Karlsruhe Institute of Technology

A Comparative Study of Overlapping Community Detection Algorithms

Bachelor Thesis of

John Gelhausen

At the Department of Informatics
Institute of Theoretical Informatics

Reviewers:	Prof. Dr. Dorothea Wagner
	Prof. Dr. Peter Sanders
Advisor:	Michael Hamann

Time Period: 1st February 2019 – 31st May 2019

Statement of Authorship

I hereby declare that this document has been composed by myself and describes my own work, unless otherwise acknowledged in the text.

Karlsruhe, June 18, 2019

Abstract

The goal of detecting communities in networks is to find groups of nodes which are densely connected to each other and sparsely connected to the rest of the network. Overlapping communities allow nodes to be part of multiple communities. We review a total of nine algorithms for overlapping community detection and compare them to each other by conducting experiments on synthetic benchmark networks and real networks. The algorithms are empirically evaluated using performance metrics that evaluate the similarity of detected communities to reference communities, such as the Normalized Mutual Information (NMI). We carry out additional experiments to gain more insights into the behaviour of the algorithms, such as verifying if the algorithms detect too many or too few, too small or too large communities. Our results show that, overall, OSLOM and MOSES perform the best. Whereas OSLOM performs better on smaller networks, MOSES performs better on larger networks. Our results also show that it is very important to use complementary metrics to evaluate the performance of overlapping community detection algorithms. Performance metrics, such as the NMI or the Omega Index, only measure the overall quality of a detected cover. Whereas, complementary metrics give us more information about the behaviour of each algorithm at detecting overlapping communities. Finally, while some algorithms perform well on synthetic networks, none of the algorithms are able to detect the community structure in the real networks. This is due to the detected communities of the algorithms being substantially different to the communities defined by the meta-data.

Deutsche Zusammenfassung

Das Ziel der Erkennung von Communities in Netzwerken ist es, Gruppen von Knoten zu finden welche dicht zueinander und spärlich zu dem Rest des Netzwerkes verbunden sind. Überlappende Communities erlauben es den Knoten Teil mehrerer Communities zu sein. Wir evaluieren neun Algorithmen für die Erkennung von überlappenden Communities, indem wir sie mit Hilfe von Experimenten auf synthetischen Benchmark Netzwerken und realen Netzwerken miteinander vergleichen. Die Algorithmen werden empirisch evaluiert anhand von Performanz-Metriken welche die Ähnlichkeit der gefundenen Communities zu denen der Referenz Communities berechnen. Eine solche Metrik ist zum Beispiel die Normalized Mutual Information (NMI). Wir haben zusätzliche Experimente durchgeführt, um die Algorithmen detaillierter analysieren zu können. Zum Beispiel haben wir überprüft, ob die Algorithmen zu wenige oder zu viele, zu kleine oder zu große Communities finden. Unsere Resultate zeigen, dass OSLOM und MOSES am besten sind, um überlappende Communities zu finden, wobei OSLOM bessere Resultate auf kleinen Netzwerken und MOSES bessere Resultate auf größeren Netzwerken liefert. Unsere Resultate zeigen auch, dass es sehr wichtig ist ergänzende Metriken zu benutzen um die Qualität von den Algorithmen zu evaluieren. Qualitäts-Metriken, wie die NMI oder der Omega Index, berechnen nur die Gesamtqualität eines gefundenen Covers. Dagegen liefern uns ergänzende Metriken mehr Informationen über das Verhalten der Algorithmen. Schlussendlich, während einige Algorithmen gute Resultate auf synthetischen Netzwerken liefern, sind keine der Algorithmen in der Lage die Community Struktur in realen Netzwerken zu erkennen. Das kommt daher, dass die erkannten Communities grundlegend unterschiedlich von den Communities sind die durch Meta-Daten definiert sind.

Contents

1. Introduction

Networks are used in different disciplines, such as social [MMG+07], computer sciences or biology [Bar04], to represent and analyse complex data. A network consists of *nodes* and *edges*, which connect a pair of vertices. Many of these networks show community structure, which means that the nodes in the network are part of groups of nodes, called communities. A community is commonly a subnetwork of a network that is densely connected internally but is sparsely connected to the rest of the network. An example of communities are groups of acquaintances in social networks, such as Facebook, where the nodes represent the users and the edges represent a friendship or relationship between two users. Other examples exist, not only in computer science, but also in other disciplines, such as biology (protein interaction networks). The most important types of communities are *disjoint* communities, called *partitions*, where each node belongs to at most one community and *overlapping* communities, called *covers*, where each node can belong to multiple communities. We consider *crisp* covers, which means that nodes share multiple communities with equal strength [FH16].

Detecting communities in such networks is important, because it may grant additional information of how networks are structured. This gives an overview of the network and thus those could get more easily readable and understandable.

Most community detection algorithms have been developed for disjoint community detection [FH16], but in networks such as the social network Facebook the users can belong to multiple different communities which can easily overlap [XKS13]. For example, people can be part of communities of close friends or of colleagues. This observation of possible overlapping communities in networks has led to the development of overlapping community detection algorithms.

Evaluating an algorithm's performance can be done by taking networks with well-defined communities, called *ground-truth* communities, and testing how good the algorithm can recover these communities. Preferably, we would run the algorithms on real networks, such as the Facebook network, which are not computer generated. However, not a lot of real networks with well-defined communities exist. This has led to the development of computer generated benchmark networks with built-in ground-truth communities, called *synthetic* networks. The most popular synthetic benchmark networks are the LFR networks [LFR08, LF09], which consider the heterogeneity in the distribution of node degrees and the community sizes. A recent addition to synthetic benchmark networks are CKB networks [CKB+14].

This work presents a comparative study of overlapping community detection algorithms on synthetic and real networks. There already exist several empirical comparisons of overlapping community detection algorithms of which some are described in Section 2. Building on these comparisons, we carry out several new experiments to gain a more precise insight in how well the algorithms perform. For example, we compare each detected community to each ground-truth community, to see which ground-truth communities are recovered by the algorithms. Furthermore, we use newly published synthetic benchmark networks, the CKB networks, in addition to the popular LFR networks.

As for overlapping community detection algorithms, we use a few popular algorithms, often seen in comparisons, such as OSLOM [LRRF11], MOSES [MH10], GCE [LRMH10] and GANXiS (SLPA) [XSL11]. Furthermore, we use the overlapping extension of the popular disjoint community detection algorithm InfoMap [RAB09], which has been extended by the authors to allow for overlapping community detection. Another overlapping community detection algorithm, often used in comparisons is COPRA [Gre10], however we use BMLPA [WLG+12], because it is an improved version of COPRA. We also use BigClam [YL13b], which is an overlapping community detection algorithm that scales to large networks with millions of nodes and edges. Another overlapping community detection algorithm that we use is the EgoSplitting algorithm [ELL17], which showed promising results and also a better performance than DEMON [CRGP14]. As a last algorithm, we use OLP [RAK07], which is a simple label propagation algorithm able to detect overlapping communities. All the algorithms that we use in our comparison are described in more detail in Section 4.

The goal of this study is to provide an extensive empirical review of nine state-of-the-art overlapping community detection algorithms by conducting different experiments on synthetic and real networks, such as verifying if the algorithms detect too many or too few communities or too small or too large communities, or verifying if the algorithms are able to detect overlapping nodes, which are nodes that belong to more than one community.

Section 2 presents a short overview of related work. In Section 3 important definitions of graph theory and community detection are given. In Section 4 a short description of each algorithm studied in this work is presented. Section 5 provides an overview of the methodology used to carry out the evaluations and experiments. In Section 6 the results of the conducted experiments are presented and discussed. Conclusions are given in Section 7.

2. Related Work

In recent years, a range of comparisons of overlapping community detection algorithms have been published. In this section a short overview of work that has already been done in this field will be shown. First of all, we take a look at recent surveys and reviews that have been published to compare the performance of overlapping community detection algorithms.

Xie et al. [XKS13] provide a review of fourteen overlapping community detection algorithms. A wide range of tests are performed on synthetic networks generated by using the LFR graph generator. The NMI and Omega Index [CD88] were used as performance metrics to measure the quality of the detected communities. Additionally, complementary experiments were carried out to gain further insight into how the algorithms perform. For example, they verified if the algorithms are able to detect the correct overlapping nodes. Furthermore the community size distribution of detected communities is compared to the ground-truth, to see if the algorithms detect too small or too large communities. The results of the review show that not only performance metrics are needed to evaluate the algorithms, but also complementary evaluations to gain a better insight in how algorithms behave. Metrics such as NMI only analyses the accuracy of a algorithm globally, whereas complementary metrics could provide a more precise analysis.

Due to a lack of real networks with ground-truth communities, J. Leskovec et al. [YL13a] published a set of real networks, which are accessible from the Stanford Network Analysis Project's collection of datasets[1], for which they defined ground-truth communities from the meta-data of the real networks. In addition, they gathered thirteen different commonly used community scoring functions, which all build on the intuition that communities are densely connected internally and sparsely connected to the rest of the network. Communities, which are intuitively defined as densely connected internally and sparsely connected to the rest of the network, should provide good results for each of these community scoring function. They used six of these community scoring functions to define the top 5000 communities for each real network, which show the highest quality for each of the six aforementioned community scoring functions.

Hric et al. [HDF14] evaluated a total of eleven community detection algorithms on various synthetic and real networks. Four of the algorithms are able to identify overlapping communities whereas the rest can only detect disjoint communities. The datasets were split into two groups. The first group contained the synthetic networks generated by using

[1] http://snap.stanford.edu/data

the LFR benchmark generator and a set of classic real networks, as tests for community detection algorithms, were used such as the Zachary karate club network. The second group consisted of more recent and larger real networks such as the Amazon or DBLP network. Note that some of these real networks do not contain overlapping ground-truth communities. To measure the similarity between the detected communities and the ground-truth communities, the NMI was used. Looking at the results, the algorithms perform the best on the LFR networks and the classic real networks, but do not provide good results on the larger real networks. Furthermore, they pointed out that the community detection algorithms only rely on the network structure to detect communities in networks, which we will call structural communities. However, the ground-truth communities of the real networks are not defined by the network structure, instead they were defined by the meta-data of the nodes in the network. The poor performance of the algorithms for the large real networks is due to the structural communities detected by the algorithms being substantially different to the meta-data communities of the real networks.

Harenberg et al. [HBG+14] compared nine disjoint community detection algorithms and four overlapping community detection algorithms, published up to 2013, on large-scale real networks with ground-truth communities. For the overlapping community detections, they used the top 5000 ground-truth communities described by J. Leskovec et al. [YL13a]. No synthetic networks were used. To evaluate the algorithms, they used four community scoring functions, the edge density, conductance, clustering coefficient and the triangle participation ratio, described in [YL13a]. In addition the similarity between the set of ground-truth communities and the set of detected communities were measured by using performance metrics such as Recall, Precision, F1-Score and the Normalized Mutual Information (NMI). The results of their study show that the goodness metrics are not equivalent to the performance metrics. In other words an algorithm that finds communities with good structural properties does not necessarily return good performance metrics when compared to ground-truth communities. This means that some of the ground-truth communities of the real networks defined by the meta-data are not good communities in such that they do not provide good results for some of the community scoring functions. By good communities, we mean communities that are densely connected internally and sparsely connected to the rest of the network. Combining these results with the results of the aforementioned work [HDF14], this begs the question if the ground-truth communities in the real networks, that are solely based of the meta-data, are reliable.

Comparisons of the performance of overlapping community detection algorithms are not only done in reviews or surveys. If a new overlapping community detection algorithm is published, the authors usually compare their new algorithm to older overlapping community detection algorithms to see if their algorithm performs better.

Lee et al. [LRMH10] introduced an overlapping community detection algorithm, *GCE*. They compared the performance to detect overlapping communities to three other overlapping community detection algorithms (COPRA [Gre10], CFinder [PDFV05] and LFM [LFK09]). They evaluated the algorithms on various LFR networks and on a protein-protein interactions network. To measure the similarity of the detected communities to the ground-truth communities, they used the performance metric NMI. To verify if the algorithms are able to detect overlapping communities, they used LFR networks with increasing amount of communities per node. The results show that GCE performs the best on networks with high overlap.

McDaid et al. [MH10] introduced *MOSES*, which is a community detection algorithm capable of detecting overlapping communities. They used four other community detection algorithms (GCE, LFM, COPRA and Louvain [BGLL08]). They carried out the same experiments as the aforementioned work, by using the performance metric NMI to measure

the similarity of each detected community to the ground-truth. Their results show that MOSES performs the best on LFR networks with many communities per node, while the other algorithms are only able to detect overlapping communities on networks with few communities per node. However, MOSES also has the highest running time.

Lancichinetti et al. [LRRF11] introduced an overlapping community detection algorithm, *OSLOM*. To compare the performance of each algorithm to detect overlapping communities, they used LFR networks with varying amount of overlapping nodes and varying amount of communities per node. They compared OSLOM with MOSES and COPRA. Their results show that OSLOM performs clearly better than COPRA in detecting overlapping communities. OSLOM and MOSES perform similarly on networks with many communities per node, however MOSES's performance worsens faster as the number of overlapping nodes increases.

3. Preliminaries

In this section, a brief overview of important definitions, that are used throughout this work, is given.

Graph Theory. A *graph* or *network* G is a pair (V, E) consisting of a set of *nodes* V and a set of *edges* E $\subseteq \{\{u, v\}|(u, v) \in V^2 \wedge u \neq v\}$, where an edge is a 2 element subset of V. This type of graph is called *undirected simple* graph. Let n and m be the number of nodes and edges inside the graph G. The *degree d(v)* of a node v is the amount of neighbours of v. A *triplet* is a tuple of three nodes (u, v, w) where (u, v) and (v, w) $\in$ E. If (w, u) $\in$ E then the triplet is *closed* otherwise it is *open*.

Communities and Cover. A *community* is a subgraph $S = (V_S, E_S)$ of a graph G, where $V_S \subseteq V$ and $E_S \subseteq E$. Let $k_S^{int} = |\{(u, v)|u \in S \wedge v \in S\}|$ and $k_S^{ext} = |\{(u, v)|u \in S \wedge v \notin S\}|$ be the internal and external degree of the community S. The *total degree* k_S is defined as follow: $k_S = |\{(u, v)|u \in S\}|$.

For disjoint community detection, the set of communities is called a
partition $P = \{p_1, p_2, ..., p_k\}$. In a partition a node is only allowed to belong to one community. In the context of overlapping community detection, the set of communities is called a *cover* $C = \{c_1, c_2, ..., c_k\}$. In a cover, a node is allowed to be part of multiple communities. We consider only *crisp* covers which means that nodes share multiple communities with equal strength. *Reference* communities are communities embedded in the network that are considered the correct result. They are also referred to as *ground-truth* communities.

4. Algorithms

This section provides a short overview of all the algorithms for overlapping community detection, that are used in this study (Table 4.1). For some of these algorithms we have to define specific parameters. The parameters, we chose for each algorithms, are specified in the following. For a better overview, the algorithms are categorized using the classes proposed by Xie et al. [XKS13]. Some algorithms are not part of any of these classes.

Table 4.1: Overview of algorithms used in this study

Algorithms	Reference	Complexity
EgoSplitting_(PLP)	[ELL17]	?
EgoSplitting_(LPPotts_par)	[ELL17]	?
OSLOM	[LRRF11]	$O(n^2)$
GCE	[LRMH10]	$O(mc)$
GANXiS	[XSL11]	$O(tm)$
OLP	[RAK07]	?
InfoMap	[ER11]	?
BMLPA	[WLG+12]	$O(n \log n)$
MOSES	[MH10]	$O(mn^2)$
BigClam	[YL13b]	?

1. *Label Propagation*

   Initially every node is initialized with a unique label. After that every node checks its neighbour's labels and replaces its own label with the one that is the most common among its neighbours. This process is repeated a number of times. Finally, all nodes having the same label form a community.

   GANXiS, also known as SLPA, Speaker-listener Label Propagation Algorithm, has been introduced by Xie et al. [XSL11]. This label propagation algorithm allows each node to save a list of labels, instead of just one. Initially, every node is initialized with an unique label. In each iteration, the nodes are shuffled. Then, the algorithm iterates over all nodes. Each neighbour v of the currently selected node u selects a random label l from its list of labels with a probability proportional to the occurrence frequency of the label l in its list of labels and sends the label l to the node u. Node u then adds the most common label received to its list of labels. This process is repeated

a maximum number of times t. Finally a probability distribution is created for all labels in each node's list of labels and every label that has a lower probability than the threshold parameter r is deleted. For GANXiS, we set the threshold parameter r to 0.01, which allows the algorithm to detect overlapping communities. Note that a threshold parameter r of 0.5 or higher results in GANXiS outputting partitions. The time complexity is *O(tm)*, where m is the number of edges and t is the maximum number of iterations.

BMLPA, Balance Multi-Label Propagation Algorithm, has been introduced by Wu et al. [WLG+12]. Instead of initializing every node with an unique label, a rough core extraction algorithm is used to give the initial labels to some of the nodes. Every node u has a set of pairs (c, b) where c is a community and b is the belonging coefficient. All belonging coefficients sum to 1. After that each propagation step sums the belonging coefficient of each community over every neighbour of node u. Then the community c_{max} with the highest belonging coefficient b_{max} is selected and every community is removed for which: $\frac{b}{b_{max}} \geq p$ does not hold, where p is a user defined threshold parameter. For BMLPA, we set the threshold parameter p to 0.75, which is the preferred value proposed by the authors. The time complexity is $O(n \log n)$.

OLP is an Overlapping Label Propagation algorithm by Raghavan et al. [RAK07] which allows overlapping community detection. This algorithm allows each node to retain up to k most common labels. Then the nodes are assigned to each corresponding community, based on the labels they retained. The parameter k is set to 3 and only communities of size larger than 5 are kept. The implementation used was provided by Armin Wiebigke[1].

2. *Local Expansion*

Local Expansion algorithms usually revolve around growing a natural community [LFK09]. In general, initial nodes are taken as seeds, which are expanded to communities until a certain condition of a fitness function is met.

OSLOM, Order Statistics Local Optimization Method, has been introduced by Lancichinetti et al. [LRRF11]. OSLOM tries to detect statistically significant communities in the network. The statistical significance of a community is defined as the probability of finding a community with similar properties in a random network without community structure. Communities are then detected by estimating the community's statistical significance through adding and removing nodes from the community. The worst case time complexity is $O(n^2)$.

GCE, Greedy Clique Expansion, has been introduced by Lee et al. [LRMH10]. GCE first starts by taking maximal cliques as a set of seeds, then expanding them by greedily maximizing a local fitness function. Finally a check is performed to remove near-duplicates of cliques and communities. To remove near-duplicates of cliques, a Clique Coverage Heuristic (CCH) is used. Each clique is removed if more than a proportion ϕ of its nodes is contained in at least two already accepted larger cliques. Furthermore every seed that is within some distance η of an already accepted community is discarded. For GCE, we set the minimum clique size k to 4, the overlap to discard η to 0.6 and the CCH threshold ϕ to 0.75, which are the preferred values proposed by the authors. The time complexity is *O(mc)*, where m is the number of edges and c is the number of cliques to be expanded.

3. *Others*

[1] https://github.com/ArminWiebigke/networkit/tree/Dev/

MOSES, Model-based Overlapping Seed ExpanSion, has been introduced by McDaid et al. [MH10]. MOSES builds upon a modified OSBM (Overlapping Stochastic Block Modeling), introduced by Latouche et al. [LBA11]. Initially edges are randomly selected. Each edge represents an initial community. After that, the communities are greedily expanded by maximizing a global fitness function. Finally entire communities are periodically removed to see if that improves the fitness function. In addition after the expansion of the edges, nodes are removed from the communities to which they belong and then added to different communities to see if that improves the fitness function. The time complexity is $O(mn^2)$.

Egosplit has been introduced by Epasto et al. [ELL17]. This algorithm functions in two steps. In the first step, for every node u, a subnetwork induced by the neighbourhood of u is constructed, which is called an ego-net. Then, the ego-net is partitioned by a disjoint community detection algorithm. After that, for each community in the partition, a replica node v of the node u is created which is associated uniquely with one of the communities in the partition. Then, each node u, in the original network, is replaced by their replica nodes. This outputs a new network, called the *persona* network. In the last step the persona network is partitioned by another disjoint community detection algorithm. The partition in the persona network then represent a cover in the original network. For example, if a node u is replaced by two replica nodes, the algorithm will place these two replica nodes into two different communities and if you merge the two replica nodes back to its initial node u, then the node u belongs to two communities in the original network. In this study two different partitioning algorithms are used, a Label Propagation Algorithm using the Absolute Potts Model technique (LPPotts_par) and another Label Propagation algorithm (PLP) proposed in [RN10] and [RAK07] respectively. The implementation used was provided by Armin Wiebigke[2].

InfoMap has been introduced by Rosvall et al. [ER11]. The algorithm optimizes the map equation [RAB09] by combining the problem of detecting communities in a graph and the problem of finding a description of minimum length of a random walk in the graph. The idea behind the random walker is, that it will remain a long time inside a community and movements between communities are rare.

BigClam, Cluster Affiliation Model for Big Networks, has been introduced by Yang et al. [YL13b]. BigClam builds on models of affiliation of nodes to communities maximising an objective function, using non-negative matrix factorization. Each node-community pair is assigned a non-negative factor which represents the degree of membership of that node to the community. Then, the probability of an edge between two nodes is modelled as a function of the shared community affiliations. The intuition behind this model is that nodes are more likely to be neighbours when they share more communities. For BigClam, we set the number of communities to detect c to -1 (detect automatically), the minimum and maximum number of communities to try mc and xc to 5 and 100 respectively, which are the default values proposed by the authors.

[2]https://github.com/ArminWiebigke/networkit/tree/Dev/

5. Evaluation Methodology

This section describes the methodology we used to evaluate the overlapping community detection algorithms. In the following, we present the datasets and the metrics that we use to evaluate the performance of the algorithms.

5.1 Data Sets

The experiments were carried out on several synthetic benchmark networks and real networks, which contain ground-truth communities. We use small (5000 nodes) and large (50000 nodes) synthetic benchmark networks. We use three different models to generate the synthetic benchmark networks: LFR [LFR08], CKB [CKB$^+$14] models and Erdös-Renyi [ER59] model as control. The parameter selection for the LFR and CKB networks are specified in the tables 5.1 and 5.2 respectively.

Synthetic Networks. While the initially proposed LFR model considers only disjoint communities, they extended the LFR model to generate overlapping communities [LF09].The LFR model provides heterogeneity in the distribution of community sizes and node degrees which are features often seen in real networks. Additionally, the LFR model provides a vast number of parameters to control the topology of the network. We set the majority of the parameters similar to other comparison studies, such as [XKS13] or [LRMH10]. Node degrees and community sizes are controlled by power law distributions with exponents $\tau_1 = 1$ and $\tau_2 = 2$, respectively. Community sizes range from small ([10, 20]) and large communities ([20, 100]), the average degree and maximum degree are 20 and 50 respectively. However, for the networks with increasing membership, both the average degree and maximum degree increase as well to ensure that every overlapping node has a reasonable amount of links to each community it is part of. The number of overlapping nodes O_n is set between 0.8 and 1.0 to ensure high amount of overlap in the networks. The number of communities to which each overlapping node belongs is set to 2, unless it is varied. One major drawback of the typical parametrization is that all overlapping nodes have the exact same number of memberships which is unrealistic. While the authors explicitly state, that other distributions can be chosen, we are not aware of any paper or implementation that uses this possibility.

Therefore, we use another type of networks, the CKB networks [CKB$^+$14]. The CKB benchmark generator not only provides a power law distribution of community sizes, but also for the number of communities a node belongs to. The parameters for the CKB networks

Table 5.1: Overview of parameter selection for LFR networks

Description		Mix. parameter	Membership	Overlap
N	number of nodes	5000/50000	5000/50000	5000/50000
k	average degree	20	$20 + O_m \cdot 10$	20
k_{max}	max degree	50	$50 + O_m \cdot 10$	50
C_{min}	min. comm. size	10/20	20	10/20
C_{max}	max. comm. size	50/100	100	50/100
τ_1	degree exponent	1	1	1
τ_2	comm. exponent	2	2	2
μ	mixing parameter	**0.0-1.0**	0.3	0.3
O_n %	overlapping nodes	0.8	1.0	**0.0-1.0**
O_m	comms. per node	2	**1-8**	2

Table 5.2: Overview of parameter selection for CKB networks

Description		Small CKB	Large CKB
N	number of nodes	5000	50000
X_{min}	min. comms per node	1	1
X_{max}	max. comms per node	500	5000
M_{min}	min. comm. size	20	20
M_{max}	max. comm. size	500	5000
β_1	membership exponent	2.5	2.5
β_2	comm. exponent	2.5	2.5
α	edge prob. inside comms.	4	4
γ	edge prob. inside comms.	0.5	0.5
ϵ	num. edges inside ϵ-comm.	0.0004	0.00004

are chosen following the suggestions of [SHW17]. Which are the same as the parameters proposed in the original paper except for an higher number of minimum communities. Note that both the community sizes and communities per node follow a power law distribution with exponents 2.5. To generate these CKB networks, the implementation provided by [SHW17] was used.

Furthermore, we use Erdös-Renyi networks [ER59] as random networks. The Erdös-Renyi network generator only takes two parameters. The amount of nodes N and the edge probability p. Each pair of vertices is then connected to each other with probability p. This results in a network which should not have any community structure.

Real Networks. For real networks with overlapping ground-truth communities, we use four networks from the Stanford Network Analysis Project's collection of datasets[1]:

1. *Amazon product co-purchasing network* is a network where nodes represent products and an edge between product i and j signifies that product i was frequently co-purchased with product j. Each product category defines a ground-truth community.

2. *DBLP collaboration network* is a network where nodes represent authors and an edge between two authors means that they published at least one paper together. A set of authors who published to a certain journal form a ground-truth community.

[1] http://snap.stanford.edu/data

3. *Youtube* and *Live-Journal social networks* are networks where nodes represent users and edges between users represent friendships. The ground-truth communities are user-defined groups.

We use the top 5000 communities for each of the real networks. J. Leskovec et al. [YL13a] define these top 5000 communities by using six community scoring functions, such as conductance and triangle participation ratio. The top 5000 communities show the highest quality for each of the six aforementioned community scoring functions. Note that the ground-truth communities of each real network are not defined by the structure of the corresponding network, but by the meta-data of the nodes. We use community detection algorithms that only use the network structure to detect communities and not the meta-data of the nodes. We refer to these communities as structural communities. Hric et al. [HDF14] state that most scholars assume that the structural communities detected by the algorithms correspond to the communities defined by the meta-data. However, the results of Hric et al. show that the structural communities are substantially different than those defined by meta-data.

5.2 Metrics

Evaluating the quality of a detected cover is not trivial. There are a range of metrics to measure the similarity of partitions of which a few have been extended to covers. In the following, the metrics that were used in this study are presented.

Performance Metrics are metrics which measure the similarity of a detected cover with a reference cover. Two widely used performance metrics are the Normalized Mutual Information (NMI) and the Omega-Index.

1. The *NMI* was first introduced by Fred et al. [AJ] and later extended for covers by Lancichinetti et al. [LFK09, Appendix B.]. The normalization bounds the values of the mutual information to [0,1], where 1 signifies that both covers are identical and 0 that they are independent. However, the implementation provided by Lancichinetti et al. shows some unintuitive behaviour [MGH11] where their implementation can overestimate the similarity of two covers. Therefore, the implementation provided by McDaid et al. [MGH11] is used in this study which fixes this issue.

2. The *Omega Index* was first introduced by Collins et al. [CD88] which is the extension of the *Adjusted Rand Index* (ARI) [HA85] for covers. The Omega Index is based on pairs of nodes that are clustered in the exact same number of communities in both covers. The value of the Omega Index is highest at 1, which indicates perfect matching of the two covers.

3. The *F1 Score* is the harmonic mean of the precision and recall. The F1 Score reaches its best value at 1 and worst value at 0. *Precision* is the fraction of retrieved items that are relevant for the query and defined as: $\mathrm{P}(S', \mathrm{S}) = \frac{|S \cap S'|}{|S'|}$. *Recall* is the fraction of relevant items that are successfully retrieved and defined as: $\mathrm{R}(S', \mathrm{S}) = \frac{|S \cap S'|}{|S|}$. S' and S are the sets of retrieved and relevant items respectively. In this study, we use the F1 Score at community and node level.

   The F1 Score at community level was used in [ELL17]. Each community in a cover is compared to the best-matching reference community in terms of highest F1 Score. The F1 Score to compare two covers is the average over all communities in the detected cover, i.e., $F_1(C', C) = \frac{1}{|C'|} \cdot \sum_{S' \in C'} \max_{S \in C} F_1(S', S)$, where C' is a detected cover and C is a reference cover. The implementation of the F1 Score at community level was provided by NetworKit[2].

[2] https://networkit.github.io/

We use the F1 Score at node level to compare the distribution of the number of communities per node of a detected cover to a reference cover. In this case, S' is the set of the number of communities assigned to each node of the detected cover and S is the set of the number of communities assigned to each node of the reference cover.

Community Scoring Functions quantitatively measure certain properties of communities. Various community scoring functions were presented and evaluated in [YL13a]. In this study, we use five such community scoring functions to measure the structural properties of the detected communities. These functions were also used in [HBG+14]. In the following, we describe them briefly:

1. The *Edge Density* measures the fraction of possible edges that exists in the community and thus how strongly connected the nodes of the community are connected to each other: $\mathrm{F(S)} = \frac{2 \cdot |E_S|}{|V_S| \cdot (|V_S| - 1)}$. An edge density of 1 means that the community is densely connected, whereas an edge density of 0 means that no node is connected to another node inside the community. Therefore, the higher the edge density the better.

2. The *Conductance* normalizes the number of edges to other communities by the total number of edges incident to nodes in the community: $\mathrm{F(S)} = \frac{k_S^{ext}}{k_S}$. A conductance of 0 means that the community is completely isolated from the rest of the graph, whereas a conductance of 1 means that there are only edges between nodes inside the community and nodes outside of the community. Therefore a low conductance is preferred.

3. The *Clustering coefficient* measures the fraction of closed triangles compared to possible triangles: $\mathrm{F(S)} = \frac{|T_{Closed}|}{|T_{Closed}| + |T_{Open}|}$ where T_{Closed} is the set of closed triplets and T_{Open} is the set of open triplets.

4. The *Triangle Participation Ratio* measures the fraction of nodes that belong to a triangle: $\mathrm{F(S)} = \frac{|\{v \in V_S | v \in T \wedge T \in T_{Closed}\}|}{|V_S|}$. The clustering coefficient and the triangle participation ratio are based on the premise that pairs of nodes are with common neighbours are more likely connected to each other. Therefore, a high clustering coefficient and triangle participation ratio are preferred.

5. The *Size* of the community.

Ratio of detected communities to reference communities, for a detected cover and a reference cover, measures the fraction of the number of detected communities to the number of reference communities.

6. Results

In this section, the results of the overlapping community detection algorithms, that were run on various synthetic and real networks, are presented and discussed. For synthetic networks, 10 instances for each set of parameters were generated and for real networks, each algorithm was run 10 times. We allow a maximum run time of 4 hours for each algorithm. The experiments were run on a server consisting of a 4 core Intel Processor (Intel Core i7-2600K CPU @ 3.40GHz) with Hyper-Threading activated and 32 GB of RAM.

First the results of each experiment carried out on the LFR, CKB and Erdös-Renyi networks are presented and discussed. Then the results of each experiment carried out on the real networks are presented and discussed.

6.1 Synthetic Networks

First of all, we take a look at the run time of each algorithm on the LFR networks where the mixing parameter μ and the number of memberships O_m vary from small values to large values and we take also a look at the run time of each algorithm on the CKB networks.

Figures 6.1 and 6.2 show the results for the run time of each algorithm on LFR networks with varying mixing parameter and membership respectively. Looking at the figures, we can see that in general OSLOM, BigClam, InfoMap and MOSES have the highest run time, of which OSLOM and BigClam are the worst. However, the run time of MOSES scales better with the number of nodes in the graph than the other aforementioned algorithms, except for BigClam. BMLPA and GANXiS have a lower run time than the other aforementioned algorithms, however on large networks, they take longer to complete than MOSES. Finally, the Egosplitting algorithms, OLP and GCE have the lowest run times of all the algorithms. Looking at the results in Figure 6.1 and Figure 6.2, we can see that each increase in mixing parameter and membership will result in a higher run time for most algorithms. Figure 6.3 shows the results for the run time of each algorithm on CKB networks. The results are mostly identical to the results for the LFR networks. For small CKB networks, MOSES has the worst run time of all algorithms, however on large networks, OSLOM takes more than four hours, whereas MOSES is able to complete in less than four hours. The run time for GCE is 10 times higher on CKB networks than on LFR networks. Note that InfoMap needs more than 32 GB of RAM on large LFR networks with high membership and on large CKB networks, therefore no results are presented for those cases.

Next the overall performance, measured by NMI, is examined. Looking at the results for the LFR networks with varying overlap (Figure 6.4), both OSLOM and MOSES perform the

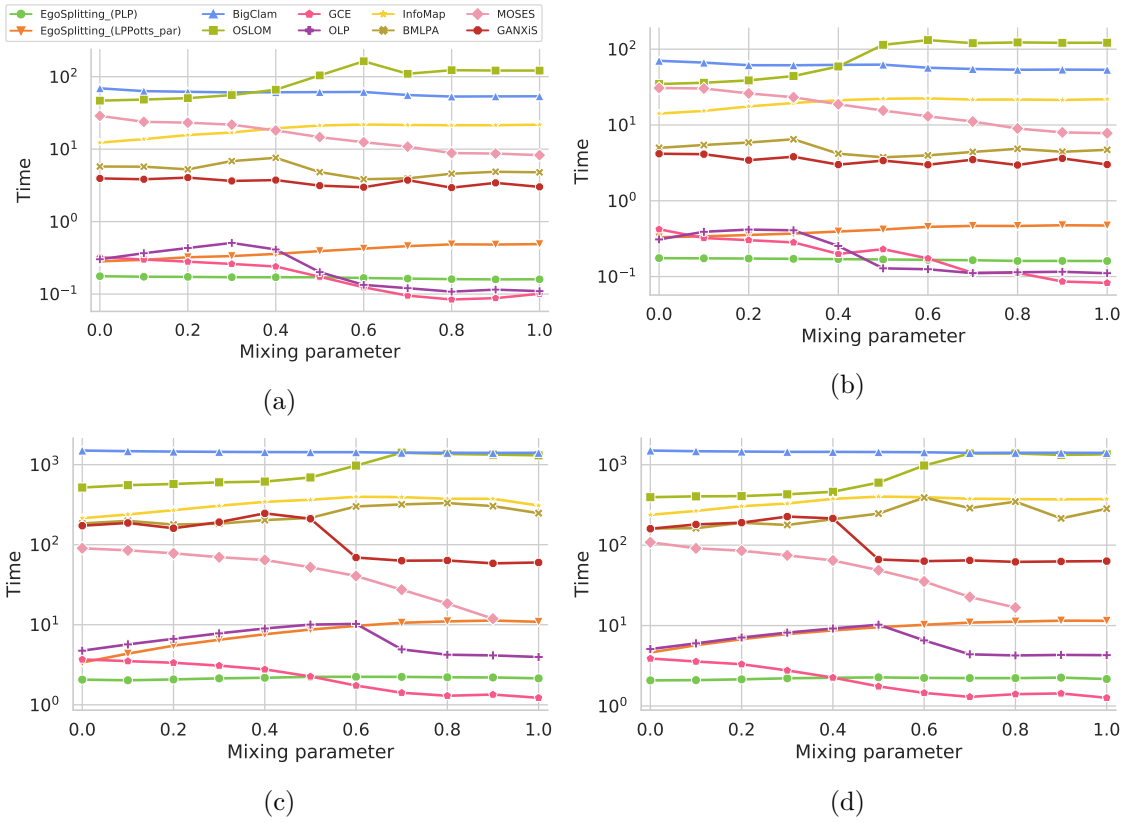


Figure 6.1: The average run time on LFR networks with varying mixing parameter. The parameters are $N = 5000$ (**A**, **B**) and 50000 (**C**, **D**), $k = 20$, $k_{max} = 50$, community sizes = [10, 50] (**A**, **C**) and [20, 100] (**B**, **D**), $\tau_1 = 1$, $\tau_2 = 2$, $\mu =$ **0.0-1.0**, $O_n = 0.8$, $O_m = 2$

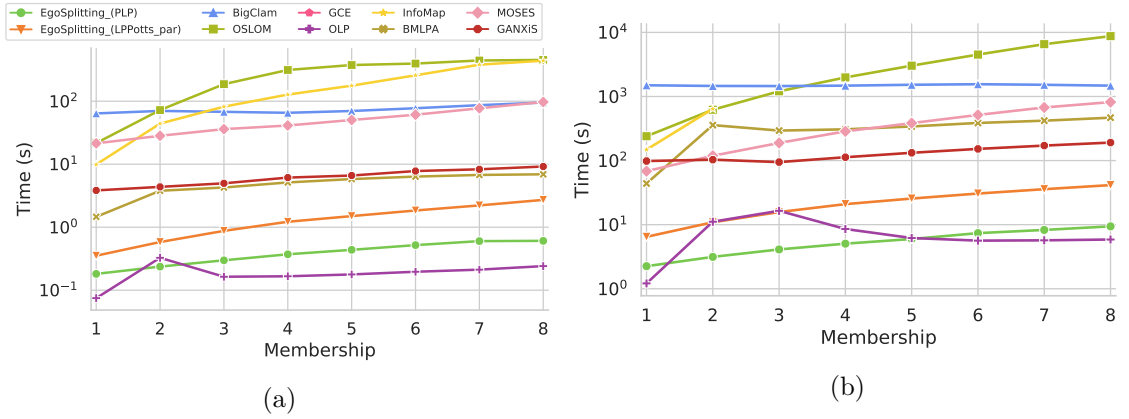


Figure 6.2: The average run time on LFR networks with varying membership. The parameters are $N = 5000$ (**A**) and 50000 (**B**), $k = 20 + O_m \cdot 10$, $k_{max} = 50 + O_m \cdot 10$, community sizes = [20, 100] , $\tau_1 = 1$, $\tau_2 = 2$, $\mu = 0.3$, $O_n = 1.0$, $O_m =$ **1-8**

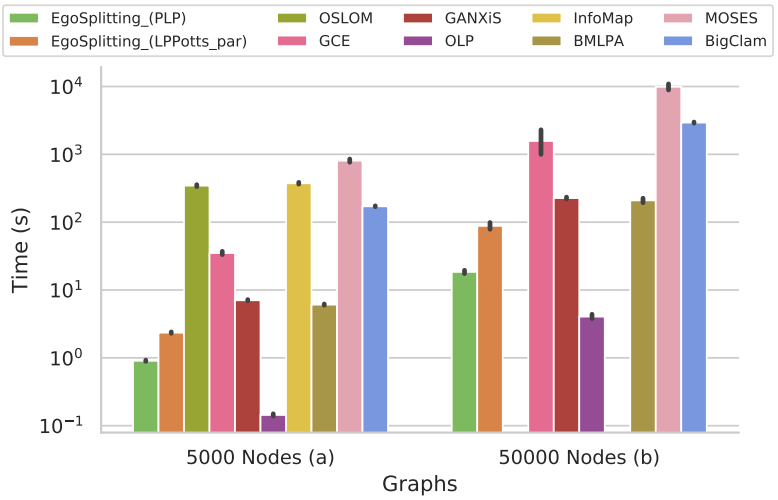


Figure 6.3: Bar plot showing the average run time on CKB networks. The vertical black lines show the standard deviation around the average using error bars. The parameters are $N = 5000$ (**A**) and 50000 (**B**), $X_{min} = 1$, $X_{max} = 500$ (**A**) and 5000 (**B**), $M_{min} = 20$, $M_{max} = 500$ (**A**) and 5000 (**B**)

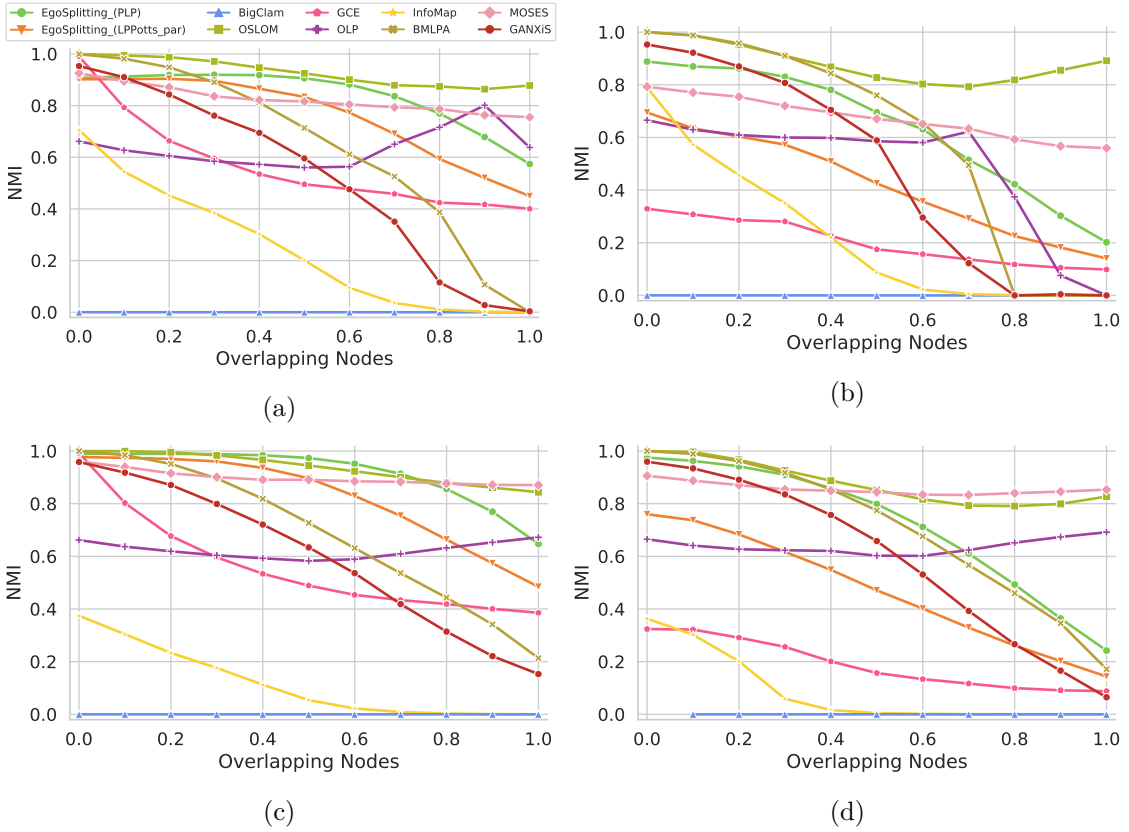


Figure 6.4: The average NMI on LFR networks with varying overlap. The parameters are $N = 5000$ (**A**, **B**) and 50000 (**C**, **D**), $k = 20$, $k_{max} = 50$, community sizes = [10, 50] (**A**, **C**) and [20, 100] (**B**, **D**), $\tau_1 = 1$, $\tau_2 = 2$, $\mu = 0.3$, $O_n =$ **0.0 - 1.0**, $O_m = 2$

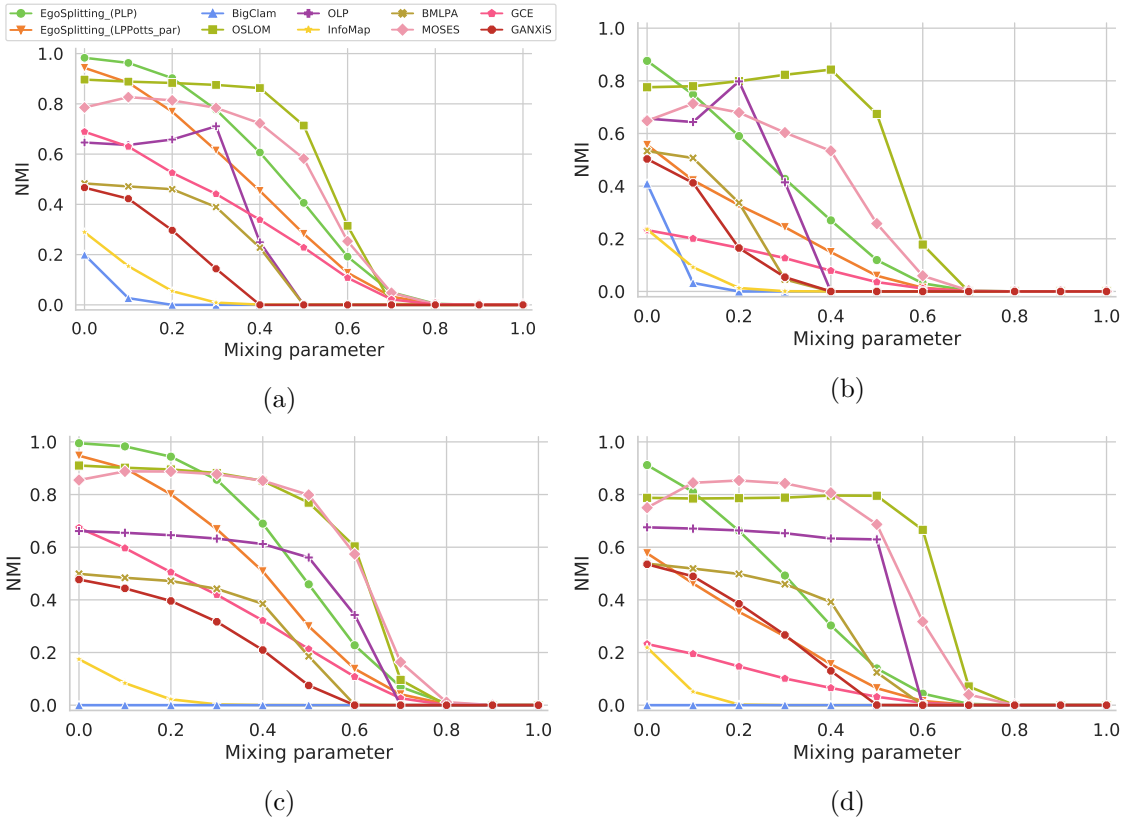


Figure 6.5: The average NMI on LFR networks with varying mixing parameter. The parameters are $N = 5000$ (**A**, **B**) and 50000 (**C**, **D**), $k = 20$, $k_{max} = 50$, community sizes = [10, 50] (**A**, **C**) and [20, 100] (**B**, **D**), $\tau_1 = 1$, $\tau_2 = 2$, $\mu =$ **0.0-1.0**, $O_n = 0.8$, $O_m = 2$

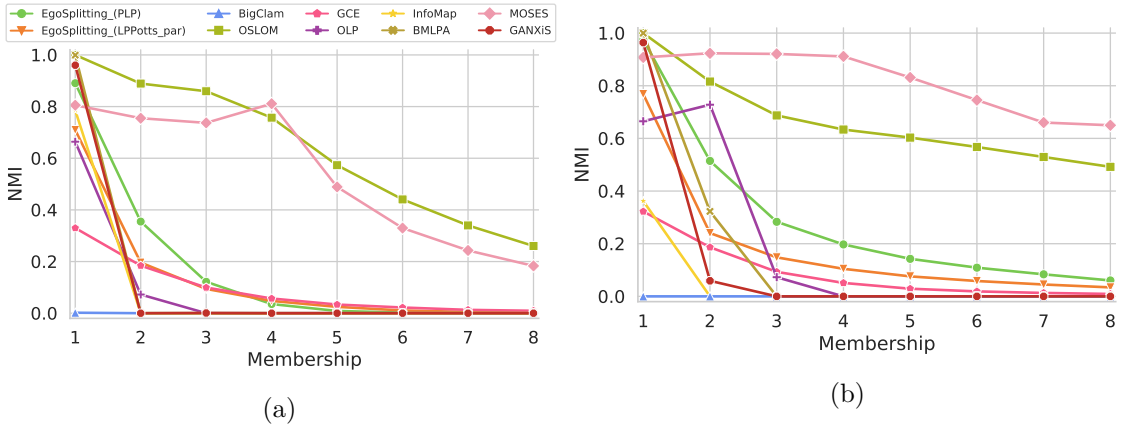


Figure 6.6: The average NMI on LFR networks with varying membership. The parameters are $N = 5000$ (**A**) and 50000 (**B**), $k = 20 + O_m \cdot 10$, $k_{max} = 50 + O_m \cdot 10$, community sizes = [20, 100] , $\tau_1 = 1$, $\tau_2 = 2$, $\mu = 0.3$, $O_n = 1.0$, $O_m =$ **1-8**

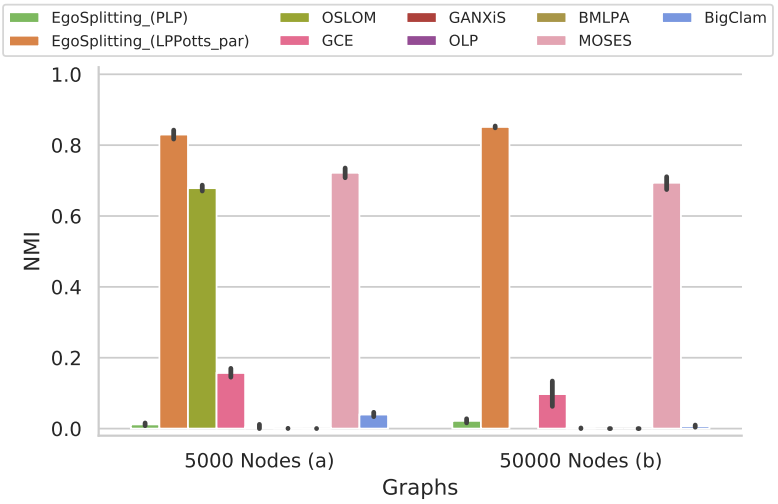


Figure 6.7: Bar plot showing the average NMI on CKB networks. The vertical black lines show the standard deviation around the average using error bars. The parameters are $N = 5000$ (**A**) and 50000 (**B**), $X_{min} = 1$, $X_{max} = 500$ (**A**) and 5000 (**B**), $M_{min} = 20$, $M_{max} = 500$ (**A**) and 5000 (**B**)

best. OSLOM performs better on small networks than MOSES, but on larger networks both algorithm's performance is almost equal. As the number of overlapping nodes increases, the performance of all algorithms suffers moderately to greatly, except for OSLOM and MOSES. An exception are the small networks with big communities where the performance of OSLOM and MOSES decreases moderately. Both EgoSplitting algorithms do not perform well on networks with big communities, but provide acceptable results on networks with small communities. As for the label propagation algorithms, both the performance of GANXiS and BMLPA decline progressively as the number of overlapping nodes increases. OLP's performance, on the other hand, while moderate for low overlap, does not worsen for high overlap. In contrary in 3 out of 4 cases, the performance even improves. GCE's performance is moderate, but does not worsen as fast as for BLMPA or GANXiS, but provides poor results on networks with big communities, due to them being more sparsely connected which also results in less cliques in the network. InfoMap's performance declines similarly to GANXiS and provides worse results on large networks. BigClam does not detect any overlapping communities.

For the LFR networks with varying mixing parameter (Figure 6.5), as expected the larger the mixing parameter, the lower the performance is, especially for a μ greater than 0.5, due to there being less connections inside the communities. On these networks, OSLOM and MOSES perform the best, where OSLOM performs better than MOSES on small networks. However on large networks, MOSES is on par with OSLOM and on large networks with big communities, MOSES even outperforms OSLOM for small values of μ. Both EgoSplitting algorithms also provide good results for networks with small communities. However, as seen in the former experiment, both algorithms do not run well on networks with big communities, as their performance suffers a lot. GCE's performance once again worsens on networks with big communities. Looking at the label propagation algorithms, it is interesting to see that a simple label propagation algorithm, like OLP, outperforms the other label propagation algorithms, BMLPA and GANXiS. The poor performance of BMLPA and GANXiS is due to their poor performance on networks with high amount of overlap. Finally, both InfoMap and BigClam do not perform well, this is also due to their poor performance on networks with high overlap 6.4.

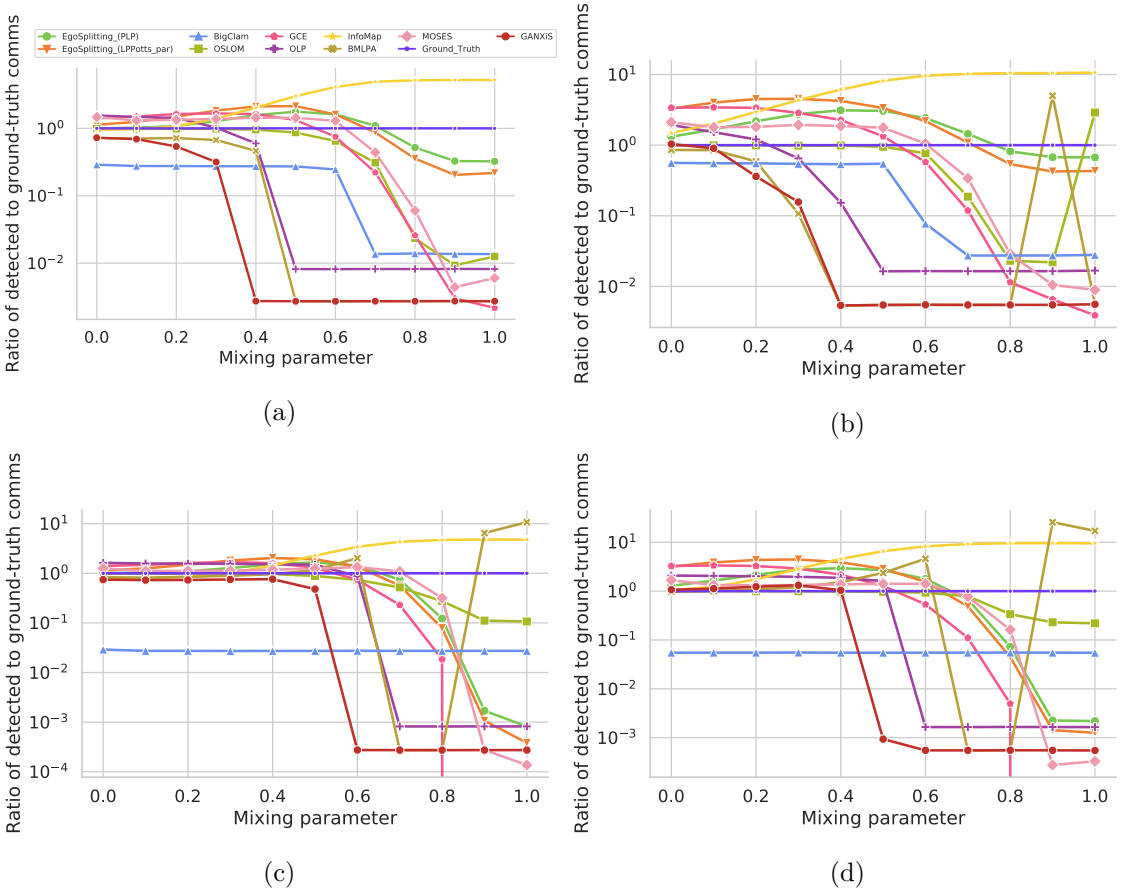


Figure 6.8: The average ratio of detected to ground-truth communities on LFR networks with varying mixing parameter. The parameters are $N = 5000$ (**A**, **B**) and 50000 (**C**, **D**), $k = 20$, $k_{max} = 50$, community sizes = [10, 50] (**A**, **C**) and [20, 100] (**B**, **D**), $\tau_1 = 1$, $\tau_2 = 2$, $\mu =$ **0.0-1.0**, $O_n = 0.8$, $O_m = 2$

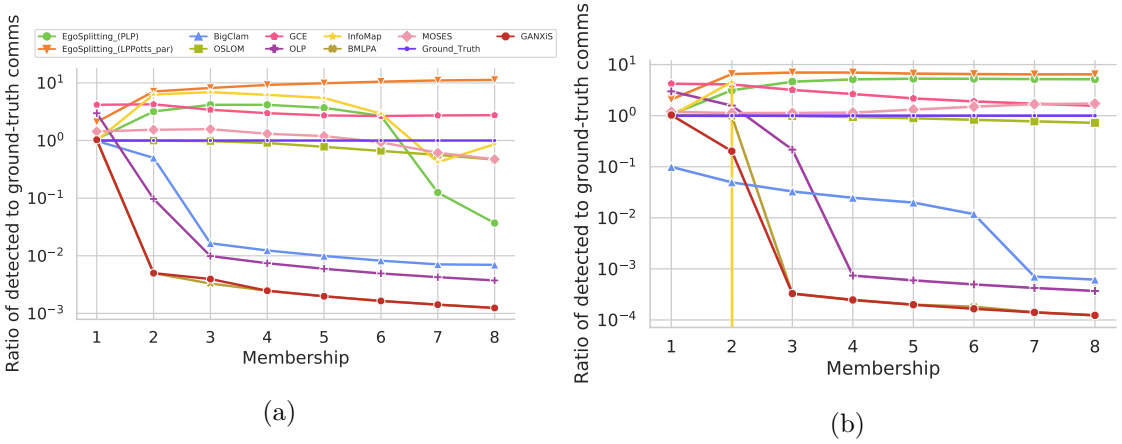


Figure 6.9: The average ratio of detected to ground-truth communities on LFR networks with varying membership. The parameters are $N = 5000$ (**A**) and 50000 (**B**), $k = 20 + O_m \cdot 10$, $k_{max} = 50 + O_m \cdot 10$, community sizes = [20, 100] , $\tau_1 = 1$, $\tau_2 = 2$, $\mu = 0.3$, $O_n = 1.0$, $O_m =$ **1-8**

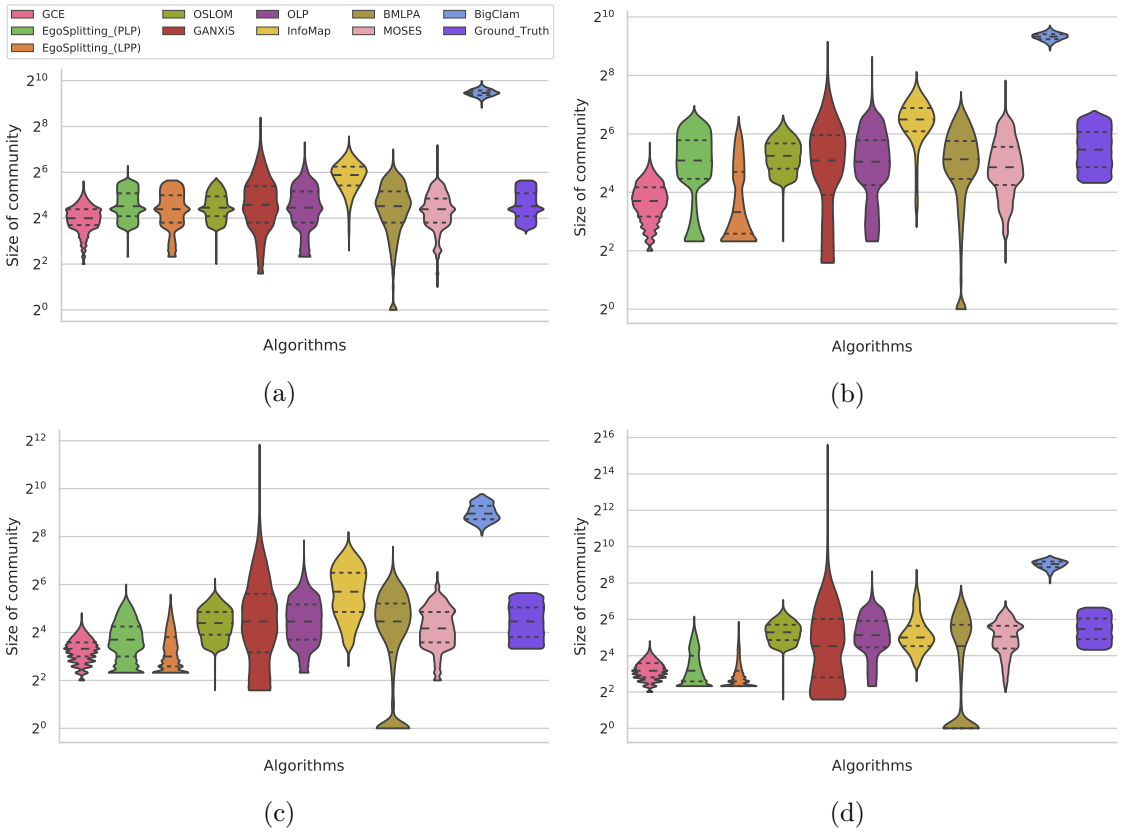
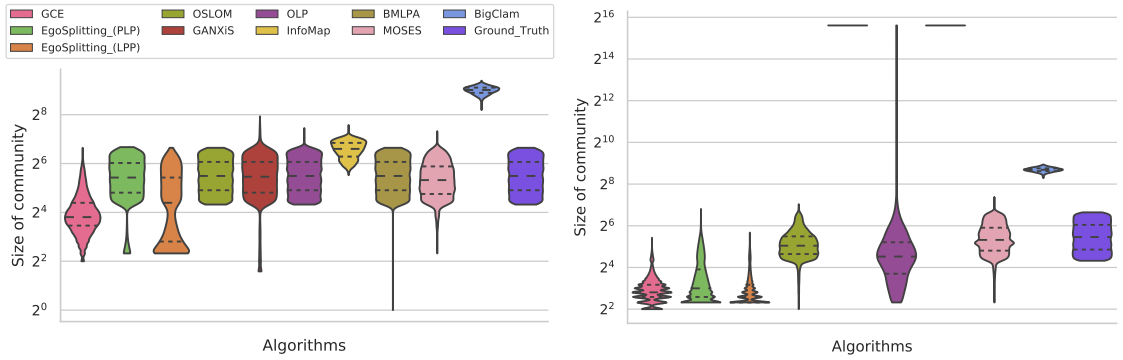


Figure 6.10: Violin plot showing the distribution of the community sizes of the detected and the ground-truth communities on LFR networks with mixing parameter $\mu = 0.0$ (**A**, **B**) and 0.4 (**C**, **D**). The horizontal lines represent the quartiles. The parameters are $N = 50000$, $k = 20$, $k_{max} = 50$, community sizes = [10, 50] (**A**, **C**) and [20, 100] (**B**, **D**), $\tau_1 = 1$, $\tau_2 = 2$, $O_n = 0.8$, $O_m = 2$

The results for the LFR networks with varying membership (Figure 6.6) show clearly that OSLOM and MOSES perform the best, where OSLOM performs slightly better on small networks and MOSES performs much better on larger networks. All the other algorithms perform poorly on both small and large networks. However, the performance for most algorithms, such as OLP, EgoSplitting (PLP and LPPotts_par) and BMLPA improve noticeably on large networks. Maybe large networks are easier as multi-node overlaps between two communities are rarer and the edge probability between nodes not sharing a community decreases. InfoMap, GANXiS, BigClam and BMLPA perform poorly due to their bad performance on networks with high overlap (Figure 6.4).

Figure 6.7 shows the results of each algorithm on CKB networks. We can clearly see that EgoSplitting(LPPotts_par) performs the best. OSLOM and MOSES perform slightly worse and all the other algorithms do not perform well on CKB networks. OSLOM took too long on the large CKB networks. Note that MOSES builds on the Erdös-Renyi model and each community in the CKB networks is modelled as an Erdös-Renyi network. This could explain the good performance of MOSES.

Overall, summarizing the results of the previous experiments, both OSLOM and MOSES show the best performance. However this good performance also comes with a high run time (Figure 6.1 and 6.2). Additionally, both OSLOM and MOSES perform worse on LFR networks with big communities. The EgoSplitting algorithms perform well overall and have a low run time, but their performance suffers severely once they are run on LFR networks



(a) (b)

Figure 6.11: Violin plot showing the distribution of the community sizes of the detected and the ground-truth communities on LFR networks with membership $O_m = 1$ (**A**) and 3 (**B**). The horizontal lines represent the quartiles. The parameters are $N = 50000$, $k = 20 + O_m \cdot 10$, $k_{max} = 50 + O_m \cdot 10$, community sizes = [20, 100] , $\tau_1 = 1$, $\tau_2 = 2$, $\mu = 0.3$, $O_n = 1.0$

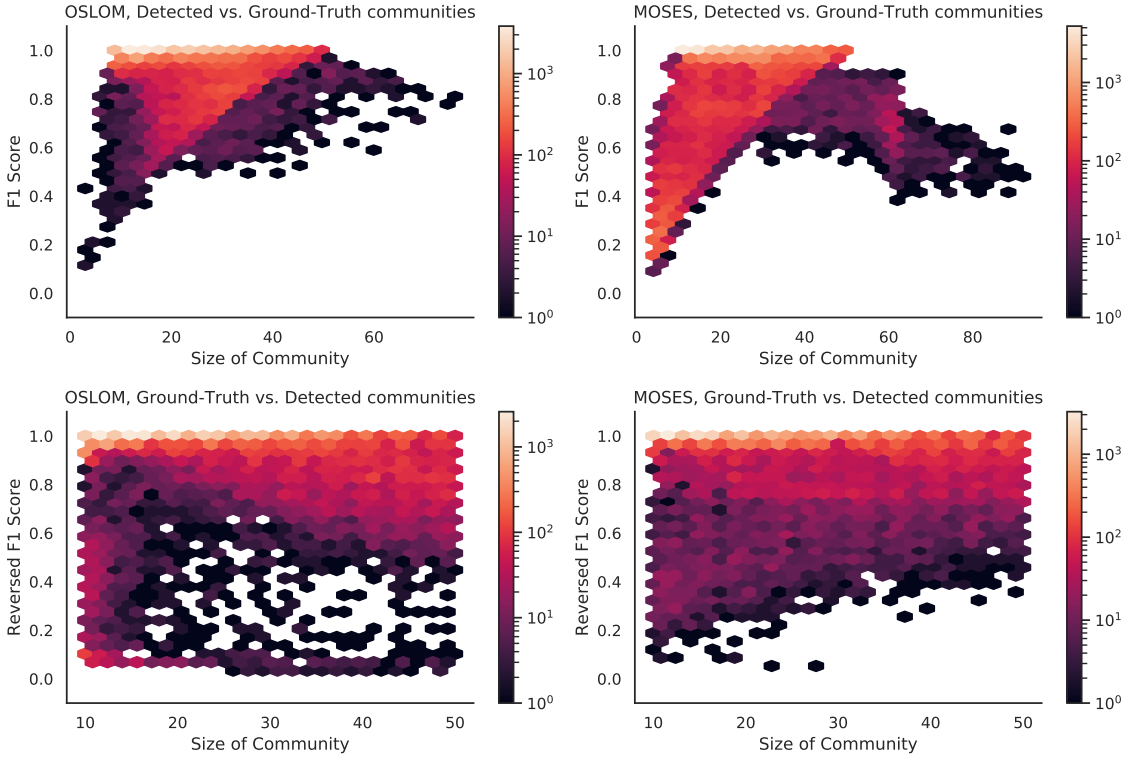


Figure 6.12: The results for OSLOM and MOSES on LFR networks with mixing parameter $\mu = 0.4$. The two upper plots show how well a detected community corresponds to any ground-truth community, by comparing each detected community to each ground-truth community in terms of F1 Score. The two lower plots show how well a ground-truth community is detected, by comparing each ground-truth community to each detected community in terms of F1 Score. The parameters are $N = 50000$, $k = 20$, $k_{max} = 50$, community sizes = [10, 50], $\tau_1 = 1$, $\tau_2 = 2$, $O_n = 0.8$, $O_m = 2$

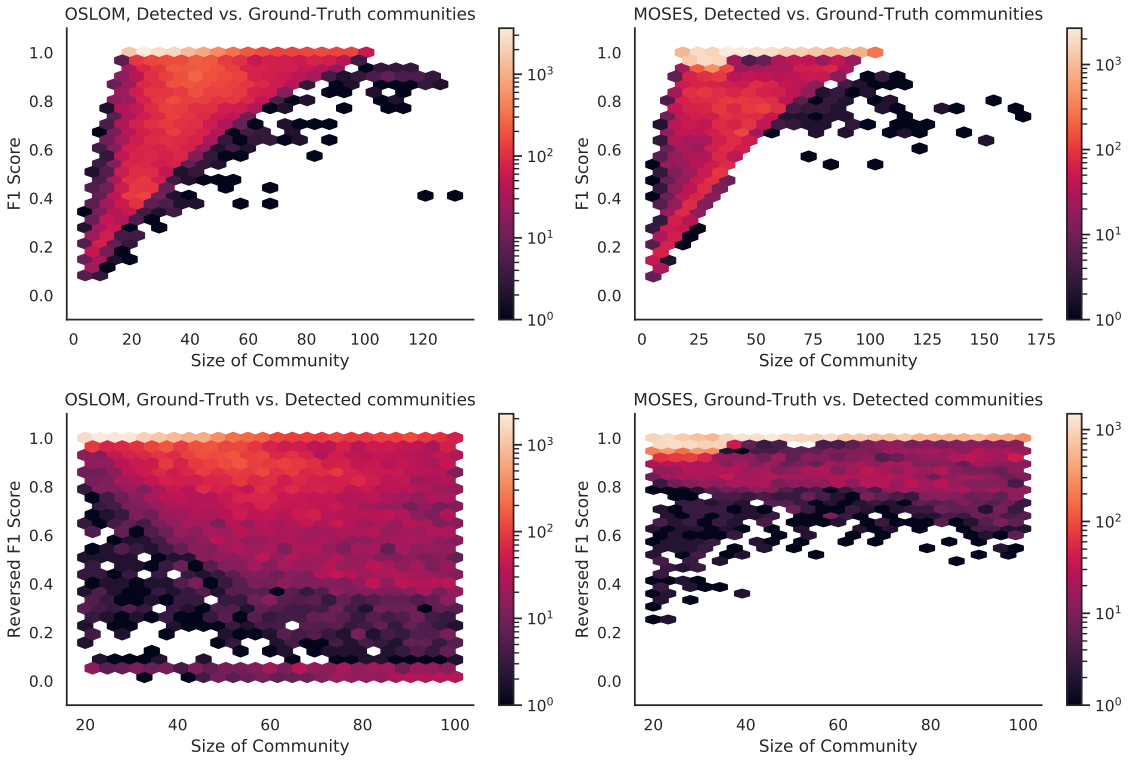


Figure 6.13: The results for OSLOM and MOSES on LFR networks with membership $O_m = 3$. The two upper plots show how well a detected community corresponds to any ground-truth community, by comparing each detected community to each ground-truth community in terms of F1 Score. The two lower plots show how well a ground-truth community is detected, by comparing each ground-truth community to each detected community in terms of F1 Score. The parameters are $N = 50000$, $k = 50$, $k_{max} = 80$, community sizes = [20, 100], $\mu = 0.3$, $\tau_1 = 1$, $\tau_2 = 2$, $O_n = 1.0$

with big communities or on LFR networks with an high number of communities per node. Interestingly, EgoSplitting(LPPotts_par) shows really good performance on CKB networks, whereas EgoSplitting(PLP) performs poorly on them. GCE performs slightly worse than the EgoSplitting algorithms on LFR networks and its performance also suffers on LFR networks with big communities. InfoMap, BigClam, GANXiS and BMLPA perform poorly on LFR networks, due to having significant problems detecting overlapping communities in LFR networks with high overlap. Finally, OLP performs well on LFR networks with high overlap, but is not able to detect any communities on LFR networks with high membership. Another observation made, is that larger LFR networks result in better performance for most algorithms. This is not the case for the CKB networks.

While performance metrics, such as NMI, only provide an overall measure of the algorithm's performance, they do not provide enough information of why the algorithms perform so well or so poorly. Therefore, the results of the complementary experiments are presented and discussed below.

At first, the results for verifying if algorithms detect more or less communities than the ground-truth are presented. Figures 6.8 and 6.9 show the ratio of the detected communities to the known number of ground-truth communities on LFR networks with varying mixing parameter and varying membership, respectively. As already observed in the last experiments, the performance of each algorithm worsens once the mixing parameter is larger than a value of 0.5 and the performance of most algorithms increases on large networks.

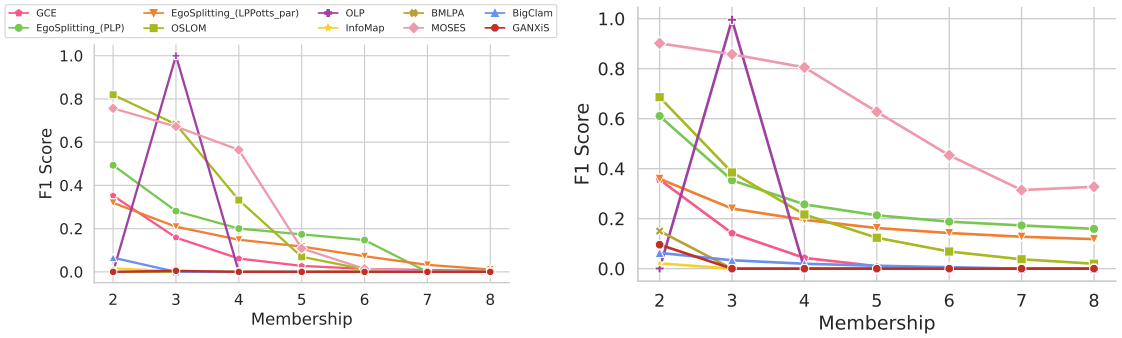


Figure 6.14: Overlapping node detection on LFR networks with varying membership. The F1 Score measures how similar the distribution of the number of communities per node of the detected cover is to the ground-truth cover. The parameters are $N = 5000$ (**A**) and 50000 (**B**), $k = 20 + O_m \cdot 10$, $k_{max} = 50 + O_m \cdot 10$, community sizes = [20, 100] , $\tau_1 = 1$, $\tau_2 = 2$, $\mu = 0.3$, $O_n = 1.0$, $O_m =$ **1-8**

OSLOM is by far the best algorithm to detect a similar number of communities compared to the ground-truth. OSLOM only detects slightly less communities on networks with high memberships. MOSES detects slightly more communities overall. Both EgoSplitting algorithms and GCE tend to detect more communities and their performance worsens on networks with big communities. The worst performing algorithm is InfoMap, which detects way too many communities. BMLPA, OLP and GANXiS detect too few communities in general. However, for large networks with low mixing parameter, OLP and BMLPA tend to detect more communities compared to the ground-truth. Finally, BigClam always produces too few communities.

In general, aside from OSLOM, all the other algorithms either detect too few or too many communities. Extreme cases of detecting too many communities are GCE and InfoMap. Extreme cases of detecting too few communities are BigClam for all networks and all the label propagation algorithms for networks where the average and maximum degree increases as the memberships of each overlapping node increases as well.

Next, the sizes of detected communities are compared to the ground-truth communities to see if some algorithms detect too small or too large communities. Figures 6.10 and 6.11 show the number of detected communities with a certain size of each algorithm. We only show the results for networks with 50000 nodes, because the larger networks result in higher performance for most algorithms. For the networks with varying mixing parameter, we take a look at the community sizes for $\mu = 0.0$, where the NMI is the highest, and for $\mu = 0.4$, where the NMI is lower. For the networks with varying membership, we take a look at the community sizes for $O_m = 1$ and for $O_m = 3$. OSLOM shows the best results, however it detects also a few communities that are too small or too large. MOSES also shows good results, but in general detects many communities that are too small. The EgoSplitting algorithms detect, in all cases, too many too small communities, especially on networks with big communities which explains their low NMI on such networks. The communities GCE detects are usually too small. InfoMap mostly detects communities that are too large. All the label propagation algorithms detect too small communities in networks with $O_m \geq 2$ and varying mixing parameter. GANXiS only detects large communities on networks with $O_m \geq 3$. For $O_m = 1$, their performance is really good compared to the ground-truth.

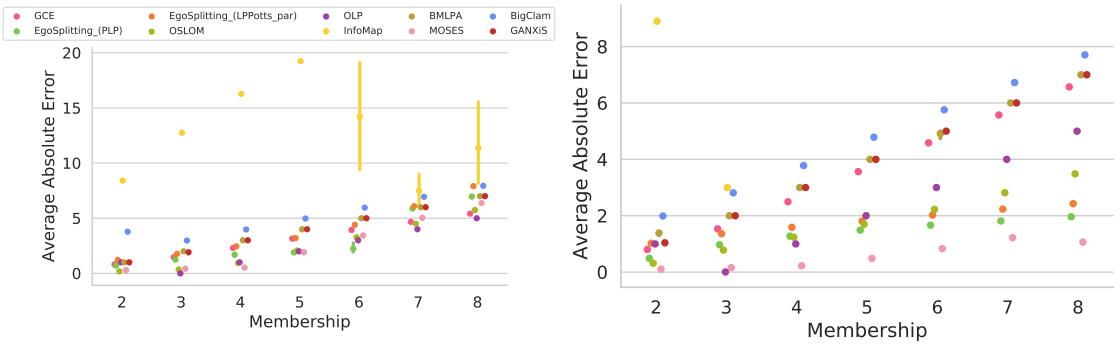
In general, most algorithms are not able to detect communities of similar size to the ground-truth. They either detect many too small communities or many too large communities.

OSLOM is the only algorithm that is able to detect communities sized similarly to the ground-truth, however it still detects a few communities that are too small.

While this experiment gives us a better insight if algorithms are able to detect similar sized communities, it is not able to show us if the detected communities that have the correct size actually fit the ground-truth communities. For example, if we take a look at the detected community sizes of OSLOM and MOSES on the large LFR networks with small communities (Figure 6.10, **C**), we can see that OSLOM performs better than MOSES. Intuitively, this would mean that OSLOM also performs better overall than MOSES. However, if we take a look at the results in terms of NMI (Figure 6.5, **C**, $\mu = 0.4$), we see that both algorithms have a similar NMI. To explain this result, we take a look at the results shown in Figure 6.12. The two upper plots show the results where we compare each detected community to the best-matching ground-truth community in terms of F1-Score. These plots show that, as expected, the communities that are either too small or too big do not fit well with the ground-truth communities and that a lot of the smaller detected communities do not fit well, too. In the two lower plots, we compare each ground-truth community to the best-matching detected community instead. These plots are interesting, because for MOSES, they show us that the majority of the ground-truth communities fit well with the detected communities. However, for OSLOM, we can see that a lot of the smaller ground-truth communities are not recovered. In other words, OSLOM does not detect all of the smaller ground-truth communities. In general, MOSES detects many too small and few too big communities, but it detects most of the ground-truth communities well. OSLOM, on the other hand, does not detect too many too small or too big communities, but it does not detect a lot of the smaller ground-truth communities. Another example can be seen on Figure 6.11 (**B**), where OSLOM and MOSES perform equally well, but looking at the results shown in Figure 6.6 (**B**, $O_m = 3$), we can see that MOSES performs significantly better than OSLOM. Figure 6.13 shows the results of the same experiment, but on a different LFR network. Again, if we take a look at the two lower plots, we can see that MOSES detects most of the ground-truth communities, whereas OSLOM does not detect a lot of the bigger communities, which also explains the lower NMI than MOSES.

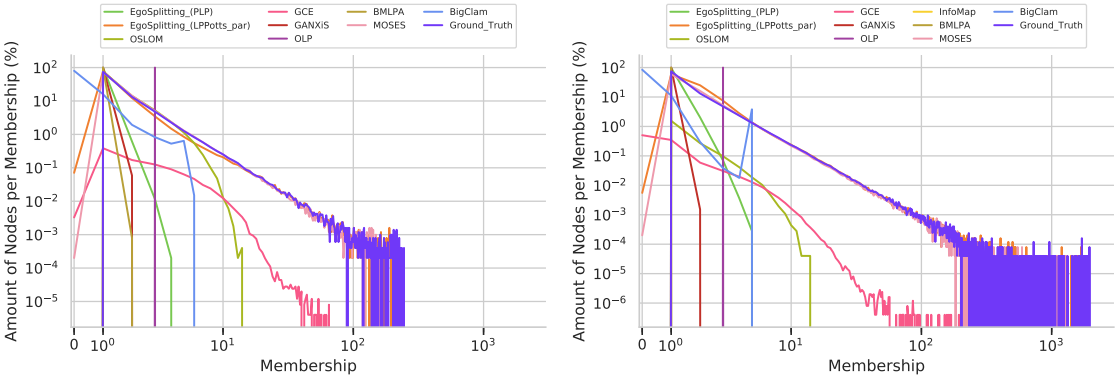
In general, this experiment shows the importance of using the F1-Score at community level. The performance metric NMI and the experiment to see if algorithms detect too many too small or too big communities, are not enough to say how good the algorithms perform. If we use the F1-Score on community level, we gain a more precise insight if the algorithms actually detect communities that are similar to the ground-truth. For example, OSLOM does not detect all the ground-truth communities correctly, whereas MOSES is able to do that. In addition, we see what kind of extra or too few communities are detected.

It is also important to verify if the algorithms are able to detect overlapping nodes in networks with overlapping communities. This is important, because some of the overlapping nodes can act as bridges or communicators between communities. Figure 6.14 shows the results of the algorithms if they are able to detect overlapping nodes on LFR networks with varying membership. The plots show how well each algorithm is at assigning each node the right number of communities. The higher the F1 Score, the better the algorithm is at assigning each node the right number of communities. In general, the ability to detect overlapping nodes degrades as the number of communities per node increases and as seen before the performance increases overall on larger networks. MOSES performs by far the best which is also reflected by the high NMI values seen in Figure 6.6. OLP performs the worst, because it puts each node into three communities. This also explains the high F1 score for networks with membership 3. OSLOM and the EgoSplitting algorithms perform slightly better than the other algorithms, aside from MOSES, however their ability to detect the overlapping nodes correctly is not that good for networks with high memberships. GCE,



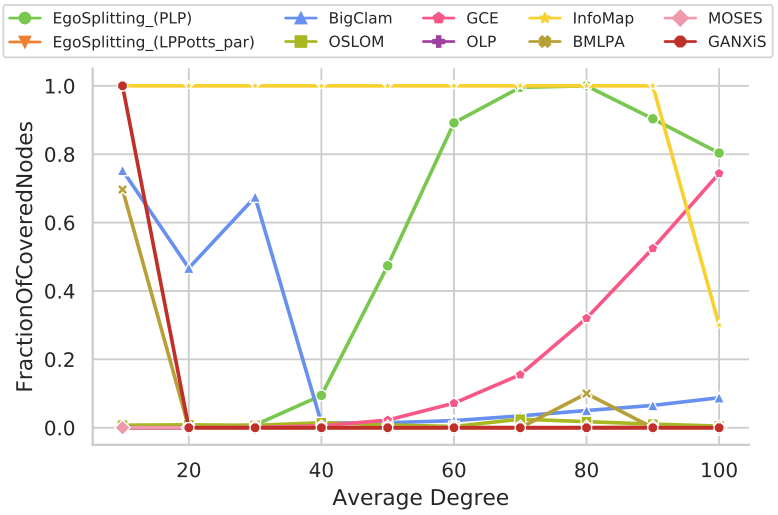
(a) (b)

Figure 6.15: The average absolute error of detecting the correct number of memberships per node on LFR networks with varying membership. The points at each tick are separated to account for better visibility. The vertical black lines show the standard deviation around the average using error bars. The parameters are $N = 5000$ (**A**) and 50000 (**B**), $k = 20 + O_m \cdot 10$, $k_{max} = 50 + O_m \cdot 10$, community sizes = [20, 100] , $\tau_1 = 1$, $\tau_2 = 2$, $\mu = 0.3$, $O_n = 1.0$, $O_m =$ **1-8**



(a) (b)

Figure 6.16: Overlapping node detection on CKB networks. The parameters are $N = 5000$ (**A**) and 50000 (**B**), $X_{min} = 1$, $X_{max} = 500$ (**A**) and 5000 (**B**), $M_{min} = 20$, $M_{max} = 500$ (**A**) and 5000 (**B**)



(a)

Figure 6.17: The fraction of nodes belonging to communities with more than one and less than 5000 nodes. The experiment was carried out on Erdos-Renyi networks with parameters: $N = 5000$ and $p = \frac{\mathbf{a} \cdot 100}{N}$, where $a \in \{10, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$.

GANXiS, BMLPA, InfoMap and BigClam, all have difficulties to detect the overlapping nodes correctly.

Figure 6.15 shows the results of the experiment where the absolute error of memberships of each node is measured and then averaged over all nodes inside the graph. The lower the average absolute error, the better the algorithm is at detecting the correct memberships for each node. In general, we can see the higher the number of memberships, the harder it is for the algorithms to detect the correct number of memberships for each node. We can clearly see that InfoMap is performing poorly and is not able to detect the correct number of memberships for each node. Overall, MOSES shows the best performance and is able to detect the correct number of memberships for each node for LFR networks with a low number of memberships per overlapping node. All the other algorithms have difficulties of detecting the correct number of memberships for each node, even on LFR networks with a low number of memberships for each overlapping node.

Figure 6.16 shows the results of the algorithms on CKB networks. The plot shows if the algorithms are able to assign each node the correct number of communities insofar that at the end the distribution of the number of communities per node follows a power law distribution. Looking at the results, we can see that MOSES and EgoSplitting(LPPotts_par) are able to detect a power law distribution of the number of communities per node similar to the ground-truth. This also reflects the high NMI for both algorithms (Figure 6.7). An exception is OSLOM, which is not able to detect a power law distribution of the number of communities per node, but still has a high NMI compared to the other algorithms. All the other algorithms are not able to detect a power law distribution of the memberships.

In general, summarizing the results for the LFR and CKB networks, we can see that MOSES obtains the best results and that most of the other algorithms are not able to detect the correct number of communities to which each overlapping node belongs to.

In the experiments presented so far, we only tested whether or how good the algorithms are able to detect overlapping communities. However, it is also important to see if the algorithms recognize an absence of community structure in networks which do not hold

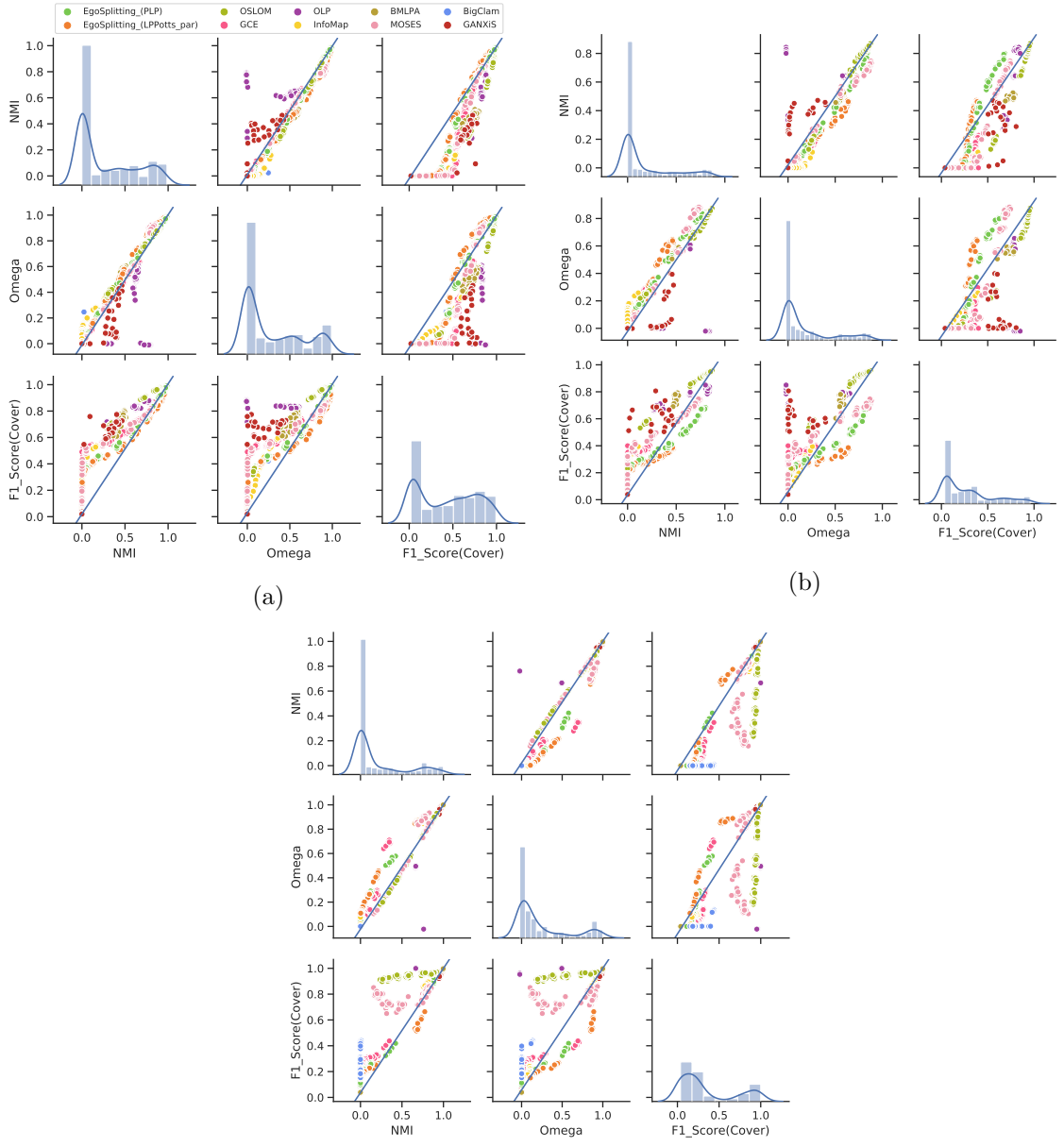


Figure 6.18: Correlation of NMI, Omega-Index and F1-Score for covers on LFR networks with varying mixing parameter and membership. The parameters are $N = 5000$, $k = 20$ (**A**, **B**) and $20 + O_m \cdot 10$ (**C**), $k_{max} = 50$ (**A**, **B**) and $50 + O_m \cdot 10$ (**C**), community sizes = [10, 50] (**A**) and [20, 100] (**B**, **C**), $\tau_1 = 1$, $\tau_2 = 2$, μ = 0.0-1.0 (**A**, **B**) and 0.3 (**C**), $O_n = 0.8$ (**A**, **B**) and 1.0 (**C**), $O_m = 2$ (**A**, **B**) and 1-8 (**C**)

any community structure. For that, we measure the fraction of nodes that belong to communities with more than one and less than 5000 nodes, which is similar to what the authors of [LRRF11] did. For this experiment we use Erdös-Renyi networks, which are random networks, so they, in general, do not hold a community structure. Preferably, the algorithms detects a cover where each node is also a community or a cover where each node belongs to the same community. If an algorithm finds many communities with at least two nodes and smaller than 5000 nodes, then the algorithm is not able to detect the absence of community structure. Figure 6.17 shows the result of the experiment, where the lower the value the better the result. In that case, we can see that OSLOM, MOSES, EgoSplitting(LPPotts_par) and OLP are able to detect the absence of community structure within random networks. BMLPA and GANXiS are also able to detect the absence of community structure, however not for a low average degree $k = 10$. BigClam has difficulties recognizing the absence of community structure in networks with an average degree of less than 40. The ability to recognize the absence of community structure worsens for EgoSplitting(PLP) and GCE once the average degree increases. The overlapping version of InfoMap fails to recognize the absence of community structure in random networks.

Finally, Figure 6.18 shows the results, in terms of NMI, Omega-Index and F1-Score for covers, of each algorithm on small LFR networks with varying mixing parameter and membership. Ideally, the points seen on each plot should stick to the diagonal line which means while three different approaches to compare two covers are used, they return similar results. However, as we can see, this is not always the case. If we compare the NMI to the Omega-Index, we can see that the points stick to the diagonal line for the majority of points, but some outliers exist, such as the results of OLP, which show a negative Omega-Index or an Omega-Index of zero and a non-zero NMI. OLP detects covers, where each pair of nodes in the detected cover and in the ground-truth cover belong to a different number of communities. This results in a negative or very low Omega-Index. If we compare the F1-Score for covers to the other metrics, we can clearly see that a lot of the points do not follow the diagonal line. In other words, while there exist different metrics to compare two covers, one has to be aware that they do not return identical results. Important to note here is that the F1 Score compares each detected cover if it corresponds to any ground-truth and then takes the average. In other words, if all the detected communities correspond to the same ground-truth this would still result in a high F1 Score, even though the other ground-truth communities were not recovered. This also explains the higher F1 Score compared to the NMI, as the detected communities do not have to correspond to all the ground-truth communities. To compensate for this shortcoming, one needs to use the F1 Score to compare each ground-truth community to the detected communities. This shows which ground-truth communities are recovered and which are not.

In Table 6.1 we summarize all the results and rank the algorithms based on their performance on each conducted experiment. We took the average of each algorithm's results for each experiment and ranked each algorithm based of the averages. For the experiments carried out on the LFR networks with varying mixing parameter, we only take the average of the results up to a mixing parameter of 0.6. For the overlapping node detection on CKB networks, we ranked each algorithm if they output a power law distribution of the number of communities per node. For the community sizes, we put each algorithm in one of four categories. From few too small or too large detected communities to many too small or too large detected communities. To get the global rank, we take the average over each individual rank. Overall, OSLOM and MOSES perform the best on synthetic networks. OSLOM detects a number of communities 6.8 and a community size distribution similar to the ground-truth 6.10, but fails to detect some ground-truth communities 6.12. MOSES is better at detecting the overlapping nodes correctly 6.14. EgoSplitting(LPPotts_par) detects overlapping communities reasonably well on LFR networks with few communities per node.

Table 6.1: Ranking of all algorithms

Rank	Algorithm	Avg.NMI				Avg. rat. det. comms		Ov. node det.			Erdös	Comm. Size
		vr. O_n	vr. μ	vr. O_m	CKB	vr. μ	vr. O_m	$F1$	err.	CKB		
1	OSLOM	0.91	0.77	0.65	0.68	0.95	0.84	0.25	2.12	−	0.01	++
2	MOSES	0.81	0.68	0.68	0.72	1.44	1.24	0.46	1.58	+	0	+
3	EGO(LPP)	0.62	0.43	0.16	0.83	1.60	7.37	0.16	2.82	+	0	−−
4	OLP	0.60	0.48	0.14	0	1.19	0.49	0.14	2.29	−	0	−
5	EGO(PLP)	0.78	0.59	0.24	0.01	1.80	3.37	0.24	2.17	−	0.52	−−
6	BMLPA	0.66	0.30	0.15	0	0.93	0.19	0.01	4.02	−	0.08	−−
7	GCE	0.38	0.27	0.09	0.13	1.24	2.95	0.16	3	−	0.19	−−
8	GANXiS	0.55	0.22	0.13	0	0.34	0.14	0	4	−	0.1	−−
9	BigClam	0	0.02	0	0.04	0.21	0.11	0.04	5.01	−	0.22	−−
10	InfoMap	0.17	0.05	*	*	3.04	*	0	9.39	−	0.93	−−

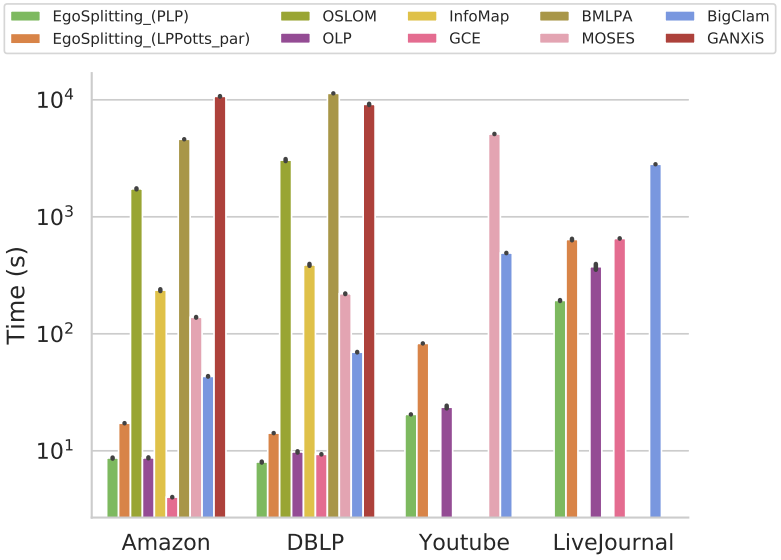


Figure 6.19: Bar plot showing the average run time on the real networks: Amazon, DBLP, Youtube and LiveJournal with the top 5000 communities. The vertical black lines show the standard deviation around the average using error bars.

However, as the number of communities per node increases, the algorithm's performance deteriorates fast. This also counts for EgoSplitting(PLP). Both algorithms also detect too many communities in general. On the other hand, though, EgoSplitting(LPPotts_par) performs the best on CKB networks, even better than OSLOM and MOSES. However, EgoSplitting(LPPotts_par) is not able to detect the absence of community structure in random networks without community structure 6.17. OLP performs reasonably well too on LFR networks with few communities per node, but once again, as the number of communities per node increases, its performance worsens fast. OLP is also not able to detect any overlapping communities in CKB networks. OLP performs in general as good as MOSES at detecting a community size distribution similar to the ground-truth. However, on networks with a high number of communities per node, OLP performs worse 6.11. The performance of the other algorithms, GCE, GANXiS, BMLPA, BigClam, and InfoMap, is poor. This is due to all these algorithms not being able to detect any overlapping communities in LFR networks with a high number of overlapping nodes.

6.2 Real Networks

First of all, we take a look at the run time of each algorithm on the real networks. The results are shown in Figure 6.19. Note that some algorithms took longer than four hours to complete on some of the real networks and are thus omitted. The algorithms, OLP, GCE and the EgoSplitting algorithms, that have the lowest run time on synthetic networks, also have the lowest run time on the real networks. The other label propagation algorithms, BMLPA and GANXiS, surprisingly have a significantly higher run time than OSLOM and MOSES. InfoMap has a low run time, but is not able to run on the larger real networks, Youtube and LiveJournal, because InfoMap needs more than 32 GB of RAM to complete on these real networks. Interestingly, BigClam, which shows a high run time in our experiments on synthetic networks, has a lower run time than most of the other algorithms, such as OSLOM, MOSES, InfoMap and BMLPA.

Next, we take a look at how well the overlapping community detection algorithms recover the ground-truth communities on the real networks. Figure 6.20 shows the results of the NMI for each algorithm on each real network. Additionally, we have also removed the

communities of size less than 5, as those small communities are less informative. As we can see, none of the algorithms are able to detect the correct community structure of the real networks. However, a low NMI does not necessarily mean that the algorithm is not able to detect any ground-truth communities at all. We verify this for the four best performing algorithms in terms of NMI on the real network Amazon. Figure 6.21 shows how well a detected community corresponds to a ground-truth community by using the F1-Score to compare the communities. As we can see, the majority of the detected communities are not similar to any ground-truth community, but a few detected communities exist that are similar to a ground-truth community. Figure 6.22 shows how well each ground-truth community is recovered by the algorithm by using the F1-Score to compare the communities. We can see that a lot of the ground-truth communities are not recovered, only a few ground-truth communities are recovered by the algorithms. Even though the quality of the detected covers is poor, the algorithms are still able to recover some of the ground-truth communities. The EgoSplitting algorithms fail to detect the smallest communities, because they discard communities of size 4 or less. By removing the smaller communities, the NMI generally does not improve. The NMI only slightly worsens for OSLOM and MOSES as they detect some of the smaller communities 6.21. Figure 6.23 shows the structural properties of the detected communities that have a F1 Score equal or higher than 0.6. In other words, these are the detected communities that correspond the best with a ground-truth community. As we can see all the communities are well separated from the rest of the network, but are not that densely connected internally. Our results match those of Hric et al. [HDF14], where they show that the algorithms, they used, are only able to recover few of the ground-truth communities of the Amazon, DBLP, Youtube and LiveJournal networks. They used nine algorithms of which we use three (GANXiS, GCE and InfoMap). However, for two of the algorithms (DEMON, COPRA), we use EgoSplitting and BMLPA, which have been shown that they perform better.

We also take a look at the community sizes of the detected communities, to verify if the algorithms detect either too small or too big communities. Figure 6.24 shows the size of each community detected by each algorithm. The detected community size distribution of each algorithm is in general different than the community size distribution of the ground-truth. GCE, InfoMap and BMLPA detect communities that are too small. The EgoSplitting algorithms and OLP do not detect the smallest communities for most of the real networks, due to them discarding communities of size 4 or less. OSLOM, GANXiS and OLP detect too many too large communities. BigClam detects communities that are too large. For the Amazon network, the community size distribution of MOSES is similar to the ground-truth.

Finally, we measure the structural properties of all detected and all ground-truth communities. The results are shown in Figures 6.25, 6.26, 6.27, 6.28 for the Amazon, DBLP, Youtube and LiveJournal network, respectively. First, we take a look at the structural properties of the detected covers of each algorithm. OSLOM, GANXiS, BigClam and OLP detect many large communities which are for the most part good separated from the rest of the network, but not internally dense. The EgoSplitting algorithms and MOSES detect more small networks, which are not well separated from the rest of the network, but for the most part internally dense. BMLPA and InfoMap detect many communities that are well separated from the rest of the network, this is due to them detecting many communities of size 1, which are single nodes.

Next, we take a look at the structural properties of the ground-truth communities of each real network. The top 5000 communities of the Amazon network are all very well separated from the rest of the network. Few of the communities are sparsely connected internally. A large number of the top 5000 communities of the DBLP and the LiveJournal network are not well separated from the rest of the network, but are in general densely connected internally. The quality of the communities of the Youtube network is poor in terms of

structural properties. Many of Youtube's communities have a high edge density, because a lot of its communities are of size 2, which is simply an edge and results automatically in a edge density of 1. If we remove the communities of size less than 5, we can see that most communities in Youtube's network are not internally dense. If we combine these results with the results of the overall performance of each algorithm, we see that the performance of each algorithm is significantly better on the Amazon and DBLP networks and slightly better on the LiveJournal network than on the Youtube network. The quality of the communities of the Amazon, DBLP and LiveJournal networks is also significantly better than the quality of the communities of the Youtube network. These results match those of Hric et al. [HDF14], where the authors state that the "results depend more on the network than on the specific method adopted." In our case, the algorithms are able to detect more communities similar to the ground-truth communities in real networks with communities that show good structural properties.

Finally, we take a look if there are inconsistencies between the results of the real networks and those of the synthetic networks. In terms of NMI, while OSLOM and MOSES perform the best on synthetic networks, they do not perform the best on real networks. Not only do the EgoSplitting algorithms perform better in terms of NMI, they also have a lower run time than most of the algorithms used. While OSLOM is able to detect the smaller communities in synthetic networks, it detects too many of the smaller communities in the real networks. InfoMap mostly detects large communities on synthetic networks, but on real networks, it mostly detects communities of size 1.

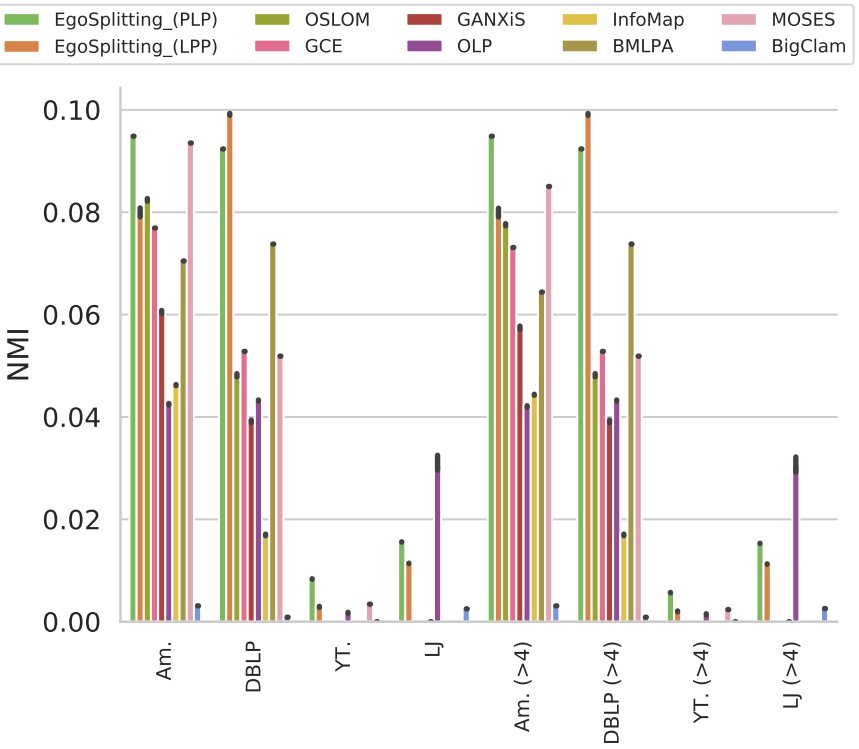


Figure 6.20: Bar plot showing the average NMI on the real networks: Amazon, DBLP, Youtube and LiveJournal with the top 5000 communities and the same networks with communities of size greater than 4. The vertical black lines show the standard deviation around the average using error bars.

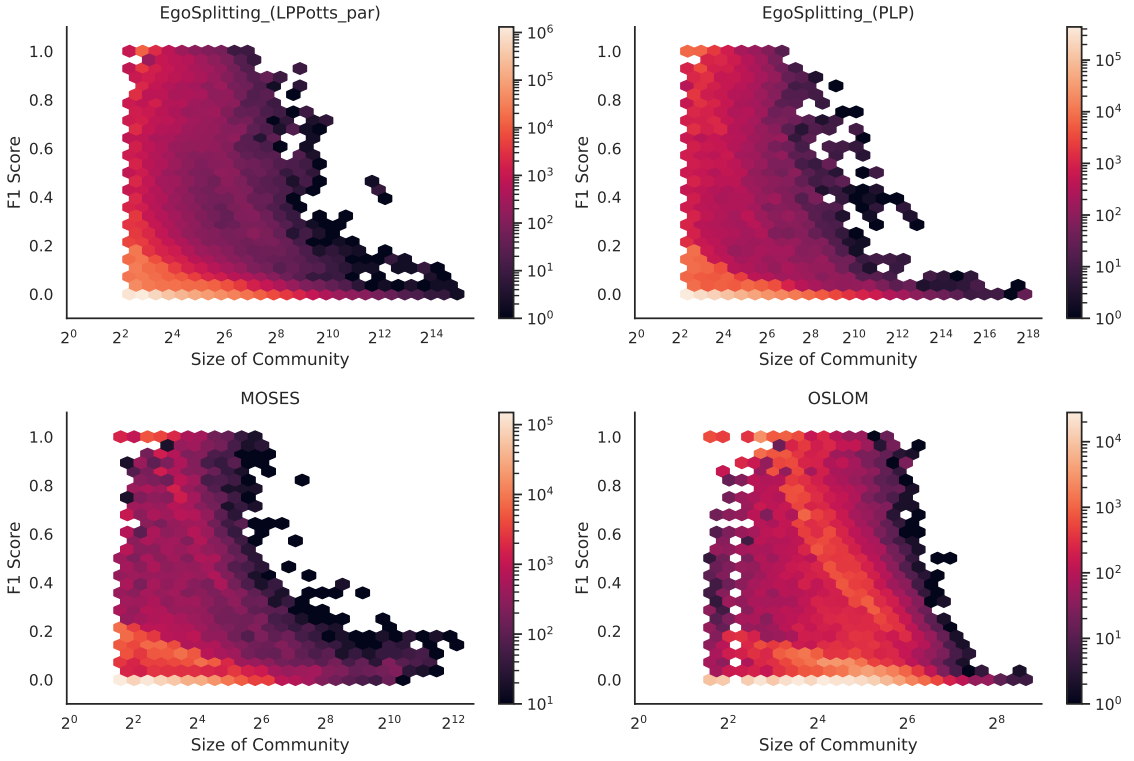


Figure 6.21: The plots show the density of how well the detected communities correspond to any ground-truth community, by comparing each detected community to each ground-truth community in terms of F1 Score on the Amazon network with the top 5000 communities.

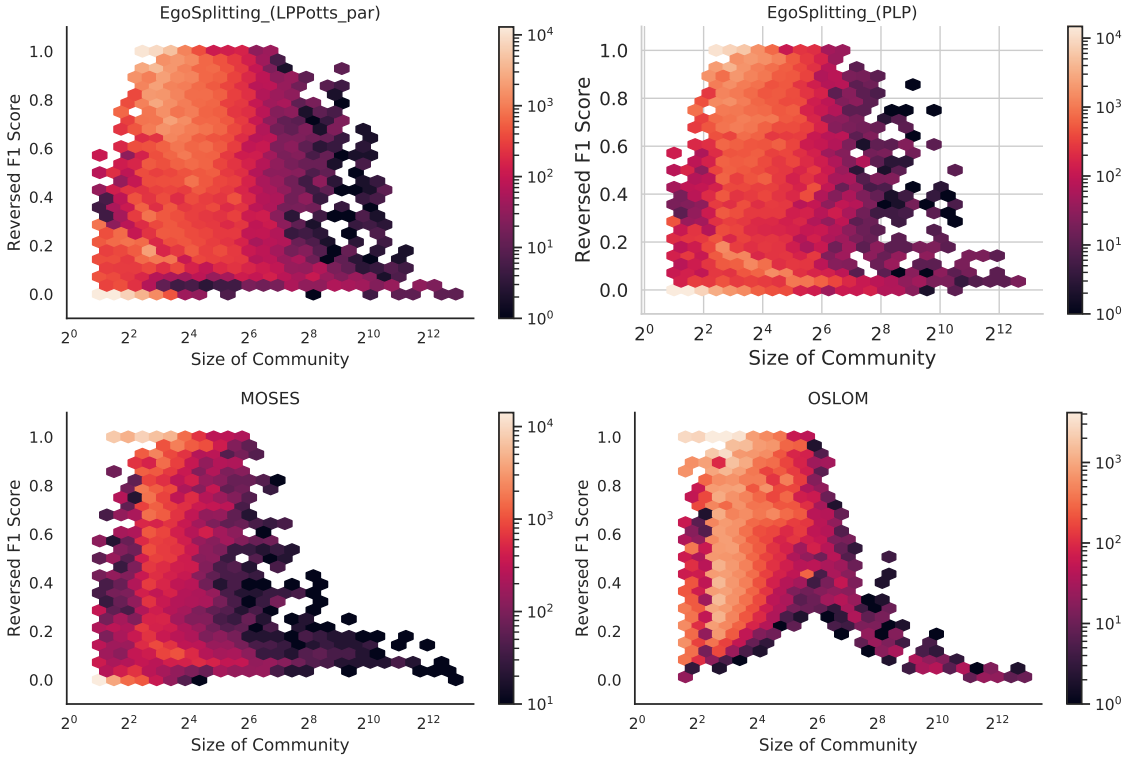


Figure 6.22: The plots show the density of how well the ground-truth communities are detected, by comparing each ground-truth community to each detected community in terms of F1 Score on the Amazon network with the top 5000 communities.

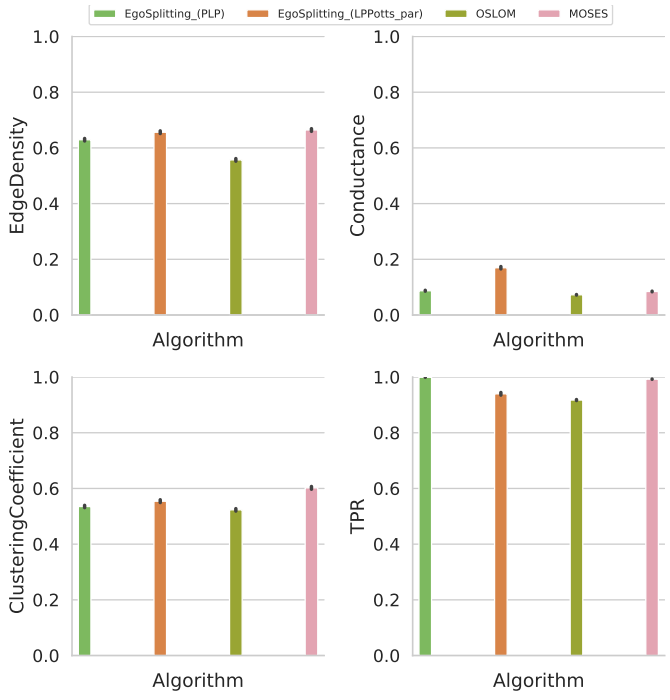


Figure 6.23: Bar plots showing the structural properties of the detected communities that have a F1 Score equal or higher than 0.6 on the Amazon network with the top 5000 communities. The algorithms shown, are the best performing in terms of NMI. The vertical black lines show the standard deviation around the average using error bars.

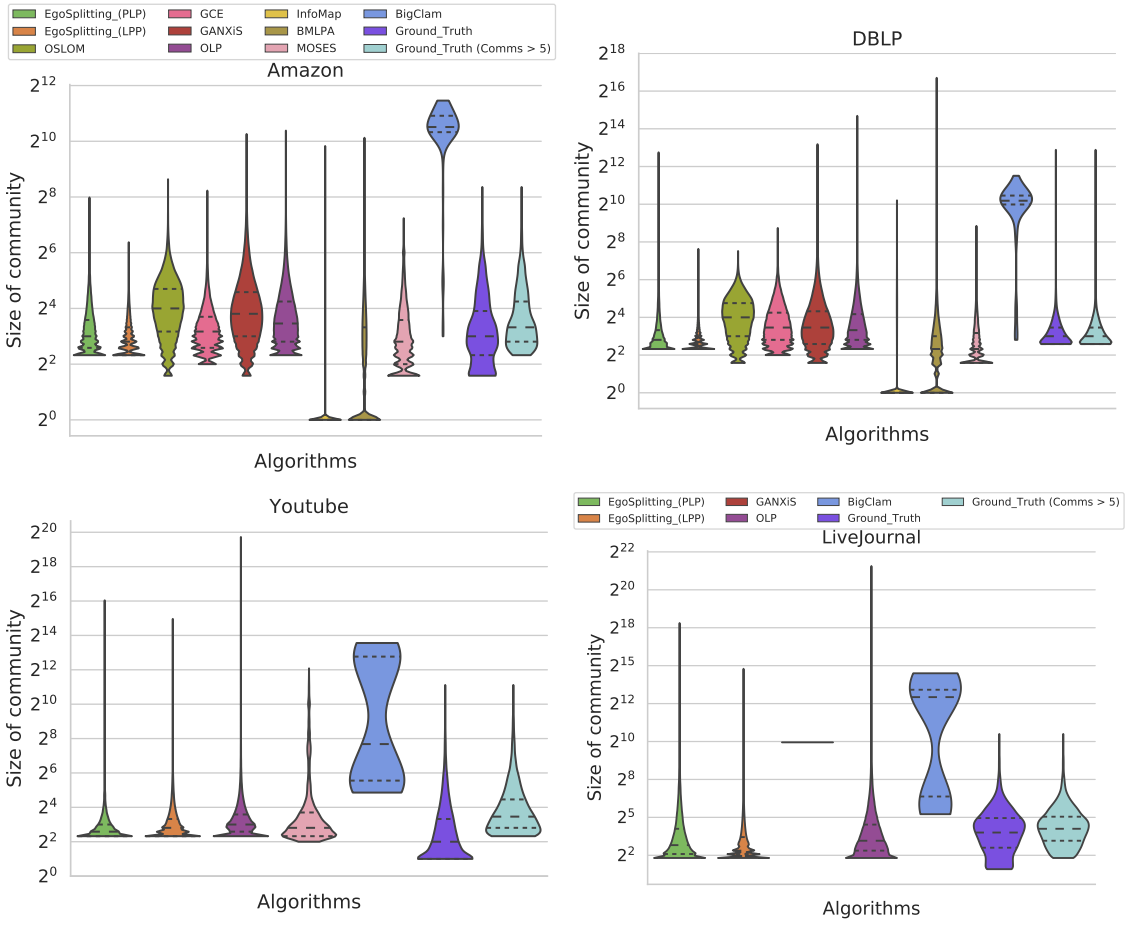


Figure 6.24: The community size distribution of each algorithm on each real network with the top 5000 communities.

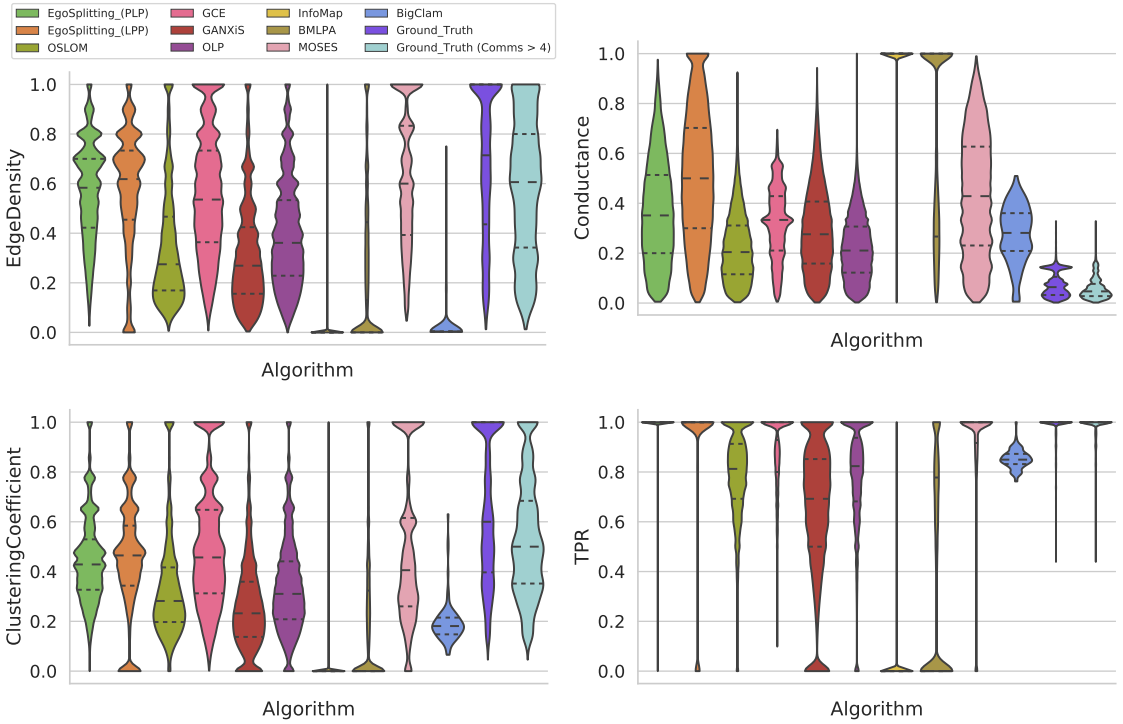


Figure 6.25: The structural properties of each algorithm on the Amazon network with the top 5000 communities.

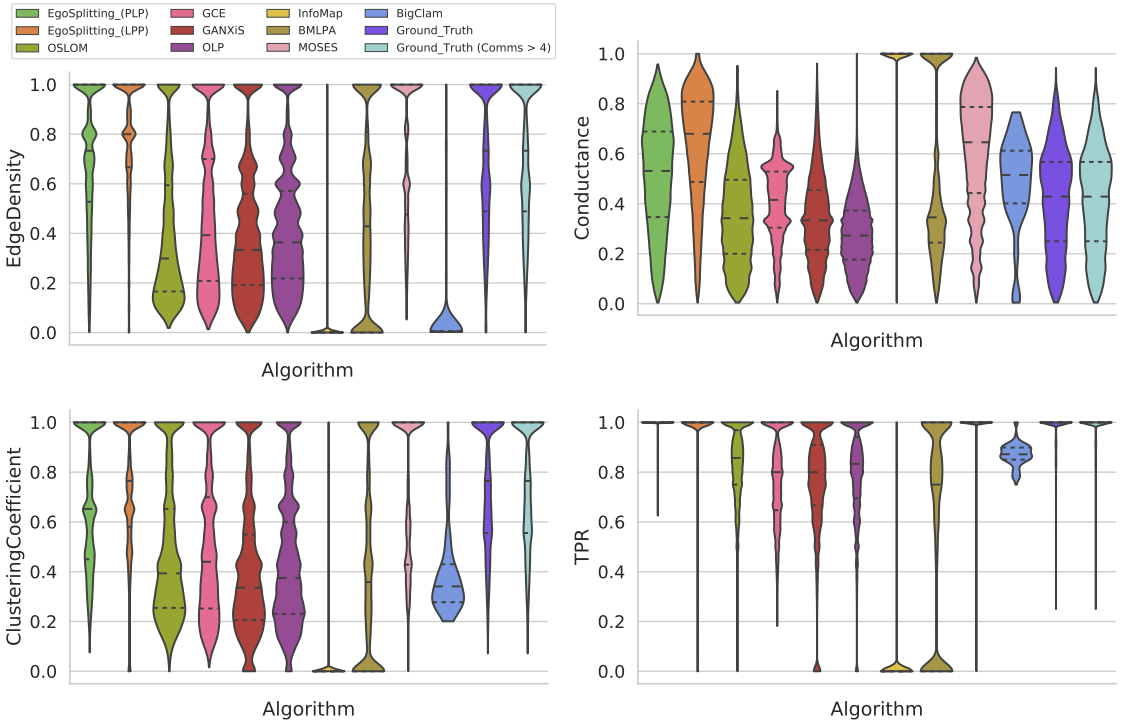


Figure 6.26: The structural properties of each algorithm on the DBLP network with the top 5000 communities.

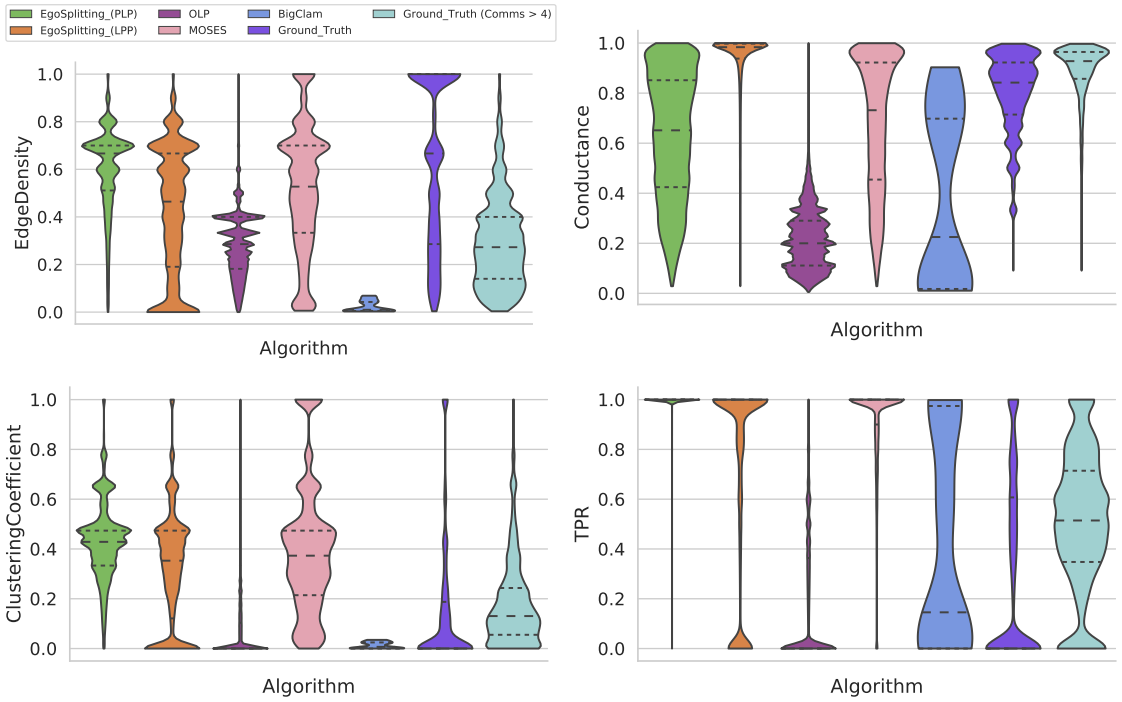


Figure 6.27: The structural properties of each algorithm on the Youtube network with the top 5000 communities.

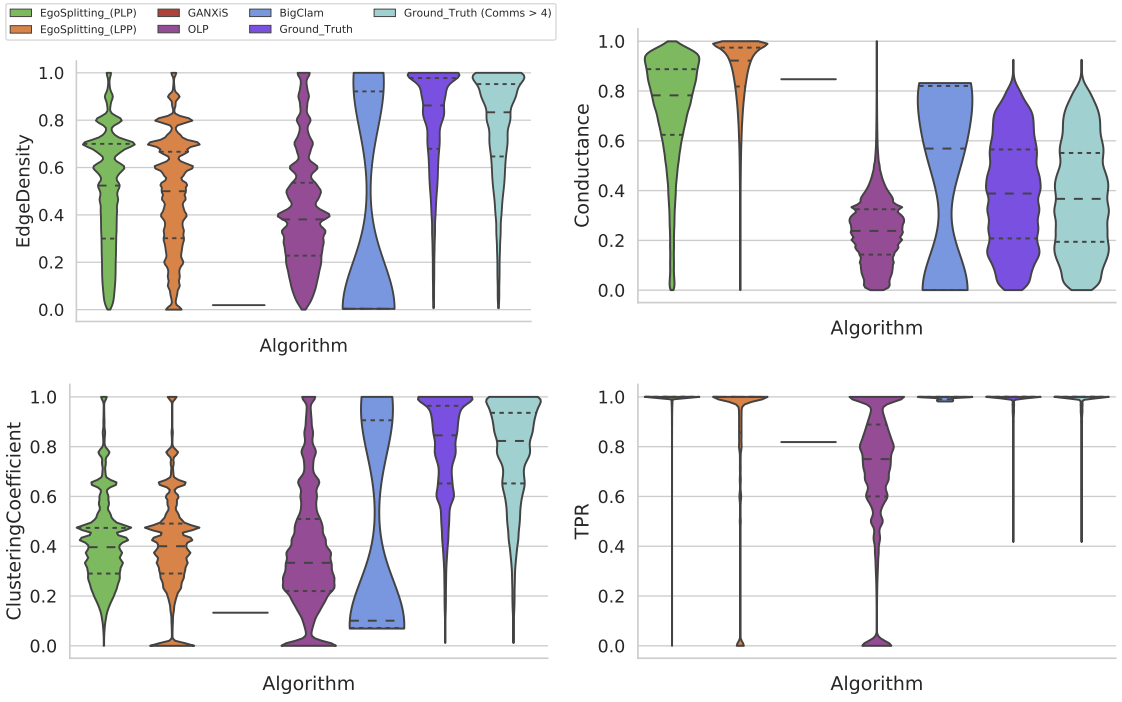


Figure 6.28: The structural properties of each algorithm on the LiveJournal network with the top 5000 communities.

7. Conclusion

In this comparative study, we have empirically evaluated nine overlapping community detection algorithms. To evaluate the performance of each algorithm, we carried out several experiments on synthetic networks, incorporating different network structure, and on real networks with ground-truth communities. To measure the quality of the detected covers, we used performance metrics, such as the NMI, F1 Score and Omega Index. Furthermore, we used complementary metrics on community level, such as verifying if the algorithms detect too many, too few or too small, too large communities or if some ground-truth communities are not detected by the algorithms. We used additional complementary metrics on node level to see if the algorithms are able to assign each overlapping node the correct amount of communities.

Our results show that the quality of the detected covers by the majority of the algorithms, we used, worsens as the number of overlapping nodes in a network increases. Additionally, the performance of these algorithms worsens significantly as the number of communities per node increases. Furthermore, the majority of the algorithms, aside from MOSES, are not good at assigning each node to the correct number of communities. In general, the majority of the algorithms, we evaluated, are only able to detect overlapping communities in networks with a low number of overlapping nodes and low number of communities per node. Only OSLOM and MOSES perform well on networks with a high number of overlapping nodes and both algorithms are also able to detect covers in networks where each overlapping node belongs to a high number of communities. However, both algorithms also have the highest run time of all the algorithms. Our results also show that it is very important to use complementary metrics to evaluate the performance of overlapping community detection algorithms. Performance metrics, such as the NMI or the Omega Index, only measure the overall quality of a detected cover. Whereas, complementary metrics give us more information about the behaviour and what kind of flaws each algorithm has at detecting overlapping communities. For example, as our results have shown, OSLOM detects a similar community size distribution as the ground-truth. However, it is not able to recover most of the smaller communities. By only using metric, such as NMI, we would not be able to see this behaviour of OSLOM. This information could help to improve existing community detection algorithms.

Next, our results, regarding the real networks, match those of Hric et al. [HDF14]. None of the algorithms are able to detect the correct community structure in the real networks that we used. However, most algorithms perform better on real networks where the ground-truth communities show good structural properties, such as the ground-truth communities of

the Amazon network. The authors of [HDF14] point out that the structural communities detected by the algorithms are substantially different than the ground-truth communities defined by the meta-data. This would also explain our results insofar as that the algorithms, we used, perform poorly on real networks with bad structured ground-truth communities and slightly better on real networks with good structured ground-truth communities.

The EgoSplitting algorithm shows promising results on both synthetic and real networks. Depending on which partitioning algorithm is used, the EgoSplitting algorithm is able to detect overlapping communities in CKB networks really well and is also the best performing algorithm on the real networks. The EgoSplitting algorithm is also one of the fastest algorithms, we used in this comparative study.

Finally, as our results show, the majority of the overlapping community detection algorithms are not performing well on networks with a high number of overlapping nodes or a high number of communities per node. Therefore, *future work* could focus on developing new overlapping community detection algorithms that are able to perform well on networks with a high number of overlapping nodes and where their performance is more stable on networks with a high number of communities per nodes.

Recent comparison studies only use the LFR benchmark networks as synthetic networks. Future comparison studies could use additional recent developed benchmark generators, such as the CKB benchmark generator.

The implementation of the LFR benchmark generator could also be extended. The current implementation only allows the overlapping nodes to belong to the same number of communities per node which is unrealistic. This could be extended to allow for other distributions.

Bibliography

[AJ] L.N.F. Ana and A.K. Jain. Robust data clustering. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.* IEEE Comput. Soc.

[Bar04] Albert-László Barabási. Evolution of networks: From biological nets to the internet and WWW evolution of networks: From biological nets to the internet and WWW , s. n. dorogovtsev and j. f. f. mendes oxford u. press, new york, 2003. $95.00 (264 pp.). ISBN 0-19-851590-1. *Physics Today*, 57(10):81–82, October 2004.

[BGLL08] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, October 2008.

[CD88] Linda M. Collins and Clyde W. Dent. Omega: A general formulation of the rand index of cluster recovery suitable for non-disjoint solutions. *Multivariate Behavioral Research*, 23(2):231–242, apr 1988.

[CKB+14] Kyrylo Chykhradze, Anton Korshunov, Nazar Buzun, Roman Pastukhov, Nikolay Kuzyurin, Denis Turdakov, and Hangkyu Kim. Distributed generation of billion-node social graphs with overlapping community structure. In *Complex Networks V*, pages 199–208. Springer International Publishing, 2014.

[CRGP14] Michele Coscia, Giulio Rossetti, Fosca Giannotti, and Dino Pedreschi. Uncovering hierarchical and overlapping communities with a local-first approach. *ACM Trans. Knowl. Discov. Data*, 9(1):6:1–6:27, August 2014.

[ELL17] Alessandro Epasto, Silvio Lattanzi, and Renato Paes Leme. Ego-splitting framework: from non-overlapping to overlapping clusters. 2017.

[ER59] P. Erdös and A. Rényi. On random graphs, I. *Publicationes Mathematicae (Debrecen)*, 6:290–297, 1959.

[ER11] Alcides Viamontes Esquivel and Martin Rosvall. Compression of flow can reveal overlapping-module organization in networks. *Physical Review X*, 1(2), dec 2011.

[FH16] Santo Fortunato and Darko Hric. Community detection in networks: A user guide. *Physics Reports*, 659:1–44, November 2016.

[Gre10] Steve Gregory. Finding overlapping communities in networks by label propagation. *New Journal of Physics*, 12(10):103018, October 2010.

[HA85] Lawrence Hubert and Phipps Arabie. Comparing partitions. *Journal of Classification*, 2(1):193–218, dec 1985.

[HBG+14] Steve Harenberg, Gonzalo Bello, L. Gjeltema, Stephen Ranshous, Jitendra Harlalka, Ramona Seay, Kanchana Padmanabhan, and Nagiza Samatova.

Community detection in large-scale networks: a survey and empirical evaluation. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6(6):426–439, jul 2014.

[HDF14] Darko Hric, Richard K. Darst, and Santo Fortunato. Community detection in networks: Structural communities versus ground truth. *Physical Review E*, 90(6), December 2014.

[LBA11] Pierre Latouche, Etienne Birmelé, and Christophe Ambroise. Overlapping stochastic block models with application to the french political blogosphere. *The Annals of Applied Statistics*, 5(1):309–336, mar 2011.

[LF09] Andrea Lancichinetti and Santo Fortunato. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 80(1), jul 2009.

[LFK09] Andrea Lancichinetti, Santo Fortunato, and János Kertész. Detecting the overlapping and hierarchical community structure in complex networks. *New Journal of Physics*, 11(3):033015, mar 2009.

[LFR08] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4), oct 2008.

[LRMH10] Conrad Lee, Fergal Reid, Aaron McDaid, and Neil Hurley. Detecting highly overlapping community structure by greedy clique expansion. *arXiv e-prints*, page arXiv:1002.1827, Feb 2010.

[LRRF11] Andrea Lancichinetti, Filippo Radicchi, José J. Ramasco, and Santo Fortunato. Finding statistically significant communities in networks. *PLoS ONE*, 6(4):e18961, apr 2011.

[MGH11] Aaron F. McDaid, Derek Greene, and Neil Hurley. Normalized Mutual Information to evaluate overlapping community finding algorithms. *arXiv e-prints*, page arXiv:1110.2515, Oct 2011.

[MH10] Aaron F. McDaid and Neil J. Hurley. Using Model-based Overlapping Seed Expansion to detect highly overlapping community structure. *arXiv e-prints*, page arXiv:1011.1970, Nov 2010.

[MMG+07] Alan Mislove, Massimiliano Marcon, Krishna P. Gummadi, Peter Druschel, and Bobby Bhattacharjee. Measurement and analysis of online social networks. In *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement - IMC 07*. ACM Press, 2007.

[PDFV05] Gergely Palla, Imre Derényi, Illés Farkas, and Tamás Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, June 2005.

[RAB09] M. Rosvall, D. Axelsson, and C. T. Bergstrom. The map equation. *The European Physical Journal Special Topics*, 178(1):13–23, nov 2009.

[RAK07] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3), sep 2007.

[RN10] Peter Ronhovde and Zohar Nussinov. Local resolution-limit-free potts model for community detection. *Physical Review E*, 81(4), April 2010.

[SHW17] Neha Sengupta, Michael Hamann, and Dorothea Wagner. Benchmark generator for dynamic overlapping communities in networks. In *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, nov 2017.

[WLG+12] 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, jan 2012.

[XKS13] Jierui Xie, Stephen Kelley, and Boleslaw K. Szymanski. Overlapping community detection in networks: the state of the art and comparative study. *ACM Computing Surveys*, 45(4):1–35, aug 2013.

[XSL11] Jierui Xie, Boleslaw K. Szymanski, and Xiaoming Liu. SLPA: Uncovering Overlapping Communities in Social Networks via A Speaker-listener Interaction Dynamic Process. *arXiv e-prints*, page arXiv:1109.5720, Sep 2011.

[YL13a] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42(1):181–213, oct 2013.

[YL13b] Jaewon Yang and Jure Leskovec. Overlapping community detection at scale. In *Proceedings of the sixth ACM international conference on Web search and data mining*. ACM Press, 2013.