# Guided Interaction and Collaborative Exploration in Heterogeneous Network Visualizations

Linnaeus University Dissertations

No 343/2019

# Guided Interaction and Collaborative Exploration in Heterogeneous Network Visualizations

Björn Zimmer

LINNAEUS UNIVERSITY PRESS

**Guided Interaction and Collaborative Exploration in Heterogeneous Network Visualizations**
Doctoral Dissertation, Department of Computer Science and Media Technology, Linnaeus University, Växjö, 2019

ISBN: 978-91-88898-33-3 (print), 978-91-88898-34-0 (pdf)
Published by: Linnaeus University Press, 351 95 Växjö
Printed by: DanagårdLiTHO, 2019

## Abstract

Zimmer, Björn (2019). *Guided Interaction and Collaborative Exploration in Heterogeneous Network Visualizations,* Linnaeus University Dissertations No 343/2019, ISBN: 978-91-88898-33-3 (print), 978-91-88898-34-0 (pdf).

The visual exploration of large and complex network structures remains a challenge for many application fields, such as systems biology or social sciences. Often, various domain experts would like to work together to improve the analysis time or the quality of the analysis results. Collaborative visualization tools can facilitate the analysis process in such situations. Moreover, a growing number of real world networks are multivariate and often interconnected with each other. Entities in a network may have relationships with elements of other related data sets, which do not necessarily have to be networks themselves, and these relationships may be defined by attributes that can vary greatly. A challenge is to correctly assign the attributes and relations between different data sets and graphs in order to be able to analyze them visually afterwards. The navigation between the resulting visualizations is also difficult. How can users be guided to other interesting data points relevant to their current view and how can this information be additionally displayed in a graph without losing the overview of the data?

In this dissertation, we propose our new web-based visualization environment OnGraX, which supports distributed, synchronous and asynchronous collaboration of networks and related multivariate data sets. In addition to standard collaboration features like event tracking or synchronizing, our client/server-based system provides a rich set of visualization and interaction techniques for better navigation and overview of the input network. Changes made by specific analysts or even just visited network elements can be highlighted by heat maps, which enable us to visualize user behavior data without affecting the original graph visualization. We evaluate the usability of the heat map approach against two alternatives in a user experiment.

Additional features of OnGraX include a comprehensive visual analytics approach that supports researchers to specify and subsequently explore attribute-based relationships across networks, text documents, and derived secondary data. Our approach provides an individual search functionality based on keywords and semantically similar terms over an entire text corpus to find related network nodes. For examining these nodes in the interconnected network views, we introduce a new interaction technique, called Hub2Go, which facilitates the navigation by guiding the user to the information of interest. To showcase these features, we use a large text corpus collected from papers listed in the IEEE VIS publications data set (1990--2015) that consists of 2,752 documents. We analyze relationships between various heterogeneous networks, a Bag-of-Words index, and a word similarity matrix, all derived from the initial corpus and metadata. We also propose a design for the interactive specification of degree-of-interest functions, which can be used to provide and evaluate configurations for guidance based on network attributes and logged user data in heterogeneous networks.

Keywords: Information Visualization, Multivariate Networks, Visual Analytics, Exploration, Interaction, Collaboration, Provenance, Guidance

# *Abstract*

The visual exploration of large and complex network structures remains a challenge for many application fields, such as systems biology or social sciences. Often, various domain experts would like to work together to improve the analysis time or the quality of the analysis results. Collaborative visualization tools can facilitate the analysis process in such situations. Moreover, a growing number of real world networks are multivariate and often interconnected with each other. Entities in a network may have relationships with elements of other related data sets, which do not necessarily have to be networks themselves, and these relationships may be defined by attributes that can vary greatly. A challenge is to correctly assign the attributes and relations between different data sets and graphs in order to be able to analyze them visually afterwards. The navigation between the resulting visualizations is also difficult. How can users be guided to other interesting data points relevant to their current view and how can this information be additionally displayed in a graph without losing the overview of the data?

In this dissertation, we propose our new web-based visualization environment OnGraX, which supports distributed, synchronous and asynchronous collaboration of networks and related multivariate data sets. In addition to standard collaboration features like event tracking or synchronizing, our client/server-based system provides a rich set of visualization and interaction techniques for better navigation and overview of the input network. Changes made by specific analysts or even just visited network elements can be highlighted by heat maps, which enable us to visualize user behavior data without affecting the original graph visualization. We evaluate the usability of the heat map approach against two alternatives in a user experiment.

Additional features of OnGraX include a comprehensive visual analytics approach that supports researchers to specify and subsequently explore attribute-based relationships across networks, text documents, and derived secondary data. Our approach provides an individual search functionality based on keywords and semantically similar terms over an entire text corpus to find related network nodes. For examining these nodes in the interconnected network views, we introduce a new interaction technique, called Hub2Go, which facilitates the navigation by guiding the user to the information of interest. To showcase these features, we use a large text corpus collected from papers listed in the IEEE VIS publications data set (1990–2015) that consists of 2,752 documents. We analyze relationships between various heterogeneous networks, a Bag-of-Words index, and a word similarity matrix, all derived from the initial corpus and metadata. We also propose a design for the interactive specification of degree-of-interest functions, which can be used to provide and evaluate configurations for guidance based on network attributes and logged user data in heterogeneous networks.



**Keywords:**

Information Visualization, Multivariate Networks, Visual Analytics, Exploration, Interaction, Collaboration, Provenance, Guidance

# *Acknowledgments*

I would like to express my sincere gratitude and appreciation to everyone who was involved in the completion of this thesis. I would especially like to thank my supervisor Prof. Dr. Andreas Kerren for his guidance during my doctoral studies. This work would not have been possible without his exceptional supervision and his never ending patience at the end of this journey, while it took me a bit longer to finish the writing of this thesis.

I would also like to thank my friends and colleagues Ilir Jusufi, Daniel Cernea, Kostiantyn Kucher, Anna Wingkvist, and Rafael Messias Martins for all the support, lively discussions, and fun that we had during my time in Sweden.

Special thanks also to all other colleagues from Linnaeus University and to all the motivated people that I met on conferences, seminars, and workshops. Keep up the good work, and I hope to see you somewhere on the next interesting conference.

My deepest gratitude goes to my parents, who supported me during the years of my studies in Germany and supported me when I moved to Sweden and a couple of years later to Austria. I also want to thank my partner Nina for understanding all the hours I had to spend on my thesis.

Finally, I have to thank my cats for their emotional support. After rolling over my keyboard on countless of occasions, I claim them responsible for all the typos that might still be in this thesis.

# Table of Contents

**Abstract** iii

**Acknowledgments** v

**Table of Contents** vii

**List of Figures** x

**List of Publications** xii

**1 Introduction** 1
- 1.1 Research Problems and Goals of this Thesis . . . 4
- 1.2 Thesis Outline . . . 5

**2 Background Information and Related Work** 9
- 2.1 Terms and Definitions . . . 10
  - 2.1.1 Network Attributes . . . 10
  - 2.1.2 Network Centralities . . . 10
  - 2.1.3 Static and Dynamic Networks . . . 13
- 2.2 Visualization of Networks . . . 13
  - 2.2.1 Node-Link Representations . . . 14
  - 2.2.2 Matrix Representations . . . 15
  - 2.2.3 Hybrid Representations . . . 16
  - 2.2.4 Visualizing Graphs with Multivariate Attributes . . . 16
  - 2.2.5 Visualization in a Web Browser . . . 21
- 2.3 Heterogeneous Networks . . . 21
  - 2.3.1 Definition of Heterogeneous Networks . . . 23
  - 2.3.2 Visualization and Navigation in (Heterogeneous) Networks . 25
- 2.4 Guidance in Network Visualizations . . . 27
  - 2.4.1 Degree-of-Interest (DOI) Functions . . . 29
- 2.5 Collaboration and Provenance in Network Visualizations . . . 30

**3 OnGraX – the Online Graph Exploration System** 35
- 3.1 Technical Requirements . . . 36
- 3.2 General System Overview . . . 37
- 3.3 OnGraX – Server . . . 39

3.3.1 MySQL and Neo4j Databases . . . 39
3.3.2 Authentication and User Sessions . . . 41
3.4 OnGraX – Web Client . . . 41
3.4.1 Action and Conflict Handling . . . 42
3.4.2 Increased Performance with WebGL . . . 43
3.4.3 WebGL Graph Rendering . . . 44
3.5 OnGraX – Data Set Server . . . 45
3.6 Visualization of Graphs from External Applications with OnGraX . 45
3.7 Discussion . . . 46
3.8 Summary . . . 48

**4 Collaborative Visual Analysis of Networks 49**
4.1 Visualization of User Behaviour . . . 50
4.1.1 Requirements . . . 51
4.1.2 Design Decisions . . . 52
4.1.3 Interaction and Visualization Techniques . . . 52
4.2 Visualization of User Data with Heat Maps . . . 59
4.2.1 Calculation of the Heat Map Values . . . 60
4.2.2 Logged User Views Compared to Eye Tracking Data . . . 61
4.3 Annotations and Chat Links . . . 62
4.4 Tracking and Replaying User Actions . . . 63
4.5 Evaluation and Expert Review . . . 64
4.5.1 Heat Map Evaluation . . . 64
4.5.2 Expert Review . . . 67
4.6 Summary . . . 69

**5 Matching of Attributes across Heterogeneous Networks 71**
5.1 Visualization of Documents . . . 72
5.2 Design . . . 74
5.2.1 Visual Design . . . 75
5.2.2 Hub2Go . . . 77
5.3 Example Application . . . 78
5.3.1 Text Preprocessing . . . 80
5.3.2 Bag-of-Words . . . 81
5.3.3 Word Similarity . . . 81
5.3.4 Matching . . . 82
5.3.5 Use Case . . . 84
5.4 Discussion . . . 90
5.5 Summary . . . 90

**6 Guided Interaction in Heterogeneous Networks** **93**
6.1 Problem Description . . . . . 94
6.2 Data for Guidance . . . . . 95
6.3 Similarity Calculation . . . . . 97
6.4 Visualizing and Navigating Guidance Results . . . . . 98
6.4.1 Configuring the DOI Function . . . . . 98
6.4.2 Enhanced Hub2Go . . . . . 99
6.4.3 Exploring Values of Multiple Guidance Results in Detail . . 101
6.4.4 Automatic On-The-Fly Suggestions . . . . . 103
6.5 Discussion . . . . . 105
6.6 Summary . . . . . 105

**7 Conclusions and Outlook** **107**
7.1 Conclusions . . . . . 107
7.2 Outlook . . . . . 111

**Bibliography** **113**

# List of Figures


1.1 How to explore relationships between heterogeneous networks and related data . . . 2
1.2 Collaborative visualization scenarios . . . 3
1.3 The six building blocks of this thesis . . . 6

2.1 Three techniques for the visualization of network centralities . . . 11
2.2 Overview of the network visualization tool ViNCent . . . 13
2.3 Three types of main graph visualization techniques . . . 14
2.4 Crossings and symmetry . . . 14
2.5 Different ordering of nodes in matrix representations . . . 16
2.6 NodeTrix . . . 17
2.7 An example for multiple coordinated views . . . 18
2.8 Glyphs in node-link diagrams . . . 18
2.9 Semantic substrates . . . 19
2.10 PivotGraph . . . 20
2.11 JauntyNets . . . 20
2.12 Overview of interconnected networks . . . 22
2.13 Three-level network example . . . 23
2.14 Heterogeneous network definition – data structure . . . 24
2.15 Caleydo . . . 25
2.16 Entourage . . . 26
2.17 Signposts for navigation . . . 26
2.18 Enriched wedges for navigation . . . 27
2.19 Characterization of guidance . . . 28
2.20 A decision tree for guidance degrees . . . 30
2.21 Refinery: Visualization based on a user query . . . 31
2.22 A news/leak system with guidance . . . 31
2.23 Modular DOI specification . . . 32

3.1 OnGraX' architecure . . . 38
3.2 OnGraX' database schema . . . 40
3.3 Action handling between clients . . . 43
3.4 Structure of supported files for the import of additional data sets . . . 46
3.5 Sequence of loading and visualizing a graph from an external application . . . 47

4.1 Overview of the OnGraX system . . . . . 53
4.2 Heat map visualization . . . . . 55
4.3 Different color schemes for the heat map . . . . . 56
4.4 Robust heat map creation . . . . . 57
4.5 Viewport filtering . . . . . 58
4.6 Dialog for heat map settings . . . . . 59
4.7 Heatmap based on graph changes . . . . . 60
4.8 Comparison of eye tracker data to logged user views . . . . . 62
4.9 Transition to a previous graph state . . . . . 64
4.10 Task 1, mean error rate . . . . . 66
4.11 Task 1, mean completion time . . . . . 66
4.12 Task 2, mean completion time . . . . . 66

5.1 How to visualize different, interconnected networks and data sets . . 73
5.2 Overview of an attribute matching session in progress . . . . . 76
5.3 Usage of the Hub2Go widgets . . . . . 78
5.4 Overview of the data sets . . . . . 79
5.5 Dialog to configure the mapping between graphs and data sets . . . 83
5.6 Overview of a search result by author . . . . . 85
5.7 Distribution of 8 selected terms in van Wijk's papers . . . . . 87
5.8 Network view consisting of all papers mentioning the term "traffic_flows" . . . . . 88
5.9 TF-IDF chart and paper network for the term "traffic_flows" and six related terms . . . . . 89

6.1 Selection of bins . . . . . 99
6.2 Overview of the enhanced Hub2Go visualization . . . . . 100
6.3 Enhanced Hub2Go explained . . . . . 101
6.4 Visualizing all DOI results for a node with the ViNCent metaphor . 102
6.5 Detailed view for visualizing automatic guidance results on nodes . 103
6.6 Overview of visualizing automatic guidance results on nodes . . . . 104

# List of Publications

**This thesis is based on the following refereed publications:**

1. Björn Zimmer, Magnus Sahlgren and Andreas Kerren. Visual Analysis of Relationships between Heterogeneous Networks and Texts: An Application on the IEEE VIS Publication Data Set. *Informatics*, 4(2):11, 2017, MDPI AG. Materials appear in Chapter 5.

2. Björn Zimmer and Andreas Kerren. OnGraX: A Web-Based System for the Collaborative Visual Analysis of Graphs. *Journal of Graph Algorithms and Applications*, 21(1). Pages 5–27, 2017. Materials appear in Chapter 4.

3. Björn Zimmer and Andreas Kerren. Displaying User Behavior in the Collaborative Graph Visualization System OnGraX. *Graph Drawing and Network Visualization: 23rd International Symposium on Graph Drawing and Network Visualization*, GD '15, Los Angeles, CA, USA, September 24–26, 2015, Revised Selected Papers. Springer. Pages 247–259. Materials appear in Chapter 4.

4. Björn Zimmer and Andreas Kerren. Harnessing WebGL and WebSockets for a Web-Based Collaborative Graph Exploration Tool. *Engineering the Web in the Big Data Era: 15th International Conference*, ICWE '15, Rotterdam, The Netherlands, June 23–26, 2015, Proceedings. Pages 583–598. Materials appear in Chapter 3.

5. Björn Zimmer and Andreas Kerren. Sensemaking and Provenance in Distributed Collaborative Node-Link Visualizations, Abstract paper, *IEEE VIS 2014 Workshop: Provenance for Sensemaking*, Paris, France, 2014. Materials appear in Chapter 4.

6. Björn Zimmer and Andreas Kerren. Applying Heat Maps in a Web-Based Collaborative Graph Visualization, Poster Abstract, *IEEE Information Visualization, InfoVis '14*, Paris, France, 2014. Materials appear in Chapter 4.

7. Björn Zimmer, Ilir Jusufi, and Andreas Kerren. Analyzing Multiple Network Centralities with ViNCent, *In Proceedings of the SIGRAD '12 Conference on Interactive Visual Analysis of Data*, Pages 87–90, Växjö, Sweden, 2012. Linköping University Electronic Press. Materials appear in Chapter 6.

8. Andreas Kerren, Harald Köstinger, and Björn Zimmer. ViNCent – Visualization of Network Centralities, *In Proceedings of the International Conference on Information Visualization Theory and Applications, IVAPP '12*, Pages 703–712, Rome, Italy, 2012. SciTePress. Materials appear in Chapter 6.

**Further publications not related to this thesis:**

1. Ilir Jusufi, Andreas Kerren, and Björn Zimmer. Visual Exploration of Relationships between Document Clusters. *Proceedings of the 5th International Conference on Information Visualization Theory and Applications, IVAPP '14*, Pages 195–203, Lisbon, Portugal, 2014. SciTePress.

2. Ilir Jusufi, Andreas Kerren, and Björn Zimmer. Multivariate Network Exploration with JauntyNets. *Proceedings of the 17th International Conference on Information Visualisation, IV '13*, 16–18 July 2013, London, United Kingdom. Pages 19–27.

3. Michael Wybrow, Niklas Elmqvist, Jean-Daniel Fekete, Tatiana von Landesberger, Jarke J. van Wijk, and Björn Zimmer. Interaction in the Visualization of Multivariate Networks. *Multivariate Network Visualization: Dagstuhl Seminar #13201*, Dagstuhl Castle, Germany, May 12–17, 2013. Revised Discussions. 97–125, Springer 2014.

*Chapter 1*

# Introduction

The visual exploration of networks[1] plays an important role in various application fields. Researchers use network visualizations to analyze publication networks to find out details about specific authors, explore citations among various papers or books, or discover publications and related work based on a keyword search. In social sciences, network visualizations are a feasible way to analyze various social networks, where nodes represent specific individuals or groups, and edges depict relationships between social entities. Network representations are also found in software engineering in the form of UML diagrams to document software architecture or database schemes. Another interesting topic is the exploration of complex biological networks, such as metabolic pathways, protein interaction networks, or gene regulatory networks. Since biological networks are usually too large to visualize in one single view, researchers analyze subsets of these graphs which are interconnected with each other. Exploring these subsets and getting an understanding of the interconnections among them remains an interesting challenge and is nowadays not only limited to biological networks, as more and more data all over the world is gathered and analyzed. For instance, publication data sets increase in size each year, and researchers are interested in exploring interconnections between different publication venues.

Additionally, the growing availability and complexity of data gives information visualization researchers the opportunity to combine and explore different multivariate networks that are not only interconnected with each other, but might also have references to additional multivariate data sets (see Figure 1.1). These networks are usually referred to as *heterogeneous* networks. Integrating heterogeneous network visualizations for *different* use cases into a single system that could assist researchers in their work is a challenging task. Researchers should be able to dynamically specify and explore relations between heterogeneous networks and additional multivariate data to find relevant or interesting links in the data. For instance, combining and analyzing publication data from different conferences and journals, consisting of citation and co-authorship networks combined with additional data from the publications, such as keywords, titles, or the actual text of the papers, could be interesting for young researchers and assist them to get an overview of their field and find relevant papers.

[1] *Networks* are also addressed as *graphs* throughout this thesis.

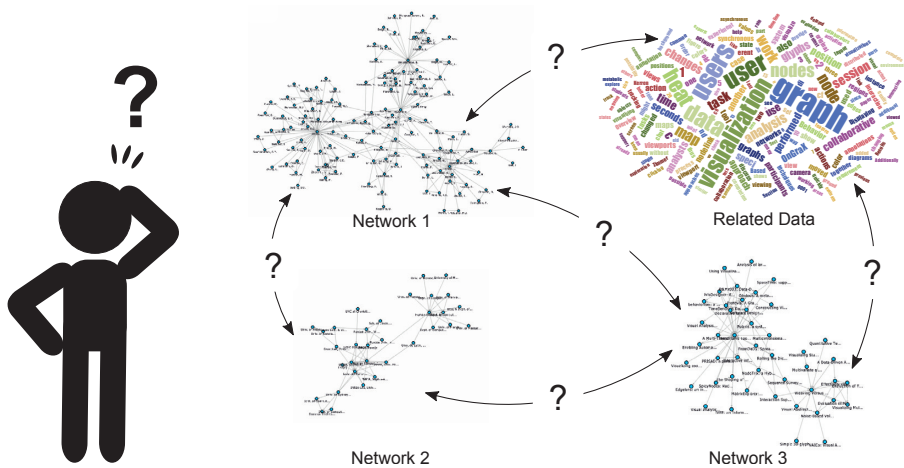



Figure 1.1: How to explore relationships between different heterogeneous networks and related data sets?

With the growing size of complex data, cooperative analysis is becoming an important new method for many data analysts as cooperation might improve the quality of the analysis process [112] and help to analyze data sets efficiently. Also, networks are explored by various domain experts, who would like to collaborate to improve the quality of the analysis process. There are various different scenarios for collaborative visualizations (see Figure 1.2). Collaboration can take place in a joint online session where everybody works *simultaneously* on one data set and discusses possible changes and insights in real-time with other users. In this case, an expert might want to see what the others are doing and if there are possibilities to coordinate the efforts and find a common ground. In another scenario, experts would like to work on the networks whenever time permits. Both situations cause specific problems that should be handled by tools which support collaborative work. For instance, while working independently, it would be helpful to see changes of the data performed by other analysts. Another interesting issue is to see which part of the data set has already been explored by others and to know who changed the data: was an established expert working on a specific part of the data, or a new staff member who might not have the same experience as the expert? In such an *asynchronous* setting, it would be interesting to quickly perceive changes that were performed by former analysts or to find out which parts of the networks were already explored by others.

Nowadays, information visualization researchers use the web to make collaborative data visualizations available [115, 154]. The biggest advantage of these web-based applications is their convenient usage. Users do not have to download and install additional software packages or plug-ins to collaborate. It suffices to share a URL

to start a collaborative session with other colleagues. This tackles a problem which was mentioned by Isenberg et al. [82] as one of the ongoing challenges in collaborative information visualization: collaborators would like to seamlessly drop in and out of ongoing work. A vast amount of interactive web applications such as text editors, online drawing tools or more advanced office suites (e.g., Google Docs) provide this feature. The online nature of these applications makes them feasible for collaboration, and an increasing amount of users spread across the globe work together to edit documents in web-based tools. These collaborative applications should assist users by providing insights about others who are working on the same document to help everyone in establishing a *common ground* [31, 67] in the shared workspace. Users should be able to notice if another user joins or leaves a session and be aware of changes that others are applying to a document.

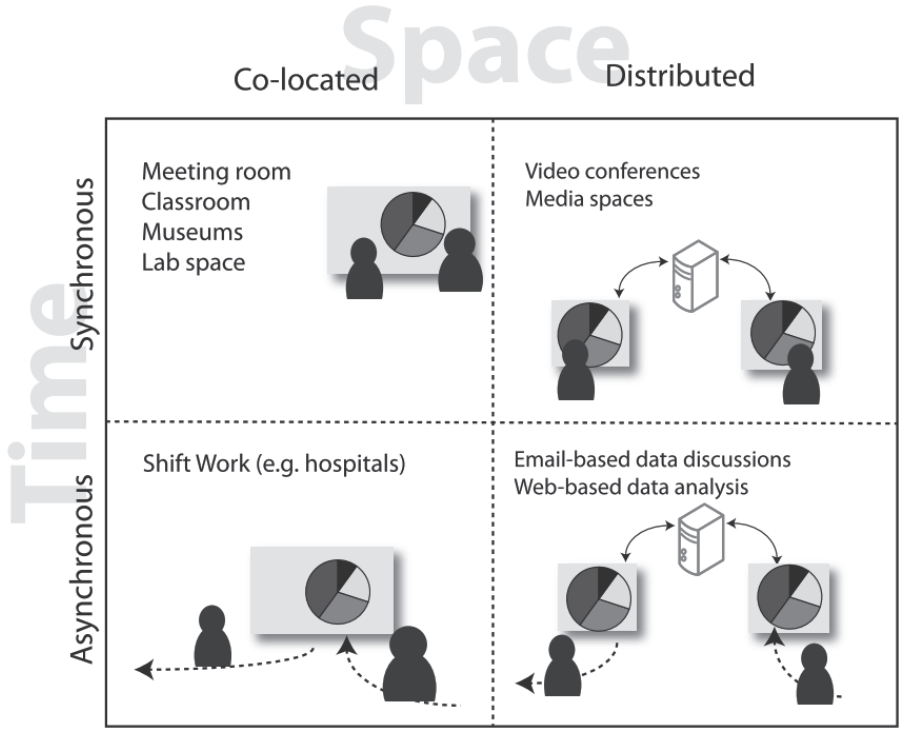



Figure 1.2: Different scenarios for collaborative visualization. Image taken from [82]. ©Information Visualization Journal.

Providing means of guidance for navigating through the resulting relationships in the various networks and data sets is also a topic of increasing importance. Metabolic pathways, for instance, are often connected with each other, and biologists are look-

ing for new possibilities to visualize and navigate these interconnections. Moreover, suggesting navigation links from one network to another could not only be done based on predefined relations in the network data: if a system would be able to collect interaction data of ongoing exploration tasks, for example, this data could also be used to recommend links between the networks. Data which is generated by users during the exploration process is typically referred to as *provenance information* [128]. If a previous user spent a lot of time analyzing a specific region of a graph and from there moved to a related area of another interconnected graph, this action might also be interesting for other analysts. Alternatively, if a system could suggest a number of related target nodes in other networks and would also know that previous users did not make use of these suggestions, it might be interesting to analyze these unvisited nodes and eventually gain new insights.
The aforementioned aspects of performing collaborative analyses of heterogeneous networks and the challenges that arise in such a setting are addressed in this thesis. The main technical contribution of this work is the development of a web-based network visualization approach, which facilitates the visual exploration of interconnected, heterogeneous networks in a collaborative environment.

## 1.1 Research Problems and Goals of this Thesis

To be able to use provenance information gathered during collaborative analysis processes, analysts need the possibility to use a comprehensive network visualization tool, which is easy to access and has the ability to record and track all actions the analysts might want to perform on the data. Thereby, the first main goal of this thesis is to design and implement an online system that serves as foundation for the joint collaboration on graphs for a number of different purposes.

**Goal 1:** Design and implementation of an online graph visualization system to enable collaborative work on graphs.

More precisely, the system has to fulfill a number of criteria. It should:

G 1.A: provide interactive visualizations to perform synchronous and asynchronous collaborative analyses of node-link diagrams,

G 1.B: store all relevant user actions and track which graph objects were viewed by the users, and

G 1.C: give other users options to visually explore this stored provenance information.

The next step is to give support for the analysis of *heterogeneous* networks. However, the relations between networks and data sets vary greatly, depending on the

kind of data; as such there are different use cases for the exploration of these networks. The system should allow analysts to interactively specify and subsequently explore relations between different graphs and related multivariate data sets.

> **Goal 2:** Extension of the graph visualization system to support specification and exploration of relationships between heterogeneous networks and additional data sets.

The extension should support:

G 2.A: features for an interactive specification of individual relationships between different networks and additional data sets, and

G 2.B: navigation between these mapped relationships with the help of visualization widgets.

After achieving the first two goals, the next step is to combine the features of both goals and devise a prototype design for guidance in heterogeneous networks. Note that this step does not include an implementation part. We create a design recommendation for a guidance system which would be able to use gathered provenance information from Goal 1, together with features for exploring heterogeneous graphs and data sets from Goal 2.

> **Goal 3:** Design of a visualization system to be able to specify and test configurations for guidance in heterogeneous networks for various use cases.

The design includes ideas on how to:

G 3.A: visualize guidance results and use them to navigate across networks, and

G 3.B: provide visualization widgets to show the computed values of the guidance results in order to give researchers the possibility to assess their usefulness and specify various settings for different use cases.

## *1.2 Thesis Outline*

This work is divided into six blocks (see Figure 1.3), which define the core contributions of this thesis. Before these blocks are addressed in detail, Chapter 2 introduces and discusses background information as well as related work in the visualization of (heterogeneous) networks. It also examines arising challenges researchers have to face if they aim to collaboratively analyze and explore networks. These challenges include the task of setting up a collaborative environment together with multiple

analysts in different locations, coordinating ongoing work, and using information gained from previous sessions to gain insights. Additionally, the task of working with various interconnected networks is addressed. How are the interconnections between these networks defined and what do researchers need to be able to explore them?

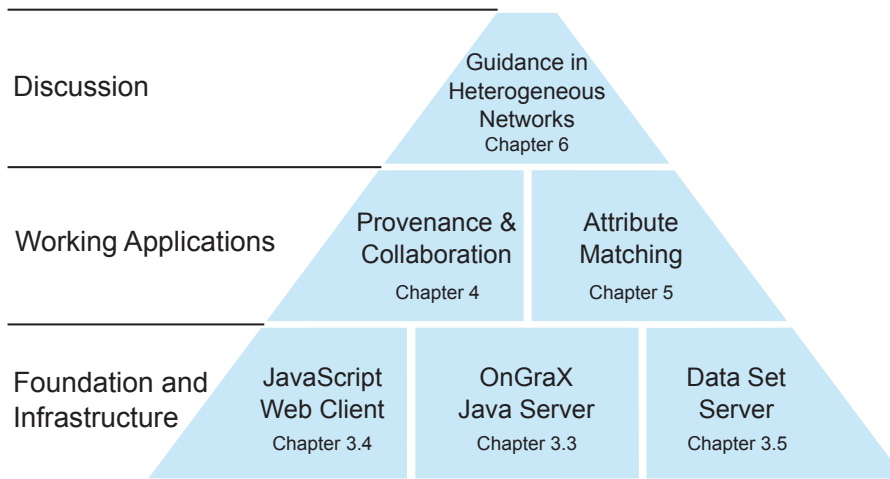



Figure 1.3: The six building blocks of this thesis.

Chapter 3 describes the architecture and implementation details of our Online Graph Exploration system (OnGraX). It provides the main infrastructure for our research goals and allows us to address them by using one comprehensive tool throughout this work. It consists of three main parts: the web client, the core server system, and the data set server, which together represent the first three building blocks and the foundation of this thesis.

Performing collaborative work on graphs is discussed in Chapter 4. To exemplify our approach, we address the collaborative analysis of metabolic networks from the Kyoto Encyclopedia of Genes and Genomes (KEGG) pathway database [104]. Biologists of different domains and experience levels want to explore networks built from experiments, check them for wrong entries or missing data, and revise the graphs wherever it is necessary. In this use case, it is important to know what part of the network has already been checked and what part still needs attention. Facilitating this provenance information can also be used as a kind of quality check: an area which has been investigated by many different experts is likely to have a higher quality than an area only investigated by one scientist.

Chapter 5 steers the focus to heterogeneous networks and discusses a comprehensive visual analytics approach that supports researchers to specify and explore attribute-based relationships across different data sets of different types, such as networks and text data. To showcase our approach, we analyze a large text corpus collected from papers listed in the IEEE VIS publications data set [84] and explore relationships between various heterogeneous networks, a Bag-of-Words index, and a word similarity matrix, all derived from the initial corpus and metadata.

Based on the knowledge gained from the previous parts of this thesis, Chapter 6 discusses possible features for realizing guidance in heterogeneous networks. Harnessing OnGraX' features to gather provenance information from users during graph exploration processes and utilizing mapping functionality across different data sets allows us to design a flexible guidance system for graph exploration. Since there are different use cases, such as revising metabolic networks, exploring citation networks, or viewing diagrams of software structures, the system has to provide researchers with a flexible functionality to specify different configurations for these cases. The goal of this chapter is to present the design of a prototype that allows researchers to create test cases for the exploration of various different heterogeneous networks. It should help users to setup configurations for the recommendation of navigation cues for different kinds of use cases. For instance, the exploration of publication networks, analysis of interconnected biological graphs, or visualization of software diagrams is based on different multivariate attributes, and every use case requires an individual configuration to calculate the navigation suggestions correctly.

The last chapter summarizes and discusses the results of this thesis and gives an outlook to future work.

*Chapter 2*

# Background Information and Related Work

While the focus of this work predominantly relies on node-link visualizations, there are various alternative techniques for the visualization of networks (also referred to as *graphs*), which might be more beneficial than others depending on the use cases at hand. Choosing the most advantageous visualization also depends on the type and quantity of additional attributes that could be of relevance and part of the network data. Moreover, networks could be derived from static data sets, whose structures and attributes do not, or do seldom change over time—or the network could be based on an interactive data set where the actual visualization of changes to the structure or the attributes could be of relevance to an analyst. This chapter discusses these challenges and gives an overview of various network visualization techniques and different types of networks. It starts with a brief discussion of terms and definitions, gives a brief overview of general network visualization techniques, and discusses related work about collaboration in network visualization systems.

## Contents


**2.1 Terms and Definitions . . . . . 10**
2.1.1 Network Attributes . . . . . 10
2.1.2 Network Centralities . . . . . 10
2.1.3 Static and Dynamic Networks . . . . . 13
**2.2 Visualization of Networks . . . . . 13**
2.2.1 Node-Link Representations . . . . . 14
2.2.2 Matrix Representations . . . . . 15
2.2.3 Hybrid Representations . . . . . 16
2.2.4 Visualizing Graphs with Multivariate Attributes . . . . . 16
2.2.5 Visualization in a Web Browser . . . . . 21
**2.3 Heterogeneous Networks . . . . . 21**
2.3.1 Definition of Heterogeneous Networks . . . . . 23
2.3.2 Visualization and Navigation in (Heterogeneous) Networks . . . . . 25
**2.4 Guidance in Network Visualizations . . . . . 27**
2.4.1 Degree-of-Interest (DOI) Functions . . . . . 29
**2.5 Collaboration and Provenance in Network Visualizations . . 30**

## 2.1 Terms and Definitions

A network or graph provides information about single elements and relationships between them. A widely used example are social networks, where persons are represented as nodes and relationships between two persons (i.e., friends, relatives, or co-workers) are represented as edges between two nodes. A (simple) *graph* $G = (V,E)$ consists of a finite set of nodes (or vertices) $V$ and a set of edges $E \subseteq \{(u,v)|u,v \in V, u \neq v\}$. An edge $e = (u,v)$ in the graph $G$ connects two nodes $u$ and $v$. Two nodes $u$ and $v$ are said to be *incident* with the edge $e = (u,v)$ and *adjacent* to each other. The degree $d(u)$ of a node $u$ is defined as the number of edges incident to this node $u$. Furthermore, we can define a *walk* on a graph as described as follows: let $(e_1,\ldots,e_k)$ be a sequence of edges in a graph $G = (V,E)$. This sequence is called a *walk* if there are nodes $v_0,...,v_k$ such that $e_i = (v_{i-1},v_i)$ for $i = 1,...,k$. If the edges $e_i$ and the nodes $v_i$ are pairwise distinct respectively, then the walk is called a *path*. The *length* of a walk/path is given by its number of edges, i.e., $k = |(e_1,\ldots,e_k)|$. A *shortest path* between two nodes $u,v$ is a path with minimal length. The *distance* $dist(u,v)$ between two nodes $u,v$ is the length of a shortest path between them.

### 2.1.1 Network Attributes

A graph can also have one or more additional attributes on each of its nodes and/or edges. In such cases, a graph is referred to as *multivariate graph*. A multivariate graph is a graph $M = (V,E,A,W)$ with a set of nodes $V$ and a set of edges $E$ as introduced before. Additionally, it contains $p$ node attributes $A = (A_1,\ldots,A_p)$ and $q$ edge attributes $W = (W_1,\ldots,W_q)$ with $p,q \in \mathbb{N}$. These attributes can be distinguished in structural and associated attributes [16].
*Structural attributes* are implicitly contained in the graph and may be computed from its structure. A common example for a structural attribute is the *node degree*, which equals the number of incident edges of a node. Further and more complex structural attributes are network centralities, which are discussed in the next subsection.
*Associated attributes* are other attributes such as node names, the age of a person, the latitude and longitude to position nodes of a network on a map, or the $x$ and $y$ positions of a layout algorithm to visualize a graph on a 2D-canvas. Numerical attributes on edges are usually referred to as weights. They are often used to describe costs or flows of the connections between nodes.

### 2.1.2 Network Centralities

Network centralities are used to discover the relative importance of nodes within a complex network topology. The comparison of several different centrality values in larger networks is especially a challenge. Identifying communities and central actors in social networks or the calculation of the importance of specific nodes in

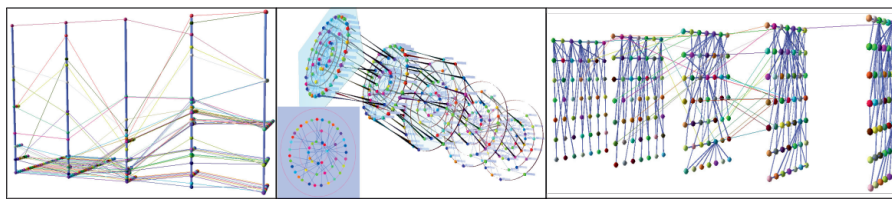


Figure 2.1: Three techniques for the visualization of network centralities: 3D parallel coordinates display the centrality values for each vertex (left), an orbit-based comparison method (middle), and hierarchy-based comparison (right). Image taken from [40]. ©Australian Computer Society, Inc.

biochemical networks are some examples where multiple network centralities are used. Typical tasks during network centrality analysis include finding nodes with high/low centrality values or finding nodes with high values in several centralities across a large number of nodes. Especially the latter task raises challenging analytical problems. How to visualize the data in a meaningful way for researchers? How to minimize occlusions and visual clutter? Or how to build a flexible solution in order to deal with a large number of centrality values at the same time? Note that not every centrality measure can be applied to every graph.

A network centrality $C$ is a function that assigns a value $C(u)$ to a node $u \in V$ of a given graph $G = (V,E)$. In order to compare network centralities according to their importance, $u$ is more important than $v$ if $C(u) > C(v)$, with $u, v \in V$. By using network centralities, analysts are able to better understand the structure of networks and to identify central actors. Typical examples are *degree*, *eccentricity*, or *random walk betweenness* [123]. Whereas the *degree* of a node is a very simple centrality that orders the nodes according to their degree values, *eccentricity* is calculated by using the distance (based on shortest paths) between nodes of an undirected, connected graph and is defined as $ecc(u) := max_{v \in V} dist(u,v)$ and the corresponding centrality as $C_{ecc}(u) := \frac{1}{ecc(u)}$. More central nodes have therefore a higher value of $C_{ecc}$.

The problem of choosing suitable centralities differs from network to network. For the computation of centralities, data about the functional properties of networks is often missing. This information would allow us to choose the "right" centrality measures [102]. Therefore, this analysis is usually done by visual comparison of centrality values on the networks via correlations, scatter plots, and parallel coordinates. Dwyer et al. [40] presented three techniques to visualize network centralities (see Figure 2.1). Coordinates-based comparison is based on parallel coordinates to visualize multivariate data. Standard approaches typically deal with two dimensions. This one uses 3D to stack visual representations of a network according to one centrality into the third dimension. Thus, each 2D plane contains the information for a particular centrality. 3D parallel coordinates-based comparison gives a good

overview of the centrality values within the network and about how many nodes fall into a certain value range. However, this approach does not reveal the actual network structure. In orbit-based comparison, nodes are placed in an orbit-based visualization that has some advantages over the previous approach: the network topology is shown and thus the relationships between the nodes can be identified. In more detail, for each centrality a new 2D orbit-based plane is added to the 3D drawing. The ordering of the planes takes edge crossing minimization and minimization of inter-plane edge lengths into account. As a single orbit provides information about the centrality values and as the network structure can be seen, this approach outperforms the previous one when revealing both structure and centrality values. Drawbacks are occlusions in the middle of the orbits, and it is hard to keep track of changes within the single centrality measures. Hierarchy-based comparison is conceptually similar compared to the 3D parallel coordinates approach, but it divides the nodes according to their centrality values into layers. Those layers are then drawn as horizontal lines, having an ordering on the line as well. This could be, for example, a decreasing centrality value from the left to the right. The top layers in the visualization are considered to show larger centrality values. There might be even connecting edges between nodes on the same layer. Filtering and thresholds are used to reduce visual clutter between two planes.

Junker et al. presented a different approach with the CentiBiN tool [86]. It uses standard node-link diagrams based on a set of graph drawing algorithms. In addition to the displayed graph, single centrality values are displayed next to the visualization in a table. Interactions with the table, like selecting certain centrality values, are coordinated with the network visualization as well, allowing the user to locate certain values within the network. Simple histograms are used to compare data. CentiBiN has advantages with respect to the amount of available centralities, as it can calculate up to 17 centrality measures for networks. Another tool was introduced by Scardoni et al. [136]. Their tool CentiScaPe is able to compute several network centralities and provides analyses of existing relationships between user data (based on experiments) and centrality values. It was implemented as a Cytoscape [139] plugin. However, the supported interactive visualizations are restricted.

Our own approach ViNCent [170] uses a circular view to visualize the structure of a network together with its network centralities (see Figure 2.2). Analyses can then be focused on the exploration of the centrality values including the network structure, without dealing with visual clutter or occlusions of nodes. Filtering based on statistical data concerning the network elements and centrality values supports this and helps keeping the network itself readable.

Further details about tools for biochemical network visualization can be found in a survey paper from Albrecht et al. [3] and in the work of Junker and Schreiber [87]. For further details in the field of social networks, we refer to the work of Corra and Ma [35].

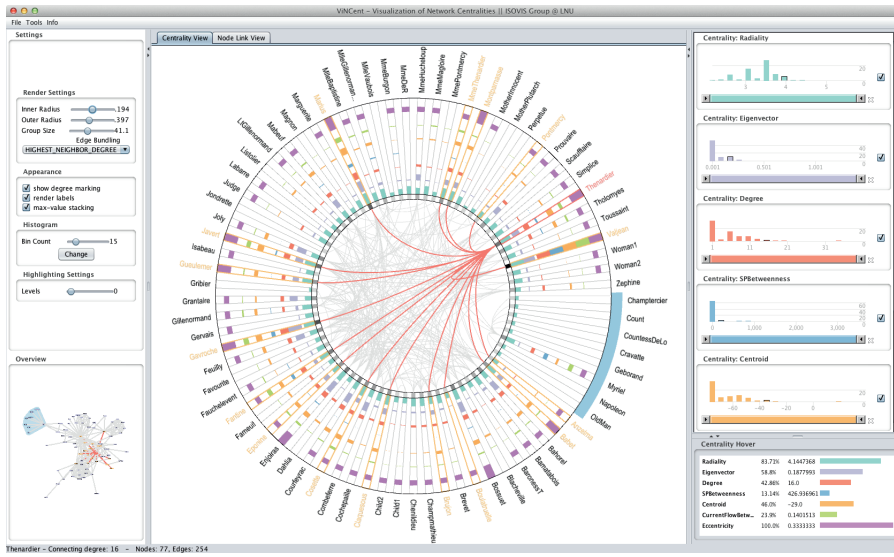



Figure 2.2: Overview of our network visualization tool ViNCent [96, 170]. The network is drawn as a circular view together with network centralities as bars. The right side displays the corresponding histograms of the network centralities as well as detailed values of the network centralities for the currently hovered node “Thenardier”. ©Linköping University Electronic Press.

### 2.1.3 Static and Dynamic Networks

In this work, we address various use cases for different kinds of networks. They can be static networks which do not change, or dynamic networks, which evolve over time. The dynamics of networks can be distinguished into *structural*- and *attribute*-related changes [156]. *Structural changes* are represented by adding or removing nodes and/or edges of a network and have an impact on the computed structural attributes of a network. Adding or removing nodes and/or edges can also have an impact on the associated attributes of a graph, for instance, if the removal of a node leads to a recalculation of the layout for a node-link diagram visualization. Therefore, structural changes can also lead to attribute changes. *Attribute changes* refer to changes of attributes from nodes or edges, such as the name, size, or shape of a node, or the weight of an edge attribute.

## *2.2 Visualization of Networks*

Network visualizations should help to grasp the topology of the network and clearly visualize the nodes and edges of a graph, as well as any additional attributes that might be of interest to the user. Based on the work of Landesberger et al. [156],

techniques for displaying graphs can be divided into three main groups which are discussed in the following: *node-link based*, *matrix-based*, and *hybrid* visualizations (see Figure 2.3).

Figure 2.3: Three types of general graph visualization techniques: node-link diagram (left), adjacency matrix (middle), hybrid (right). Image taken from [76]. ©IEEE.

### 2.2.1 Node-Link Representations

Traditional graph drawing methods focus on creating a 2D—or sometimes a 3D—layout of the nodes and edges in form of a node-link diagram [62]. In these diagrams, nodes are usually visualized as circles or rectangles, and edges between nodes are drawn as lines. Node-link visualizations play an important role in various network visualization applications, for instance, for the visualization of biological pathways [29], co-authorship networks [1], or social networks [57]. Available methods for graph visualizations are comprehensively surveyed in several surveys and books, for instance [14, 53, 97, 99, 156].

The readability of the results from a graph layout algorithm is affected by aesthetic criteria [23, 29, 36, 95], such as minimizing the number of edge crossings, edge bends, edge length, node overlap, and drawing area of the graph as well as maximizing the overall symmetry. These criteria often contradict each other, for instance, maximizing symmetry might introduce more edge crossings (see Figure 2.4), thus the challenge lies in finding the right balance to produce a satisfying layout.

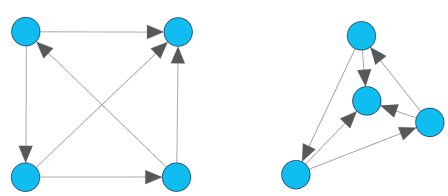


Figure 2.4: Two different drawings of the same graph. The graph on left was drawn with a focus on symmetry, but has more edge crossings than the graph on the right, which looks in turn less symmetric.

There are two main approaches to produce satisfying layouts. For small graphs it is feasible to layout the nodes and edges manually, which is supported by most graph drawing tools. For larger graphs, the use of automatic layout algorithms is necessary. Research in this field has already been done for more than 40 years, and there is a vast amount of algorithms to compute aesthetic graph layouts. Well-known examples are force-directed solutions [41, 48], algebaric approaches [63, 65, 101], and multi-level algorithms [61]. These approaches usually only focus on generating a layout for the nodes of a graph with straight edges between the nodes, additional solutions exist to route edges as polylines or orthogonal drawings [113, 163, 164]. Edge-bundling is another approach to deal with a lot of edges and reduce the clutter in visualizations of graphs with many edges. Zhou et al. give a good overview of edge-bundling approaches [169].
Most of these algorithms focus on specific aesthetic graph drawing criteria. However, the problem of finding an optimal layout usually remains NP-complete [37], and most of these algorithms only provide approximations of the optimal solution. With an increasing number of edges, the layout quality of a graph usually decreases and leads to "hairball" visualizations which are almost impossible to read. A suitable layout of a graph can provide a good overview of the overall structure and support manual exploration, for instance, to find a path between two given nodes [52, 94]. To be able to assess if a layout is actually "suitable" various different studies have been performed [22, 52, 64, 79, 80, 81, 127, 165], which evaluated structure-based tasks such as response time and accuracy or cognitive load. These studies revealed that the size of a graph (e.g., the number of nodes and edges), the layout of these nodes and edges, and the visual mapping of attributes have a significant impact on the effectiveness of a node-link visualization for specific tasks.

### 2.2.2 Matrix Representations

An alternative to node-link visualizations are matrix representations, which put an emphasis on a view of the edges of a graph by visualizing its adjacency matrix. Each row and column in the matrix represent a node and each cell depicts if an edge between the nodes in the respective column and row exists. Matrix representations visualize edges without edge crossings and, in contrast to node-link diagrams, guarantee the visibility of edges [143]. Matrix representations are used in the same application fields as node-link visualizations, e.g., the visualization of protein-interaction networks [43], co-authorship networks [77], or social networks [74]. The readability of a graph and its structure depends on the ordering of the matrix' rows and columns [15]. Changing the ordering of rows and/or columns can affect if an analyst is able to perceive cliques, edges between nodes, or the number of outgoing edges of a node (see Figure 2.5). There are several algorithms for matrix ordering, for instance [43, 74, 120].
The effectiveness of matrix representations versus node-link representations has been compared by Ghoniem et al. [52]. The results show that matrices are the better

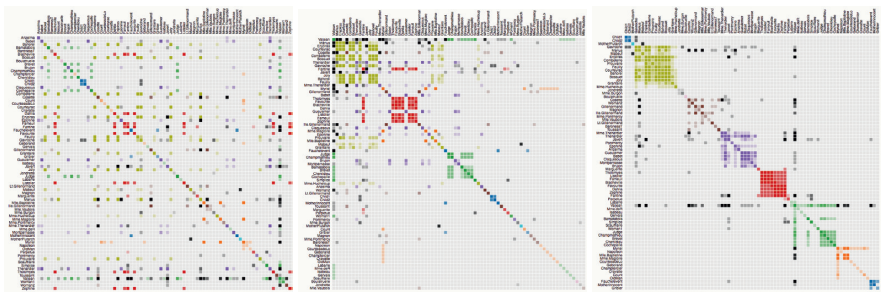


Figure 2.5: Three matrix representations of all character co-occurrences in the novel "Les Misérables": Ordered by name (left), by frequency (middle), and by cluster (right). Image taken from [117].

choice for large and dense graphs. In contrast to node-link visualizations, matrix representations are usually not practical for identifying paths along several nodes. For this problem, researchers suggest superimposing visual links to show paths between nodes, for instance, interactively by selecting two nodes [75, 143]. Another problem of a matrix representation is the required screen space, which increases quadratically with the number of nodes [94, 156]. To solve this issue, various zoom and aggregation techniques were proposed [2, 43, 150]. Matrix visualizations are also rarely useful for sparse graphs, because most of the cells are left empty, if there are no connections between nodes.

### 2.2.3 Hybrid Representations

Hybrid representations combine node-link and matrix representations to combine the advantages of the individual approaches. NodeTrix [76] uses a hybrid representation to visualize a co-authorship network, that is locally dense, but globally sparse (see Figure 2.6). It is often hard to decide which combination of representations one should use for which data. For this case, there are automatic approaches [7] and alternative approaches, which leave this decision to the analyst and support switching between different representations, for instance, for the visualization of social networks [76] or the structure of a database [168].

### 2.2.4 Visualizing Graphs with Multivariate Attributes

Graphs often have additional attributes, which can be integrated in both node-link and matrix visualizations. In node-link diagrams, multivariate glyphs could replace the traditional node representation to show additional data, which is attached to nodes. Edge attributes could be represented through different edge colors, thickness of the edge, edge shapes, or even labels along an edge. In case of matrix representations, cells could have different colors or sizes [152], or they could also be

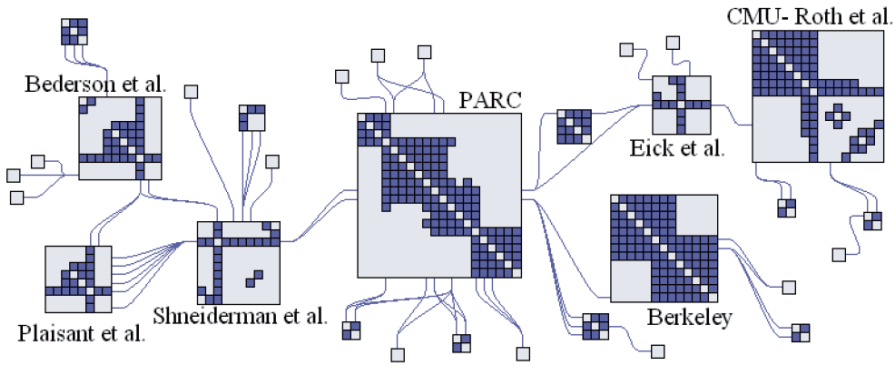



Figure 2.6: Hybrid graph representation: NodeTrix — Representation of the largest component of the InfoVis Co-authorship Network. Dense nodes are visualized as matrix representations, while sparse connections use the node-link metaphor. Image taken from [76]. ©IEEE.

replaced by icons or labels to show additional edge attributes [77]. The problem with these solutions is that they introduce more clutter into the visualizations and do not scale well for bigger networks. Kerren et al. give a good overview of different techniques that try to solve these issues [97]. Their suggestions are based on the work of Jusufi [88]. Several approaches can be found in the literature that offer solutions for the problem of visualizing multivariate networks: multiple and coordinated views, integrated approaches, semantic substrates, attribute-driven layouts, and hybrid approaches. These approaches are discussed in the following.

*Multiple and coordinated views:* Solutions in this category combine several views and present them together, which allows the user to choose the most powerful visualization techniques [45, 130]. One example is the work of Shannon et al. [140]. Their approach consists of two distinct views: a parallel coordinate approach for the visual representation of the network attributes combined with a traditional node-link drawing of a graph (see Figure 2.7). Both views are coordinated by linking and brushing techniques [145]. The drawback of multiple views is that they split the displayed data because of the spatial separation of the visual elements.

*Integrated approaches:* To provide a combined picture, attributes and the underlying graph can be displayed in one single view [56]. Borisjuk et al. [19] use small diagrams (e.g., bar charts) instead of representing the nodes as simple circles, dots, or rectangles (see Figure 2.8). Each diagram shows experimental data that is related to the regarded node. The embedding of the visualizations into the nodes consumes a lot of space, which may affect the readability of the network due to the visual clutter that may appear when the number of nodes and the attributes is high [95]. However, the problem of space usage and additional clutter can be alleviated by interaction techniques, such as [89].

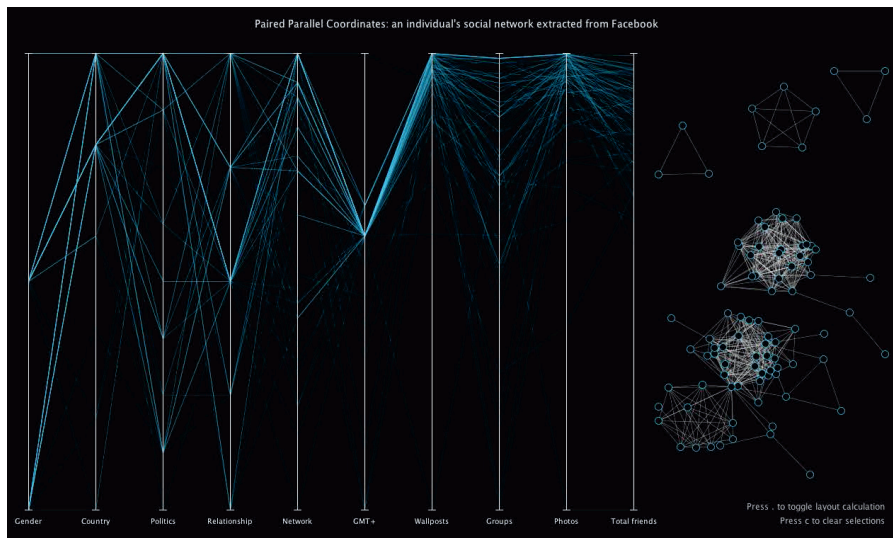



Figure 2.7: A view of a parallel coordinates plot enhanced with a corresponding node-link graph drawing, allowing selections in one view to be represented in another. This data is from a social network from Facebook, with 89 nodes, 434 edges and 10 data dimensions [140]. ©University College Dublin.

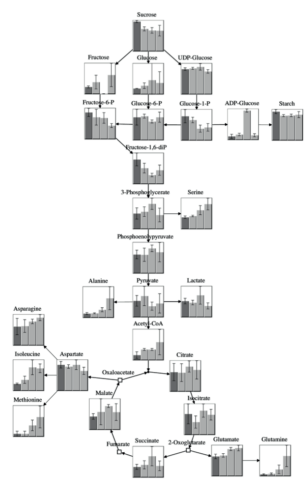



Figure 2.8: Visualization of experimental data in the context of a metabolic network. Additional data is shown as a diagram on each node. Image taken from [19]. ©IOS Press.

*Semantic substrates:* In order to further avoid clutter in multivariate network visualizations, some researchers realized the idea of so-called semantic substrates that "are non-overlapping regions in which node placement is based on node attributes". Shneiderman and Aris [144] introduced this idea and combined it with sliders to interactively control the edge visibility and thus to ensure comprehensibility of the edges' end nodes. Their tool efficiently improves the situation of visual clutter that happens with large multivariate networks. However, the underlying graph topology is not (completely) visible with this approach (see Figure 2.9).

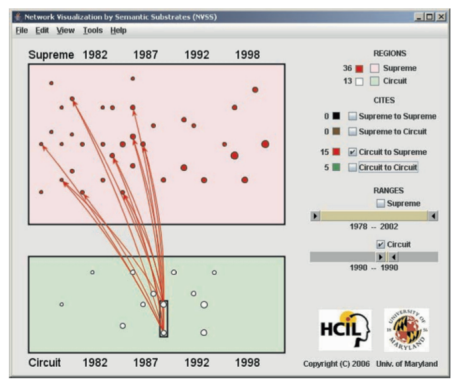



Figure 2.9: Semantic substrates: Nodes are placed in different regions according to their attributes. Image taken from [144]. ©IEEE.

*Attribute-driven layouts:* The display of the network elements is used to present insight about the attached multivariate data instead of visualizing the graph topology itself. In contrast to semantic substrates, this technique does not necessarily place the nodes into specific regions. Instead, it controls the placement of a node in the graph layout by considering the node's attributes. An example is PivotGraph [157] which shows the relationships between (node) attributes and links within a 2D grid-layout (see Figure 2.10). This concrete approach scales well for some situations, because of the inherent node aggregation (nodes on the same grid position share the same attribute values) but is restricted to discrete attribute values and only two attribute dimensions.

*Hybrid approaches:* This idea combines at least two of the previously discussed techniques. The most common combinations are multiple coordinated views with any of the integrated approaches. For instance, Rohrschneider et al. [132] integrate additional attributes of a biological network inside the nodes and edges. The authors also use other visual metaphors for creating multiple coordinated views to show time-related data of the network. Another hybrid approach is the JauntyNets tool, which was suggested by Jusufi et al. [90]. It combines multiple coordinated views with an attribute-driven node-link layout (see Figure 2.11).

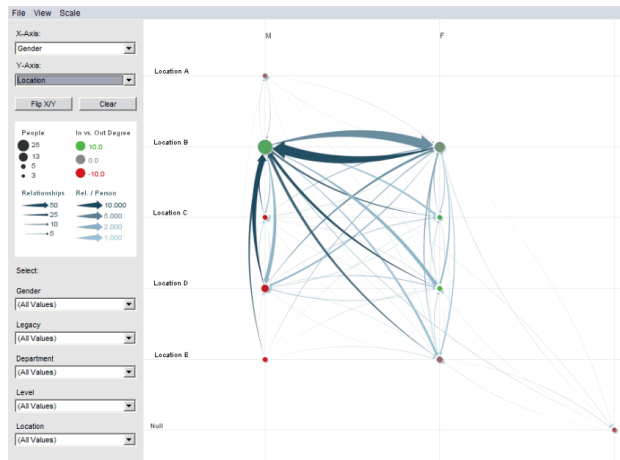



Figure 2.10: PivotGraph: Nodes are placed on a 2D grid based on their attributes. Image taken from [157]. ©IEEE.

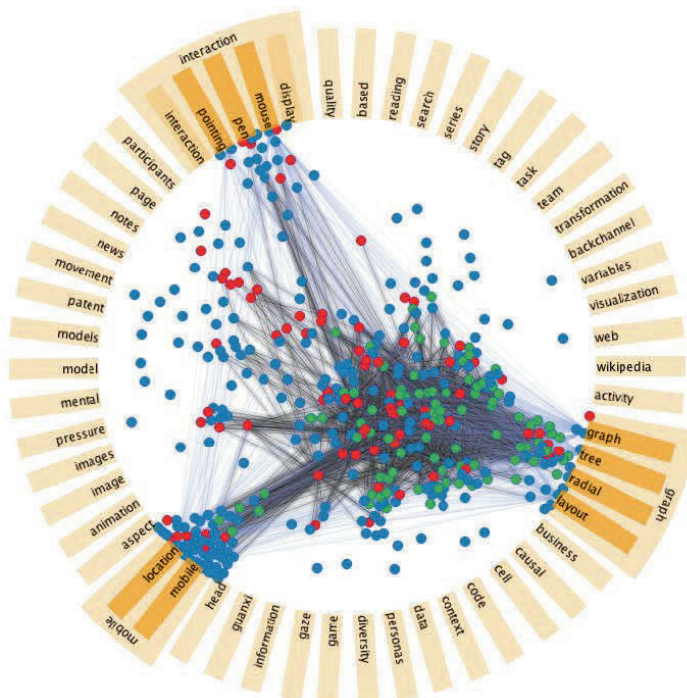



Figure 2.11: JauntyNets: Nodes are steered towards their corresponding attributes. Image taken from [90]. ©IEEE.

### 2.2.5 Visualization in a Web Browser

To facilitate collaborative visualization of networks on-the-fly in a web browser, we already defined the implementation of a web-based system as **Goal 1** of this thesis (see Section 1.1). Of course, desktop applications for single users with the possibility to visualize graphs, such as Cytoscape [139], Gephi [12], Pajek [13], or Tulip [8], have been used before the advent of the Web 2.0. However, the idea of working collaboratively on complex data sets is becoming more and more attractive with the rapidly growing amount of data available. Thus, a good number of web-based visualizations were already introduced that support the public exploration of complex data sets [70]. Users are able to add comments on interesting visualizations and also discuss new insights with other users over the web. Several systems support those features, such as ManyEyes [154] or Sense.us [71]. In these web-based visualization systems, users are able to save bookmarks of specific views on a data set, and it is possible to add graphical annotations directly to the visualization. Dashiki [115] enables users to collaboratively build visualization dashboards with the help of a wiki-like syntax and interactive editors. However, the aforementioned systems are not suitable for our tasks, since they do not support the interactive visualization of node-link diagrams in a web browser. And, they do not provide any features for real-time interactions during synchronous collaboration settings.
Drawing graphs in a web browser is usually done with the help of already existing JavaScript libraries. Arbor.js [135] and Sigma.js [4] render the input graphs on a HTML5 canvas or via SVG-images, but they do not support OpenGL-enhanced rendering which limits the number of nodes and edges that can be rendered with a suitable frame rate. A faster solution is provided by VivaGraphJS [6]: this library uses a WebGL renderer to draw graphs and provides high performance during rendering, but gives only limited support for different and more complex node shapes. Additionally, these libraries are not able to efficiently render a lot of text, e.g., node labels for all nodes visible at the same time, which is one of the basic requirements for our web-based tool.

## *2.3 Heterogeneous Networks*

The multivariate networks from different research fields, that were discussed so far, are often also interconnected with additional networks or other, not necessary network related data sources (see Figure 2.12). These networks are usually referred to as *multimodal networks*, *network of networks*, or *heterogeneous networks* and can also be assigned to different levels (or scales). Examples for application domains are biological network visualizations, social networks, or visualizations for the structures of programs in software engineering, for instance. Schreiber et al. give a comprehensive introduction, examples, and challenges for practical applications of heterogeneous networks in [137].

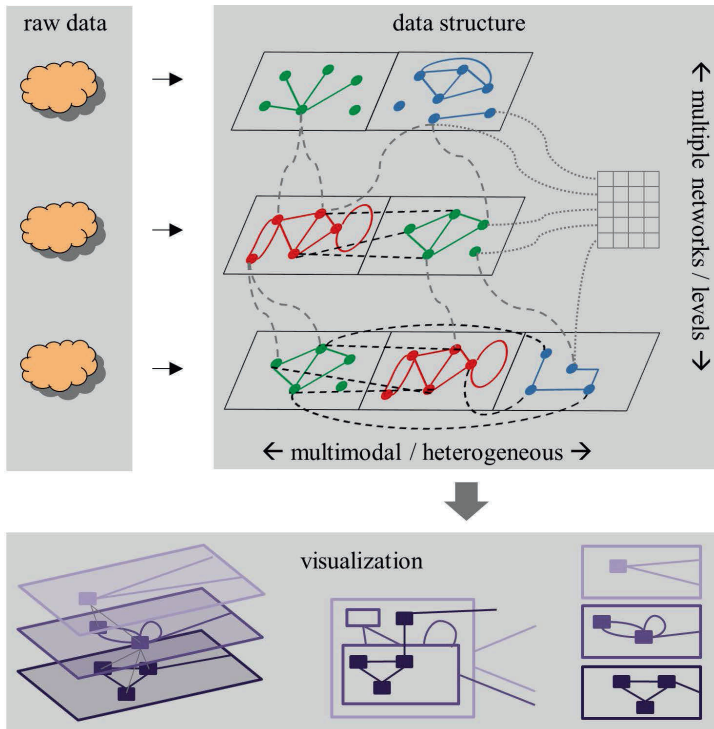



Figure 2.12: An overview of various interconnected networks. Different networks on the same level can have connections with each other, as well as with other networks on other levels. Additional attributes from a different data source could also have connections to the various networks, as symbolized by the small grey data matrix on the right hand side. Image taken from [137]. ©Springer International Publishing Switzerland 2014.

One of the discussed examples of heterogeneous networks on different levels are collaboration and citation networks. A group of authors from the same institution could form an affiliation network on the lowest level, where nodes represent authors and edges between those nodes are added if the authors wrote a paper together. The top level could represent aggregations of the lower levels, for instance, different institutions or different journals where papers are published. Figure 2.13 shows an example of different levels for collaboration and citation networks. The problem of exploring different, interconnected heterogeneous networks, as well as combining and visualizing such data from different sources is defined as **Goal 2** (see Section 1.1) of this thesis and our solution is discussed in Chapter 5.

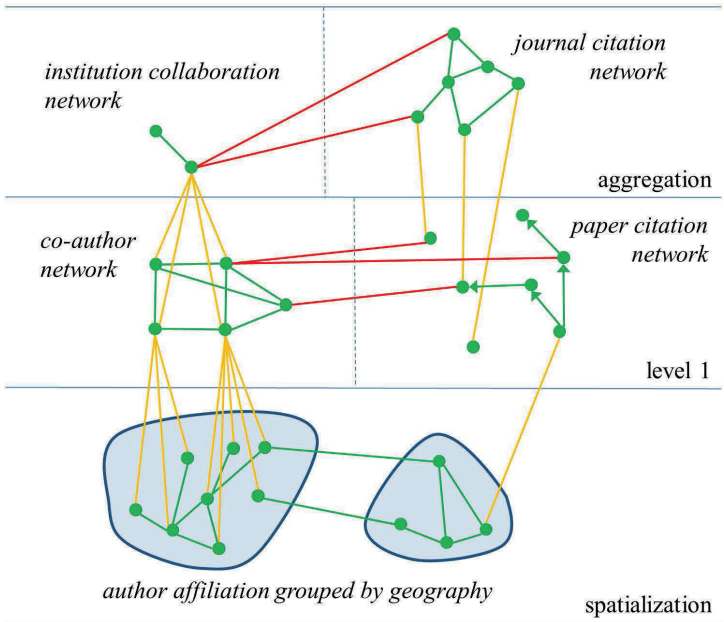



Figure 2.13: Example of a three-level network of different collaboration and citation networks. Image taken from [137]. ©Springer International Publishing Switzerland 2014.

### 2.3.1 Definition of Heterogeneous Networks

For the sake of completeness, this section gives a formal specification of heterogeneous networks which was taken from the work of Schreiber et al. [137]. Figure 2.14 shows a visualization of the data structure used in the following specification. Let $G = (V, E, L)$ be a labeled graph with a finite set of nodes $V = \{v_1, \ldots, v_n\}$, a finite set of edges $E = (v_i, v_j) | v_i, v_j \in V$, and a finite set of node and edge labels $L = l_1, \ldots, l_p$. Each node or edge of the graph is required to have a not necessarily unique label, and $l : V, E \to L$ gives the label for each node or edge. The label is used to encode a vector of additional node-/edge-specific data, i.e., the multivariate attributes.

Let $G_1, \ldots, G_m$ be a set of labeled graphs with $G_i = (V_i, E_i, L_i)$. Each graph may be directed, undirected, or mixed, and represents a specific network of the application domain (note that a graph might also be unconnected). For example, in the biological domain, a gene regulatory network may be represented by a directed graph, a protein interaction network by an undirected graph, and a metabolic network by a directed bipartite graph, respectively (cf. the green colored graphs in Figure 2.14). As mentioned at the beginning of this section, we allow networks arranged at var-

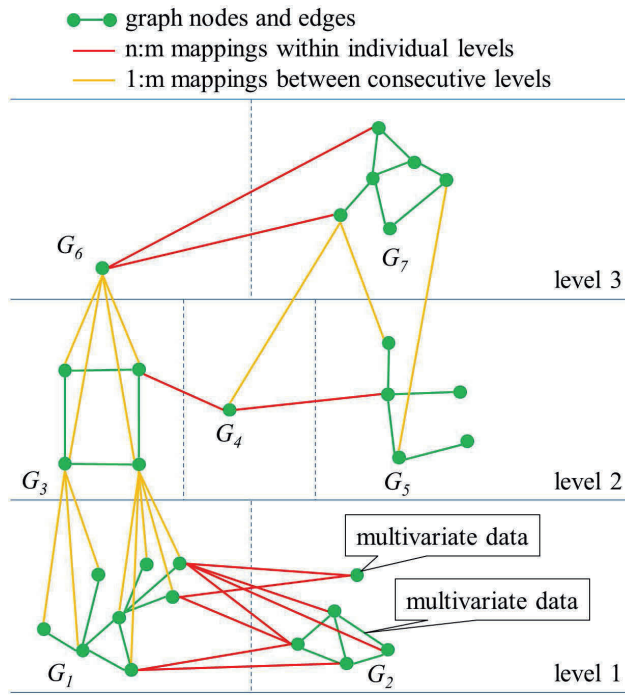



Figure 2.14: The data structure from the definition of heterogeneous networks (see Section 2.3.1). It shows three levels with two heterogeneous networks $G_1$, $G_2$ in level 1, three heterogeneous networks $G_3$, $G_4$, $G_5$ in level 2, and two heterogeneous networks $G_6$, $G_7$ in level 3. Image taken from [137]. ©Springer International Publishing Switzerland 2014.

ious levels. For modeling this property, let $S = \{s_1, \ldots, s_k\}$ be a set of consecutive levels. Each level can contain several graphs of $G_1, \ldots, G_m$, but each graph belongs to only one level. A level therefore groups graphs. The function $s : G, V, E \rightarrow S$ gives the level for each graph, node, or edge.

To connect graphs with each other, we introduce mappings between nodes of different graphs. Let $M = \{(v_i, v_j) | v_i \in V_i, v_j \in V_j\}$ be a mapping which connects a node from graph $G_i$ with one node from a different graph $G_j$. Note that mappings within a graph are not allowed (those "intragraph mappings" are represented by the normal edges). Furthermore, the resulting structure could be seen as a new (global or union) graph $G_G$ which merges all nodes and edges from $G_1, \ldots, G_m$ and the mappings (edges) $M$.

For simplicity, the mapping $M$ will be restricted in the following way. For $m = (v_i, v_j)$ with $v_i \in V_i$ and $v_j \in V_j$: if both nodes belong to the same level $s(v_i) = s(v_j)$ (see the red links in the figure) there will be no restriction. However, if both nodes

belong to different levels (yellow edges), a mapping is only allowed if the following two conditions hold:

1. both levels are consecutive (neighboring) levels, $s(v_i) = s(v_j) - 1$ or $s(v_j) = s(v_i) - 1$, and
2. there is a $1 : n$ mapping from the higher to the lower level, i.e., if $s(v_i)$ is the higher level ($s(v_i) = s(v_j) + 1$), then there is no other node $v_k$ in level $s(v_i)$ with $(v_k, v_j) \in M$.

### 2.3.2 Visualization and Navigation in (Heterogeneous) Networks

The visualization of links across related networks has so far mostly been addressed for biological pathways. Due to the sheer size of these networks, biologists usually create a number of smaller subsets, which are then analyzed using different techniques. Caleydo [108], for instance, shows networks side by side in a 2.5D environment (see Figure 2.15).

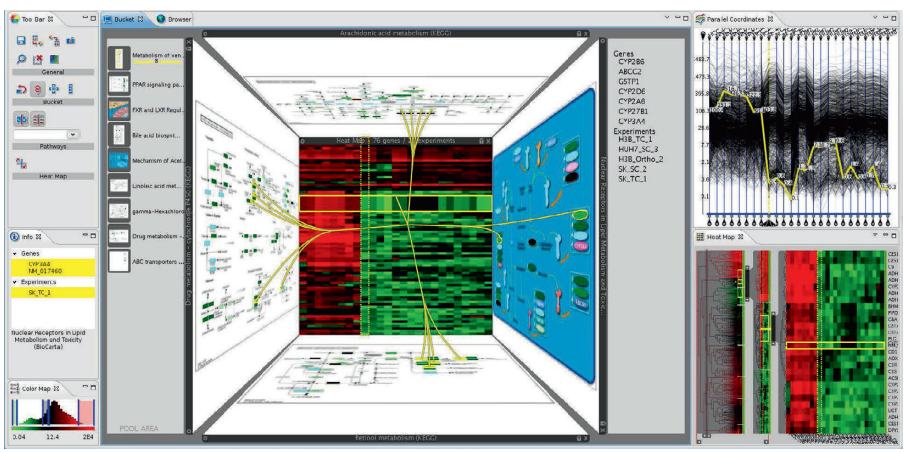


Figure 2.15: Screenshot of Caleydo with a 2.5D view that connects relations between networks with visual links, a parallel coordinates view, a heat map, and some meta-information [108]. ©IEEE.

Entourage [107] uses a focus+context approach to explore relationships in biological pathways. While analysts explore a pathway and focus on specific nodes, parts of related pathways are shown in additional views as contextual subsets (see Figure 2.16). Related to this approach and an extension of Caleydo is the work of Partl et al., who display associated experimental data next to a selected path [125]. An early example of a visualization tool for heterogeneous social networks was presented by Shen et al. [141]. They use semantic and structural abstraction to filter large networks and create a comprehensible network visualization.

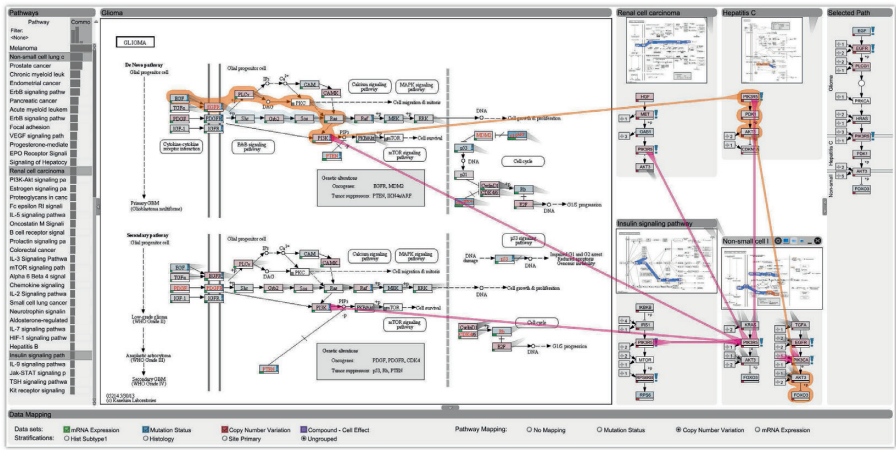


Figure 2.16: Entourage showing a biological pathway in detail and contextual information of multiple related pathways [107]. ©IEEE.

Exploring the connections in just one large network is often already a challenging task: navigating across links to adjacent nodes is difficult if the nodes are far away from each other and not in the current view. This usually requires a lot of panning and also zooming of the graph view and puts a lot of cognitive load on the user. To tackle this problem, May et al. [114] proposed signposts to provide cues about off-screen regions in a graph visualization (see Figure 2.17). As already discussed in the section about hybrid representations, Henry et al. [76] combine traditional node-link diagrams with adjacency matrices for a compact view on a network. The

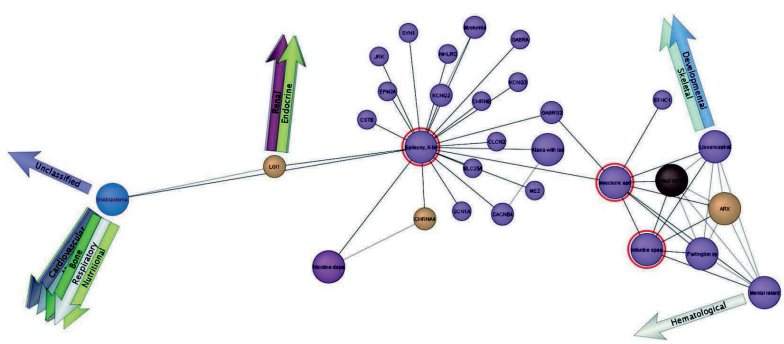



Figure 2.17: Navigation in a large graph using signposts to provide context for focused nodes [114]. ©The Eurographics Association and Blackwell Publishing Ltd.

links between these matrix representations can be considered as connections between subnetworks within one large network. Another approach from Moscovich et al. [119] introduced Link Sliding to automatically move the camera across links to adjacent nodes, and the Bring & Go technique which moves adjacent nodes into the current view, close to the selected node. Gladisch et al. [54] introduce enriched wedges at the corners of the screen of a node-link visualization to recommend interesting nodes (see Figure 2.18). A similar approach is used by Frisch et al. to visualize nodes, which are not in the current view, at the corners of the screen [47] There are also tools to assist users in creating and exploring multiple networks from raw tabular data, such as Orion [68] or Ploceus [111]. These tools use single node-link and matrix views to visualize the generated networks.

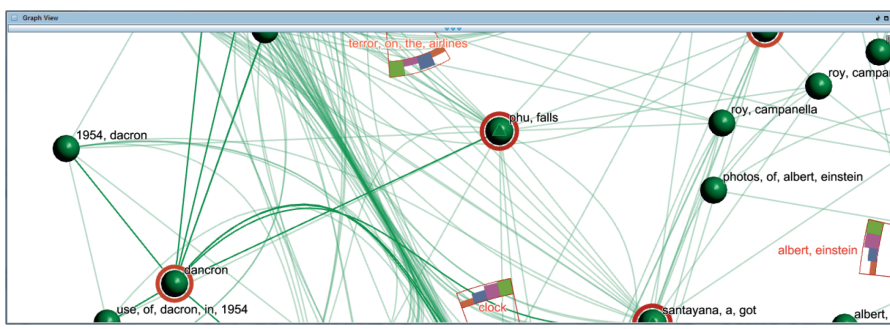



Figure 2.18: Enriched wedges indicate off-screen targets that might be of interest to the user. Image taken from [54]. ©Springer-Verlag Berlin Heidelberg 2013.

## 2.4 Guidance in Network Visualizations

The ever-growing amount of data available to users makes it difficult to maintain an overview and perform effective analyses. It is often difficult to decide which visualization method to use, or how to correctly set parameters to view the data. Another important point is how navigation takes place in larger data visualizations. How do users effectively move from one point in the data to another and how do they select the next point to look at? First solutions for the problem of navigation in the field of graph analysis have already been discussed in the previous section, but more flexible ways to get assistance while analyzing data are needed. When users are guided through the data to a certain degree by a system, this process is referred to as *guidance* in the literature, and there are already some tools and first attempts to create models for guidance in visualization and visual analytics [25, 138]. Since visualization systems are often used by different users, it is important for such systems to react flexibly to the behavior and experience of the users and to create dynamic suggestions for further navigation in the data depending on the task at hand.

Ceneda et al. [25] characterize guidance in terms of four main aspects: knowledge gap, input and output, as well as guidance degree (see Figure 2.19). These aspects are shortly discussed in the following (details can be found in the work of Ceneda et al.).

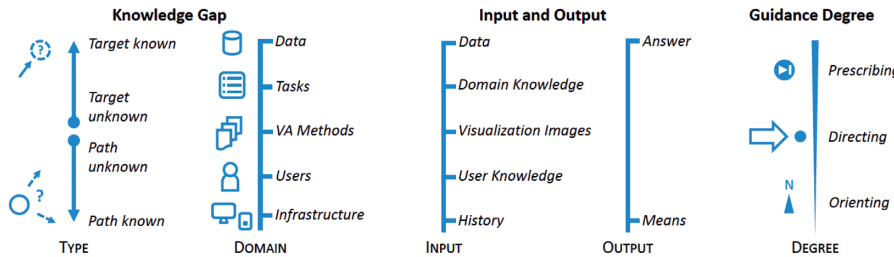



Figure 2.19: Terms of the main aspect of guidance: knowledge gap, input and output, and guidance degree. Image taken from [25]. ©IEEE.

Knowledge gap stands for the question: *What does the user need to know to make progress?* and is divided into two types, where the *target is unknown* and the user does not know the desired result, or where the *path is unknown*, which means that the user does not know how to reach the desired result. Another aspect of knowledge gap is the *domain* to which it belongs. Ceneda et al. describe five domains that are relevant for guided visual analytics:

- The *data* domain describes cases where users need guidance in which subset or feature they could look at next and is usually based on some kind of "interestingness" definition such as degree-of-interest (DOI) functions.
- The *task* domain, where users need assistance in structuring a goal into a series of tasks to solve the goal.
- The domain of *Visual Analytics Methods*, where the user needs help with the available visual analytical and interactive methods. Guidance could suggest visualization techniques or parameters for algorithms to assist the users.
- The *users* domain is important when analysts with different expertise collaborate together. Guidance could provide advice which analyst would be suitable to work on specific tasks.
- The *infrastructure* domain assists users on which hardware and software to use.

The input is based on the question about the basis for generating guidance. And the output is concerned with what the answer to a user's problem is and how it should be presented. The main aspect here is the input, as it builds the basis upon which

guidance is generated. Ceneda et al. identify the input categories: *data, domain knowledge, visualization images, user knowledge, and history*. *Data* usually refers to already available sources, such as the data set itself or statistical properties of the data. *Domain knowledge* uses information about the specific application problem, whereas *visualization images* represents information about how the data is presented to the user. The *user* is also source for guidance, as users typically input additional data into the system, such as annotations or DOI functions. Finally, *history* stands for tracking user input, such as interaction steps, parameters, or which part of the data the user has visited.
The last important aspect is the *guidance degree*, which concerns the question of how much guidance is provided by a system. Does the system only provide guidance if the user asks for it, should it automatically give suggestions, or should it even change aspects of the visualization without asking the user first? Ceneda et al. describe three degrees of guidance:

- *Orienting:* Provides basic orientation to build and maintain the user's mental map. Examples include showing connections between different parts of data or pointing the user into the right direction with highlights/markers.

- *Directing:* Here, the user is provided with a set of different options to choose from, such as preview techniques that guide the user to a new part of a data set to gain more insight, or navigation recommendations that the user can click/interact with, to move the camera to a specific part of the data set.

- *Prescribing:* This degree represents a largely automated process, where the user is merely watching a visual presentation with automated camera movements and stops, rewinds, or resumes it to gain knowledge.

We classify the guidance level of OnGraX based on a publication of Ceneda et al. [26] (see also Figure 2.20). Since OnGraX provides the user with a number of suggested nodes, with a priorization based on the DOI results, we classify our guidance approach as *directed guidance*. The aim is to support users in navigating between different heterogeneous networks and data sets. For this purpose, a 1:1 mapping between previously defined attributes of the networks and data sets is performed based on a focus node marked by the user. Details are discussed in Chapter 5. The next step, described in Chapter 6, includes extended guidance functionality which is based on multiple attributes and DOI functions that are interactively created by users. For this purpose, the next subsection gives a brief overlook about DOI functions in network visualizations.

### 2.4.1 Degree-of-Interest (DOI) Functions

The concept of DOI functions for trees was introduced by G. W. Furnas [49] and later applied and extended by van Ham and Perer to general graphs [151]. The

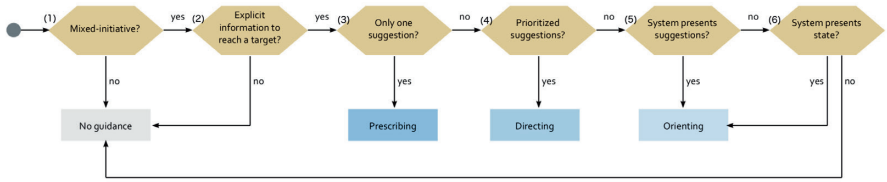



Figure 2.20: A decision tree to illustrate the key differences between guidance and plain visual approaches. Image taken from [26]. ©The Eurographics Association.

general idea is that information items have different levels of importance to different users, depending on their viewpoint. A user's initial point of interest (or focus) can be used to compute a numerical value for each data item that indicates its degree-of-interest (DOI). These DOI values are then used to display extracts of the full data, by only displaying items above a certain "interestingness" threshold. Van Ham and Perer used user data for their DOI function in the form of search entries for text and certain attribute values that apply to nodes in the graph. DOI functions have been adapted and extended for various use cases in recent years. Gladisch et al. [54] extend the DOI function around a *KNOW* component, which takes into account whether a node has already been explored by the user.
Kairam et al. [92] also use a DOI function to allow users to browse through heterogeneous networks. Their system shows views of subgraphs that are relevant to a user query in a single view (see Figure 2.21), but does not visualize the DOI calculations for verification. Müller et al. [121] present another DOI function for graphs with multiple types of entities. They also visualize an interesting subgraph around a focal node and offer additional information about possible further steps during the exploration process (see Figure 2.22). Users can interactively set interests in different entity types and entity relations. Abello et al. [1] presented a modular DOI specification to explore large dynamic networks and also support users in adapting and changing the DOI function (see Figure 2.23). This idea is similar to our design, which is discussed in Chapter 6. We plan to extend this idea by designing visualizations that enable analysts to review and analyze the results of a DOI calculation across multiple heterogeneous graphs and additional data sets.

## 2.5 Collaboration and Provenance in Network Visualizations

Analysts often wish to collaborate for increasing the quality of the analysis result, as well as the speed of the analysis process. Isenberg et al. [82] give a good overview of definitions, tasks, and sample visualizations in the field of collabora-

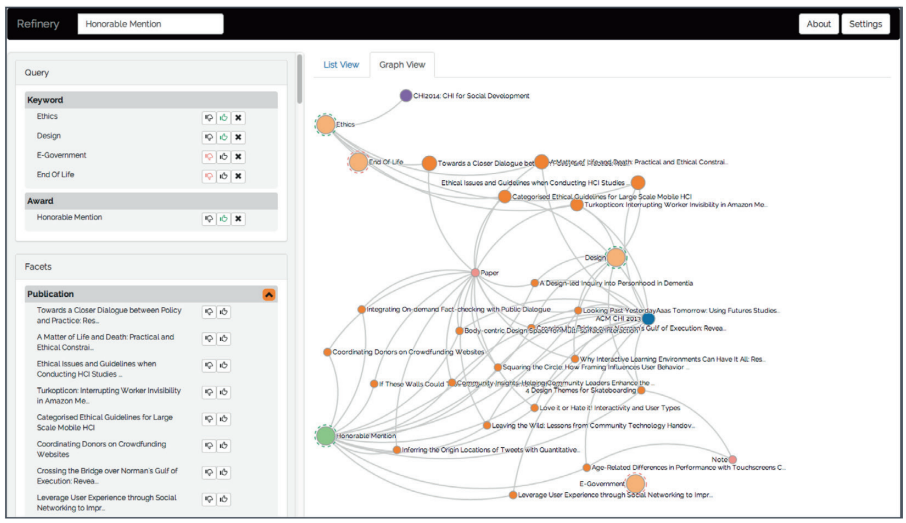



Figure 2.21: Refinery: The visualization is based on a user query and shows the most relevant items as subgraph. Image taken from [92]. ©IEEE.

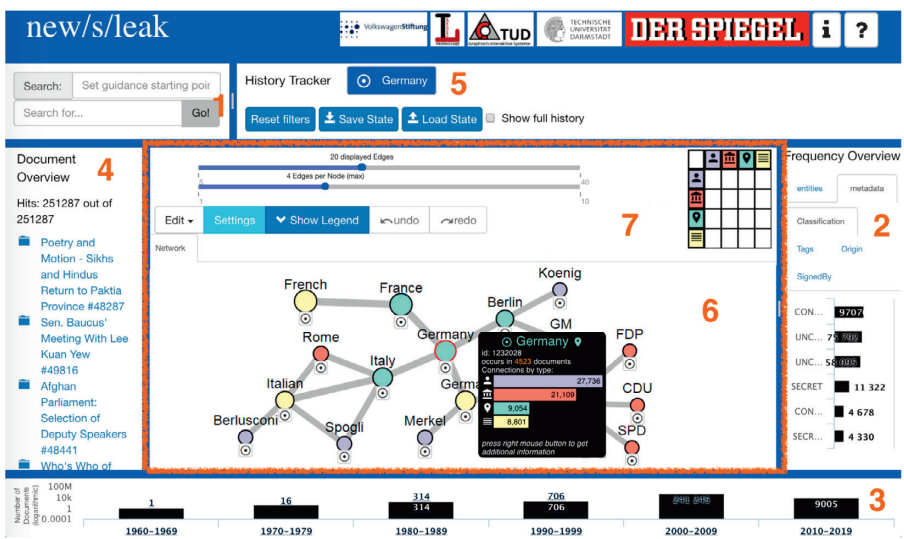



Figure 2.22: A system for guidance in multi-type entity graphs. A subgraph, based on the user's interest is shown around a user's focal node. The system informs the user about further possible exploration steps and further information. Image taken from [121]. ©The Eurographics Association.

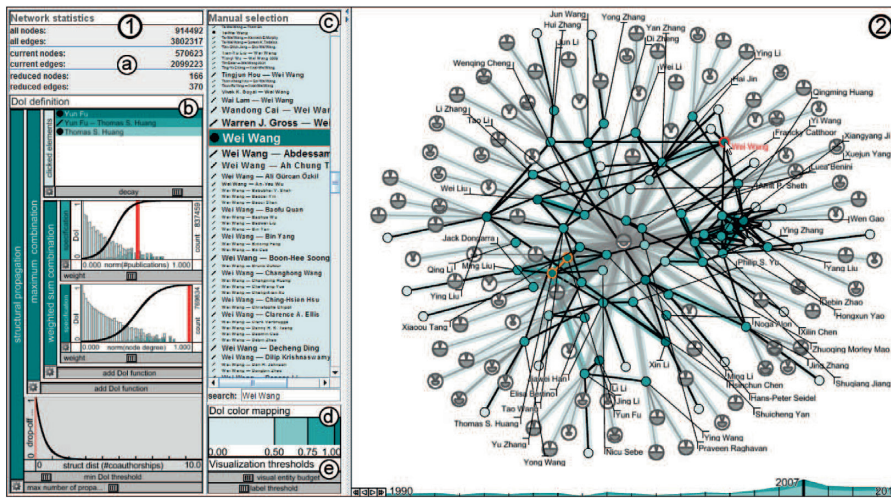



Figure 2.23: Modular DOI specification. DOI view (1) and network view (2). The DOI view gives details about the network, used DOI functions, and individual DOI values for the hovered node. Image taken from [1]. ©IEEE.

tive visualization. The authors define collaborative visualization as "the shared use of computer-supported, (interactive,) visual representations of data by more than one person with the common goal of contribution to joint information processing activities". They also provide an excellent summary of ongoing challenges in this field. We follow their terminology and classify our approach as a distributed, synchronous/asynchronous collaborative visualization method. In this thesis, we restrict ourselves to the collaborative visual analysis of networks. For further readings on general aspects of visualization, human-computer interaction, computer-supported collaborated work, and collaborative visualization of other data types, we refer to the standard literature, for instance [10, 39, 67, 70, 95, 100]. Please note that this list does not claim to be exhaustive.

The benefits of collaborative work were also discussed in an article on social navigation presented by Dieberger et al. [38]. Being able to see the usage history and annotations of former users might help analysts to filter and find relevant information more quickly. In order to be able to work together during a synchronous session, users have to know each other's interactions and views on the data set, usually referred to as "common ground" [31, 67].

There are also existing groupware frameworks and libraries to handle collaborative editing in the web with concurrency control, such as ShareJS [51] or Apache Wave [46]. They usually center on manipulating the DOM of collaborative websites and editing text in online documents without creating conflicts. Our approach concentrates on collaboration and awareness in a node-link based graph visualiza-

tion which does not require to edit a lot of textual data, i.e., we focus on changing the structure and attributes of the graph instead. For this case, it suffices to use a traditional lock-based approach.

Building an understanding of the collaborative work in node-link visualizations is an interesting challenge and has not yet been discussed in the literature. The first techniques for supporting provenance involved the collection of everything that happened during an analysis session with the intention of somehow making sense of this data later. However, the recorded interaction data quickly grows in size and becomes sometimes more complex than the original data set. Some tools address this issue by reducing the complexity of the interaction history and by filtering unnecessary views [70]. Revisiting old snapshots of an analytic session or replaying every single event is usually not sufficient enough to reveal the same insights that an analyst might have had during the initial analysis. Providing analysts to add *data aware annotations* [69] to parts of a data set during the sensemaking process is a good method to discuss insights, ideas or questions with other users. Tools such as ManyEyes [154] or Sense.us [71] address this issue, however they focus on the visualization of data sets with standard diagrams and do not support annotations in more complex node-link visualizations, which we see as one of the existing challenges in distributed collaborative graph visualization.

To find a common ground in node-link visualizations, we apply the techniques from the work of Gutwin and Greenberg [60]. They used secondary viewports and radar views to indicate other users' view areas and mouse cursor positions. We use a similar approach and show the viewports of other users as rectangles in the background of the graph visualization. Another work by Isenberg et al. [83] introduced the concept of collaborative brushing and linking, which "allows users to communicate implicitly, by sharing activities and progress between visualizations". The authors considered sharing activities during synchronous collaborations on a tabletop visualization for document collections. We adapt the concept and utilize it in node-link diagrams with the help of a heat map visualization (see Chapter 4) for the exploration of interaction information in both asynchronous and synchronous distributed sessions.

The special problem that arises during the distributed analysis of graphs is that the topology of a graph is independent to its layout. Analysts might change the layout drastically during the analysis process, which complicates the task of keeping track of the graph objects and areas that users were most interested in. Adding tools for subsequent analysts to be able to quickly review the previous collaborators' interests is another interesting challenge. It could also be useful to track changes that were performed on graph objects by others, such as renaming a node or changing its attributes. Additionally, annotations pinned to specific graph objects or regions of a graph could loose their context when the layout or the structure of a graph changes. In case of graphs that change their topology during the analysis process (referred to as dynamic graphs in the following), single nodes or complete graph regions could be deleted from a graph, rendering old user annotations useless without the

possibility to view them in their historical context. This would require to either having to save a snapshot of the graph for each user annotation or to track all changes performed on the graph layout and structure to be able to undo them on the fly and revisit old graph states and their related annotations. Providing tools for the analysis of dynamic graphs in a *distributed collaborative* environment proves to be even more complex. Should the users who work at the same time in a *synchronous* session always have the same graph layout as the others or should they be able to create their own views? How would other users in such a case be able to follow what their collaborators are seeing, and how would it be possible for subsequent analysts to follow the original sensemaking process of multiple collaborating users?
We summarize these questions and our challenges of sensemaking and provenance in distributed collaborative node-link visualizations as follows:

1. Give users tools to discuss and review insights in dynamic node-link diagrams.
2. Get an overview of graph objects and regions that users were interested in.
3. Review graph changes performed by former users.
4. Follow the analysis processes of other users while they look at a different layout of the same graph.
5. Review the sensemaking process of multiple users that worked collaboratively with different views of the same graph.

These challenges are addressed in the next two chapters. Chapter 3 describes the architecture and implementation details of our Online Graph Exploration system OnGraX, whereas Chapter 4 discusses collaborative work on node-link visualizations with our system.

*Chapter 3*

# OnGraX – the Online Graph Exploration System

Existing graph visualization libraries for the web do not support the special requirements for our goals and use cases, such as tracking of user actions and viewed graph objects. Furthermore, most libraries use an svg-rendering approach for graphs, which stores the svg elements in the DOM [159] of the browser. This technique is sufficient for smaller networks with less than 1,000 nodes—depending on the computers hardware. To reach the goals in this work, a faster approach is needed, as we want to visualize networks with thousands of nodes and edges. Furthermore, we need features for distributed collaborative graph exploration and the possibility to gather and visualize provenance information of our users. This chapter is based on the publication [173]. It describes the technical requirements and gives a detailed description for each of the components of our Online Graph Exploration System OnGraX.

## Contents


**3.1 Technical Requirements . . . . . 36**
**3.2 General System Overview . . . . . 37**
**3.3 OnGraX – Server . . . . . 39**
3.3.1 MySQL and Neo4j Databases . . . . . 39
3.3.2 Authentication and User Sessions . . . . . 41
**3.4 OnGraX – Web Client . . . . . 41**
3.4.1 Action and Conflict Handling . . . . . 42
3.4.2 Increased Performance with WebGL . . . . . 43
3.4.3 WebGL Graph Rendering . . . . . 44
**3.5 OnGraX – Data Set Server . . . . . 45**
**3.6 Visualization of Graphs from External Applications with OnGraX . . . . . 45**
**3.7 Discussion . . . . . 46**
**3.8 Summary . . . . . 48**

## 3.1 Technical Requirements

Our goal was to design a single central system for analyzing complex graphs in distributed collaborative sessions, instead of various small prototype systems for each of our planned and future use cases in the field of heterogeneous network visualization. The initial process of beginning a collaboration should be as fast and easy as possible, thus the tool should also be available on the fly without requiring users to install specific software, plug-ins, or Java applets first. Additionally, users should be able to drop in and out of ongoing collaborative work without having to setup and plan each session individually—experts would like to work on a data set whenever they find the time and do not want to wait for others to join before they can start their analysis process.
In this case, users who are joining an already ongoing session should be able to quickly catch up on what has been done before. Hence, the system should support logging all actions that are performed in a session and provide a way to quickly retrieve and analyze them to show important information to subsequent users. These actions include not only changes performed on a graph (such as deleting or changing the attributes of a node), but also the tracking and logging of every user's camera position. This assists subsequent analysts to identify regions of a graph that other users were already interested in if they work on the data set asynchronously. During a synchronous session, however, changes applied to the graph should be distributed to all connected clients as fast as possible, and users should be able to track each other's mouse and camera positions in real time to establish a common ground and be aware of everything that is going on in a session. The system also has to store the complete graph structure and additional graph attributes in an efficient way. And, it should handle conflicts that could arise during a collaborative session if two users want to change the same object in the graph at the same time.
In addition to the graph structure itself, nodes usually hold attributes (e.g., age, gender, or income for social networks). Rendering the nodes as simple dots is therefore not sufficient enough for our application areas since the shape, size, and color of nodes may have specific meanings; nodes may also have labels attached. Thus, our system should be able to render graphs with a considerable number of nodes and edges of different shapes and sizes with additional text labels on every node in an acceptable frame rate. Moreover, running computationally expensive tasks—such as calculating the layout of a graph, computing additional graph metrics, or aggregating the camera positions of users to show interesting regions—should not have a negative impact on the rendering performance on the clients. We summarize the technical requirements (TRs) of our tool as follows:

TR 1. It should be possible to start a collaborative session on the fly without having to install software or plug-ins first.

TR 2. Most of the graphs from our use cases consist of up to 10.000 nodes and edges. As such, the system should be able to render these graphs with a good performance on standard computers.

TR 3. The system should be able to render nodes in various shapes and colors, with additional text labels on every node.

TR 4. Changes to the graphs and mouse positions as well as viewports of other users should be distributed to all connected clients in (soft) real time.

TR 5. Computationally expensive processes, such as calculating the layout of a graph, should not interfere with the rendering performance of the graph visualization.

TR 6. The server should efficiently store the graph structure as well as the complete action history of a graph session in order to use this data for subsequent analysis processes.

TR 7. The system should be able to manage concurrent graph changes during a collaborative session.

TR 8. As the most graphs come with node and/or edge labels, the system should support fast and efficient text rendering.

## 3.2 General System Overview

We decided to implement OnGraX as a web-based tool to address our first technical requirement (TR 1). OnGraX' client/server-based architecture enables us to provide an easily accessible tool for graph exploration over the web which allows analysts to simply open a web browser to start a collaborative session. Figure 3.1 illustrates the basic architecture of our system. On the client side, all graphs are rendered with the help of the client computer's GPU by using WebGL [59], a JavaScript API for rendering 3D graphics natively in modern web browsers. For our special case, this is faster than using SVG-based node-link visualization approaches, such as the d3.js visualization library [116]. To ease the process of low-level OpenGL programming, we utilize the JavaScript library three.js [129]. This approach tackles the second and third technical requirements (TR 2-3) and enables us to achieve a high-performance rendering of node-link diagrams.
The server side of OnGraX is implemented in Java EE and currently runs as a web application on a standard Apache Tomcat 8 server installation. Communication between server and clients is done with the relatively new WebSocket protocol [160]. It suffers less from network overhead since it is layered over TCP instead of HTTP which removes the overhead from HTTP header fields and allows for a

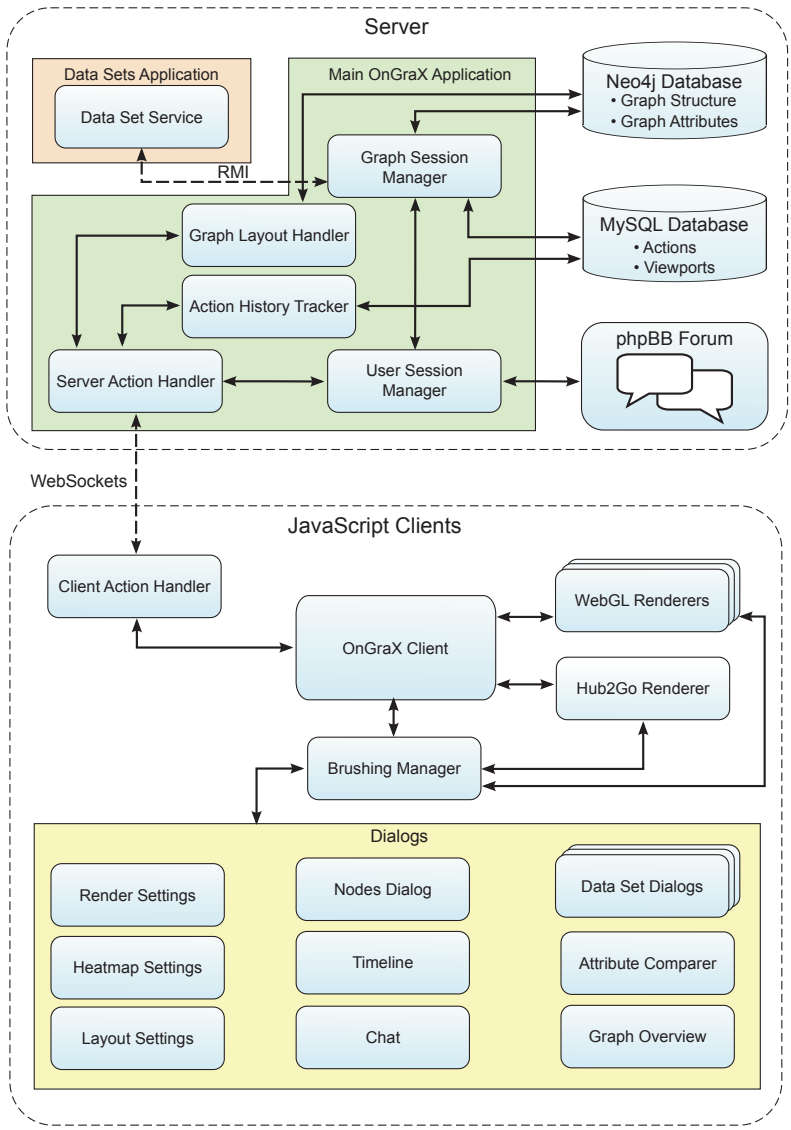



Figure 3.1: General architecture of our system. Graph sessions are initialized on demand whenever a user joins a graph analysis session. The client-server communication is done via WebSockets. The server stores each graph in a separate Neo4j database, whereas all performed actions of a session are logged in a dedicated MySQL table. The main components of the JavaScript Client are the WebGL renderer for the graph visualization, the Hub2Go renderer to allow the visualization of attribute matches across heterogeneous networks, and the brushing manager to handle user interactions across OnGraX' various dialogs.

low latency two-way communication. Clients do not have to poll the server in regular intervals anymore to ask for updates. Instead, the server can send a message to all connected clients whenever an update on the client side is required. This allows our system to track the viewports and mouse positions of other users and distribute this information among all clients in real time and addresses our fourth technical requirement (TR 4).

## 3.3 OnGraX – Server


Technical requirement number five is addressed by using the server side part of our system for complex processes (TR 5). The Tomcat server currently runs on a Dell PowerEdge R720 with two Intel Xeon E5-2650 2.00GHz processors (eight cores each), 128GB RAM, and a Value MLC 3G SSD hard drive. This configuration provides more than enough computing power for our current purpose and future extensions like the calculation of complex graph layouts or running graph analysis algorithms on the server and distributing the results to all clients. Many graphs analyzed with our tool already have precomputed layouts, but computing a layout for other graphs is also possible. OnGraX uses the yFiles for Java graph layout library [166], and the import of graphs is possible via the graphML specification [20].

### 3.3.1 MySQL and Neo4j Databases

The server stores each graph in a separate Neo4j [122] database—the default configuration puts all databases in the directory *usr/local/ongrax2_graphs*. As soon as a user joins a graph session, the respective Neo4j database is initialized as an embedded database service. As a graph-based database, Neo4j offers a convenient way to store our graphs together with all of their node/edge attributes and supports graph-like queries, such as shortest path calculations. It also simplifies other graph-related queries for community detection or applying clustering algorithms. For the remaining data, such as user data and login information, our system uses a MySQL database in conjunction with a phpBB forum installation. Figure 3.2 shows the most important MySQL tables of OnGraX. For each imported graph, a new *ongrax_graphActions* table is created, which stores all performed actions and all camera positions that are generated by the users while they are exploring the graph. This data can be used later to calculate and visualize regions of a graph that were modified or viewed by other users. The graph structure as well as all node and edge attributes are stored in the Neo4j database, whereas all meta-attributes, such as the

mapping of the attributes to node size, shape, or color, are saved in the MySQL table *ongrax_graphs*. This table also has the references to the Neo4j database on the file system and other MySQL tables to track ongoing user sessions and read/write permissions for each graph. Using Neo4j to efficiently store the graph structure in conjunction with a MySQL database to log all events enables us to tackle our sixth technical requirement (TR 6).

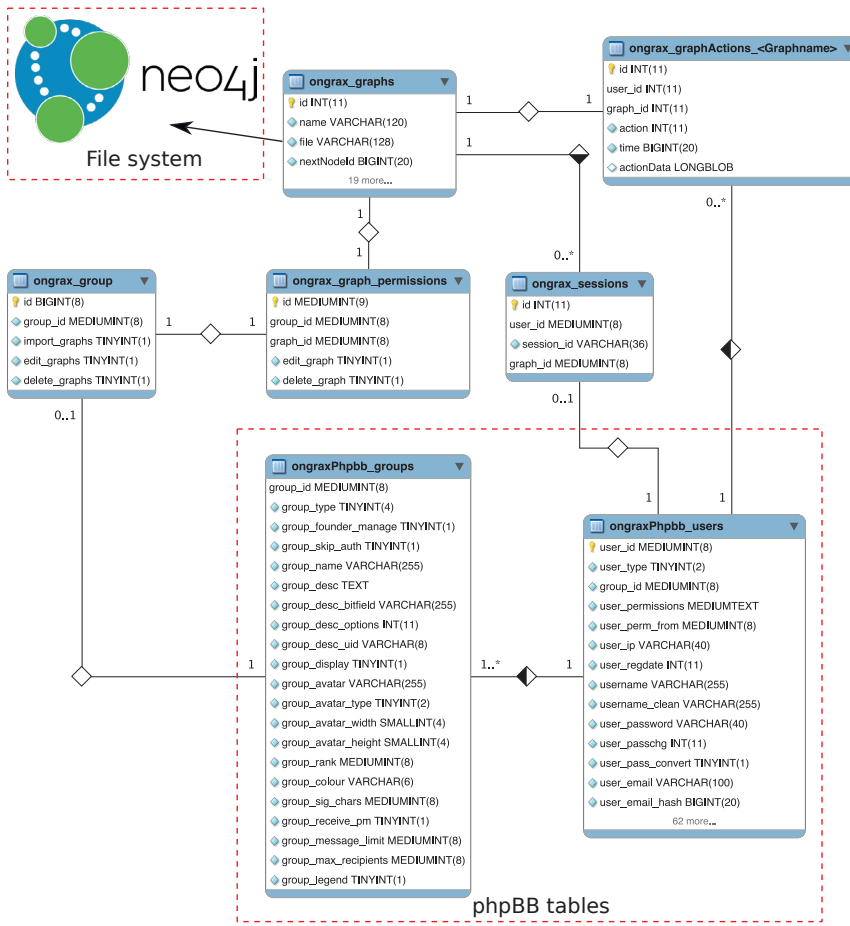



Figure 3.2: OnGraX' database schema with connections to the Neo4j graph databases and the tables of a phpBB forum, which is used for user registration, authentication, and group management to limit access to graphs for certain user groups.

### 3.3.2 Authentication and User Sessions

OnGraX uses existing functionality of a phpBB forum installation for user registration and authentication. To prevent vandalistic behavior, all users have to create a forum account first, which can then be used as login credentials in the OnGraX web-application. The phpBB forum already ships with group management functionality, which is used by OnGraX to grant different access levels to imported graphs. Groups can be limited to only have read access to a graph, to enable guests or less experienced users the exploration of a graph, without the possibility to edit the graph structure or attributes. Newly registered users are put into the default members group of the forum and are only able to explore graphs that were added to this group. A graph can be added to multiple groups. Each graph can only have one graph session; upon joining an already opened graph, a user is added to the respective graph session and will be able to collaborate (e.g., chat, follow other mouse cursors and camera positions, and read/add annotations, see Chapter 4) with all other users who also opened this graph. OnGraX also supports the exploration of multiple graphs at once, which is discussed in Chapter 5.

## *3.4 OnGraX – Web Client*


The OnGraX Web Client can be used in three different modes. These modes are adapted to the different tasks for which OnGraX is frequently used.

**Mode 1** Collaborative analysis and adaptation of graphs with a precomputed layout.

**Mode 2** Bottom-up exploration of multiple graphs with a dynamically calculated layout.

**Mode 3** Bottom-up exploration of graphs from an external application, with automatic login through the external application.

The first mode, which is discussed in Chapter 4, is intended for the collaborative analysis of biological graphs with an existing layout. In most cases, these layouts were created manually (which also includes the size, shape and color of nodes) and are only slightly adjusted during an analysis process. In addition, the number of nodes and edges is limited—these graphs consist of up to 10,000 nodes and edges—and can easily be displayed and analyzed in their entirety. OnGraX loads the entire

graph in this mode and also records the camera positions of all users during the analysis processes, so that they can be used afterwards to highlight areas that were viewed by certain or by all users. To facilitate the collaborative exploration and analysis of these graphs, users also see each others camera and mouse positions in this mode.
The second mode uses a bottom-up approach, where the analysis is started with a search query on a few specific nodes and more context is added afterwards as needed [151, 156]. This approach is suitable for larger graphs and graphs with a high number of edges. Since these graphs do not have a predefined layout, the layout of the graphs in this mode is calculated on the fly when new nodes are added. In this mode, OnGraX first displays a list of all nodes of the graph and also offers the possibility to search for specific node names. Nodes can be added from the list, along with their nearest $n$ neighbors. Whereby $n$ can be chosen freely. Recording camera positions and displaying other users is disabled in this mode, because tracking each view through the dynamic analysis process, where nodes are often added and removed again, would be too costly and is furthermore not relevant to our use case, which is explained in Chapter 5.
The third mode also uses a bottom-up approach and is used to automatically load and display graphs from external applications into OnGraX without users having to register and log in to OnGraX beforehand. This mode is explained in Section 3.6 of this chapter.

### 3.4.1 Action and Conflict Handling

Since OnGraX focuses on the collaborative exploration of graphs and not on the collaborative editing of text documents, we do not need sophisticated concurrency control systems which are usually used in such a case (e.g., operational transformation [42]). A pessimistic locking approach is sufficient enough for our scenario, since changes are usually performed directly on one or a couple of single nodes and node attributes. To address our seventh technical requirement (TR 7), we use a simple server-side queue to ensure that all clients visualize the same data structure. Whenever a user performs an action that would change the graph structure or any of the graph object's attributes—such as moving a node, changing the shape of a node, or adding a new edge between two nodes—an action request event is sent to the server. The server uses Neo4j's transaction system to open a new transaction and apply the changes to the stored graph. Only if the transaction is successful, the server reports this back to all connected clients, including the client who initiated the action. Figure 3.3 illustrates this approach. All incoming actions are handled in a server-side queue, and an event is only applied if it is not in conflict with a previous event. This could happen if a user deletes a node, while a second user tries to add an edge to this node at exactly the same time. If the server processes the node delete action first, the second user's action will not be performed on the graph structure, and the user will receive a short notification instead, while his/her local

graph visualization is updated to reflect the changes that where performed by the first user.

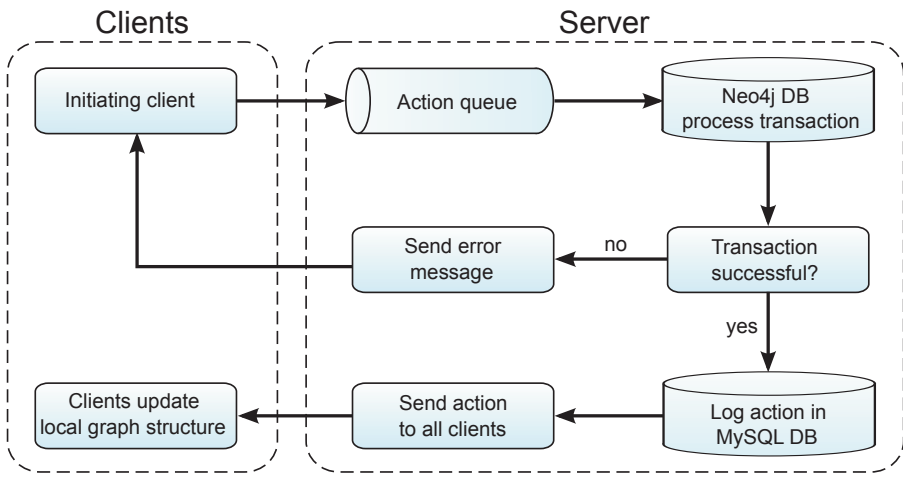



Figure 3.3: Action handling between clients during a graph session. To avoid concurrent changes, actions that would affect the graph structure are sent as an action request event to a queue on the server. All clients only update their local graph visualization, if the transaction was applied successfully to the Neo4j database. In case of a conflict, the action is not applied, and the initiating client receives an error message.

### 3.4.2 Increased Performance with WebGL

Upon joining a graph session, the complete graph structure is transferred to the client. This approach is fast enough for graphs with up to 10,000 nodes and edges. Depending on the connection and hardware, the download and initialization of all graphical objects on the client takes up to twenty seconds. Unfortunately, this approach does not work for graphs with more than 10,000 nodes as it takes simply too long to transfer the whole data set to the client and generate all graphical objects in one single step. This may eventually lead to an error message in the client's browser stating that JavaScript is not responding anymore. Streaming the graph to the client by only sending an initial part and expanding the information as soon the user zooms out or further explores the graph would be a more convenient step and is planned for a future version of our tool. Picking up the idea of Gretarsson et al. [58], it would also be possible to render parts of the graph on the server and send them as images to the client where they are shown until the complete data is loaded.
After the initialization phase, our tool renders a zoomed-out overview of our test graphs—which usually contain about 3,000 labeled nodes and edges in typical ap-

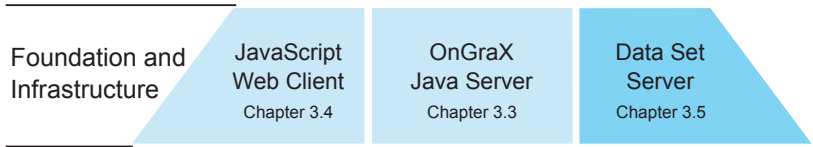
plication scenarios—with 20 frames per second and 30-40 frames per second in a standard zoomed-in view on an early 2011 MacBook Pro (2,2GHz i7, 8GB memory, Radeon HD 6750M with 1,024MB video memory) with a screen resolution of 1,680×1,050 pixels. We also did a performance test with the same computer on a 4K monitor with a resolution of 3,840×2,160 pixels. Here, the overview is rendered with 10 frames per second and 20-28 frames per second for the zoomed-in view. The biggest graph visualized by with our system by now had 10,000 nodes and 7,500 edges. The initial data transfer while joining the session took about ten seconds. While rendering the complete overview of the graph was only possible with five frames per second, the standard zoom level of our users while working within the graph was rendered at around 10 to 15 frames per second with a 1,680×1,050 resolution. While this is not an optimal frame rate, the visualization still proved to be useable by our test users. Increasing the performance for bigger graphs is one of the next planned steps in our future work as discussed in the following section.

### 3.4.3 WebGL Graph Rendering

Utilizing a custom WebGL renderer instead of already existing graph rendering libraries which are mostly using a slower svg approach has various advantages. OnGraX' renderer is able to visualize thousands of nodes in different shapes and colors. The rendering approach could also be improved in future versions. Currently, a new GL-geometry and a new GL-material is created for every single node shape in the graph. Common reoccurring shapes could be buffered and reused, which would increase the rendering performance. The renderer also only renders the graph if something in the graph view has changes since the last rendering of the graph (for instance if a node is deleted or moved to another position, or if the user zooms in or out).

Drawing a large number of node labels was another challenge and requirement (TR 8) that we had to solve. The most convenient way—which can also be found in three.js forums—is to draw each node label on an offscreen canvas, render this canvas to an image and use this image as a sprite texture. This works well for a small amount of text strings. But for graphs with more than just a couple of hundred nodes, the JavaScript engine would have to create and render a large amount of textures, which is not fast enough and will eventually crash the JavaScript engine. To solve this, we adapted and modified the JavaScript-based solution from an online article of Heikkinen [72] for bitmap fonts, a common technique used in standard OpenGL rendering. Instead of creating a texture for each label, we just use one texture with all required letters on it. For each letter in a node label, two triangles are drawn and only the part of the texture with the position of the letter is mapped to this geometry. This is an extremely fast solution, and the only drawback is that the fonts do not scale nicely at very high zoom levels.

## 3.5 OnGraX – Data Set Server




OnGraX supports not only the analysis and exploration of graphs, but also the analysis of additional attributes that are not directly stored in the node and edge attributes of the graphs. Moreover, these attributes do not necessarily have to be assignable to a single specific graph. As an example, our application in Chapter 5 uses lists containing Bag-of-Words (BoW), a word importance index, and an additional word similarity matrix (details can be found in Chapter 5.3). These data sets are often very large (in our case about 3 gigabytes) and therefore require a lot of storage space. OnGraX must be able to access these attributes quickly so that the interactive matching between different graphs and data sets can be calculated and displayed in a few milliseconds. For this purpose, the required data records are currently loaded completely into the main memory of the server. In case the used server does not have enough storage capacity or is already under heavy load by other applications, the loading and searching of these data sets was outsourced to an independent data set server, which communicates with OnGraX via *remote method invocation* (RMI). Through this approach, several data sets can be distributed on different servers and accessed by OnGraX. The only prerequisite for this is that the network connection between OnGraX and the additional servers is fast enough to maintain the high interactivity of the attribute matching. Another advantage of this approach is that the large data sets do not have to be constantly reloaded during further development and updates of OnGraX or if an update requires a restart of the Tomcat server.
Currently, OnGraX supports loading lists and matrices (see Figure 3.4) as CSV files, whose attributes can then be mapped to attributes of other data sets or graph attributes. If a data set server is installed on a Tomcat server, it automatically searches for files in the folder */usr/local/ongrax_data/* and loads suitable files into the main memory. OnGraX configuration has to be adapted manually to include the configuration for each data set server. Right now, this is done directly in the source code but could be adapted to an xml-file configuration, for instance.

## 3.6 Visualization of Graphs from External Applications with OnGraX

OnGraX also offers the option to programmatically import and visualize graphs from external systems. We used this option to enable collaborators from the Center for Bioinformatics at Saarland University, Germany, to visualize data from their

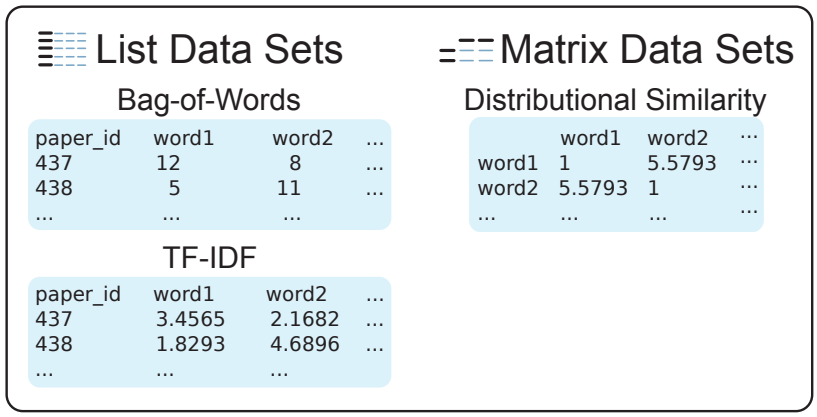



Figure 3.4: Structure of supported files for the import of additional data sets. The example shown here is for a specific use case in text analysis, which is discussed in Chapter 5.

GeneTrail2 system [9, 27]. Figure 3.5 shows the process to automatically login an application to OnGraX and upload and open a graph in the web browser. GeneTrail2 uses a special user account for the login and a session ID, which can be used for uploading a graph and for further communication. After the graph has been successfully imported into OnGraX, GeneTrail2 can open the graph directly via a special URL, so that the user no longer has to log on to the system. This mode uses the same bottomup approach as OnGraX' second mode, to allow users to manually search for nodes of interest and add them to the graph visualization. After a session is finished and the user closes the browser tab, OnGraX deletes the graph and all related data. In addition, no user input is logged in this external application mode, and it is not possible to analyze the graph with other users at the same time. If users want to work collaboratively, they can always fall back to OnGraX' main functionality and import the graph with a standard user account.

## 3.7 Discussion

We made our tool available to various experts in systems biology and bioinformatics at Monash University and University of Melbourne, Australia, and received mostly positive and also constructive feedback. They liked the idea of working together on their data sets by simply opening the visualization in a browser window. Seeing each others camera position in a synchronous session made it a lot easier for them to discuss and change specific parts of the graph, although one group of experts missed a voice chat directly in the tool. They would have preferred to be able to talk to each other directly without having to fall back to other programs, such as Skype

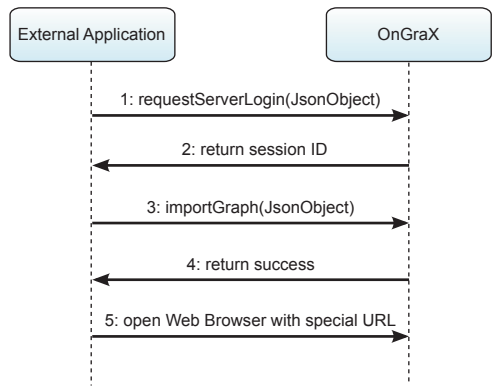



Figure 3.5: Sequence of loading and visualizing a graph with OnGraX from an external application. The application requests a login from OnGraX and gets a session ID, which is used for all further communication. If the graph is imported successfully into OnGraX, the application can open OnGraX with a special URL to visualize the Graph in external application mode.

or Google Hangouts. This could be addressed in a future version of our tool with the help of the new WebRTC standard [162] for real-time communications in browsers. Another group of biologists found the heat map visualization of user behavioral data quite useful. They would like to use OnGraX for the education of their students. A use case here would be to give students an already edited graph and ask them to revise the data set further as well as to verify the changes that have already been performed by previous users. Afterwards, the supervisors could review the steps that the students performed and also discuss and reflect the process online together with the students in a collaborative session.

There are still some technical issues that have to be addressed to improve the performance and usability of OnGraX. One task is to utilize Web Workers [161] to speed up the client-side part of the heat map generation. The user interface sometimes becomes unresponsive—depending on the client's hardware—for up to two seconds while the JavaScript engine generates the heat map texture. This caused a small inconvenience among some of our test users. Web Workers could be used to finish this calculation in a thread-like manner in the background of the web application, allowing the web page and user interface to remain responsive all the time.

In general, our tool is able to handle graphs with 10,000 nodes and edges on faster computers as we stated in [171], but some users with slower computers had performance issues during the exploration of graphs with more than 6,000 nodes. Using streaming techniques together with the idea of WiGis [58] to render parts of the

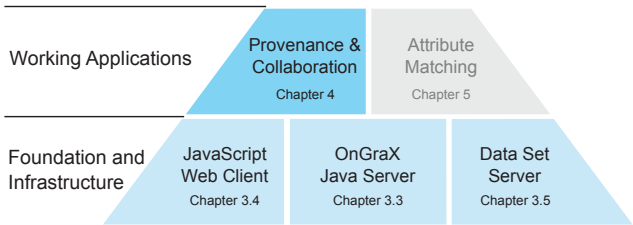
graph on the server and only show them as images on the clients would be another possibility to improve the rendering performance and initialization times. Another technique would be to still send the whole data to the clients, but to render specified areas of a graph only once into a texture instead of rendering all graph objects in every frame. Users could then manually select areas to be rendered as a texture and only switch to full rendering on demand. If a subgraph that is rendered as a texture for one user is changed on another client, the changes only have to be rendered once again into the texture on the first user's client to update the information. Adapting this technique would enable OnGraX to visualize even bigger graphs with more than 10,000 nodes.

## 3.8 Summary

This chapter presented the basic building blocks of OnGraX, our collaborative system for visualizing graphs with several thousands of nodes and edges in a web-based environment. By using WebGL, the system is able to provide an interactive graph visualization with up to 10,000 labeled nodes and edges. With the help of our client/server-based system, analysts do not have to install any additional applications or browser plug-ins anymore. The start of a collaborative analysis session is simply done by opening a URL in a browser window. The first three building blocks of this thesis, the *OnGraX Server*, *Web Client*, and *Data Set Server* together form a foundation for the realization of further functionalities for different use cases. If required, external applications can use OnGraX for fast visualization and exploration of graphs. Optional tracking of user actions allows us to provide various features for collaborative work on graphs. This collaborative aspect is discussed in more detail in the next chapter.

*Chapter 4*

# Collaborative Visual Analysis of Networks




This chapter discusses OnGraX' interactive visualization techniques to support coordinating work in a collaborative setting for node-link diagrams which may change their topology during the analysis process (referred to as dynamic graphs in the following). We also propose techniques to assist analysts to identify previous activities performed by former users on these networks. This chapter is based on the publications [172] and [174]. An accompanying video is available on vimeo.[1]

## Contents


**4.1 Visualization of User Behaviour . . . . . 50**
4.1.1 Requirements . . . . . 51
4.1.2 Design Decisions . . . . . 52
4.1.3 Interaction and Visualization Techniques . . . . . 52
**4.2 Visualization of User Data with Heat Maps . . . . . 59**
4.2.1 Calculation of the Heat Map Values . . . . . 60
4.2.2 Logged User Views Compared to Eye Tracking Data . . 61
**4.3 Annotations and Chat Links . . . . . 62**
**4.4 Tracking and Replaying User Actions . . . . . 63**
**4.5 Evaluation and Expert Review . . . . . 64**
4.5.1 Heat Map Evaluation . . . . . 64
4.5.2 Expert Review . . . . . 67
**4.6 Summary . . . . . 69**


[1] `https://vimeo.com/135034649`

## 4.1 Visualization of User Behaviour

With the growing size and availability of large and complex data, the cooperative analysis of such data sets is becoming an important new method for many data analysts as cooperation might improve the quality of the analysis process [112] and help to analyze data sets efficiently. One crucial observation is that collaborators—who are often spread across the globe—would like to seamlessly drop in and out of ongoing work [85]. On the one hand, the collaborative analysis process can take place in a joint online session where everybody is working simultaneously on one data set, discussing and changing it together in real-time to create better analysis results. Here, different experts might want to see what the others are doing, and if there are possibilities to coordinate their efforts and find a common ground [31, 67]. On the other hand, the experts work on the data set whenever they find the time (i.e., asynchronously) to avoid having to schedule and organize a virtual or physical meeting with a larger group of colleagues. Both situations cause specific problems that should be handled by tools which support collaborative work. For instance, while working independently, it would be helpful to see changes of the data performed by other analysts. Another interesting issue is to see which part of the data set has already been explored by others. Here, it is also interesting to know who changed the data: was an established expert working on a specific part of the data, or a new staff member who might not have the same experience as the expert? After implementing the fundamental functions of OnGraX (discussed in the previous chapter), we are now able to tackle the aforementioned problems in the context of collaborative network analyses. In the following, we propose interactive visualization techniques that

- help to coordinate work in a collaborative setting for *node-link diagrams* which may change their topology during the analysis process (referred to as dynamic graphs in the following) and
- assist analysts to identify previous activities performed by former users on these networks.

We exemplify our visualization approaches with the help of the collaborative analysis of metabolic networks from the Kyoto Encyclopedia of Genes and Genomes (KEGG) pathway database [104] due to our long lasting research collaborations with biologists/bioinformaticians at several research institutions [3, 91, 98, 99]. Building such biological networks is often based on complex experiments. In consequence, biologists of different domains and experience levels want to explore the resulting networks and check them for wrong entries or missing data and revise the networks wherever it is necessary. Usually, they only check parts of a network that are specific to their own field of expertise or interest. In this case it is important to know, what part of the network has already been checked and what part still needs attention. This can also be used as a kind of quality check: an area which has been

investigated by many different experts is likely to have a higher quality than an area only investigated by one scientist. OnGraX supports such analysis tasks by providing methods for data awareness and coordination. Note that we retain this usage scenario in the rest of this chapter except in the heat map evaluation (cf. Subsection 4.5.1) in order to attract a higher number of test subjects.

### 4.1.1 Requirements

Our overall goal was to develop a visualization system that allows analysts spatially spread across multiple research labs or even countries to quickly start an analysis session and to work on large and complex networks together. A special problem that arises during the distributed analysis of graphs is that topology and structure of a graph are independent to the layout. Analysts might change the layout drastically during the analysis process, which complicates the task of keeping track of the graph objects and areas that users were most interested in. We also want our tool to support tracking and subsequent visualizing of all actions and graph changes performed by the users. This includes to keep track of the users' camera positions and use this data later to assist users in finding parts of a graph that were interesting to other analysts or have been edited a lot. The reason behind this is that users in a collaborative working environment do not always find the time to work together simultaneously. They would prefer to work on the data set whenever it is convenient for them. And in such a case, they would like to review changes that have been performed on the data set by other analysts in the past. Maybe, they also want to find out which part of the data set another analyst was looking at, since he/she might be an expert in the underlying application field and has another exploration pattern compared to less experienced users. Showing this data—the camera and mouse positions, the logged user views, and changes to specific objects—in the graph without changing the original node-link visualization was an important requirement for our users. Biologists are accustomed to existing layouts and drawing conventions of graphs from the KEGG pathway database, for instance. Thus, changing positions, color, or the shape of nodes to show the data which is collected during collaborations is not an option for our analysis tasks.
During their work, analysts would also like to share their thoughts, insights, and questions about specific nodes, edges or regions with other users. This could happen during a *synchronous* session where collaborators want to discuss their findings, or in an *asynchronous* session where users would like to share messages and pointers on specific nodes. Heer and Agrawala discuss these ideas as "Common Ground and Awareness" and "Reference and Deixis" in their work on collaborative visual analytics [67]. In case of graphs that change their topology during the analysis process, single nodes or complete graph regions could be deleted from a graph, rendering old user annotations useless without the possibility to view them in their historical context. Thus, analysts need a way to quickly view the graph in a state when the

annotation was originally written. Based on this discussion, we categorize our requirements as described in the following.

Collaboration Requirements (C-R)

1. Users should be aware of the position of other users in the same synchronous session.
2. Users should have possibilities to establish and keep a common ground with other users. Everyone should be aware of performed changes on the graph during a session.
3. They should have an option to discuss ongoing work through persistent chat channels and annotations.

Visualization Requirements (V-R)

1. Annotations should be viewable in their historical context. Thus, it should be possible for users to review old graph states.
2. Provide an easy and intuitive way for analysts to find out which regions of a graph were viewed and/or changed by former users.
3. Additionally, the visualization of this data should not interfere with the original node-link diagram.

### 4.1.2 Design Decisions

We carefully designed our system in terms of visual representations, interaction techniques, and analysis processes to support biologists/bioinformaticians in exploring and curating graphs from the KEGG pathway database. We decided to focus our work on node-link diagrams, since this is still the most accepted and preferred graph drawing metaphor, and our users are familiar with this kind of visualization.

### 4.1.3 Interaction and Visualization Techniques

Figure 4.1 shows an overview of OnGraX right after joining an ongoing graph analysis session. In this case, the user has joined a session where two other users, Bob and Sue, are already working in. Their viewports are represented as two dashed rectangles: Bob's view is shown in blue (bottom left) and Sue's view is shown in green (bottom right). All users in a session are listed as small icons at the left hand side of the screen. By clicking on one of the user icons, the camera moves to his/her current position in the graph. This feature provides a quick way to join and discuss another user's viewing area. Visualizing the viewports of other users helps us to tackle our first collaboration requirement (cf. C-R 1). An overview of the graph

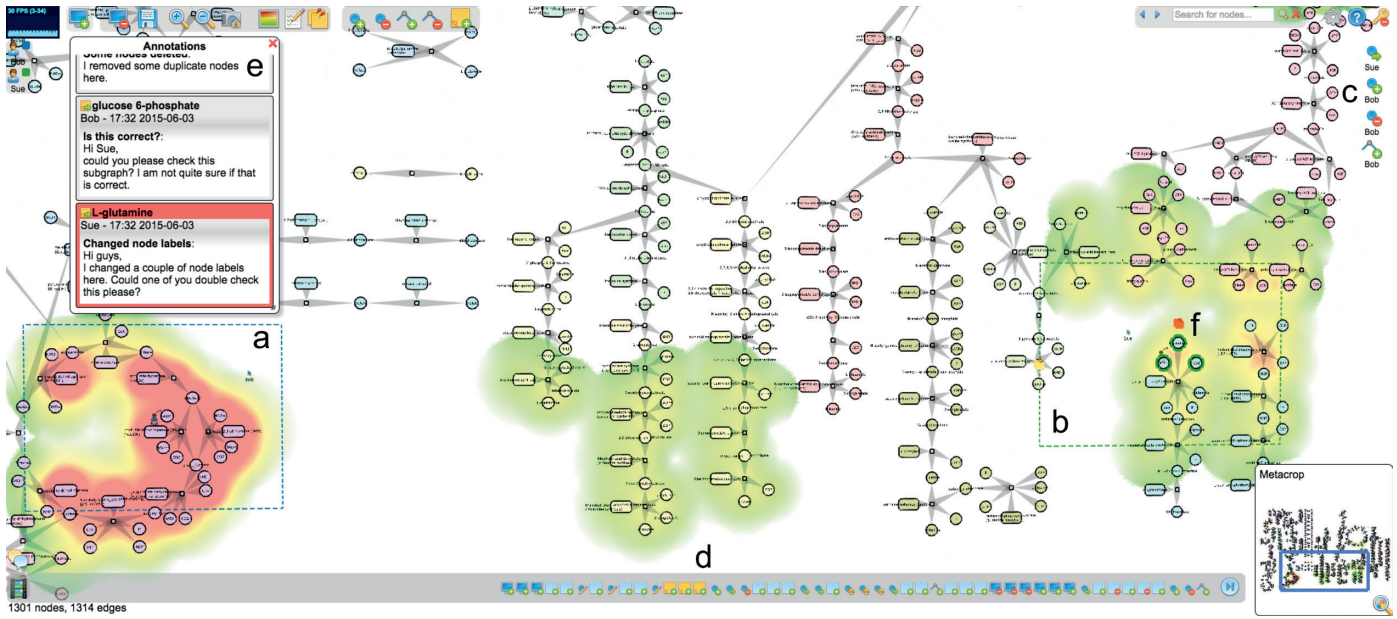



Figure 4.1: Overview of our system. The image shows a part of a biochemical network with 1,301 nodes and 1,314 edges. The blue and green dashed rectangles (see (a) and (b)) are the viewing areas (viewports) of two other users who are also exploring this graph simultaneously. In this concrete case, the underlying heat map highlights those nodes that were in the viewing area to all users during the last hour. Symbols in the top-right corner of the screen (c) assist analysts to keep track of recent actions performed by other users. The timeline (d) is used to temporarily revert the graph to a previous state and to replay applied changes. Analysts can pin text annotations to nodes and edges to discuss tasks, insights and questions with each other (e+f). Image taken from [174]. ©Journal of Graph Algorithms and Applications.

is rendered in the bottom right corner of the screen. Here, the user's camera position is shown as a blue rectangle. As in many other standard visualizations that use overview+detail [33], this rectangle can be dragged to another position in the overview in order to modify the detail view (the same can be done by clicking on the new position in the overview).
We use a standard node-link metaphor to visualize graphs in our system. The visualization uses tapered edges for directed graphs, as suggested by Holten and van Wijk [79], since they provide users with a faster way to find connected nodes as opposed to arrowhead edges. If another user selects one or more nodes, this will be visible to all other participants of the analysis session. An outline in the respective user color is added to a selected node; thereby the system adapts the outline shape to the corresponding node shape.
With the help of OnGraX, analysts are able to explore static graphs, and they can also edit the structure and topology of a graph. There are different kinds of use cases during the exploration and analysis of networks. Depending on the data, an analyst might want to keep the topology of a graph but completely change its layout, in order to find specific structures, hubs, or communities in the data. Also, mapping data attributes to the colors and size of nodes, or to the width and length of edges could change the layout of a graph. In another situation, analysts could actually alter the topology of a graph by adding or removing edges and nodes. This could encompass a few nodes or edges, but it could also affect the graph on a bigger scale, for instance, if a complete subgraph is removed or added. And then, if the topology of a graph is altered on a huge scale, its layout could also change drastically. Comparing different states of a graph and keeping track of changes in such cases is a challenge which is addressed by Hascoët and Dragicevic [66].
The use case in this chapter still includes altering the structure and layout of graphs, but on a smaller scale. The metabolic networks, which are analyzed and revised with OnGraX, already have an existing, handcrafted layout which is not changed by mapping different data attributes to the graph objects. Tasks for which OnGraX is used in this case, usually only require adding or removing a few nodes and edges and/or changing their label, size or color manually. To make such graph changes more obvious—when they are performed by other users during a synchronous session, and to address the second collaboration requirement (cf. C-R 2)—we use short animations on the affected objects, similarly to the work of Gutwin and Greenberg [60]. For instance, the outlines for other users' node selections are animated shortly while they are added or removed, nodes are slowly moved to new positions instead of just jumping there after being moved by another user, and deleted nodes slowly vanish instead of just disappearing.
In order to provide users with a way to quickly find out which nodes or regions of a graph were viewed and/or changed by others (cf. V-R 2), we considered several options. It would be possible to map the corresponding data to the colors or the size of the nodes. Another option would be to use additional glyphs on/around the nodes which represent this data. Using glyphs would also allow us to show both the

viewport data and the data for graph changes at the same time, as small bar charts for instance. The third option is a heat map-based visualization in the background of the graph visualization. We decided to omit mapping the data to the size of nodes, as this would interfere too much with the original graph layout and could introduce too many node overlaps. Additional options would have been to use contour lines [5] or bubble sets [34], but for our use case the focus usually lies on finding and marking single nodes and small groups instead of bigger regions in a graph. We also wanted to visualize the actual numerical values of the data, which is not possible with the aforementioned approaches. Finally three options are left over, and we exemplify them in Figure 4.2.

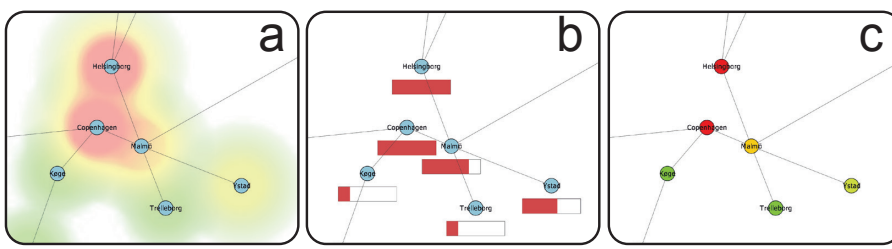



Figure 4.2: Heat map visualization (a) and two alternative approaches: glyphs (b) and node color (c). They are used to indicate which parts of the entire graph were viewed or changed by other users. Image taken from [174]. ©Journal of Graph Algorithms and Applications.

One disadvantage of glyphs in this context is the increased clutter in the graph visualization. Additionally, depending on the size of the glyphs, it could be hard to see the actual data values in highly zoomed-out views of the graph. Changing the color coding of nodes in a graph as alternative is in conflict with our last visualization requirement (cf. V-R 3), because the color coding can be already mapped to another attribute. Thus, heat maps could provide a good alternative to visualize additional data without directly changing the attributes of objects in a node-link diagram. We performed a user experiment (cf. Section 4.5) to assess how the heat map approach compares against glyphs and node colors. During the experiment, we used a three-color gradient for the heat map visualization. After getting diverse feedback about the preferred color gradient, we decided to give users the option to choose between three different gradient options: the default three-color gradient with colors ranging from green over yellow to red, a monochrome version, and a two-colored gradient for color blind users (cf. Figure 4.3). We believe that the colored gradients are better at indicating the actual values that are mapped to the heat map, whereas the grayscale gradient could be used in cases where perceiving the original node colors is more important. The opacity of the heat map can also be adjusted to make it easier to perceive the actual colors of the nodes in case one of the colored heat map versions is used.

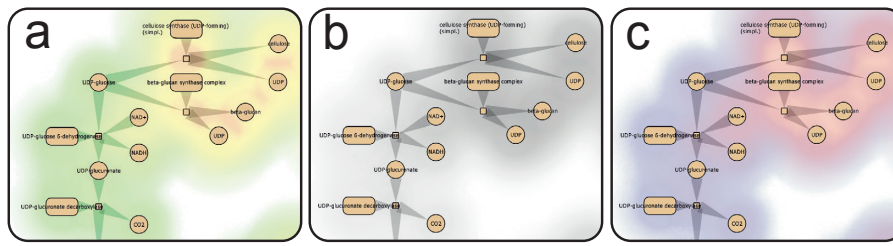



Figure 4.3: Three different approaches for the visual encoding of the heat map: a three-color gradient (a), a grayscale gradient (b), and a two-color gradient for color blind users (c). Image taken from [174]. ©Journal of Graph Algorithms and Applications.

If the heat map visualization is selected, all calculated values are transferred to the requesting client, where they are used to draw an alpha map on an off-screen canvas element. For every node, a circular gradient is drawn which is based on the size and position of the node. See Section 4.2.1 for a more detailed explanation. The actual values, which are mapped to the glyphs, node colors or heat map, can be computed based on two different data sources: viewports and graph changes.

## Displaying Viewports

In the first case, values are calculated based on the amount of seconds that nodes have been in the viewing areas of users (visitation rate). For aggregating this data, OnGraX stores each user's viewport together with the time spent on the position whenever the viewport is changed. Additionally, each time a node is moved, the old position is logged. The server correlates all logged user views and node positions to calculate the values, thus making them robust against changes in the layout of the graph. Figure 4.4 illustrates this approach. In this small example, three stored viewports of one user and two node movements from another user—whose viewports are ignored here—are taken into account. The user arrived at position A at exactly 10:00 AM, stayed there for 10 seconds, moved his viewport to position B for 5 seconds and finally stayed 16 seconds at position C. In viewport A, node 1 was visible for 10 seconds, but in viewport C, it was only visible for 12 seconds, as the node was only moved into the viewport 4 seconds after the user arrived at the position, resulting in a complete viewing time of 22 seconds for node 1. The viewing time of node 2 is only 2 seconds, as it was moved into viewport B 13 seconds after 10:00 AM, and the user arrived there at 10 seconds after 10:00 AM and left 5 seconds later.

For zoomed-out views that show a lot of nodes, it is clear that the user does not attend to all nodes in such a view. To solve this issue, users can adjust the settings to filter out these "big views" and only use zoomed-in views to calculate the heat map.

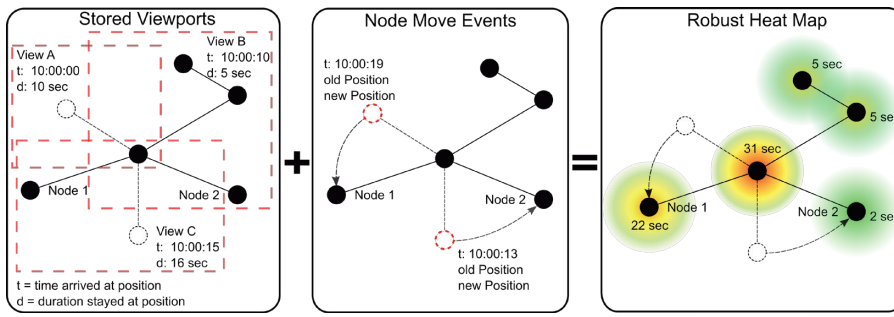



Figure 4.4: Illustration for the correlation of all stored viewports with all node move actions to create a heat map that is robust against layout changes of the graph. Image taken from [174]. ©Journal of Graph Algorithms and Applications.

Figure 4.5 illustrates this idea. In (a) and (b), all viewports are used to calculate the values, (c) and (d) only show viewports with a high zoom level. Views are also only tracked if the user is actively working on the graph: if a user switches to another window or tab, then the tracking is stopped. It is also stopped if the mouse is not moved for a while (currently 20 seconds) to avoid tracking views of inactive users. This approach does still include nodes in the views that might not have had attention by an active user, but it gives a better estimate about the viewed graph regions without asking a user to mark every inspected node manually or requiring all users to use an eye tracker during the analysis process, for instance. Section 4.2.2 gives a short comparison of the user view approach to real eye tracking data.

## Displaying Graph Changes

In the second case, OnGraX calculates values based on changes that have been performed on nodes. Seven actions (name changed, shape changed, node moved, node added, node selected, edge added, edge removed) are tracked and can be used to calculate the heat map values in this case. A multiplier is specified in a configuration dialog for each individual action type to give it more or less weight during the calculation (see Figure 4.6). This enables analysts to highlight only nodes that were moved and had their names changed, for instance. Figure 4.7 gives an example for such a case: the heat map highlights three regions with renamed and moved nodes. The visualization can be configured to only show a specific user or to show the data for all users together (the selection of user groups would also be possible and could easily be added to the system). Furthermore, it is possible to select a time frame, for instance, the last five minutes of the current analysis session, or a specific start and end date. This enables an analyst to review changes done in a collaborative session during a specific time frame or to check the work of a single user.

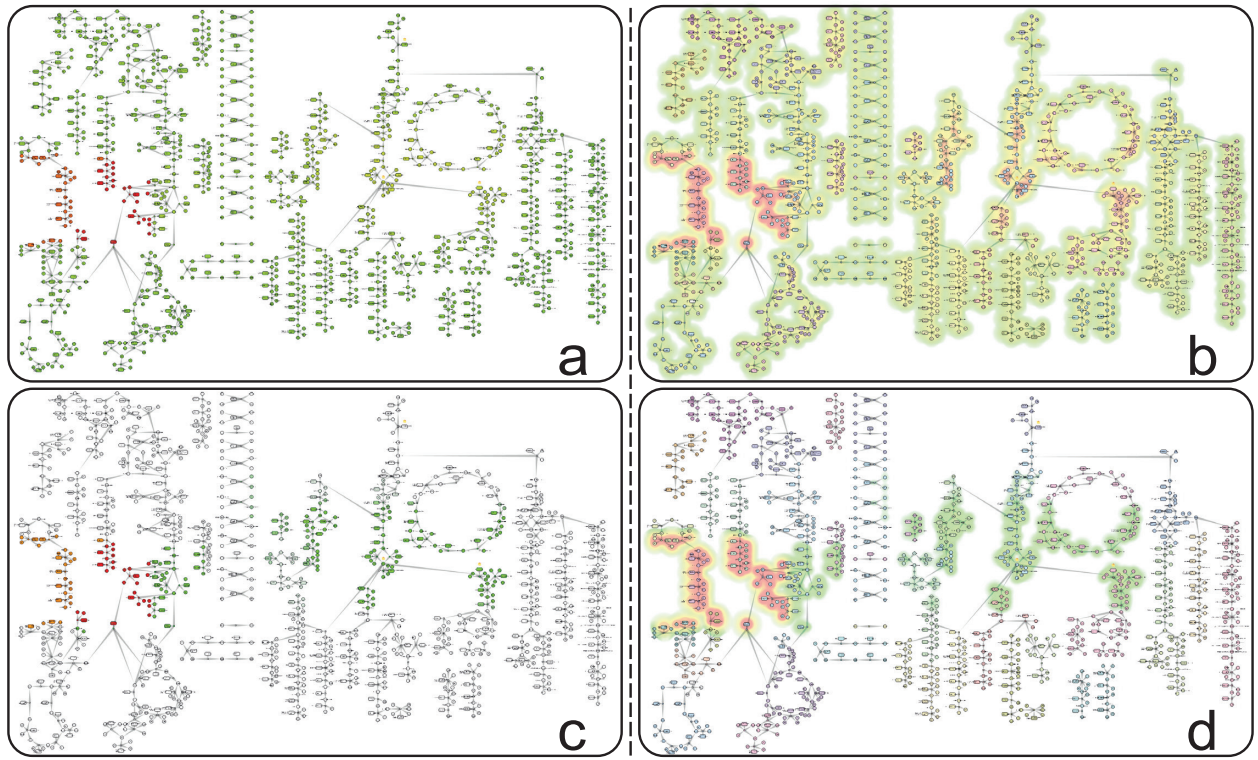



Figure 4.5: Example of filtering out user viewports with a low zoom level. (a) and (b) show all viewports, whereas (c) and (d) only show zoomed-in viewports of one selected user who was looking at a specific part of the graph. Here, (a) and (c) show the viewports mapped to *node colors*, whereas (b) and (d) show the same data visualized as *heat map* to preserve the original node colors. Image taken from [174]. ©Journal of Graph Algorithms and Applications.

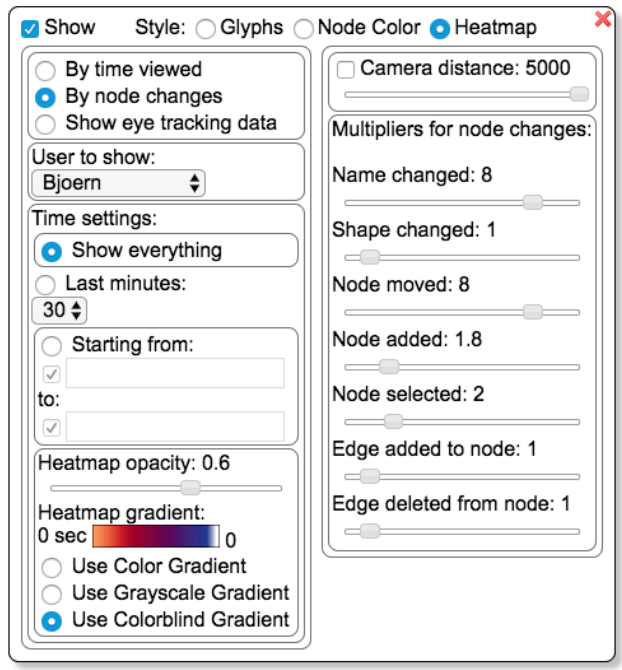



Figure 4.6: Dialog for the heat map settings. Users can chose between showing logged user views, highlighting performed changes on the nodes of the graph, or visualizing eye tracking data. If the dialog is set to show performed changes on the nodes, the user can choose which types of changes should be used as multipliers to calculate the heatmap (see "Multipliers for node changes" in the right part of the image).

## 4.2 Visualization of User Data with Heat Maps

The heat map-based visualization in the background of the node-link diagram (cf. Figure 4.1) does either show aggregated values of all logged user views to find regions of interest or applied node changes to get an overview of changes that were performed on a graph. If the heat map configuration is set to show the logged user views, the server calculates the heat map values by aggregating all logged user views with all node positions from all move actions that have been performed previously on the nodes. This process results in a heat map visualization that is robust against layout changes of the graph.

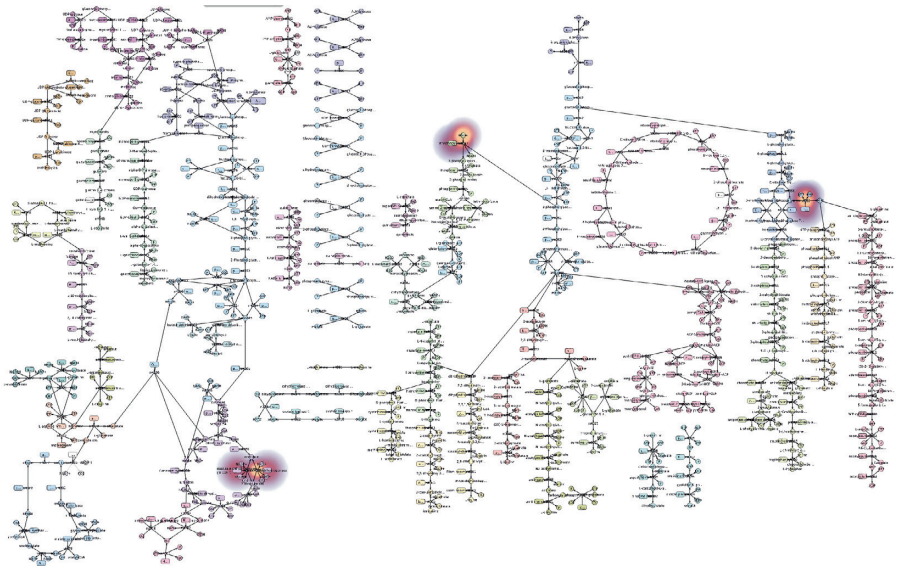


Figure 4.7: Example of highlighting changes that were performed on nodes. In this case, the heatmap shows three regions where nodes were moved and renamed by a specific user.

### 4.2.1 Calculation of the Heat Map Values

After the values are calculated and sent to the requesting client, they have to be visualized in the web browser without negative performance effects on the interactive graph visualization. To avoid having to render a lot of additional objects, we first draw the heat map values on an off-screen canvas element and create a single OpenGL texture from this canvas afterwards. The canvas represents the heat map values as an alpha map—for every node, a circular gradient is drawn which is based on the size and position of its related node. This creates an image with grayscale values ranging from 0 to 255. To create the actual colors for the heat map, each pixel value is used to lookup the color from a $1 \times 256$ pixel-wide color gradient. The gradient colors range from white, over green to red. Based on these color values, an OpenGL texture is created and put on a mesh in the background of the graph visualization (see Figure 4.2, (a)). By using an alpha map as basis, we could also draw a heat map based on mouse positions or eye tracking data easily.

Getting the heat map values for all actions that were applied to all nodes of a graph only takes around 4 milliseconds on the server for a typical graph analysis session. Calculating the heat map based on user views and node positions takes considerably more time, because the server has to correlate all stored views with all logged node positions to get the number of seconds every node of the graph was viewed by a

user while also considering layout changes. During our test sessions, the server calculated those values in about 60 milliseconds if the complete time frame was selected. The time also depends on the amount of stored user actions and node positions and will of course increase slightly if a graph analysis session is used for a longer time period and more actions are logged. However, since analysts are usually only interested in specific time frames which span over the course of a few days, this is not an important issue. Right now, the bottleneck during the heat map generation lies on the client side. The largest graph that is currently visualized with our system spans an area of 5,800×3,600 pixels. Drawing the alpha map on a canvas with the same size would take approximately 20 seconds, which is not adequate for generating a real-time heat map. Another problem occurs during the creation of the OpenGL texture, as the maximum texture size is limited by the client's graphic card. Current graphic cards support textures with up to 16,384×16,384 pixels, whereas older computers are limited to 2,048×2,048 pixels. We avoid this performance problem on the client side and achieve a real-time rendering for the heat map by drawing a scaled-down version of the alpha map if the graph area exceeds the size of 2,048×2,048 pixels. The resulting texture is then put on a mesh in the background of the visualization and stretched to the appropriate size of the graph.
An alternative and faster approach for generating the heat map would be to create the alpha map and the texture array directly on the server and send the array for the texture to the requesting client(s). But this would drastically increase the amount of traffic between server and clients. Therefore, we decided against this idea and settled for the slightly slower approach of drawing the heat map on the client side.

### 4.2.2 Logged User Views Compared to Eye Tracking Data

In a normal collaborative setting, users do not have access to eye tracking hardware. Additionally, it would be too much work to setup and calibrate an eye tracker for every session. As such, employing logged views to estimate interesting regions of a graph is a good alternative, even though this data is not as accurate as data from an eye tracker. To demonstrate the possibilities of logging user views with eye trackers, we compared eye tracking data raised during a study with two different eye trackers against the user views as part of a bachelor thesis by Bilgic and Vulgari [18]. Figure 4.8 shows a heat map based on OnGraX' logged views (a) and a heat map based on eye tracking data (b) from one of the users of the study. The graph in the screenshots shows the Euroroad network of Europe and Asia. Nodes represent major cities and edges are added between two nodes if the cities are connected with a highway. We asked the user to first locate the cities Barcelona and Berlin. Afterwards, the user had to find a path along the network from the first to the second city. Figure 4.8 shows, that real eye tracker data is more accurate than OnGraX' logged views. In future work, the accuracy of the eye tracking data could be improved even further by calculating the fixation points of the raw eye tracking data [78, 124]. Nevertheless, the steps required for setting up the eye tracking hardware, and calibrating the eye

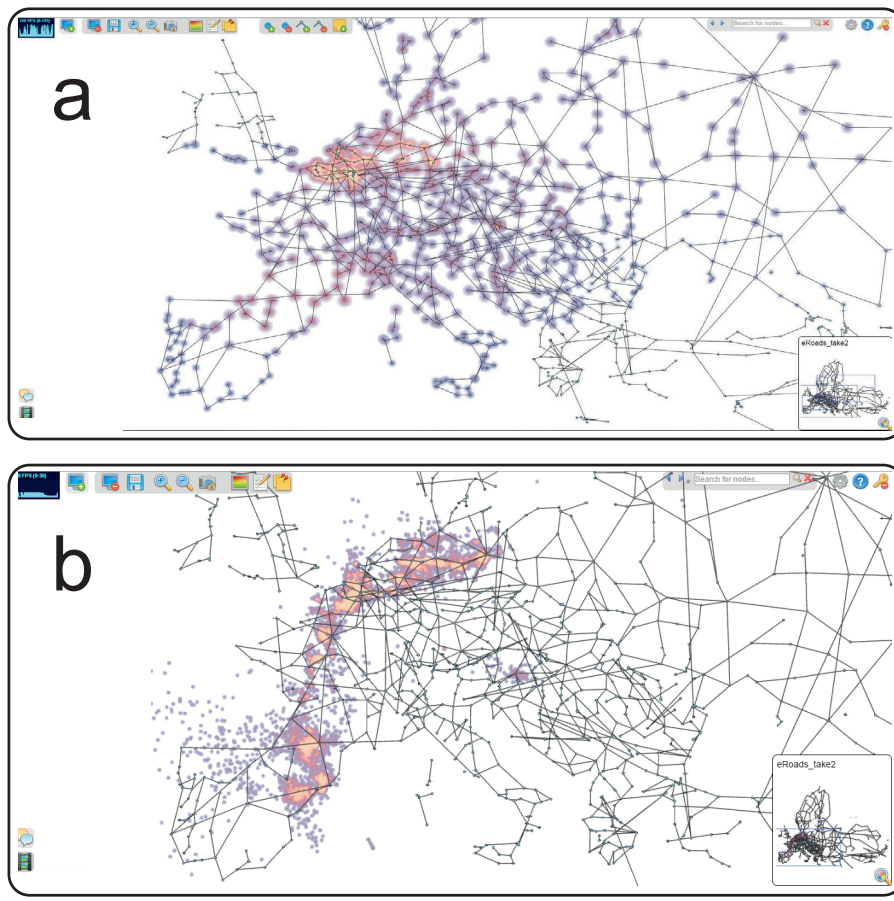



Figure 4.8: Comparison of logged user views to eye tracker data: (a) shows a heat map that was generated based on the logged user views during an exploration session. (b) visualizes the users eye tracking data from the same session.

tracker for each user are usually too time consuming, and eye trackers are not available everywhere. Thus, OnGraX' logged user views provide a good compromise between accuracy and ease of use.

## 4.3 Annotations and Chat Links

In order to improve the communication among collaborators, our tool has a persistent chat channel for every graph session and offers the possibility to link chat messages to a position or a node in the graph. Users can use those chat links to

move the camera to the linked object or position. A link to an arbitrary position might become obsolete after changes to the graph layout, but a message linked to a node or edge will always be valid as long as the object is not deleted. In addition, users can attach textual annotations directly to nodes or edges (cf. Figure 4.1, (e+f)). These annotations work as pointers from the graph visualization to text and vice versa. Clicking on an annotation in the graph visualization opens the annotation dialog and highlights the linked message. A click on an annotation in the dialog moves the camera to the object's position in the graph visualization. With the chat and annotation features, we address our last collaboration requirement (cf. C-R 3).
One problem with textual annotations and chat messages linked to objects is, that the original context in which an annotation or message was initially written could get lost if the respective graph region—where the link is pointing to—is changed during the course of a session or if the object with this link is deleted. We solve this problem by enabling analysts to temporarily revert the complete graph to an old state (similar to the timeline feature, see next Section 4.4) by right clicking on a chat link or an annotation, giving them the possibility to view the graph in a state in which the annotation was originally written. This feature addresses our first visualization requirement (cf. V-R 1).

## 4.4 Tracking and Replaying User Actions

Actions performed by other users during a *synchronous* session are shown at the right corner of the screen (cf. Figure 4.1, (c)) together with the name of the user who initiated the action. A right-click is used to dismiss a recent action and a left-click moves the camera to the location of the action in the graph. Another left-click on the same action moves the camera back to its original position. Thus, users can quickly check what their collaborators are doing and then return to their own work, without having to navigate to every performed action manually. To provide our users with the possibility to keep track of *all* actions that occurred in a session, we use a scrollable timeline at the bottom border of the screen that shows the complete action history of the graph session (cf. Figure 4.1, (d)). The mouse tooltip for the symbols in the timeline shows the action time and the name of the user who performed the action. The timeline can also be used to revisit old graph states and replay previous actions. If a user clicks on a symbol, all actions performed since this specific action are replayed in reverse order. The visualization will show the graph in a state before the action was performed. Shortly after the graph has been transformed to its old state, the clicked action is reapplied, animating the graph to the requested point in time.
Figure 4.9 shows an example of such a transition. This feature gives users a tool to revisit old graph states and replay old actions allowing them to assess what work has been done by other collaborators. Clicking on the rightmost symbol reverts the graph back to its present state. While viewing an old graph state, it is *not* possible

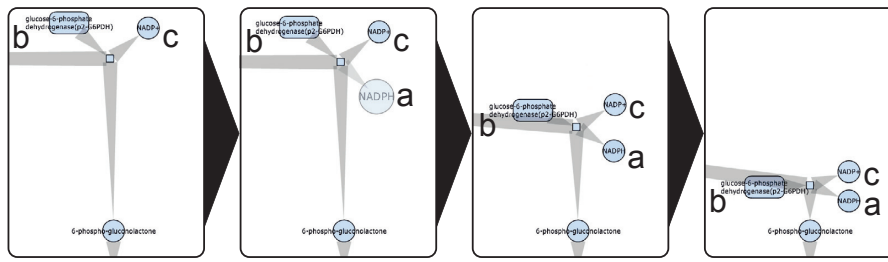



Figure 4.9: Transition to a previous graph state. The figure shows the *reverse animation* for a node delete action (a) and node move actions for the nodes marked with (b) and (c). Image taken from [174]. ©Journal of Graph Algorithms and Applications.

to apply any changes to the graph. We decided against this feature as it would open the possibility to create numerous new branches of different graph states. This is an interesting aspect and actively researched [155], but currently not the focus of our work.
It is also possible to inspect the complete action history of a graph session as a list in an additional dialog. To not get overwhelmed with uninteresting notifications in this dialog or the timeline, users are able to configure which types of actions are tracked or filtered out.

## *4.5 Evaluation and Expert Review*

We performed two studies: a user experiment and an expert review to get an impression of the usability of the most important aspects of OnGraX. The user experiment had the goal to evaluate the usefulness and acceptance of our heat map approach to visualize user behavior data in comparison to glyphs and node coloring. Here, the test subjects worked independently of each other, i.e., asynchronously. In contrast, the expert review aimed to assess OnGraX' features for collaboratively revising a graph during synchronous analysis sessions in the domain of metabolic network analysis.

### 4.5.1 Heat Map Evaluation

We recruited 15 participants (7 undergraduate students, 7 graduate students, and 1 post-graduate; average age = 28; 5 female, 10 male). Seven participants had a background in computer science and eight a background in media technology. Eight participants never worked with node-link diagrams before, but everyone was familiar with them.

All 15 sessions were recorded on video and the participants were instructed to employ a think-aloud protocol. Before starting the actual tasks, the tool and the three visualization approaches for user behavior data (glyphs, node color, heat map) and their meaning were introduced by the experimenter, and each participant could explore a sample graph to get accustomed to the tool. Each session took about 25-30 minutes, and we asked the participants to solve each task as quickly as possible, but the time for the tasks was not limited by us. All participants had to solve two tasks for nine different graphs with the help of the three visualization approaches. Both tasks were described as follows:

**Task 1 – explore graph changes:** Find and count all nodes that were moved by a specific user (9-14 single marked nodes per graph).

**Task 2 – explore viewports:** Find all regions that a specific user was most interested in (1-3 marked regions per graph).

The experiment was conducted following a within-subjects design, and users were divided into three different groups. Every group explored all graphs in the same order but with a different sequence of visualization approaches. Six graphs were generated randomly: the first three graphs consisted of 1,000 nodes/edges and the following three of 2,000 nodes/edges. For the last three graphs, we used existing metabolic networks with 1,300 to 1,800 nodes/edges.

#### *4.5.1.1 Quantitative Results*

We started measuring the task time in seconds for each task as soon as the visualization of the user behavioral data was enabled by the participants and stopped the time as soon as they reported a number. For Task 1, we show the number of nodes that were not found by the users (mean error rate), see Figure 4.10. In Task 2, all participants found all marked regions, regardless of the visualization approach. Therefore, we only report the error rate for Task 1.
Initial Friedman tests showed that both tasks had statistically significant differences in task completion time. Task 1: $\chi^2 = 34.881$, $p < 0.001$. Task 2: $\chi^2 = 16.812$, $p < 0.001$. We conducted a post hoc analysis with Wilcoxon signed-rank tests for our not normally distributed data. For Task 1, the median interquartile range (IQR) task completion times were 26 (Glyphs), 39 (Node Colors), and 20 (Heat Map), see Figure 4.11. Both, glyphs vs. heat maps ($Z = -3.678, p < 0.001$) and node colors vs. heat maps ($Z = -5.334, p < 0.001$) had a significant reduction in task completion time. For Task 2, the median (IQR) task completion times were 19 (Glyphs), 15 (Node Colors), and 13 (Heat Map), see Figure 4.12. Here, the heat map approach also performed significantly better in comparison with glyphs ($Z = -3.678, p < 0.001$) and node colors ($Z = -2.406, p = 0.016$).

#### *4.5.1.2 Qualitative Results*

We asked all participants which visualization approach they preferred. Everyone favored the heat map visualization. For them, the heat map was the easiest to perceive, and it also provided the most convenient way

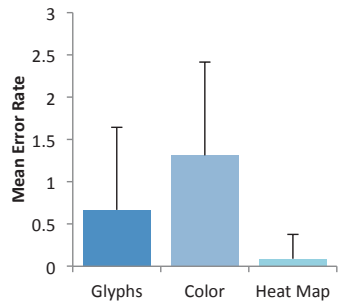



Figure 4.10: Task 1, mean error rate. Image taken from [174]. ©Journal of Graph Algorithms and Applications.

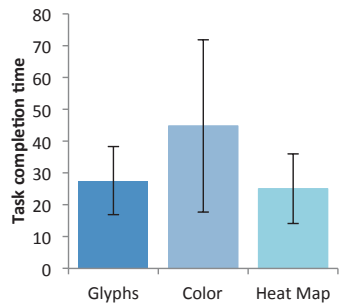



Figure 4.11: Task 1, mean completion time in seconds. Image taken from [174]. ©Journal of Graph Algorithms and Applications.

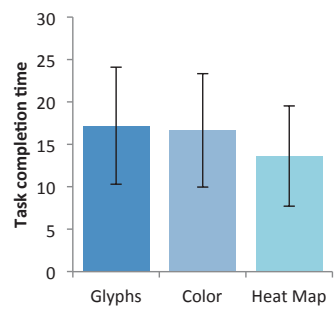



Figure 4.12: Task 2, mean completion time in seconds. Image taken from [174]. ©Journal of Graph Algorithms and Applications.

to find single nodes with high values, even at lower zoom levels. While performing the second task, four participants mentioned that the glyph approach introduced too much clutter in the view, especially for the metabolic networks. They said that glyphs were hard to distinguish from the actual nodes, because both the nodes and glyphs sometimes had a similar shape.

### 4.5.2 Expert Review

Expert reviews are an alternative to assess interface usability and find possible problems without having to perform a full scale user study [147]. Goal of this expert review was to get informal feedback about the usability of OnGraX' features for synchronous sessions from a small group of experts, who use the tool to revise metabolic networks collaboratively. More precisely, we wanted to get a first impression if OnGraX could address our three collaboration requirements (cf. C-R 1-3), i.e., (1) if the participants could keep track of each other's view position in the graph, (2) if they could keep a common ground while working on the graph, and (3) if they could coordinate their work by using the chat and annotation features of OnGraX. The review was performed with an earlier prototype of our tool. Please note that this review has an informal character and is by no means a full scale user study, which would require more detailed preparation (cf. the work of Carpendale on evaluating information visualizations [24]).

***4.5.2.1 Participants and Review Setup*** We asked three experts from the Bioinformatics research group at Monash University, Australia, to collaboratively work on a network in a *synchronous* session. All three experts were male. One of the experts was a full professor of Bioinformatics (45 years of age), and the two others were research fellows (one postdoctoral researcher, one very experienced research assistant; both around 30 years old) with degrees in Bioinformatics as well. The professor already knew OnGraX, because we asked him to provide some general feedback related to specifics in Bioinformatics (e.g., typical shapes of nodes or use of color) before the review took place. Consequently, he also introduced the task for the review to the two other experts. We asked the experts to revise the graph shown in Figure 4.1. All experts worked with this graph previously. For this expert review, however, a small part of the network was removed by the experimenters, and the experts had to manually edit and add the missing nodes and edges together. Every expert was sitting alone in his own office and was not supervised by us directly. Before they started, the participants had to watch a six minute introduction video for OnGraX and afterwards opened a web browser on their computers to join the graph session in OnGraX at the same time. The heat map visualization was turned off by default, as we did not aim to assess the heat map features in this review, but the feature was mentioned in the introduction video. We did not specify a time limit for the review, and the experts worked for about one hour on the graph. One experimenter

also joined the session remotely to observe the experts during the review. His own viewport (zoomed out completely) was visible to the expert reviewers, and he could answer urgent questions via the online chat.

***4.5.2.2 Results*** After the experiment, we asked the participants to write down their thoughts about the tool and if the available features helped to fulfill our three collaboration requirements. The first participant found it very easy to follow the work and positions of the other online users with the help of the viewports and tracking possibility of the mouse cursor, while the other two experts thought that the symbol for displaying the center of the other users' viewports together with all mouse cursors was a bit confusing. For them, it would have been enough to only show the colored rectangles and mouse cursors of specific users. Additionally, when clicking on a user icon in the top left corner of the screen (see Figure 4.1, (c)), they would have preferred to move to the respective user's mouse cursor rather than to the center of his view. We fixed these issues in the current version of OnGraX and also added the possibility to automatically follow another user's mouse and/or camera position in real time. The experts thought that this would be a useful feature to allow collaborators to just follow one user's work on a graph while they discuss possible changes with each other.

All participants found the tool quite interesting, but they missed some features. In particular, everyone asked for an "undo" function, and they would have liked to be able to hide the timeline at the bottom of the screen. Both features have already been added in the current version of OnGraX. Even though the experts found the chat window helpful, they would have preferred to talk directly via an in-browser voice chat. They said it was easy to follow and track applied changes during the session, but discussing the changes that should actually be performed was quite cumbersome as they had to write down every question in the chat window. Alternatively, collaborators could use tools like Skype or Google Hangouts, but this would require more time during the start of a collaborative session. It would be more convenient if a voice chat functionality could be archived in the browser without having to rely on external programs or additional plugins. This could be addressed in a future version of OnGraX with the help of the new WebRTC standard [162] for build-in real-time communications in browsers.

The feedback which we got from this review indicated that being able to see the viewports and mouse cursors of a selection of users, helps analysts to keep a common ground in a synchronous session (cf. C-R 1). The experts also found the short animations when nodes and edges are added, deleted or modified to be helpful for following ongoing changes of other users (cf. C-R 2). Even though we did not ask them to use the heat map feature, two of the experts actually turned on this functionality and used it to track the nodes down that were added during the course of the review. The experts evaluated the heat map and chat feature as helpful, whereas the functions to replay user actions and add annotations were a bit less important

for them. This is likely because they worked in a synchronous session and did not need to replay former actions or read old annotations (cf. C-R 3), since they could use the chat window to communicate with each other directly.

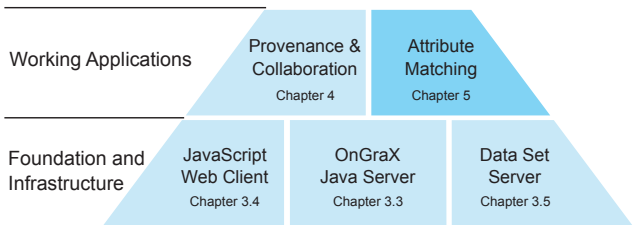
## 4.6 Summary

In this chapter, we presented visualization and interaction techniques for analyzing data sets synchronously *and* asynchronously in a distributed environment. This addresses **Goal 1.A** of this thesis (see Section 1.1). With the help of OnGraX, users can seamlessly drop in and out of ongoing sessions and do not have to wait for other users to start or finish their work. All actions performed during a session as well as the users' camera positions are tracked and can be visualized along with the graph data by using underlying heat map representations. This helps experts to analyze regions of a graph that were of interest or have been edited by former users, i.e., social navigation and guidance is maintained by the aggregated user activity in form of heat maps. Here, we address the "Collaboration & Awareness" challenge in web-based collaborative visualization that was raised by Heer et al. [70] and achieve **Goal 1.B and 1.C** of this thesis (see Section 1.1). We propose using heat maps to efficiently show additional data without affecting the original graph visualization. Based on a user experiment, we show that the heat map-based approach compares better against glyphs or changing the background color of nodes. In our specific use case, graph changes are usually limited to a couple of nodes, thus the tracking of *all* actions and visualizing this data is not an issue here. This could become problematic if a graph or a subgraph is changed drastically. In this case, additional options to set the granularity for tracked events and alternative visualization techniques would be required including a newly designed evaluation.
In the synchronous collaboration case, we provide another contribution (cf. "Pointing & Reference" challenge in [70]) namely that participants of an analysis session can follow the activities of the others (shared viewing areas) and point to specific nodes or edges by using brushing (shared node markers and mouse cursors). The use of WebSockets enables us to distribute this user data in real time among all connected clients.
With the help of OnGraX and all its implemented visualization and tracking functions, we have now build a solid foundation for future projects, extensions to OnGraX, and specific use cases and can now address the next two goals of this thesis:

- **Goal 2:** the extension of OnGraX to support specification and exploration of relationships between heterogeneous networks and additional data sets is discussed in Chapter 5 and
- **Goal 3:** the design of a visualization system for guidance in heterogeneous networks is discussed in Chapter 6.

*Chapter 5*

# Matching of Attributes across Heterogeneous Networks




A growing number of real world networks are multivariate and often interconnected with each other. Entities in a network may have relationships with elements of other related data sets, which do not necessarily have to be networks themselves, and these relationships may be defined by attributes that can vary greatly. The following chapter presents a comprehensive visual analytics approach that supports researchers to specify and subsequently explore attribute-based relationships across networks, text documents, and derived secondary data. The approach provides an individual search functionality based on keywords and semantically similar terms over the entire text corpus to find related network nodes. This chapter is based on the publication [175]. Two videos about the overview of the approach and the use case are available on vimeo.[1]

## Contents


**5.1 Visualization of Documents . . . . . 72**
**5.2 Design . . . . . 74**
5.2.1 Visual Design . . . . . 75
5.2.2 Hub2Go . . . . . 77
**5.3 Example Application . . . . . 78**
5.3.1 Text Preprocessing . . . . . 80
5.3.2 Bag-of-Words . . . . . 81
5.3.3 Word Similarity . . . . . 81
5.3.4 Matching . . . . . 82
5.3.5 Use Case . . . . . 84
**5.4 Discussion . . . . . 90**
**5.5 Summary . . . . . 90**


[1] Overview: `https://vimeo.com/216708596` ; Use Case: `https://vimeo.com/216708834`

## 5.1 Visualization of Documents

The combination of different heterogeneous networks and related textual data is crucial for various application domains. For instance, a network of telephone calls or email traffic and the organizational tree of a company could be interesting for social scientists to discover various patterns of communication inside a company. Libraries, for example, are nowadays interested in analyzing (known or hidden) relationships among various collections of books, which might be related to each other even though they do not share the same author or topic. Based on an initial book search, analysts want to find out what terms were used in a specific book, find related ones that might use the same or similar terms, and also visualize the direct neighborhood network of those books which could, for instance, consist of other books written by the same authors.

Tools for the visual analysis of documents and literature publications are related to our use case application. A general overview of text visualization techniques is available in the interactive Text Visualization Browser [103][2]. Furthermore, Federico et al. [44] provide a broad survey about various techniques and open challenges for analyzing scientific literature.

More specifically, Görg et al. [55] use Jigsaw for the exploration and sensemaking of document collections. They use a list view to show the relationships among co-authors, keywords, and other attributes, together with a clustered view of related papers. Shen et al. [142] use an interactive table together with multiple stacked network planes in a 2D or 3D view to visualize interconnections between various paper, author, and publication networks. PivotSlice [167] uses a more general approach to visualize relationships in data sets. It supports dynamic queries on attributes and divides data sets into facets and connected views on the data. Phrase Nets [153] offers techniques to depict relationships in a text and map it into a network based overview. Refinery [92] shows combined heterogeneous networks with subgraphs generated from user queries in one network view. Chen et al. [30] analyze the metadata of document collections and visualize identified topics together with co-authors in a single network visualization to show collaborations over time. Instead of only using titles, abstracts, keywords, and other metadata as the previous tools, our system also supports exploring the complete text corpus of a document collection, which gives analysts the possibility to find documents based on terms that might otherwise not be visible. There are various other systems that use the complete text corpora of document collections. For instance, TextPioneer [110] uses on automatic topic extraction to build a hierarchical overview of interesting topics across multiple text corpora, whereas our approach focuses more on the exploration of specific terms of a document corpus. CiteRivers [73] aims at visualizing citation patterns of scientific document collections over time but does not feature the exploration of author collaborations as we do in our network visualizations.

[2]The Text Visualization Browser: `http://textvis.lnu.se`

Similar text corpora may be derived from conference proceeding publications, such as the IEEE VIS conference proceedings, which we use as application example and use case in this chapter. A researcher could be interested in finding out more about a specific topic and therefore wants to explore publications that use a number of specific keywords or terms. However, related publications might use different terms but still talk about the same idea. Or those publications mention certain terms that do not appear in the title or their keyword list. For less experienced researchers it might help to alleviate the initial process of getting an overview of their new field if they would have access to tools that are able to find these concealed connections.

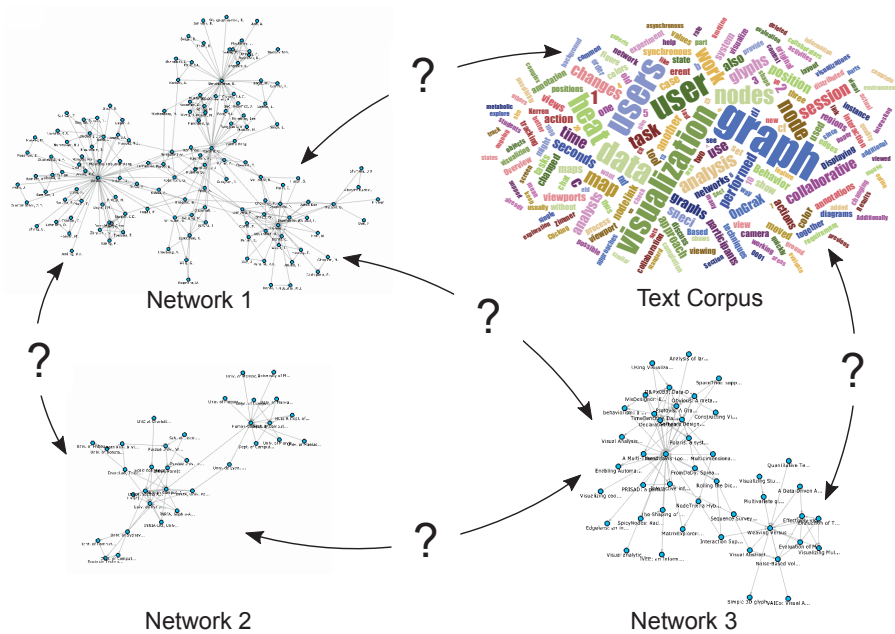



Figure 5.1: Different networks and data sets can be interconnected with each other, but these connections may not be immediately apparent. How can users visualize and explore them interactively? Image taken from [175]. ©Informatics.

Integrating different network visualizations together with related multivariate data and helping analysts to discover relevant information—in context of this work by providing them with interactive possibilities to view and explore the relations between networks and large text corpora—is still an open challenge and has not yet been sufficiently addressed to the best of our knowledge. This approach is also important for other application domains such as biology, where the correlation of large metabolic pathways, protein-protein-interaction networks, or regulatory networks is an important issue, and the interconnections between various networks are not al-

ways obvious. Different networks come with numerous multivariate attributes (e.g., coming from experimental data) and exploring just one network and its related multivariate data is already a challenge in itself. This has been addressed by various techniques and tools, see the state-of-the-art survey of Kerren et al. [97]. Another survey for the visual analysis of graphs and open challenges in this area is provided by von Landesberger et al. [156]. In a collection of data sets (see the illustration in Figure 5.1), an attribute in one network might be related to other attributes in various other networks, even though they might not have the same identifier or name. To make interesting assessments about the data, analysts need to search for specific keywords, terms, authors, or other metadata across different networks that might or might not be related.
If they also have text corpora available in addition to the network data, it can be interesting to see the relationships between documents which result from analyzing the text corpora. Such relations could be derived from the usage of specific terms in the documents, from a list of papers written by a specific author, or be based on texts written at the same institution. It could also be worth investigating whether authors who use similar terms throughout their publications have a connection to each other. Examples of such a connection could be co-authoring a paper, performing research in the same field, or having the same affiliation. Maybe the authors never really worked together, but one author was citing another author's work and was influenced by the choice of words in the original document. In this chapter, we address these challenges and analysis tasks. Our main contributions in the field of heterogeneous network exploration in context of visual text analytics are:

- we present a scalable system that gives analysts the possibility to interactively *specify and explore mappings* across interconnected heterogeneous networks with thousands of nodes and related data derived from text corpora;
- we provide a way to perform a search based on *main keywords and semantically similar terms* over the entire text corpus to find related nodes which represent, for instance, papers, authors, or affiliations;
- we introduce the technique "Hub2Go", which enables users to *quickly add and examine* these nodes in the interconnected networks views; and
- we give the option to directly *compare the usage of specific terms* across a selection of nodes.

## 5.2 Design

Depending on the use case, it is not always possible to merge several heterogeneous networks into one view. For instance, some metabolic networks from the Kyoto Encyclopedia of Genes and Genomes (KEGG) pathway database [104] have

a precomputed layout which is preferred by biologists, but the networks are still connected with each other. In this case it would be beneficial to have multiple views on the networks and provide means to navigate the relationships, similar to the approach facilitated by Entourage [107]. In another scenario, multiple networks could be merged into one single network view. The Apolo system [28] uses this technique to make sense of large network data using machine learning and user interaction to find nodes that might be of interest to the user. Our goal was to design a flexible system which can be used for different application areas. We also wanted to be able to visualize additional related multivariate data across the networks. It is possible to show attributes of target nodes directly in a graph visualization, for instance as information wedges at the borders of the visualization [54], but the number of attributes that can be visualized with this technique is limited and not applicable for all possible use cases. Consequently, we decided to use multiple coordinated views together with brushing & linking techniques [131] for our approach.

### 5.2.1 Visual Design

Graphs are visualized as interactive node-link views (see Figure 5.2(a)) whereas list-based data sets, such as the bag-of-words index, are shown as horizontal bar charts (see Figure 5.2 ((d+e)) due to space efficiency. The distributional similarity dialog (see Figure 5.2(e)) displays the most similar terms of a selected word in the related BoW chart. The attribute comparer (see Figure 5.2(f)) helps to compare quantitative attributes, such as the BoW values for instance, among a selection of nodes that can be added from any network view by performing a right-click on a node. It also supports sorting of nodes based on a selected attribute. The concrete processing depends on the network type in our case: if a node is added from the affiliation or the co-authorship network, the accumulated value from all papers for that affiliation or author is visualized, whereas a node from the paper network shows the actual BoW or TF-IDF value for that paper directly. If a node in one of the network views or an entry in the charts is hovered with the mouse, all related nodes or entries in all other views are highlighted, which helps to interactively explore the relationships between the data sets.
Node-link visualizations of networks with many thousands of nodes and edges are usually quite cluttered and difficult to explore without further filtering or clustering. To circumvent this issue, we facilitate a bottom-up approach to explore the data sets: when the application is started, all network views are empty and the BoW/TF-IDF charts show all terms over the complete data set. Nodes from networks are added into or removed from the views on demand, either by directly searching for a specific node in the node list (see Figure 5.2(g)), finding nodes by keywords from the BoW chart, or by requesting connected matches from a node in another node-link view. If a network does not already have precomputed node positions, the network views use a force-directed layout from the yFiles library [166]. The layout is calculated whenever new nodes are added to the view and tries to minimize position changes

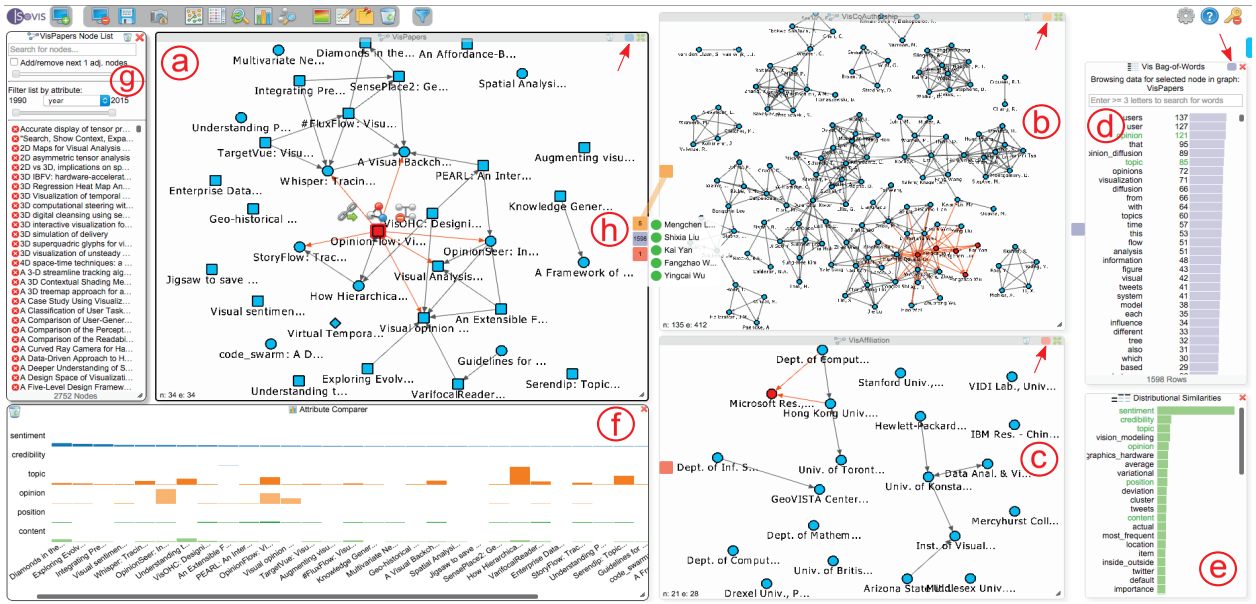



Figure 5.2: An analysis session in progress. Here, a mapping between a paper citation network (a), a co-authorship network (b), and an affiliation network (c) is explored. The bag-of-words chart (d) shows all terms for the currently selected paper in the paper network, and the chart at (e) shows terms used in a similar context as the word "sentiment" throughout the complete text corpus of the IEEE VIS conference proceedings. The attribute comparer (f) visualizes the occurrence of six terms over a selection of papers. The dialog at (g) contains the complete list of nodes for the active network that the user is currently exploring; in this case the paper network. The three colored rectangles at (h) realize our interactive Hub2Go, which currently indicates matches from the paper "Opinion Flow", which is selected in the paper network, to three other networks or data sets. The small rectangles at the top right of the three network views and the bag-of-words chart (marked with red arrows) represent the colors used for the Hub2Go instances (h) to indicate matches across networks and data sets. Image taken from [175]. ©Informatics.

of already existing nodes in order to preserve the mental map. If necessary, users can adjust the layout settings and select different layout algorithms such as circular or hierarchical graph layouts.

### 5.2.2 Hub2Go

In order to perform a bottom-up exploration across interconnected heterogeneous networks and related data views, it is essential to quickly find nodes of interest. Users should also have the option to seamlessly add nodes into a related target network and swiftly go to their location to explore their local neighborhood. Moreover, it should be possible to remove nodes from a search result, if it did not yield interesting information. Depending on the use case, it is also not always feasible to directly show all interconnections to a target node if the node has a high number of interconnections.

To provide a flexible solution for these problems, we introduce the *Hub2Go* technique. Whenever a search is requested by clicking on a node in a network view or by selecting a term in the BoW or TF-IDF charts, the system represents the search results as Hub2Go instances at the border of the initial network view or chart (see Figure 5.2(h)). The hubs show the connections to other data sets: in context of Figure 5.2, the user clicked on the paper "Opinion Flow" (marked in red) in the paper network (see Figure 5.2(a)) and initialized a search for connected entities in other networks or data sets based on the configured mappings. Three Hub2Go instances show the result: five nodes in the co-authorship network, 1,598 terms in the BoW chart, and one node in the affiliation network.

To create the concrete network views given in Figure 5.2, we started by searching for the term "sentiment" in the BoW chart (see Figure 5.3(a)). By clicking on the resulting term, the system matches the term over all configured networks and visualizes the result as three Hub2Go instances (see Figure 5.3(b)). In this case, the hubs show 34 papers, 166 authors, and 21 affiliations that mention this term at least once. Hovering over the blue hub, which is related to the paper network, shows a list of all 34 found papers. Left-clicking the hub adds all nodes from the search result to the target network (see Figure 5.3(c)); or removes them, if they are already in the view. Alternatively, users could browse through the list and manually add or remove nodes of interest by left-clicking them. Going back to Figure 5.2, we also added all matching nodes to the other node-link views using our Hub2Go technique. Additionally, users can perform a right-click on a hub or a node in a hub-list to add all or specific nodes to the attribute comparer. In this case, we added all 34 papers to the attribute comparer (see Figure 5.2(f)) by right-clicking on the blue Hub2Go instance and, to analyze the occurrence of related terms among these papers, we also added "sentiment", "credibility", "topic", "opinion", "position", and "content" to the attribute comparer by right-clicking them in the distributional similarity dialog, which currently shows the most similar terms for the term "sentiment".

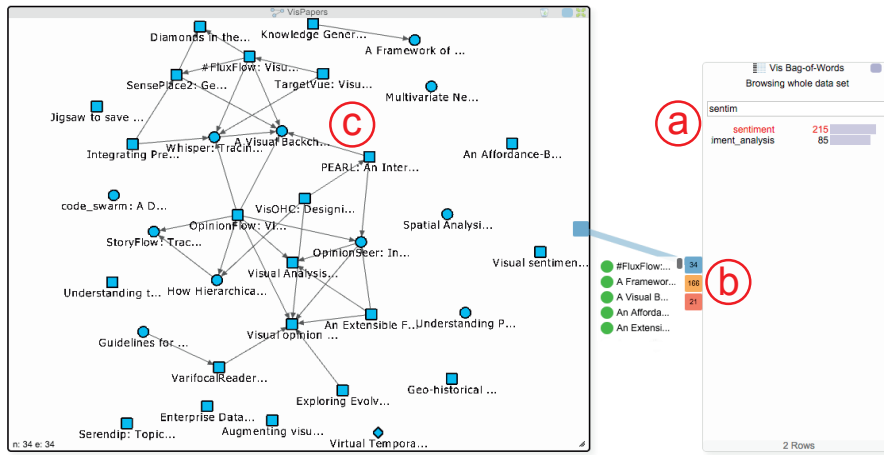



Figure 5.3: Usage of the Hub2Go instances: The user searched for the term "sentiment" in the BoW chart (a) and clicked on it. The resulting matches across the networks are shown as three Hub2Go instances at the border of the dialog (b). Hovering the blue hub shows a scrollable list with all related papers. Clicking the hub adds all papers to the paper network view (c). Image taken from [175]. ©Informatics.

In Figure 5.2 the user is currently hovering the orange hub—associated to the co-authorship network—with the mouse cursor, which brings up a list with all found authors for the selected paper and also highlights all connected nodes in the other views. Hovering the mouse over a specific item in the hub-list automatically centers the connected target node in the relevant view, enabling the analyst to swiftly explore its local neighborhood. To indicate whether nodes of a search result are already in the view, circles in this list appear green if they are already added to the network view, or red if they have *not* been added to the view. When clicking on a node in a hub list, the neighborhood of this node can optionally also be added to the network view (based on a user setting). This is helpful to quickly find related nodes in the data set, for instance, citations from or to other papers, or authors who also wrote other papers together with the initial author of a paper.

## 5.3 Example Application – Analyzing Relationships between Heterogeneous Networks and Texts

Generally, OnGraX can be used to specify and explore mappings across any networks and related quantitative data. For our use case employed throughout this chapter, we focus on analyzing the complete text corpus which we derived from all papers in the IEEE VIS conference proceedings with the help of the visualization publication data set [84]. We created three networks from the initial data, which help us to explore relations within the data. Nodes in the first network (2,752 nodes

and 10,021 edges) represent all papers from 1990 to 2015, and edges represent paper citations. Node shapes in the resulting network visualization encode the VIS conferences: rectangles for VAST, circles for InfoVis, triangles for SciVis, and diamonds for older Vis papers. Nodes in the second network (4,890 nodes and 14,023 edges) represent all authors and co-authors from the data set. Edges are added between authors if they wrote a paper together at least once. Nodes in the third network (1,539 nodes and 5,773 edges) represent all author affiliations, and edges are added if authors from different affiliations published a paper together. As we did not prune the initial VIS data set, the co-authorship and affiliation networks may contain duplicates with slightly different wording. Tools such as D-dupe [17] could be used to reconcile this issue.

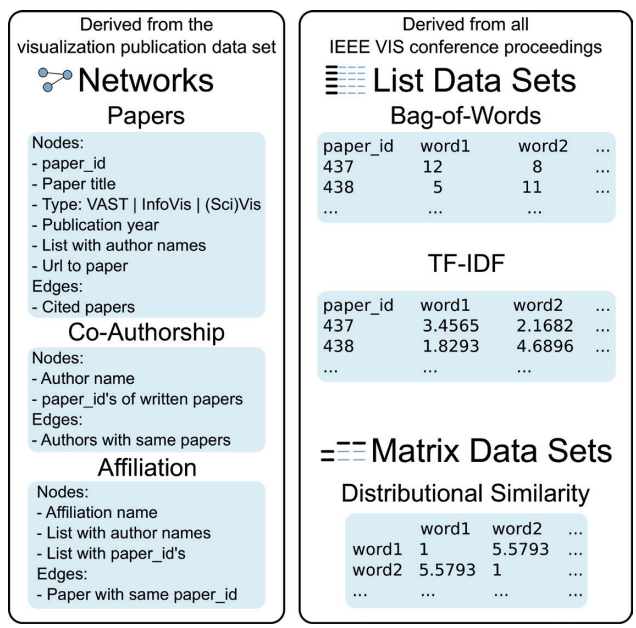



Figure 5.4: Overview of our data set: all networks, node attributes, and edge types were taken from the visualization publication data set, with additional textual data derived from all papers in the IEEE VIS conference proceedings. Image taken from [175]. ©Informatics.

Moreover, we did a pdf-to-text conversion for all papers in the IEEE VIS conference proceedings and used the resulting text corpus to create a word *occurrence* index (a so-called Bag-of-Words (BoW) index) and a word *importance* index (a so-called TF-IDF index) to find the most frequent and most important terms. The resulting indices contain 12,346 words/terms each and can also be used to uncover the main representative words/terms in a specific paper or to find terms that were used by a

specific author or even in an affiliation. The indices contain single words as well as bigrams, which are discussed in Section 5.3.1. Additionally, we use a word *similarity* matrix to find and analyze the relationships between terms that are used in the same context between various papers. Figure 5.4 shows the structure and attributes of the networks and data sets in detail. The resulting list and matrix data sets are stored as plain, space delimited text files on the server. At runtime, the server loads and keeps a copy of all required data sets in memory to provide faster query times for mapping requests during the exploration of the networks.

Note that the difference between the BoW and TF-IDF indices is that the former only quantifies the frequency of occurrence of terms in the data (i.e., has a term occurred or not?), whereas the latter tries to qualify the importance of occurrences (i.e., is a term important or not?). Having identified the main keywords in a paper (or author/affiliation) of interest, it is also possible to find other papers (authors/affiliations) that have used the same terms. As an example, we may be interested in a specific paper about "sentiment analysis". By using the BoW index, we can find all other papers that have mentioned this term, and we can subsequently analyze the co-authorship networks of this group of papers (see Section 5.2 for details on this process). If we use the TF-IDF index instead, we find other papers for which the term is a useful index term (i.e., for which it is important). Finally, the word similarity matrix enables us to find other terms that have been used near-synonymously (or, more generally, *in the same way*) as some keyword. Searching for the term "sentiment analysis" again, we can use the word similarity matrix to find other terms that might be of interest to us; perhaps terms such as "opinion mining" and "topic detection", which could then be included in the current analysis. In the following, we provide details on how we produced the BoW, TF-IDF indices, and word similarity matrices.

### 5.3.1 Text Preprocessing

After the pdf-to-text conversion, we removed the punctuation from the resulting text and also lower-cased and tokenized it. We even identified salient bigrams in the data, such as the term "traffic_flows" from our use case in Section 5.3.5, using a variant of mutual information that is commonly used for bigram detection in natural language processing:

$$p(a,b) = \frac{f_{a,b} - \delta}{f_a f_b} \tag{5.1}$$

where $f_{a,b}$ is the co-occurrence frequency of $a$ and $b$, $f_a$ and $f_b$ are the individual frequencies of $a$ and $b$, and $\delta$ is a discounting factor that counteracts the tendency of mutual information to promote infrequent items, which we set to $\delta = 5$. The resulting mutual information scores are sorted, and only the highest-scoring bigrams are included in the data. The threshold for the mutual information scores is a trade-off between precision and recall: a lower threshold leads to more but less precise

bigrams, while a higher threshold leads to fewer but more precise bigrams. We opt for a more conservative threshold in this paper.

### 5.3.2 Bag-of-Words

Bag-of-Words (BoW) is a standard text representation formalism in natural language processing (NLP), which represents text simply as a bag containing frequency counts of the words that occur in the text. Formally, the BoW representation of a text is a vector:

$$\vec{w} = [t_1, \ldots, t_n] \tag{5.2}$$

whose dimensionality $n$ is the size of the vocabulary (i.e., each word in the vocabulary is represented by one separate dimension), and $t_i$ is some weight that quantifies the importance of the word in the text. There are many ways to quantify the importance of a word in a document; the arguably most common term weighting scheme is TF-IDF, which in its simplest form is defined as:

$$\text{TF-IDF}_{i,j} = \text{TF}_{i,j} \cdot \log \frac{N}{\text{DF}_i} \tag{5.3}$$

where $\text{TF}_{i,j}$ is the frequency of word $i$ in document $j$, $\text{DF}_i$ is the number of documents word $i$ occurs in, and $N$ is the total number of documents in the data. It should be noted that there are many variations, refinements and alternatives to using TF-IDF to extract useful terms as indicated by Chuang et al. [32], for example. However, we opt for the standard TF-IDF measure in this work, since it is simple to compute and produces useful results. Using BoW representations enables us to find all documents that contain a certain term (i.e., all documents for which the value of the dimension representing the term in question is not zero). The TF-IDF representation also enables us to list the most useful index terms for a document (i.e., the terms with highest TF-IDF weights for that particular document).

### 5.3.3 Word Similarity

In order to produce the word similarity matrix, we use *distributional semantics*, which is the practice of using information about the co-occurrence patterns of terms in order to quantify semantic similarity. In a standard Distributional Semantic Model (DSM), each word is represented by a *distributional vector*, $\vec{w}_f = [w_1, \ldots, w_m]$ where $w_i$ is a function of the co-occurrence count between the focus word $w_f$ and each context word $w_i$ that has occurred within a window of $k$ tokens around the focus word [149]. Words that have co-occurred with the same *other* words (i.e., that are interchangeable in context) get similar distributional vectors, which means we can use the resulting model to quantify semantic similarity between terms.

There are many variations of DSMs, ranging from simple count-based methods to the currently more popular, and more complex, models based on factorization techniques or neural networks [118, 126, 148]. We opt for an approach based on an

incremental random projection of a count-based model, called Random Indexing (RI) [93], which accumulates distributional vectors $\vec{v}(a)$ by summing sparse random index vectors $\vec{r}(b)$ that act as fingerprints for the context items:

$$\vec{v}(a) \leftarrow \vec{v}(a_i) + \sum_{j=-c, j\neq 0}^{c} w(x^{(i+j)})\pi^j \vec{r}(x^{(i+j)}) \tag{5.4}$$

where $c$ is the extension of the context window, $w(b)$ is some weight function that quantifies the importance of context term $b$, and $\pi^j$ is a permutation that rotates the random index vectors according to the position $j$ of the context items within the context windows, thus enabling the model to take word order into account [134]. We use 2,000-dimensional vectors, $c = 2$, and the weight function suggested in Sahlgren et al. [133]. There are several reasons why we use RI instead of one of the currently more popular DSM models. One is the ability of RI to encode word order, which is not possible to do using, e.g., the `word2vec` library or the `GloVe` model. Another reason is the scalability and efficiency of RI, which makes it suitable for collections with large amounts of data. It should also be mentioned that the most current DSMs perform similarly, given that parameters have been optimized for the specific data and task at hand.
In order to produce the word similarity matrix, we compute pairwise similarities between all words with frequency $\geq 50$ in the data using cosine similarity.

$$\cos(a,b) = \frac{\sum_{i=1}^{n} a_i b_i}{\sqrt{\sum_{i=1}^{n} a_i^2}\sqrt{\sum_{i=1}^{n} b_i^2}} \tag{5.5}$$

We utilize a comparatively high frequency cut-off for the word similarity matrix, since low-frequent words have insufficient distributional statistics to produce reliable distributional vectors.

### 5.3.4 Matching

Before exploring the interconnections and relations between networks and additional data sets, analysts have to specify which attributes they want to compare throughout the complete data collection. Instead of having to do this programmatically, our application enables analysts to create and subsequently explore arbitrary mappings between all graphs and additional data sets (e.g., lists or matrices) on the fly by using a small bipartite graph visualization as shown in Figure 5.5. Nodes either represent a graph or a data set that was imported into the application. The node colors are selected by the tool from a color list created with ColorBrewer [21]. They are also used as background colors for the hubs during the exploration of a mapping to indicate the target data set (see Figure 5.2(h)). Every dialog with a view on a graph or data set has a small colored rectangle in the top right corner to indicate its mapped color. Users can add unidirectional and bidirectional mappings by dragging

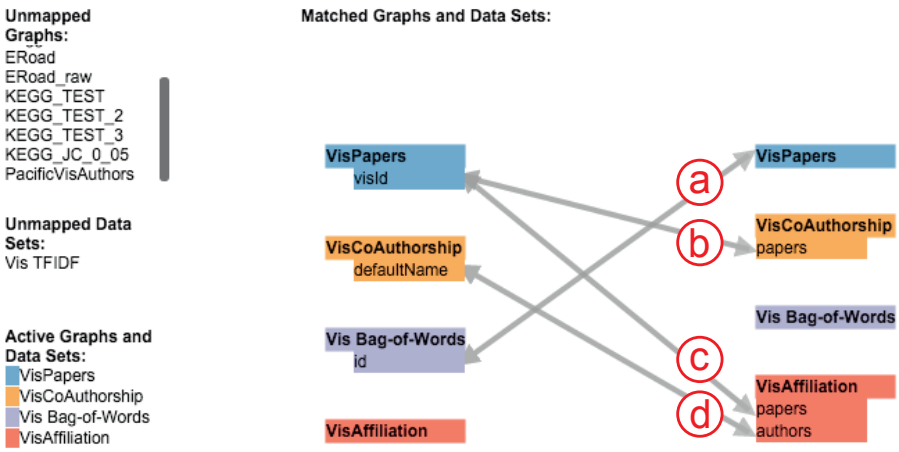



Figure 5.5: The dialog to configure the mapping between graphs and data sets. This figure shows four configured mappings between three networks and the BoW index. Image taken from [175]. ©Informatics.

the mouse from a source to a target node. The application will ask which attribute from the source should be mapped against the target.

As performance can be an issue if a large number of graphs and data sets with a lot of entries are loaded, users can add and remove graphs and data sets (that are already imported into the system) to the matcher on the fly. Depending on a priori knowledge, mappings can be defined from and to specific attributes or be more general, to iterate over all possible attributes of a target graph or data set. For the data from our use case, all matches across the data sets are calculated and visualized in less than a second, but to provide better scalability for bigger data sets with a large number of entities and additional attributes, specific mappings could be helpful to reduce the required time to iterate over all possible matches.

An actual mapping is performed dynamically, i.e., if a user clicks on a node in one of the network views or an entry in one of the additional bar charts. In Figure 5.5, the analyst declared four mappings between three graphs and the BoW index. The first mapping (see Figure 5.5(a)) from the BoW index to the papers network is used if a user clicks on a node in the respective network view. The application will then iterate through the BoW index, locate the row with the matching paper ID and fetch all terms used in that paper to show their BoW values in the BoW chart. As this mapping is bidirectional, the mapping will also be performed if a term is clicked in the BoW chart. In this case, the application will fetch all papers that use this specific term. The next two mappings (see Figure 5.5(b+c)) from the paper publications network to the co-authorship and affiliation networks are used to find related authors and affiliations for a clicked paper. The application will map the paper to

all authors in the co-authorship network and affiliations that have the paper ID in their list of papers. The last mapping (see Figure 5.5(d)) maps an author's name from the co-authorship network to the affiliation network, thereby enabling the user to find all affiliations a specific author wrote papers for and also to find all authors of a specific affiliation. Currently, our application only discovers matches between attributes with identical values. For instance, the last mapping (see Figure 5.5(d)) does only find completely identical strings between the configured attributes of the co-authorship and affiliation network. To be able to support fuzzy matching, we plan to extend the mapping feature in a future version. String similarities and configurable range values for numerical attributes could be used to find and display similar results.
In our example, the latter three attribute mappings (see Figure 5.5(b+c+d)) are defined to specific attributes in the source and target nodes, as the exact attributes, interconnections, and relations are already known for these data sets. Alternatively—if the attributes of a newly imported graph are unknown—it is possible to specify an attribute mapping which tries to find identical attribute values by iterating over all entities and their attributes of a mapped graph or data set. The mapping in Figure 5.5(a), for instance, implies the paper network without declaring a specific target attribute. In this case, the application will try to match all possible attributes of the target to the specified source attribute. If an attribute can be mapped depends on the type of the source attribute (e.g., nominal or ordinal). The application will only try to map between attributes of the same type.

### 5.3.5 Use Case

Our analysis goal for the use case described in the following is to discover relationships between papers that cannot be spotted by an analysis of pure metadata, such as the data set provided by *vispubdata.org*. For instance, we may want to discover thematic outliers in the paper collection of a specific author and analyze if there are similarities between those outliers and other papers written by the author (or by others). The three network views could also be of variable interest to different users. Students or researchers usually want to explore research results and are interested in the relations between existing papers and authors, whereas libraries or funding organisations might also be interested in co-authorships and the related affiliations to explore the results of an interdisciplinary project.
Let us assume for this use case that a student previously read an interesting paper about "traffic flows", and that the student remembers one of the authors of this paper: "Jarke J. van Wijk". The student wants to find out more about the author and might also have a few questions related to the paper and wants to use our system to answer them:

**Q1 - author question:** Which words/terms does van Wijk use in his papers? Answering this question would provide the student a first impression about van Wijk's most important research areas.

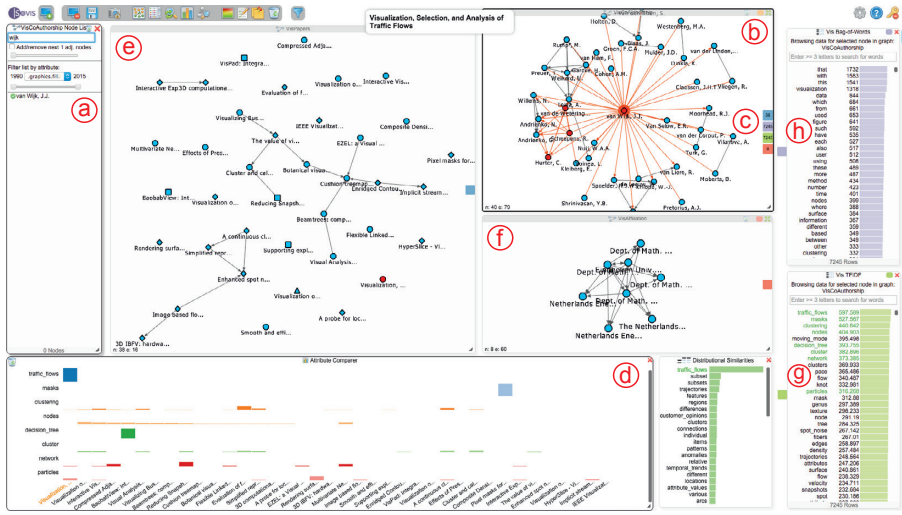



Figure 5.6: After searching for the author "Jarke J. van Wijk" in the node dialog (a), adding his corresponding node and all of his co-authors to the co-authorship network (b), and performing a matching over all graphs and data sets (c), the system also allows the exploration all of van Wijk's papers (e), affiliations (f), and the terms he used throughout his papers with the help of the TF-IDF (g) and BoW (h) charts. The attribute comparer (d) shows TF-IDF values of van Wijk's papers for eight chosen terms, which are marked in green in the TF-IDF chart. Currently, the paper "Visualization, Selection, and Analysis of Traffic Flows" has been selected in the VisPapers and Attribute Comparer views. Image taken from [175]. ©Informatics.

**Q2 - author question:** Does he usually focus on the same terms throughout his papers, or does he address various different topics, which would indicate a broad research interest?

**Q3 - paper question:** Are there other papers related to the term "traffic_flows", that might be of interest to the student?

**Q4 - paper question:** Considering the term "traffic_flows": are other terms in other papers used in a similar context, which would make those papers also interesting to the student?

***Analysis Step 1*** Aiming to answer Q1, the student first adds the TF-IDF chart to this analysis (see Figure 5.6(g)) to get the most important words among selected papers. He/she then activates the co-authorship network view, and searches for van Wijk's name in the search dialog, see Figure 5.6(a), and adds van Wijk's corresponding author node together with all other nodes of authors who collaborated

with him to the egocentric network view (see Figure 5.6(b)). After clicking on van Wijk's node in this view, the system shows four Hub2Go instances (see Figure 5.6(c)) which indicate 38 related papers in the paper network (blue), 8 nodes in the affiliation network (terra cotta), and 7,245 terms in the BoW (violett) and TF-IDF charts (jade green). The two charts already give the student an overview of all terms used in van Wijk's papers, thereby answering Q1. Left-clicking on the blue hub adds all papers to the paper network view, and the student also puts all of van Wijk's affiliations into the affiliation network view (note that in this case, some of the affiliations are actually duplicates with slightly different wording in the visualization publication data set).

The student could now further explore which terms are used in a specific paper by clicking on its node in the paper network. The student could also explore which of his own (or his co-authors') papers van Wijk cites in the VIS publication data set. For doing so, we use a pop-up menu that appears while an interesting node (i.e., a specific paper) is hovered with the mouse in order to add all nodes that are adjacent to the hovered node into the network view. This way the student is able to easily identify papers that cite or are cited by one of van Wijk's papers. By using the pop-up menu, he/she could also directly open the IEEE-Xplore link, which is saved in the URL attribute of each node, to take a closer look at the actual papers.

Hovering over a node in the VisPapers view also highlights connected nodes in the other networks, enabling the student to see the authors and affiliations of specific papers. Furthermore, the student could also investigate the co-authorship network and hover over a node to highlight all papers that an author wrote together with van Wijk.

***Analysis Step 2*** To answer Q2, the student investigates the TF-IDF chart which shows the terms for all of van Wijk's papers, as long as his node is selected in the co-authorship network. The student adds eight of van Wijk's most used/important terms "traffic_flows", "masks", "clustering", "nodes", "decision_tree", "cluster", "network", and "particles" to the attribute comparer (see Figure 5.7) and right clicks the blue hub in the co-authorship view to also add van Wijk's papers to the comparer. The student notices that the terms "clustering", "nodes", "cluster", "network", and "particles" are used quite evenly throughout van Wijk's paper collection. But the other three terms are interesting outliers: "traffic_flows", "masks", and "decision_tree" are each uniquely used in three different papers. In this case, the student discovers from the chosen terms, that van Wijk addresses similar topics in most of his papers (in case of the terms that are used evenly), but the three outliers could also indicate a broader research interest. The student could now further investigate and use the paper network view to open the links to the papers, and he/she could also use the TF-IDF chart to get an overview of the other terms that were used in those papers.

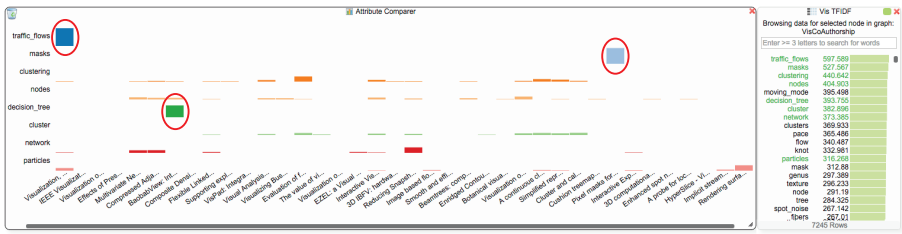



Figure 5.7: Distribution of 8 selected terms in van Wijk's papers. The terms "traffic_flows", "masks", and "decision_tree" (marked by red ellipsoids) are each uniquely used in different papers. Image taken from [175]. ©Informatics.

***Analysis Step 3*** The student now decides to address Q3 and to find more papers about the term "traffic_flows". Figure 5.7 already revealed that van Wijk only has one single paper that mentions this term and searching for the term in the titles of all papers via the node list of the paper network also only reveals van Wijk's paper. The addition of the BoW and TF-IDF indices makes it possible to search for terms in the data, rather than being limited to search for keywords in the titles of papers. Therefore, the student can now find all papers that mention some specific term, regardless of whether it is in the title of the paper or not. To see if there are other papers in the data set that use this term, the student clicks on "traffic_flows" in the TF-IDF chart and gets three Hub2Go instances linking to 12 papers, 42 authors, and 9 affiliations (see Figure 5.8). After clearing the attribute comparer from the previous search queries, the student adds all 12 papers to the paper network and the attribute comparer using the blue hub. Van Wijk's paper has by far the highest frequency of the term, but the student could now also open the other papers to see if they are relevant for his/her research.

***Analysis Step 4*** The student can now utilize the TF-IDF (or BoW) values together with the distributional similarities to further investigate these 12 papers and find additional work that might be thematically related to traffic flows, thereby addressing Q4. The student selects van Wijk's paper in the paper network to get the TF-IDF values for that paper and adds the next three terms with the highest values ("aircraft", "traffic", "trajectories") to the comparer. A well-known issue with keyword search is vocabulary variation (sometimes referred to as *the synonymy problem*), which means that several different terms can be used to refer to the same thing. In the case of "traffic_flows", there could be other terms that are also used by researchers to refer to this type of visualization. The student therefore consults the distributional similarities view (see Figure 5.9), which shows the most similar terms, and identifies four interesting ones "trajectories", "features", "regions", and "patterns", which might be relevant for investigation besides the term "traffic_flows". The first term from the distributional similarities "trajectories" was already in the

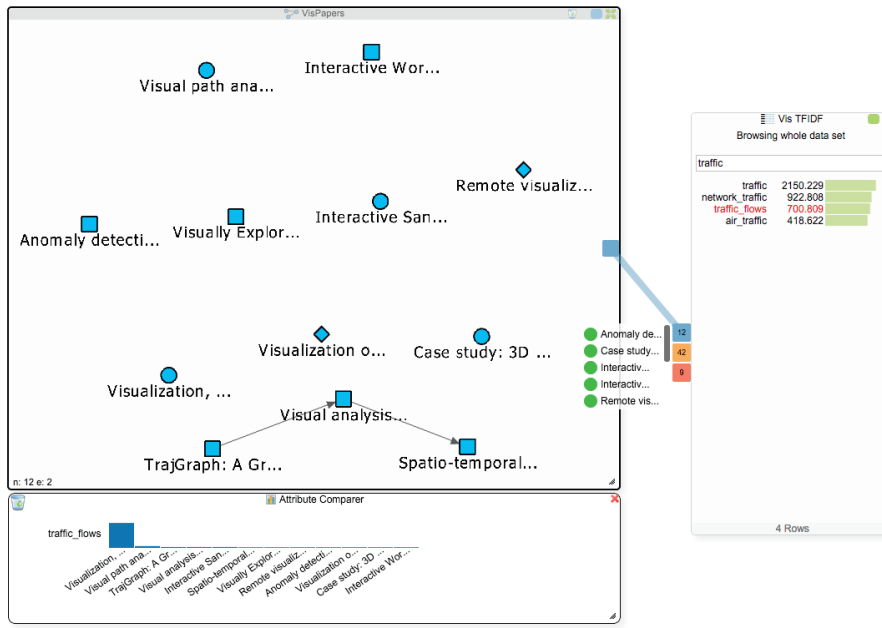



Figure 5.8: The network view shows all papers mentioning the term “traffic_flows”. The attribute comparer shows the distribution of that term among the papers. Image taken from [175]. ©Informatics.

TF-IDF chart, which encourages the student to further investigate the papers mentioning this term.

Right-clicking on the term “trajectories” in the attribute comparer orders the papers in the chart based on the TF-IDF values for this term. The student finds five new papers that mention this term (see Figure 5.9(1–5) at the bottom in the attribute comparer dialog). They could be of interest, and the student decides to add them into the paper network view that still contains all of van Wijk’s papers. Right-clicking the papers in the attribute comparer adds them to the paper network view: four of these papers (1–4) are actually connected to each other, and one of van Wijk’s papers (see Figure 5.9(6); “Composite Density Maps for Multivariate Trajectories”) is part of this group, as it cites two papers with the term “trajectories”. Since it does not mention the term “traffic_flows”, it was not added to the attribute comparer in the initial search. To investigate the occurrence of the already selected terms in this paper, the student also adds it to the attribute comparer by right clicking on it in the paper network. The student finds out that the paper also uses the terms “trajectories”, “features”, and “patterns”, which suggests that he/she might also take a closer look at this paper. Note that the distributional similarities encode similarity of usage in the data rather than some generic (or “objective”) notion of semantic

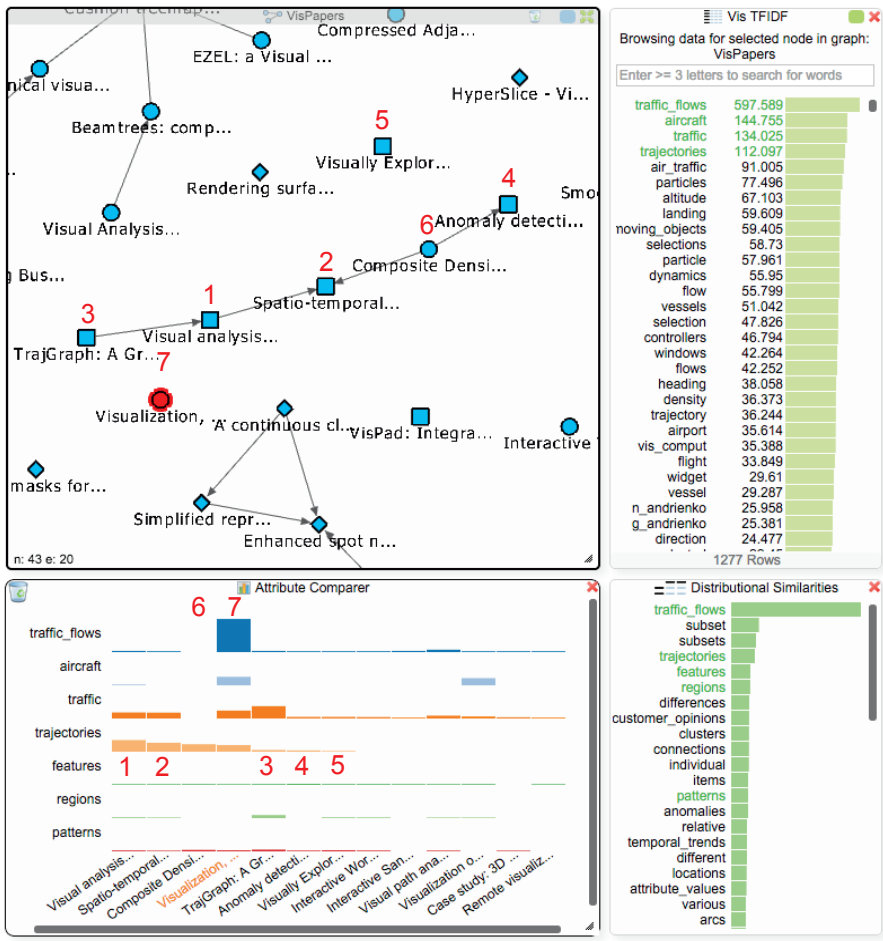



Figure 5.9: The TF-IDF chart now shows all values for the same focus paper (7; currently selected in the paper network view) as in Figure 5.6. Adding all papers mentioning the term "traffic_flows" together with six related terms from the TF-IDF chart and the distributional similarities to the attribute comparer reveals five potentially interesting papers (1–5), which we added to van Wijk's initial paper network view. Four of these papers are connected due to citations and one of van Wijk's papers (6) is also part of the resulting group. Image taken from [175]. ©Informatics.

similarity. This means that we will find terms that have been used in the same way in this view. Such similarities may sometimes be somewhat surprising, and perhaps even counter-intuitive (in this case the terms "subset" and "subsets" are not related to the initial search term), but are a direct effect of word usage in the data at hand.

## 5.4 Discussion

The employment of multiple coordinated views has the advantage that we can explore various interconnected networks at once. This is of course limited to the available screen space. On a 2K desktop resolution, three to four networks in addition to the BoW and distributional similarity charts are a realistic usage scenario (Figure 5.2 and 5.6 were taken at a 1,900 $x$ 1,080 resolution), whereas a 4K resolution offers even more space for additional network views. A higher resolution is essential for bigger document collections, for instance, if users plan to explore the interconnections between various different conference proceedings and would have three or four paper citation networks and not just a single one as we did in our use case.
Analyzing a bigger text corpus should not have a big impact on the performance of the system and the interaction between the views, as all word indices and similarities are calculated beforehand and loaded into the main memory on the OnGraX server. Exploring a larger number of different networks also scales well for realistic use cases. Our mapping system on the server uses multiple parallel threads and can iterate through up to 16 data sets or networks at the same time to find matches. As such, only the highest number of nodes $(n)$ and the number of mapped attributes $(m)$ in a graph have an impact on the performance and a matching runs in $\mathcal{O}(nm)$ time. Additionally, all networks are rendered on the client computer's GPU by using WebGL [59]. This approach is faster than using the more common SVG-based node-link visualization approaches, and our system provides good scalability for up to 10,000 nodes and edges (depending on the hardware specifications of the client). As we use a bottom up approach and only load nodes into the views on demand, the number of nodes that have to be visualized at once is usually quite small and as such also scales well with bigger networks.
While users can quickly specify new attribute mappings between imported graphs and data sets, this process usually requires some a priori knowledge about the structure and attributes of the data. OnGraX currently only visualizes attributes of a graph or data set during the creation of a new mapping in the mapping dialog. Our plan is to improve this view to give users a more intuitive access to this information.

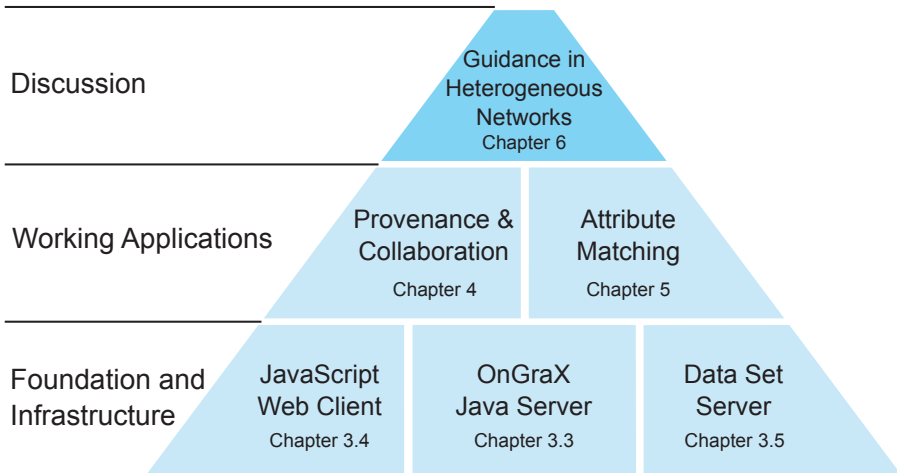
## 5.5 Summary

In this chapter, we presented an extension for OnGraX to explore the structure and relations between heterogeneous interconnected networks and additional metadata and thereby address **Goal 2** of this thesis (see Section 1.1). The inclusion of large textual information, in the form of both a BoW/TF-IDF index and a word similarity matrix, together with the possibility to map this data across different networks, addresses **Goal 2.A**, constitutes a novel contribution in visual text analytics, and allows us to extensively explore relations between related papers, authors, or affiliations based on selected keywords. Being able to search for terms and semantically

similar used words and visualize their usage throughout the complete text corpus can reveal relations between thematically related documents, which would otherwise not be apparent if an analyst could only search through the titles and keywords of a document collection. **Goal 2.B** is addressed by our novel Hub2Go technique. Hub2Go assists users to quickly add (or remove) interconnected nodes in interactive network views and charts, and assists in navigating to the locations of these nodes to explore their local network structure. This approach was developed in close collaboration with NLP experts from the StaViCTA project [109].

*Chapter 6*

# Guided Interaction in Heterogeneous Networks




## Contents


**6.1 Problem Description . . . . . 94**
**6.2 Data for Guidance . . . . . 95**
**6.3 Similarity Calculation . . . . . 97**
**6.4 Visualizing and Navigating Guidance Results . . . . . 98**
6.4.1 Configuring the DOI Function . . . . . 98
6.4.2 Enhanced Hub2Go . . . . . 99
6.4.3 Exploring Values of Multiple Guidance Results in Detail 101
6.4.4 Automatic On-The-Fly Suggestions . . . . . 103
**6.5 Discussion . . . . . 105**
**6.6 Summary . . . . . 105**


Now that we have the ability to connect networks and data sets via a 1:1 mapping of attributes, this chapter presents possible extensions of OnGraX for improved analysis and exploration of connected heterogeneous networks. The focus here lies not on a concrete use case, but on the possibility of quickly connecting different heterogeneous networks and data sets to explore and analyze relations visually. Based

on a focus node, OnGraX should offer some kind of guidance to assist users during their exploration. The matching of attributes between networks and data sets should not only use the 1:1 mappings, which where discussed in the previous chapter. Instead or in addition, OnGraX should suggest interesting nodes on the basis of *degree-of-interest* (DOI) functions [49, 151] which were discussed in Section 2.4. The DOI functions provide suggestions for navigation based on the attributes of nodes in the connected networks. We do not specify concrete DOI functions in this chapter: the proposed design for OnGraX should provide users the possibility to intuitively specify their own DOI functions based on specific attributes and should help users to evaluate these functions for different use cases by visualizing results of DOI calculations and the involved attributes. With this approach, DOI functions are not just used as "black boxes" that give suggestions without insight on how they were derived. We want to give analysts the possibility to explore the guidance suggestions and make an assessment if the DOI settings actually make sense for a specific use case. Examples for use cases could include collaboration or publication networks, biological networks/systems, social networks, or UML-diagrams of complex software architectures.

Please note, that this is a discussion about **additional functionality** for OnGraX and that these features are not implemented yet. All images in this chapter are mockups that serve as a basis for a possible future extension.

## 6.1 Problem Description

In some cases, such as for different co-citation networks, the interconnections are usually straightforward, since certain nodes (e.g., authors, institutions, or papers) can occur in different networks, and the relations are therefore defined via same author names or institutions. However, there are other applications in which previously unknown relations between larger networks and data sets are of interest.
An example are data sets of patients, which are to be connected and analyzed over several systems, for example from online-databases of pharmacies, data from doctor visits, and hospital stays [146]. The simplest use case in this example is to connect the data of a particular patient by using a name or social security number. However, other tasks are also conceivable, such as the precise analysis of a disease based on similar incidents. In this case, various attributes such as the age of a patient or certain ranges of laboratory results, such as blood or hormone values, could be interesting.
The activity of genes and/or proteins that influence cellular processes is another data source that can be used to cluster patient groups and analyze similarities or differences in their data [50]. A challenge here are biological networks (see also Chapter 2.3.2), which are created by so-called high-throughput analysis methods. They are often divided into smaller subsets due to their sheer size and contain many multivariate attributes. Biologists are interested in finding out how different elements

are connected between these networks, by analyzing these multivariate attributes as well as the structure of the networks.
To be able to visually explore such networks and data sets, an analyst should have the possibility to adapt the guidance settings based on the specific task. It should be possible to select different value ranges over various attributes to adapt the guidance calculation, as attributes and their values change drastically for different tasks. For instance, if a biologist wants to analyze various forms of cancer, the settings have to be adapted for each specific type of cancer that is to be analyzed. Additionally, during the analysis the settings could be adapted on the fly, if new insights are gained or a if a specific setting does not yield any interesting results. A typical analysis session with OnGraX, that includes various networks and data sets, could consist of the following activities:

**Data Set Selection** The analyst has to select which networks and data sets should be used during the session.

**Attribute selection** Attributes of interest have to be selected for the calculation of the DOI values.

**Exploration** The analyst starts with exploring a network and related data sets.

**Guidance on Demand** The system performs DOI calculations for specific nodes on demand and shows the related entries throughout the data sets and networks. The analyst choses interesting entities from the suggestions and explores the interconnections between the networks and data sets.

**Value Range Adaption** Based the initial suggestions and insights gained from the exploration, the analyst might adapt the DOI function in such a way that only specific value ranges are used to show more accurate suggestions or to filter out uninteresting entries.

**Continued Exploration and Adaption** The analyst continues to explore the connected networks and data sets and can interactively adapt the DOI calculations and filter the results of the guidance suggestions.

The next section takes a closer look on the various forms of data that could be used to facilitate such a guidance approach. This is followed by a detailed description about the calculation for the DOI functions and a design proposal for the visualization and navigation of the guidance results.

## *6.2 Data for Guidance*

The data required to provide means of guidance is not limited to the existing data of the networks and data sets loaded into the system. Data that is collected during the

analysis processes and user-related data, such as user expertise, can also be used for guidance. Ceneda et al. [25] identify five different sources of information that are categorized as follows:

**Data** includes all kinds of information already available or derivable from the data to be analyzed. These are in our case the networks and data sets itself, the attributes of the nodes and edges of the networks, the network topology, and other deriveable attributes, such as network centralities.

**Domain Knowledge** refers to information that originates from the application domain, such as expert systems, domain models, workflows, or conventions.

**Visualization Images** include the visual data representations and information about mapping parameters. They can be useful for understanding what the user is actually seeing.

**User Knowledge** is about information that users input to the system, including annotations or DOI functions, or other information that the system can infer from the user.

**History** relates to keeping track of interactive changes, including logging all interaction steps, employed algorithms, applied parameterizations, or visited parts of the data.

OnGraX uses the categories *data*, *user knowledge*, and *history* as the basis for the calculation of the guidance suggestions. The already existing multivariate attributes of the networks fall into the *data* category, and we can use additional history-related data that are already collected during the observation and analysis of the networks with OnGraX (e.g., if a user joins a session, changes graph elements, viewports, annotations, chat messages). Moreover, information from individual users, such as their education or current field of research, could be used for guidance by OnGraX, which falls into the *user knowledge* category. The following five attribute categories, that are based on *history* and *user knowledge*, are available through the use of OnGraX' logging functionality:

**Viewtime** The amount of seconds a node was viewed by (specific) users. This value can be calculated by viewing rectangles or based on eye tracking data.

**Number of Selections** How many times did a (specific) user select a node.

**Number of Edits** How many times was a node edited (for instance, node movements or attributes, such as size, color, name, shape changed).

**Number of Navigations** This data is based on previous guidance suggestions. How many times did a user navigate to a related node after inspecting DOI values in another graph.

**User Expertise** Different users have varying levels of expertise and interests. A user could be a seasoned expert or a student. This information could also have an impact on a guidance suggestion.

In many networks, edges also have additional attributes. For example, in a social network, the number of emails or phone calls sent between people can be stored in the attributes of the edges. Since the aim in our case is the navigation between different graphs, and there are usually no interconnecting edges between these graphs, our focus currently lies on node attributes.
To provide an automatic selection for the proposal of nodes, relevant attributes must be mapped from a source node to possible target nodes. This mapping can be done in three different ways:

**1:1 Match** Similar to our approach used in the previous chapter, a specific node is only considered if it has the exact same value for an attribute as the value of the selected source node.

**Selected Value Range** Only include nodes that have attribute values within a specific user selected range.

**Similarity** A calculated similarity, for instance, string similarity in case of textual attributes (e.g., *Levenshtein Distance* [106]) or the similarity of numeric attributes based on the distance of the attribute value of the source node.

Our strategy is to allow users to use all three approaches during an analysis session. This requires an interactive dialog that allows users to quickly determine whether a 1:1 match is desired for certain attributes, a certain value range is required, or whether a similarity calculation is desired. The following section gives details about how the similarity calculation could be done. The interface design for our guidance approach is discussed afterwards.

## 6.3 Similarity Calculation

Given $N$ numeric attributes the basic similarity for each numeric attribute can be calculated as:

$$attribute\ similarity = 1 - \frac{|A_{source} - A_{target}|}{A_{max} - A_{min}} \tag{6.1}$$

where $A_{source}$ is the attribute value of the source node and $A_{target}$ the attribute value of the target node, $A_{max}$ the maximum overall value and $A_{min}$ the minimum overall value of all target attributes. The *degree-of-interest* (DOI) is calculated as the summarized similarity of all attributes for a target node:

$$DOI(target, source) = \frac{\sum_{i=1}^{N} attribute\ similarity_i}{N} \tag{6.2}$$

where $N$ is the total number of configured attributes in the DOI function. The resulting node similarities are sorted in descending order, and only the $x$ highest (depending on the user's choice) results of each target graph are visualized. As we want to have the possibility to have weighted similarities, in case the analyst decides to make one or more particular attributes more important, we calculate a weighted similarity as:

$$weighted\ attribute\ similarity = \left(1 - \frac{|A_{source} - A_{target}|}{A_{max} - A_{min}}\right) * weight \tag{6.3}$$

and the corresponding DOI with the summarized weighted similarity values for a target node as:

$$DOI(target, source) = \frac{\sum_{i=1}^{N} weighted\ attribute\ similarity_i}{W} \tag{6.4}$$

where $W$ is the sum of all weights. For string-based attributes, such as names of authors, we use a string similarity metric, for instance, the Levenshtein distance [106] instead of *attribute similarity*.

## 6.4 Visualizing and Navigating Guidance Results

In this section, we discuss suggestions on how an interface for configuring and interactively adapting DOI calculations can be structured and how the results can be visualized in such a way that the system is able to display relevant nodes to a user and, at the same time, support navigation through several heterogeneous graphs. Additionally, the interface must provide the ability to quickly switch between mapping types (1:1 matching, value range, similarity), and the attributes of each proposed node should be easily comparable so that the effects of the parameter settings for the guidance suggestions can be analyzed in detail.

### 6.4.1 Configuring the DOI Function

Figure 6.1 shows a mockup to configure the attributes for the DOI calculation. With this approach, the matching between graphs and data sets is less strict and offers a more flexible calculation and guidance suggestions of interesting nodes than our previous 1:1 matching approach from Chapter 5. Attributes are no longer explicitly mapped between certain graphs/data sets and certain attributes. The guidance works based on the attributes of a selected focus node, in a graph that a user is currently exploring. In this case, the mockup shows the configuration for a DOI function that uses four node attributes to calculate node similarities across heterogeneous networks: the *name* of a node as string similarity, the *income* attribute, the *age* attribute, and the *viewtime* attribute. The calculation of the DOI results and the

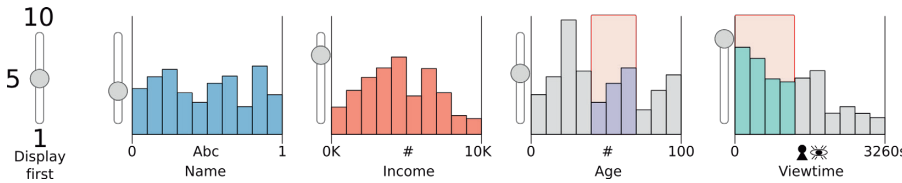



Figure 6.1: Mockup for configuring node attributes for guidance suggestions. The distribution of each attribute in all graphs is visualized as histograms. The user can directly select a specific bin or bin range, which is indicated as red rectangles in this mockup for the *Age* and *Viewtime* attributes. By doing this, the *similarity* calculation for this specific attribute is replaced by a binary 0...1 function, and only nodes whose attributes are within the selected bins are added to the result. Their corresponding similarity value for this attribute is always 1. The slider on the far left of the figure is used to set the number of nodes that should be shown in the enhanced Hub2Go visualization. The other sliders to the left of each attribute are used to set a weight for each attribute. For instance, the user could increase the weight for the viewtime to focus his/her exploration of nodes that have been viewed by other, specific users. The first attribute in this Figure is a string-based attribute (*Abc*) and is calculated as Levenshtein distance with values between 0 and 1.

detailed attribute values for each suggested node can be visualized in three different ways: the *enhanced Hub2Go* technique to show the most similar results, the *ViNCent* view to explore a larger amount of results in detail, and *on-the-fly suggestions on nodes* for an automatic visualization of results directly in a graph view. The three designs are explained in more detail in the following.

### 6.4.2 Enhanced Hub2Go

To show suggested nodes from related graphs and to be able to compare their attributes in detail, we extend our Hub2Go technique from Chapter 5 with an additional visualization that is shown at the border of a graph window after the user clicks on a node. Figure 6.2 shows a mockup of the design. The visualization employs a similar technique that we have chosen for our tool ViNCent [170], which is used for the analysis of network centralities[1]. The enhanced Hub2Go visualization is interactive, which means that the user can hover the mouse over the suggested nodes and their attributes for a better comparison of the values and to quickly view each node and its neighborhood in the relevant target graph. Figure 6.3 shows the enhanced Hub2Go and its interaction possibilities in detail. In this case, it shows the first five results for nodes that were matched from two target graphs based on the DOI function with four attributes. Nodes are represented as wedges and for each of the attributes, a small arc is drawn to represent the attribute values.

[1] ViNCent Video: `https://vimeo.com/128268456`

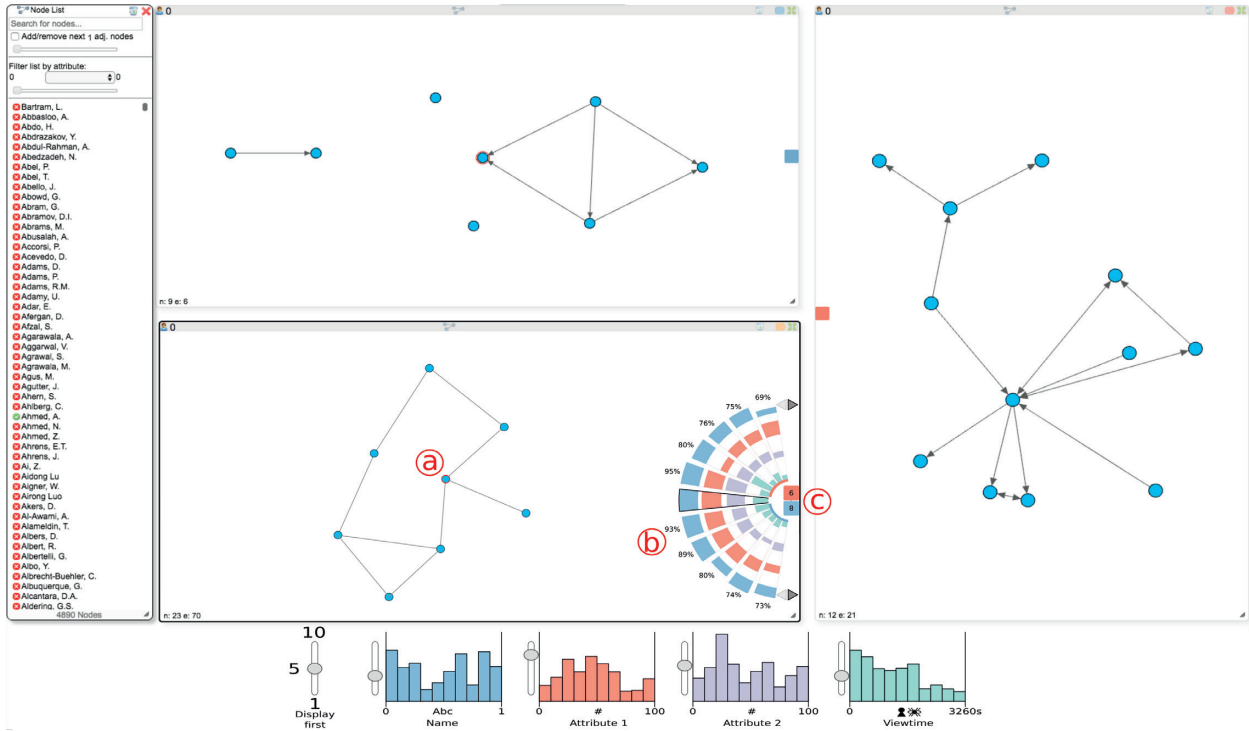



Figure 6.2: Mockup for visualizing guidance across heterogeneous networks. The user initially has to choose which attributes are used for the DOI calculation, which is shown at the bottom of the screen. In this case, four attributes are configured: the name of the node – a string-based attribute, two numerical attributes – that are part of the graph data, and the viewtime of the nodes – which is a numerical attribute selected by OnGraX while users explore the graphs. After clicking on a node in a graph (a), OnGraX shows the most similar nodes that are found in other graphs based on the DOI calculation as enhanced Hub2Go visualization (b+c).

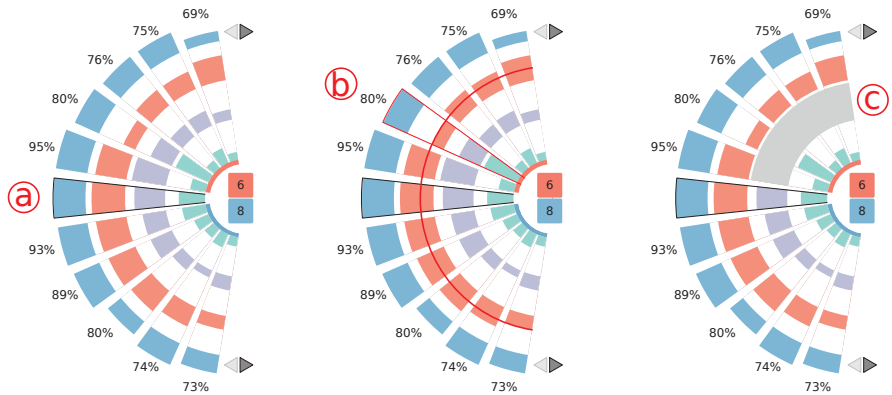



Figure 6.3: Mockup for the Enhanced Hub2Go visualization. The left part shows the hub after the user clicks on a source node. The wedge with a black frame represents the source node (a). The wedges above and below show the five most similar nodes in the two target graphs. The system found 6 nodes for the upper graph and 8 nodes for the graph below, as indicated by the numbers in the two rectangles. In this case, the hub only shows the first 5 results. The triangles on the top and bottom can be used to scroll through the results and visualize the other nodes. Each of the colored arcs shows the attribute value of the corresponding node, allowing the user to quickly compare attribute values of target nodes with the source node. The overall similarity values, compared to the source node are shown as percentages on each of the wedges. Hovering over a specific attribute arc (b) highlights the node in the graph and adds a red circle segment to the hub, which allows to better compare that specific value with the values from the other nodes. If the nodes of a target graph do not have an attribute that is present in the source graph, the hub will show a grey arc instead of the attribute arcs (c).

### 6.4.3 Exploring Values of Multiple Guidance Results in Detail

If a user is interested in more detailed results, and the number of nodes displayed in the Enhanced Hub2Go visualization is not sufficient, a reimplementation of ViNCent in WebGL should be considered. We can use the idea of ViNCent to visualize the calculated values of the DOI functions together with the relationships of nodes in a radial ViNCent view. Figure 6.4 shows a mockup of this visualization design. The view is displayed on the request with the selected source node (a). All matched nodes are displayed in ViNCent with the respective calculated attributes from the DOI function per target node. The interactions are similar to the Enhanced Hub2Go approach, hovering over a wedge adds a circle to the view for a better comparison of that specific attribute value.

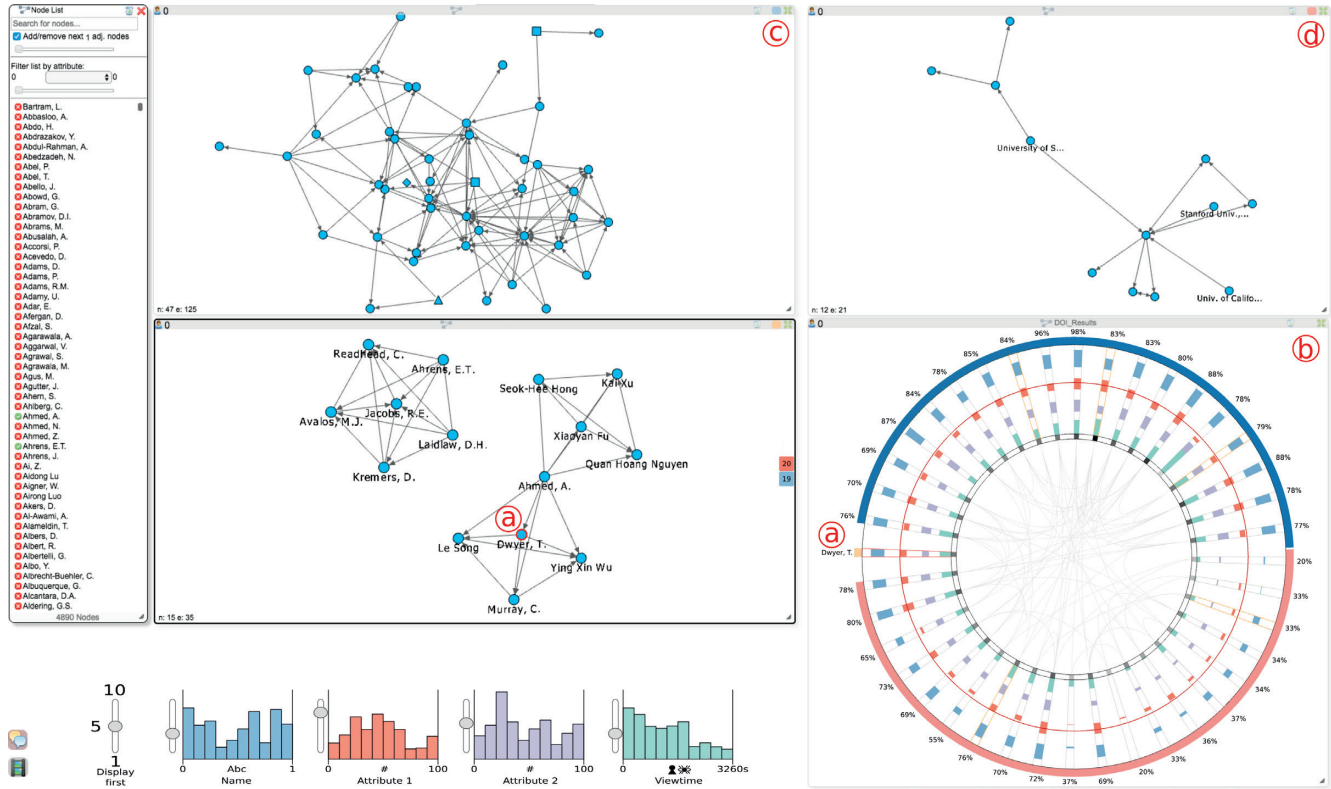



Figure 6.4: Mockup for a visualization that displays the DOI results for all target nodes based on a source node. The attributes of the source node (a) can be compared with the target nodes in the ViNCent view (b). The camera of the other graph visualizations (c+d) automatically centers the corresponding nodes if they are hovered with the mouse in the ViNCent view.

### 6.4.4 Automatic On-The-Fly Suggestions

So far we have discussed a guided exploration approach that only works "on demand". Related nodes from other graphs are only calculated and visualized if a user directly clicks on a specific node of interest. A more fluid approach would be to show relevant suggestions automatically during an exploration session on all nodes that are currently visible to the user. To be able to provide this feature, suggestions have to be automatically calculated in the background while the user explores a graph. In this case, the DOI values are calculated for each node in the current graph view and visualized as wedges around the nodes. Figure 6.5 shows a zoomed-in view of the visualization around the nodes, and an overview for this approach is shown in Figure 6.6. The visualization should be limited to a certain zoom level, as the screen would get too cluttered if the wedges would be visualized on too many nodes. This approach does not work for dense graphs, as the visualization requires enough free space around the nodes. Another challenge is the server-side implementation of the DOI calculation in this case. Calculating the DOI for multiple nodes has to be fast enough to facilitate a real time exploration of the guidance results. A smart caching function to store matching results of multiple nodes would be necessary.

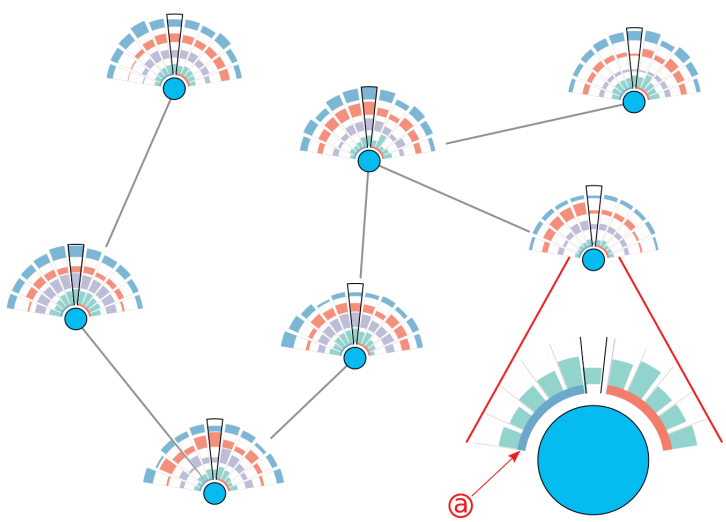



Figure 6.5: Mockup of visualizing the DOI results for each target node directly in the graph visualization around each source node. In this example, the wedges are rotated around the top of the nodes, instead of facing to the left, for aesthetical reasons. The small colored wedges directly at the outer part of each node—see the zoomed-in excerpt at (a)—illustrate the relevant target graphs for each wedge.

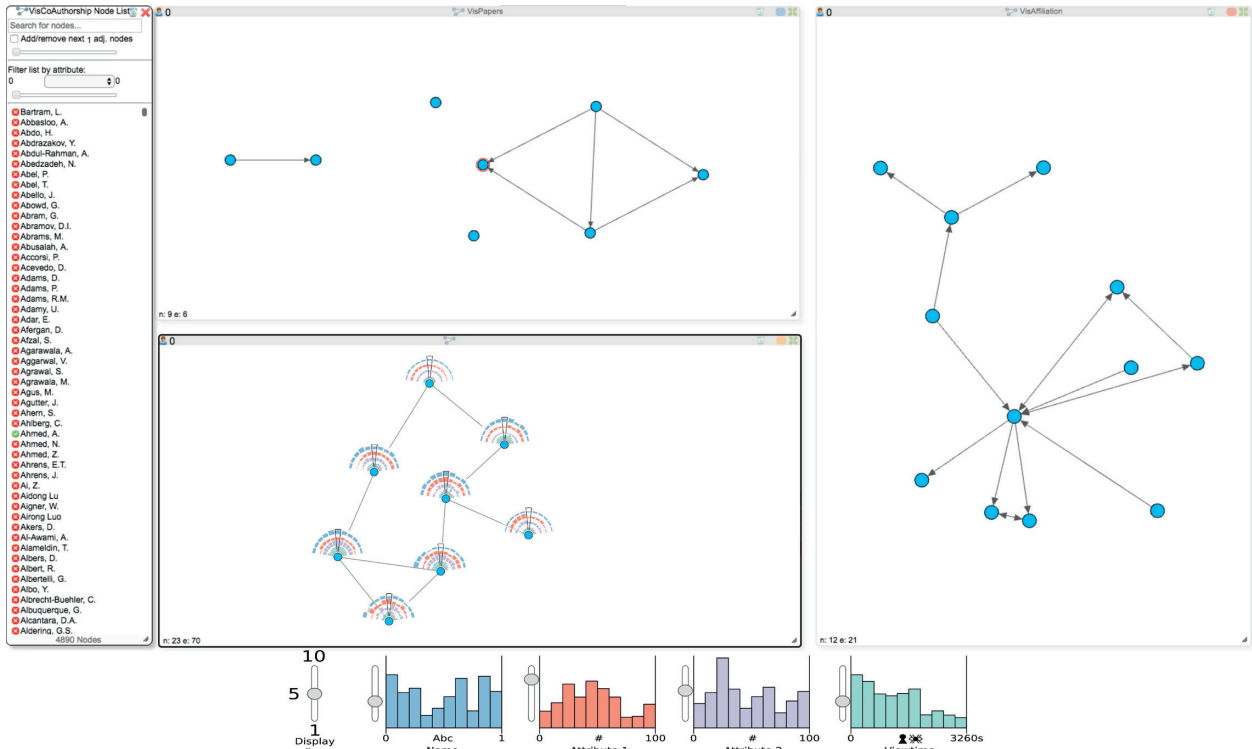



Figure 6.6: Mockup for guiding users on the fly during exploration of a graph. In this case, the user is actively panning the camera in the bottom left graph, and OnGraX calculates the DOI values for all nodes on the fly. The results are then visualized on the nodes in the source graph, allowing the user to compare the attributes of the source node to relevant target nodes. The visualization on the nodes provides the same highlight functionality as the previous approaches: the camera in the target graph automatically centers the corresponding target node which allows the user to examine the target graphs structure.

## 6.5 Discussion

The proposed designs allow users to quickly define and test new DOI functions for various use cases. However, the DOI calculation for the interactive guidance currently focuses on numerical attributes and string similarities for text-based attributes. We did not yet consider categorical attributes (such as gender or car brands) in our current design. A common approach is to assign a sequence of numbers to the categorical values [158]. Once the mapping is done, it can be treated as numerical data. This would require an additional dialog to perform such a mapping.
The Enhanced Hub2Go visualization at the border of the graph views does have some limitations regarding the number of interconnected nodes that can be shown at the same time. The design supports scrolling through nodes, when a guidance result contains more than 5 nodes (see Figure 6.3). Alternatively, the degree of individual wedges for each node could also be reduced, to display more results. This would also be necessary if a guidance result calculates relationships between more than three graphs (a source graph and two target graphs). The number of related networks is limited to two target graphs in the current designs and could get too cluttered if the results for multiple target graphs have to be visualized. Displaying the hubs directly on a node (see Figure 6.6), does increase the possible number of wedges and thereby also the number of target graphs, as they can be drawn in a complete circle around each node. However, this would also increase the clutter in the graph visualization.

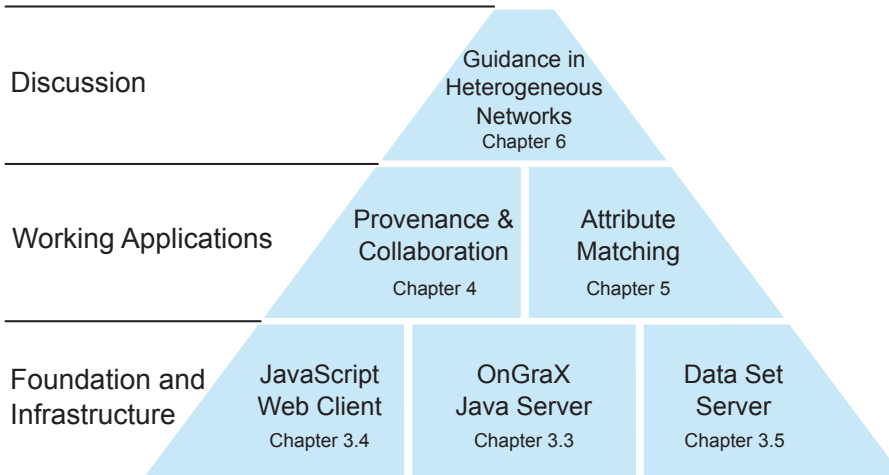
## 6.6 Summary

In this chapter, we presented possible designs for interactive guidance across heterogeneous networks. We proposed visualization and interaction techniques that allow users to define variable DOI functions depending on different use cases. These DOI functions are then used to determine relevant nodes in other graphs. By using OnGraX as a foundation, we are able to calculate the degree of interest not only for attributes that are already included in the graph data, but also for attributes that are based on additional data sets that can be combined with the graphs. Furthermore, we can also use provenance information from users, such as the view times, navigation interactions between graphs, or edit operations on nodes (node movements, resizing, etc.) to calculate the degree of interest of related nodes.
We proposed *three different designs* to visualize the results of the DOI calculation which are used to guide the user in choosing relevant nodes for further exploration. The first design is the Enhanced Hub2Go visualization, which shows guidance results and details about node attributes as wedges at the border of a graph visualization on demand after the user selects an interesting source node. The second design is based on a reimplementation of ViNCent for OnGraX. With the ViNCent view, the attributes of multiple relevant nodes and their neighborhood can be evaluated

by analysts in detail. The third proposed design enables users to explore relationships across graphs on the fly by adding small wedges with attribute visualizations for nodes in related graphs directly into the source graph visualization. With these designs, we achieve **Goals 3.A and 3.B** of this thesis (see Section 1.1).

*Chapter 7*

# Conclusions and Outlook




## *7.1 Conclusions*

The following section offers an overview of the research results in context to the goals defined in Chapter 1. The scientific contributions in the field of graph visualization and interaction are summarized, and open challenges and issues are discussed.

The visualization of large and complex networks is still one of the most challenging tasks in the information visualization and graph drawing communities [105]. Collaborative work on networks plays an important role as the amount of available data continues to increase. The challenge here lies in the fast and seamless provision of tools to enable scientists to analyze networks together. It is important that collaborators can perform joint work without any problems in order to be able to communicate findings easily. This process is very complex, since usually special tools have to be installed before a collaborative session can take place. In addition, current tasks in network analysis, especially in the life sciences, require the visualization and analysis of several large networks and data sets, which are also interconnected with each other. Up to now, no system has been able to meet the high demands for a high-performance visualization of these networks.

**Performant collaborative graph visualization** In this thesis, the distributed collaborative analysis of heterogeneous networks was discussed. In the course of the work, the system OnGraX for synchronous and asynchronous analysis of node-link diagrams was designed and implemented, which is discussed in Chapter 3. This addresses **Goal 1.A**. The main contribution is the easily accessible graph environment provided by a web application, through which collaborators can start a session to work together without much effort.
One challenge was the high-performance display of larger graphs with up to 10,000 nodes and edges in the web browser, which could be realized by using WebGL. Compared to other network visualization libraries, OnGraX has the advantage that nodes with different shapes, colors, and labels can be displayed with high performance. Many WebGL frameworks for the visualization of graphs only offer standard shapes and rudimentary support for text rendering, which leads to performance problems for large graphs with several thousand nodes.

**Real time interactions with WebSockets** WebSockets are now an integral part of current JavaScript libraries and frameworks, as they provide fast communication between clients and servers and generate less overhead than traditional AJAX calls. At the beginning of the implementation of OnGraX, this technique was not yet very widespread—OnGrax was among the first systems to make use of this new technique, which provided users with a real time view of the mouse and camera positions of other collaborators in a graph session.

**Accessible data provenance and annotations** In order to help users to follow the interactions of other users in real time and to suggest and discuss further changes, appropriate features have been implemented in OnGraX, which are described in detail in Chapter 4. OnGraX logs all executed actions and viewed areas of the graphs to allow users to easily find already edited parts of the graphs, or areas that require additional editing, which tackles **Goal 1.B** of this thesis. Users can also leave annotations for subsequent users directly in the graph visualization to communicate necessary work other users have to perform or to explain already finished adjustments to the data. This eliminates the time-consuming process of writing e-mails with screenshots of the graphs to explain and discuss required work in detail.

**Visualizing data in graph visualizations with heat maps** Based on a user study, we have shown that additional information can be effectively displayed in graph visualizations as heat maps. Especially for graphs, which already visualize attributes by different node shapes, node sizes and node colors, heat maps provide a further possibility for additional data visualization in graphs. We use this technique in OnGraX to display changes on graph elements, or alternatively to mark areas that were viewed by other users. The visualization of these provenance informations addresses **Goal 1.C**. In the future it is also conceivable to visualize other already existing nodes or edge attributes in this way.

✔ **Goal 1:** Design and implementation of an online graph visualization system to enable collaborative work on graphs.

**G 1.A**: provide interactive visualizations to perform synchronous and asynchronous collaborative analyses of node-link diagrams,

**G 1.B**: store all relevant user actions and track which graph objects were viewed by the users, and

**G 1.C**: give other users options to visually explore this stored provenance information.

**Extendable system for various use cases with heterogeneous networks and data sets** Another challenge was to build the architecture of OnGraX in such a way that the system is easily expandable. The implementation forms the basis for additional extensions and use cases. This allows us to address **Goal 2.A**. Users can load large data sets into OnGraX, which have relations to heterogeneous graphs, and can interactively create implicit connections to other related graphs for further analysis.

**Exploration of scholarly networks with full text corpora** OnGraX offers users the possibility to explore scholarly networks in detail. This includes the analysis of the full text corpus of these networks, which has the advantage that users can not only perform a search for keywords based on the paper meta meta, but also for similar terms that occur in the full text of the papers. In our specific case, we used a publication network, a co-author network, and an affiliation network, as well as a word similarity matrix, consisting of the complete text corpus of all papers from the Vis conference. Users are given the opportunity to analyze relevant publications and the associated authors and affiliates in more detail. In addition, the use of the word similarity matrix offers the possibility to find further relevant papers in the data set due to similar sentence constructions for certain keywords, which could be interesting for a certain topic. Exploring these relationships with various visualization widgets tackles **Goal 2.B**. A detailed use case for this approach is explained in Chapter 5.

✔ **Goal 2:** Extension of the graph visualization system to support specification and exploration of relationships between heterogeneous networks and additional data sets.

**G 2.A**: features for an interactive specification of individual relationships between different networks and additional data sets, and

**G 2.B**: navigation between these mapped relationships with the help of visualization widgets.

**Guided interaction in heterogeneous networks** The existing implementation of OnGraX provides the basis for advanced techniques to facilitate the exploration of heterogeneous networks with the help of guidance technologies. In Chapter 6, we propose design and interaction suggestions that allow analysts to quickly define and evaluate their own degree-of-interest functions for various tasks in heterogeneous networks and thereby achieve **Goal 3.A**. The interface allows analysts to calculate and display previously unknown relations between networks and data sets, and to quickly switch back and forth between the network visualizations to explore the closer structure of the neighborhood of relevant nodes, which fulfills **Goal 3.B**.

✔ **Goal 3:** Design of a visualization system to be able to specify and test configurations for guidance in heterogeneous networks for various use cases.

**G 3.A**: visualize guidance results and use them to navigate across networks, and

**G 3.B**: provide visualization widgets to show the computed values of the guidance results in order to give researchers the possibility to assess their usefulness and specify various settings for different use cases.

**Summary of contributions** The three goals set out in Chapter 1 are fulfilled. The resulting scientific contributions of the research described in this thesis are:

1. Design and implementation of the flexible web-based tool OnGraX for the interactive visualization of node link diagrams with up to 10.000 nodes and edges. Furthermore, the system stores provenance information of analysis sessions which assists users in performing synchronous and asynchronous distributed collaboration.

2. Interaction and visualization techniques for collaborative editing of node link diagrams, including highlighting of nodes with heat maps, annotations for graph objects and regions, and a timeline function, which allows users to view old states of a graph and to replay changes that have already been performed on a graph.

3. OnGraX allows the connection of external applications for a seamless and automatic upload. This allows researchers to quickly visualize and explore large graphs in a web browser.

4. An extension of the system to allow users to interactively specify 1:1 relationships between heterogeneous networks and additional data sets. This includes the novel visualization technique Hub2Go, which is used to assist the user in exploring these relationships.

5. A design proposal for an additional extension that allows guided interaction in heterogeneous networks by allowing users to specify DOI functions to connect interesting nodes across multiple networks on demand for different use cases. The detailed results of the DOI functions can be scrutinized with an enhanced Hub2Go visualization technique and a web-based reimplementation of ViNCent.

## 7.2 Outlook

**Implementation** The design proposals for guidance across heterogeneous networks discussed in the previous chapter have yet to be implemented. In addition to the visual design, a high-performance implementation of the DOI calculation is also necessary. Especially for the *on the fly* approach, smart caching functions are necessary to calculate the guidance proposals for several nodes in the view fast enough to be able to display them in real time while the user explores the graph.
Furthermore, an extension of the available dialogs and improvement of the graph interactions in OnGraX should be done. For example, the dialogs for importing graphs and data sets can be extended and improved. Moreover, the editing possibilities of graphs in OnGraX are still quite limited, since our focus was on small adjustments of existing graphs. For more extensive tasks, the interface should be adapted accordingly.

**Evaluation** We performed a detailed user study for the visualization of user provenance data in form of a heat map and an expert review for the collaborative features of OnGraX, but not all functionalities of OnGrax have been evaluated in detail.
Subsequently, a user study to evaluate the interactive guidance visualizations, the Hub2Go technique, and the dialogs for the DOI specification has to be carried out. In addition, concrete use cases in the various application areas should be examined in detail by experts to determine the usability of different DOI specifications.

**Social implications and ethical issues** Although collaborative work can often be advantageous, users sometimes want to work on their own without having their work observed and followed by others. Independent data analysis by just one person may also sometimes provide better results [11]. Since the implementation of OnGraX is focused on collaborative work, the possibilities to perform a graph session independently from other persons and not to track the analysis steps and changes made to the networks are limited. The only possibility in this case is to create an independent user group with only one user, so that other participants have no access to the graph(s). If the result should later be made available to several people without including the logged changes, all edited graphs would first have to be exported from OnGraX and imported again, since there is currently no possibility in the user interface to delete the logs.

The logging of user data also raises questions about data security, especially in respect to the recently passed General Data Protection Regulation (EU-DSGVO). Users must be expressly informed which personal data is stored by OnGraX. Additionally, users may also not want their interactions to be recorded and tracked by others. Disabling this feature is not yet possible for all modes in the current version of OnGraX.

**Application areas** The features of OnGraX can be of particular interest in the areas of social sciences and life sciences. For the former, OnGraX offers the possibility to analyze scientific and scholarly networks in detail. Due to the possibilities for an accurate semantic analysis of text corpora, it can be used to explore citation networks of papers and journals and collaboration networks of authors and institutions and could be of interest to libraries and students.
In the life sciences, OnGraX offers with its features the possibility of international cooperation in the revision of graphs that depict biological processes. We already collaborate with the bioinformatics research group around Prof. Dr. Hans-Peter Lenhof at the Center for Bioinformatics at Saarland University in Saarbrücken, Germany. They use OnGraX together with their GeneTrail2 system as an external application for the visualization and analysis of biological processes.

# Bibliography


[1] James Abello, Steffen Hadlak, Heidrun Schumann, and Hans-Jörg Schulz. A modular degree-of-interest specification for the visual analysis of large dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):337–350, March 2014.

[2] James Abello and Frank van Ham. Matrix zoom: A visual interface to semi-external graphs. In Matthew O. Ward and Tamara Munzner, editors, *Proceedings of the IEEE Symposium on Information Visualization*, INFOVIS '04, pages 183–190. IEEE Computer Society, 2004.

[3] Mario Albrecht, Andreas Kerren, Karsten Klein, Oliver Kohlbacher, Petra Mutzel, Wolfgang Paul, Falk Schreiber, and Michael Wybrow. On open problems in biological network visualization. In *Proceedings of the 17th International Conference on Graph Drawing*, volume 5849 of *GD'09*, pages 256–267, Berlin, Heidelberg, 2010. Springer-Verlag.

[4] Alexis Jacomy. sigma.js. `http://sigmajs.org`. Accessed December 2018.

[5] Basak E. Alper, Nathalie Henry Riche, and Tobias Hollerer. Structuring the space: A study on enriching node-link diagrams with visual references. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems*, CHI '14, pages 1825–1834, New York, NY, USA, 2014. ACM.

[6] Andrei Kashcha. VivaGraphJS. `https://github.com/anvaka/VivaGraphJS`. Accessed December 2018.

[7] Daniel Archambault, Tamara Munzner, and David Auber. Topolayout: Multilevel graph layout by topological features. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):305–317, March 2007.

[8] David Auber. Tulip: Data visualization software. In Petra Mutzel, Michael Jünger, and Sebastian Leipert, editors, *Graph Drawing*, pages 435–437, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg.

[9] Christina Backes, Andreas Keller, Jan Kuentzer, Benny Kneissl, Nicole Comtesse, Yasser A. Elnakady, Rolf Müller, Eckart Meese, and Hans-Peter Lenhof. Genetrail—advanced gene set enrichment analysis. *Nucleic Acids Research*, 35:186–192, April 2007.

[10] Ronald M. Baecker. *Readings in GroupWare and Computer-Supported Cooperative Work: Assisting Human-Human Collaboration*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1st edition, 1994.

[11] Aruna D. Balakrishnan, Susan R. Fussell, Sara Kiesler, and Aniket Kittur. Pitfalls of information access with visualizations in remote collaborative analysis. In *Proceedings of the 2010 ACM Conference on Computer Supported Cooperative Work*, CSCW '10, pages 411–420, New York, NY, USA, 2010. ACM.

[12] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. Gephi: An open source software for exploring and manipulating networks. In Eytan Adar, Matthew Hurst, Tim Finin, Natalie S. Glance, Nicolas Nicolov, and Belle L. Tseng, editors, *Proceedings of the Third International Conference on Weblogs and Social Media*, San Jose, California, USA, May 2009. The AAAI Press.

[13] Vladimir Batagelj and Andrej Mrvar. *Pajek - Analysis and Visualization of Large Networks*, volume 2265 of *Lecture Notes in Computer Science*. Springer, 2002.

[14] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 1998.

[15] Jacques Bertin. *Semiology of Graphics*. University of Wisconsin Press, 1983.

[16] Anastasia Bezerianos, Fanny Chevalier, Pierre Dragicevic, Niklas Elmqvist, and Jean-Daniel Fekete. Graphdice: A system for exploring multivariate social networks. In *Proceedings of the Eurographics / IEEE - VGTC Conference on Visualization*, volume 29 of *EuroVis'10*, pages 863–872, Chichester, UK, 2010. The Eurographs Association & John Wiley & Sons, Ltd.

[17] Mustafa Bilgic, Louis Licamele, Lise Getoor, and Ben Shneiderman. D-dupe: An interactive tool for entity resolution in social networks. In *IEEE Symposium on Visual Analytics Science and Technology*, pages 43–50, October 2006.

[18] Nazli Bilgic and Sophia-Kyriaki Vulgari. Comparison of two eye trackers for the visualization of eye tracking data in node-link diagrams (thesis). `http://urn.kb.se/resolve?urn=urn:nbn:se:lnu:diva-55733`, 2016. Linnaeus University, Department of Computer Science, Sweden.

[19] Ljudmilla Borisjuk, Mohammad-Reza Hajirezaei, Christian Klukas, Hardy Rolletschek, and Falk Schreiber. Integrating data from biological experiments into metabolic networks with the dbe information system. *In Silico Biology*, 5(2):93–102, 2005.

[20] Ulrik Brandes, Markus Eiglsperger, Jürgen Lerner, and Christian Pich. Graph markup language (graphml). In *Handbook of Graph Drawing and Visualization*. Chapman and Hall/CRC, October 2016.

[21] Cynthia A. Brewer. Colorbrewer. `http://colorbrewer2.org/`. Accessed December 2018.

[22] Michael Burch, Corinna Vehlow, Natalia Konevtsova, and Daniel Weiskopf. Evaluating partially drawn links for directed graph edges. In *Proceedings of the 19th International Conference on Graph Drawing*, GD'11, pages 226–237, Berlin, Heidelberg, 2012. Springer-Verlag.

[23] Stuart K. Card, Jock D. Mackinlay, and Ben Shneiderman, editors. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[24] Sheelagh Carpendale. Evaluating information visualizations. In Andreas Kerren, John T. Stasko, Jean-Daniel Fekete, and Chris North, editors, *Information Visualization*, pages 19–45. Springer-Verlag, Berlin, Heidelberg, 2008.

[25] Davide Ceneda, Theresia Gschwandtner, Thorsten May, Silvia Miksch, Hans-Jorg Schulz, Marc Streit, and Christian Tominski. Characterizing guidance in visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 23(1):111–120, January 2017.

[26] Davide Ceneda, Theresia Gschwandtner, Thorsten May, Silvia Miksch, Marc Streit, and Christian Tominski. Guidance or no guidance? A decision tree can help. In *Proceedings of the 9th International EuroVis Workshop on Visual Analytics (EuroVA)*, pages 19–23. Eurographics Digital Library, Eurographics Digital Library, 2018.

[27] Chair of Bioinformatics, Saarland University. Genetrail 2. `https://genetrail2.bioinf.uni-sb.de/start.html/`. Accessed December 2018.

[28] Duen Horng Chau, Aniket Kittur, Jason I. Hong, and Christos Faloutsos. Apolo: Making sense of large network data by combining rich user interaction and machine learning. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 167–176, New York, NY, USA, 2011. ACM.

[29] Chaomei Chen. *Information Visualization. Beyond the Horizon*. Springer-Verlag, London Berlin Heidelberg, 2nd edition, 2004.

[30] Francine Chen, Patrick Chiu, and Seongtaek Lim. Topic modeling of document metadata for visualizing collaborations over time. In *Proceedings of the 21st International Conference on Intelligent User Interfaces*, IUI '16, pages 108–117. ACM, 2016.

[31] Mei C. Chuah and Steven F. Roth. Visualizing common ground. In *Proceedings of the International Conference on Information Visualization*, IV '03, pages 365–372, Washington, DC, USA, July 2003. IEEE Computer Society.

[32] Jason Chuang, Christopher D. Manning, and Jeffrey Heer. "Without the clutter of unimportant words": Descriptive keyphrases for text visualization. *ACM Transactions on Computer-Human Interaction*, 19:1–29, 2012.

[33] Andy Cockburn, Amy Karlson, and Benjamin B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys (CSUR)*, 41(1):2:1–2:31, January 2009.

[34] Christopher Collins, Gerald Penn, and Sheelagh Carpendale. Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1009–1016, November 2009.

[35] Carlos D. Correa and Kwan-Liu Ma. Visualizing social networks. In Charu Aggarwal, editor, *Social Network Data Analytics*, pages 307–326. Springer, 2011.

[36] Giuseppe Di Battista, Peter Eades, Roberto Tamassia, and Ioannis G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall, 1999.

[37] Josep Díaz, Jordi Petit, and Maria Serna. A survey of graph layout problems. *ACM Comput. Surv.*, 34(3):313–356, September 2002.

[38] Andreas Dieberger, Paul Dourish, Kristina Höök, Paul Resnick, and Alan Wexelblat. Social navigation: Techniques for building more usable systems. *Interactions*, 7(6), November 2000.

[39] Alan Dix, Janet E. Finlay, Gregory D. Abowd, and Russell Beale. *Human-Computer Interaction*. Prentice Hall, London, UK, 3rd edition, 2003.

[40] Tim Dwyer, Seok-Hee Hong, Dirk Koschützki, Falk Schreiber, and Kai Xu. Visual analysis of network centralities. In Kazuo Misue, Kozo Sugiyama, and Jiro Tanaka, editors, *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation (APVis'06)*, pages 189–198, Darlinghurst, Australia, 2006. Australian Computer Society, ACM International Conference Proceeding Series, vol. 164.

[41] Peter Eades. A heuristic for graph drawing. In *Congressus Numerantium*, volume 42, pages 149–160, 1984.

[42] Clarence A. Ellis and Simon J. Gibbs. Concurrency control in groupware systems. *SIGMOD Rec.*, 18(2):399–407, June 1989.

[43] Niklas Elmqvist, Thanh-Nghi Do, Howard Goodell, Nathalie Henry, and Jean-Daniel Fekete. Zame: Interactive large-scale graph visualization. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 215–222, March 2008.

[44] Paolo Federico, Florian Heimerl, Steffen Koch, and Silvia Miksch. A survey on visual approaches for analyzing scientific literature and patents. *IEEE Transactions on Visualization and Computer Graphics*, 23(9):2179–2198, September 2017.

[45] Wim Fikkert, Marco D'Ambros, Torsten Bierz, and T.J. Jankun-Kelly. Interacting with visualizations. In Kerren et al. [95], pages 77–162.

[46] The Apache Software Foundation. Apache wave. `https://incubator.apache.org/wave/`. Accessed December 2018.

[47] Mathias Frisch and Raimund Dachselt. Visualizing offscreen elements of node-link diagrams. *Information Visualization*, 12(2):133–162, 2013.

[48] Thomas M. J. Fruchterman and Edward M. Reingold. Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21(11):1129–1164, November 1991.

[49] George W. Furnas. Generalized fisheye views. *SIGCHI Bull.*, 17(4):16–23, April 1986.

[50] Nils Gehlenborg, Seán I. O'Donoghue, Nitin S Baliga, Alexander Goesmann, Matthew A Hibbs, Hiroaki Kitano, Oliver Kohlbacher, Heiko Neuweger, Reinhard Schneider, Dan Tenenbaum, and Anne-Claude Gavin. Visualization of omics data for systems biology. *Nature Methods*, 7:S56, March 2010.

[51] Joseph Gentle. Sharejs – live concurrent editing in your app. `http://sharejs.org`. Accessed December 2018.

[52] Mohammad Ghoniem, Jean-Daniel Fekete, and Philippe Castagliola. A comparison of the readability of graphs using node-link and matrix-based representations. In *Proceedings of the IEEE Symposium on Information Visualization*, INFOVIS '04, pages 17–24, Washington, DC, USA, 2004. IEEE Computer Society.

[53] Helen Gibson, Joe Faith, and Paul Vickers. A survey of two-dimensional graph layout techniques for information visualisation. *Information Visualization*, 12(3-4):324–357, 2013.

[54] Stefan Gladisch, Heidrun Schumann, and Christian Tominski. Navigation recommendations for exploring hierarchical graphs. In *Advances in Visual Computing: 9th International Symposium, ISVC 2013, Rethymnon, Crete, Greece, July 29-31, 2013. Proceedings, Part II*, pages 36–47. Springer Berlin Heidelberg, 2013.

[55] Carsten Görg, Zhicheng Liu, Jaeyeon Kihm, Jaegul Choo, Haesun Park, and John Stasko. Combining computational analyses and interactive visualization for document exploration and sensemaking in Jigsaw. *IEEE Transactions on Visualization and Computer Graphics*, 19(10):1646–1663, 2013.

[56] Carsten Görg, Mathias Pohl, Ermir Qeli, and Kai Xu. Visual representations. In Kerren et al. [95], pages 163–230.

[57] Liang Gou, Xiaolong Zhang, Airong Luo, and Patricia F. Anderson. Socialnetsense: Supporting sensemaking of social and structural features in networks with interactive visualization. In *Proceedings of the IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 133–142, October 2012.

[58] Brynjar Gretarsson, Svetlin Bostandjiev, John O'Donovan, and Tobias Höllerer. Wigis: A framework for scalable web-based interactive graph visualizations. *Graph Drawing*, pages 1–12, 2010.

[59] Khronos Group. WebGL Specification. Editor's Draft 1 July 2014. `http://www.khronos.org/registry/webgl/specs/latest`. Accessed December 2018.

[60] Carl Gutwin and Saul Greenberg. Design for individuals, design for groups: Tradeoffs between power and workspace awareness. In *Proceedings of the 1998 ACM Conference on Computer Supported Cooperative Work*, CSCW '98, pages 207–216, New York, NY, USA, 1998. ACM.

[61] Stefan Hachul and Michael Jünger. Drawing large graphs with a potential-field-based multilevel algorithm. In *Proceedings of the 12th International Conference on Graph Drawing*, GD '04, pages 285–295, Berlin, Heidelberg, 2004. Springer-Verlag.

[62] Stefan Hachul and Michael Jünger. An experimental comparison of fast algorithms for drawing general large graphs. In Patrick Healy and Nikola S. Nikolov, editors, *Graph Drawing*, pages 235–250, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[63] Kenneth M. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 17:219–229, November 1970.

[64] Harry Halpin, David J. Zielinski, Rachael Brady, and Glenda Kelly. Exploring semantic social networks using virtual reality. In Amit Sheth, Steffen Staab, Mike Dean, Massimo Paolucci, Diana Maynard, Timothy Finin, and Krishnaprasad Thirunarayan, editors, *The Semantic Web – ISWC*, pages 599–614, Berlin, Heidelberg, 2008. Springer Berlin Heidelberg.

[65] David Harel and Yehuda Koren. Graph drawing by high-dimensional embedding. In *Revised Papers from the 10th International Symposium on Graph Drawing*, GD '02, pages 207–219, London, UK, 2002. Springer-Verlag.

[66] Mountaz Hascoët and Pierre Dragicevic. Interactive graph matching and visual comparison of graphs and clustered graphs. *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI '12)*, pages 522–529, 2012.

[67] Jeffrey Heer and Maneesh Agrawala. Design considerations for collaborative visual analytics. *Information Visualization*, 7(1):49–62, March 2008.

[68] Jeffrey Heer and Adam Perer. Orion: A system for modeling, transformation and visualization of multidimensional heterogeneous networks. *Information Visualization*, 13(2):111–133, December 2012.

[69] Jeffrey Heer and Ben Shneiderman. Interactive dynamics for visual analysis. *Commun. ACM*, 55(4):45–54, April 2012.

[70] Jeffrey Heer, Frank van Ham, Sheelagh Carpendale, Chris Weaver, and Petra Isenberg. Creation and collaboration: Engaging new audiences for information visualization. In Andreas Kerren, John T. Stasko, Jean-Daniel Fekete, and Chris North, editors, *Information Visualization*, pages 92–133. Springer-Verlag, Berlin, Heidelberg, 2008.

[71] Jeffrey Heer, Fernanda B. Viégas, and Martin Wattenberg. Voyagers and voyeurs: Supporting asynchronous collaborative information visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 1029–1038, New York, NY, USA, 2007. ACM.

[72] Ilmari Heikkinen. Animating a Million Letters Using Three.js. `http://www.html5rocks.com/en/tutorials/webgl/million_letters/`. Accessed December 2018.

[73] Florian Heimerl, Qi Han, Steffen Koch, and Thomas Ertl. CiteRivers: Visual analytics of citation patterns. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):190–199, January 2016.

[74] Nathalie Henry and Jean-Daniel Fekete. MatrixExplorer: A dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):677–684, September 2006.

[75] Nathalie Henry and Jean-Daniel Fekete. MatLink: Enhanced matrix visualization for analyzing social networks. In Cécilia Baranauskas, Philippe Palanque, Julio Abascal, and Simone Diniz Junqueira Barbosa, editors, *Human-Computer Interaction – INTERACT 2007*, pages 288–302, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[76] Nathalie Henry, Jean-Daniel Fekete, and Michael J. McGuffin. NodeTrix: A hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, 2007.

[77] Nathalie Henry Riche. *Exploring Social Networks with Matrix-based Representations*. PhD thesis, University of Sydney, July 2008.

[78] Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Halszka Jarodzka, and Joost van de Weijer, editors. *Eye Tracking – A comprehensive guide to methods and measures*. Oxford University Press, 2011.

[79] Danny Holten and Jarke J. van Wijk. A user study on visualizing directed edges in graphs. In *Proceedings of the SIGCHI Conference on Human factors in Computing Systems (CHI '09)*, pages 2299–2308, New York, New York, USA, 2009. ACM Press.

[80] Weidong Huang, Peter Eades, and Seok-Hee Hong. Measuring effectiveness of graph visualizations: A cognitive load perspective. *Information Visualization*, 8(3):139–152, June 2009.

[81] Weidong Huang, Seok-Hee Hong, and Peter Eades. Layout effects on sociogram perception. In Patrick Healy and Nikola S. Nikolov, editors, *Graph Drawing*, pages 262–273, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[82] Petra Isenberg, Niklas Elmqvist, Daniel Cernea, Jean Scholtz, Kwan-Liu Ma, and Hans Hagen. Collaborative visualization: Definition, challenges, and research agenda. *Information Visualization*, 10(4):310–326, October 2011.

[83] Petra Isenberg and Danyel Fisher. Collaborative brushing and linking for co-located visual analytics of document collections. *Computer Graphics Forum*, 28(3):1031–1038, June 2009.

[84] Petra Isenberg, Florian Heimerl, Steffen Koch, Tobias Isenberg, Panpan Xu, Charles D. Stolper, Michael Sedlmair, Jian Chen, Torsten Möller, and

John Stasko. Vispubdata.org: A metadata collection about IEEE visualization (VIS) publications. *IEEE Transactions on Visualization and Computer Graphics*, 23(9):2199–2206, September 2017.

[85] Petra Isenberg, Anthony Tang, and Sheelagh Carpendale. An exploratory study of visual information analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*, pages 1217–1226, New York, NY, USA, 2008. ACM.

[86] Björn Junker, Dirk Koschutzki, and Falk Schreiber. Exploration of biological network centralities with CentiBiN. *BMC Bioinformatics*, 7(1):219, 2006.

[87] Björn H. Junker and Falk Schreiber. *Analysis of Biological Networks*. Wiley Series on Bioinformatics, Computational Techniques and Engineering. Wiley-Interscience, March 2008.

[88] Ilir Jusufi. *Multivariate Networks : Visualization and Interaction Techniques*. PhD thesis, Linnaeus University, Department of Computer Science, 2013.

[89] Ilir Jusufi, Yang Dingjie, and Andreas Kerren. The network lens: Interactive exploration of multivariate networks using visual filtering. In *Proceedings of the 14th International Conference Information Visualisation*, IV '10, pages 35–42, Washington, DC, USA, 2010. IEEE Computer Society.

[90] Ilir Jusufi, Andreas Kerren, and Björn Zimmer. Multivariate network exploration with JauntyNets. In *Proceedings of the 17th International Conference on Information Visualisation*, IV '13, pages 19–27, July 2013.

[91] Ilir Jusufi, Christian Klukas, Andreas Kerren, and Falk Schreiber. Guiding the interactive exploration of metabolic pathway interconnections. *Information Visualization*, 11(2):136–150, 2012.

[92] Sanjay Kairam, Nathalie Henry Riche, Steven Drucker, Roland Fernandez, and Jefrey Heer. Refinery: Visual exploration of large, heterogeneous networks through associative browsing. In *Proceedings of the 2015 Eurographics Conference on Visualization*, EuroVis '15, pages 301–310, Aire-la-Ville, Switzerland, Switzerland, May 2015. Eurographics Association.

[93] Pentti Kanerva, Jan Kristofersson, and Anders Holst. Random indexing of text samples for latent semantic analysis. In *Proceedings of CogSci*, page 103, 2000.

[94] René Keller, Claudia M. Eckert, and P. John Clarkson. Matrices or node-link diagrams: Which visual representation is better for visualising connectivity models? *Information Visualization*, 5(1):62–76, March 2006.

[95] Andreas Kerren, Achim Ebert, and Jörg Meyer, editors. *Human-Centered Visualization Environments*, volume 4417 of *LNCS Tutorial*. Springer, 2007.

[96] Andreas Kerren, Harald Köstinger, and Björn Zimmer. ViNCent – visualisation of network centralities. In *Proceedings of the International Conference on Information Visualization Theory and Applications*, IVAPP '12, pages 703–712. INSTICC, 2012.

[97] Andreas Kerren, Helen Purchase, and Matthew O. Ward. *Multivariate Network Visualization*, volume 8380 of *Lecture Notes in Computer Science*. Springer, 2014.

[98] Andreas Kerren and Falk Schreiber. Toward the role of interaction in visual analytics. In *Proceedings of the Winter Simulation Conference*, WSC '12, pages 420:1–420:13. Winter Simulation Conference, 2012.

[99] Andreas Kerren and Falk Schreiber. Network visualization for integrative bioinformatics. In Ming Chen and Ralf Hofestädt, editors, *Approaches in Integrative Bioinformatics*, pages 173–202. Springer, Heidelberg, 2014.

[100] Andreas Kerren, John T. Stasko, Jean-Daniel Fekete, and Chris North, editors. *Information Visualization, Human-Centered Issues and Perspectives*, volume 4950 of *Lecture Notes in Computer Science*. Springer, 2008.

[101] Yehuda Koren, Liran Carmel, and David Harel. Drawing huge graphs by algebraic multigrid optimization. multiscale modeling and simulation. *Simulation*, 1:645–673, 2003.

[102] Dirk Koschützki and Falk Schreiber. Comparison of centralities for biological networks. In J. Stoye R. Giegerich, editor, *In Proceedings of the. German Conference on Bioinformatics*, GCB '04, pages 199–206, 2004.

[103] Kostiantyn Kucher and Andreas Kerren. Text visualization techniques: Taxonomy, visual survey, and community insights. In *Proceedings of the 8th IEEE Pacific Visualization Symposium*, PacificVis '15, pages 117–121. IEEE, 2015.

[104] Kanehisa Laboratories. Kegg: Kyoto encyclopedia of genes and genomes. `http://www.genome.jp/kegg/`. Accessed December 2018.

[105] Robert S. Laramee and Robert Kosara. Challenges and unsolved problems. In Kerren et al. [95], pages 231–254.

[106] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Dokl. Akad. Nauk SSSR*, volume 163, pages 845–848. Soviet Physics Doklady, 1965.

[107] Alexander Lex, Christian Partl, Denis Kalkofen, Marc Streit, Samuel Gratzl, Anne Mai Wassermann, Dieter Schmalstieg, and Hanspeter Pfister. Entourage: Visualizing relationships between biological pathways using contextual subsets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2536–45, December 2013.

[108] Alexander Lex, Marc Streit, Ernst Kruijff, and Dieter Schmalstieg. Caleydo: Design and evaluation of a visual analysis framework for gene expression data in its biological context. In *Proceedings of the IEEE Pacific Visualization Symposium, PacificVis '10*, pages 57–64. Ieee, March 2010.

[109] Linneaus University and Lund University and Gavagai AB. Advances in the description and explanation of stance in discourse using visual and computational text analytics – StaViCTA funded by the swedish research council (vetenskapsrådet), grant no. 2012-5659. `http://cs.lnu.se/stavicta/`. Accessed December 2018.

[110] Shiia Liu, Yang Chen, Hao Wei, Jing Yang, Kun Zhou, and Steven M. Drucker. Exploring topical lead-lag across corpora. *IEEE Transactions on Knowledge and Data Engineering*, 27(1):115–129, January 2015.

[111] Zhicheng Liu, Shamkant B. Navathe, and John T. Stasko. Ploceus: Modeling, visualizing, and analyzing tabular data as networks. *Information Visualization*, 13(1):59–89, 2014.

[112] Gloria Mark and Alfred Kobsa. The effects of collaboration and system transparency on cive usage: An empirical study and model. *Presence: Teleoperators and Virtual Environments*, 14(1):60–80, February 2005.

[113] Kim Marriott, Peter J. Stuckey, and Michael Wybrow. Seeing around corners: Fast orthogonal connector routing. In Tim Dwyer, Helen Purchase, and Aidan Delaney, editors, *Diagrammatic Representation and Inference*, pages 31–37, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[114] Thorsten May, Martin Steiger, James Davey, and Jörn Kohlhammer. Using signposts for navigation in large graphs. *Computer Graphics Forum*, 31:985–994, 2012.

[115] Matt McKeon. Harnessing the information ecosystem with wiki-based visualization dashboards. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1081–1088, November 2009.

[116] Mike Bostock. D3 data-driven documents. `https://d3js.org`. Accessed December 2018.

[117] Mike Bostock. Les misérables co-occurrence. `https://bost.ocks.org/mike/miserables/`. Accessed December 2018.

[118] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems – Volume 2*, NIPS '13, pages 3111–3119, USA, 2013. Curran Associates Inc.

[119] Tomer Moscovich, Fanny Chevalier, Nathalie Henry, Emmanuel Pietriga, and Jean-Daniel Fekete. Topology-aware navigation in large networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 2319–2328, New York, NY, USA, 2009. ACM.

[120] Christopher Mueller, Benjamin Martin, and Andrew Lumsdaine. A comparison of vertex ordering algorithms for large graph visualization. In *Proceedings of the 6th International Asia-Pacific Symposium on Visualization*, APVIS '07, pages 141–148, February 2007.

[121] Martin Müller, Kathrin Ballweg, Tatiana von Landesberger, Seid Yimam, Uuli Fahrer, Chris Biemann, Marcel Rosenbach, Michaela Regneri, and Heiner. Ulrich. Guidance for multi-type entity graphs from text collections. In *EuroVis Workshop on Visual Analytics, EuroVA '17, Barcelona, Spain, 12-13 June 2017*, pages 1–5, 2017.

[122] Neo Technology, Inc. Neo4j. `http://neo4j.com`. Accessed December 2018.

[123] Mark E. J. Newman. *Networks: An Introduction*. Oxford University Press, 2010.

[124] Marcus Nyström and Kenneth Holmqvist. An adaptive algorithm for fixation, saccade, and glissade detection in eyetracking data. *Behavior Research Methods*, 42(1):188–204, February 2010.

[125] Christian Partl, Alexander Lex, Marc Streit, Denis Kalkofen, Karl Kashofer, and Dieter Schmalstieg. enRoute: dynamic path extraction from biological pathway maps for exploring heterogeneous experimental datasets. *BMC Bioinformatics*, 14(19):S3, November 2013.

[126] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, EMNLP '14, pages 1532–1543, 2014.

[127] Helen C. Purchase. Performance of layout algorithms: Comprehension, not computation. *Journal of Visual Languages and Computing*, 9(6):647 – 657, 1998.

[128] Eric D. Ragan, Alex Endert, Jibonananda Sanyal, and Jian Chen. Characterizing provenance in visualization and data analysis: An organizational framework of provenance types and purposes. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):31–40, January 2016.

[129] Ricardo Cabello. three.js. `http://threejs.org`. Accessed December 2018.

[130] Jonathan C. Roberts. Exploratory visualization with multiple linked views. In Jason Dykes, Alan M. MacEachren, and Menno-Jan Kraak, editors, *Exploring Geovisualization*, International Cartographic Association, pages 159–180. Elsevier, Oxford, 2005.

[131] Jonathan C. Roberts. State of the art: Coordinated & multiple views in exploratory visualization. In *Proceedings of the Fifth International Conference on Coordinated and Multiple Views in Exploratory Visualization*, CMV '07, pages 61–71, Washington, DC, USA, 2007. IEEE Computer Society.

[132] Markus Rohrschneider, Alexander Ullrich, Andreas Kerren, Peter F. Stadler, and Gerik Scheuermann. Visual network analysis of dynamic metabolic pathways. In George Bebis, Richard Boyle, Bahram Parvin, Darko Koracin, Ronald Chung, Riad Hammoud, Muhammad Hussain, Tan Kar-Han, Roger Crawfis, Daniel Thalmann, David Kao, and Lisa Avila, editors, *Advances in Visual Computing*, pages 316–327, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[133] Magnus Sahlgren, Amaru Cuba Gyllensten, Fredrik Espinoza, Ola Hamfors, Anders Holst, Jussi Karlgren, Fredrik Olsson, Per Persson, and Akshay Viswanathan. The Gavagai Living Lexicon. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC '16)*, Paris, France, May 2016. European Language Resources Association (ELRA).

[134] Magnus Sahlgren, Anders Holst, and Pentti Kanerva. Permutations as a means to encode order in word space. In *Proceedings of the 30th Annual Conference of the Cognitive Science Society*, pages 1300–1305. Cognitive Science Society, 2008.

[135] Samizdat Drafting Co. Arbor.js. `http://arborjs.org`. Accessed December 2018.

[136] Giovanni Scardoni, Michele Petterlini, and Carlo Laudanna. Analyzing biological network parameters with CentiScaPe. *Bioinformatics*, 25(21):2857–2859, 2009.

[137] Falk Schreiber, Andreas Kerren, Katy Börner, Hans Hagen, and Dirk Zeckzer. *Multivariate Network Visualization: Dagstuhl Seminar #13201, Dagstuhl Castle, Germany, May 12-17, 2013, Revised Discussions*, chapter Heterogeneous Networks on Multiple Levels, pages 175–206. Volume 8380 of *Lecture Notes in Computer Science* [97], 2014.

[138] Hans-Jörg Schulz, Marc Streit, Thorsten May, and Christian Tominski. Towards a characterization of guidance in visualization. In *Proceedings of IEEE Information Visualization, Poster Session*, 2013.

[139] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S. Baliga, Jonathan T. Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, and Trey Ideker. Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Research*, 11(13):2498–2504, 2003.

[140] Ross Shannon, Thomas Holland, and Aaron Quigley. Multivariate graph drawing using parallel coordinate visualisations. *University College Dublin, School of Computer Science and Informatics*, 2008.

[141] Zeqian Shen, Kwan-Liu Ma, and Tina Eliassi-Rad. Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1427–1439, September 2006.

[142] Zeqian Shen, Michael Ogawa, Soon Tee Teoh, and Kwan-Liu Ma. Biblioviz: A system for visualizing bibliography information. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation*, volume 60 of *APVis '06*, pages 93–102, Darlinghurst, Australia, 2006. Australian Computer Society, Inc.

[143] Zeqian Sheny and Kwan-Liu Ma. Path visualization for adjacency matrices. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization*, EUROVIS '07, pages 83–90, Aire-la-Ville, Switzerland, 2007. Eurographics Association.

[144] Ben Shneiderman and Aleks Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, September 2006.

[145] Robert Spence, editor. *Information Visualization: Design for Interaction*. Pearson/Prentice Hall, 2007.

[146] Mark Streit, Hans-Jorg Schulz, Alexander Lex, Dieter Schmalstieg, and Heidrun Schumann. Model-driven design for the visual analysis of heterogeneous data. *IEEE Transactions on Visualization and Computer Graphics*, 18(6):998–1010, June 2012.

[147] Melanie Tory and Torsten Möller. Evaluating visualizations: Do expert reviews work? *IEEE Computer Graphics and Applications*, 25(5):8–11, 2005.

[148] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[149] Peter D. Turney and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of Artificial Intelligence Research*, 37(1):141–188, January 2010.

[150] Frank van Ham. Using multilevel call matrices in large software projects. In *Proceedings of the IEEE Symposium on Information Visualization 2003*, pages 227–232, October 2003.

[151] Frank van Ham and Adam Perer. "search, show context, expand on demand": Supporting large graph exploration with degree-of-interest. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):953–960, November 2009.

[152] Frank van Ham, Hans-Jörg Schulz, and Joan M. Dimicco. Honeycomb: Visual analysis of large scale social networks. In Tom Gross, Jan Gulliksen, Paula Kotzé, Lars Oestreicher, Philippe Palanque, Raquel Oliveira Prates, and Marco Winckler, editors, *Human-Computer Interaction – INTERACT 2009*, pages 429–442, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[153] Frank van Ham, Martin Wattenberg, and Fernanda B. Viegas. Mapping text with phrase nets. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1169–1176, November 2009.

[154] Fernanda B. Viégas, Martin Wattenberg, Frank Van Ham, Jesse Kriss, and Matt Mckeon. Many eyes: A site for visualization at internet scale. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1121–1128, November 2007.

[155] Tatiana von Landesberger, Sebastian Fiebig, Sebastian Bremm, Arjan Kuijper, and Dieter W. Fellner. Interaction taxonomy for tracking of user actions in visual analytics applications. In Weidong Huang, editor, *Handbook of Human Centric Visualization*, pages 653–670. Springer New York, 2014.

[156] Tatiana von Landesberger, Arjan Kuijper, Tobias Schreck, Jörn Kohlhammer, Jarke van Wijk, Jean-Daniel Fekete, and Dieter Fellner. Visual analysis of large graphs: State-of-the-art and future research challenges. *Computer Graphics Forum*, 30(6):1719–1749, September 2011.

[157] Martin Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI'06, pages 811–819, New York, NY, USA, 2006. ACM.

[158] Seung won Hwang. Optimizing ranked retrieval over categorical attributes. In *Proceedings of the 19th IEEE Symposium on Computer-Based Medical Systems (CBMS '06)*, pages 51–56, June 2006.

[159] World Wide Web Consortium. Document object model (dom). `https://www.w3.org/DOM/`. Accessed December 2018.

[160] World Wide Web Consortium. The WebSocket API. `http://dev.w3.org/html5/websockets/`. Accessed December 2018.

[161] World Wide Web Consortium. Web Workers. `http://www.w3.org/TR/workers/`. Accessed December 2018.

[162] World Wide Web Consortium. WebRTC. `http://www.w3.org/TR/2015/WD-webrtc-20150210/`. Accessed December 2018.

[163] Michael Wybrow, Kim Marriott, and Peter J. Stuckey. Incremental connector routing. In Patrick Healy and Nikola S. Nikolov, editors, *Graph Drawing*, pages 446–457, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[164] Michael Wybrow, Kim Marriott, and Peter J. Stuckey. Orthogonal connector routing. In David Eppstein and Emden R. Gansner, editors, *Graph Drawing*, pages 219–231, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[165] Kai Xu, Chris Rooney, Peter Passmore, and Dong-Han Ham. A user study on curved edges in graph visualisation. In Philip Cox, Beryl Plimmer, and Peter Rodgers, editors, *Diagrammatic Representation and Inference*, pages 306–308, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[166] yWorks the diagramming company. yFiles. `https://www.yworks.com/products/yfiles-for-java-2.x`. Accessed December 2018.

[167] Jian Zhao, Christopher Collins, Fanny Chevalier, and Ravin Balakrishnan. Interactive exploration of implicit and explicit relations in faceted datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2080–2089, December 2013.

[168] Shengdong Zhao, Michael J. McGuffin, and Mark H. Chignell. Elastic hierarchies: combining treemaps and node-link diagrams. In *Proceedings of the IEEE Symposium on Information Visualization*, InfoVis '05, pages 57–64, October 2005.

[169] Hong Zhou, Panpan Xu, Xiaoru Yuan, and Huamin Qu. Edge bundling in information visualization. *Tsinghua Science and Technology*, 18(2):145–156, April 2013.

[170] Björn Zimmer, Ilir Jusufi, and Andreas Kerren. Analyzing multiple network centralities with ViNCent. In *Proceedings of the SIGRAD 2012 Conference on Interactive Visual Analysis of Data*, pages 87–90. Linköping University Electronic Press, 2012.

[171] Björn Zimmer and Andreas Kerren. Applying heat maps in a web-based collaborative graph visualization. In *Poster Abstracts, IEEE Information Visualization*, InfoVis '14, Paris, France, 2014.

[172] Björn Zimmer and Andreas Kerren. Displaying user behavior in the collaborative graph visualization system OnGraX. In *Revised Selected Papers of the 23rd International Symposium on Graph Drawing and Network Visualization – Volume 9411*, GD 2015, pages 247–259, New York, NY, USA, 2015. Springer-Verlag New York, Inc.

[173] Björn Zimmer and Andreas Kerren. Harnessing WebGL and WebSockets for a web-based collaborative graph exploration tool. In Philipp Cimiano, Flavius Frasincar, Geert-Jan Houben, and Daniel Schwabe, editors, *Proceedings of the 15th International Conference on Web Engineering, ICWE '15*, volume 9114 of *Lecture Notes in Computer Science*, pages 583–598. Springer, 2015.

[174] Björn Zimmer and Andreas Kerren. OnGraX: A web-based system for the collaborative visual analysis of graphs. *Journal of Graph Algorithms and Applications*, 21(1):5–27, 2017.

[175] Björn Zimmer, Magnus Sahlgren, and Andreas Kerren. Visual analysis of relationships between heterogeneous networks and texts: An application on the IEEE VIS publication dataset. *Informatics*, 4(2), 2017.

## Linnaeus University Dissertations

Below please find a list of recent publications in the series Linnaeus University Dissertations. For a full list and more information: Lnu.se

332. Thomas Nordström, 2018, *Measures that matter: facilitating literacy through targeted instruction and assistive technology,* (psychologi/ psychology) ISBN: 978-91-88898-02-9 (print), 978-91-88898-03-6 (pdf).

333. Innocent Ngao Wanyonyi, 2018, *Migrant ('dago') fishers in coastal East Africa: Understanding fisher migration and its role in artisanal fisheries,* (biologi/biology) ISBN: 978-91-88898-04-3 (print), 978-91-88898-05-0 (pdf).

334. Martin Holgersson, 2018, *Finans och Existens: Tolkning av vardagslivets finansialisering* (företagsekonomi/business administration) ISBN: 978-91-88898-12-8 (print), 978-91-88898-13-5 (pdf).

335. Suejb Memeti, 2018, *Programming and Optimization of Big-Data Applicatoins on Heterogeneous Computing Systems* (datavetenskap/computer science) ISBN: 978-91-88898-14-2 (print), 978-91-88898-15-9 (pdf).

336. Susanna Nordmark, 2018, *A Multimodal Seamless Learning Approach Supported by Mobile Digital Storytelling (mDS)* (data- och informationsvetenskap/computer and information science) ISBN: 978-91-88898-14-2 (print), 978-91-88898-15-9 (pdf).

337. Josefin Rahmqvist, 2018, *Forensic care for victims of violence and their family members in the emergency department* (hälso- och vårdvetenskap/caring science) ISBN: 978-91-88898-18-0 (print), 978-91-88898-19-7 (pdf).

338. Sofia Svensson, 2018, *Språkhandlingar i flerspråkiga elevers gruppsamtal – en studie av identitetskonstruktion.* (svenska språket/Swedish language) ISBN: 978-91-88898-20-3 (tryckt), 978-91-88898-21-0 (pdf).

339. Fredrik Ahlgren, 2018, *Reducing ships' fuel consumption and emissions by learning from data* (sjöfartsvetenskap/maritime science) ISBN: 978-91-88898-22-7 (print), 978-91-88898-23-4 (pdf)

340. Oscar Nordahl, 2018, *Intraspecific diversity of pike (Esox lucius) in the Baltic Sea and new insights on thermoregulation in fish,* (ekologi/ecology) ISBN: 978-91-88898-24-1 (print), 978-91-88898-25-8 (pdf).

341. Annelie E. K. Johansson, 2018, *Lärares bedömningsspråk; Språkhandlingar, bedömning och språklig utformning i grundskolans skriftliga omdömen,* (svenska språket/Swedish language) ISBN: 978-91-88898-26-5 (tryckt), 978-91-88898-27-2 (pdf).

342. Katarina Ståhlkrantz, 2019, *Rektors pedagogiska ledarskap – en kritisk policy-analys*, (pedagogik/pedagogy) ISBN: 978-91-88898-29-6 (tryckt), 978-91-88898-30-2 (pdf).

343. Björn Zimmer, 2019, *Guided Interaction and Collaborative Exploration in Heterogeneous Network Visualizations*, (datavetenskap/computer science) ISBN: 978-91-88898-33-3 (print), 978-91-88898-34-0 (pdf)