Community Structure, Spectral Analysis, and its generalization

Chenkai Wang

Abstract

Community detection is a fundamental problem in the domain of complex network analysis. It has received great attention, and many community detection methods have been proposed in the last decade. We first give the definition of community and explain its significance in section 1. After that, we introduce some community detection algorithms, especially spectral analysis. Moreover, a generalization spectral analysis is proposed to distinguish a network with three communities.

Keywords: Community Structure, Spectral Analysis

SUSTech Southern University of Science and Technology

Contents

1 A Brief introduction to community and its detection

1.1 Community: Illustration, Definition, and Examples

Social networks are full of easy-to-spot communities, something that scholars have noticed decades ago[7]. The employees of a company are more likely to interact with their coworkers than with employees of other companies. Meanwhile, students in SUSTech tend to communicate with schoolmates rather than those at Shenzhen University. Consequently, workplaces and schools appear as densely interconnected communities within the social network. Communities could also represent circles of friends, a group of individuals who pursue the same hobby together, or individuals living in the same neighborhood.

More precisely, for an unweighted and undirected network $G = (V, E)$ where V and E are the node set and the edge set with $|V| = n$ and $|E| = m$, respectively. A community structure of network G is a partition of the network, denote as $C = \{C_1, C_2, \cdots, C_{\mathcal{K}}\}$, where $C_i \subset V$, $\cup_{i=1}^{\mathcal{K}} C_i = V$ and $C_i \cap C_j = \emptyset$ $(i, j = 1, 2, \cdots, \mathcal{K}$ and $i \neq j)$. With the concept of community, an additional constraint

$$\sum_{i,j=1}^{\mathcal{K}} |\{(u,v) \mid (u,v) \in E, u \in C_i, v \in C_i\}| >> \sum_{i,j=1}^{\mathcal{K}} \left|\left\{(u,v) \mid (u,v) \in E, u \in C_i, v \in C_j, i \neq j\right\}\right|$$

is always attached to the partition. Intuitively, a community is a group of nodes with much more edges connecting to nodes within the community than to nodes from other communities.

Figure 1 gives an example of community structure. To understand the importance of community intuitively, we will provide several examples immediately.

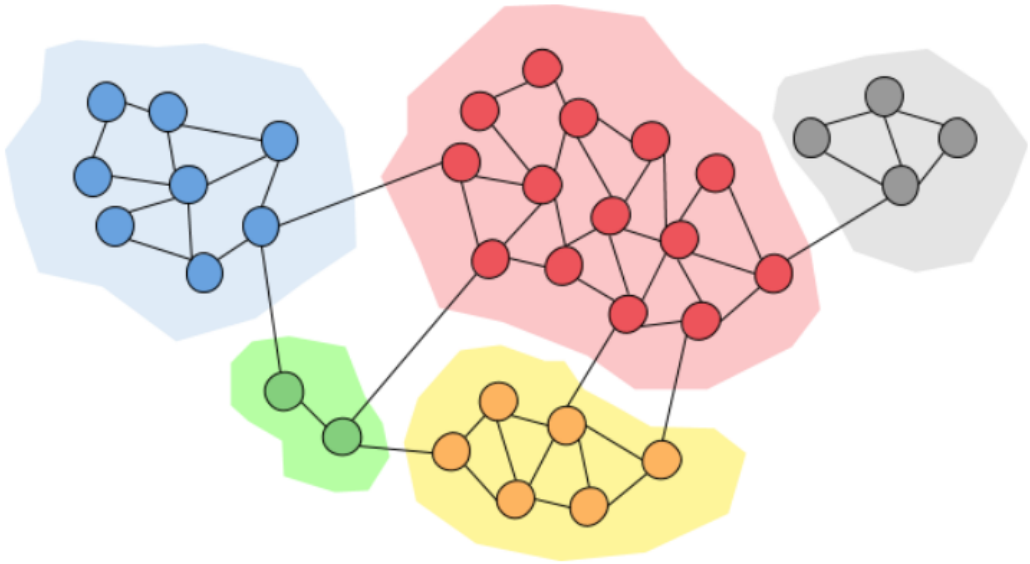

Figure 1: An network with five communities painted in 5 different colors

Communities play a vital role in understanding human diseases. Indeed, proteins involved in the same disease tend to interact with each other[4]. This finding inspired the disease module hypothesis[3], stating that each disease can be linked to a well-defined neighborhood of the cellular network. With these discoveries, we can develop new medicines faster to fight diseases like COVID-19 and keep us better.

Communities are also crucial in the collaboration network of scientists. For beginners in scientific research, through the community, they can quickly find influential scholars to master the general situation of this field better. Scholars in this field can follow the frontier work faster. Moreover, scientists who want to try cross-field can rapidly understand the proximity and development of the target field to make more meaningful work.

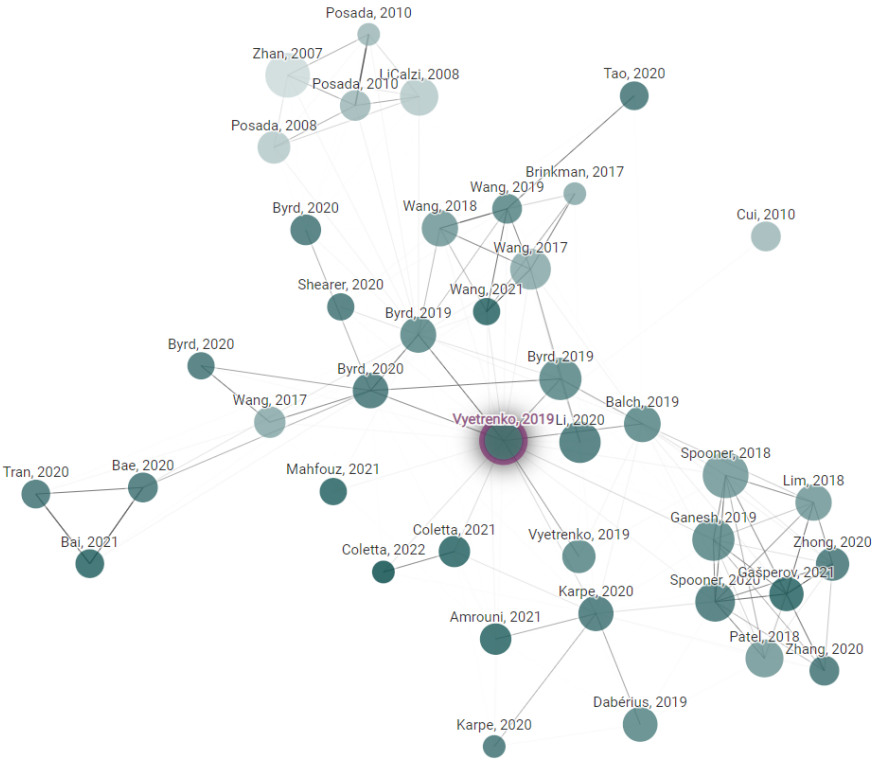

Figure 2: The collaboration network for a specific paper[6] from [1]

A community structure in a network consists of users and their preferences. Internet companies can benefit from such a network by identifying its community structure. For example, the users who have bought basketball and football are divided into two different communities, denoted as A and B, respectively. Then when the company wants to launch an advertisement for a new basketball shoe, the effect of advertising to the users of A will be better than that of club B, which provides a new idea for improving the efficiency and benefit of advertisement and commodity recommendation.

1.2 Community Detection Algorithms

In this section, we will briefly introduce the famous two: the agglomerative algorithm and the divisive algorithm.

The basic idea of the agglomerative algorithm is to calculate the similarity between each pair of nodes in a certain way and then add the edge in each step to the primitive empty network where the number of nodes is n. The number of

edges is 0 based on the descending order of the similarity. This process can stop at any step, and the community structure is obtained simultaneously. A tree can also represent the whole process from empty to origin, as shown in figure 3. The green circles at the bottom represent the nodes in the network. As the blue horizontal dashed line moves up from the bottom of the tree, the nodes merge into a larger community. When the line moves to the top, the network becomes one community. The line at any point in the tree corresponds to a community structure. In the figure, nodes belonging to the same community are circled by a black ellipse according to the location of the line.

Oppositely, the divisive algorithm tries to find the edge with minimum similarity and remove it in each step. Repeat this process, and the whole network is gradually divided into smaller communities. Similarly, the algorithm can terminate at any step, and the community structure is obtained simultaneously. Similar to the agglomerative algorithm, we can use a tree to represent the division process, which can better describe the continuous process that the whole network is split into several smaller communities. The relationship between the agglomerative algorithm and the divisive algorithm is shown below.

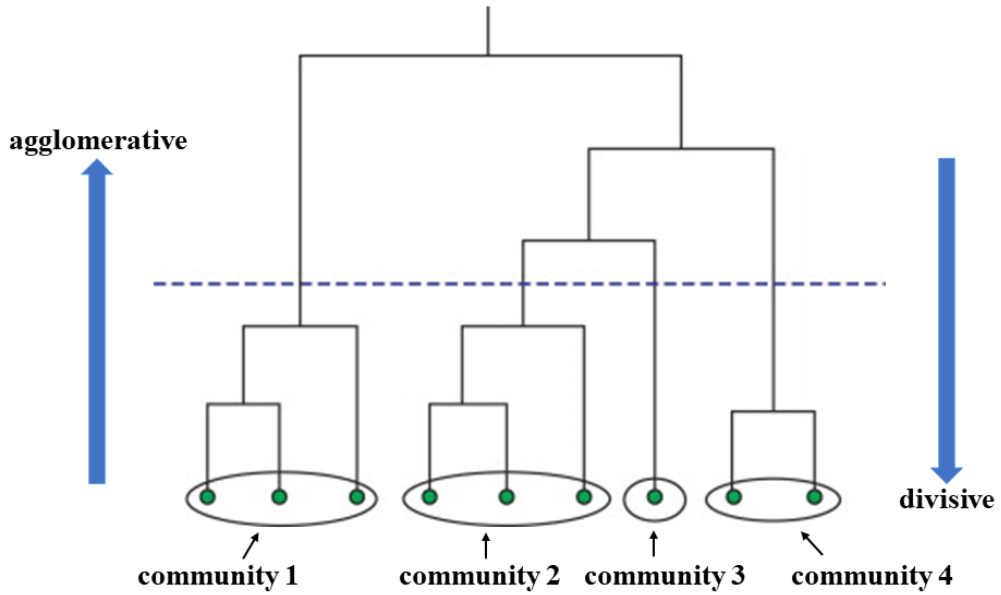

Figure 3: An illustration of agglomerative algorithm and divisive algorithm

In the next section, we will introduce the spectral analysis algorithm, specifically for meeting the requirement and its sound theoretical principles and appliances.

2 spectral analysis for networks with two communities

2.1 Notations and Definitions

For community detection, the spectral methods utilize the eigenspectra of various types of the network-associated matrix to identify the community structure. Before stating the algorithm formally, we first introduce some notations in the below chart.

notation	meaning
A	adjacency matrix
D	diagonal matrix that D_{ii} = degree of node i
L	Laplacian matrix which equals $D - A$
S	column vector records nodes information
k_{ij}	number of edges of node i in community j

Table 1: some notations and the corresponding meanings

The notation above will be illustrated by an example from figure 4.

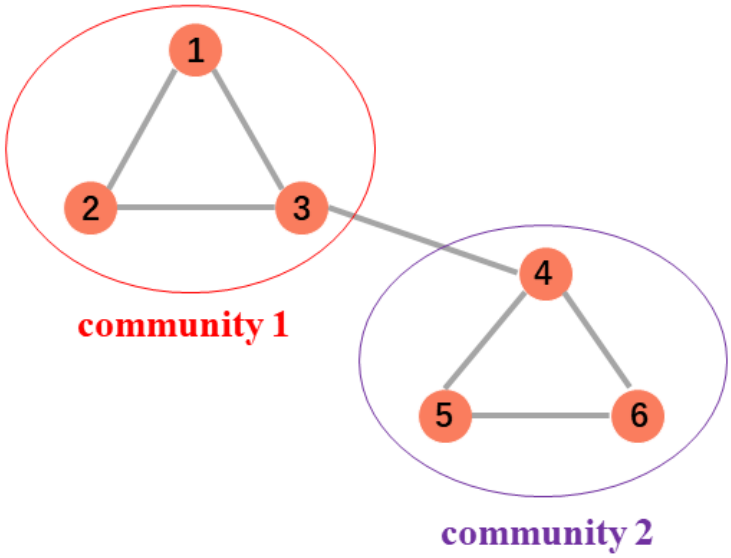

Figure 4: An illustration network with two communities

For the network shown above, we have

$$S = [1, 1, 1, -1, -1, -1]^T$$

where T represents transpose.

Meanwhile,

$$k_{11} = k_{21} = k_{31} = 2, \sum_{i=1}^{3} k_{i1} = 6 \text{ and } k_{42} = k_{52} = k_{62} = 2, \sum_{i=4}^{6} k_{i2} = 6$$

Similarity we have

$$k_{12} = k_{22} = 0, k_{32} = 1, \sum_{i=1}^{3} k_{i2} = 1 \text{ and } k_{41} = 1, k_{51} = k_{61} = 0, \sum_{i=4}^{6} k_{i2} = 1$$

Moreover,

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} D = \begin{bmatrix} 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 2 \end{bmatrix}$$

and $L = D - A$. In the above formulas, red and purple indicates that they record the information within community 1 and community 2 in figure 4, respectively. In addition, brown records the edges between communities.

2.2 A General Instruction Case

Now consider a general case with $|V| = n$, $\mathcal{K} = 2$. Also, there are m and $n - m$ nodes in community 1 and community 2, respectively. Suppose we are already familiar with the community structure. Namely, S is undisputed where

$$S = [1, \cdots, 1, -1, \cdots, -1]$$

We are interested in $S^T LS$, which is

$$S^T LS = S^T (D - A)S = S^T DS - S^T AS$$

After simplification,

$$S^T DS = (\sum_{i=1}^{m} k_{i1} + \sum_{i=m+1}^{n} k_{i2}) + (\sum_{i=1}^{m} k_{i2} + \sum_{i=m+1}^{n} k_{i1})$$

Similarly,

$$-S^TAS = -(\sum_{i=1}^{m} k_{i1} + \sum_{i=m+1}^{n} k_{i2}) + (\sum_{i=1}^{m} k_{i2} + \sum_{i=m+1}^{n} k_{i1})$$

Therefore, $S^TLS = 2 * \left(\sum_{i=1}^{m} k_{i2} + \sum_{i=m+1}^{n} k_{i1}\right)$, which is four times the number of edges between two communities. Then According to our definition of community, finding the best partition is equivalent to minimizing S^TLS.

2.3 Spectral Analysis

This process is only applicable when there are two communities in the network. If satisfies, we can minimize S^TLS to distinguish them. However, we can only write S precisely once we work on the community structure. There is no need to find S. We start from the properties of L, the Laplacian matrix.

Consider a network $G = (V, E)$ with $V = n$ and $\mathcal{K} = 2$, where $\mathcal{K} = 2$ comes from the prior knowledge or our observation. Here we do not add any constraint to $|E|$. Then L corresponding to G has the following properties we care about:

- $0 = \lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_n$ where λ_i represents eigenvalue of L,
- $< v_i^T, v_i >= 1, < v_i^T, v_j >= 0$ for $i \neq j$, $i, j = 1, \cdots, n$ where $< \bullet, \bullet >$ and v_i represents inner product of vectors and eigenvector of λ_i, respectively.

Then

- $\text{span}\{v_1, v_2, \cdots, v_n\} = \mathbb{R}^n$
- $\forall S \in \mathbb{R}^n, S = a_1v_1 + a_2v_2 + \cdots + a_nv_n$

Hence we have

$$\begin{aligned} S^TLS &= (a_1v_1 + a_2v_2 + \cdots + a_nv_n)^T L (a_1v_1 + a_2v_2 + \cdots + a_nv_n) \\ &= (a_1v_1 + a_2v_2 + \cdots + a_nv_n)^T (\lambda_1 a_1v_1 + a_2v_2 + \cdots + a_nv_n) \\ &= (a_1v_1^T + a_2v_2^T + \cdots + a_nv_n^T) (\lambda_1 a_1v_1 + a_2v_2 + \cdots + a_nv_n) \\ &= a_1^2\lambda_1 + a_2^2\lambda_2 + \cdots + a_n^2\lambda_n \end{aligned}$$

where $a_i = v_i^TS$. Then minimizing $S^TLS \approx$ maximizing $a_2^2\lambda_2$ subject to $\sum_{i=1}^{n} a_i^2 = n$.

3 Spectral analysis for networks with three communities

Now consider a network with three communities. An illustration example is shown below.

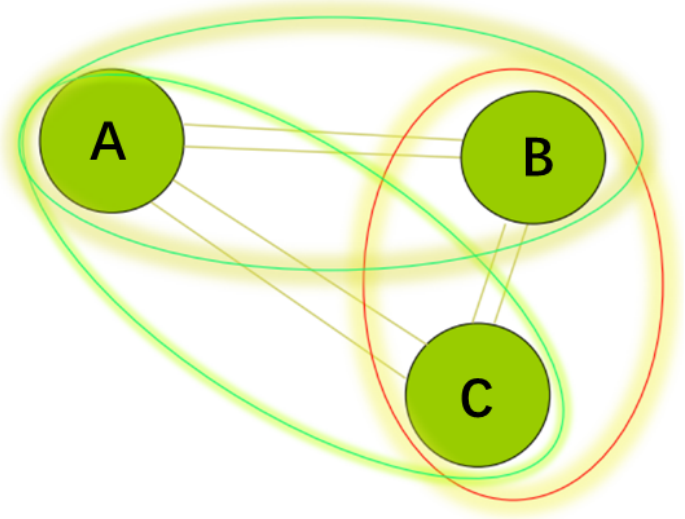

Figure 5: An illustration network with three communities

To detect the community structure similar to figure 5, we can apply the spectral algorithm for three rounds and use model-selecting thinking to derive the final result.

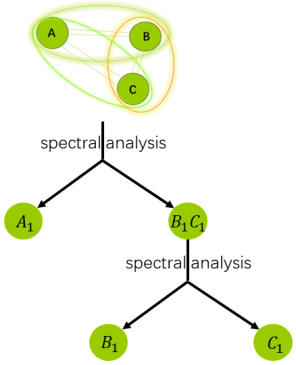

Figure 6: round 1

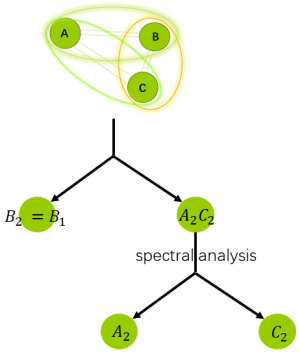

Figure 7: round 2

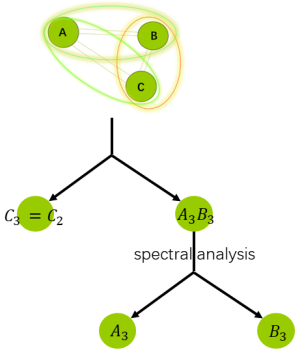

Figure 8: round 3

Here is the description. For graph G, A, B, C are the true communities and A_i, B_i, C_i are communities from round i, $i = 1, 2, 3$. In round 1, we use

spectral analysis twice, the first time for the entire network to get A_1 and B_1C_1 as a whole, the second time for B_1C_1 to get B_1 and C_1. In round 2, we set $B_2 = B_1$ and denote the remaining nodes as A_2C_2. Then we use spectral analysis for A_2C_2 to get A_2 and C_2. In round 3, we set $C_3 = C_2$, and the rest procedure is similar to round 2. Moreover, all A, B, and Cs are sets whose elements are the index of nodes in G. We call the whole process as ***roundabouts***.

The next step is finding the difference. Here we apply the following algorithm.

Algorithm 1 Find the different elements of three sets

Require: : Sets A, B, C
Create S_{same} and $S_{diffent}$
for u in A: **do**
  if u in B and C simultaneously **then**
    add u to S_{same}
  end if
end for
while $A \neq B \neq C$ **do**
  Find u such that $\{u \mid u \in A, u \notin S_{same}\}$ $u \in A$, move such u from A to $S_{different}$
  Find u such that $\{u \mid u \in B, u \notin S_{same}\}$ $u \in A$, move such u from B to $S_{different}$
  Find u such that $\{u \mid u \in C, u \notin S_{same}\}$ $u \in A$, move such u from C to $S_{different}$
end while
Ensure: $S_{same}, S_{different}$

For A_i, B_i, C_i we run the algorithm 1, we will get S_{samei} and $S_{differenti}$, $i = 1, 2, 3$. Define $S_{total\_same} = \{S_{same1} \cap S_{same2} \cap S_{same3}\}$, $S_{total\_different} = V \backslash S_{total\_same}$. Next, we consider arranging the elements in S to attain the best community structure. To achieve this, we use modularity[5] to evaluate the arrangement. We will not present its formula here for brevity. Instead, we will give an intuitive example to explain what it means.

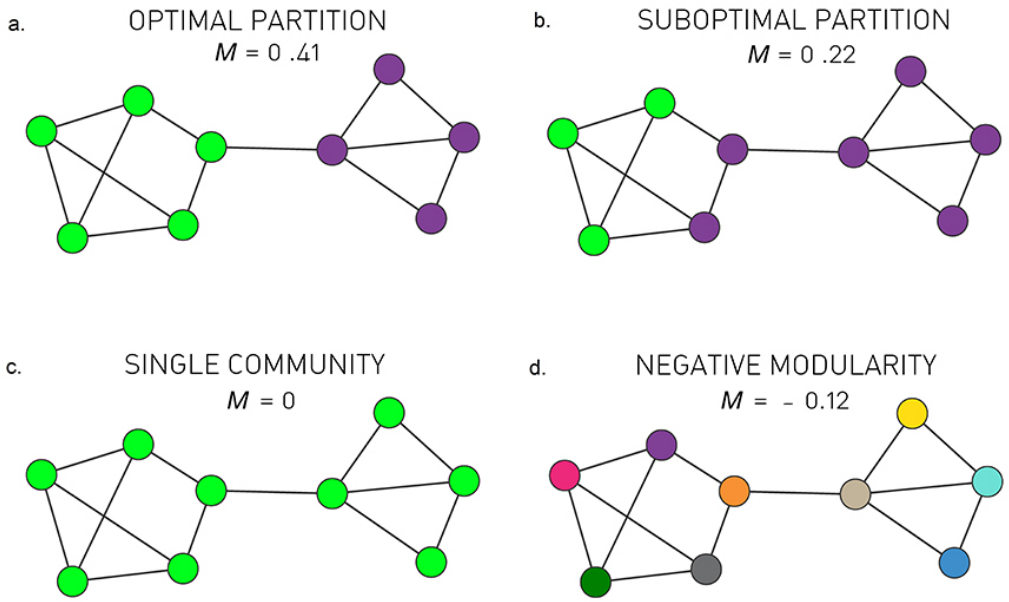

Figure 9: An illustration for modularity from [2]

In short, the better the community is divided, the higher the modularity should be. We use this principle and the idea of greed to deal with S, which is summarized as algorithm 2 presented in Appendix.1. The inputs are

$$A = B = C = S_{total\_same}$$

and

$$S = S_{different\_total}$$

In conclusion, for a network $G = (V, E)$ with $\mathcal{K} = 3$, we can get the community structure via the following steps:

- get A_i, B_i, C_i by ***roundabouts*** method,$i = 1, 2, 3$,
- get S_{total_same} and $S_{total\_different}$ via algorithm 1,
- gets the best partition via algorithm 2.

References

[1] https://www.connectedpapers.com/.

[2] http://networksciencebook.com/chapter/9.

[3] Albert-Laszlo Barabasi, Natali Gulbahce, and Joseph Loscalzo. "Network medicine: a network-based approach to human disease". In: *Nature reviews genetics* 12.1 (2011), pp. 56–68.

[4] Kwang-Il Goh et al. "The human disease network". In: *Proceedings of the National Academy of Sciences* 104.21 (2007), pp. 8685–8690.

[5] Mark EJ Newman. "Spectral methods for community detection and graph partitioning". In: *Physical Review E* 88.4 (2013), p. 042822.

[6] Svitlana Vyetrenko et al. "Get real: Realism metrics for robust limit order book market simulations". In: *Proceedings of the First ACM International Conference on AI in Finance*. 2020, pp. 1–8.

[7] Wayne W Zachary. "An information flow model for conflict and fission in small groups". In: *Journal of anthropological research* 33.4 (1977), pp. 452–473.

Appendix.1

Algorithm 2 Find the best partition

Require: Sets A, B, C, S

Create n, S_{temp}
Create function ***ArrangeNode***(A, B, C, u)
Add u to A and calculate modularity M_{A_u}, then delete u from A
Add u to B and calculate modularity M_{B_u}, then delete u from B
Add u to C and calculate modularity M_{C_u}, then delete u from C
if $max\{M_{A_u}, M_{B_u}, M_{C_u}\} = M_{A_u}$ **then**
    Move u to A
else if $max\{M_{A_u}, M_{B_u}, M_{C_u}\} = M_{B_u}$ **then**
    Move u to B
else if $max\{M_{A_u}, M_{B_u}, M_{C_u}\} = M_{C_u}$ **then**
    Move u to C
else
    Move u to S_{temp}
end if
End ***ArrangeNode***(A, B, C, u)
for u in S: **do**
    run ***ArrangeNode***(A, B, C, u)
end for
$n \leftarrow |S_{temp}|$
for u in S_{temp}: **do**
    run ***ArrangeNode***(A, B, C, u)
    if $|S_{temp}| = n$ **then**
        Generate random number $e \in [0, 1]$
        if $e < \frac{1}{3}$ **then**
            Move u to A
        else if $\frac{1}{3} \leq e < \frac{2}{3}$ **then**
            Move u to B
        else
            Move u to C
        end if
    end if
    $n \leftarrow |S_{temp}|$
end for

Ensure: A, B, C