Published online 25 November 2011

Nucleic Acids Research, 2012, Vol. 40, No. 3 ***e18***
doi:10.1093/nar/gkr1103

An iterative network partition algorithm for accurate identification of dense network modules

Siqi Sun, Xinran Dong, Yao Fu and Weidong Tian*

State Key Laboratory of Genetic Engineering, Institute of Biostatistics, School of Life Sciences, Fudan University, Shanghai 200433, P.R. China

Received December 1, 2010; Revised October 2, 2011; Accepted November 5, 2011

ABSTRACT

A key step in network analysis is to partition a complex network into dense modules. Currently, modularity is one of the most popular benefit functions used to partition network modules. However, recent studies suggested that it has an inherent limitation in detecting dense network modules. In this study, we observed that despite the limitation, modularity has the advantage of preserving the primary network structure of the undetected modules. Thus, we have developed a simple iterative Network Partition (iNP) algorithm to partition a network. The iNP algorithm provides a general framework in which any modularity-based algorithm can be implemented in the network partition step. Here, we tested iNP with three modularity-based algorithms: multi-step greedy (MSG), spectral clustering and Qcut. Compared with the original three methods, iNP achieved a significant improvement in the quality of network partition in a benchmark study with simulated networks, identified more modules with significantly better enrichment of functionally related genes in both yeast protein complex network and breast cancer gene co-expression network, and discovered more cancer-specific modules in the cancer gene co-expression network. As such, iNP should have a broad application as a general method to assist in the analysis of biological networks.

INTRODUCTION

The continuing development of high-throughput technologies has presented biologists with unprecedented opportunities to study thousands of genes in parallel. Currently, genome-scale omics data, including transcriptomics, proteomics, metabolomics, etc. are being generated on a daily base. How to store, organize and interpret such a tremendous amount of data, however, creates a significant challenge in the field of computational biology. Biological network provides a convenient platform for displaying and visualizing the complex relationships between genes/proteins in cell, and has been quickly adopted as a general tool for genome-wide analysis (1–10). The most popular and widely studied biological network is the protein–protein interaction network in which the nodes correspond to proteins while the edges represent the physical interactions between proteins (1–8,10). Other examples include gene co-expression network, genetic interaction network, etc. (5,11).

Although a network is easy to construct, it is impractical to use the network directly for computational analysis, given the sheer volume of network data and the complicated network structure. It is therefore often necessary to partition a network into modules or subcomponents before the application of further analysis. A network module is generally defined as a local dense community inside which nodes have more edges with each other than with those outside the module (12). For biological networks, the network modules are often associated with protein complexes (7,13), or enriched with specific functions (10), making it useful to generate biologically meaningful hypothesis. So far, a range of definitions on network modules have been proposed, and various heuristic methods based on those definitions have been developed (4,14,15). Hierarchical clustering is one of the earliest used methods for network partition (16). Although it has the advantage of not specifying the size or number of modules to partition, given the stringent tree structure of network produced by hierarchical clustering, it is often difficult to decide where to cut the tree in order to get the best division (17). Modularity proposed by Newman and

*To whom correspondence should be addressed. Tel: +86 21 55665169; Fax: +86 21 55665643; Email: weidong.tian@fudan.edu.cn
Present address:
Weidong Tian, State Key Laboratory of Genetic Engineering, Institute of Biostatistics, School of Life Sciences, Fudan University, Shanghai 200433, P.R. China.

The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

© The Author(s) 2011. Published by Oxford University Press.
This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (http://creativecommons.org/licenses/by-nc/3.0), which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

Girvan is a benefit function that quantifies how well a network is divided into modules (4,14). The higher the modularity, the denser the network modules and the more sparse connections between network modules are. Because of its simple mathematics form, modularity has become the most popular benefit function and quickly been adopted by many algorithms for network partition (18).

However, on the one hand, maximizing modularity is a NP-hard problem (19). Though a number of heuristic methods, such as greedy (4,14,15,20), Qcut (21), simulate annealing (22) and Spectral Clustering (SC) (14,23,24) have been developed, these methods are not always capable of finding the best modularity. On the other hand, as Fortunato *et al.* (25) pointed out, there is an inherent resolution limit in the modularity function, making it difficult to detect smaller communities by maximizing modularity. Recognizing the resolution limit of the modularity function, a number of methods have been developed to either develop an alternative quality function to modularity (26,27), or optimize or post-process the network partition on the basis of modularity function (21,28). The alternative quality functions include modularity density developed by Li *et al.* (26), a spin model-based formulation proposed by Reichardt *et al.* (27), and etc. However, these alternative methods still suffer a resolution limit. Ruan *et al.* (21) developed a recursive procedure named hQcut to partition networks on the basis of a modularity-based method, Qcut, and demonstrated in the benchmarks that the recursive partition achieved significant improvement in the quality of network partition than that without the iteration. However, because of the need to partition a large number of random graphs at every step of iteration by Qcut to evaluate whether to continue or stop the iteration, this method suffers high computational time especially for large networks, making it impractical for analyzing a large number of real biological networks in parallel.

In this study, we observed that despite the resolution limitation, modularity tends to preserve the primary network structure of the undetected dense network modules. In addition, though different modularity-based methods might divide networks with different quality, the modularity produced by them tends to be similar to each other. Therefore, similar to Ruan *et al.*'s (21) hQcut method, in this study, we proposed an iterative Network Partition (iNP) algorithm that recursively partitions pre-partitioned network modules. However, what make iNP different from hQcut are the followings. First, iNP provides a general framework in which any modularity-based methods can be implemented in the network partition step, and we tested iNP with three modularity-based algorithm: Multi-Step Greedy (MSG) (15), SC (24) and Qcut (21) in this study. Second, given that different modularity-based methods tend to generate networks with similar modularity value, at the iteration step, we chose MSG, a greedy method that runs extremely efficient though with poor quality of network division, to partition the random graph generated from the parent module, in order to quickly evaluate the statistical significance of the iteration. In a benchmark with hundreds of simulated networks with different degree of complexity, we demonstrated that iNP achieved significant improvement in the quality of network division over the original modularity-based algorithms. In two real biological network cases: a yeast protein complex interaction network and a breast cancer gene co-expression network, compared with the traditional methods, the iNP algorithm was able to identify more gene modules with significantly better enrichment of functionally related genes in both networks, and identify more breast cancer-specific gene modules in the cancer co-expression network. What's more, we found that although MSG performed much worse than SC and Qcut in simulated networks, iNP-MSG performed comparably to iNP-SC and iNP-Qcut in relatively simple simulated networks and the two real biological networks. Because it took only a minute for iNP-MSG to finish partitioning the breast cancer gene co-expression network that consists of 9112 genes, compared with 10 and 100 min by iNP-Qcut and iNP-SC, respectively, this makes iNP-MSG an appealing choice for analyzing large number of real biological networks.

MATERIALS AND METHODS

Modularity and the MSG, SC, Qcut and simulated annealing (SA) algorithms

Modularity is a popular benefit function to evaluate the quality of a network division. The definition of modularity is as the followings:

$$M = \sum_{i=1}^{N_m} \left[\frac{l_i}{L} - \left(\frac{d_i}{2L} \right)^2 \right]$$

where M is the modularity, N_m is the total number of modules, l_i is the sum of edges in module i, d_i is the sum over the degree of nodes in module i where the degree of a node is defined as the number of nodes in the network it is connected to, and L is the sum of edges in whole network. In practice, the higher the modularity, the better quality of the network division is. Generally, the modularity of a good partition is >0.30. In this study, we investigated the following three modularity-based algorithms for network partition, which all aim to maximize the modularity by different means. Below, we briefly describe the procedures of these algorithms. For detailed description of these algorithms, please refer to the original publications.

MSG is a greedy method. Compared with traditional greedy methods that iteratively merge two communities contributing the most to modularity maximization, MSG optimizes modularity by merging multiple pairs of communities at each step of iteration. In addition, a Vertex Mover algorithm is implemented after the convergence of modularity maximization to adjust the 'misplaced' vertices to neighboring communities in order to further improve the modularity. MSG is highly computationally efficient, with a complexity of approximately $O(N\log^2 N)$ for sparse networks, where N is the number of nodes in the network. Detailed description of MSG can

be found in (15). The MSG source code was obtained from its author Philipp Schütz.

SC is a spectral approach-based clustering method. It involves mapping of the similarity matrix in a high-dimensional space to a low-dimensional space, clustering the nodes by k-means, and selecting the best 'k' that gives the maximum modularity value (24). It is currently one of the most popular network partition algorithms. The SC source code was obtained from its author Mark Newman.

Qcut is a graph partitioning algorithm. Compared with other graph partitioning algorithms, Qcut uses modularity to automatically determine the optimal partitioning and the number of partitions, and is parameter free. In addition, Qcut also combines spectral graph partitioning with local search to optimize modularity, making it an efficient heuristic algorithm (21). The source code was obtained from its author Weixiong Zhang.

All algorithms in this study were tested on a Dell PowerEdge R700 server with Intel Xeon X5650 2.67 GHz CPU. Single CPU core was used in order to evaluate the computational time by each algorithm.

The iNP algorithm

In iNP, a modularity-based algorithm, such as MSG or SC, is first applied to divide a network into modules. Then, for each module, the modularity-based algorithm is iteratively applied to further partition it into smaller ones until all smaller modules are identified. To determine whether to stop or continue partitioning on a pre-partitioned module, the modularity value (M-value) after partitioning this module is recorded. Then, similar to Ruan *et al.*'s (21) hQcut procedure, a number of random graphs with similar degree distribution to the pre-partitioned module are generated and partitioned in order to obtain the mean and variance of the M-value of the random graphs. However, unlike hQcut that uses Qcut to partition the random graphs, which is not computationally efficient, MSG is used to partition the random graphs irrespective of which modularity-based algorithms are used in iNP. Furthermore, to speed up the process, a summed modularity difference defined by the following equation: $S = \sum_{i}^{N} (M - M_i)$ is computed, where M is the M-value of the pre-partitioned module and M_i is the M-value of the ith random graph. For the first 10 random graphs, i.e. $N = 10$, if $S \geq 1.0$, then we continue the iteration, and if $S \leq 0.2$, then we stop the iteration. If S is in between 0.2 and 1.0, then we continue to generate 50 random graphs, in order to compute the mean and variance of the M-value of the random graphs from which we compute the Z-score of the M-value of the pre-partitioned module. If Z-score ≥ 2, we continue the iteration; otherwise, we stop the iteration. The source code of iNP is available for downloading at http://202.120.224.143/index/software.

Generation of simulated networks

To benchmark the iNP method, we generated several hundreds of simulated networks. Each network consists of a total number of 1000 nodes. To generate modules in the network, we first defined the mean community size of the modules in a network. For example, if the mean community size is 25 nodes, then the network has around 40 modules. By adjusting the mean community size (from 25, 50 . . . to 150 nodes, with an interval of 25 nodes) and the variance (from 10% to 40% of the mean community size), we could generate networks with different number of modules. Then, to achieve different degree of network complexity, we defined an in-out-degree ratio for the nodes inside a module. The in-out-degree ratio of a node is defined as the number of links this node has in the module to that in the network. The higher the in-out-degree ratio, the clearer the modular structure of the network is. By experimenting the mean in-out-degree ratio and the variance, we found that when the in-out-degree ratio and the variance were set at (0.5, 0.2), (0.6, 0.2) and (0.8, 0.2), we could generate networks with a modularity value around 0.4, 0.5 and 0.65 based on the defined modules, representing complex, medium complex and simple networks, respectively. For each combination of the above-mentioned three parameters (the mean community size, the variance of community size and the in-out-degree ratio), we generated nine networks. Finally, we obtained a total number of $6 \times 4 \times 3 \times 9 = 648$ simulated networks.

The Jaccard accuracy to measure the quality of a network division

Because the exact components of each module in a simulated network are known by definition, after a network partition, we can compare the partitioned modules with the defined modules to evaluate the quality of the network division. Here, we used the Jaccard accuracy measure for this purpose (29). Given the partitioned modules $G_1, \ldots, G_i, \ldots, G_n$ and the originally defined modules $A_1, \ldots, A_j, \ldots, A_m$, where G_i and A_j refers to the i-th partitioned and the jth generated module, respectively, we computed $J_{G_i,A_j} = |G_i \cap A_j| / |G_i \cup A_j|$ to indicate the overlap between G_i and A_j. Then, for every G_i, we computed $\mathrm{Jac}_{G_i} = \max_j J_{G_i,A_j}$, from which we computed $\mathrm{Jaccard\_accuracy} = \sum_{i=1}^{n} |G_i| \cdot \mathrm{Jac}_{G_i} / \sum_{i=1}^{n} |G_i|$, where $|G_i|$ is the number of nodes in G_i and n is the total number of partitioned modules, to represent the overall similarity of the partitioned network to the generated network. The Jaccard accuracy ranges from 0 to 1, with 1 indicating a perfect network division.

Construction of a yeast protein complex interaction network and a breast cancer gene co-expression network

We downloaded the protein complex interaction network from Yu *et al.*'s (30) paper published in 2008. There are a total number of 1622 genes with 9070 interactions.

We downloaded a breast cancer dataset (GSE10780) (31,32) from the Gene Expression Omnibus (GEO) database. There are a total number of 185 samples in the data set, in which 143 are normal while 43 are cancer samples. We first filtered out those probes that have the lowest 30% expression value and variance. Then, for each gene, we used the median value of the

corresponding probe set as its expression value, and conducted log2 transformation. The missing values were filled by the impute.knn package in R. The total number of genes in this data set is 21 561. We used quantile-normalization (limma package in R) to normalize the gene expression values in each sample. Because some samples in the data set are very similar to each other, to remove the redundant samples, we computed the Pearson correlation coefficient (PCC) for each pair of the 185 samples based on their gene expression values, and only kept those samples that have a PCC <0.99 between each other. This resulted in a total number of 162 samples, among which 123 are normal while 39 are cancer samples.

We followed Zhang *et al.*'s (33) procedures to construct the breast gene co-expression network. Specifically, we first computed the PCC between each pair of genes, and ranked all gene pairs according to their PCC. Then, we selected the top 1, 0.8, 0.6 . . . 0.01% gene pairs to construct the co-expression networks, respectively. For each network, we computed the linear regression coefficient between the log10 transformed degree k (the degree of a node is defined as the number of edges this node has in the network) and the frequency of k (the frequency of nodes with a given degree), and chose the network that gave the best correlation. Such network is considered to have the scale free property. Based on this measure, we chose the network constructed from the top 0.1% gene pairs, which includes 9112 genes with a total number of 244 928 edges. When partitioning the co-expression network, we considered the network as a binary network.

Measurements on the quality of network partition on real biological networks

Unlike the simulated networks in which the exact components of each module are known in advance, there is no answer on how real biological networks should be partitioned. To evaluate the quality of the partition of real biological networks, in this study, we have developed a functional linkage enrichment (FLE) score, which is defined as: $\mathrm{FLE} = \sum_{i}^{N} \left(\mathrm{funsim}_{\mathrm{avg},i} - \mathrm{funsim}_{\mathrm{rand}}\right)$, where i refers to the ith partitioned module, N is the total number of partitioned module, $\mathrm{funsim}_{\mathrm{avg},i}$ is the averaged funsim scores of all gene pairs in the i-th partitioned module, and $\mathrm{funsim}_{\mathrm{rand}}$ is the random funsim score of a pair of genes in the genome. The funsim score of a pair of genes is computed followings Schlicker *et al.*'s (34) description. A funsim score ranges from 0 to 1, with a higher score indicating stronger functional linkage between a pair of genes. The funsim score was designed such that all GO terms associated with the two genes as well as the specificity of each GO term are taken into consideration. Here, only those genes with known GO biological process terms (GO annotations with evidence codes of 'IEA' and 'RCA' are excluded) are taken into consideration, while a module is considered only if it has at least two pairs of known genes.

The rationales of developing the FLE score are 2-folds. First, the main purpose of partitioning real biological networks is to obtain biologically meaningful gene modules, while the more functionally related genes are in a module, the more biologically meaningful it is. Therefore, the greater the averaged funsim score of all pairs of genes in a partitioned module than the random funsim score, the more biologically relevant the module is. Second, to compare the network partition quality by different methods, the method producing more biologically meaningful modules should be favored. Thus, with the FLE score, we can directly compare the performance of different methods in partitioning a real biological network.

However, FLE may be biased to networks with more number of modules. For example, if a biologically relevant module that has a higher averaged funsim score than random is randomly partitioned into two smaller modules, then the resulted new network may have a higher FLE score. This may be especially a problem for iterative partitioning, because by design it will generate more number of modules than the non-iterative methods do. Therefore, to rule out the possibility that a higher FLE score by iNP may be resulted from random partitioning, we have developed two additional scores on the basis of functional linkages: a functional cohesiveness score (FCS), and a functional distinctiveness score (FDS). Suppose a module m was partitioned by a non-iteratie method, and was further partitioned into N modules by iNP, then FCS is defined as $\mathrm{FCS} = \left(\sum_{i=1}^{N} \mathrm{funsim}_i\right)/N/(\mathrm{funsim}_m)$, where funsim_m is the averaged funsim score between genes in module m, and funsim_i is the averaged funsim score between genes in module i, and N is the total number of newly generated modules from module m. FDS is defined as

$$\mathrm{FDS} = \sum_{i \le N, j \le N, i \ne j} \left(\mathrm{funsim}_i + \mathrm{funsim}_j\right)/\left(\mathrm{funsim}_{i,j} + \mathrm{funsim}_{j,i}\right)/ N(N+1)/2,$$

where funsim_i and funsim_j are the funsim scores of module i and j, respectively, while $\mathrm{funsim}_{i,j}$ is the averaged funsim scores between genes in module i and genes in module j, and $\mathrm{funsim}_{j,i}$ equals to $\mathrm{funsim}_{i,j}$. If a module is subjected to random partitioning, then both the FSC and FDS scores will be equal to 1. However, if the iterative partition results in more biological relevant modules, then both scores will be >1, indicating that the genes inside the new modules are more functionally cohesively related to each other, while the genes between the modules are functionally distinctive from each other. In contrast, if the iterative partition destroys the inner structure of the original module, then both scores will be <1.

Gene Ontology annotation and enrichment analysis

We downloaded the human and yeast gene GO annotation database released in August 2010 by Gene Ontology database (35), and filtered out GO annotations with evidence codes of 'IEA' and 'RCA'. For enrichment

analysis, we focused only on those GO terms with the number of annotated genes ranging from 10 to 300. When analyzing the gene modules from the protein complex interaction network, we considered GO terms of 'Biological Process' and 'Molecular function'. For breast cancer gene co-expression network, we focused only on GO terms from 'Biological Process' branch. Fisher's Exact test (R package Fisher's test) was used to compute the enrichment *P*-value, which was adjusted by the 'FDR' method (36). The enrichment threshold of *P*-value was set at 0.1.

Identification of the breast cancer-specific gene co-expression modules

To identify the gene co-expression modules conferring breast cancer specificity, for a given partitioned gene module in the gene co-expression network, we first computed the median expression value of the genes inside the module for each sample in the gene expression data set. Then, because the sample status (cancer or normal) was known for every sample in the dataset, for each gene co-expression module, we plotted a ROC curve using its median gene expression value to classify the cancer (labeled 1) versus normal (labeled 0) samples. The median expression value of a given module can be either higher (upregulated) or lower (downregulated) in cancer samples than in normal samples, or indifferent in between them. Consequently, the corresponding area under curve of ROC (AUC_{ROC}) can be greater, smaller than, or close to 0.5, respectively. The more deviated the corresponding AUC_{ROC} is from 0.5, the more specific the gene module is in distinguishing cancer from normal samples. Here, we selected those gene modules with an AUC_{ROC} of either >0.8 or <0.2 as upregulated or downregulated cancer-specific gene modules, respectively. GO enrichment analysis were then performed on each cancer-specific gene module.

RESULTS

Despite the resolution limitation, modularity preserves the primary network structure of undetected dense modules

As described in the 'Materials and Methods' section, we generated 648 simulated networks with different degree of complexity and different number of generated modules. Here, we applied three modularity-based algorithms: MSG, SC and Qcut, to partition those networks. Among these three methods, SC is currently one of the best and most popular network partition methods. For description about these methods, refer to the 'Materials and Methods' section. In terms of computational speed, MSG was the most efficient one, with an average computational time of 0.7 s per network, while both SC and Qcut were ~20 times slower than MSG.

The modularity values produced by different methods are similar to each other for all simulated networks, though MSG produced slightly smaller modularity values (Figure 1A). However, the quality of network division by different methods is noticeably different from each other: the Jaccard accuracy by MSG is significantly worse than that that by SC and Qcut, while SC and Qcut performed comparably to each other, with Qcut slightly better than SC (Figure 1B). For example, for simple, medium complex and complex networks, the median Jaccard accuracy produced by MSG is 0.79, 0.72 and 0.68, respectively, in contrast to 1, 0.94 and 0.82 by Qcut, and 1, 0.93 and 0.83 by SC, respectively. Thus, both SC and Qcut outperformed MSG with significant margin in terms of the quality of network division. However, even for both SC and Qcut, there are still ~35% of simple networks whose Jaccard accuracy is <1, indicating there are still a lot of rooms to improve the quality of network partition.

To find out a way to improve the quality of network partition, we first inspected whether the reasons why these methods failed to accurately partition some networks were because of overaggressive or because of overconservative partitions. Figure 1C shows the box plot of the ratio of the number of partitioned modules to the number of defined modules by different methods. Interestingly, none of these three modularity-based algorithms produced a network with more modules than that of the defined ones in any simulated network, indicating that modularity-based methods are not aggressive in partitioning networks. However, there are still two possibilities based on this observation. One, several defined modules may be merged into one partitioned module. Two, a defined module may be split apart and then merged with other partitioned modules. To find out which one is more likely to happen, we computed the fraction of defined modules whose primary structure is preserved in the network. Here, the primary network structure of a defined module is considered preserved if 80% of its nodes are kept in the same module after a network partition. As can be seen in Figure 1D, for all three methods, the median value of this fraction is still close to 1 even for complex networks, though the chances that the primary network structure of a defined module is affected are higher when the network becomes more complex. Therefore, it can be concluded that for most simulated networks, especially the relatively simpler ones, maximizing modularity tends to merge several dense modules into larger ones, with the primary network structure of each module preserved. This property of modularity promoted us to reason that an iterative use of the modularity-based method to partition the pre-partitioned modules may gradually resolve the network structures.

The iNP algorithm significantly improves the quality of network division

Given that the primary network structure of most modules are persevered by modularity-based methods, in this study, we have developed an iNP algorithm to recursively partition the pre-partitioned modules using modularity-based methods (see 'Materials and Methods' section for details). iNP provides a general platform in which any modularity-based methods can be implemented in the iNP step. Similar to an iterative version of Qcut that was named as hQcut, at each iteration step, a number of random graphs with similar degree distribution to the

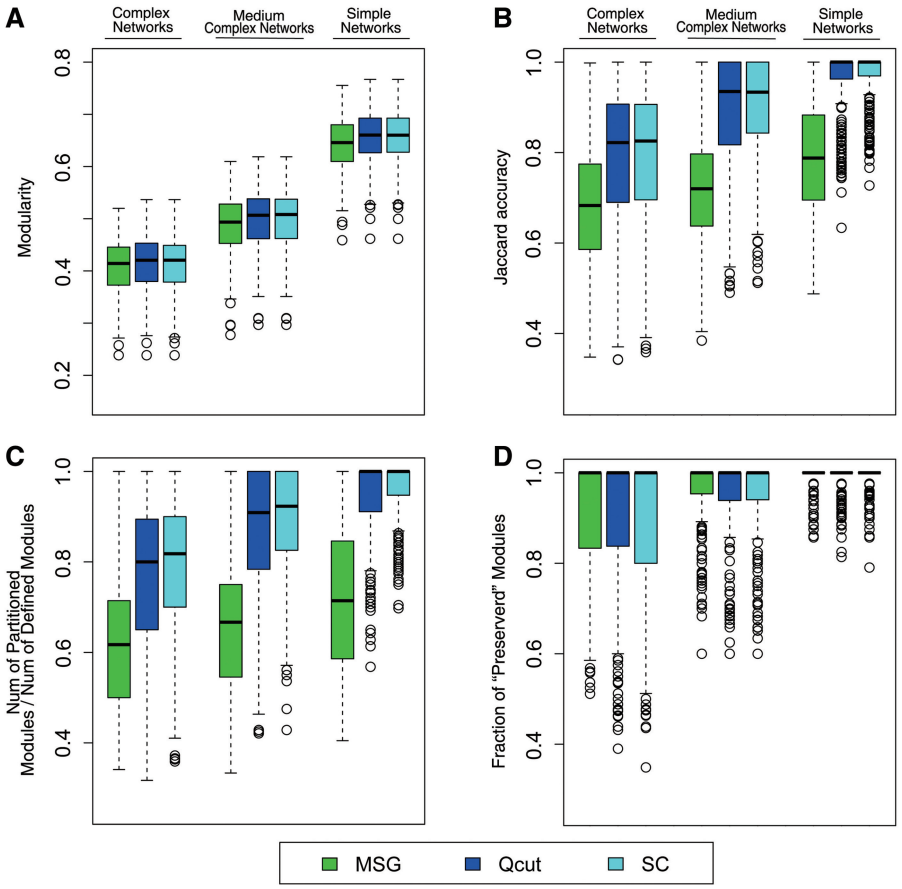

Figure 1. Performance of the modularity-based methods in partitioning the simulated networks. Three modularity-based methods, MSG, Qcut and SC, were compared with each other. (**A** and **B**) show the modularity value and the Jaccard accuracy of the networks partitioned by different methods, respectively. (**C**) The ratio of the number of partitioned modules to that of defined modules in a given network for different methods. (**D**) The fraction of the defined modules whose primary network structure is preserved after network partition by different methods. In all subfigures, the results are shown in boxplot for the complex, the medium complex and the simple networks separately from left to right. The network complexity is defined by the modularity value computed using the defined modules (see 'Materials and Methods' section for details).

pre-partitioned module are generated in order to determine the modularity threshold of continuing or stopping the iteration. However, unlike hQcut that uses Qcut to partition the random graphs in order to obtain the modularity of the random graphs, which is very time-consuming, we implemented MSG to partition the random graphs. This is because though MSG performed much worse than SC and Qcut in terms of Jaccard accuracy, the modularity value produced by MSG is similar to that by SC and Qcut (Figure 1A). Because MSG is extremely fast, using MSG to partition the random graph can greatly reduce the time consumption of iNP. Here, we tested iNP with MSG, SC and Qcut as the main partitioning algorithms, and for simplicity named them as iNP-MSG, iNP-SC and iNP-Qcut, respectively.

Figure 2A–C show the Jaccard accuracy by iNP-MSG, iNP-Qcut and iNP-SC in comparison with that by MSG, Qcut and SC, respectively. As can be seen, compared with the corresponding non-iterative methods, iNP achieved significant improvement in the quality of network division, especially for iNP-MSG. There were also a large number of networks whose Jaccard accuracy remained the same before and after the iteration, indicating that these networks were not subjected for iteration. As shown in Figure 2D, the fraction of networks partitioned by MSG, Qcut and SC that were subjected for iteration by iNP was 81, 39 and 38%, respectively, which is consistent with the observation that MSG performed much worse than both Qcut and SC. In Figure 2E, we compared the performance of all methods with each other. From the comparison, we can see that though MSG was significantly worse than both SC and Qcut, iNP-MSG now outperformed both Qcut and SC in partitioning the medium complex networks, and performed comparably to them in both complex and simple networks. Though iNP-MSG was still slightly worse than both iNP-SC and iNP-Qcut, the difference between them has been significantly reduced, compared with that before the iteration. The average time consumption by iNP-MSG across all 648 networks was about 20 s, while iNP-Qcut and iNP-SC consumed about 54 and 40 s on average per network, respectively. However, the relatively smaller difference in time consumption between these methods after the iteration was due to the fact that only ~38% of networks partitioned by Qcut and SC were

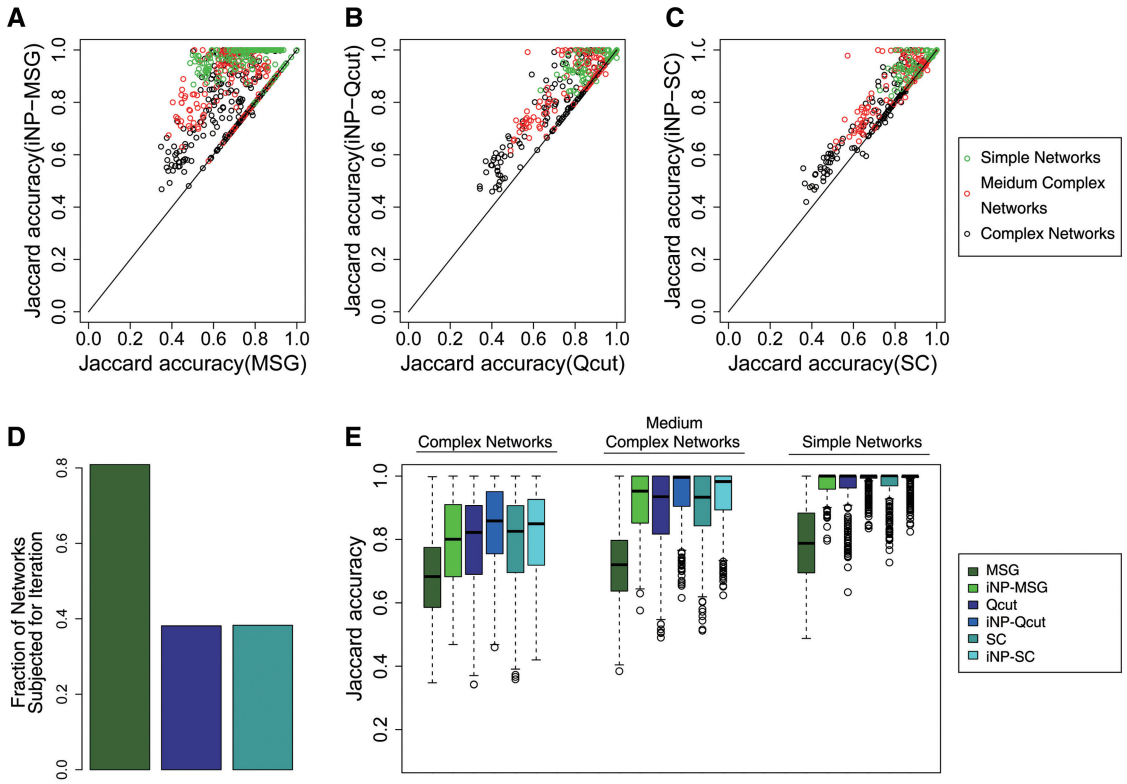

Figure 2. Performance of the iNP method in partitioning the simulated networks. (**A–C**) The Jaccard accuracy of iNP-MSG versus MSG, iNP-Qcut versus Qcut, and iNP-SC versus SC, respectively, with the black, red, and green circles representing the complex, the medium complex and the simple networks, respectively. (**D**) The fractions of the simulated networks initially partitioned by MSG, Qcut and SC that were further subjected for iterative partitioning by iNP-MSG, iNP-Qcut and iNP-SC, respectively. (**E**) The boxplots of the Jaccard accuracy of different algorithms on partitioning the complex, the medium complex and simple networks, respectively.

subjected to iterative partitioning, compared with 80% of MSG partitioned networks. In summary, we have proven that by iteratively partitioning, iNP achieved significant improvement over the traditional modularity-based methods including SC, the currently one of the most popular methods, in the quality of network division.

Partitioning the yeast protein complex interaction network by iNP

To further demonstrate the usefulness of iNP, we applied it to partition a yeast protein complex interaction network. The yeast protein complex interaction network consists of 1540 genes, and has a simple network structure, with the modularity ~0.79 partitioned by MSG, Qcut and SC without iteration. The number of modules partitioned by MSG, Qcut and SC is 197, 197 and 58, respectively. The reason why SC partitioned such significantly less number of modules than the other two methods was because a large number of small isolated modules were simply merged into large modules by SC. In contrast, iNP-MSG, iNP-Qcut and iNP-SC partitioned the yeast complex interaction network into 254, 249 and 158 modules, respectively, indicating that even though the network structure of yeast network is simple, there were still some modules unresolved by using the traditional modularity-based methods. The time consumption on partitioning the yeast complex network by iNP-MSG, iNP-Qcut and iNP-SC was 9, 66 and 180 s, respectively.

To evaluate whether the partitioned modules by iNP are biologically meaningful, we have developed a FLE score to measure the quality of network division of real biological network. For details about FLE, please refer to the 'Materials and Methods' section. Briefly, FLE is the summed difference of the averaged fumsim score of all gene pairs in a module from the random fumsim score over all partitioned modules. The fumsim score between a pair of genes corresponds to the functional linkage between these two genes, and was computed by considering all GO terms associated with these two genes (34). A random funsim score for a pair of genes in the yeast interaction network is 0.21. Thus, if a module has an averaged fumsim score greater than the random funsim score, then it includes more genes that are functionally related to each other than random. Accordingly, the higher the FLE of a partitioned network, the more number of modules with enriched functionally related genes the network has. As shown in Figure 3A, before the iteration, the FLE score of the yeast network partitioned by MSG, Qcut and SC was 85.7, 85.4 and 18.8, respectively. In contrast, the FLE score of the yeast network partitioned by iNP-MSG, iNP-Qcut and iNP-SC was 104.7, 104.0 and 53.5, respectively, all

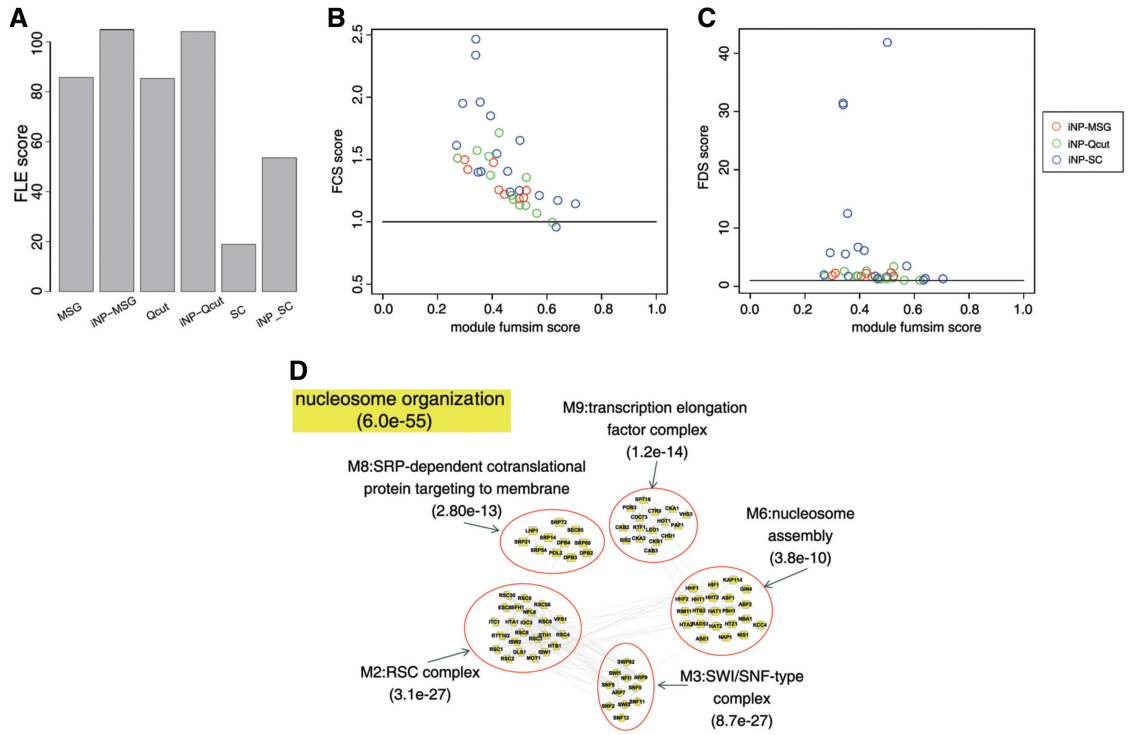

Figure 3. Partitioning the yeast complex network by iNP. (**A**) The FLE scores of the yeast networks partitioned by different methods. (**B** and **C**) show the FCS and the FDS scores of the modules initially partitioned by a non-iterative method that were further partitioned by iNP, respectively, with each circle representing a module, and the color corresponding to different iNP methods. Module funsim score is the averaged funsim score of all gene pairs in a module. For definitions of FLE, FCS and FDS, refer to 'Materials and Methods' section. (**D**) An example module initially partitioned by MSG that was further partitioned by iNP-MSG. The most significantly enriched GO terms in each module are shown next to the module (only GO terms with a size between 10 and 300 genes were considered and the threshold of the adjusted *P*-value was set <0.1).

showing significant improvement over the corresponding non-iterative methods.

However, if a module is enriched with functionally related genes, then random partitioning it into more modules may result in the increase of FLE score as well. To rule out the possibility that the improvement in FLE by iNP was due to this reason, we further developed a FCS and a FDS to inspect the modules that have been subjected for iterative partitioning. For details about both FCS and FDS, please refer to the 'Materials and Methods' section. Briefly, for a module that is further partitioned into several smaller modules by iNP, the FCS assesses the relative enrichment of functional-related genes in the new modules to that in the parent module, while the FDS evaluates whether the newly generated modules have relatively distinctive functions between each other. If the original module includes several functionally distinctive smaller modules, then a successful partition will result in both the FCS and FDS >1. In contrast, a random partitioning will result in both FCS and FDS equal to 1, while an unsuccessful partitioning will result in both FCS and FDS <1. In Figure 3B and C, we show that for all modules that have been subjected for further partitioning by iNP, both the FCS and FDS were >1, indicating that the significant increase of FLE by iNP was resulted from successfully partitioning of the original modules into functionally distinctive ones.

As an example, Figure 3D shows that the transcription elongation complex, nucleosome assembly complex, SWI/SNF-type complex, RSC complex and signal recognition particle were merged together by MSG; in contrast, iNP-MSG successfully partitioned these complexes apart. Thus, compared with traditional modularity-based methods, iNP was able to partition the yeast protein interaction network into more biologically meaningful gene modules.

Partitioning a breast cancer gene co-expression network by iNP

In addition to the yeast protein complex interaction network, we also applied iNP to partition a breast cancer gene co-expression network (see 'Materials and Methods' section for how we constructed the network). This network consists of 9112 genes, with a modularity value around 0.6 partitioned by MSG, Qcut, and SC. MSG, Qcut and SC partitioned the network into 222, 210 and 14 gene modules, respectively. Again, SC simply merged many isolated modules into large modules. The networks partitioned by traditional modularity-based methods are dominated by the largest modules, with the top four largest modules partitioned by MSG, Qcut and SC accounting for around 91, 74 and 62% of all genes in the network, respectively. In comparison, iNP-MSG, iNP-Qcut and iNP-SC partitioned the network into 412,

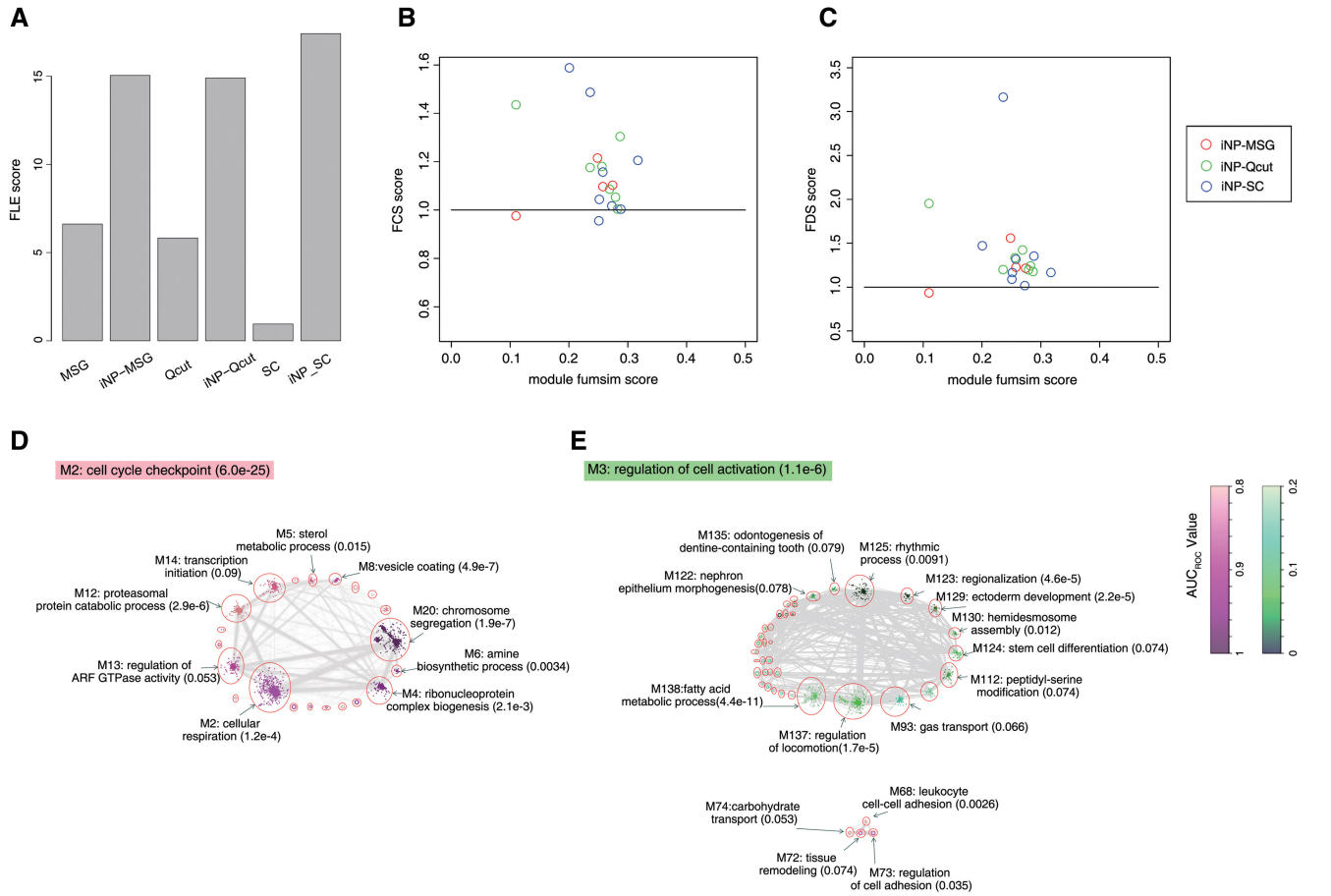

Figure 4. Partitioning the breast cancer gene co-expression network by iNP. (**A**) The FLE scores of the breast network partitioned by different methods. (**B** and **C**) The FCS and the FDS scores of the modules initially partitioned by a non-iterative method that were further partitioned by iNP, respectively, with each circle representing a module, and the color corresponding to different iNP methods. Module funsim score is the averaged funsim score of all gene pairs in a module. (**D** and **E**) show an upregulated and a downregulated cancer-specific gene module partitioned by MSG that was further partitioned into more modules by iNP-MSG, respectively. In both (D and E), *M* refers to 'module', and the numbers following *M* refer to the index of a module. The nodes inside the module are colored according to the color intensity bar that indicates the AUC_{ROC} of each module in distinguishing cancer versus normal samples. The most significantly enriched GO terms in each module are shown next to the module (only GO terms with a size between 10 and 300 genes were considered and the threshold of the adjusted *P*-value was set <0.1). The colored text in both (B and C) corresponds to the most significantly enriched GO terms of the modules partitioned by MSG.

372 and 345 gene modules, respectively, with the top four largest modules consisting of 33, 41 and 37% of all genes, respectively. As for time consumption on partitioning the breast network, MSG, Qcut and SC spent 11s, 323s and 5404s, respectively, while iNP-MSG, iNP-Qcut and iNP-SC spent 95s, 691s and 5933s, respectively.

Similar to analyzing the network partition results of the yeast network, we used the FLE scores to evaluate the quality of the partitioned breast cancer gene co-expression networks by different methods. The FLE scores of the networks partitioned by MSG, Qcut and SC are 6.6, 5.8 and 0.9, respectively. In contrast, it was improved to 15.0, 14.8 and 17.4 by iNP-MSG, iNP-Qcut and iNP-SC, respectively (Figure 4A). As shown in Figure 4B and C, for nearly all modules that were subject for iterative partitioning by iNP, both the FCS and FDS scores were >1, indicating that the improvement in FLE by iNP was resulted from successful partitioning the functionally distinctive modules unresolved by the non-iterative methods. Thus, compared with the traditional modularity-based methods, iNP was able to partition the breast cancer gene co-expression network into more biologically relevant gene modules. In addition, the extremely fast speed and the good performance of iNP-MSG in partitioning the large-scale breast cancer gene co-expression network makes it an appealing choice for analyzing large number of biological networks in parallel.

As an application of the partitioned gene co-expression modules, we further identified gene modules showing strong specificity to cancer. The details on how to identify the cancer-specific gene, co-expression modules can be found in the 'Materials and Methods' section. Briefly, the cancer specificity of a gene module is defined as the AUC_{ROC} of the module of using the median gene expression value of the genes inside the module in distinguishing cancer versus normal samples. Thus, an upregulated or a downregulated gene module will have a AUC_{ROC} greater or smaller than 0.5, respectively, and the

more deviated the AUC_{ROC} is from 0.5, the more specific it is to cancer. Figure 4D and 4E show examples of an upregulated and a downregulated gene module partitioned by MSG that were further partitioned by iNP-MSG into more cancer-specific gene modules, respectively. The upregulated module partitioned by MSG consists of 2313 genes, with an AUC_{ROC} of 0.91 in distinguishing cancer versus normal samples. The most significantly enriched biological process GO terms in this module is 'cell cycle checkpoint' ($P = 6.0e-25$). With iNP-MSG, this module was further partitioned into 65 modules, with 21 having an AUC_{ROC} >0.8. Among these modules, the most significantly upregulated one has an AUC_{ROC} of 0.98, with the most significant GO term being 'chromosome segregation' ($P = 7.9e-27$). This module includes genes well known to be related to breast cancer, such as BRCA1 (37,38), further confirming its important roles in breast cancer. The downregulated gene module by MSG consists of 2233 genes, with an AUC_{ROC} of 0.066. The most significantly enriched biological process GO term is 'regulation of cell activation' ($P = 1.1e-17$). This module was further partitioned by iNP-MSG into 89 modules, in which 37 have an AUC_{ROC} <0.2. Surprisingly, four modules have AUC_{ROC} >0.8, indicating that they were mis-classified into the downregulated module by MSG. Among them, the most significantly downregulated one has an AUC_{ROC} of 0.026, with the most significantly enriched GO term being 'rhythmic process' ($P = 0.009$). Therefore, compared with using traditional modularity-based methods, the use of iNP to partition the breast cancer gene co-expression network allowed for identification of more cancer-specific gene modules with distinctive functions, making it possible to derive biologically relevant hypothesis to investigate the mechanisms of cancer cell development and progression.

DISCUSSION AND CONCLUSIONS

Modularity is one of the most popular benefit functions used for network partition (4). However, there are inherent limitations of the modularity function in detecting dense modules (25). Though, a number of studies have been conducted to resolve the limitation, this is still an open problem to the community. In this study, we tackled this problem by developing an iNP algorithm. There are two main features of iNP. First, it provides a general framework in which any modularity-based methods can be implemented in the iterative partition step. Second, MSG, an extremely fast greedy-based method, is implemented to control the iteration step by partitioning the random graphs in order to evaluate the statistical significance of the modularity value of partitioning a pre-partitioned module. In both benchmarks of simulated networks and real biological networks, we have demonstrated that iNP achieved significant improvement in the quality of network division over traditional modularity-based methods, including one of the currently most popular one—SC developed by Newman *et al*. What makes iNP especially appealing is that with iNP-MSG, partitioning a large-scale biological network, such as the breast cancer gene co-expression network that consists of 9112 genes and 244 928 edges, only took <2 min, yet the resulted network is of better quality. Besides the breast cancer gene expression data set used to construct the co-expression network in this study, there are thousands of gene expression data sets available for human and many more for other model organisms in public databases. Each gene expression data set can be converted into a gene co-expression network, allowing for the identification of biologically meaningful modules and the inference of the function of unknown genes. Biological networks can also be constructed using other relationships, such as correlated phylogenetic profiles, sequence similarity, similar transcription binding sites, similar histone modification patterns, etc. Each of these networks can provide some knowledge about the functions of genes, while analyzing these networks in parallel can undoubtedly provide more insights toward unraveling the complicated functional organization of genes in cell. iNP that is not only accurate but also fast thus makes analysis of large number of biological networks computationally feasible.

The success of iNP can be attributed to the following two reasons. First, though modularity has limitations in detecting dense modules, we found that modularity tends to preserve the primary network structure of undetected dense modules, making it possible to develop a recursive procedure to gradually resolve the network structure. Second, the use of statistical significance to control the iteration has been well demonstrated in hQcut developed by Ruan *et al*. However, in hQcut, Qcut is used to partition both the pre-partitioned module and the generated random graphs. Given that Qcut is >20 times slower than MSG, this makes it computationally inefficient for large-scale biological network analysis. In fact, in a follow-up study by Ruan *et al*. (39), Qcut instead of hQcut was used to partition a gene co-expression network. In contrast, having observed that MSG can partition networks with similar modularity value to that by both Qcut and SC, we adopted MSG for partitioning the random graphs, which greatly reduced the time consumption of iNP.

However, because different modularity-based algorithms can be implemented in iNP, the performance of iNP is also dependent on the choice of the modularity-based algorithms. For example, though iNP-MSG improved significantly over MSG, compared with iNP-Qcut and iNP-SC, it still performs worse in simulated networks, suggesting that the choice of partitioning algorithm plays an important role in deciding the quality of final network division. However, for networks with simple structure, such as real biological networks that usually have sparse structure, iNP-MSG performed comparably to iNP-Qcut and iNP-SC. Therefore, for simple networks such as real biological networks, probably the iterative use of most modularity-based methods by iNP could achieve good performance; for relatively complex networks, in order to further improve the quality of network division, development of new and more accurate modularity-based algorithms may be necessary.

On the other hand, because iNP simply recursively partitions the pre-partitioned modules, it is unlikely to correct the mistakes resulted from a previous partition. To solve this problem, more efforts on designing a new benefit function, or allowing for shuffling of nodes in between modules, may be needed.

One of the difficulties in developing algorithms for partitioning real biological networks is to evaluate the quality of network division. Usually, the performance of an algorithm can be estimated from simulated networks. However, given that real biological networks are much different from the simulated ones, direct inferring the performance of an algorithm on real biological networks from its performance on simulated networks may be inappropriate. GO enrichment analysis has been used in a number of studies to indicate the quality of real biological network division (33,40). However, because of the complicated parent-child relationships between enriched GO terms, GO enrichment analysis is not convenient for direct comparison of the network partition results between different methods and between different types of networks. In this study, we have developed a simple FLE score that takes into consideration both the relative enrichment of functionally related genes inside a module and the number of biologically meaningful modules, allowing for direct comparison between the networks partitioned by different methods and from different types of data. In addition, we also developed a FCS and a FDS to evaluate whether further partitioning on an existing module is biologically meaningful, with FCS indicating the relative improvement in capturing functionally related genes in the newly generated modules and FDS indicating the relative functionally distinctiveness between the newly generated modules. These scores provide quantitative measurement on evaluating the quality of dividing a biological network, allowing for development of more sophisticated algorithms on partitioning the real biological networks by maximizing the FLE score. One thing worth to be noting is that when calculating the FLE score, we excluded GO annotations from computational predictions. A recent study on function association found that using predicted GO terms would benefit for making biological hypothesis (41). Therefore, in the future, we will test the use of predicted GO annotations in the FLE score. Finally, as a general method, iNP can also be readily adopted to analyze other biological networks, such as the metabolic network, enabling us to uncover more biologically interesting findings.

ACKNOWLEDGEMENTS

We thank Qingtian Gong and Guodong Zhan for their help with preparing the figures.

FUNDING

Funding for open access charge: The National Basic Research Program of China (Grant No. 2010CB529505, Grant No. 2012CB316505); Shanghai Pujiang Program (Grant No. 09PJ1401000); the National Natural Science Foundation of China (Grant No. 30971643); FDUROP, Fudan's Undergraduate Research Opportunities Program (to S.S.).

Conflict of interest statement. None declared.

REFERENCES

1. Vazquez,A., Flammini,A., Maritan,A. and Vespignani,A. (2003) Global protein function prediction from protein-protein interaction networks. *Nat. Biotechnol.*, **21**, 697–700.
2. Brun,C., Chevenet,F., Martin,D., Wojcik,J., Guenoche,A. and Jacq,B. (2003) Functional classification of proteins for the prediction of cellular function from a protein-protein interaction network. *Genome Biol.*, **5**, R6.
3. Li,S., Armstrong,C.M., Bertin,N., Ge,H., Milstein,S., Boxem,M., Vidalain,P.O., Han,J.D., Chesneau,A., Hao,T. *et al.* (2004) A map of the interactome network of the metazoan C. elegans. *Science*, **303**, 540–543.
4. Newman,M. (2004) Detecting community structure in networks. *Eur. Phys. J. B-Condensed Matter and Complex Syst.*, **38**, 321–330.
5. Segal,E., Friedman,N., Koller,D. and Regev,A. (2004) A module map showing conditional activity of expression modules in cancer. *Nat. Genet.*, **36**, 1090–1098.
6. Chen,J. and Yuan,B. (2006) Detecting functional modules in the yeast protein-protein interaction network. *Bioinformatics*, **22**, 2283–2290.
7. Altaf-Ul-Amin,M., Shinbo,Y., Mihara,K., Kurokawa,K. and Kanaya,S. (2006) Development and implementation of an algorithm for detection of protein complexes in large interaction networks. *BMC Bioinformatics*, **7**, 207.
8. Chua,H.N., Sung,W.K. and Wong,L. (2006) Exploiting indirect neighbours and topological weight to predict protein function from protein-protein interactions. *Bioinformatics*, **22**, 1623–1630.
9. Chikina,M.D., Huttenhower,C., Murphy,C.T. and Troyanskaya,O.G. (2009) Global prediction of tissue-specific gene expression and context-dependent gene networks in Caenorhabditis elegans. *PLoS Comput. Biol.*, **5**, e1000417.
10. Sharan,R., Ulitsky,I. and Shamir,R. (2007) Network-based prediction of protein function. *Mol. Syst. Biol.*, **3**, 88.
11. Eisen,M.B., Spellman,P.T., Brown,P.O. and Botstein,D. (1998) Cluster analysis and display of genome-wide expression patterns. *Proc. Natl Acad. Sci. USA*, **95**, 14863–14868.
12. Hartwell,L.H., Hopfield,J.J., Leibler,S. and Murray,A.W. (1999) From molecular to modular cell biology. *Nature*, **402**, 47.
13. Spirin,V. and Mirny,L.A. (2003) Protein complexes and functional modules in molecular networks. *Proc. Natl Acad. Sci. USA*, **100**, 12123–12128.
14. Clauset,A., Newman,M. and Moore,C. (2004) Finding community structure in very large networks. *Phys. Rev. E*, **70**, 66111.
15. Schuetz,P. and Caflisch,A. (2008) Multistep greedy algorithm identifies community structure in real-world and computer-generated networks. *Phys. Rev. E*, **78**, 26112.
16. Perlman,R. (1985) Hierarchical networks and the subnetwork partition problem. *Comput. Networks and ISDN*, **9**, 297–303.
17. Chen,F., Chen,Z., Liu,Z., Xiang,L. and Yuan,Z. (2007) Finding and evaluating the hierarchical structure in complex networks. *J. Phys. A: Math. Theor.*, **40**, 5013.
18. Agarwal,G. and Kempe,D. (2008) Modularity-maximizing graph communities via mathematical programming. *Eur. Phys. J. B-Condensed Matter and Complex Sys.*, **66**, 409–418.
19. Brandes,U., Delling,D., Gaertler,M., Goerke,R., Hoefer,M., Nikoloski,Z. and Wagner,D. (2006) Maximizing modularity is hard. arXiv: physics, 0608255.
20. Newman,M.E. and Girvan,M. (2004) Finding and evaluating community structure in networks. *Phys. Rev. E Stat. Nonlin. Soft Matter Phys.*, **69**, 026113.
21. Ruan,J. and Zhang,W. (2008) Identifying network communities with a high resolution. *Phys. Rev. E*, **77**, 016104.

22. Guimera,R. and Nunes Amaral,L.A. (2005) Functional cartography of complex metabolic networks. *Nature*, **433**, 895–900.
23. Ng,A., Jordan,M. and Weiss,Y. (2001) *On spectral clustering: Analysis and an algorithm. Advances in Neural Information Processing Systems*, Vol. 14. MIT Press, Cambridge, USA, pp. 849–856.
24. Newman,M.E. (2006) Modularity and community structure in networks. *Proc. Natl Acad. Sci. USA*, **103**, 8577–8582.
25. Fortunato,S. and Barthelemy,M. (2007) Resolution limit in community detection. *Proc. Natl Acad. Sci. USA*, **104**, 36–41.
26. Li,Z., Zhang,S., Wang,R.S., Zhang,X.S. and Chen,L. (2008) Quantitative function for community detection. *Phys. Rev. E Stat. Nonlin. Soft Matter Phys.*, **77**, 036109.
27. Reichardt,J. and Bornholdt,S. (2006) Statistical mechanics of community detection. *Phys. Rev. E Stat. Nonlin. Soft Matter Phys.*, **74**, 016110.
28. Khadivi,A., Ajdari Rad,A. and Hasler,M. (2011) Network community-detection enhancement by proper weighting. *Phys. Rev. E Stat. Nonlin. Soft Matter Phys.*, **83**, 046104.
29. Song,J. and Singh,M. (2009) How and when should interactome-derived clusters be used to predict functional modules and protein function? *Bioinformatics*, **25**, 3143–3150.
30. Yu,H., Braun,P., Yildirim,M.A., Lemmens,I., Venkatesan,K., Sahalie,J., Hirozane-Kishikawa,T., Gebreab,F., Li,N., Simonis,N. *et al.* (2008) High-quality binary protein interaction map of the yeast interactome network. *Science*, **322**, 104–110.
31. Chen,D.T., Nasir,A., Culhane,A., Venkataramu,C., Fulp,W., Rubio,R., Wang,T., Agrawal,D., McCarthy,S.M., Gruidl,M. *et al.* Proliferative genes dominate malignancy-risk gene signature in histologically-normal breast tissue. *Breast Cancer Res. Tr.*, **119**, 335–346.
32. Chen,Y. and Xu,D. (2005) Understanding protein dispensability through machine-learning analysis of high-throughput data. *Bioinformatics*, **21**, 575–581.
33. Zhang,B. and Horvath,S. (2005) A general framework for weighted gene co-expression network analysis. *Stat. Appl. Gen. Mol. Biol.*, **4**, 1128.
34. Schlicker,A., Domingues,F.S., Rahnenführer,J. and Lengauer,T. (2006) A new measure for functional similarity of gene products based on Gene Ontology. *BMC Bioinformatics*, **7**, 302.
35. Harris,M., Clark,J., Ireland,A., Lomax,J., Ashburner,M., Foulger,R., Eilbeck,K., Lewis,S., Marshall,B. and Mungall,C. (2004) The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res.*, **32**, D258.
36. Benjamini,Y. and Hochberg,Y. (1995) Controlling the false discovery rate: a practical and powerful approach to multiple testing. *J. Roy. Stat. Soc. Ser. B*, **57**, 289–300.
37. Wooster,R., Bignell,G., Lancaster,J., Swift,S., Seal,S., Mangion,J., Collins,N., Gregory,S., Gumbs,C. and Micklem,G. (1995) Identification of the breast cancer susceptibility gene BRCA2. *Nature*, **378**, 789–792.
38. Miki,Y., Swensen,J., Shattuck-Eidens,D., Futreal,P.A., Harshman,K., Tavtigian,S., Liu,Q., Cochran,C., Bennett,L.M. and Ding,W. (1994) A strong candidate for the breast and ovarian cancer susceptibility gene BRCA1. *Science*, **266**, 66.
39. Ruan,J., Dean,A.K. and Zhang,W. (2010) A general co-expression network-based approach to gene expression analysis: comparison and applications. *BMC Syst. Biol.*, **4**, 8.
40. Ulitsky,I. and Shamir,R. (2007) Identification of functional modules using network topology and high-throughput data. *BMC Syst. Biol.*, **1**, 8.
41. Hawkins,T., Chitale,M. and Kihara,D. (2010) Functional enrichment analyses and construction of functional similarity networks with high confidence function prediction by PFP. *BMC Bioinformatics*, **11**, 265.