Performance Evaluation of Community Detection Algorithms Based on Relationship Strength Measurement

Soom Behera, Haoye Lu, Amiya Nayak
School of Electrical Engineering & Computer Science, University of Ottawa, Canada
{hlu044, sbehe070, nayak}@uottawa.ca

***Abstract*—The study and analysis of networks have been a significant field of research as it holds its roots in varied disciplines like Biology, Chemistry, Sociology, Computer Applications and many more. Human beings have ventured into the era of a network with the rise of all kinds of networks such as the Internet and Social networks. Detection of community structure in real networks is vital in terms of both theoretical and practical value. Many community detection methods are derived from specific backgrounds and their reliability is still questionable. Researchers hardly focus on the general definition of communities and the general community detection algorithms. The discrepancy between the two might pose some obstacles to the optimization of them so that it is hard to use one algorithm to perfect the other. Recently, Lu et al. [1] have proposed a general method for constructing network model from the real-world problem. Also, they have given a general definition of community structure as well as a complete procedure for detecting communities. In this paper, we apply the efficient resistance distance function [2] on Lu et al.'s algorithm and compare its performance with other community detection algorithms that allow for overlaps.**

I. Introduction

Detection of communities in real-world topologies such as large social networks, web graphs, and biological networks is a problem of considerable practical interest that has received a great deal of introspection [3]. A network community (also sometimes referred to as a module or cluster) is typically thought of as a group of nodes with more and/or better interactions amongst its members than between its members and the remainder of the network [3, 4].

To find such cluster of nodes, one generally picks up a target function that captures the above instinct of a group as an arrangement of nodes with preferable inside integration over the connectivity with the outer network. At that point, following the goal is commonly NP-difficult to optimize exactly [3], one uses heuristics [5] or approximation algorithms [6] to discover such set of nodes that more or less optimizes the target function and that can be comprehended or interpreted as real communities.

However, one might presume communities operationally to be the result of a community detection procedure, owing to the fact that they bear some relationship to the intuition as to what it means for a set of nodes to be a strong community. Although there is no such ubiquitous definition of what a community is, it can be viewed as a set of nodes that has a shared identity or a frequent interaction to each other than others. Modules or communities are intermediate-level structures between the microscopic level of nodes and the macroscopic panorama of the whole network [7]. Once extracted, such clusters of nodes are often interpreted as organizational units in social networks, functional units in biochemical networks, ecological niches in food web networks, or scientific disciplines in citation and collaboration networks [8].

We stress the fact when we find the communities within a certain topology is the division of it in certain sets so that the edges appear within a confined set more often than that across other groups. But, this boils down to an important argument that if the node belongs to the clusters which have an equal number of edges, then which cluster does it belong to?

Usually in the real time scenario, we have a big picture of what a complex network is, but we do not figure out the individual groups, modules or communities underlying it. The objective of the community detection algorithm is to find the modules or communities of a network using only the interactions between its members, which may be a (un)weighted and (un)directed graph. This again raises a question: if the graph partition does the same then what is the need for a community detection method?

For the former method, we are aware of the modules in advance we need to partition the graph into; whereas for the latter case, we are not aware of it. We need to find the number of modules and the partition of the network into modules at the same time. Of course, this leads to different applications. Graph partitioning is used to match a graph to external criteria such as a set of processors. Community detection is used to find the "natural" structure of a graph.

Lu et al. [1] have recently proposed a general characterization of community structures and a procedure for detecting communities. In this paper, we apply the efficient resistance distance function [2] on Lu et al.'s algorithm and compare its performance with other community detection algorithms that allow for overlaps. Specifically, we compare their algorithm with **CFinder** [9], greedy clique expansion (**GCE**) [10], **Link** [11], connected iterative scan (**CIS**), and community overlap propagation algorithm (**COPRA**) [12].

Since we will focus on the algorithms that allow for overlaps, we will use LFR benchmark networks [13] developed by Andrea Lancichinetti, Santo Fortunato and Filippo Radicchi to generate test cases. After that, we will use both community-level and node-level performance measurements to evaluate the performance of the algorithms.

The rest of the paper is organized as follows. Section II gives

the background and some related work in the area of network community detection. Section III gives a brief introduction to Lu et al.'s algorithm and the efficient resistance distance. Further, we discuss on the LFR Benchmark Network and the evaluation metrics (Section IV) and other algorithms for comparison (Section V). We make the comparison in Section VI and finally conclude with some observations in Section VII.

II. Related Work

The community detection methods are always presented from an author's perspective which emphasizes on the speed, performance and the cost of the process. There is no such unique definition of what community is, instead, the idea of communities is different and has been evolving depending on the field that defines it[11]. The reason being we have so many different algorithms existing. So under these scenarios, comparing the performances of the different algorithms which has targeted different fields has less relevance. We would like to refer the interested reader to the more comprehensive and popular reviews and books [3, 14].

In this section, we are going to explain some important milestones of the community detection algorithms from our subjective point of view and also elaborate how these selected algorithms constitute the base of our research plan.

Hierarchical Clustering is one of the traditional methods used to find the communities. Being an agglomerative approach, it starts assuming each node as a different cluster and finds similarities between the every pair of nodes included in that network. Then, the nodes are merged forming the communities ranging from highest to the lowest score. It continues computing similarity with the newly formed group with the rest of the cluster of nodes.There are basically 3 main approaches to finding the similarity between nodes, nodes and clusters and between clusters. The single-linked method uses the highest similarity score of two nodes between two groups, the complete-linked method uses the lowest similarity score of two nodes, and the average-linked method takes the average of all similarities between groups [14]. Eventually, each of these approaches will give us a network of the nested set of communities. Cosine similarity is used to find the similarity between the nodes by not only finding the neighbours common between the two nodes i and j but also normalizing it n_{ij}. It produces values that ranges between 0 and 1. If both nodes have the same neighbours $\theta_{ij} = 1$, and if they have no common neighbours $\theta_{ij} = 0$ [15].

Although there is no best choice between one of the methods (single, average and complete) but it affects the quality of clustering the method is employed. The major drawback of this type of clustering is that the similarity metrics give higher scores to the edges more central to the communities than the edges less central. It results that the node that are connected with a single link to the rest of the network are added later and thus result in the formation of the singleton communities [3, 14].

The divisive algorithms are based on the idea of the between centrality, first proposed by Freeman [16]. Girvan & Newman proposed the popular edge-betweenness metric to find the edges on the boundaries [3]. The algorithm tries to find the edges that are at the boundaries of the communities instead which are at the core of the communities. The betweenness of an edge is measured by the number of the shortest paths that traverse through it. It first counts the number of edges that passes through each edge. It is quite likely that if there is an edge on the boundary of the communities there will be many shortest path routes along it. Thus, they will have a higher betweenness score. If an edge is at the core of a community, it will have a low betweenness value as the shortest paths between pairs of vertices are likely to use other edges. This property distinguishes inter-community edges, which link many vertices in different communities and have high betweenness, from intra-community edges, whose betweenness is low. Eventually, the whole network which is considered as a single community at first results in dendrogram by the deletion of the edges one by one from the highest edge-betweenness value to the lowest one. The leaves of the dendrogram are individual nodes.

The drawback of those algorithms is that it is NP-complete and not suitable for the smaller networks with the lesser number of nodes (≈ 1000). Moreover, no effective greedy algorithm has been developed for this method, which makes this algorithm impractical. Also, when a link is removed, the centralities needs to be recalculated which requires a large amount of computer power and, for a network of size n with m links, the speed of calculating all link betweenness in one step still remains $O(m^2n)$ for unweighted networks [17].

In an extension to the above shortest path centrality, the same authors presented two different approaches to calculating the betweenness centrality. In the first method, the network is looked upon as an electric circuit where each edge between two nodes are assigned a unit resistance and two nodes are selected that we define as unit voltage source and sink. We obtain a measure like that of centrality by calculating the current flow by the Kirchhoff's laws. Those links with the lowest resistance (shortest path) carry the most current and, therefore, are the most central. The same approach is used by Wu-Huberman which is discussed further in this section. While the second approach employs random walks to calculate the betweenness centrality of the links. The network is used as a substrate for signals that perform a random walk between pairs of nodes. The link betweenness, in this case, is simply the rate of flow of random walkers through a particular link summed over all pairs of vertices. The drawback of these approaches is that they require higher computation and do not improve the accuracy when compared to the earlier centrality method.

When a network is partitioned into different communities, the problem is looked upon as an evaluation of how good the partition is. Girvan and Newman [3] proposed a simple approach. Considering an arbitrary partition of a given network into N_c communities. We can define a $N_c \times N_c$, size matrix where the elements represent the fraction of total links starting at a node in partition i and ending at a node in partition j. Then, the sum of any row of $a_i = \Sigma_j e_{ij}$, corresponding to the fraction of links connected to i.

The expected value of the fraction of the links can be calculated by within partitions can be estimated. The probability

that a link begins at a start of node i say a_i multiplied by the fraction of links that end at a node in j say a_j. The expected number of community links is given as $a_i \times a_j$.The real fraction of links exclusively within a partition is e_{ij}. Comparing the two and summing over all the partitions in the graph, we get

$$Q = \sum_{i=1}^{c} (e_{ij} - a_{ij}) \quad (1)$$

This is known as modularity function. To illustrate with an example, if we have two partitions, corresponding exactly to the two components, modularity will have a value of 1. As we had discussed earlier the drawbacks of hierarchical clustering being NP-complete and no effective greedy algorithm has been found out yet. So, Newman considered this approach as one of the standalone community detection methods as it was able to limit those drawbacks [3].

A greedy algorithm is used to optimize the value of Q. Starting from a configuration where each node corresponds to one community, the authors computed all the changes in modularity obtained by joining any possible pair of nodes. The highest increment is selected and the two communities are joined, and the process is repeated until a maximum value of Q is obtained. This method used is really fast and the recalculation of the increments only uses local information. It can analyze a network in almost linear time. However, the accuracy achieved is the lowest of all the modularity optimizing methods. The resolution limit is also the drawback of this methods. It works well for the modules which are similar in size and degree [15].

Another approach introduced the idea overlapping communities. It says that a particular node can belong to various "thematic" communities (i.e. one can belong to a scientific group, a family, a sports team), which usually share a certain amount of nodes, a common scenario. The methodology to find the overlapped communities is based on the concept of 'k-clique communities'. We discuss here maximal cliques as we have used in our algorithm to find the maximal communities and would like the readers to know about it. A Clique in an undirected graph $G = (V, E)$ is a subset of the vertex set $C \subseteq V$, such that for every two vertices in C, there exists an edge connecting the two. Maximum Clique is a clique of the largest possible size in a given graph and cannot be extended by including one more adjacent vertex. A k-clique is a group of nodes that is a complete subgraph, and a 'k-clique community' is the union of all k-clique that are adjacent (two k-cliques are adjacent if they share $k-1$ nodes [17]. It has certainly interesting and useful applications, i.e., it can be used to observe the level of relationship between communities or to determine the communities where a certain node belongs.

Approximate resistance networks although a similar concept founded by Newman but Wu et al. presented an extension of this resistance approach method to reduce the time complexity of it [18]. The algorithm identifies communities based on the properties of resistor networks [18]. It is a method for partitioning graphs in two sections, similar to that of spectral bisection, where it partitions in an arbitrary number of communities, obtained by repeated iterations. The graph is transformed into a resistor network where each edge has a unit resistance. A unit potential difference is set between two randomly chosen vertices named as source and sink.

The underlying idea is that if there is a visible gap between voltage values for vertices at the borders between the clusters, then it makes two separate communities. The voltages are calculated by solving Kirchhoff's equations, an exact solution would be time-consuming, but it is possible to find a reasonably good approximation in a linear time for a sparse graph with a distinct community structure, so the more time-consuming part of the algorithm is the sorting of the voltage values, which takes time $O(nlogn)$ [18]. Any possible vertex pair can be chosen to set the initial potential difference, so the procedure should be repeated for all possible vertex pairs. The authors showed that this is not essential and that a certain number of sampling pairs is sufficient to get desired results, so the algorithm scales as $O(nlogn)$ and is very fast.

An interesting feature of the method is that it can quickly find the natural community of any vertex, without determining the complete partition of the graph. For that, one uses the vertex as the source voltage and places the sink at an arbitrary vertex. The same feature is present in an older algorithm by Flake et al., where one uses max-flow instead of current flow [19].

The algorithm proposed by Orponen and Schaeler [20] is based on the same principle, but it does not need the specification of target sources as it is based on diffusion in an unbounded medium. The limitation of such methods is the fact that one has to input the number of clusters, which the user is not aware of before. The accuracy of this method is dependent on how many times the iterative step is repeated and we are not sure of the maximum limit the step should be followed. It is also dependent on having a good idea of the sizes of communities, which make it difficult to use it in large networks. However, this is one of the few methods that is able to identify the community around one node in linear time.

III. Lu et al.'s Community Detection Algorithm and Efficient Resistance Function

Lu et al. [1] proposed a general algorithm to detect community structures in networks. They assume that the relation strength between any pair of nodes is always metrizable so that they can define a special function, named relation strength measurement (**RSM**) from any pair of nodes to nonnegative real numbers.

Based on RSM, they can find all the maximal communities in a network such that the relation strength (computed by RSM) between any pair of nodes in a single community is greater than some positive threshold named community parameter(**CP**)

The authors claim that there are some other measurements discovered by others satisfying the definition of RSM and they also provide how their algorithm works when the efficient resistance function (**ERF**) [2] is applied.

In this paper, we will simply use ERF as the RSM of Lu et al.'s algorithm.

IV. LFR Benchmark Network and Performance Metrics

When any algorithm detects communities, it must be evaluated based upon how accurately it detects it. For that, we need to define some performance parameters first to measure its performance.

A. LFR Benchmark Network and Normalised Mutual Information

Benchmark networks are used to be the test cases for evaluations of community detection algorithms. For each benchmark network test case, we know all the community structures a priori. In this paper, we use LFR benchmark network which is developed by Lancichinetti, Fortunato and Radicchi [13]. Lancichinetti et al. use normalized mutual information (**NMI**) (a measure of similarity) to compare the pre-known community structures with the one delivered by the tested algorithm. In our paper, we simply apply NMI for evaluation.

B. Precision, Recall and F-Score

While NMI is community-level performance measurement, we also use precision, recall and F-Score which are node -level performance measurements [21].

Since Lu et al.'s algorithm allows for overlaps (i.e, the nodes in networks can be assigned to more than one community), the commonly used pairwise precision and recall measures for clustering algorithms need to be revised. Here, we use Yang et al.'s idea [22].

Suppose C_{test} is the set of communities given by the benchmark network test case. $C_{solution}$ is the set of communities generated by the algorithm for testing. Then we can define two new node pair sets by

$$O_n := \{\{u, v\} | \exists S \in C_{test}.\ u, v \in S\} \tag{2}$$

$$O_m := \{\{u, v\} | \exists S \in C_{result}.\ u, v \in S\} \tag{3}$$

Note. *We do not care the order of nodes here.*

Then, we can compute the pairwise *precision* and *recall* as follows:

$$\textit{precision} = \frac{|O_n \cap O_m|}{|O_n|},\ \textit{recall} = \frac{|O_n \cap O_m|}{|O_m|} \tag{4}$$

Note that precision is the proportion of the correct detection results to all results given by the tested algorithm, while the recall is the proportion of the correct results generated by the tested algorithm to the solutions of the test case. In other words, precision indicates the correctness of the result and recall shows the solution coverage of the result.

Then, the *F-Score* can be defined which can reveal the quality of the tested algorithm's solution.

$$\textit{F-Score} = 2 \times \frac{precision \times recall}{precision + recall} \tag{5}$$

The ranges of precision, recall and F-Score are all from 0 to 1. Moreover, the higher the values are, the better the solution is.

V. Other Algorithms For Comparison

Clique Percolation Method (**CPM**) is based on the assumption that a network is composed of cliques which overlap with each other. CPM finds overlapping communities by searching for adjacent cliques. As a vertex can be a member of more than one clique, so overlap is possible between communities. The parameter k ($k \sim n$ for n cliques in graph theory) is of utmost importance in finding communities via CPM, Empirically small values of k have shown effective results. An efficient implementation of the CPM method is **CFinder**. CPM is suitable for dense graphs where cliques are present. In case there are few cliques only CPM fails to produce meaningful covers. However, it also fails to terminate in many large social networks.

On the contrary, Greedy Clique Expansion (**GCE**) [10], proposed by Lee et al. in 2010, starts with all maximal cliques as initial communities, and removes communities that are too similar, using a local community quantity measure. Thus, it produces communities in which there exists, for each maximal clique, at least one community that fully contains it.

Ahn, Bagrow and Lehmann [11] proposed the **LINK** based algorithm to detect overlapping communities.The algorithm first calculates the similarity of all edges of the network and assign each edge to its own community. At each step, the method chooses the pair of edges with the largest similarity and merges their respective communities until all edges become a single cluster. The history of clustering process is stored in a dendrogram, and the partition with the largest partition density is chosen as the final result. This algorithm is effective in finding the smaller communities; however, it fails to give a larger scope of the topology.

Connected Iterative Scan (**CIS**) examines each node of the network, adding or removing it if the density of the set is increased as a result. The scans are repeated until the set is locally optimal with respect to a defined density metric. The choice of the seed sets is not predetermined: they can be the nodes or the edges of the network. To ensure the connectivity of the identified communities, a number of scans are repeated for a set until no change of the set occurs. Then the set is declared to be a community [21]. Once a scan is finished, the set's connectivity is examined. If the set consists of multiple connected components, it is replaced by the connected component with the highest density, after which the next scan starts. The major disadvantage of this algorithm is that it finds a large number of highly overlapping communities.

Diffusion-based approaches tackle the problem of community detection using a communication paradigm. They rely on the assumption that information is more efficiently exchanged between nodes of the same community. Therefore, communities can be detected by considering how information is propagated in the network. Community Overlap Propagation Algorithm (**COPRA**) is one of the popular methods followed.

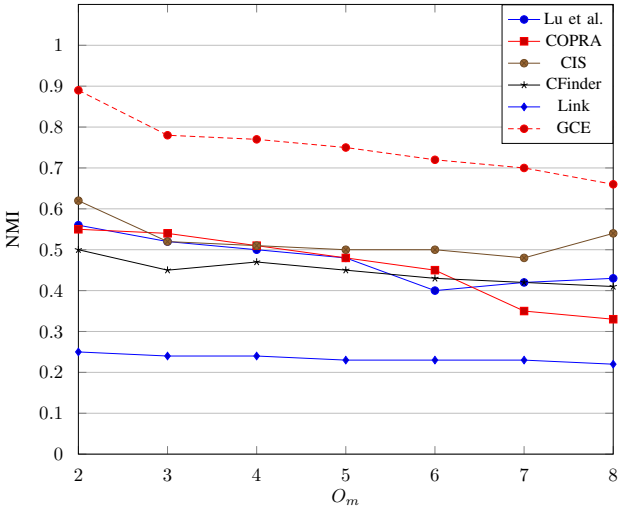

Fig. 1: NMI vs. O_m

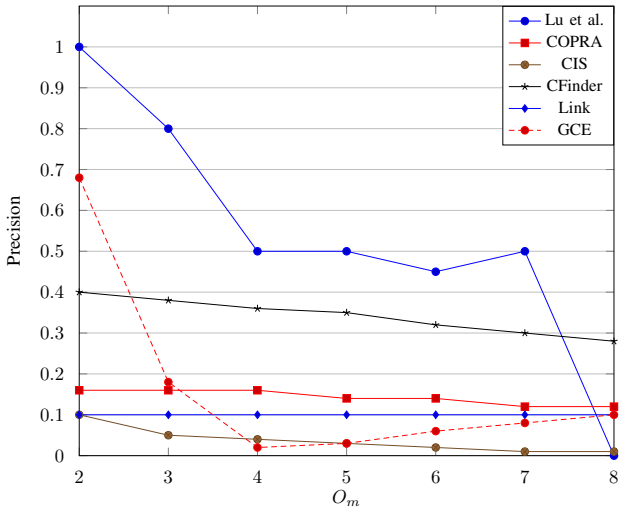

Fig. 2: Precision vs. O_m

COPRA is an extended version of Raghavan et al.'s Label Propagation algorithm, proposed by Gregory [12]. The information takes the form of a label, and the propagation mechanism relies on a vote between neighbours. Initially, each node is labelled with a unique value. Then an iterative process takes place, where each node takes the label which is the most spread in its neighbourhood (ties are broken randomly). This process goes on until convergence, i.e. each node has the majority label of its neighbours. Communities are then obtained by considering groups of nodes with the same label. By construction, one node has more neighbours in its community than in the others. This algorithm is faster than most other algorithms.

VI. Comparison Results

With the help of the metrics we introduced before, we can compare the performance of Lu et al.'s algorithm with others.

We use the similar configuration in Xie et al.'s paper [23]. We generate LFR benchmark networks with size $n = 100$ for each instance. The average degree is kept at $k = 6$. The degree of overlapping is determined by two parameters. O_n defines the number of overlapping nodes and is set to 10% of all nodes. O_m defines the number of communities to which each overlapping node belongs and varies from 2 to 8 indicating the diversity of overlap. By increasing the value of O_m, we create harder detection tasks. Other parameters are as follows: node degrees and community sizes are governed by the power laws with exponents 2 and 1; the maximum degree is 50; the community size varies from 20 to 100; the expected fraction of links of a node connecting it to other communities, called the mixing parameter μ, is set to 0.3. Moreover, since Lu et al.'s algorithm needs CP given by the user, we simply use the CP that gives the best performance here.

We calculated the related parameter values and plotted them with respect to O_m (number of memberships of the overlapping nodes) for Lu et al.'s algorithm and compared them with the other algorithms as shown in the Fig. 1 - Fig. 4. Although the F-score of our algorithm has a negative correlation with the O_m

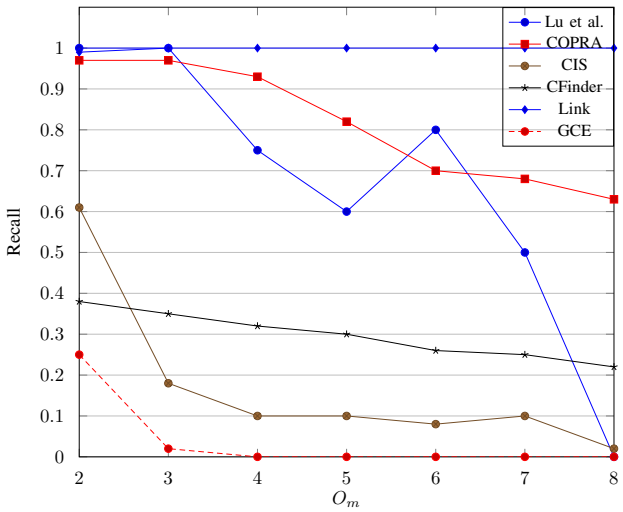

Fig. 3: Recall vs. O_m

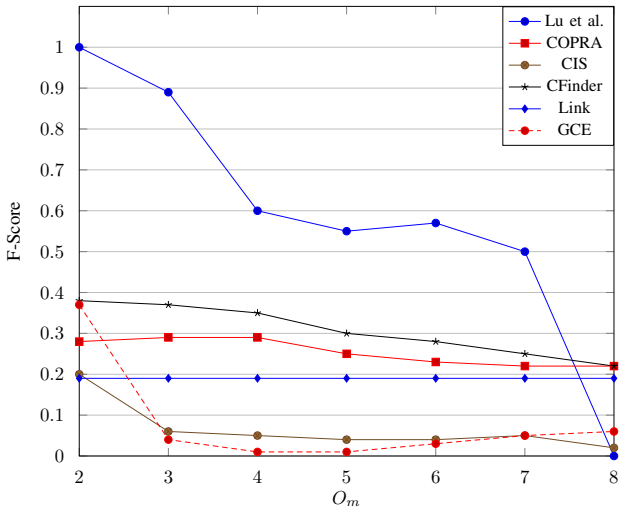

Fig. 4: F-Score vs. O_m

like other algorithms with the increase in O_m but it, however, achieves a reasonable better performance.

Particularly, when the O_m is less than 3, the average F-score of Lu et al.'s algorithm is almost 0.90. This means Lu et al.'s methodology can discover exactly the full local community structure from the given node. Also, the score is reasonably better till when $O_m \leq 7$ which again shows that the algorithm is able to find results with the harder community detection. But beyond that, the proposed algorithm suffers performance degradation and becomes ineffective to detect communities.

The high precision of Lu et al.'s algorithm (also CFinder and GCE for $O_m = 2$) shows that clique-like assumption of communities helps us to identify overlapping nodes. Taking both community level performance (NMI) and node level performance (F-score) into account, we conclude that Lu et al.'s algorithm performs well in the LFR benchmarks.

VII. Conclusion

In this paper, we presented Lu et al.'s algorithm based on *RSM*. We justified how this approach could be effective than other methods when we take into account some of the random arbitrary networks. Experiments on LFR benchmark network show good results. The detection of communities by Lu et al.'s algorithm is solely based on the strength between the nodes. So the users should always give a proper community parameter (CP), which is difficult to find an accurate one.

References

[1] H. Lu and A. Nayak, "A general definition of network communities and the corresponding detection algorithm," *arXiv:1801.07783*, 2018.

[2] D. Klein and M. Randic, "Resistance distance," *Journal of Mathematical Chemistry*, vol. 12, no. 1, pp. 81–95, 1993.

[3] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of the National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[4] F. Radicchi, C. Castellano, F. Cecconi, V. Loreto, and D. Parisi, "Defining and identifying communities in networks," *PNAS*, vol. 101, no. 9, pp. 2658–2663, 2004.

[5] S. Arora, S. Rao, and U. Vazirani, "Expander flows, geometric embeddings and graph partitioning," *J. ACM*, vol. 56, no. 2, pp. 5:1–5:37, Apr. 2009.

[6] D. A. Spielman and S.-H. Teng, "Spectral partitioning works: Planar graphs and finite element meshes," *Linear Algebra and its Applications*, vol. 421, no. 2-3, pp. 284 – 305, 2007, special Issue in honor of Miroslav Fiedler.

[7] M. Rosvall and C. T. Bergstrom, "An information-theoretic framework for resolving community structure in complex networks," *Proceedings of the National Academy of Sciences*, vol. 104, no. 18, pp. 7327–7331, 2007.

[8] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney, "Community structure in large networks: Natural cluster sizes and the absence of large well-defined clusters," *Internet Math.*, vol. 6, no. 1, pp. 29–123, 2009.

[9] B. Adamcsek, G. Palla, I. J. Farkas, I. Derényi, and T. Vicsek, "Cfinder: Locating cliques and overlapping modules in biological networks," *Bioinformatics*, vol. 22, no. 8, pp. 1021–1023, Apr. 2006.

[10] C. Lee, F. Reid, A. McDaid, and N. Hurley, "Detecting highly overlapping community structure by greedy clique expansion," *arXiv:1002.1827*, 2010.

[11] Y.-Y. Ahn, J. P. Bagrow, and S. Lehmann, "Link communities reveal multiscale complexity in networks," *Nature*, vol. 466, no. 7307, pp. 761–764, 08 2010.

[12] S. Gregory, "Finding overlapping communities in networks by label propagation," *New Journal of Physics*, vol. 12, no. 10, p. 103018, 2010.

[13] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E*, vol. 78, p. 046110, Oct 2008.

[14] S. Fortunato, "Community detection in graphs," *Physics Reports*, vol. 486, pp. 75 – 174, 2010.

[15] S. Fortunato and M. Barthelemy, "Resolution limit in community detection," *Proceedings of the National Academy of Sciences*, vol. 104, no. 1, pp. 36–41, 2007.

[16] L. C. Freeman, *The Development of Social Network Analysis: A Study in the Sociology of Science*. Empirical Press, 2004.

[17] I. Derényi, G. Palla, and T. Vicsek, "Clique percolation in random networks," *Phys. Rev. Lett.*, vol. 94, p. 160202, Apr 2005.

[18] F. Wu and B. Huberman, "Finding communities in linear time: a physics approach," *The European Physical Journal B - Condensed Matter and Complex Systems*, vol. 38, no. 2, pp. 331–338, 2004.

[19] G. W. Flake, S. Lawrence, C. L. Giles, and F. M. Coetzee, "Self-organization and identification of web communities," *Computer*, vol. 35, no. 3, pp. 66–71, Mar. 2002.

[20] P. Orponen and S. Schaeffer, "Local clustering of large graphs by approximate fiedler vectors," *Experimental and Efficient Algorithms*, vol. 3503, pp. 524–533, 2005.

[21] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: the state of the art and comparative study," *ACM Computing Surveys*, vol. 45, no. 4, pp. 1–35, 2013.

[22] T. Yang, R. Jin, Y. Chi, and S. Zhu, "Combining link and content for community detection: a discriminative approach," *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 927–936, 2009.

[23] J. Xie and B. Szymanski, "Towards linear time overlapping community detection in social networks," P.-N. Tan, S. Chawla, C. Ho, and J. Bailey, Eds. Springer Berlin Heidelberg, 2012, vol. 7302, pp. 25–36.