Discrete Mathematics, Algorithms and Applications
Vol. 5, No. 4 (2013) 1350031 (16 pages)
© World Scientific Publishing Company
DOI: 10.1142/S1793830913500316

THE MAXIMUM COMMUNITY PARTITION PROBLEM IN NETWORKS

ZAIXIN LU[*,§], WEILI WU[*,¶], WEIDONG CHEN[†,‖],
JIAOFEI ZHONG[‡,**], YUANJUN BI[*,††], ZHENG GAO[*,‡‡]

[*]*Department of Computer Science*
University of Texas at Dallas
Richardson, TX 75080, USA

[†]*Department of Computer Science*
South China Normal University
Guangzhou, P. R. China

[‡]*Department of Mathematics and Computer Science*
University of Central Missouri
Warrensburg, Missouri 64093, USA
[§]*zaixinlu@utdallas.edu*
[¶]*weiliwu@utdallas.edu*
[‖]*chenwd2007@hotmail.com*
[**]*zhong@ucmo.edu*
[††]*yuanjun.bi@utdallas.edu*
[‡‡]*zheng.gao@utdallas.edu*

Received 15 May 2013
Accepted 5 June 2013
Published 10 October 2013

The community structure detection is an important problem in many areas such as biology network, computer network and social network. The objective of this problem is to analyze the relationships among data via the network topology. In the literature, many works have been done for partitioning a network into communities or clustering data into groups. In this paper, we define a series of conditions for communities and formulate the community detection problem into a combinatorial optimization problem which aims at partitioning a given network into disjoint communities such that all the communities satisfy the community conditions. We show that the maximization version of this problem is $\mathcal{NP}$-hard for general networks under some natural conditions, and we develop a greedy heuristic algorithm for it. We also develop a refine algorithm to improve the modularity score of a community partition, subject to the community conditions.

Keywords: Community partition; $\mathcal{NP}$-hard; optimization.

Mathematics Subject Classification: 05C82

1. Introduction

In many real-world network systems, such as biology network, computer network and social network, the network architectures can be represented as graphs, and the communities are defined as sub-graphs whose vertices are densely connected within the sub-graphs and loosely connected with vertices outside. Because of this characteristic, community structures are widely used in many applications. For example, social networks often include groups based on locations, interests and occupations, and metabolic networks often have communities based on families and functions. These applications motivate great interests of researches from different areas including physics, computer science, biology and sociology. In the last decade, a lot of works have been done on community structure detection (see examples in [15, 11, 12, 14, 17, 18]).

In the literature, most community detection methods are studied in a qualitative way based on some natural community measures. Among all the measures, the modularity score (also called Q measure) which is proposed by Newman is the most popular one [15, 12]. It is designed to evaluate the strength of communities of a network. That is, communities with high modularity score have dense connections among the vertices within each community and sparse connections between vertices in different communities. However, there are some drawbacks of modularity based algorithms. For example, as shown in [14] and [17], communities of small sizes cannot be detected by algorithms based on modularity score. In addition, some communities obtained by such algorithms are not even connected.

Considering the limits of modularity score, some alternative methods are proposed which are based on community definitions, such as the self-reference community definition and the comparative community definition [11, 19]. These two measures are shown to be able to overcome the above drawbacks of modularity based algorithms. In this paper, we define four comparative community conditions, and we study the problem of partitioning a network into disjoint communities such that all the communities satisfy certain community conditions.

Our contributions are as follows:

(1) We consider the community conditions by incorporating the connectivity constraint (i.e., a community is valid if and only if it is connected), which is not considered in the previous works.

(2) We apply community conditions into community partition to overcome the limits of modularity score. Moreover, we develop a greedy heuristic algorithm to partition a network into communities which satisfy the community conditions, and then use these communities as input to obtain a new community partition that not only satisfies the community conditions but also has good modularity score.

(3) In order to get a flexible initial community set, we investigate the Maximum Community Partition Problem (MCPP), and we prove that it is $\mathcal{NP}$-hard to

partition a general network into the maximum number of valid disjoint communities, even considering the proposed community conditions independently.

The rest of this paper is organized as follows. In Sec. 2, we introduce the related works of community partition. In Sec. 3, we give four comparative community conditions for community partition. In Sec. 4, we present our community partition problem according to the comparative conditions as well as two heuristic algorithms. In Sec. 5, we show the hardness of the maximum version of our Community Partition Problem (CPP). In Sec. 6, we conduct experiments on some well-known real-world data sets and also give the performance analysis. In Sec. 7, we conclude this paper.

2. Related Work

In recent years, there have been tremendous interests in finding communities in different kinds of network systems, and many nice works have been done. Most of these works have been devoted to formalizing the intuition that a community is a set of vertices having dense connections.

The first investigation on community detection was done by Weiss *et al.* [3]. For subsequent approaches used for community detection, there are mainly four categories: Hierarchical-based methods [2], Spectral-based methods [4], Density-based methods [1] and Modularity-based methods [12]. Particularly, Newman's notion of modularity has been a very popular measure in the recent community detection literature. It measures the internal connectivity with reference to a randomized null model. In spite of its theoretical foundations and good performance on many real-world data sets, this family of approaches usually has the "resolution limit" problem [11], i.e., they prefer to choose large communities rather than small communities.

To overcome the "resolution limit" problem, some researchers investigate new approaches or measures for community detection. A Qualified Min-Cut Algorithm (QMCA) from the aspect of combinatorial optimization was proposed in [19]. The algorithm selects one vertex randomly each time and finds the least close vertex to it. If these two vertices can form a community, then group these two vertices as a new community. Otherwise, combine them as a single vertex. The process runs iteratively until there is no possible new community. It has been shown that QMCA can overcome the "resolution limit" problem in modularity optimization. However, it ignores the connectivity issue of community. It is clear that disconnected graph is not an optimal result according to modularity score, because if we partition the disconnected components into communities, the modularity score for the entire community set will be higher than the original partition [15]. Therefore, the connectivity issue should be considered when doing community partition.

In [16], Hu defined an attractive force function to partition vertices into communities such that each vertex stays in the community with the largest attractive force. The algorithm is able to detect small communities such as cliques with three or more vertices. Thus it also solves the “resolution limit” problem. However, since the attractive function for each vertex will be updated when a vertex joins or leaves the original community and each vertex always searches the community with the most attractiveness, the algorithm may be unable to converge for some special graphs.

There also exist many other works which view communities from somewhat different perspectives. To learn more about the large body of research in community detection and partition, please see [20–25] for recent works.

3. Comparative Definitions for Communities

In this section, we give four comparative conditions for communities. We focus on community detection in general networks without directions and self loops.

Four Comparative Definitions: Given a graph $G = \langle V, E\rangle$ with adjacent matrix $A\{a_{i,j}\}$ where $a_{i,j} = 1$ if there is an edge between vertices i and j and $a_{i,j} = 0$ otherwise, a vertex partition

$$\mathcal{P} = \{C_1, C_2, \ldots, C_K\}, \quad 1 \leq K \leq |V|,$$

subject to (1) $\bigcup_{k=1}^{K} C_k = V$, (2) $C_k \cap C_t = \emptyset, \forall\, k \neq t$ and (3) $C_k \neq \emptyset, \forall\, k$, is called an *x–valid* community partition of G if $\forall\, k$, the sub-graph $G(C_k)$ is connected and $\mathcal{P}$ satisfies condition x, where x is defined as the following conditions:

$$\textit{Condition } 1\text{:}\ \sum_{\forall\, j \in C_k} A_{i,j} > \sum_{\forall\, j \in V \setminus C_k} A_{i,j}, \quad \forall\, i \in C_k,\ \forall\, k \tag{1}$$

$$\textit{Condition } 2\text{:}\ \sum_{\forall\, j \in C_k} A_{i,j} > \sum_{\forall\, j \in C_t} A_{i,j}, \quad \forall\, i \in C_k,\ \forall\, k \neq t \tag{2}$$

$$\textit{Condition } 3\text{:}\ \sum_{\forall\, i,j \in C_k} A_{i,j} > \sum_{\forall\, i \in C_k, \forall\, j \in V \setminus C_k} A_{i,j}, \quad \forall\, k \tag{3}$$

$$\textit{Condition } 4\text{:}\ \sum_{\forall\, i,j \in C_k} A_{i,j} > \sum_{\forall\, i \in C_k,\ \forall\, j \in C_t} A_{i,j}, \quad \forall\, k \neq t. \tag{4}$$

The first and second conditions check the validity of communities at the vertex level, i.e., check whether the internal degree of each vertex is large enough or not; meanwhile the third and fourth conditions check the validity of communities globally, i.e., check whether the total internal degree of each community is large enough or not. It is easy to see that a *condition* 1*–valid* community partition of any graph G is also a *condition* 2*–valid* community partition of G and a *condition* 3*–valid* community partition of G is also a *condition* 4*–valid* community partition of G. By simple computation, we can also see that a *condition* 1*–valid* community partition of any graph G is also a *condition* 3*–valid* and *condition* 4*–valid* community partition of G. But a *condition* 2*–valid* community partition of G may not be a *condition* 3*–valid* community partition of G.

Generally, using the first condition will result in less but larger communities; using the last condition will result in more but smaller communities; and using the second and third conditions will result in more but smaller communities than the first condition, and less but larger communities than the last one.

4. The Community Partition Problem

Problem Description: Given a graph $G = \langle V, E\rangle$ with adjacent matrix A and one or multiple community conditions, the Community Partition Problem (CPP) is to partition the graph into disjoint communities such that all the communities satisfy the community conditions.

The Community Partition Problem under the first two community conditions has been studied in [16] and [11] respectively, and the maximization version of CPP is proven to be $\mathcal{NP}$-hard for the third condition in [19]. In this section, we present two algorithms to solve CPP. It can partition a graph into valid communities such that each community is connected. Given a graph G and a set of community conditions as the input, our first algorithm, called Community Partition Algorithm (CPA), has the following steps:

(1) Let $\mathcal{P} = (V)$ (there is only one community at the beginning).
(2) For each community C' in $\mathcal{P}$, select a vertex v with the smallest degree randomly from C' as the root of a new community C.
(3) Check each neighbor vertex of C if it can be merged into C to make the partition $(C'\backslash C, C)$ satisfy the community conditions.
(4) If no such neighbor vertex in Step 3 exists, select the one with the smallest difference value between external degree and internal degree. Put it into C and repeat Step 3. If the partition $(C'\backslash C, C)$ is valid, then recursively run the algorithm on all the communities.
(5) If C grows up to the whole graph C', mark v as an invalid vertex and repeat Steps 2 to 5 (invalid vertices will not be considered).
(6) The algorithm terminates when no more new community can be found for all the communities in $\mathcal{P}$.

The objective of our first algorithm is to partition the graph into small valid communities. Each time CPA finds a new community which is the smallest one among all the valid candidates and it stops at a step that no more community can be found. It is worthy to mention that cutting a community from a graph may separate the rest vertices into many disconnected parts. To overcome this problem, in the third step of CPA, we not only check the community conditions but also check the connectivity of the rest vertices. If there exist more than one components, we put all the components except the largest one to the newly found community C and check the community validity again for all the communities.

Our second algorithm takes the result of the first one as input to modify the community partition according to the modularity score and the community conditions. We call it Community Merge Algorithm (CMA), and its working steps are as follows:

(1) Let $\mathcal{P} = (C_1, C_2, \ldots, C_K)$ as a set of communities for graph G.
(2) Find two communities C_i and C_j such that merging them together will result in a better modularity score.
(3) If C_i and C_j are connected, remove C_i and C_j from $\mathcal{P}$ and put $C_i \cup C_j$ into $\mathcal{P}$. Repeat Steps 2 and 3 until no such C_i and C_j exist.

Theorem 1. *Merging any two communities into one will keep the community validity according to community Conditions* 1 *and* 3.

Proof. The proof is simple, because merging communities will only increase the internal degree and decrease or remain the external degree at both vertex and community levels for all the communities. Therefore, Theorem 1 holds. □

According to Theorem 1, merging communities together will not violate the community conditions when considering *Conditions* 1 and 3. Therefore, each time we only have to check whether the merged two communities are connected.

5. Hardness Proof

In order to get a good input for our second algorithm, we want to partition a network into the maximum number of valid disjoint communities. In this section we prove that the Maximum Community Partition Problem (MCPP) is $\mathcal{NP}$-hard for any community condition defined in Sec. 3.

Theorem 2. *The Maximum Community Partition Problem* (*MCPP*) *under Condition* 1 *is* $\mathcal{NP}$*-hard.*

To prove Theorem 2, we first show the following Lemma.

Lemma 1. *For any graph* $G\langle V, E\rangle$ *and a set* $D \subseteq V$ *of* p *vertices, if the induced graph of* D *is a cycle* $(d_1 \rightarrow d_2 \rightarrow \cdots \rightarrow d_p \rightarrow d_1)$ *in which all the even-numbered vertices have no connections with vertices in* $V \backslash D$, *then* D *cannot be separated into two or more communities according to Condition* 1.

Proof. To satisfy *Condition* 1, the internal degree of each vertex has to be greater than its external degree. For any even-numbered vertex $d_i \in D$, since it has no connections with vertices in $V \backslash D$, it must stay with its two neighbor vertices d_{i-1} and d_{i+1} in D. By a simple transfer process, it is easy to verify that all the vertices in D must stay within the same community according to *Condition* 1. □

We next prove Theorem 2.

Proof. To show that MCPP under *Condition* 1 is $\mathcal{NP}$-hard, we reduce the Half Clique (HC) problem to MCPP in polynomial time. The HC problem asks, given a graph G with an even number of vertices, whether there exists a clique of G consisting of exactly half the vertices of G. It can be easily shown that the HC problem is $\mathcal{NP}$-hard by giving a reduction from the Clique problem, which is known to be $\mathcal{NP}$-complete. Let $G\langle V, E\rangle$ and k be the inputs of a Clique problem. If $k \geq \frac{|V|}{2}$, add $2k-|V|$ vertices to G without adding any edges. If $k < \frac{|V|}{2}$, add $|V|-2k$ vertices and completely connect them to all the vertices in V, including the $|V|-2k$ vertices themselves. Now, it is easy to see that the original graph has a k Clique if and only if the new graph has a Half Clique. We next show a polynomial time reduction from the HC problem to MCPP.

Let $G\langle V, E\rangle$ be an instance of the HC problem that has $|V| = n$ vertices, we construct an instance of MCPP as follows.

First, construct a graph G' by copying all the vertices and edges in G.

Second, for all the vertices in G' that are directly copied from G, we call them original vertices. For each original vertex $v \in G'$, we create additional $deg_{G'}(v) + 3$ new vertices and connected them with vertex v, where $deg_{G'}(\cdot)$ denotes the degree of vertex "$\cdot$" in G'.

Third, create a circle $D = (d_1 \to d_2 \to \cdots \to d_{2n} \to d_1)$ that consists of $2n$ vertices. For each vertex $d_i \in D$, if d_i is an even-numbered vertex, we call it a connector. For each nonconnector vertex, connect it with all the original vertices in G'. Therefore, the n nonconnector vertices in D and all the original vertices in G' form a bipartite complete graph.

The input of MCPP consists of G', D and the $n \times (deg_{G'}(v) + 3)$ new vertices. Let G'' denote the entire graph. We next show that G'' can be partitioned into two or more communities if and only if the HC problem is a "yes" instance.

Consider the existence of a Half Clique in G. Then there exists a complete graph of $\frac{n}{2}$ vertices in G'. Let H' denote the graph induced by the $\frac{n}{2}$ vertices as well as their attaching vertices (each original vertex has $deg_{G'}(v) + 3$ distinct attaching vertices). We partition G'' into two communities $(G''\backslash H', H')$. On one hand, for any original vertex $v' \in H'$, the degree of v' in H' is equal to

$$\frac{n}{2} - 1 + deg_{G'}(v') + 3 = \frac{|V|}{2} + deg_{G'}(v') + 2,$$

where $deg_{G'}(v')$ denotes the degree of vertex v' in graph G'. The degree of v' in G'' is equal to

$$n + 2 * deg_G(v') + 3 < 2 * \left(\frac{n}{2} + deg_G(v') + 2\right).$$

Therefore, all the original vertices in H' is valid according to *Condition* 1. On the other hand, for any original vertex $v' \in G''\backslash H'$, the degree of v' in $G''\backslash H'$ is no less than $n + deg_{G'}(v') + 3$. Therefore all the original vertices in $G''\backslash H'$ are valid too according to *Condition* 1. In addition, it is easy to see all the attaching vertices and vertices in D are valid according to *Condition* 1. Thus, $(G''\backslash H', H')$ is a *Condition* 1-valid community partition.

Consider the HC problem is a "no" instance. Then there exists no complete graph of size $\frac{n}{2}$ in G'. Assume, for the sake of contradiction, that there exists a *Condition* 1-valid community partition $(G''\backslash H', H')$ for G''. By Lemma 1, we know that D cannot be separated. Without loss of generality, assume D belongs to H'. According to *Condition* 1, a vertex of degree $n+2$ must stay with at least $\frac{n}{2}+2$ of its neighbor vertices. Therefore, H' must contain at least $\frac{n}{2}$ original vertices. Since G' has no complete graph of size $\frac{n}{2}$, there exists an original vertex $v' \in G''\backslash H'$ of degree $\frac{n}{2} - 2 + deg_{G'}(v') + 3$ or less, where $deg_{G'}(v')$ denotes the degree of v' in G'. The degree of v' in G'' is $n + 2 * deg_G(v') + 3$. Thus, the degree of vertex v' in $G''\backslash H'$ is not greater than $\frac{deg_{G''}(v')}{2}$, which contradicts that $(G''\backslash H', H')$ is a *Condition* 1-valid community partition.

In summary, we prove G has a Half Clique if and only if G'' can be separated into two valid communities according to *Condition* 1. Thus, MCCP under *Condition* 1 is $\mathcal{NP}$-hard. □

Theorem 3. *The Maximum Community Partition Problem* (*MCPP*) *under Condition* 2 *is $\mathcal{NP}$-hard.*

Proof. From the proof of Theorem 2, we can also get that to partition a graph into two communities is $\mathcal{NP}$-hard according to *Condition* 1. When there are only two communities, *Conditions* 1 and 2 are equivalent, which directly implies Theorem 3. □

Theorem 4. [19] *The Maximum Community Partition Problem* (*MCPP*) *under Condition* 3 *is $\mathcal{NP}$-hard.*

Theorem 5. *The Maximum Community Partition Problem* (*MCPP*) *under Condition* 4 *is $\mathcal{NP}$-hard.*

Proof. We prove Theorem 5 by reducing the 3-Dimensional Matching problem to MCCP. For 3 disjoint sets X, Y and Z, let J be a subset of $X \times Y \times Z$. $M \subseteq J$ is 3-Dimensional Matching if for any two distinct triples $(x_1, y_1, z_1) \in M$ and $(x_2, y_2, z_2) \in M$, we have $x_1 \neq x_2$, $y_1 \neq y_2$ and $z_1 \neq z_2$. Given a set J and an integer n, decide whether there exists a 3-Dimensional Matching with $|M| = n$ is $\mathcal{NP}-$hard even if $|X| = |Y| = |Z| = n$.

Given an instance of 3-Dimensional Matching $\mathcal{M}$ with inputs X, Y, Z and J where $|X| = |Y| = |Z| = n$, we construct the input graph of MCCP as follows.

First, construct a vertex set V. For each element $x_i \in X$, create a vertex v_{x_i} in V. For each element $y_i \in Y$, create a vertex v_{y_i} in V. For each element $z_i \in X$, create a vertex v_{z_i} in V.

Second, construct a vertex set T for J. For each triple $t = (x_i, y_j, z_k)$ in J, create a vertex v_t in T, and connect v_t with the corresponding vertices v_{x_i}, v_{y_j} and v_{z_k} in V respectively.

Third, construct a gadget set D, consisting of $|J| - n$ gadgets $d_1, d_2, \ldots, d_{|J|-n}$. Each gadget is a complete graph of 3 vertices. Connect all the vertices in D with all the vertices in T, i.e., construct a bipartite complete graph between D and T.

Fourth, construct a complete graph K_m consisting of m vertices, where $m = \mu(n + |J|)^2$ and μ is a large constant. Let v_1 and v_2 be the randomly selected two vertices in K_m, connect v_1 with all the vertices in $V \cup T \cup D$ and connect v_2 with all the vertices in $T \cup D$.

Let G be the entire graph, we next show that the 3-Dimensional Matching is a "yes" instance if and only if G can be partitioned into $|J| + 1$ valid communities according to *Condition* 4.

If $\mathcal{M}$ is a "yes" instance, i.e., there are n disjoint triples in J. Consider the corresponding n disjoint triple vertices in V and the corresponding n vertices in T. If we combine them into n communities: $J_1, J_2, \ldots, J_n$, each one consists of 4 vertices, then $\forall\, J_k$ $(0 < k \leq n)$, $\sum_{v_i, v_j \in J_k} A_{i,j} = 6$. Combine the remainder $|J| - n$ vertices in T and the $|J| - n$ gadgets in D into $|J| - n$ communities: $J_{n+1}, J_{n+2}, \ldots, J_{|J|}$. Then $\forall\, J_{k'}$ $(n < k' \leq |J|)$, $\sum_{v_i, v_j \in J_{k'}} A_{i,j} = 10$. It is easy to see the m vertices in K_m directly form a valid community. In such a case, we have

$$\forall\, J_i, J_k\ (0 < i < k \leq n), \qquad \sum_{v_{i'} \in J_i, v_{k'} \in J_k} A_{i',j'} = 0 \tag{5}$$

$$\forall\, J_i, J_k\ (0 < i \leq n < k \leq |J|), \qquad \sum_{v_{i'} \in J_i, v_{k'} \in J_k} A_{i',k'} \leq 4 \tag{6}$$

$$\forall\, J_i, J_k\ (n < i < k \leq |J|), \qquad \sum_{v_{i'} \in J_i, v_{k'} \in J_k} A_{i',k'} = 4 \tag{7}$$

$$\forall\, J_i\ (0 < i \leq n), \qquad \sum_{v_{i'} \in J_i, v_{k'} \in K_m} A_{i',k'} = 5 \tag{8}$$

$$\forall\, J_i\ (n < i \leq |J|), \qquad \sum_{v_{i'} \in J_i, v_{k'} \in K_m} A_{i',k'} = 8. \tag{9}$$

Therefore, G can be partitioned into $|J| + 1$ communities.

Conversely, assume $\mathcal{M}$ is a "no" instance. We next show that G cannot be partitioned into $|J| + 1$ communities. We claim that

(1) The complete graph K_m in G cannot be separated into 2 or more communities.
(2) The vertices in V cannot form communities without using vertices in T.
(3) The $|j| - n$ gadgets cannot form communities without using vertices in T.

Note that there are only two vertices v_1 and v_2 in K_m which are connected with vertices in $G \backslash K_m$. Thus, if we partition K_m into more than 3 communities, at least 2 of them are disconnected with vertices in $V \cup T \cup D$. Let K_m^1 and K_m^2 be the two communities. Without loss of generality, assume $|K_m^1| \geq |K_m^2|$, then on one hand we have

$$\sum_{v_i, v_j \in K_m^2} A_{i,j} = |K_m^2|(|K_m^2| - 1).$$

On the other hand we have

$$\begin{aligned}\sum_{v_i \in K_m^2, v_j \in K_m^1} A_{i,j} &= |K_m^1||K_m^2| \geq |K_m^2|^2 \\ &> |K_m^2|(|K_m^2| - 1),\end{aligned}$$

which implies K_m^2 is not a valid community according to *Condition* 4. Thus, we cannot partition K_m into more than 3 communities. It is easy to see that K_m cannot be partitioned into 3 communities neither. Assume, for the sake of contradiction, that K_m can be partitioned into 3 communities: K_m^1, K_m^2 and K_m^3. Assume, $|K_m^1| \geq |K_m^2| \geq |K_m^3|$. Combine K_m^2 with all the vertices in $T \cup V \cup D$ to form a new communities $K_m^{2'}$. Assume J is a large set, we have

$$\begin{aligned}\sum_{v_i, v_j \in K_m^{2'}} A_{i,j} &\leq |K_m^{2'}|(|K_m^{2'}| - 1) + 2(3n + 2|J| + 6(|J| - n)) \\ &\quad + 2(3|J| + 2|J|(|J| - n) + 3(|J| - n)) \\ &< |K_m^{2'}|(|K_m^{2'}| - 1) + 5|J|^2,\end{aligned}$$

in which the term $3n+2|J|+6(|J|-n)$ corresponds to the number of edges between K_m^2 and $V \cup T \cup D$ and the term $3|J| + 2|J|(|J| - n) + 3(|J| - n)$ corresponds to the number of edges in the graph induced by vertices in V, T and D.

Consider the edges between K_m^1 and $K_m^{2'}$, we have

$$\begin{aligned}\sum_{v_i \in K_m^{2'}, v_j \in K_m^1} A_{i,j} &= |K_m^1||K_m^2| \\ &> \sum_{v_i, v_j \in K_m^{2'}} A_{i,j},\end{aligned}$$

in which the last inequality follows from

$$\begin{aligned}&|K_m^1||K_m^2| - |K_m^2|(|K_m^2| - 1) - (5|J|^2) \\ &= (|K_m^1| - |K_m^2|)|K_m^2| - (5|J|^2) \\ &\geq \frac{20}{3}(n + |J|^2 - |K_m^2|)|K_m^2| - (5|J|^2) \\ &> 0.\end{aligned}$$

Thus, K_m^2 is not a valid community according to *Condition* 4. We can say that K_m cannot be partitioned into 3 communities.

By a similar argument, it can be shown that K_m cannot be partitioned into 2 communities. Therefore K_m cannot be separated into 2 or more communities. The second and third claims hold trivially since a community must be a connected graph and K_m cannot be separated. Therefore, we need at least a gadget and a vertex in T to form a community and we can form at most $|J| - n$ communities by using the vertices in T and D. To show G cannot be partitioned into $|J| + 1$ communities, it is sufficient to show that two vertices in V and one vertex in T cannot form a valid community. This is fair since for any such kind of community J', we have $\sum_{v_i \in J', v_k \in K_m} A_{i,k} = 4$ and $\sum_{v_{i'}, v_{k'} \in J'} A_{i',k'} = 4$, which contradicts the definition of community structure according to *Condition* 4.

In summary, we prove $\mathcal{M}$ is a "yes" instance if and only if G can be partitioned into $|J| + 1$ valid communities. The proof of Theorem 5 is complete. □

6. Experiment Result

In this section we analyze the performances of our algorithm in several real networks. We are concerned about the community validity, the partition size, the intra community connectivity and the modularity score in this experiment. Although our algorithm is not a modularity based algorithm, it may help to improve the performance on modularity while combining with other algorithms. This improvement will be discussed at the end of this section.

6.1. *Simulation environments*

The network data sets used in this experiment including Zachary's Karate Club (ZKC) [5], Les Miserables (LM) [6], Word Adjacencies (WA) [13], American College Football (ACF) [7], Dolphin Social Network (DSN) [8] and Politics Books (PB), which can be download at [26], and St. Martin (STM), Sea Grass (SG), Grassland (GL), Ythan (YTH) and Little Rock (LR) can be download at [27]. The Jazz Musician (JM) network is available at [28], and the USAir97 (USA) network is available at [29].

6.2. *Algorithms*

In addition to the proposed algorithm in Sec. 4, we also implement two heuristics Qualified Minimum Cut Algorithm (QMCA) [19] and Attractive Force Algorithm (AFA) [16] for comparison purposes. QMCA was proposed by Zhang *et al.* to solve the Community Partition Problem under *Condition* 3. In order to compare it with our algorithm, we modify it slightly to be able to fit any community conditions. Using "$\geq$" instead of "$>$" in the conditions in Sec. 3, we will have additional four conditions, we call them relaxed conditions. AFA only works on the relaxed version of the *Condition* 2. Thus, we only compare our algorithm with the results of AFA for the relaxed version of the *Condition* 2. In this experiment, we call *Conditions* $1, 2, 3$ and 4 the strongest condition, strong condition, weak condition and weakest condition respectively.

The modified QMCA works as follows. Given a community C, QMCA begins with a set A that contains a randomly selected vertex in $V(C)$. Add to A the most tightly connected vertex from $V(C)\backslash A$ until $A = V(C)$. Let the last two vertices be s and t, respectively. If the two parts corresponding to the minimum $s - t$ cut in C are valid communities, it is a valid cut. Then QMCA shrinks C by merging vertices s and t, and repeat the process to find the minimum valid cut until it is a trivial graph (which contains only one vertex). Such minimum valid cut partitions C into two communities. When there are multiple communities, QMCA runs this process on each communities.

The AFA begins with a randomized partition of graph G. That is, each vertex and its randomly selected half neighbors form a block. Then, it computes the attracting power for each vertex. For any partition of graph G, the attracting power of a vertex is defined as the maximum number of neighbors within any block in the partition. For any vertex u, if the attracting power is from a block rather than u's block, then u will move to that block. The above attracting power-computing and vertex-moving procedure is repeated until no vertex can be moved or the given maximum repeat number is attained. Considering the algorithm cannot converge in some cases, we modify the algorithm slightly. In the modified AFA, once a vertex u moves to a block B, any vertex in block B is not allowed to move out. We next show the simulation results.

6.3. *Simulation Results*

Tables 1 and 2 show the partition size K generated by QMCA, AFA and our algorithm (MCCP) for all the conditions and their relaxed conditions.

From Tables 1 and 2, one notes that the weaker sense results in larger partition sizes. For example in the graph of American College Football, the relaxed strong

Table 1. The partition size K under Relaxed Weakest (RWKST), Weakest (WKST), Relaxed Weak (RW) and Weak (WK) conditions. The number in the parentheses of the QMCA columns is the number of disconnected communities in the partition.

Partition Size		RWKST		WKST		RWK		WK	
Graph	V	MCCP	QMC	MCCP	QMC	MCCP	QMC	MCCP	QMC
ZKC	34	11	8 (2)	7	6 (2)	4	4 (2)	4	4 (1)
STM	45	11	5 (0)	7	5 (0)	3	2 (1)	3	2 (0)
SG	49	13	11 (2)	10	9 (3)	3	2 (0)	3	3 (0)
DSN	62	19	18 (4)	15	13 (4)	7	7 (3)	7	7 (1)
LM	77	15	14 (6)	13	11 (5)	8	8 (4)	8	7 (4)
GL	88	29	26 (8)	23	19 (6)	17	16 (7)	15	14 (4)
PB	105	24	19 (3)	20	16 (6)	8	8 (2)	8	9 (0)
WA	112	37	12 (5)	29	22 (6)	4	2 (2)	4	2 (2)
ACF	115	19	17 (1)	17	16 (0)	10	12 (0)	10	12 (0)
YTH	135	34	11 (6)	25	7 (5)	4	2 (2)	4	2 (2)
LR	183	6	4 (3)	6	4 (3)	4	4 (2)	4	3 (1)
JM	198	22	12 (2)	19	8 (2)	5	4 (1)	5	4 (2)
USA	332	74	28 (7)	44	16 (11)	18	17 (5)	11	11 (6)

Table 2. The partition size K under Relaxed Strong (RSTG), Strong (STG), Relaxed Strongest (RSTGST) and Strongest (STGST) conditions. NA means the algorithm cannot partition the graph under the corresponding condition.

Partition Size		RSTG			STG		RSTGST		STGST	
Graph	V	MCCP	QMC	AF	MCCP	QMC	MCCP	QMC	MCCP	QMC
ZKC	34	6	6	5	2	2	3	3	2	2
STM	45	2	2	2	NA	NA	2	NA	NA	NA
SG	49	2	NA	NA	NA	NA	2	NA	NA	NA
DSN	62	11	5	9	2	NA	4	2	2	NA
LM	77	13	4	12	4	3	7	3	4	2
GL	88	22	21	23	7	4	13	10	5	4
PB	105	11	NA	6	3	NA	NA	4	2	3
WA	112	3	NA	NA	NA	NA	2	NA	NA	NA
ACF	115	14	15	13	10	14	4	7	4	7
YTH	135	2	NA	NA	NA	NA	2	NA	NA	NA
LR	183	2	NA	3	2	NA	2	NA	2	NA
JM	198	2	NA	4	NA	NA	2	NA	NA	NA
USA	332	16	12	19	8	2	12	10	6	2

sense and strong sense form more communities than that resulted by the relaxed weak sense and weak sense respectively, which verifies the conclusion we mentioned in Sec. 3.

In most case, our algorithm outperforms QMCA and AFA in terms of the valid partition size. MCCP can find more latent community structures even under the strongest sense, but QMCA fails. Especially, MCCP performs much better than QMCA on the relaxed weakest sense and weakest sense. It is probably because those two graphs basically use pair of vertices and 3-vertices complete graph as community template, which meet the goal of our algorithm (to cut a qualified piece of vertices as small as possible from the graph).

Another important criterion for communities structure detection is the connectivity. The relaxed strong sense, strong sense, relaxed strongest sense and strongest sense can naturally avoid the connectivity check. A disconnected valid community under those senses can be simply separated into several connected valid communities which still satisfy the definitions. Our algorithm guarantees the intra community connectivity by applying the fragments collection process, but the communities resulted by QMCA are always intra disconnected. Take the graph USAir97 as an example, QMCA detects 28 and 16 communities on the relaxed weakest sense and weakest sense respectively, but 7 of the relaxed weakest communities (respectively 11 of the weakest communities) are disconnected.

One application for the MCCP is that the outputs can be used as the inputs of any clustering algorithm such as the Fast Community Detecting Algorithm (FCDA) [9, 10], which iteratively merges two communities to yield a partition with a better modularity.

The modularity is defined as

$$Q = \sum_{i=1}^{K} \left[\frac{|L(C_i)|}{m} - \left(\frac{Vol(C_i)}{m} \right)^2 \right],$$

where $L(C_i)$ represents the number of links within community C_i, m is the total links of the whole graph G, and $Vol(C_i)$ is the sum of the degrees of each vertex in C_i.

We claim that merging any communities under strongest or weak sense or their relaxed versions will keep the community validity respectively unless they are not connected. It is trivial since merging communities will only add to the intra community degree and do nothing with the external degree either for vertex nor community itself. Therefore, the claim holds and it is free to do agglomerative hierarchical clustering for communities under those conditions.

Table 3 gives the modularity ratio of using MCCP and FCDA.

Note that using the results of MCCP as inputs of the FCDA will not lose much on modularity as shown in Table 3 and it always keep the validity of community. It is interesting that some results are improved by using MCCP pattern. For example, in the Les Miserables network and American College Football network on relaxed strong sense, the modularity scores resulted by a combination of MCCP and FCDA are 5% and 6% better than that resulted by FCDA.

In some cases, for instance the Karate Club and the USAir97 on strongest sense, the modularity score decades in some extent when applying MCPA. This is because the strongest sense is too strict to have enough small communities which reduces the flexibility when applying FCDA to merge them. Actually the Karate Club and the USAir97 can only be separated into two parts under the strongest sense condition and they cannot be merged by FCDA. Generally speaking, a stronger sense will

Table 3. The modularity ratio. The left "F" means the merged community set from the outputs of MCCP is invalid according to the corresponding condition and the right "F" means the merged community set by FCDA is invalid.

Graph	V	RWKST	WKST	RWK	WK	RSTG	STG	RSTGST	STGST
ZKC	34	97%	94%	100%	100%	102% F	35% F	68% F	35% F
STM	45	90%	77%	92%	91%	100% F	NA F	100% F	NA F
SG	49	82%	81%	75%	75%	65% F	NA F	65% F	NA F
DSN	62	91% F	94% F	69% F	90% F	104% F	45% F	96% F	45% F
LM	77	92%	93%	85%	98%	105% F	85% F	87% F	85% F
GL	88	97%	96%	92% F	91% F	F 92%	62% F	78% F	55% F
PB	105	99% F	96% F	72% F	72% F	100% F	99% F	67% F	99% F
WA	112	90%	85%	80% F	82% F	63% F	NA F	60% F	NA F
ACF	115	91%	93%	89%	90%	106% F	104% F	92% F	84% F
YTH	135	98%	100%	77% F	77% F	82% F	NA F	82% F	NA F
LR	183	87%	87%	46%	46%	58% F	43% F	58% F	43% F
JM	198	F 77% F	F 84% F	76% F	76% F	50% F	NA F	50% F	NA F
USA	332	102%	95% F	79% F	79% F	33% F	25% F	33% F	25% F

lead to less, but larger communities and less flexible to tune to meet other criteria while a weaker sense can have more small communities and gain more flexibility to cluster. Therefore, one can select a proper sense when applying MCPA for clustering algorithms.

7. Conclusion

In this paper, we collect a series of unified definitions for community structures. Taking the graph connectivity into consideration, we formulate the community structure detection into a combinatorial optimization problem which aims at identifying as many valid communities as possible for a given network. This maximum community partition problem is proved to be $\mathcal{NP}$-hard under some natural community conditions, and two heuristic algorithms are proposed. The performance of the proposed algorithms are demonstrated through simulation. The simulation result show that our algorithms are better than the existing algorithm QMCA and AFA for community partition on some well-studied real networks. Moreover, the communities of our algorithm can be used as inputs for any agglomerative hierarchical clustering algorithms to improve the clustering quality.

References

[1] S. Mancoridis, B. S. Mitchell and C. Rorres, Using automatic clustering to produce high-level system organizations of source code, in *Proc. of the 6th Int. Workshop on Program Comprehension*, Ischia, Italy, June 24–26, 1998 (IEEE, 1998) pp. 45–53.

[2] B. Bollobás, *Modern Graph Theory*, Series in Graduate Text in Math., Vol. 184 (Springer-Verlag, NY, 1998).

[3] R. S. Weiss and E. Jacobson, A method for the analysis of the structure of complex organizations, *Amer. Socio. Rev.* **20**(6) (1955) 661–668.

[4] R. Kannan, S. Vempala and A. Vetta, On clusterings: Good, bad and spectral, *J. ACM* **51**(3) (2004) 497–515.

[5] W. W. Zachary, An information flow model for conflict and fission in small groups, *J. Anthropological Research* **33** (1977) 452–473.

[6] D. E. Knuth, *The Stanford Graph Base: A Platform for Combinatorial Computing* (Addison-Wesley, Reading, MA, 1993).

[7] M. Girvan and M. E. J. Newman, Community structure in social and biological networks, *Proc. of the Nat. Acad. Sci.* **99** (2002) 7821–7826.

[8] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten and S. M. Dawson, The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations, *Behav. Ecol. Sociobio.* **54** (2003) 396–405.

[9] M. E. J. Newman, Fast algorithm for detecting community structure in networks, *Phys. Rev. E* **69** (2004) 066133.

[10] A. Clauset, M. E. J. Newmana and C. Moore, Finding community structure in very large networks, *Phys. Rev. E* **70** (2004) 066111.

[11] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto and D. Parisi, Defining and identifying communities in networks, *Proc. of the Nat. Acad. Sci.* **101**(9) (2004) 2658–2663.

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

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

[14] S. Fortunato and M. Barth'elemy, Resolution limit in community detection, *Proc. of the Nat. Acad. Sci.* **104**(1) (2007) 36–41.

[15] U. Brandes, D. Delling, M. Gaertler, R. Gorke, R. M. Hoefer, Z. Nikoloski and D. Wagner, On modularity clustering, *IEEE Tran. Knowl. And Data Eng.* **20**(2) (2008) 172–188.

[16] Y. Hu, H. Chen, P. Zhang, M. Li, Z. Di and Y. Fan, Comparative definition of community and corresponding identifying algorithm, *Phys. Rev. E* **78**(2) (2008) 026121.

[17] Z. Li, S. Zhang, R. Wang, X. S. Zhang and L. Chen, Quantitative function for community detection, *Phys. Rev. E* **77**(3) (2008) 036109.

[18] S. Fortunato, Community detection in graphs, *Physics Reports* **486** (2010) 75–174.

[19] X. S. Zhang, Z. Li, R. S. Wang and Y. Wang, A combinatorial model and algorithm for globally searching community structure in complex networks, *J. Combin. Optim.* **23**(4) (2010) 425–442.

[20] M. Gaertler, Clustering, in *Network Analysis: Methodological Foundations*, series in Lect. Notes in Comp. Sci., Vol. 3418, eds. U. Brandes and T. Erlebach (Springer-Verlag, NY, 2005), pp. 178–215.

[21] A. Lancichinetti and S. Fortunato, Community detection algorithms: A comparative analysis, *Phys. Rev. E* **80** (2009) 056117.

[22] S. Schaeffer, Graph clustering, *Comp. Sci. Rev.* **1**(1) (2007) 27–64.

[23] R. Andersen, F. Chung and K. Lang, Local graph partitioning using PageRank vectors, in *Proc. of the 47th Annual IEEE Symp. on Foundations of Comp. Sci.*, Berkeley, USA, October 21–24, 2006 (IEEE, 2006) pp. 475–486.

[24] C. Pizzuti, Community detection in social networks with genetic algorithms, in Proc. of the 10th Annual Conf. on Genetic and Evolutionary Computat., Atlanta, USA, July 12–16, 2008 (ACM, NY, USA, 2008) pp. 1137–1138 .

[25] E. A. Leicht and M. E. J. Newman, Community structure in directed networks, *Phys. Rev. Lett.* **100**(11) (2008) 118703.

[26] M. E. J. Newman, http://www-personal.umich.edu/∼mejn/netdata.

[27] J. Camacho and A. A. Arenas, Universal scaling in food-web structure, http://www.cosinproject.org/.

[28] A. Arenas, http://deim.urv.cat/∼aarenas/data/welcome.htm.

[29] V. Batagelj and A. Mrvar (2006), http://vlado.fmf.uni-lj.si/pub/networks/data/default.htm.