{-1/2}$.
4. Find the K largest eigenvectors of L, $x_1, x_2, \cdots, x_K$, and form matrix $U_K = [x_1, x_2, \cdots, x_K]$ in $R^{n \times K}$, where K is an upper bound of the number of clusters.
5. For each integer k, $2 < k < K$:

(a) Form matrix U_k using the first k columns of U_K. Scale each row vector of U_k to have unit length.
(b) Cluster the row vectors of U_k using k-means clustering, and calculate the Q^t value for the result.

6. Select a k that gives a clustering with the highest Q^t.

Experimental Results

Computer-generated Networks

We first tested our methods on networks with known community structures embedded to evaluate their performance. We generated a large number of unweighted networks of 100 nodes, divided into four communities of 25 nodes each. Edges were randomly placed with probability p_{in} for nodes within the same community and with probability p_{out} for nodes across communities. We varied p_{in} from 0.8 to 0.2, representing networks with highly connected to loosely connected communities. For each p_{in}, we varied p_{out} from 0 to $p_{in}/2$ with an interval of $p_{in}/6$. With the trivial case of $p_{in} = 0$, there is no inter-community edges. When $p_{out} = p_{in}/3$, the total numbers of inter- and intra-community edges are roughly equal. When $p_{out} > p_{in}/3$, each node has more inter- than intra-community edges on average, although edge densities within communities are still higher than other regions of the network. For each network G and its adjacency matrix A, we computed matrices B and C with Equations (1) and (2), and different combinations of them. We clustered them given the correct number of clusters. To measure clustering accuracy, we computed the minimal Wallace Index (Wallace 1983) between the true clustering Γ and the predicted clustering Γ', which is defined as following:

$$W(\Gamma, \Gamma') = \min\left(N_{11}/S(\Gamma), N_{11}/S(\Gamma')\right), \qquad (5)$$

where N_{11} is the number of node pairs in the same cluster in both Γ and Γ', and $S(\Gamma)$ is the number of intra-cluster node pairs in Γ.

Fig. 1 shows W as a function of p_{out}, for p_{in} equals to 0.6 and 0.3, representing dense and sparse community structures, respectively. For both cases, C alone results in significantly better clustering than A for $p_{out} > p_{in}/3$, where the Q value drops to below 0.3 (Fig. 1(b) and (d)), indicating weak community structures. This suggests that C is indeed able to remove many of the inter-community edges that are unlikely to be in any triangles. On the other hand, for sparse communities, C is in fact worse than A when p_{out} is small (Fig. 1(c)), since a significant portion of the intra-community edges may be removed in this case. The clustering accuracy of $A + C$ is always better than that of A and only slightly worse than that of C for dense communities. On the other hand, B alone is not better than A in general, and in some cases may be worse. However, if B is combined with A, or $A + C$, it always produces good results.

Next, we generated a set of weighted complete graphs of 100 nodes with four equal-sized communities. The intra- and inter-community edges have weights randomly drawn from the positive half of normal distributions $N(\mu_1, \sigma_1)$ and $N(\mu_2, \sigma_2)$, respectively. We fixed μ_2 at 0 and $\sigma_1 = \sigma_2 = 1$, while varied μ_1 from 0.3 to 1. We used these

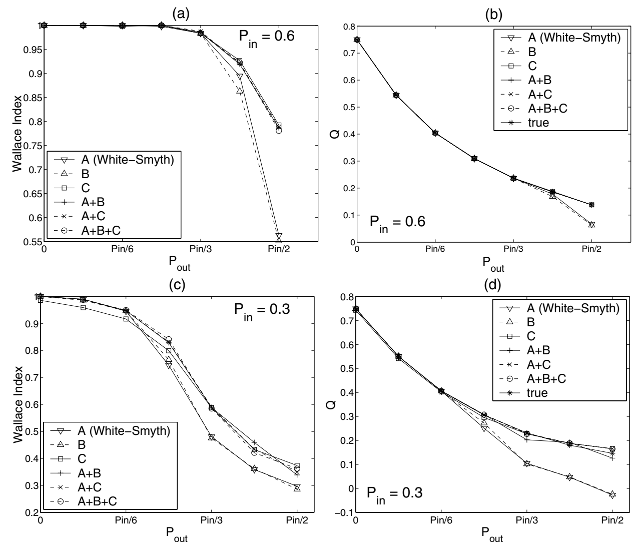

Figure 1: Clustering results on unweighted networks. (a),(c): Wallace Indices between true and predicted clusters. (b),(d): Q values of the true and predicted clusters. Each data point is an average of 100 runs.

graphs to demonstrate that the C matrix can be generalized to weighted networks, and our method for estimating the thresholded Q values on weighted graphs can be used to identify the correct number of clusters.

As shown in Fig. 2(a), given k, the correct number of clusters, C often results in higher W values than A. Furthermore, the combination of $A + B + C$ works no better than C alone, although still better than A. When k is not given, both Q^t and Q can often give good estimations of k, with Q^t being slightly better for smaller μ_1 (Fig. 2(c)). In contrast, the scaled cost function (Chan, Schlag, & Zien 1993) is not able to recover k even in the simplest cases where Q and Q^t make no mistakes. An advantage of Q^t over Q is that Q^t is more meaningful than Q in quantifying cluster qualities. As shown in Fig. 2(b), the Q^t values for these networks range from 0.4 to 0.1, representing networks with strong to weak communities. Indeed, for the networks with $Q^t > 0.3$, our method makes very few mistakes in recovering the original structures, a phenomenon similar to unweighted graphs. In contrast, the Q values tend to be much smaller and do not quantify cluster strengths very well.

Real-world Networks with Known Structures

We also evaluated our methods on two real-world networks with known community structures. The first example is from one of the classical social network studies. In this study, Zachary observed over two years the social interactions among 34 members of a karate club. In this period, the club was split into two smaller ones, due to a dispute between the club's instructor and administrator. Fig. 3(a) shows the network and the actual split of the club. Applying our method to the network, the best result was obtained with matrix $A + B + C$ and $A + B$, where we per-

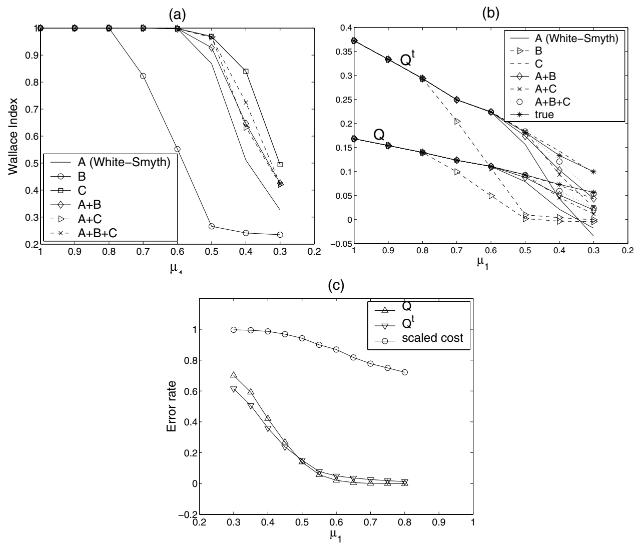

Figure 2: Clustering results on weighted networks. (a) Wallace Indices between true and predicted clusters. (b) Q and Q^t values of the true and predicted clusters. (c) The percentage of incorrect predictions to the number of clusters.

fectly predicted the division of the members, with a Q value 0.372. In comparison, the White & Smyth method disagreed with the actual division on node 3, and had slightly lower Q value (0.36). Interestingly, with our method, the maximal Q value (0.42) occurred when the network was split into four clusters, as indicated by the four different node types in Fig. 3(a). These splits seem to be reasonable: the five hexagonal nodes form a connected sub-community that has no paths to the community led by the instructor (node 33), other than through the administrator (node 1); the 10 circular nodes on the upper left are more tightly connected to nodes 33 and 34 than the 6 nodes on the bottom left. In contrast, the Q values are much smaller for the White & Smyth method to split the network into $k > 2$ clusters.

The second real-world example we examined is a network of 115 NCAA Division I-A American college football teams, where a node is a team and an edge represents a game played by two teams in year 2000. The teams were officially organized into 11 conferences, and each team played more intra- than inter-conference games. Therefore, the conference structure represents the communities that we would like to identify (network not shown due to space limit). Indeed, as shown in Fig. 3(c), the maximal Q value for our method corresponds to 11 clusters, which is exactly the number of conferences. Furthermore, with these clusters, each team was correctly assigned to its own conference, except for eight teams that do not belong to any of the conferences. The clustering by the White & Smyth method with 11 clusters is the same as ours. On the other hand, with other numbers of clusters, our method often identified better community structures than theirs. Since the clustering coefficient is relatively high for this network (0.412), the combination of $A + B + C$ performs slightly worse than other combinations or C alone. In comparison, the clustering coefficient for the karate club network is 0.298, and as a result the combination of $A + B + C$ is better than C alone.

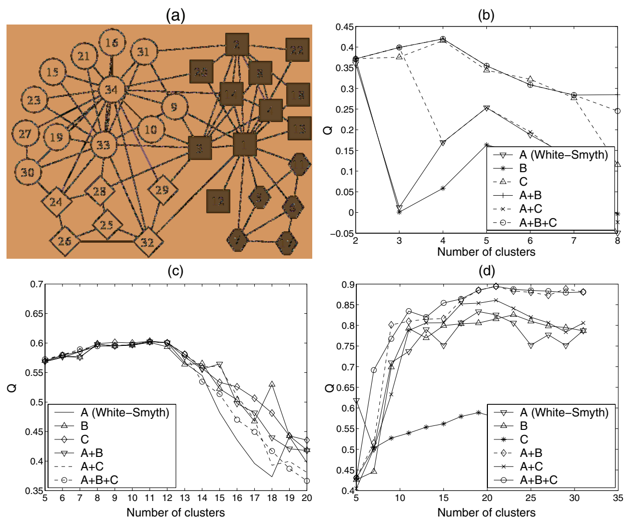

Figure 3: (a) The karate club network. Light and dark nodes represent the actual division of the club. Shapes correspond to the division predicted by our algorithm. (b) Q values from clustering the karate club network. (c) Q values from clustering the NCAA football teams. (d) Q values from clustering the CiteSeer collaboration network.

Real Applications

Finally, we applied our method to two networks for which the true community structures were not well understood. The first application was to cluster a network of collaborations among computer scientists embedded in the CiteSeer co-authorship database. In order to focus on community structures, we selected authors (nodes) who have at least 15 collaborators. The largest connected component in this network contains 275 nodes and 417 edges. We run our algorithm with k from 5 to 31; the Q values of the results are shown in Fig. 3(d). The network has very strong community structures, in that the best Q value, 0.89, was achieved by $A + B + C$ and $A + B$ at $k = 21$. On the other hand, the best Q value obtained by A is only 0.82, indicating that our new method is also effective on this application. Note that matrix C alone produced very poor results, as shown by the low Q values for all k. This is because the collaboration network is very sparse and has a low clustering coefficient of 0.29. Fig. 4 shows the network structure and the best clustering given by $A + B + C$. It is evident that many edges in the network are not in any triangles, which explains why C did not work well on this application.

The clustering clearly reflects various sub-communities among computer scientists, such as machine learning, multi-agent, software engineering, compiler, and cryptography. Some clusters are specific and contain a few nodes, such as the researchers working on PVM or the CiteSeer search engine. These communities tend to be near the perimeter of the

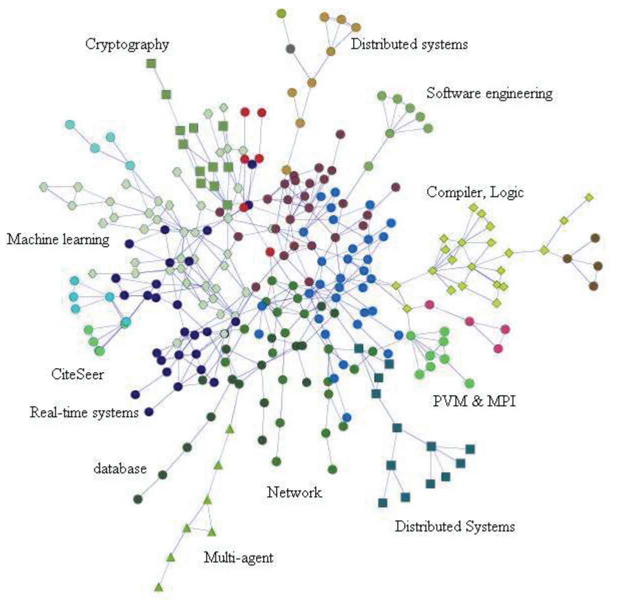

Figure 4: Clustering CiteSeer collaboration network, $k = 21$. Best viewed in color.

network and have very few inter-community edges. On the other hand, some groups are not well-defined and connect to many other communities; examples include networks, distributed systems, and real-time systems communities.

The second application was to cluster a genetic network of 800 yeast cell-cycle genes. Expression profiles of these genes at 77 time points during cell cycles were obtained from (Spellman *et al.* 1998). The network was constructed as a weighted complete graph, where each node represents a gene, and the weight of an edge is the Pearson correlation coefficient between the expression profiles of a pair of genes, scaled to within $[0, 1]$. The graph was then transformed by Equation (3) and fed into the spectral clustering algorithm to find 2 to 10 clusters. As shown in Fig. 5(a), our method using C achieved the best result based on the Q^t measurement, while $A+B+C$ also gave slightly better results than A. The maximal Q^t value, 0.37, was reached at $k = 4$ clusters. The high Q^t value indicates strong community structures with the clustering result. Importantly, it turns out that the clusters obtained correspond very well to the four phases in a cell cycle (G1, S, G2, and M). As shown in Fig 5(b), the average expression profile of each cluster shows good periodicities, and the shift from one phase to another is evident. In comparison, the original Q function failed to predict the number of clusters for this case.

Conclusions and Discussion

We proposed a method for identifying weak community structures in large networks. We showed how two types of local structure properties, the number of triangles and the number of paths of length two, can be exploited to improve the quality of network clustering. We used these local neighborhood structures to derive two local operations (in matrices B and C in the paper). We empirically studied, using

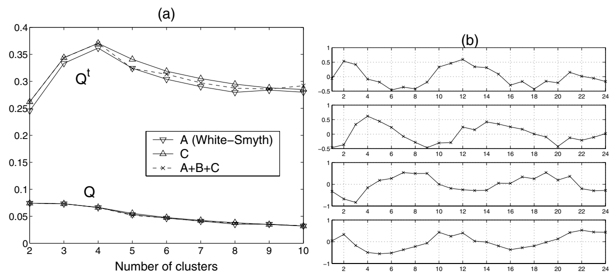

Figure 5: Clustering genetic network (a) Q and Q^t values. (b) Average gene expression profiles of the best 4 clusters.

many known and unknown network structures, the strength of these operations and their combinations. Among many other things, the most important conclusion is that the combination of these operations, along with the original graph, is very effective in revealing weak community structures in both sparse and dense networks.

We also extended the work by Newman and Girvan for quantifying the strength of community structures to weighted complete graphs. We showed that, on both synthetic and real-world networks, the generalization allowed us to unbiasedly evaluate the clustering quality, and automatically determine the best number of clusters without prior knowledge of network structures.

In short, our extensive experiments and applications to many types of networks showed that our methods are effective in discovering high-quality weak community structures in large networks.

Acknowledgments. This research was supported in part by NSF grants ITR/EIA-0113618 and IIS-0535257. Thanks to Mark Newman for the Karate club and NCAA football data, to Marko Rodriguez for the Citeseer data, and to the anonymous referees for comments.

References

Chan, P. K.; Schlag, M. D. F.; and Zien, J. Y. 1993. Spectral k-way ratio-cut partitioning and clustering. In *ACM/IEEE Design Automation Conference*, 749–754.

Jain, A. K.; Murty, M. N.; and Flynn, P. J. 1999. Data clustering: A review. *ACM Comput. Surv.* 31(3):264–323.

Newman, M., and Girvan, M. 2004. Finding and evaluating community structure in networks. *Phys. Rev. E* 69:026113.

Newman, M. 2003. The structure and function of complex networks. *SIAM Review* 45:167–256.

Newman, M. 2004. Fast algorithm for detecting community structure in networks. *Phys. Rev. E* 69:066133.

Ng, A. Y.; Jordan, M. I.; and Weiss, Y. 2001. On spectral clustering: Analysis and an algorithm. In *NIPS*, 849–856.

Spellman, P., *et al.* 1998. Comprehensive identification of cell cycle-regulated genes of the yeast saccharomyces cerevisiae by microarray hybridization. *Mol Biol Cell* 9(12):3273–97.

Wallace, D. L. 1983. Comment. *Journal of the American Statistical Assocation* 78:569–576.

White, S., and Smyth, P. 2005. A spectral clustering approach to finding communities in graph. In *SIAM Data Mining*.