Local Community Detection in Complex Networks

M2 Project in Erasmus Mundus Master's in Complex Systems Science

ÖMER SALİH YÜKSEL

Department of Computer Science
Networks and Systems
Chalmers University of Technology

Gothenburg, Sweden 2013
Master's Thesis 2013:1

Local Community Detection in Complex Networks

ÖMER SALİH YÜKSEL

© ÖMER SALİH YÜKSEL, June 2013.

Examiner: PHILIPPAS TSIGAS

Chalmers University of Technology
University of Gothenburg
Department of Computer Science and Engineering
SE-412 96 Göteborg
Sweden
Telephone + 46 (0)31-772 1000

Department of Computer Science and Engineering
Göteborg, Sweden, June 2013

Abstract

Community structure is an important aspect of network analysis, with a variety of real-life applications. Local community detection algorithms, which are relatively new in literature, provide the opportunity to analyze community structure in large networks without needing global information. We focus our work on a state-of-the-art algorithm developed by Yang and Leskovec and evaluate it on three different networks: Amazon, DBLP and Soundcloud.

We highlight various similarities and differences between the geometry and the sizes of real and annotated communities. The algorithm shows robustness to the seed node, which is also demonstrated by its rather high level of stability. By using two different methods of seed selection from the literature, we demonstrate further improvement on the quality of the communities returned by the algorithm. Finally, we try to detect real-life communities and show that the local algorithm is comparable to global algorithms in terms of accuracy.

Acknowledgements

This project has been funded with support from the European Commission. This publication reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.

The author is thankful to his supervisors; Philippas Tsigas and Farnaz Moradi for the guidance and support they provided; and the fellow members of Complex Adaptive Systems and Complex Systems Science for their constructive feedback during the project work.

Ömer Salih Yüksel, Göteborg, 22/06/2013

Contents

1

Introduction

NETWORKS appear in many real life structures such as social groups, energy distribution systems, the World Wide Web, and the molecular structure of materials. The emergence of network science allowed researchers to analyze these different entitities within the same mathematical framework. With the help of this common framework, it was discovered that such different structures share many similar characteristics, e.g. power-law distributions of the degrees [1] or the 'small world' phenomenon[2].

One such characteristic is the groups that hold members with high number of connections among themselves and low number of connections to those outside of the group. Such groups are generally called 'communities', and the problem of finding such groups is referred to as 'community detection' in the literature [3].

The majority of the community detection algorithms in the literature are ”global” algorithms. That means, they require the complete information of the network, and try to find all communities in it. In this project, however, we work on local community detection algorithms, which are relatively new in the literature.

While the global algorithms have a graph-centric approach, i.e. focus on finding tightly connected groups of nodes in a graph; local algorithms have a node-centric approach, focusing on finding the communities around a node. As such, local community detection algorithms tend to start from a 'seed node', then grow the communities around the node in the subsequent iterations.

One of the most well-known algorithms is an extension of the Local Spectral Clustering algorithm by Yang and Leskovec [4]. Our work focuses on its applications and evaluation on various networks and possible methods to improve the algorithm's performance.

1.1 Motivation

Community detection is one of the widely discussed problems in network science, and it has a wide range of applications. These applications include detecting friend circles in online groups, grouping similar types of proteins in protein-interaction networks, and uncovering clusters that are separated by characteristics such as geographic location.

A brute-force approach to the problem is computationally expensive, which necessitates 'smarter' solutions. There exists various methods proposed (which we discuss in the following sections) to approach the problem. However, there is no 'perfect method' to handle the task. Community detection remains an open research problem, and any new approach yields the possibility of potential breakthroughs and expanding the algorithm's possible areas of use.

Local algorithms provide a new approach to the problem. They allow the detection of communities in the situations where one is only interested in a certain subset of the network, without having to involve the whole network in their calculations. In the situations where one deals with very large networks, such an approach is particularly desirable [5]. However, as it is a relatively new concept, few works propose or evaluate local community detection algorithms, and even fewer works deal with the particulars such as community characteristics or detecting real-life communities.

As stated before, our work focuses on Yang and Leskovec's algorithm[4]. We aim to evaluate the performance of the algorithm, describe the properties of the communities detected, discuss possible methods of improvement that have been proposed in the literature and use the algorithm to detect the communities found in real life. By meeting these goals, we hope to provide new insights to local community detection algorithms, and the community detection problem in general.

1.2 Definitions

Our work is heavily involved with the concepts of network theory. This sections contains the definition of such concepts, and the explanations about our naming choices.

Graph: The entity $G(V,E)$ is defined as a graph with V as the set of vertices (singular: vertex) and E as the set of vertices; such that:

$$\forall (u,v) \in E : u \in V, v \in V$$

In this work, we use the words *graph* and *network* interchangably. However, we prefer to use the term *graph* when referring to an abstract mathematical entity, and *network* when referring to a real-world structure (such as Soundcloud network).

Vertex: A *vertex*, or a *node*, is a single element of the set V in the graph $G(V,E)$. Vertices are the main building blocks of the graphs. We generally use the term **node**.

Edge: An *edge*, or a *connection* is an element of the set E in the graph $G(V,E)$. The element consists of a pair, like (u,v), and both sub-elements in the pair belong to the set of vertices, V. A vertex can be connected to any other vertex in the graph.

Degree: The number of edges a node is connected to.

Subgraph: A subgraph G' of the graph G is defined as follows: $G'(V',E')$ is a graph such that $V' \subset V$, $E' \subset E$. G' is also a graph, and satisfies the requirements of one.

We prefer the term *subgraph* when mentioning the abstract entity. In other cases, we prefer to use the informal term 'group of nodes'.

Shortest path: The *shortest path* between two nodes (u, v) is a sequence of edges, in which the number of edges required to traverse when trying to reach from node u to node v is minimum.

Distance: The number of edges in the shortest path between two nodes. If there exists no paths between two nodes, then the distance is considered to be ∞.

Diameter: The maximum distance in a graph or a subgraph.

Eccentricity: The maximum distance between a node and all the other nodes in a graph (or subgraph) [6].

Community: A group of nodes where the number of internal edges exceeds the number of edges connecting the nodes outside the group. Figure 1.1 shows the communities detected in the author's Facebook network. See the following section for the discussion about its definition.

Annotated community: A group of nodes that define a 'community' in real life sense, such as Facebook groups. Annotated communities can be used as a basis to evaluate the usefulness of a community detection algorithm. We also use the terms 'communities based on ground-truth' and 'annotated communities' interchangeably [4].

Boundary: The boundary of the community C for a graph $G(V,E)$ is defined as the set of edges E', such that $E' \subset E$, $E = \{(u,v)|(u,v) \in E, u \in C, v \notin C\}$ [5]. We also call E' the set of 'inter-community' or 'bordering' edges.

Neighborhood: A neighborhood, or an egonet of a node, is the set of vertices that are connected to that node [7].

1.3 Background

Community detection is about finding tightly connected groups of nodes with few connections leading outside. However, the problem remains ambiguous in the literature: there is no consensus on what a community is, and some amount of 'common sense' is required to identify one [3].

Following from the ambiguity of the definition of a community, there are also multiple measures of how 'fit' a group of nodes is to be a community, such as modularity or conductance. By using these measures, we can define the community detection as a problem of optimizing these fitness functions, which is an NP-Complete problem [4]. Therefore, many community detection methods are actually approximation algorithms.

We can then give examples of various fitness functions. Let G(V, E) a graph, C the set of nodes in a community, e_{in} the number of edges where both nodes are within C, e_{bnd} the number of edges where one node is in C and one is outside (boundary edges), and e_out the number of edges where both nodes are outside C. Let k_i be a node's degree, and let A be the 'adjacency matrix' of the graph so that $A_{ij} = 1$ when an edge exists between the nodes numbered i and j. Let $s_i = 1$ when the node numbered i is in the

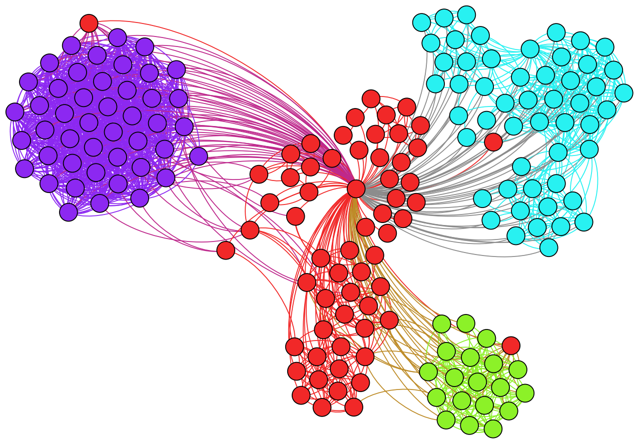


Figure 1.1: The results of a global community detection algorithm (greedy modularity maximization) on the author's Facebook ego network. The first three communities roughly correspond to: magenta-high school, green-workplace, blue-university. The red-colored nodes form a union of the remaining smaller communities in the network.

community, and$s_i = -1$ otherwise. The definitions of various fitness functions are as follows:

Conductance [8]**:** The actual definition of conductance is:

$$f(C) = \frac{e_{bnd}}{min((e_{bnd} + e_{in}), (e_{bnd} + e_{out}))}$$

For local community detection, where the community is considerably smaller than the graph, conductance can be defined as the fraction of the edges in C that point out of the community and be simplified as:

$$f(C) = \frac{e_{bnd}}{e_{bnd} + e_{in}}$$

Modularity[9] **:** Difference between the number of edges in the community, and the expected number of edges in a given graph with the same degree distribution.

$$f(C) = \frac{1}{4} \sum_{i,j} [s_i s_j (A_{ij} - \frac{k_i k_j}{2|V|})]$$

Triad participation ratio [4]: The fraction of nodes in the graph that belong to a triangle.

$$f(C) = \frac{|u : u \in C, (v,w) : v,w \in C, (u,v) \in E, (u,w) \in E, (v,w) \in E|}{|V|}$$

Cut Ratio[3]: Fraction of the edges pointing outside the community, to all possible such edges.

The best known examples of community detection algorithms are what we denote 'global' algorithms. They are generally concerned with finding all communities in a given graph. However, there are several networks with a massive volume or a dynamic structure. In such networks, attempting to detect communities using global algorithms becomes infeasible [10].

Our work, however, is concerned with local community detection. Local methods start from one seed node and grow a community around it. In the following subsections we explain these in more detail, and give examples of both global and local methods.

1.3.1 Global Community Detection

The vast majority of the work done in the field of community detection is on global methods. We included several well-known methods in this section and grouped them into different categories. It should be noted, however, that global algorithms are not limited to these categories.

Clustering: Community detection problem is analogous to cluster analysis in machine learning. Cluster analysis, or clustering is the problem of finding group of objects that show more similarity to each other than the other objects [11]. It has applications in many different areas, such as bioinformatics, text analysis and recommendation systems. For the community detection problem, 'similarity' can be defined as the number of common neighbors.

Modularity maximization: Modularity is one of the most widely used measures in community detection methods. One such way to approach the modularity maximization problem is to use greedy algorithms. An algorithm is denoted 'greedy' when it makes decisions based on reaching local optima in each step [12]. The first known greedy method for modularity maximization, based on hierarchical clustering, was developed by Newman[13]. In a later work, Clauset, Newman and Moore [14] developed one of the most well-known greedy algorithms, which is able to detect communities in graphs with size up to 10^6. In this paper, we used the algorithm on some of our datasets to allow the reader to make a comparison.

Besides greedy algorithms, there are several other methods of optimization, such as *simulated annealing.* [15]

Random walk based methods: If the communities in a graph are well-separated, we can expect a random walker to spend considerable amount of time in a community, due to the high number of interconnecting edges and low number of 'bordering' edges. Some examples using random walks are "Walktrap" by Latapy and Pons, which is based on probabilities of reaching a node in a number of fixed steps [16]; and methods based on Markov chains by Van Dongen[17] and by Weinan et al. [18].

Shortest-path based methods: The method of Girvan and Newman, based on *edge betweenness*, is the most well known among the methods based on shortest paths. [19]. Betweenness of an edge is defined as the number of shortest paths that go through the pairs of the nodes in the edge. [20]

The algorithm runs as follows:

1. Calculate betweenness values for all edges
2. Remove the edge with highest betweenness
3. Recalculate the edge betweenesss values
4. Go to step 2

As the algorithm runs, communities are transformed into components that are disconnected from each other. An application of the algorithm by Gleiser and Danon on musician networks reveal meaningful communities: Musicians are shown to be separated by race and geographic location. [21]

With the present computational power, the algorithm is only feasible to run on graphs with up to 10,000 nodes [3]. However, there are variants that are designed to decrease the time complexity [22].

Overlapping community detection: Most of the community detection algorithms find communities with distinct nodes. This, however, is not the case for the communities found in real-life. In the Figure 1.1, for example, the community (2) is composed of the author's co-workers and classmates. As some of the people belong to both these groups, it is actually a group consisting of two overlapping communities.

Since considering overlaps adds another dimension of complexity, the problem is generally ignored. There are, however, methods that are designed with overlapping communities in mind.

Clique percolation method is the most widely known example [23]. A 'clique' is a group of nodes that are all connected to one another, and a set of such k number of nodes is called a $k - clique$. The idea of the clique percolation method is that, due to the definition of a community, cliques are likely to exist within the community, and unlikely to exist at the community boundaries. Therefore, detection of the cliques forms the essential part of the community detection process.

1.3.2 Local Community Detection

The main difference between local algorithms is that they do not require complete information about the network. A seed node is selected, and the community is 'grown' around the seed node using a bottom-up approach.

In order to properly evaluate a local algorithm, one must study the properties of the detected communities. In general, we expect the returned communities to have an optimal fitness value (e.g. low conductance, high modularity) and display similarities to communities found in real-life networks. The size and the shape of the communities are particularly important. Most of the works about community detection briefly mention the community sizes. Few works also address the aspects of the community geometry, such as the diameter [24][25].

As local methods require a seed node to run, the selected seed is one of the parameters that influence the algorithm's performance. Several works propose methods of seed selection [26][27][7], and we evaluate two of these methods in our work. Another aspect of the algorithm's performance is its stability. Local algorithms use a bottom-up approach to reveal the community structure. We call an algorithm 'stable' if we can detect the same community structure using different seeds that belong to it. This area is generally neglected in the literature, however, there exists mentions of overlapping communities with different seeds [27][25].

Lastly, the results of a community detection algorithm can be tested against real data. A considerable number of real-life networks contain annotated communities; such as groups in Facebook and departments in a university network. Using the algorithm to predict these communities will give an idea of how accurate the community detection process works. Note that this task is challenging, as annotated communities do not always share the cohesive structure as the communities returned by metrics like conductance and modularity. Nevertheless, some of the present work also focus on this area [4][24].

Previous Work

The local methods are relatively new in literature. Clauset [5] proposed the first local community detection method based on 'local modularity'. This new definition of modularity was required since Girvan and Newman's definition[19] of modularity uses global information, such as the number of edges in the complete network. Local modularity is based on the edges in the community boundary. With $B \subset E$ as the community boundary, the local modularity for the community C can be defined as:

$$\frac{|\{(u,v)|(u,v) \in Bu \in C, v \in C\}|}{|B|}$$

Another well-known work on local methods was developed by Luo, Wang and Promislow [25], with a new definition of local modularity. They proposed methods based on three different heuristics for optimization. The communities detected by the algorithm displayed small-world properties and overlaps in certain cases. The algorithm is claimed to be better suited for recommendation engines compared to Clauset's.

Bagrow [28] proposed an algorithm based on the concept of a node's 'outwardness', which is defined as the normalized difference between the node's neighbors outside the community and those inside the community. The outwardness value is used in the search process as the main criterion of a node's inclusion into the community.

Local spectral clustering algorithm, which Yang and Leskovec's algorithm is based on, is particularly worth mentioning. Spielman and Teng proposed a method for graph partitioning based on the mixing of random walks, using an algorithm named Nibble [29].

Andersen and Lang applied these methods to detect communities in various networks [30]. In addition, they developed an improved extension to the original algorithm, called PageRank-Nibble [8].

There are several works that deal with the seed selection process. Gleich and Seshadri proposed selecting low-conductance neighborhoods as communities, and presented results on using such neighborhoods in the seed selection process [7]. Pan et al. also presented a method of seed selection, based on ‘PageRank centrality’, a modified version of the PageRank algorithm [27]. They denote these selected nodes as ”leaders”, and expanded the community according to a fitness function after the leader selection. Chen and Fang, on the other hand, select seed nodes based on a ”locally maximal degree” heuristic, and use a local community detection method based on adding neighbors that have the ”greatest common neighbors” [26].

2

Methodology and Problem Setting

It is necessary to provide the reader more in-depth information before moving on to the results. In this chapter we explain our methods; the choice of programming languages and the libraries, then give more information about the concepts related to our work, summary and interpretation of the network data, and finally the workings of the algorithm we are evaluating.

2.1 Methodology

Our work involves testing the local community detection algorithm in various networks. The majority of these networks have been obtained from Stanford's SNAP project. In addition to these, we decided to gather network data from SoundCloud, the music distribution platform with over 42,000,000 users. Surprisingly, our research found no social network analysis work using the network.

For our calculations, we used NetworkX[31] (Python) and SNAP[32] (C++) libraries. SNAP was mainly used for community detection and NetworkX was mainly used to calculate network statistics.

For the collection of Soundcloud data, we alternated between random sampling and breadth-first-search, so that we could capture local neighborhood information while covering different parts of the network. SQLAlchemy for Python was utilized to provide a seamless interface to the database in the data collection process [33]. More information about the Soundcloud network is given in the following section.

With the exception of the parts where other seed selection methods are stated, we always used uniform random sampling with replacement to produce community statistics.

2.2 Datasets

Besides Soundcloud, we included DBLP and Amazon networks in our work, both obtained from the SNAP project[34]. DBLP is a citation network, the edges denote co-authoring on an article, and ground-truth communities denote the journal, conference or organization the authors have appeared together. Amazon dataset contains a co-purchasing network of products from the website of the same name. The annotated communities denote various product categories in the website.

2.2.1 Soundcloud

As mentioned before, Soundcloud is an online social network about sound sharing, containing over 42 million users.

By the time we started community detection work, we had ca. 1.5 million users in the giant connected component; meaning that our work is focused on a small subset of the whole userbase. As stated before, local algorithms focus on the vicinity of a node and do not require complete information about a network to perform, so we do not expect to encounter any significant issues about the lack of global data. Nevertheless, data collection is an ongoing process, and we are going to publish a more complete version of the Soundcloud data along with the current subset.

Similar to Twitter, users can follow each other. As a result, unlike Facebook's friendships, the connections are not necessarily reciprocal. Popular artists tend to attract a large number of followers. The users can comment on the tracks in the website.

The users also can form and join groups. We use these groups to make comparison between the detected communities and the real-life communities.

We investigated two possible ways to create a network out of the data. The first, and obvious one, is the network of 'follow' relations. The users are nodes, and the graph contains a directed edge (u,v) if the user u follows the user v.

The second type of network we investigated was the 'co-commenting' network. If two users commented on the same track, an undirected edge is formed between them. The network is also weighted: the number of tracks commented on by both users is the weight of the edge.

In the end, we decided to only use the follows network for our algorithm. In the following section, we give the reasoning behind our initial decision for investigating the co-comments network, and later not following through with it.

Co-comments network

The followers network provides a good start for the community detection task. However, we had certain concerns about detecting annotated communities. Following relations are partially due to friendships, and partially due to the user being interested in the tracks produced by another user. Groups, on the other hand, are largely formed by the users with similar taste in music, not due to friendships. Therefore, the follows that are due to personal contacts would actually lead to false positives when detecting communities: the algorithm would detect a group of friends as a structural community, but no Soundcloud group would correspond to it.

Users commenting on the same track, on the other hand, are shown to express interest in the same kind of music, and would therefore would have a higher probability of being in the same group. Therefore, a co-commenting network could lead to more reliable results in terms of community classification.

Another incentive for choosing to work on the comments network is the fact that comments are timestamped, whereas follows are not. Studying the comments network gives us the possibility to study the evolution of the communities.

The disadvantages of the co-comments network is that they are much more dense and clustered in comparison to other real-life networks. For example, a track with 100 users commenting on it would create a clique with 100 nodes in the network. This means all of these 100 nodes would have an edge between each other. With over 100,000,000 tracks, this leads to an immense density and clustering, and makes the community detection task very challenging. Figure 2.1 shows the weight distribution of the comments, with the 'weight' being the number of co-comments between two users.

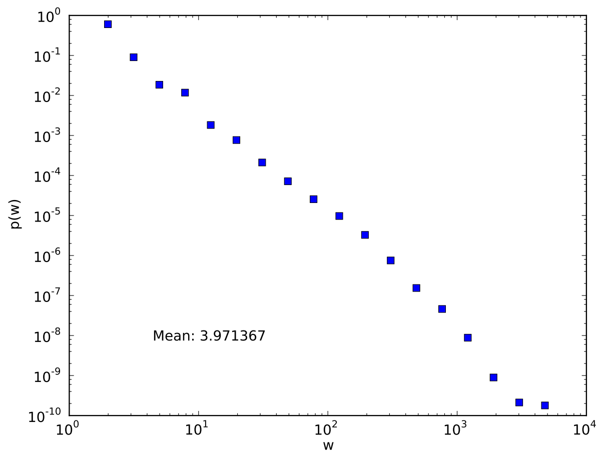


Figure 2.1: Weight distribution for the co-comments network.

Dealing with this density problem requires us to come up with a method to eliminate the majority of the edges. Several measures such as setting up a threshold (number of co-comments), a timestamp based filter(only comments that are posted in similar time periods create an edge), additional criteria for eligibility (comments on the same track and the group) were proposed.

Despite these measures, crawling and creating co-comments network is computationally much more expensive, compared to the follows network. In addition, enforcing these measures increases the computational cost even further. Therefore, as it was infeasible to create a co-comments network at a large size, we decided to run the algorithm on the follows network only.

2.2.2 Network properties

The Table 2.1 shows the basic statistics about the three networks. While they have various differences, e.g. in size, diameter and density; they also share characteristics that are common in many real life networks. One such characteristic is the hierarchical distribution of the degrees (see Figure 2.2).

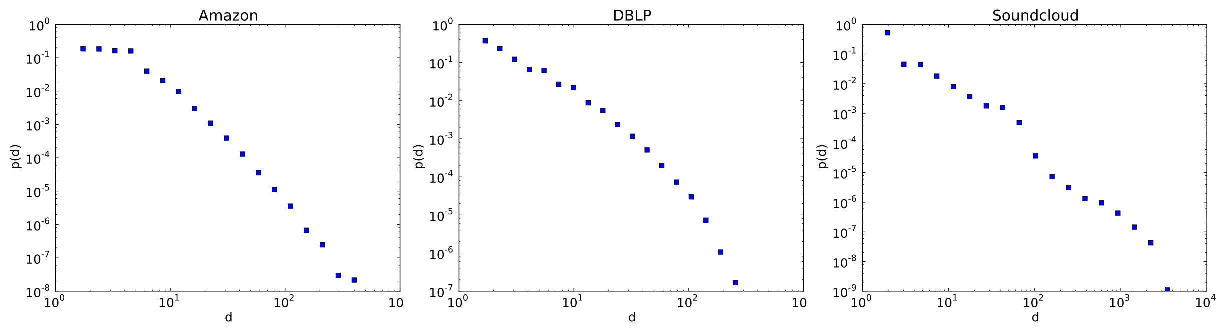


Figure 2.2: Degree distributions, with logarithmic binning.

All three distributions follow a similar trend, with strongly decreasing density as the degrees increase. In the case of Amazon, the distribution starts with a relatively moderate decline for the low-degree nodes, then follows a power-law. DBLP, on the other hand, shows a steeper decline in density. Soundcloud also has a sharp decline for the high degrees and several shifts in its slope. It should be noted that Soundcloud has the highest maximum degree among all three networks, which is expected as it is the largest network among the three.

Table 2.1: Network statistics for Amazon, DBLP and Soundcloud.

	Soundcloud	Amazon	DBLP
Nodes	1,583,460	334,863	317,080
Edges	3,009,036	925,872	1,049,866
Groups	48,091	271,570	13,477
Fraction of the nodes with group memberships	0.07	0.96	0.82
Average group size	24.4	11.7	53.4
Average membership size	0.74	9.5	2.3

It should be also noted that Soundcloud is a directed network, whereas Amazon and DBLP are not. The algorithm, however, always assumes an undirected network. We note that this could lead to interesting results, and discuss the possible effects of this assumption in the following section.

3

Algorithm

In this chapter we present an outline of the Yang and Leskovec's algorithm (denoted YL). We refer the interested readers to their paper for the details [4].

The algorithm is based on random walks, and is an extension of the Local Spectral Clustering algorithm [29] [30]. As we mentioned in the previous chapter, random walks are one of the methods to approach the community detection problem in the literature, and there exist a variety of random walk based-algorithms in the literature, such as Walktrap.

YL uses a personalized PageRank method. The personalized PageRank assumes that a random-walker begins its motion in a particular seed node, and returns the distribution of its probabilities of being in each node in the graph, as a vector. The nodes with a high probability of visiting have greater PageRank scores. The implementation of PageRank that is used in the algorithm is called PageRank-Nibble, an approximation algorithm by Andersen, Chung and Lang [8]. The implementation ignores the nodes that score below a threshold value and thereby limits the computation to the seed node's vicinity. An overview of the algorithm is given in the following subsection.

The pseudocode of the method can be seen in the Algorithm 1. The plot of the conductance score with respect to the number of nodes in the community is called the 'sweep curve'. The algorithm uses the local minima of the curve as a means of detecting the communities. Since all of the nodes in the sweep curve belong to the same sequences, the local minima correspond to 'nested' communities. Leskovec and Yang generally select the first local minimum in their work. In order to maintain consistency and allow comparison of results, we also use the first local minimum.

The score of a node is proportional to its PageRank score, and inverse proportional to its degree. Consequently, the algorithm avoids high-degree nodes (which lead to greater conductance) and is attracted towards nodes with a high probability of a random walker visiting.

The implementation we are using works with conductance as the fitness function [32].

As the original work shows that conductance provides the most effective score to match annotated communities, we used this implementation without modifications.

Our work focuses on aggregated results, but we present an example on the author's Facebook ego-network for the reader's consideration (see Table 3.1) .

Algorithm 1: Local community detection [4]

```
1: Compute the PageRank-Nibble scores $r_u$ from the seed $s$
2: Order nodes $u$ by the decreasing value of $r_u/d(u)$, $d(u)$ being the degree
3: for k = 1 to len(PageRank vector) do
4:     S = First k nodes in the list
5:     Compute the fitness function $f(S)$
6: end for
7: Detect the local optima of $f$
8: Each local optimum corresponds to a community found. For each $k_{min}$, get the
   first $k_{min}$ nodes, $C_{k_{min}}$ and insert into $S$: $S = S \cup C_{k_{min}}$
```

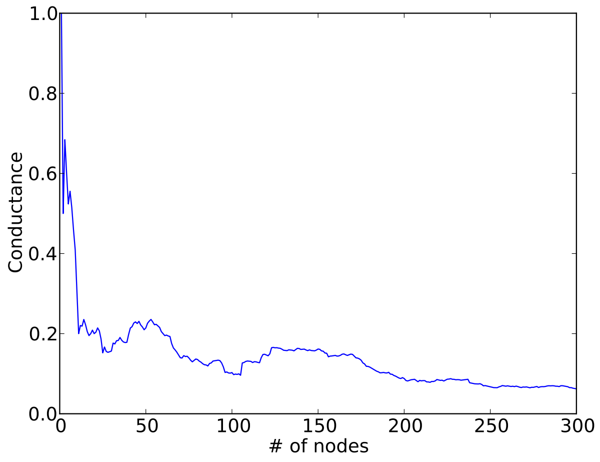


Figure 3.1: The sweep curve of the algorithm, with the conductance score plotted against the number of nodes, ordered by their modified PageRank score.

3.0.3 Directed networks

It was previously mentioned that the algorithm assumes an undirected network. Converting the directed edges to undirected causes loss of information to some extent, and in this section we discuss the consequences of such conversion.

The difference starts with the definition of conductance. Let us call C the nodes that belong to a community, $\bar{C}$ the nodes in the graph that do not belong to the community,

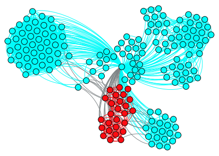
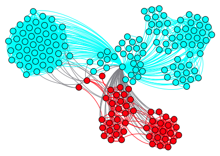
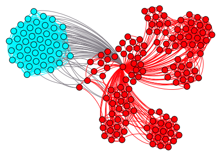
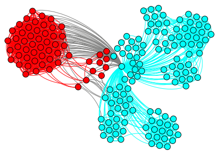
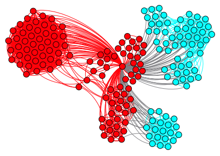
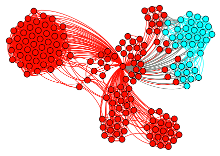
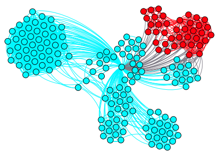
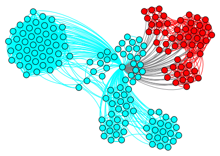
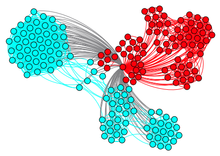
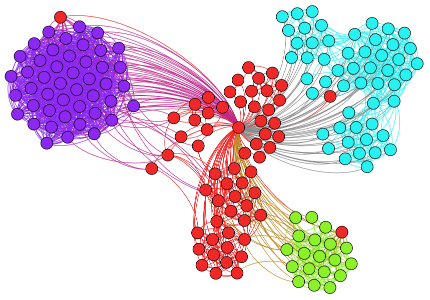
Local seed 1			
Local seed 2			
Local seed 3			
Global			

Table 3.1: An example of the communities detected by the algorithm, on the author's Facebook network. The last row contains the communities detected by the greedy modularity maximization algorithm [14]. The columns denote the local minima with increasing order and the red nodes denote the detected local communities.

e_o the edges leading from C to $\bar{C}$, e_i the edges leading from $\bar{C}$ to C, and e_c all the edges that are incident with C.

With directed networks, conductance of C is e_o/e_c [35]. Note that we do not involve e_i. In contrast, with the undirected networks, we consider all edges that are connected to a node outside the community. That means, when we convert a directed network to an undirected one, the conductance of C becomes $(e_i + e_o)/e_c)$. Therefore, the community scoring is affected, with certain communities having increased conductance. This potentially leads to losing some of the local minima that would appear with a directed network.

PageRank scoring is also affected. The original algorithm was intended for the world wide web, with web pages as nodes, and links as the edges. As the links are not required to be reciprocal, the network is directed.

Converting non-reciprocal directed edges into undirected ones can significantly affect the movements of a random walker. Consider a web page which has no links, but is linked by a great number of other pages. In the directed version of the network, the random walker would be trapped in the node. As a contrast, when the network is converted to undirected, the random walker can move out of such nodes. That would mean spending less time in that node, and would lead to a lower PageRank score than before.

It should be remembered that the algorithm is trying to find groups with low conductance, and uses the PageRank score as the main criterion of finding such nodes. Converting the node to undirected causes some nodes to have decreased PageRank scores, and some groups to have increased conductance. Consequently, some of the communities that would potentially appear in a directed network are ‘lost’.

4

Characteristics of the Communities

In this chapter, we investigate several aspects of the communities including their size and shape. This area, while important for the evaluation of the local algorithms, is mostly left unexplored in the literature.

4.1 Size

The community size forms a basic but important measure in community detection. It would be desirable for an algorithm to be able to cover the various community sizes that appear in the real-life networks. Figure 4.1 shows the distribution of community sizes, and Table 4.1 shows the mean values, compared to the annotated communities.

For Soundcloud, the algorithm provides a near-perfect match of the mean community size. It returns considerably smaller communities for DBLP (less than the half), and larger communities for Amazon. The algorithm covers community sizes between 1 to $\sim 10^3$. For the cases of Amazon and DBLP, the annotated communities cover an even larger extent, and several communities over the size of 1,000 are beyond the algorithm's reach.

The distributions show similarities, and in case of Soundcloud, overlaps for small-sized communities. However, looking at the curve, it is clear that annotated communities and detected communities belong to different types of underlying distributions.

4.2 Geometry

The geometric analysis of a community involves its various properties about the shape and structure of the community. In this section, we address three different areas: diameter, symmetry and roundness.

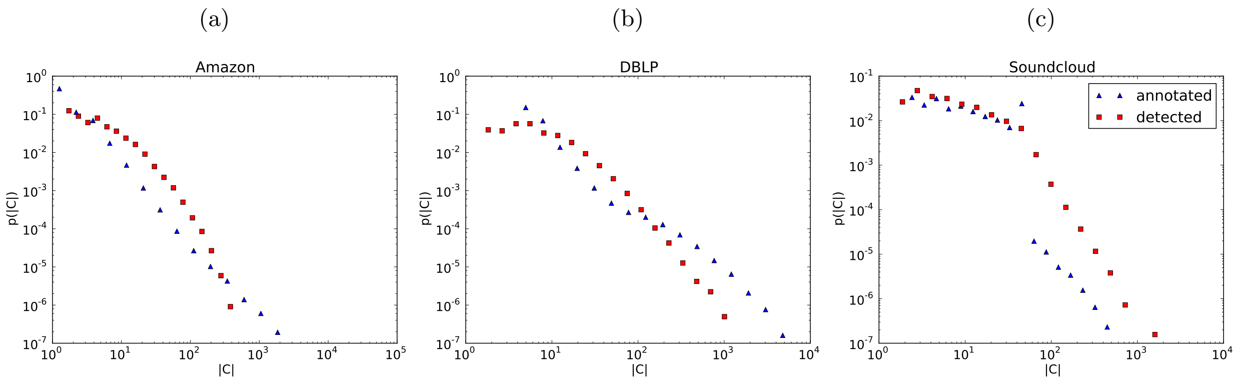


Figure 4.1: Size distribution of the detected communities in comparison with the annotated communities (log-binned).

Table 4.1: The mean values of community size, $|C|$ for the detected and annotated communities.

	Annotated	Detected
Amazon	11.7	17.1
DBLP	53.4	25.8
Soundcloud	31.8	31.8

The diameter of a community is the largest distance between any nodes in it. As communities tend to be closely connected groups with many edges inside, a small diameter value is desirable. However, it is also shown that some real-life communities can have a large diameter.

The diameter distributions are shown in Figure 4.2. The algorithm finds communities with a relatively larger diameter for DBLP. This fact is also reflected in the distribution: while all three distributions are skewed to the right, the skewness of the distribution is more pronounced for Amazon and Soundcloud. In contrast, DBLP has the peak of the distribution shifted to the right.

Soundarajan and Hopcroft [24] explain the distinction of ‘round’ and ‘long’ communities. Round communities have edges distributed throughout the community, whereas long communities contain peripheral nodes that have a considerably high distance from each other. They also maintain that fitness functions such as conductance and modularity return round communities in general. As a measure of ‘roundness’, they propose measuring the ratio of the community’s diameter to a graph with the same number of nodes and edges. Communities that have a ratio of 1 are considered round.

It has been shown that annotated communities generally have larger diameters compared to the random graphs of the same size, with the ratio varying between 1.1 to 2[24].

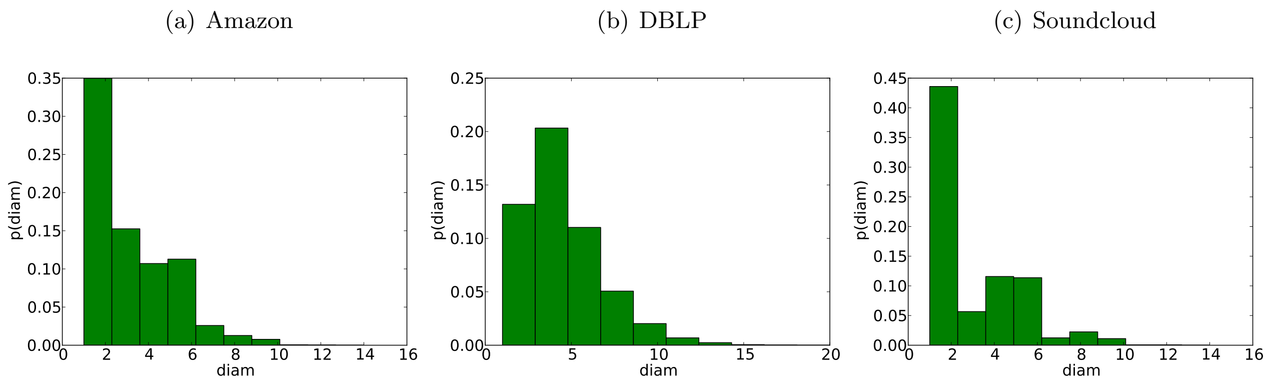


Figure 4.2: Diameter distributions for the detected communities, with the mean values of (a)3.14 (b)4.20 (c)3.28

Table 4.2: 'Roundness' measures for the communities detected by our algorithm, with various thresholds. See Soundarajan and Hopcroft's paper for the results on various annotated communities [24].

	Amazon	DBLP	Soundcloud
(#nodes $\geq$ 50)	1.28	1.45	0.50
(#nodes $\geq$ 75)	1.31	1.50	0.57
(#nodes $\geq$ 100)	1.34	0.57	0.59

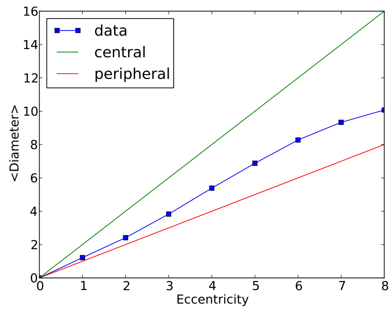
We calculate the roundness of the communities detected by the algorithm using the same metric. Both Amazon and DBLP display a longer structure than those of the Erdös-Renyi networks of the same size. For comparison, the annotated communities in Amazon have the diameter ratious of 1.72, 1.84, and 1.91 for the thresholds of 50, 75 and 100, respectively [24]. The communities detected in these networks are rounder in comparison to the real-life ones, but longer than the random subgraphs.

Surprisingly, the communities returned from the Soundcloud network are the exact opposite: the communities have much smaller diameter than the random network of the same size, implying the existence of considerably more tightly-knit communities in the network. It can be speculated that this could be the result of the asymmetric "follower-artist" relationships in Soundcloud, which create communities of fans with a popular artist in the centre. With such communities, a small diameter can be expected, as the artist in the centre of the community will act as a 'hub' between any two nodes in the community.

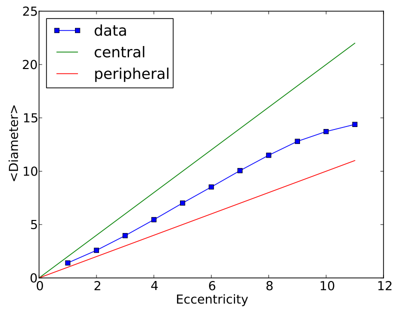
Another aspect of the geometry is the symmetry of the community 'growth' process. If the seed node is in the centre of the community detected by the algorithm, we can conclude that the growth is symmetric. There are many measures of centrality of a node, and for this particular case, we choose eccentricity: the longest distance of the seed to any node in the community. If the eccentricity value is close to the diameter, this means that the seed node is in the periphery. If the eccentricity value is half of the diameter or less, the seed node is considered in the core part of the community. Figure 4.3 shows the comparison of eccentricity values to the diameter. It can be seen that the seed node is closer to the periphery than the central area, and this difference becomes more pronounced as the diameter gets larger.

The fact that the seed node is not consistently in the centre is actually desirable and is a sign of the algorithm's robustness. Suppose the opposite situation, where the algorithm consistently returns communities centered around the seed node. That means, the algorithm would be actually ignoring the community structure, and is only trying to find the nodes that are close to the seed node, regardless of its position in the community. The following chapter provides a more detailed analysis regarding the robustness.

(a) Amazon



(b) DBLP



(c) Soundcloud

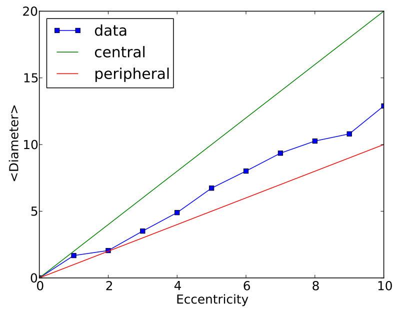


Figure 4.3: Eccentricity of the seed node compared to the reference values. The green reference line denotes the cases when eccentricity is equal to half of the diameter, and the red line denotes when eccentricity is equal to the community diameter.

5

Quality

THERE are multiple indicators of how well a community detection algorithm performs. In this chapter, two main aspects of the quality of community detection is explored: the fitness(conductance) and the stability of the detection process.

Building the community using a bottom-up approach from a seed node is one of the biggest strengths of the local algorithms, giving a considerable flexibility. Surpisingly, however, the dependency on the seed nodes also constitutes a drawback in terms of the reliability of the algorithm. Ideally, a local algorithm is expected to reveal the community structure a seed node belongs to, independent of the node's location within the community. In the previous chapter, our results about the seed node's eccentricity implied that the algorithm possesses some degree of robustness regarding to the seed node. Nevertheless, it is possible to improve the results further by using a 'smart' heuristic to select the seed. We evaluate the conductance scores and possible methods of seed selection in the first section of this chapter

The reliance on the seed node also affects the stability of the community detection process. Let C_u and C_v be two communities detected by a local algorithm, with the seed nodes u and v, respectively. Assume that u and v belong to the same structural community according to several fitness functions, such as modularity and conductance. In an ideal case with clear-cut communities, we would expect $C_u = C_v$. However, this is often not the case, partly because the algorithms are not 'perfect', and partly because the community structure in the real networks is prone to overlaps and unclear boundaries. Nevertheless, the degree of overlap between C_u and C_v is an indicator of stability, and higher values are more desirable.

5.1 Conductance and Seed Selection

Figure 5.1 shows the distribution of conductance when we randomly sample the seed nodes. All three distributions are right-skewed, with the peak at the low values of conductance. Amazon displays the lowest values of conductance among the three networks.

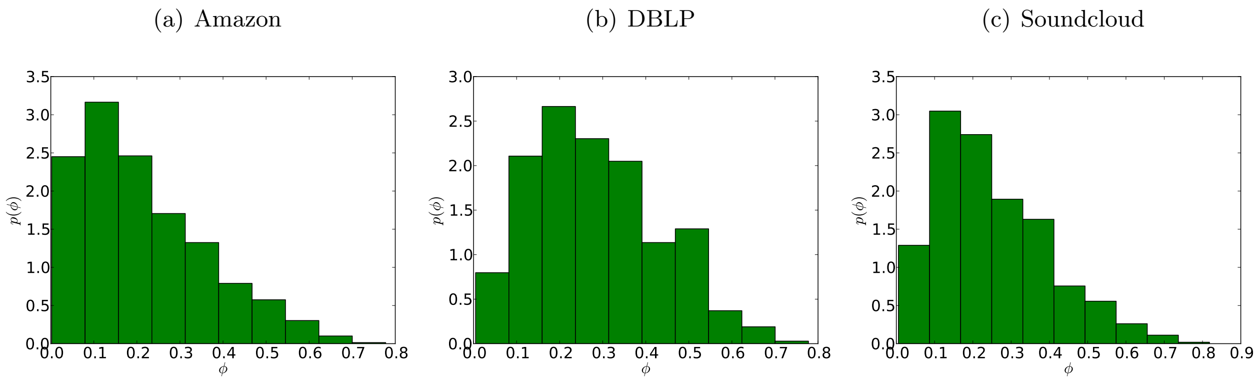


Figure 5.1: Distribution of conductance(ϕ) over randomly sampled seed nodes from the networks.

We first investigate the relation between a seed's degree and the conductance of the detected community. The intuition behind this investigation is that highly connected nodes have a higher chance of being in the 'core' part of the community than the periphery. As such, nodes with high degree are potentially good seeds for the algorithm.

Figure 5.2 shows the relation between the seed's degree and the community fitness. With Amazon and DBLP, there is an obviously decreasing trend. In contrast, Soundcloud shows a jump in conductance as the seed degree is increased from 1 to 5, and shows a constant fluctuation rather than a decreasing trend.

The relation between the seed node's degree and the conductance is also confirmed by the correlation coefficients. Spearman's correlation coefficient[36] yields values between -1 and 1, the former denoting a monotonically increasing relationship and the latter denoting the opposite. The correlation coefficients for the degree and conductance values are -0.22, -0.16 and 0.27 with a p-value of 0; for Amazon, DBLP and Soundcloud, respectively.

While certain high-degree nodes provide good seeds, simply picking the nodes with the highest degrees in the graph is not enough to have a significant decrease in the conductance values. More complex criteria are required to find low-conductance communities.

Chen and Fang propose one such criterion, and demonstrate it using another local community detection algorithm developed by themselves [26]. They select seeds with 'locally maximal' degrees, which have a higher degree than all of their neighbors.

Another method of seed selection, 'minimum conductance neighborhoods', is proposed by Gleich [7]. According to this heuristic, the conductance of each node's egonet

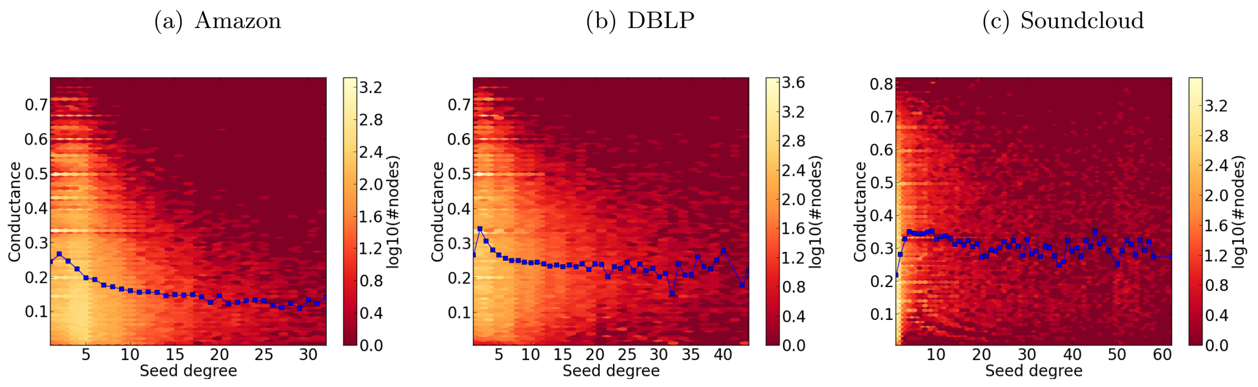


Figure 5.2: Two dimensional histograms showing the distribution of the conductance over the seed node's degree. The colour denotes the number of nodes, in log_{10}. The blue line indicates the mean conductance for a given seed node degree. We display the mean values when there are at least 20 samples for a given degree.

Table 5.1: The values of $< \phi >$ for two different seed selections methods, and randomly sampled nodes for comparison. The values in the parantheses indicate the percentage of the seed nodes to the whole population.

	Locally max. deg.	Min. conductance neighborhoods	Random sampling
Soundcloud	0.16 (%3)	0.19 (%3)	0.25
Amazon	0.13 (%3)	0.15 (%4)	0.21
DBLP	0.1(%0.5)	0.17 (%4)	0.28

(first-step neighborhood) is calculated (called EC). Afterwards, the nodes that have a lower EC value than all of their neighbors are selected as seeds. Also, only the egonets with size 6 or higher are considered.

We try both of the aforementioned methods, and compare them to randomly selected seeds (see Table 5.1). Both seed selection methods allow the algorithm to find communities with a lower conductance value than randomly sampled nodes. Locally maximal degrees provide lower conductance values in comparison to min. conductance neighborhoods, but also covers a narrower range of nodes in the network. For the DBLP network, this difference is significantly more pronounced, in which the min. conductance selection covers 4% while locally max. degrees method covers only 0.5%.

5.2 Stability

We propose the following method to test the algorithm's stability: two random seed nodes (u,v) are selected with various distances and the overlap of the detected communities C_u and C_v are computed. When two nodes are adjacent (distance of 1), an average overlap of nearly 100% would be ideal, since they most probably belong to the same structural community. We would expect the overlap to decrease with distance.

Jaccard similarity is used to measure the overlaps. The Jaccard similarity of two sets A and B is defined as follows:

$$J(A,B) = \frac{A \cap B}{A \cup B}$$

[11]

J(A,B) returns a value between 0 and 1,, with 0 denoting disjoint sets, and 1 denoting identical sets.

The results are shown in the Figure 5.3. Amazon network shows the highest stability, with a Jaccard index of ~ 0.8 for the adjacent nodes. DBLP and Soundcloud have an average overlap of 0.7 for the adjacent nodes, with the Soundcloud data showing more variance compared to other two networks.

As expected from the previously given community diameters, the Jaccard index drops sharply as distance increases. Amazon shows a relatively mild drop in the similarity as the distance increases, while Soundcloud shows the sharpest decline.

(a) Amazon (b) DBLP (c) Soundcloud

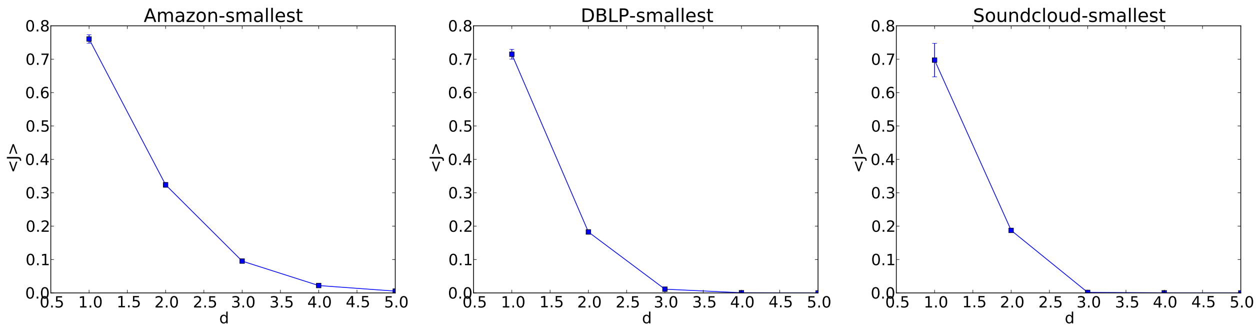


Figure 5.3: The average Jaccard similarity of the detected communities in comparison to the distance between the seed nodes. The bars denote the standard error of the mean.

6

Detecting Annotated Communities

An important application of community detection algorithms is to uncover the actual communities that appear in real-life. The annotated communities in our datasets provide a reference for the performance of such an application. We use the algorithm to detect the online groups in Soundcloud, item categories in Amazon and the publication media in DBLP.

The greatest challenge in this task is the discrepancy between structural and annotated communities. For example, in online networks, anyone can join an open group, so a group can contain nodes that have no connections to anyone else in the group. On the other hand, in co-authoring networks, two people that co-authored a paper appear in the same publication venue. As a result, for the networks like DBLP, the definition of the annotated communities is redundant with the definition of an edge.

We use a basic indicator that can give an idea about how well the annotated and the structural communities match. Let C_G be an annotated community in a graph denoted as $G(V,E)$. Consider the following probability:

$$p_G = p(u \in C_G \wedge v \in C_G | u,v \in V, (u,v) \in E)$$

In other words, p_G is the probability of the two nodes belonging to the same annotated community, given that they are connected by an edge.

The logic behind this measure is as follows: consider an 'ideal' network where each ground-truth community corresponds to a structural community. This means that each community, when taken as a subgraph, consists of one giant connected component without any disjoint nodes. In such a graph, there are two possibilities when two nodes share an edge: 1) They are in a boundary between two communities, 2) They belong to the same community. As the network gets larger, we expect the intra-community edges to greatly exceed the inter-community edges, and p_G to approximate towards 1.

Table 6.1: The probability of adjacent nodes belonging to the same annotated community, p_G.

	Soundcloud	Amazon	DBLP
All nodes	0	0.89	0.89
Nodes that belong at at least one community	0.03	0.98	0.99

As a result, p_G gives us an indicator of how dissimilar the annotated communities are from the structural ones. A value close to 1 does not guarantee high accuracy, but is a prequisite of it.

The Table 6.1 shows the probabilities. The sparseness of community memberships in Soundcloud, which was stated before in the previous chapters, can also be seen on the first row. Soundcloud has a large number of users, small number of groups, and the users are generally not interested in joining the groups. Amazon, on the other hand, has a considerably high groups-to-user ratio, but it turns out that most of these groups cover only a small portion of the node set.

The second row of the table is more important for the classification task, because we focus on the nodes with group memberships. Amazon and DBLP have a $p_G \approx 1$, whereas Soundcloud groups hardly have a match to the edges. Consequently, we can only expect accurate classification for Amazon and DBLP networks.

We reproduce the steps by Leskovec and Yang[4] to test the accuracy of the community detection. From each annotated community, a random node is selected as a seed node. The algorithm is then run on these seed nodes and the nodes corresponding to the first local minimum is chosen as the detected community. The detected community and the annotated community to which the seed belongs to are compared, which is quantified by the F1-score.

The F1-score (also known as F-score) of a classification is calculated using the true positive(TP), false positive(FP) and false negative(FN) values. Let C_D be the sets of nodes in the detected community and C_A be of those in the annotated community.

$$TP = |C_A \cap C_D|, \quad FP = |C_D \backslash C_A|, \quad FN = |C_A \backslash C_D|$$

Following from these, we define precision(p), recall (r) and F-score as[11]:

$$p = \frac{TP}{TP + FP}, \quad r = \frac{TP}{TP + FN}$$

$$F_{score} = 2 \cdot \frac{p \cdot r}{p + r}$$

In the original work, the communities are ranked by various fitness scores. The authors used their algorithm on the top 5000 communities only. We run the algorithm on separately for the full set and the top 5000 communities, compare the latter results with the original work.

During our calculations, we came across a behaviour that was not addressed in the original work: for some of the seed nodes, the sweep curve is monotonically decreasing, so there are no local minima to obtain. For such occasions, we tried four different approaches:

1. Try another seed node from the same annotated community. Do step (2) if all seeds fail to return a community.
2. Return a 'detected' community that contains the seed node only. Note that this returns a precision of 1 for the failures.
3. Return an empty community, which gives an F1-score of 0.
4. Ignore and discard the community.

The table 6.2 shows the results of the F1-scores detected by the algorithm, compared to the results on DBLP and Amazon in the original paper. In order to provide a reference, we also included the results for a global method, fast greedy modularity maximization algorithm by Clauset, Newman and Moore [14]. Due to the incompleteness of Soundcloud data and the computational cost of running the algorithm on such a large network, we decided to run the algorithm on DBLP and Amazon only.

Surprisingly, our implementation returns slightly different average F1-scores for the "top 5000" communities in comparison to the original work. We assume that the authors either used a different method to handle the aforementioned behavior, or they had different a parameter choice for the approximated PageRank algorithm. The local algorithm returns higher F1-scores than the global algorithm in all cases.

As shown in the tables 6.1 and 2.1, the majority of the Soundcloud's userbase does not show any interest in the groups, and the existing groups have a structure that is independent from the connections between the individuals. As a result of these aspects of the network, the algorithm does not perform well on Soundcloud.

Table 6.2: The classification results according to F1-score. The first three columns denote the way of handling the seeds with no local minima: (1) Re-try with another seed (2) Return a single-node community (3) Return an empty community. (4) Ignore the community. The column denoted 'Global' is Clauset et al.'s greedy modularity maximization algorithm, provided as a reference. The last column shows the percentage of the seeds with no local minima. The rows denoted with "LY" are the results from the original paper [4].

	Average F1-score					
Network/Community	**(1)**	**(2)**	**(3)**	**(4)**	**Global**	**undefined%**
DBLP-All	0.37	0.36	0.33	0.41	0.23	20%
Amazon-All	0.44	0.45	0.41	0.45	0.36	9.6%
Soundcloud	0.05	0.06	0.05	0.06	-	12%
DBLP-Top 5000	0.56	0.55	0.53	0.58	0.47	7.8%
Amazon- Top 5000	0.90	0.90	0.90	0.90	0.87	0.02%
DBLP-Top 5000 - LY	0.61					
Amazon-Top 5000 - LY	0.87					

7

Conclusion

We evaluated a state-of-the-art local community detection algorithm by Yang and Leskovec. Our findings provide new insights into the workings of the algorithm, and bring the possibility of further research on local algorithms.

We ran the algorithm on three different datasets; Amazon, DBLP and Soundcloud, in order to find communities with minimum conductance scores. We analyzed certain characteristics of the communities and highlighted the differences and similarities between the annotated and the detected ones. Annotated communities, while showing partial overlaps, follow a different size distribution from the detected communities and have a longer extent. Moreover, we revealed that the communities grow asymmetrically, so that a seed node shifted away from the centre as more nodes are added to the community. We explained that this property of the algorithm is desirable, as it implies that the algorithm shows a degree of robustness to the selected seed. Further looking into the geometry of the communities, we compared the diameters of the communities to Erdös-Renyi subgraphs of the same size. DBLP and Amazon have a larger diameter compared to the random graphs. Compared to Soundarajan and Hopcroft"s results for Amazon, however, the detected communities still have a smaller diameter value compared to the annotated ones. In addition, Soundcloud has communities even 'rounder' than those found in random graphs, due to its communities being centered around artists with a large number of followers.

We then analyzed the quality of the community detection process, and various methods to improve it. We discussed the importance of seed selection to improve the quality and tested two seed selection methods against random sampling. Both "locally maximal degree" and "minimal conductance neighborhood" methods provide communities with lower conductance, and the former provides a better improvement at the cost of a narrower seed set. Then we discussed the effects of seed selection on the algorithm's stability and revealed that the algorithm is fairly stable. Running the algorithm on two adjacent seeds returns communities with a large overlap on average.

Finally, we provided a short discussion about the relationship between the structural and annotated communities. We proposed a basic measure to quantify how well the annotated and structural communities correspond to each other. It was revealed that the type of the network and the definition of the annotated communities play an important role: in the case of DBLP and Amazon, the structural communities provide a better match to the annotated ones, while Soundcloud's online communities have a more disjointed structure. With this information in mind, we tried to use the local algorithm to detect communities in these networks. The local results confirm the aforementioned findings - the algorithm provides relatively accurate matches for DBLP and Amazon, and quite low accuracy for Soundcloud. Compared to the global modularity maximization algorithm, the local algorithm returns higher F1-scores in all datasets.

Community detection remains as an important area of research in the field of Network Science. Local methods have been proposed to address some of the issues with the traditional methods of community detection, such as lack of global information, and computational complexity of working with large networks. The method we evaluated provides promising results, and the area of local algorithms warrants further research by the scientific community.

7.1 Future Work

We provided a number of results and new information related to the workings of the algorithm. There is, nevertheless, possibility of further research into the area. In order to maintain the consistency with Leskovec and Yang, we generally focused on the first local minimum - smallest community returned by the algorithm. Considering all of the local minima or the global minimum could provide interesting results. The detected communities would have a lower conductance value, but a larger size, and could be desirable in the networks where the annotated communities are larger than expected.

Another possible area to explore is the combination of the seed selection methods with the detection of the annotated communities. It could be interesting to see if using locally maximal degree or low conductance neighborhood seeds improve the accuracy of the detected communities. It was also shown that the results are quite dependant on the type of the networks and the communities . This could be further investigated by testing the algorithm on a wider range of datasets and analyzing the results with relation to each other. This would provide better insights into both the algorithm's workings and the networks themselves.

When combined with datasets containing timestamps, local methods could be utilized to study community evolution. This area is often neglected due to the lack of such datasets and the extra computational complexity. Local methods can provide a computationally inexpensive method to study the changes in selected communities over time.

Bibliography

[1] A.-L. Barabási, R. Albert, Emergence of scaling in random networks, science 286 (5439) (1999) 509–512.

[2] D. J. Watts, S. H. Strogatz, Collective dynamics of ‘small-world’networks, nature 393 (6684) (1998) 440–442.

[3] S. Fortunato, Community detection in graphs, Physics Reports 486 (3) (2010) 75–174.

[4] J. Yang, J. Leskovec, Defining and evaluating network communities based on ground-truth, in: Proceedings of the ACM SIGKDD Workshop on Mining Data Semantics, ACM, 2012, p. 3.

[5] A. Clauset, Finding local community structure in networks, Physical review E 72 (2) (2005) 026132.

[6] P. Hage, F. Harary, Eccentricity and centrality in networks, Social networks 17 (1) (1995) 57–63.

[7] D. F. Gleich, C. Seshadhri, Vertex neighborhoods, low conductance cuts, and good seeds for local community methods, in: Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2012, pp. 597–605.

[8] R. Andersen, F. Chung, K. Lang, Local graph partitioning using pagerank vectors, in: Foundations of Computer Science, 2006. FOCS'06. 47th Annual IEEE Symposium on, IEEE, 2006, pp. 475–486.

[9] M. E. Newman, Modularity and community structure in networks, Proceedings of the National Academy of Sciences 103 (23) (2006) 8577–8582.

[10] J. Chen, O. Zaïane, R. Goebel, Local community identification in social networks, in: Social Network Analysis and Mining, 2009. ASONAM'09. International Conference on Advances in, IEEE, 2009, pp. 237–242.

[11] J. Han, M. Kamber, J. Pei, Data mining: concepts and techniques, Morgan kaufmann, 2006.

[12] T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, Introduction to algorithms, MIT press, 2001.

[13] M. E. Newman, Fast algorithm for detecting community structure in networks, Physical review E 69 (6) (2004) 066133.

[14] A. Clauset, M. E. Newman, C. Moore, Finding community structure in very large networks, Physical review E 70 (6) (2004) 066111.

[15] R. Guimera, M. Sales-Pardo, L. A. N. Amaral, Modularity from fluctuations in random graphs and complex networks, Physical Review E 70 (2) (2004) 025101.

[16] P. Pons, M. Latapy, Computing communities in large networks using random walks, in: Computer and Information Sciences-ISCIS 2005, Springer, 2005, pp. 284–293.

[17] S. Van Dongen, A cluster algorithm for graphs, Report-Information systems (10) (2000) 1–40.

[18] E. Weinan, T. Li, E. Vanden-Eijnden, et al., Optimal partition and effective dynamics of complex networks, Proceedings of the National Academy of Sciences 105 (23) (2008) 7907–7912.

[19] M. Girvan, M. E. Newman, Community structure in social and biological networks, Proceedings of the National Academy of Sciences 99 (12) (2002) 7821–7826.

[20] L. C. Freeman, A set of measures of centrality based on betweenness, Sociometry (1977) 35–41.

[21] P. M. Gleiser, L. Danon, Community structure in jazz, Advances in complex systems 6 (04) (2003) 565–573.

[22] D. M. Wilkinson, B. A. Huberman, A method for finding communities of related genes, Proceedings of the National Academy of Sciences of the United States of America 101 (Suppl 1) (2004) 5241–5248.

[23] I. Derényi, G. Palla, T. Vicsek, Clique percolation in random networks, Physical review letters 94 (16) (2005) 160202.

[24] S. SOUNDARAJAN, J. E. HOPCROFT, Use of local group information to identify communities in networks.

[25] F. Luo, J. Z. Wang, E. Promislow, Exploring local community structures in large networks, in: Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence, IEEE Computer Society, 2006, pp. 233–239.

[26] Q. Chen, M. Fang, An efficient algorithm for community detection in complex networks.

[27] L. Pan, C. Dai, C. Wang, J. Xie, M. Liu, Overlapping community detection via leader-based local expansion in social networks, in: Tools with Artificial Intelligence (ICTAI), 2012 IEEE 24th International Conference on, Vol. 1, 2012, pp. 397–404.

[28] J. P. Bagrow, Evaluating local community methods in networks, Journal of Statistical Mechanics: Theory and Experiment 2008 (05) (2008) P05001.

[29] D. A. Spielman, S.-H. Teng, Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems, in: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing, ACM, 2004, pp. 81–90.

[30] R. Andersen, K. J. Lang, Communities from seed sets, in: Proceedings of the 15th international conference on World Wide Web, ACM, 2006, pp. 223–232.

[31] A. Hagberg, D. Schult, P. Swart, D. Conway, L. Séguin-Charbonneau, C. Ellison, B. Edwards, J. Torrents, Networkx. high productivity software for complex networks, Webová strá nka https://networkx. lanl. gov/wiki.

[32] J. Leskovec, Stanford network analysis package (snap), URL http://snap. stanford. edu.

[33] M. Bayer, Sqlalchemy-the database toolkit for python,”, URL http://www. sqlalchemy. org/. Accessed on the 13th of November.

[34] J. Leskovec, Stanford large network dataset collection, URL http://snap. stanford. edu/data/index. html.

[35] F. Chung, Laplacians and the cheeger inequality for directed graphs, Annals of Combinatorics 9 (1) (2005) 1–19.

[36] G. W. Corder, D. I. Foreman, Nonparametric statistics for non-statisticians: a step-by-step approach, John Wiley & Sons, 2009.