Overlapping community detection in networks based on link partitioning and partitioning around medoids

Alexander Ponomarenko[1,*], Leonidas Pitsoulis[2], Marat Shamshetdinov[3]

1 National Research University Higher School of Economics, Nizhny Novgorod, Russia
2 University of Thessaloniki, Greece
3 Intel Corp, Nizhny Novgorod, Russia

* aponomarenko@hse.ru

Abstract

In this paper we present a new method for detecting overlapping communities in networks with a predefined number of clusters. The overlapping communities in the graph are obtained by detecting the disjoint communities in the associated line graph by means of link partitioning and partitioning around medoids. Partitioning around medoids is done through the use of a distance function defined on the set of nodes of the linear graph. In the present paper we consider the commute distance and amplified commute distance functions as distance functions. The performance of the proposed method is demonstrated by computational experiments on real life instances.

Introduction

Detection of overlapping communities in a network is the task of grouping the nodes of the network into a family of subsets called clusters, so that each cluster contains nodes which are similar with respect to the overall network structure. Overlapping means that clusters can intersect each other, so that a node can belong to several clusters, in contrast with disjoint community detection where the clusters form a partition of the node set.

To this day there is no widely accepted formal definition for the notion of community in a network. This leads to different community definitions and allows for the existence of a variety of graph clustering methods that can be compared only with respect to their computational complexity and the empirical evaluation of their proposed communities. The common approach to formalize the notion of community in a network is through the use of quality functions which attempt to quantify the degree of community structure captured by a given partition of the nodes. That is, a quality function will in principle attain extreme values for clusterings of the nodes which best reflect the community structure of the graph. Given such a quality function then, community detection translates to an optimization problem.

One of the most well known such quality function is modularity [1], where it has been used by many methods that solve the related optimization problem with varying success. However, it is still an open question of what are the properties of a good quality function [2].

Community detection in networks is still an actively developing area connected to many fields of science that need tools for a complex network analysis including molecular biology, sociology, data mining and unsupervised machine learning. Network clustering methods can be classified according to the approaches they are based on.

There is a plethora of different methods and approaches for overlapping community detection in graphs, a fact which can be partially attributed to the absence of a well defined and widely accepted quality function for overlapping communities, as it is the case with non-overlapping community detection. An attempt to axiomatize quality functions for non-overlapping graph clustering in the form of intuitive properties that any such function should satisfy is presented in [2]. The authors in [2], driven by similar results on distance based clustering present six such properties. For instance, the value of a clustering quality function should not decrease if for a given clustering we add edges between nodes in the same clusters. Moreover, they showed that modularity does not satisfy some of these properties. In a more recent and related work, the authors in [3] compiled a survey of the currently known families of quality functions, or metrics as they call them, for both non-overlapping and overlapping graph clustering. Even more so, the authors in [3] present computational experiments on sets of benchmark instances with known community structure, the compare these quality functions in terms of how do they perform in identifying the communities. The most recent overview and classifiction of the state of the art methods for overlapping community detection, as well as a computational comparison of existing

methods and benchmark instance evaluation can be found in [4]. In [4] the authors present fourteen different algorithms and they propose a unified framework for testing them.

One approach for overlapping community detection is link partitioning also known as link communities identification. The idea of this approach is the following. If we assume that the nodes of a network represent the entities of a system and the edges the binary relations betweem them, instead of partitioning the nodes to form communities which will be non-overlapping, partition the edges in the sense that the relations between the nodes define the community structure and not the nodes themselves. A node will belong to the communities so defined by its adjacent edges. For example, a person may play soccer with a group of playmates on the weekends and go to work with coworkers on other days. Given that a coworker can also be a playmate we have overlapping communities. That person has two types of relations with other persons: "plays soccer with" and "works with". Thus, the person belongs to two communities: the community of soccer players and the community of his colleagues. Such a person is can be considered to be an overlapped node. Despite the fact that link partitioning for overlapping community detection seems very natural, historically the methods that exploit it appeared relatively late. Thus, in 2009 and later in 2010 Evans and Lambiotte [5], [6] were the first who making node partition of a line graph to get an edge partition of the original graph. So they projected the network into a weighted line graph whose nodes are the links of the original graph and after that they applied one of disjoint community detection algorithm. In 2011, Kim and Jeong [7] proposed a modified version of the map equation method (also known as Infomap [8]) to detect link communities under the Minimum Description Length (MDL) principle. Also Evans [9] in 2010 extended line graph approach to using clique graph , wherein cliques of a given order are represented as nodes in a weighted graph. The membership strength of a node i to community c is given by the fraction of cliques containing i which are assigned to c.

In the present paper we present research of one combination of methods which previously has not been studied in the literature.
The paper is organized as follows. In the section Materials and Methods we give a formal definition of the clustering problem, describe the proposed method and give description of the datasets and compared methods that were used in computational experiments. We briefly discuss how to choose input parameters and give the estimation of computational cost of the proposed method in the section Discussion. And traditionally we finalize the paper with the Conclusion section.

Materials and Methods

Problem statement

Let $G(V, E)$ be a graph with n nodes $V = \{v_1, v_2, ..., v_n\}$ and m edges $E \subseteq V \times V$. For a given natural number k define a *cover* as a family of k subsets of nodes

$$\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$$

where each C_i is called a *cluster* or *community*. The goal in community detection is to find a cover $\mathcal{C}$ which best describes the community structure of the graph, in the sense that nodes within clusters are more densely connected than the clusters themselves. We can also associate with $\mathcal{C}$ an *affiliation matrix* $\mathbf{F}_{\mathcal{C}} \in \mathbb{R}^{|V|\times|\mathcal{C}|}$ where F_{vc} corresponds to the degree of affiliation of vertex v with community $c \in \mathcal{C}$. If we impose the following constraints

$$\sum_{c\in\mathcal{C}} F_{vc} = 1, \qquad \forall v \in V \tag{1}$$

$$0 \leq F_{vc} \leq 1, \qquad \forall v \in V, \forall c \in \mathcal{C}. \tag{2}$$

then the values of the affiliation matrix are also known as *belonging coefficients* [10]. In the case of non-overlapping community detection we have that $\mathcal{C}$ must be a partition of V, or equivalently, equation (2) is replaced by the binary constraint $F_{vc} \in \{0, 1\}$.

Proposed method

The proposed method is based on non-overlapping link partitioning. Thus, the task of overlapping community detection is reduced to the problem of finding non-overlapping communities in the set of edges. That also corresponds to the problem of finding non-overlapping communities on a line graph $L(G)$ whose vertices correspond to edges of the original graph G. Two vertices are connected by an edge in $L(G)$ if the corresponding edges in G have a common node. In order to determine disjoint communities in the line graph $L(G)$ we build a distance matrix $D = (d_{ij}) \in \mathbb{R}^{m\times m}$ based on the structure of $L(G)$, and for doing this we utilize a distance function on the nodes of a graph. For this purpose we tested two distance functions; the commute distance [11] and the amplified commute distance [12].

Given that we seek to find k overlapping communities in the original graph G, we compute a set $S = \{s_1, s_2, ..., s_k\}$ of vertices from $L(G)$ which can be considered the medians with respect to the distances in D, that is

$$S := \underset{T,|T|=k}{\arg\min} \sum_{j \in E(G)} \sum_{c \in T} d_{jc} x_{jc}, \tag{3}$$

$$x_{jc} = \begin{cases} 1, & \text{if } d_{jc} \leq d_{js},\ s \in S, \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

Thus, x_{jc} is indicator variable which takes the value 1 when the edge j of the original graph G belongs to cluster c and 0 otherwise. Together expressions 3 and 4 constitute the k-median problem also known as *facility location problem* which is known to be NP-complete [13, 14].

The matrix of belonging coefficients for the final covering is calculated as follows.

$$F_{ic} = \begin{cases} 1, & \text{if } \frac{\sum_{(i,j) \in E} x_{jc}}{d_i} \geq \theta, \\ 0, & \text{otherwise} \end{cases} \tag{5}$$

for every $i \in V(G)$ and $c \in S$, where θ is a threshold parameter, and d_i is the degree of vertex i in the graph G. Thus, the belonging coefficient F_{ic} of node i to cluster c is proportional to the number of adjacent edges belonging to the cluster c.

In summary, in order to find k overlapping communities in a graph G our proposed method **L**ink **P**artitioning **A**round **M**edoids (LPAM) consists of the following steps:

1. Build the line graph $L(G)$
2. Find k disjoint communities in $L(G)$:
   (a) Compute the distance matrix D between each pair of nodes based on compute distance or amplified computed distance
   (b) Solve the k-median problem based on the distances in D and compute the medians $S = \{s_1, s_2, ..., s_k\}$
3. Build a cover for the original graph G based on the affiliation matrix $\mathbf{F}_{\mathcal{C}}$ which is constructed from S and a threshold value θ.

Distance functions

There are various options when it comes to choose a distance function on the nodes of a graph. Intuitively we would like a distance function that reflects the relationship between nodes within the same cluster in a community, so that vertices from the same cluster should have a shorter distance between one another than the distance between them if they were to belong to different clusters. In this paper we employed two distance functions, namely the commute distance and the amplified commute distance.

Commute distance

Commute distance [15] is also known as resistance distance [16] in the literature. The resistance distance can be thought of as the effective resistance between two nodes in a graph, if we consider this graph to be an electrical circuit. It is defined as

$$d_{\text{r}}(i,j) = \frac{K^{(i,j)}}{K_{(ij)}}, \tag{6}$$

where $K_{(ij)}$ is the minor of Kirchhoff matrix, and $K^{(i,j)}$ is a second order algebraic complement, that is, a determinant of the matrix obtained from the Kirchhoff matrix by deleting two rows and two columns i, j.

Commute distance $d_{\text{cm}}(i,j)$ and resistance distance $d_{\text{r}}(i,j)$ are connected by the relation

$$d_{\text{cm}}(i,j) = \text{vol(G)} d_{\text{r}}(i,j), \tag{7}$$

where $\text{vol}(G) = \sum_{v \in V(G)} d_v$ is the volume of the graph G, and d_v is the degree of vertex v. The value of the commute distance $d_{\text{cm}}(v,w)$ between node v and node w on the graph G can be interpreted as the expected number of steps that a random walk needs to take in order to reach node w from v and return back. Intuitively the commute distance seems like a good candidate for capturing the communities in a graph, in the sense that nodes within the same community

should have higher probability to be reachable to each other than nodes from different communities. The number of possible paths between two nodes is directly proportional to the commute distance between these two nodes, and one should expect that pairs of nodes within the same community should have a higher number of paths than pairs from different communities. However it is theoretically flawed when it comes to large graphs. When the size of the graph becomes sufficiently large, the probability to reach a node from another one becomes dependent only on the degree of the destination node, as it was proven in [12]. The authors in [12] call this effect *lost in space*. In order to overcome this drawback, in the same paper the authors proposed the amplified commute distance as a possible improvement.

Amplified commute distance

The amplified commute distance can be expressed as

$$d_{\text{amp}}(i,j) = \frac{d_{\text{cm}}(i,j)}{\text{vol(G)}} - \frac{1}{d_i} - \frac{1}{d_j} + \frac{2w_{ij}}{d_i d_j} - \frac{w_{ii}}{d_i^2} - \frac{w_{jj}}{d_j^2}, \tag{8}$$

where the purpose of the negative terms is to reduce the influence of the edges adjacent to i and j, which completely dominate the behavior of the resistance distance. The term *amplify* is intended to emphasize the general role of the first term. As well as the original commute distance the amplified commute distance is Euclidean [12].

Benchmarks and evaluation

To evaluate the quality of the tested algorithms we employ an implementation of the Normalized Mutual Information measure for sets of overlapping clusters (ONMI) [17]. We used it to measure the difference between the covering produced by the examined algorithm and the known ground truth. In the recent literature, ONMI values have become one of the most widely used measures to calculate the difference between two coverings. Given that many papers in overlapping clustering (e.g. [4, 18, 19]) include the ONMI values for comparison purposes with benchmark graph instances with known ground truth, it enables us to compare our proposed method without necessarily implementing the other methods, given that we use the same benchmark instance set.

An example

In order to help the readers gain some intuition with respect to what covering result to expect from the proposed method, we created a pedagogical example which is illustrated in Figure 5. Given the regular 8×8 lattice, which naturally does not contain any community structure, we applied our method for $k = 4$ number of overlapping communities. As it can be seen in Figure 5 our method, which produced the same results for both the commute and the amplified commute distance, identifies four equal and overlapping communities in such a way such that each community overlaps exactly with other two. The medoids on the line graph are presented by big circles, while the communities in the lattice are identified with colors and the corresponding medoids with bold edges. Similarly, if we choose $k = 2$ we get two equal overlapping communities.

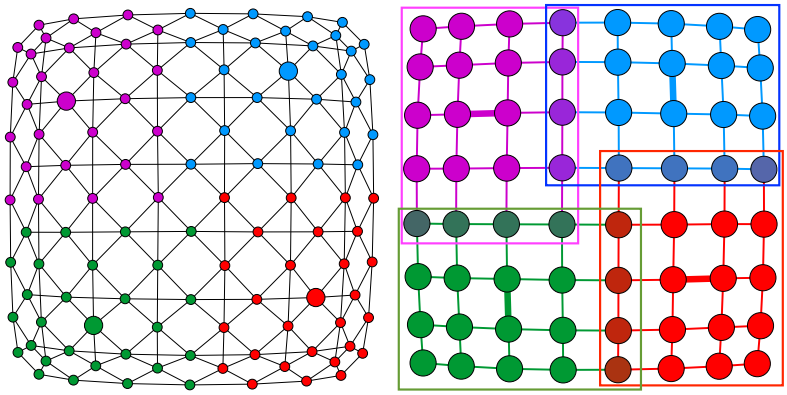


Fig 1. The 8×8 lattice on the right and its line graph on the left.

Compared Methods

Although the performance of the proposed method could in principle be compared with other published methods based solely on the ONMI value, for sake of consistency we used the publicly available implementations of the following three overlapping community detection methods.

- **Greedy Clique Extension** method (GCE) [20]. A method for detecting highly overlapping community structure by greedy clique expansion.
- **OSLOM** [21]. This method uses the metric of the importance of the cluster. The algorithm is based on optimizing this metric by adding new vertices to the cluster or deleting them. This method has the ability to define overlapping clusters, as well as build clusters hierarchies.
- **COPRA** [22]. An iterative method, based on the idea of multi-label propagation with computation complexity close to linear.

Datasets

As real word datasets we used the following four well known network benchmarks with known ground truth.

- Zachary's karate club: social network of friendships between 34 members of a karate club at a US university in the 1970s [23].
- Word adjacencies: adjacency network of common adjectives and nouns in the novel David Copperfield by Charles Dickens [24]
- Books about US politics: A network of books about US politics published around the time of the 2004 presidential election and sold by the online bookseller Amazon.com. The edges between the books represent frequent co-purchasing of books by the same buyers. The dataset can be found on on Valdis Krebs' website `http://www.orgnet.com`
- American College Football: Graph of the games between college football teams which belong to 12 different confederations [25].

Moreover, we constructed a set of synthetic networks with the prefix `bench` with known ground truth, using the instance generation tool which is based on the algorithm published in [26]. The parameters that were used for generating these networks can be found in the `flags.dat` file in the corresponding dataset directory in our git repository. Also we used recently published FARZ network generator [27]

All the relevant information for the above mentions benchmark networks can be seen in the Table 1.

Table 1. A basic statistics of the networks used in computational experiments

Dataset Name	$\|V\|$	$\|E\|$	$\hat{k}$	$\hat{o}$	$\hat{c}$
School Friendship	69	194	6	1	0.49
Karate Club	34	78	2	0	0.588
Football league	62	613	11	8	0.403
Adj-Noun	112	425	2	0	0.19
Politics Book	105	441	3	0	0.676
bench 30	30	120	3	1	0.419
bench 40	40	220	3	5	0.416
bench 50	50	282	4	4	0.396
bench 60	60	260	6	4	0.577
bench 60 dense	60	262	6	5	0.313
FARZ $k = 5, \beta = 1$	200	901	5	0	0.6
FARZ $k = 5, \beta = 0.95$	200	930	5	0	0.591
FARZ $k = 5, \beta = 0.9$	200	907	5	0	0.546
FARZ $k = 5, \beta = 0.85$	200	914	5	0	0.521
FARZ $k = 5, \beta = 0.8$	200	923	5	0	0.486
FARZ $k = 5, \beta = 0.75$	200	913	5	0	0.454
FARZ $k = 5, \beta = 0.7$	200	915	5	0	0.444
FARZ $k = 5, \beta = 0.65$	200	947	5	0	0.423
FARZ $k = 5, \beta = 0.6$	200	931	5	0	0.403
FARZ $k = 5, \beta = 0.55$	200	938	5	0	0.436
FARZ $k = 5, \beta = 0.5$	200	945	5	0	0.402
lattice 8x8	64	112	4	0	0

$|V|$ – number of nodes; $|E|$ – number of links; $\hat{k}$ – number of known clusters; $\hat{o}$ – number of overlapping nodes in ground truth; $\hat{c}$ – avg. clustering coefficient.

Implementation

The main code of the LPAM algorithm is implemented in Java and is accompanied by a set of Jupyter notebooks with Python scipts for running the experiments, as well as the implementations of the comparted methods (OSLOM, GCE, COPRA).

Recall that the LPAM method requires the solution of a k-median problem. For large graphs we solve the k-median problem using an implementation of the CLARANS heuristic [28] from the `smile` library [29]. For comparison reasons we also solved the benchmark problems with an exact solution of the k-median problem, by employing an efficient mixed integer linear programming model by Goldengorin [30]. The exact solutions of this model were found using the publicly available `lp_solve` solver.

All the source codes including all Jupyter notebooks and data sets are publicly available on github at `https://github.com/aponom84/lpam-clustering`.

Results

We have implemented four versions of the LPAM method, given that we solve the k-median problem exactly and with a heuristic, as well as that we chose to use both the commute distance and the amplified commute distance. For each method tested, the ONMI value was calculated between computed clustering and the ground truth. For non-randomized methods we selected the best values of ONMI for the corresponding parameters. For randomized methods we made a sequence of experiments for 10 randomly selected values of a random seed parameter while fixing the other parameters, collect ONMI values and get the average (see S6 Appendix). The summary of the results with the ONMI values for each method are presented in Table 2.

As it can be seen from Table 2, no method dominates the rest with respect to the proximity to the ground truth covering for all data sets. We can see however a superiority of the OSLOM method and the LPAM with the amplified commute distance. The LPAM method with the amplified commute distance get the best results for `Politics Book` graph and for the four synthetic networks: `bench 30`, `bench 40` and `bench 60`. Also in our experiments the same combination gives the second best result for `School Friendship`, `Karate club`, `bench 60` and `bench 60 dense` networks. An example of method output for the `School Friendship` instance can be seen in the Figure 2 and the associated line graph in Figure 3.

We should also note that the LPAM method for the regular lattice example naturally produces the covering that matches to the intuitive separation in contrast to the GCE method which doesn't produce any result for this example, and the OLSOM method can only give an accidentally good result for appropriate choosing the input parameters.

The OMNI values with an asterisk in Table 2 correspond to cases where the LPAM method with the heuristic solution of the k-median problem produced slightly better result compared to the LPAM method with the exact solution. Apparently this was caused due to the fact that sometimes the ground truth communities structure does not match to the neighbors of medoids. Thus, the mistake in identification of the global minimum of the k-median problem may lead to the solutions which are closer to the ground truth. Moreover, the ground truth may not necessarily correspond to the *true* community structure, given that we do not have an exact definition of a community in a network.

Discussion

Tuning parameters

The behavior of the ONMI value depending on threshold parameter θ for the LPAM method is shown in Figure 4. As it can be seen in most cases the maximum ONMI value is reached when threshold value θ lies between 0.3 and 0.6. This can be attributed to the fact that with respect to the proximity to the ground truth covering, it is usually better to either assign a vertex to one cluster or no cluster rather than to several clusters. The final clustering covering for the four different combination of the exact/heuristic version of the LPAM method with the commute and amplified commute distances for all datasets are presented in S1 Appendix,S2 Appendix, S3 Appendix, S4 Appendix. The resulting pictures for the best ONMI values and the full study of the dependence of ONMI value from the input parameters for the GCE, OSLOM and COPRA methods can be found in S5 Appendix, S6 Appendix, S7 Appendix correspondingly.

Computational Complexity

Aside from the solution of the k-median problem which is NP-Hard, the computational complexity of the method is the following. To build line graph $L(G(V, E))$ we need $\Theta(|E|)$ time. Computing the distance matrix depends on the type

Table 2. Comparison ONMI results of methods for overlapping communities detection

Name of network (Dataset)	LPAM Exact CM	LPAM Heuristic-CM	LPAM Exact-ACM	LPAM Heuristic-ACM	GCE	OSLOM	COPRA
School Friendship	0.555983	0.577798*	0.623965	0.640249*	0.745248	0.5651837	0.64575
Karate Club	0.91796	0.91796	0.91796	0.91796	0.685455	0.9586149	0.590609
Football league	0.71194	0.71194	-	0.856281	0.880363	0.9077013	0.853344
Adj-Noun	0.00953164	0.00953164	0.00490969	0.00490969	0.00463105	0.0620827	0.00463105
Politics Book	0.43517	0.440109*	0.464154	0.422712	0.398558	0.4382833	0.409727
bench 30	0.931866	0.931866	0.931866	0.931866	0.348233	0.9246478	0.776329
bench 40	0.203899	0.203899	0.346589	0.273879	0.0969355	0.6039149	0.477972
bench 50	0.309034	0.275624	0.845306	0.845306	0.392884	0.7614544	0.509912
bench 60	0.405531	0.307907	0.60211	0.586147	0.257251	0.7709094	0.485213
bench 60 dense	0.112146	0.0671985	0.466922	0.283863	0.156888	0.512886	0.193731
FARZ $k = 5, \beta = 1$	-	-	-	1.0	0.7384528	1.0	0.9062872
FARZ $k = 5, \beta = 0.95$	-	-	-	0.775677	0.5970368	0.8747809	0.7499637
FARZ $k = 5, \beta = 0.9$	-	-	-	-	0.6797227	0.8343755	0.6824643
FARZ $k = 5, \beta = 0.85$	-	-	-	0.818325	0.5696628	0.7033592	0.6015953
FARZ $k = 5, \beta = 0.8$	-	-	-	0.763537	0.4424587	0.5353789	0.4884105
FARZ $k = 5, \beta = 0.75$	-	-	-	0.629307	0.5824832	0.5896019	0.2363482
FARZ $k = 5, \beta = 0.7$	-	-	-	0.436827	0.4510196	0.4923299	0.0713048
FARZ $k = 5, \beta = 0.65$	-	-	-	0.52277	0.3750271	0.5435125	0.1046695
FARZ $k = 5, \beta = 0.6$	-	-	-	0.342269	0.1883288	0.3167039	0.0007918
FARZ $k = 5, \beta = 0.55$	-	-	-	0.295493	0.1943838	0.3986279	0.0304626
FARZ $k = 5, \beta = 0.5$	-	-	-	0.105063	0.1531077	0.2954157	0.0003813

Each cell contains ONMI values between ground truth covering and the result of corresponding method. Bright green cell background marks the highest (the best) ONMI value. Light green background means the second best result.

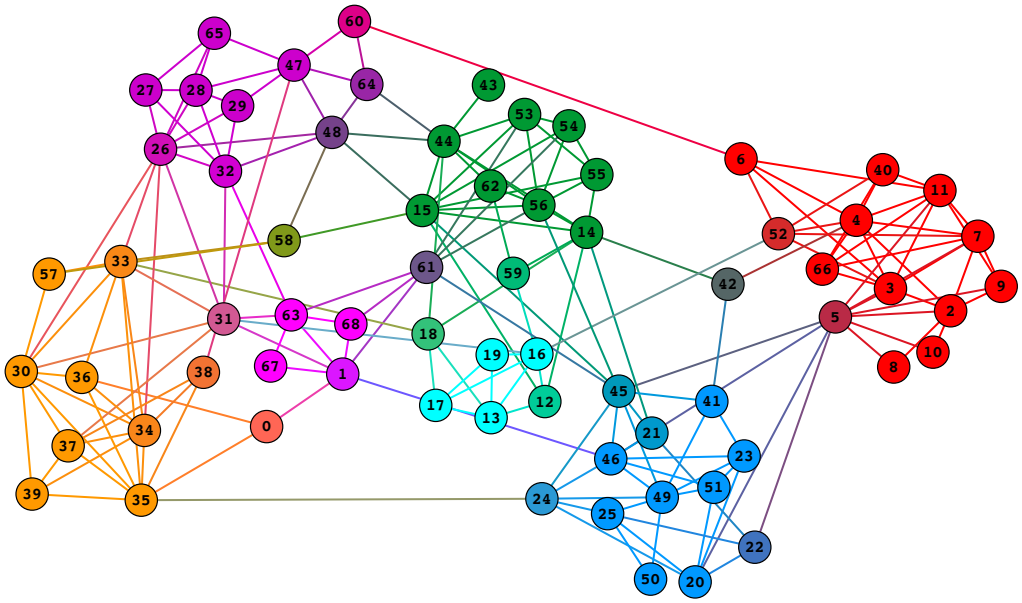


Fig 2. Clustering results of exact version of LPAM method with commute distance function for `School Friendship` network. $\theta = 0.5$

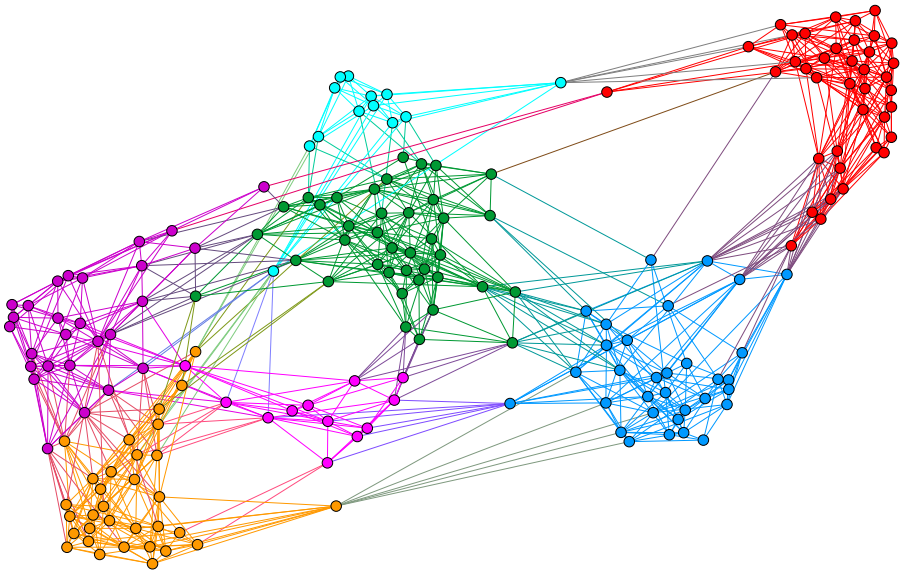


Fig 3. A line graph which is produced by the exact version of LPAM method with commute distance function for `School Friendship` network. $\theta = 0.5$

of the distance function used. So, the matrix with the shortest path distances can be calculated with the Floyd–Warshall algorithm [31] with cubic time on the number of nodes of the linear graph, which means $\Theta(|E|^3)$ time. Also $\Theta(|E|^3)$ time is needed to compute distance matrix with the commute distance function.

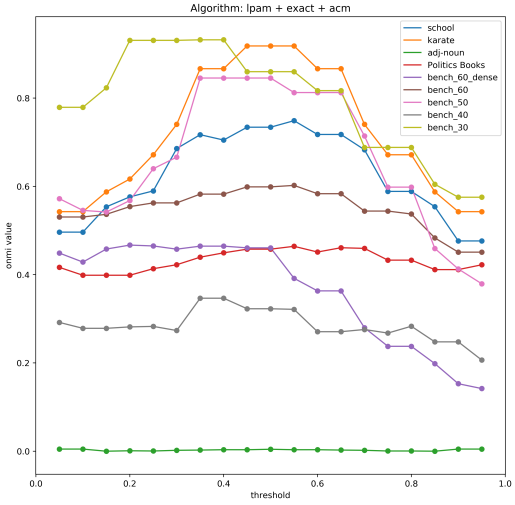


Fig 4. The ONMI results for exact version of LPAM method with amplified commute distance depending on the threshold parameter θ.

Conclusion

In this paper we propose a new method for the detection of overlapping communities in networks with a predefined number of clusters. The proposed method is based on finding disjoint communities on the line graph of the original network, by partitioning around medoids. The resulting link partitioning naturally produces an overlapping community structure for the original graph. The link partitioning is done using the commute distance and its variation which produces more accurate results.

Experimental results on a set of well known benchmark instances as well as artificially generated instances with known ground truth, demonstrate that the proposed method has competitive performance with respect to existing methods in the literature, which provides a motivation to further improve the method.

Supporting information

S1 Appendix. Computational results of LPAM-AMP-Exact. The computational results and clustering result pictures of exact method Link Partitioning by Partitioning Around with Amplified Commute Distance function.

S2 Appendix. Computational results of LPAM-AMP-Heuristic The computational results and clustering result pictures method Link Partitioning by Partitioning Around with Amplified Commute Distance function using CLARANCE heuristic to solve P-median problem.

S3 Appendix. Computational results of LPAM-CM-Exact. The computational results for exact version of LPAM method with commute distance function.

S4 Appendix. Computational results of LPAM-CM-Heuristic The computational results for heuristic version of LPAM method with commute distance function.

S5 Appendix. Computational results of GCE The computational results for Greedy Clique Expansion method.

S6 Appendix. Computational results of OSLOM The computational results for OSLOM method.

S7 Appendix. Computational results of COPRA The computational results for COPRA method.

Acknowledgments

Authors thank Anna Yaushkina and Nikita Putikhin for the help with implementation of amplified commute distance function and for adding heuristic version of LPAM method. Also thank Eldar Yusupov who set up the computational cluster of LATNA Labaratory of HSE.
The work of Alexander Ponomarenko was conducted at National Research University Higher School of Economics and supported by RSF grant 14-41-00039

References

[1] Michelle Girvan and Mark EJ Newman. Finding and evaluating community structure in networks. *Proceedings of the national academy of sciences*, 69:026113, 2004.

[2] Twan van Laarhoven and Elena Marchiori. Axioms for graph clustering quality functions. *Journal of machine learning research*, 15:193–215, 2014.

[3] Tanmoy Chakraborty, Ayushi Dalmia, Animesh Mukherjee, and Niloy Ganguly. Metrics for community analysis: A survey. *ACM Comput. Surv.*, 50(4):54:1–54:37, August 2017.

[4] Jierui Xie, Stephen Kelley, and Boleslaw K Szymanski. Overlapping community detection in networks: The state-of-the-art and comparative study. *Acm computing surveys (csur)*, 45(4):43, 2013.

[5] T. S. Evans and R. Lambiotte. Line graphs, link partitions, and overlapping communities. *Phys. Rev. E*, 80:016105, Jul 2009.

[6] Tim S Evans and Renaud Lambiotte. Line graphs of weighted networks for overlapping communities. *The European Physical Journal B-Condensed Matter and Complex Systems*, 77(2):265–272, 2010.

[7] Youngdo Kim and Hawoong Jeong. Map equation for link communities. *Physical Review E*, 84(2):026110, 2011.

[8] Martin Rosvall and Carl T Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, 105(4):1118–1123, 2008.

[9] Tim S Evans. Clique graphs and overlapping communities. *Journal of Statistical Mechanics: Theory and Experiment*, 2010(12):P12037, 2010.

[10] Hua-Wei Shen, Xue-Qi Cheng, and Jia-Feng Guo. Quantifying and identifying the overlapping community structure in networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2009(07):P07042, jul 2009.

[11] Luh Yen, Denis Vanvyve, Fabien Wouters, François Fouss, Michel Verleysen, and Marco Saerens. clustering using a random walk based distance measure. In *ESANN*, pages 317–324, 2005.

[12] Ulrike V Luxburg, Agnes Radl, and Matthias Hein. Getting lost in space: Large sample analysis of the resistance distance. In *Advances in Neural Information Processing Systems*, pages 2622–2630, 2010.

[13] Robert J Fowler, Michael S Paterson, and Steven L Tanimoto. Optimal packing and covering in the plane are np-complete. *Information processing letters*, 12(3):133–137, 1981.

[14] Teofilo F Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985.

[15] L. Lovász. Random walks on graphs: A survey. In D. Miklós, V. T. Sós, and T. Szőnyi, editors, *Combinatorics, Paul Erdős is Eighty*, volume 2, pages 353–398. János Bolyai Mathematical Society, Budapest, 1996.

[16] Douglas J Klein and Milan Randić. Resistance distance. *Journal of mathematical chemistry*, 12(1):81–95, 1993.

[17] Aaron F. McDaid, Derek Greene, and Neil Hurley. Normalized mutual information to evaluate overlapping community finding algorithms. October 2011.

[18] Alexander J Gates, Ian B Wood, William P Hetrick, and Yong-Yeol Ahn. On comparing clusterings: an element-centric framework unifies overlaps and hierarchy. *arXiv preprint arXiv:1706.06136*, 2017.

[19] Hamideh Sadat Cheraghchi and Ali Zakerolhosseini. Mining dynamic communities based on a novel link-clustering algorithm. *International Journal of Information & Communication Technology Research*, 9(1):45–51, 2017.

[20] Conrad Lee, Fergal Reid, Aaron McDaid, and Neil Hurley. Detecting highly overlapping community structure by greedy clique expansion. *arXiv preprint arXiv:1002.1827*, 2010.

[21] Andrea Lancichinetti, Filippo Radicchi, José J Ramasco, and Santo Fortunato. Finding statistically significant communities in networks. *PloS one*, 6(4):e18961, 2011.

[22] Zhi-Hao Wu, You-Fang Lin, Steve Gregory, Huai-Yu Wan, and Sheng-Feng Tian. Balanced multi-label propagation for overlapping community detection in social networks. *Journal of Computer Science and Technology*, 27(3):468–479, 2012.

[23] Wayne W Zachary. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 33(4):452–473, 1977.

[24] Mark EJ Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 74(3):036104, 2006.

[25] Michelle Girvan and Mark EJ Newman. Community structure in social and biological networks. *Physical review E*, 99(12):7821–7826, 2002.

[26] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical review E*, 78(4):046110, 2008.

[27] Justin Fagnan, Afra Abnar, Reihaneh Rabbany, and Osmar R Zaiane. Modular networks for validating community detection algorithms. *arXiv preprint arXiv:1801.01229*, 2018.

[28] Raymond T. Ng and Jiawei Han. Clarans: A method for clustering objects for spatial data mining. *IEEE transactions on knowledge and data engineering*, 14(5):1003–1016, 2002.

[29] Smile statistical machine intelligence and learning engine. `https://haifengl.github.io/smile/`. Accessed: 2010-09-30.

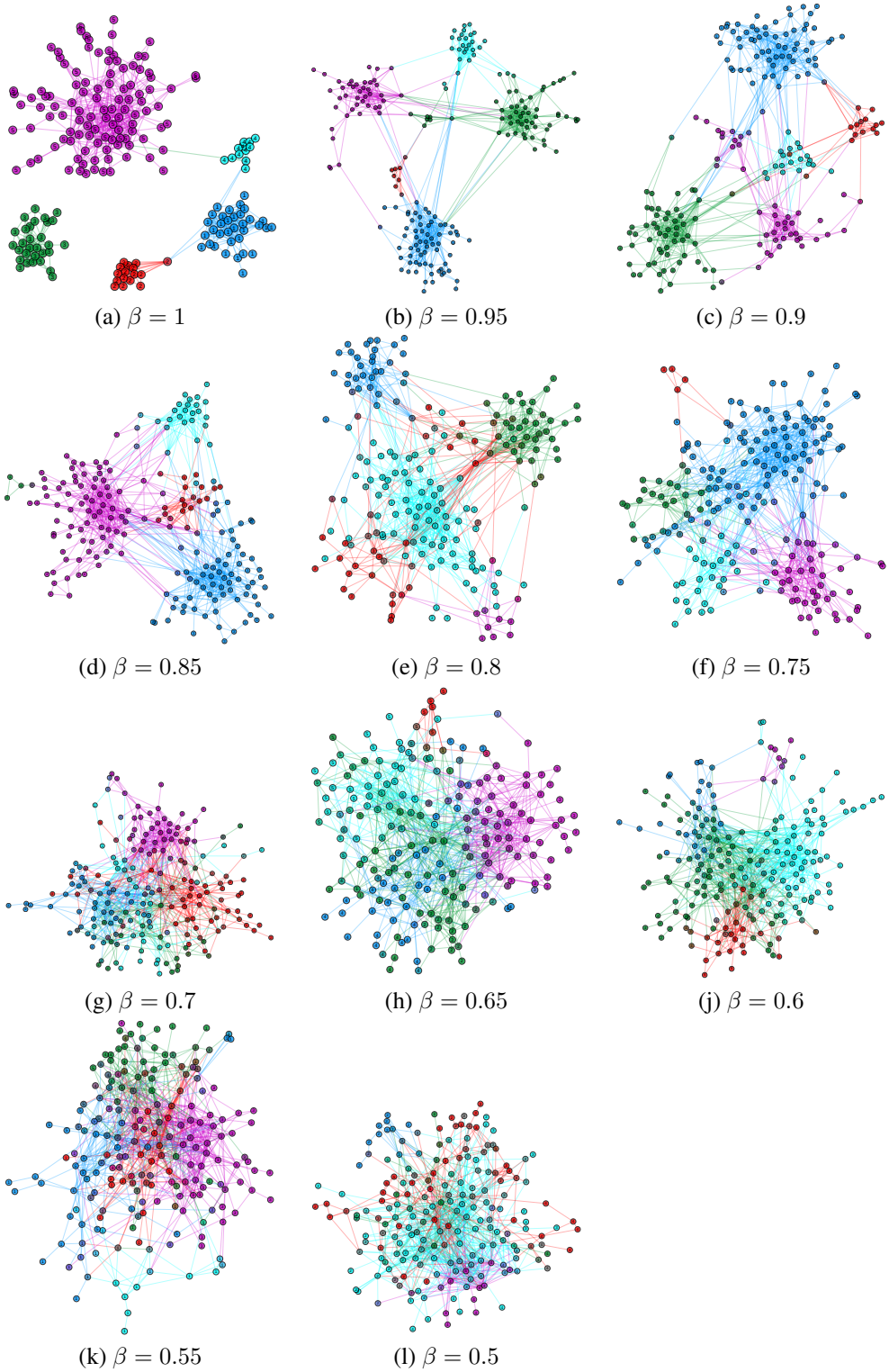
[30] Bader F AlBdaiwi, Diptesh Ghosh, and Boris Goldengorin. Data aggregation for p-median problems. *Journal of Combinatorial Optimization*, 21(3):348–363, 2011.

[31] Robert W Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.

Appendix: Extended Results

Fig 5. Clustering result of LPAM method with amplified commute distance function on FARZ network $n = 1000, m = 7, k = 20, \beta = 0.75$

Table 3. Clustering results of heuristic version of LPAM method with Amplified Commute Distance for FARZ networks with 200 nodes and 20 communities



(a) $\beta = 1$ (b) $\beta = 0.95$ (c) $\beta = 0.9$

(d) $\beta = 0.85$ (e) $\beta = 0.8$ (f) $\beta = 0.75$

(g) $\beta = 0.7$ (h) $\beta = 0.65$ (j) $\beta = 0.6$

(k) $\beta = 0.55$ (l) $\beta = 0.5$