Budapest University of Technology and Economics
Faculty of Electrical Engineering and Informatics
Department of Measurement and Information Systems

Anomaly detection in networks

Candidate
Phan Anh Nguyen

Advisor
Ágnes Salánki

2015

TABLE OF CONTENTS

Összefoglaló 5
Abstract 6
1. Introduction to outlier detection in graphs and networks 7
1.1. Motivation 7
1.2. Challenges 9
1.3. Outline and organization 10
2. The concept of outliers in static graphs 11
2.1. Plain and attributed graphs 11
2.2. Types of outliers 12
3. Outlier detection techniques 13
3.1. Oddball 13
3.2. SCAN – Structural Clustering Algorithm for Networks 14
3.3. Autopart – Parameter-Free Graph Partitioning 16
4. Dataset descriptions 18
4.1. Discussion network 20
4.1.1. Defining outliers in a discussion network 20
4.2. Social network 20
4.2.1. Defining outliers in a social network 22
4.3. Road network 22
4.3.1. Defining outliers in a road network 23
4.4. Market basket network 23
4.4.1. Defining outliers in a basket network 24
4.5. Graph features of datasets 24
4.5.1. Degree distribution and the scale-free property 25
5. Application of outlier detection techniques on datasets 27
5.1. Oddball results 28
5.2. SCAN results 30
5.2.1. SCAN at work: social network 30
5.2.2. SCAN at work: discussion and road network 33
5.2.3. Authors' graph choice: basket network 35

5.3. Autopart results 35
5.3.1. Autopart at work: basket network 36
5.3.2. Limitations of Autopart 38
6. Extending the techniques to dynamic graphs 40
6.1. Oddball in a dynamic context 40
6.2. SCAN in a dynamic context 41
6.3. Autopart in a dynamic context 42
7. Summary and conclusions 43
7.1. Future work and possible improvements 44
Index of figures 45
Index of tables 47
Bibliography 48
Appendix 54
Tables pertaining to the parameter tuning of SCAN 54

HALLGATÓI NYILATKOZAT

Alulírott Nguyen Phan Anh, szigorló hallgató kijelentem, hogy ezt a szakdolgozatot/diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen, a forrás megadásával megjelöltem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2015. 12. 12.

.....

Nguyen Phan Anh

Összefoglaló

A klasszikus sokdimenziós adatkészletek anomáliáinak (kilógó pontjainak) detektálására számos hatékony algoritmus létezik, ezek nagy része azonban feltételezi, hogy az egyes rekordok egymástól függetlenek. A detektáló algoritmusok egy speciális részcsoportját alkotják azok a módszerek, amelyek az egyes rekordokat valamilyen kontextusban összekapcsolják, például időt vagy fizikai elhelyezkedést reprezentáló dimenziókon keresztül.

Az egyre növekvő, nyílt hozzáférésű hálózati adatsoroknak köszönhetően az utóbbi évtizedben az anomália detektálás területe dinamikus fejlődésnek indult. Ennek feladata a szokatlan gráf minták és elemek felfedezése, amelyek a közösségi hálózatok véleményvezéreinek keresésekor vagy bűnügyi hálók felderítésekor kiemelt jelentőségűek lehetnek.

A szakdolgozat magában foglalja a jelenleg jellemzően használt detektálási algoritmusok feltárását, illetve azok közvetlen használhatóságának vizsgálatát több választott szakterületen: az internetes fórum közösségek, a szociális hálók, a fizikai úthálózatok és a vásárlói kosár tartalmát reprezentáló hálók területén.

Abstract

Anomaly detection in networks is a dynamically growing field with compelling applications in areas such as security (detection of network intrusions), finance (frauds), and social sciences (identification of opinion leaders and spammers). Its applicability is propelled by an ever increasing availability of network data: the ubiquity of handheld devices gave rise to a plethora of community and network-based services that in turn generate a wide spectrum of graph data in the most different domains.

This work addresses the problem of outlier detection in plain, static graphs. We analyze three fundamental, a feature, a network structure and an information theory driven anomaly detection technique. We demonstrate their effectiveness and results on four real-world datasets from the domains of discussion, social, spatial, and market basket networks. Each network's unique characteristic is presented along with an overarching set of features allowing for network comparison. Finally, we offer an outline to extend the examined anomaly detection techniques to the dynamic context of graphs. We conclude with a discussion on possible directions of future work.

1. Introduction to outlier detection in graphs and networks

Hawkins defined the concept of an outlier [1] in the following way:

> *"An outlier is an observation which deviates so much from the other observations as to arouse suspicions that it was generated by a different mechanism."*

In the field of data mining, outliers are also referred to as anomalies, abnormalities, discordant observations, or deviants. Other application domains may use terms like exceptions, surprises, peculiarities or contaminants. All these terminologies are capturing a deviation from an assumed normal data model. The detection and characterization of that deviation provide useful domain-specific insights. Among other domains, intrusion detection, fraud detection and spam filtering are relying on and applying outlier detection effectively.

Outlier detection is related to, but distinct from noise removal, which aims to exclude unwanted data values originating from errors or inaccuracy of measurements. Considering this definition, noise represents the intermediate range between normal data and true outliers. It could be modeled as a *weak outlier* [2], the deviation of which is not yet significant enough to be of interest for the analyst.

Another related topic is novelty detection, *"the identification of new or unknown data,* [unknown features] *that a machine learning system is not aware of during training"* [3] [4]. The main difference between outliers and novelties is that the latter is typically integrated into the normal data model after its detection.

This work considers outliers as extraordinary observations that bear significance in their contexts. We search for these particularities, highlight them in their environment and put them under detailed examination in order to gain insights about what roles they might be fulfilling. We regard them as the exact opposite of noise that is usually filtered and removed. We do not think of them as novelties, for the contexts at work are static and do not change in the dimension of time.

1.1. Motivation

[5] highlights the potential for outlier detection in graphs: the nature of outliers are relational in certain domains. Performance monitoring is an example where the failure of a machine could cause the breakdown of others dependent on it. Therefore some data objects cannot be treated as points lying in a multi-dimensional space independently. Graphs, on the other hand, represent that inter-dependent nature of the data well.

Interdependency, a key component of social networks, is heavily investigated in the fields of social network analysis and network analysis in general. In the past decade, these have been gaining momentum and significance, and so has outlier detection in graphs. Internet capable devices are becoming ever more ubiquitous, encouraging the development of community based services, all of which are aiming to create a large base network of users.

Facebook incentivizes individuals to connect along friendships [6], Twitter provides a platform for microblogging [7], LinkedIn specializes in professional and career relations [8], Couchsurfing facilitates hospitality exchange [9], Uber assists transportation in cities [10], Tinder organizes dates based on proximity [11], and the list would continue endlessly. As several of these companies hold a large amount of data on user interactions, research and data analysis are intensively pursued for product development and usage trend comprehension.

A wide spectrum of fields builds on anomaly detection for successful operation, using either conventional multi-dimensional or relational data. Some use cases and the context of outliers are briefly summarized in Table 1-1.

Focus of detection	Example	Description
Intrusion	Network intrusion	Ding et al. identified network intrusions by detecting anomalous network flow data, communication that does not respect community structure [12].
Fraud	Subscription	Cortes et al. revealed subscription fraud in telecommunication network, building on the assumption that *"fraudsters tend to be closer to other fraudsters than random accounts are to fraud"* [13].
	Fake personalities in auction sites	Chau et al. uncovered fraudulent personalities in networks of online auctioneers by leveraging user level features along with network level features that capture interactions between different users [14].
	Trading fraud cases	Li et al. recognized distinctive patterns – black-holes and volcanoes, sets of nodes that contain only in-links and out-links, respectively, to those sets from the rest of the graph – in traders' network to unveil cross-account collaborative fraud cases [15].
Spam	Web pages	Carlos et al. detected web spam pages based on link-based and content-based features as well as the topology of the web graph by exploiting the link dependencies among the web pages. They found that linked hosts tend to belong to the same class of spam or non-spam [16].
	Messages in social networks	Gao et al. filtered spam messages in online social networks using incremental clustering, based also on network-level features such as the interaction history between users [17].

Table 1-1. Applications of outlier detection

1.2. Challenges

What is considered to be normal or anomalous is not straightforward. Therefore navigating on the boundary between the two is difficult: an anomalous observation could appear to be normal, and vice versa. Consider spam and hijacked account detection in online social networks. A user with a conversation history containing only one-on-one communication initiates a group conversation with all his/her contacts. One interpretation could be that the account was hacked and was thus used to spread spam messages to infect additional users. The activity would be flagged, and actions might be taken against it. However, it could also have been a genuine help request to complete a survey, which requires a larger pool of audience.

The exact notion of anomaly varies from domain to domain. In the *wait-for graph* of deadlock detection in relational database and operating systems, a small cycle implies a blocking that needs to be resolved, otherwise normal execution might completely halt [18]. Conversely, in the friendship network of communities, small cycles are common creations of typical social behavior: *"If two people in a social network have a friend in common, then there is an increased likelihood that they will become friends themselves at some point in the future."* [19] If such a friendship is established, the three persons would form a cycle.

Anomalies may be the result of malicious actions. In such situations, the adversaries often adapt themselves to conceal their true intentions and *appear to be normal*. This makes the detection even more difficult. In the example of trading frauds [15], a trading ring – a group of traders that are engaged in illegal activities – tries to align to standard commercial behavior and relies on the very high number of transactions to conceal itself.

Most datasets do not have ground truth. Although there are plenty of datasets that contain relational data, most of these *lack the predefined knowledge* behind them to verify that the detected unusualities are indeed anomalies. Additionally, classifying unlabeled data faces obstacles of ensuring a consistent, standard labeling of data, a result highly dependent on the annotators. Imagine a task of scoring the inappropriateness of forum posts on a scale of ten: were the effort saved by outsourcing the task on the e.g. Amazon Mechanical Turk [20], the new challenge of normalizing scores would appear in its stead.

Mislabeling data could lead to adverse consequences. In health care systems, dismissing a seriously ill patient as healthy could cause fatal consequences. In astronomy, if imaging instruments suspect most unknown signs to be new celestial objects, the astronomers cannot keep pace with verifying the potential discoveries [21].

It is difficult to cope with the scale of the datasets. The enormous size of the networks and the immense number of transactions they generate require approaches that are *both efficient and scalable.*

Class imbalance and asymmetric error have to be taken into consideration. Amidst the data created through normal functioning of the systems are hidden the outliers, which *constitute only a small fraction of the whole.*

1.3. Outline and organization

Section 2 provides a more specific definition of outliers in the context of static graphs. Section 3 follows with the description of fundamental outlier detection techniques. Our assembled network dataset as well as additional referenced datasets are introduced in section 4. Section 5 demonstrates and analyzes the application of outlier detection techniques on the datasets. Section 6 reflects on the possibility of extending the techniques to dynamic graphs. Finally, a brief summary and the conclusions are presented in Section 7.

2. The concept of outliers in static graphs

In this section, we formulate the concept of outliers in *static snapshots of graphs*. The problem is to find graph objects (nodes/edges/subgraphs) that are different from the majority of the other reference objects in the graph. We make the distinction between *plain graphs* and *attributed graphs*, each of which carries a different particularity. In the former case, particularity could be isolation (far-away points in an n-dimensional space) or unusual structural characteristics, such as the presence of certain patterns (cliques and stars). In the latter case, it could be a rare combination of labels (e.g. a scholar having publications in remotely connected fields such as biology and astrology).

2.1. Plain and attributed graphs

Plain graphs consist of only nodes and edges, they do not hold more information than the graph structure. Attributed graphs, on the other hand, may have features associated with their components. For example in a social network, users may disclose their gender and favorite hobbies, and connections between users may be labeled to designate a relation of acquaintances/friends/family members.

Anomaly detection techniques on plain graphs rely exclusively on structural information. The survey in [5] categorizes the techniques as *feature-based*, *proximity-based* and *community-based* depending on their concepts of similarity between graph objects (Table 2-1).

Techniques	**Description**
Feature-based	Extracts features like in/out degrees, *betweenness centrality* [22], *clustering coefficient* [23], *modularity* [24], etc.
Proximity-based	Measures closeness of graph objects. PageRank [25] is a famous example for rating web pages based on their linking from one to another.
Community-based	Finds densely connected groups of nodes.

Table 2-1. Anomaly detection techniques in plain, static graphs

Anomaly detection techniques on attributed graphs differ in that they rely on structural as well as labeled information to find patterns and spot anomalies. Their categorization is the same as in the case of plain graphs. However, as a result of the additional information provided by labels, the number of extractable features are higher, graph objects could either be connected structurally or through similar labeling, and communities could be defined by density as well as class labels.

2.2. Types of outliers

Three types of outliers are differentiated: *node, linkage* and *subgraph outliers* [26].

Node outliers are vertices with unusual characteristics in the graph. They could be defined in various ways: node outliers may be structurally (in)significant, by being isolated from the rest of the vertices, or by being in the center of a star shaped pattern. In attributed graphs, they could hold a rare combination of categorical attribute values or simply differ in labeling compared to their neighbors.

Linkage outliers are edges with unusual characteristics in the graph. These are generally defined as edges that connect two disparate, but each densely connected partitions/communities of the network.

Subgraph outliers are defined as parts of the graph which exhibit unusual characteristics with respect to the normal patterns in the complete graph. They could form particular patterns such as stars or cliques, or they could simply be distinctively different from the frequent patterns observed in the graph. In attributed graphs, subgraph outliers could also be defined based on the repetitions in the labels on the nodes.

3. Outlier detection techniques

In the following subsections, we discuss the basic outlier detection techniques in static, plain graphs. *Oddball* [27] is a feature-based technique for identifying node outliers. *SCAN* [28] and *Autopart* [29] are community-based techniques that are detecting node and linkage outliers, respectively.

3.1. Oddball

The aim of this technique, as the same suggests ("odd ball"), is to find anomalous nodes. It builds its solution on the analysis of *ego networks*. It takes in a graph as input, and produces a list of node outlier candidates as output.

Ego network is defined as the one-step neighborhood around a central node "ego". It includes the central node, its direct neighbors and all the edges among these nodes. In other words, the ego network is the subgraph of *one-step neighborhood of the central node*.

Figure 3-1 highlights the ego network of the aqua-colored node. The red nodes are the neighboring vertices.

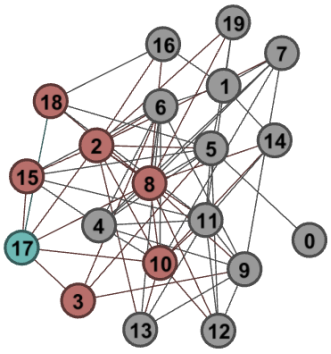


Figure 3-1. Ego network

Our analysis focuses on nodes partaking in patterns. The nodes whose neighbors are well connected (near cliques, Figure 3-2) or sparsely connected (near stars, Figure 3-3) are considered particular: in social networks, the previous indicates a regular and intense interaction in the history of the clique members; the latter suggests an influential person in a central position, who is capable of reaching a wide, but independent audience.

The technique can be broken down to four parts.

1. *Ego network extraction*: get all ego networks from the input graph.
2. *Feature selection*: choose features of ego networks that could indicate anomalies; compute these features for all ego networks.
3. *Analysis*: pinpoint anomalies using any outlier detection method in point clouds [30] [31].

Two of the features the authors presented that are successful in detecting outliers are *number of nodes* and *number of edges* in the ego network. (Their attempts using number of neighbors of degree 1, principal eigenvalues of ego networks, and other features did not yield significant insights. [32]) Plotting the number of nodes against the number of edges reveals near cliques (Figure 3-4) and stars (Figure 3-5). The green line represents the maximum number of edges in an n node ego network ($n * (n-1)/2$), while the blue line the minimum number of edges ($n-1$). The closer the ego network lies to the lines, the more remarkable it is likely to be.

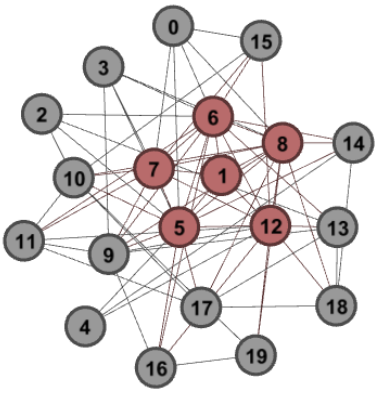


Figure 3-2. Clique in graph *A*

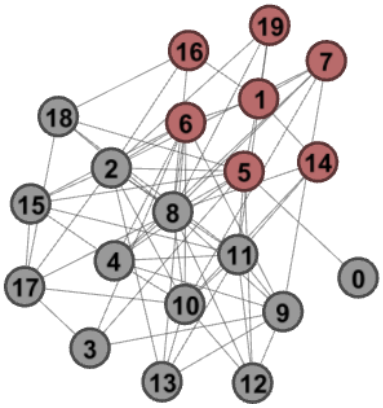


Figure 3-3. Star in graph *B*

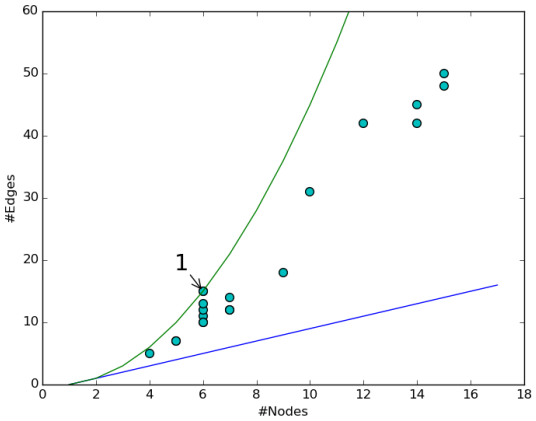


Figure 3-4. Revealing cliques in graph *A*

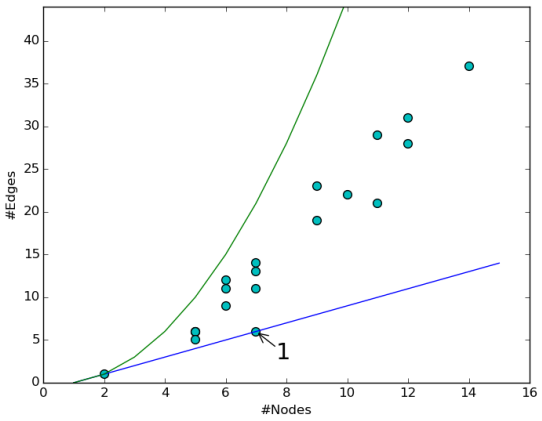


Figure 3-5. Revealing stars in graph *B*

3.2. SCAN – Structural Clustering Algorithm for Networks

The purpose of this technique – similarly to Oddball's – is to identify node outliers. The authors distinguish two types of nodes that play special roles:

- *Outliers*: nodes that are marginally connected to *clusters*
- *Hubs*: nodes that bridge clusters

Clusters are groups of nodes that have a dense set of edges running within the clusters, and have a relatively low number of edges that run between the clusters. They are densely connected graph parts.

In the context of this algorithm, hubs play a significant role due to their interconnecting properties. They are the targets in viral marketing, individuals who exert great influence in the process of opinion or information spreading. In contrast, the term of outliers *in this context*, bear no importance and may be discarded or isolated as noise in the data.

SCAN takes in a graph and two parameters (ε, μ) as input, and yields a list of clusters, hubs and outliers as output. ε captures the rigorousness of the condition of a node to be considered part of a cluster. An analogy would be that a sect imposes a requirement on newcomers that they must have at least ε number of common acquaintances with an existing member to join. On the other hand, μ determines the minimum number of vertices

a cluster must have. An illustration would be a general rule that states groups must have at least μ number of members to be legally considered a sect.

In the Figure 3-6, the algorithm places all nodes into a single cluster with $\varepsilon = 0.6, \mu = 2$. A low ε draws a low line of requirement for being a member of a cluster, thus all nodes would be grouped together. Increasing ε tightens the coherence inside a cluster, and the initial all-encompassing cluster would be broken up to smaller groups. In the Figure 3-7, the original interpretation is retrieved: clusters {1, 2, 3, 4, 5, 6} and {8, 9, 10, 11, 12, 13}, 7 as a hub and 14 as an outlier. Figure 3-8 further decomposes the two clusters, thus identifying 10 also as a hub, because it neighbors two clusters. At the extreme case in Figure 3-9, the conditions to form a cluster are so high, that none was identified, thus all nodes are taken to be outliers. It is worth to note that $\varepsilon = 0.7$ and $\mu = 7$ would also lead to the extreme case, because there is no combination of seven nodes that are closely connected.

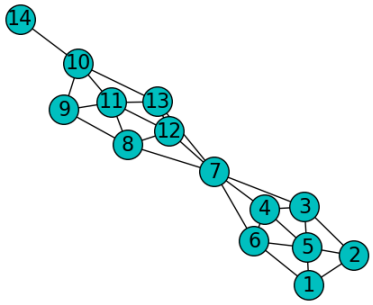


Figure 3-6. A network with two clusters, a hub and an outlier

The algorithm works in the following way. At the beginning, all nodes are labeled as unclassified. SCAN performs one pass of the nodes, and classifies them either as a cluster member or a non-member based on structure connectivity (for an exhaustive definition, see [28]). At the end, when all clusters are found, the non-members are classified further as hubs or outliers, based on the cluster membership of their neighbors. (Remember, hubs are nodes that bridge separate clusters.)

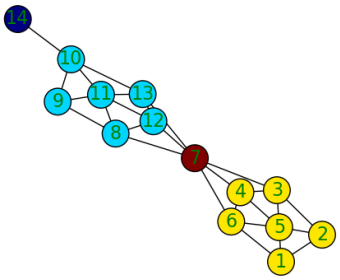


Figure 3-7. $\varepsilon = 0.7, \mu = 2$

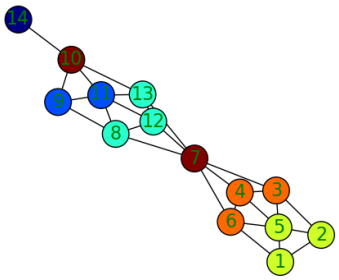


Figure 3-8. $\varepsilon = 0.8, \mu = 2$

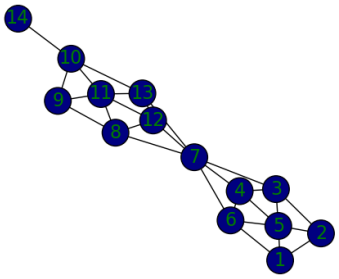


Figure 3-9. $\varepsilon = 0.9, \mu = 2$

Analyzing the example network with Oddball would highlight 14 as an isolated node, for its ego network consists of only two nodes and one edge. However, 7 would not stand out among the rest of the nodes. (Figure 3-10). Its ego network has a ratio of edges to nodes similar to the ego networks of other nodes.

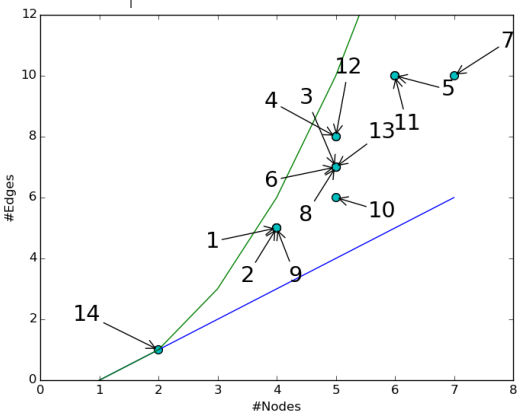


Figure 3-10. Oddball at work

3.3. Autopart – Parameter-Free Graph Partitioning

Autopart is capable of identifying anomalous edges. Its primary purpose is to (automatically) partition the graph into clusters without user intervention – hence the parameter-free attribute. After finding a partitioning – a set of clusters – it proposes a method to measure the outlierness of edges that bridge separate clusters. The algorithm takes in a graph as input, and produces a partitioning and a list of link outlier candidates as output.

The main idea is to measure the goodness of the partitioning with how well it can be compressed and then transmitted. A better compression results in a lower transmission cost, which implies a good partitioning. The application of information theory to outlier detection is frequently used in the classical, multi-dimensional context [33], and has been extended to graphs in the Autopart algorithm.

This technique specifically uses the adjacency matrix as graph representation. A partitioning is a reordering of rows and columns in a way that nodes belonging to the same cluster are placed next to each other (Figure 3-11).

In consequence, the adjacency matrix is broken down to blocks: the squares located on the diagonal of the matrix capture the edges running inside the clusters; the rectangles represent the edges bridging the corresponding clusters.

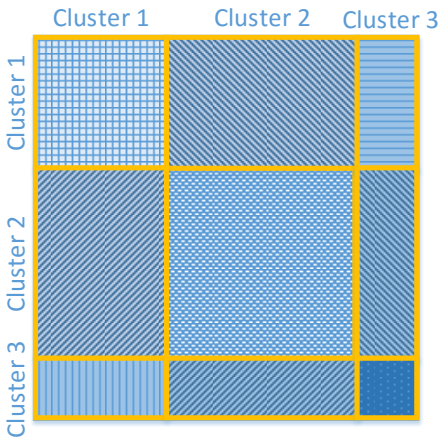


Figure 3-11. A partitioning of an adjacency matrix

A good partitioning yields homogeneous blocks, which in turn, can be compressed efficiently. At the extreme, with n clusters, there could be n^2 perfectly homogenous blocks. However, the compression scheme accounts for this as well.

The author proposes a two-part code for the adjacency matrix. The total cost is comprised of a *description cost* and a *code cost*.

Description cost holds the information about the rectangular/square blocks. It is the transmission cost of the following terms:

- number of nodes
- node permutation (which row represents which node)
- number of clusters
- number of nodes in each cluster
- number of ones in each block (the number of edges bridging the given clusters)

Code cost holds the information about the content of the blocks. It is the transmission cost of the blocks calculated using the Shannon entropy function [34].

Description cost penalizes a high number of blocks; on the other hand, code cost penalizes heterogeneous blocks. Thus a good partitioning maintains a balance between a low number of clusters and a high homogeneity of blocks. The algorithm finds the tradeoff point between the two aspects and yields a construction with the minimal total cost.

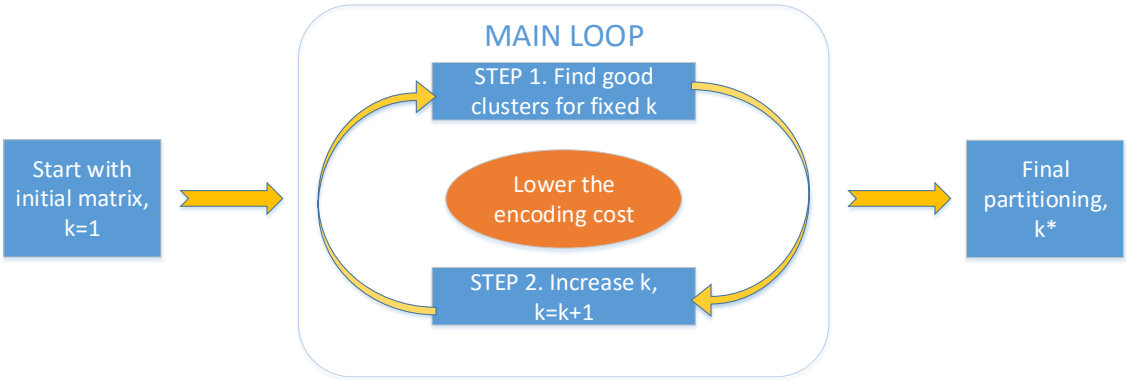


Figure 3-12. Steps of the Autopart algorithm

It starts with an initial adjacency matrix, where all nodes belong to one cluster (k = 1). Inside the main loop, the total cost is iteratively reduced until no improvements can be made, and the final partitioning together with the final cluster count k^* is outputted (Figure 3-12). The iterative reduction is made up of two steps: first, a good partitioning given the number of clusters is found. Second, the number of clusters is increased to allow for better partitioning.

Once the final partitioning is found, Autopart marks the anomalous edges. Outliers show deviation from the normal patterns, so they hurt attempts to compress data. Therefore those edges, whose removal reduces the total cost the most are marked as outliers.

4. Dataset descriptions

In order to validate the practical usability of the algorithms, we applied them on four datasets. The first is the discussion community network of a Hungarian news portal [35], which we assembled using the available forum activity data. This choice reflects our ambition to work with a Hungarian dataset.

The second dataset, the social network of Facebook users [36] was chosen for its availability of ground truth. The ground truth consists of manually-labeled circles – friendship groups defined by social network users based on their relation, history and background with their peers – which allows us to test partitioning (clustering) algorithms. Since two out of three basic anomaly detection algorithms presented in Section 3 are based on cluster identification, it becomes essential to verify and measure their accuracy and performance.

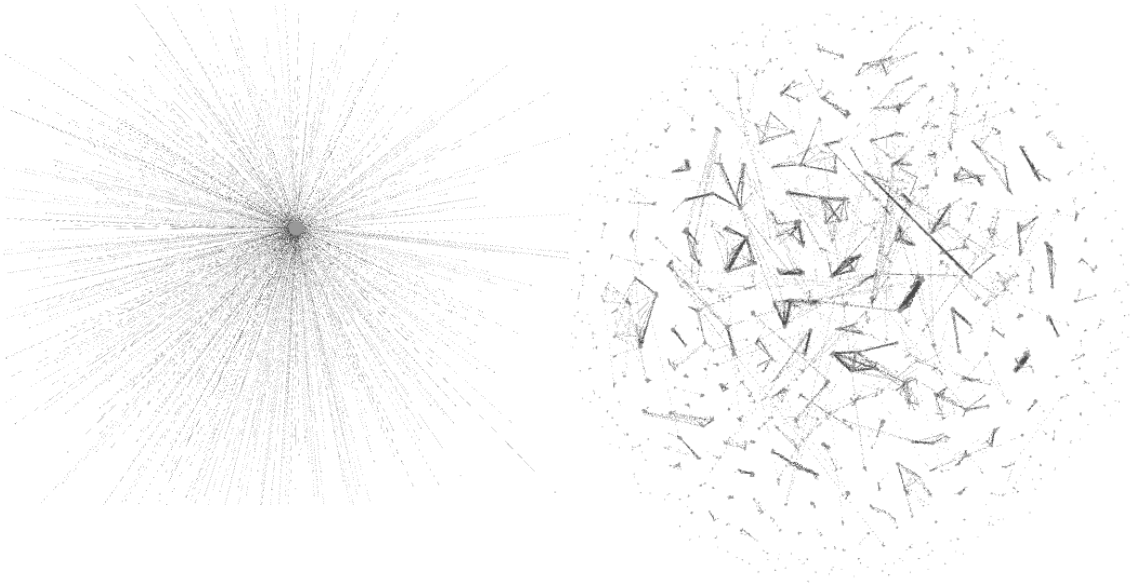
The third dataset, the road network of Stockton city in San Joaquin County (California, U.S.) [37] was selected for its apparent difference in structure, compared to the previous two. While it only takes a registration to create a new node, and reply on a comment/confirmation of a friend request to link two existing nodes in the online discussion/social network, careful architectural, material and civic planning is required for road construction. In addition to the different pace of network evolution, the number of roads starting from the same point (for example a crossing point) is physically constrained, and two remote points cannot be connected directly. In consequence, this network will have a distinctly different layout and structural metric values.

The final dataset is a market basket network, a network of books about U.S. politics sold on Amazon [38]. Compared to the previous three datasets, this is much smaller in size, allowing us to test and analyze algorithms that do not scale well.

Figure 4-1 visualizes the large networks using OpenOrd [39], a force directed layout algorithm for distinguishing clusters in large graphs. It can be seen that the discussion network is dominated by a single cluster, the social network is comprised of multiple, smaller clusters, and the road network is evenly distributed, showing few signs of clustering.

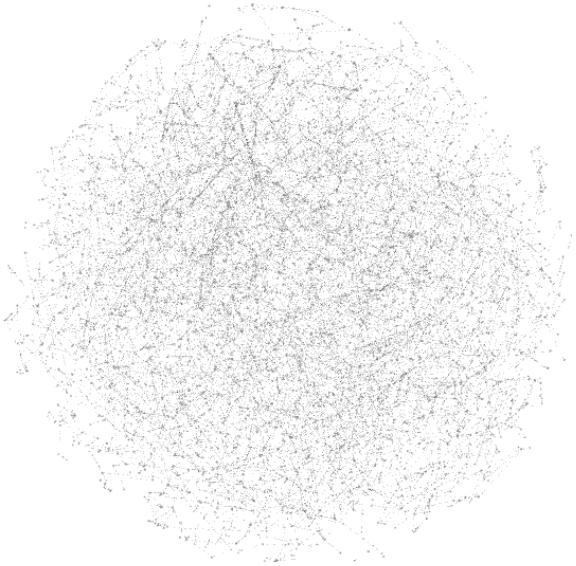
Figure 4-2 displays the small network of market basket data using ForceAtlas [40], also a force directed layout, developed for the visualization of small and medium sized networks.

We review the four datasets in detail in the following subsections.



a) Discussion network

b) Social network



c) Road network

Figure 4-1. Introduction of the large datasets

Figure 4-2. Small network of books

4.1. Discussion network

We examine an online news portal that publishes articles in a broad range of topics, with the majority of articles open to unmoderated commentary. The site uses Disqus [41] – a blog comment hosting service for web sites and online communitites – to provide a framework for posting comments. Using the REST API of Disqus, we retrieved the complete commentary history of the year 2014.

We constructed a simple undirected graph from the commentary history in the following way: nodes represent users, while edges represent replies one user posted in response to another user's comment. Each node is labeled with its associated username. The repetition of replies are not reflected in the addition of edges, thus this model can capture neither the frequency, nor the quality of responses.

4.1.1. Defining outliers in a discussion network

We search for two types of user characteristics that exert remarkable impact and at the same time show deviating traits from the majority of users. First, opinion leaders in the blogosphere are those that disseminate new information to the masses and capture the most representative opinions in the social network [42]. In the context of discussion communities, we regard the opinion leaders as the influential users who interpret and analyze the published news articles in a way that evoke approving and/or supplementary responses. Second, spammers are those who engage in antisocial behavior, meaning that they negatively affect other users by trolling, flaming, bullying, and harassing [43].

Note that both opinion leaders and spammers belong to the category of node outliers.

4.2. Social network

Facebook is an online social networking service where registered users can connect with each other. In the network dataset, nodes are users, and edges are undirected friendship connections between the users.

The dataset was assembled from the friendship data of the Kaggle [44] data science competition participants. According to the *small-world phenomenon* [45], or the *"six degrees of separation"*, social networks exhibit a certain characteristics that any two members should be connected by a relatively short path [46]. However, probably because of the national diversity of the participants, who were scattered across different geographical locations, this network is comprised of several larger components, leaving this the only unconnected network among our datasets.

Small-world phenomenon is the observation that any two individuals in a social network are likely to be connected through a short sequence of intermediate acquaintances.

Small-world networks are characterized by relatively short *diameters*, *radii* and *average shortest path lengths*. Figure 4-3 shows that both the discussion and the social network have relatively short network distances, while the road network has values greater by an order of magnitude. The basket network displays the shortest distances, which in this case we attribute to its size rather than to its domain.

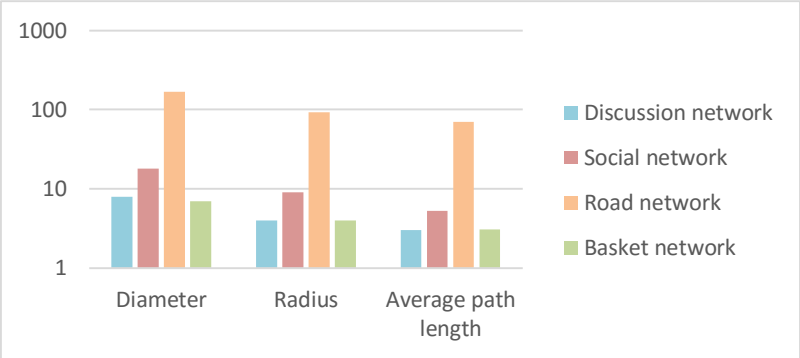


Figure 4-3. Network distances

Diameter is the longest shortest path between network nodes.

Radius is the minimum *eccentricity* of the network nodes.

Eccentricity of a node n is the maximum distance between n and any other node of the network.

The main difference between the social network and the discussion network is that the previous revolves around presenting user activities of friends for friends, while the latter focuses on displaying articles of journalists for everyone. Therefore we expect the concept of densely connected circles in the social network to be more dominant. In terms of graph features, the average *clustering coefficient* of the social network is expected to be higher, which means it is more likely to contain cliques, or near-cliques. Figure 4-4 confirms our anticipation.

Clustering coefficient is a measure of how nodes are embedded in their neighborhood. Formally,

$$C_i = \frac{\left|\{e_{jk}: v_j, v_k \in N_i, e_{jk} \in E\}\right|}{\frac{k_i(k_i-1)}{2}}$$

where C_i is the clustering coefficient of node i, $e_{jk}$is the edge between node v_j and v_k, N_i is the neighborhood of node i, E is the set of edges of the graph, and k_i is the number of neighbors of node i. Note that members of a clique would have a clustering coefficient 1.

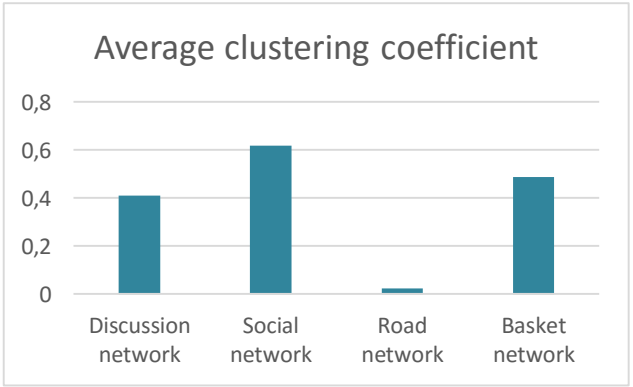


Figure 4-4. Average clustering coefficient of networks

4.2.1. Defining outliers in a social network

In a context where we have ground truth about existing circles, we attribute significance to and search for those users, who are connected to several tightly knit groups. The fact that they are not confined to one single circle means that they have access and view of multiple group activities. They are the ones bridging the circles, constituting the *weak ties* [47] for those who are involved exclusively in one circle.

Consider the example on Figure 4-5. Except for node 1, none has information, or connection to other groups. In exchange, node 1 is not as embedded in, or completely part of any of the groups. It is, however, providing the weak ties (1-2, 1-6, 1-7, 1-11, 1-12 and 1-16) for the respective members of the groups, serving as the sole information channel.

This illustrates the idea that *"the best job leads come from acquaintances rather than close friends"* [48]. Close friends move in the same environment and therefore are exposed to similar news and information. A new change of environment might be brought about by acquaintances, who have access to information we otherwise would not necessarily hear about.

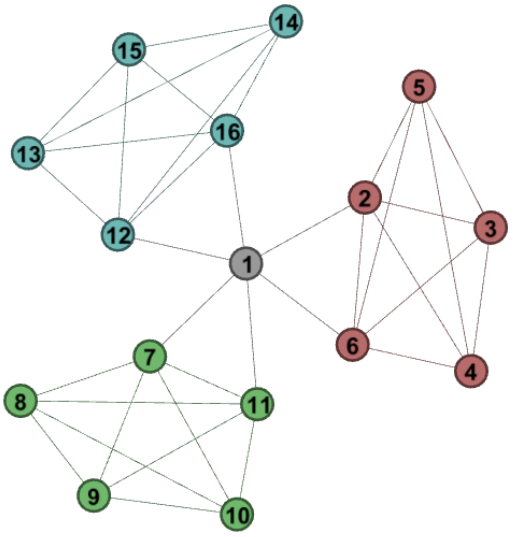


Figure 4-5. The strength of weak ties

4.3. Road network

The road network of Stockton city in San Joaquin County (U.S.) [37] combines geographic and graph-theoretic information in one structure. Nodes represent road intersections, and edges represent road segments that join such points. Generally, road networks contain geographic information: vertices are labeled with longitude and latitude coordinates, and edges are labeled with their length.

As demonstrated earlier (Figure 4-3), a significant difference between the road network and the online networks are network distances; the road network is a non-small-world network. This could be attributed to the low-paced network evolution as well as the spatial limitations: it certainly takes numerous hops to drive from one secluded corner to the spatially opposite point of the city.

Another noteworthy difference stemming from spatial constrains is that intersections can be the endpoint of a limited number of roads. In contrast, forum commenters have the freedom to reply to any other user, and social network users are free to befriend anyone they find. This is reflected in the node degree measures, where there may be a whole order of magnitude in difference (Figure 4-6).

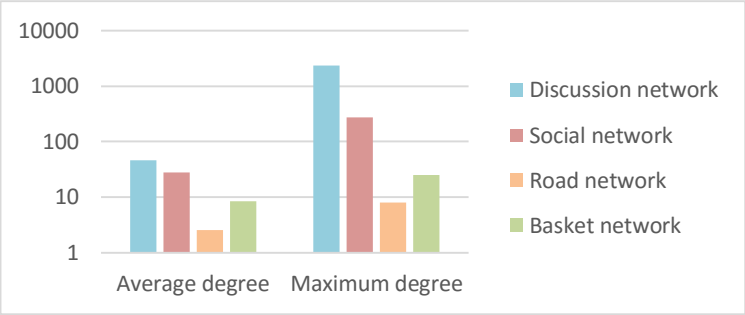


Figure 4-6. Degree measures

4.3.1. Defining outliers in a road network

Objects in spatial networks – in our case, intersections and road segments – usually have two dimensions along which attributes are defined: (i) *spatial attributes* include location, altitude and other topological properties; (ii) *non-spatial attributes* include values of e.g. traffic or climate metrics.

Our dataset contains exclusively topological properties of spatial objects. Consequently we define the outliers based on graph connectivity: we assign special characteristics to and search for nodes with high number of neighbors, and edges that bridge otherwise separate partitions of the network.

4.4. Market basket network

The basket network of political books [38] is the result of market basket analysis, a technique closely related to association mining [49] in the field of data sciences. The analysis rests on the assumption that from a collection of products commonly bought together, it could be inferred what else a consumer might be interested to buy. The aim is to leverage this information to build recommendation systems capable of delivering targeted offers. Furthermore, *"it can suggest new store layouts; it can determine which products to put on special; it can indicate when to issue coupons"* [50].

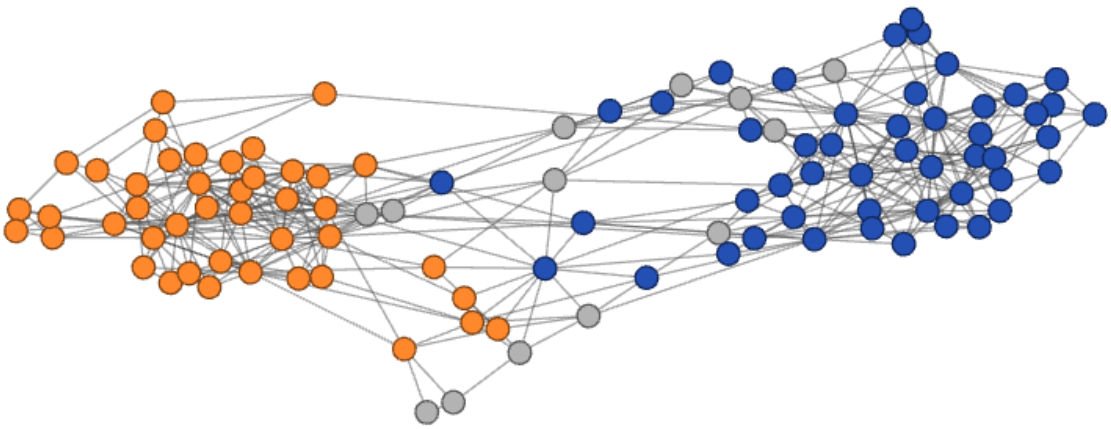


Figure 4-7. Book labeling

Nodes in the network represent books, edges represent frequent co-purchasing of those books. The books were pre-labeled as liberal, neutral, or conservative [51] (Figure 4-7).

4.4.1. Defining outliers in a basket network

Figure 4-7 shows two clearly discernable liberal and conservative clusters of books, in-between which a couple of neutral books are scattered. This suggests that books labeled as the same category are often purchased together (except for neutral books that form no distinct pattern). While there are a few conservative books drawn near to the liberal clusters, the same does not hold for liberal books. In this context, we target edges that connect otherwise densely connected partitions. The results might coincide with co-purchases that contain both liberal and conservative books.

4.5. Graph features of datasets

We summarized the network features of our datasets in Table 4-1. It provides an overview of the networks' distances, clustering and degree measures.

Feature	Discussion network	Social network	Road network	Basket network
Graph type	Undirected | Simple			
	Connected	Unconnected	Connected	Connected
#Nodes	15109	26457	18263	105
#Edges	350507	372227	23797	441
Avg. degree	46.397	28.138	2.536	8.400
Max. degree	2324	276	8	25
Diameter	8	18	167	7
Radius	4	9	93	4
Avg. path length	3.025	5.25	70.577	3.079
Avg. clustering coefficient	0.411	0.619	0.023	0.488

Table 4-1. Graph features of datasets

4.5.1. Degree distribution and the scale-free property

Scale-free network is a network whose node degree exhibits a power law distribution.

In scale-free networks, the probability that a node has k links $P(k)$ follows a power law $P(k) \sim k^{-\gamma}$. *"This feature was found to be a consequence of two generic mechanisms: (i) networks expand continuously by the addition of new vertices, and (ii) new vertices attach preferentially to sites that are already well connected."* [52]

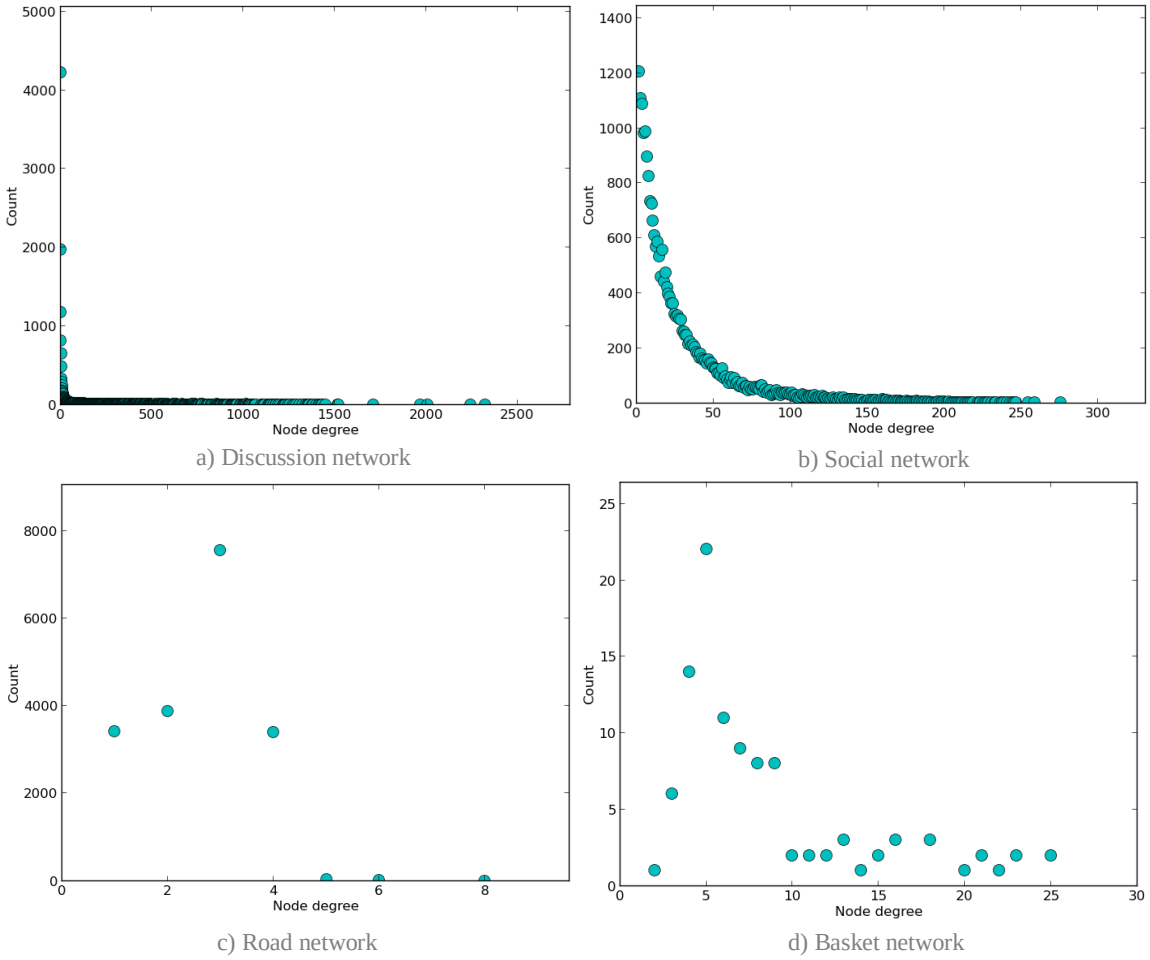


Figure 4-8. Node degree distributions

Figure 4-8 shows that both the discussion network and the social network display a long tail, a characteristic of power laws [53]. To confirm that online networks have the scale-free property, we converted the power law relationship $y = Cx^{-\gamma}$ to an expected linear relationship by taking its logarithm: $y' = \log(C) - \gamma\, x'$ where $x' = \log(x)$ $y' = \log(y)$. From the two online networks, the discussion network indeed displayed a linear pattern, while the social network did not (Figure 4-9). In the case of discussion network, we used least squares fitting on the cloud point to get a rough estimate $\gamma = 1.06$ and $C = 1200$ (Figure 4-10).

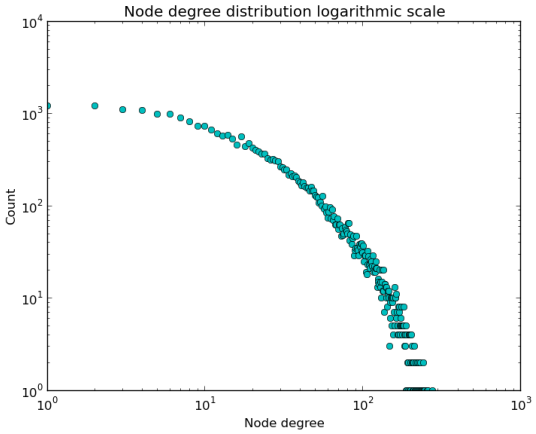


Figure 4-9. Node distribution of social network in logarithmic scale

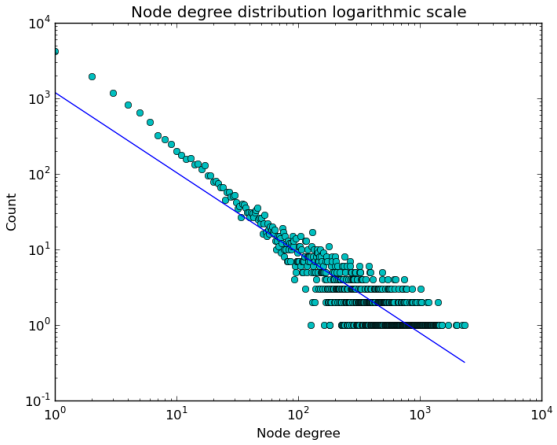


Figure 4-10. Least squares fitting

The node degree of the road and the basket network clearly do not follow a power-law distribution. We concluded that out of the four datasets, only the discussion network was scale-free (Figure 4-11).

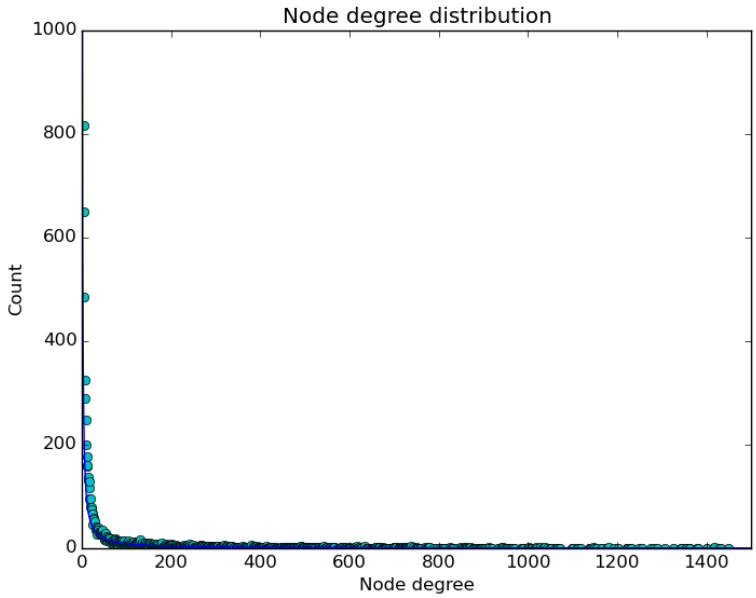


Figure 4-11. Scale-free property of the discussion network

5. Application of outlier detection techniques on datasets

The algorithms were implemented in Python[1] [54], and the NetworkX[2] [55] library was used for graph related operations. For Oddball and SCAN, we have a reliable implementation that scales well to graphs with nodes in the order of 10 thousands and edges in the 100 thousands. Autopart, on the other hand, operates with a complexity which renders it inapplicable to our large graphs.

The algorithms are summarized briefly in Table 5-1. Oddball is the most flexible in terms of input graphs. The reason is that an ego network is practically a subgraph of the complete graph, and the operation of taking subgraphs is independent of the graph type. Furthermore, the features can also be selected based on the input graph type. For instance, the total weight of edges in an ego network might prove useful for directed, weighted multigraphs.

Oddball may be considered parameter-free, however, the feature selection – a pre-requisite for its operation – is decisive in its usefulness. Selecting the appropriate features for a given problem might require exhaustive experimentation. The other parameter-free technique is Autopart. The reason it can be parameter free is because it attempts to determine the number of clusters k by starting from 1 and increasing it step-by-step. This approach has a high computational expense, which is reflected in its complexity $O(e\,k^{*2})$, where e and k^* are the number of edges, and the final cluster count, respectively.

The complexity of Oddball depends on the selected feature. While a simple choice, such as the number of nodes may take $O(n)$ (n = #nodes) steps to finish, a more computationally demanding choice, such as node eccentricity, may require $O(n^3)$ steps, in order to compute all the shortest paths in the network (using the Floyd-Warshall algorithm [56]).

	Oddball	**SCAN**	**Autopart**
Input graph	Simple | Multi Plain | Attributed Directed | Undirected Weighted | Unweighted	Simple Plain Undirected Unweighted	Simple Plain Directed | Undirected Unweighted
Parameters	Parameter-free	2 parameters	Parameter-free
Output	Node outliers	Node outliers Clustering	Edge outliers Clustering
Complexity	Feature dependent	$O(e)$	$O(e\,k^{*2})$

Table 5-1. Algorithm applicability

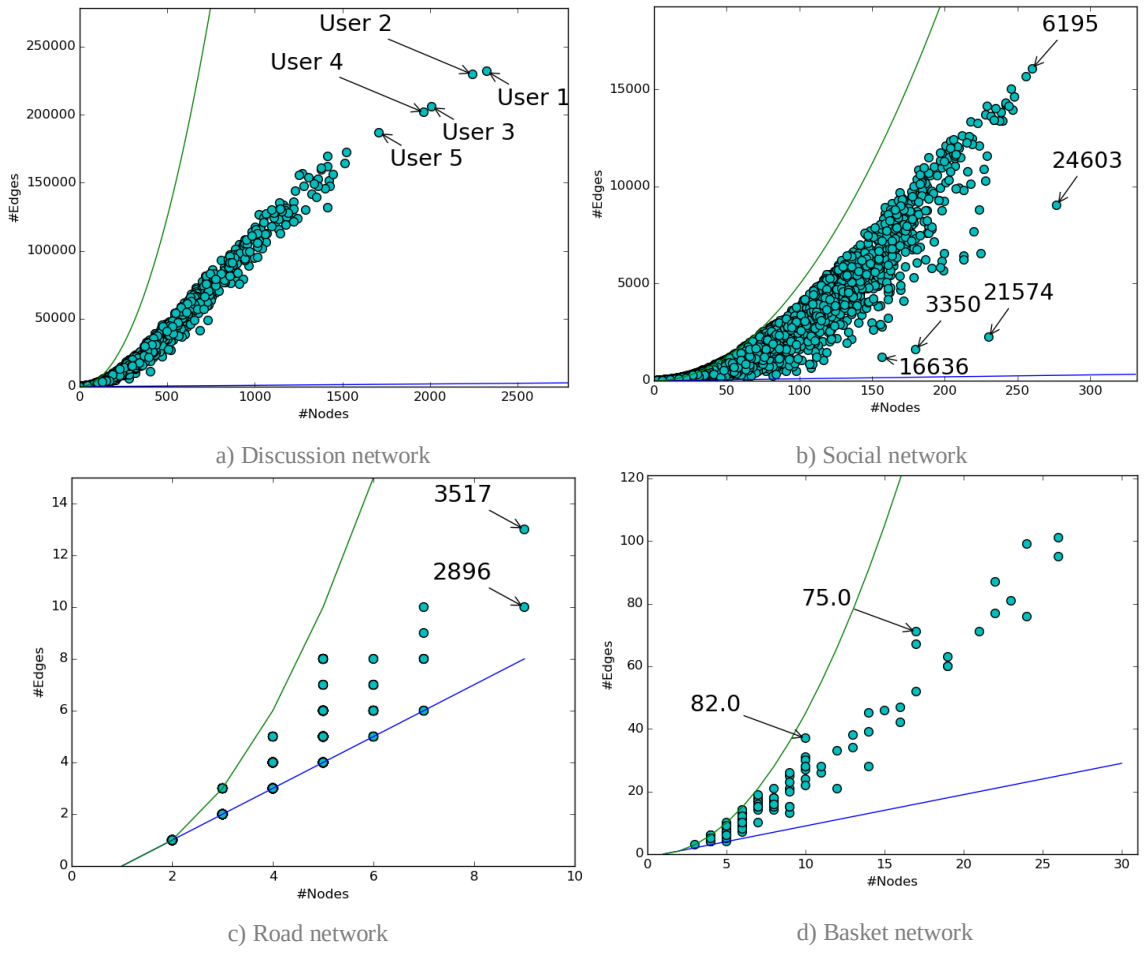
The implementation is accessible in a public repository at

https://github.com/tonnpa/opleaders

[1] Version 2.7, for compatibility with the NetworkX drawing functions
[2] Version 1.10

5.1. Oddball results

Oddball[1] identifies outliers that could be organized into two categories. The first attracted attention by its excessively high number of graph components in the ego network (Figure 5-1/a, b, c). These are distinctly separable from the rest of the data points and are not specifically targeted by Oddball. The second category contains the patterns recognizable from the plotting of number of nodes against the number of edges: near-cliques (Figure 5-1/d) and near-stars (Figure 5-1/b, c).



a) Discussion network

b) Social network

c) Road network

d) Basket network

Figure 5-1. Outliers identified by Oddball

The data points in the discussion network exhibit a near linear pattern. The labeled points are five users who are especially active in posting their opinions under news articles. As a result, they established a high number of links with a high number of users. Their intensive forum activity is reflected in the user statistics of Disqus that lists 4 of them to be the top 4 commenters[2].

The social network demonstrated well the purpose of Oddball and our choice of features. Three nodes were found to have near-star ego networks (Figure 5-2). Using ForceAtlas, it becomes obvious that the identified nodes indeed provide bridges between clusters.

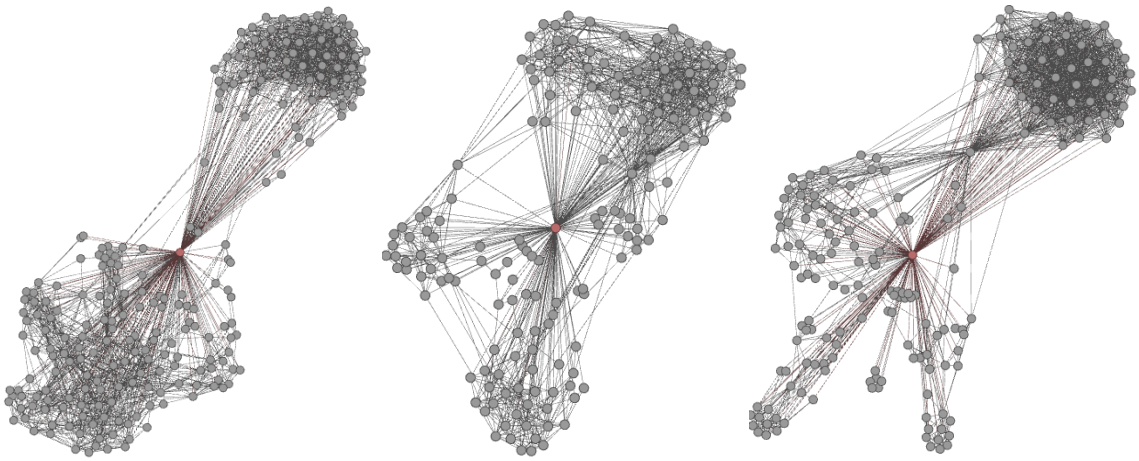
[1] Oddball has an existing implementation in the combination of Python and MATLAB scripts on the author's web page [67].

[2] According to Disqus statistics in May 2015.

Particularly node 21574 (Figure 5-2/a) embodies a transition between two otherwise weakly connected partitions.

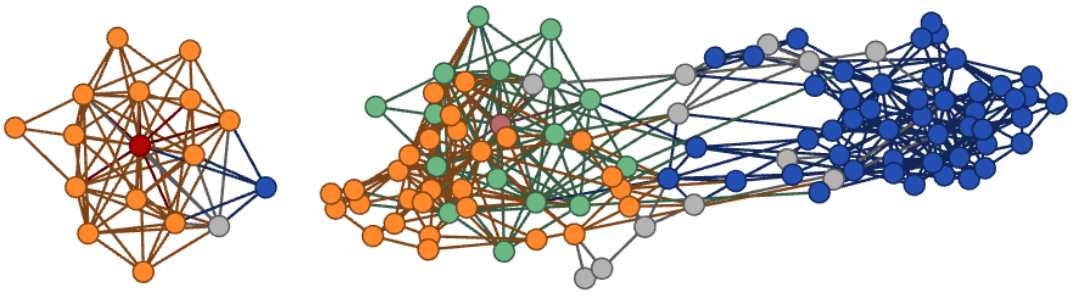
In the case of road network, two points were emphasized for their especially high number of neighbors. These had an ego network of 9 nodes, which indeed covers a unique crossing point from which 8 roads originate. There were many stars in this dataset, which could be explained as the result of minimizing connection redundancy between closely positioned locations.

The basket network contained two near cliques with a relatively high number of nodes in the ego network (Figure 5-3). Both of these turned out to be liberal books that happened to be each other's neighbor.

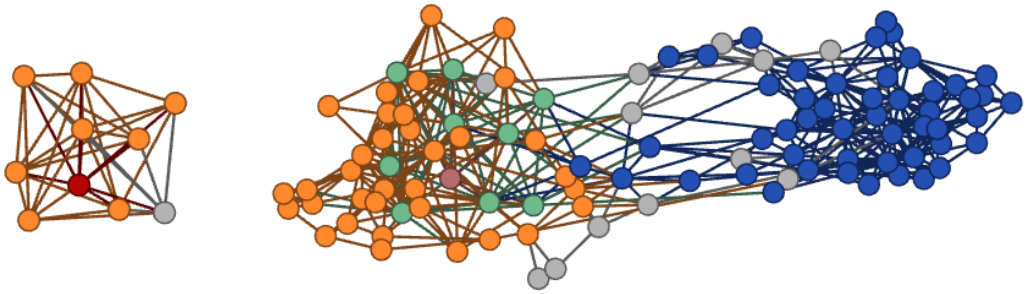


a) User 21574 b) User 16636 c) User 3350

Figure 5-2. Near-star ego networks in the social network



a) The ego network of book 75 (Worse Than Watergate) and its location in the complete graph



b) The ego network of book 82 (The Politics of truth) and its location in the complete graph

Figure 5-3. Near-clique ego networks in the basket network

5.2. SCAN results

All four datasets were inspected using the algorithm. We conducted the analysis on the discussion, social and road network, and refer to the SCAN authors' work in the case of basket network. In the upcoming subsection, the social network is inspected thoroughly: using this example we show how SCAN can be combined with Oddball for anomaly verification. Afterwards, the discussion and road network are examined briefly, in which we show that applying the algorithm can reveal further insights about the domain.

Note that what we consider as outliers, the extraordinary observations that fulfill special roles, are here referred to and detected as hubs.

5.2.1. SCAN at work: social network

In our detailed analysis of the algorithm we leverage the available ground truth, the 455 predefined friendship circles. SCAN requires two input parameters (ε, μ). Based on the author's recommendation on parametrization, we ran the algorithm multiple times, tuning ε in the range of 0.5 and 0.7, 0.02 steps in-between, and $\mu = 2, 3, 4$.

In order to compare the results of different parametrizations, we defined a performance measure based on ground truth. Since we do not have complete information about all circles in the social network, we focused on how well the given 455 circles appear in the resulting clustering. We converted the problem to a maximum weighted bipartite matching:

- the two disjoint sets of vertices are the (i) ground truth circles and the (ii) clusters found by the algorithm
- there is an edge between a circle and a cluster, if they have at least one common node, that is they are not disjoint groups of nodes in the network
- the weight of the edges is the *Jaccard similarity* [57] of the connected circle and cluster

Jaccard similarity is defined as

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}, \qquad 0 \leq J(A,B) \leq 1$$

where A, B are groups of nodes. It equals the number of common nodes divided by the number of distinct nodes in the groups.

The maximum weighted matching yields a one-to-one pairing of circles and clusters, maximizing the similarity between the pairs. The final performance score assigned to a clustering is the average similarity of its pairs: $\frac{\Sigma_{pair} J(pair_{circle}, pair_{cluster})}{\#circles}$.

The scores calculated for the different parametrizations are displayed in Figure 5-4. It can be seen that the score decreases by increasing μ, or by moving ε to the extremes of the recommended range. Increasing μ raises the minimum number of nodes there has to be in a cluster. Consequently, fewer graph parts meet the higher requirement and the number of clusters decreases (Figure 5-5). At the same time, changes occur in the classification of nodes that were previously parts of small clusters in case of $\mu = 2$: these nodes are

converted from cluster members to outliers. Thus the percentage of outliers increases (Figure 5-6).

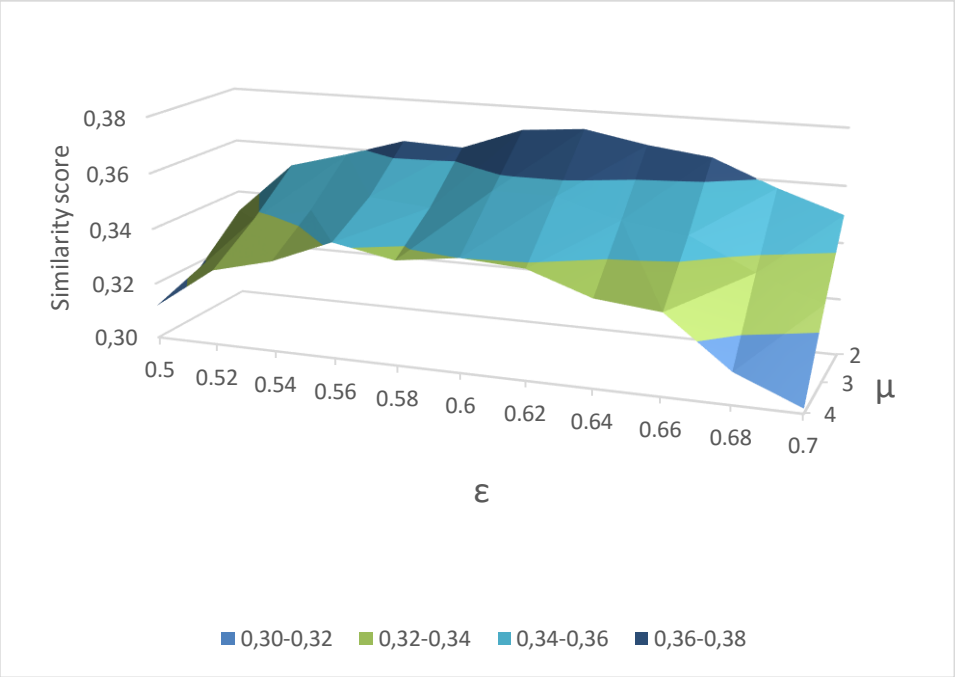


Figure 5-4. Clustering performance score of varying (ε, μ)

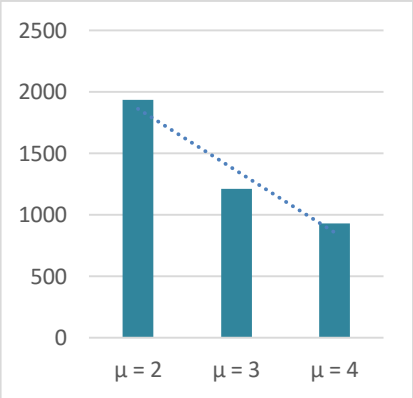


Figure 5-5. Number of clusters $\varepsilon = 0.62$

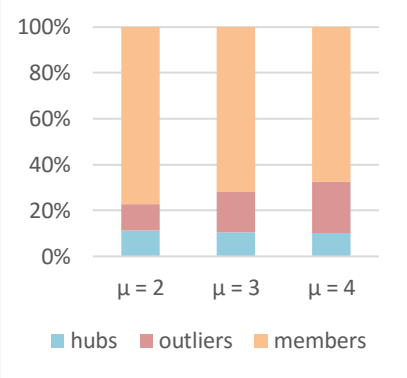


Figure 5-6. Cluster composition $\varepsilon = 0.62$

Moving ε to the extremes of the recommended range has two opposite effects. A low ε causes the merging of small clusters, and results in a low number of large clusters. On the other hand, a high ε causes the decomposition of large clusters, and results in a high number of small clusters (Figure 5-7, Figure 5-8). A balance between the two extremes could estimate the right number of clusters of the appropriate size. To maximize the similarity score, we settled the parameters $\varepsilon = 0.62, \mu = 2$. Further analysis is conducted on the results of that parameterization.

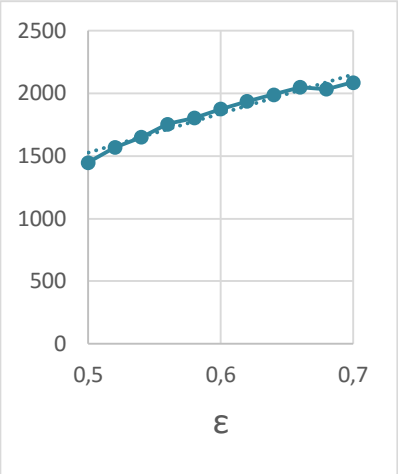


Figure 5-7. Number of clusters $\mu = 2$

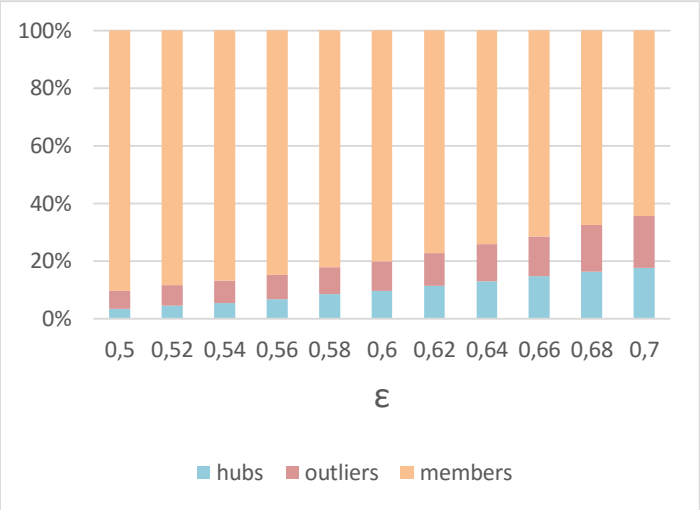


Figure 5-8. Cluster composition $\mu = 2$

One disadvantage of SCAN is the predominant number of small clusters as shown in Figure 5-9. The ground truth only had circles with at least five members, and pairing these up with mostly two or three-sized groups yielded a low average similarity. Figure 5-10 shows that a notable number of circles (~ 60 out of 455) had no, or very insignificant matches. Only 8 circles were found completely, while the rest were paired to clusters with similarities distributed evenly between 0 and 1.

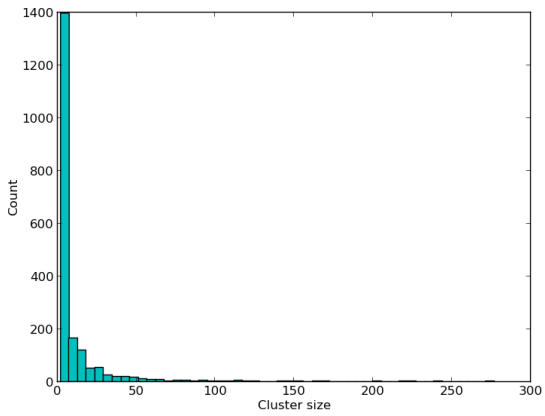


Figure 5-9. Histogram of cluster size

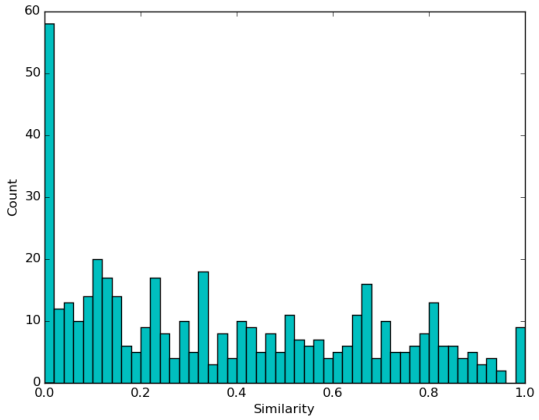
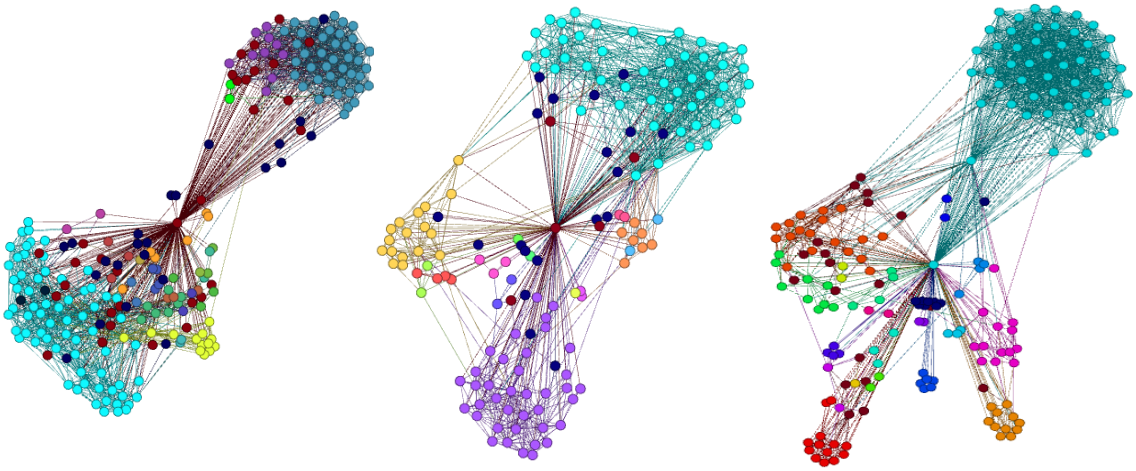


Figure 5-10. Histogram of similarity

Another disadvantage of SCAN compared to Oddball is the high number of identified anomalies. Outliers, in the context of SCAN, do not have significance, and may be discarded as noise. Hubs, on the other hand, play an important interconnecting role. In our social network of more than 26 000 nodes, over 3000 nodes were marked as hubs. Further work has to be conducted to differentiate the most impactful ones.

We inspected whether the near-stars detected by Oddball would appear here as hubs. Figure 5-11 displays the ego networks of the particular nodes. Hubs are dark red and outliers are dark blue. The remaining colors represent clusters. Similar, or matching colors between the subfigures do not represent the same clusters. It can be seen that two out of

three stars were classified as hubs. In the third case, the central node was rendered to be part of the largest cluster in its vicinity.



a) User 21574 b) User 16636 c) User 3350

Figure 5-11. Near-star ego networks colored by the clustering of SCAN

5.2.2. SCAN at work: discussion and road network

Datasets without ground truth about clusters or circles do not provide feedback on the quality of parametrization. Therefore we introduced a few intuitive conditions based on which we selected the parameters:

1. Provided the size of these datasets, more than 1 cluster has to be identified.
2. The execution of the algorithm should yield hubs and outliers.
3. The number of outliers and hubs should not nearly equal or exceed the number of cluster members.
4. The number of outliers should not exceed 10% of the network's node count.

Following these guidelines, we selected an ε value for our networks (we retained $\mu = 2$). The clustering of the discussion network had to be conducted with an unexpectedly low $\varepsilon = 0.15$ value. Figure 5-12 shows that for lower values, no hubs were found, which violates condition #2. However, for higher values, the number of outliers exceeds the number of cluster members, violating condition #3 and #4. Therefore we settled for a transitional value in the middle of the two extremes.

The high number of outliers is accompanied by another characteristic feature of the clustering: the presence of a single encompassing cluster with nearly 6000 members (Figure 5-13). The remaining clusters have only a few (≤ 11) members. This analysis confirmed our first impression about the network on Figure 4-1, which visualized the discussion network in a star shape, with a dense nucleus in the center.

A single dominating cluster of thousands of members accompanied by numerous outliers suggests an activity pattern in discussion forums. Registered users are either active commenters, thus becoming a member of the discussion community, or passive observers who are mainly reading the news articles and might leave a comment or two in a few cases.

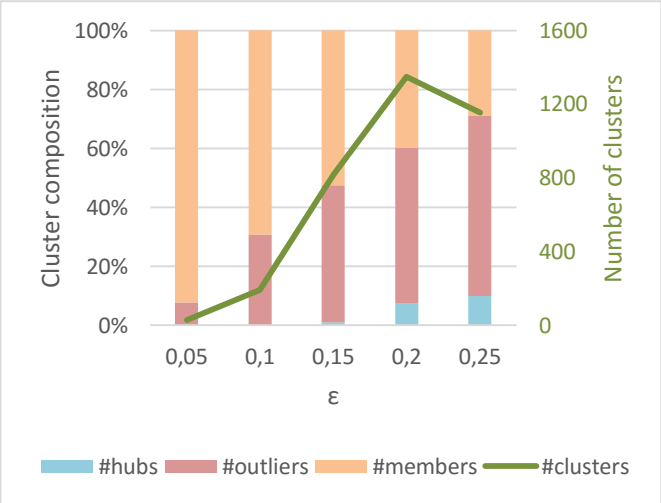


Figure 5-12. Cluster count and composition in ε tuning of discussion network

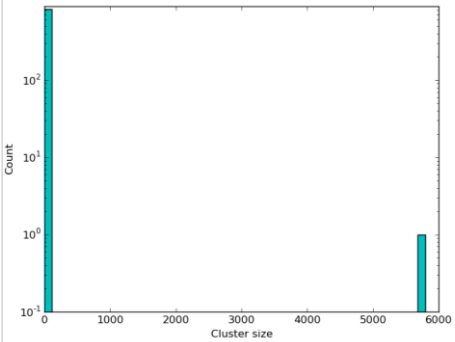


Figure 5-13. Histogram of cluster size, discussion network

We used ForceAtlas to display the distinct rift between the two types of users. All clusters were painted black, except for the single large one, which is displayed in orange. Outliers were kept dark blue. What can be seen is a large orange nucleus, surrounded by sharp, dark blue spikes (Figure 5-14).

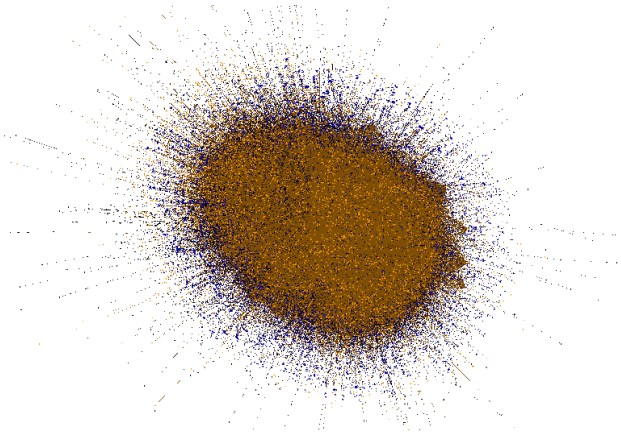


Figure 5-14. Separation of the two user types in the discussion network

We conducted the clustering of the road network with $\varepsilon = 0.5$. Figure 5-15 shows that there is no significant difference between an ε value of 0.45 or 0.5, so we settled arbitrarily on the latter. The resulting clusters, similar to the previous cases, showed a tendency of containing only a few members (Figure 5-16). Interestingly, there was a cluster of nearly 700 members, which is unexpected for a relatively homogeneous road network. Figure 5-17 shows (in a graph structure-centric, not geography-centric layout) that the cluster is composed of several circle and tree-like structures chained together. Further work could be conducted to explain the appearance of such cluster, or the reason why there were not more of that.

Road intersections marked as remote outliers by Oddball were not marked as hubs by SCAN, instead they all have been merged into a part of a cluster (Figure 5-18). Note that coloring is influenced by additional nodes not displayed in the figure.

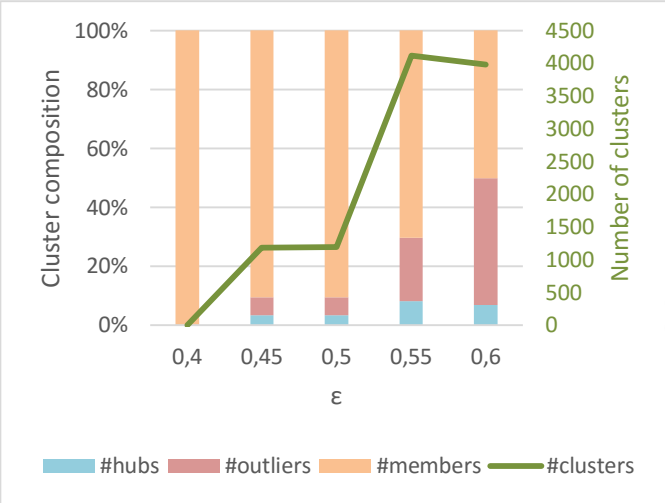


Figure 5-15. Cluster count and composition in ε tuning of road network

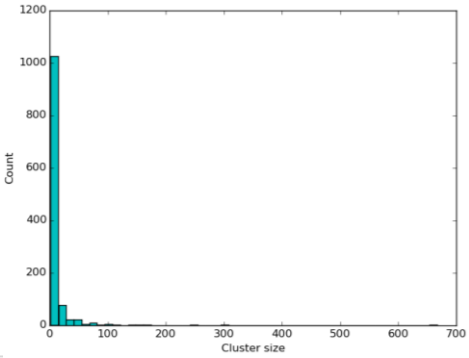


Figure 5-16. Histogram of cluster size, road network

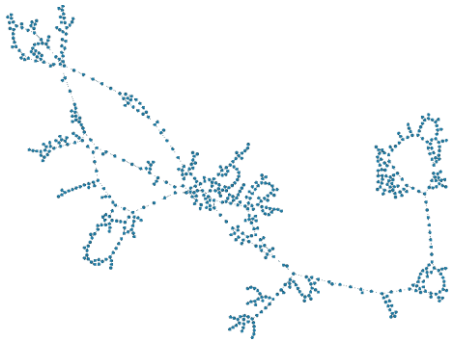


Figure 5-17. The largest cluster of the road network

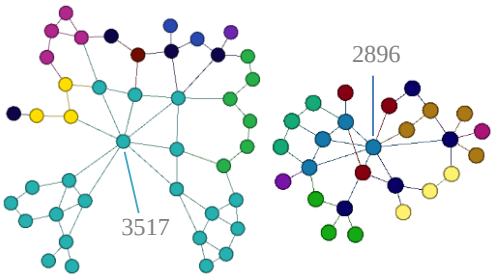


Figure 5-18. Vicinity of Oddball outliers

Specific values of cluster count and composition related to the parametrization of SCAN can be found in the Appendix.

5.2.3. Authors' graph choice: basket network

The author of [28] applied the algorithm using parameters $\varepsilon = 0.35, \mu = 2$. The three clusters representing conservative, neutral and liberal books were reported to have been found. For further analysis and illustration, refer to work [28].

5.3. Autopart results

In the case of Autopart, our implementation may yield suboptimal results. The main idea is to decompose the adjacency matrix of the inspected graph into easily compressible and transmittable parts. It is proposed that outliers could be identified by their diminishing impact on the compression rate. This technique relies on information theory concepts that are not straightforward to verify. Although we leveraged the similarity with the author's previous work on matrix decomposition [58] (also an information theory driven algorithm), we did not manage to completely clarify all claims. We proceeded assuming that the equations and the theorems were correct and adjusted our implementation for fault

acceptance. In terms of algorithm application, it means the final output could be a suboptimal construction compared to the theoretical optimum, which is compressed and transmitted using the least cost.

Our implementation works with operations that have high computing cost. The iterations in the main body of the algorithm involves frequent changes in the adjacency matrix, which in implementation means row and column manipulations. The data structure employed by the NetworkX graph library are the sparse matrices of the SciPy [59] package. These structures (compressed sparse row matrix/compressed sparse column matrix/linked list sparse matrix) do not support efficient row and column insertion and deletion, a frequently performed operation. This has a significant adverse impact on performance.

In addition to the lack of support of frequent operations, the algorithm has an inherent complexity $O(e\ k^{*2})$, which leaves a runtime proportional to the number of edges multiplied by the square of the final number of clusters. Here k^* indicates the number of clusters at which the algorithm terminates, in consequence of not finding further ways to reduce the total coding cost.

Due to the high-cost operation, and an inherently high algorithm complexity, our implementation does not scale up to the size of the large datasets. We could not verify the author's claim that Autopart *"scales linearly with the problem size, and is thus applicable to very large matrices"*.

In the following subsection, we conduct the analysis on the smaller basket network. Afterwards, we describe the measurements related to the limitations of our implementation.

5.3.1. Autopart at work: basket network

The algorithm settles at four clusters ($k^* = 4$), and reaches that state in 12 iterations. These include 3 *adjustments* of cluster count, and 9 *re-partitioning*. An adjustment is the splitting of the group with the highest entropy. A repartitioning involves moving nodes between groups.

Figure 5-19 displays the change of the adjacency matrix. The nodes line up on the x, y axes, and the black dots indicate the presence of edges between the corresponding nodes. Orange separators mark the border of clusters and divide the matrix into blocks that represent the connectivity between those clusters. A clear tendency can be discerned: the black dots accumulate in the right and lower part of the matrix, which indicates that high-degree nodes are grouped together. The algorithm halts when no further improvement is found.

We evaluate the effectiveness of the clustering based on the ground truth, the three classes in the labeling of books. In order to allow comparison of SCAN and Autopart, we adopted the measurement of SCAN's authors, the *adjusted Rand index* [60]. We also follow their reasoning for this choice [61].

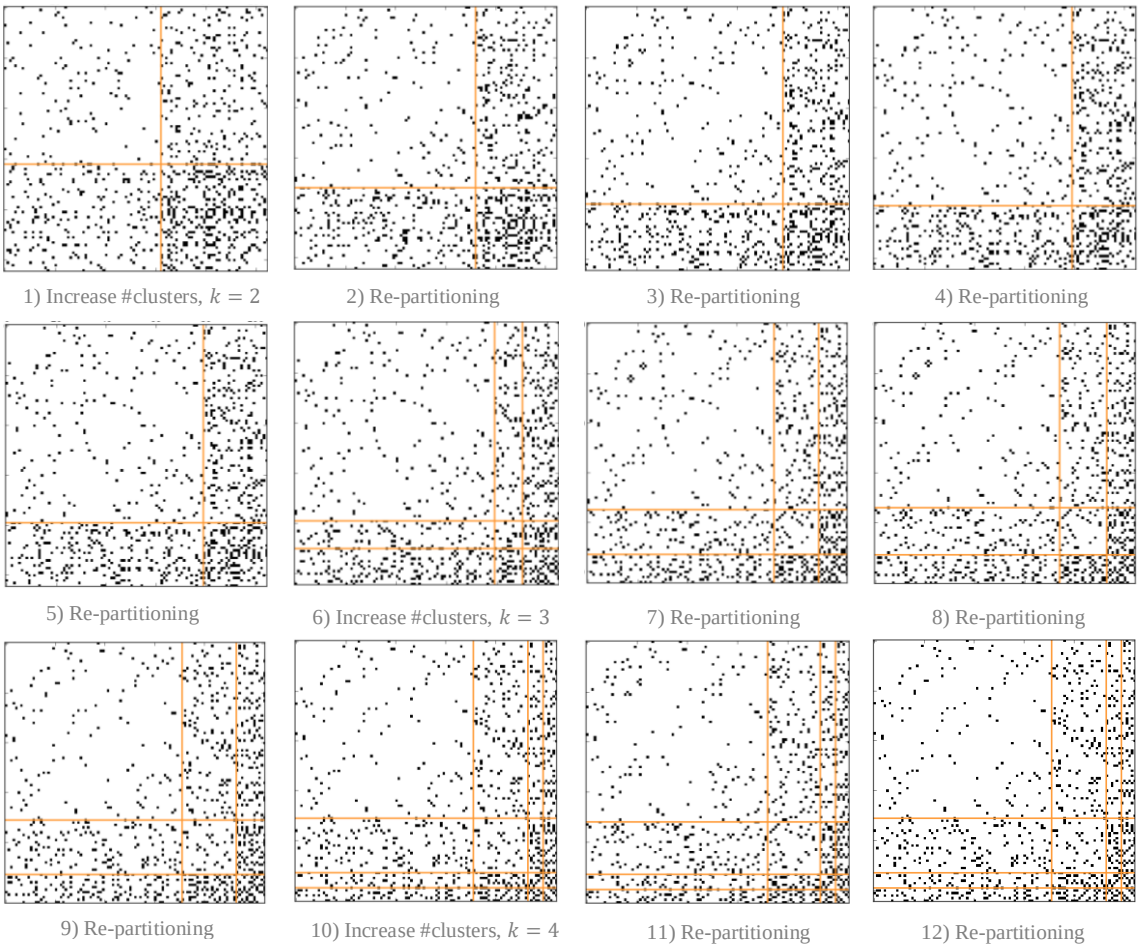


Figure 5-19. Autopart steps, changing of the adjacency matrix

The higher the index value, the greater the similarity is between the clustering and the labeling. The results indicate that Autopart yields a significantly less accurate clustering compared to SCAN, which could be partially explained by the different number of identified clusters.

	Autopart	*SCAN*
ARI	-0.03	0.71

However, an important feature and advantage of Autopart over SCAN is the overall picture of partitioning. While Autopart provides an easily visualizable, fairly intuitive overview of the clustering based on the density of connections between clusters, SCAN yields numerous clusters of various sizes that are challenging to visualize and to interpret.

Once Autopart has produced the final partitioning, it utilizes that information to mark certain edges as outliers. It reasons that those edges whose removal reduces the total encoding cost the most are the outliers. Therefore the algorithm finds the block where removal of an edge incurs the greatest reduction in cost. Since all edges within the same block contribute equally to the encoding cost, all of them are considered as edge outliers.

In the case of basket network, the 4 clusters create 16 blocks. Table 5-2 displays the reduction in total cost incurred by the removal of an edge from the given block. Since we are searching for link outliers that bridge different clusters, edges between cluster 1 and 2 are the final results (Figure 5-20).

Clusters	1	2	3	4
1	5.4	3.4	2.8	2.2
2	3.4	3.6	1.6	1.3
3	2.8	1.6	1.5	0.9
4	2.2	1.3	0.9	1.1

Table 5-2. Reduction in total cost

There are 136 edges bridging the two clusters, a result too broad to interpret. This brings us to the limitations of this technique.

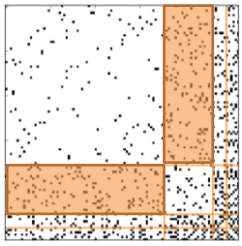


Figure 5-20. Edge outliers

5.3.2. Limitations of Autopart

The outlier detection quality of this technique is coarse in the sense that there is no differentiation between edges that reside in the same block. In consequence, when the partitioning yields a few, but large clusters, the number of edges marked as outliers is likely to be high. This raises yet another problem for the user, as no further hint is given to which specific cases should be put under scrutiny.

We created a graph construction to measure the impact of increasing graph size on the algorithm runtime. We multiplied the basket network and considered the union of multiple basket networks to be a single, unconnected graph. Following this method, we created input networks linearly growing in size. We inspected whether the result of this experiment would be a linearly increasing runtime that yields partitionings with a cluster count of linear growth.

Our measurements do not show a linear relationship between runtime and input size (Figure 5-21). They also provided a more specific insight as to why the implementation did not scale up to our large datasets: it took over 20 minutes to finish[1] for a graph of less than 500 nodes.

(The values are the average runtime of three repeated algorithm executions.)

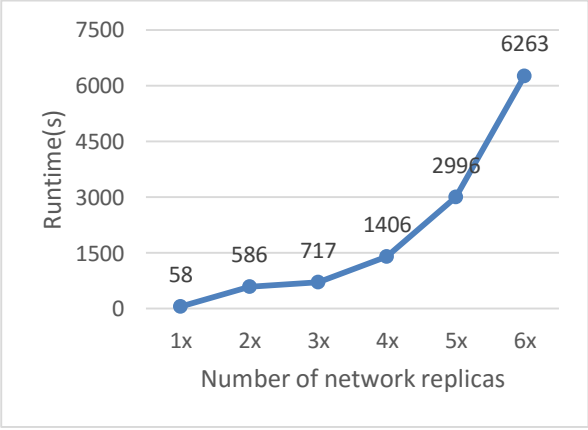


Figure 5-21. Runtime vs. input size

[1] Hardware: Intel® Core™ i5-2410M CPU @ 2.30GHz

The author's observation that *"the execution time grows linearly with the number of edges"* fails to take into consideration the special graph type on which the experiments were executed. *"Caveman graphs are highly clustered, sparse graphs that consist of isolated cliques or caves."* [62] (An example is in Figure 5-22.) The runtime measurements were conducted on special 3-cave graphs, which do not form new clusters in consequence of adding more edges. However, the evolution of complex networks, such as social networks, often brings about the emergence of new structures. An example would be that a certain social networking service becomes available in a previously unengaged geographical area. Thus the conclusion of the algorithm scaling to large graphs is highly questionable.

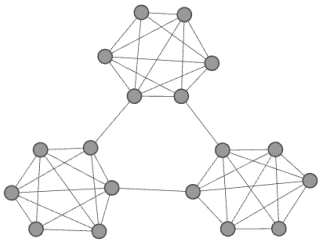


Figure 5-22. 3-cave graph

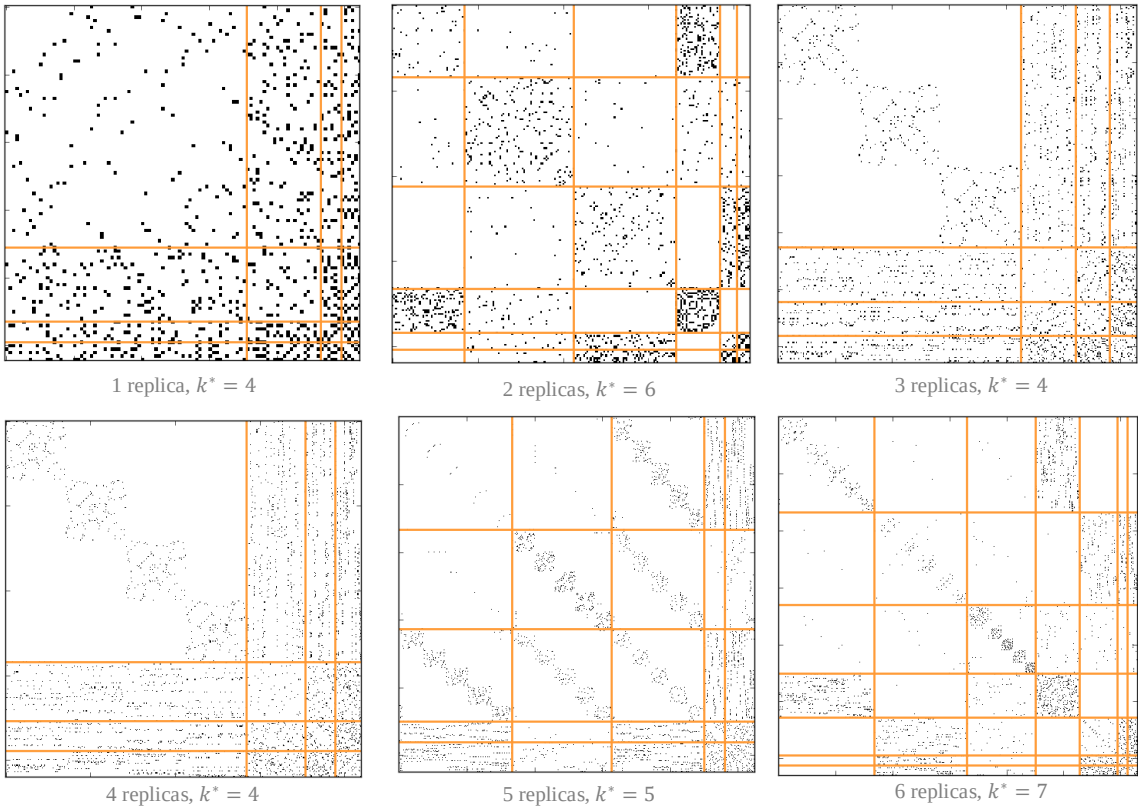


Figure 5-23. Final partitionings

Figure 5-23 displays the final partitionings of the algorithm. These hold information about the number of identified clusters. It can be observed that the case of 3 and 4 replicas did not, but the case of 2, 5, and 6 did have a dramatic increase in runtime. An explanation is provided by k^*, which, when increased, clearly disrupts any trace of linearity.

Does a pattern emerge in the grouping of nodes in reaction of a predictably changing input graph? That question remained open for us.

6. Extending the techniques to dynamic graphs

In this section, we introduce the problem of outlier detection in the context of *dynamic* or *time-evolving/temporal* graphs. Dynamic graphs are a sequence of static snapshots of the same, but evolving graphs. The problem is to find the timestamps that correspond to a change, as well as the graph objects (nodes/edges/subgraphs) that contribute most to the change [5].

The surveys about graph-based anomaly detection [5] and evolutionary network analysis [63] provide a comprehensive overview of the state-of-the-art methods for anomaly detection in a dynamic context. However, none of these discuss the usability of Oddball, SCAN and Autopart. In the following subsections, we propose a way to adapt the static outlier detection techniques to dynamic settings.

6.1. Oddball in a dynamic context

Oddball could be adapted to be a *feature-based* method [5]. The main idea is to monitor a set of selected graph properties and flag the timestamps and their corresponding snapshots where the value change of the properties exceed a predefined threshold.

"The general approach in detecting anomalous timestamps in the evolution of dynamic graphs can be summarized in the following steps:

1. *Extract a summary from each snapshot of the input graph.*
2. *Compare consecutive graphs using a distance or similarity function.*
3. *When the distance is greater than a manually or automatically defined threshold, characterize the corresponding snapshot as anomalous."* [5]

Oddball works with the building blocks of ego networks. Thus the summary extracted from snapshots would be the set of properties calculated for each ego network. In case of numeric properties – such as the number of edges or nodes – the difference of the old and new values would be an adequate distance measure. A manual threshold can be drawn based on the nature of the properties and the size of the ego networks.

In a static context, outlier detection algorithms ideally would scale linearly with the size of the input graphs (number of edges, or nodes). In a dynamic context, these algorithms should also be linear on the size of changes of the input graphs. Running Oddball repeatedly on every small change in the input graph would soon run into scalability and runtime walls. We propose a set of basic events to which network evolution can generally be decomposed to:

- Addition / Removal of an edge
- Addition of a node that connects to the existing graph with (a few) edges
- Removal of a node and all its existing connection to other nodes

It is essential for the algorithms to handle these events efficiently in order to be applicable in practice. Oddball would generally react to these in the following way.

Event	Event handling
Edge addition or removal	The ego network properties of the nodes at the endpoints of the inserted/removed edges have to be updated.
Node addition	A new set of properties is created for the new ego network. The ego networks of the new node's neighbors have to be updated.
Node removal	Properties of the node's ego network is deleted. The ego networks of the deleted node's neighbors have to be updated.

Note that there are certain graph properties that require more than the update of the vicinity of the change. For instance, removal of a highly connected node could affect the eccentricity of nodes far away from the removed node. When operating with these properties, further analysis of the problem is required.

6.2. SCAN in a dynamic context

SCAN could be adapted to be a *community or clustering-based* method. The main idea is, *"instead of monitoring the changes in the whole network,* [we] *monitor graph communities or clusters over time and report an event when there is structural or contextual change in any of them."* [5]

SCAN depends on two initial properties (ε, μ) and produces a classification of nodes into clusters, hubs and outliers. Over time, both the initial properties and the classification may be subject to change. The initial properties may have to be re-tuned, therefore a periodic update of the parameters may be conducted to keep them attuned to the network. The adjustment could be fired by different triggers, such as the elapse of predefined length of time, or the occurrence of a certain number of graph object changes.

SCAN would react to the basic events in the following way:

Event	Event handling
Edge addition	The connectivity of the nodes on the endpoint has to be re-examined. Depending on the parameters and their neighborhood, they might form a new small cluster, or merge two already existing clusters.
Edge removal	The connectivity of the nodes on the endpoint has to be re-examined. If they were part of a small cluster, that cluster may have to be removed.
Node addition	Based on the classification of the inserted node's neighbors, the node may be classified as a cluster member, hub, or an outlier.
Node removal	The connectivity of the deleted node's neighbors have to be re-examined. An existing cluster may disappear, rendering the previously node members to be either outliers as hubs. An existing cluster may also be split in two.

Unlike Oddball, which may operate with non-local graph properties, SCAN is capable of handling all the basic events by re-classifying nodes exclusive to an affected graph area. The reason is that SCAN builds the clustering on structural connectivity.

Once the clustering is adaptive to the dynamic context, SCAN can provide the hubs and outliers that appeared or vanished at a certain timestamp.

6.3. Autopart in a dynamic context

Autopart – similar to SCAN – could also be adapted to be a community or clustering-based method. In contrast to Oddball and SCAN, the event handling cannot be narrowed down to the adjustment of aggregated properties, or re-classification of a restricted area of the graph. The information theory driven algorithm, that once finished and produced a final partitioning, has to be resumed.

However, what can and should be utilized from a previous running of the algorithm is the partitioning of the adjacency matrix. Following the removal of graph objects, the nodes will be re-ordered according to the adjusted block properties. Addition of a new node could be handled by inserting the node into a random existing group and the algorithm could be resumed to see where it moves the newly inserted node.

For each timestamp and its corresponding snapshot of the graph, Autopart provides the block whose edges diminish the compression efficiency the most.

7. Summary and conclusions

We addressed the problem of anomaly detection in graphs. The problem was approached pragmatically: we selected three fundamental outlier detection techniques, and applied them on a diverse range of real-world network datasets. The networks were assembled from various domains of online discussion forums, social networking services, spatial road mappings and market basket analysis. Building on the different unique characteristics of these domains, we evaluated the networks with respect to the concept of small-world phenomenon, also known as the six degrees of separation, and the scale-free property, a structure attributed to evolution following preferential attachment. In addition, we summarized the essential graph-centric properties [5] for comparing network datasets.

For each dataset, we defined outliers in respect of its domain. We aimed to identify opinion leaders and spammers in a discussion community; users bridging multiple, but not committing to a single of friendship circles in a social network; intersections that connect an unusually high number of road sections; and finally, purchases that contain politically contrasting books in a basket network.

The three techniques employed to the detection of the predefined anomalies were feature, network structure, and information theory driven: Oddball, SCAN, and Autopart, respectively. Oddball and SCAN targeted node outliers, and Autopart located edge outliers.

Oddball detected visually conspicuous, select outliers in all networks. It identified especially active users among the forum commenters. In order to further classify these as either opinion leaders, or spammers, deeper analysis is required. A possible way for that is to embed the characteristics proposed in work [64] that differentiate spammers from regular users. Oddball also located high-degree intersections of the road network. Its straightforwardness makes it a highly effective, easily applicable technique.

SCAN found hubs positioned on the border of densely connected graph parts. Although the results are outputted quickly, their high numbers rule out case-by-case examination. We combined SCAN with Oddball to focus on the particularly interesting cases. This proved its usefulness in the social network, where users with potentially numerous weak ties were discovered.

Autopart marked compression-reducing edges in the basket network. Its frequent matrix operations and complexity require carefully chosen data structures to make it scalable. Although this technique also produces too many outliers for a case-by-case analysis, it provides a general overview of them in the feasibly visualizable partitioned adjacency matrix.

Finally, we delineated the possible extension of the algorithms to a dynamic context.

7.1. Future work and possible improvements

The parameter estimation of SCAN were mostly carried out according to the authors' suggestions and repeated measurements. Work has been conducted toward the automatic tuning of ε [65] which could be imported into the current implementation for a more refined method of parameter assignment.

Moreover, SCAN could be utilized to improve on Autopart. In a combination of the two algorithm, SCAN – which scales more efficiently for large graphs – could provide the initial cluster count k for Autopart. The reason k is set to 1 in the beginning is that Autopart aimed to remain parameter free. However, the computation cost in turn is really high ($O(e\ k^{*2})$) which in practice could be substantially reduced by providing an estimation of lower bound for k^*.

Our discussion network contains many attributes that have yet to be fully exploited. The dates on the individual comments allow for analysis of dynamic graphs. The registration dates of forum users enables the observation of user life cycles. The ratings, like and dislike scores of comments open up a new dimension on response quality that could reveal insights without reading all the comment text themselves.

Could the emergence of new community structures be spotted in their development? What are the typical activity patterns exhibited by newcomers? Are opinion leaders distinguishable by rating reputation alone?

These are questions that guide in directions worth exploring.

Index of figures

Figure 3-1. Ego network 13
Figure 3-2. Clique in graph *A* 14
Figure 3-3. Star in graph *B* 14
Figure 3-4. Revealing cliques in graph *A* 14
Figure 3-5. Revealing stars in graph *B* 14
Figure 3-6. A network with two clusters, a hub and an outlier 15
Figure 3-7. $\varepsilon = 0.7, \mu = 2$ 15
Figure 3-8. $\varepsilon = 0.8, \mu = 2$ 15
Figure 3-9. $\varepsilon = 0.9, \mu = 2$ 15
Figure 3-10. Oddball at work 15
Figure 3-11. A partitioning of an adjacency matrix 16
Figure 3-12. Steps of the Autopart algorithm 17
Figure 4-1. Introduction of the large datasets 19
Figure 4-2. Small network of books 19
Figure 4-3. Network distances 21
Figure 4-4. Average clustering coefficient of networks 21
Figure 4-5. The strength of weak ties 22
Figure 4-6. Degree measures 23
Figure 4-7. Book labeling 23
Figure 4-8. Node degree distributions 25
Figure 4-9. Node distribution of social network in logarithmic scale 26
Figure 4-10. Least squares fitting 26
Figure 4-11. Scale-free property of the discussion network 26
Figure 5-1. Outliers identified by Oddball 28
Figure 5-2. Near-star ego networks in the social network 29
Figure 5-3. Near-clique ego networks in the basket network 29
Figure 5-4. Clustering performance score of varying (ε, μ) 31
Figure 5-5. Number of clusters $\varepsilon = 0.62$ 31
Figure 5-6. Cluster composition $\varepsilon = 0.62$ 31
Figure 5-7. Number of clusters $\mu = 2$ 32

Figure 5-8. Cluster composition $\mu = 2$ 32
Figure 5-9. Histogram of cluster size 32
Figure 5-10. Histogram of similarity.....32
Figure 5-11. Near-star ego networks colored by the clustering of SCAN 33
Figure 5-12. Cluster count and composition in ε tuning of discussion network.....34
Figure 5-13. Histogram of cluster size, discussion network.....34
Figure 5-14. Separation of the two user types in the discussion network 34
Figure 5-15. Cluster count and composition in ε tuning of road network.....35
Figure 5-16. Histogram of cluster size, road network.....35
Figure 5-17. The largest cluster of the road network 35
Figure 5-18. Vicinity of Oddball outliers 35
Figure 5-19. Autopart steps, changing of the adjacency matrix 37
Figure 5-20. Edge outliers 38
Figure 5-21. Runtime vs. input size 38
Figure 5-22. 3-cave graph.....39
Figure 5-23. Final partitionings 39

Index of tables

Table 1-1. Applications of outlier detection 8
Table 2-1. Anomaly detection techniques in plain, static graphs 11
Table 4-1. Graph features of datasets 24
Table 5-1. Algorithm applicability 27
Table 5-2. Reduction in total cost 38

Bibliography

[1] D. M. Hawkins, Identification of outliers, Springer, 1980.

[2] E. M. Knorr and R. T. Ng, "Finding intensional knowledge of distance-based outliers," in *VLDB*, 1999, pp. 211-222.

[3] M. Markou and S. Singh, "Novelty detection: A review - Part 1: Statistical approaches," *Signal processing*, pp. 2481-2497, 2003.

[4] M. Markou and S. Singh, "Novelty detection: A review - Part 2: Neural network based approaches," *Signal processing*, pp. 2499-2521, 2003.

[5] L. Akoglu, H. Tong and D. Koutra, "Graph based anomaly detection and description: a survey," *Data Mining and Knowledge Discovery*, pp. 626-688, 2014.

[6] Facebook Inc., "Facebook Data Science," [Online]. Available: https://www.facebook.com/data?_rdr=p.

[7] Twitter Inc., "The Twitter Data Blog," [Online]. Available: https://blog.twitter.com/data.

[8] LinkedIn Inc., "Data | LinkedIn Engineering," [Online]. Available: https://engineering.linkedin.com/data.

[9] Couchsurfing International Inc., "Stay with Locals and Make Travel Friends | Couchsurfing," [Online]. Available: https://www.couchsurfing.com/.

[10] Uber Inc., "#uberdaat | Uber Global," [Online]. Available: http://newsroom.uber.com/tag/uberdata/.

[11] Tinder Inc., "Tinder," [Online]. Available: https://www.gotinder.com/.

[12] Q. Ding, N. Katenka, P. Barford, E. Kolaczyk and M. Crovella, "Intrusion as (anti) social communication: characterization and detection," in *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM, 2012, pp. 886-894.

[13] C. Cortes, D. Pregibon and C. Volinsky, Communities of interest, Springer, 2001.

[14] D. H. Chau, S. Pandit and C. Faloutsos, "Detecting fraudulent personalities in networks of online auctioneers," in *Knowledge Discovery in Databases: PKDD 2006*, Springer, 2006, pp. 103-114.

[15] Z. Li, H. Xiong, Y. Liu and A. Zhou, "Detecting blackhole and volcano patterns in directed networks," in *IEEE 10th International Conference on Data Mining (ICDM)*, Sydney, 2010.

[16] C. Castillo, D. Donato, A. Gionis, V. Murdock and F. Silvestri, "Know your neighbors: Web spam detection using the web topology," in *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, 2007.

[17] H. Gao, Y. Chen, K. Lee, D. Palsetia and A. N. Choudhary, "Towards Online Spam Filtering in Social Networks," in *Proceedings of the 19th Annual Network & Distributed System Security Symposium*, 2012.

[18] A. Silberschatz, P. B. Galvin, G. Gagne and A. Silberschatz, "Deadlocks," in *Operating system concepts*, Addison-Wesley Reading, 1998, pp. 283-313.

[19] A. Rapoport, "Spread of information through a population with socio-structural bias: I. Assumption of transitivity," *The bulletin of mathematical biophysics*, vol. 15, no. 4, pp. 523-533, 1953.

[20] Amazon.com Inc, "Amazon Mechanical Turk," [Online]. Available: https://www.mturk.com/mturk/welcome.

[21] H. J. Escalante, "A comparison of outlier detection algorithms for machine learning," in *Proceedings of the International Conference on Communications in Computing*, 2005.

[22] L. C. Freeman, "A set of measures of centrality based on betweenness," *Sociometry*, pp. 35-41, 1977.

[23] D. J. Watts and S. H. Strogatz, "Collective Dynamics of 'Small-world' Networks," *Nature*, vol. 393, pp. 440-442, 1998.

[24] M. E. Newman, "Modularity and community structure in networks," *Proceedings of the National Academy of Sciences*, vol. 103, pp. 8577-8582, 2006.

[25] L. Page, S. Brin, R. Motwani and T. Winograd, "The PageRank citation ranking: bringing order to the Web," Stanford InfoLab, 1999.

[26] C. C. Aggarwal, "Outlier Detection in Graphs and Networks," in *Outlier Analysis*, Springer, 2013, pp. 343-345.

[27] L. Akoglu, M. McGlohon and C. Faloutsos, "OddBall: Spotting Anomalies in Weighted Graphs," in *Advances in Knowledge Discovery and Data Mining*, Springer, 2010, pp. 410-421.

[28] X. Xu, N. Yuruk, Z. Feng and T. A. Schweiger, "SCAN: A Structural Clustering Algorithm for Networks," in *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2007.

[29] D. Chakrabarti, "Autopart: Parameter-free graph partitioning and outlier detection," in *Knowledge Discovery in Databases: PKDD 2004*, Springer, 2004, pp. 112--124.

[30] E. M. Knorr, R. T. Ng and V. Tucakov, "Distance-based Outliers: Algorithms and Applications," *The VLDB Journal*, vol. 8, no. 3-4, pp. 237-253, 2000.

[31] M. M. Breunig, H.-P. Kriegel, R. T. Ng and J. Sander, "LOF: Identifying Density-based Local Outliers," *SIGMOD Rec.*, vol. 29, no. 2, pp. 93-104, 2000.

[32] L. Akoglu, M. McGlohon and C. Faloutsos, "Anomaly Detection in Large Graphs," 2009.

[33] C. Böhm, K. Haegler, N. S. Müller and C. Plant, "CoCo: coding cost for parameter-free outlier detection," in *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2009.

[34] C. E. Shannon, "A note on the concept of entropy," *Bell System Tech. J,* vol. 27, pp. 379-423, 1948.

[35] Magyar Jeti Zrt., "444," [Online]. Available: http://444.hu/.

[36] Kaggle, "Learning Social Circles in Networks | Kaggle," May 2014. [Online]. Available: https://www.kaggle.com/c/learning-social-circles/data. [Accessed September 2015].

[37] T. Brinkhoff, "Real Datasets for Spatial Databases: Road Networks and Category Points," 9 September 2005. [Online]. Available: https://www.cs.utah.edu/~lifeifei/SpatialDataset.htm. [Accessed September 2015].

[38] V. Krebs, "Social & Organizational Network Analysis software & services for organizations, communities, and their consultants," [Online]. Available: http://www.orgnet.com/. [Accessed September 2015].

[39] S. Martin, W. M. Brown, R. Klavans and K. W. Boyack, "OpenOrd: An open-source toolbox for large graph layout," in *IS&T/SPIE Electronic Imaging*, 2011.

[40] M. Jacomy, S. Heymann, T. Venturini and M. Bastian, "Forceatlas2, a continuous graph layout algorithm for handy network visualization," *Medialab center of research*, vol. 560, 2011.

[41] Disqus, Inc., "Disqus," [Online]. Available: https://disqus.com.

[42] X. Song, Y. Chi, K. Hino and B. Tseng, "Identifying Opinion Leaders in the Blogosphere," in *Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management*, ACM, 2007, pp. 971-974.

[43] C. Justin, D.-N.-M. Cristian and L. Jure, "Antisocial Behavior in Online Discussion Communities," *CoRR*, 2015.

[44] Kaggle Inc., "Kaggle: The Home of Data Science," [Online]. Available: https://www.kaggle.com/.

[45] S. Milgram, "The small world problem," *Psychology today*, vol. 2, pp. 60-67, 1967.

[46] A. Mislove, M. Marcon, K. P. Gummadi, P. Druschel and B. Bhattacharjee, "Measurement and analysis of online social networks," in *Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, 2007.

[47] M. S. Granovetter, "The strength of weak ties," *American journal of sociology*, pp. 1360-1380, 1973.

[48] D. Easley and J. Kleinberg, "The Strength of Weak Ties," in *Networks, crowds, and markets: Reasoning about a highly connected world*, Cambridge University Press, 2010, pp. 50-51.

[49] R. Agrawal, T. Imielinski and A. Swami, "Mining association rules between sets of items in large databases," in *ACM SIGMOD Record*, 1993.

[50] M. J. Berry and G. Linoff, "Market basket analysis and association rules," in *Data mining techniques: for marketing, sales, and customer support*, John Wiley & Sons, Inc., 1997, pp. 287-289.

[51] M. Newman, "Network data," 19 April 2013. [Online]. Available: http://www-personal.umich.edu/~mejn/netdata/. [Accessed September 2015].

[52] A. L. Barabási and A. Réka, "Emergence of scaling in random networks," *Science,* vol. 286, pp. 509-512, 1999.

[53] A. Clauset, C. R. Shalizi and M. E. Newman, "Power-law distributions in empirical data," *SIAM review,* vol. 51, no. 4, pp. 661-703, 2009.

[54] Python Software Foundation, "Python," [Online]. Available: https://www.python.org/.

[55] NetworkX developer team, "NetworkX," [Online]. Available: https://networkx.github.io/.

[56] R. W. Floyd, "Algorithm 97: shortest path," *Communications of the ACM,* vol. 5, no. 6, p. 345, 1962.

[57] P. Jaccard, "The distribution of the flora in the alpine zone," *New phytologist,* vol. 11, pp. 37-50, 1912.

[58] D. Chakrabarti, S. Papadimitriou, D. S. Modha and C. Faloutsos, "Fully automatic cross-associations," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, 2004.

[59] SciPy developers, "Sparse matrices (scipy.sparse)," [Online]. Available: http://docs.scipy.org/doc/scipy/reference/sparse.html.

[60] L. Hubert and P. Arabie, "Comparing partitions," *Journal of classification,* vol. 2, no. 1, pp. 193-218, 1985.

[61] G. W. Milligan and M. C. Cooper, "A study of the comparability of external criteria for hierarchical cluster analysis," *Multivariate Behavioral Research,* vol. 21, no. 4, pp. 441-458, 1986.

[62] D. J. Watts, "Networks, Dynamics, and the Small-world Phenomenon 1," *American Journal of Sociology*, vol. 105, no. 2, pp. 493-527, 1999.

[63] C. Aggarwal and K. Subbian, "Evolutionary network analysis: A survey," *ACM Computing Surveys (CSUR),* vol. 47, no. 1, p. 10, 2014.

[64] J. Cheng, C. Danescu-Niculescu-Mizil and J. Leskovec, "Antisocial Behavior in Online Discussion Communities," in *AAAI International Conference on Weblogs and Social Media (ICWSM),* 2015.

[65] H. Sun, J. Huang, J. Han, H. Deng, P. Zhao and B. Feng, "gSkeletonClu: Density-based Network Clustering via Structure-connected Tree Division or

Agglomeration," in *2010 IEEE 10th International Conference on Data Mining (ICDM)*, 2010.

[66] L. Akoglu, "Leman Akoglu, Stony Brook," November 2015. [Online]. Available: http://www3.cs.stonybrook.edu/~leman/pubs.html. [Accessed September 2015].

Appendix

Tables pertaining to the parameter tuning of SCAN

μ\ε	0.5	0.52	0.54	0.56	0.58	0.6	0.62	0.64	0.66	0.68	0.7
2	0.332	0.352	0.358	0.364	0.364	0.372	0.374	0.369	0.367	0.357	0.350
3	0.318	0.338	0.344	0.350	0.348	0.356	0.356	0.350	0.346	0.336	0.327
4	0.312	0.327	0.332	0.341	0.337	0.340	0.339	0.331	0.329	0.311	0.302

Social network: clustering performance score of varying (ε, μ)

feature \ μ	2	3	4
#clusters	1937	1212	930
#hubs	3004	2808	2705
#outliers	2989	4635	5892
#members	20464	19014	17860

Social network: Number of hubs, outliers and cluster members for varying parametrization ($\varepsilon = 0.62$)

feature \ ε	0.5	0.52	0.54	0.56	0.58	0.6	0.62	0.64	0.66	0.68	0.7
#clusters	1449	1567	1651	1755	1803	1875	1937	1990	2047	2032	2087
#hubs	917	1221	1451	1805	2237	2573	3004	3449	3900	4338	4696
#outliers	1656	1851	2057	2191	2473	2676	2989	3376	3657	4253	4709
#members	23884	23385	22949	22461	21747	21208	20464	19632	18900	17866	17052

Social network: number of hubs, outliers and cluster members for varying parametrization ($\mu = 2$)

feature \ ε	0.05	0.1	0.15	0.2	0.25
#clusters	30	191	818	1348	1154
#hubs	0	0	166	1135	1486
#outliers	1158	4654	7007	7988	9269
#members	13951	10455	7936	5986	4354

Discussion network: number of hubs, outliers and cluster members for varying parametrization ($\mu = 2$)

	0.4	0.45	0.5	0.55	0.6
#clusters	1	1180	1194	4116	3978
#hubs	0	612	618	1498	1248
#outliers	1	1122	1123	3920	7879
#members	18262	16529	16522	12845	9136

Road network: number of hubs, outliers and cluster members for varying parametrization ($\mu = 2$)