Friedrich-Alexander-Universität Erlangen-Nürnberg
Technische Fakultät, Department Informatik

MANUEL HUBERT

MASTER THESIS

CRAWLING AND ANALYSING CODE REVIEW NETWORKS ON INDUSTRY AND OPEN SOURCE DATA

Submitted on 2 March 2020

Supervisor:
Michael Dorner, M. Sc
Prof. Dr. Dirk Riehle, M.B.A.

Professur für Open-Source-Software
Department Informatik, Technische Fakultät
Friedrich-Alexander-Universität Erlangen-Nürnberg

Versicherung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Erlangen, 2 March 2020

License

Erlangen, 2 March 2020

Abstract

Social network analysis is a wide studied area and applyied to various fields. One field is the code review process in software development. By reviewing parts of code and sharing knowledge the developers interact in a network structure to each other. However, the field of code review networks is not studied as extensive as other fields. Therefore, the following research shows an approach for analysing structural information of code review networks in their evolution over a specific time. Additionally, the thesis focusses on the impact of structural network information on the code review duration. The result of this thesis is a network model for comparing code review data of heterogenous data sources with the findings of a decrease on the code review duration by having more complexity in the network structure over the time.

Contents

1 Introduction

1.1 Original Thesis Goals

It was the original thesis goal to gain an overview over the two industry and one open source code review data sets by doing an exploratory data analysis. In order to perform this analysis, the heterogenous data sets should be prepared and harmonized. Furthermore, a model is needed to analyse each data set independent of the data source and to make the code review data compareable. Potential findings of the network structure should be compared with the semantic information of the code review process in the data.

Therefore the original thesis goals were:

- Exploratory analysis of code review networks
- Creation of a model for analysing code review networks from heterogenous data sets

1.2 Extensions to Thesis Goals

With the creation of the model for analysing the code review networks the original thesis goals were extended, due to the fact that the open source data set was incomplete and additional semantic information of the code review process were missing. Therefore, a crawler should be implemented to collect all needed information for further analysis in the thesis.

The extended goal can be described as:

- Development of a crawler to collect open source code review data

2 Research Chapter

2.1 Introduction

"[P]oorly-reviewed code has a negative impact on software quality in large systems using modern reviewing tools" (McIntosh, Kamei, Adams & Hassan, 2016, p. 2147). Therefore, to ensure quality in software the code review process is an important element in software development. The review process in large systems can be displayed as a network of developers interacting with each other. Due to the fact that modern reviewing tools and version control systems track code review information and the interactions of the involved persons, there is a mass of data that can be used to gain knowledge in order to improve the code review process and software quality in long term.

Social network analysis is a wide studied field and gained an increasing importance nowadays. It relies on the mathematical graph theory, showing the social structure of involved persons and their behaviour to each other (Otte & Rousseau, 2002). With the availability of data and the importance of code review in terms of software quality, there is a need for research in the field of code review networks.

2.2 Related Work

Although social network analysis gains more and more importance in the various field of applications there is a gap in the case of code review network analysis. The literature concentrates in exracting and preparing code review data and identifiying key members and their impact. Furthermore the related work focusses only on open source projects and not involving industry data for more detailed insights.

Hamasaki et al. (2013) examined the different roles in the code review process of open source software. The publication provides an extraction methodology with an example on the android open source project to prepare code review data sets

for further analysis with already identified roles in the project.

Another approach for an extraction methodology of code review data from five different open source projects was done by X. Yang, Kula, Yoshida and Iida (2016). The goal was to achieve an easier structure of the code review data to retrieve people, the process and information about the product itself.

Kerzazi and El Asri (2016) created different types of networks in order to identify common patterns and experts that should be suggested as reviewer. The types of the networks are divided by the different interactions of co-editing files, commenting and performing the review process.

With also having the goal of suggesting the best reviewer for a specific code commit Ouni, Kula and Inoue (2016) developed a search based genetic algorithm that goes through past reviews and analyses the collaboration. Part of the algorithm was the collaboration network of the review process that represents the interactions of the developers an reviewers.

Finally, Bosu and Carver (2014) identified the impact of reputation of an open source developer for a review request. By examining code review networks of different open source projects, the research showed that developers with a higher reputation are getting faster feedback on review requests as well as the code review process was finished in a shorter time. Additionally, the acceptance rate of code changes in the core project was higher due to the reputation of the developer.

2.3 Research Question

This thesis performs an exploratory analysis of heterogenous code review datasources. It is unknown what information relies in code review networks and if there are any common patterns in the evolution of the data over a given period of time. Furthermore, in terms of the code review process an important factor is the performance of the review itself. Independent of the fact if it is positive or negative to have a specific review duration, the performance can be measured and shows an semantic factor of the process. Overall, the research in this thesis can be described with the following two questions:

- RQ1: What structural network information contains a code review network?
- RQ2: Do structural network information have an impact on the code review duration?

2.4 Research Approach

2.4.1 Crawling Github Open Source Code Review Data

In order to crawl the whole code review data from an open source project, the term of code review and the underlying process has to be inspected and declared for this thesis. A code review is defined by an IEEE standard as a "meeting at which software code is presented to project personnel, managers, users, customers, or other interested parties for comment or approval" (IEE, 2017, p. 74). Therefore, the main parts of the process is a bring in of software code and all different kinds of interactions and reactions to it.

The software development plattform and open source repository Github describes all the changes and commits of software code in a pull request, while interactions on it are interpreted as comments that can be optionally bundled in a review entity (GitHub, 2020a). A pull request entity contains an unique identifer for the creator that represents the person who wants to commit his software code, as well as a list of all comments which are interactions on the commit. All of the available comments contain an unique identifer for the person that reacts on the commit, too. The review entity of Github extends the interactions between persons with also containing comments related to a pull request. (GitHub, 2020b, 2020c)

In addition to the data of the code review process itself, meta data like concrete timestamps and count of changed files are useful for a structured analysis. Hence, the following data needs to be obtained:

- Pull requests
- Comments
- Code review entity with comments
- Edited files

The succeeding part describes the implementation, shown in figure 2.1, of all relevant data provided by accessing the Github API. The Implementation was done with the framework agithub (Paugh, 2019).

The root element for each interaction between each developer is the pull request. As for this the first sequence of the crawler calls the Github API to get all pull requests available at access time of the call. Secondly, the crawler saves the responses of the API in a JSON-file. Afterwards, the program iterates through each additional part shown above depending on the pull request number. The number describes an unique identifer for each sub-part. The responses of containing the comments, files or reviews are also saved in a JSON-file structured by the

number of pull request. This way, the created files can be parsed depending on the pull request as root element. With this approach, every information needed to create a code review network is saved in a JSON-file structured by the unique pull request number as initial start of the code review. Consequently, the goal of developing a crawler to collect open source code review data is achieved.

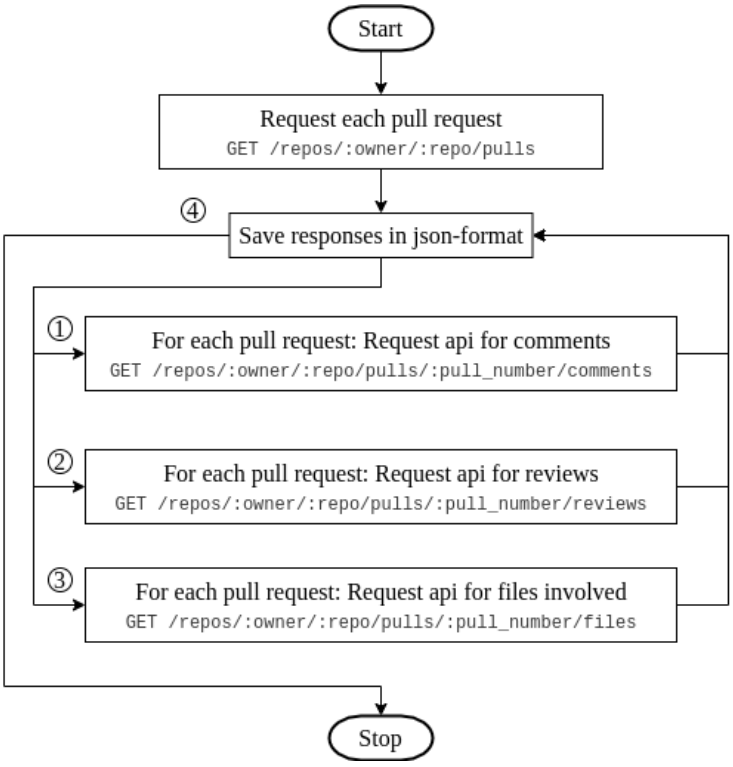

Figure 2.1: Program sequence of crawling Github

2.4.2 Creating Code Review Networks from Heterogenous Data Sets

The following part describes the process from having heterogenous data sets of different sources to create a code review network idependent of the preceding data structure. The first subpart handles the translation of each data structure in a common scheme, whereas the second subpart describes the creation of an network from the common code review entity.

As for now, there is the raw data of two companies and one crawled open source project in different file formats, containing different kind of code review information and meta information about the review process. With the aim to create a code review network from this data sets, each one needs to be parsed and

summarized in an common entity described in table 2.1. The entity mainly ensures that for each raw data set common information about the author, reviewer, timestamps and involved files are in the same scheme for building up a network of this entities. The main task of the parsers is to provide the functionality for looping through each raw data entry and mapping the containing information to the common structure.

Table 2.1: Code review entity

Name	Type	Description
id	string	An identifier for each review process. Multiple interactions can happen on a single review process. The id is needed to combine each interaction into a single review.
author	string	Person that commits a software code.
reviewer	string	Person that interacts with the committed software code and therefore takes part in the code review process.
start	datetime	Timestamp of the beginning of a code review process.
end	datetime	Timestamp of the end of a code review process. It is not filled if the data entry has not the information, if the code review process is completed.
happened_on	datetime	Timestamp of the interaction of given through the raw data entry. Equally to end field if the raw data has the information, that the code review process has ended with this interaction.
year	int	Year of the happened_on field for easier analysis.
files	int	Number of files involved during the code review process.
finished	bool	Flag if the raw data entry has the information that the code review process has finished and the end value is filled.

Furthermore, each parsing step includes a validation of the raw data. The validation checks contraints for each entry to decide if the created entity should be added to the network afterwards. The common contraints of any raw data entry is that the author can not be the reviewer at the same time, because the main

goal is to create a network of interactions between persons and not the interaction with itself. Apart from this, there are constraints specific to one data set. The data set of the travel search company included a file with unique identifier for bots that for example autonomously close pull requests. Either the author or the reviewer could be such a bot. Due to the fact that the resulting networks represent interactions between natural persons, the entry is signed as invalid. Lastly, if anything fails during the parsing process, e.g. the date format can not be interpreted, the entry is also signed invalid.

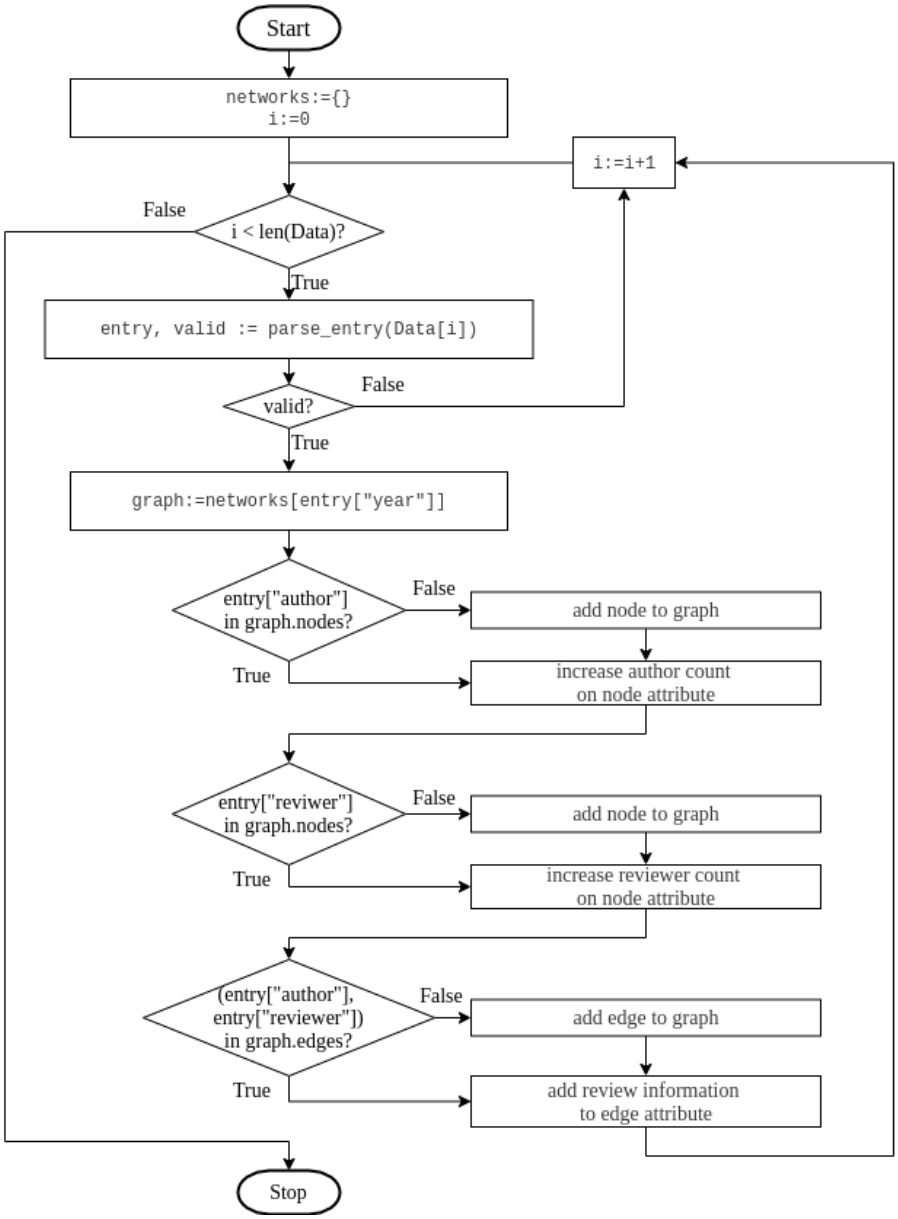

Figure 2.2: Add review data entry to graph

As result of the first step, there is a parsing object for each available data set that is able to translate the raw data entries into the common code review entity. The second step is to build a network structure of these entities. The framework used for creating the different networks is networkX (Hagberg, Schult & Swart, 2008). Figure 2.2 shows the implementation of firstly parsing the raw data and secondly adding each entity to a specific graph. In order to anlayse the evolution of the different projects, the entities are added to different graphs respecting the year of the *happened_on* field of the entry.

In the first step, the implementation initializes a dictionary containing a graph per data set, per year called *networks*. After the initialization, the program iterates through the raw data set and parses each given entry. By having a return value of the common data entry, as shown above, and additionally a valid flag. The implementation checks by the flag if the parsed entry can be added to the graph or not. If it is invalid, the program continues with the next entry of the raw data. Otherwise the graph object of the *networks* dictionary is selected by the *year* field of the code review entity. After that, the program checks wheter a node in the graph already exists for the author or not. If a node does not already exist, a new one is added to the graph with the unique identifier of the author. The same procedure is applied to the reviewer information and to the edge between the two nodes.

Beside the information of which natural person is interacting with whom, the code review entity includes more meta information like timestamps or reviewed files. In order to save this information within the network structure itself, the classes *EdgeAttribute* and *NodeAttribute* are designed to hold it. For each node and edge of the graph an instance of the respective class is added as attribute. This way every element of the network contains all relevant information in the term of the code review process and the graph can be splitted into subparts without loosing relevant information. A detailed view of the class diagrams is shown in figure 2.3.

The *NodeAttribute* contains a list of review ids and counts for the person beeing an author or reviewer. The list of ids is needed in case of having multiple interactions between two persons within a single review process. The counts increase if the review id is not in the list of ids. The *EdgeAttribute* class has the property weight, that represents the count of interactions between two nodes. In addition to that, the class holds a list of finished review entities that contain the start and end timestamp of the review and also the involved files within this review. Based on this structure of the network, each attribute class has different implementations of collecting the information for analysing.

Summarizing the section, a process was described to parse and standarize the heterogenous raw data and creating a code review network per year, containing all information in the network structure itself.

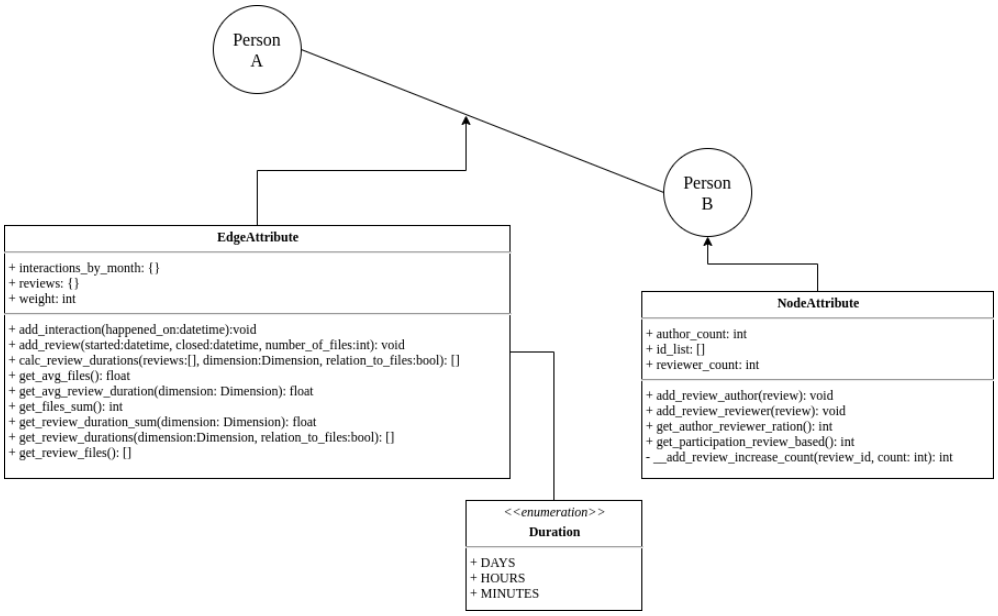

Figure 2.3: Node and edge structure in detail

2.4.3 Network Model for Analysing Code Review Data

Since the raw data sets can be transformed into code review networks, this section explains the model used on implementing evaluation methods on the created network data. The main approach hereby is to integrate the different parsers of each data set into one construct with the goal of analyzing the network structures itself and possible sub networks, as well as the evolution between each year. Additionally, the focus at this was to implement analysing methods only once and beeing able to call them on each graph or sub graph, that can be created in the concrete analysing process.

The overall model is shown in figure 2.4. The main parts of the model are the classes *AbstractParser*, *Analyser* and *Network*. The Parser class basically provides an interface and the algorithm described in section 2.4.2. It is responsible for parsing the raw data and creating a dictionary of code review network by year. The main parsing logic is individually implemented by a subclass for each specific raw data set. The *Analyser* class receives the instance of the parser and gets the network dictionary by calling the parsing method. As for this, the analyser is the instance of holding the whole data dictionary as it is. Therefore, it holds the possibility of comparing each year of the dictionary.

As last part of the model, there is the *Network* class. Each networkX graph of the dictionary is enveloped in an instance of Network. The class only knows as parameter the graph itself and is unknown about previous or upcomming years

of the dictionary. The implementation of evaluation methods specific for one single graph is done within it. All implemented methods can be seen in the class diagram in figure 2.4. Additionally to the evaluation methods, the class can hold a list of sub graphs called *communities_list*. The member of this list are also instances of the same class with a sub graph as value of the graph property. Additionally, the child parent relation between them is hold by a parent attribute pointing to the higher instance containing this sub network. This way, all the methods for evaluting the network are only implemented once and reused for each instance.

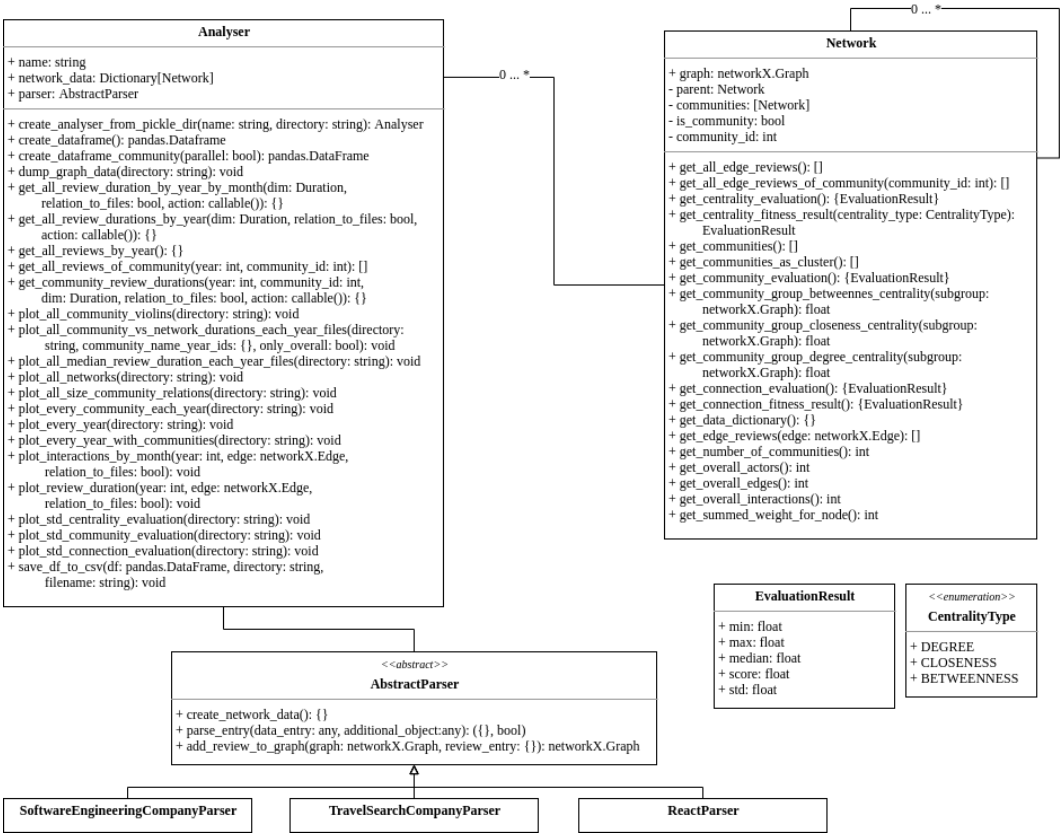

Figure 2.4: Class diagram of network model

As result of the section, there is a model that is capable of interpreting heterogenous raw data and creating a structure for evaluting the networks itself, as well as the evolution of the networks containing common context information about the code review process of the different data sources. With this step the goal of creating a model for analysing code review networks from heterogenous data sets is completed.

2.4.4 Analysing Code Review Networks

After the basic structure for analysis is given through the created network model, the following section deals with the concrete approach for answering the research question what structural information contains a code review network and do structural network information have an impact on the code review duration. Therefore, the section is divided into two different parts. The first part shows the analysis of the structural information of the network and sub networks theirself, whereas the second part deals with identifaction of the most outstanding sub network. The final part compares the outstanding sub network and the overall network through the code review information, hidden in the network edges and nodes. All analysed metrics are summarized in table 2.2.

Structural information analysis

The first step of the structural information analysis is to get an overview of the whole network created. For this, the evolution of the metrics [M1]-[M4] will be analysed. Actors describe the nodes in the network, whereas the edges show between which actor has been an interaction. The concrete interaction is explained through the weight of each edge. For every single interaction the weight is increased. Finally, the degree value shows the count of connections for every node. Each metric in this thesis is either a single value in terms of describing a whole network or an sub part of it. If the metric is defined on a node level, for example degrees, the measured sizes are the maximum, mean, median, minimum and standard deviation over the network or sub parts of it.

After getting an overview of the structure, the second step deals with the common centrality analysis of networks [M5]-[M7]. The three centrality metrics are perhaps the most used in literature and define the term centrality on a node based level of the graph. Betweenness centrality determines the centrality of a node with using the sum of fractions of paths between two other nodes, that contain the measured node and the overall number of paths between them. Closeness centrality stands for the reciprocal value of the distance from a node to the number of all reachable nodes. Finally, degree centrality referes to the degree value of a single node in relation to the number of nodes contained in the graph. (Freeman, 1977)

Therefore, the higher each value is, the more central is the node in terms of degrees, closeness and betweenness network. Both previous steps focused on the overall network therefore the third step tries to split the graph into subgraphs with a common behaviour. An approach for discovering those subgraphs are community detection algorithms.

Z. Yang, Algesheimer and Tessone (2016) compared different kinds of detection algorithms in terms of accuracy and performance. Thus there is not any information given on the raw data of concrete communities, the selection of the algorithm for this thesis depends on the performance and the approach of the algorithm itself. One of the best algorithm, depending on the sizes of the data sets, is Infomap with a performance of $\mathcal{O}(E)$, based on the number of Edges E (Z. Yang et al., 2016). The detection uses the idea of random walks accross the graph and describing the information flow between nodes by using coding theory (Rosvall & Bergstrom, 2008). The algorithm is additionally choosen because of the context of code reviews, where one of the main actions is the exchange of information between persons. Therefore, the approach of the algrorithm also covers this semantic action.

Table 2.2: Metrics for analysing the networks

Identifier	Name	Level
[M1]	Actors	Graph
[M2]	Edges	Graph
[M3]	Degrees	Node
[M4]	Interactions	Node
[M5]	Betweenness centrality	Node
[M6]	Closeness centrality	Node
[M7]	Degree centrality	Node
[CM1]	Average internal degree	Subgraph
[CM2]	Conductance	Subgraph
[CM3]	Cut-ratio	Subgraph
[CM4]	Edges-inside	Subgraph
[CM5]	Expansion	Subgraph
[CM6]	Fraction over median degree	Subgraph
[CM7]	Group centrality	Subgraph
[CM7.1]	- Betweenness	
[CM7.2]	- Closeness	
[CM7.3]	- Degree	
[CM8]	Internal edge density	Subgraph
[CM9]	Out degree fraction [ODF]	Subgraph
[CM9.1]	- Average	
[CM9.2]	- Flake	
[CM9.3]	- Max	
[CM10]	Triangle participation	Subgraph

As result of the detection, each graph of each year of each datasource is splitted into serveral subgraphs. With the set of the communities and the parent graph, the community metrics [CM1]-[CM10] can be calculated and analysed. Radicchi, Castellano, Cecconi, Loreto and Parisi (2004) are using quantitative definitions of a community, where the density of internal connections, as well as the number of edges that are leading outside the cluster are used. Furthermore, they distinguish the definition of a community in a strong and a weak sense. "In a strong community each node has more connections within the community than with the rest of the graph" (Radicchi et al., 2004, p. 2659), whereas communities in a weak sense are defined through the sum of all degrees inside the communities being larger than the sum pointing to the rest of the network. As result, the average internal degree [CM1], edges-inside [CM2], internal edge density [CM8] and the expansion [CM5], as number of edges that are pointing to the rest of the network, are useful metrics to evaluate a communtity in a graph. (Radicchi et al., 2004)

By crawling websites and identifying communities, Flake, Lawrence and Giles (2000) defined a community in a Graph $G = (V, E)$ "as a vertex subset $C \subset V$, such that for all vertices $v \in C$, v has at least as many edges as connecting to vertices in C as it does to vertices in $(V - C)$" (Flake et al., 2000, p. 152). Furthermore, with calculating the edges pointing inside and pointing outside per node, they defined a identifaction of communities by calculating the minimum cut of the two sizes. This leads to the metrics [CM9.1] with the average and [CM9.3] as the maximum fraction of edges that lead outside of a community, as well as [CM9.2] as the fragment of nodes that have a lower count of edges pointing inside a subset than outside. (Flake et al., 2000)

In addition to that, J. Yang and Leskovec (2015) are using the metrics conductance [CM2], cut-ratio [CM3], fraction over median degree [CM6] and the triangle participation [CM10] for the scorring and identification of communities. Conductance is explained as "fraction of total edge volume that points outside the cluster" (J. Yang & Leskovec, 2015, p. 184). The cut-ratio describes the fraction of existing edges leaving the cluster, whereas the fraction over median degree is measured by the fraction of nodes, that have inside a community a higher degree than the median value of degrees per node in the whole graph. At least, the triangle participation is given through the the fraction of nodes that belong to a triad. (J. Yang & Leskovec, 2015)

Finally, the last group of metrics [CM7.1] - [CM7.3] is not explained by the community internal structure or the relation to other communities, but the centrality in the whole graph. The centrality metrics refer to the betweenness, closeness and degree metrics on node level, but just for a whole subgraph. Degree group centrality stands for the fraction of non-community members pointing to inside members. The group closeness is defined as the "sum of the distances from the group to all vertices outside the group" Everett and Borgatti (1999, p. 8), that

describes how close the group is to other nodes in the graph. Lastly, the group betweenness centrality metric measures the sum of proportions of all shortest paths that goes through any node in the community. (Everett & Borgatti, 1999)

In summary, there are three parts in analysing the network from the overall structure to measuring the centrality on node level and finalizing it with splitting the graph into subgraphs for further analysis. Overall, the metrics are implemented with the framework CDlib (Rossetti, Milli & Cazabet, 2019), which builds additionally functions for community detection or evaluation methods on top of networkX.

Identifying the most outstanding sub network

After getting a structural overview of the evolution of graphs and their subparts with the given metrics to answer the first research question, there is a need to structure the given information to gain a concept for answering the question, if the structural information of the networks have an impact on the code review duration. The analysis is focused on information of the whole graph and subparts of it. Therefore, the approach for comparing the code review duration is to find the most outstanding community by the discovered metrics and compare the review duration of it with the whole graph to identify a relation.

The approach of finding the most outstanding community is to create a score calculated by the community metrics [CM1] - [CM10]. The metrics can be clustered into three groups of targets:

- Centrality metrics
- Internal community metrics
- Leaving community metrics

The centrality metrics describe the location of the community in comparision to the whole network. The cluster contains the group betweenness [CM7.1], group closeness [CM7.2] and group degree centrality [CM7.3]. In comparision to the centrality cluster, the leaving community cluster concentrates on the nodes inside a community that are pointing outside the community and not the relation to the whole network. Inside this cluster are the metrics conductance [CM2], cut-ratio [CM3], expansion [CM5] and the out-degree-fractions [CM9.1] - [CM9.3]. As third cluster, the internal metrics only evaluate the internal structure of a community and have no measurments for the relation to other subgraphs or the overall network. The cluster contains the average internal degree [CM1], edges inside [CM4], fraction over median degree [CM6] and triangle participation [CM10] metric.

With the creation of the three metric clusters, there would be three possible scores

to identify the most outstanding community. In order to calculate a score from the different metrics, each measurment has to be compareable and has to be in the same range. For example, the metrics that calculate a proportion are in a range of $[0.0 - 1.0]$, whereas the edges inside for example can be any natural number. For this reason the given metrics for each cluster are normalized by min-max scaling. After having compareable numbers, the relation for each metric inside a cluster has to be declared in order to gain a way for combining them to one cluster score. Each given metric over all clusters have the same pattern in the way of the bigger the value is, the better it is. For example, the internal community strength by the definitions above. So each normalized value is summed into a score for each group.

The scores will be called centrality, internal and leaving score in the upcomming parts. In an additional step, the sum of each score creates an overall score for the most outstanding community in the network. Due to the fact that each cluster score consist of a different number of metrics, each score is again normalized by min-max scaling before summing it up to the overall score. As result of this part, there are four scores based on the network analysis for each year that can be used to identify the most outstanding community for each cluster or of the whole graph for later comparison.

Analysis of the code review duration

The last step explains the analysis of the code review duration, based on the information contained in each edge of the graph or subgraph in terms of communities. Each edge of the graph contains list of all finished code reviews and also the involved files in it. First of all, there are some limitations that have to be set by the raw data and the split of the graphs per year:

- *Code reviews durations over years or seconds:* Each data set contains reviews that have a duration over years or are finished the moment they have started. These durations cause the fact, that the comparision of the average review duration with the average duration in communities might not reflect the real impact of the community with outstanding structural measurements. Therefore, the median value is choosen for the comparision.
- *Overall network has the community duration inside:* In the calculation of the whole network the graph also contains the community. That means, the median code review duration might be increased or reduced, due to the fact that the compared subgraph is also part of it.
- *Network is focused on concrete timestamps of interactions:* While parsing the raw data, the split into each year is done by the *happened_on* field that respresents the concrete time value of when the interaction occured. On

the other hand, the code review process is represented as a time period by subtracting the start value from the end value. In order to compare set the duration in relation to the structural information, the period has to be matched to a concrete time value. Due to that, the reviews are sorted by the start date and afterwards assigned to the year of the start date. For example, if the review started in December 2018 and finished in January 2019, the review is included in the median value of 2018.

Algorithm 1 shows the process for collecting the review information from the overall graph in pseudo code. Mainly, the program is iterating through each year of the data set and collecting every review duration, given on each edge of the graph. The duration is calculated by the substraction of the start date from the end date. Additionally, for evaluating the duration per file in a review process the duration is optionally divided by the number of files, given through an boolean parameter. The algorithm for collecting the duration from a subgraph behaves analogously to the shown with only iterating through a subset of the network data. Afterwards, all the median values of every collected time period is calculated. With this approach, the evolution of the overall graph and of the most outstanding community each year can be obtained and compared afterwards.

```
Data: networks: dict[Network], relation_to_files: bool
Result: median_durations: [float]
median_durations:= [];
durations_by_year:= {};
while year in networks.keys do
    while edge in networks[year].edges do
        while review in edge.review_data do
            if relation_to_files then
                duration := (review.end-review.start)/involed_files;
            else
                duration := (review.end-review.start);
            end
            durations_by_year[review.start.year].append(duration);
        end
    end
end
while year in durations_by_year.keys do
    median_value := median(durations_by_year[year]);
    median_durations.append(median_value);
end
```

Algorithm 1: Calculate median code review duration per year

2.5 Used Data Sources

The data used for this thesis is obtained by a software engingeering company, a travel search service provider company and the Github open source project React. The following section contains an overview of the basic conditions and dimensions for each data source.

The raw data of the software engingeering company contains 10'901 entries of 13-24 developer over a date range from 03/05/2013 to 11/23/2017. Each entry represents a single code review process and comprises an unique identifier for author and reviewer. Furthermore, each entry contains timestamps of the start and the actual review date as well as the number of files that were reviewed.

The code review data of the travel search service provider company was given through a database containing two tables. One with all pull request on several repositories and the other table containing all actions on each pull request. All pull requests include the code review information of the author and the involved files. Each action consists of an unqiue identifier of the actor and also timestamps when the action happenend. By joining the tables by the repository and pull request id the dataset obtains 406'865 entries that represent interactions within a code review. With a date range of 02/19/2016 to 06/22/2018 it represents collection of 288 to 456 developers.

Finally the third data source is the Github open source project React. The code review data crawled from this project is described in section 2.4.1. Overall the dataset contains 8'590 pull request with every comment, review, review comments and edited files in it. It was accessed on the 10/07/2019 and comprises interactions of 45 to 383 devlopers from a pull request range from 05/29/2013 to 09/19/2019.

All in all, the used data is heterogenous with an open source project and two industry data sources in different file formats with various meta information. Furthermore, the sizes of developers involved vary in a wide range from 13 to 456 over all data sources united.

2.6 Research Results

2.6.1 Code Review Network Structure Analysis

The following section shows the result after executing the approach described above. It is subdived into the structural analysis of the overall networks each year, the analysis of the communities inside the networks each year with identifying the

most outstanding community and the context analysis how the most outstanding community differentiates in terms of code review duration against the overall network. Every subsection contains the most significant data in this explorative analysis, whereas all results discovered are shown in the appendices A to E.

Structural analysis of data from software engineering company

The first raw data set that is transformed into a network is from the software engineering company. Figure 2.5 shows each graph created by year. Comparing to the upcomming graphs, the network contains only a small amount of actors. Because of that, the visualization of the graph is clearly arranged in the figure. Just with the plotted graph, it can already been seen that the project becomes more complex over the years and with an increasing amount of nodes, the interactions between the actors also have increased. Furthermore, the graph is fully connected. Only in the years 2015 and 2017 some nodes occure that have only a single connection to the rest of the network.

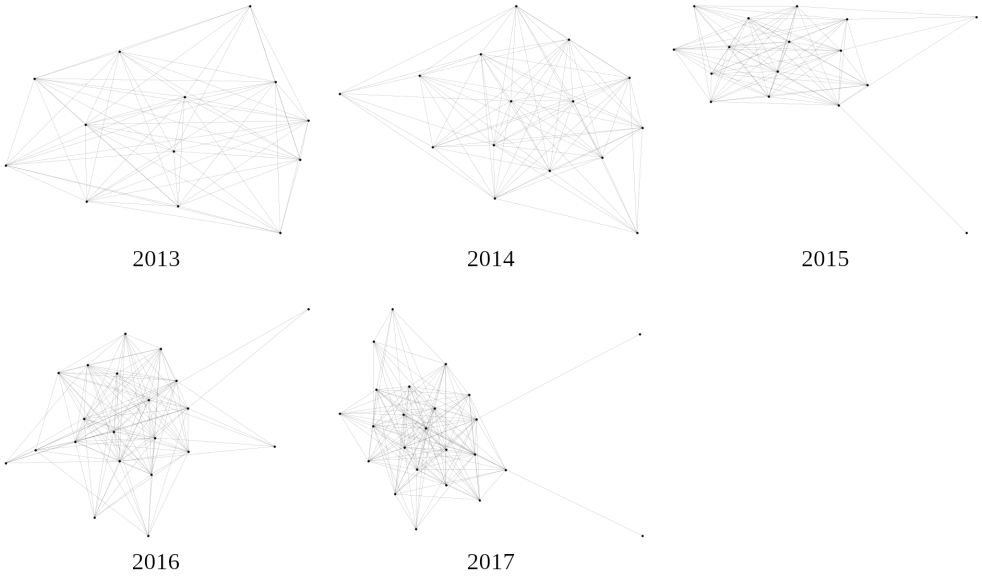

Figure 2.5: Code review network over years (Software engineering company)

A more profound look on the created graphs is shown in table 2.5. The table shows only metrics in terms of the overall network structure. The numbers of actors increased from 13 to 24 in a range of 4 years. With the raise of the involved actors, the edges increased with nearly the same growth rate. The actors increase with an average growth of 16%, whereas the edges have an average addition of 25%. The degree value based on the node level is splitted into the

maximum, minimum and mean value. In addition to the fact that the graph gets more complex with every year, the maximum value shows also an increase. That means, with every increase on the actor count, the nodes are creating more connections and the overall structure get more complex, instead of beeing stable. The minimum degree of 1 to 2 in the years 2015 to 2017 could have reasons like business modification or new employees in the end of the year period, but can not be explained with the given data. The mean value of the degree is close to the maximum value. This implies the distribution per node is equal and the code review process does not base on single active nodes.

Table 2.3: Network structure overview (Software engineering company)

	2013	2014	2015	2016	2017
Actors	13	15	16	21	24
Edges	70	88	89	137	165
Degree					
- max	12	14	14	19	21
- min	8	7	1	2	1
- mean	10.77	11.73	11.13	13.05	13.75
Interactions	1205	1822	1821	2431	2783
Interactions per node					
- mean	185.34	242.93	227.63	231.52	231.92
- median	180	243	253	239	241,5

As last metric of the overall network structure, the interactions are shown for the overall network in sum and the interaction on each node in mean and median value. The interactions are calculated by the sum of the edges multiplied with the corresponding weight. The overall interactions reflect, as well as the actors and edges, the growing complexity over the analysed years. Furthermore, the years 2014 and 2015 show that if the participating developer are stable, all the resulting metrics remain also stable. The last fact that can be retrieved from the data is that the mean and median value of the interactions per node hardly vary. This confirms that the interactions in the code review process are equally distributed over every node.

Lastly, the common centrality metrics are calculated in the overall graph analysis. Each range of centrality metric per year is shown in figure 2.6. Every range is visualized by a bar where the dot shows the average value, whereas the thicker bar displays the standard deviation and the thiner bar the range between maximum and minimum value of the centrality metric. The plotted range of the degree centrality shows a downward trend in the average value with a percentual change

of 7% to 12% from a value of 0.897 to 0.598 over the years. In the first two years both displayed ranges vary from the subsequent years. In the first part the standard deviation is around 0.1 and the range between maximum and minum is about 0.4, whereas in the subsequent years the range doubled to a value around 0.23, in terms of standard deviation, and 0.86 for the range between maximum and minimum. To be highlighted is also the fact that the maximum value in the first two years is 1.0, that means that at least one actor exists that has an edge to every other actor in the network.

In the visualization of the betweenness centrality a slightly increase of the average value from 0.009 to 0.019 is identifiable. The plotted metric can be divided into two parts. In the first part the standard deviation has only a small size with a value of 0.005 and 0.008, whereas in the subsequent years the value is quadrupled compared to the first two years. Also the range between maximum and minimum value increased from the year 2015. Especially noticeable is the year 2015 with the highest values in terms of standard deviation and range from maximum to minimum. Furthermore, the standard deviation outranges the minimum value, that means the centrality betweenness of each node is very widely spread. Due to the meaning of the metric, it shows that with increase of the overall network the shortest path between nodes, that go through a single node, vary more widely.

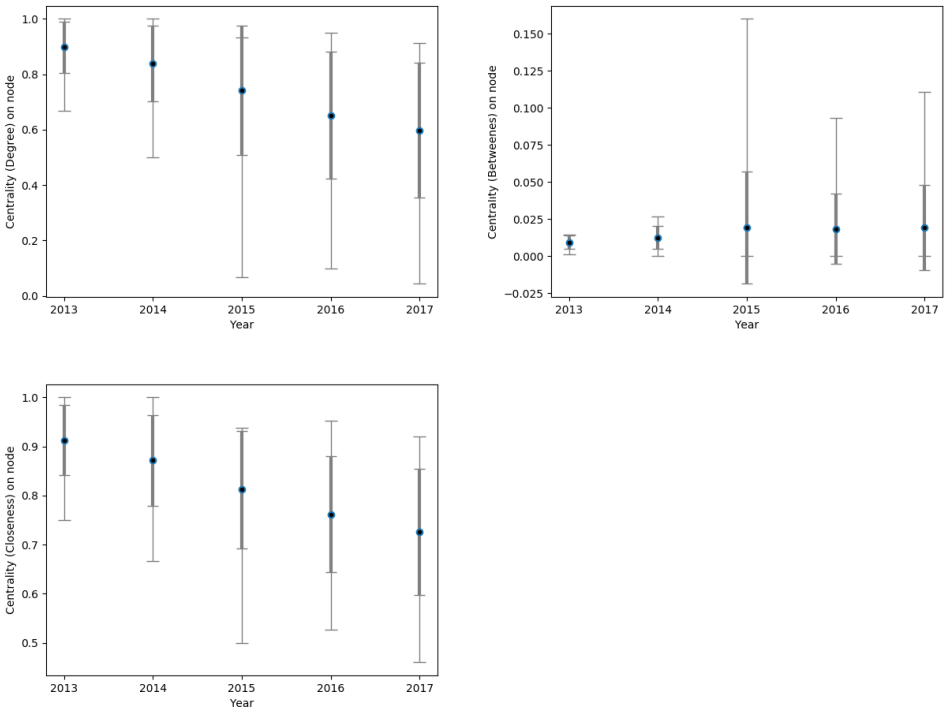

Figure 2.6: Centrality metrics over years (Software engineering company)

Finally, the closeness centrality also sees a downwards trend from an average value of 0.913 in year 2013 to 0.726 in year 2017. The decrease of the value is nearly constant by a percentual loss of 5%. The standard deviation first slightly increased from 0.071 to 0.120 in the years 2013 to 2015 and then stays stable with a value about 0.120. The year 2015 also stands out in the closeness metric, because in every other year the standard deviation is roughly placed in the center of the maximum and minimum value, whereas in the year 2015 it is placed on the top end of the range. This can be explained with the fact that there is at least one node with an extremly low value of the closeness centrality. In figure 2.5 a single node with only one edge to the rest of the graph can be seen, what explains the placement of the standard deviation in the year 2015.

Structural analysis of data from Travel search company

The second raw data set, that is analysed with the described approach, comes from the travel search service company. Figure 2.7 shows the plotted graphs from the years 2016 to 2018. Due to the mass of involved actors compared to the first data set, only few a few findings can be made from the visualization. The complexity of the graph increased and it can be seen that subgraphs or communities are gradually forming over the years. Except the year 2018, there are a few nodes that only have a connection between them and not the the rest of the network. In the last year every node is connected to each other.

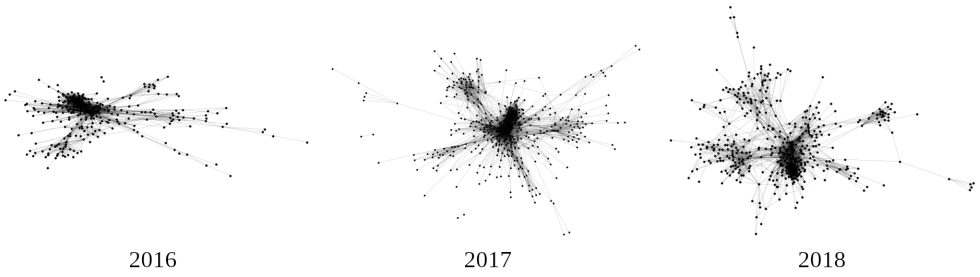

Figure 2.7: Code review network over years (Travel search company)

A more detailed breakdown of the graphs is given in table 2.4. The participating actors raise from 288 in the year 2016 to 456 in the year 2018. The last year is not fully given through the raw data as the others and only shows half of a year. Nevertheless, the number of involved persons are increased and the project gains in complexity over the years. The edges in the overall network raise from a value of 2'201 to 4'053 and decrease afterwards to 3'097. The same pattern can be seen on the degree metrics, as well as the overall interactions summarized in the table. The minimum degree count for every year is 1 and therefore not shown

in the table. Worth mentioning is the fact that the relation of the maximum degree value per node in relation to the participating actors decreases over the period of observation. This implies that the graph is more connected over various actors instead of single nodes. Additionally to be mentioned is the fact that by increasing the number of actors from 288 to 456, nearly the same amount of overall interactions took place at half of the time range in 2018 compared to 2016.

The mean value of interactions has a small decrease of 4% points from 492.94 to 470.90 and a bigger decrease of 39% points to 287.64 in the last year. The median value however increased from 204.5 to 248 and decreased afterwards to 180. The drop from the year 2017 to 2018 may be caused due the fact that the metrics of 2018 are calculated with a raw data set of half the year, whereas the previous year contains a fully time range of a year. The opposite growth of mean and median value of the first two years implies that the participation in the code review process is more evenly distributed than in the first year.

Table 2.4: Network structure overview (Travel search company)

	2016	2017	2018
Actors	288	455	456
Edges	2201	4053	3097
Degree			
- max	113	143	71
- mean	15.5	17.82	13.58
- median	10	12	10
Interactions	69998	107130	65582
Interactions per node			
- mean	492.94	470.90	287.64
- median	204.5	248.0	180

In case of centrality metrics, shown in figure 2.8, the same visualization method, as described in the first data set, is used. Starting with the degree centrality, the average value decreased from 0.055 over 0.039 to a value of 0.030 with a loss about 25% over the three years. Also the range between maximum and minimum degree centrality decreased from 0.40 over 0.31 to 0.15. This leads to the result that the standard deviation also smallers it range from a value of 0.055 over 0.040 to 0.028 with a loss about 28% points each year. Evaluating the evolution, it means that the degrees got more evenly distributed over the nodes and outliers that have clearly higher degree value. In terms of the code review process, this means that

more persons take part in the review process with more different actors than in the previous years.

The closeness centrality behaves equally to the degree centrality. The average value per node starts at value 0.351 then took a loss of 5% to a value of 0.333. Subsequently, the loss doubled to a resulting mean value of 0.299. The distance between maximum and minimum value became smaller as well. The range decreased from 0.57 over 0.51 to a value of 0.28. A difference between the metrics can be seen in the placement of the average dot and the standard deviation on the min-max bar. In the graphic the average is clearly more centralized in the bar, that means the closeness centrality value per node is more equally distributed on every node than the degree centrality. Overall, the visualization indicates that although the average closeness of the nodes to each other decreased, every node is more involved in the network structure due to the fact that the range between maximum and minimum become smaller over the time.

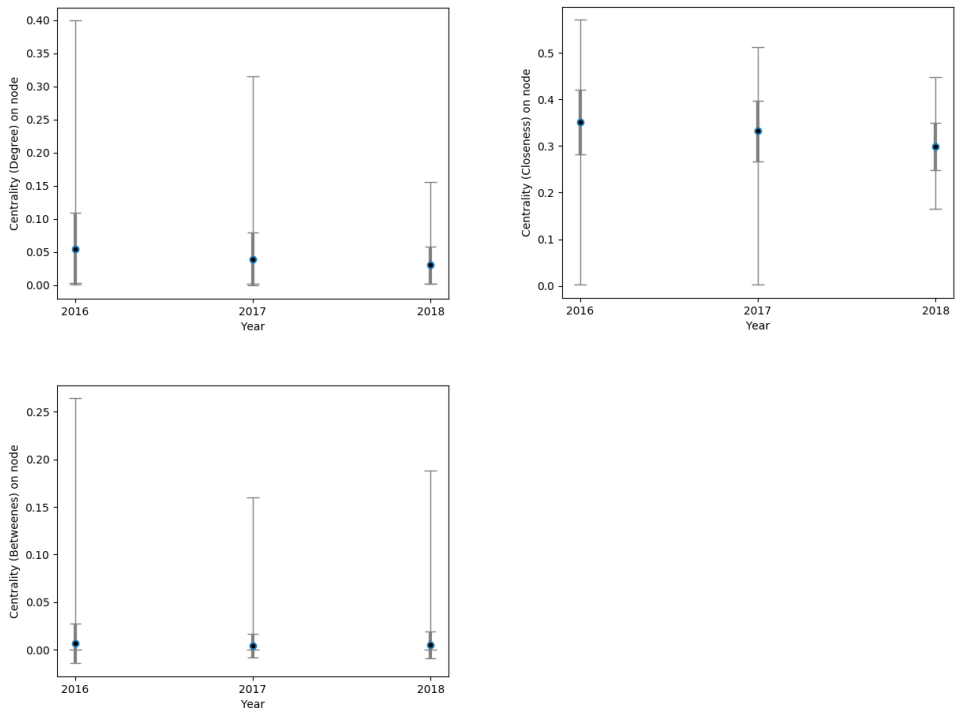

Figure 2.8: Centrality metrics over years (Travel search company)

Lastly, the betweenness centrality decreased from the first to the second year with an average value of 0.007 and a loss of 33% points to the value 0.004. Afterwards, it slightly increased to a value of 0.005. Where in comparison the previous two metrics have a constant decrease in the range between maximum and minimum, as well as standard deviation value the betweenness centrality also

firstly decreased and then slightly increased again. The min-max range started with a difference of 0.26 to 0.16 and finished with a value of 0.19. Worth to mention is the placement of the average and the standard deviation bar. It is placed low on the min-max bar that means the majority of nodes do not appear on the shortest path between other nodes, but there are some outstanding nodes that are relevant for the shortest path.

Structural analysis of data from React

The last analysed raw data is from the open source project react. With 7 years of review interactions the data set has the longest period of time. Figure 2.9 visualizes the evolution of the created graphs. From visual point of view the graph evolves till 2017 and then decomposes slightly. Furthermore, especially in the later years the graphs has some key nodes that connect subgraphs with the rest of the network.

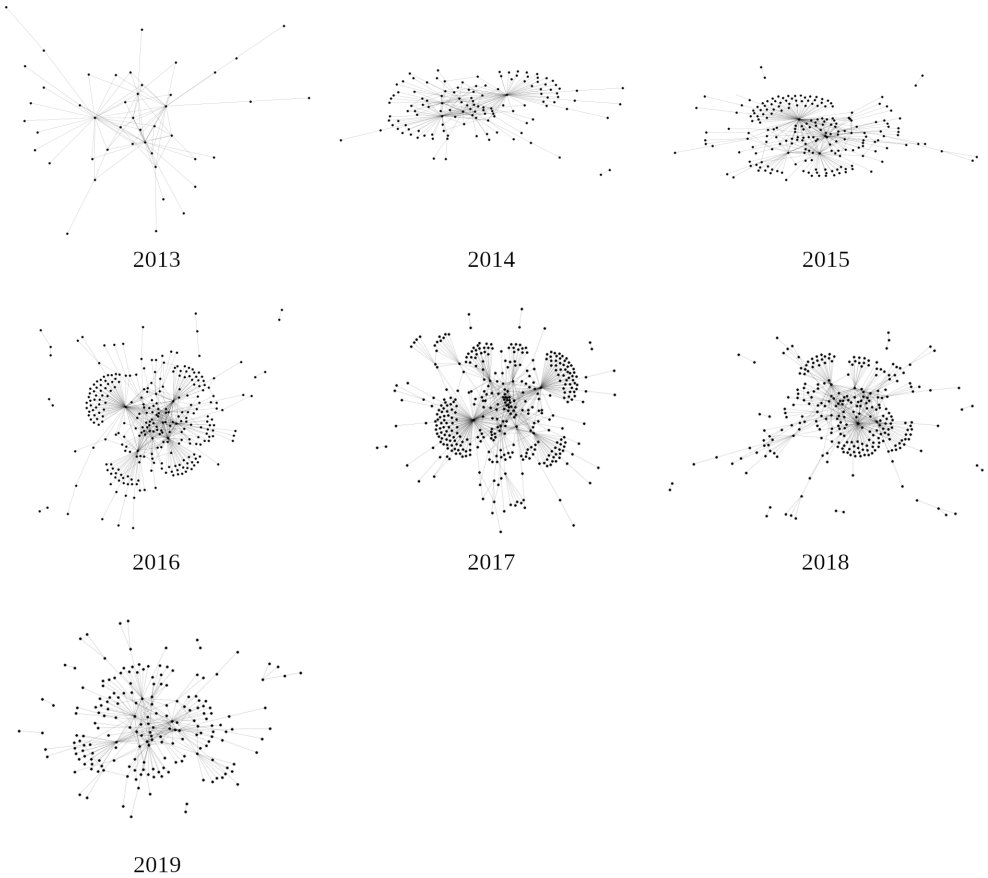

Figure 2.9: Code review network over years (React)

Additionally, the majority of nodes do only have one edge directing towards the center. Overall, comparing to the other data sets, the plotted graph appears in a star shapped form. Visually, the code review network implies that the process is ensured by individually actors. A more founded look is provided in table 2.11.

The visually assumption of 2017 beeing the peak of the participating actors, can be proven by the calculated numbers. The graph starts in year 2013 with 45 actors and gains on complexity till 2017 with the highest value of 383 persons. Afterwards, the number of actors decreased to a value of 191 in the last year. The same progress of increase to the year 2017 and then decrease to the last year can be seen by the overall edges in the graph. The lowest value of edges is in the year 2013 with constant increase to 558 in the year 2017 and after that a decrease to the value 286 in the last year. Worth to be mentioned is the relation of the actors and edges, shown in figure 2.10. The relation over the years is nearly linear with a R^2 value of 0.985.

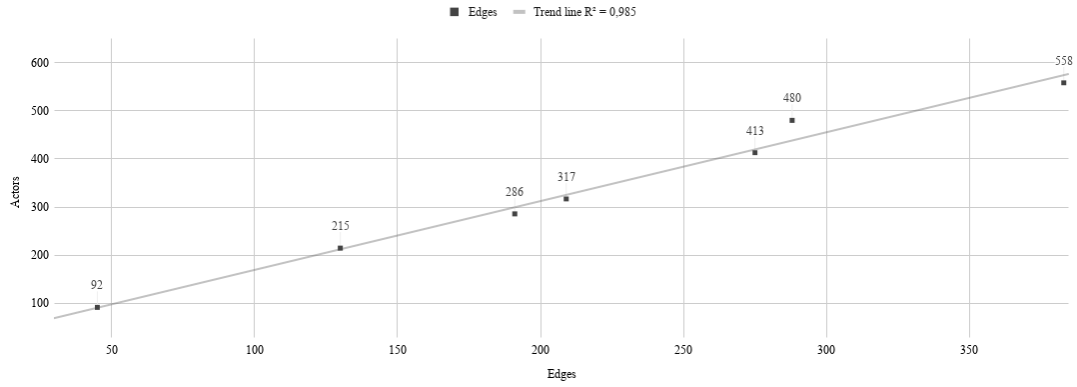

Figure 2.10: Relation between actors and edges (React)

Continuing with the degree value, the calculated numbers show a huge difference between the maxmium, mean and median value. While the mean degree is relatively stable around value 3, the maximum degree increases from 25 in 2013 to 148 in 2017 with a short drop in the year 2016 with 99. Afterwards, it decreased again to the value 35. This reinforces that the network relies on few key nodes with a much higher degree than the rest, as also seen in the plotted graph. Additionally, this can be seen on the median value that stays at value 1 from the 2014 to 2019. The same pattern is also recognizable from the interactions count per node. The mean value on every year is many times over the median interactions per node. The progress of the overall interactions of the network reflect the progress of the actors or edges. Starting from 591, the value increases to 6435 in 2017 and decreased again to a value of 3155.

Figure 2.11 shows the evolution of the centrality metrics of the data set. Begin-

ning with the degree centrality on the top left, the average value behaves exactly the opposite way as the actors or edges. Starting with a value of 0.093 in 2013, the value sinks with a loss around 20-70% points till the value 0.008 in the year 2017. Then the value raises up to 0.016 from 2017 to 2019. In comparison to the actors and edges each year, the opposite behaviour can also be explained by the fact that the network is controlled by single key nodes and has a star shapped form. With each actor added to the graph over the years, the key nodes with high degree centrality stay the same, while the average sinks. The pattern can also be seen in the evolution of the standard deviation, that stays at the bottom of the min-max bar with decreasing values till 2017 following with a raise till the last year.

Table 2.5: Network structure overview (React)

	2013	2014	2015	2016	2017	2018	2019
Actors	45	130	209	288	383	275	191
Edges	92	215	317	480	558	413	286
Degree							
- max	25	65	111	99	148	114	35
- mean	4.09	3.31	3.03	3.33	2.91	3.00	2.99
- median	2	1	1	1	1	1	1
Interactions	591	865	1146	3738	6435	4113	3155
Interactions per node							
- mean	26.27	13.31	10.97	25.96	33.60	29.91	33.04
- median	5	2	2	2	3	3	2

The average value on the closeness centrality has nearly a constant decrease from the value 0.425 in 2013 to 0.273 in 2019. Only in the year 2017 there is a little increase with 3% points compared to the previous year. The range between maximum and minimum also repeats this behaviour except for the first year. From the second year the range began with a value of 0.63 and sinked to 0.56 in the year 2016. After a raise to 0.58, the range decreased again to a final value of 0.47. Due the fact that the minimum value stays stable by having at least one node with a closeness centrality about 0.004, the evolution of the min-max range also represents the evolution of the maximum closeness centrality node. Overall, this can also be explained by the shape of the network. The nodes near the minimum value are in the outer part of the network, whereas each node that is more central gets a higher closeness centrality value by having the shortest path to the outer part in it.

Lastly, the betweenness centrality per node shows the same evolution as the de-

gree centrality with a smaller value range. The average centrality value decreased from a value of 0.034 to 0.005 in the years 2013 to 2017. Then it raises again up to a value of 0.010. The minimum value stays constant by the value 0 due to the fact that nodes in the outer part only have one single edge pointing to the center and not a single shortest path pointing through it. Also worth to be mentioned is the fact, that the standard deviation is sinking over the whole period of observation except in the year 2017. It can be explained due to the fact that even if the range between maximum and minimum increased, the increase of actors over the whole time period adds only few key nodes in the center, but a high number of nodes in the outer part. Therefore, the standard deviation tends to become smaller due to the majority of outer nodes.

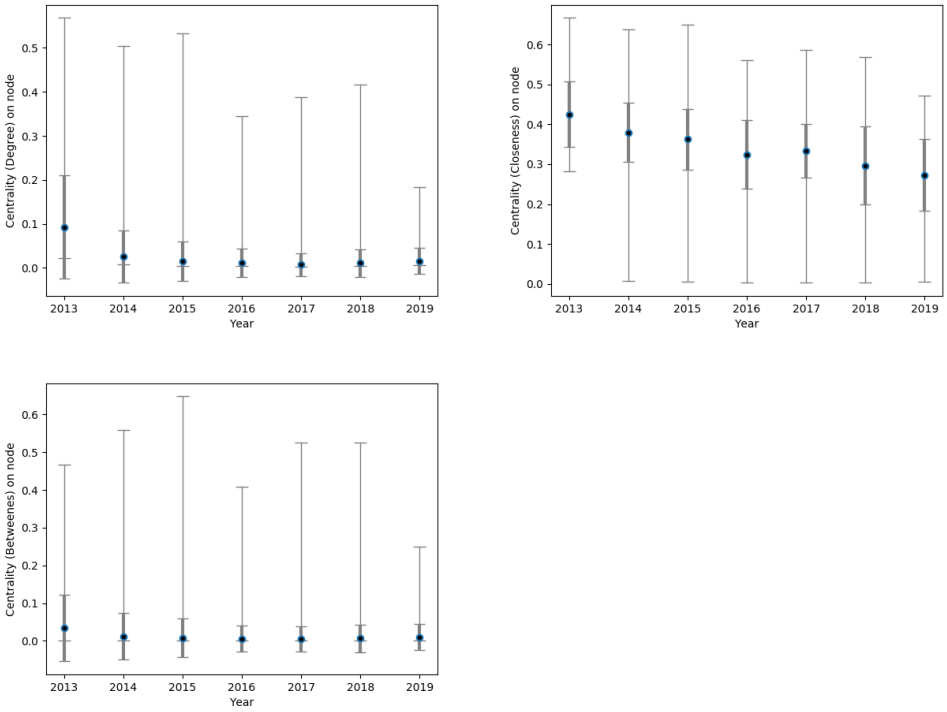

Figure 2.11: Centrality metrics over years (React)

Comparing network structure of all data sets

After analysing the overall graph structure for each data set itself, the following subpart concentrates on comparing all of them. The previous analysis was split in two parts for each data set. First part consisted of the key facts on the overall graph level, whereas the second focused on the centrality on node level. Due to the difference of each data set on graph level, the comparision is limited to the centrality metrics on the node level which is more compareable. Figure

2.12 shows a visualization of the mean centrality metrics calculated each year for each data set. Each finding of the comparision is afterwards set in relation with the facts of the findings on the overall graph level.

Beginning with the closeness centrality, all given data set show the same evolution. Since the start of the period of observation the closeness centrality on each data set has decreased constantly. They all have in common that the graphs get more complex every year with a raising number of actors and edges. Due to the fact that the closeness centrality measures the shortest path distance over all reachable nodes, the fact that more actors are involved over the time decrease this factor and the closeness of each node to another sinks. Also the closeness centrality of the software engineering company data compared to the other data sets nearly has the doubled value on each year. The reason for this could be the overall size of the graph. Lastly, the industry data set of the travel search company and the open source data set from react have nearly the same closeness values in the years 2016 to 2018. Although they differ much in terms of actors, edges and interactions and overall shape this has no impact on the closeness centrality in this years.

Figure 2.12: Centrality metrics over years in compare

In terms of betweenness centrality, the data sets are not sharing a common pattern. Where the smaller graphs of the software engineering company increased the value of betweenness centrality, the graphs of react decreased their mean

value over the time. At the year 2017 when the graph of the react data set has the most complexity, the centrality value is at the lowest point. After that, the complexity of the graph gets lower and the centrality value raises again. The evolution of the open source data set implies that the complexity of the network has an impact on the betweenness centrality. This implication can not be proved by the software engineering company data, because with raising complexity the centrality value also increased within. Beside complexity, both data sets differ in overall size and the shape of the growing network. Therefore, the different behaviour can only be explained through that. The travel search company shows the same pattern as react but with a period of three years it also could be by chance.

Lastly, the average degree centrality has a common behaviour in all three data sets. It started from a higher value and has a constant decrease over the years. Especially the software engineering company data has a more strongly decrease from the value 0.897 to 0.598 from the year 2013 to 2017. The fact that compared to the other data sets the average centrality of the software engineering company graphs is multiplie times higher implies that the size of the network, has the highest impact in terms of degree centrality in code review networks. Furthermore, every graph of each data set got more complex by each year. Therefore, with more actors involved the centrality sinks. The open source data set has overall the lowest degree centrality compared to the other two data sets. This might be caused by the star-shape of the graph that causes a lower degree of outer nodes. Summarizing the comparison, the three data sets have in common that the closeness and degree centrality decreases, while the graphs are gaining more complexity over the time. The smaller network structure of the Software engineering company has a positive impact on the average node centrality. Especially in terms of betweenness centrality, the evolution differs by having a smaller actor size. Between open source and industry data in a compareable size, there are hardly any differences to be seen.

2.6.2 Community Structure Analysis

The following subsection shows the analysis of the community structure. The community detection algorithm Infomap (Rosvall & Bergstrom, 2008) is applied to each year of each data set. Following the research approach the community metrics shown above are calculated for each discovered community. While focussing on identifying the most outstanding community the subsection only covers the most remarkable findings and main facts for each data set in the first part. The full results can be seen in appendix D to E.

Community analysis of data from software engineering company

The first data set the community detection algorithm was applied is the industry data set of the software engineering company. Due to the fact that the size compared to the other to data sets is much lower the algorithm can not discover any communities in any year of the data set. In addition to that as seen in the network structure analysis above every node has a very high degree centrality. This means the algorithm can not discover any subgraphs by a lower degree value as information flow cut in the graph. Overall the data set can be seen as a single community already analysed in the previous section. In the further thesis the data set is no longer considered due to the fact that no community data can be discovered.

Community analysis of data from Travel search company

The community detection algortihm discovered several communities in the three years of the travel search company data set. In the year 2016 there 12 communities identifyable with an average size of 23.67 actors. The number of communities raised in the year 2017 to 17 communities with also raising the average number 26.76 of actors in it. In the last year the algorithm detects 24 subgraphs with an lower average member size of 19.00. Figure 2.13 shows a visualization of the graphs by highlighting each community in a different color.

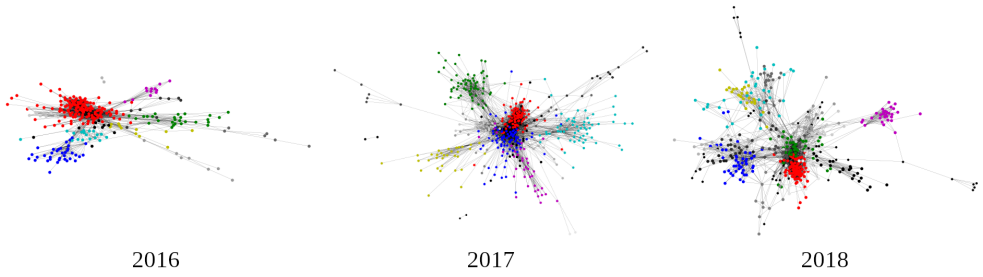

Figure 2.13: Communities over years (Travel search company)

From a visual point of view the graph in 2016 is determined through the single red highlighted community that takes a central role in the whole network. The outer parts of the graph are separate communities connecting to the central red subgraph. Worth mentioning is the dark blue colored community in the bottom left hand corner. The subgraph is completly separated from the red central community. The light blue subgraph at the top of it acts as a connection to the central part.

In the upcomming years the graphs get more divided in a star like shape with more different communities in the outer parts of the network. In the year 2017 there are two central communities colored in red and blue that connect the outer subgraphs to each other. Especially in the last year the growth of communities in the overall shape can be seen. Furthermore, each community has visually fewer members and the centrality of the communities sinks. Every year has at least one community that takes the center part of the network and connects the outer subgraphs with each other. For a deeper look in the community structure figure 2.14 shows the evolution of the community metrics fraction over median degree and the triangle participation ratio in a violin chart. The chart shows the distribution of communities in relation to the community metric on the vertical axis. While the white dot inside the area shows the median value and the thicker bar shows the interquartile range. The widness of the area visualizes the distribution of the communities with the value of the metric.

In terms of fraction over median degree a trend of more communities having a higher value can be seen in the evolution over the years. Even though the median value of the year 2017 is with a value of 0.46 slightly higher than the median of 2018 with 0.44 the interquartile range of the last year is visually smaller and therefore there are more communities in the the higher range of the metric. Overall with getting more and smaller communities over the period of observation there is a trend of having a higher internal community connection than the overall graph.

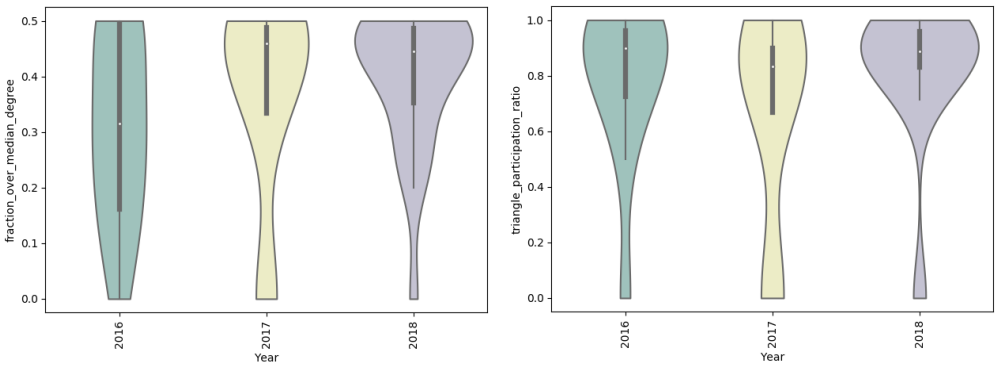

Figure 2.14: Community metrics in a violin chart (Travel search company)

Another indication for having more and especially stronger communities over the years is the triangle participation ratio. Triangle participation describes the ratio of nodes that take place in a triad. Even in the first year the communities have a very high ratio of having the triad relation inside. With gaining more communities in the second year the interquartile range slightly drops with a mean value of 0.83. Afterwards, it raises again to a value of 0.89. Especially the

distribution of the last year implies that the more and smaller the communities are inside the network, the more the triangle participation ratio and therefore the internal strength raises.

Figure 2.15 shows the relation of the community size to the edges inside and the internal edge density. The horizontal axis shows the size of each community, whereas the vertical axis displays the measured metric. Each year is visualized in a different color. Starting with the edges inside the graph shows a nearly linear relation up to a size of about 80 actors and 600 edges inside. Three communities are outliers of this relation between the sizes and edges. Each outlier is the community with the biggest size of actors for each year. Almost every community is arranged in the bottom left hand corner by having a size between 1 and and 50 and about 1 to 250 edges inside. Three subgraphs of the year 2017 fall out of the pattern by having up to 80 members and about 600 edges inside, but are also alligned at the linear relationship between the metrics size and edges inside.

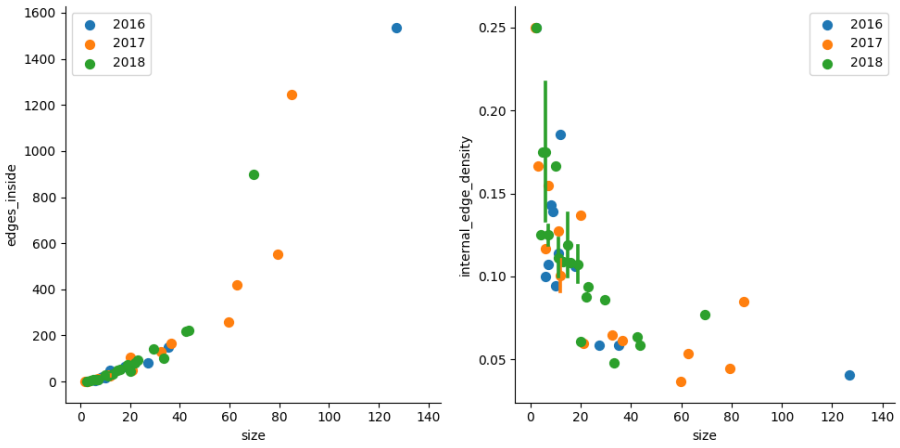

Figure 2.15: Community metrics in relation to size of community (Travel search company)

The right plot of the figure shows the relation between the internal edge density and the size of the community. The vertical lines that occure in the green color are shown if multiple communities have the same size but different edge density. Overall it can be seen that the higher the size of the community gets the lower is the internal edge density. Two communities of the year 2016 and 2017 fall out of this pattern by having a higher density compared to subgraphs in the same size range. They are also the communitie with the most members corresponding to their year. Additionally the plot shows that only a few subgraphs have a member count higher than 20 while the other years are more evenly distributed. Summarizing the findings, there are some communities in the data set that are more outstanding than others in different ways. The network color visualization

shows communities that are more centralized than others and therefore also have more connections outside their community zone. In terms of size there are also communities that fall of the pattern.

Community analysis of data from React

The network of the open source data is more structured by having multiple key nodes that connect subgraphs to the rest of the network as described in chapter 2.6.1. After applying the community detection algorithm the star shape like form can be seen more clearer in figure 2.16. The number of discovered communities starts with 4 at the year 2013 and increased up to a count of 42 member in the year 2017. Afterwards, the value decreased again to 29 in the last year. The average member size of the communties varies from 6.19 to 11.25. The years from 2015 to 2019 have an average member count about 8.

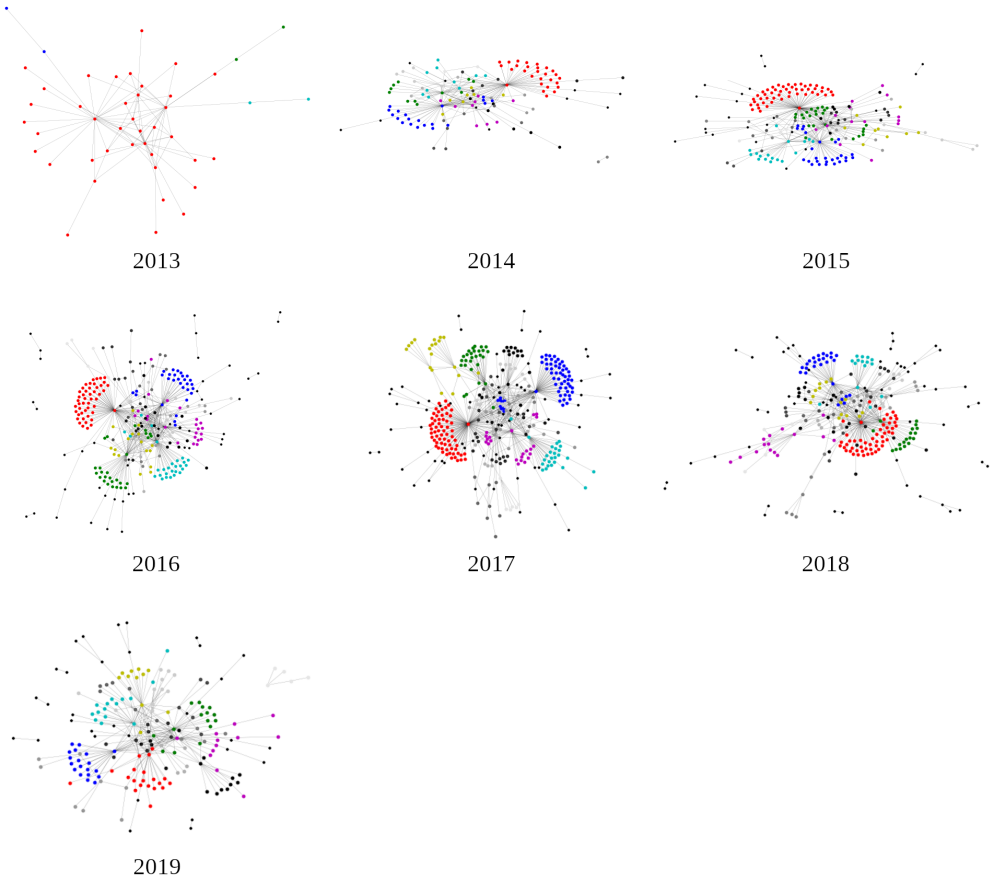

Figure 2.16: Communities over years (React)

Visually the most complex and also most community containing network in year 2107 has an average size of 9.12 actors. With a count of 65 actors the year 2017 has also the biggest community over all years. Due to the fact that the network is structured through several key nodes every node on the outer side of the network is combined to a community by the fact to which key node it has a connection. Despite the first year each upcomming graph also contains nodes that connect to multiple key members of the overall graph. These nodes are mostly combined to a community by the pattern of having the same connections to key nodes. Additionally every visualized graph despite the year 2013 has at least two nodes showed as a community that have no other connection than to each other. Therefore, the smallest community size is two for every year observed.

In figure 2.17 the distribution per year of the metrics edges inside and conductance can be seen in a violin chart. The distribution of the edges inside show that the majority of communities except the year 2013 have an edges inside count below 20. Especially if the distribution is compared to the average member count of the subgraphs each year the average edge count per node in the majorities of communities is about 1 to 2. Only a small set of communities have a higher edges inside count. Except the first year with a different network structure compared to the other years the most edges inside of a subgraph are also in the most complex year 2017. Also worth mentioning is the fact that the median edges inside count is the lowest in the year 2017 with a value of 2. Therefore with an increasing count of actors participating in the network the edges inside the communities decrease due to the star shaped form.

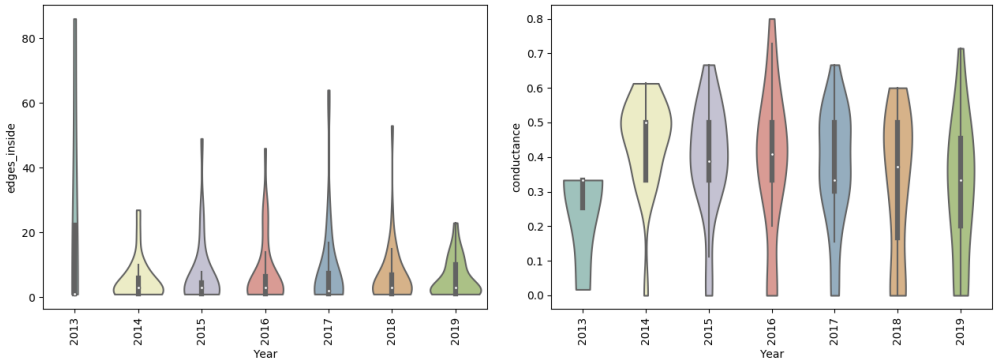

Figure 2.17: Community metrics in a violin chart (React)

The distribution of the conductance value that calculates the fraction of total edge volume that points outside of a subgraph shows that in the year 2014 most of the communities have a fraction of 50% with connections leaving the community. In the upcomming years the median value decreased to 33% and is more wideley distributed than in the beginning years. The community with the highest

conductance value is in the year 2016. The year 2013 acts completly different in both visualization because of the different overall network structure in the beginning of the project.

As seen in the community analysis of the travel search company the size of the communties are an important factor for the community structure. Therefore, the relation between size and the metrics average internal degree as well as internal edge density is shown in figure 2.18. Up to a size of 10 actors the average internal degree doubles the size from 1 to nearly 2. With a member count higher than 10 the average internal degree stagnates around the value 2. Only 6 communities have an higher average internal degree value. Especially the year 2013 falls out of the pattern by having an average value of 4.5 which can be explained of the small numbers of subgraphs detected and the different network structure in the first year.

Comparing the size against the internal edge density also a clearly pattern can be discoverd. The bigger the size of the community the smaller is the value of the internal edge density. The relation can be described as a negative power function. Only one community falls out of the pattern by having a higher internal edge density with a size of 39 members compared to the other subgraphs. Furthermore, only four communities, all of them in the year 2019, have an internal edge density higher than 0.10, whereas the communities with the lowest density and a member size over 45 are only in the year 2017.

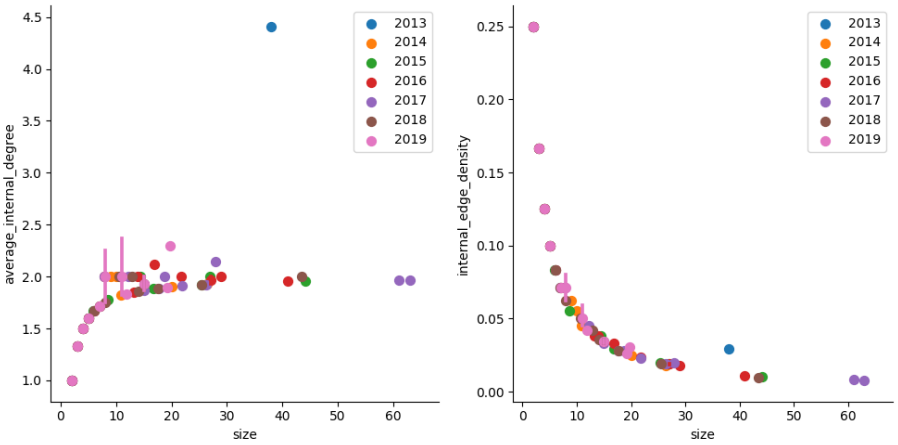

Figure 2.18: Community metrics in relation to size of community (React)

Overall both plots both show the expected relations for the given network structures. The open source data set is defined through several key nodes that also define the given communities. Therefore, all nodes in the outer part of the network hardly have more than a single edge connecting to the rest of the graph. Compared to the industry data set the network is strongly influenced by the open

source structure of having different contributers and only several maintainers for the project.

Identifying the most outstanding community

After calculating every community metric described in table 2.2 the three scores internal, leave and centrality are calculated following the approach above. The results for the travel search company can be seen in table 2.6 and for the open source project react in table 2.7. Each community is identified by a given id through the network model. The tables show the community with the highest overall score that is combination of the normalized value of the internal, leave and centrality score. The value in the parenthesis gives the highest score given in the communities of the calculated year. Due to the fact that the overall score is the sum of three normalized values the highest possible value is 3.

Table 2.6: Scores of communities (Travel Search company)

Year	ID	Internal Score	Leave Score	Centrality Score	Overall Score
2016	1	3.71 (3.71)	1.24 (4.09)	2.72 (2.72)	2.09
2017	2	2.70 (3.99)	5.54 (5.54)	2.73 (2.73)	2.43
2018	3	2.53 (3.49)	5.49 (6.49)	2.25 (2.25)	2.19

Table 2.7: Scores of communities (React)

Year	ID	Internal Score	Leave Score	Centrality Score	Overall Score
2013	1	4.06 (4.06)	0.93 (1.62)	3.00 (3.00)	2.16
2014	9	1.65 (1.68)	3.85 (4.99)	0.81 (2.03)	1.24
2015	1	0.90 (1.65)	2.83 (3.85)	2.22 (2.22)	1.31
2016	15	1.15 (2.21)	5.56 (5.56)	0.81 (1.54)	1.39
2017	1	1.06 (1.82)	2.93 (3.92)	1.79 (1.79)	1.23
2018	8	1.57 (1.65)	3.89 (3.89)	0.91 (1.89)	1.26
2019	1	1.82 (1,86)	2.42 (5,71)	1.10 (1.21)	1.14

The most outstanding communities of the travel search company have an overall score of 2.09, 2.43 and 2.19. Only four communities exists that have an overall score above the value 2. In the year 2018 there is also another community with a score of 2.11 with having the maximum leave score of 6.49. But overall the

community with id 3 in the year 2018 is the most outstanding because of having the maximum value in the centrality score. Worth to mention is the fact that the community with the highest centrality score has always the highest overall score. This implies that the centrality of a subgraph has the most impact for the score calculation.

Compared to the industry data set the overall scores of the open source project communties are except the first year always below the value 1.4. This is caused by the fact that each community is identified corresponding to the key node as maintainer of the open source project. Therefore, the communities have not that many connections among theirselves as the industry network. However, this due to the fact that the industry data set follows the pattern of having the most outstanding community in relation to the most central subgraph. In the open source data set no pattern is visible at the first look of the results. Therefore, the weight of each score is more evenly distributed. Overall with the identification of the most outstanding community through the community metrics the goal of having a detailed analysis of the code review networks is achieved.

2.6.3 Code Review Duration

This subsection handles the evolution of the code review duration over the years and is divided into two parts. The first part focusses on the median review duration of all data sets for the whole network and the change of it each year. Afterwards, the median review duration of the most outstanding communities is compared to the overall networks. Each median value is also represented in a duration per file value, due to the fact that each data set and review entry contains the involved number of files.

Code review duration of the overall networks

The median code review duration is shown in figure 2.19. The x-axis displays each year of the data set, whereas the y-axis shows the median code review durations. The scale of the duration is either in minutes, hours or days depending of the data set. Starting with the software engineering company, the median review duration takes in between 13 and 25 minutes. The first year has a median value of 23.4 minutes which increased to a value of 24.22 and afterwards constantly decreased to a value of 13.5 minutes in the year 2017. Therefore, only one increase of the duration exists and otherwise the duration gets smaller every year. The decrease could have many reasons, such as an increase of performance or better tooling set, but can not be explained through the given data. Therefore, no conclusion are drawn and the thesis only focusses on the result of having a change in the duration.

Unlike the overall median duration, the duration per file in the review process decreased constantly from an value of 1.58 in the first year to a value of 0.96 minutes per file in the year 2017. Therefore, the increase from the year 2013 to the year 2014 can be explained to some reviews containing a mass of files that increased the median duration.

In case of the travel search company data set, the median code review duration per year shows also an decrease. The duration starts at a value of 10.52 hours in 2016 over 4.23 hours in 2017 and finishes with a value of 3.35 hours in the year 2018. Therefore, the duration gets smaller every year while the networks structure gains on complexity over the same time. Worth to mention is also a drop of 6.29 hours from the first to the second year of observation. Besides, the duration per file constantly decreased with a value of 0.65 hours per file each year. Especially, the year 2016 has a much higher overall median duration value than the per file value in the upcomming years. This implies that the number of files involved in the first year were higher than in the rest.

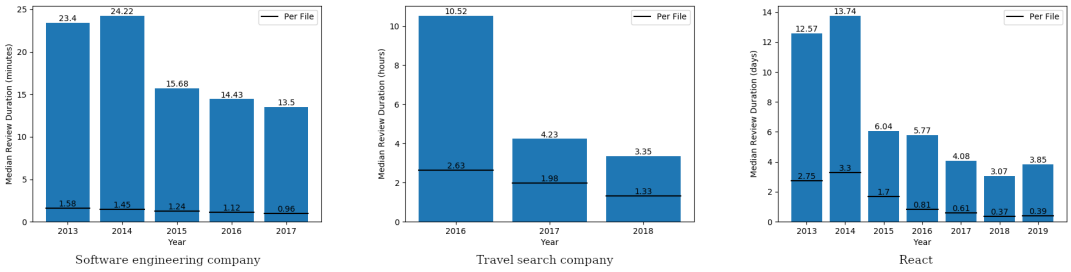

Figure 2.19: Median code review duration each year

After analysing the results of the industry data sets, the open source project shows a similiar evolution with a decrease of the median code review duration over the time. The decrease of the durations got only interrupted from the step between the years 2013 and 2014 and between the years 2018 and 2019. In 2014 the duration has the highest value by 13.74 hours, whereas the shortest duration with 3.07 hours can be found in the year 2018. As seen in the previous data sets, the react project has also a big drop of the duration with a value of 7.7 after the second year of observation. The relation between duration per file and overall duration has been about 25% in the first years whereas it decreased to 15% in the progress of the project.

Compared to the remaining data sets, the open source project took multiple days for each review, whereas the industry data set in the same scale took only multiple hours. The shortest durations are taken in the small industry data set that can be explained through the small network size. Summarizing it all, each network of every data set has in common that the overall duration of code reviews decreases

over the time. An explaination of this behaviour through all data set might be given in structural information of the networks. Therefore, the upcomming part analyses the impact of the community structure to the code review duration.

Community code review duration compared to network (travel search company)

After identfiying the most outstanding community through calculation of an overall score depending on the measuered community metrics, a comparision of the median review duration with the whole network is shown in figure 2.20. The most outstanding subgraph starts with a median review duration of 21.45 hours in the first year and decreased by 84.4% in the second year to a value of 3.04. From the year 2017 to 2018 there is also an decrease to a value of 1.73 hours. Compared to the whole network the review duration of the first year, the review process took twice as long as in the whole network. Afterwards, the performance of the outstanding community slightly faster than in the rest of the network. Therefore, the fact of having more actors and a more complex network structure has a shortening impact for the review duration in the selected communities.

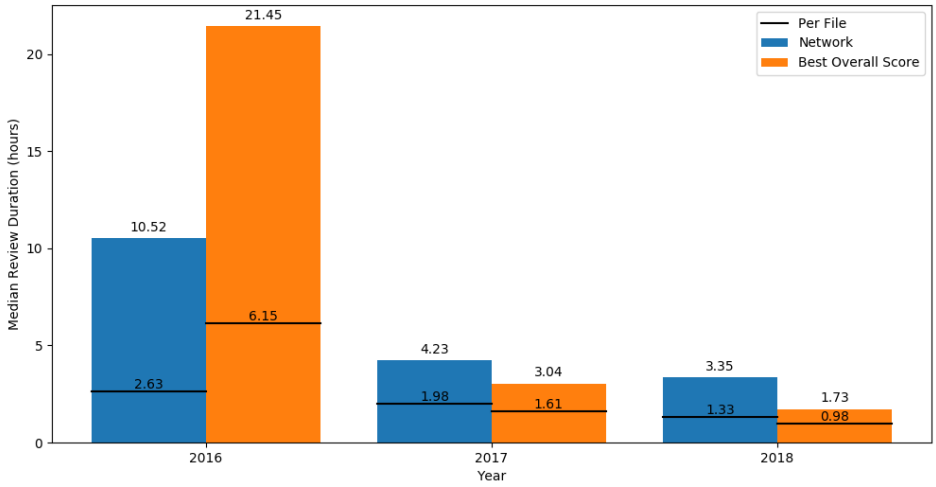

Figure 2.20: Median code review duration of most outstanding community compared to network (Travel search company)

Additionally, by analysing the ratio between review duration per file and the total review duration the most outstanding community has a bigger value than the whole network each year. This implies that even if the median review duration in the year 2016 took longer in the community, the subgraph reviewed the involved

files more precisly than the rest of the network. That leads to a positive impact of having good community structure measuered by the given metrics.

The comparision of the review duration can be broken down by comparing the individual scores to the rest of the network, as seen in figure 2.21. In the year 2016 each community independent of the score has a longer review duration. The subgraph with having the most connections to other subgraphs has additionally the lowest duration per file in the review process. The two following years have a common pattern that each community but the one with the highest internal score have a lower review duration than the whole network. The subgraph with the most internal connections, and therefore the highest internal score, has a median review duration that is multiple times higher than in the rest of the network.

Also the ratio between duration per file, compared to the overall duration, is in the best internal score community nearly the half of the duration of the whole network. This implies the fact the more connections a subgraph internaly has, the lower gets the duration of processing each individually file in the code review. Furthermore, the communities of the best centrality and leave score have in common that the review duration decreased over the years. On the other side the community with the best internal score increased over the same time.

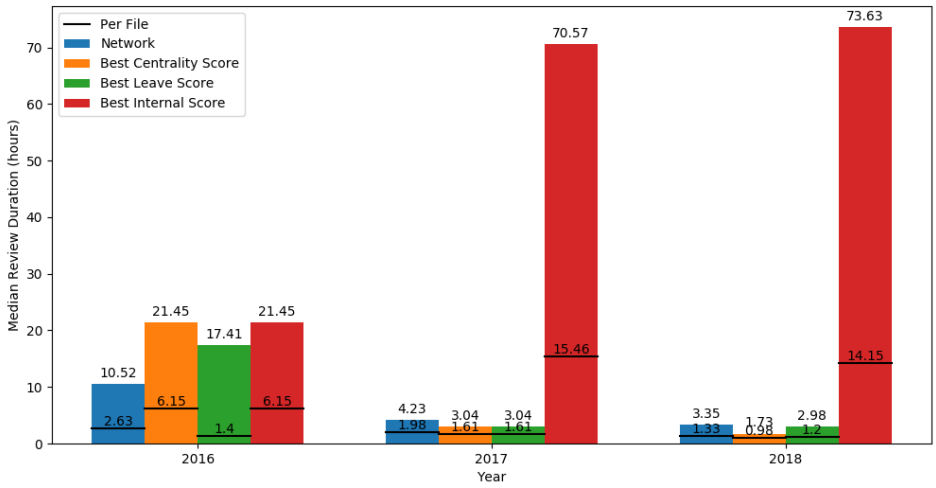

Figure 2.21: Median code review durations of communities compared to network (Travel search company)

Community code review duration compared to network (React)

Figure 2.22 shows the comparision of code review duration between the overall network and the most outstanding community. Three out of the seven years

show that duration of the review process took longer in the community than on the rest of the network. The longest duration in the community is taken in the year 2014 with an median value of 82.6 days that is 6 times longer than the overall network took. Additionally, the duration per file has a value slightly under the same and is displayed due to rounding as 82.6 days. This means that the community only did reviews involving a single file or reviews that took no time compared to the rest with more than a single file.

The second community that took longer than the rest is in the year 2017 with a median duration of 8.9 days. The year 2017 was identified as the most complex year of the network in the previous sections and seems to have impact on the duration of the code review process in the most outstanding community. The year after also has a community that took one 1 day longer in the review process than the rest of the network. Every community but one in 2018 has in common that the ratio between overall duration and duration per file is higher than in the whole network. This means each identified community either took reviews with a smaller number of files at the same average time or took more time in reviewing the same amount of files. Either way, this implies that the structural information has a impact in having a longer review time per single file.

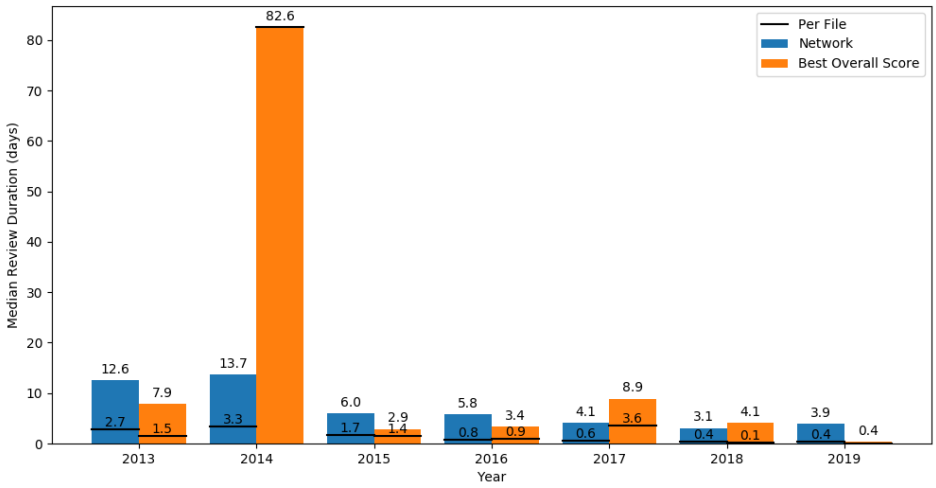

Figure 2.22: Median code review duration of most outstanding community compared to network (React)

Figure 2.23 displays the comparison of communities with the best scores in their individual cluster with the overall network. The years 2013, 2015 and 2016 have in common that each identified community has a shorter duration in the code review than the rest of the network. Every other year has a least one subgraph that took

longer in the code review process. Especially, in the years 2017 and 2018 each identified community took longer than the overall network. The longest median duration has the community with the best leave score at a duration value of 32.8 days in 2017. No impact of structural community information is recognizeable in the open source data set. Additionally, in the ratio between duration per file and the overall duration in the community no pattern can be identified.

Summarizing the evolution of the code review duration, the industry data shows more a common behaviour over the analysed years than the open source data on the community level. Comparing each data set on the whole network, both industry data sets, as well as the open source data set, show a common pattern by reducing the duration of the code review process over the time. This gives a clear hint that the gaining complexity of each network over the years has a impact in the code review process but can not cleary discovered on community level. Therefore, goal 4 of this thesis by discovering information between code review and network structure is achieved.

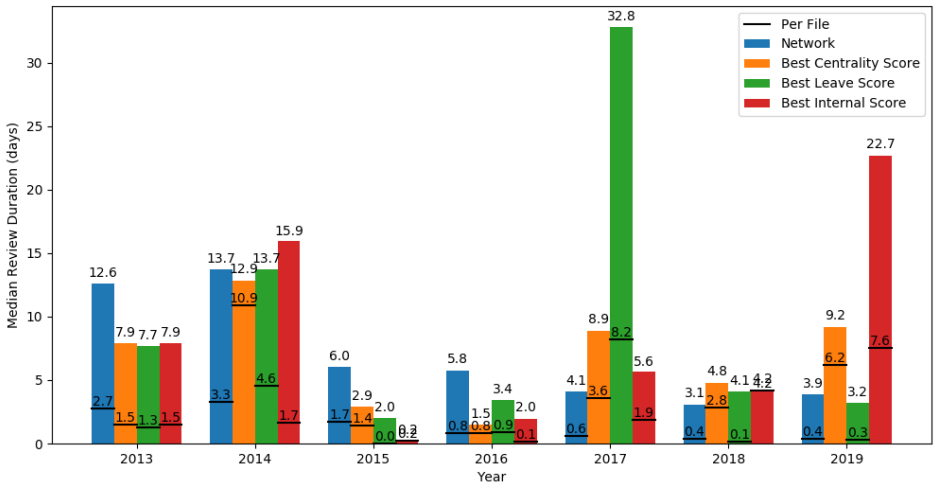

Figure 2.23: Median code review durations of communities compared to network (React)

2.7 Limitations

The data crawling and subsequently explorative analysis of code review networks in this thesis has several limitations. The following part underlines the focus and limitations of the selected data, the creation of the code review networks, the identification and scoring of communties as well as the analysis of the network

and evolution of code review duration.

Beginning with the limitations of the data sets, all discovered results of the open source data are discovered in a single data set and should be validated with more open source data. Compared to the open source data, the industry data set had two limitations in this thesis. The small industry data set did not have the size and structure for a community detection. Therefore, only a subpart of this thesis could be done with two industry data sets. As second limitation the other industry data set of the travel search company did not cover a long period of time. The data was limited to a range of 2.5 years which makes it difficult to see a pattern in the evolution of the data, as in the other data sets. Summarizing the data quality of this thesis, the discovered results are only valid for the inspected data sets and can not be generalized without any further validation on other data of different open source or industry projects.

Secondly, the creation of the network in this research relies on a 1-to-1 relationship between author and reviewer in the code review process. On the other hand the structure of the open source project supports a 1-to-n relationship in the process, by having a single person with a code commit and multiple persons interacting on it. In order to align with the industry data each interaction was handled as a 1-to-1 code review process. For further research the network creation should be extended to interpret also information such as the relationships between authors and reviewers.

Continuing with the creation of the network, the meta information of involved files in the code review process that were collected inside the review entity may falsify the calculated results. By working with central code repositories and version control, every change like adding whitespaces lead to a change in the file and increases the involved files count. This can lead to the fact that the calculation of review duration per file is possibly falsified by code commits that do not change the semantic content of the code. Another approach for getting more clearly results is the use of changes in the code instead of the simple file count.

The next limitation consists in the discovery of the communities by applying a single algorithm choosen by the performance and methodology. For a more detailed and valid result, different algorithm for detecting communities should be used. In case of scoring each identified community metrics were clusterd on a top level and do have different targets in analysing the network structure. The results could be more clearler with clustering them into smaller groups or analysing each individual for getting a more clearer impact of structural information on the code review process.

Summarizing the limitations, this thesis only focusses on a very specific creation of code review networks and aligning the given data into it. From the given

data over the creation, the community detection and scoring to the concrete analysis of the whole network or parts of it each step can be extended to gain more valid results and more generalized implications for code review networks. Nevertheless, this thesis gives a starting point in the explorative analysis of code review networks by comparing structural with semantic information.

2.8 Conclusion

The explorative analysis of code review networks give hints about many undiscovered facts for a better understanding and improving the code review process by analysing the structural information of a code review network. The thesis showed a method for crawling open source data and a way for creating code review networks from heterogenous data sets. Apart from this, a network analysis model was introduced to calculate metrics on different levels on the network for comparision.

The thesis issued the research questions what structural network information contains a code review network and secondly if structural network information do have an impact on the code review duration. As main point discovered in this thesis, the code review networks of the industry data do increase over the analysed years. In comparision, the crawled open source data gains on complexity till a specific year and decreased afterwards again. Overall, the networks of the open source project has a quite different structure compared to the industry that can be explained through the open source design. The network structure of the open source data is similiar to a star-shape connected via single key nodes, whereas the industry data is connected via bigger subgraphs in the center and does have more individual connections on the outer side.

The main structural information given through the networks is that the average closeness and degree centrality correlates with the complexity of the networks. With an increase of complexity by having more actors and edges, the centrality values decrease over all data sets. The evolution of the betweenness centrality depends on the size of the network. In the small industry data set a constant growth was found, whereas the bigger industry and open source data sinks over the analysed time. Furthermore, the growth of the open source network can be calculated via a direct correlation between actors and edges.

In case of communities, a growth in the number of discovered communities was identified accross the bigger data sets. The detected communities vary in the structure. In the open source data nodes are combined to a community by having the same key node that interacts as connection to the rest of the network. In the industry data set the communities are way more connected internally as

discovered by different community metrics. Especially, the size of the discovered subgraphs seems to be an important factor for the resulting community metrics. To summarize it, the code review networks do share same behaviour in terms of overall network centralities, but also differ in case of communties and connection or layout structure.

To answer the second research question, the code review duration was compared among each data set in the overall network and secondly with the most outstanding communities by the calculated community metrics. The analysis showed that by growing complexity of the overall network the median code review duration was reduced over all given data sets. This gives a clear hint that the network structure and especially the complexity has a clear impact on the code review duration. In case of communities, the industry data set gave some hints underlying the impact, but as main fact the results show that the identified communities take longer for each individual file, compared to the rest of the network. The open source data communities did not show any pattern that confirmed the assumptions.

Overall, the thesis presents a comparision of explorative analysed structural information to semantic information in terms of code review duration. The research is limited in several ways as described in the previous section. Therefore, by extending the network model and validating the findings on more different data sets can be done in future work. Also, the research approach can be extended by applying different community detection algorithms and comparing each community metric individually with semantic information of a code review process. Additionally, in future work more different semantic information like identifying key reviewer or key authors can be compared to structural information.

Appendix A Network Analysis Data (Software Engineering Company)

	2013	2014	2015	2016	2017
Actors	13	15	16	21	24
Edges	70	88	89	137	165
Interactions	1205	1822	1821	2431	2783
Number of Communities	1	1	1	1	1
Cetrality (Betweenness)					
- max	0.015	0.027	0.160	0.093	0.111
- mean	0.009	0.012	0.019	0.018	0.019
- median	0.009	0.011	0.008	0.010	0.006
- min	0.001	0	0	0	0
- std	0.005	0.008	0.038	0.024	0.029
Centrality (Closeness)					
- max	1.000	1.000	0.938	0.952	0.920
- mean	0.913	0.871	0.812	0.762	0.726
- median	0.923	0.875	0.858	0.769	0.719
- min	0.750	0.667	0.500	0.526	0.460
- std	0.071	0.092	0.120	0.119	0.128
Centrality (Degree)					
- max	1.000	1.000	0.933	0.950	0.913
- mean	0.897	0.838	0.742	0.652	0.598
- median	0.917	0.857	0.833	0.700	0.609
- min	0.667	0.500	0.067	0.100	0.043
- std	0.093	0.137	0.233	0.229	0.243
Avg. ODF					
- max	0	0	0	0	0
- mean	0	0	0	0	0
- min	0	0	0	0	0
- std	0	0	0	0	0
Avg. internal degree					
- max	10.77	11.73	11.13	13.05	13.75
- mean	10.77	11.73	11.13	13.05	13.75
- min	10.77	11.73	11.13	13.05	13.75
- std	0.00	0.00	0.00	0.00	0.00
Conductance					
- max	0	0	0	0	0
- mean	0	0	0	0	0
- min	0	0	0	0	0
- std	0	0	0	0	0
Cut-ratio					
- max	0	0	0	0	0
- mean	0	0	0	0	0
- min	0	0	0	0	0
- std	0	0	0	0	0

	2013	2014	2015	2016	2017
Degree					
- max	12	14	14	19	21
- mean	10.77	11.73	11.13	13.05	13.75
- median	11	12	12.5	14	14
- min	8	7	1	2	1
- std	1.12	1.91	3.50	4.57	5.58
Egdes inside (Communities)					
- max	70	88	89	137	165
- mean	70.00	88.00	89.00	137.00	165.00
- min	70	88	89	137	165
- std	0.00	0.00	0.00	0.00	0.00
Expansion					
- max	0	0	0	0	0
- mean	0	0	0	0	0
- min	0	0	0	0	0
- std	0	0	0	0	0
Flake ODF					
- max	0	0	0	0	0
- mean	0	0	0	0	0
- min	0	0	0	0	0
- std	0	0	0	0	0
Fraction over median degree					
- max	0.231	0.400	0.500	0.429	0.458
- mean	0.231	0.400	0.500	0.429	0.458
- min	0.231	0.400	0.500	0.429	0.458
- std	0	0	0	0	0
Interactions per node					
- max	357	488	531	546	611
- mean	185.38	242.93	227.63	231.52	231.92
- median	180	243	253.5	239	241.5
- min	30	15	1	4	1
- std	103.21	138.84	136.32	152.19	156.71
Internal edge density					
- max	0.224	0.210	0.185	0.163	0.149
- mean	0.224	0.210	0.185	0.163	0.149
- min	0.224	0.210	0.185	0.163	0.149
- std	0	0	0	0	0
ODF					
- max	0.00	0.00	0.00	0.00	0.00
- mean	0.00	0.00	0.00	0.00	0.00
- min	0.00	0.00	0.00	0.00	0.00
- std	0.00	0.00	0.00	0.00	0.00
Normalized cut-ratio					
- max	0.00	0.00	0.00	0.00	0.00
- mean	0.00	0.00	0.00	0.00	0.00
- min	0.00	0.00	0.00	0.00	0.00
- std	0.00	0.00	0.00	0.00	0.00

	2013	2014	2015	2016	2017
Size (Community)					
- max	13	15	16	21	24
- mean	13	15	16	21	24
- min	13	15	16	21	24
- std	0	0	0	0	0
Triangle Participation					
- max	1.000	1.000	1.000	1.000	1.000
- mean	1.000	1.000	0.938	1.000	0.917
- min	1.000	1.000	0.938	1.000	0.917
- std	0	0	0	0	0

Appendix B Network Analysis Data (Travel Search Company)

	2016	2017	2018
Actors	284	455	456
Edges	2201	4053	3097
Interactions	69998	107130	65582
Number of Communities	12	17	24
Cetrality (Betweenness)			
- max	0.264	0.160	0.188
- mean	0.007	0.004	0.005
- median	0.001	0.001	0.001
- min	0	0	0
- std	0.021	0.012	0.014
Centrality (Closeness)			
- max	0.571	0.513	0.448
- mean	0.351	0.333	0.299
- median	0.357	0.334	0.294
- min	0.004	0.002	0.166
- std	0.069	0.065	0.051
Centrality (Degree)			
- max	0.399	0.315	0.156
- mean	0.055	0.039	0.030
- median	0.035	0.026	0.022
- min	0.004	0.002	0.002
- std	0.055	0.040	0.028
Avg. ODF			
- max	4.27	9.92	11.27
- mean	1.88	3.16	3.07
- min	0.00	0.00	0.29
- std	1.48	3.03	2.83
Avg. internal degree			
- max	22.26	28.93	23.32
- mean	6.16	7.13	6.40
- min	1.00	1.00	1.00
- std	5.35	6.79	4.37
Conductance			
- max	0.5526	0.6000	0.5930
- mean	0.2419	0.2775	0.3113
- min	00	00	0.0311
- std	0.1766	0.2134	0.1663
Cut-ratio			
- max	0.0157	0.0264	0.0255
- mean	0.0073	0.0076	0.0071
- min	00	00	06
- std	0.0054	0.0076	0.0066

	2016	2017	2018
Degree			
- max	113	143	71
- mean	15.50	17.82	13.58
- median	10	12	10
- min	1	1	1
- std	15.43	18.28	12.73
Egdes inside (Communities)			
- max	1536	1244	898
- mean	164.50	178.06	94.38
- min	1	1	1
- std	415.48	308.26	177.84
Expansion			
- max	4.273	9.924	11.267
- mean	1.883	3.165	3.074
- min	0	0	0.286
- std	1.479	3.034	2.831
Flake ODF			
- max	0.444	0.500	0.500
- mean	0.110	0.145	0.152
- min	0	0	0
- std	0.160	0.185	0.172
Fraction over median degree			
- max	0.500	0.500	0.500
- mean	0.305	0.369	0.399
- min	0	0	0
- std	0.164	0.179	0.122
Interactions per node			
- max	4724	5827	1838
- mean	492.94	470.90	287.64
- median	204.5	248.0	180.0
- min	1	1	1
- std	653.40	648.79	321.97
Internal edge density			
- max	0.250	0.250	0.250
- mean	0.116	0.121	0.116
- min	0.041	0.037	0.048
- std	0.056	0.070	0.047
ODF			
- max	26	100	60
- mean	8.33	17.47	15.08
- min	0	0	1
- std	6.612	23.472	14.944
Normalized cut-ratio			
- max	0.562	0.601	0.620
- mean	0.257	0.294	0.323
- min	0	0	0.033
- std	0.172	0.219	0.170

	2016	2017	2018
Size (Community)			
- max	138	86	77
- mean	23.67	26.76	19.00
- min	2	2	2
- std	35.67	27.42	16.41
Triangle Participation			
- max	1	1	1
- mean	0.786	0.660	0.822
- min	0	0	0
- std	0.278	0.374	0.260

Appendix C Network Analysis Data (React)

	2013	2014	2015	2016	2017	2018	2019
Actors	45	130	209	288	383	275	191
Edges	92	215	317	480	558	413	286
Interactions	591	865	1146	3738	6435	4113	3155
Number of Communities	4	21	27	35	42	37	29
Centrality (Betweenness)							
- max	0.468	0.560	0.648	0.409	0.525	0.525	0.251
- mean	0.034	0.012	0.008	0.006	0.005	0.006	0.010
- median	0.0	0.0	0.0	0.0	0.0	0.0	0.0
- min	0	0	0	0	0	0	0
- std	0.088	0.061	0.051	0.034	0.034	0.036	0.035
Centrality (Closeness)							
- max	0.667	0.638	0.650	0.561	0.587	0.568	0.473
- mean	0.425	0.379	0.362	0.324	0.334	0.296	0.273
- median	0.411	0.388	0.392	0.338	0.339	0.306	0.297
- min	0.282	0.008	0.005	0.003	0.003	0.004	0.005
- std	0.082	0.074	0.075	0.086	0.067	0.098	0.090
Centrality (Degree)							
- max	0.568	0.504	0.534	0.345	0.387	0.416	0.184
- mean	0.093	0.026	0.015	0.012	0.008	0.011	0.016
- median	0.045	0.008	0.005	0.003	0.003	0.004	0.005
- min	0.023	0.008	0.005	0.003	0.003	0.004	0.005
- std	0.118	0.060	0.045	0.033	0.026	0.031	0.030
Avg. ODF							
- max	0.500	2.714	2.250	4.000	2.417	2.636	3.750
- mean	0.394	1.318	1.085	1.255	0.979	0.928	0.947
- min	0.077	0	0	0	0	0	0
- std	0.183	0.679	0.627	1.075	0.674	0.739	0.930
Avg. internal degree							
- max	4.41	2.00	2.00	2.12	2.14	2.00	2.36
- mean	1.85	1.42	1.42	1.46	1.41	1.45	1.52
- min	1.00	1.00	1.00	1.00	1.00	1.00	1.00
- std	1.48	0.37	0.35	0.38	0.40	0.38	0.42
Conductance							
- max	0.333	0.613	0.667	0.800	0.667	0.600	0.714
- mean	0.254	0.446	0.393	0.376	0.364	0.315	0.301
- min	0.017	0	0	0	0	0	0
- std	0.137	0.133	0.161	0.212	0.171	0.208	0.199
Cut-ratio							
- max	0.013	0.022	0.011	0.014	0.007	0.010	0.020
- mean	0.012	0.011	0.005	0.005	0.003	0.004	0.005
- min	0.012	0	0	0	0	0	0
- std	0.001	0.006	0.003	0.004	0.002	0.003	0.005

	2013	2014	2015	2016	2017	2018	2019
Degree							
- max	25	65	111	99	148	114	35
- mean	4.09	3.31	3.03	3.33	2.91	3.00	2.99
- median	2	1	1	1	1	1	1
- min	1	1	1	1	1	1	1
- std	5.18	7.71	9.34	9.39	10.00	8.50	5.67
Egdes inside (Communities)							
- max	86	27	49	46	64	53	23
- mean	22.25	5.29	6.81	7.43	8.26	6.54	5.93
- min	1	1	1	1	1	1	1
- std	36.81	6.56	10.72	10.44	13.91	9.55	5.88
Expansion							
- max	0.500	2.714	2.250	4.000	2.417	2.636	3.750
- mean	0.394	1.318	1.085	1.255	0.979	0.928	0.947
- min	0.077	0	0	0	0	0	0
- std	0.183	0.679	0.627	1.075	0.674	0.739	0.930
Flake ODF							
- max	0	0.714	0.500	0.750	0.750	0.500	0.750
- mean	0	0.192	0.221	0.172	0.151	0.135	0.084
- min	0	0	0	0	0	0	0
- std	0	0.200	0.216	0.214	0.211	0.166	0.162
Fraction over median degree							
- max	0.487	0.500	0.500	0.500	0.500	0.500	0.400
- mean	0.122	0.167	0.200	0.186	0.135	0.172	0.181
- min	0	0	0	0	0	0	0
- std	0.211	0.178	0.191	0.159	0.157	0.172	0.142
Interactions per node							
- max	251	363	519	1745	3239	1791	971
- mean	26.27	13.31	10.97	25.96	33.60	29.91	33.04
- median	5	2	2	2	3	3	2
- min	1	1	1	1	1	1	1
- std	49.87	44.92	50.66	129.61	202.75	160.40	135.12
Internal edge density							
- max	0.250	0.250	0.250	0.250	0.250	0.250	0.250
- mean	0.195	0.151	0.147	0.141	0.152	0.141	0.135
- min	0.029	0.018	0.010	0.011	0.008	0.010	0.026
- std	0.096	0.086	0.085	0.086	0.094	0.089	0.083
ODF							
- max	2.00	38.00	62.00	53.00	84.00	62.00	21.00
- mean	1.25	7.29	6.85	8.74	7.43	6.73	5.83
- min	1.00	0.00	0.00	0.00	0.00	0.00	0.00
- std	0.43	9.94	12.95	13.45	14.25	11.42	6.88
Normalized cut-ratio							
- max	0.339	0.656	0.673	0.808	0.670	0.604	0.740
- mean	0.308	0.469	0.409	0.389	0.374	0.327	0.314
- min	0.217	0	0	0	0	0	0
- std	0.053	0.143	0.167	0.219	0.174	0.215	0.209

	2013	2014	2015	2016	2017	2018	2019
Size (Community)							
- max	39	28	50	47	65	53	20
- mean	11.25	6.19	7.74	8.23	9.12	7.43	6.59
- min	2	2	2	2	2	2	2
- std	16.02	6.51	10.62	10.22	13.79	9.37	5.32
Triangle Participation							
- max	0.615	0.300	0.214	0.375	0.250	0.273	0.455
- mean	0.154	0.014	0.012	0.034	0.014	0.009	0.033
- min	0	0	0	0	0	0	0
- std	0.266	0.064	0.045	0.084	0.051	0.045	0.103

Appendix D Community Network Analysis Data (Travel Search Company)

Yearid	Actors	Edges	Interactions	Group Centrality			Avg. internal degree	Conductance	Cut-Ratio	Expansion	Fraction over median degree
				Betweenness	Closeness	Degree					
2016 1	138	1536	46489	0.641	0.57	0.336	22.26	0.057	0.009	1.355	0.493
2	36	148	5126	0.003	0.434	0.065	8.22	0.063	0.002	0.556	0.5
3	27	82	4841	0.056	0.443	0.105	6.07	0.226	0.007	1.778	0.296
4	18	65	774	0.001	0.416	0.064	7.22	0.193	0.006	1.722	0.333
5	12	49	4586	0.011	0.414	0.051	8.17	0.169	0.006	1.667	0.5
6	11	25	3314	0.02	0.447	0.07	4.55	0.485	0.016	4.273	0.273
7	10	17	214	0.014	0.45	0.106	3.4	0.553	0.015	4.2	0.5
8	9	20	251	0	0.397	0.047	4.44	0.487	0.015	4.222	0.333
9	8	16	1174	0.002	0.427	0.043	4	0.319	0.007	1.875	0.125
10	7	9	168	0	0.372	0.007	2.57	0.1	0.001	0.286	0.143
11	6	6	101	0	0.294	0.011	2	0.25	0.002	0.667	0.167
12	2	1	5	0	0	0	1	0	0	0	0
2017 1	86	1244	37934	0.108	0.556	0.336	28.93	0.18	0.017	6.36	0.488
2	79	552	12599	0.47	0.6	0.439	13.97	0.415	0.026	9.924	0.468
3	63	419	11923	0.009	0.452	0.11	13.3	0.083	0.003	1.206	0.492
4	60	260	9122	0.002	0.431	0.084	8.67	0.132	0.003	1.317	0.483
5	37	164	5031	0.041	0.452	0.112	8.86	0.21	0.006	2.351	0.459
6	32	128	7634	0	0.424	0.054	8	0.138	0.003	1.281	0.5
7	21	50	1796	0.016	0.439	0.118	4.76	0.554	0.014	5.905	0.476
8	20	104	3855	0.02	0.458	0.117	10.4	0.388	0.015	6.6	0.45
9	12	29	2421	0	0.371	0.014	4.83	0.094	0.001	0.5	0.5
10	12	24	356	0.036	0.438	0.084	4	0.575	0.012	5.417	0.5
11	11	28	2555	0.049	0.461	0.101	5.09	0.591	0.017	7.364	0.364
12	7	13	788	0.015	0.408	0.045	3.71	0.49	0.008	3.571	0.429
13	6	7	101	0	0.272	0.002	2.33	0.067	0	0.167	0.333
14	3	2	13	0	0.288	0.002	1.33	0.2	0.001	0.333	0.333
15	2	1	10	0	0	0	1	0	0	0	0
16	2	1	4	0	0.319	0.004	1	0.6	0.003	1.5	0
17	2	1	4	0	0	0	1	0	0	0	0
2018 1	77	898	16887	0.127	0.456	0.185	23.32	0.149	0.011	4.091	0.481
2	44	221	4107	0.065	0.391	0.1	10.05	0.193	0.006	2.409	0.477
3	42	218	3235	0.425	0.532	0.304	10.38	0.496	0.025	10.214	0.5
4	33	101	2975	0.09	0.396	0.087	6.12	0.235	0.004	1.879	0.394
5	29	140	4208	0.001	0.317	0.016	9.66	0.031	0.001	0.31	0.483
6	23	95	5101	0.001	0.34	0.035	8.26	0.136	0.003	1.304	0.435
7	22	81	1663	0.001	0.362	0.032	7.36	0.11	0.002	0.909	0.5
8	20	46	620	0.01	0.423	0.11	4.6	0.456	0.009	3.85	0.25
9	19	66	3443	0.027	0.421	0.085	6.95	0.337	0.008	3.526	0.316
10	19	81	2539	0.035	0.419	0.089	8.53	0.293	0.008	3.526	0.421
11	16	52	3381	0.03	0.367	0.052	6.5	0.257	0.005	2.25	0.5
12	15	42	3046	0	0.34	0.034	5.6	0.184	0.003	1.267	0.467
13	15	58	1913	0.217	0.464	0.204	7.73	0.593	0.026	11.267	0.4
14	13	34	707	0.029	0.418	0.111	5.23	0.544	0.014	6.231	0.462
15	11	27	1868	0.013	0.339	0.049	4.91	0.419	0.008	3.545	0.455
16	11	22	167	0.004	0.348	0.047	4	0.405	0.006	2.727	0.364
17	10	30	1641	0.006	0.39	0.076	6	0.504	0.014	6.1	0.5
18	7	10	192	0	0.318	0.004	2.86	0.091	0.001	0.286	0.286
19	7	11	145	0.001	0.366	0.042	3.14	0.522	0.008	3.429	0.429
20	6	13	414	0	0.351	0.018	4.33	0.235	0.003	1.333	0.5
21	6	8	903	0	0.264	0.004	2.67	0.2	0.001	0.667	0.5
22	5	7	562	0	0.215	0.002	2.8	0.125	0.001	0.4	0.2
23	4	3	57	0	0.324	0.011	1.5	0.455	0.003	1.25	0.25
24	2	1	28	0	0.298	0.004	1	0.5	0.002	1	0

Year	id	Internal edge density	Normalized Cut-Ratio	Max ODF	Avg ODF	Flake ODF	Triangle participation	Score			
								Internal	Leave	Centrality	Sum
2016	1	0.041	0.181	26	1.355	0	0.949	3.71	1.24	2.72	2.09
	2	0.059	0.068	5	0.556	0	0.944	2.40	0.45	0.87	0.87
	3	0.058	0.238	11	1.778	0.074	0.852	1.78	1.60	1.07	0.91
	4	0.106	0.200	6	1.722	0	0.944	2.20	1.25	0.84	0.92
	5	0.186	0.174	9	1.667	0	1.000	2.99	1.18	0.82	1.16
	6	0.114	0.495	11	4.273	0.273	0.727	1.78	3.61	0.93	1.17
	7	0.094	0.562	14	4.200	0.400	0.800	2.17	4.09	1.01	1.40
	8	0.139	0.496	8	4.222	0.444	1.000	2.28	3.91	0.77	1.33
	9	0.143	0.323	5	1.875	0.125	1.000	1.86	1.94	0.81	0.90
	10	0.107	0.100	2	0.286	0	0.714	1.39	0.44	0.64	0.45
	11	0.100	0.251	3	0.667	0	0.500	1.17	1.06	0.51	0.43
	12	0.250	0	0	0	0	0	1.00	0	0	0.02
2017	1	0.085	0.269	43	6.360	0.012	0.977	3.99	2.97	1.86	2.14
	2	0.045	0.516	100	9.924	0.127	0.899	2.70	5.54	2.73	2.43
	3	0.054	0.093	22	1.206	0	0.968	2.74	0.84	1.02	1.09
	4	0.037	0.142	8	1.317	0	0.850	2.26	0.89	0.91	0.90
	5	0.062	0.221	13	2.351	0.054	0.919	2.34	1.57	1.07	1.10
	6	0.065	0.143	8	1.281	0.031	0.875	2.34	0.95	0.83	0.91
	7	0.060	0.569	16	5.905	0.429	0.810	2.04	4.42	1.03	1.42
	8	0.137	0.405	24	6.600	0.150	1.000	2.77	3.59	1.06	1.54
	9	0.110	0.094	3	0.500	0	0.750	2.25	0.47	0.65	0.74
	10	0.091	0.583	21	5.417	0.417	0.833	2.21	4.37	0.98	1.45
	11	0.127	0.601	21	7.364	0.455	0.818	2.13	5.01	1.08	1.56
	12	0.155	0.493	14	3.571	0.286	0.857	2.37	3.26	0.80	1.27
	13	0.117	0.067	1	0.167	0	0.667	1.76	0.27	0.46	0.48
	14	0.167	0.200	1	0.333	0	0	1.29	0.75	0.48	0.41
	15	0.250	0	0	0	0	0	1.00	0	0	0.02
	16	0.250	0.600	2	1.500	0.500	0	1.00	3.38	0.54	0.74
	17	0.250	0	0	0	0	0	1.00	0	0	0.02
2018	1	0.077	0.216	22	4.091	0.026	0.961	3.49	2.00	1.38	1.65
	2	0.058	0.212	28	2.409	0	0.932	2.46	1.59	0.98	1.10
	3	0.063	0.565	48	10.214	0.262	0.929	2.53	5.49	2.25	2.19
	4	0.048	0.245	13	1.879	0	0.879	1.97	1.42	1.00	0.92
	5	0.086	0.033	2	0.310	0	0.966	2.56	0.21	0.57	0.77
	6	0.094	0.141	6	1.304	0	0.957	2.42	0.86	0.65	0.85
	7	0.088	0.113	5	0.909	0	0.864	2.38	0.66	0.68	0.82
	8	0.061	0.468	19	3.850	0.250	0.850	1.62	3.22	0.97	1.08
	9	0.096	0.348	13	3.526	0.053	0.947	2.11	2.29	0.94	1.08
	10	0.118	0.304	15	3.526	0.105	0.895	2.44	2.27	0.96	1.19
	11	0.108	0.263	21	2.250	0.063	0.875	2.44	1.78	0.78	1.05
	12	0.100	0.188	4	1.267	0	0.733	2.15	0.98	0.64	0.79
	13	0.138	0.620	60	11.267	0.467	1.000	2.55	6.49	1.58	2.11
	14	0.109	0.557	35	6.231	0.308	1.000	2.43	4.41	0.99	1.53
	15	0.123	0.426	14	3.545	0.273	0.909	2.38	3.00	0.70	1.19
	16	0.100	0.410	12	2.727	0.273	0.818	1.96	2.72	0.69	1.01
	17	0.167	0.514	23	6.100	0.400	1.000	2.81	4.30	0.83	1.58
	18	0.119	0.091	1	0.286	0	0.714	1.74	0.38	0.54	0.52
	19	0.131	0.526	8	3.429	0.429	0.857	2.24	3.55	0.71	1.23
	20	0.217	0.237	3	1.333	0	1.000	2.97	1.15	0.63	1.07
	21	0.133	0.201	2	0.667	0	0.833	2.35	0.85	0.45	0.76
	22	0.175	0.125	1	0.400	0	0.800	1.92	0.53	0.36	0.54
	23	0.125	0.455	5	1.250	0.250	0	0.93	2.37	0.56	0.57
	24	0.250	0.500	2	1.000	0.500	0	1.00	2.92	0.51	0.66

Appendix E Community Network Analysis Data (React)

Yearid	Actors	Edges	Inter-actions	Group Centrality			Avg. internal degree	Conduct-ance	Cut-Ratio	Expan-sion	Fraction over median degree
				Between-ness	Close-ness	Degree					
2013 1	39	86	581	0.8	0.667	0.5	4.41	0.017	0.013	0.077	0.487
2	2	1	2	0	0.406	0.023	1	0.333	0.012	0.5	0
3	2	1	1	0	0.384	0.023	1	0.333	0.012	0.5	0
4	2	1	1	0	0.384	0.023	1	0.333	0.012	0.5	0
2014 1	28	27	41	0.305	0.604	0.373	1.93	0.413	0.013	1.357	0.036
2	20	19	39	0.217	0.585	0.3	1.9	0.493	0.017	1.85	0.05
3	11	10	29	0.04	0.515	0.126	1.82	0.524	0.017	2	0.091
4	10	10	16	0.027	0.506	0.092	2	0.474	0.015	1.8	0.3
5	9	9	19	0.007	0.511	0.099	2	0.514	0.017	2.111	0.444
6	7	6	13	0.02	0.498	0.073	1.71	0.613	0.022	2.714	0.143
7	6	5	14	0.001	0.473	0.032	1.67	0.474	0.012	1.5	0.333
8	5	4	25	0.009	0.488	0.056	1.6	0.5	0.013	1.6	0.2
9	4	3	4	0.004	0.481	0.04	1.5	0.6	0.018	2.25	0.5
10	4	3	5	0.002	0.481	0.04	1.5	0.5	0.012	1.5	0.25
11	4	3	6	0	0.472	0.04	1.5	0.5	0.012	1.5	0.5
12	3	2	4	0	0.464	0.024	1.33	0.5	0.01	1.333	0.333
13	3	2	2	0	0.393	0.016	1.33	0.333	0.005	0.667	0.333
14	2	1	1	0	0.421	0.016	1	0.5	0.008	1	0
15	2	1	1	0	0.401	0.008	1	0.333	0.004	0.5	0
16	2	1	1	0	0	0	1	0	0	0	0
17	2	1	2	0	0.465	0.023	1	0.6	0.012	1.5	0
18	2	1	1	0	0.388	0.008	1	0.333	0.004	0.5	0
19	2	1	1	0	0.401	0.008	1	0.333	0.004	0.5	0
20	2	1	1	0	0.454	0.016	1	0.5	0.008	1	0
21	2	1	1	0	0.401	0.008	1	0.333	0.004	0.5	0
2015 1	50	49	98	0.413	0.614	0.39	1.96	0.388	0.008	1.24	0.02
2	27	27	95	0.088	0.513	0.148	2	0.386	0.007	1.259	0.111
3	25	24	40	0.109	0.52	0.163	1.92	0.5	0.01	1.92	0.04
4	17	16	22	0.014	0.485	0.063	1.88	0.36	0.006	1.059	0.059
5	14	14	26	0.002	0.48	0.041	2	0.417	0.007	1.429	0.286
6	9	8	11	0	0.462	0.02	1.78	0.36	0.005	1	0.444
7	6	5	16	0	0.457	0.02	1.67	0.474	0.007	1.5	0.333
8	5	4	8	0	0.41	0.005	1.6	0.111	0.001	0.2	0.4
9	5	4	4	0	0.445	0.02	1.6	0.429	0.006	1.2	0.2
10	4	3	3	0.001	0.458	0.02	1.5	0.5	0.007	1.5	0.5
11	4	3	4	0.001	0.469	0.024	1.5	0.571	0.01	2	0.25
12	4	3	4	0.003	0.461	0.029	1.5	0.571	0.01	2	0.5
13	4	3	8	0	0.439	0.01	1.5	0.25	0.002	0.5	0.25
14	4	3	7	0	0.452	0.015	1.5	0.538	0.009	1.75	0.5
15	4	3	4	0.002	0.448	0.024	1.5	0.6	0.011	2.25	0.5
16	3	2	3	0	0.438	0.01	1.33	0.333	0.003	0.667	0.333
17	3	2	2	0	0.431	0.01	1.33	0.333	0.003	0.667	0.333
18	3	2	2	0	0.423	0.01	1.33	0.333	0.003	0.667	0.333
19	2	1	2	0	0.407	0.005	1	0.333	0.002	0.5	0
20	2	1	1	0	0.429	0.01	1	0.5	0.005	1	0
21	2	1	1	0	0	0	1	0	0	0	0
22	2	1	1	0	0.468	0.019	1	0.667	0.01	2	0
23	2	1	2	0	0.439	0.01	1	0.5	0.005	1	0
24	2	1	2	0	0.407	0.005	1	0.333	0.002	0.5	0
25	2	1	1	0	0.407	0.005	1	0.333	0.002	0.5	0
26	2	1	1	0	0.39	0.01	1	0.5	0.005	1	0
27	2	1	1	0	0	0	1	0	0	0	0

Year	id	Actors	Edges	Inter-actions	Group Centrality			Avg. internal degree	Conduct-ance	Cut-Ratio	Expan-sion	Fraction over median degree
2016	1	47	46	136	0.184	0.578	0.22	1.96	0.366	0.005	1.128	0.021
	2	29	29	58	0.093	0.541	0.158	2	0.448	0.006	1.621	0.103
	3	27	26	48	0.077	0.512	0.107	1.93	0.422	0.005	1.407	0.037
	4	27	27	46	0.105	0.554	0.176	2	0.491	0.007	1.926	0.111
	5	22	22	49	0.053	0.526	0.105	2	0.47	0.007	1.773	0.136
	6	17	18	58	0.019	0.515	0.074	2.12	0.544	0.009	2.529	0.235
	7	14	14	37	0.002	0.496	0.044	2	0.509	0.008	2.071	0.357
	8	13	12	49	0.005	0.503	0.044	1.85	0.5	0.007	1.846	0.231
	9	8	8	48	0.004	0.471	0.032	2	0.407	0.005	1.375	0.25
	10	6	5	6	0.005	0.483	0.025	1.67	0.474	0.005	1.5	0.333
	11	6	5	10	0	0.444	0.011	1.67	0.375	0.004	1	0.333
	12	5	4	24	0.006	0.487	0.032	1.6	0.692	0.013	3.6	0.4
	13	5	4	13	0	0.483	0.021	1.6	0.429	0.004	1.2	0.4
	14	5	4	10	0	0.393	0.007	1.6	0.273	0.002	0.6	0.4
	15	4	3	15	0.001	0.491	0.035	1.5	0.727	0.014	4	0.25
	16	4	3	4	0	0.417	0.018	1.5	0.455	0.004	1.25	0.25
	17	4	3	5	0.001	0.469	0.032	1.5	0.647	0.01	2.75	0.25
	18	4	3	4	0	0.372	0.007	1.5	0.333	0.003	0.75	0.5
	19	4	3	3	0	0.394	0.011	1.5	0.333	0.003	0.75	0.25
	20	3	2	3	0	0	0	1.33	0	0	0	0.333
	21	3	2	3	0	0.419	0.007	1.33	0.333	0.002	0.667	0.333
	22	3	2	2	0	0.449	0.014	1.33	0.5	0.005	1.333	0.333
	23	3	2	3	0	0.391	0.004	1.33	0.2	0.001	0.333	0.333
	24	3	2	3	0	0	0	1.33	0	0	0	0.333
	25	2	1	4	0.001	0.49	0.028	1	0.8	0.014	4	0
	26	2	1	1	0	0.355	0.003	1	0.333	0.002	0.5	0
	27	2	1	2	0	0.355	0.003	1	0.333	0.002	0.5	0
	28	2	1	1	0	0	0	1	0	0	0	0
	29	2	1	1	0	0	0	1	0	0	0	0
	30	2	1	2	0	0.414	0.007	1	0.5	0.003	1	0
	31	2	1	2	0	0.398	0.01	1	0.6	0.005	1.5	0
	32	2	1	3	0	0	0	1	0	0	0	0
	33	2	1	3	0	0.355	0.003	1	0.333	0.002	0.5	0
	34	2	1	1	0	0	0	1	0	0	0	0
	35	2	1	1	0	0.359	0.003	1	0.333	0.002	0.5	0
2017	1	65	64	474	0.324	0.568	0.264	1.97	0.396	0.004	1.292	0.015
	2	60	59	125	0.072	0.498	0.102	1.97	0.263	0.002	0.7	0.017
	3	28	30	164	0.01	0.478	0.045	2.14	0.259	0.002	0.75	0.179
	4	26	25	40	0.027	0.492	0.053	1.92	0.296	0.002	0.808	0.115
	5	22	21	66	0.056	0.495	0.069	1.91	0.455	0.004	1.591	0.045
	6	19	19	105	0	0.412	0.011	2	0.156	0.001	0.368	0.263
	7	18	17	39	0.004	0.459	0.027	1.89	0.32	0.002	0.889	0.111
	8	15	14	47	0.051	0.488	0.054	1.87	0.533	0.006	2.133	0.067
	9	15	14	37	0.064	0.505	0.063	1.87	0.525	0.006	2.067	0.133
	10	12	12	22	0.001	0.482	0.03	2	0.547	0.007	2.417	0.417
	11	8	8	38	0	0.392	0.005	2	0.158	0.001	0.375	0.25
	12	7	6	15	0	0.461	0.021	1.71	0.5	0.005	1.714	0.286
	13	7	6	26	0	0.446	0.024	1.71	0.5	0.005	1.714	0.429
	14	6	5	25	0	0.392	0.005	1.67	0.167	0.001	0.333	0.167
	15	5	4	6	0.001	0.44	0.016	1.6	0.5	0.004	1.6	0.2
	16	5	4	20	0.003	0.48	0.029	1.6	0.579	0.006	2.2	0.2
	17	5	4	8	0	0.447	0.013	1.6	0.467	0.004	1.4	0.2
	18	4	3	9	0	0.352	0.005	1.5	0.333	0.002	0.75	0.5
	19	4	3	4	0	0.383	0.003	1.5	0.333	0.002	0.75	0.5
	20	4	3	22	0.001	0.426	0.011	1.5	0.6	0.006	2.25	0.25
	21	3	2	2	0	0.383	0.003	1.33	0.2	0.001	0.333	0.333
	22	3	2	4	0.001	0.423	0.013	1.33	0.556	0.004	1.667	0.333
	23	3	2	8	0	0.391	0.005	1.33	0.5	0.004	1.333	0.333
	24	3	2	4	0	0.383	0.003	1.33	0.333	0.002	0.667	0.333
	25	2	1	2	0	0.432	0.01	1	0.667	0.005	2	0
	26	2	1	1	0	0.383	0.003	1	0.333	0.001	0.5	0
	27	2	1	1	0	0.353	0.003	1	0.333	0.001	0.5	0
	28	2	1	2	0	0.411	0.005	1	0.5	0.003	1	0
	29	2	1	2	0	0	0	1	0	0	0	0
	30	2	1	2	0	0.396	0.005	1	0.5	0.003	1	0
	31	2	1	1	0	0	0	1	0	0	0	0
	32	2	1	2	0	0.34	0.003	1	0.333	0.001	0.5	0
	33	2	1	18	0	0.396	0.005	1	0.5	0.003	1	0
	34	2	1	5	0	0	0	1	0	0	0	0
	35	2	1	2	0	0.383	0.003	1	0.333	0.001	0.5	0
	36	2	1	2	0	0.383	0.003	1	0.333	0.001	0.5	0
	37	2	1	2	0	0.366	0.005	1	0.5	0.003	1	0
	38	2	1	4	0	0.367	0.005	1	0.5	0.003	1	0
	39	2	1	1	0	0	0	1	0	0	0	0
	40	2	1	2	0	0.383	0.003	1	0.333	0.001	0.5	0
	41	2	1	2	0	0.353	0.003	1	0.333	0.001	0.5	0
	42	2	1	2	0	0.335	0.003	1	0.333	0.001	0.5	0

Year	id	Actors	Edges	Inter-actions	Group Centrality			Avg. internal degree	Conduct-ance	Cut-Ratio	Expan-sion	Fraction over median degree
2018	1	53	53	302	0.323	0.618	0.279	2	0.373	0.005	1.189	0.057
	2	26	25	73	0.05	0.512	0.104	1.92	0.383	0.005	1.192	0.038
	3	18	17	52	0.037	0.508	0.093	1.89	0.424	0.005	1.389	0.056
	4	15	14	39	0.022	0.527	0.077	1.87	0.462	0.006	1.6	0.067
	5	15	15	19	0.038	0.459	0.031	2	0.286	0.003	0.8	0.4
	6	14	13	34	0.038	0.499	0.069	1.86	0.519	0.008	2	0.071
	7	13	13	66	0.01	0.489	0.034	2	0.409	0.005	1.385	0.462
	8	11	11	26	0.031	0.501	0.061	2	0.569	0.01	2.636	0.273
	9	8	7	16	0.002	0.483	0.034	1.75	0.533	0.007	2	0.125
	10	8	7	7	0	0.475	0.026	1.75	0.481	0.006	1.625	0.25
	11	7	6	10	0.03	0.494	0.037	1.71	0.556	0.008	2.143	0.286
	12	7	6	19	0.002	0.477	0.03	1.71	0.5	0.006	1.714	0.143
	13	7	6	20	0.015	0.479	0.03	1.71	0.455	0.005	1.429	0.286
	14	6	5	15	0	0.377	0.015	1.67	0.286	0.002	0.667	0.5
	15	6	5	10	0	0.44	0.015	1.67	0.444	0.005	1.333	0.5
	16	6	5	5	0	0.422	0.007	1.67	0.167	0.001	0.333	0.333
	17	5	4	9	0.001	0.451	0.026	1.6	0.467	0.005	1.4	0.2
	18	4	3	8	0	0.439	0.022	1.5	0.5	0.006	1.5	0.5
	19	4	3	3	0	0	0	1.5	0	0	0	0.25
	20	4	3	6	0	0.35	0.007	1.5	0.25	0.002	0.5	0.25
	21	3	2	2	0	0.344	0.004	1.33	0.2	0.001	0.333	0.333
	22	3	2	5	0	0.435	0.011	1.33	0.5	0.005	1.333	0.333
	23	3	2	3	0	0.317	0.004	1.33	0.2	0.001	0.333	0.333
	24	3	2	6	0	0	0	1.33	0	0	0	0.333
	25	2	1	1	0	0	0	1	0	0	0	0
	26	2	1	1	0	0.429	0.011	1	0.6	0.005	1.5	0
	27	2	1	2	0	0	0	1	0	0	0	0
	28	2	1	3	0	0.441	0.011	1	0.6	0.005	1.5	0
	29	2	1	1	0	0.414	0.004	1	0.333	0.002	0.5	0
	30	2	1	2	0	0	0	1	0	0	0	0
	31	2	1	10	0	0.432	0.007	1	0.5	0.004	1	0
	32	2	1	1	0	0	0	1	0	0	0	0
	33	2	1	4	0	0.354	0.004	1	0.333	0.002	0.5	0
	34	2	1	1	0	0	0	1	0	0	0	0
	35	2	1	1	0	0	0	1	0	0	0	0
	36	2	1	1	0	0.359	0.004	1	0.333	0.002	0.5	0
	37	2	1	1	0	0	0	1	0	0	0	0
2019	1	20	23	480	0.096	0.515	0.105	2.3	0.378	0.008	1.4	0.3
	2	19	18	30	0.015	0.491	0.087	1.89	0.294	0.005	0.789	0.053
	3	15	15	33	0.12	0.532	0.131	2	0.508	0.012	2.067	0.2
	4	15	14	36	0.14	0.54	0.102	1.87	0.391	0.007	1.2	0.133
	5	12	11	25	0.047	0.504	0.112	1.83	0.476	0.009	1.667	0.333
	6	11	10	38	0.089	0.497	0.083	1.82	0.487	0.01	1.727	0.091
	7	11	10	11	0.005	0.408	0.033	1.82	0.231	0.003	0.545	0.091
	8	11	13	48	0	0.404	0.017	2.36	0.212	0.004	0.636	0.182
	9	8	9	27	0.077	0.537	0.098	2.25	0.591	0.018	3.25	0.375
	10	8	7	16	0	0.389	0.016	1.75	0.176	0.002	0.375	0.375
	11	7	6	55	0.003	0.429	0.033	1.71	0.455	0.008	1.429	0.143
	12	5	4	8	0	0.44	0.027	1.6	0.429	0.006	1.2	0.2
	13	5	4	5	0.001	0.412	0.027	1.6	0.385	0.005	1	0.2
	14	5	4	17	0	0	0	1.6	0	0	0	0.4
	15	4	3	9	0.028	0.473	0.059	1.5	0.714	0.02	3.75	0.25
	16	4	3	7	0	0.394	0.027	1.5	0.455	0.007	1.25	0.25
	17	3	2	10	0	0.376	0.011	1.33	0.333	0.004	0.667	0.333
	18	3	2	3	0.001	0.442	0.032	1.33	0.6	0.011	2	0.333
	19	3	2	3	0	0.348	0.005	1.33	0.2	0.002	0.333	0.333
	20	3	2	2	0	0.361	0.005	1.33	0.2	0.002	0.333	0.333
	21	3	2	4	0	0.348	0.005	1.33	0.2	0.002	0.333	0.333
	22	2	1	1	0	0	0	1	0	0	0	0
	23	2	1	1	0	0.36	0.005	1	0.333	0.003	0.5	0
	24	2	1	2	0	0.358	0.005	1	0.333	0.003	0.5	0
	25	2	1	1	0	0.358	0.005	1	0.333	0.003	0.5	0
	26	2	1	2	0	0	0	1	0	0	0	0
	27	2	1	2	0	0	0	1	0	0	0	0
	28	2	1	1	0	0	0	1	0	0	0	0
	29	2	1	2	0	0	0	1	0	0	0	0

Year	id	Internal edge density	Normalized Cut-Ratio	Max ODF	Avg ODF	Flake ODF	Triangle participation	Score			
								Internal	Leave	Centrality	Sum
2013	1	0.029	0.217	2	0.077	0	0.615	4.06	0.93	3.00	2.16
	2	0.250	0.339	1	0.500	0	0	1.00	1.62	0.66	0.61
	3	0.250	0.339	1	0.500	0	0	1.00	1.62	0.62	0.60
	4	0.250	0.339	1	0.500	0	0	1.00	1.62	0.62	0.60
2014	1	0.018	0.505	38	1.357	0.036	0	0.69	2.92	2.03	1.20
	2	0.025	0.580	33	1.850	0.050	0	0.65	3.48	1.75	1.19
	3	0.045	0.575	15	2.000	0.182	0	0.68	3.55	1.07	0.99
	4	0.056	0.516	11	1.800	0.100	0.300	1.68	3.07	0.98	1.17
	5	0.063	0.558	9	2.111	0.333	0	1.50	3.73	0.97	1.23
	6	0.071	0.656	9	2.714	0.714	0	0.82	4.99	0.92	1.23
	7	0.083	0.495	4	1.500	0.333	0	1.22	2.99	0.78	0.95
	8	0.100	0.519	7	1.600	0.200	0	0.99	3.00	0.86	0.91
	9	0.125	0.621	5	2.250	0.250	0	1.65	3.85	0.81	1.24
	10	0.125	0.514	5	1.500	0.250	0	1.15	2.94	0.80	0.93
	11	0.125	0.514	3	1.500	0.250	0	1.65	2.92	0.79	1.07
	12	0.167	0.509	2	1.333	0.333	0	1.43	2.87	0.74	0.98
	13	0.167	0.338	2	0.667	0	0	1.43	1.43	0.62	0.69
	14	0.250	0.505	1	1.000	0	0	1.00	2.12	0.66	0.70
	15	0.250	0.336	1	0.500	0	0	1.00	1.27	0.62	0.53
	16	0.250	0	0	0	0	0	1.00	0	0	0.10
	17	0.250	0.607	3	1.500	0.500	0	1.00	3.48	0.75	0.96
	18	0.250	0.336	1	0.500	0	0	1.00	1.27	0.60	0.53
	19	0.250	0.336	1	0.500	0	0	1.00	1.27	0.62	0.53
	20	0.250	0.505	2	1.000	0.500	0	1.00	2.79	0.71	0.83
	21	0.250	0.336	1	0.500	0	0	1.00	1.27	0.62	0.53
2015	1	0.010	0.491	62	1.240	0.020	0	0.90	2.83	2.22	1.31
	2	0.019	0.442	27	1.259	0.037	0.111	1.05	2.34	1.18	0.92
	3	0.020	0.576	30	1.920	0.120	0	0.67	3.29	1.24	1.00
	4	0.029	0.389	12	1.059	0.059	0	0.64	1.93	0.87	0.63
	5	0.038	0.449	6	1.429	0.214	0.214	1.49	2.48	0.81	0.95
	6	0.056	0.374	3	1.000	0.222	0	1.40	1.97	0.73	0.81
	7	0.083	0.488	2	1.500	0.500	0	1.22	2.97	0.73	0.93
	8	0.100	0.113	1	0.200	0	0	1.39	0.43	0.63	0.50
	9	0.100	0.438	4	1.200	0.200	0	0.99	2.26	0.71	0.73
	10	0.125	0.509	3	1.500	0.500	0	1.65	3.04	0.73	1.07
	11	0.125	0.584	5	2.000	0.500	0	1.15	3.61	0.75	1.03
	12	0.125	0.584	4	2.000	0.250	0	1.65	3.26	0.75	1.12
	13	0.125	0.253	1	0.500	0	0	1.15	1.00	0.68	0.55
	14	0.125	0.549	3	1.750	0.500	0	1.65	3.32	0.71	1.11
	15	0.125	0.614	4	2.250	0.500	0	1.65	3.85	0.72	1.21
	16	0.167	0.336	1	0.667	0	0	1.43	1.32	0.68	0.69
	17	0.167	0.336	2	0.667	0.333	0	1.43	1.78	0.67	0.76
	18	0.167	0.336	2	0.667	0	0	1.43	1.34	0.65	0.68
	19	0.250	0.335	1	0.500	0	0	1.00	1.20	0.62	0.52
	20	0.250	0.503	2	1.000	0.500	0	1.00	2.66	0.66	0.79
	21	0.250	0	0	0	0	0	1.00	0	0	0.10
	22	0.250	0.673	4	2.000	0.500	0	1.00	3.82	0.74	1.02
	23	0.250	0.503	2	1.000	0.500	0	1.00	2.66	0.68	0.80
	24	0.250	0.335	1	0.500	0	0	1.00	1.20	0.62	0.52
	25	0.250	0.335	1	0.500	0	0	1.00	1.20	0.62	0.52
	26	0.250	0.503	2	1.000	0.500	0	1.00	2.66	0.60	0.77
	27	0.250	0	0	0	0	0	1.00	0	0	0.10

Year	id	Internal edge density	Normalized Cut-Ratio	Max ODF	Avg ODF	Flake ODF	Triangle participation	Score			
2016	1	0.011	0.423	53	1.128	0.021	0	0.86	2.42	1.54	1.00
	2	0.018	0.497	41	1.621	0.034	0.103	1.04	2.80	1.24	1.02
	3	0.019	0.462	28	1.407	0.111	0	0.68	2.53	1.08	0.81
	4	0.019	0.545	46	1.926	0.074	0.111	1.05	3.23	1.32	1.12
	5	0.024	0.511	28	1.773	0.136	0.136	1.10	2.92	1.06	1.00
	6	0.033	0.589	14	2.529	0.471	0.235	1.49	3.89	0.94	1.24
	7	0.038	0.539	9	2.071	0.286	0.214	1.64	3.17	0.83	1.12
	8	0.038	0.525	11	1.846	0.308	0	0.97	3.04	0.85	0.91
	9	0.071	0.419	9	1.375	0.125	0.375	1.75	2.21	0.78	0.97
	10	0.083	0.483	5	1.500	0.167	0	1.22	2.46	0.78	0.86
	11	0.083	0.381	3	1.000	0.167	0	1.22	1.86	0.69	0.72
	12	0.100	0.711	7	3.600	0.600	0	1.39	5.00	0.80	1.36
	13	0.100	0.435	3	1.200	0.200	0	1.39	2.17	0.77	0.85
	14	0.100	0.276	2	0.600	0	0	1.39	1.10	0.60	0.61
	15	0.125	0.744	8	4.000	0.750	0	1.15	5.56	0.81	1.39
	16	0.125	0.460	2	1.250	0.250	0	1.15	2.32	0.66	0.78
	17	0.125	0.658	9	2.750	0.500	0	1.15	4.21	0.77	1.14
	18	0.125	0.336	1	0.750	0	0	1.65	1.34	0.57	0.72
	19	0.125	0.336	3	0.750	0	0	1.15	1.36	0.61	0.59
	20	0.167	0	0	0	0	0	1.43	0	0	0.23
	21	0.167	0.335	2	0.667	0	0	1.43	1.29	0.64	0.67
	22	0.167	0.504	4	1.333	0.333	0	1.43	2.62	0.70	0.92
	23	0.167	0.201	1	0.333	0	0	1.43	0.73	0.59	0.56
	24	0.167	0	0	0	0	0	1.43	0	0	0.23
	25	0.250	0.808	8	4.000	0.500	0	1.00	5.40	0.79	1.31
	26	0.250	0.334	1	0.500	0	0	1.00	1.17	0.54	0.49
	27	0.250	0.334	1	0.500	0	0	1.00	1.17	0.54	0.49
	28	0.250	0	0	0	0	0	1.00	0	0	0.10
	29	0.250	0	0	0	0	0	1.00	0	0	0.10
	30	0.250	0.502	2	1.000	0.500	0	1.00	2.60	0.64	0.77
	31	0.250	0.603	3	1.500	0.500	0	1.00	3.19	0.62	0.87
	32	0.250	0	0	0	0	0	1.00	0	0	0.10
	33	0.250	0.334	1	0.500	0	0	1.00	1.17	0.54	0.49
	34	0.250	0	0	0	0	0	1.00	0	0	0.10
	35	0.250	0.334	1	0.500	0	0	1.00	1.17	0.55	0.49
2017	1	0.008	0.475	84	1.292	0.015	0	1.06	2.93	1.79	1.23
	2	0.008	0.303	33	0.700	0	0	1.00	1.54	1.04	0.72
	3	0.020	0.279	16	0.750	0	0.179	1.37	1.33	0.82	0.72
	4	0.019	0.315	19	0.808	0.038	0	0.83	1.54	0.88	0.62
	5	0.023	0.486	25	1.591	0.045	0	0.65	2.52	0.95	0.76
	6	0.028	0.162	2	0.368	0	0.158	1.37	0.65	0.64	0.54
	7	0.028	0.335	10	0.889	0.056	0	0.75	1.56	0.75	0.56
	8	0.033	0.562	20	2.133	0.333	0	0.65	3.37	0.90	0.89
	9	0.033	0.553	23	2.067	0.133	0	0.78	3.08	0.96	0.90
	10	0.045	0.573	10	2.417	0.417	0.250	1.82	3.57	0.78	1.23
	11	0.071	0.161	1	0.375	0	0	1.14	0.64	0.60	0.46
	12	0.071	0.511	6	1.714	0.286	0	1.10	2.77	0.74	0.87
	13	0.071	0.511	8	1.714	0.143	0	1.39	2.61	0.72	0.91
	14	0.083	0.168	2	0.333	0	0	0.89	0.65	0.60	0.38
	15	0.100	0.507	3	1.600	0.200	0	0.99	2.55	0.69	0.78
	16	0.100	0.589	11	2.200	0.200	0	0.99	3.21	0.78	0.93
	17	0.100	0.473	5	1.400	0.400	0	0.99	2.63	0.70	0.80
	18	0.125	0.336	2	0.750	0	0	1.65	1.32	0.54	0.71
	19	0.125	0.336	1	0.750	0	0	1.65	1.31	0.58	0.72
	20	0.125	0.608	3	2.250	0.750	0	1.15	3.93	0.66	1.06
	21	0.167	0.201	1	0.333	0	0	1.43	0.72	0.58	0.55
	22	0.167	0.560	3	1.667	0.333	0	1.43	2.90	0.66	0.96
	23	0.167	0.504	2	1.333	0	0	1.43	2.10	0.60	0.80
	24	0.167	0.335	1	0.667	0	0	1.43	1.26	0.58	0.64
	25	0.250	0.670	3	2.000	0.500	0	1.00	3.60	0.67	0.96
	26	0.250	0.334	1	0.500	0	0	1.00	1.15	0.58	0.50
	27	0.250	0.334	1	0.500	0	0	1.00	1.15	0.54	0.48
	28	0.250	0.502	2	1.000	0.500	0	1.00	2.56	0.63	0.76
	29	0.250	0	0	0	0	0	1.00	0	0	0.10
	30	0.250	0.502	2	1.000	0.500	0	1.00	2.56	0.60	0.75
	31	0.250	0	0	0	0	0	1.00	0	0	0.10
	32	0.250	0.334	1	0.500	0	0	1.00	1.15	0.52	0.48
	33	0.250	0.502	2	1.000	0.500	0	1.00	2.56	0.60	0.75
	34	0.250	0	0	0	0	0	1.00	0	0	0.10
	35	0.250	0.334	1	0.500	0	0	1.00	1.15	0.58	0.50
	36	0.250	0.334	1	0.500	0	0	1.00	1.15	0.58	0.50
	37	0.250	0.502	2	1.000	0.500	0	1.00	2.56	0.56	0.74
	38	0.250	0.502	2	1.000	0.500	0	1.00	2.56	0.56	0.74
	39	0.250	0	0	0	0	0	1.00	0	0	0.10
	40	0.250	0.334	1	0.500	0	0	1.00	1.15	0.58	0.50
	41	0.250	0.334	1	0.500	0	0	1.00	1.15	0.54	0.48
	42	0.250	0.334	1	0.500	0	0	1.00	1.15	0.51	0.48

Year	id	Internal edge density	Normalized Cut-Ratio	Max ODF	Avg ODF	Flake ODF	Triangle participation	Score			
2018	1	0.010	0.453	62	1.189	0.019	0.057	1.12	2.63	1.89	1.23
	2	0.019	0.421	26	1.192	0.038	0	0.68	2.17	1.04	0.74
	3	0.028	0.454	24	1.389	0.056	0	0.64	2.39	0.99	0.75
	4	0.033	0.491	19	1.600	0.200	0	0.65	2.76	0.97	0.81
	5	0.036	0.301	6	0.800	0.067	0	1.37	1.43	0.80	0.73
	6	0.036	0.552	18	2.000	0.214	0	0.65	3.18	0.93	0.87
	7	0.042	0.431	6	1.385	0.077	0	1.50	2.15	0.81	0.90
	8	0.050	0.603	15	2.636	0.364	0.273	1.57	3.89	0.91	1.26
	9	0.063	0.553	9	2.000	0.375	0	0.77	3.30	0.79	0.88
	10	0.063	0.497	5	1.625	0.250	0	1.02	2.70	0.77	0.84
	11	0.071	0.574	9	2.143	0.286	0	1.10	3.33	0.85	1.00
	12	0.071	0.515	7	1.714	0.286	0	0.82	2.87	0.78	0.81
	13	0.071	0.467	7	1.429	0.143	0	1.10	2.38	0.80	0.82
	14	0.083	0.291	1	0.667	0	0	1.55	1.17	0.60	0.67
	15	0.083	0.454	3	1.333	0.333	0	1.55	2.49	0.69	0.93
	16	0.083	0.169	2	0.333	0	0	1.22	0.66	0.65	0.50
	17	0.100	0.475	7	1.400	0.200	0	0.99	2.46	0.73	0.78
	18	0.125	0.507	6	1.500	0.250	0	1.65	2.66	0.70	1.00
	19	0.125	0	0	0	0	0	1.15	0	0	0.15
	20	0.125	0.252	2	0.500	0	0	1.15	0.98	0.54	0.50
	21	0.167	0.201	1	0.333	0	0	1.43	0.73	0.52	0.53
	22	0.167	0.505	2	1.333	0.333	0	1.43	2.61	0.67	0.91
	23	0.167	0.201	1	0.333	0	0	1.43	0.73	0.48	0.52
	24	0.167	0	0	0	0	0	1.43	0	0	0.23
	25	0.250	0	0	0	0	0	1.00	0	0	0.10
	26	0.250	0.604	3	1.500	0.500	0	1.00	3.20	0.66	0.89
	27	0.250	0	0	0	0	0	1.00	0	0	0.10
	28	0.250	0.604	3	1.500	0.500	0	1.00	3.20	0.68	0.89
	29	0.250	0.335	1	0.500	0	0	1.00	1.18	0.63	0.52
	30	0.250	0	0	0	0	0	1.00	0	0	0.10
	31	0.250	0.502	2	1.000	0.500	0	1.00	2.60	0.66	0.78
	32	0.250	0	0	0	0	0	1.00	0	0	0.10
	33	0.250	0.335	1	0.500	0	0	1.00	1.18	0.54	0.49
	34	0.250	0	0	0	0	0	1.00	0	0	0.10
	35	0.250	0	0	0	0	0	1.00	0	0	0.10
	36	0.250	0.335	1	0.500	0	0	1.00	1.18	0.55	0.49
	37	0.250	0	0	0	0	0	1.00	0	0	0.10
2019	1	0.030	0.429	18	1.400	0.100	0.300	1.82	2.42	1.10	1.14
	2	0.026	0.321	15	0.789	0	0	0.64	1.55	0.93	0.58
	3	0.036	0.562	21	2.067	0.200	0.200	1.30	3.41	1.21	1.19
	4	0.033	0.423	18	1.200	0.067	0	0.78	2.23	1.19	0.83
	5	0.042	0.511	20	1.667	0.083	0	1.17	2.83	1.04	1.00
	6	0.045	0.520	15	1.727	0.182	0	0.68	2.97	1.02	0.87
	7	0.045	0.242	4	0.545	0	0	0.68	1.04	0.69	0.42
	8	0.059	0.225	3	0.636	0	0.455	1.86	1.06	0.64	0.75
	9	0.080	0.636	13	3.250	0.375	0	1.51	4.61	1.10	1.43
	10	0.063	0.182	1	0.375	0	0	1.27	0.74	0.62	0.52
	11	0.071	0.472	6	1.429	0.143	0	0.82	2.48	0.71	0.72
	12	0.100	0.439	5	1.200	0.200	0	0.99	2.30	0.71	0.74
	13	0.100	0.393	3	1.000	0	0	0.99	1.75	0.67	0.63
	14	0.100	0	0	0	0	0	1.39	0	0	0.22
	15	0.125	0.740	10	3.750	0.750	0	1.15	5.71	0.86	1.44
	16	0.125	0.463	3	1.250	0	0	1.15	2.11	0.64	0.73
	17	0.167	0.337	2	0.667	0	0	1.43	1.35	0.59	0.66
	18	0.167	0.610	6	2.000	0.333	0	1.43	3.50	0.73	1.09
	19	0.167	0.202	1	0.333	0	0	1.43	0.76	0.53	0.54
	20	0.167	0.202	1	0.333	0	0	1.43	0.76	0.55	0.55
	21	0.167	0.202	1	0.333	0	0	1.43	0.76	0.53	0.54
	22	0.250	0	0	0	0	0	1.00	0	0	0.10
	23	0.250	0.335	1	0.500	0	0	1.00	1.21	0.55	0.50
	24	0.250	0.335	1	0.500	0	0	1.00	1.21	0.55	0.50
	25	0.250	0.335	1	0.500	0	0	1.00	1.21	0.55	0.50
	26	0.250	0	0	0	0	0	1.00	0	0	0.10
	27	0.250	0	0	0	0	0	1.00	0	0	0.10
	28	0.250	0	0	0	0	0	1.00	0	0	0.10
	29	0.250	0	0	0	0	0	1.00	0	0	0.10

References

Bosu, A. & Carver, J. C. (2014). Impact of developer reputation on code review outcomes in oss projects: An empirical investigation. In *Proceedings of the 8th acm/ieee international symposium on empirical software engineering and measurement*. ESEM '14. Torino, Italy: Association for Computing Machinery. doi:10.1145/2652524.2652544

Everett, M. G. & Borgatti, S. P. (1999). The centrality of groups and classes. *The Journal of Mathematical Sociology*, *23*(3), 181–201. doi:10.1080/0022250X.1999.9990219. eprint: https://doi.org/10.1080/0022250X.1999.9990219

Flake, G. W., Lawrence, S. & Giles, C. L. (2000). Efficient identification of web communities. In *Proceedings of the sixth acm sigkdd international conference on knowledge discovery and data mining* (pp. 150–160). KDD '00. Boston, Massachusetts, USA: Association for Computing Machinery. doi:10.1145/347090.347121

Freeman, L. (1977). A set of measures of centrality based on betweenness. *Sociometry*, *40*, 35–41. doi:10.2307/3033543

GitHub. (2020a). Features · code review · github. Retrieved January 15, 2020, from https://github.com/features/code-review

GitHub. (2020b). Pull requests — github developer guide. Retrieved January 16, 2020, from https://developer.github.com/v3/pulls

GitHub. (2020c). Reviews — github developer guide. Retrieved January 16, 2020, from https://developer.github.com/v3/pulls/reviews

Hagberg, A. A., Schult, D. A. & Swart, P. J. (2008). Exploring network structure, dynamics, and function using networkx. In G. Varoquaux, T. Vaught & J. Millman (Eds.), *Proceedings of the 7th python in science conference* (pp. 11–15). Pasadena, CA USA.

Hamasaki, K., Kula, R. G., Yoshida, N., Cruz, A. E. C., Fujiwara, K. & Iida, H. (2013). Who does what during a code review? datasets of oss peer review repositories. In *2013 10th working conference on mining software repositories (msr)* (pp. 49–52). doi:10.1109/MSR.2013.6624003

IEE. (2017). Iso/iec/ieee international standard - systems and software engineering–vocabulary. *ISO/IEC/IEEE 24765:2017(E)*, 1–541. doi:10.1109/IEEESTD.2017.8016712

Kerzazi, N. & El Asri, I. (2016). Who Can Help to Review This Piece of Code? In *17th Working Conference on Virtual Enterprises (PRO-VE)* (Vol. AICT-480, pp. 289–301). Collaboration in a Hyperconnected World. Part 7: Co-development and Co-working - Collaborative Engineering. Porto, Portugal. doi:10.1007/978-3-319-45390-3_25

McIntosh, S., Kamei, Y., Adams, B. & Hassan, A. E. (2016). An empirical study of the impact of modern code review practices on software quality. *Empirical Software Engineering*, *21*(5), 2146–2189. doi:10.1007/s10664-015-9381-9

Otte, E. & Rousseau, R. (2002). Social network analysis: A powerful strategy, also for the information sciences. *Journal of Information Science*, *28*, 441–453. doi:10.1177/016555150202800601

Ouni, A., Kula, R. G. & Inoue, K. (2016). Search-based peer reviewers recommendation in modern code review. In *2016 ieee international conference on software maintenance and evolution (icsme)* (pp. 367–377). doi:10.1109/ICSME.2016.65

Paugh, J. (2019). Agithub. https://github.com/mozilla/agithub/releases/tag/v2.2.1. GitHub.

Radicchi, F., Castellano, C., Cecconi, F., Loreto, V. & Parisi, D. (2004). Defining and identifying communities in networks. *Proceedings of the National Academy of Sciences of the United States of America*, *101*, 2658–63. doi:10.1073/pnas.0400054101

Rossetti, G., Milli, L. & Cazabet, R. (2019). Cdlib: A python library to extract, compare and evaluate communities from complex networks. *Applied Network Science*, *4*(1), 52. doi:10.1007/s41109-019-0165-9

Rosvall, M. & Bergstrom, C. T. (2008). Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences*, *105*(4), 1118–1123. doi:10.1073/pnas.0706851105. eprint: https://www.pnas.org/content/105/4/1118.full.pdf

Yang, J. & Leskovec, J. (2015). Defining and evaluating network communities based on ground-truth. *Knowl. Inf. Syst. 42*(1), 181–213. doi:10.1007/s10115-013-0693-z

Yang, X., Kula, R. G., Yoshida, N. & Iida, H. (2016). Mining the modern code review repositories: A dataset of people, process and product. In *Proceedings of the 13th international conference on mining software repositories* (pp. 460–463). MSR '16. Austin, Texas: Association for Computing Machinery. doi:10.1145/2901739.2903504

Yang, Z., Algesheimer, R. & Tessone, C. J. (2016). A comparative analysis of community detection algorithms on artificial networks. *Scientific Reports*, *6*(1), 30750. doi:10.1038/srep30750