Community Detection via Neighborhood Overlap and Spanning Tree Computations

Ketki Kulkarni[1], Aris Pagourtzis[2(✉)], Katerina Potika[1], Petros Potikas[2], and Dora Souliou[2]

[1] San Jose State University,
One Washington Square, San Jose, CA 95192, USA
{ketki.kulkarni,katerina.potika}@sjsu.edu
[2] School of Electrical and Computer Engineering,
National Technical University of Athens, 15780 Zografou, Greece
{pagour,ppotik}@cs.ntua.gr, dsouliou@gmail.com

Abstract. Most social networks of today are populated with several millions of active users, while the most popular of them accommodate way more than one billion. Analyzing such huge complex networks has become particularly demanding in computational terms. A task of paramount importance for understanding the structure of social networks as well as of many other real-world systems is to identify *communities*, that is, sets of nodes that are more densely connected to each other than to other nodes of the network. In this paper we propose two algorithms for community detection in networks, by employing the *neighborhood overlap* metric and appropriate spanning tree computations.

Keywords: Community detection · Neighborhood overlap · Hierarchical clustering · Edge betweenness · Modularity · Social networks · Spanning trees

1 Introduction

Over the last few decades, advances in technology and the rise of the Internet have led to numerous online social networks, like Facebook, Twitter, LinkedIn, and Instagram, where people interact and exchange information at an unprecedented rate forming a plethora of virtual groups, communities and societies. Apart from its own interest, the study and analysis of social networks finds applications on complex networks that appear in various other scientific fields. Scientists working on different disciplines like sociology, computer science, anthropology, psychology, biology, and physics are interested in the discovery of various structural and statistical properties that characterize complex networks [2]. One of the most important problems in analyzing such networks is the detection of

---

The order of authors is alphabetical; each author had an equal contribution to this work.

communities based on observable connections and interaction among users or components of the network. Prediction of human emotions, influence propagation, sentiment analysis, opinion dynamics, protein interaction are some of the ever-expanding fields for which community detection is highly relevant.

In this work we are proposing two new algorithms for community detection in networks that can be represented by unweighted graphs, that is, networks in which only information regarding connections between parts of the network is available. Our algorithms are hierarchical clustering methods, that make novel use of the Neighborhood overlap (nover) metric and spanning tree computations. We compare our algorithms with two related well-known algorithms Louvain and Girvan-Newmann (GN) by performing experiments on random graphs as well as on real-world networks. The contribution of this work is twofold: first, we manage to obtain a fast parallelizable algorithm based on spanning tree computations and second we reveal cases where the use of the nover similarity measure can enhance community detection.

1.1   Community Detection

A community in a network is a collection of nodes that are more densely connected to each other than to nodes outside the community. Detecting communities thus helps us identify nodes with common preferences, properties or behavior, unveil interactions and evaluate relationships among them and often discover hidden information.

Currently, there are quite a few methods and techniques that deal with finding communities. As an example, a lot of techniques identify edges that link different communities. In order to find such edges, various centrality measures, such as node or edge betweenness, are used. Popular approaches attempt to discover a hierarchical structure in a network, and create communities that maximize or minimize some evaluation function. Well-known community detection algorithms are the Girvan-Newman algorithm, which is based on the *edge betweenness* metric [8], the Louvain algorithm [4], and the Label propagation algorithm [14], to name only a few.

Often, edges within the same communities tend to have lower traffic, in case of information or other flow among nodes, a fact that is reflected to smaller edge betweenness (see below) compared to that of edges belonging to different communities. Thus removing edges with high edge betweenness seems a reasonable approach in order to partition the network into communities. While doing so, it is usual to keep track of the quality of the formed partitions using a metric called *modularity*, which is a well established community quality measure. A third measure of interest in this work is the *neighborhood overlap* which reveals the strength of bonds between a pair of nodes in terms of the fraction of neighbors that are common to both. These three measures are described in more detail below.

1.2   Terminology

For a graph $G(V, E)$, which models a network, where V is the set of nodes (users), and E is the set of edges (connections between nodes), we define the following notions and measures.

Edge Betweenness. Edge betweenness (eb) of an edge $e \in E$ defines how important that edge is with respect to shortest paths that connect each pair of nodes in G. More specifically, $\mathsf{eb}(e)$ is defined as the sum, over all pairs of nodes i, j, of the ratio of the number of shortest paths between i and j passing through edge e over the total number of shortest paths between i and j. It is based on the assumption that if much of the traffic of a network passes through an edge (assuming that traffic is routed through shortest paths) then that edge is likely to connect different communities.

Neighborhood Overlap.   The neighborhood overlap (nover) of an edge $e = (u, v)$ is a measure of embeddedness of e defined as the ratio of the number of common neighbors of u and v to the number of nodes that are neighbors of either u or v:

$$\mathsf{nover}(u, v) = \frac{|N_u \cap N_v|}{|N_u \cup N_v|}, \tag{1}$$

where N_u denotes the set of neighbors of node u.

When an edge e is a local bridge, then $\mathsf{nover}(e) = 0$, and edges with very small nover value can be seen as almost local bridges. An edge with low nover score is considered a weak tie and an edge with high nover score is a strong tie. Note that $\mathsf{nover}(u, v)$ is in fact the Jaccard index of the two neighborhood sets N_u, N_v and measures the nodes similarity.

Modularity. One way to measure the quality of the formed community structure is the modularity [13]. Modularity Q is a scalar value, $-1 \leq Q \leq 1$, and it measures the density of the nodes within the same community compared to a random assignment of edges. The larger the modularity score, the better the partitioning of the nodes into communities. It is used to compare the communities obtained by different methods. It is calculated as,

$$Q = \frac{1}{2m} \cdot \sum_{i,j} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \cdot \delta(c_i, c_j) \tag{2}$$

where m is the number of edges, A_{ij} is the weight of an edge between nodes i and j, k_i is the degree of node i, c_i is the community to which node i belongs to, and δ is a function such that $\delta(c, c') = 1$ if $c = c'$ else 0. A modularity value close to 0 or negative indicates low community structure, while a value well above 0 indicates high community structure.

1.3   Related Work

Various approaches have been proposed in recent years to solve the community detection problem. Most popular among these are: (a) optimization methods, which aim to maximize or minimize an objective function, and (b) hierarchical methods that are either divisive or agglomerative.

In the seminal paper of Girvan and Newman [8] they define the eb centrality measure and propose the GN algorithm that uses this measure. GN iteratively removes edges of higher eb centrality, thus forming connected components that correspond to communities. The main disadvantage of this algorithm is that it is computationally expensive (since it recomputes the eb values for all edges in each step) and thus not scalable. The running time in the worst case is $O(|E|^2|V|)$.

A two phase algorithm for weighted graphs was proposed in [4], known as the Louvain algorithm, that runs in $O(|E|)$ time. In the first phase, for each node it iteratively calculates the modularity obtained by including the node to the community of each of its neighbors, and then places this node into the community that gives the highest modularity. In the second phase, it creates a meta-graph in which communities are represented as meta-nodes and self-loops represent edges internal to the communities. The two phases are repeated on the meta-graph. This algorithm has a tendency to overlook small communities. In general, methods that use the modularity metric to optimize the community detection are known to suffer from the *resolution limit* effect [7], which refers to the fact that communities smaller than some threshold may not be discovered. Furthermore, the Louvain algorithm cannot efficiently explore the hierarchical structure of the network (if such a structure is present).

The main idea in [12] is to use the nover score to differentiate weak from strong ties. The nover scores are stored in a minimum heap and all edges with a score smaller than a threshold value are considered weak ties and are removed. The problem with this approach is correctly deciding the threshold value. Larger values of the threshold value could disintegrate the communities and smaller values could make two different communities merge.

Regarding other successful methods for community detection in the setting where overlapping communities are also sought, BigClam [16] should be mentioned, a method that uses matrix factorization in order to discover overlapping and non overlapping communities in large scale networks, and another approach by Ahn, Bagrow and Lehmann [2] that also discovers overlapping communities by partitioning edges instead of nodes. Both these approaches work on the global structure of the network. In a different direction, a hierarchical scalable edge clustering algorithm presented in [9] aims at discovering overlapping local communities of a seed node; a similar approach when the graph is given as a stream is described in [10]. The latter manages to maintain minimal information about the whole graph and the formed communities, thus using space sublinear in the number of edges.

1.4   Our Contribution

In order to propose scalable and parallelizable algorithms that build on the importance of the nover score we introduce two approaches. Our first approach modifies Louvain algorithm by assigning weights equal to their nover score; recall that Louvain is designed to work on weighted graphs. We run several experiments in order to estimate the importance of this modification. The obtained results in most cases show that the modified algorithm performs better, therefore nover values seem worth taking into account.

Our second approach uses the eb centrality in conjunction with the nover metric in order to remove edges and form connected components (communities). The eb metric is also used by the GN algorithm; however, in contrast to the GN approach, we start by computing a maximum spanning tree using the nover weights, thus considerably reducing the eb computations which take place on the tree only. This yields a faster and parallelizable approach. It turns out that the use of a spanning tree together with nover scores helps sparsify a graph without substantial loss of information. In particular, the experimental results show that while the time gain of our ST algorithm compared to GN is important, the modularity values are of the same order of magnitude and in some cases even exceed those of the GN algorithm. Moreover, the high time requirements of the GN algorithm render it inapplicable to larger networks, while ST does not seem to suffer from similar limitations.

2   Neighborhood Overlap-Based Approaches

We consider unweighted, undirected graphs. In both our algorithms, we use the nover score of edges Eq. (1) in order to assign weights to them. Intuitively, we want to differentiate weak from strong ties. We consider strong ties as more probable to connect nodes within the same community. Let us assume $G(V, E)$ is an input graph. The first step of both our algorithms is to calculate the nover score for every edge $e \in E$. Thus, we get a weighted graph.

In Algorithm 1 (nover-Louvain), this preprocessing is the only modification with respect to the original Louvain algorithm. Note that this increases by at most an $O(\Delta)$ factor the time complexity of the algorithm, where Δ is the maximum degree of the network. This is because we need $O(|E|\Delta)$ time for computing the nover of all edges, since computing the common neighbors of an edge can be done in $O(\Delta)$ time. Combining with the $O(|E|)$ complexity of the original Louvain we get a total time complexity of $O(|E|\Delta)$.

In Algorithm 2 (ST), in the first phase we make use of the nover edge weights in order to perform the maximum spanning tree computation. In the second phase, we calculate the eb score of each edge, taken over the constructed spanning tree. We then repeatedly remove one edge at a time, in non-increasing order of their eb score; thus, in each repetition we increase the number of communities by one. Since the optimal number of communities is not known beforehand we keep repeating until all edges are removed; the output is the set of communities C that yields the highest modularity.

Algorithm 1. Modified Louvain community detection by neighborhood overlap (nover-Louvain).

Input: $G(V, E)$
Output: Set of communities C of maximum modularity Q
  for each edge $e = (u, v) \in E$ **do**
    nover$(e) = |N_u \cap N_v| \; / \; |N_u \cup N_v|$
    $w(e) \leftarrow$ nover(e)
  end for
  $C \leftarrow$ Louvain(G, w)
  return C

The time complexity of Algorithm 2 is analyzed as follows:

- $O(|E|\Delta)$, for computing the nover of all edges of the graph, as explained above
- $O(|E| + |V| \log |V|)$, for computing the maximum spanning tree using Prim's algorithm
- $O(|V|)$, for computing the eb values for all edges in the spanning tree
- $O(|V| \log |V|)$ for sorting the eb values of the tree edges
- $O(|E||V|)$, for the *while* loop since $O(|E|)$ suffices in order to compute the communities and modularity in each iteration.

Therefore, an upper bound on the total running time is $O(|E||V|)$. Note that this may be slightly improved by a more tight analysis of the *while* loop, but such an analysis would not improve the worst case bound.

3   Experimental Results

We evaluate our methods using experiments on various datasets and compare them against the most established algorithms, namely the Louvain[1] algorithm [4] and the Girvan-Newman (GN) algorithm [8]. These two algorithms were already implemented in the `python igraph` library so we have chosen to implement our new algorithms nover-Louvain and ST using the same library. We focus on two criteria for the comparison of the algorithms, namely the modularity and the number of communities found.

We have experimented with two types of networks and corresponding datasets: synthetic and real-world data. Regarding synthetic datasets we employed two different random graph models, namely the Barabási-Albert model and the Erdős-Rényi model; as for real-world datasets we considered the Zachary's karate club network, and the Facebook network. Our results are summarized in Tables 1, 2, 3 and are discussed below.

---

[1] Note that the plain Louvain algorithm, can be applied on unweighted graphs by setting all edge weights equal to 1.

Algorithm 2. Community detection by neighborhood overlap and maximum spanning tree (ST).

Input: $G(V, E)$
Output: Set of communities C with maximum modularity Q

    for each edge $e = (u, v) \in E$ **do**
        $\mathsf{nover}(e) = |N_u \cap N_v| \; / \; |N_u \cup N_v|$
        $w(e) \leftarrow \mathsf{nover}(e)$
    end for
    $G'(V, E') \; \leftarrow \;$ calculate Maximum Spanning Tree(G, w)
    for each $e \in E'$ **do**
        $\mathsf{eb}(e) \leftarrow$ calculate Edge Betweenness on e
    end for
    Initialize $C \leftarrow \{V\}$, $Q \leftarrow$ modularity of C in $G(V, E)$ $\triangleright$ one community
    Sort all edges in E' in non-increasing order of $\mathsf{eb}(e)$
    while E' is nonempty **do**
        Remove the edge e of highest $\mathsf{eb}(e)$ from E' $\triangleright$ next edge in sorted list of edges
        $C' \; \leftarrow \;$ community structure implied by E' $\triangleright$ set of components, partitioning V
        $Q' \; \leftarrow \;$ modularity of C' in $G(V, E)$ $\triangleright$ modularity is wrt the original graph
        if $Q' > Q$ **then**
            $Q \; \leftarrow \; Q'$
            $C \; \leftarrow \; C'$
        end if
    end while
    return C

3.1 Synthetic Network Models

Barabási-Albert Graphs. We use the Barabási-Albert [3] synthetic random graph model, which is a well known model for generating random networks. These networks have a power-law distribution property. A power-law distribution property implies that only few nodes in a network have a high degree, a property that is common in social networks and the Internet.

We use different values on parameters for our experiments in order to find out in which cases our new algorithms behave better than the existing ones and when the results are getting worse. The number of nodes varies from 40 to 1000, while the number of edges vary from 114 to 1997. The dataset generator actually does not permit to determine the number of edges but only the minimum degree. The minimum degree value in our experiments is either 2 or 3. Since the graphs resulting from the given parameters are random, the obtained results sometimes vary considerably (see Table 1). We have chosen to present two experiments for each parameter setting, the most extreme ones with respect to the best modularity observed among experiments of the same setting.

A remarkable case is that of graphs with 150 nodes, 297 edges and minimum node degree 2, in which the Louvain algorithm achieves modularity 0.139 for a certain graph and 0.457 for another one, both graphs having been obtained by using the same parameters. The nover-Louvain algorithm achieves, in both graphs, better results; the difference in modularity in the same graphs is again

remarkable (0.189 and 0.459, respectively). For the GN algorithm this difference is even larger compared to the other algorithms. Our ST algorithm on the contrary produces results close to those of Louvain and nover-Louvain; in fact it outperforms Louvain for the first of the two graphs.

In the case of networks consisting of 400 nodes, 1194 edges and minimum degree 2, the results of our new algorithms are encouraging. In almost all cases, nover-Louvain outperforms all other algorithms and the results of ST are comparable to the GN results, which is quite interesting if we take into consideration the much better running time of ST compared to GN.

Table 1. Results for Barabási-Albert graphs (mod. = modularity, #cl. = number of clusters).

#nodes	deg	#edges	Louvain [4]		nover-Louvain		GN [8]		ST	
			mod.	#cl.	mod.	#cl.	mod.	#cl.	mod.	#cl.
40	3	114	0.245	6	0.254	5	0.133	9	0.198	3
40	3	114	0.25	6	0.249	5	0.088	7	0.218	7
80	3	234	0.282	8	0.269	6	0.167	32	0.255	7
80	3	234	0.289	7	0.327	7	0.224	21	0.263	4
100	2	197	0.43	9	0.448	9	0.42	10	0.41	7
100	2	197	0.41	7	0.446	9	0.373	15	0.375	8
150	2	297	0.448	8	0.457	9	0.42	11	0.401	9
150	2	297	0.453	8	0.456	9	0.44	11	0.413	10
150	3	444	0.139	9	0.189	6	0.026	10	0.147	4
150	3	444	0.457	9	0.459	9	0.43	13	0.409	9
200	3	594	0.339	9	0.333	9	0.254	41	0.267	6
200	3	594	0.334	8	0.334	8	0.257	40	0.281	8
400	2	1194	0.476	14	0.486	16	0.446	23	0.427	11
400	2	1194	0.487	12	0.489	13	0.441	27	0.409	6
400	3	1194	0.358	12	0.36	13	0.297	44	0.284	7
400	3	1194	0.353	12	0.364	12	0.287	57	0.296	9
1000	2	1997	0.511	19	0.51	20	0.467	51	0.404	7
1000	2	1997	0.511	19	0.513	20	0.479	50	0.43	9

We observe that the modified nover-Louvain gives slightly better results compared to Louvain in almost all cases. This leads us to conclude that the nover similarity is an important measure which should be further taken into consideration for community detection purposes.

Erdős-Rényi Graphs. In this random graph model [6], a random graph $G(n, p)$ is constructed by connecting n nodes randomly. Each edge exists in G with probability p, independent from every other edge. The average degree is np.

Table 2. Results for Erdős-Rényi graphs (mod. = modularity, #cl. = number of clusters).

#nodes	prob	Louvain [4]		nover-Louvain		GN [8]		ST	
		mod.	#cl.	mod.	#cl.	mod.	#cl.	mod.	#cl.
40	0.1	0.363	6	0.378	5	0.358	8	0.327	6
40	0.1	0.363	6	0.415	6	0.298	11	0.323	7
80	0.1	0.273	6	0.277	7	0.199	25	0.24	5
80	0.1	0.249	8	0.271	6	0.172	26	0.217	5
120	0.1	0.23	8	0.233	8	0.123	54	0.171	11
120	0.1	0.233	8	0.234	6	0.098	67	0.196	8
200	0.1	0.172	9	0.175	8	0.05	105	0.122	7
200	0.1	0.185	7	0.178	8	0.054	109	0.133	11

The graphs we created have 40 to 200 nodes and the probability of the existence of an edge is 0.1. The Erdős-Rényi graphs follow the Poisson degree distribution and can be used for evaluating community detection algorithms [15].

In our experiments in Table 2, one can see that as the number of nodes increases, keeping the same edge probability, the modularity (columns labeled 'mod.') tends to decrease steadily and the number of clusters (columns labeled '#cl.') tends to increase with the same pace for all algorithms, except for GN. For GN the modularity decreases faster and the number of communities increments a lot, and one can assume that a lot of smaller communities are formed. A possible explanation is that as the average degree increments, the GN algorithm will have to remove more edges, evenly distributed through the network, in order to get the partitions. On the other hand, Louvain and nover-Louvain tend to create larger communities since nodes exhibit the preferential attachment behavior. The ST algorithm also creates larger communities by providing a more robust initial structure through the nover scores and the spanning tree computation.

3.2   Real-World Networks

Synthetic datasets allow us to work with randomly generated graphs in which the structure is determined by certain parameters. On the other hand, it makes sense to evaluate our algorithms' performance in real datasets where we know in advance the community structure in some cases. We have thus chosen to experiment with the Zachary's karate club network and one large network, namely Facebook.

Zachary's Karate Club. This is a famous network model, due to Zachary's observations, explained in his study [18]. Over the course of two years, he monitored members of a karate club and relationship among those members. Over time, dispute arose between team's instructor and administrator which resulted in club splitting in two separate clubs. Half of the original club members joined

the new club [8]. Zachary built a network model based on his observations, which included 34 nodes and 78 links.

For Zachary's dataset we get some interesting results. When we compare the algorithms in respect to the modularity value as shown in Table 3, our algorithms do not perform as good as Louvain or GN. However, when we compare them regarding the number of clusters, ST performs much better. It correctly found the 2 communities (Fig. 1a) as in the paper and a small one containing just two nodes (node 8 and 30 in Zachary's paper [18]). This is very close to the split of nodes that actually took place in the Karate Club network. This example provides evidence that modularity, although a generally accepted measure for evaluating a proposed community partition, may fail to correctly reflect the actual community structure. On the other hand, GN finds 5 communities (Fig. 1b), while both nover-Louvain and Louvain find 4 communities (Figs. 1c, d).

Facebook Friendship Network. This is a friendship network between Facebook users [1,11]. In this network a node represents a user and an edge indicates
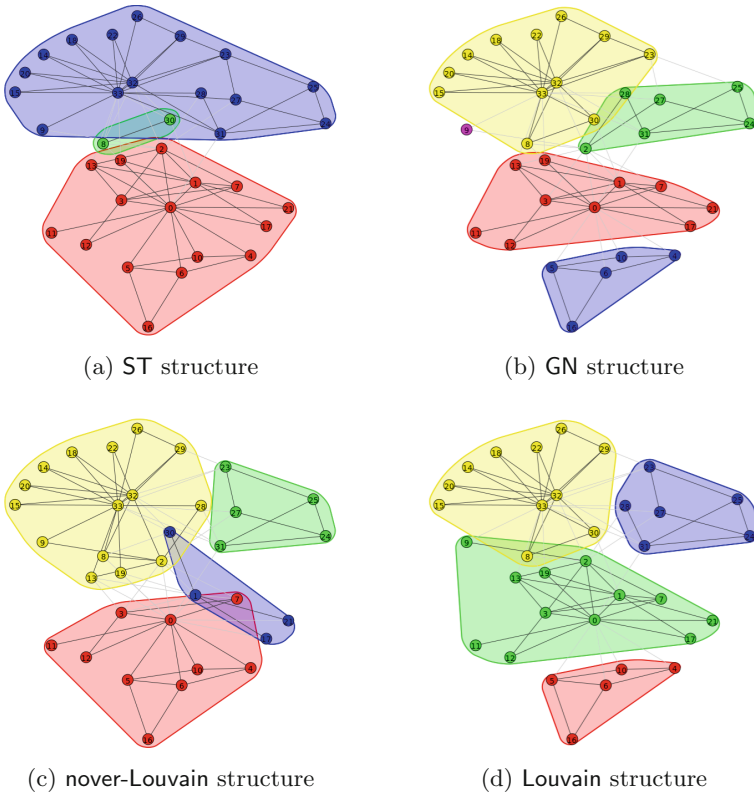

(a) ST structure

(b) GN structure

(c) nover-Louvain structure

(d) Louvain structure

Fig. 1. Zachary's Karate club: community structure as obtained by ST, GN, nover-Louvain, and Louvain algorithms.

friendship between the corresponding users. This is an unweighted directed network with 2888 nodes and 2981 edges. For the purpose of this study we have ignored the edge directions. It is clear that this network is sparse, hence some of the nodes are too far from each other.

For larger graphs it makes sense to focus on parallelizable algorithms, appropriate for distributed implementation. We thus experimented with the Facebook network using GN and ST algorithms, since the Louvain algorithm in its original form seems to be inherently sequential. Our results show that ST yields a modularity very close to the one obtained by GN (see Table 3); since ST is a much faster, parallelizable algorithm it provides a possibly useful alternative whenever time gains and parallelization are more important than a slight modularity decrease.

Table 3. Results for real-world datasets.

Dataset	#nodes	#edges	GN [8]		ST	
			modularity	#clusters	modularity	#clusters
Zachary's karate club	34	78	0.401	5	0.372	3
Facebook	2888	2981	0.807	8	0.804	7

4   Conclusions

The volume of available data and the size of networks of today in various fields, such as social and biological networks, call for efficient distributed methods to identify communities in large scale networks.

Towards this goal we first introduced the ST algorithm, that uses the nover score to assign weights on edges in order to compute a spanning tree and then uses the eb centrality to split the tree into a forest. Each tree in the forest yields a community. The main advantage of the ST algorithm, is that by using nover and the spanning tree it renders computations less time consuming (without losing much on the quality of the results) compared to the GN algorithm. Note also that the ST algorithm is parallelizable and thus is appropriate for distributed implementation.

We next explored the idea that the nover score, being a measure of the similarity of two neighbor nodes, may improve existing algorithms if used in a preprocessing phase. We thus obtained a second algorithm by adding such a preprocessing to the Louvain algorithm. The new algorithm nover-Louvain indeed performed better in several cases.

As future work, we plan to compare these algorithms using different objective criteria to evaluate the communities formed. As mentioned earlier, modularity is a measure of the quality of the communities that may have some drawbacks, as indicated by the Zachary's karate club example. Thus we intend to explore additional measures, such as the Normalized Mutual Information (NMI) score

[5] and the conductance [17] in order to compare the different approaches. Furthermore, we also plan to experiment with alternative graph metrics to obtain edge weights for the preprocessing phase, in order to see if they can lead to further improvement of the proposed algorithms nover-Louvain and ST.

References

1. Facebook (nips) network dataset - KONECT, April 2017. http://konect.uni-koblenz.de/networks/ego-facebook
2. Ahn, Y.Y., Bagrow, J.P., Lehmann, S.: Link communities reveal multiscale complexity in networks. Nature **466**(9), 761–764 (2010)
3. Barabási, A.L., Albert, R.: Emergence of scaling in random networks. Science **286**(5439), 509–512 (1999)
4. Blondel, V.D., Guillaume, J.L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. J. Stat. Mech. Theory Exp. **2008**(10), P10008 (2008)
5. Danon, L., Díaz-Guilera, A., Duch, J., Arenas, A.: Comparing community structure identification. J. Stat. Mech. Theory Exp. **2005**(09), 09008 (2005)
6. Erdös, P., Rényi, A.: On random graphs I. Publicationes Mathematicae Debrecen **6**, 290 (1959)
7. Fortunato, S., Barthelemy, M.: Resolution limit in community detection. Proc. Nat. Acad. Sci. **104**(1), 36–41 (2007)
8. Girvan, M., Newman, M.E.: Community structure in social and biological networks. Proc. Nat. Acad. Sci. **99**(12), 7821–7826 (2002)
9. Liakos, P., Ntoulas, A., Delis, A.: Scalable link community detection: a local dispersion-aware approach. In: IEEE International Conference on Big Data, pp. 716–725 (2016)
10. Liakos, P., Ntoulas, A., Delis, A.: COEUS: community detection via seed-set expansion on graph streams. In: IEEE International Conference on Big Data, pp. 676–685 (2017)
11. McAuley, J., Leskovec, J.: Learning to discover social circles in ego networks. In: Advances in Neural Information Processing Systems, pp. 548–556 (2012)
12. Meghanathan, N.: A greedy algorithm for neighborhood overlap-based community detection. Algorithms **9**(1), 8 (2016)
13. Newman, M.E., Girvan, M.: Finding and evaluating community structure in networks. Phys. Rev. E **69**(2), 026113 (2004)
14. Raghavan, U.N., Albert, R., Kumara, S.: Near linear time algorithm to detect community structures in large-scale networks. Phys. Rev. E **76**(3), 036106 (2007)
15. Reichardt, J., Bornholdt, S.: Statistical mechanics of community detection. Phys. Rev. E **74**, 016110 (2006)
16. Yang, J., Leskovec, J.: Overlapping community detection at scale: a nonnegative matrix factorization approach. In: 6th ACM WSDM 2013, pp. 587–596 (2013)
17. Yin, H., Benson, A.R., Leskovec, J., Gleich, D.F.: Local higher-order graph clustering. In: Proceedings of the 23rd ACM SIGKDD, pp. 555–564 (2017)
18. Zachary, W.W.: An information flow model for conflict and fission in small groups. J. Anthropol. Res. **33**(4), 452–473 (1977)