Building ego-community based on a non-closed neighborhood

Ahmed OULD MOHAMED MOCTAR and Idrissa SARR

Department of Mathematics and Computer Science
Cheikh Anta Diop University
Dakar - Fann BP, 5005, Senegal
amed.mohameden@gmail.com, idrissa.sarr@ucad.edu.sn

RÉSUMÉ. La détection de communautés attire, de plus en plus, l'attention de la communauté scientifique vu son rôle central dans la recherche des caractéristiques des réseaux, de même pour en exploiter les informations. Toutefois, les travaux existants s'intéressent plus au partitionnement des réseaux en plusieurs communautés, appelées souvent "communautés plates". Une autre façon consiste à détecter les communautés à partir des noeuds d'intérêt. Ainsi, nous parlons de "communautés ego-centrées". Elles permettent de comprendre, outre les interactions entre un élément et ses voisins, les interactions qui se passent entre les voisins eux mêmes. Dans un travail précédent, nous avons proposé un algorithme de détection de communautés ego-centrées qui se limite au voisinage direct. Ceci ne reflète pas la réalité de communication, car les amis des amis d'une personne peuvent avoir un impact significatif sur son comportement. L'objectif de ce papier est de proposer une extension de l'algorithme précédent permettant la détection de communautés ego-centrées au-dela du voisinage direct. Une évaluation de la nouvelle solution avec un jeu de données réel montre son gain sur la précédente approche.

ABSTRACT. Community detection arouses, more and more, the interest of scientific community given its central role in research on network characteristics and in the mining of network information. However, most existing works are interested in partitioning networks into communities, called "flat communities". Another way is to detect communities from nodes of interest, such communities are called "ego-communities". They allow to better understand not only the interactions between an element and their neighbors, but also the interactions that are happening between neighbors themselves. In a previous work, we proposed an algorithm for detecting one-step ego-centered communities. That is, the approach is limited only to close neighborhood (*i.e.* nodes directly linked to the *ego*). However such an approach does not reflect the communication real state since friends of a person's friends can impact on its behavior. The purpose of this paper is to extend our previous algorithm by enlarging the neighborhood range. We also implement our algorithm and use adolescent friendship network to demonstrate the effectiveness of our algorithm.

MOTS-CLÉS : détection de communauté, fonction de qualité, communauté ego-centrée

KEYWORDS : community detection, quality function, ego-community

1. Introduction

Social networks analysis has shown benefits of studying and/or analyzing the structural properties of social networks. One of the main topics of social network analysis is community detection, which is a process of decomposing the network into groups that satisfy a set of characteristics. In fact, detecting communities is splitting the network into groups that can be helpful for several use cases. For instance, identifying the group of people who has been in contact (by any communication device) with an Ebola's patient give a huge boost on the target to be monitored or to be informed about the risk they incur.

In short, community detection studies can be divided into two main categories, namely, global community detection (*i.e.*, macroscopic vision of the network) and local community detection (*i.e.*, microscopic vision of some nodes of the network). The first category seeks to partition the network into several communities without any focus on nodes and their role. In contrary, the second category builds communities from a few nodes of the network that we call nodes of interest or "ego". Communities obtained from nodes of interests are named ego-centered communities. Basically, an ego-centered community can be built from an "ego" and its closed neighborhood (alters) as well as it can be defined based on a larger neighborhood (the neighbors of the alters).

In [8], we proposed an algorithm for detecting ego-communities based on closed neighborhood. This paper aims to extend the previous algorithm in order to detect ego-communities based on a larger neighborhood. Moreover, we revised one of the metric, called affinity degree, we used in [8] to select node candidates that form a community. The main contributions of our work can be summarized as follows :

– An improvement of the affinity degree in order to prevent evaluating relevances of nodes that doe not have outgoing links from the ego-community ;

– An algorithm for detecting ego-communities while considering both the alters and their neighbors, the neighbors of their neighbors and so forth. Actually, the algorithm works in two phases :*1*) a selection-phase that helps choosing nodes and *2*) a removing-phase that defines step-by-step the community structure ;

– A validation of our algorithms through experiments conducted over a real data sets that shows the feasibility of our approach and its efficiency.

For this purpose, our paper is organized as follows : firstly, we define, in section 2, the preliminary concepts needed to understand the rest of this article. Then, we present, in section 3, the main existing ego-centered community detection approaches. In Section 4, we present the weakness of the affinity degree that we proposed in a previous paper and we explain how we correct these weaknesses. Section 5 is used for describing the process of building ego-communities beyond direct neighborhood. Finally, we evaluate the efficiency of our solution in section 6 and conclude in section 7.

2. Definitions

In this section, we define the basic concepts needed to fully understand the rest of the article.

Définition 1 (Eccentricity). *Eccentricity of a node u, denoted $e(u)$, designates the number of links used to connect u to the most distant node in network. Otherwise, $e(u)$ repre-*

sents the maximum distance that can exist between u and any other node in network. The eccentricity of a node u is given by the following formula :

$$e(u) = max_{\forall v \in V(G)} d(u, v) \quad [1]$$

Définition 2 (Node of interest)**.** *We call a node of interest or "ego" any node that, by its status, can influence the behavior of other nodes in a given network.*

Définition 3 (k-neighborhood)**.** *We define a k-neighborhood of an "ego" e as all nodes that can be reached with a path length lesser or equal to k. Formally, a node j is in the k-neighborhood of e if $d(e, j) \leq k$. The value of k is between 1 and the eccentricity of u. Ego's neighbors can be linked to each other.*

The figure 1 shows an example. The nodes colored in orange are direct neighbors while pinks are in 2-neighborhood as well as greens are 3-neighborhood. For sake of simplicity, the "ego" with its k-neighborhood is called "ego-network" in the remainder.

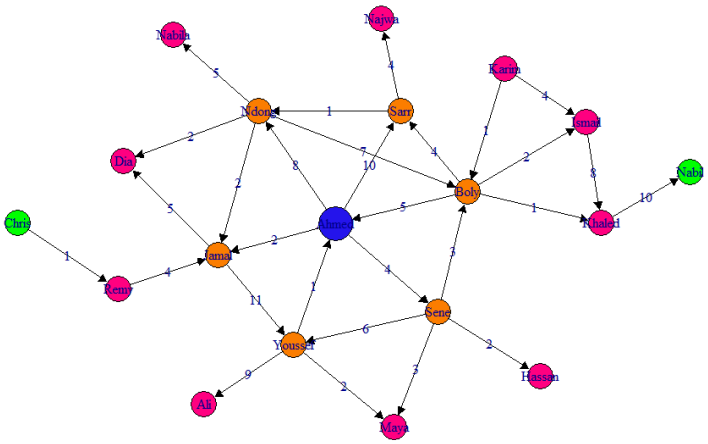


Figure 1 – Illustration of a directed and weighted ego-neighborhood.

Définition 4 (Ego-centered community)**.** *An ego-centered community is a group of nodes made of an ego and its k-neighbors so that the interactions number within the group is predominant compared to the one between group elements and the rest of network.*

3. Related work

To our best knowledge, there is not yet an algorithm for detecting ego-centered community beyond the direct neighborhood. However, we present in this section the existing algorithms for direct ego-community detection. Among the existing works, two approaches have attracted our attention.

3.1. Approach of quality function optimization

Considered as the most used approach [9, 3, 7], it consists in expressing the evaluation criteria of community relevance in the form of metric, called "quality function". The metric associates any community with a quality score. But, since applying a one-time quality

function does not guarantee community relevance, the result of the quality function can be optimized. The goal of this optimization is to minimize/maximize the quality score in order to detect a more relevant community than the original one. There are several ways to express a quality function that depends on how the algorithm defines a community.

The optimization of a quality function is done in three steps. It consists of starting with the node(s) of interest and adding or removing, in a iterative manner, the node that maximizes or minimizes the quality function score until we reach a given threshold.

3.2. Approach based on proximity

Such an approach was introduced recently in [5]. The overall principal is to find out nodes that are enough similar to the “ego" and then gather them as a community. To this end, proximities values are calculated for all nodes with respect to the “ego". Thus, nodes are ranked based on their proximities values in order to figure out whether there exists skew. In other words, if several nodes are almost similar to the “ ego ”, then, they form a stage on the curve of proximities. Rather, nodes that are strictly different follow the stage from afar since they have low values. Once this classification done, nodes of the stage are considered as the community of the “ ego ”. Nonetheless, the proximity curve usually decreases steadily with a non-scale law fashion, which means that a node may belong to several communities of different sizes or does not belong to any one.

Even though there exists plethora of ego-community detection solutions, we can mention a set of drawbacks or limitations as follows :

1) the detection approach considers only the “ego" and its alters ;

2) the topological structure are the only features they use ignoring their intensity for example ;

3) the links direction are not taken into account while computing metrics.

In this paper, we envision a solution facing these limitations.

It should be noted that there are also some supervised classification methods derived from machine learning but often used in community detection. Let’s take for example the k-nearest neighbors algorithms which classify the target elements (nodes) according to their distance from the nodes constituting the learning sample. Thus, these algorithms detect flat communities as they classify the nodes of network in different communities. However, in this paper, we are interested in ego-community detection algorithms.

4. Affinity degree

In a previous paper [8], we have proposed an algorithm for detecting ego-community detection based only on direct neighborhood. This algorithm seeks to optimize a quality function taking into account the weight and orientation of links. To this end, we have also defined a metric that called “affinity degree” allowing to select the most connected node at each iteration of optimization process of the quality function. In this section, we illustrate the weakness of the affinity degree we proposed in [8]. Then, we explain how to correct it.

In fact, the affinity degree $\mathcal{A}_v(\mathcal{C}_u)$ captures how connected is a node v regarding to the community $\mathcal{C}_u$. We evaluate the affinity degree of a given node according to 2 aspects :

– Separation of node from the rest of network : does the node have more links within the community than outside ? Does it communicate more with the nodes of the community or with the nodes located elsewhere ?

– Level of connectivity within the community : how much the node is connected to the nodes of the community ? Does it communicate further with them ?

To handle the first aspect, we propose the criterion $Separation_v(\mathcal{C}_u)$ which allows to measure the separation of a node v from a community C_u :

$$Separation_v(\mathcal{C}_u) = \frac{|N_v \cap \mathcal{C}_u|}{d(v)} \times \frac{d_w^{\mathcal{C}_u}(v)}{d_w(v)} \quad [2]$$

Where :

– $|N_v \cap \mathcal{C}_u|$ is the number of common neighbors between v and $\mathcal{C}_u$;

– $d(u)$ represents the centrality degree of u ;

– $d_w^{\mathcal{C}_u}(u)$ is the sum of weights of all links sharing between the node v and $\mathcal{C}_u$;

– $d_w(u)$ is the weighted degree of u.

Note that $Separation_v(\mathcal{C}_u)$ value's varies between 0 and 1. If $Separation_v(\mathcal{C}_u) = 0$, thus any v's neighbor is in $\mathcal{C}_u$ while $Separation_v(\mathcal{C}_u) = 1$ refers that all neighbors of v are in $\mathcal{C}_u$.

Although $Separation_v(\mathcal{C}_u)$ can classify nodes according to their level of separation from the community, $Separation_v(\mathcal{C}_u)$ does not distinguish between a node having only one neighbor in the community and another one having as many. In fact, this weakness is manifested in the case where the nodes do not have links outside the community, thus, $Separation_v(\mathcal{C}_u)$ considers that all these nodes are at the same level of relevance, which is not necessarily the case. Therefore, we found it necessary to add another criterion allowing to get an idea about the connectivity degree of nodes within the community.

To evaluate the connectivity degree of a given node v to a community C_u, we propose the following formula :

$$Connectivity_v(\mathcal{C}_u) = \frac{1}{d(v) \times d_w(v)} \quad [3]$$

The intuitive idea behind is that the more a node is very connected to C_u (having many neighbors inside and communicating further with them), the more the score of this criterion will be close to 0.

In the worst case, we can find a node having only one link in the community whose the weight equal to 1, which means that $Connectivity_v(\mathcal{C}_u) = 1$. In the best case, the value of $Connectivity_v(\mathcal{C}_u) \simeq 0$.

Now, we define our degree of affinity as follows :

$$\mathcal{A}_v(\mathcal{C}_u) = Separation_v(\mathcal{C}_u) \times (1 - Connectivity_v(\mathcal{C}_u)) \quad [4]$$

In formula 4, we calculate $1 - Connectivity_v(\mathcal{C}_u)$ so that the bounds of $Separation_v(\mathcal{C}_u)$ and $Connectivity_v(\mathcal{C}_u)$ are coherent.

This way of selection has the advantage to define a selection order according to the affinity degree values. The β nodes with the smallest value are selected first. The intuition behind this is that nodes whose affinity degree is small are more likely not to be part of the community. However, this approach is time consuming since we need to estimate the affinity degree of each block of nodes before deciding whether it will be removed or not.

Moreover, the complexity is calculated based on the formula 4. Actually, the first part is computed with a complexity of $4n$, the second part is computed with a complexity of $2n$, which leads to an overall selection cost of $\frac{6n}{\beta}$.

5. Ego-community detection

In this section, we describe the procedure that follows our algorithm to detect k-step ego-centered communities.

5.1. k-neighborhood extraction

To extract the k-neighborhood of a given node u, we use the BFS[1] Algorithm [2]. Our implementation of BFS is to extract neighbors from the ego by step of neighborhood. Otherwise, we extract, first, the set of nodes N_1 that are directly connected to the ego. Next, we extract the set of nodes N_2 that are linked to N_1. Then, we extract the set of nodes N_3 that are linked to N_2. The process of nodes extraction continues until we reach the set of nodes N_k that are linked to the ego through a path length k. We use a vector I for saving the extracted nodes. At each iteration, we save the obtained nodes in I at the condition that they are not already present.

Let's take as example figure 1, the table 1 illustrates the process of 3-neighborhood extraction of node Ahmed. As illustrated on row 7 of table 1, at the second iteration, the neighbors of Sene are Hassan and Maya. Furthermore, Youssef's neighbors are Maya and Ali. But, since Maya is already added to the vector I. Our algorithm adds this time only Ali. The same case occurs between Jamal and Ndong since they have Dia as a common neighbor. Note that the final result is the union of results of all iterations after deleting the repeated neighbors.

First iteration	
Result : Boly, Sene, Youssef, Jamal, Ndong, Sarr	
Second iteration	
Node	*Neighbors*
Boly	Karim, Ismail, Khaled
Sene	Hassan, Maya
Youssef	Maya, Ali
Jamal	Remy, Dia
Ndong	Dia, Nabila
Sarr	Najwa
Result of second iteration : Karim, Ismail, Khaled, Hassan, Maya, Ali, Remy, Dia, Nabila, Najwa	
Third iteration	
Node	*Neighbors*
Khaled	Nabil
Remy	Chris
Result of third iteration : Nabil, Chris	

Table 1 – Illustration of the 3-neighborhood extraction process using BFS algorithm.

1. Breadth-First Search.

The time complexity of BFS can be expressed as $O(n+m)$ such as n is the number of nodes in k-neighborhood and m the number of links.

5.2. Sorting and selecting nodes

After extracting and saving the k-neighborhood of the ego in a vector, we use quick-Sort algorithm's [6] to sort the vector in ascending order according to the affinity degree of nodes. The time complexity of sorting and selecting nodes is $O(n + n\log(n))$. The overall complexity of our algorithm to detect an ego-community from k-neighborhood is $\frac{n}{\alpha}O(2n + n\log(n))$. Such as $\frac{n}{\alpha}$ is the number of iterations done throughout the optimization process of the quality function.

5.3. The Algorithm

Our algorithm initializes the ego community with the k-neighborhood. Then, it selects from the vector I a block of the most useless nodes, namely, those whose affinity degree is the smallest. Next, if the withdrawing of the nodes block from the community reduces more the quality score, they will be removed, otherwise, we leave them in the community and we move on to other ones. This procedure is repeated until the quality score reaches a given threshold.

The block size α is a parameter of our algorithm. In this paper, we are not interested in the impact of block size on the resulting communities.

6. Experimentation

In this section, we aim at assessing the efficiency of the proposed solutions and their feasibility through experimentation. To this end, we implement our algorithms with igraph R package [4] and use an adolescent friendship network [1] containing 2539 nodes and 12969 links. See Fig. 2. It's a directed and weighted network that was created from a study that includes a 90-minute home interview and aims to improve adolescent health in the United States. Each student was asked to list his 5 best female and his 5 male friends. A node represents a student and a link between two students shows that the left student has chosen the right one as a friend. The higher the weight value, the more students interact between them. We mention the existence of 6 big ego-neighborhoods. However, for a purpose of experiment, we focus only on a single ego-neighborhood, namely, the one whose the ego-node label is " 1 " because of its predominance (it's the lowest neighborhood in Fig. 2, colored in purple). Note that in our experiments, we set $k = 2$ that means we build community with a 2-neighborhood at most.

In the remainder of this section, we compare the ego-community algorithm with a larger neighborhood versus ego-community with only alters [8] to prove that the detected ego-communities by the first one are more relevant. Then, we show an example of detected ego-communities of each algorithm to illustrate the difference.

6.1. Algorithms effectiveness

In this part, we aim at evaluating the effectiveness of the previous algorithm we proposed in [8] and that one we proposed in this paper. To this end, we compare them regarding to the 2 criteria (ego-community isolation and separation). Fig. 3a and Fig. 3b show the quality function scores of the 2 algorithms.

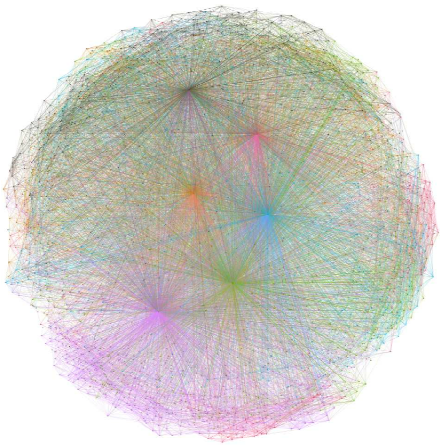


Figure 2 – Visualization of adolescent health network. For readability reasons, we did not display neither the nodes labels nor the links directions.

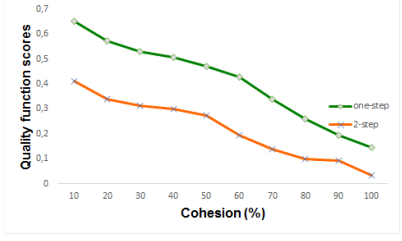
On Fig. 3a, we notice that the more the cohesion increases, the more the quality score decreases, this is explained by the fact that the closer we get to the total cohesion (complete sub-graph), the more the majority of nodes becomes relevant in terms of topological connections. However, we notice that the quality of k-neighborhood algorithm is always better than that of one-step ; we interpret the space difference between the 2 curves on Fig. 3a as follows : as the 2-neighborhood ego-community includes more nodes than 1-neighborhood ego-community, the quality of the first one will always be better by increasing the cohesion.

By analogy, we run another set of experiments in order to evaluate the quality function regarding to the separation of the ego-community.

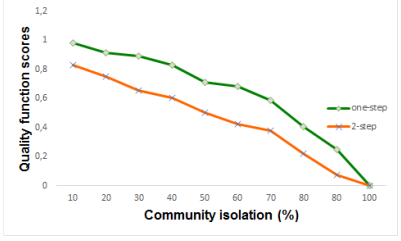
In this respect, we vary the community isolation (its disconnection with the rest of the network) from 10% (*i.e.*, higher communication of the group with the rest of the network) to 100% (total isolation). The community isolation is calculated using the following formula :

$$separation\_rate = \frac{\frac{\mathcal{W}_C^{in}}{|E_C|}}{\frac{\mathcal{W}_C^{in}}{|E_C|} + \frac{\mathcal{W}_C^{out}}{|E_C|}}$$

[h]



(a) Quality functions of 1-neighborhood algorithm vs 2-neighborhood algorithm regarding to the cohesion.



(b) Quality functions of 1-neighborhood algorithm vs 2-neighborhood algorithm regarding to separation of the rest of network.

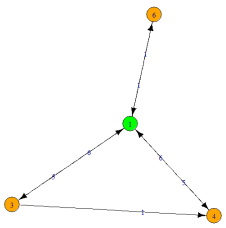
Figure 3 – Quality functions of 1-neighborhood algorithm vs 2-neighborhood algorithm.

Precisely, we divide the intensity of the communication within the group by the intensity of all communication involving one member of the group.

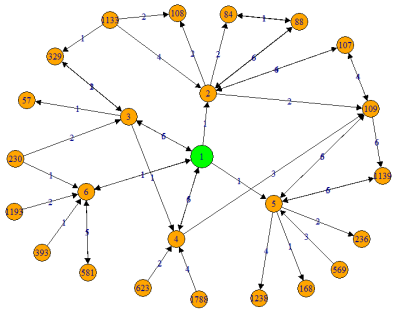
We observe on Fig. 3b three things : firstly, more the communication intensity increases, more the quality scores become small. Secondly, the quality of k-step algorithm is always better than that given by one-step algorithm. Thirdly, we also notice that where the community isolation is maximum, the two algorithms tend towards the same quality, namely 0, because if the isolation reaches 100%, it means that the value of outgoing link weights is zero, thus, the quality score will also be 0.

6.2. Algorithms impact on community structure

To show the pertinence of our k-neighborhood algorithm on the community structure of node whose label is " 1 " of the network depicted on Fig. 2 and extract both the 1-neighborhood and 2-neighborhood ego-communities using algorithm cited in [8] and the one we presented in this paper (i.e. k-neighborhood algorithm).



(a) 1-neighborhood ego-community of ego-node whose label is " 1 ", detected by the algorithm cited in [8].



(b) 2-neighborhood ego-community of ego-node whose label is " 1 ", detected by the k-neighborhood algorithm.

Figure 4 – Detected ego-communities of node " 1 " by one-step and 2-steps algorithms.

As each student lists his 5 best male/female friends, the one-step neighborhood of each node is composed of 6 nodes, including the ego-node, and 15 links, maximally. The minimization process of quality function carried out by the algorithm, cited in [8], led to the rejection of nodes labeled by "2" and "5". Therefore, the one-step ego-community extracted from the neighborhood of ego-node "1" consists only of 3 ego's neighbors, see Fig. 4a. This calls into question the weakness of one-step algorithm [8], because, often, the one-step neighborhood may not be rich of information, that is why, it is necessary to be able to go back towards the top of the hierarchy in order to understand the topological and semantic connections of the node of interest.

On Fig. 4b, we show the ego-community detected by k-neighborhood algorithm. Then, it is clear that with the possibility of broadening the neighborhood depth, we can have an idea about friends of friends of a person, which allows us to, either make a suggestion of friends, or predict a new behavior that the person inspired by its neighbors. Note that for the 2-neighborhood ego-community to be readable, we set the quality threshold ε to 0.4, since if ε is very small, by having a lot of nodes and links, the detected ego-community may be illegible.

To conclude, we proposed the k-neighborhood ego-community detection algorithm in order to cover the weakness of the algorithm cited in [8] which is limited to the direct neighborhood. Experiments have shown that the results provided by k-neighborhood algorithm are more relevant than those of one-step, in terms of internal cohesion and separation from the rest of the network.

7. Conclusion

In this article, we proposed a k-neighborhood ego-community detection algorithm in order to overcome the weakness of the algorithm cited in [8] which is limited to direct one-step neighborhood.

This work can be extended in several ways. For instance, the principle of functioning of the propositions is easily parallelized since the creation of each ego-centered community is made independently of the others. Thus, it is possible to assign a thread to each community building in order to improve the performance of the algorithm. Finally, it is also possible to take into account the detection of multi-level hierarchical communities by recursively applying the algorithm to each community detected.

Références

[1] Adolescent health network dataset – KONECT, September 2016. Available at http://konect.uni-koblenz.de/networks/moreno_health.

[2] Alan Bundy and Lincoln Wallen. Breadth-first search. In *Catalogue of Artificial Intelligence Tools*, pages 13–13. Springer, 1984.

[3] Jean Creusefond, Thomas Largillier, and Sylvain Peyronnet. On the evaluation potential of quality functions in community detection for different contexts. In *International Conference and School on Network Science*, pages 111–125. Springer, 2016.

[4] Gábor Csárdi, Tamás Nepusz, and Edoardo M Airoldi. Statistical network analysis with igraph. 2016.

[5] Maximilien Danisch, Jean-Loup Guillaume, and Bénédicte Le Grand. Multi-ego-centered communities in practice. *Social network analysis and mining*, 4(1) :1–10, 2014.

[6] Charles AR Hoare. Quicksort. *The Computer Journal*, 5(1) :10–16, 1962.

[7] Alexandre Hollocou, Thomas Bonald, and Marc Lelarge. Multiple local community detection. In *Performance Evaluation*, 2017.

[8] Ahmed Ould Mohamed Moctar and Idrissa Sarr. Ego-centered community detection in directed and weighted networks. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, ASONAM '17, pages 1201–1208, New York, NY, USA, 2017. ACM.

[9] Ju Xiang, Tao Hu, Yan Zhang, Ke Hu, Jian-Ming Li, Xiao-Ke Xu, Cui-Cui Liu, and Shi Chen. Local modularity for community detection in complex networks. *Physica A : Statistical Mechanics and its Applications*, 443 :451–459, 2016.