Community Detection In Complex Networks

by
Jiahao Guo

A dissertation submitted to the Department of Physics,

College of Natural Sciences and Mathematics

in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in Physics

Chair of Committee: Kevin E. Bassler

Committee Member: Arthur B. Weglein

Committee Member: Claudia Ratti

Committee Member: Greg Morrison

Committee Member: Amy K. Sater

University of Houston
May 2021

ABSTRACT

We introduce an ensemble learning scheme and a new metric for community detection in complex networks. The scheme uses a Machine Learning algorithmic paradigm we call Extremal Ensemble Learning. It uses iterative extremal updating of an ensemble of network partitions, which can be found by a conventional base algorithm, to find a node partition that maximizes a metric. At each iteration, core groups of nodes that are in the same community in every ensemble partition are identified and used to form a reduced network. Partitions of the reduced network are then found and used to update the ensemble. The smaller size of the reduced network makes the scheme efficient. We use the scheme to analyze the community structure in a set of commonly studied benchmark networks and find that it outperforms all other known methods for finding the partition with maximum modularity. The new metric that we call generalized modularity density Q_g eliminates the well-known resolution limit problem at any desired resolution and is easily extendable to study weighted and hierarchical networks. We also propose a benchmark test to quantify the resolution limit problem, examine various modularity-like metrics to show that the new metric Q_g performs best, and show that Q_g can identify modular structure in real-world and artificial networks that is otherwise hidden.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

1.1 Fundamentals of Complex Networks

Many systems can be represented as networks or graphs [1]. A network consists of a set of nodes and links connecting them in pairs. Nodes stand for entities in the system of interest, while links reflect the relationship between them. In complex networks, we focus on those networks with non-trivial topological features. Networks that are very uniform, like lattices or random networks, are not our focus. Those networks with interesting non-trivial features are mainly real-world networks such as computer networks, social networks, biological networks, and so on [2, 3].

Depending on different application scenarios, there can be different properties of links. For instance, links can be weighted or unweighted, directed or undirected. Self-loops and multi-links can be allowed or prohibited. Nodes are usually simple but can be assigned weights in case of need. A famous example of a network is the Internet. A partial map of it is shown in Figure 1. The nodes are different IP addresses, and the links indicate the delay between those two IP addresses. This network is weighted and undirected. Another example that is popularly studied is a social network first constructed by Wayne W. Zachary in the 1970s [4]. A visualization of the Karate Club is shown in Figure 2. The nodes represent different members of the club. The link between two nodes represents their friendship, as discussed by Zachary. This network is unweighted and undirected.

A directed network can be like a hyperlink network, which describes the hyperlinks between many different web pages, or a citation network, which shows how scholars cite each other. Networks can also be static or dynamic. A dynamic network changes its topological structure by adding or removing nodes or links as it evolves. Researchers can use dynamic networks to do social simulations [5]. We can also consider the dynamics on networks, where something is evolving on the fixed network structure, such as transportation among cities. Cities are nodes, and highways are links. Cars are moving around the network and may form some patterns, depending on the network structure. A famous result is the SIR model that is used for simulating the spread of

global pandemics within an infectious population [6, 7]. In this thesis, we only consider the static networks that keep their number of nodes and links and the connections all the time. Some more examples of networks used later in this thesis can be found in Table 1.

Besides the number of nodes and the number of links, we also define some more properties of networks to study them conveniently. For example, a node's degree is the number of nodes connected to it. A connected component (or simply component) of a network is a group of nodes such that each pair can be connected by a path of links, and no more nodes in the rest of the network can be added into the group. A clique is a group of nodes that all pairs are connected. Link density (or simply density) is defined as the ratio of the number of links to the maximum possible number of links for unweighted networks. Assuming a network has n nodes, m links, and no self-loop or multi-loop, the definition is

$$Density = \frac{m}{n(n-1)/2}. \tag{1}$$

The diameter of a connected network is defined as the longest distance of all shortest paths in the network. The distance on an unweighted network can be measured by the number of links on the path between two nodes.

We can also define many connectivity measures, such as degree distribution, clustering coefficient, and assortativity. The degree distribution is a histogram of the degree of all nodes. Clustering coefficient C is a measure of how neighbors of a node are connected. More precisely, the definition of a node is

$$C = \frac{e}{k(k-1)/2}, \tag{2}$$

where e is the number of links between its neighbors and k is the number of neighbors, which is also the degree of the node. The clustering coefficient of a network is defined as the average of the clustering coefficients of all the nodes. Assortativity is a preference for the nodes to connect with other nodes that are similar in some way. For example, in social networks, nodes tend to connect with other nodes with a similar degree. On the contrary, high degree nodes tend to connect with

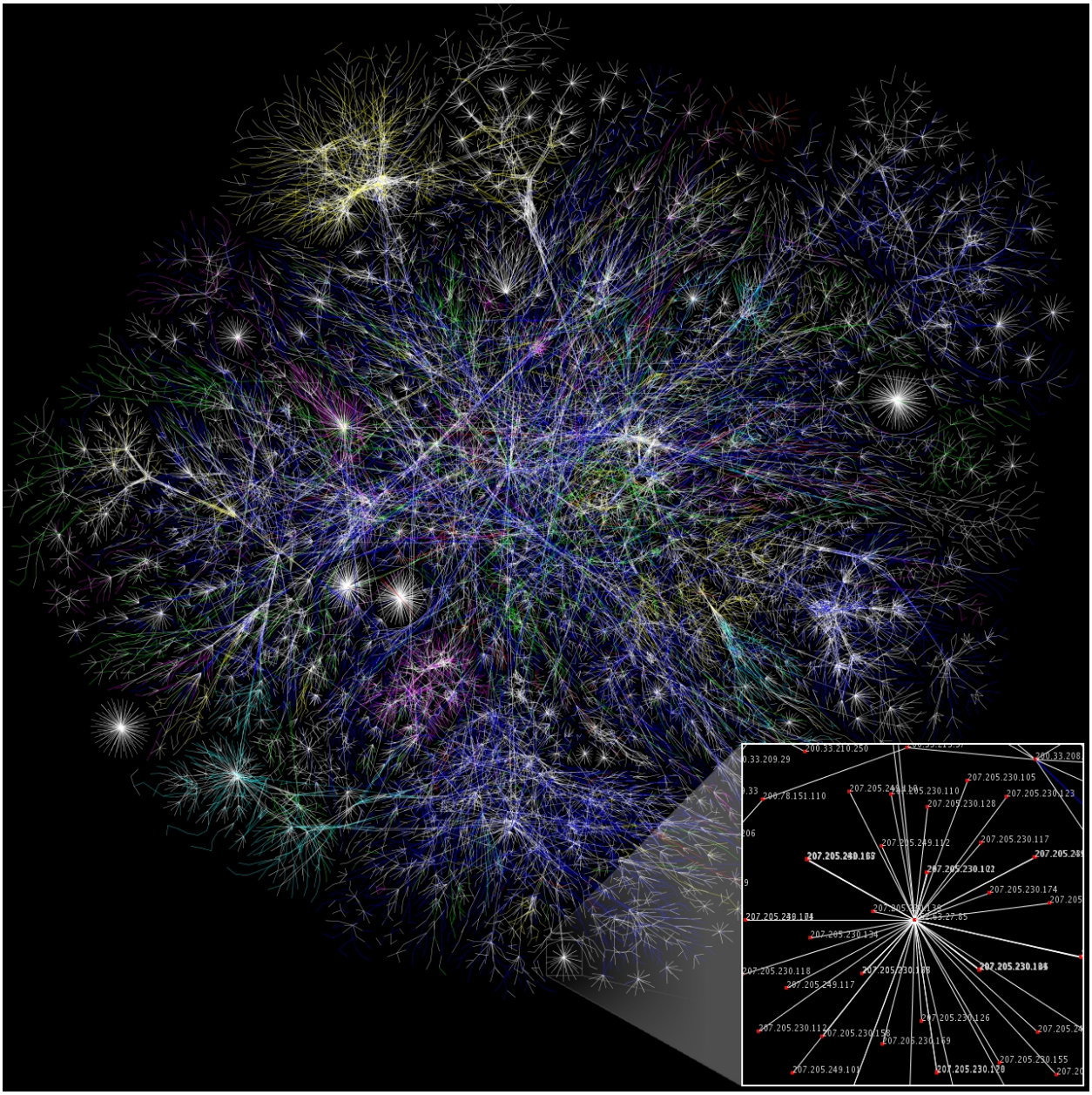


Figure 1: **Partial map of the Internet based on the January 15, 2005 data.** Each line is drawn between two nodes, representing two IP addresses. The length of a line indicates the delay between those two nodes. Figure created by the Opte Project (www.opte.org).

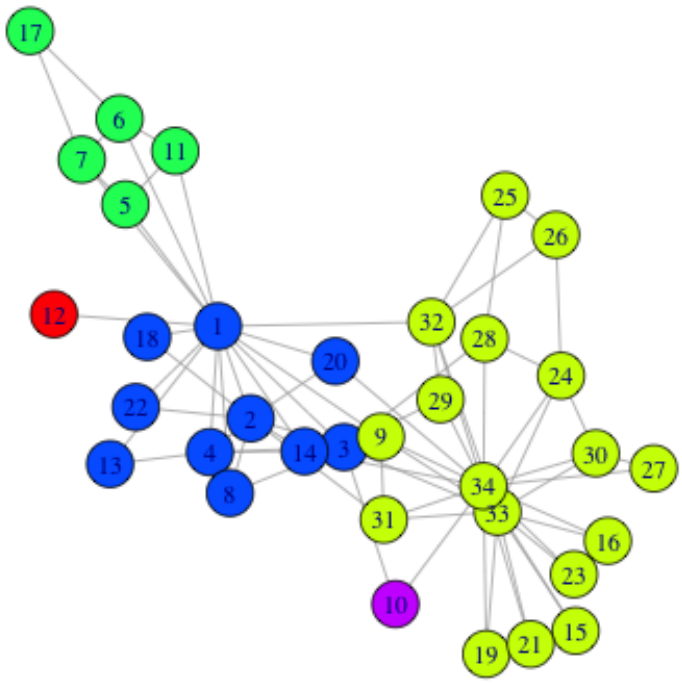


Figure 2: **Visualization of Karate Club.** Each node is a member of the club. Each link is the friendship between two members. Groups of nodes in the same color here represent communities found by some community detection algorithm.

Table 1: **Benchmark networks.** A list of empirical and synthetic networks.

Network	**Node description**	**Link description**
Adjnoun [9]	the most commonly occurring adjectives and nouns in the novel "David Copperfield" by Charles Dickens	pair of words that occur in adjacent position in the text of the book
Jazz [10]	musician	collaboration
Metabolic [11, 12, 13]	metabolites (e.g., proteins) (in C. elegans)	interaction between them
Email [14]	members	email interchanges
Polblog [15]	weblogs on US politics	hyperlink
Netscience [9]	scientists working on network theory and experiment	coauthorship
Power [16]	either a generator, a transformator or a substation	power supply line
PGPgc [17]	users of the Pretty Good Privacy (PGP) algorithm	interaction
Astro-ph [18]	scientists	coauthorship in preprints on the Astrophysics E-Print Archive between Jan 1, 1995 and December 31, 1999.
Memplus [19]	memory circuit elements	connections
As-22july06 [20]	autonomous systems	data connection
Cond-mat-2005 [18]	scientists	coauthorship in preprints on the Condensed Matter E-Print Archive between Jan 1, 1995 and March 31, 2005.
Smallworld [16]	synthetic	synthetic
CAIDARouterLevel [21]	routers	links

low degree nodes in networks like the Internet [8]. This is called disassortativity.

Centrality is another important property that measures the importance of nodes. It helps us to identify the most important nodes. Based on different application scenarios, there are many different importance measurements, which are degree centrality, closeness centrality [22], betweenness centrality [23], eigenvector centrality [24], Katz centrality [25], PageRank centrality [26], percolation centrality [27], and cross-clique centrality [28].

Two important and well-known classes of complex networks are scale-free networks [29] and small-world networks [30, 31]. A network is scale-free if its degree distribution follows a power law. The power law implies that there is no special scale of degree distribution. A network is considered a small-world network if the small-world phenomenon presents itself. The idea is that the common distance between two nodes doesn't grow much as the number of nodes increases. A more mathematical definition is that the typical distance L between two randomly selected nodes grows proportionally to the logarithm of the number of nodes N in the network as follows.

$$L \propto logN \tag{3}$$

Researchers have been using networks to explore the statistical properties of different systems for many years. Some properties such as small-world effect, high clustering coefficient, assortativity or disassortativity among nodes, and heavy-tailed degree distributions have been found to be common in many real-world networks [32]. Here we focus on another common finding, the property of community structure. Each community is a group of nodes that are tightly connected to each other, and the connection between communities is relatively loose [32]. Most real-world examples of complex networks are far from being random and have a community or modular structure within them [33, 34, 35]. The community structure has been found to control much of their dynamical or functional behavior [2, 36, 37, 38].

1.2 Community Detection

Assigning each node to some community is considered as finding a partition of the network. There are two kinds of partitioning. One is that each node only belongs to one community. This is to find the non-overlapping community structure. The other one is the overlapping community, where each node can belong to multiple communities. Overlapping community structure is found to be a significant feature of many real-world networks [39, 40, 41]. But it's more complicated to detect. And, we can use an ensemble of non-overlapping partitions to mimic the overlapping partition. In this thesis, we only focus on the non-overlapping community structure.

There are many approaches to detect the community structure. For example, the oldest algorithm is the minimum cut method [42], where the network is divided into a customized number of communities by minimizing the number of links between communities. Usually, the communities are of the same size approximately.

Another popular method is hierarchical clustering [43]. Hierarchical clustering is a classic clustering method in machine learning and data mining where each data point is described by a set of features so that the distance between two nodes is easier to define. On the contrary, clustering in networks is harder because the networks are usually sparse; thus, the distances of many pairs are missing. To do hierarchical clustering in complex networks, we first need to define distance or similarity for each pair of nodes or communities. Then, we iteratively join the nodes or communities by picking the nearest pair until there is only one community left. During this process, we obtain a hierarchical dendrogram of an ensemble of partitions.

The third method for community detection is the Girvan–Newman algorithm [32]. This method first identifies links between communities and removes them, leaving the communities detected. The identification step makes use of the betweenness centrality mentioned above.

Spectral clustering [44] is to detect the community structure by finding the eigenvalues and eigenvectors of the graph laplacian [45, 46]. Considering the matrix formed by a set of eigenvectors with the lowest eigenvalues as the new set of features for all nodes, we can then use methods like K-means to detect the community structure. The number of eigenvectors used can be a choice of

the user.

We can also detect communities using a stochastic block model of networks [47, 48]. The model generates networks with a given partition. Generally, links are generated with higher probability within communities and lower probability between communities. The community structure is determined by finding the partition with the biggest probability of generating the given network. This method is popular mainly because it has a simple mathematical definition and can be analyzed and implemented quickly. But for many real-world networks, the model doesn't reflect the actual structure; thus, the result is not helpful.

The last method we want to introduce here is the Markov Clustering (MCL) algorithm [49]. It identifies the community structure by a steady-state distribution of simulated random flows in the network. MCL is scalable and fast. However, it requires specification of external parameters (inflation, expansion). A proper choice of these parameters is important for the algorithm to converge. There is no definite way to select these parameters to ensure convergence. Other limitations of MCL include unknown convergence time and its48unsuitability for application to networks with large diameters [49].

The method that we focus on in this thesis is to detect the community structure by maximizing a metric. The metric is designed or defined to quantify the extent to which a partition is modular. Given a network G and a partition P, the metric can be considered as a function of (G, P). For example, *Modularity* Q is such a metric [32]. For an unweighted undirected network, Q can be expressed as

$$Q = \frac{1}{2m} \sum_{c} \left(2m_c - \frac{K_c^2}{2m} \right), \tag{4}$$

where m is the number of total links of the network, m_c is the number of links within a community c, K_c is the sum of degrees of all nodes in community c. Q measures the difference between the fraction of links within communities and the expected fraction if the links were randomly placed. For a given metric, this becomes just an optimization problem. Based on the well-known "No Free Lunch theorem" for optimization problems [50], we know that no method has the advantage over

any other across all possible community detection tasks theoretically [51, 52]. But in practice, for some kinds of community detection tasks, there still exists some method that is better than any other because the method successfully captures the underlying structure or organization of interest. By comparing the fraction of links within communities and the expected fraction if the links were randomly placed, Q successfully captures the community structure in many real-world networks [32].

For the convenience of following discussion, we'll elaborate the ideas by using Q, but it can be replaced by any other metric. By finding a partition that corresponds to the maximum Q_{max}, we consider the partition as the community structure of the network. Finding this partition, however, is an NP-hard problem [53]. As we can estimate, supposing a network has N nodes and at most N communities, the number of possible partitions will be of the order of N^N. To find the true maximal will take too long to make any practical sense. Thus, it is of considerable interest and importance to develop an algorithm that robustly finds an accurate solution to this optimization problem that completes in polynomial time. Besides the extremely large search space, there is another difficulty of this optimization problem. A searching algorithm will likely be trapped in a local maximal because the space formed by all possible partitions is very bumpy, given some neighboring definition [54]. The accuracy of a solution can be measured by how close the value Q of the partition found is to the value of $Q_{\max}$. Since we may never know the true maximum, any solution provides a lower bound estimate of the value of $Q_{\max}$. Therefore, the higher a solution's value of Q is, the more accurate it and its estimate of $Q_{\max}$ is.

Many optimization algorithms can be used for maximizing the metric. For example, simulated annealing is used in [55]. But the time it takes to converge can be very long if we want a decent result. A number of polynomial time complexity algorithms for finding a network partition that enables $Q_{\max}$ to be estimated have been proposed. Some are quite fast, such as random greedy agglomeration [56, 57, 58] and the Louvain method [59]. These algorithms, however, don't generally find very accurate solutions. Far more accurate solutions can generally be found with spectral clustering algorithms [9, 60] that iteratively bisect the set of nodes. The most accurate algorithm of

this type [61] combines bi-sectioning based on the eigenvector of the largest eigenvalue of the modularity matrix [9], tuning with generalized Kernighan–Lin refinements [62, 63] and agglomeration. Until recently, this was the most accurate algorithm known. Virtually all algorithms for maximizing modularity are partially stochastic, as they make random choices at intermediate steps among what are seemingly equivalent options at that point. These choices can affect the final partition, and, thus, different runs can produce different partitions. Because of this, to find the partition that provides the best estimate of the maximum modularity, algorithms are often run multiple times to produce an ensemble of partitions, and the best of those partitions is chosen.

It has, however, recently been demonstrated that partitions with even more accurate estimates of $Q_{\max}$ can be obtained with a scheme that uses information contained within an ensemble of partitions generated with conventional algorithms. This idea is known as ensemble learning. Its use distinguishes a new class of modularity maximizing algorithms [64, 65]. An ensemble learning scheme known as Iterative Core Group Graph Clustering (CGGCi) [66] was the most accurate algorithm for finding the network partition that maximizes modularity in the 10^{th} DIMACS Implementation Challenge [67]. The CGGCi scheme starts with an ensemble of partitions obtained by using a conventional "base algorithm" and identifies "core groups" of nodes that are grouped together in the same community in every partition in the ensemble. It then transforms the original network into a weighted reduced network by collapsing each of these core groups into a single "reduced" node and summing all link weights between original nodes to assign weights to the links between the reduced nodes. A base algorithm is then used to find an ensemble of partitions of the reduced network, and that ensemble is used to find a new reduced network. This procedure is iterated until no further improvement in Q is found. The best partition of the final reduced network is then mapped back onto the original network to identify the communities.

In this thesis, we introduce a different ensemble learning scheme for network community detection. It uses an algorithmic paradigm we call Extremal Ensemble Learning (EEL). Our scheme, which we refer to as Reduced Network Extremal Ensemble Learning (RenEEL) [68], starts with an ensemble of partitions obtained using a conventional base algorithm and then iteratively updates

the partitions in the ensemble until a consensus about which partition is best is reached within the ensemble. To find the partitions used to update the ensemble efficiently, core groups of nodes are identified and used to form a reduced network that is partitioned using a base algorithm. RenEEL then uses a partition of the reduced network to update the ensemble through extremal updating. We will show that an algorithm using the RenEEL scheme improves the quality of community structure discovered, especially for larger networks for which estimating the partition with $Q_{\max}$ becomes challenging. Testing our scheme on a wide range of real-world and synthetic benchmark networks, we show that it outperforms all other existing methods, consistently finding partitions with the highest values of Q ever discovered.

1.3 Community Detection Metric

Let us move back to think of the choice of the metric. A metric conveys a specific understanding of the modular structure, which is essential to community detection. Modularity, Q, though widely used, has a fundamental problem in detecting community structure. Namely, communities smaller than a certain size in a large network may not be detected. This *Resolution Limit* (RL) problem [69, 70] reduces the domain of applicability of Q and is often a significant issue when analyzing empirical networks.

In recent years, alternate metrics have been proposed to mitigate the RL problem [71, 72, 73, 74, 75, 76, 77, 78, 79]. Some of these metrics [75, 76, 77, 78, 79] are known as modularity density metrics. They have weights that are functions of the internal link density of communities in addition to the definition of modularity, Q. This thesis proposes a new metric of this form, which we call *generalized modularity density* Q_g [80]. For an unweighted undirected network, we have

$$Q_g = \frac{1}{2m}\sum_c (2m_c - \frac{K_c^2}{2m})\rho_c^\chi, \tag{5}$$

where m is the number of total links of the network, m_c is the number of links within a community c, K_c is the sum of degrees of all nodes in community c, ρ_c is the link density of community c, the

exponent χ is a control parameter. Here we assume that χ is a non-negative real number. Q_g is an extension of Q, as it reduces to Q when $\chi = 0$. The main reasons for introducing this new metric are as follows. First, it has an adjustable parameter χ that controls the resolution density of the communities that are detected. Second, Q_g can be extended to detect communities in weighted networks in a way that has a clear interpretation and is independent of the scale of the link weights.

The RL problem can be seen in the simple example of cliques arranged in a ring connected to one another in series by single links [69]. The expectation, in this case, is that the cliques should be detected as separate communities. Unfortunately, with some metrics, pairs of cliques are merged into the same community. Of course, if all possible cross-links between two cliques are present, then it is sensible to merge them into one community as they simply form a clique of larger size. However, when cliques are connected by an intermediate number of links or when the network is weighted, it is unclear whether the cliques should be merged or separated [81]. Intuitively, it makes sense to merge two cliques at a sufficiently high density of cross-links. Generally, methods of community detection that use different metrics have a different critical value for this density. The answer may also depend on the specific application being considered. Thus, it is useful to have some flexibility in allowing the communities to be separated or merged. Q_g achieves this goal by varying a parameter χ. We will show that, for a properly chosen value of χ, the partition that maximizes Q_g separates two cliques at any desired strength of inter-connectivity. This tunability of our metric is extremely useful for analyzing networks that exhibit hierarchical community structure [33], which is found in many real-world networks. A common way to investigate these hierarchical structures is to perform community detection within detected communities iteratively [82]. Using our approach, one can simply vary χ.

Finally, we compare the performance of our metric against other modularity density metrics by using them to find the structure in a more complex benchmark network than a simple ring of cliques. Our analysis indicates that Q_g performs better than all other metrics considered. We then use Q_g to find structure in a variety of empirical and artificial networks to demonstrate its ability to detect hidden community structure. We find that it eliminates the resolution limit problem

that we consider and that it is applicable to a wider range of problems than other metrics. Armed with this new metric Q_g, together with the RenEEL algorithm, we can now detect the community structure of accuracy and flexibility.

1.4 Gene Networks

Despite the recent surge in the abundance of genome-wide biological data and computational capacity, functional annotation of genes still remains challenging [83, 84, 85]. Gene metabolic and regulatory networks contain information about interactions among particular genes. Gene communities (also referred to as "clusters" or, when derived from very large diverse data, "regulons") are groups of co-expressed genes. A community of genes that displays similar expression patterns across a very wide range of experimental condition is likely to contain co-regulated genes and genes participating in the same biological process; thus, identifying gene communities is extremely important to understand their functions [86, 87, 88, 89, 90, 91]. Network-based analyses of gene expression have led to hypotheses on the roles of genes of completely unknown function, which were later experimentally verified [92, 93].

The goal of working on gene networks is to show the advantage of our newly proposed clustering method and metric (RenEEL and Q_g) in detecting gene communities that represent specific biological functions. As a case study, we apply network-based analysis to infer functional gene modules (*communities*) in the model species *Saccharomyces cerevisiae* (yeast). We use RNA abundance profiles generated from raw RNA-Seq data representing the expression of 6692 annotated genes across 691 runs from 44 studies, including samples representing a wide range of developmental conditions, strains, growth media, and times [94] to construct gene co-expression networks using Pearson's pairwise correlations and Context Likelihood of Relatedness (CLR) approaches. Then, we use four community detection methods to find communities of interest within the Pearson's and CLR gene networks. Specifically, these are (i) Markov Clustering (MCL), (ii) Modularity, (iii) Excess Modularity Density, and (iv) Generalized Modularity Density. The clusters found by each of these methods are compared to the *S. cerevisiae* genes and Gene Ontology (GO) leaf term

(GO terms having no children) associations [95]. The statistical overrepresentation of genes in the clusters is determined using the hypergeometric test. By comparing gene communities identified by each method with the GO term assigned to each gene, we find that Q_g communities perform the best.

2 Methods

2.1 Reduced Network Extremal Ensemble Learning

2.1.1 Community detection via modularity maximization

We will use Modularity Q as the metric to elaborate the use of Reduced network Extremal Ensemble Learning (RenEEL). The first term in Equation 4 is the fraction of links inside communities, and the second term is the expected fraction if all links of the network were randomly placed. For a weighted network, m_c, K_c and m are sums of link weights instead of numbers of links. Modularity measures the deviation of the structure of a network partition from that expected in a random null model. The *community structure of a network* corresponds to the partition P that maximizes Q. The number of communities in P is free to vary. The challenge of detecting the community structure of a network, therefore, is to find the partition with the maximum modularity $Q_{\max}$.

2.1.2 Reduced networks

To find a reduced network G' starting from a network G and an ensemble of partitions of it $\mathcal{P}$, we first identify the core groups in G. A *core group* is a set of nodes that are found together in the same community in every partition in the ensemble. Any node that is not found in the same community with some other node in every partition in $\mathcal{P}$ is itself a core group. G' is then formed by collapsing core groups of nodes into single nodes and combining their links to other nodes by summing their weights. An example of this is shown in Figure 3. Each circle containing multiple nodes of G that are colored the same in Figure 3(a) denotes a core group. Two nodes that do not belong to any circle are shown in black and dark green. The core groups are collapsed to single nodes of the same color in the reduced network G' shown in Figure 3(b). The link weights in the reduced network are the sum of link weights between core groups in the original network. The weighted self-loops in G' result from the total internal weights of the core groups in G.

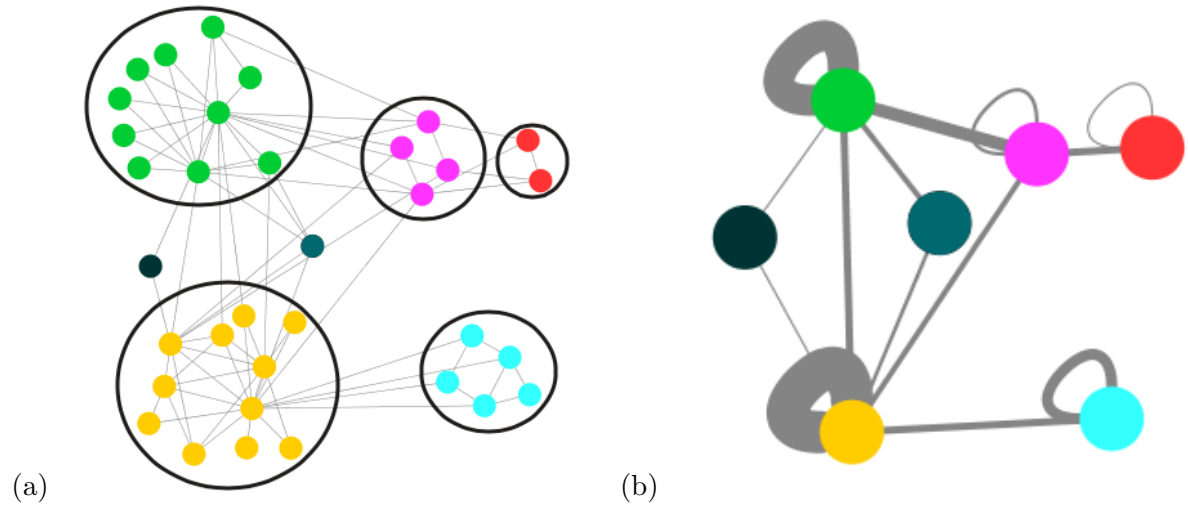


Figure 3: **Construction of a reduced network.** (a) An example network showing seven core groups of nodes. The nodes of the same color belong to the same core group. The nodes inside each of the five circles are collapsed to single nodes in the reduced network, while the two isolated nodes remain isolated. (b) The reduced network after collapsing the core groups into single nodes. The nodes in the reduced network are colored according to the core group nodes in the original network and thickness of each link is proportional to its weight.

2.1.3 RenEEL scheme

The RenEEL scheme is summarized in the flowchart[1] shown in Figure 4 and Figure 5. It is also described as follows. First, an ensemble $\mathcal{P}$ of at most k_{max} partitions P of the network G is obtained from multiple runs of a base algorithm. The base algorithm can be, for example, any of the conventional ones that have been developed to find the partition with $Q_{\max}$. Alternatively, a set of base algorithms can be used to find $\mathcal{P}$. The partitions in $\mathcal{P}$ are then ordered according to their modularity values, from the one with the largest value P_{best} to the one with the smallest value P_{worst}. Next, the core groups of nodes in the ensemble $\mathcal{P}$ are identified and used to construct the reduced network G'. An ensemble $\mathcal{P}'$ consisting of k' partitions P' of G' is then obtained using a base algorithm. The base algorithm used for this step can either be the same as or different from the base algorithm used to find $\mathcal{P}$. The steps in which a base algorithm is used to find the ensembles $\mathcal{P}$ and $\mathcal{P}'$ are shown in red in Figure 4. The partition in $\mathcal{P}'$ with the largest modularity value P'_{best}

[1]The flowcharts adhere to the ISO 5807:1985 standard (Information processing – Documentation symbols and conventions for data, program and system flowcharts, program network charts and system resources charts).

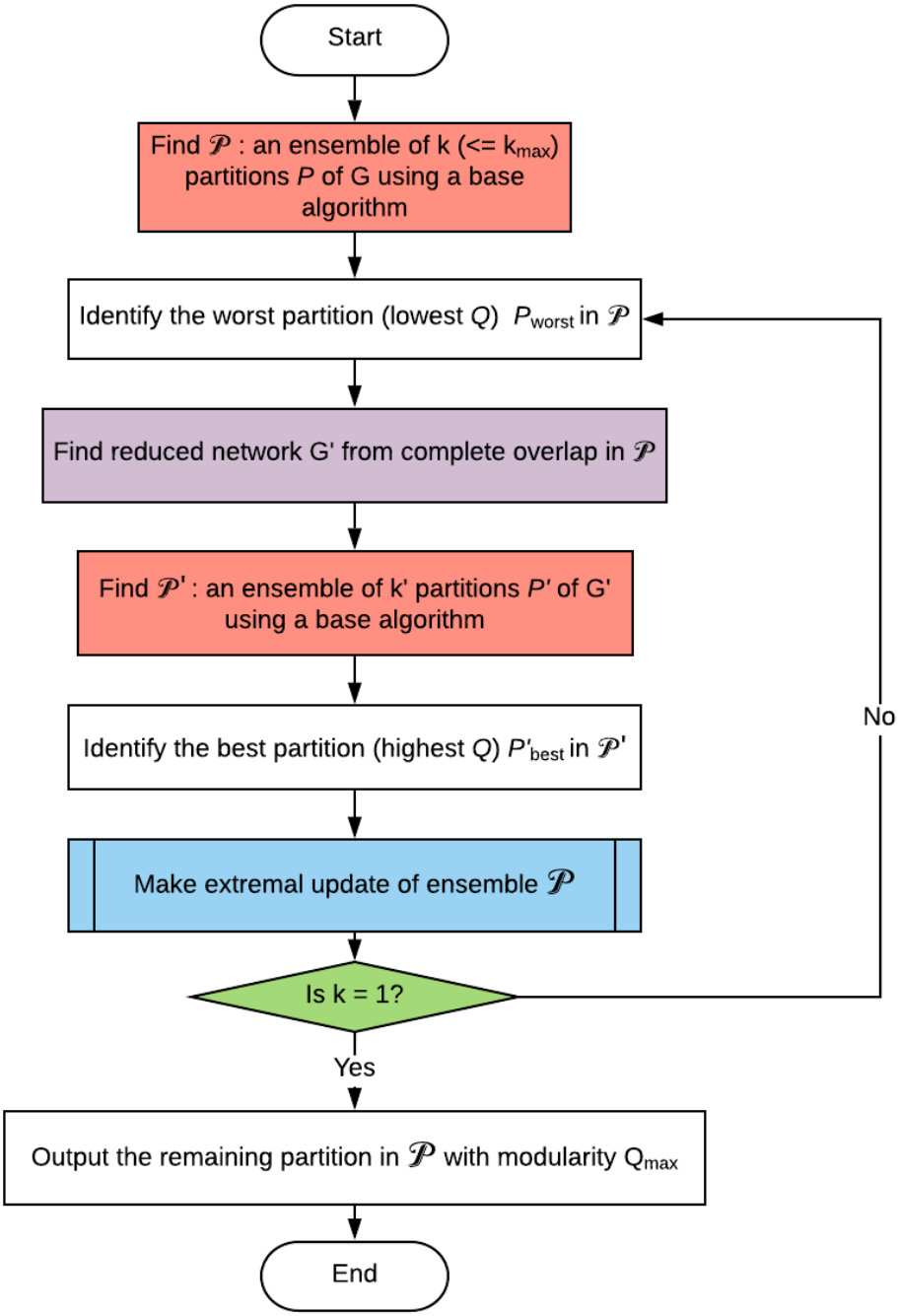


Figure 4: **The RenEEL scheme.** The steps of an efficient ensemble learning scheme to find the network partition that maximizes modularity Q are shown in this flow chart. In the two steps shown in red a base algorithm is used to obtain an ensemble of partitions. The step shown in purple collapses the core groups to find the reduced network. The ensemble $\mathcal{P}$ gets updated with extremal criteria in the step shown in blue and is described in Figure 5.

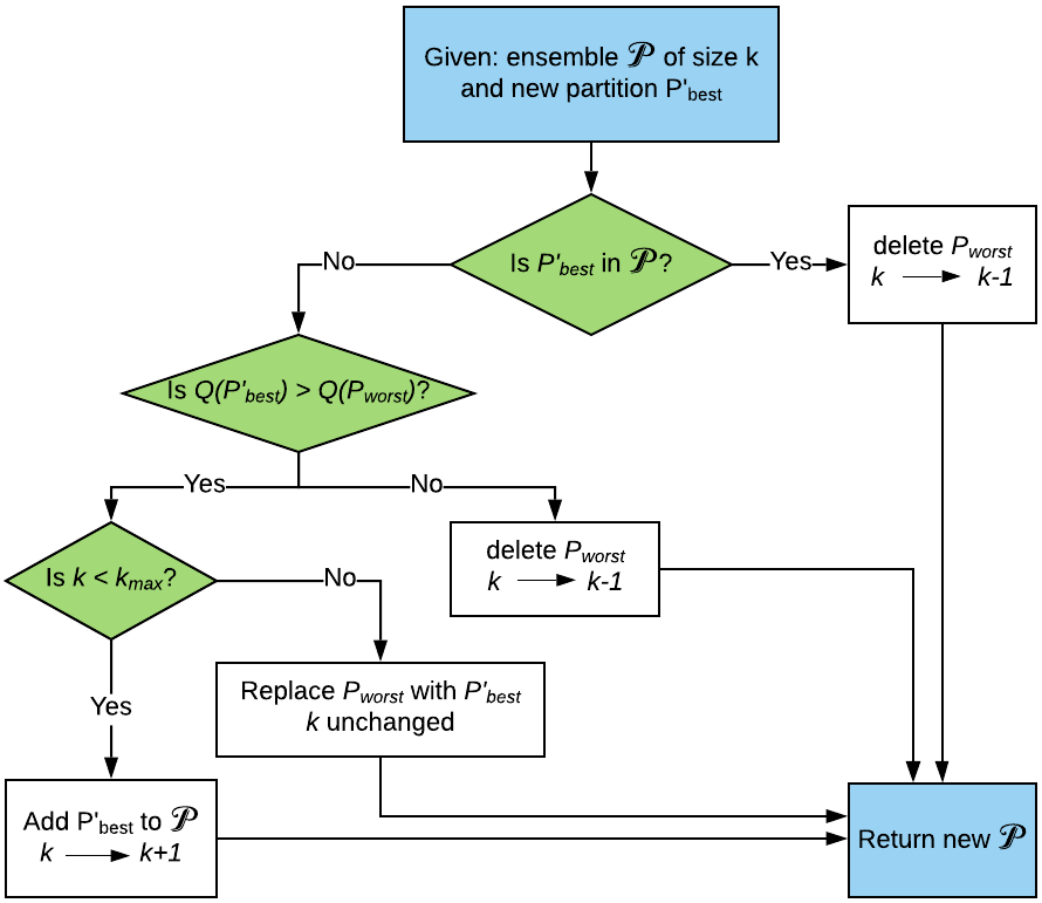


Figure 5: **The procedure of the extremal updating of ensemble $\mathcal{P}$.** The updating steps guarantee algorithmic termination in a finite network.

is then identified and used to perform an extremal update of ensemble $\mathcal{P}$. This step is shown in blue in Figure 4 and detailed in Figure 5. If $Q(P'_{\text{best}}) > Q(P_{\text{worst}})$, then P'_{best} is expanded into a partition of G and either used in place of P_{worst} in $\mathcal{P}$ (if $k = k_{max}$) or added to the ensemble $\mathcal{P}$ (if $k < k_{max}$) as shown in Figure 5. In doing so $\mathcal{P}$ is enriched with a better quality partition. However, it is possible that at any iteration either P'_{best} is already contained in $\mathcal{P}$, or $Q(P'_{\text{best}}) < Q(P_{\text{worst}})$. In both cases, in order to move toward consensus within $\mathcal{P}$, its current size k is reduced by 1 by deleting P_{worst} from it. This procedure is repeated until there is only one partition left in the ensemble $\mathcal{P}$. This consensus partition is the optimal partition that defines the communities of the network.

2.1.4 Computational complexity, base algorithms and practical implementation

The most computationally complex and time consuming steps of the RenEEL scheme are those that use a base algorithm to find an ensemble of partitions. These steps are colored in red in the flowchart in Figure 4. Assuming that the size of the ensembles $\mathcal{P}$ and $\mathcal{P}'$ are fixed, the computational complexity of executing these steps is simply a fixed multiple of the computational complexity of the base algorithm used. Worst-case scaling of the computational complexity of base algorithms is between $\mathcal{O}(n^2)$ and $\mathcal{O}(n^3)$, where n is the number of nodes in the network. All other steps of the scheme have less complexity; the steps of network reduction, colored purple in Figure 4, and network expansion both have a worst case computational complexity that is $\mathcal{O}(n^2)$, and the rest all have a computational complexity that is $\mathcal{O}(1)$. Thus, since each iteration of the scheme has only one step that uses the base algorithm a fixed number of times, each iteration has a computational complexity that scales the same as that of the base algorithm used. As the scheme progresses, however, the size of the reduced network monotonically decreases, significantly increasing the speed of later iterations. The number of iterations required for scheme to complete appears to grow approximately linearly with $k_{\max}$. Thus, the overall complexity of a RenEEL algorithm is expected to scale as the base algorithm times k' times $k_{\max}$.

The base algorithm used to obtain the results presented in this thesis is a randomized greedy agglomerative hierarchical clustering algorithm [58]. It is commonly used to find the community structure in complex networks [66] and has an expected time complexity that scales as $\mathcal{O}(m \ \ln n)$ [58], where m is the number of links in the network. There can be, at most, $\mathcal{O}(n^2)$ links. The overall worst case complexity of the algorithm used here scales approximately as $\mathcal{O}(k_{\max} k' n^2 \ \ln n)$. The particular choice of parameters k_{max} and k' is important for the quality of community structure as well as the computational time. In general, higher k' and k_{max} yield higher $Q_{\max}$.

2.1.5 Co-clustering analysis

In order to visualize the evolution of the clustering results in the RenEEL scheme, co-clustering matrices at various stages of the scheme are shown in Figure 6. In Figure 7 the results of the core group co-clustering at the different stages are combined to show their evolution. A co-clustering matrix S is a matrix whose elements s_{ij} are defined as the fraction of times node i and node j are in the same community in an ensemble of partitions $\mathcal{P}$. The order of the nodes in Figs. 6 and 7 was determined using simulated annealing to optimize the block-diagonal structure of the matrices. Starting from a random ordering of the nodes, their order was rearranged to minimize a cost function, or "Hamiltonian", that is a function of minimum distance of matrix elements (i, j) from the diagonal d_{ij} assuming periodic boundary conditions on the order:

$$H = \sum_{i<j} s_{ij} \ d_{ij}^{\alpha}, \tag{6}$$

where α is an arbitrary factor that controls the non-linear dependence of H on d_{ij}. The results in Figs. 6 and 7 were obtained using $\alpha = 3$. Simulated annealing seeks to find the order of nodes that minimizes H. For the Monte Carlo updates in our simulated annealing, Metropolis rates [96] with Boltzmann factor $e^{-(\Delta H)/T}$ were used. Starting from a relative high temperature where the order of the nodes is random, the temperature was systematically lowered each Monte Carlo step until the node order stabilized.

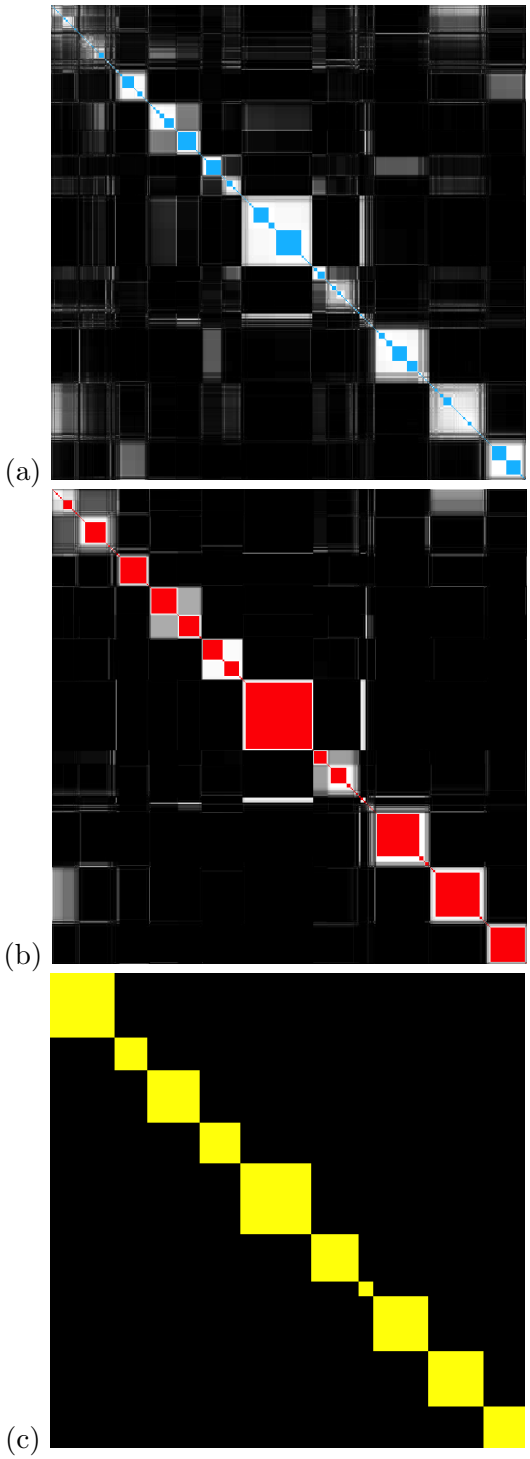


Figure 6: **Ordered co-clustering matrix with core groups.** Co-clustering matrix after the nodes have been reordered by simulated annealing. (a) after the first iteration (b) at the intermediate stage (c) at completion. The intensity of white in each pixel is proportional to the co-clustering frequency of the corresponding pair of nodes, except when the pair of nodes are always grouped together and, thus, belong to the same core group. In that case the pixel is colored blue in (a), red in (b), and yellow in (c).

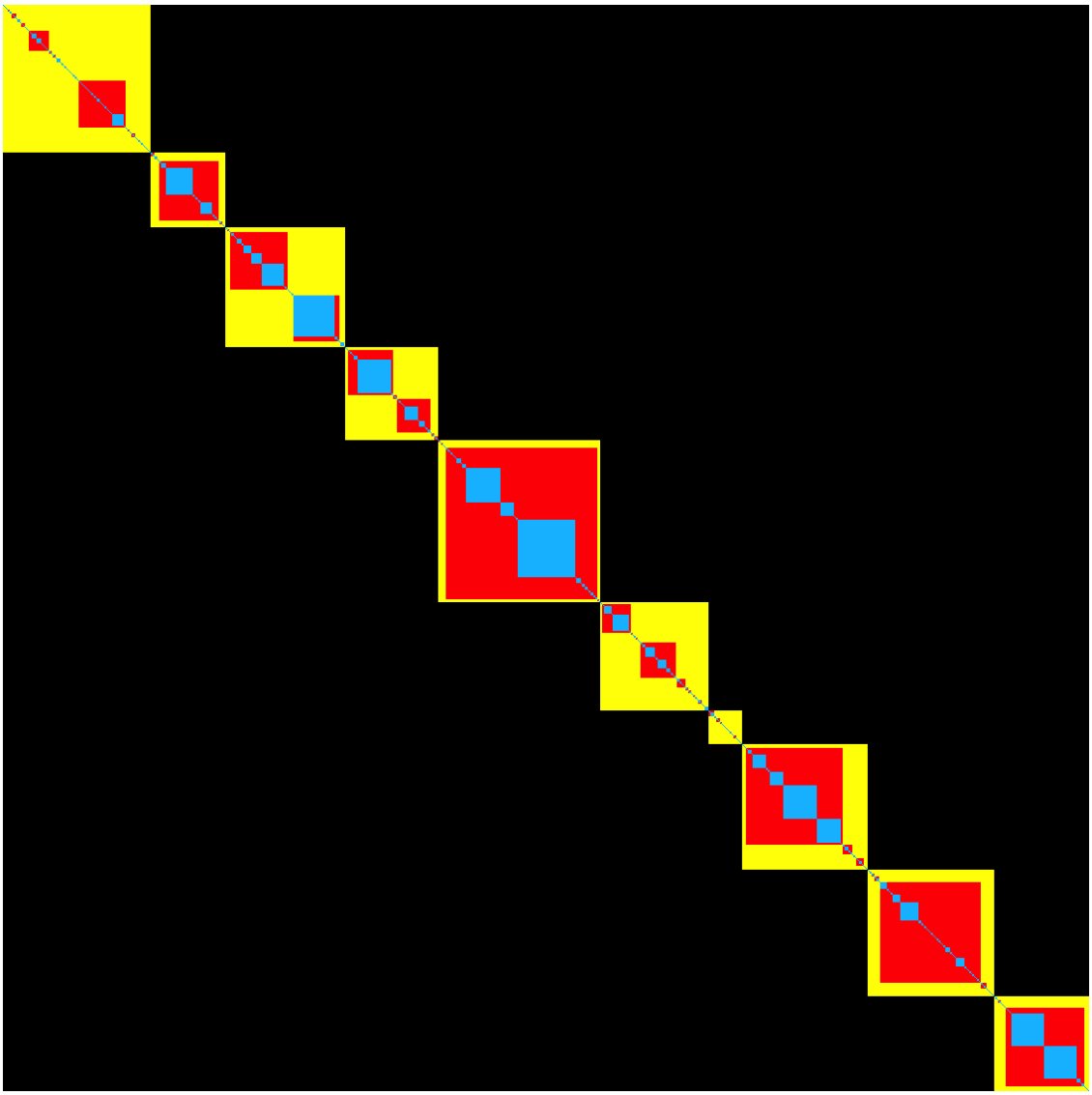


Figure 7: **Growth of core groups.** Colors blue, red, and yellow represent the core groups after the first iteration, at an intermediate stage, and at the end when the core groups have reached a stable state, respectively. The core groups can only grow. The process is agglomerative.

To get the three co-clustering matrices shown in Figure 6, which respectively show results at the initial, intermediate, and final stages of the RenEEL scheme, the following procedure was used in the simulated annealing Monte Carlo. First, nodes were reordered by considering swaps of random pairs of nodes so as to minimize H in the final stage co-clustering matrix. Then, swaps of pairs of final stage core groups and swaps of pairs of nodes within the final stage core groups were considered, to minimize H in the intermediate stage co-clustering matrix. Finally, swaps of pairs of final stage core groups, swaps of pairs of intermediate stage core groups within finial stage core groups, and swaps of pairs of nodes within the intermediate stage core groups were considered to minimize H in the initial stage co-clustering matrix. The order of nodes that resulted is used in all three co-clustering matrices in Figure 6 and in Figure 7.

2.1.6 Benchmark networks used for comparison

To test the effectiveness of our methods of community detection we studied a set of networks that were used in the 10^{th} DIMACS challenge [67]. The networks we studied are listed and described in Table 1. These networks have been complied from various sources and cover a wide range of sizes, functions and other characteristics. Hence they are often used as benchmarks for testing community detection methods. The network data for Email, Jazz, PGPgc, Metabolic can be downloaded from [97], Adjnoun, Polblog, Netscience, Power, Astro-ph, As-22july06, Cond-mat-2005 can be downloaded from [20], Memplus can be downlaoded from [98], and Smallworld can be downloaded from [99].

2.2 Generalized Modularity Density

2.2.1 Metric properties

We define the *Generalized Modularity Density* of a node partition of unweighted network as

$$Q_g = \frac{1}{2m}\sum_c (2m_c - \frac{K_c^2}{2m})\rho_c^\chi \tag{7}$$

where m is the number of total links of the network, m_c is the number of links within a community c, K_c is the sum of degrees of all nodes in community c, ρ_c is the link density of community c, the exponent χ is a control parameter. Here we assume that χ is a non-negative real number. The link density of a community is the ratio of the number of links that exist in c to the number of possible links that can exist

$$\rho_c = \frac{2m_c}{n_c(n_c - 1)} \,, \tag{8}$$

where n_c is the number of nodes in c. Q_g is an extension of modularity, i.e. at $\chi = 0$, $Q_g = Q$.

The metric Q_g, like the Modularity metric Q (Equation 4), can be easily extended to weighted networks. For Q this is done by simply replacing the number of links with the sum of link weights in m, m_c and K_c [100, 101]. Extending the definition of modularity density metrics to weighted networks is complicated by the fact that they depend on link density, and link density can be problematic to use with weighted networks. One way to deal with these problems is to simply ignore the link weights and calculate the link density as if the network was unweighted [75, 78]. Unfortunately, this loses the information contained in the link weights. The correct way is to use a normalized definition of link density, where the sum of the weight of all internal links divided by the maximum value that sum would have if the community were fully connected with links of weight equal to the maximum weight of any link in the network,

$$\rho_c = \frac{2m_c}{n_c(n_c - 1)w_{\max}} \tag{9}$$

where m_c is the sum of the weights within community c, n_c is the number of nodes in c, and $w_{\max}$ is the maximum weight of any link in the network. This definition of ρ_c is consistent with the definition for unweighted networks, but it can be problematic because it involves the global variable $w_{\max}$. The community structure found using some metrics, such as those proposed in Refs. [75, 77, 79], can be very sensitive to the value of $w_{\max}$. This makes their use potentially troublesome, especially in empirical studies where the value of $w_{\max}$ can be difficult to accurately measure. Additionally, if there is a wide distribution of link weights and $w_{\max} \to \infty$, then $\rho_c \to 0$

for all communities and the algorithms for finding the partition that maximizes the modularity density metric become numerically unstable.

Generalized Modularity Density, unlike other modularity density metrics, does not have problems with $w_{\max}$. Both $2m_c$ and $\frac{K_c^2}{2m}$ in Equation 7 are weighted by the same function $\rho_c^\chi = (\frac{2m_c}{n_c(n_c-1)w_{max}})^\chi$, where w_{max} can be factored out and simply modifies the value of Q_g for every possible partition by the same constant factor. It is, thus, irrelevant for determining the partition that maximizes Q_g. So, instead of the absolute link density, Equation 9, a relative link density, given by Equation 8 with m_c being the sum of the weight of links in c, can be used in the metric Q_g without affecting results. The community partitions found with Generalized Modularity Density are also independent of the scale of the link weights. As it is with Modularity, multiplying all link weights by a common factor does not affect the results obtained with Q_g. This important property is needed for preserving the information in the link weights.

2.2.2 Resolution Density

The Resolution limit (RL) problem can be viewed as a problem with a metric, when using it yields a partition that merges two "well separated" communities. A resolution-limit-free metric is expected to resolve these communities. Conversely, a metric should also avoid splitting two groups of nodes that are "well connected" to each other. The RL problem is clear at these two extremes. However, more generally, the notion of well separated/connected communities is not well defined. It is unclear whether two partially connected communities should be merged or not.

Consider the benchmark network shown in Figure 8. This network consists of three parts: two cliques and an external arbitrary component to which the cliques are weakly connected. As the cliques are fully connected, they have no internal community structure. Assume clique 1 has n_1 nodes, clique 2 has n_2 nodes, and both n_1 and $n_2 \geq 3$. Without loss of generality, we assume $n_2 \geq n_1$. Let m_{12} be the sum of weights of links between the two cliques, and let m_{1a} and m_{2a} be the sum of the weights of links that connect each clique with the arbitrary component. n_a and m_a are number of nodes and the sum of weights of links within the arbitrary component, respectively.

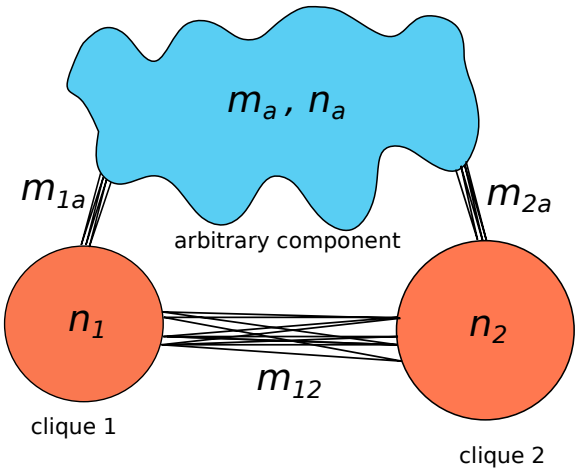


Figure 8: **Benchmark network for studying the resolution limit problem.** The network consists of two cliques of sizes n_1 and n_2 and an arbitrary component with n_a nodes and m_a links. The two cliques share m_{1a} and m_{2a} links with the arbitrary component, respectively, and have m_{12} links between. The links of the network can be weighted, in which case, m_a, m_{1a}, m_{2a} and m_{12} are the sums of link weights.

Also, assume that $m_{1a} \ll n_1^2 w_{\max}$ and $m_{2a} \ll n_2^2 w_{\max}$, so that the cliques are only weakly connected to the arbitrary component. The RL question concerning this network is whether or not the two cliques should be merged or split, and whether or not using a given metric will meet this expectation. This choice of network gives greater flexibility to explore the RL problem than a simple ring of cliques, since the external component can have an arbitrary structure and the strength of inter-connectivity between the two cliques can be varied. Generally, there is a threshold, or critical, value of m_{12} below which the cliques are separated and above which they are merged. We impose an arbitrary *expected critical value* $m_{\exp}$ such that the cliques should be merged if $m_{12} \geq m_{\exp}$ and separated if $m_{12} < m_{\exp}$. Instead of using the values of m_{12} and $m_{\exp}$, it is convenient to use normalized inter-clique link density

$$d = \frac{m_{12}}{n_1 n_2 w_{\max}} \tag{10}$$

and normalized expected critical resolution link density

$$\delta_{\exp} = \frac{m_{\exp}}{n_1 n_2 w_{\max}} \, . \tag{11}$$

For unweighted networks, $w_{\max} = 1$.

Given a metric, we can examine the RL question in the benchmark network. For a given set of network parameters, the two cliques are either merged or split. Accordingly, the parameter space can be divided into Merged (M) and Split (S) phases. The value of the link density at the boundary of the two phases is δ. At the same time, there is an expected result, corresponding to a specific understanding of the problem, given by δ_{exp}. The metric can then be evaluated by comparing the results obtained by using it with the expected results. Specifically, we define a resolution-limit-free metric as one for which $\delta \geq \delta_{\text{exp}}$ for all parameters of the benchmark network. Then, the metric is resolution-limit-free with respect to the expected resolution density.

3 Results

3.1 Reduced Network Extremal Ensemble Learning

3.1.1 Evolution of core groups

The essence of how the RenEEL scheme works and why it is efficient can be seen by the evolution of the co-clustering of the nodes across the ensemble $\mathcal{P}$. Figure 6 shows the co-clustering results during a typical realization of the scheme on the Email network [14] (see Table 1) at the initial, intermediate and final stages. In the three sub-figures, the intensity with which a pixel (i, j) is colored white corresponds to the frequency that nodes i and j are in the same community in the member partitions of $\mathcal{P}$. The pixels colored blue, red and yellow indicate that the nodes are in the same community in all member partitions. The nodes in the blue, red and yellow blocks on the diagonal are the core groups that are used to form that reduced network. Nodes are listed in the same order in each of the three sub-figures. Figure 7 shows the evolution of just the core groups in the same realization.

The Email network has $n = 1133$ nodes. Initially, as shown in Figure 6(a), there are 446 core groups, most of which contain only one or two nodes. After 100 iterations of the scheme, as shown in Figure 6(b), the number of core groups is reduced to 192. Finally, in the stable state,

after about 300 iterations of the scheme, only 10 core groups remain, as shown in Figure 6(c). This reduction, from the original network of 1133 nodes to a reduced network of 10 nodes, is a tremendous simplification and greatly improves the overall speed of network clustering.

Within a network G it is generally "easy" to determine that certain groups of nodes should be clustered together. All partitions group them together. These are the core groups of nodes. The hard work in finding the optimal partition is to determine whether nodes that are grouped together in only some of the partitions should indeed be in the same community, that is, to determine whether or not core groups should combine. This is precisely what RenEEL focuses on. The formation and evolution of core groups in RenEEL is an agglomerative process [102]. Once a core group is formed, RenEEL never subsequently divides it. As the scheme progresses, core groups grow and merge with each other and the number of core groups monotonically decreases.

3.1.2 Evolution of the ensemble $\mathcal{P}$

A defining characteristic of RenEEL is that the ensemble of partitions $\mathcal{P}$ evolves as the scheme progresses. The ensemble "learns" what the partition with $Q_{\max}$ is by using extremal updating to incorporate new partitions, replace existing ones with higher quality ones, or remove low quality partitions. The new partitions are partitions of the reduced network G'. They are used in RenEEL to improve the quality of $\mathcal{P}$ at every iteration of the scheme until a consensus is reached about what the optimal partition is.

A typical way that $\mathcal{P}$ evolves as the scheme progresses can be seen with the results shown in Figure 9 from an example run of RenEEL that partitions the As-22july06 network [20]. (Also see table 1.) In this example run, $k_{\max} = 100$ and $k' = 20$. Figure 9(a) and (b) show the modularity value Q of P_{best} the best partition in $\mathcal{P}$ (red dots), of P_{worst} the worst partition in $\mathcal{P}$ (black dots), and of P'_{best} the new partition of G' considered for the enrichment of $\mathcal{P}$ (blue dots) as a function of the number of iterations. The main panel of Figure 9(a) shows the full results of the scheme, from start to finish. An enlarged view of the results for the initial 150 iterations is shown in the inset of Figure 9(a). The main panel of Figure 9(b) shows an enlarged view of the vertical Q axis

near the final result of the entire scheme. An enlarged view of both axes at the end stages of the scheme is shown in the inset of Figure 9(d). Figure 9(c) shows the size of the reduced network, or equivalently the number of core groups, as a function of the number of iterations. The main panel of Figure 9(c) shows the results on linear axis scales, and the inset shows the same results on log scales. Figure 9(d) shows the ensemble size k as a function of the number of iterations.

In the example run, as can be seen from the inset of Figure 9(a), for the first 100 iterations the modularity of the new partitions $Q(P'_{best})$ are all significantly better than that of the worst in the ensemble $Q(P_{worst})$. In fact, all the first 100 new partitions generated by RenEEL are better than every one the 100 original ones in $\mathcal{P}$ generated by the base algorithm. (The number of partitions in $\mathcal{P}$ initially is $k_{\max} = 100$.) So, for the first $k_{\max}$ iterations RenEEL systematically replaced each of the original partitions. There is large increase in $Q(P_{worst})$ at iteration 100. Although it's difficult to see in the figure, there are other similar, significant increases in $Q(P_{worst})$ at iterations 200 and 300, indicating that RenEEL also replaces its first and second 100 new partitions with entirely new sets in the second and third 100 iterations, respectively. After the first 300 iterations, the quality of the new partitions starts to become comparable to the existing partitions. Throughout the process, the $Q(P_{best})$ intermittently raises when a new best partition is discovered.

Figure 9(c) shows that the size of the reduced network keeps decreasing as the scheme progresses. It initially decreases exponentially, then there is what appears to be a power-law decay from iteration 100 to iteration 1000 (see inset of Figure 9(c)), followed by a sharp, perhaps exponential, decay in the final iterations of the scheme. The original size of this network, $n = 22963$, is reduced to 38 core groups at the termination step. The size of the ensemble, shown in Figure 9(d), varies when new partitions are discovered and added to $\mathcal{P}$ or when low quality partitions are deleted as the scheme drives $\mathcal{P}$ toward consensus. The plot shows that, as the ensemble learns, its size grows and shrinks multiple times before its size falls to unity and the scheme terminates. There are two main periods in which the size of the ensemble grows, one beginning at about iteration 900 and the other at about iteration 1200. During these periods the value of $Q(P_{best})$ increases quickly, as can be seen in thee main panel and inset of Figure 9(b). These are periods when the ensemble $\mathcal{P}$ has

made a "breakthrough" by discovering a new set of high quality partitions. The example run ends with a consensus choice that a partition with modularity $Q_{\max} = 0.678579$ is the optimal one for this network, a value higher than any previously reported results. (See Table 2.)

3.1.3 Distribution of results for $Q_{\max}$

Since virtually all conventional algorithms are stochastic, ensemble learning schemes that use them as base algorithms will also be stochastic. Thus, a range of results for $Q_{\max}$ are possible with each realization of virtually all methods of modularity maximization. As an example, Figure 10 shows the distribution of $Q_{\max}$ that three different methods of community detection produce for the Email network. For each method, results from 250 realizations are shown. Results from the RenEEL, CGGCi ensemble learning schemes, and naive ensemble analyses are shown in red, green, and blue, respectively. The results for all three of these schemes were obtained using a randomized greedy algorithm as the base algorithm and an ensemble size of $k_{\max} = 100$. Each of the blue data points were obtained by running the algorithm 100 times and choosing the largest value of $Q_{\max}$ from those runs. The distributions from the three different methods are all non-overlapping, with the RenEEL results having the largest values, followed those of CGGCi and then those of the naive ensemble analyses with the conventional algorithm. The distribution of $Q_{\max}$ for RenEEL is also narrower than those of the other two schemes, which suggests that the results from RenEEL are close to the true value of $Q_{\max}$ for the network.

3.1.4 Application to benchmark networks

To test the accuracy of the RenEEL scheme, we applied it to the benchmark networks listed in Table 1. In Table 2, the maximum modularity value $Q_{\max}$ found for these networks by RenEEL is compared to the best previously published values. Many of these values were the best result in the 10^{th} DIMACS challenge [103]. To be consistent, all realizations had $k_{\max} = 100$ and $k' = 20$ and used the randomized greedy algorithm as a base network. 100 different realizations of RenEEL were run on the smaller networks, up to and including the Netscience network, and 5 were run on the

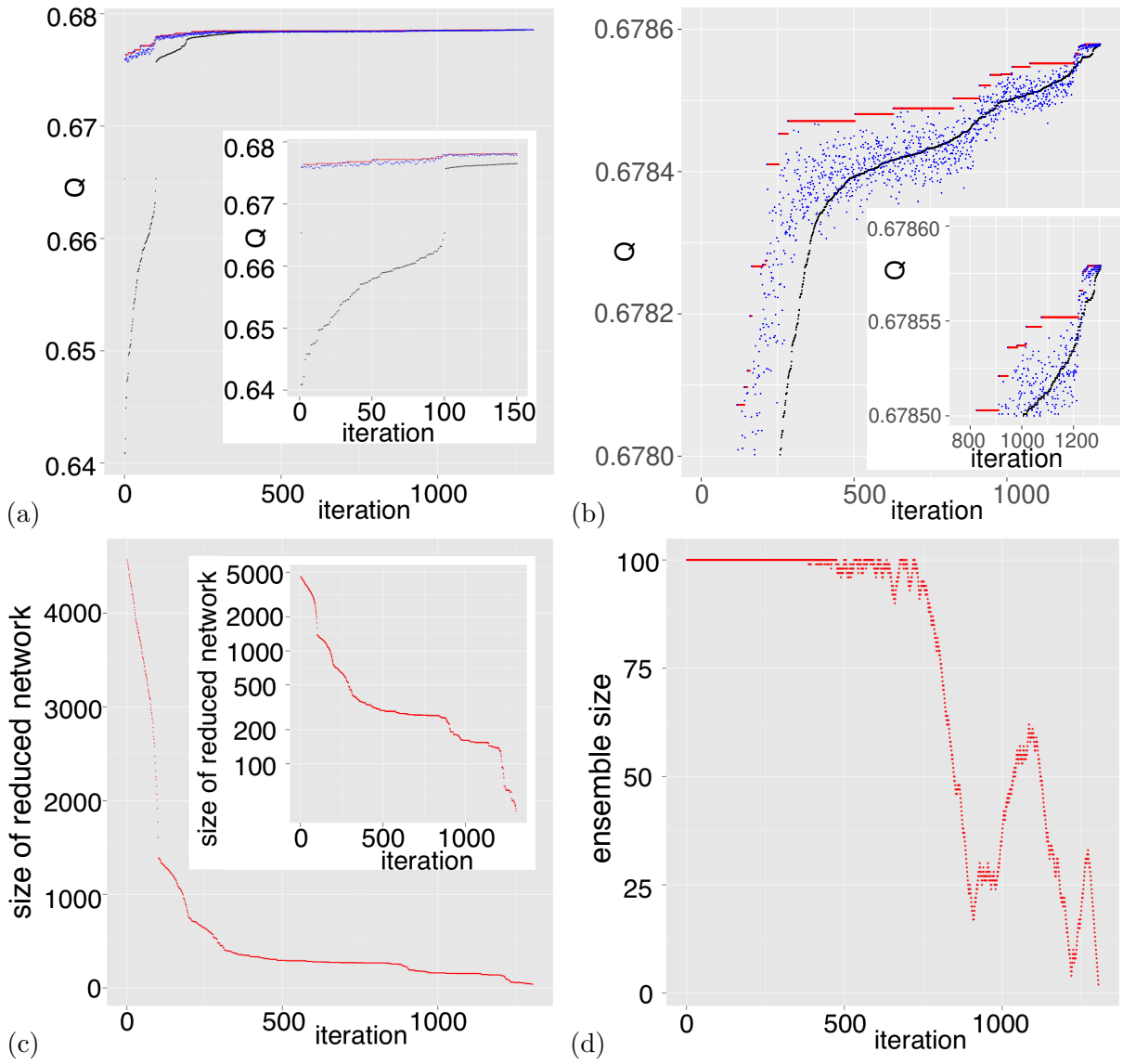


Figure 9: **Evolution of the ensemble of partitions $\mathcal{P}$ for a typical run of RenEEL.** (a) Modularity Q of partitions P_{best}, P'_{best} and P_{worst} at each iteration of the scheme is shown in red, blue and black, respectively. The inset is an enlargement of the results for the first 150 iterations. (b) Same results as in (a), but showing only the upper portion of the plot. The inset shows an enlargement of the upper-right corner of the plot. (c) Evolution of the size of the reduced network. The inset shows the same plot on a logarithmic scale. (d) Evolution of the size of the ensemble $\mathcal{P}$.

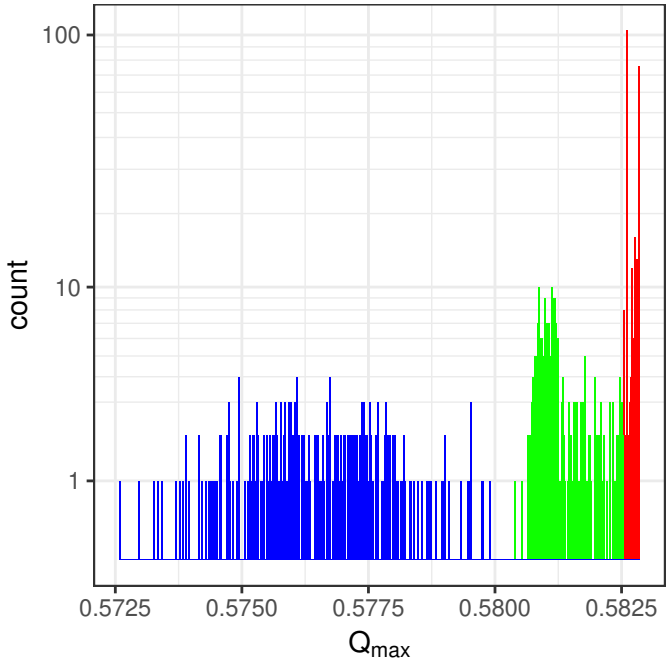


Figure 10: **Distribution of $Q_{\max}$ obtained by various methods.** Frequency plot of $Q_{\max}$ for the Email network obtained by multiple realizations of three different methods. Blue corresponds to a naive ensemble analysis scheme, green corresponds to CGGCi scheme, and red corresponds to RenEEL scheme. The y-axis has a logarithmic scale. In this particular example, there is no overlap between the distributions from the different methods.

Table 2: **Comparison of results using RenEEL to the previous best results for benchmark networks.** Maximum modularity $Q_{\max}$ obtained by the RenEEL scheme compared to the previous best reported values.

Network	**Nodes**	**Links**	**RenEEL result**	**Previous best**
Adjnoun	112	425	0.313367	0.313367 [104]
Jazz	198	2742	0.445144	0.445144 [104]
Metabolic	453	2025	0.453248	0.453248 [104]
Email	1133	5451	0.582829	0.582829 [104]
Polblog	1490	16715	0.427105	0.427105 [104]
Netscience	1589	2742	0.959900	0.959900 [66]
Power	4941	6594	0.940938	0.940851 [104]
PGPgc	10680	24316	0.886853	0.886564 [103]
Astro-ph	16706	121251	0.745614	0.744621 [104]
Memplus	17758	54196	0.700591	0.700473 [103]
As-22july06	22963	48436	0.678579	0.678360 [66]
Cond-mat-2005	40421	175693	0.748187	0.746445 [66]
Smallworld	100000	499998	0.793175	0.793099 [66]
caidaRouterLevel	192244	609066	0.872086	0.872042 [103]

larger networks. For the smaller networks, the value of $Q_{\max}$ reported in Table 2 was consistently obtained. For the larger networks, a range of results were obtained and the largest one is listed. As the table shows, the partitions found by RenEEL have a value of $Q_{\max}$ that is higher than or equivalent to the best previously reported value for every benchmark network. The difference between $Q_{\max}$ found by RenEEL and the previous best values increases with network size. This is due to the fact that for small networks it is generally easier to find the optimal partition, but the task becomes more challenging for larger networks where the superiority of a method can make a significant difference.

3.2 Generalized Modularity Density

3.2.1 Benchmark test

We now analytically study the extent to which the RL exists in the benchmark network of Figure 8 when Q_g is used as the metric. We also compare the results to that obtained when using other metrics. Whether the use of the metric Q_g will split the cliques or not is determined by the sign of $\Delta Q_g = Q_g^{merge} - Q_g^{split}$, where Q_g^{merged} and Q_g^{split} are the values of Q_g if the cliques are merged or split, respectively. Let us define the variables

$$p = \frac{n_1}{n_2} \tag{12}$$

and

$$t = \frac{m_a}{n_1 n_2 w_{\max}} \, . \tag{13}$$

$p \in (0, 1]$ is the ratio size of the cliques. $t \in [0, \infty)$ measures the external influence on them. Then,

$$\begin{aligned} \Delta Q_g \sim \Big(\frac{r+2d}{r+2}\Big)^{\chi} \Big(2d + r - \frac{(r+2d)^2}{r+2d+2t}\Big) \\ - \Big(r - \frac{r^2 - 2 + 2d^2 + 2dr}{r+2d+2t}\Big) \end{aligned} \tag{14}$$

where $r = p + 1/p$. If $\Delta Q_g < 0$, splitting is preferred, and if $\Delta Q_g > 0$, merging is preferred. Equation 14 determines whether the use of the metric Q_g, for a given value of χ, will lead to M or S phase as a function of the variables (p, d, t). The value of d at which the phase boundary separating the M and S phases occurs is δ_{Q_g}.

In the limit of large external influence parameter t, which is often the situation encountered in empirical studies where RL problems are considered problematic, the value of δ_{Q_g} for a given value of χ is

$$\lim_{t \to \infty} \delta_{Q_g} = \frac{r}{2} \left[\left(1 + \frac{2}{r}\right)^{\frac{\chi}{\chi+1}} - 1 \right] \, . \tag{15}$$

This limit increases from $\delta_{Q_g} = 0$, when $\chi = 0$, to $\delta_{Q_g} = 1$, when $\chi \to \infty$, for all values of p.

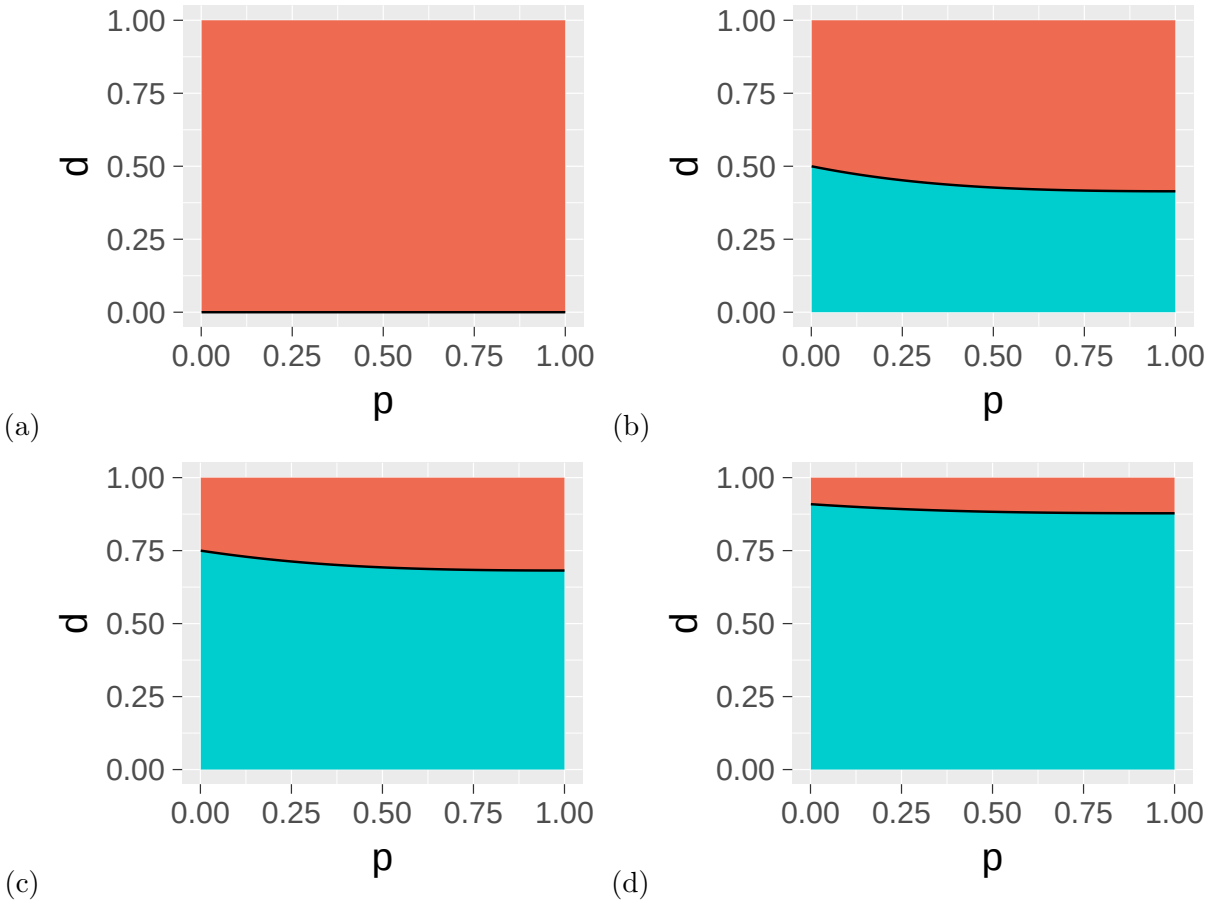


Figure 11: **Phase diagram of clique splitting with generalized modularity density at large external influence as χ is varied.** The values of clique size ratio p and link density d where the M phase occurs are shown in orange and where the S phase occurs are shown in blue. Results are for different values of the control parameter χ: (a) $\chi = 0$, (b) $\chi = 1$, (c) $\chi = 3$, (d) $\chi = 10$. The external influence parameter is $t = 10^6$

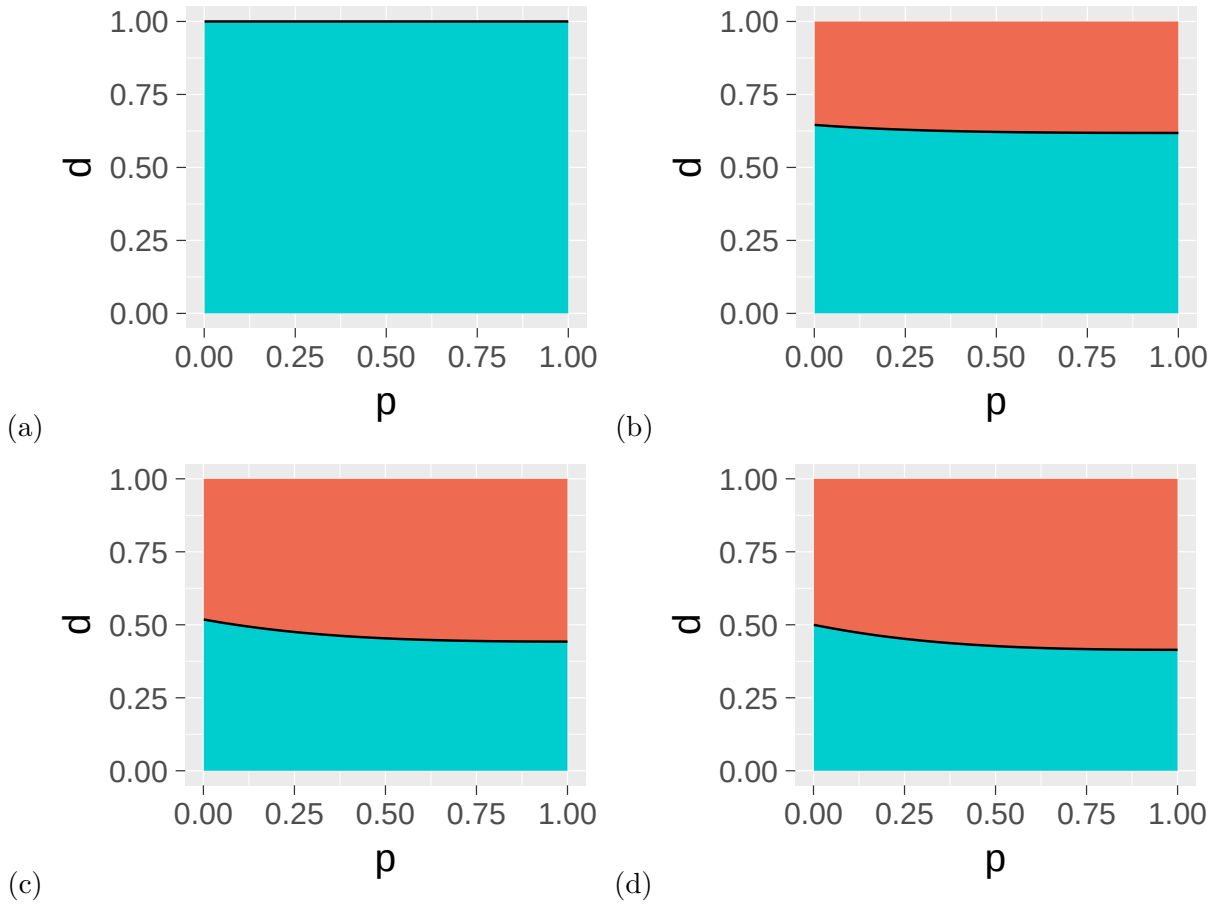


Figure 12: **Phase diagram of clique splitting with generalized modularity density at fixed χ as the external influence is varied.** The values of clique size ratio p and link density d where the M phase occurs are shown in orange and where the S phase occurs are shown in blue. Results are for $\chi = 1$ and different choices of the external influence parameter t: (a) $t = 0$, (b) $t = 1$, (c) $t = 10$, (d) $t = 10^6$.

At intermediate values of χ, the result is only weakly dependent on p, being just slightly larger at small p, as can be seen in Figure 11. The figure shows the phase diagram as a function of p and d at various values χ for large t. For $\chi = 0$, when $Q_g = Q$, the cliques are merged at all values of p and d as shown in Figure 11(a). For $\chi > 0$ at smaller values of d the cliques separate and are, thus, resolved. As χ increases, δ_{Q_g} also increases and approaches 1 in the limit of large χ, Figs. 11(b)-(d), meaning that at large χ the cliques are always resolved.

The effect of varying t at fixed χ on the (p, d) phase diagram are shown in Figure 12. As shown in Figure 12(a), at $t = 0$ when there is no influence by the external component on the two cliques, the S phase occupies the entire space and $\delta_{Q_g} = 1$ for all p. In this case, the cliques are always separated unless they are fully connected to each other. For $t > 0$, when there is some influence from an external component, the cliques are merged and, thus, not resolved for large values of d. As t increases, shown in Figs. 12(b)-(d), the M phase occupies an increasing area and δ_{Q_g} decreases until reaching the limiting value given by Equation 15.

These results show that, as the control exponent χ is varied, a wide range of δ_{Q_g} results. The range increases with t and varies from 0 to 1, the complete possible range, in the limit of large t. This freedom gives leeway in applications to choose χ so that δ_{Q_g} matches the expected critical resolution link density δ_{exp}.

In general, as χ increases, the number of communities found also increases, but gives stable results for a range of χ. (See the example discussed in Section 3.2.3.) Increasing χ thus tends to result in smaller communities being detected. The appropriate, or best, choice of χ depends on the problem. If there is some "ground truth" knowledge about the community structure in the network, or in similar networks, that knowledge can be used to select a χ that results in communities that match the ground truth. If there is no ground truth knowledge, then a default choice of $\chi = 1$ may be appropriate. That choice results in a critical resolution density of $\delta_{Q_g} = 1/2$ in the limit of large t and r (Equation 15). Thus, an advantage of Q_g is that even for the extreme values of t and r, the metric has a positive lower bound of δ_{Q_g} that can be controlled by χ.

In contrast to Q [32], Q_{ds} [75], Q_x [79] and Q_{AFG} [72] (see Section 3.2.4), Q_g has a finite non-zero

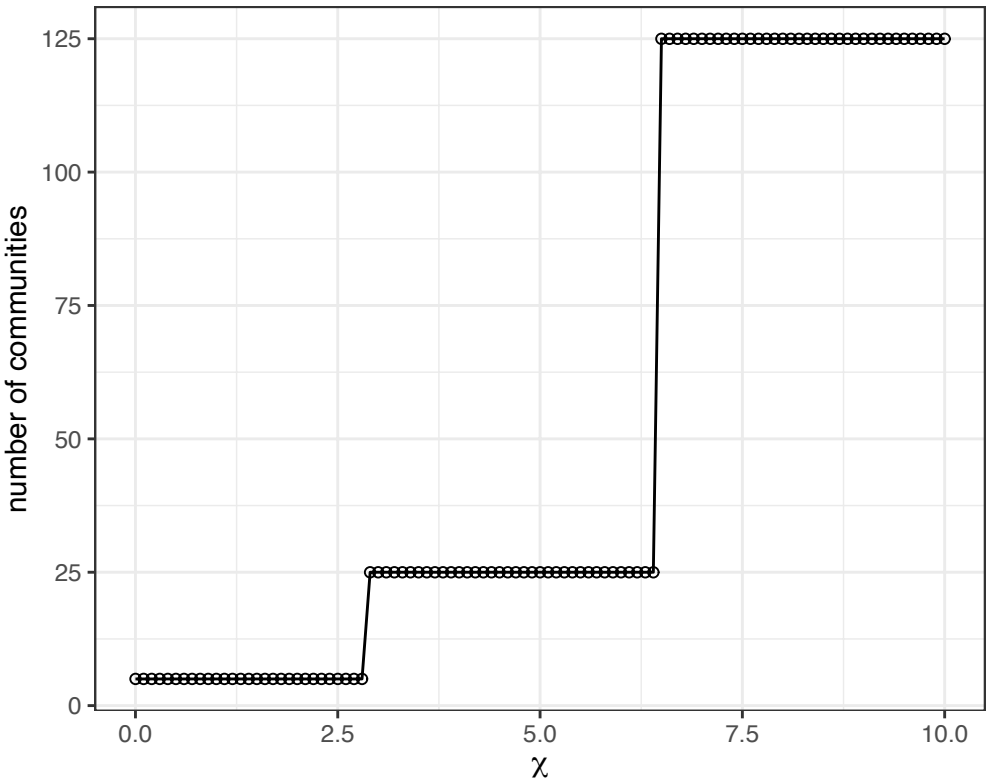


Figure 13: **Number of communities found using Q_g with different χ.** Communities in each level (from the largest to the smallest) in the hierarchy are revealed as χ is varied.

lower limit of δ, which implies that for d smaller than this value, the two cliques of the benchmark network are guaranteed to be split for all possible values of (r, t). Thus, Q_g can successfully avoid resolution limit problem in these extreme cases (See last paragraph in Section 2.2.2). While the metric Q_w (see Section 3.2.4) also shows this lower limit (Table 3), the advantage of Q_g is that the lower limit of δ_{Q_g} can be adjusted by tuning the parameter χ for any desired resolution density. Table 3 summarizes the kind of resolution problems with Q, Q_{ds}, Q_x, Q_w and Q_{AFG} that would be encountered when tested on the benchmark network (See Section 3.2.6 for details).

In principle, a reasonable δ_{exp} is always in $[0, 1]$ but a given metric can still have a δ that is out of this range. Since δ_{exp} is strictly positive (no matter how small), if it is possible to construct a network for which $\delta \to 0$ then that metric presents a resolution limit problem. Even worse, if $\delta < 0$, it would result in merging of disconnected communities. On the other hand $\delta \to 1$ does not pose a resolution problem as long as $\delta \leq 1$ and $\delta \geq \delta_{\text{exp}}$ is satisfied. However, higher δ_{exp} imposes a stricter criterion for merging. But if $\delta > 1$, it will have the unwanted consequence of cliques being subdivided. Thus, a metric is problematic if it can not avoid $\delta \to 0$, $\delta < 0$ or $\delta > 1$.

Table 3: **Resolution limit problems of different metrics.** Q, Q_{ds}, Q_x, Q_w and Q_{AFG} have different resolution limits problems. ρ, which appears in Q_x, is the global link density. s is used in the metric Q_{AFG} as a weight to every node (equivalent to adding a self-loop to every node) and thereby modifying the strength of a community. Q_{AFG} reduces to modularity at $s = 0$, and by controlling s substructures ($s > 0$) or superstructures ($s < 0$) can be explored.

Metric	**Resolution limit problem**
Q	$\delta \to 0$ when $t \to \infty$
Q_{ds}	$\delta < 0$ when p is small
Q_x	$\delta < 0$ when p and ρ are small
Q_w	$\delta_{min} = 0.236$ when $t \to \infty$ and $p = 1$
Q_{AFG}	$\delta < 0$ when $s < 0$ and p is small $\delta > 1$ when $s > 0$ and p is small

3.2.2 American college football network

We use the Q_g metric to detect communities in the network of American college football games between Division IA colleges during the regular season of Fall 2000 [32, 105]. A link between two

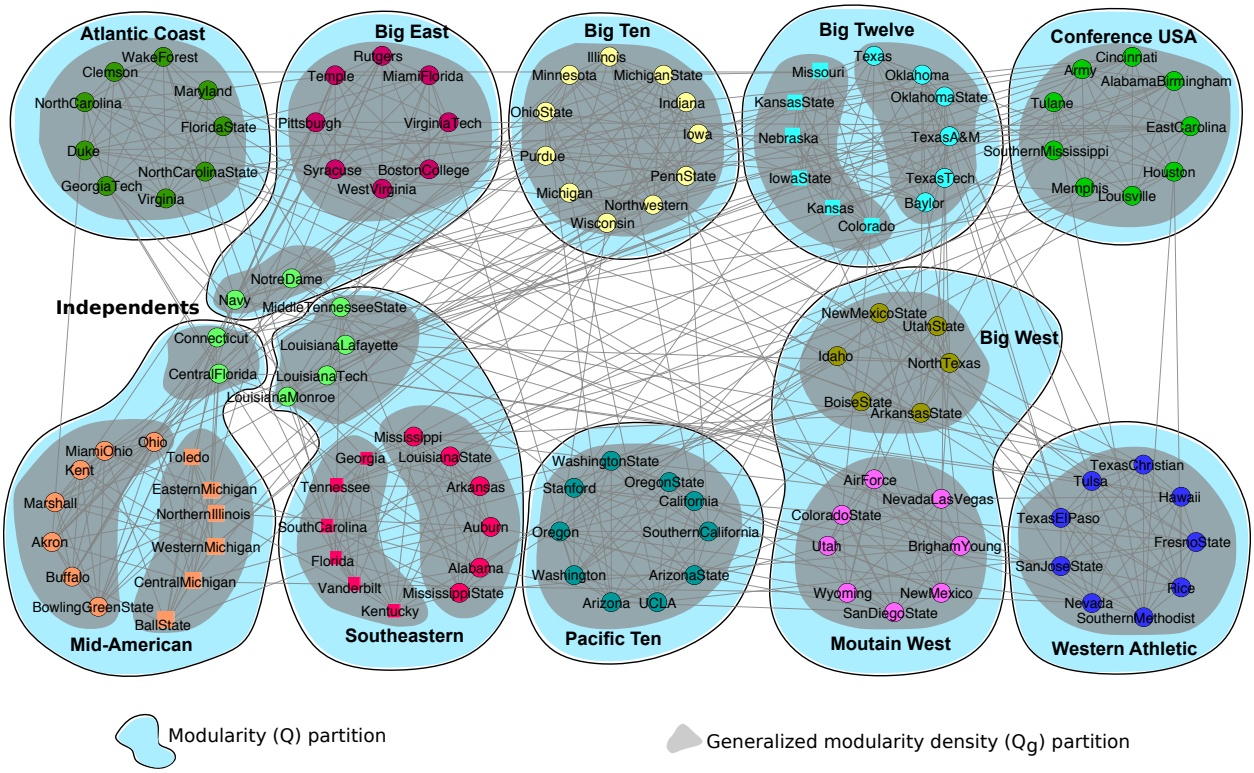


Figure 14: **Communities found in American college football network.** Blue blobs show the communities detected by modularity and gray blobs show the communities found by generalized modularity density.

colleges is present if they played a game against each other. Colleges play games within the same conference more frequently, thus, a community detection algorithm should be able to recover these conferences from the network data. First, we show the result of using modularity (Q) that are indicated by light blue blobs in Fig 14. It matches the conference memberships (distinguished by node color) well except Independents, which are absorbed by three communities and that it groups Big West and Mountain West in the same community. Using $Q_g(\chi = 3)$ in this network we find communities that are shown by gray blobs. There are some key differences between the Q and Q_g partitions. First, the Q_g partition does not merge the Independents with other conferences. Instead, it divides them into three disjoint communities. Second, it successfully identifies the Big West and Mountain West as two different groups. But more interestingly, unlike modularity, it divides each of the Mid-American, Southeastern, and Big Twelve conferences into two communities. This apparent deviation from ground truth actually turns out to be a major advantage of using Q_g. Each of these three conferences have subdivisions within them that are in perfect agreement (considering their membership as of year 2000) with the partition found by Q_g. Mid-American conference has East Division and West Division, Southeastern also has Eastern Division and Western Division, whereas Big-Twelve conference has Northern Division and Southern Division. These subdivisions are indicated by different node shapes (circles and squares) in Figure 14.

3.2.3 Artificial network with hierarchical community structure

To demonstrate the ability of Q_g for detecting the community structure at different resolution densities, we construct a hierarchical network. Similar constructions have been used as a model for hierarchical network structure [72]. We consider a structure shown in Figure 15 that includes four levels of hierarchy, although it can be extended to include any number of levels. The elementary level (level 1) is a clique formed by fully connecting five nodes with links weighted α_1. To construct a level 2 network, we use the clique network from level 1 as a *generalized node* to form a clique of size 5 with links weighted α_2. Link between two generalized nodes is achieved by connecting all the internal nodes from one generalized node to those in another generalized node. Similarly, level

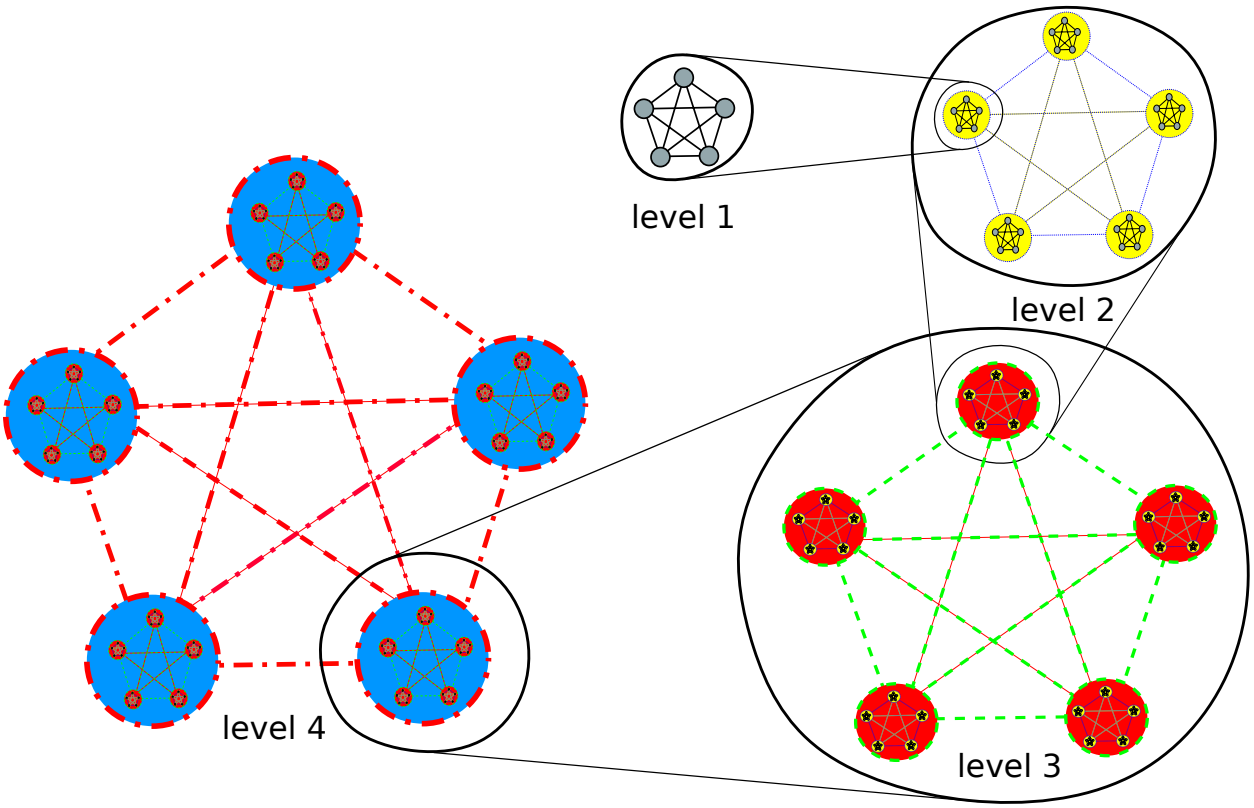


Figure 15: **Example hierarchical network.** Level 1: A clique of five nodes. Level 2: A clique of five level 1 cliques. Level 3: A clique of five level 2 cliques. Level 4: A clique of five level 3 cliques.

k network is constructed by using level $k-1$ network as a generalized node to form a clique of size 5 with links weighted α_k. Here we keep $\alpha_1 > \alpha_2 > ... > \alpha_k$ so that the hierarchy of structure is preserved.

We use the metric Q_g on a level 4 network with $\alpha_k = 5-k$ and show that it successfully detects the planted hierarchical communities at every level. The level 4 network consists of 125 level 1 cliques, and 625 nodes in total. The results obtained by maximizing Q_g are shown in Figure 13. We observe that the 5 level 3 cliques are detected when $\chi < 2.8$, the 25 level 2 cliques are detected when $2.9 < \chi < 6.4$, the 125 level 1 cliques are detected when $\chi > 6.5$. There are 3 stages, corresponding to 3 levels of construction. There should not be a single "best" choice of χ by the

nature of the problem. The choice of χ or desired resolution density should be based on specific requirement and the background information of the particular problem.

3.2.4 Other metrics

Besides the metric Q and Q_g, we test the performance of the following metrics. Each variable has the same meaning as Equation 7 unless otherwise noted.

Weighted Modularity [77]:

$$Q_w = \frac{1}{2m} \sum_c \left(2m_c - \frac{K_c^2}{2m} \right) (\rho_c + 1) \tag{16}$$

Excess Modularity Density [79]:

$$Q_x = \frac{1}{2m} \sum_c \left[2m_c(\rho_c - \rho) - \frac{K_c^2(\rho_c - \rho)^2}{2m} \right] \tag{17}$$

Here $\rho = 2m/[n(n-1)]$ is the global link density.

Modularity Density introduced in [75] has a term that corresponds to Split Penalty. But as discussed in [79], this term may be problematic. Therefore, here we analyze a modified version of modularity density without the Split Penalty term:

$$Q_{ds} = \frac{1}{2m} \sum_c \left(2m_c\rho_c - \frac{K_c^2\rho_c^2}{2m} \right) \tag{18}$$

AFG method of modularity Q_{AFG} in [72] can have different resolution densities by assigning self-loop weighted s to each node and tuning the value of s. It still finds the partition by maximizing modularity after assigning the self-loops.

$$Q_{AFG} = \frac{1}{2m + 2Ns} \sum_c \left[(2m_c + 2n_c s) - \frac{(K_c + 2n_c s)^2}{2m + 2Ns} \right] \tag{19}$$

where N is total number of nodes, n_c is the number of nodes of community c.

3.2.5 Derivation of equation of phase for modularity

We assess the performance of modularity Q using the benchmark test described before. In the form shown in Equation 4, modularity is the sum of the quantity within parenthesis over each community. Thus, two partitions of splitting or merging the two cliques yield the following values of modularity Q.

$$Q_{split} = Q_1 + Q_2 + Q_{ex} \tag{20}$$

$$Q_{merge} = Q_{(1+2)} + Q_{ex} \tag{21}$$

where Q_1, Q_2 are the two terms corresponding to clique 1 and 2 as separate communities, $Q_{(1+2)}$ is the corresponding term when the two cliques are merged. Q_{ex} is the sum over the remaining communities in the external component, which do not change in the two partitions. The difference between the two modularity values is given by

$$\Delta Q = Q_{merge} - Q_{split} = Q_{(1+2)} - Q_1 - Q_2. \tag{22}$$

Using Equation 4, we have:

$$\Delta Q = \frac{1}{2m}\left[(2m_{(1+2)} - \frac{K_{(1+2)}^2}{2m}) - (2m_1 - \frac{K_1^2}{2m} + 2m_2 - \frac{K_2^2}{2m})\right] \tag{23}$$

According to the construction of the example network, we can rewrite Equation 23 with $(n_1, n_2, m_{12}, m_a, n_a, m_{1a}, m_2$
To simplify the expression and capture the principle features, we take $n_1, n_2 >> 1$. Recall the construction of separation of the two cliques from the external component, we also have $n_1^2 >> m_{1a}, n_2^2 >> m_{2a}$. Plugging all in ΔQ, we obtain:

$$\Delta Q = \frac{1}{2m}\left(2m_{12} - \frac{2(n_1^2 n_2^2 + (n_1^2 + n_2^2)m_{12} + m_{12}^2)}{n_1^2 + n_2^2 + 2m_a + 2m_{12}}\right) \tag{24}$$

Equation 24 can be rewritten more concisely by omitting the normalization factors and using

variables (d, r, t) defined in Section 3.2.1

$$\Delta Q \sim 2d - \frac{2(1 + rd + d^2)}{r + 2d + 2t} \tag{25}$$

The space is reduced to three principal dimensions (d, r, t), where $0 \leq d \leq 1$, $r \geq 2$ and $t \geq 0$. Equation 25 is the equation of phase that is used to plot the phase diagram of Figure 16. We obtain δ_Q, which determines the phase boundary as the value of d for which $\Delta Q = 0$,

$$\delta_Q = \sqrt{t^2 + 1} - t \tag{26}$$

3.2.6 Benchmark test for other metrics

By carrying out similar mathematical analyses as in the last section, we can obtain an equation of phase for each metric and we can identify possible RL problems of each metric. Some advantages of using this benchmark test include that it covers a wider range of cases, it can be used by working on the formula without any guess or speculation of specific network, and it provides a clear view of metric performance including all RL problems previously reported. The difference between values of a metric between merge and split cases, for other metrics can be written as follows.

For weighted modularity Q_w

$$\begin{aligned} \Delta Q_w = \frac{2r + 2d + 2}{r + 2}(2d + r - \frac{(r + 2d)^2}{r + 2d + 2t}) \\ -2(r - \frac{r^2 - 2 + 2d^2 + 2dr}{r + 2d + 2t}). \end{aligned} \tag{27}$$

For excess modularity density Q_x

$$\begin{aligned} \Delta Q_x = (2d + r)(\frac{2d + r}{r + 2} - \rho) - \frac{(r + 2d)^2}{r + 2d + 2t}(\frac{r + 2d}{r + 2} - \rho)^2 \\ - \left(r(1 - \rho) - \frac{r^2 - 2 + 2d^2 + 2dr}{r + 2d + 2t}(1 - \rho)^2 \right). \end{aligned} \tag{28}$$

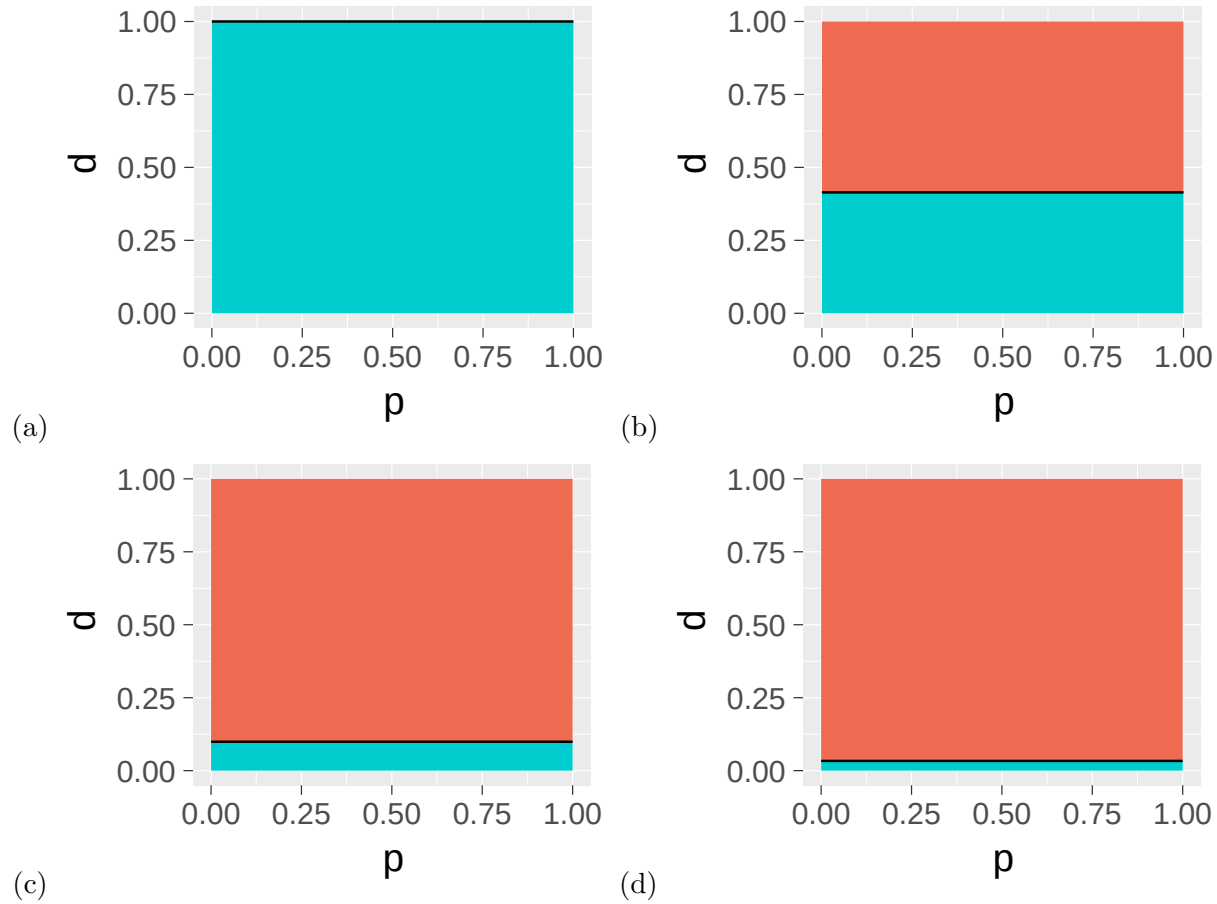


Figure 16: **Phase diagram of clique splitting with modularity Q as the external influence is varied.** The values of clique size ratio p and link density d where the M phase occurs are shown in orange and where the S phase occurs are shown in blue. Results are for different choices of the external influence parameter t: (a) t=0 (b) t=1 (c) t=5 (d) t=15.

For modified (without the split penalty term) modularity density Q_{ds}

$$\Delta Q_{ds} = \frac{(2d+r)^2}{r+2} - \frac{(2d+r)^4}{(r+2d+2t)(r+2)^2} - (r - \frac{r^2-2+2d^2+2dr}{r+2d+2t}) \tag{29}$$

For Q_x (Equation 17), in addition to (d, r, t), the phase space consists of an extra principal variable ρ, which is the global link density and its maximum value ρ_{max} is obtained when n_a is smallest as other variables (n_1, n_2, m_{12}, m_a) are fixed.

The phase diagrams of Q, Q_{ds}, Q_w, $Q_x(\rho = \rho_{max})$, $Q_{AFG}(n_a = \sqrt{2m_a})$ and $Q_g(\chi = 1)$ are

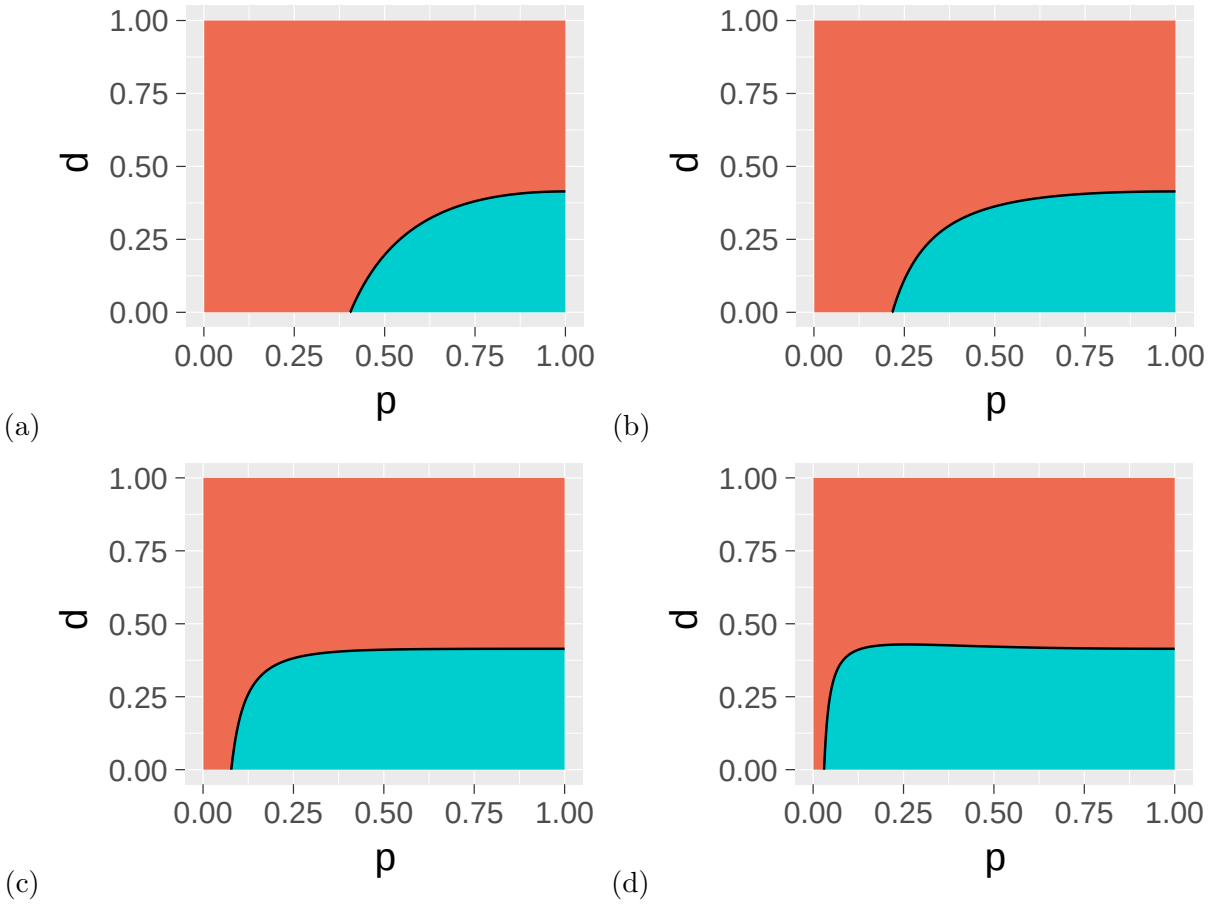


Figure 17: **Phase diagram of clique splitting with modularity density Q_{ds} as the external influence is varied.** The values of clique size ratio p and link density d where the M phase occurs are shown in orange and where the S phase occurs are shown in blue. Results are for different choices of the external influence parameter t: (a) t=0 (b) t=1 (c) t=5 (d) t=15.

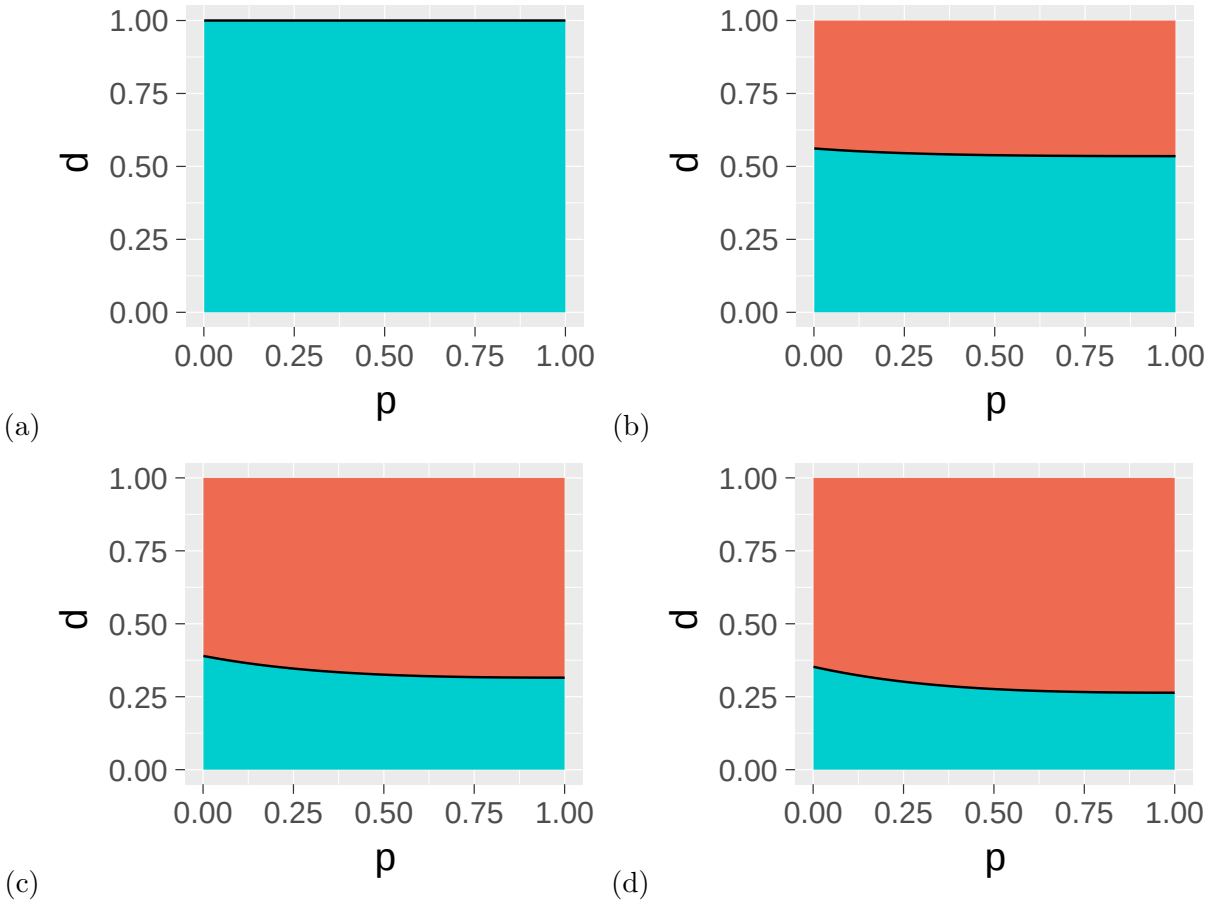


Figure 18: **Phase diagram of clique splitting with weighted modularity Q_w as the external influence is varied.** The values of clique size ratio p and link density d where the M phase occurs are shown in orange and where the S phase occurs are shown in blue. Results are for different choices of the external influence parameter t: (a) t=0 (b) t=1 (c) t=5 (d) t=15.

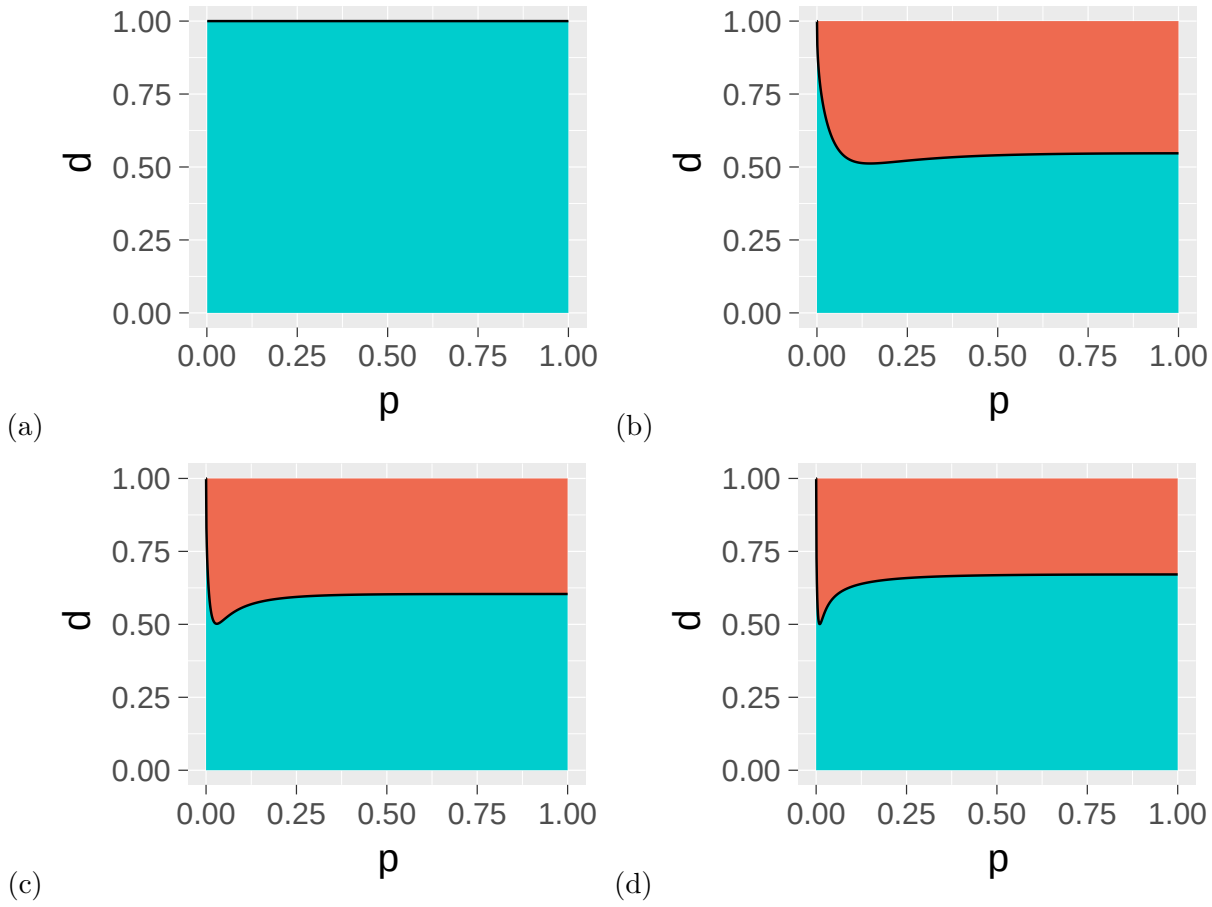


Figure 19: **Phase diagram of clique splitting with excess modularity density Q_x as the external influence is varied.** The values of clique size ratio p and link density d where the M phase occurs are shown in orange and where the S phase occurs are shown in blue. Results are for $\rho = \rho_{max}$ and different choices of the external influence parameter t: (a) t=0 (b) t=1 (c) t=5 (d) t=15.

shown respectively in Figure 16, Figure 17, Figure 18, Figure 19, Figure 20 and Figure 12. As shown in the figures, the behavior varies a lot across different metrics and the particular choice of other variables. In the following, we will observe some general characteristics of all phase diagrams. Then, we will examine each one in more detail and demonstrate that Q_g performs better than other metrics.

First, there are two phases (M (red) and S (blue) phase) in the phase diagram as expected and M phase is above S phase, implying that nearly all metrics tend to merge the two cliques when d is close to 1 and to split when d is close to 0. This meets the common expectation in extreme cases. But different metrics disagree when d is in intermediate range. Other variables such as (t, ρ) also dictate the performance in this range. Figure 16 shows the RL problem of modularity with a much clear view. We know that $\delta = \sqrt{t^2+1} - t$, as $t \to \infty$, we have $\delta \to 0$. This trend is also shown in Figure 16. Therefore, given any value $d > 0$, we can construct a network with large enough t so that $d > \delta$, which means the two cliques, as long as they are connected, will be merged into one community if the external component has enough links. This is the RL problem of modularity.

However, if $d = 0$, there is no RL because for any $t \geq 0$, $\delta > 0$ is always true and modularity maximization would not merge two disconnected cliques. More generally it can be shown that, if two subgroups of the network are disconnected, they are guaranteed to be split.
As shown in Figure 17, Q_{ds} depends strongly on p. A different type of RL problem can be seen in the figures. If p is small enough, $\delta = 0$ can always be true whatever d is. It means that if the sizes of two cliques are different enough, they will be merged even if $d = 0$ [79]. It clearly violates our expectation. This problem gets alleviated as $t \to \infty$. But it always exists for arbitrary t.

For the phase diagram of Q_w shown in Figure 18, the phase boundary moves down as $t \to \infty$. But it has a lower bound which means, when d is small enough, the two cliques of example network will always be split whatever other variables are. Thus, it has no extreme cases of RL as Q and Q_{ds}. Note that M phase is reduced to a straight line $d = 1$ here in Figure 18(a), which means the extreme case of expectation is satisfied.

As for Q_x, because there is one more variable ρ, the analysis is more complicated. As we can

see from Equation 17 and Equation 18, $Q_x(\rho \to 0) \to Q_{ds}$ which means Q_x will behave the same as Q_{ds} when global link density $\rho = 0$. Because of the arbitrary external component, it can be easily achieved. Also we should be aware of the fact that most real-world networks are sparse, thus $\rho \to 0$ is a common case where Q_x will fail to solve RL as Q_{ds}. In Figure 19, we show the phase diagram when ρ equals its maximum. The phase boundary, starting from $d = 1$, goes down first and then rises up again. So, when $t \to \infty$, the two cliques will always be split as long as $d < 1$. But this requires both ρ, t are very large which is uncommon for most real-world networks.

We use the AFG method [72] on the benchmark network, which attempts to solve the RL problem by assigning a self loop of weight s to each node. This method allows one to explore communities at different resolution densities by controlling s. Using Q_g, this is achieved by controlling χ so the two methods are similar in spirit. However, irrespective of the choice of s, the metric Q_{AFG} will behave like modularity Q and fail to resolve clusters if n_a in the benchmark network is sufficiently large. To avoid that, we show the phase diagram (Figure 20) for $n_a = \sqrt{2m_a}$, which is the smallest possible n_a for a fixed m_a (in the large m_a limit) and perhaps the best case scenario for Q_{AFG}. Moreover, if a specific resolution density is desired, then s must be selected according to the network size, unlike Q_g, which has the lower bound that is independent of the network size. Even if s is chosen according to the network size, the phase diagram in Figure 20 (a), (c), (d) shows that the metric Q_{AFG} will fail when the two cliques are somewhat different in size (small p). When p is small and $s \neq 0$, it either merges two disconnected cliques (Figure 20 (a)), or splits a larger clique formed by clique 1 and clique 2 (Figure 20 (c) and (d)). When $s = 0$ (Figure 20 (b)), the metric Q_{AFG} is the same as modularity Q and it will have the same problems as outlined before. The phase diagram also shows that, for a non-zero value of s, the resolution density varies a lot as a function of p. This implies that merging or splitting the two cliques is heavily influenced by their relative sizes. Thus, in a network with a wide range of community sizes, this method will be biased either towards merging well separated communities or splitting well connected communities, an observation also made in [81].

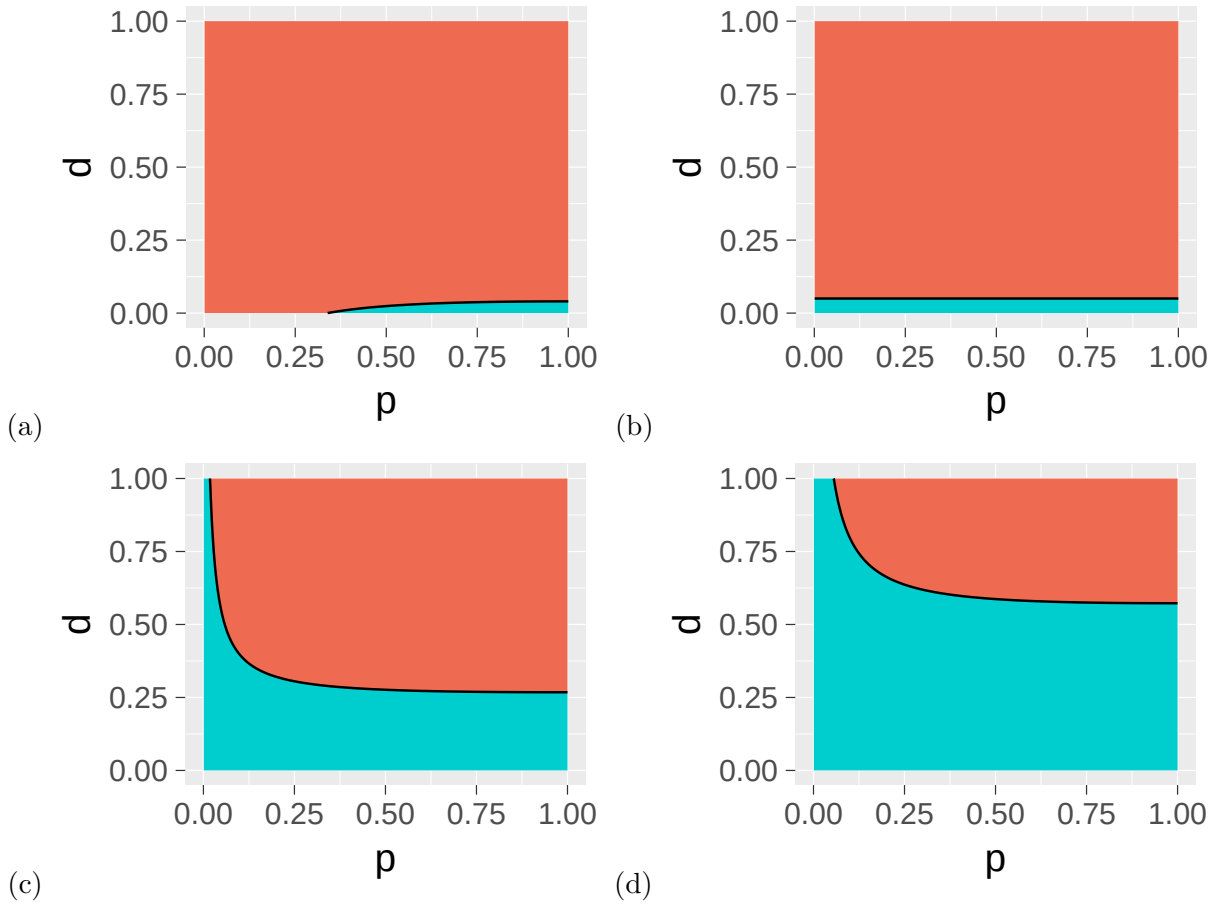


Figure 20: **Phase diagram of clique splitting with Q_{AFG} at fixed external influence t as the parameter s is varied.** The values of clique size ratio p and link density d where the M phase occurs are shown in orange and where the S phase occurs are shown in blue. Results are for $n_a = \sqrt{2m_a}$, $t = 10$ and different choices of s: (a) $s = -\frac{m}{2N}$ (b) $s = 0$ (c) $s = \frac{m}{2N}$ (d) $s = \frac{m}{N}$.

4 Application: Gene Networks

Because of the control parameter in generalized modularity density Q_g, we can detect communities at any desired resolution. Thus, the metric is particularly desirable for studying hierarchical community structure in biological networks, and identifying functional gene communities that are hidden in other modularity approaches because of their small size. Besides, RenEEL can help us maximize the metric Q_g efficiently and accurately. We apply the Q_g and RenEEL to some gene networks [106].

The first step is to generate networks from the gene co-expression data from *S. cerevisiae*, which is a matrix based on some measure of the co-expression. We implemented three popular methods, which are Pearson's correlation method, mutual information method, context likelihood of relatedness method.

Pearson's Correlations distinguish positive and negative gene interactions, providing important biological information. However it misses non-linear relationships among gene expression.

Information theoretic methods that use mutual information (MI) between gene pairs as similarity measure have been used to capture these non-linear variations, which can help uncover strong pairwise relationships between genes that are not detected by linear measures [107]. However, negative relationships cannot be distinguished from positive associations.

Context Likelihood of Relatedness (CLR), an algorithm for unsupervised network inference [108, 109], uses the *distribution* of the MI of two genes combined with the *value* of the MI between these two genes to compute a *relatedness* score [108]. The score of pair (i, j) is defined as $\sqrt{z_i^2 + z_j^2}$, where z_i and z_j are the z-scores of $MI_{i,j}$ from the marginal distributions, and the score is the joint likelihood measure [108]. The CLR algorithm performed better than several other inference methods tested at inferring the regulatory interactions among genes in microarray expression data from the prokaryote, *E. coli* [108].

Then, we used four community detection methods to detect community structure in each network. These methods are MCL [49], and community detection by maximizing Q, Q_x, and Q_g.

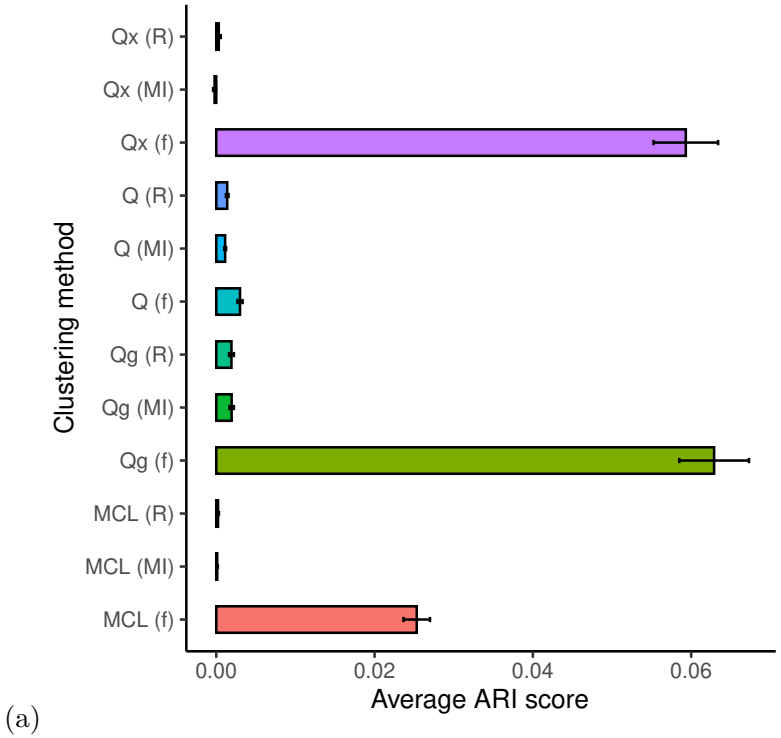
Each network generation method combined with a different community detection method results in a different partition of genes into communities, i.e., four different community detection methods and three different networks, yielding twelve sets of communities.

To determine which combination of network generation method and community detection method is more relevant biologically, we compare our results to the Gene Ontology terms [110]. Specifically, we compare all twelve partitions to the structure of GO terms using the average ARI [111] (Figure 21 (a)). First, we observe that all community detection methods show improved performance when applied to relatedness network constructed by the CLR method. Other network inference methods showed similar and relatively low performance. Second, we find that the results from the relatedness network combined with generalized modularity density (denoted by $Q_g(f)$) shows a better match (higher average ARI) with the GO terms than other methods (although closely followed by $Q_x(f)$). We also show the overall distribution of ARI when using the relatedness network in Fig 21(b) to show that Q_g consistently outperforms Q, Q_x and MCL.

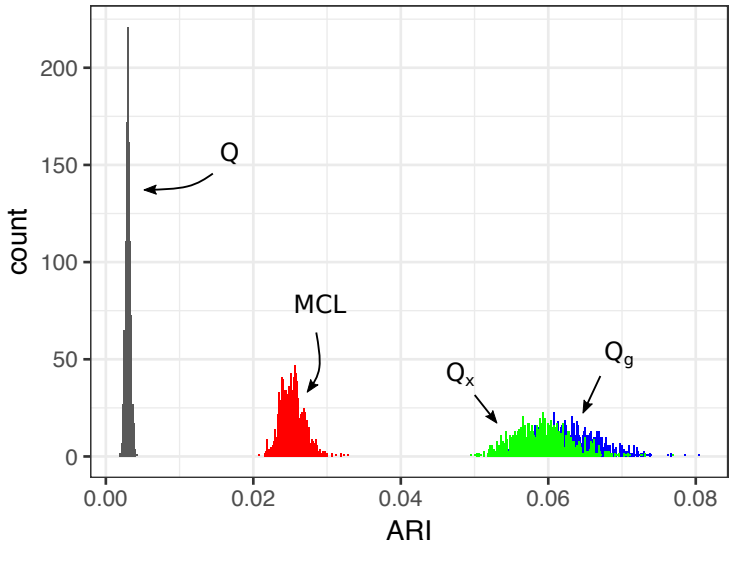
Table 4 shows the top ten matches of the same enrichment analysis for the communities that contain orphan genes. Orphan genes are genes encoding species-specific proteins without detectable homologues in other lineages [112]. They are present across eukaryotics and prokaryotics, and are thought to play an important role in speciation [113, 114, 115, 116, 117]. The functions of the vast majority of orphan genes remain unknown. Of those that have been researched, many are implicated in defense and offense/predation, either as secreted molecules, or as interactors of internal defense responses [114, 117]. Identifying communities with orphan gene members and with highly significant associations with GO-terms is particularly important in that it provides an avenue towards building hypotheses about the functions of these orphan genes.

Table 4: **GO term enrichment of communities found using generalized modularity density containing orphan genes in them.** Ten most significant associations for orphan containing communities found by Q_g and relatedness network with threshold = 3.0.

community	**GO term**	**genes in community**	**genes in GO term**	**genes in common**	**p-value**	**orphans**
1	cytochrome-c oxidase activity	19	20	8	7.28E-15	9
1	heme binding	19	27	5	9.74E-08	9
1	Group I intron splicing	19	12	4	2.04E-07	9
1	mitochondrial electron transport, cytochrome c to oxygen	19	14	3	2.68E-05	9
34	carnitine O-acetyltransferase activity	187	3	3	0.0010	5
1	proton-transporting ATP synthase activity, rotational mechanism	19	17	2	0.0027	9
34	fructose 2,6-bisphosphate metabolic process	187	4	3	0.0032	5
1	homing of group II introns	19	1	1	0.0062	9
1	movement of group I intron	19	1	1	0.0062	9
66	protein targeting to vacuole involved in ubiquitin-dependent protein catabolic process via the multivesicular body sorting pathway	15	22	3	0.0074	6



(a)



(b)

Figure 21: **Comparison between GO term associations and gene communities found by different network inference and community detection methods.** (a) Average Adjusted Rand Index (ARI) score for 1000 realizations comparing GO-terms and communities found by modularity Q, excess modularity density Q_x, Markov Clustering (MCL), and generalized modularity density Q_g on relatedness (f), Mutual Information (MI), and Pearson correlation (R) networks. A higher ARI indicates a better match. The error bars show one standard deviation interval around the mean. (b) The distribution of ARI scores between gene communities and GO terms over 1000 realizations using modularity Q (black), MCL (red), excess modularity density Q_x (green), and generalized modularity density Q_g (blue) for relatedness (f) network.

5 Conclusions

Recent advances in Machine Learning and Artificial Intelligence have enabled progress toward solving a range of challenging computational problems [118]. This thesis has introduced a powerful algorithmic paradigm for graph partitioning that we call Extremal Ensemble Learning (EEL). EEL is a form of Machine Learning. An EEL scheme creates an ensemble of partitions and then uses information within the ensemble to find new partitions used to update the ensemble using extremal criteria. The ensemble learns how to form improved partitions through the updating procedure, as it works toward a conclusion by achieving consensus among its member partitions about what the optimal partition is.

The particular EEL scheme we have introduced, Reduced Network Extremal Ensemble Learning (RenEEL), uses information in the ensemble of partitions to create a reduced network that can be efficiently analyzed to find a new partition with which we can update the ensemble. We have used RenEEL to find the partition that maximizes the modularity of networks. This is a difficult, NP-hard computational problem [53]. We have shown that an algorithm using the RenEEL scheme outperforms all existing modularity maximizing algorithms when analyzing various commonly studied benchmark networks. For those networks, it finds partitions with the largest modularity ever discovered. For the larger benchmark networks, the partitions that we discovered are novel.

Although we have only demonstrated the effectiveness of our algorithm for the well-known problem of finding the network partition that maximizes modularity, the EEL paradigm and the RenEEL scheme can be used to solve other network partitioning problems. For example, the algorithm we used can be straightforwardly adapted to optimize other metrics such as modularity density [78], excess modularity density [79] or the new metric, generalized modularity Q_g, which we introduce in this thesis.

There is potential to improve upon our results using the RenEEL scheme. As previously discussed, any conventional algorithm can be used as the base algorithm of the scheme. There is

also the freedom to vary the size of the ensembles used in the scheme. Which base algorithm and what ensemble sizes are best depends on the network to be analyzed. Using a high-quality base algorithm, though, such as the Iterative Spectral Bisectioning, Tuning, and Agglomeration algorithm [61], is likely to yield more accurate results for many of the networks studied. There is also potential to improve the RenEEL scheme itself. For instance, a naive ensemble analysis of partitions of the reduced network is currently used to find a new partition to update the ensemble. Another method, such as a recursive use of the RenEEL scheme, may yield better results. Also, once the original ensemble of partitions is created, no new information is ever added to the system during the learning processes. It may be beneficial to occasionally use a new partition of the original network instead of the reduced network to update the ensemble. Work is in progress to explore if these ideas lead to improved results.

Finally, the principal reasons why the RenEEL scheme is both efficient and effective should be noted. Its efficiency stems from its use of an ensemble of partitions to form reduced networks. The smaller size of the reduced networks allows them to be partitioned much more quickly than the original network. Also, because the scheme is so effective, highly accurate results can be obtained even if a fast but low-quality base algorithm is used. This allows significantly larger networks to be analyzed than what would otherwise be possible. The remarkable effectiveness of RenEEL, even relative to other Ensemble Learning schemes, is mainly due to its extremal updating of the ensemble of partitions. It is, of course, just one example of a scheme using the EEL paradigm. Its success, though, suggests that EEL is an algorithmic paradigm that will be useful for solving a variety of graph-theoretic problems.

Community detection by maximizing modularity sometimes yields unexpected community structure. Resolution limit, for example, is an unwanted but inevitable consequence of modularity maximization. Other such metrics, namely modularity density measures, which attempt to fix this problem, also differ in the community structure they obtain and may violate our general expectation. While the cutoff of the number of cross-links between two strongly connected groups of nodes that is enough to make the two be considered as a single community remains mostly subjective

and vague, our metric Q_g provides a quantifiable notion and solves the resolution limit problem. In particular, with a free parameter χ, one can control this threshold of merging two cliques. It is quite appealing to have a metric that can be adjusted to meet the specific requirement set by the user because the idea of a community may vary from one application to another and be particular to the network under consideration. At the same time, due to its ability to detect communities at many resolution densities, it is also helpful in uncovering the hierarchical community structure, an inherent characteristic observed in many complex networks. The existing benchmarks, e.g., the ring of cliques, are too restrictive to evaluate and compare the performance of different metrics for solving the specific resolution limit problem. In this thesis, we consider a more general yet simple network structure, which can be used to examine the limits of metrics such as modularity quantitatively. Using this general framework, we demonstrated that our metric Q_g eliminates resolution limit problems at any desired resolution density, shows better performance, and is straightforward to extend for studying weighted and directed networks. Among other essential problems, finding communities at high resolution is instrumental in inferring gene regulatory networks where the goal of functional annotation of genes is to find particular gene functions [91].

We systematically evaluated twelve different combinations of clustering methods by network inference methods using gene expression data. Our investigation empirically shows that the relatedness network obtained by the CLR method combined with community detection using generalized modularity density Q_g metric outperforms other approaches. Previous studies using the CLR method have demonstrated its effectiveness in inferring gene interactions [108]. Concerning the improved accuracy of Q_g, we surmise that it performs better than other metrics because it does not suffer from the resolution limit problem [69, 80] and can detect smaller communities with higher resolution.

Furthermore, we performed GO term enrichment analysis using the communities obtained by generalized modularity density Q_g on relatedness networks. Our focused analysis reveals statistically highly significant associations between communities, many of which contain orphan genes and GO terms. These novel modules can potentially enrich our understanding of the roles of these

orphan genes and their relationship with other genes by providing possible directions for the experiments to explore the function of the orphan genes. Then, we can use the feedback from those experiments to improve our methods, such as tuning the value of the control parameter in Q_g.

Bibliography

[1] S. Strogatz, Exploring complex networks. Nature 410, 268–276 (2001).

[2] M. E. J. Newman, Networks, An Introduction. Oxford University Press (2010).

[3] R. Albert, A.-L. Barabási, Statistical mechanics of complex networks. Reviews of Modern Physics 74 (1): 47–49 (2002).

[4] W. W. Zachary, An Information Flow Model for Conflict and Fission in Small Groups. Journal of Anthropological Research 33 (4): 452–473 (1977).

[5] M. Ezzatabadipour, W. Zhang, K. E. Bassler, and R. K. P. Zia, Fluctuations and correlations in a model of extreme introverts and extroverts in preparation.

[6] T. Harko, F. S. N. Lobo, and M. K. Mak, Exact analytical solutions of the Susceptible-Infected-Recovered (SIR) epidemic model and of the SIR model with equal death and birth rates. Applied Mathematics and Computation 236: 184–194 (2014).

[7] M. Kröger, R. Schlickeiser, Analytical solution of the SIR-model for the temporal evolution of epidemics. Part A: Time-independent reproduction factor. Journal of Physics A 53 (50): 505601 (2020).

[8] M. E. J. Newman, Assortative Mixing in Networks. Physical Review Letters American Physical Society (APS) 89 (20): 208701 (2002).

[9] M. E. J. Newman, Finding community structure in networks using the eigenvectors of matrices. Physical Review E 74, 036104 (2006).

[10] P. Gleiser, L. Danon, Community structure in jazz. Advances in Complex Systems 6, 565–573 (2003).

[11] J. Duch, A. Arenas, Community detection in complex networks using extremal optimization. Physical Review E 72, 027104 (2005).

[12] H. Jeong, B. Tombor, R. Albert, Z. N. Oltvai, and A.-L. Barabási, The large-scale organization of metabolic networks. Nature 407, 651–654 (2000).

[13] R. Overbeek et al., WIT: integrated system for high-throughput genome sequence analysis and metabolic reconstruction. Nucleic Acids Research 28, 123–125 (2000).

[14] R. Guimera, L. Danon, A. Diaz-Guilera, F. Giralt, and A. Arenas, Self-similar community structure in a network of human interactions. Physical Review E 68, 065103 (2003).

[15] L. A. Adamic, N. Glance, The political blogosphere and the 2004 us election: divided they blog. In Proceedings of the 3rd International Workshop on Link discovery 36–43 (2005).

[16] D. J. Watts, S. H. Strogatz, Collective dynamics of 'small-world'networks. Nature 393, 440 (1998).

[17] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas, Models of social networks based on social distance attachment. Physical Review E 70, 056122 (2004).

[18] M. E. J. Newman, The structure of scientific collaboration networks. Proceedings of the National Academy of Sciences 98, 404–409 (2001).

[19] T. A. Davis, Y. Hu, The university of Florida sparse matrix collection. ACM Transactions on Mathematical Software (TOMS) 38, 1 (2011).

[20] Newman's Network data. http://www-personal.umich.edu/~mejn/netdata/.

[21] CAIDA Skitter Router-Level Topology and Degree Distribution. http://www.caida.org/data/router-adjacencies.

[22] A. Bavelas, Communication patterns in task-oriented groups. The Journal of the Acoustical Society of America 22 (6):725–730, (1950).

[23] L. C. Freeman, A set of measures of centrality based upon betweenness. Sociometry 40 (1): 35–41 (1977).

[24] C. F. A. Negre, U. N. Morzan, H. P. Hendrickson, R. Pal, G. P. Lisi, J. P. Loria, I. Rivalta, J. Ho, and V. S. Batista. Eigenvector centrality for characterization of protein allosteric pathways. Proceedings of the National Academy of Sciences 115 (52): E12201–E12208 (2018).

[25] L. Katz, A New Status Index Derived from Sociometric Index. Psychometrika 39–43 (1953).

[26] "US7058628B1–Method for node ranking in a linked database–Google Patents". Google Patents.

[27] M. Piraveenan, M. Prokopenko, and L. Hossain, Percolation Centrality: Quantifying Graph-Theoretic Impact of Nodes during Percolation in Networks. PLOS ONE 8 (1): e53095 (2013).

[28] M. G. Everett, S. P. Borgatti, Analyzing clique overlap. Connections 21 (1): 49-61 (1998).

[29] A. Barabasi, E. Bonabeau, Scale-Free Networks. Scientific American 288 (5): 50–59 (2003).

[30] S. H. Strogatz, D. J. Watts, Collective dynamics of "small-world" networks. Nature 393 (6684): 440–442 (1998).

[31] H. E. Stanley, L. A. N. Amaral, A. Scala, and M. Barthelemy, Classes of small-world networks. Proceedings of the National Academy of Sciences 97 (21): 11149–52 (2000).

[32] M. Girvan, M. E. J. Newman, Community structure in social and biological networks. Proceedings of the National Academy of Sciences 99: 8271-8276 (2002).

[33] M. E. J. Newman, The Structure and Function of Complex Networks. Society for Industrial and Applied Mathematics Review 45 (2): 167-256 (2003).

[34] L. Danon, A. Diaz-Guilera, J. Duch, and A. Arenas, Comparing community structure identification. Journal of Statistical Mechanics P09008 (2005).

[35] S. Fortunato, Community detection in graphs. Physics Reports 486, 3 (2010).

[36] R. Chauhan, J. Ravi, P. Datta, T. Chen, D. Schnappinger, K. E. Bassler, G. Balazsi, and M. L. Gennaro, Reconstruction and topological characterization of the sigma factor regulatory network of Mycobacterium tuberculosis. Nature communications 7, 11062 (2016).

[37] S. Treviño III, Y. Sun, T. F. Cooper, and K. E. Bassler, Robust detection of hierarchical communities from Escherichia coli gene expression data. PLOS Computational Biology 8 (2): e1002391 (2012).

[38] S. K. Bhavnani, G. Bellala, S. Victor, K. E. Bassler, and S. Visweswaran, The role of complementary bipartite visual analytical representations in the analysis of SNPs: a case study in ancestral informative markers. Journal of the American Medical Informatics Association 19 (e1): e5-e12 (2012).

[39] F. Reid, A. F. McDaid, and N. J. Hurley, Partitioning breaks communities. Mining Social Networks and Security Informatics Lecture Notes in Social Networks 79–105 (2013).

[40] S. Kelley, M. Goldberg, M. Magdon-Ismail, K. Mertsalov, and A. Wallace, Hand-book of Optimization in Complex Networks, Springer Chapter 6 (2011).

[41] A. Lancichinetti, S. Fortunato, and J. Kertész, Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics 11 (3), Article 033015 (2009).

[42] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to Algorithms (2nd ed.), MIT Press and McGraw-Hill, p. 563,655,1043 (2001).

[43] L. Rokach, O. Maimon, Clustering methods. Data mining and knowledge discovery handbook. Springer US 321-352 (2005).

[44] A. Y. Ng, M. I. Jordan, and Y. Weiss, On spectral clustering: analysis and an algorithm. In Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic 849–856 (2002).

[45] D. M. Cvetković, M. Doob, and H. Sachs, Spectra of Graphs: Theory and Applications, 3rd Revised and Enlarged Edition New York: Wiley (1998).

[46] D. Babić, D. J. Klein, I. Lukovits, S. Nikolić, and N. Trinajstić, Resistance-Distance Matrix: A Computational Algorithm and Its Applications. International Journal of Quantum Chemistry 90, 166-176 (2002).

[47] L. Massoulié, Community detection thresholds and the weak Ramanujan property. In Proceedings of the forty-sixth annual ACM symposium on Theory of computing (STOC '14) 694–703 (2014).

[48] E. Abbe, C. Sandon, Community Detection in General Stochastic Block models: Fundamental Limits and Efficient Algorithms for Recovery. IEEE 56th Annual Symposium on Foundations of Computer Science, pp. 670-688 (2015).

[49] S. Van Dongen, Graph Clustering by Flow Simulation. PhD thesis, University of Utrecht, (2000).

[50] D. H. Wolpert, The lack of a priori distinctions between learning algorithms. Neural Computation 8 (7): 1341–1390 (1996).

[51] L. Peel, D. B. Larremore, and A. Clauset, The ground truth about metadata and community detection in networks. Science Advances, 3 (5) (2017).

[52] A. D. McCarthy, T. Chen, S. Ebner, An Exact No Free Lunch Theorem for Community Detection. Complex Networks and Their Applications VIII, COMPLEX NETWORKS 2019, Studies in Computational Intelligence, vol 881 Springer (2020).

[53] U. Brandes, *et al.*, On modularity clustering. IEEE Transactions on Knowledge and Data Engineering 20, 172–188 (2008).

[54] B.H. Good, Y.-A. de Montjoye, and A. Clauset, Performance of modularity maximization in practical contexts. Physics Review E 81 046106 (2010).

[55] J. Reichardt, S. Bornholdt, Statistical mechanics of community detection. Physics Review E 74 16110 (2006).

[56] A. Clauset, M. E. J. Newman, and C. Moore, Finding community structure in very large networks. Physical Review E 70, 066111 (2004).

[57] M. E. J. Newman, Fast algorithm for detecting community structure in networks. Physical Review E 69, 066133 (2004).

[58] M. Ovelgönne, A. Geyer-Schulz, Cluster cores and modularity maximization. In Data Mining Workshops (ICDMW), 2010 IEEE International Conference, 1204–1213 (2010).

[59] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment P10008 (2008).

[60] M. E. J. Newman, Modularity and community structure in networks. Proceedings of the National Academy of Sciences 103, 8577–8582 (2006).

[61] S. Treviño III, A. Nyberg, C. I. Del Genio,and K. E. Bassler, Fast and accurate determination of modularity and its effect size. Journal of Statistical Mechanics: Theory and Experiment P02003 (2015).

[62] B. W. Kernighan, S. Lin, An efficient heuristic procedure for partitioning graphs. The Bell System Technical Journal 49, 291–307 (1970).

[63] Y. Sun, B. Danila, K. Josić, and K. E. Bassler, Improved community structure detection using a modified fine-tuning strategy. Europhysics Letters 86, 28004 (2009).

[64] R. Polikar, Ensemble based systems in decision making. IEEE Circuits and Systems Magazine 6, 21–45 (2006).

[65] O. Sagi, L. Rokach, Ensemble learning: A survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 8, e1249 (2018).

[66] M. Ovelgönne, A. Geyer-Schulz, An ensemble learning strategy for graph clustering. Graph Partitioning and Graph Clustering 588, 187 (2012).

[67] 10th DIMACS Implementation Challenge. https://www.cc.gatech.edu/dimacs10/.

[68] J. Guo, P. Singh, and K. E. Bassler, Reduced network extremal ensemble learning (RenEEL) scheme for community detection in complex networks. Scientific Reports 9: 14234 (2019).

[69] S. Fortunato, M. Barthelemy, Resolution limit in community detection. Proceedings of the National Academy of Sciences 104, 36–41 (2007).

[70] V. A. Traag, P. Van Dooren and Y. Nesterov, Narrow scope for resolution-limit-free community detection. Physics Review E 84, 016114 (2011).

[71] P. Ronhovde, Z. Nussinov, Local resolution-limit-free Potts model for community detection. Physics Review E 81, 046114 (2010).

[72] A. Arenas, A. Fernández, and S. Gomez, Analysis of the structure of complex networks at different resolution levels. New Journal of Physics 10 (5): 053039 (2008).

[73] C. Granell, S. Gomez and A. Arenas, Hierarchical multiresolution method to overcome the resolution limit in complex networks. International Journal of Bifurcation and Chaos 22 (07): 1250171 (2012).

[74] R. Aldecoa, I. Marin, Deciphering Network Community Structure by Surprise. PLOS ONE 6 (9): e24195 (2011).

[75] M. Chen, T. Nguyen, and B. K. Szymanski, A New Metric for Quality of Network Community Structure. ASE Human Journal 2 (4): 226-240 (2013).

[76] F. Botta, C. I. Del Genio, Finding network communities using modularity density. Journal of Statistical Mechanics 123402 (2016).

[77] N. F. Haq, M. Moradi, and Z. J. Wang, Community structure detection from networks with weighted modularity. Pattern Recognition Letters 122: 14-22 (2019).

[78] M. Chen, K. Kuzmin, and B. K. Szymanski, Community detection via maximization of modularity and its variants. IEEE Transactions on Computational Social Systems 1, 46–65 (2014).

[79] T. Chen, P. Singh, and K. E. Bassler, Network community detection using modularity density measures. Journal of Statistical Mechanics: Theory and Experiment, 053406 (2018).

[80] J. Guo, P. Singh, and K. E. Bassler, Resolution limit revisited: community detection using generalized modularity density. In preparation (2020).

[81] A. Lancichinetti, S. Fortunato, Limits of modularity maximization in community detection. Physics Review E 84 (6): 066122 (2011).

[82] J. Park, I. B. Wood, E. Jing, A. Nematzadeh, S. Ghosh, M. D. Conover, and Y. Y. Ahn, Global labor flow network reveals the hierarchical organization and dynamics of geo-industrial clusters. Nature Communications 10 (1): 3449 (2019).

[83] T. Z. Berardini, S. Mundodi, L. Reiser et al., Functional annotation of the Arabidopsis genome using controlled vocabularies. Plant Physiology 135 (2): 745–755 (2004).

[84] R. Overbeek, M. Fonstein, M. D'Souza, G. D. Pusch and N. Maltsev, The use of gene clusters to infer functional coupling. Proceedings of the National Academy of Sciences 96 (6): 2896-2901 (1999).

[85] U. Singh, M. Hur, K. Dorman, and E. S. Wurtele, MetaOmGraph: a workbench for interactive exploratory data analysis of large expression datasets. Nucleic Acids Research 48 (4): e23 (2020).

[86] A. K. Jain, M. N. Murty, P. J. Flynn, Data clustering: a review. ACM Computing Surveys 31: 264-323 (1999).

[87] M. B. Eisen, P. T. Spellman, P. O. Brown, D. Botstein, Cluster analysis and display of genome-wide expression patterns. Proceedings of the National Academy of Sciences 95: 14863-14868 (1998).

[88] P. D'Haeseleer, How does gene expression clustering work? Nature Biotechnology 23: 1499-1501 (2005).

[89] D. Chaussabel, N. Baldwin, Democratizing systems immunology with modular transcriptional repertoire analyses. Nature Reviews Immunology 14: 271-280 (2014).

[90] S. Van Dam, U. Vosa U, A. Van der Graaf, L. Franke, J. P. de Magalhaes, Gene co-expression analysis for functional classification and gene–disease predictions. Briefings in Bioinformatics 19 (4): 575-92 (2018).

[91] W. I. Mentzen, E. S. Wurtele, Regulon organization of Arabidopsis. BMC Plant Biology 8 (1): 99 (2008).

[92] L. Li, W. Zheng, Y. Zhu et al., QQS orphan gene regulates carbon and nitrogen partitioning across species via NF-YC interactions. Proceedings of the National Academy of Sciences 112 (47): 14734-14739 (2015).

[93] M. Qi et al., QQS orphan gene and its interactor NF-YC4 reduce susceptibility to pathogens and pests. Plant Biotechnology Journal 17 (1): 252–263 (2019).

[94] J. Li, U. Singh, Z. Arendsee, and E. S. Wurtele, Landscape of the Dark Transcriptome Revealed through Re-mining Massive RNA-Seq Data bioRxiv 671263 (2020).

[95] S. S. Dwight, M. A. Harris, K. Dolinski et al., Saccharomyces Genome Database (SGD) provides secondary gene annotation using the Gene Ontology (GO). Nucleic Acids Research 30 (1): 69-72 (2002).

[96] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Optimization by simulated annealing. Science 220, 671–680 (1983).

[97] Alex Arenas datasets. http://deim.urv.cat/ alexandre.arenas/data/welcome.htm.

[98] Hamm/memplus — SuiteSparse Matrix Collection. https://sparse.tamu.edu/Hamm/memplus.

[99] 10th DIMACS Implementation Challenge. https://www.cc.gatech.edu/dimacs10/archive/clustering.shtml.

[100] M. E. J. Newman, Analysis of weighted networks. Physics Review E 70 (5): 056131 (2004).

[101] E. A. Leicht, M. E. J. Newman, Community Structure in Directed Networks. Physics Review Letters 100 (11): 118703 (2008).

[102] L. Rokach, O. Maimon, Clustering Methods. Springer, 321–352 (2005).

[103] Index of /dimacs10/results. https://www.cc.gatech.edu/dimacs10/results/.

[104] D. Aloise, *et al.*, Modularity maximization in networks by variable neighborhood search. In Graph Partitioning and Graph Clustering (2012).

[105] T. S. Evans, Clique graphs and overlapping communities. Journal of Statistical Mechanics P12037 (2010).

[106] P. Singh, J. Guo, J. Li, U. Singh, E. S. Wurtele, and K. E. Bassler, Assessing network inference and clustering methods for geneco-expression networks.

[107] C. O. Daub, R. Steuer, J. Selbig, and S. Kloska, Estimating mutual information using B-spline functions–an improved similarity measure for analysing gene expression data. BMC Bioinformatics 5 (1): 118 (2004).

[108] J. J. Faith et al., Large-scale mapping and validation of Escherichia coli transcriptional regulation from a compendium of expression profiles. PLOS Biology 5: e8 (2007).

[109] S. Treviño III, Y. Sun, T. F. Cooper and K. E. Bassler, Robust detection of hierarchical communities from Escherichia coli gene expression data. PLOS Computational Biology 8 (2): e1002391 (2012).

[110] Gene Ontology Consortium. The Gene Ontology project in 2008. Nucleic acids research volume 36. Database issue: D440-4 (2008).

[111] L. Hubert, P. Arabie, Comparing partitions. Journal of Classification 2 (1): 193-218 (1985).

[112] D. Tautz, T. Domazet-Lošo, The evolutionary origin of orphan genes. Nature Reviews Genetics 12, 692–702 (2011).

[113] S. Chen, B. H. Krinsky, and M. Long, New genes as drivers of phenotypic evolution. Nature Reviews Genetics 14 (9): 645-660 (2013).

[114] Z. W. Arendsee, L. Li, and E. S. Wurtele, Coming of age: orphan genes in plants. Trends in Plant Science 19 (11): 698-708 (2014).

[115] Y. Suenaga, et. al., NCYM, a Cis-antisense gene of MYCN, encodes a de novo evolved protein that inhibits GSK3β resulting in the stabilization of MYCN in human neuroblastomas. PLOS Genet 10 (1): e1003996 (2014).

[116] B. Heames, J. Schmitz, and E. Bornberg-Bauer, A continuum of evolving de novo genes drives protein-coding novelty in Drosophila. Journal of Molecular Evolution 88: 382–398 (2020).

[117] U. Singh, E. S. Wurtele, How new genes are born. eLife 9: e55136 (2020).

[118] M. Mohammed, M. B. Khan, and E. B. M. Bashier, Machine Learning: Algorithms and Applications. CRC Press (2016).