J Syst Sci Complex (2021) 34: 180–205

Formalising and Detecting Community Structures in Real World Complex Networks*

KUMAR Pawan · DOHARE Ravins

DOI: 10.1007/s11424-020-9252-3
Received: 16 September 2019 / Revised: 15 January 2020

Abstract Community structure is an integral characteristic of real world networks whichever processes or areas they emerge from. This paper addresses the problem of community structure detection theoretically as well as computationally. The authors introduce a number of concepts such as the neighbourhood and strength of a subgraph, p-community, local maximal p-community, hubs, and outliers that play elemental role in formalising the concept of community structure in complex networks. A few preliminary results have been derived that lead to the development of an algorithm for community structure detection in undirected unweighted networks. The algorithm is based on a local seed expansion strategy that uses the concept of interaction coefficient. The authors have analysed the algorithm on a number of parameters such as accuracy, stability, and quality on synthetic and real world networks from different areas.

Keywords Algorithm, hub, local maximal p-community, outlier, p-community, strength of a subgraph.

1 Introduction

The era of network science has emerged as a consequence of the study of graphs representing real data from different areas such as, but not limited to, biology[1], physics[2, 3] and social sciences[4]. However, due to large size, real world networks are often expressed into smaller, yet cohesive subgraphs that have least possible interconnections. Such subgraphs are loosely referred to as "communities". Quite a lot of efforts were made to define communities quantitatively in the last few decades, which resulted in a plethora of definitions. This may possibly be due to the diversity of the areas where real networks, and of course the researchers, belong to.

KUMAR Pawan
School of Sciences, Indira Gandhi National Open University, New Delhi 110068*, India.*
DOHARE Ravins (Corresponding author)
Centre for Interdisciplinary Research in Basic Sciences, Jamia Millia Islamia, New Delhi 110025*, India.*
Email: ravinsdohare@gmail.com.
*This paper was supported by the Science and Engineering Research Board, D.S.T., Govt. of India, India under Grant No. EEQ/2016/000509.
◇ *This paper was recommended for publication by Editor DI Zengru.*

Traditionally a clique (or a maximal clique) was treated as a definition of a community, especially by social scientists[5]. Such a definition is, however, so stringent that one can easily overlook the subgraphs which lack only a few edges to be a maximal clique. In the last half a century, a number of variations of the definition of community including LS-sets[6], n-clique[7], n-club and n-clan[8], k-core and k-plex[9], λ-sets and α-sets[10] were proposed and studied. Two meaningful characteristics one can infer from these definitions are "connectedness" and "cohesiveness". Connectedness implies that in a community there must be a path (a chain of nodes and edges) between every pair of its members. Cohesiveness generally means that the intra-community interactions are higher than the inter-community interactions. For example, Flake, et al. defined web communities as the vertex subsets with each vertex having internal degree no smaller than its external degree[11]. In the same spirit, Radicchi and co-workers proposed definitions of community in strong and weak sense[12]. They define a weak community as a subgraph whose internal degree is higher than its external degree and a strong community as a subgraph whose each vertex has internal degree greater than its external degree.

Many researchers even do not define communities explicitly. Instead, they treat a partition algorithmically optimised through some quality measure[13] as a plausible community structure of the input network. While optimisation of a quality function can offer appealing results in some cases, it carries with it a risk of overestimating or underestimating the number of communities. This phenomenon is often called the "resolution limit"[14]. And possibly, there is a very narrow scope for development of resolution limit-free methods that depend on global optimisation of a quality function[15].

Along with communities, real world networks also contain "hubs"[16, 17]. However, just like the case of communities, there is lack of clarity about what it means to be a hub. A general sense prevailing about hubs is that they have high degree and connect the different "parts" (cores or communities) of a network. Sometimes they form groups called "rich clubs" of a network[18]. While there are several methods that detect communities and cores[3, 11, 19], and even rich clubs[18, 20] in real world networks, there is no method which can detect both communities as well as hubs.

In this article, we aim, primarily to build a necessary framework for community structure theoretically. We introduce a number of concepts such as the neighbourhood and strength of a subgraph, p-community, local maximal p-community, hubs, and outliers that play elemental role in formalising the concept of community structure in complex networks. A few preliminary results have been derived that lead to the development of an algorithm for community structure detection in undirected unweighted networks. The algorithm is based on a local seed expansion strategy that uses the concept of interaction coefficient. We have tested the algorithm on real world networks of different sizes and from different areas and found satisfying results.

2 Theoretical Framework

2.1 Basic Terminology

For the theory that follows, we have assumed that $G = (V, E)$ is an undirected unweighted graph. A subgraph C of a graph G is said to be induced by a set $S \subseteq V(G)$ if $V(C) = S$ and $E(C) = \{(u, v) \in E(G) : u, v \in S\}$. Note that a single vertex of a graph is an induced subgraph of the graph. We shall use the notation $C \subseteq G$ to denote that C is an induced subgraph of G. For $S \subseteq V(G)$ and $C \subseteq G$, whenever we write $S \cap C, S \cup C$ or $S \backslash C$, we mean that the operation intersection, union, or subtraction respectively takes place between the set S and the vertex set of C. Let C_1 and C_2 be two subgraphs of G, then an edge-cut or simply a cut of C_1 and C_2, denoted by $\text{cut}(C_1, C_2)$, is the set of all edges that have one end point in C_1 and the other in C_2. Symbolically,

$$\text{cut}(C_1, C_2) = \{(u, v) \in E(G) : u \in C_1, v \in C_2\}.$$

When $v \notin C$, we shall use the notation $\text{cut}(v, C)$ to denote the set of edges with one endpoint as v, and another in C. Thus, $\text{cut}(v, C) = \varnothing$ is equivalent to $N_v \cap C = \varnothing$.

We generalise the neighbourhood of a vertex to the neighbourhood of a subgraph as follows.

Definition 2.1 (Neighbourhood of a subgraph) Let G be a graph and $C \subseteq G$. The neighbourhood of C, denoted as N_C, is defined as

$$N_C = \{u \in G \backslash C : u \in N_v \text{ for some } v \in C\}.$$

Definition 2.2 (Vertex/edge disjoint subgraphs) Let G be a graph and $C_1, C_2 \subseteq G$. Then, C_1 and C_2 are said to be vertex-disjoint if $V(C_1) \cap V(C_2) = \varnothing$. C_1 and C_2 are said to be edge-disjoint if $V(C_1) \cap V(C_2) = \varnothing$ and $\text{cut}(C_1, C_2) = \varnothing$.

While detecting the community structure of a graph, it is natural to take union of two or more subgraphs to form a larger subgraph. Recall that the union of two graphs G_1 and G_2 is the graph G with $V(G) = V(G_1) \cup V(G_2)$ and $E(G) = E(G_1) \cup E(G_2)$[21]. However, this definition cannot be applied to two subgraphs C_1 and C_2 of the same graph, because it does not take into account the edges in $\text{cut}(C_1, C_2)$. For this reason, below we present a generalised definition of the union of two subgraphs.

Definition 2.3 (Union of subgraphs) Let G be a graph and $C_1, C_2 \subseteq G$. The union of C_1 and C_2 is the subgraph $C_1 \cup C_2$ induced by $V(C_1) \cup V(C_2)$.

To clarify the definition, we have $V(C_1 \cup C_2) = V(C_1) \cup V(C_2)$ and $E(C_1 \cup C_2) = E(C_1) \cup E(C_2) \cup \text{cut}(C_1, C_2)$.

2.2 Strength of a Subgraph

Here we present the concepts prerequisite to the notion of a community. Two important characteristics of a community as we have mentioned in the introduction are connectedness and cohesiveness. To address the cohesiveness, we present the notion of the strength of a subgraph.

Definition 2.4 (Strength of a subgraph) Let G be graph. Let $C \subseteq G$ with $d(C) \neq 0$.

Then the strength of C, denoted as $s(C)$, is defined as

$$s(C) = \frac{d^{\text{int}}(C) - d^{\text{ext}}(C)}{d(C)}, \tag{1}$$

where $d^{\text{int}}(C) = \sum_{v \in C} |N_v \cap C|$, $d^{\text{ext}}(C) = \sum_{v \in C} |N_v \backslash C|$, and $d(C) = \sum_{v \in C} |N_v|$ are respectively the internal degree, the external degree and the degree of C.

Note that the strength of a subgraph that contains all isolated vertices is not defined, because such a subgraph would have total degree zero. In any case, such subgraphs are unimportant as far as we are concerned with community structure detection. However, the strength of an isolated vertex may be needed, so we set it to 1. Now, from the definition of strength, note that

$$\left|d^{\text{int}}(C) - d^{\text{ext}}(C)\right| \leq d^{\text{int}}(C) + d^{\text{ext}}(C) = d(C).$$

This implies $s(C) \in [-1, 1]$. It can easily be observed that when C has no internal edges $s(C) = -1$ and when C has no external edges $s(C) = 1$. Thus the strength of a subgraph indicates the level of cohesiveness it possesses. Note that cohesiveness have been defined in many other ways too, for example, as the density of edges inside a subgraph, i.e., the ratio of the number of edges present in the subgraph to the maximum possible number of edges. Here, the strength offers an alternate view on cohesiveness. It compares the the number of internal edges with the external edges.

Note that in the terminology of strength, Radicchi's weak community is a subgraph with positive strength, i.e., a subgraph C of a graph G is said to be a weak community if $s(C) > 0$. If we define $d_C^{\text{int}}(v) = |N_v \cap C|$, and $d_C^{\text{ext}}(v) = |N_v \backslash C|$, then Radicchi's strong community is a subgraph C with $d_C^{\text{int}}(v) > d_C^{\text{ext}}(v)$ for all $v \in C$. However, defining a community merely on the basis of strength is severely flawed. For instance, look at the subgraphs C_1, C_2 and C_3 in Figure 1. We find that the strength of each of them is 1, which is the maximum. But C_1 is not even connected, and C_2 is minimally connected in the sense that removal of any edge can disrupt the communication between the other nodes. The subgraph C_3, on the other hand, is maximally connected, i.e., it is a clique. By Radicchi's definition, all the three subgraphs C_1, C_2, and C_3 are weak as well as strong communities. Thus Radicchi's definitions fail to distinguish between the three subgraphs.

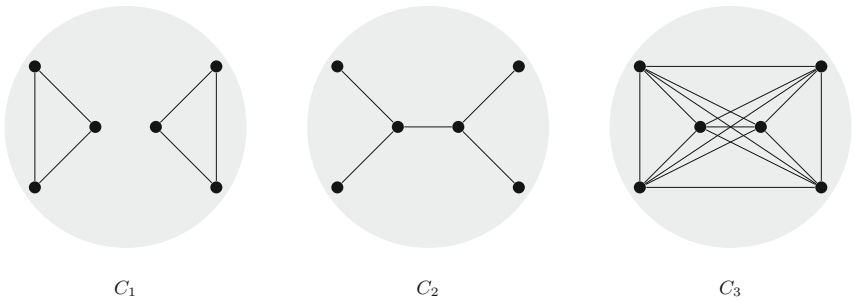


Figure 1 Three subgraphs with the same strength, but with different internal degrees

Before we come to our definition of community, let us look at a few preliminary results concerning the strengths of the union of two subgraphs.

Theorem 2.5 *Let G be a graph. Let C_1 and C_2 be two vertex-disjoint subgraphs of G with $s(C_1) = p > -1$ and $s(C_2) = q > -1$. Also, let $d^{\text{int}}(C_1) = d_1, d^{\text{int}}(C_2) = d_2$ and $k = |\text{cut}(C_1, C_2)|$. Then*

$$s(C_1 \cup C_2) = \frac{(1+p)(1+q)2k + (1+p)qd_2 + (1+q)pd_1}{(1+p)d_2 + (1+q)d_1}. \tag{2}$$

Proof Let $k_1 = |\text{cut}(C_1, G\backslash(C_1 \cup C_2))|$ and $k_2 = |\text{cut}(C_2, G\backslash(C_1 \cup C_2))|$. Then, note that

$$p = s(C_1) = \frac{d_1 - (k + k_1)}{d_1 + (k + k_1)}.$$

It can be rewritten as

$$k + k_1 = \left(\frac{1-p}{1+p}\right) d_1. \tag{3}$$

Similarly, we can write

$$k + k_2 = \left(\frac{1-q}{1+q}\right) d_2. \tag{4}$$

These give us

$$\begin{aligned} s(C_1 \cup C_2) &= \frac{(d_1 + d_2 + 2k) - (k_1 + k_2)}{(d_1 + d_2 + 2k) + (k_1 + k_2)} \qquad (5)\\ &= \frac{d_1 + d_2 + 4k - \left(\frac{1-p}{1+p}\right) d_1 - \left(\frac{1-q}{1+q}\right) d_2}{d_1 + d_2 + \left(\frac{1-p}{1+p}\right) d_1 + \left(\frac{1-q}{1+q}\right) d_2} \\ &= \frac{(1+p)(1+q)2k + (1+p)qd_2 + (1+q)pd_1}{(1+p)d_2 + (1+q)d_1}. \end{aligned}$$

The proof is finished. ∎

Theorem 2.5 gives us a precise formula for computation of the strength of the union of two vertex-disjoint subgraphs. The following corollary follows immediately from Theorem 2.5.

Corollary 2.6 *Let C_1 and C_2 be two edge-disjoint subgraphs of a graph G with strengths p and q, respectively. Then*

$$\min\{p, q\} \leq s(C_1 \cup C_2) \leq \max\{p, q\}.$$

If we are given that the strength of two subgraphs is greater than their individual strengths then there must be at least a certain number of edges crossing the two subgraphs. This is stated below.

Theorem 2.7 *Let C_1 and C_2 be two disjoint subgraphs of a graph G, with strengths $p > -1$ and $q > -1$,respectively. Then, if $s(C_1 \cup C_2) > \max\{p, q\}$, then $\text{cut}(C_1, C_2) \neq \varnothing$.*

The proof of this theorem is given in the Appendix. It can be seen from the proof of Theorem 2.7 that its converse is not true. For a counter-example, look at the subgraphs $C_1 = \{1, 2, 3, 4, 5\}$ and $C_2 = \{6, 7, 8, 9\}$ in Figure 2. Here $s(C_1) = \frac{2}{3}$ and $s(C_2) = \frac{1}{11}$. Also $s(C_1 \cup C_2) = \frac{15}{23}$, which is not greater than the maximum of $s(C_1)$ and $s(C_2)$, although $\text{cut}(C_1, C_2) \neq \varnothing$.

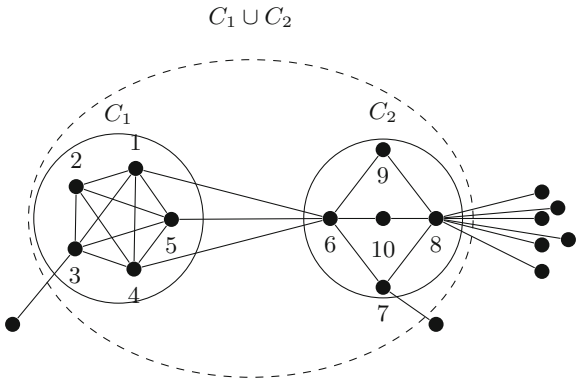


Figure 2 C_1 and C_2 are the subgraphs induced by $\{1, 2, 3, 4, 5\}$ and $\{6, 7, 8, 9, 10\}$, respectively. $s(C_1) = \frac{2}{3}$ and $s(C_2) = \frac{1}{11}$. Also $s(C_1 \cup C_2) = \frac{15}{23}$, which is no greater than $\max\{\frac{2}{3}, \frac{1}{11}\}$

If we take $p = q$ in Theorem 2.7, we get a much stronger result as given in the following corollary.

Corollary 2.8 *Let C_1 and C_2 be two subgraphs of a graph with strength $-1 < p < 1$. Then $s(C_1 \cup C_2) > p$ iff $\text{cut}(C_1, C_2) \neq \varnothing$.*

Proof Trivial. ∎

All the results discussed above are aimed at computing the lower bounds on the strength of the union of two vertex-disjoint subgraphs. Now, let us consider the subgraphs C_1 and C_2 of a graph G as shown in Figure 3. Note that $s(C_1) = 1/3 = s(C_2)$. Also you can compute that $s(C_1 \cup C_2) = 2/3$. So in this case we have

$$s(C_1 \cup C_2) \geq s(C_1) + s(C_2). \tag{6}$$

Is this true for all pairs of subgraphs of G? Clearly, the answer is no. Because, for example, if we look at the subgraphs C_2 and C_3 of G in Figure 3 we find that $s(C_2 \cup C_3) = 19/21$ which is smaller than $s(C_2) + s(C_3)$.

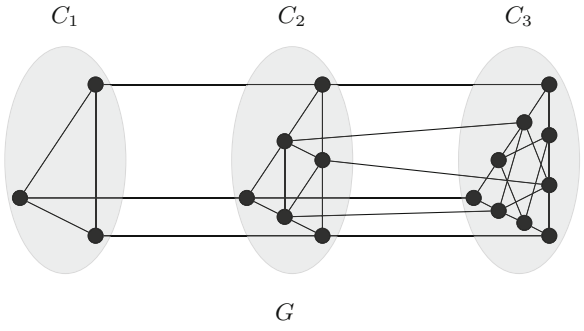


Figure 3 Three subgraphs C_1, C_2 and C_3 of a graph G with $s(C_1) = 1/3 = s(C_2)$ and $s(C_3) = 2/3$

So under what conditions on C_1 and C_2, we can ensure that (6) holds? The answer lies in the following theorem.

Theorem 2.9 *Let C_1 and C_2 be two vertex-disjoint subgraphs of a graph G with $s(C_1) = p, s(C_2) = q$ and $p + q < 1$. Then $s(C_1 \cup C_2) \geq p + q$ if and only if*

$$|\text{cut}(C_1, C_2)| \geq \frac{1}{2}\left[\left(\frac{q}{1-p}\right) d^{\text{ext}}(C_1) + \left(\frac{p}{1-q}\right) d^{\text{ext}}(C_2)\right]. \tag{7}$$

The proof of Theorem 2.9 is given in the Appendix. An important thing about Theorem 2.9 is that its premise and conclusion imply each other. It gives us the precise condition under which the strength of the union of two subgraphs is greater than or equal to the sum of the strengths of the individual subgraphs. We shall return to it in Section 3 to see its application.

2.3 Community Structure at Local Level

Here we shall present the basic ingredients essential for defining the "community structure" in a graph.

Definition 2.10 (Partition) Let G be a graph, and $\mathcal{P}$ a collection of subgraphs of G. Then $\mathcal{P}$ is called a partition of G if

i) each $C \in \mathcal{P}$ is an induced subgraph of G,

ii) $V(C_1) \cap V(C_2) = \varnothing$, whenever $C_1, C_2 \in \mathcal{P}$ and $C_1 \neq C_2$,

iii) $\cup_{C \in \mathcal{P}} V(C) = V(G)$.

When the condition (ii) is relaxed, $\mathcal{P}$ is called a cover of G.

So what kind of subgraphs can $\mathcal{P}$ hold? Here we propose different notions essential for determining the community structure of a graph. First we define the notion of p-community.

Definition 2.11 (p-community) A connected subgraph C of a graph G with strength $s(C) = p > 0$ is called a p-community of G.

From a user's point of view p-communities with p closer to 1 might be useful. Since, 1-communities are precisely the components of the graph, they can be treated the most generalised communities. However, if the size of a component is large it is possible that it contains many meaningful communities of small size and with comparatively small p. On the other hand p-communities with p close to 0 might be numerous, and hence they would also be unrealistic.

Thus, it would be desirable to search for those p-communities that have maximised p in the interval $(0, 1)$, at local or global scale. However, global maximisation of p may overlook meaningful communities at local level. This is what Corollary 2.8 tells us about. For example, in the graph given in Figure 4 the subgraphs C_1 and C_2 have strengths equal to 29/35 each. But, their union has strength $s(C_1 \cup C_2) = 31/35$ which is greater than 29/35. In fact, it can be verified that

$$s(C_1 \cup C_2 \cup C_3) > \max\{s(C_1 \cup C_2), s(C_3)\}.$$

Thus, a global optimisation of the strengths would not resolve the individual communities C_1, C_2, C_3 etc. This motivates us to look for p-communities which have maximum strengths within their neighborhoods. We define them as follows.

Definition 2.12 (Local maximal p-community) Let C be a p-community of a graph G. Then C is said to be a local maximal p-community if

$$s(C \cup X) < p, \quad \text{for all} \quad X \subseteq N_C. \tag{8}$$

The definition of a local maximal p-community can be distinguished from all the other existing definitions of community. We shall present a few examples. First a local maximal p-community need not be a clique or an n-clique. Neither it is necessarily a k-core or a k-plex. This is because, the definition of local maximal p-community emphasises the requirement of minimum external degree and/or maximum internal degree at local level, i.e., within the neighbourhood. As a result, it is quite possible that a local maximal p-community may include nodes with internal degree smaller than their external degree if their inclusion does not decrease its strength . This means that a local maximal p-community need not be a strong community or an LS-set.

Example 2.13 Consider the subgraph $C = \{1, 2, 3, 4, 5\}$ in Figure 4, where C is a web community and 3/17-community. But, you can see that C is not a local maximal 3/17-community, because $s(C \cup N_C) = 7/27 > 3/17$. However, $C \cup N_C$ is a local maximal 7/27-community.

Example 2.14 Consider the graph in Figure 4 again. A little computation reveals that $s(C_1) = s(C_2) = s(C_4) = 29/35$ and $s(C_3) = s(C_5) = 7/9$. Now, it can be proved that each of C_1, C_2 and C_4 is a local maximal 29/35-community, whereas each of C_3 and C_5 is a local maximal 7/9-community.

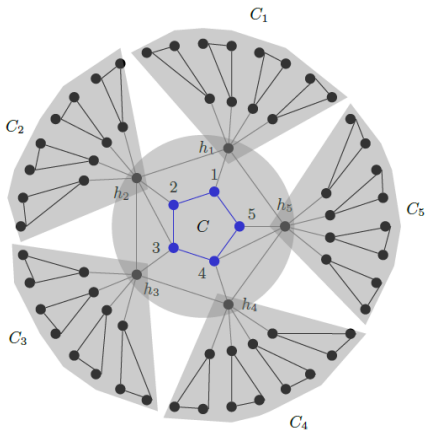


Figure 4 A graph showing with local maximal p-communities

The notion of p-community or local maximal p-community captures just one aspect of real world systems. Networks emerging, for example, from biological[16, 18] and social sciences[22] are often composed of "hubs", that inescapably influence their dynamics. Hubs have been defined in different ways, depending on their role and the type of networks. A hub generally possesses high degree, connects different communities, and may participate in a rich club of the network[20]. Let us look at a simple description of a modular network in Figure 5. The node v connects the communities C_1, C_2, and C_3. No communication would take place among C_1, C_2, and C_3 if v is removed. Thus, v works as a 'hub' through which information is exchanged between different modules of the network. Formally, we define it as follows.

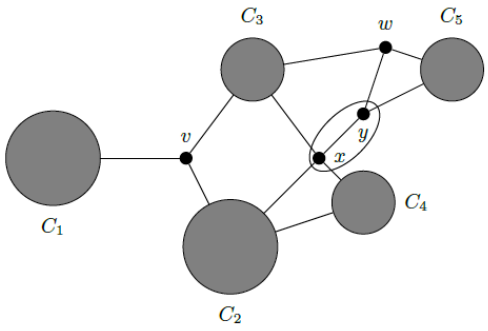


Figure 5 A model graph showing a community structure with hubs and outliers

Definition 2.15 (Hub/outlier) Let G be a graph with partition $\mathcal{P}$. Then a vertex $v \in \mathcal{P}$ is called a hub if

i) there exist at least three subgraphs $C_1, C_2, C_3 \in \mathcal{P}$ such that for each $1 \leq i \leq 3$, $s(C_i) > 0$ and $\text{cut}(v, C_i) \neq \varnothing$.

ii) whenever $C_1, C_2 \in \mathcal{P}$ are such that $C_1 \neq C_2$, $\text{cut}(v, C_1) \neq \varnothing$ and $\text{cut}(v, C_2) \neq \varnothing$, then $\text{cut}(C_1, C_2) = \varnothing$.

A connected subgraph $C \in \mathcal{P}$ is called an outlier if $s(C) \leq 0$.

A hub essentially controls the flow of information among the modules that are connected to it. On the other hand, outliers are usually small connected subgraphs attached to a hub or a p-community. Look at the community structure in Figure 5 again. Here

$$\mathcal{P} = \{C_1, C_2, C_3, C_4, C_5, v, w, \{x, y\}\},$$

where we have assumed that for each $1 \leq i \leq 5$, $s(C_i) > 0$. We can see that v is a hub, because it satisfies all the three conditions of a hub. However, node w is not a hub, as the strength of the subgraph $\{x, y\}$ is $-1/3$. Rather, w is an outlier. Likewise, $\{x, y\}$ is also an outlier.

Note that if $\mathcal{P}$ is a partition of a graph G, and $C \in \mathcal{P}$ is connected, then C must be either a p-community or a hub, or an outlier. So, now we can give a formal meaning to the term "community structure". For this we need a quality measure.

Definition 2.16 (Disjoint/overlapping community structure) Let G be a graph and $\mathcal{P}$ a partition of G. Then $\mathcal{P}$ is called a disjoint community structure of G with respect to a quality measure M if

i) every $C \in \mathcal{P}$ is a local maximal p-community, a hub, or an outlier;

ii) $M(\mathcal{P}) \geq 0.5$.

If $\mathcal{P}$ is a cover, instead of a partition, then $\mathcal{P}$ is called an overlapping community structure of G w.r.t. M.

Obviously, there can be multiple partitions/covers of a graph. One has to look for the one which is optimal with respect to some quality measure M, that is, for which M attains the maximum value. Unfortunately not all quality measures correlate strongly[4]. The traditional quality measure called "modularity" by Newman and Girvan[13] has serious drawbacks. Once such drawback is its resolution limit[14] which shows that the communities smaller than a specific size are undetectable by the modularity. But due to lack of other sophisticated quality measures it was almost treated as a benchmark quality measure in the last decade. Recently, the quality measure overlapping modularity (Q_{ov}) of Nicosia, et al.[23] was compared with a new measure called the weighted overlapping modularity (Q_{wo})[24]. It was found that the Q_{wo} measure evaluates the quality of overlapping community structures more sharply than Q_{ov}. Both of these measures can equally be applied on disjoint community structures in undirected and unweighted real world networks. Below we reproduce the formula for Q_{wo} for a cover $\mathcal{P}$

$$Q_{\text{wo}}(\mathcal{P}) = \frac{1}{|\mathcal{P}|} \sum_{C \in \mathcal{P}} \left(1 - \frac{\sum_{v \in C} |m_v|}{|C| \,.\, |\mathcal{P}|}\right) \frac{d^{\text{int}}(C)}{d(C)},$$

where m_v is the set of all the memberships of v. When $\mathcal{P}$ is a partition, the formula reduces to

$$Q_{\text{wo}}(\mathcal{P}) = \left(1 - \frac{1}{|\mathcal{P}|}\right) \left[\frac{1}{|\mathcal{P}|} \sum_{C \in \mathcal{P}} \frac{d^{\text{int}}(C)}{d(C)}\right]. \tag{9}$$

This is because for all $C \in \mathcal{P}$

$$\sum_{v \in C} |m_v| = |C| \,.$$

Arguably, every overlapping community structure can be expressed into a disjoint community structure. To see how, let $\mathcal{P}$ be a cover of a graph and $C \in \mathcal{P}$. Assume that C is connected, otherwise we can write it as the disjoint union of connected components. Assume that $|C| > 1$. Then $s(C) > 0$ or $s(C) \leq 0$ which implies that either C is a p-community or an outlier. On the other hand, when $|C| = 1$, either C is a hub or an outlier.

Let us look at a particular example. Consider the cover

$$\mathcal{P} = \big\{\{1, 2, 3, 4, 5, h_1, h_2, h_3, h_4, h_5\}, C_1, C_2, C_3, C_4, C_5\big\}$$

of the graph in Figure 4. Note that each h_i has dual membership. We can compute that

$$Q_{\text{wo}}(\mathcal{P}) = \frac{1}{6}\left[\frac{45}{60} \times \frac{34}{54} + \frac{5}{6}\left(3 \times \frac{32}{35} + 2 \times \frac{32}{36}\right)\right] = 0.71.$$

If we look at the community structure closely, we find that removal of h_i's leads to several disjoint communities. In fact, for each i, $C_i \backslash h_i$ represents four local maximal p-communities, each with strength $p = 5/7$. This means each h_i could be a hub if we decompose the overlapping structure into a disjoint one. Let the new community structure be represented by the partition $\mathcal{P}'$ defined as

$$\mathcal{P}' = \big\{\{1,2,3,4,5\}\big\} \bigcup_{i=1}^{5} \{h_i\} \bigcup_{i=1}^{5} \{C_i \backslash h_i\}.$$

In this partition, $\{1,2,3,4,5\}$ is a 3/17-community, each h_i is a hub, and for each i, $C_i \backslash h_i$ is group of 4 disjoint local maximal p-communities each with strength $p = 5/7$. Thus, there are total 26 subgraphs in $\mathcal{P}'$ whereas $\mathcal{P}$ contains just 6 subgraphs. Thus $\mathcal{P}'$ gives a finer community structure than $\mathcal{P}'$. Let us now compute the Q_{wo} score for $\mathcal{P}'$. We have

$$Q_{\text{wo}}(\mathcal{P}') = \frac{1}{26}\left(1 - \frac{1}{26}\right)\left(\frac{10}{17} + 5 \times 0 + 20 \times \frac{6}{7}\right) = 0.66.$$

Likewise we computed the overlapping modularity scores for both the partitions, and obtained $Q_{\text{ov}}(\mathcal{P}) = 0.84$ and $Q_{\text{ov}}(\mathcal{P}') = 0.82$. So, qualitywise $\mathcal{P}$ is slightly better than $\mathcal{P}'$. Nevertheless, $\mathcal{P}'$ gives deeper and finer structure than $\mathcal{P}$.

2.4 Interaction Coefficient

Broadly there are two kinds of methods, namely global methods[25] and local methods[26], that explore the community structure of a network. The algorithm we shall develop in the next section, is based on a local method that expands seeds through their neighbourhoods. In the expansion phase of such a method, there is a subgraph C and a node v in C's neighbourhood, between which the interaction takes place. Inclusion of v into C depends on the strength of their interaction. To quantify the interaction of a node with a community, we define the following coefficient.

Definition 2.17 (α-interaction coefficient) Let C be a subgraph of a graph G and $0 \leq \alpha \leq 1$. The α-interaction coefficient of a vertex $v \in V(G)$ with C is defined as

$$\xi_\alpha(v, C) = \frac{\alpha\,|N_v \cap C| + (1-\alpha)\,|N_{vC}|}{d_v}, \tag{10}$$

where $N_{vC} = N_v \cap N_C$, and d_v is the degree of v.

By definition, $\xi_\alpha(v, C)$ is the weighted average of the quantities $|N_v \cap C|$ and $|N_{vC}|$, where the first one represents the number of neighbours of v in C, and the second one the number of neighbours of v in the neighbourhood of C. It is clear that $\xi_\alpha(v, C) \geq 0$. The lower bound is achieved when $N_v \cap C = N_{vC} = \varnothing$. In order to estimate the upper bound, note that

$$\xi_\alpha(v, C) = \begin{cases} \alpha, & \text{when } N_v \subseteq C, \\ 1 - \alpha, & \text{when } N_v \subseteq N_C. \end{cases}$$

This implies that $\xi_\alpha(v, C) \leq \max\{\alpha, 1-\alpha\}$. Let us consider an example.

Example 2.18 Consider the graph in Figure 5. We can compute that

$$\xi_\alpha(v, C_1) = \frac{\alpha \cdot 1 + (1-\alpha) \cdot 0}{3} = \frac{\alpha}{3}.$$

To compute $\xi_\alpha(w, C_5)$, note that $|N_w \cap C_5| = |N_{wC_5}| = 1$. So,

$$\xi_\alpha(w, C_5) = \frac{\alpha \cdot 1 + (1-\alpha) \cdot 1}{3} = \frac{1}{3}.$$

From the example above we can see that $\xi_\alpha(v, C)$ does not always depend on α. In fact, it can easily be seen that $\xi_\alpha(v, C)$ is independent of α when $|N_v \cap C| = |N_{vC}|$. So, what exactly is the importance of α? We shall address this question in the next section. At the moment, let us look at the role the interaction coefficient plays in detecting the community structure of a network. Note that $\xi_\alpha(v, C)$ measures the strength of interaction between v and C. This information can be used by an algorithm that detects communities through a seed expansion technique. Assume, for example, an algorithm starts at a subgraph C as an 'initial community' and expands it by adding nodes from its neighbourhood to it as long a certain condition is satisfied. Then to get the maximum possible expansion of C into a full community, one needs to add nodes that have highest interaction coefficient with C at each step. This is because, if $\xi_\alpha(v, C)$ is highest for some $v \in N_C$, then it indicates that a large fraction of the neighbours of v is in $C \cup N_C$.

3 The Algorithm

3.1 Rationale of the Algorithm

The theoretical framework that we have proposed in the preceding section throws the fundamental question: Can we precisely detect the local maximal p-communities? If yes, how efficiently? The answer to the first question demands further work on the structure of the local maximal p-communities. To answer second, consider a subgraph C with $|N_C| = \ell$ and $s(C) = p$. Then there are $\sum_{i=0}^{\ell} \binom{\ell}{i} = 2^\ell - 1$ nonempty subsets T of N_C for which we have to test the condition (8). Thus it would take exponential time $O(2^\ell)$ for checking whether C is local maximal p-community or not. However, we can always develop a local expansion type algorithm that begins each community with a single vertex and extends it as long as a certain quality function is increased[27]. Here we can employ strength as a quality function. Usually expansion algorithms expand seeds by adding one vertex at a time. We shall develop a method that allows a bunch of vertices to be added at each step during expansion phase. For this task we need another function that can control the vertices to be added at each step. We shall use the interaction coefficient for this task as it essentially determines the closeness between a vertex and a subgraph.

3.2 Description of the Algorithm

The algorithm takes a graph G as input, and returns a partition that contains p-communities, hubs and outliers. It uses primarily three functions, namely EXPAND, GET-NEAREST-

COMS, and GET-HUBS. First we describe the main algorithm, and then each of these functions, in detail.

Algorithm 1: Algorithm to find p-communities, hubs and outliers in an unweighted undirected network .

Data: A graph G
Result: p-communities, hubs and outliers of G
$U \leftarrow V(G), \mathcal{P} \leftarrow \varnothing$;
$i \leftarrow 1$;
while $U \neq \varnothing$ **do**
  $C \leftarrow \{u\}$, where u is a seed chosen from U;
  EXPAND(G, C, U);
  $U \leftarrow U \backslash C$;
  if $0 \leq s(C) < 0.5$ *and* $\mathcal{P} \neq \varnothing$ **then**
    $L \leftarrow$GET-NEAREST-COMS$(G, \mathcal{P}, C)$;
    if $L \neq \varnothing$ **then**
      Pick $\ell \in L$ randomly.;
      $\mathcal{P}[\ell] \leftarrow \mathcal{P}[\ell] \cup C$;
    else
      $\mathcal{P}[i] \leftarrow C$;
      $i \leftarrow i + 1$;
    end
  else
    $\mathcal{P}[i] \leftarrow C$;
    $i \leftarrow i + 1$;
  end
end
$H \leftarrow$GET-HUBS$(G, \mathcal{P})$;
$\mathcal{O} \leftarrow \varnothing$;
foreach $C \in \mathcal{P}$ **do**
  if $C \notin H$ and $s(C) \leq 0$ **then**
    $\mathcal{O} \leftarrow \mathcal{O} \cup \{C\}$;
  end
end

Main Algorithm The main algorithm (given in Algorithm 1) maintains four sets namely $U, \mathcal{P}, H$, and $\mathcal{O}$. The set U stores the vertices that are still unexplored, i.e., the vertices whose memberships to a community is yet to be decided. The set $\mathcal{P}$ stores the communities that have been explored. The sets H and $\mathcal{O}$ store hubs and outliers, respectively. Note that $\mathcal{P}$ is an indexed set, and $\mathcal{P}[i]$ indicates the community in $\mathcal{P}$ at the i^{th} index. Step 1 initializes U with $V(G)$, and $\mathcal{P}$ with $\varnothing$. The variable i holds the index of the community being expanded. Then Steps 3–20 describe a while loop which runs as long as U is nonempty. In Step 4, a set

C is initialized with a randomly chosen vertex u from U. The set C, actually, represents the community being expanded currently. To expand C, the function EXPAND is called at line 5. Afterwards, C is removed from U.

Now the condition at line 7 checks whether or not $0 \leq s(C) < 0.5$ and $\mathcal{P} \neq \varnothing$. Note that at the first encounter of this condition $\mathcal{P}$ is empty, which means that C is the first community. So, C is stored in $\mathcal{P}$ at the i^{th} index, and i is incremented by 1 (lines 16–19). On the other hand, if the condition at line 7 is true, the indices of the communities that are nearest to C are obtained in the set L, using the function GET-NEAREST-COMS. Next, if L is nonempty, an index ℓ is chosen from L randomly and C is stored in $\mathcal{P}$ at the ℓ^{th} index. Otherwise, $L = \varnothing$, which means that there is no community in $\mathcal{P}$ nearest to C. In this case again, C is stored in $\mathcal{P}$ at the i^{th} index, and i is incremented by 1.

The next task is to detect the hubs in the network, which is done by calling the function GET-HUBS. Finally, the lines 22–27 detect the outliers, which are stored in $\mathcal{O}$.

Seed Expansion The function EXPAND (given in Procedure 2) takes a set C as input along with a set U, and the input graph G. The set C, which initially contains a single node, is expanded by adding to it the vertices from U. For this task, a set S is needed that stores the nodes to be added to C. The variables p and p_{new} are initialised to -1. Now a repeat-until loop (lines 3–17) runs until p becomes larger than p_{new}. In this loop, look at the line 5, which puts in S those nodes of U that lie in the neighbourhood of C. Note that $S = \varnothing$ means that C cannot be expanded further, and so we must break out of the loop.

Here we shall discuss how the choice of α is crucial to the expansion phase of the algorithm. Since in the beginning C contains just one vertex, the expansion of C solely depends on the distribution of the edges among N_C. This is because each vertex $v \in N_C$ has exactly one neighbour in C, with rest of the neighbours in N_C or possibly outside $C \cup N_C$. So, in this case we take $\alpha = 0$. Consequently, Equation (10) reduces to

$$\xi_\alpha(v, C) = \frac{|N_{vC}|}{d_v}.$$

This indicates that if v has a large fraction of neighbours in N_C, its chances for inclusion to C are high. Now as C grows there are sufficient internal members in C, so v's chance for inclusion to C must depend on the number of neighbours of v in C, rather than in N_C. Due to these observations, we fix the value of α as follows.

$$\alpha = 1 - \frac{1}{|C|}.$$

This means, as C grows larger, S must contain fewer, and carefully selected vertices. Accordingly, S must retain only those vertices v in it for which $\xi_\alpha(v, C)$ is higher than or equal to the average value of α-interaction coefficients of all the vertices in S with C (see line 10). Moreover, when C reaches a certain size threshold, which can vary from network to network, expansion of C must be even more sophisticated. It is reasonable to assume that when C attains the size $\sqrt{|G|}$, the set S could be refined by repeating the process at line 10 one more time (this is done

at lines 11–13). Finally, S is included to C, strength of C is computed and stored in p_{new}, and S is removed from U.

Note that the loop repeat-until terminates in two cases, namely when $S = \varnothing$ or $p_{\text{new}} < p$. In case, it terminates due to $p_{\text{new}} < p$, S must be removed from C in order to retain the old version of C.

Procedure `EXPAND`(G, C, U)

Result: C
$S \leftarrow \varnothing$;
$p \leftarrow -1, p_{\text{new}} \leftarrow -1$;
repeat
  $p \leftarrow p_{\text{new}}$;
  $S \leftarrow N_C \cap U$;
  if $S = \varnothing$ **then**
    break;
  end
  $\alpha \leftarrow 1 - 1/|C|$;
  $S \leftarrow \left\{ v \in S : \xi_\alpha(v, C) \geq \frac{1}{|S|} \sum_{u \in S} \xi_\alpha(u, C) \right\}$;
  if $|C|^2 \geq |G|$ **then**
    $S \leftarrow \left\{ v \in S : \xi_\alpha(v, C) \geq \frac{1}{|S|} \sum_{u \in S} \xi_\alpha(u, C) \right\}$;
  end
  $C \leftarrow C \cup S$;
  $p_{\text{new}} \leftarrow s(C)$;
  $U \leftarrow U \backslash S$;
until $p_{new} < p$;
if $S \neq \varnothing$ **then**
  $C \leftarrow C \backslash S$;
end

Getting Nearest Communities The function GET-NEAREST-COMS (given in Procedure GET-NEAREST-COMS) takes the parameters $G, \mathcal{P}$, and C and returns the indices of the communities in $\mathcal{P}$ that are 'nearest' to C. Note that $\mathcal{P}$ is an indexed set that stores the communities that have been explored. The criterion for deciding whether a community $C' \in \mathcal{P}$ is nearest to C or not is as follows.

Let $p = s(C)$ and $q = s(C')$ be such that $0 < p < 0.5$ and $0 < q < 0.5$. Then Theorem 2.9 implies that $s(C \cup C')$ is greater than or equal to $p + q$ if and only if

$$|\text{cut}(C, C')| \geq \frac{1}{2} \left[\left(\frac{q}{1-p} \right) d^{\text{ext}}(C) + \left(\frac{p}{1-q} \right) d^{\text{ext}}(C') \right]. \tag{11}$$

So, when the above condition is satisfied the index of C' is stored in the set L.

Procedure GET-NEAREST-COMS$(G, \mathcal{P}, C)$

Result: L
$L \leftarrow \varnothing$;
foreach $C' \in \mathcal{P}$ **do**
  $p \leftarrow s(C), q \leftarrow s(C')$;
  if $0 \leq q < 0.5$ **then**
    if $\text{cut}(C, C') = \varnothing$ **then**
      continue;
    end
    if Condition (11) is true **then**
      $L \leftarrow L \cup \{\ell\}$, where ℓ is the index of C' in $\mathcal{P}$;
    end
  end
end

Getting Hubs To detect the hubs we use the function GET-HUBS (given in Procedure GET-HUBS) which takes G, and $\mathcal{P}$ as parameters and returns a set H containing all the hubs in $\mathcal{P}$. The procedure is as follows.

Let $C \in \mathcal{P}$ be such that $C = \{u\}$, for some $u \in G$. Let L contain the indices of all those communities in $\mathcal{P}$ that contain a neighbour of u. Also let *adj_pcoms*, initialised to zero, contain the count of the p-communities that contain a neighbour of u. Look at the **for** loop at lines 7–13 now. Its task is to increase the counter *adj_pcoms* if there are any p-communities containing a neighbour of u. The loop stops as soon as *adj_pcoms* reaches to 3. Thus, at the end of this loop if *adj_pcoms* still remains smaller than 3, it means that u is not a hub, and as a consequence the rest of the code is skipped.

On the other hand, if *adj_pcoms* is equal to 3, we test whether any three p-communities adjacent to u are mutually edge-disjoint or not. If yes, u is included to H (see lines 18–31).

4 Results and Analysis

Community detection algorithms are apparently heuristic, that is, they use some kind randomness at some phase. For example, the seed expansion phase of our algorithm selects seeds randomly. Not surprisingly, then such algorithms produce different partitions at different runs. Accordingly, we analyse our algorithm in terms of stability and accuracy. Besides we shall also see how the performance of our algorithm can be improved. Finally we shall look at its running time on different real world networks.

4.1 Real World Networks and Quality Measures

In this section we apply our algorithm on real world networks and analyse the community structures using Q_{ov} and Q_{wo}. The selected real world networks are given in Table 1. These networks are taken from diverse areas such as social sciences (POLBLOGS), technology (INTERNET, POWER-GRID), communication (FACEBOOK, EMAIL-ENRON, LOC-

Procedure `GET-HUBS`$(G, \mathcal{P})$

Result: H
$H \leftarrow \varnothing$;
foreach $C \in \mathcal{P}$ **do**
  if $|C| = 1$ **then**
    Let u be the single node in C;
    Let L contain the indices of the communities in $\mathcal{P}$ that contain the neighbours of u;
    $adj\_pcoms \leftarrow 0$;
    foreach $\ell \in L$ **do**
      if $adj\_pcoms < 3$ **then**
        if $s(\mathcal{P}[\ell]) > 0$ **then**
          $adj\_pcoms \leftarrow adj\_pcoms + 1$;
        end
      end
    end
    if $adj\_pcoms < 3$ **then**
      continue;
    end
    $flag \leftarrow true$;
    foreach $\ell \in L$ **do**
      foreach $\ell' \in L$ **do**
        if $\ell' > \ell$ and $\text{cut}(\mathcal{P}[\ell], \mathcal{P}[\ell']) \neq \varnothing$ **then**
          $flag \leftarrow false$;
          break;
        end
      end
      if $flag = false$ **then**
        break;
      end
    end
    if $flag = true$ **then**
      $H \leftarrow H \cup \{u\}$;
    end
  end
end

BRIGHTKITE), biology (YEAST-PPI), and scientific collaborations (NETSCIENCE, HEP-THEORY, ASTRO-PHSICS, COND-MATTER). We run the algorithm 10 times on each of these networks, and record the average number of total communities, p-communities, and hubs in Table 2, along with the average scores of Q_{ov} and Q_{wo}.

Table 1 Details of networks studied in this paper

Network	**Nodes**	**Edges**	**Ref.**
POLBLOGS	1490	16715	[28]
NETSCIENCE	1589	2742	[25]
YEAST-PPI	2361	6646	[29]
FACEBOOK	4039	88234	[30]
POWER-GRID	4941	6594	[31]
HEP-THEORY	5835	13815	[32]
ASTRO-PHYSICS	14845	119652	[32]
INTERNET	22963	48436	[33]
EMAIL-ENRON	36693	183831	[34]
COND-MATTER	39577	175692	[32]
LOC-BRIGHTKITE	58228	214078	[35]

Table 2 Assessment of community structures of real world networks using Q_{ov} and Q_{wo}

Networks	**p-coms**	**hubs**	**outliers**	Q_{ov}	Q_{wo}
POLBLOGS	4	0	3	0.78	0.56
NETSCIENCE	36	1	8	0.80	0.70
YEAST-PPI	50	3	97	0.52	0.40
FACEBOOK	50	0	20	0.90	0.60
POWER-GRID	540	30	60	0.75	0.68
HEP-THEORY	430	12	138	0.69	0.63
ASTRO-PHYSICS	100	2	118	0.64	0.60
INTERNET	310	0	380	0.55	0.44
EMAIL-ENRON	1500	6	94	0.68	0.86
COND-MATTER	1510	0	120	0.60	0.88
LOC-BRIGHTKITE	1480	5	705	0.60	0.44

It can be seen that both Q_{ov} and Q_{wo} achieve moderate to high score for all the networks except YEAST-PPI, INTERNET and LOC-BRIGHTKITE where Q_{wo} scores are below 0.50. The largest number of hubs are observed in POWER-GRID.

4.2 Stability

The stability of an algorithm indicates how similar two partitions produced at different runs are. There are some well known measures, e.g., normalised mutual information (NMI)[36], that can be used to assess the stability of community detection algorithms.

To analyse the stability of our algorithm we use the NMI measure, which in the context of disjoint community structure can be defined as follows. Let G be a graph, and $\mathcal{P}_1$ and $\mathcal{P}_2$ its two partitions. Then a confusion matrix $N = (N_{ij})$ is defined with $|\mathcal{P}_1|$ rows and $|\mathcal{P}_2|$ columns, where N_{ij} is the number of nodes in the community i of $\mathcal{P}_1$ that appear in the community j of $\mathcal{P}_2$. Then the NMI measure of similarity between two partitions $\mathcal{P}_1$ and $\mathcal{P}_2$ is:

$$\text{NMI}(\mathcal{P}_1, \mathcal{P}_2) = \frac{-2\sum_{i=1}^{|\mathcal{P}_1|}\sum_{j=1}^{|\mathcal{P}_2|} N_{ij}\log\left(\frac{N_{ij}|G|}{N_{i\bullet}N_{\bullet j}}\right)}{\sum_{i=1}^{|\mathcal{P}_1|} N_{i\bullet}\log\left(\frac{N_{i\bullet}}{|G|}\right) + \sum_{j=1}^{|\mathcal{P}_2|}\log\left(\frac{N_{\bullet j}}{|G|}\right)}.$$

Here $N_{i\bullet} = \sum_{j=1}^{|\mathcal{P}_2|} N_{ij}$ and $N_{\bullet j} = \sum_{i=1}^{|\mathcal{P}_1|} N_{ij}$. The NMI score of a pair of partitions can vary from 0 to 1, where 0 indicates that the partitions are completely different, and 1 that they are identical.

We select four real world networks namely, POWER-GRID, HEP-THEORY, EMAIL-ENRON, and COND-MATTER, and run the algorithm 10 times on each of these networks. The NMI scores between each pair of partitions of these networks produced by the algorithm are shown in the image plots in Figure 6. The four plots in this figure correspond to the four selected networks. The rows and columns in a plot refer to the partition index, and the intensity of shade of the (i, j)th box refers to the NMI score of Partition i and Partition j. The shades of the boxes in the figure vary from dark (0.3) to white (1). A darker shade indicates that the NMI score between the corresponding partitions is low, whereas a fairer one indicates a higher NMI score.

It can be seen that in the networks POWER-GRID and HEP-THEORY, all the NMI scores are above 0.80, validating the high partition similarity. In EMAIL-ENRON, most of the NMI scores are above 0.75, and in COND-MATTER most of the NMI scores are above 0.60. These results reasonably indicate that the algorithm is stable.

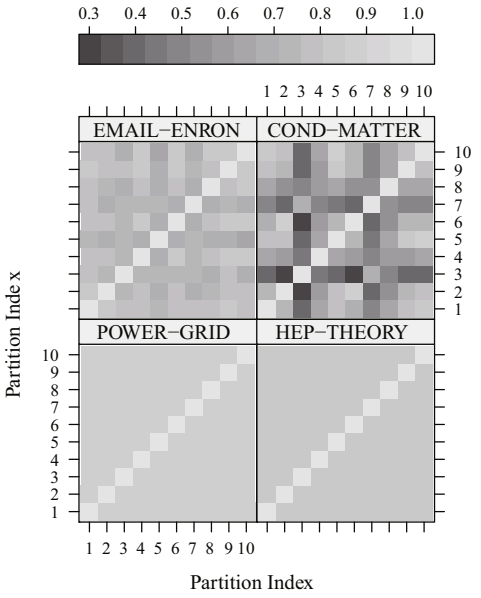


Figure 6 NMI scores between mutual pairs of partitions produced in 10 runs of the algorithm on each of the networks POWER-GRID, HEP-THEORY, EMAIL-ENRON, COND-MATTER

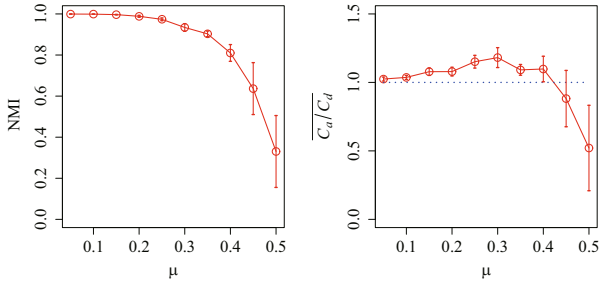
4.3 Accuracy

To assess the accuracy of the algorithm, we use the Lancichinetti-Fortunato-Radicchi (LFR) benchmarks. The benchmark offers a reasonable platform for assessment of community structure detection algorithm. It has a number of parameters, of which the most relevant ones for our study are N — the number of vertices, k — the average degree, $k_{\max}$ — the maximum degree and μ — the mixing parameter. We leave other parameters at their default value. Of special interest is the parameter μ which varies from 0 to 1. When $\mu = 0$, the the communities have no edges between them. As μ increases the edges between communities start falling. When $\mu = 0.5$, the vertices in all communities have roughly as many neighbors inside their own communities as in the rest of the network. For the values of μ in the range 0.5 to 1, the community structure is rather unrealistic.

We assess the accuracy of our algorithm using two parameters. The first one is again the NMI measure. However, this time we compute NMI scores between the so called "actual communities" and the detected communities in LFR as a function of μ. The second parameter is the mean of the quantity C_a/C_d as a function of μ, where C_a is the number of actual communities and C_d is the number of communities detected by our algorithm in LFR.

So, we generated 10 instances of LFR benchmarks for each value of μ in the range $\{0.0, 0.1, 0.2, \cdots, 0.5\}$, with other parameters at $N = 1000, k = 5, k_{\max} = 100$. For each instance we ran our algorithm once. Thus for each value of μ we have 10 values of NMI and C_a/C_d scores. The mean of these scores is plotted as a function of μ in Figure 7. In Figure 7(a), the horizontal axis represents the parameter μ of LFR, and the vertical axis represents the mean NMI scores,

which are plotted as circles. The length of the vertical bar crossing each circle represents the standard deviation in the NMI scores. In Figure 7(b), the horizontal axis is the same as in part (a), but the vertical axis is the mean of C_a/C_d. The length of the vertical bar crossing each circle is the standard deviation between the C_a/C_d scores. We can see that as μ increases NMI scores decreases gradually until $\mu = 0.35$, with low standard deviations. As μ crosses 0.35 there is a rapid decrease in the mean NMI scores with increased standard deviations.



(a) (b)

Figure 7 (a) Mean NMI as a function of μ; (b) Mean C_a/C_d as a function of μ

4.4 Quality Improvement

Let us revisit Equation (9). Let $\mathcal{P}$ be any partition of a graph. For any $C \in \mathcal{P}$, we can write

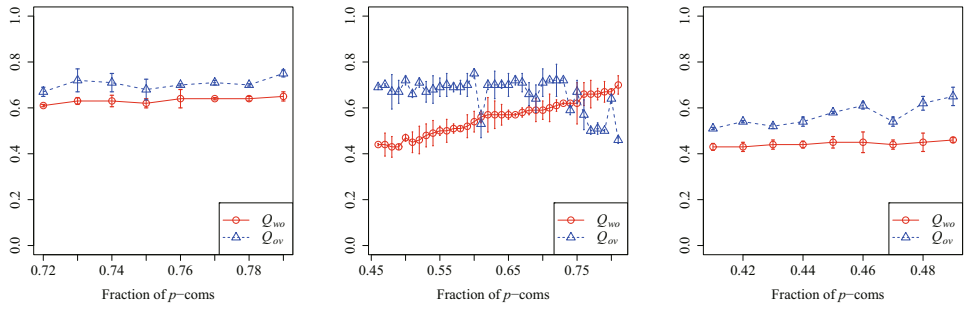
$$\frac{d^{\text{int}}(C)}{d(C)} = s(C) + \frac{d^{\text{ext}}(C)}{d(C)}.$$

So, if C is a p-community, the right hand side in the equation above is greater than or equal to p as $d^{\text{ext}}(C)/d(C) \geq 0$. Thus in order to make the quantity $\frac{1}{|\mathcal{P}|}\sum_{C\in\mathcal{P}} \frac{d^{\text{int}}(C)}{d(C)}$ large, $\mathcal{P}$ must contain a large number of p-communities, with high values of p.

To validate this observation, we select the networks HEP-THEORY, ASTRO-PHYSICS, INTERNET, EMAIL-ENRON, COND-MATTER, and LOC-BRIGHTKITE. On each of these networks we run our algorithm 100 times. Thus for each network, we got 100 partitions. Then we sorted these 100 partitions in the ascending order of the fraction of p-communities (correct upto 2 decimal places) they contain. If any two partitions have the same fraction of p-communities, they are grouped together. For each group then the mean scores of Q_{wo} and Q_{ov} is computed along with the standard deviations. The results are shown in Figure 8. For comparison of the results, we also select another algorithm called SLPA[37] which has been used extensively in the last decade to detect disjoint as well as the overlapping community structure in complex networks. So, we have repeated the above procedure for SLPA with its parameter $r = 0.45$ and restricting it to detect only disjoint community structure, the results of which are given in Figure 9.

Let us first look at Figure 8, where the horizontal axis represents the Fraction of p-communities obtained in the partitions. The vertical axis represents the mean scores of Q_{wo} and Q_{ov}, marked by circles and triangles, respectively. The length of the vertical line on each point (circle or triangle) shows the standard deviation in the scores.

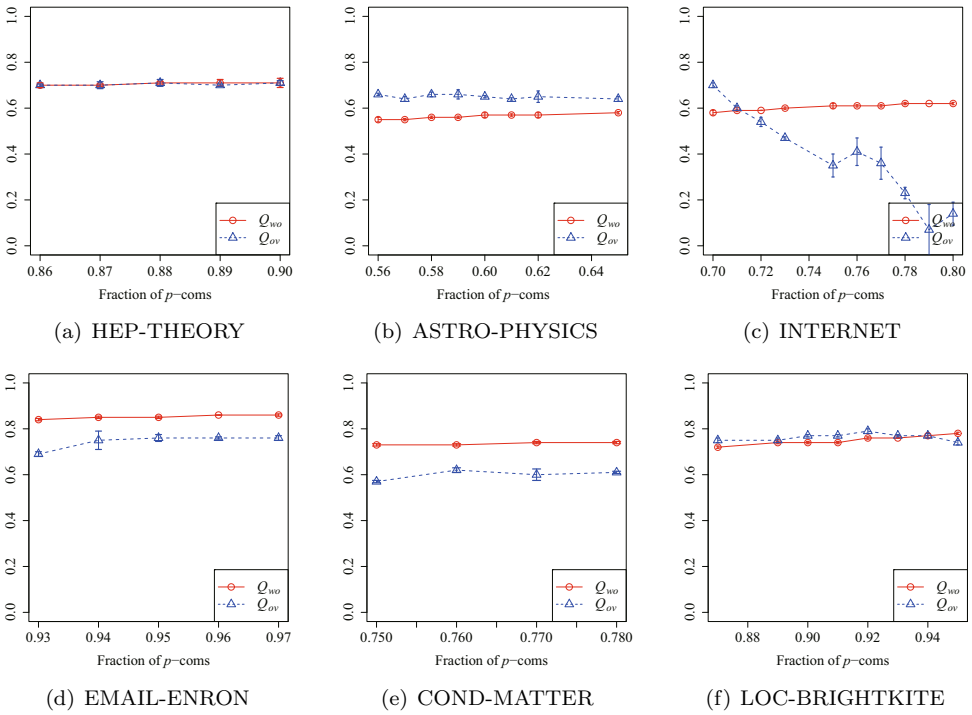
We can see that Q_{wo} increases with small deviations, as the fraction of p-communities increases in the network partitions. However, Q_{ov} does not show the same behaviour in all the networks. In particular, its behaviour in COND-MATTER seems strange (Figure 8(e)). The Q_{ov} score increases for some time, and then start decreasing as the partitions contain more and more p-communities. Now look at Figure 9. Here also x and y-axes are the same as in Figure 8. In this figure too, we get the same observation except for Figure 9(c) in which Q_{ov} decreases as the Fraction of p-communities increases. This somewhat contrasting behaviour of Q_{ov} can be attributed to the notion it is derived from. Its formulation depends on an arbitrary choice of the function f, whose behaviour is not well understood. On the other hand, the formulation of Q_{wo} is based on the simple assumption that a partition is good if it contains a large number of communities with high internal degrees. Thus treating Q_{wo} as a reference of quality, the performance of our algorithm can be improves when the partition contains large number of p-communities. It means that to get high quality partition it is sufficient to run our algorithm a number times, say 10, and keep the partition that contains the largest fraction of p-communities.



(a) HEP-THEORY (b) ASTRO-PHYSICS (c) INTERNET

(d) EMAIL-ENRON (e) COND-MATTER (f) LOC-BRIGHTKITE

Figure 8 Effect of increase in the fraction of p-communities on the partition quality



(a) HEP-THEORY (b) ASTRO-PHYSICS (c) INTERNET

(d) EMAIL-ENRON (e) COND-MATTER (f) LOC-BRIGHTKITE

Figure 9 Effect of increase in the fraction of p-communities on the partition quality

4.5 Running Time of the Algorithm

Running time analysis of an algorithm shows how its run time scales as the input network size increases. For this task, again we choose the real world networks listed in Table 1. The runtime (in seconds) of our algorithm are shown in Figure 10. It can be seen that on the network LOC-BRIGHTKITE, which contains 58, 228 nodes and 2, 14, 078 edges, the algorithm takes roughly one minute on a machine with 4 processors and 8GB RAM.

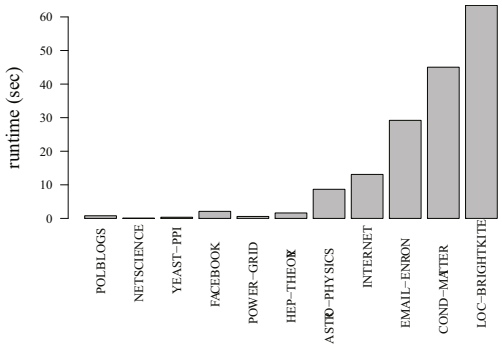


Figure 10 Running time plot (in seconds) of our algorithm on real networks

5 Conclusion

In this article, we have presented a theoretical and computational framework for community structure detection in complex networks. Theoretically, we have addressed the fundamental questions such as when a subgraph is called a community?, what is a community structure?, is a community structure all about communities, or it includes a wider class of subgraphs? In this connection we have proposed the concepts such as strength of a subgraph, p-community, and local maximal p-community, hubs, and outliers. We have defined that a community structure includes local maximal p-communities. However, as we have seen there are practical difficulties in checking whether a p-community is a local maximal p-community or not. So, to detect disjoint community structures in a network we have restricted our attention to the partitions that just contain p-communities, hubs and outliers.

Accordingly, we have developed a local seed based expansion algorithm for disjoint community structure detection in real world networks. For seed expansion we have proposed a new technique called the interaction coefficient, tunable by a parameter α. Presently, the algorithm works only for undirected and unweighted networks. We have analysed our algorithm on a number of parameters and found satisfactory results. Finally, we hope the study in this paper would enrich the knowledge of community structure detection in complex networks.

References

[1] Qi Y and Ge H, Modularity and dynamics of cellular networks, *PLOS Computational Biology*, 2006, **2**(12): e174.

[2] Newman M E J, Detecting community structure in networks, *Eur. Phys. J. B*, 2004, **38**(2): 321–330.

[3] Newman M E J, Communities, modules and large-scale structure in networks, *Nat. Phys.*, 2012, **8**(1): 25–31.

[4] Yang J and Leskovec J, Defining and evaluating network communities based on ground-truth, *Knowl. Inf. Syst.*, 2013, **42**(1): 181–213.

[5] Luce R D and Perry A D, A method of matrix analysis of group structure, *Psychometrika*, 1949, **14**(2): 95–116.

[6] Lawler E L, Cutsets and partitions of hypergraphs, *Networks*, 1973, **3**(3): 275–285.

[7] Alba R D, A graphtheoretic definition of a sociometric clique, *The Journal of Mathematical Sociology*, 1973, **3**(1): 113–126.

[8] Mokken R J, Cliques, clubs and clans, *Quality and Quantity*, 1979, **13**(2): 161–173.

[9] Seidman S B, Network structure and minimum degree, *Social Networks*, 1983, **5**(3): 269–287.

[10] Borgatti S P, Everett M G, and Shirey P R, LS sets, lambda sets and other cohesive subsets, *Social Networks*, 1990, **12**(4): 337–357.

[11] Flake G W, Lawrence S, and Giles C L, Efficient identification of web communities, *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '00, New York, NY, USA, ACM, 2000, 150–160.

[12] Radicchi F, Castellano C, Cecconi F, et al., Defining and identifying communities in networks, *PNAS*, 2004, **101**(9): 2658–2663.

[13] Newman M E J and Girvan M, Finding and evaluating community structure in networks, *Phys. Rev. E*, 2004, **69**(2): 026113.

[14] Fortunato S and Barthlemy M, Resolution limit in community detection, *PNAS*, 2007, **104**(1): 36–41.

[15] Traag V A, Van Dooren P, and Nesterov Y, Narrow scope for resolution-limit-free community detection, *Phys. Rev. E*, 2011, **84**(1): 016114.

[16] Towlson E K, Vrtes P E, Ahnert S E, et al., The rich club of the c. elegans neuronal connectome, *J. Neurosci.*, 2013, **33**(15): 6380–6387.

[17] van den Heuvel M P and Sporns O, Network hubs in the human brain, *Trends Cogn. Sci.* (*Regul. Ed.*), 2013, **17**(12): 683–696.

[18] Sporns O, Structure and function of complex brain networks, *Dialogues Clin. Neurosci.*, 2013, **15**(3): 247–262.

[19] Rombach P, Porter M, Fowler J, et al., Core-periphery structure in networks (revisited), *SIAM Rev.*, 2017, **59**(3): 619–646.

[20] Sporns O and Betzel R F, Modular brain networks, *Annu. Rev. Psychol.*, 2016, **67**: 613–640.

[21] West D B, *Introduction to Graph Theory*, Pearson, 2nd Edition, Prentice Hall, Inc. Upper Saddle River, NJ, 2000.

[22] Kempe D, Kleinberg J, and Tardos E, Maximizing the spread of influence through a social network, *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, New York, NY, USA, ACM, 2003, 137–146.

[23] Nicosia V, Mangioni G, Carchiolo V, et al., Extending the definition of modularity to directed graphs with overlapping communities, *J. Stat. Mech.*, 2009, (3): P03024.

[24] Kumar P and Dohare R, A neighborhood proximity based algorithm for overlapping community structure detection in weighted networks, *Front. Comput. Sci.*, 2019, **13**(6): 1353–1355.

[25] Newman M E J, Finding community structure in networks using the eigenvectors of matrices, *Phys. Rev. E*, 2006, **74**(3): 036104.

[26] Clauset A, Finding local community structure in networks, *Phys. Rev. E*, 2005, **72**(2): 026132.

[27] Lancichinetti A, Fortunato S, and Kertsz J, Detecting the overlapping and hierarchical community structure in complex networks, *New J. Phys.*, 2009, **11**(3): 033015.

[28] Adamic L A and Glance N, The political blogosphere and the 2004 U.S. election: Divided they blog, *Proceedings of the 3rd International Workshop on Link Discovery*, LinkKDD '05, New York, NY, USA, ACM, 2005, 36–43. DOI: 10.1145/1134271.1134277.

[29] Bu D B, Zhao Y, Cai L, et al., Topological structure analysis of the proteinprotein interaction network in budding yeast, *Nucleic Acids. Res.*, 2003, **31**(9): 2443–2450.

[30] Leskovec J and Krevl A, SNAP Datasets: Stanford large network dataset collection, http://snap.stanford.edu/data, 2014.

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

[32] Newman M E J, The structure of scientific collaboration networks, *PNAS*, 2001, **98**(2): 404–409.

[33] Newman M E J, Network datasets from newman, http://www-personal.umich.edu/mejn/netdata/.

[34] Leskovec J, Lang K J, Dasgupta A, et al., Community structure in large networks: Natural

cluster sizes and the absence of large well-defined clusters, *Internet Mathematics*, 2009, **6**(1): 29–123.

[35] Cho E, Myers S A, and Leskovec J, Friendship and mobility: User movement in location-based social networks, *Proceedings of the* 17*th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '11, New York, NY, USA, ACM, 2011, 1082–1090.

[36] Danon L, Daz-Guilera A, Duch J, et al., Comparing community structure identification, *J. Stat. Mech.*, 2005, (9): P09008.

[37] Xie J R, Szymanski B K, and Liu X M, SLPA: Uncovering overlapping communities in social networks via a speaker-listener interaction dynamic process, *IEEE*, 2011, 344–349.

Appendix

Proof [Proof of Theorem 2.7] From Equation (2), we have

$$s(C_1 \cup C_2) = \frac{(1+p)(1+q)2k + (1+p)qd_2 + (1+q)pd_1}{(1+p)d_2 + (1+q)d_1}.$$

Without loss of generality, we can assume that $\max\{p, q\} = p$. Then

$$\begin{aligned}
s(C_1 \cup C_2) > p &\iff \frac{(1+p)(1+q)2k + (1+p)qd_2 + (1+q)pd_1}{(1+p)d_2 + (1+q)d_1} > p \\
&\iff (1+p)(1+q)2k + qd_2 + pqd_2 + pd_1 + pqd_1 > pd_2 + p^2d_2 + pd_1 + pqd_1 \\
&\iff 2(1+p)(1+q)k > (1+p)(p-q)d_2 \\
&\iff k > \frac{1}{2}\left(\frac{p-q}{1+q}\right)d_2 \\
&\implies k > 0 \\
&\implies \mathrm{cut}(C_1, C_2) \neq \varnothing.
\end{aligned}$$

The proof is finished. ∎

Proof [Proof of Theorem 2.9] From Equation (2), we can see that

$$\begin{aligned}
s(C_1 \cup C_2) \geq p+q &\iff \frac{(1+p)(1+q)2k + (1+p)qd_2 + (1+q)pd_1}{(1+p)d_2 + (1+q)d_1} \geq p+q \\
&\iff (1+p)(1+q)2k + (1+p)qd_2 + (1+q)pd_1 \geq (p+q)((1+p)d_2 + (1+q)d_1) \\
&\iff (1+p)(1+q)2k \geq p(1+p)d_2 + q(1+q)d_1 \\
&\iff |\mathrm{cut}(C_1, C_2)| \geq \frac{1}{2}\left[\left(\frac{q}{1+p}\right)d_1 + \left(\frac{p}{1+q}\right)d_2\right] \\
&\iff |\mathrm{cut}(C_1, C_2)| \geq \frac{1}{2}\left[\left(\frac{q}{1-p}\right)d^{\mathrm{ext}}(C_1) + \left(\frac{p}{1-q}\right)d^{\mathrm{ext}}(C_2)\right].
\end{aligned}$$

The last equality follows from the observation that

$$d_1 = \left(\frac{1-p}{1+p}\right)d^{\mathrm{ext}}(C_1) \quad \text{and} \quad d_2 = \left(\frac{1-q}{1+q}\right)d^{\mathrm{ext}}(C_2).$$

The proof is finished. ∎