View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Washington University St. Louis: Open Scholarship

Washington University in St. Louis

Washington University Open Scholarship

All Computer Science and Engineering Research

Computer Science and Engineering

Report Number: WUCSE-2006-20

2006-01-01

Discovering weak community structures in large biological networks

Jianhua Ruan and Weixiong Zhang

Identifying intrinsic structures in large networks is a fundamental problem in many fields, such as biology, engineering and social sciences. Motivated by biology applications, in this paper we are concerned with identifying community structures, which are densely connected sub-graphs, in large biological networks. We address several critical issues for finding community structures. First, biological networks directly constructed from experimental data often contain spurious edges and may also miss genuine connections. As a result, community structures in biological networks are often weak. We introduce simple operations to capture local neighborhood structures for identifying weak communities. Second, we consider the issue...
Read complete abstract on page 2.

Follow this and additional works at: https://openscholarship.wustl.edu/cse_research

 Part of the Computer Engineering Commons, and the Computer Sciences Commons

Recommended Citation

Ruan, Jianhua and Zhang, Weixiong, "Discovering weak community structures in large biological networks " Report Number: WUCSE-2006-20 (2006). *All Computer Science and Engineering Research.*
https://openscholarship.wustl.edu/cse_research/171

Department of Computer Science & Engineering - Washington University in St. Louis
Campus Box 1045 - St. Louis, MO - 63130 - ph: (314) 935-6160.

This technical report is available at Washington University Open Scholarship: https://openscholarship.wustl.edu/cse_research/171

Discovering weak community structures in large biological networks

Jianhua Ruan and Weixiong Zhang

Complete Abstract:

Identifying intrinsic structures in large networks is a fundamental problem in many fields, such as biology, engineering and social sciences. Motivated by biology applications, in this paper we are concerned with identifying community structures, which are densely connected sub-graphs, in large biological networks. We address several critical issues for finding community structures. First, biological networks directly constructed from experimental data often contain spurious edges and may also miss genuine connections. As a result, community structures in biological networks are often weak. We introduce simple operations to capture local neighborhood structures for identifying 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. We extend an existing work of a modularity function for evaluating community structures to weighted graphs. Third, we propose a spectral clustering algorithm to optimize the modularity function, and a greedy partitioning method to approximate the first algorithm with much reduced running time. We evaluate our methods on many networks of known structures, and apply them to three real-world networks that have different types of network communities: a yeast protein-protein interaction network, a co-expression network of yeast cell-cycle genes, and a collaboration network of bioinformaticians. The results show that our methods can find superb community structures and the correct numbers of communities. Our results reveal several interesting network structures that have not been reported previously.

Department of Computer Science & Engineering

Washington University in St. Louis
School of Engineering & Applied Science

2006-20

Discovering weak community structures in large biological networks

Authors: Jianhua Ruan, Weixiong Zhang

Corresponding Author: jruan@cse.wustl.edu

Web Page: http://cic.cs.wustl.edu/kcuts/

Abstract: Identifying intrinsic structures in large networks is a fundamental problem in many fields, such as biology, engineering and social sciences. Motivated by biology applications, in this paper we are concerned with identifying community structures, which are densely connected sub-graphs, in large biological networks. We address several critical issues for finding community structures. First, biological networks directly constructed from experimental data often contain spurious edges and may also miss genuine connections. As a result, community structures in biological networks are often weak. We introduce simple operations to capture local neighborhood structures for identifying 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. We extend an existing work of a modularity function for evaluating community structures to weighted graphs. Third, we propose a spectral clustering algorithm to optimize the modularity function, and a greedy k-way partitioning method to approximate the first algorithm with much reduced running time. We evaluate our methods on many networks of known structures, and apply them to three real-world networks that have different types of network communities: a yeast protein-protein interaction network, a co-expression network of yeast cell-cycle genes, and a collaboration

Type of Report: Other

BIOINFORMATICS

Vol. 00 no. 00 2003
Pages 1–10

Discovering weak community structures in large biological networks*

Jianhua Ruan [a], Weixiong Zhang [a,b,†]

[a]Department of Computer Science and Engineering, and [b]Department of Genetics, Washington University in Saint Louis, Saint Louis, MO 63130, USA

ABSTRACT

Identifying intrinsic structures in large networks is a fundamental problem in many fields, such as biology, engineering and social sciences. Motivated by biology applications, in this paper we are concerned with identifying community structures, which are densely connected sub-graphs, in large biological networks. We address several critical issues for finding community structures. First, biological networks directly constructed from experimental data often contain spurious edges and may also miss genuine connections. As a result, community structures in biological networks are often weak. We introduce simple operations to capture local neighborhood structures for identifying 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. We extend an existing work of a modularity function for evaluating community structures to weighted graphs. Third, we propose a spectral clustering algorithm to optimize the modularity function, and a greedy k-way partitioning method to approximate the first algorithm with much reduced running time. We evaluate our methods on many networks of known structures, and apply them to three real-world networks that have different types of network communities: a yeast protein-protein interaction network, a co-expression network of yeast cell-cycle genes, and a collaboration network of bioinformaticians. The results show that our methods can find superb community structures and the correct numbers of communities. Our results reveal several interesting network structures that have not been reported previously.

Supplementary information: http://cic.cs.wustl.edu/kcuts/

1 INTRODUCTION AND OVERVIEW

Complex network structures have drawn much interest lately in many fields of research, ranging from biological studies (e.g., genetic networks, [2]), engineering (e.g., the Internet, [8]), and social sciences (e.g., scientific collaborations [11]). 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. genes in a regulatory networks), 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 discovery of the so-called community structure, a relatively densely connected sub-graph. Discovering such structures is fundamentally important for understanding the dynamics and design principles of complex systems. In the context of computational biology, large-scale experiments and integration of large sets of experimental data have produced maps of several biological networks, such as metabolic networks, genetic interaction networks, protein-protein physical interaction networks, transcriptional regulatory networks and gene co-expression networks. Several attempts have been made to identify patterns from these networks including community structures, which are often referred to as functional modules in bioinformatics literatures [14, 20, 23].

Identifying community structures in a network amounts to clustering nodes into groups. Many clustering algorithms have been proposed in diverse areas, including bioinformatics, data mining, VLSI design, and social networks. However, most conventional algorithms, e.g. k-means, self-organizing maps, hierarchical clustering, and many other surveyed in [6], are not designed specifically for clustering networks. These methods usually cluster objects in a metric space, and make strong assumptions of the statistical or topological properties of the clusters, e.g., Gaussian distributions and spheric shapes. Most real networks do not agree well with these assumptions. Another critical issue in clustering networks is to determine the right number of clusters. This fundamentally important problem is difficult in general, and requires deep insight into the data of interest. A few ideas have been proposed for this problem, however, with limited success [3, 24].

Recently, Newman and Girvan [12] proposed a network clustering algorithm that depends heavily on the topological information of a network. Their method is based on the concept of edge betweenness centrality [5], which measures how likely that an edge will connect two nodes in two communities rather than within the same community. Furthermore, they also proposed a modularity function, called Q, to quantify

*This research was supported in part by an NSF grants ITR/EIA-0113618 and IIS-0535257, and a grant from Monsanto Corporation.

†Corresponding author. Email: {jruan,zhang}@cse.wustl.edu.

© Oxford University Press 2003.

the strength of community structures [12]. 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. Their method has been successful on a variety of networks [12].

Another family of approaches for clustering that have been studied extensively in computer science is the spectral clustering algorithms. According to spectral graph theory, it is well-known that many graph properties are related to the eigenvalues and eigenvectors of the Laplacian matrix of a graph [4]. Several algorithms have been designed to successfully partition graphs by embedding nodes into eigen space and clustering them with geometric algorithms in the eigen space [17, 13]. Recently, Whilte and Smyth [22] showed that partitioning networks using spectral methods is equivalent to optimizing the modularity function Q in a relaxed sense that ignores the discreteness constraints.

We make three contributions in this paper. First, most existing network clustering methods[12, 22] assume sparse networks and networks with strong or tight community structures, which have more intra-community edges than inter-community edges. In the context of biological networks, edges are often derived logically (e.g. in co-expression networks), and/or measured with techniques that have a high error rate (e.g. in protein-protein interaction networks determined by yeast two-hybrid experiments). Therefore, many spurious edges are often included whereas some genuine edges may be missing or have incorrect weights in biological networks. In such networks, community structures are typically weak in the sense that there may be more edges crossing the boundary of a cluster than within the cluster, although the edge densities within clusters may still be higher than other regions of the network. To discover weak communities, we introduce a set of simple operations to capture local neighborhood structures of a node. Our method is based on a transitivity property of many real networks, i.e., two nodes that are connected to a third node are likely to be directly connected as well [10]. In social networks, this means that a friend of my friend is likely to be my friend as well. Consequently, two nodes in the same community are likely to be linked by short paths or loops than those not in the same community. By exploiting such local information, and combining it with spectral clustering, our method can identify weak community structures with significantly improved accuracies.

Second, the modularity function Q proposed in [12] provides a potential solution to automatically determining the most appropriate number of communities in a network. One can run any clustering algorithm multiple times, each time specifying a different value k for the number of clusters to be returned. The correct number of clusters can then be determined by choosing the k that optimizes Q. However, this method can be very costly for large networks with thousands of nodes, which can be potentially partitioned into hundreds of clusters. We develop a k-way partitioning algorithm under the framework of spectral clustering to optimize Q function greedily. Empirical studies show that the best Q obtained by this greedy approach approximates the costly iterative method very well with much reduced computation.

Third, we address the issue of automatically determining the most appropriate number of communities in *weighted dense graphs* in many applications where the similarities or distances of objects can be measured (e.g., similarity of gene expression profiles). Under a good community structure, 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. Furthermore, the Q function is not stable with respect to rank-preserving transformations of edge weights. We propose a simple extension to estimate the Q function on weighted and dense graphs by a rank-based transformation of edge weights that produces much meaningful results.

The paper is organized as follows. We first investigate local neighborhoods in real-world networks and propose two local operations in Section 2.1. We then discuss modularity and number of clusters in Section 2.2 and present two algorithms in Section 2.3. In Section 3, we extensively evaluate our methods on various networks of known structures, and apply them to three real-world applications: an protein-protein interaction network in yeast *S. cerevisiae*, a co-expression network of yeast cell-cycle genes, and a collaboration network of bioinformatics researchers. Finally, we conclude in Section 4.

2 METHODS

2.1 Local structures

Real-world networks often possess intrinsic properties that are lacked in random graphs, such as heavy-tail distributions of node degrees and small-world structures [10]. In particular, it has been observed that real-world networks often have surprisingly higher clustering coefficients than random graphs [10]. 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 real-world networks, $0.1 \leq c \leq 0.5$, while for random networks of n nodes, $\lim_{n\to\infty} c = 0$ [10].

Formally, 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 $\hat{B}_{ij} = \sum_k A_{ik}A_{jk}$ 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 common triangles shared by 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 they are directly connected or not, they have a higher chance to be in the same community than random. Therefore, we define two normalized matrices:

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

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

in which $B_{ij} = \hat{B}_{ij}/\sqrt{D_{ii}D_{jj}}$ and $C_{ij} = \hat{C}_{ij}/\sqrt{D_{ii}D_{jj}}$. 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 the 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 lowered weights. However, the edges that are not in any triangle are simply removed. This may cause some problems if a community is sparse, which 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 can consider the combination of the three:

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

where α and β are parameters. Ideally, smaller α and β are preferred for dense graphs or graphs with high clustering coefficients, and larger α and β should be used for sparse graphs. As to be seen in Section 3, we found that simply taking α and β such that $(\alpha \times A)$, $(\beta \times B)$ and C have the same maximal weight 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 turned out to be better than any other combination in our study (see Section 3).

2.2 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 [12] is defined as

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

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 connecting to cluster i. 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 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 no partitioning, Q is always equal to zero.

A nice feature of modularity Q is that it provides a global quality measurement of community structures. It has been found that most real-world networks have $Q > 0.3$, which was suggested as a threshold for good community structures [12]. We have also found that networks with $Q > 0.3$ are relatively easy to cluster because 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 Section 3).

The definition of Q can be generalized to weighted networks by extending e_{ii} and a_i to corresponding fractions of edge weights, instead of fractions of edges. However, the generalization is only meaningful for sparse networks. Although most real networks are sparse, weighted networks from real applications are often dense or complete graphs. This is because the weights in a weighted graph are usually similarities between nodes (objects), so that a similarity can be computed for any pair of nodes, resulting in a dense or complete graph. On dense weighted networks, the Q function often fails to produce meaningful results, as we will see in Section 3.

A simple way to fix this problem is to use a threshold to filter out some edges to make a dense weighted graph sparse. However, without knowing the underlying community structures 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 alone is not a good criterion.

Here we propose a method to determine an edge-weight threshold for edge removal so that meaningful community structures can be revealed, and the corresponding Q value can be used to unbiasedly compare results from different clustering methods. 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 select a threshold to remove low-weight edges so that in the ideal case, the remaining edges would be just enough to form completely connected communities with no inter-community edges. Therefore, in the ideal case, perfect community structures have all the weights of intra-community edges higher than that of inter-community edges, and the threshold can be uniquely determined given the communities.

Based on this insight, our method for determining the edge-weight threshold is the following. 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 rest edges. In case where the s-th edge is tied with other edges, we may remove all or none of them, while keeping the number of remaining edges as close to s as possible. The Q value of the resulting clustering can be computed according to Equation (4).

Due to the last step in handling ties, the above method works for sparse and un-weighted graphs as well. 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 possible. For example, we could use the weight of the s-th edge as the threshold, while not changing the weights of the remaining edges. The other variant is that, instead of keeping the top s edges, we can keep only a fraction, p, where $0 < p \leq s$, of them if we require the community structure to be sparse. From our experience, however, we have found the results of these variants are very similar, given that p is not too small.

Note that the procedure we proposed for estimating the thresholded Q value on a weighted dense graph only estimates the strength of community structures after partitioning. We also considered an EM-like method to iteratively optimize Q^t explicitly. With this method, we first cluster the network using all the edge weights as usual. Based on the threshold that is used to calculate Q^t, we remove those edges with weights below the threshold and re-cluster the network. This procedure iterates multiple times, and the clustering with the best Q^t value is returned as the final solution. This iterative procedure, although not guaranteed to converge, can in practice often improve clustering quality and help estimate the correct number of clusters, as we will see in Section 3.

2.3 Algorithms

Based on our method for constructing a new graph from the adjacency matrix of a network and the method for automatically determining the number of clusters, our overall algorithm is generic since it can be combined with any clustering algorithm. In our study, we considered spectral clustering, due to its close relationship with maximizing Q as shown by White and Smyth [22]. In this research, we adopted the widely used spectral clustering algorithm in [13]. Given a graph $G = (V, E)$ and its adjacency matrix $A = (A_{ij})$, our first algorithm, called *g-cuts*, executes the following steps:

1. Compute matrices B and C by Equations (1) and (2).
2. Compute $H = \alpha A + \beta B + C$, where α and β are chosen such that the maximal weights in αA, βB and C are approximately the same. For a weighted complete graph, we suggest $\alpha = \beta = 0$.
3. Let D be a diagonal matrix with $D_{ii} = \sum_j H_{ij}$ and construct a matrix $L = D^{-1/2} H D^{-1/2}$, following [13].
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.

If a network is large and contains many clusters, this algorithm can be inefficient for two reasons. First, deriving a large number of eigen-vectors may require a substantial amount of computation. In the worst-case, the running time for computing K eigen-vectors is $O(n^3)$ if K is nearly linear in n, where n is the number of nodes in the network and K the number of eigen-vectors to compute. Second, the algorithm needs to run k-means many times in order to select the k that optimizes the Q value. With no prior knowledge, the upper-bound of k for a given network could be proportional to the number of nodes. To be conservative, we may have to over-estimate the number of clusters significantly. Therefore, in many real applications it is impractical to iterate over all possible k.

We now propose the second algorithm, called *k-cuts*, which is an approximation and faster version of the first. It follows a greedy strategy to recursively partition nodes into clusters to optimize Q. Unlike the algorithm in [22] and [17] that always bisect nodes, we consider a k-way partitioning at each step, where k is typically a small integer in $[2, 10]$, and select a k at each step that maximizes the Q value of the clustering. Each cluster is then split recursively if by doing so the Q value increases. The *k-cuts* algorithm is as follows.

1. Given an adjacency matrix A, compute B, C, and H as in algorithm *g-cuts*.
2. Initialize Γ to be a single cluster with all nodes, and $Q_{old} = 0$.
3. For each cluster P in Γ
   a. Let $\hat{H}$ be the sub-graph of H that corresponds to the nodes in P. Let D be a diagonal matrix with $D_{ii} = \sum_j \hat{H}_{ij}$ and construct a matrix $L = D^{-1/2}\hat{H}D^{-1/2}$.
   b. 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 partitions at each step, and typically $k \in [2, 10]$.
   c. For each integer j, $2 \le j \le k$,
      (1) Form a matrix U_j from the first j columns of U_k, and scale the rows of U_j to unit lengths. Cluster the row vectors into j clusters, Γ_j^P, with k-means.
      (2) Calculate Q value of the clusters as $Q_{new}^j = Q(\Gamma \bigcup \Gamma_j^P \setminus P)$.
   d. Find the j that gives the best Q value, $j^* = \arg\max j(Q_{new}^j)$.
   e. If $Q_{new}^{j^*} > Q_{old}$, accept the partition by replacing P with $\Gamma_{j^*}^P$, $\Gamma = \Gamma \bigcup \Gamma_j^P \setminus P$, and set $Q_{old} = Q_{new}^{j^*}$.
   f. Advance to the next cluster in Γ, if there is any.
4. At termination, Γ contains the best clustering, and Q_{old} is the best Q value achieved.

Note that in step 4(d)(2), if the network is sparse, i.e., Q can be calculated by Equation (4) instead of Q^t, we do not need to re-calculate Q for the whole network. Since the contribution of each cluster towards Q is independent of the other clusters, updating Q after partitioning a cluster into sub-clusters can be computed locally as well. Therefore, this step is very efficient. The advantage of the greedy method is two-fold. First, it reduces the time required for calculating eigen-vectors. Instead of calculating the top K eigenvectors of the graph, where K is linear to the number of nodes in the network, we only need to calculate the top k eigen-vectors on the graph for the first partitioning, where k is a small integer independent on the size of the network. The computation of eigenvectors in subsequent steps is negligible since they are calculated on much smaller subgraphs. Second, the program improves Q progressively. When terminated, it reports both the maximum Q and the best number of clusters. Therefore, the problem of over-estimating the number of clusters disappears.

3 EXPERIMENTAL RESULTS

3.1 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 un-weighted networks of 100 nodes, divided into four communities of 25 nodes each. Edges were randomly placed with probability p_{in} for edges within the same community and with probability p_{out} for edges 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 edge and all the communities are disconnected. 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-community edges 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 using Equations (1) and (2), and applied the *g-cuts* algorithm with the exact number of clusters. To measure the accuracy of the results, we computed the minimal Wallace Index [21] between the true clusters Γ and the predicted clusters Γ', which is defined as follows:

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

where N_{11} is the number of edges in the same cluster in both Γ and Γ', and $S(\Gamma)$ is the number of intra-cluster edges 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 communities, respectively. As shown for both cases, using C alone results in significantly better clustering than using A for $p_{out} > \frac{p_{in}}{3}$, where the Q value drops to below 0.3, as shown in 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, using C is worse than using A for small p_{out}, since a significant portion of the intra-community edges may be removed in this case. A community may also be broken into several components and the clustering of those components becomes random. This problem can be addressed by combining C and A. As shown in Fig. 1, the clustering accuracy using $A + C$ is always better than using A and only slightly worse than using C for dense communities. On the other hand, using B alone is not better than using A in general, and in some cases may be worse. However, if B is combined with A, or A and C, it always produces good results.

Next, we generated a set of weighted complete graphs of 100 nodes with four equal-size 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 graphs to demonstrate that the C matrix can be generalized to weighted graphs, and our method for estimating Q^t values can be used to identify the correct number of clusters.

As shown in Fig. 2(a), for a given number of clusters k, using C often results in higher W values than using A. On the other hand, the combination of $A + B + C$ works no better than C alone, although still better than A. When k is not given,

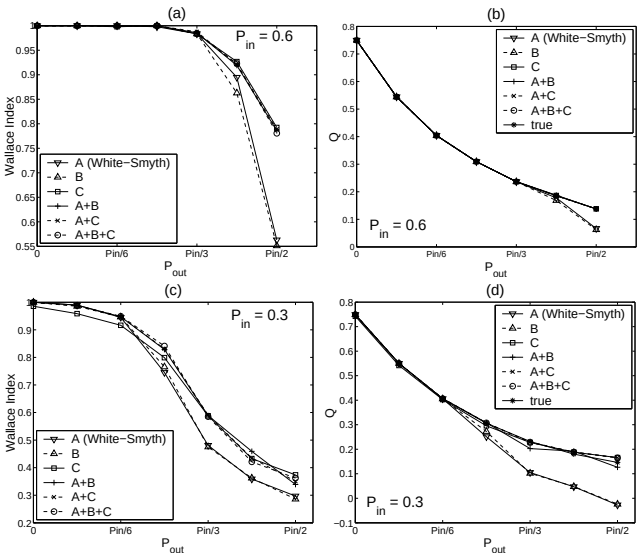

Fig. 1. Clustering accuracy on un-weighted networks with *g-cuts*. (a),(c): Wallace Indices between true and predicted clusters. (b),(d): Q values for the true and predicted clusters.

both Q^t and Q can often give good estimations of k, with Q^t being slightly better for smaller μ. In contrast, the scaled cost function [3] is not able to recover k even in the simplest cases where Q and Q^t make no mistake. An advantage of Q^t over Q is that Q^t is more meaningful than Q in representing cluster qualities. As shown in Fig. 2(c), 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 un-weighted graphs. In contrast, the Q values tend to be much smaller and do not quantify very well cluster strengths.

3.2 Real-world networks with known structures

We now analyze our methods on two real-world networks with known community structures. The first real-world network is from one of the classical social network studies; we obtained the network data from [12]. 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 (nodes in different colors) observed by Zachary. Applying the *g-cuts* algorithm to the network, the best result was obtained with matrix $A+B+C$ and $A+B$, where we perfectly predicted the division of the members. The Q value for the division is 0.372, indicating a strong community structure. In comparison, The White-Smyth method [22] 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) occurs when the network is split into four clusters, as indicated by the different node types in Fig. 3(a). These splits seem to be reasonable: the five nodes on the lower right form a connected sub-community that has no path to the community led by the instructor (node 33), other than through the administrator (node 1); the 10 nodes on the upper left of the network are more tightly connected to nodes 33 and 34 than the 6 nodes on the bottom left. In contrast, the Q values are small for the White-Smyth method to split the network into $k > 2$ clusters, indicating that the divisions are much poorer.

The second real-world example we examined is the network of 115 NCAA Division I-A college football games in 2000, where a node is a team and an edge represents two teams played against each other. The community structures are known, corresponding to 11 conferences. Due to the lack of space, the actual network is not shown here. Indeed, as shown in Fig. 3(c), the maximal Q value for our *g-cuts* algorithm corresponds to 11 clusters, which is the exact number of conferences. Furthermore, with these clusters, each team was correctly assigned to its own conference, except for 8 teams that do not belong to any of the conferences. The clustering by the White-Smyth method with $k = 11$ is the same as ours. On the other hand, with $k \neq 11$, 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 $A + C$ 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.

3.3 A protein-protein interaction network

We obtained a protein-protein interaction (PPI) network for yeast *S. cerevisiae* from the Database of Interacting Proteins (DIP, January 2006 release) [16], which contains curated interactions from both large- and small-scale experimental studies. This dataset consists of 17186 physical interactions involving 4928 yeast proteins. With this dataset, we constructed an un-weighted network, where nodes are proteins and edges are interactions. The largest connected component in this network contains 4873 nodes and 17158 edges.

Because of the size of the network, we only applied the *k-cuts* algorithm. It took about two minutes on a personal computer (Celeron 2.53G Hz, 512MB RAM) to reach a local maximum of Q. We run the algorithm several times with different choices of α, β and k. The maximum $Q = 0.50$ was achieved at $\alpha = 0.01$, $\beta = 0$ and $k = 5$. The choices of α and β are rather interesting. It means that the C term, which counts the number of triangles that pass along an edge, is the most important for clustering the PPI network. This phenomenon indicates that the PPI network has a high rate of false-positive edges [2], which can be effectively suppressed by giving higher weights to the edges that belong to more triangles. We also found that using $k = 2$ always resulted in worse Q values than using larger k values, which supports our choice of a multi-way partitioning in the *k-cuts* algorithm. The

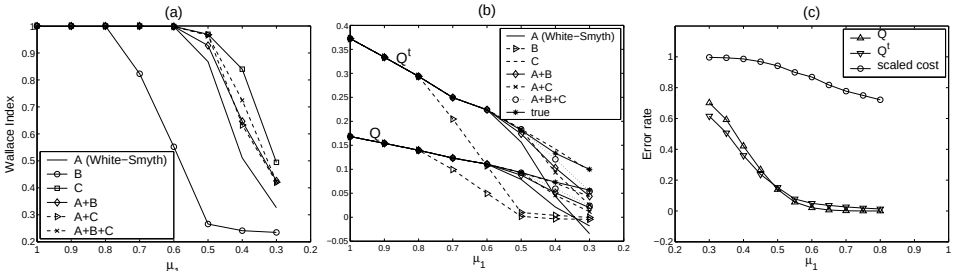

Fig. 2. Clustering accuracy on weighted networks with *g-cuts*. (a) Wallace Indices between true and predicted clusters. (b) Q and Q^t values for the true and predicted clusters. (c) The percentage of incorrect predictions to the number of clusters.

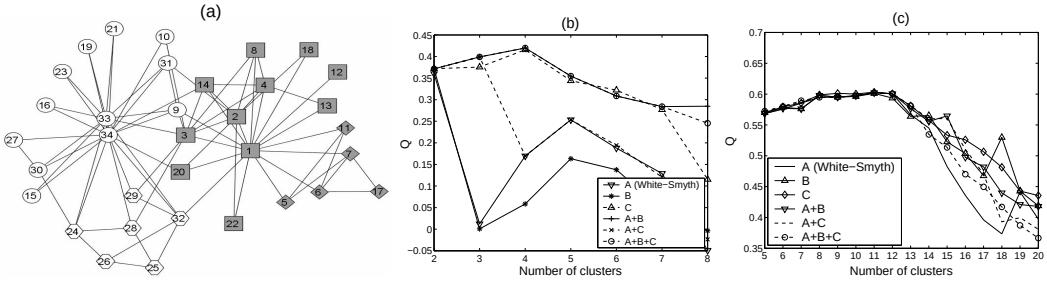

Fig. 3. (a) The Zachary's karate club network. Light and dark nodes represent the actual division of the club. Different shapes within each sub-club corresponds to the division predicted by our *g-cuts* algorithm. (b) Q values for clustering the karate club network with different matrices using *g-cuts*. (c) Q values for clustering the American college football network with different matrices using *g-cuts*.

results using $k = 5$ or 10 differ only slightly. In comparison, it took the *g-cuts* algorithm more than one hour to search for the best partitioning on the sparse matrix A with the maximum number of clusters K set to 50.

In the final clustering returned by *k-cuts*, the network was partitioned into 42 communities. The sizes of the communities range from 2 to 520, with median value 24. To assess the biological significance of the detected communities, we evaluated the consistency of the protein classifications within each individual cluster using a variety of external biological information. For each protein in a cluster, we obtain its functional categories, protein classes, and sub-cellular locations from the MIPS database [9], and pathway classifications from the KEGG database [7]. In addition, we also compared the community structures with the experimentally curated protein complexes in the MIPS database [9] to see how many known complexes are recovered by the algorithm. These different classification schemes should represent different types of biological relevance and serve well to assess the biological significance of the detected communities.

For each cluster, we tested the enrichment of each level-one category in the five classification schemes. The significance of the enrichment was estimated with a cumulative hypergeometric test [1]. The raw p-values were adjusted by Bonferroni corrections for multiple tests [1]. We then counted the total number of enriched categories below a p-value threshold (corrected p-value < 0.05). For comparison, we randomly shuffled the clusters by fixing the size of each cluster and randomly assigning genes to clusters, and performed the same enrichment analysis as for true clusters. Table 1 shows the statistics for the true clusters and the randomly shuffled clusters. As can be seen, the communities found by our algorithm are strongly functionally correlated compared to the random shuffling case. The total number of enriched categories in these communities in any of the five classification scheme is much higher than would be expected by random. In addition, the total number of communities with at least one enriched category is also considerably higher than random.

Fig. 4 shows a condensed community structure for the PPI network, where communities are drawn as circles, with sizes varying roughly with the logarithm of the number of nodes in the communities. The lines between communities indicate inter-community interactions, with the thickness of the lines varying in proportion to the total number of edges connecting the two communities, minus the number of edges that would be expected by random. The nodes in the network were further annotated with three types of information. The different shapes and colors of nodes represent the most significant MIPS function and subcellular location categories enriched in each community. A grey or triangle node means no significant function or subcellular location categories found for the community. Each node was also labeled with names of complexes that overlapped most significantly with the cluster members.

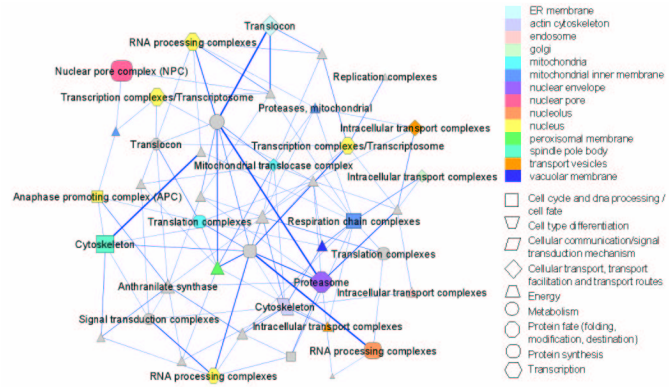

Fig. 4. Clusters of the PPI network. Nodes are color and shape coded. See main text. Better viewed on the supplementary website.

Table 1. Functional enrichment within communities

	Funcat	Protcat	Location	KEGG	Complex
	Number of enriched categories				
Community	50	17	77	23	44
Random	3.0 ± 1.8	$0.58 \pm .75$	1.5 ± 1.3	0.52 ± 0.86	0.48 ± 0.61
Z-score	26.1	21.9	58.1	26.1	71.3
	Number of communities with enriched categories				
Community	22	14	22	12	24
Random	2.6 ± 1.2	0	1.7 ± 1.3	0	0.57 ± 0.74
Z-score	16.2	inf	15.6	inf	31.7

Random: nodes are randomly assigned to communities; Funcat: MIPS functional categories; Protcat: MIPS proteinc classes; KEGG: KEGG pathway classifications. Complex: MIPS curated complexes.

As shown in Fig. 4, most communities (30 out of 42) are annotated by at least one of the three classification schemes. In comparison, only 4.5 ± 1.9 communities in the random clusters contain at least one significant categories. Many of the detected clusters represent previously characterized protein complexes, e.g. the transcription complexes, translation complexes, RNA processing complexes, replication complexes, intracellular transport complexes, signal transduction complex, etc. Overall, the MIPS database contains 65 protein complex families. Among them, 42 had at least three members in the PPI network, and 31 (74%) of them were significantly enriched in the communities detected by our method.

Some complexes were split into multiple communities. With a closer examination, however, we found that such splits correspond very well to lower-level categorization of functional modules. For example, the two transcription complex communities correspond to RNA polymerases and transcription factors, respectively. The four communities of intercellular transport complexes comprise of the Clathrin-associated protein complexes, Coat complexes, SNAP Receptors, and transport protein particle complexes, respectively. The two cytoskeleton communities contain microtubes and actin filaments respectively. One of the translocon communities consists of primarily ER protein-translocation complex and Oligosaccharyltransferase, while the other translocon community contains almost exclusively signal peptidases.

On the other hand, some communities are large and contain multiple categories under a given classification scheme. By inspecting the categories that are enriched in the same community, we found that the genes in these categories are often closely related to some common functions or frequently coordinated in certain biological process. For example, one of the communities is enriched with Histone acetyltransferase complexes (p-value $< 10^{-12}$) and Transcription complexes/Transcriptosome (p-value $< 10^{-14}$). It is known that Histone acetylation is an important mechanism for gene transcriptional regulation [15]. The other example is a community enriched with Spindle pole body complexes ($p < 10^{-14}$) and Cytoskeletons ($p < 10^{-14}$) [18]. Similar phenomena were observed for other classification schemes as well. For example, the categories of transcription and RNA processing were simultaneously enriched in several communities.

3.4 A gene co-expression network

In contrast to traditional methods to treat gene expression profiles as points in the metric space, here we treat clustering gene expression data as a problem of network clustering. We constructed a completely connected network for 800 cell-cycle related genes in yeast *S. cerevisiae*. The gene expression levels of these genes at 77 time points within cell cycles were obtained from [19]. 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 patterns of a pair of genes, scaled to within $[0, 1]$. The graph was then transformed by Equation 3 and fed into the *g-cuts* algorithm to find 2 to 10 clusters. As shown in Fig. 5(a), *g-cuts* on 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, 0.36, was reached at $k = 4$ clusters. The high Q^t value indicates strong community structures among the genes. Importantly, it turns out that the 4 clusters obtained correspond very well to the four phases, i.e., G1, S, G2, and M phases, in a cell cycle. As shown in Fig. 5(b), the average expression pattern of the genes in each cluster shows good periodicity, 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 (see Fig. 5(b)).

We also tested the idea to iteratively improve Q^t (see Section 2.2). The method converged very quickly in 2-3 iterations, and improved the Q^t value to 0.38. Next, we deliberately altered the initial clustering so that it contained more than 4 clusters (from 5 to 9), and applied the iterative method to refine the clusters. Surprisingly, the algorithm improved the original clustering and converged to 4 clusters in less than 10 steps.

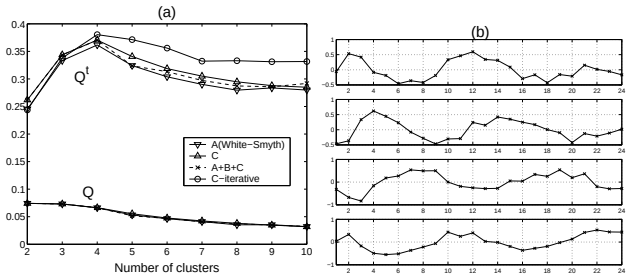

Fig. 5. (a) Q and Q^t values resulted from clustering the 800 cell-cycle genes. C-iterative: the iterative method was used to optimize Q^t. (b) The average gene expression profile with $k = 4$. Only a subset of the time points corresonding to CDC15 are shown here.

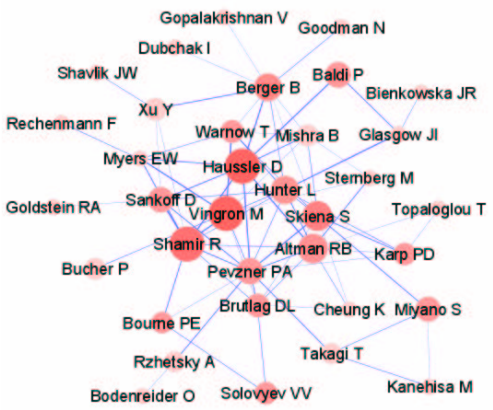

Fig. 6. Clusters of a collobration network of bioinformaticians.

Although further studies are needed, it seems that this iterative procedure works well in practice to reach a local maximal of Q^t. Another interesting results we observed is that, after the iterative procedure converged, the Q^t value for having 5 clusters was significantly improved. The additional cluster appeared in this case corresponds well to the early G1 phase genes, which is consistent with the result reported by [19].

3.5 A collaboration network of bioinformaticians

We also applied our methods to a network of co-authorships among bioinformatics researchers who published in three major bioinformatics conferences: ISMB, RECOMB and PSB. We downloaded the complete paper lists from their websites respectively. The raw data contains 2790 authors and 1494 papers, from which we constructed a weighted network where nodes represent authors and edges represent co-authorship of the given pair of authors. We manually curated the dataset to ensure that distinct names represent different authors (e.g. Gene Myers, E. W. Myers and Eugene Myers are the same person), and vice versa (e.g. Yin Xu and Ying Xu may both appear as Y Xu). We also eliminated co-authorships introduced by co-chairing conference sessions. The edge weight between two authors i and j are calculated as following: $w_{ij} = \sum_{k=1}^{m} 1/(n_k - 1)$, where m is the total number of papers that i and j published together, and n_k is the number of co-authors for paper k [22]. The network contains many disconnected components, so we used only the largest component. Furthermore, to focus on community structures, we selected authors (nodes) who have published at least two papers in these three conferences, which reduced the network to 526 nodes and 1348 edges.

We run the *g-cuts* algorithm with $K = 50$ and the *k-cuts* algorithm with $k = 5$ on the smaller network. The network has very strong community structures, in that the best Q value, 0.862, was achieved with the *g-cuts* algorithm on the matrix $A + B + C$ at $k = 35$. The *k-cuts* algorithm tracked the performance of *g-cuts* very well, achieved a Q value of 0.859 at $k = 27$ on the same matrix. On the other hand, the best Q obtained on A is only 0.81, indicating that the combination of local structure information is also effective on this application.

The clustering of the network corresponding to the best Q value had 35 clusters, with cluster size ranging from 3 to 41 and median size at 12. To better visualize the network, we condensed it to show only the groups. We drew each cluster as a circle, with size varying roughly with the number of individuals in the group. The color intensity of a node represents roughly the total number of papers published by the authors within the group. The lines between groups indicate collaborations between group members, with the thickness of the lines varying in proportion to the total weights of the edges connecting the two clusters. We also labeled each group with a single author who published the largest number of papers in these three conferences. As shown in Fig. 6, the network is centered around several large groups in the middle, each of which is dominated by a well-known scientist.

After a further examination of the authors within each cluster, we found that many communities were grouped according to geographical proximity. For example, we found clusters of authors primarily at Max Planck Institute (Vingron), EBI (Bucher), ORNL (Xu), UCSD (Pevzner), University of Wisconsin (Shavlik), French (Rechenmann) or Japan (Miyano). There are also several communities whose members share common research interests. For example, majority of the authors in the Sankoff cluster (e.g. Tompa M, Blanchette M, Buhler J, Siggia ED, Li H, Bussemaker JH and Sinha S) have worked on motif finding; the group dominated by Brutlag consists of researchers who developed the Gibbs Sampler motif finder (e.g., Lawrence CE and Liu JS); many people in the cluster of Baldi are involved in machine learning and Hidden Markov Models (e.g. Krogh A, Durbin R, Holmes I). The algorithm also clustered together several researchers conducting protein 3D structure studies (Glasgow JI, Lathrop RH, Fortier S and Rost B). These sub-areas (motif finding, HMM, protein structures) are relatively "old-fashioned" and well-defined in bioinformatics research, therefore the annotation

of these groups was relative easy. On the other hand, several communities closer to the center cannot be annotated easily (e.g, the communities dominated by Hunter, Haussler, and Shamir). The reason may be that bioinformatics is a rather new field. Many sub-areas emerge just recently and are not well-defined. Since we collected data from a relatively long period (the first ISMB was held in 1993), a number of researchers within these communities have actually been involved in many sub-areas of bioinformatics and they frequently collaborate with scientists in other communities.

4 CONCLUSIONS AND DISCUSSION

We have proposed a method for identifying weak community structures in large networks. We introduced two local structure properties, the number of triangles and the number of paths of length two, and exploited them using two local operations (in matrices B and C in the paper) to discover weak communities in large networks. We empirically studied the strength of these operations and their combinations using many known and unknown network structures. 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. Combining these local operations with a spectral clustering algorithm, our method was able to optimize the modularity measure and automatically determine the right number of clusters in many applications. In order to handle large networks, we further introduced an approximation version of the method to greedily optimize the modularity measure. The approximation algorithm discovers near optimal community structures and runs significantly faster than the optimal method.

Furthermore, we have also proposed a method that extends the work by Newman and Girvan for quantifying the strength of community structures to weighted and dense graphs. We demonstrated on both computer-generated and real-world networks that the generalization allows us to unbiasedly evaluate clustering quality and determine the best number of clusters without prior knowledge of network structures.

In addition to these novel methods for network clustering, we also produced useful community structures on a yeast PPI network, a co-expression network of yeast cell-cycle genes and a network of co-authorship among researchers in bioinformatics. Without any prior knowledge, our results showed that proteins indeed form interacting communities to provide complex biological functions, cell-cycle related genes are synchronized into different cell-cycle phases, and researchers in the fast growing bioinformatics field form communities based on their interests and geographical locations as well as ethnic background. These results also illustrated the effectiveness and performance of our methods on large experimental data. For further work, it will be interesting to combine different networks from diverse experimental data to further gain deep insight into complex biological systems. For example, it will be interesting to integrate the yeast PPI networks with the gene co-expression networks.

REFERENCES

[1]D. G Altman. *Practical Statistics for Medical Research*. Chapman & Hall/CRC, 1991.

[2]G. Bader and C. Hogue. Analyzing yeast protein-protein interaction data obtained from different sources. *Nat Biotechnol*, 20:991–7, 2002.

[3]P. Chan, M. Schlag, and J. Zien. Spectral *k*-way ratio-cut partitioning and clustering. In *DAC*, pages 749–754, 1993.

[4]F. R. K. Chung. *Spectral Graph Theory*. American Mathematical Society, 1997.

[5]M. Girvan and M.E. Newman. Community structure in social and biological networks. *PNAS U S A*, 99:7821–6, 2002.

[6]A. Jain, M. Murty, and P. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31:264–323, 1999.

[7]M. Kanehisa et al. The KEGG resource for deciphering the genome. *Nucleic Acids Res*, 32:D277–80, 2004.

[8]J. Kleinberg and S. Lawrence. Network analysis. the structure of the web. *Science*, 294:1849–50, 2001.

[9]H. Mewes, et al. MIPS: analysis and annotation of proteins from whole genomes in 2005. *Nucleic Acids Res*, 34:D169–72, 2006.

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

[11]M.E. Newman. Coauthorship networks and patterns of scientific collaboration. *PNAS U S A*, 101 Suppl 1:5200–5, 2004.

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

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

[14]J. Pereira-Leal, A. Enright, and C. Ouzounis. Detection of functional modules from protein interaction networks. *Proteins*, 54:49–57, 2004.

[15]S.Y. Roth, J.M. Denu, and C.D. Allis. Histone acetyltransferases. *Annu Rev Biochem*, 70:81–120, 2001.

[16]L. Salwinski et al. The database of interacting proteins: 2004 update. *Nucleic Acids Res*, 32:D449–51, 2004.

[17]J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22:888–905, 2000.

[18]M. Snyder. The spindle pole body of yeast. *Chromosoma*, 103:369–80, 1994.

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

[20]V. Spirin and L. Mirny. Protein complexes and functional modules in molecular networks. *PNAS U S A*, 100:12123–8, 2003.

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

[22]S. White and P. Smyth. A spectral clustering approach to finding communities in graph. In *SDM*, 2005.

[23]D. Wilkinson and B. Huberman. A method for finding communities of related genes. *PNAS U S A*, 101:5241–8, 2004.

[24]L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *NIPS*, 2004.