Ilir Jusufi

Multivariate Networks:
Visualization and Interaction Techniques

Doctoral Dissertation

Computer Science

2013

Linnæus University

PhD Dissertation Presented in Fulfillment of the Requirements for the degree of Doctor of Philosophy in Computer Science at Linnæus University, 07 June, 2013.

Multivariate Networks:
Visualization and Interaction Techniques
Ilir Jusufi

ISBN: 978-91-87427-32-9
Linnæus University
Department of Computer Science
SE-351 95 Växjö, Sweden
http://www.lnu.se/

To my wife, Cele

Abstract

As more and more data is created each day, researchers from different science domains are trying to make sense of it. A lot of this data, for example our connections to friends on different social networking websites, can be modeled as graphs, where the nodes are actors and the edges are relationships between them. Researchers analyze this data to find new forms of communication, to explore different social groups or subgroups, to detect illegal activities or to seek for different communication patterns that could help companies in their marketing campaigns. Another example are huge networks in system biology. Their visualization is crucial for the understanding of living beings. The topological structure of a network on its own could give insight into the existence or distribution of interesting actors in the network. However, this is often not enough to understand complex network systems in real-world applications. The reason for this is that all the network elements (nodes or edges) are not simple one-dimensional data. For instance in biology, experiments can be performed on biological networks. These experiments and network analysis approaches produce additional data that are often important to be analyzed with respect to the underlying network structure. Therefore, it is crucial to visualize the additional attributes of the network while preserving the network structure as much as possible. The problem is not trivial as these so-called *multivariate networks* could have a high number of attributes that are related to their nodes, edges, different groups, or clusters of nodes and/or edges.

The aim of this thesis is to contribute to the development of different visualization and interaction techniques for the visual analysis of multivariate networks. Two research goals are defined in this thesis: first, a deeper understanding of existing approaches for visualizing multivariate networks should be acquired in order to classify them into categories and to identify disadvantages or unsolved visualization challenges. The second goal is to develop visualization and interaction techniques that will overcome various issues of these approaches.

Initially, a brief survey on techniques to visualize multivariate networks is presented in this thesis. Afterwards, a small task-based user study investigating the usefulness of two main approaches for multivariate network visualization is discussed. Then, various visualization and interaction techniques for multivariate network visualization are presented. Three different software tools were implemented to demonstrate our research efforts. All features of our systems are highlighted, including a description of visualization and interaction techniques as well as disadvantages and scalability issues if present.

Keywords: Information Visualization, Multivariate Networks, Visual Analytics, Exploration, Interaction

This thesis is based on the following refereed publications:

1. Ilir Jusufi, Yang Dingjie, and Andreas Kerren. The Network Lens: Interactive Exploration of Multivariate Networks using Visual Filtering. *14th International Conference on Information Visualisation (IV '10)*, 35-42, London, UK, 2010. IEEE Computer Society Press.
   Serves as core material for Chapter 5

2. Ilir Jusufi, Andreas Kerren, and Björn Zimmer. Multivariate Network Exploration with JauntyNets. *17th International Conference on Information Visualisation (IV '13)*, London, UK, 2013. IEEE Computer Society Press, (to appear).
   Serves as core material for Chapter 6

3. Ilir Jusufi, Andreas Kerren, and Falk Schreiber. Exploring Biological Data: Mappings Between Ontology- and Cluster-based Representations. *Information Visualization*, 2013. SAGE Publications.
   Serves as core material for Chapter 7

4. Ilir Jusufi, Andreas Kerren, Vladyslav Aleksakhin, and Falk Schreiber. Visualization of Mappings between the Gene Ontology and Cluster Trees. In *Proceedings of the SPIE 2012 Conference on Visualization and Data Analysis (VDA '12)*, Burlingame, CA, USA, 2012. IS&T/SPIE. (**Best Paper Award**)
   Serves as core material for Chapter 7

5. Andreas Kerren, Ilir Jusufi, Vladyslav Aleksakhin, and Falk Schreiber. CluMaGO: Bring Gene Ontologies and Hierarchical Clusterings Together. In *Extended Abstract, BioVis 11*, Providence, RI, USA, 2011.

Related refereed publications which are not part of this thesis directly:

1. 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, 2012. Linköping University Electronic Press.

2. Andreas Kerren and Ilir Jusufi. 3d Kiviat Diagrams for the Interactive Analysis of Software Metric Trends. In *Proceedings of the 5th international symposium on Software visualization, SOFTVIS '10*, pages 203-204, New York, NY, USA, 2010. ACM.

3. Andreas Kerren and Ilir Jusufi. Novel Visual Representations for Software Metrics Using 3d and Animation. In *Jürgen Münch and Peter Liggesmeyer, editors, Software Engineering 2009-Workshopband*, pages 147-154, Lecture Notes in Informatics (LNI), Proceedings Vol. P-150, 2009. GI-Edition.

This thesis is a direct extension of the following work:

1. Ilir Jusufi. Towards the Visualization of Multivariate Biochemical Networks, *Licentiate Thesis*, Linnaeus University, 2012

Acknowledgments

I would like to express my deepest appreciation to all those who provided me the possibility to complete this dissertation. First of all, I would like to thank Prof. Dr. Andreas Kerren, for his guidance, caring, patience, and providing me with an excellent atmosphere for doing research. I could not have imagined having a better advisor for my Ph.D. studies.

My sincere thanks goes to my dear friend Dr. Mexhid Ferati for his invaluable support. I must acknowledge Prof. Dr. Falk Schreiber and Dr. Helen Purchase who's comments and insights were extremely helpful for my research. I want to thank Daniel Cernea for his support and for proofreading my thesis. Special thanks goes to my friend Björn Zimmer for all the times he was available to help me with my research and writing. A lot of appreciation goes to all the colleagues from the Computer Science department for sharing their time and experience. I am sincerely thankful to all of my friends who gave me support during all this time. I must also acknowledge my students, Vladyslav Aleksakhin and Yang Dingjie for implementing parts of our prototypes.

My deepest gratitude goes to my parents and my brothers for their love, support and encouragement throughout my entire life. They always inspired me to pursue my dreams. I want to thank my little Mona for "staying" by my side very late at night while I was writing this thesis. Finally, I want to dedicate this dissertation to my wife, for her love and help during the time of my studies. Without her support, I would have never been able to accomplish this work.

Contents

Abstract **v**

Acknowledgments **ix**

1 Introduction **1**
1.1 Challenges . . . 3
1.2 Research Problems and Goals . . . 4
1.3 Goal Criteria . . . 5
1.4 Discussion of the Research Approach . . . 5
1.5 Thesis Outline . . . 7

2 Background Information **9**
2.1 Graph Drawing . . . 9
2.2 Graph Visualization . . . 12
2.3 Biological Network Visualization . . . 16
2.4 Summary . . . 21

3 State of the Art of Multivariate Network Visualization **23**
3.1 Characteristics of Multivariate Networks . . . 23
3.2 Visualization Approaches . . . 25
3.3 Summary . . . 34

4 Usability Study of Multivariate Network Visualization Approaches **35**
4.1 Hypotheses . . . 35
4.2 Methodology . . . 36
4.3 Results . . . 38
4.4 Summary . . . 42

5 Network Visualization Using Magic Lenses **43**
5.1 The Network Lens . . . 44
5.2 Technical Aspects . . . 50
5.3 Application Scenarios . . . 51
5.4 Summary . . . 55

6 A Hybrid Approach for Network Visualization **57**
6.1 Sample Data 58
6.2 JauntyNets 58
6.3 Technical Aspects 68
6.4 Summary 69

7 Visualization of Derived Network Attributes **71**
7.1 Sample Data 71
7.2 CluMa-GO 73
7.3 Technical Aspects 85
7.4 Improvements for Balanced Cluster Trees 89
7.5 Summary 92

8 Conclusion and Future Work **93**
8.1 Discussion 95
8.2 Future Work 98

A Usability Study Network Samples **115**

List of Figures

1.1 This figure represents the coappearance network of characters in the novel "*Les Miserables*" [54, 82]. Clutter appears in many parts of the drawing, where edges cross each other or go over nodes. This network has 77 nodes and 254 edges, while most of real life networks contain thousands of them. The image was produced by the *yEd* tool for graph visualization [144]. 2

2.1 Different visual representations of the same graph. 10

2.2 This figure represents two different drawings of the same graph. Graph A shows a drawing that emphasises the criteria for minimizing edge crossings at the expense of the symmetry. Graph B represents the drawing where the symmetry criteria was emphasized which introduced edge crossings. 11

2.3 Different graph layout conventions applied on the same graph: A polyline drawing, B straight-line drawing, and C orthogonal drawing. 11

2.4 Hierarchical edge bundles. The β parameter is used to specify the strength of the bundling. The picture was taken from [51] with permission of the authors. ©2006 IEEE. 13

2.5 Applying the EdgeLens filters out all the edges that are not connected to the nodes under the lens and making it easier to view the connections to the focused nodes. The picture was taken from [128] with permission of the authors. ©2006 IEEE. 15

2.6 The diagram shows an example of a KEGG reference pathway. The image represents the *Streptomycin biosynthesis – Reference pathway* [1]. 18

2.7 *Blue* nodes represent GO terms forming a DAG that describe the roles and properties of gene or gene products denoted with *red* nodes. 19

2.8 Gene or gene products are placed at the bottom (*red* nodes). Multivariate experimental data are represented by a heat-map. The hierarchical clustering is computed based on this data and shown as dendrogram on top. 20

3.1 Samples of multivariate data visualization techniques. The image marked with label A is made with a tool presented by Kerren *et al.* [76, 77] based on the original work of Pinzger *et al.* [96] 24

3.2 Classification of approaches. The left blue circle (A) represents the approaches that are able to represent the underlying network topology, in contrast to the right blue circle (B) that represents approaches focussing on the attribute values of the network. 28

3.3 Multiple Coordinated Views approach. The left side of the image represents a visualization of multivariate network data by parallel coordinates, while the network is shown in a separate view on the right. The picture was taken from [114] with permission of the authors. 29

3.4 An example of an Integrated Approach shows experimental data embedded into a biological network. The picture was taken from [13] showing the relative levels of different "Vicia narbonensis" lines integrated into the glycosis and citric acid cycle. Image courtesy of IPK Gatersleben. 31

3.5 An hourly ring of calls shows the frequency of calls every hour. The picture was taken from [115] with permission of the authors. ©2008 IEEE. 33

3.6 A screenshot of the ViNCent tool [146]. 34

4.1 Showing a sparse graph named SP3 visualized by using an integrated approach. It is the only tree among the sparse graphs. The node with label 10 is the correct answer for the first question, while node with label 14 is the correct answer for the second question. . . 38

4.2 Showing a dense graph named DE1 visualized by using a multiple coordinated views approach. 39

4.3 Subjects demographic data. 39

4.4 Subjects education data. 40

4.5 Knowledge about graph/network visualization. 40

4.6 Median times (in milliseconds) participants spent on performing the tasks one and two for both types of sparse (SP) and dense (DE) graphs using integrated approaches (IA) and multiple coordinated views (MCV). 41

4.7 Average participants grading of the readability of the topology for both types of sparse (SP) and dense (DE) graphs using integrated approaches (IA) and multiple coordinated views (MCV). 41

5.1 Overview of the Network Lens tool (rotated by 90°). The GUI is divided into three distinctive parts: the main network visualization area, the lens information area on the right hand side, which we call *Lens Mapping*, and the bottom part where all user-produced lenses are preserved. The multivariate network data is based on students (⇒ nodes) who share the same courses (⇒ edges) and their individual course grades and personal information (⇒ attributes). © 2010 IEEE. 46

5.2 A dialog box used to create and edit lenses. Users can specify the type of the lens, select different attributes provided by the input data set and assign their color mapping. © 2010 IEEE. 47

5.3 Two different visual representations used to display the attributes. Attributes can be color coded automatically, or the color can be specified by the user. Nominal attributes, such as *Name*, are represented by text labels on the right hand side of the glyphs. © 2010 IEEE. 48

5.4 A dialog box used to combine different lenses. © 2010 IEEE. . . . 49

5.5 Visualization of experimental data integrated into a biological network. Based on the data shown in Figure 3.4. 52

5.6 Visualization of experimental data by using the Network Lens. Based on the data shown in Figure 3.4. 53

5.7 The transparent gray circular disk in the network visualization view (a) represents the focus of the lenses discussed in the following. The top right image (b) represents the view of the lens named "Network", while the bottom right image (c) shows the view rendered by the "Network Visualization" lens. © 2010 IEEE. 54

6.1 This screenshot shows the initial visualization after the data D1 has been loaded. The *orange* blocks placed on a circle represent the attributes of the network. *Green* nodes in the middle represent the text documents. 59

6.2 The graph view on the left shows three attribute groups (based on data set D1). The green node has been highlighted after a mouse-over action, and a tooltip displays the node label. The first-level neighbors of the selected node are shown with an orange halo. Attribute nodes act as bar charts for the highlighted node, giving insight into the values it has for each attribute. The faded attribute glyphs were disabled by the user. © 2013 IEEE. 63

6.3 The screenshot displays papers from InfoVis and VAST conferences (data set D3). Two cliques marked with A and B are interesting as they are related to the created attribute groups. © 2013 IEEE. . . . 64

6.4 A cut-out of the data represented in Figure 6.3. It shows that clique B is still highly connected even after filtering out all edges with weight 1. © 2013 IEEE. 65
6.5 The screenshot displays a network of 421 nodes with 55 attributes (data set D2). The nodes are colored differently, because they have been clustered based on their attribute values. © 2013 IEEE. 67
6.6 MDS view (data set D2). The highlighted node shows a tooltip with the node label "yi.pdf", and the first-level neighbors are displayed with an orange halo. The node colors reflect the results of a previous clustering step. © 2013 IEEE. 68

7.1 The *light-blue* part on the left represents a part of the GO DAG. The *grey* part on the right represents the Cluster Tree, while the *red nodes* in the middle are shared between both of them. Note that this diagram shows an idealized situation, because the common leaves do not need to be neighbored. 72
7.2 GUI of CluMa-GO (rotated by 90°). On the left hand side, the used Gene Ontology is represented in the GO view (Levels Layout). On the right hand side, the Cluster Tree view is located. 75
7.3 Zoomed-in view using the Levels Layout approach. The red nodes represent leaf nodes (e.g., genes); the light-blue nodes represent non-terminal nodes (e.g., terms). This view provides insight into the distribution of leaf nodes in a specific DAG level. The orange nodes represent the calculated subgraph (mapping). 76
7.4 Two conceptual diagrams that show the fundamental ideas of both layering approaches. 77
7.5 GO view with visible (bundled) edges based on the Bottom Layout. The green circle in layer 3 highlights the selected GO term. 78
7.6 Sample cluster tree t. Yellow color represents the calculated backbone. 79
7.7 Spiral Tree Layout of t. The drawing algorithm was inspired by standard spiral layouts that are mostly used to represent time-series, such as [2, 129]. 80
7.8 The *red nodes* in the middle are shared between both the GO DAG (*light-blue nodes*) and the Cluster Tree (*gray nodes*) (cp. Figure 7.1). The interactively selected node is highlighted in *green*, from which we traverse the graph (*orange nodes*) until we reach all accessible leaves (*red nodes with orange background*). The leaves are used to calculate a subtree of the Cluster Tree (*orange nodes* in the right part of the figure). 81
7.9 This screenshot shows the zoomed-in GO view (with the three layers 7 - 9) on the left hand side and the Cluster Tree view with opened subtree widget on the right hand side. 82
7.10 Cut-out of a mapping in the Cluster Tree view. 83

7.11 Subtree (branch) view. The more detailed view of the selected branch (*green box glyph*) is visualized as a dendrogram. 83
7.12 Subtree (branch) view. The more detailed view of the selected branch (*green box glyph*) is visualized by following a so-called HV-drawing algorithm. 84
7.13 This screenshot shows the Detailed Mapping view. On the left hand side, the selected subgraph of the GO DAG is represented; the Cluster Tree is shown on the right hand side using a dendrogram layout. Both are connected with genes: the red nodes in the center. Some edges are thicker and blue. As seen in the cut-out of the screenshot, they represent a lot of backbone nodes which are hidden in order to avoid clutter. 86
7.14 Module architecture of CluMa-GO. 87
7.15 In this cut-out of the cluster tree, we assume that the subtrees highlighted in *green* and *blue* are too large to be abstracted into boxes. . 90
7.16 Nested Spiral Tree Layout built based on the tree sample in Figure 7.15. The red circles show the roots of the main tree (the circle in the center) and of the branches (i. e., the nested subtrees). 90
7.17 A new layout method based on recursive patterns in order to visualize more balanced cluster trees. Here, root nodes are always located in the top-left corner of each container. 91

A.1 Showing a sparse graph named SP1 visualized by using a multiple coordinated views approach. 115
A.2 Showing a sparse graph named SP1 visualized by using an integrated approach. 116
A.3 Showing a sparse graph named SP2 visualized by using a multiple coordinated views approach. 116
A.4 Showing a sparse graph named SP2 visualized by using an integrated approach. 117
A.5 Showing a sparse graph named SP3 visualized by using a multiple coordinated views approach (cp. Figure 4.1 for its corresponding version using IA). 117
A.6 Showing a sparse graph named SP4 visualized by using a multiple coordinated views approach. 118
A.7 Showing a sparse graph named SP4 visualized by using an integrated approach. 118
A.8 Showing a dense graph named DE1 visualized by using an integrated approach (cp. Figure 4.1 for its corresponding version using MCV). 119
A.9 Showing a dense graph named DE2 visualized by using a multiple coordinated views approach. 119

A.10 Showing a dense graph named DE2 visualized by using an integrated approach. 120
A.11 Showing a dense graph named DE3 visualized by using a multiple coordinated views approach. 120
A.12 Showing a dense graph named DE3 visualized by using an integrated approach. 121
A.13 Showing a dense graph named DE4 visualized by using a multiple coordinated views approach. 121
A.14 Showing a dense graph named DE4 visualized by using an integrated approach. 122

List of Tables

3.1 Multivariate Approaches. The table shows information about different approaches for visualizing multivariate networks. The *Domain* is shown through its own column. There are three sub-columns under the *Attributes* column denoting what kind of attributes are visualized by the corresponding tool with respect to the network elements. The *Approaches* column specifies to what kind of approach discussed in this chapter the tool belongs to. The sub-column *Integrated* is divided into three more columns stating to which part of the network the attribute visualization has been embedded to. 27

4.1 Four different pipelines a participant can be assigned to. The first subject is assigned to the first pipeline, the second to the second and so on. The process is repeated for all subsequent subjects. 38

Chapter 1

Introduction

Exploring, analyzing and understanding complex and large data sets is becoming an ever increasing challenge. A lot of them describe relationships between objects. Finding ways to discern between these relationships in context of other additional related data is very important for a number of different science domains. Visualization is often used to give insight into different patterns between relations of complex and large data objects and/or their individual features. Mostly, these relational data sets are represented as node and link diagrams (graphs or networks), where nodes represent the particular elements, and edges show different relations or interlinks between these elements.

Visual analysis tools have to cope with different challenges, such as to increase people's understanding of the underlying data or to avoid clutter, and face other issues related to huge, complicated and dynamic data. Sometimes, nodes and edges may overlap even when dealing with smaller data sets, for instance in cases when the graph is highly interconnected or the display resolution is limited. This will make the interpretation of the graph almost impossible, thus showing that some of these issues can be present not only with large data sets (cp. Figure 1.1). Various graph drawing algorithms aim at improving the readability of these data (graphs/networks) based on predefined aesthetic criteria (cp. the next chapter for more information).

Most of the graph drawing algorithms do not take into consideration that each network object may additionally have a number of attributes that are important to be visualized in context of the overall network presentation. Researchers from different domains, such as biology, social network analysis or software engineering face the challenge of visualizing networks together with additional data. For instance, if one wants to analyze communication patterns among workers in an organization one can analyze the email correspondence within the organization members. Notice that each employee has its salary, position, age, gender, time employed in company, address, working hours, etc. All these characteristics could be important and be regarded as additional network attributes. In order to actually discover if there are different interesting patterns in the relationships, we would need to visualize each staff's individual set of attributes, and see if some pattern related to attributes emerge in context of a communication network. For instance, do members of the same gender or age communicate more with each other? Or do groups of people belonging to the same and/or different job position exchange emails more frequently?

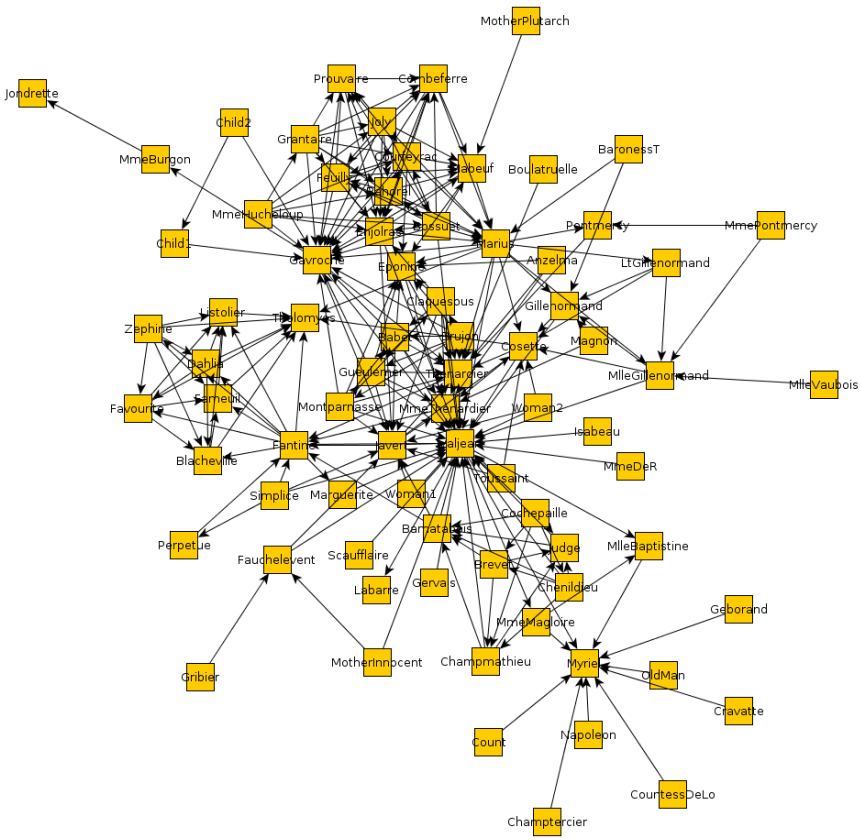


Figure 1.1: This figure represents the coappearance network of characters in the novel "*Les Miserables*" [54, 82]. Clutter appears in many parts of the drawing, where edges cross each other or go over nodes. This network has 77 nodes and 254 edges, while most of real life networks contain thousands of them. The image was produced by the *yEd* tool for graph visualization [144].

Showing the network topology together with these additional data will help us to answer these questions. The term *multivariate networks* is generally used to describe such networks, i.e., networks that have a list of different attributes attached to their vertices or edges. Section 2.1 discusses these notions in more details.

However, visualizing multivariate data on its own is often challenging enough. This data can be of different type (categorical, numerical, ordinal, etc.), thus different techniques and strategies are required in order to visualize them. The number of attributes may be too big as well making it even more challenging to show them alongside large and complex networks. Different automatic analysis approaches might even produce metadata of the data that need to be shown in context of the underlying network. Such data could be created from some sort of clustering algorithm that might produce non-conventional attribute data types such as cluster trees.

The next section continues with the motivation of this work. It explains the main problems that are addressed by this thesis.

1.1 Challenges

As a society in general, we are creating more and more data each day and researchers from different science domains produce and analyze tremendous amounts of network data. A lot of this data, such as our friendships or connections on different social networking websites, can be modelled as graphs, where nodes are actors and edges are relationships between them. Researchers analyze this data to find new forms of communication, to explore different social groups or subgroups, to detect illegal activities or to seek for different communication patterns that could help companies in their marketing campaigns. The topological structure of the network could shed light into understanding who are interesting actors in the network. For instance, different celebrities might have a lot of connections to their fans. Such actors become easily distinguishable once the network is visualized.

Visualizing networks is also important in the life sciences. Due to advances in technology, biological network data production is overwhelming. Databases such as gene regulatory networks or metabolic pathway networks are updated on a daily basis by adding or editing hundreds of elements (Section 2.3). Therefore, online databases were created to help researchers to cope with this data [68, 38]. Networks play a huge role in the understanding of living beings which, for example, could lead to the discovery of important information regarding different pharmaceutical projects. However, the complexity and the sheer amount of data being produced is hindering the interpretation and visualization of these networks.

Furthermore, these thousands of nodes are not simple one-dimensional data. Each one of the elements may contain many specific attributes. For instance, if a document network is being visualized with documents as nodes and co-authorship as edges, each node might have a number of different attributes attached to it. It might have a list of frequently used keywords in the text with the number of the co-

occurrences for each keyword. It might also hold different metadata such as creation time, the size of the file, number of words, number of pages, etc. Several computational approaches for network analyzes could be employed, such as calculating network centrality values. In biology, different experiments are usually performed on the obtained data (biological networks in this case). These experiments and network analysis approaches produce additional data that are often important to be analyzed with respect to the underlying network structure. Therefore, it is crucial to visualize the additional attributes of the network while preserving the network structure as much as possible. The problem is not trivial as these so-called multivariate networks could have a high number of attributes that are related to their nodes, edges, different groups, or clusters of nodes and/or edges. Even more issues arise if we consider the previously mentioned challenges as the visualization space becomes more expensive due to the complexity and amount of attribute and network data.

As explained above, different computational approaches for network analyzes could produce more data. Sometimes this data can be rather complex and present an additional challenge during the visualization. For instance, analyzing biological networks sometimes involves hierarchical clustering of various experimental data that produces huge data sets of tree-like structures. Usually this clustering is performed on all or a subset of network elements (nodes). This subset of elements is then "mapped" on a resulting cluster tree, as clustering in an informal sense means placing different objects into different groups (clusters) based on some predefined criteria. Therefore, we might say that these trees and networks share a considerable amount of elements, i.e., they are "connected" with each other. Often, both views are desired: the visualization of the network data together with the clustering data. This fact implies the need to show two huge and complex graphs that are of different nature, and give insight into their relationships. However, visualizing a graph of ten thousands of nodes is a challenge on its own, and in this concrete case researchers are faced with two huge network visualizations. These or similar challenges are also prominent in various science domains that deal with network analyzes. As such, solutions introduced in one of the domains could be adapted and reused for solving problems in other domains of science.

1.2 Research Problems and Goals

The aim of this thesis is to contribute to the development of different visualization and interaction techniques for the visualization of multivariate networks. Initially, a deeper understanding of existing approaches for visualizing multivariate networks should be acquired in order to classify them into categories and to identify disadvantages or unsolved challenges. Next, new visualization and interaction techniques that will overcome various issues of these approaches should be developed. With this point in mind, we define two main research goals:

1. Offer a contribution regarding the categorization of the existing approaches and perform a study regarding the usability of the main approaches.

2. Offer a contribution for the visualization challenges presented above with respect to different categories of approaches for the visualization of multivariate networks and offer a contribution for the visualization of complex data derived from multivariate networks by computational methods.

1.3 Goal Criteria

In this section, we define a number of criteria for fulfilling our research goals starting with the criteria for the first goal:

1.1 Provide a survey on the current state of the art on the visualization of multivariate networks in general. As there are several different approaches for visualizing this type of data, there is a need to categorize them and investigate strengths and weaknesses of different approach categories. Therefore, an important task here is to define the criteria and the categories for the classification of multivariate approaches.

1.2 Convey a usability study regarding the main approaches for the visualization of multivariate networks.

The criteria for fulfilling our second goal are:

2.1 Different visualization approaches introduce different advantages, but also various disadvantages and challenges. Therefore, introducing novel interaction and visualization techniques for multivariate networks that tackle these challenges is one of the criteria for fulfilling the second goal.

2.2 As explained in Section 1.1, different computational processes may produce additional complex data derived from multivariate data. For instance in biology, different experimental procedures generate multivariate data that are later hierarchically clustered. The clustering usually produces trees which can be considered as additional complex data. Therefore, a novel approach for visualizing such derived cluster data for multivariate networks shall be introduced.

1.4 Discussion of the Research Approach

Here, we briefly describe the approach to fulfill the aforementioned criteria, which lead to achieving our presented goals and aims. Our first goal is more focused on a conceptual understanding of the visualization problems related to multivariate networks. Working on this goal helps in understanding the overall complexity of the

data and provides a good introduction into the different challenges presented by different approaches for visualizing multivariate networks. The second goal focuses mainly on the development of the techniques for visualization of multivariate networks.

For the first criterion, a state of the art survey about multivariate networks should be performed. The aim of this survey is to provide a good understanding about the underlying problems and to classify different approaches into a set of categories, based on a number of specific features used in various approaches. This survey serves as a reference point for fulfilling our remaining criteria.

For Criterion 1.2, a small study was performed to compare the usability of the two main approaches typically used in the visualization of multivariate networks. The participants of the study were asked to finish a set of tasks regarding multivariate data and network analysis. The tasks were timed in order to understand which approach is more effective. With this, contributions towards achieving the first goal should be achieved.

The following criteria tackles the second goal, namely the contribution towards overcoming some of the challenges of multivariate network visualization approaches. For Criterion 2.1, we present improvements of existing techniques. The main idea of one of the approaches is to embed glyphs into graph nodes to visualize node attributes. For instance, a biologist may need to visualize different data resulting from various experiments. If each element in a biological network has specific results, one could embed a visual representation of these details into the nodes of the network. This approach leads to some issues, such as the use of space as the nodes get bigger in size. These issues are explained later in more detail. We demonstrate a way to overcome this problem through the use of interaction techniques. Additionally, we present a novel interactive way to create and use different filters for attributes through our prototype implementation. Another approach for analyzing multivariate data together with the underlying network is to use certain computational methods to specify the node positioning based on the values of attributes. This strategy often creates unreadable graph layouts, as the node placement does not follow any of the usual graph drawing aesthetic criteria. A new approach that allows users to balance the influence of the attribute values and the traditional graph layout algorithm is introduced in this thesis.

For the final criterion, a prototype tool that deals with complex multivariate data is presented. In our case, attribute values of a big biological network have been hierarchically clustered. The product of such a clustering is a relatively large binary tree. It is important for biologists to visualize the underlying network and the cluster tree together. Therefore, we implemented a prototype that is able to visualize and handle such large data sets in real-time.

1.5 Thesis Outline

The motivation behind this work and goals of the thesis, together with criteria and methodology to achieve these goals has been described so far. The rest of this thesis is organized as follows: in Chapter 2, the related work is briefly discussed, beginning with the issues of graph drawing and visualization and continuing with the description of particular biological data sets that are important for a part of this thesis. A brief state of the art survey on multivariate network visualization is presented in the next chapter where different approaches are categorized in different groups based on predefined criteria. In Chapter 4, a task based usability study comparing two groups of approaches is described. The following three chapters highlight different tools that were developed to address the problems mentioned in the previous section. For approaches that integrate multivariate attributes inside network elements interaction-based improvements are presented in Chapter 5. A new visualization and interaction technique that relies on an extended force-based layout algorithm for visual analysis of multivariate networks is presented in the next chapter. A tool that deals with hierarchical clustering data obtained from high throughput biological experiments performed on gene ontology networks is described in Chapter 7. Finally, the presented work is summarized, discussed and future work is highlighted in the last chapter of this thesis.

Chapter 2

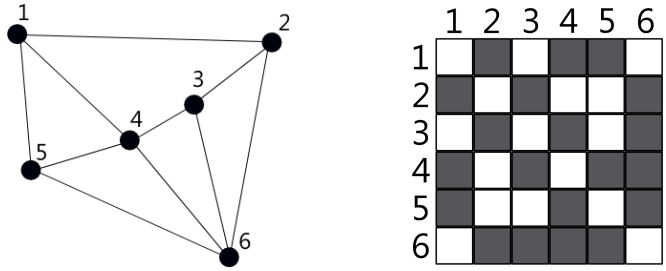
Background Information

In this chapter, related work in context of graph and network visualization is briefly discussed. Initially, general ideas in the field of graph drawing are presented in Section 2.1. Next, common approaches and techniques for network visualization in context of information visualization (Section 2.2) are discussed. Finally, the main issues of the visualization of biological networks are discussed in context of the challenges related to ontologies and clustering. These important concepts in life sciences are presented in detail as one of our tools described in this thesis deals with such data.

2.1 Graph Drawing

Before discussing the particularities of the graph drawing discipline, it is worth noting that *graphs* and *networks* are considered as different notions in this thesis. A (simple) *graph* $G = (V,E)$ consists of a finite set of vertices (or nodes) V and a set of edges $E \subseteq \{(u,v)|u,v \in V, u \neq v\}$. In contrast, a multivariate network N consists of an underlying graph G plus n additional attributes $A = \{A_1,\ldots,A_n\}$ that are attached to the nodes and/or edges. For node attributes, A_i represents a column in a table of attributes $A = (a_{ji})\,(j = 1\ldots|V|\,; i = 1\ldots n)$ and contains one attribute value per node (similar definition for edges). Thus, $a^u = (a_{u1},\ldots,a_{un})$ describes all attribute values for node u given that there is no missing data.

The development of computer technology made it possible to automatically draw graphs, which lead to the advancement of the graph drawing discipline. Various graph drawing algorithms compute a layout of the nodes and the edges, mainly based on so-called node-link diagrams (Figure 2.1(a)), while other graph representation metaphors may be used, such as matrix-based layouts (Figure 2.1(b)) [137] or space filling layouts for trees which are a special case of graphs. Space filling representations for special types of graphs are also possible [66]. We focus here on the node-link metaphor as it is more popular than matrix based layouts. A combination of these approaches is possible as well [48]. Various layout algorithms have been developed and they play a fundamental role in network visualization. Particular graph layout algorithms can give an insight into the topological structure of a network if properly chosen and implemented, or otherwise, it may conceal the nature of the underlying structure [17]. Therefore, it is important to maximize the



(a) Node-link representation of a graph

(b) Matrix-based representation of a graph

Figure 2.1: Different visual representations of the same graph.

readability of the graph drawing so that no information is hidden or lost.

The graph readability is affected by quantitative measurements called *aesthetic criteria*. Thus, graph drawing generally deals with ways of drawing graphs according to a set of predefined aesthetic criteria [75, 23, 17]. Some of the most important criteria are:

- Minimize number of edge crossings (edge crossings make it harder to follow the edges)
- Minimize number of edge bends (an edge with a large number of bends is harder to follow)
- Minimize edge length (long edges are harder to follow)
- Minimize node overlap (overlapped nodes are harder to distinguish and may hide information)
- Minimize drawing area (smaller drawing make it possible to use the free space to display other important information)
- Maximize symmetry (symmetry is useful for creating mental maps)

The graph readability is not always affected in a positive manner by enforcing these criteria. Furthermore, some criteria might work against each other. For instance, maximizing symmetry might introduce more edge crossings (Figure 2.2).

Layout conventions and *layout methods* also affect the readability. A layout convention is a basic rule followed by the drawing of graphs, such as *polyline drawing*, *straight-line drawing*, *orthogonal drawing,* etc (cp. Figure 2.3) [23]. Specific graph drawing methods are employed with respect to the desired criteria and to the particular type of graph. For instance, if we want to draw *trees*, many layout algorithms are available [81, 103, 29, 43]. *Hierarchical layouts* are a specific class of layout algorithms that place the nodes on horizontal layers according to their distance

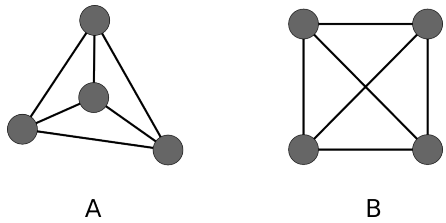


Figure 2.2: This figure represents two different drawings of the same graph. Graph A shows a drawing that emphasises the criteria for minimizing edge crossings at the expense of the symmetry. Graph B represents the drawing where the symmetry criteria was emphasized which introduced edge crossings.

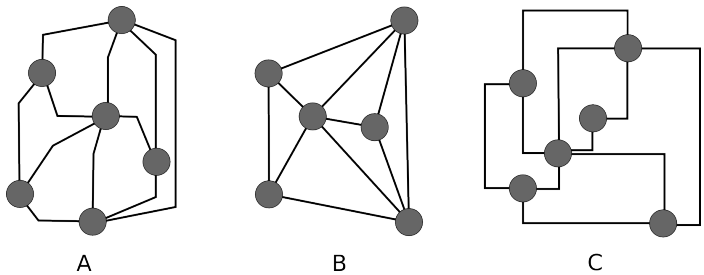


Figure 2.3: Different graph layout conventions applied on the same graph: A polyline drawing, B straight-line drawing, and C orthogonal drawing.

from the root node. Many algorithms which produce polyline drawings are designed specifically for *directed graphs*, such as *layered drawing*s.

One interesting class of layering algorithms mimics natural physical forces. These so-called *force-directed* algorithms are relatively simple to code and to understand due to their physical analogy which makes them pretty desirable for *undirected graphs* [75]. An extension of such algorithms is crucial to one of the ideas that is presented in this thesis. Therefore, a simple version of force-directed algorithm using spring and electrical repulsion forces is introduced in the following. Here, nodes can be considered as charged particles that repel each other, and edges are modeled as springs. For the x-component of the force on a node v the following holds (y-component analogous):

$$\sum_{(u,v)\in E} (sti_{uv}(d_{uv} - l_{uv}))\hat{x}_{uv} + \sum_{(u,v)\in V\times V} \frac{rep_{uv}}{d_{uv}^2}\hat{x}_{uv} \tag{2.1}$$

In this formula, the first sum represents the spring force between two nodes u and v connected with an edge and the second sum the repulsion force between v and other nodes. The Euclidean distance between u and v is denoted by d_{uv}, l_{uv} is the zero-

energy length of the spring between u and v (i.e., no force if $d_{uv} = l_{uv}$), $sti_{uv} \in [0,1]$ is the stiffness of the spring between u and v (i.e., the larger this parameter the more the tendency for d_{uv} to be close to l_{uv}), and finally rep_{uv} is the strength of the electrical repulsion between the two nodes. Note, that $\hat{x}_{uv}$ denotes the unit vector of $(x_v - x_u)$.

When implementing any kind of network visualization environment, choosing and/or designing a graph layout algorithm is one of the most important steps. In most cases, a sufficient layout algorithm would represent the underlying graph topology and reduce the scalability problem that is one of the ongoing challenges of the information visualization community in general [88].

The task of implementing a good graph drawing algorithm is often complex and time consuming. Therefore, a number of different open source libraries are available (for example, JUNG [94] and Prefuse [47] among others) that deal with graph layout issues. In most cases when dealing with smaller networks, such resources are sufficient.

2.2 Graph Visualization

In this section, problems of graph visualization from an information visualization perspective will be introduced. They differ in many aspects from those of the traditional graph drawing community. The work of Herman *et al.* [49] presents a nice overview on this subject. The use of interaction techniques to overcome problems such as clutter in case of large and/or complex network data is what usually differentiates graph visualization from traditional graph drawing. However sometimes, it is really hard to classify the approaches, especially when less or no interaction is involved as described in the following.

Visualization Techniques

There are techniques that do not require interaction, but do not fall into traditional graph drawing approaches. For instance, there exist several approaches that try to manage the edge drawing in order to avoid any overlaps or clutter. *Edge routing* is one such technique. It is used to avoid edges from overlapping nodes [26] besides the more traditional edge crossing removal algorithms which are usually NP-hard [36]. Another approach bundles the edges together in a tree layout. It reduces edge clutter while giving insight into the relations of the nodes in context of a hierarchical data set [51]. One drawback of edge bundles is that they "hide" explicit node to node links. Therefore it is useful to be able to interactively change the bundling parameter (cp. Figure 2.4). The edge bundling algorithm has been adapted to work on general graphs using force-directed layout algorithms [52]. Other improvements and adaptations of edge bundling approaches have been developed. Ersoy *et al.* [32]

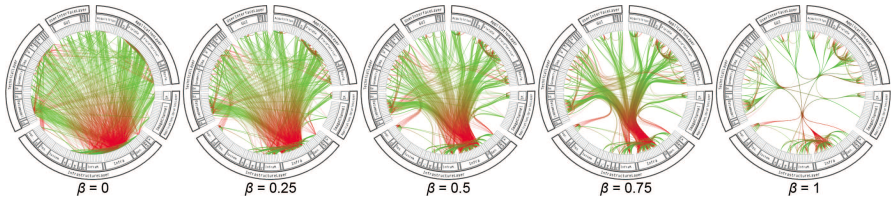


Figure 2.4: Hierarchical edge bundles. The β parameter is used to specify the strength of the bundling. The picture was taken from [51] with permission of the authors. ©2006 IEEE.

create edge bundles for general graphs, while Selassie *et al.* [111] present an approach to handle edge directions and weights.

Clustering the nodes is another technique to deal with huge data sets. It can be useful for data reduction by treating a cluster as a single representative unit, but more importantly clustering is used for finding *natural* modules and describing their properties, classifying data, and detecting outliers. There are two main clustering approaches when dealing with graphs: the first one, is about clustering the nodes based on their domain specific content. Usually, the nodes are clustered according to the distance values calculated based on the node attributes. This approach will be discussed later in this thesis in more detail. The graph structure itself is the foundation for the second approach. For instance, graph components that are strongly inter-linked in contrast to other elements of the graph are clustered together.

Optimizing the use of display space is an important task in information visualization. There are approaches that focus exactly on this aspect. Some tools use the hyperbolic geometry in order to use display space more effectively [87, 93, 92]. The main idea is to firstly run a graph layout algorithm on the hyperbolic space and then to map the results to the Euclidean space. In contrast to Euclidean space, the circumference of the circle in the hyperbolic plane increases exponentially with its radius. This means that hierarchies could be laid out in the hyperbolic space, as they tend to expand exponentially with their depth. Although the distance between parents, children and siblings is roughly the same when measured in hyperbolic geometry, it will not appear so when mapping it to Euclidean space. Lamping *et al.* [87] map the hyperbolic plane to a radial display region. The objects in the center of the view appear exponentially bigger compared to the objects around them. This introduces a need to use interaction techniques and animation, so that the users can browse the data. Each time a user clicks on a specific node, that node moves to the center of the screen, while others move away based on their topological position.

Interactive 3D graphics may present another mean to use the display space effectively. The most obvious advantage here is that a new dimension is gained that could be used to map and effectively visualize more data. Living in a 3D world gives us another advantage as we are used to this kind of environments. There is a

considerable number of tools that use the 3D approach to visualize networks especially developed in 1990's [106, 141, 104]. After this period, 3D visualization lost a bit of popularity due to some of the issues this approach presents. One such issue is that we can only project 3D scenes to a 2D display. This requires additional interaction (moving and rotating the object) to perceive the objects correctly. Navigation in 3D space is more difficult as most input devices are made with 2D in mind. Ware *et al.* [136] present more in depth analyzes of the 3D visualization problems and suggest that 2D, or 2.5D solutions are better.

The widely used node-link approaches have a conceptual drawback in terms of use of display space, as in most cases there are a lot of empty, unused areas between nodes. Alternative graph representation techniques, such as matrix-based techniques might use the space more effectively. Several other techniques have been developed to deal with this shortcoming, like Treemaps [59] or Sunburst [121]. However, we will not explain these space-filling approaches any further, as we focus on node-link visualizations in this thesis.

Interaction Techniques

The crucial role of the interaction in information visualization has been covered by the work of Yi *et al.* [143]. Here, some of the interaction techniques used in context of graph visualization are discussed shortly.

Dynamic queries are one of the most common and powerful interaction techniques in information visualization. Techniques such as range sliders could help a lot in reducing the visual clutter by filtering out unimportant data [119]. Most of these techniques could be applied directly to graph visualization. For instance, a slider could be used to filter out the nodes with low incoming edges.

Zooming and panning is another common interaction technique that deals with huge data. Often it is impossible to show the complete data in a view at a specific zoom level. In such cases users must pan, i.e., move the viewing frame. Zooming out enables to view the complete data, while zooming in will uncover more details about the data. This is a standard interaction technique used in most of the network visualization systems. However, zooming in suffers from one important conceptual drawback. When zoomed in, the overview of the area outside the zoom frame is lost. Therefore, several techniques are created to enable users to focus on a specific part of the data set while showing the context of the whole or larger chunk of data (*focus+context*). One of the most prominent focus+context techniques are distortion functions that produce an effect similar to that of a fish-eye lens [119, 8, 109].

A number of different approaches based on the fish-eye lenses have been introduced. The magic lens developed by Bier *et al.* [11, 123] is the most closely related work compared to our lens implementation discussed in Chapter 5. It is described as a user interface tool that changes the view of the object beneath it by combining its view area with an operator. The interactions of the tool have been described as to a real magnifying lens over a newspaper. Beside serving as a simple magnifica-

tion tool, their magic lens facilitates visual filtering of the object viewed through it. Usually, these magic lenses perform some image processing computations on the graphical objects or filter out some kind of object. Another interesting feature of this approach is the possibility to combine different lenses, thus producing a new lens that acquires all the features of all the lenses being combined. Of course this is not easy, especially when dealing with semantics and not just filtering and using distortion functions on graphics objects. Applying "combined" distortion on graphical objects might appear natural and intuitive to the user. However, applying such distortion on abstract data representations might result in a wrong information being presented to the user. Thiede *et al.* [126] present a model to overcome this issue.

Originally, the magic lens approach was not applied in terms of graphs. One example of using lenses in context of graphs, albeit with a slightly different aim in mind, is EdgeLens [139]. This interactive tool is used to manage the edge congestion in graphs, i.e., it is applied to graphs with high edge density in order to improve the visual perception locally. When applied, this lens "pushes" the edges around the focal point, making it easier to read focused node labels, for example. A similar approach is shown in Figure 2.5 [128]. Additionally, various lenses in context of graphs are presented by Tominski *et al.* [127].

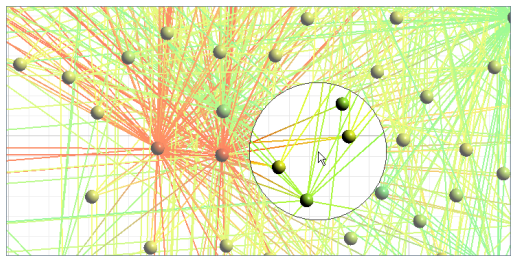


Figure 2.5: Applying the EdgeLens filters out all the edges that are not connected to the nodes under the lens and making it easier to view the connections to the focused nodes. The picture was taken from [128] with permission of the authors. ©2006 IEEE.

Magic lens-based techniques could be usable if they are combined with a number of other visualization techniques as suggested by a couple of studies. The results from the work of Baudish *et al.* [8] suggest a significant time saving in their experimental tasks and a higher subjective satisfaction when using magic lens based techniques. The authors did a study of the usability of magic lens based techniques by applying the metaphor for their focus+context interaction interface and compared it with overview+detail and pan+zoom interfaces. Another small comparative study based on three types of fish-eye view interfaces in context of graph layout tasks was performed by Gutwin *et al.* [44]. Even though there are differences between competing fish-eye varieties: the study suggest that lenses in general could be regarded as an advisable tool for network visualization environments.

2.3 Biological Network Visualization

Types of Biological Networks

In this part, different types of biological networks are briefly discussed. According to Junker *et al.* [61], there are six types of biological networks in general: *signal transduction and gene regulatory networks*, *protein interaction networks*, *metabolic networks*, *phylogenetic networks*, *ecological networks* and *correlation networks*. Zhu *et al.* [145] present a slightly different classification of biological networks types, but we will bear on the work of Junker *et al.* [61].

Signal transduction and gene regulatory networks are crucial actors in the evolution and existence of organisms. The evolution of gene regulatory networks is responsible for making the organisms different from one another. Signal transduction and gene regulatory networks are crucial factors in intracellular regulation. These networks are compact, sparse and exhibit increased clustering. They show a small-world property and scale-free topology as a result of biological evolution [98].

The majority of biological processes in living cells are regulated by proteins. They perform this task by interacting with other molecules, such as lipids, nucleic acids, low molecular weight compounds and other proteins. These interactions are modeled as networks and are extensively analyzed and visualized. The graph layout plays an important role in analyzes of protein networks, as discovering and noticing motifs and cliques are important since they play prominent roles as operational units in biological functions. They show a small-world property and scale-free topology as well [14].

The term metabolic network corresponds to a network constructed of metabolites and their biochemical reactions in an organism. Another important concept in biochemistry related to metabolic networks is that of metabolic pathways, which can be considered as small portions of a metabolic network. A metabolic pathway defines a series of biochemical reactions for a specific metabolic function, such as penicillin biosynthesis. On the other hand, a metabolic network gives a comprehensive outlook of the cellular metabolism. A complete metabolic network should describe all possible ways of material flows in the cell. This network plays a role in survival and growth of the cell as it is responsible for generating energy and synthesizing required components. The understanding of these networks is important as they are used in many ways. They are used as cellular factories to produce various chemicals, antibiotics, antibodies, and so on. Additionally, through better understanding of these networks we can control the infection of pathogens by using the metabolic differences between pathogens and humans [117]. Additional explanations on this type of networks can be found in this thesis [62].

Any graph used to visualize the evolutionary relationship between species, genomes, chromosomes, genes, or nucleotide sequences is defined as phylogenetic network [55]. Several methods for the reconstruction of phylogenetic networks exist. However, different analyzes and models, such as mechanisms operating at a

microevolutionary level are still at a relatively early stage of development [37].

Ecological networks are crucial in understanding the dynamics of the individual groups of organisms and of the entire ecosystem. They usually represent networks of consumer-resource interaction between a group of organisms, namely food webs [97]. Ecological networks show who is present and who affects whom by different interactions, such as feeding [35].

Correlation networks represent an exception among biological networks. These networks are being investigated relatively recently. What divides them from other biological networks is the way they are created. They are not a direct result of experimental data, but they are created by collecting huge amounts of high-throughput data and calculating the correlations between all elements [122].

All these different biological networks are essential for an overall understanding of living beings. The constant progress of knowledge and technology has accelerated the process of acquiring data for these networks. This has resulted in the creation of large and complex networks which are increasingly hard to interpret and visualize. An example is the network information managed in the KEGG database [68], which contains hierarchically structured pathways with in total more than 10,000 nodes representing genes, proteins/enzymes, and metabolites. Figure 2.6 shows an example of such a pathway. Although not all the network types were the aim of our research, it is important to understand their similarities and differences as many of the solutions applied to one type could be relatively easily modified for use with other types. Moreover, with little more effort, these problems could be generalized on other domains of science as well.

Tools for Visualization of Large Biological Networks

In order to cope with such large and complex networks, several systems and methods have been developed. Many tools and databases for visualization and analysis of biological networks are available online. Systems such as CellDesigner [34], Cytoscape [113], ONDEX [83], Pathway Projector [85], PathVisio [131], VANTED [60], and VisANT [53] represent some of the more popular software systems for biological network visualization. Many approaches are straightforward, such as cases of common graph drawing and analysis tools: they try to visualize the complete network and depend on common interaction techniques such as zooming and panning for navigation.

Ontologies and Clusterings

Ontologies are used by biology and medicine researchers to structure biological knowledge. Ontologies can be described as a set of controlled, relational vocabularies of terms used in a specific area of science. In life sciences, structuring and standardizing of biological knowledge to support data integration and information exchange is achieved by using ontologies. Examples are Gene Ontology (GO – to standardize gene and gene product attributes across species), Molecular Interactions

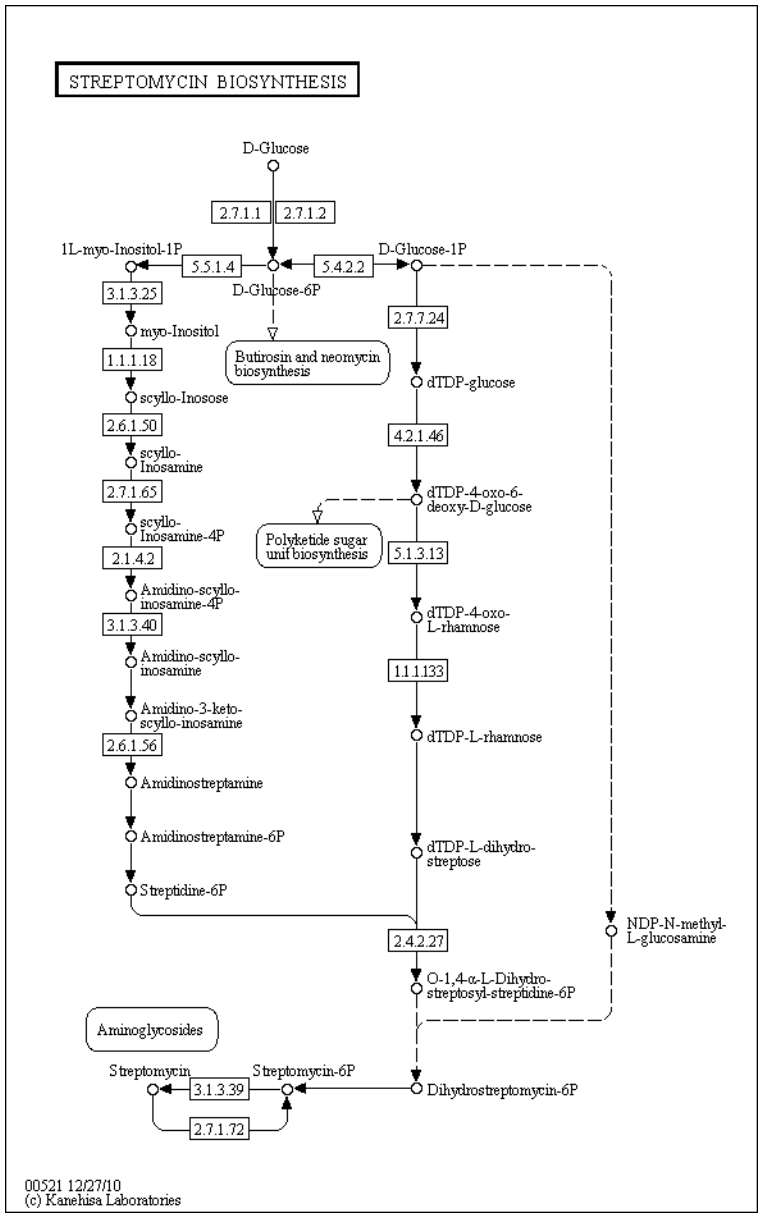


Figure 2.6: The diagram shows an example of a KEGG reference pathway. The image represents the *Streptomycin biosynthesis – Reference pathway* [1].

Ontology (PSI MI – to standardize molecular interaction and proteomics data), and Systems Biology Ontology (SBO – to standardize terms commonly used in computational modeling and systems biology). Many of these ontologies are accessible through the Ontology Lookup Service (OLS) [20], which provides a web service to query multiple ontologies from a single location, providing a unified output format. Often biological experiments (that produce experimental data) are carried out and analyzed in context of biological ontologies, for example, by means of enrichment of ontology terms to identify statistically overrepresented (inner) ontology terms.

In this thesis, we will focus on the Gene Ontology (GO) [38]. GO is an online database that provides a set of structured vocabularies (ontologies) for the annotation of genes, gene products and sequences. These vocabularies support a consistent representation of gene products in various databases and describe the roles and properties of genes or gene products in organisms. Currently, there are three independent vocabularies (or parts) that are considered by the GO: molecular function, biological process, and cellular component. Such vocabularies are used by biologists as guides to answer significant questions, e. g., "if you were searching for new targets for antibiotics, you might want to find all the gene products that are involved in bacterial protein synthesis, but that have significantly different sequences or structures from those in humans" [38]. An important feature of the GO is that new discoveries are made daily. These new findings change our understanding of roles and properties of gene or gene products, thus making GO a dynamic data set. The GO terms are interconnected and form a directed acyclic graph (DAG) [7, 124]. Figure 2.7 represents a conceptual drawing of a GO.

Large-scale experimental data in life sciences are often analyzed and visualized using hierarchical clustering [30]. It is a statistical method for finding relatively homogeneous clusters, based on two steps: (1) computing a distance matrix containing the pair-wise distances between the biological objects (such as genes) and (2) a hierarchical clustering algorithm. Hierarchical clustering algorithms can work

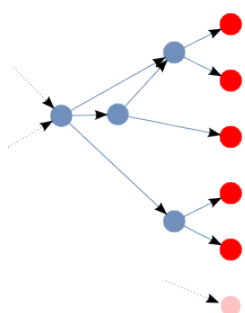


Figure 2.7: *Blue* nodes represent GO terms forming a DAG that describe the roles and properties of gene or gene products denoted with *red* nodes.

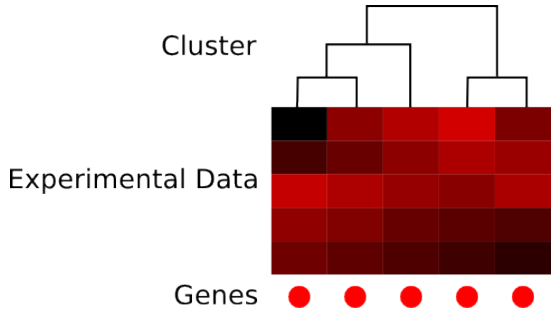


Figure 2.8: Gene or gene products are placed at the bottom (*red* nodes). Multivariate experimental data are represented by a heat-map. The hierarchical clustering is computed based on this data and shown as dendrogram on top.

in two ways (top-down or bottom-up). They will either partition clusters, starting from the complete data set, or recursively join the two closest clusters. After each clustering step, a distance matrix between the new clusters and the other clusters is recalculated. Figure 2.8 shows how the outcome of such a process might look like. This type of visualization [30, 112] is standard for representing such data, although other strategies exist as well [21, 22]. An experiment has been performed on a number of genes (red circles in Figure 2.8). The experiment resulted in a number of different values showed as "heat-map" on top of the genes. After performing the two steps of the clustering algorithm, a binary tree denoting the clusters has been created (top of the image in Figure 2.8). At this point, one might notice that the experiment produces multivariate data. The clustering is a product of such data.

The analysis of molecular-biological data obtained by high-throughput technologies is often supported by ontologies and hierarchical clustering. Data is constantly produced by these high-throughput technologies. This enables the investigation of various biological data or systems under different conditions and at different developmental stages, or even with diverse genetic backgrounds.

The main issue with ontologies and hierarchical clustering is the size of the data. Ontologies result in huge data sets with a DAG-like structure, while hierarchical clusterings usually produce large tree-like structures. Often both views are desired to support analyzes of this data: representing a data set (such as the expression levels of the genes in an organism) in the context of an ontology (such as the Gene Ontology) and in the context of a clustering of the data (such as an hierarchical clustering). There are many approaches which deal with data that are part of two trees or hierarchies or that compare two trees. However to our knowledge, no solution exists for visualizing hierarchical clustering (tree) and an ontology (DAG) of the same data set in one visualization system. Our tool presented in Chapter 7 directly addresses this challenge.

In the following, a typical example of such an exploration is presented, namely the example of transcriptomics data. The transcriptome represents the set of all RNA in a cell or population of cells. It gives a snapshot of the current gene activity within a cell and it is measured by DNA microarrays or sequencing. Hierarchical clustering is usually employed in order to identify and classify patterns of gene expression for this type of data. It results in an ordering of the genes such that clusters of co-expressed genes are visualized and can be used to infer gene function. GO however gives a hierarchical annotation of gene function. In order to better understand the roles and/or functions of the genes, a combined analysis of gene activity in both hierarchical clustering and ontologies is desirable. For instance, a small cluster of genes in the hierarchical clustering might be highlighted by selecting a specific cluster. If visual investigation of the corresponding genes in the GO shows that most of these genes belong to the same subgroup within the ontology, then this gives a strong indication that these genes are not only assigned to the same function but also that this function may be of particular importance (as the activity of these genes behaves similarly). However, it may be also of interest to investigate these functions in more detail even if the genes of a cluster in the hierarchical clustering belong to many different ontology concepts (assigned functions). Finally, the enrichment of ontology terms in the data is used to identify statistically overrepresented ontology terms, giving insight into relevant biological processes or functional modules. If the respective genes belong to the same cluster in the hierarchical clustering it means that the enrichment of clustering has been obtained independently with these two different methods. Therefore, a typical user session would be browsing the data to investigate the relation between functional annotation in the ontology and behavioral grouping of gene activity in the clustering.

The problem of comparing two or more trees with the same set of leaves could be considered as similar to the presented approach. These comparisons are common in cases of phylogenetic trees. Drawing the trees side by side in opposite directions and connect the corresponding leaves is a usual way to perform such comparisons. The challenge here is to compute good leaf ordering and tree visualizations. There exists a considerable amount of research in that direction [27, 33, 134]. However, the problem described above, i.e., comparing a tree resulted from hierarchical clustering with DAG (ontology) with the same set of final leaves is relatively new. The concept of tanglegrams for rooted phylogenetic trees was introduced by Scornavacca *et al.* [110], however it still uses the approach of representing two trees or networks side by side in opposite directions and drawing the connectors between the corresponding leaves.

2.4 Summary

In this chapter, we briefly presented basic concepts of graph drawing and visualization of networks in context of information visualization. We discussed some of

the visualization and interaction techniques that are related to the work presented in this thesis. A brief overview of different types of biological networks was discussed followed by the current approaches to visualize such networks. We concluded this chapter with the description of the concepts of ontology, focusing on GO and hierarchical clustering in biology. In the next chapter, we discuss and classify various techniques that deal with multivariate network visualization.

Chapter 3

State of the Art of Multivariate Network Visualization

The amount of data produced every day in the world is a huge challenge in understanding and extracting knowledge from it. A lot of these data are of relational nature, such as social networks, biochemical pathways, or links between software components. These networks are traditionally represented as node-link diagrams or matrix representations. Node-link diagrams are the focus of this work, because they are suitable for understanding the structure and topology of relational data and mostly used in practice. However, the topology and structure of the graph alone is not always the focus of the analysis process. These graphs usually have a huge amount of additional attributes related to nodes, edges or even node/edge clusters, although these clusters themselves are often products of attributes, thus they could be used to gain insight into the attribute data. The challenge is to show these attributes in context of the underlying graph topology. The problem is becoming more and more important as a number of researchers in many fields need solutions for their daily work.

In this chapter, a brief survey on multivariate network visualization tools and approaches is presented. The approaches are classified into several categories based on the used visualization techniques. Domain and attribute related problems are discussed too. A table listing a number of approaches/tools gives insight on the technique, nature of attributes and the domain the tool is primary used for.

3.1 Characteristics of Multivariate Networks

Multivariate Data

A formal definition of multivariate networks was presented in the previous chapter. Here, visualization approaches for such networks are investigated. However, before describing these approaches, a brief discussion about multivariate data visualization in general is presented as most of the visualization techniques can be reused or adapted for multivariate networks. The term *multivariate data* in information visualization is used to describe data that contain more than three attributes or variables [119].

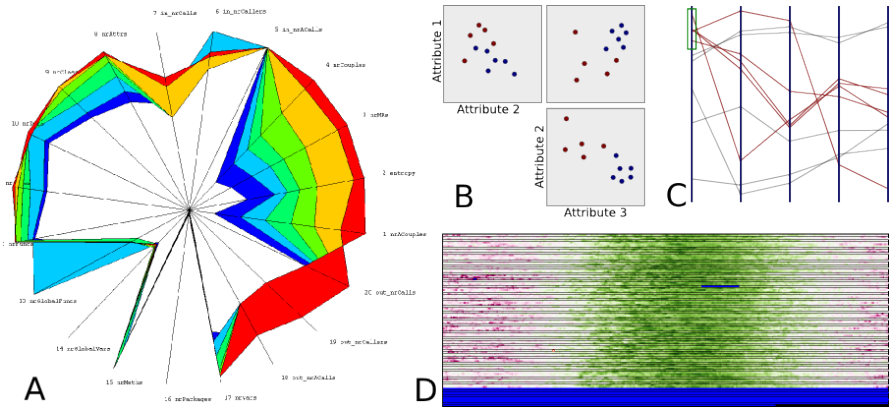


Figure 3.1: Samples of multivariate data visualization techniques. The image marked with label A is made with a tool presented by Kerren *et al.* [76, 77] based on the original work of Pinzger *et al.* [96]

A considerable number of techniques for visualizing multivariate data exist. They are generally grouped into four approaches: projection-based, coordinate axis-based, icon-based, and pixel-based. Projection-based approaches project different attribute values in a two- or three-dimensional coordinate system usually in small-multiples fashion, such as Scatter Plot Matrices [45] (cp. Figure 3.1 – B). Coordinate axis-based approaches map the attribute values into different coordinate axes. Examples of such approaches are Parallel Coordinates (cp. Figure 3.1 – C) or Star Plots diagrams (cp. Figure 3.1 – A). Icon-based approaches use glyphs to visualize the data. One of the textbook examples of this approach are Chernoff Faces [18]. And last but not least, pixel-based approaches map attribute values to a single pixel (cp. Figure 3.1 – D) [119] .

Almost all the presented approaches rely on various interaction techniques to provide additional help for coping with large multivariate data. For instance, different visual filtering approaches can reduce clutter. The use of focus+context techniques, such as different distortion techniques, can help to explore the desired data objects while keeping the overall context of a data set. More details about these issues and the definition and use of relational data, i.e., networks and graphs, were discussed in the previous chapter. Therefore, we continue with the description of multivariate attributes in networks in the following section.

Attributes in Networks

Even without any other additional information, huge networks present a great challenge to be visualized. However, the focus of this thesis is the visualization of both,

the attributes and structure of networks. Also, real life networks carry additional information such as:

- Attributes belonging to the data elements themselves (attributes belonging to the nodes of the network)
- Attributes describing the relation between the data elements (attributes belonging to the edges of the network)
- Data derived by some computation on the attributes of the network (for example, a clustering tree produced by hierarchical clustering)
- Attributes derived by some computation on the network itself (for example different network centrality values)

Let us take an example from social networking. Each person can be modeled as a node, and the friendship between persons as an edge connecting the nodes. Each node (person) has a number of different attributes, such as *name*, *age*, *gender*, *interests*, etc. An edge can also have additional attributes, such as *family relation* (mother, father, sibling, etc.) or a *friendship weight* (could be calculated based on the activities such as chatting or sharing same interests between two persons). When dealing with huge networks, some sort of clustering may be introduced. It is often important to know the average values of the elements belonging to a cluster. Additionally, some new attributes can appear as a clustering result.

Other additional algorithms are used to analyze the networks in fields such as in Biology or Social Network Analysis (SNA). These algorithms may produce additional data, such as network centrality metrics, that can be regarded in the same way as node, edge or cluster attributes, and visualized by using similar approaches. However, some tools are designed to specifically cope with these types of data [25, 79].

3.2 Visualization Approaches

Several ways to visualize the networks and their additional data exist. Some of the approaches show the topology and graph structure better than the multivariate attributes and vice versa. Investigation of the different approaches and tools have shown that many of them share some conceptual ideas in the way they represent the network and the additional attributes. Several visualization approaches are grouped with respect to the way the visualization is made. In the following, the criteria for such groupings are presented.

(A) Allow use of standard graph drawing algorithms

- A.1 Attributes visualized separately from the network visualization
- A.2 Attributes visualized together with the network visualization

(B) Use attribute values for graph drawing

- B.1 Nodes are positioned in specific non-overlapping regions, i.e., nodes with similar attribute values are placed in dedicated areas that do not overlap with other areas
- B.2 Nodes are positioned in specific positions and/or regions, i.e., nodes with similar attribute values are placed close to each other or in dedicated areas which might overlap with other areas

The top level criteria (A and B) focus on the ability of the approaches to show the underlying graph topology. Some visualization approaches allow the use of standard graph drawing algorithms in order to optimize the perception of topology (satisfying Criterion (A)), while other approaches affect the placement of network elements to emphasize the attribute values (satisfying Criterion (B)), which in effect lowers the perception of the network topology. Regarding the way attributes are visualized, criterion (A) has two sub-criteria: whether the attributes are visualized in the same view with the underlying network (A.1) or in distinct views (A.2) of which at least one has to satisfy the (A) criterion. The (B) criterion has also two sub-criteria which specify the way the network attributes influence the network topology: whether the nodes are placed in specific non-overlapping regions (B.1) or not (B.2). Some real life systems use different approaches that could satisfy more than one criteria. In the following, the groups derived based on these criteria are briefly introduced.

The approach that satisfies criterion (A.1) is *Multiple Coordinated Views* which uses a combination of two or more combined linked views to represent the network and the multivariate data. This approach is also mostly found in combination with others creating a *Hybrid Approach*. *Integrated Approaches* visualize the network and the underlying multivariate data in a single view, satisfying criterion (A.2). *Semantic Substrates* position the nodes into separate non-overlapping areas based on the node attributes (B.1). *Attribute-Driven Topology* uses a similar idea to Semantic Substrates, but instead of placing nodes to non-overlapping regions, it just affects the positioning of the nodes according to attribute values, effectively satisfying criteria (B.2).

An analysis of strengths and weaknesses of each approach separately has been performed which is explained in more detail later in this chapter. Table 3.1 presents different articles and tools that have been reviewed, while Figure 3.2 shows how these approaches are used to classify them in respect to the criteria mentioned above.

Multiple Coordinated Views

When providing several linked views together, the users should experience a seamless interaction among different views. Therefore the use of brushing and similar techniques is necessary. Changes on a particular object in one view should be reflected in all other views, unless the user specifically requires to avoid that. There is

		Attributes			Approaches							
						Integrated						
Paper / Tool	Domain	Node	Edge	Cluster	Multiple Coordinated Views	Node	Edge	Cluster	Semantic Substrates	ADT	Hybrid
Multivariate Graph Drawing using PCV [114]	General	•			•						
Network Lens[63]	General	•				•					
DBE Information System [13]	Metabolic Networks	•				•					
GraphDice [10]	Social Networks	•	•	•						•	
GeoSOM [142]	General	•								•	
Jigsaw [120]	Document Analysis	•			•						
RelVis [96]	Software	•				•					
NVSS 1.0 [116]	General	•		•					•		
PivotGraph [138]	General	•							•		
Pretorius et al. [99]	General	•		•					•		
Pretorius et al. [100]	Transition graphs	•	•	•					•		
MobiVis [115]	Mobile data		•	•	•	•		•			•
Dynamic Graph Analysis [107]	Metabolic Pathways	•	•		•	•	•				•
GrouseFlocks [4]	General	•	•	•					•		
TugGraph [5]	General	•		•					•		
PEx-graph [89]	Social Networks	•								•	
en-Route [95]	Metabolic Pathways	•	•		•						
Vehlow [133]	Biochem. Reaction Networks	•	•		•						
CluMaGO [65]	Gene Ontologies			•	•						
ViNCent [146]	Network Centralities	•			•	•					•
JauntyNets [67]	General	•			•	•				•	•

Table 3.1: Multivariate Approaches. The table shows information about different approaches for visualizing multivariate networks. The *Domain* is shown through its own column. There are three sub-columns under the *Attributes* column denoting what kind of attributes are visualized by the corresponding tool with respect to the network elements. The *Approaches* column specifies to what kind of approach discussed in this chapter the tool belongs to. The sub-column *Integrated* is divided into three more columns stating to which part of the network the attribute visualization has been embedded to.

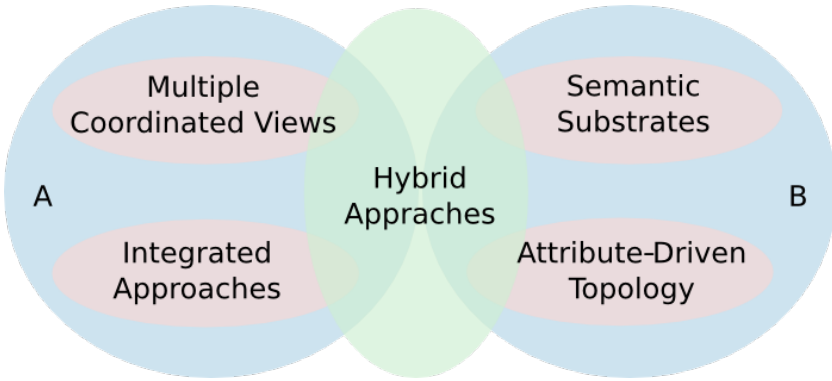


Figure 3.2: Classification of approaches. The left blue circle (A) represents the approaches that are able to represent the underlying network topology, in contrast to the right blue circle (B) that represents approaches focussing on the attribute values of the network.

a clear advantage using this approach as one may choose the most powerful visualization techniques for each specific view and data set [42, 105]. At the same time, it allows the use of the most suitable graph drawing techniques to layout the graph, as the multivariate data is visualized in a separate view and will not interfere with the layout process. However, due to the spatial separation of the visual elements, the displayed data is split in multiple views. This might introduce a scalability problem with large networks as the objects will most likely be represented in more than one view, thus consuming additional space.

The work of Shanon *et al.* [114] presents one example of this idea in the network visualization domain. They use two distinct views: one view shows a parallel coordinate approach [56], and the other view displays a node-link drawing of a graph (cp. Figure 3.3). Their tool offers a variety of visualization and interaction techniques, while maintaining coordination between the views through brushing and linking [119].

Another approach based on multiple coordinated views is Jigsaw, a tool for document analysis [120]. While using additional views to provide more information about the network, users can model views of relational data of different documents and entities as a graph representation. Although the tool is primarily designed to visually explore the relationships between different data elements, it provides possibilities to visualize additional data. For instance, the users can inspect the text of a document in a separate view, or they can visualize time-dependent data in a special calendar view. The tool uses the multiple coordinated metaphor so extensively that the authors propose that Jigsaw should be ideally used in multi-display environments or on large, high-resolution monitors.

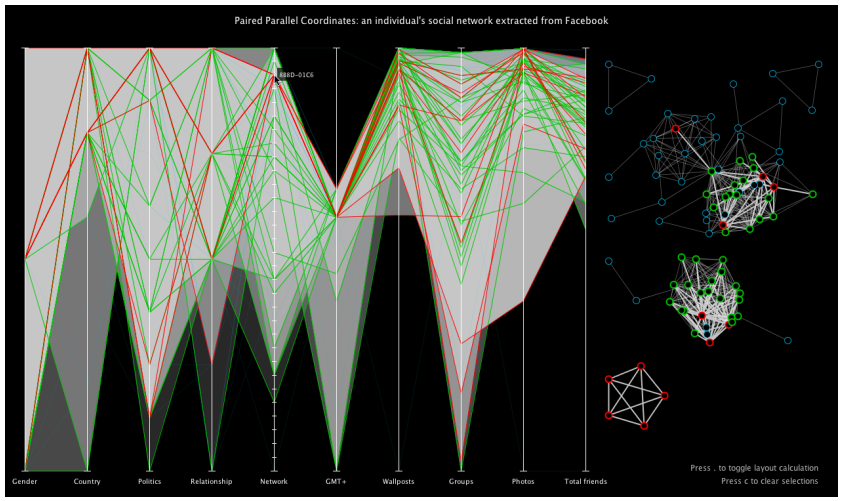


Figure 3.3: Multiple Coordinated Views approach. The left side of the image represents a visualization of multivariate network data by parallel coordinates, while the network is shown in a separate view on the right. The picture was taken from [114] with permission of the authors.

A more recent tool visualizes metabolic pathways in one view and their experimental results in another view [95]. The approach works for attributes related to nodes and edges. The users should first specify a part of the pathway they are interested in. This highlighted part is then shown in a separate view with the experimental data on the side connected to a copied part of the highlighted pathway. At the same time, a tool that visualizes biochemical reaction networks and their multivariate data in multiple coordinated views with support for uncertainty-aware analysis was presented [133].

Integrated Approaches

In contrast to the multiple coordinated views, integrated approaches provide a combined visualization, where the underlying graph and its attribute data are presented in a single view. "Integrated views can save space on a display and may decrease the time a user needs to find out relations; all data is displayed in one place." [42]. As explained earlier, attributes can belong to nodes, edges and/or clusters. Based on the data set and requirements, attributes can be visually integrated accordingly.

Visually integrating up to four or five attributes in a node is rather straightforward, of course depending on the data being visualized. Labels can be used for textual attributes and other different features, such as size, shape, color and stroke for other attributes. As the number of attributes grows, finding new visual features

to map these attributes becomes more challenging as their number comes to an end. There might be a need to introduce new visual metaphors to cope with the increasing number of attributes. Usually, some visual metaphors found in a "common" multivariate data visualization are used in such cases, for example barcharts. In general, the examples explained in this subsection are instances of embedding *glyphs* into networks. They are graphical entities that convey multiple data values via visual attributes, such as shape, color, position, or size [135].

A straightforward example of integrating node attributes is described in the work of Borisjuk *et al.* [13] on the visualization of experimental data in relation of a metabolic network. Instead of representing nodes as simple circles or rectangles, small diagrams have been embedded inside them to represent experimental data that is related to the regarded node. The diagrams were usually simple projection based visualizations of multivariate data such as barcharts (cp. Figure 3.4). This approach provides a view to all available information, but with additional costs. In order to facilitate the employment and the readability of the barchart diagrams, the size of the nodes in the graph needed to be enlarged. This issue may affect the readability of the network, especially if the number of nodes and attributes is high resulting in possible clutter and overlaps [75]. Thus, it does not scale well.

Another example of integrating well known multivariate data visualization techniques into nodes is presented by Pinzger *et al.* [96]. The authors use an improved version of so called Kiviat diagrams to represent different source code metric values in a number of software releases in the context of a network. This tool has the same advantages and disadvantages as the previous example.

However, the problem of space usage and clutter introduced by this approach can be avoided by employing different focus+context techniques. An example of such an approach visualizes the attributes inside nodes by using a fisheye lens. Users can choose between several different visual representations and make virtually unlimited number of combinations of desired attributes to be visualized [63].

Semantic Substrates

Semantic substrates were introduced in order to avoid clutter in multivariate network visualizations. The main idea is that substrates "are non-overlapping regions in which node placement is based on node attributes" [116]. Specific semantics derived from the nodes' attributes was used to place the nodes in specific substrates (regions). Additionally, the authors used sliders to control the edge visibility which ensured the comprehensibility of the edges and nodes.

PivotGraph [138] shows the relationship between (node) attributes and edges by using a grid-layout. Nodes and edges that have identical values for specific attributes are aggregated. The size of the resulting nodes and edges represent the degree of aggregation while the color is used to code the attributes.

Another approach was presented by Pretorius and van Wijk [100]. They arrange edge labels in a list located in the center of the view and place rectangular regions

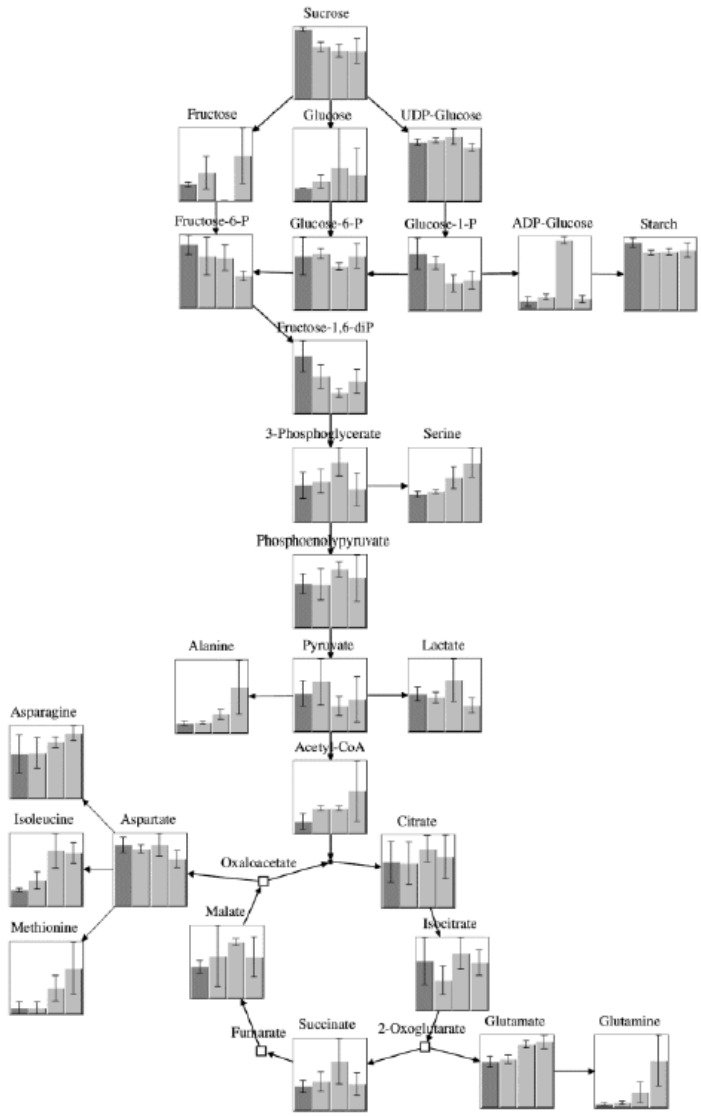


Figure 3.4: An example of an Integrated Approach shows experimental data embedded into a biological network. The picture was taken from [13] showing the relative levels of different “Vicia narbonensis” lines integrated into the glycosis and citric acid cycle. Image courtesy of IPK Gatersleben.

containing source and target nodes at each side. These regions are recursively partitioned according to the node attributes resulting in a specific positioning of nodes based on the created hierarchy. The nodes are then connected via straight lines with corresponding edge labels. Similarly, the same authors performed hierarchical clustering based on node attributes to place the nodes into specific regions in their previous work [99]. Although the nodes "overlap"—regions can be contained in other regions depending on the hierarchy level—they spatially remain in the same non-overlapping positions. Similar works have been presented by Archambault *et al.* in [5] and [4]. One conceptual drawback of these approaches is that the underlying graph topology is not (completely) visible.

Attribute-Driven Topology

Discovering different network structures is an important part of network visualization and analysis, and it is desirable to have a good graph layout algorithm when doing any kind of network visualization. In most of the cases of visualizing multivariate networks, a sufficient layout algorithm would reduce the scalability problem, which is one of the ongoing challenges in information visualization [88]. However, one could use the network layout to present insights about multivariate data belonging to network elements in cases where network topology is not the key component in the analyzes. Of course by doing this, the network topology is usually sacrificed and it becomes relatively hard to visually detect different network structures. This approach is similar to semantic substrates, however it does not necessary place the nodes into specific regions. It uses some calculations based on the node attributes to "steer" the placement of the nodes in the graph.

One example of such a tool is GraphDice [10] which extends the ScatterDice [31] tool used to visualize multidimensional data through scatterplots matrices. The system uses multiple scatterplots made of various node, edge or computed attributes to layout the graph. Each point in the scatterplot represents a data object and a line between points represents the edge of the object. Users can browse the data by selecting a combination of various attributes for the horizontal and vertical axis and see the outcome of the graph layout. This helps noticing different relations between attributes as well as identifying potential clusters.

Another example for such approaches uses spherical Self-Organizing Maps (SOMs) to layout the graph on the surface of a sphere. A SOM is an artificial neural network used as dimensionality reduction technique [84]. The high-dimensional node attributes are used to calculate the projection of nodes in a 2D plane.

In this case, the surface of the sphere is used to avoid displaying boundaries as they cannot show the relationships between data placed in the opposing borders of the 2D plane [142]. Similarly, Martins *et al.* [89] apply multidimensional scaling techniques to project nodes on a plane based on the attribute values. This approach was used for visual analyzes of social networks.

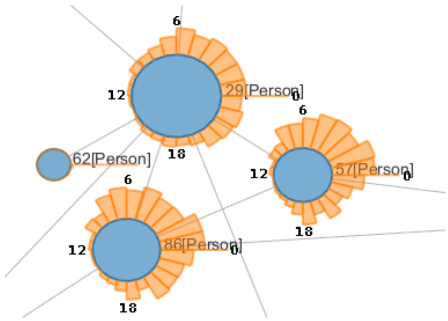


Figure 3.5: An hourly ring of calls shows the frequency of calls every hour. The picture was taken from [115] with permission of the authors. ©2008 IEEE.

Hybrid Approaches

Hybrid approaches are created by combining at least two of the already discussed approaches into one. The reason for this kind of combination is to harvest the advantages of each approach that is being used. The most common combinations are multiple coordinated views with any of the integrated approaches.

For instance MobiVis [115], a tool for visualizing social-spatial-temporal mobile data, uses a time chart to show temporal data in one view and the underlying network as a separate view as node-link diagram. It can additionally show data using "Behavior Rings" that are pie-shaped wedges placed around the nodes (Figure 3.5). The size of the wedge is mapped to the value of the attribute. The wedges are placed around the node to allow the space inside the node to be used for mapping additional attributes if necessary.

Another hybrid approach integrates additional attributes of a biological network inside the nodes and edges [107] by mapping different visual features of the nodes and edges to different attribute values. Attributes can also be integrated as small glyph inside the nodes. At the same time, it uses other visualization metaphors creating multiple coordinated views to show time related data of the network. Users can explore the evolution of the multivariate data as well as of the overall metabolic pathways through a series of different interactions.

One approach that was part of our own work, but not directly related to this thesis is ViNCent [146]. The tool visualizes multiple network centralities. It visualizes the centrality values by using staked bars on top of nodes placed in a circular layout. It uses multiple coordinated views to support the centrality analyzes. Several interaction possibilities for the integrated views are provided. Users can order nodes by centrality values or filter out specific node by using other visualization views. Figure 3.6 shows a screenshot of the tool.

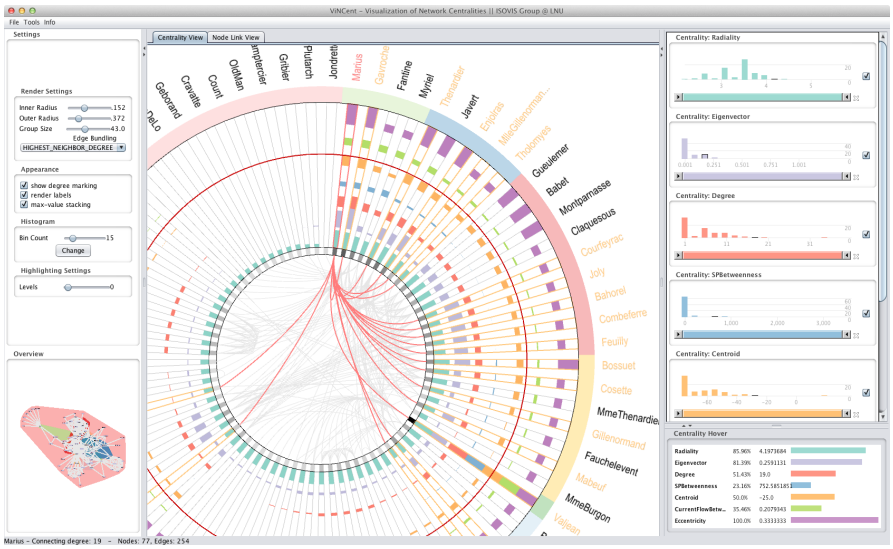


Figure 3.6: A screenshot of the ViNCent tool [146].

3.3 Summary

General multivariate data visualization aspects outside the network visualization context were discussed briefly at the beginning of this chapter. This is because a number of techniques could be directly applied to multivariate network visualizations. Then, a set of criteria for categorizing different multivariate network visualization approaches was created. For this, the ability of the approaches to show the network topology was taken into account. Additionally, the way how the additional network attributes are visualized was also considered. All discussed approaches were categorized based on these specific criteria. They helped us to define four distinct groups and a fifth hybrid category.

Chapter 4

Usability Study of Multivariate Network Visualization Approaches

In this chapter, the methodology and the results of a task-based usability study for comparing two multivariate network visualization approaches are discussed. A great resource for designing such studies is Helen C. Purchase's book that lists different practical examples of similar studies [102]. Although several groups of approaches for visualizing multivariate networks were defined in Chapter 3, the study focuses only on integrated and multiple coordinated view-based approaches. The aim of the study is to explore the differences between approaches in terms of their *effectiveness* and *efficiency* to represent the multivariate attributes in context of the objects they are attached to, and in context of the user perception of topological features of the network. Kulyk *et al.* [86] define effectiveness as "the accuracy and completeness with which users achieve certain goals." Efficiency is defined as a relation between the effectiveness with which users achieve certain goals and "the resources expended in achieving them."

The rest of this chapter is structured as follows: first, the hypotheses of our study are briefly presented. Then, the web-based tool used to convey the study is described in detail. Data about the study participants in terms of their gender, age, education and familiarity with networks are highlighted next. Finally, we present the results of the study and discuss how they compare against our initial hypothesis.

4.1 Hypotheses

The main idea of the study is to compare the effectiveness and the efficiency of integrated and multiple coordinated views approaches in context of their ability to present the multivariate data and visualize the network topology. The usability study is limited to comparing only two approaches as they allow the use of the same metaphor for the attribute visualization. Other approaches, such as attribute driven layout and semantic substrates, do not show the attribute values directly, but they place the nodes in specific areas based on attribute values. Moreover, there are many variations of how the attribute based node positioning is calculated and represented, which makes it even harder to compare them with our chosen approaches. Our initial assumption was that integrated approaches are better for identifying parts of

the networks holding interesting multivariate attributes as they insert the data inside such structures, while multiple coordinated views split the view, thus requiring additional effort from the user to perform the mapping. Another assumption is related to the perception of topological features of the network. As integrated approaches require more space to facilitate the visualization of more attributes, clutter is introduced. This clutter might affect the network readability, prompting us to believe that multiple coordinated approaches might perform better in this context. The final assumption was that there should be no significant change between the approaches when dealing with sparse graphs. Therefore, based on these assumptions, several initial hypotheses were developed:

1. Integrated approaches are better for identifying parts of the networks holding interesting multivariate data in dense (congested) graphs.

2. No significant change between approaches should be present for sparse graphs in terms of identifying parts of the networks holding interesting multivariate data.

3. Multiple coordinated views are better for gaining insight into the graph topology for dense (congested) graphs.

4. No significant change between approaches should be present for sparse graphs in terms of showing insight into the graph topology.

4.2 Methodology

The online study was mainly intended towards people who are at least university level students. The subjects were invited via email to participate in the study. The list of the email recipients included students, PhD students and faculty members at different universities. They were also asked to forward the email to other potential subjects. The data were collected on our server during ten days and was analyzed using SPSS [19]. As mentioned earlier, this was a task-based usability study, i.e., the subjects should perform certain tasks in context of multivariate network analysis. A web-based application was developed especially for this purpose. Its initial page provides information about particularities of the study. If the test person is familiar with the overall concept of the study, he/she needs to click on a “Start Survey!” hyperlink to participate. First of all, the subjects should fill out a demographics form. Here, information about *age*, *gender*, *education level*, and *major field of study* were gathered. One additional item was related to the subject’s *knowledge about graph/network visualization*, where they could chose one of the following answers: “None”, “I learned a bit in university about it”, and “Expert”.

After submitting this information, the subjects were asked to perform a list of tasks. For a given set of different graphs shown to them, four different tasks were

required to be performed for each graph. These sets of graphs were of different nature (sparse and dense), and they visualized the attributes by using both approaches. For the first task, they had to identify (by mouse-click) the node of the network with the *lowest* average attribute values. The task was timed in order to find out how long it takes for a subject to find such a node. This is important for investigating the efficiency for hypotheses 1 and 2. The accuracy of the responses would answer the issue of effectiveness for the same hypotheses. The second task is very similar to the first one. However in this case, the node with the *highest* average attribute values was required. The purpose of the third and fourth task is to answer the hypotheses 3 and 4. For the third task, the subject is asked to identify if the presented graph is a tree. He/she can answer the question by choosing "Yes", "No", or "Maybe". The third option was given to avoid getting false results if someone is not sure about what such differences mean, although a figure for explaining the two types of graphs was shown on the first page of our application. This task was timed as well. For the final tasks, subjects had to grade the readability of the network topology for the given graph with values from 1 to 5 (higher values mean better readability).

Eight different graphs were designed by hand. Four of them were sparse graphs (SP1-SP4) with little over 13 nodes in average, and four of them were dense graphs (DE1-DE4) with 47 nodes in average. Both, sparse and dense graphs were drawn in the same area. One of the graphs in each group was a tree with unspecified root. Each graph had two versions: one for integrated approaches (IA) and one for multiple coordinated views (MCV), resulting in 16 different combinations in total. In order to have the same base for comparison only one visualization metaphor for attribute representations – a bar chart – was used for both approaches. It was embedded into the nodes for the integrated approaches as shown in the Figure 4.1. Figure 4.2 shows another graph, but in this case it is a dense graph and the multiple coordinated approach was used to visualize the attributes. To mimic the effects of this approach, the attributes are placed on the right hand side of the network representation. Please check Appendix A to view a list of the graphs used in the study that are not shown in this chapter.

In order to avoid the possibility that subjects might memorize graphs, each one is shown by using only one of the approaches. For instance, sparse graph SP1 is shown to the subject only once, either by using integrated approaches (SP1-IA) or by using multiple coordinated views (SP1-MCV). Therefore, only 8 out of 16 possible graph versions are shown in one session. Another reason for this decision is to avoid too many tasks, thus losing the interest of the subjects. With such a set up, the subjects should not take more than 15 minutes to complete the study for all given examples. Therefore, several pipelines holding a sequence of different combinations of graphs and approaches were developed for this. The pipeline is picked sequentially for each new subject entering the study. Table 4.1 shows pipelines of graphs that were generated for this purpose.

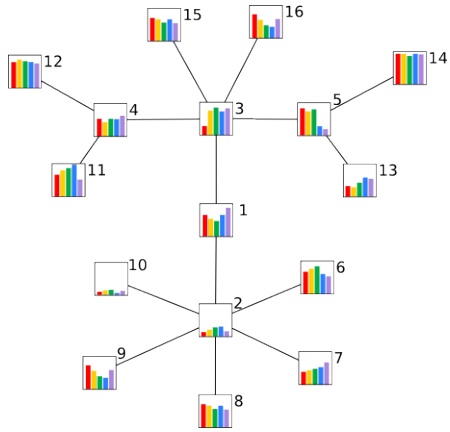


Figure 4.1: Showing a sparse graph named SP3 visualized by using an integrated approach. It is the only tree among the sparse graphs. The node with label 10 is the correct answer for the first question, while node with label 14 is the correct answer for the second question.

Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8
SP1-IA	SP3-IA	DE2-MCV	DE4-MCV	SP2-MCV	SP4-MCV	DE1-IA	DE3-IA
SP1-MCV	SP3-MCV	DE2-IA	DE4-IA	SP2-IA	SP4-IA	DE1-MCV	DE3-MCV
DE2-IA	DE3-IA	SP4-MCV	SP1-MCV	DE4-MCV	DE1-MCV	SP2-IA	SP3-IA
DE2-MCV	DE3-MCV	SP4-IA	SP1-IA	DE4-IA	DE1-IA	SP2-MCV	SP3-MCV

Table 4.1: Four different pipelines a participant can be assigned to. The first subject is assigned to the first pipeline, the second to the second and so on. The process is repeated for all subsequent subjects.

4.3 Results

There were 35 subjects who finished the study completely. Additional two subjects were disqualified. The first one did not finish task one, two and three in any of the given graph samples. The second subject was disqualified as his/her answers were varying a lot in terms of time required to complete the tasks. Two persons reported problems with the application, which we could not reproduce in any of the browsers or operating systems we tested. More than half of subjects were *male* (cp. Figure 4.3(a)), while the majority of them are *20 to 30 years old* (cp. Figure 4.3(b)). This corresponds to the level of education as considerable amount of the people are *Master* or *PhD students* (cp. Figure 4.4(a)). Most of our subjects have a background in *computing* (cp. Figure 4.4(b)). As shown in Figure 4.5, majority of the participants have *some* knowledge about network visualization. Next, the results in context of the defined hypotheses are discussed in detail.

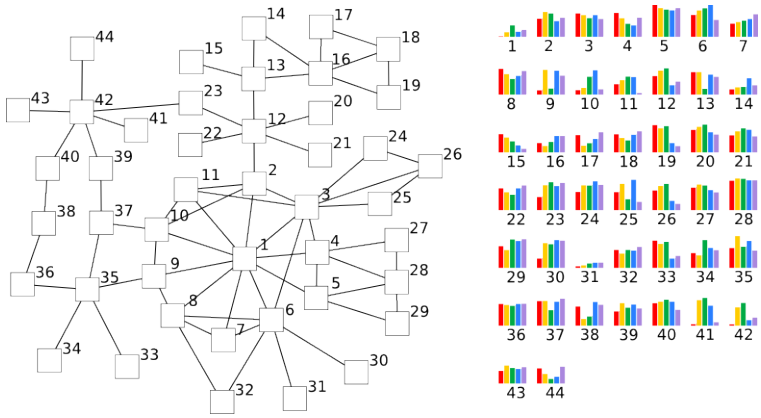
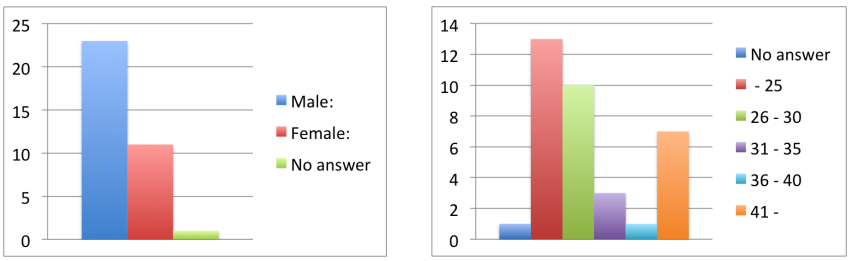


Figure 4.2: Showing a dense graph named DE1 visualized by using a multiple coordinated views approach.

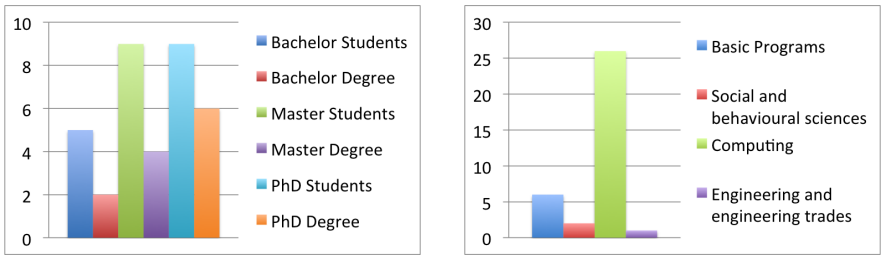


(a) Gender distribution

(b) Age distribution

Figure 4.3: Subjects demographic data.

Hypothesis 1 The accuracy of the subjects in identifying the required nodes should be higher for the integrated approaches in dense graphs in order for the first hypothesis to be true in terms of effectiveness. However, no significant difference was found for this statement ($X^2 = .25$, N=271, $p > .05$). Therefore, the time required to complete the tasks could be used directly to evaluate the efficiency of the two approaches. The tasks performed on graphs using integrated approaches were carried out significantly faster (M = 3.97, SD = .31) compared to the tasks performed on graphs using multiple coordinated view approach (M = 4.11, SD = .31), $t(269) = 3.64$, $p < .05$. Therefore, Hypothesis 1 is true in terms of efficiency, but not in terms of effectiveness, i.e., there is no difference in effectiveness between the approaches. But integrated approaches allow a more efficient identification of network structures that hold interesting data. Therefore, integrated approaches should be considered for designing time-critical visualization systems.



(a) Degree distribution (b) Education major distribution

Figure 4.4: Subjects education data.

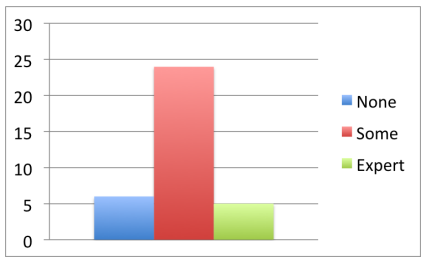


Figure 4.5: Knowledge about graph/network visualization.

Hypothesis 2 The results of the analysis for validating the second hypothesis were similar to the ones presented above. No significance was found between the approaches for the effectiveness in sparse graphs ($X^2 = 2.06$, N=276, $p > .05$). Here too, the use of integrated approaches improved the efficiency significantly (M = 3.85, SD = .29) in contrast to using multiple coordinated views (M = 4.0, SD = .32) , $t(281) = 4.11$, $p < .05$. This means that Hypothesis 2 holds in terms of effectiveness, however in terms of efficiency it does not hold, as integrated approaches were more efficient even with sparse graphs. Figure 4.6 shows the mean time subjects spent to finish the tasks using both approaches for both types of graphs. It can be noted that the difference in time increased with dense graphs (DE). Further studies should be performed to verify if this trend will continue. The conclusion here is that integrated approaches are more efficient for the given tasks, regardless of the size of the underlying graphs.

Hypothesis 3 The effectiveness of both approaches for dense graphs in terms of giving insight into the topology was tested by analyzing the results of the third task. There was no significant variation between approaches in terms of identifying the graph topology features, $X^2(2)= 3.89$, N=140, $p > .05$. The time it took participants to identify the topological features of the multivariate networks visualized using multiple coordinated views approaches (M=3.66, SD=.38) was not significantly

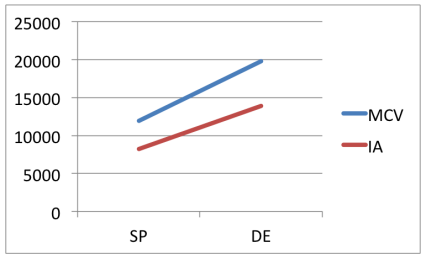


Figure 4.6: Median times (in milliseconds) participants spent on performing the tasks one and two for both types of sparse (SP) and dense (DE) graphs using integrated approaches (IA) and multiple coordinated views (MCV).

faster than in integrated approaches (M=3.69, SD=.31), t(134)=-.48, p=.63. These results suggest that our third hypothesis is false on both accounts: effectiveness and efficiency.

Hypothesis 4 The results were again similar to the previous hypothesis. Sparse graphs show no significance between approaches in terms of identifying the graph topology features, $X^2(2)= .78$, N=141, $p > .05$. This means hypothesis holds true in context of the effectiveness. Again, no significant difference was found in terms of time required to identify topological features between integrated (M=3.64, SD=.4) and multiple coordinated views approaches (M=3.56, SD=.3), t(135)=-1.26, p=.2. This confirms Hypothesis 4 for efficiency as well. Similar results were shown for dense graphs, as explained above, meaning that there is no significant difference for giving insight into the graph topology between the two approaches for any graph, for the cases we have tested. The last task for each shown graph was to grade how well the graph topology is visible. The average subject gradings are shown in Figure 4.7. It corresponds to the presented analysis, as there is no clear favorite approach. It can be noticed that the grading drops with dense graphs.

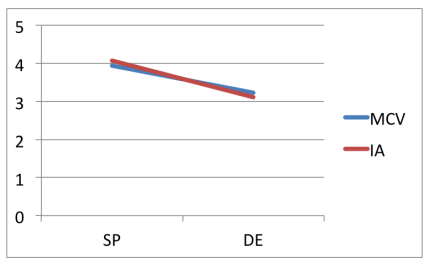


Figure 4.7: Average participants grading of the readability of the topology for both types of sparse (SP) and dense (DE) graphs using integrated approaches (IA) and multiple coordinated views (MCV).

4.4 Summary

A task-based usability study of efficiency and effectiveness of two different approaches for visualizing multivariate networks is presented in this chapter. The hypotheses and methodology was described in detail, followed by the results of the study that confirmed one of the hypotheses in context of the efficiency. Our study showed that integrated approaches are more efficient for identifying parts of the network holding interesting multivariate data. However, the findings did not show any significant difference between the approaches in terms of graph readability for our sample data. These results might serve as guidelines when designing new systems for multivariate network analysis. In the following chapters, contributions for concrete challenges for multivariate network visualization are presented.

Chapter 5

Network Visualization Using Magic Lenses

Several approaches that deal with the problem of visualizing multivariate networks were discussed in Chapter 3. Strengths and weaknesses of each approach were discussed. One of the main challenges for integrated approaches is that they tend to introduce clutter affected by the growing size of the elements (nodes or edges) where the multivariate data have been integrated. This issue can be addressed by using different focus+context techniques among others. In this chapter, the *Network Lens* will be presented, an extension of the traditional magic lens idea (cp. Section 2.2) applied to traditional node-link graph layouts. Our prototype implementation of this Network Lens enables users to interactively build various lenses by specifying different attributes and selecting different visual representations [24]. Each time the Network Lens is applied on a network element: it visualizes attributes (or a subset of them) by using small glyphs, i.e, the standard node representation is replaced by a new visualization or diagram. Let us consider one of the standard problems in biochemical network analysis: time-depended attributes. For example, time plot visualizations could be used to represent experimental data measured over time. As the data belongs to specific nodes, these plots could be attached to them. Furthermore, let us assume that a biologist wants to analyze such data at time step t_i. Then, without having a need to change the current visualization setting, a specific Network Lens instance could be used to show the data at time step t_{i-1} for a set of nodes. Thus, the visual analysis process of multivariate networks is supported by an additional generic tool that can be adapted to standard visualization systems. In this way, our approach extends already existing views to show node and possibly edge attributes of the underlying network. Lenses for specific exploration processes could be created. Additionally, it is possible to store them for later analyzes and to combine them to analyze related attribute sets.

Part of the work in this chapter was previously published [63] The **aim** of this chapter is to provide contributions towards fulfilling the Criterion 2.1 described in Section 1.3 by introducing novel interaction and visualization techniques for multivariate networks that tackle existing challenges of the integrated approaches. The rest of this chapter is structured as follows: the visualization and interaction techniques behind the idea of Network Lenses are presented in Section 5.1.

Aspects of the development of the prototype are briefly described in Section 5.2. Two different application scenarios using data from different science domains are presented in Section 5.3. Finally, Section 5.4 summarizes the work presented in this chapter.

5.1 The Network Lens

In this section, a tool that supports the interactive analyzes of complex networks using visual filtering is presented. The Network Lens preserves the overall network topology and context, and facilitates the visualization of network attributes at the same time. Our approach is independent from the graph layout and from different drawing conventions used to visualize the network. Users have the freedom to investigate the network by exploring the overall visualization of the network, in terms of topology and connectivity of particular nodes of the network. Users can get more details about *desired* attributes by focusing on specific node(s) with the help of the Network Lens. These desired attributes can be chosen from a set of all available attributes the user is interested in. Afterwards, the users can interactively explore the network elements in context of the selected attributes. It is also possible to represent the remaining attributes inside the nodes if such a feature is desired, forming a typical integrated approach technique for multivariate network visualization as discussed in Section 3.2. Such approaches have the drawback of space usage, and in consequence they may introduce clutter. In the following, the way how our approach copes with such problems is discussed. Our software prototype is presented as well including a typical use case scenario.

Approach

As briefly described in Section 2.2, the main inspiration for our approach comes from the work of Bier *et al.* [11, 123] who created lenses based on pure graphical filters. In contrast, our approach is driven by attribute semantics even though graphical filters are used too. The basic idea of magic lenses was extended in such a way that users can interactively build various lenses by selecting desired attributes and assigned visual representations in context of the network visualization. A specific number of different quantitative, ordinal and nominal attributes belonging to every network element might be important to be visualized. Each attribute or group of attributes can be represented more or less efficiently by using different visualization approaches depending on their data type. Visual comparison of attribute values could be enhanced by choosing the appropriate approach [40]. Therefore, it is important to have the option to choose the way different attribute types are visualized. Another important feature would be to allow users to combine different lenses by using drag and drop interaction in order to simplify and speed up the process of creating new lenses. Additionally, one should be able to set up and store a num-

ber of lenses for each working session as well as for later use. In consequence, *custom-built* lenses can be created, stored and reloaded for exploration of multivariate network visualization. Users can then switch between these lenses interactively.

Our prototype's GUI is divided into three parts, as shown in Figure 5.1. On the left hand side, a traditional node-link network visualization is placed. This is also the main area of the tool; it occupies the major part of the tool window. After loading the input network (using a GraphML specification [15]), an overview of the entire graph topology is displayed. Ideally, nodes could be drawn in various ways. However, nodes are only represented as rectangles at the time of writing this thesis, as such aesthetic issues were not in the focus of our research. The user can also map the value of an arbitrary attribute to the color saturation of the nodes or decide to attach a small attribute visualization in each node. If specified in the data set, the thickness of the edges will represent the value of an edge weight. Such weights usually represent the strength of the relationship between two node entities. Users can choose five different graph layout algorithms and/or modify the position of the nodes manually. All these features help to identify interesting patterns in the network. They also enable the user to rearrange the nodes by manual clustering (automatic clustering could be easily added), after which the multivariate networks can be analyzed in more details.

An active Network Lens named *Natural Sciences* is displayed in the center of the network visualization, see the screenshot in Figure 5.1. It shows a small Parallel Coordinate diagram with four quantitative attributes belonging to the corresponding node. Additionally, four nominal attributes are displayed as node labels. In the current situation, the lens covers a single node. However it is possible to access other nodes by moving the lens with the mouse or by translating the graph behind it. The size of the lens can be changed by simple mouse resizing actions similar to standard windows based GUIs. Additionally, the lenses support zoom functions which magnifies node representations without distorting them in order to avoid misinterpreting attribute values due to misshaped glyphs. In contrast, the edges are distorted to help following them as the magnification may introduce a "cut effect" similar to the way a straw might look cut in a glass of water. In some cases, this distortion cannot compensate enough and the edge flow insight could be lost or distorted too much. This side-effect can be avoided either by slightly moving the lens or by following the edge lines along the lens rim. The legend of the lens' attribute color mapping is placed on the right panel (called "Lens Mapping"). The bottom part of the tool window hosts all the lenses created and used during a working session.

In order to start the lens creation process, the lens has to be named at first. For remembering the "meaning" of the lens, the users are advised to use self-descriptive names for lenses in case a lot of them are created and stored. Let us assume our user is analyzing a data set of student relationships in context of their grades in various school subjects (see Figure 5.1). Different students might perform better or worse towards certain types of subjects, and exploring their relationships in this

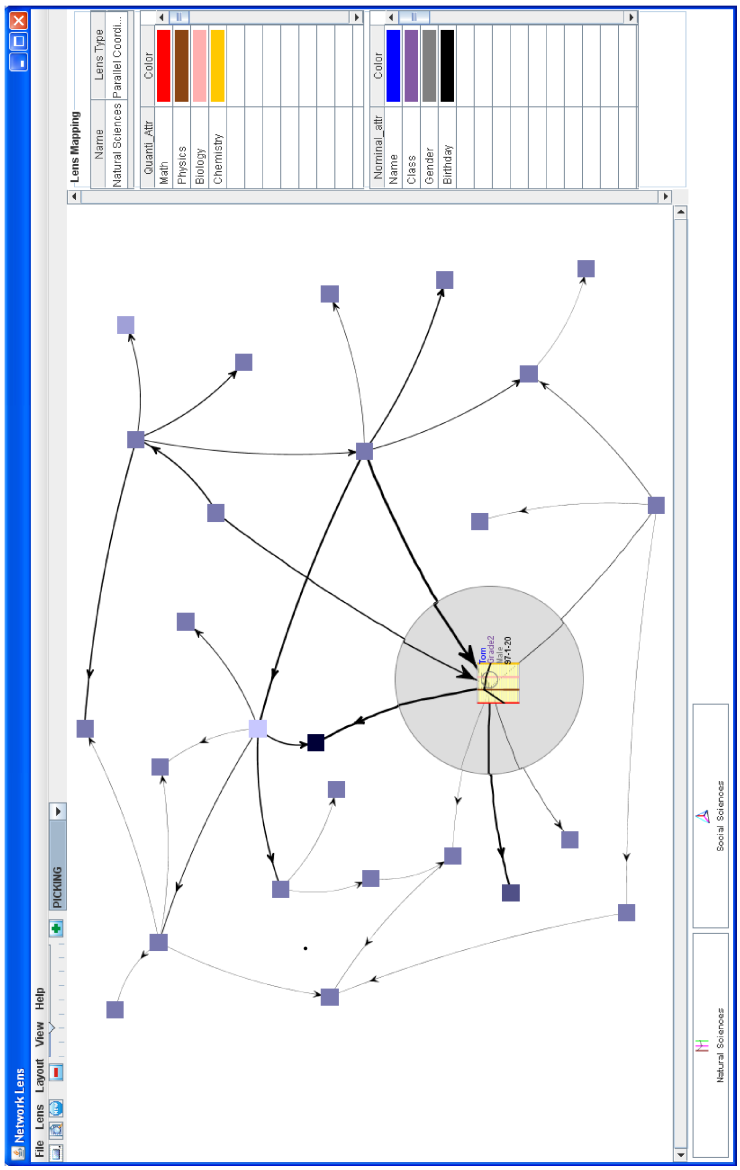


Figure 5.1: Overview of the Network Lens tool (rotated by 90°). The GUI is divided into three distinctive parts: the main network visualization area, the lens information area on the right hand side, which we call *Lens Mapping*, and the bottom part where all user-produced lenses are preserved. The multivariate network data is based on students (⇒ nodes) who share the same courses (⇒ edges) and their individual course grades and personal information (⇒ attributes). © 2010 IEEE.

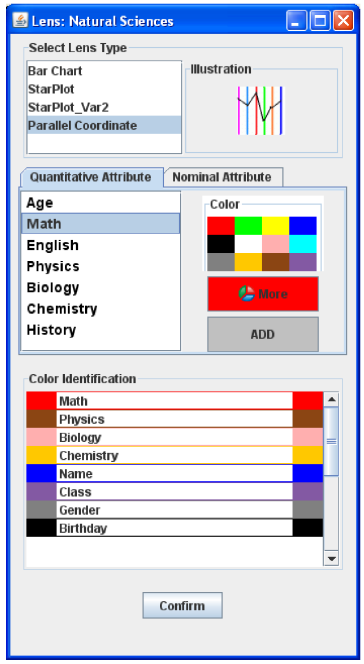
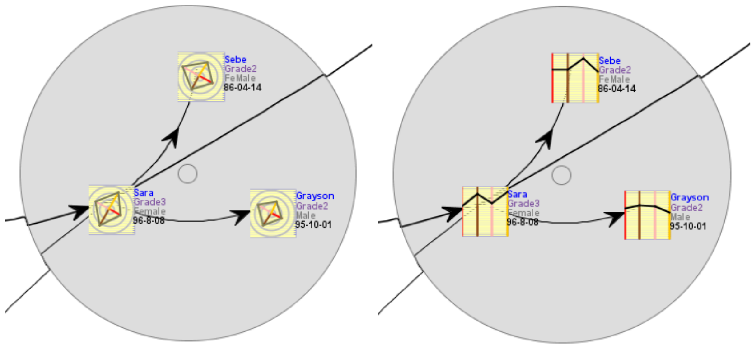


Figure 5.2: A dialog box used to create and edit lenses. Users can specify the type of the lens, select different attributes provided by the input data set and assign their color mapping. © 2010 IEEE.

context is important to our user. At this point, the user can create different lenses for certain groups of classes and name them accordingly. For instance, he/she might want to create a lens that shows the values for attributes (subjects), such as Painting, Sculpture, etc., and name it *Art Lens* and/or create a *Science Lens* with subjects like Mathematics or Physics. As shown in this example, lenses are created by following some sort of attribute semantics.

The next step is to name the lens after which the visual representation must be specified. The users need to select the desired attributes to be visualized and specify their color mapping by using the form presented in Figure 5.2. They can assign a suitable visual representation of a lens concerning quantitative and ordinal data by selecting one of the options in the "Select Lens Type" list. Our prototype currently supports two variations of star plots, one bar chart, and one parallel coordinate visualization, see Figure 5.3. A sample view of the selected lens type is shown in the "Illustration" icon at the top of the dialog box (cf. Figure 5.2). Two different tabs ("Quantitative Attribute" and "Nominal Attribute") are used to specify quantitative and/or nominal attributes to be visualized by the lens. The attributes are added to



(a) Star plot diagram, 2nd variant (b) Parallel coordinate representation

Figure 5.3: Two different visual representations used to display the attributes. Attributes can be color coded automatically, or the color can be specified by the user. Nominal attributes, such as *Name*, are represented by text labels on the right hand side of the glyphs. © 2010 IEEE.

the lens by selecting them from the list and clicking the "ADD" button. The system assigns the color automatically, if it is not specified by the user. To achieve a good visual perception, a default color palette comprised of 12 standard colors was created as suggested by C. Ware [137]. The lower part of the dialog box holds a separate list showing the final color mapping. By repeating the steps described above, the user can create several new lenses which are added in form of buttons to the bottom part of the GUI. One can then switch between lenses by clicking the desired lens button which activates the corresponding lens.

When the lens is moved or the view is panned (while the lens remains at the same place in the view), all the nodes that are covered by the lens change their original node representations to the ones specified during the creation of the lens as described above. A transparent background is set by default for all the text labels used to represent nominal attributes. This can be hard to perceive if the underlying graph has many edges because of potential overlaps. The user is able to switch to an opaque background for labels in order to avoid this problem, which in turn could lead to not very appealing lens experiences in an aesthetic sense. As discussed in Section 2.2, some edge routing techniques could be used to overcome this problem. However, the EdgeLens [139] approach seems more appropriate in our case as it could be integrated seamlessly with our Network Lens.

The procedure for creating lenses was presented so far. This process can be simplified and extended after at least two lenses were specified. In the following, the process of combining different lenses to create new ones is discussed.

We were inspired by optics and the previous work on magic lenses which enables a combinations of lenses that have different transformation functions regard-

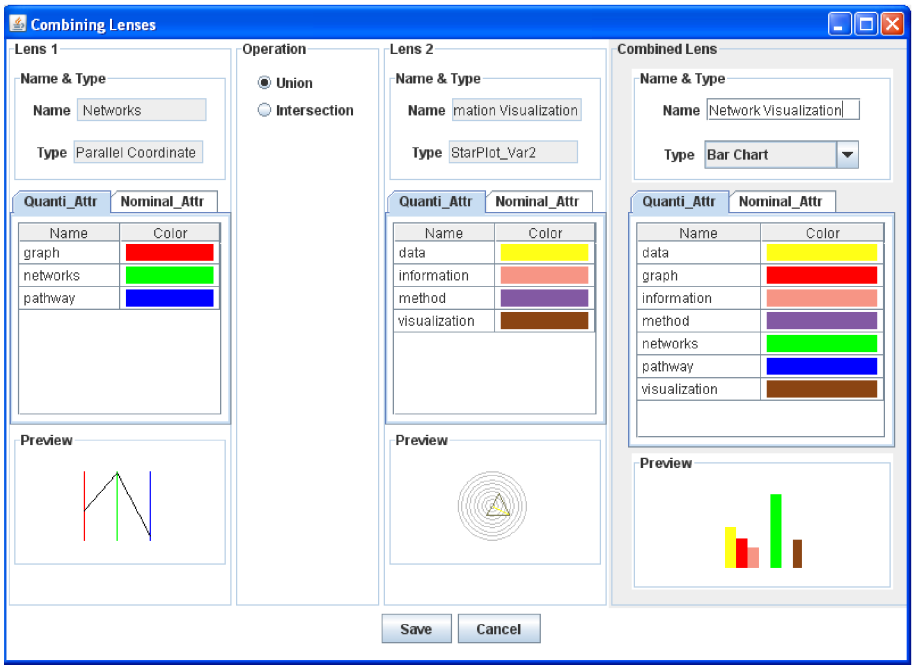


Figure 5.4: A dialog box used to combine different lenses. © 2010 IEEE.

ing the graphical objects. Accordingly, our approach enables us to combine already created lenses by *laying them one over another*. However, in our case, a different issue is introduced as described in the following. Let us assume that a user wants to combine several lenses with different visualization metaphors for the same set of attributes. Their combination will not be as straightforward as it could look in traditional, graphics-oriented magic lens approaches. But, if our user wants to combine lenses that have (partially) different sets of attributes, he/she can simply drag and drop one lens button over another one, similarly to real life optics or magic lens approaches. However, an intermediate step is additionally required where the user can decide the final outcome of the combined lens. Therefore, a new dialog box "Combining Lenses" appears as shown in Figure 5.4. This intermediate step is necessary for specifying several aspects of this lens combination. In case of the optical or standard magic lens combinations, graphical functions could be combined in a pipeline where the output of one lens is the input of the other. This cannot be done in our case; a combination of visual metaphors is hard and mostly impossible. Additionally, certain lenses might be created with different attributes making things even more complicated.

The controls of the "Combining Lenses" dialog box have been placed in four groups. Information about the lenses that are going to be combined are shown in

the control groups “Lens 1” and “Lens 2”. These control groups hold information such as the lens name and type, the list of attached attributes including their color coding, as well as a preview icon. Different set operations are used to create a new set of attributes from the given sets in “Lens 1” and “Lens 2”. The user can choose between two basic set operations: *Union* and *Intersection* in our current version of Network Lens. The “Operation” group holds two radio buttons to specify the desired set operation. The “Combined Lens” group shows the result of the chosen operation. In this way, the users can specify the desired attributes in a quick and simple way. It has the same GUI layout as “Lens 1” and “Lens 2” and is similar to the lens creation dialog box. At this point, the users should specify a name for the combined lens. If both input lenses use the same visualization type, then the default type of the combined lens corresponds to the input type. The type of the lens and attribute colors of the combined lens can be changed by the user. After saving the combined lens, a new lens button will appear again at the lower part of the main window, cp. Figure 5.1.

5.2 Technical Aspects

The JUNG graph drawing library [94] was used to implement the prototype of Network Lens. This library allows the developer to use a set of predefined node and edge visualizations as well as implement own visual representations. It offers a lot of other functions that are already implemented besides different graph layout algorithms, such as zooming and panning interactions.

JUNG also contains the implementation of a lens that has the functionality of a normal magnifying and fisheye lens. Having an already implemented distortion-based lens has eased our implementation process. Based on the existing implementation, we added the functionality of changing the shape of the nodes at each time the lens is moving over them. By using this feature, our work was focused on the development of a set of visual metaphors that are applied to network nodes each time a JUNG lens is over them, thus creating our Network Lens.

Our prototype uses GraphML files as input data [15], since GraphML has its own extension mechanism which allows to attach $<data>$-labels with different data types. They are used to store the required attributes for nodes and/or edges in a graph specification (currently, our tool only supports the visualization of node attributes). However, the GraphML reader provided by JUNG proved not flexible enough for our tasks. So, we had to implement our own parser due to the need to parse different GraphML files having different attributes, especially for future needs.

5.3 Application Scenarios

Biological Data

The main idea behind this prototype is that it is applicable to any type of multivariate networks regardless of the application domain. A hand made copy of the biological network data presented in the work of Borisjuk *et al.* [13] is used to demonstrate our approach in context of biological data. Figure 3.4 shows how this data is visualized by using their own tool. The figure represents experimental data interpreted together with a metabolic network: glycolysis and citric acid cycle. Each bar inside of the nodes represents relative substance levels of different *Vicia narbonesis* (beans) lines. Wild types of beans are shown in dark-grey. Light-grey bars represent the lines where the transgenic technology was used to increase protein accumulation as beans are an economically important protein source in food industry. The visualization of this data as shown in Figure 3.4 helps biologists to see the effect of transgenic technology on the plant metabolism.

Our tool can mimic the standard integrated approaches and can handle this type of biological data, cp. Figure 5.5. As discussed in Section 4.3, integrated approaches are efficient for identifying structures of a network holding interesting multivariate attributes. Therefore, a Network Lens–visualizing another set of attributes–could be quickly applied to such structures for further visual inspection. However, integrated approaches do not scale well spatially when the number of network elements increases and/or in case of high attribute numbers (cp. Section 3.2). In such cases, focus+context techniques like our Network Lens could alleviate the problem as there will be no need to increase the size of the nodes. Figure 5.6 shows the lens applied to a specific part of the network. While a user may investigate different parts of the network, the rest of the nodes can show different glyphs as desired.

Text Documents

In the following, analysis of another multivariate network is illustrated based on a set of research papers (24 text documents) published by our group. The documents are represented by the nodes. Each node has several attributes that correspond to the occurrences of a specific word within the document. A *similarity* between two nodes is shown by an edge connecting the corresponding nodes. This similarity is specified by the co-occurrence of attribute sets of the considered documents. The degree of the similarity is calculated as the sum of the minimum values of the co-occurred words. This degree is represented as the weight of the edge (shown as the thickness of the edge line). Frequently used words (such as stop words) were filtered out and were not included as node attributes, as they would not reveal any insight into the content of the documents. Additionally, we pruned the data by setting a threshold for the minimum occurrence of words and for minimum edge weights. A screenshot of our tool visualizing this input data set is shown in Figure 5.7(a).

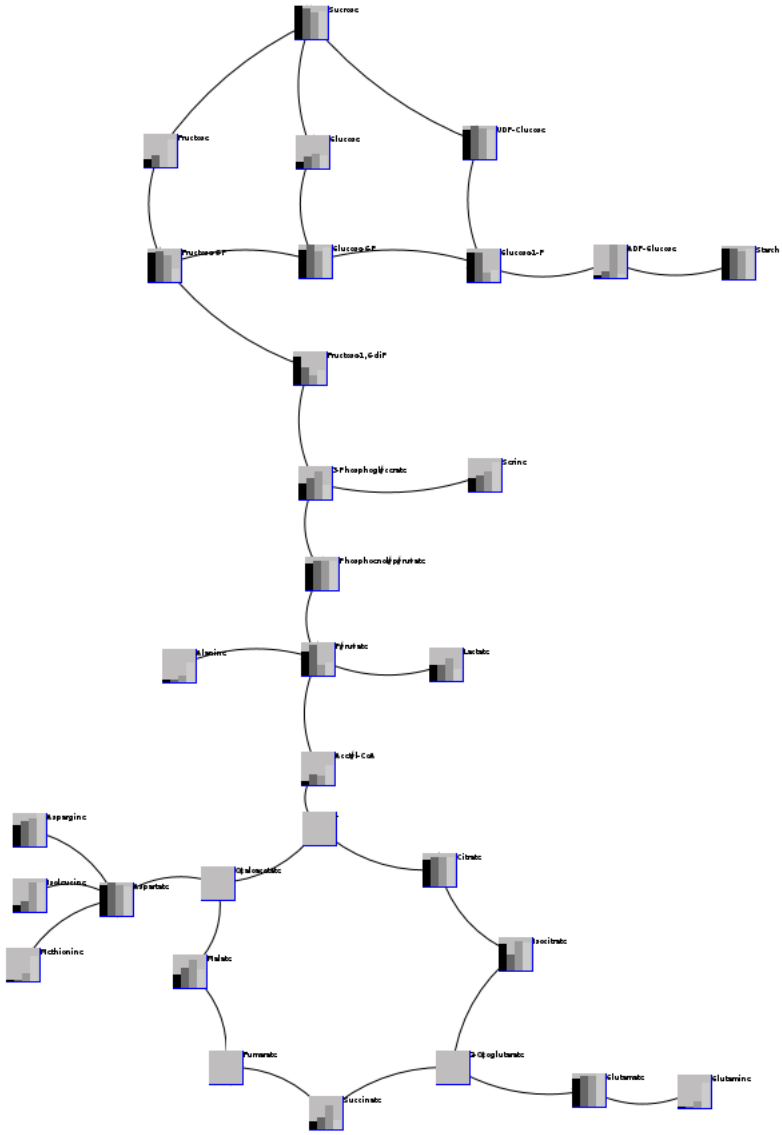


Figure 5.5: Visualization of experimental data integrated into a biological network. Based on the data shown in Figure 3.4.

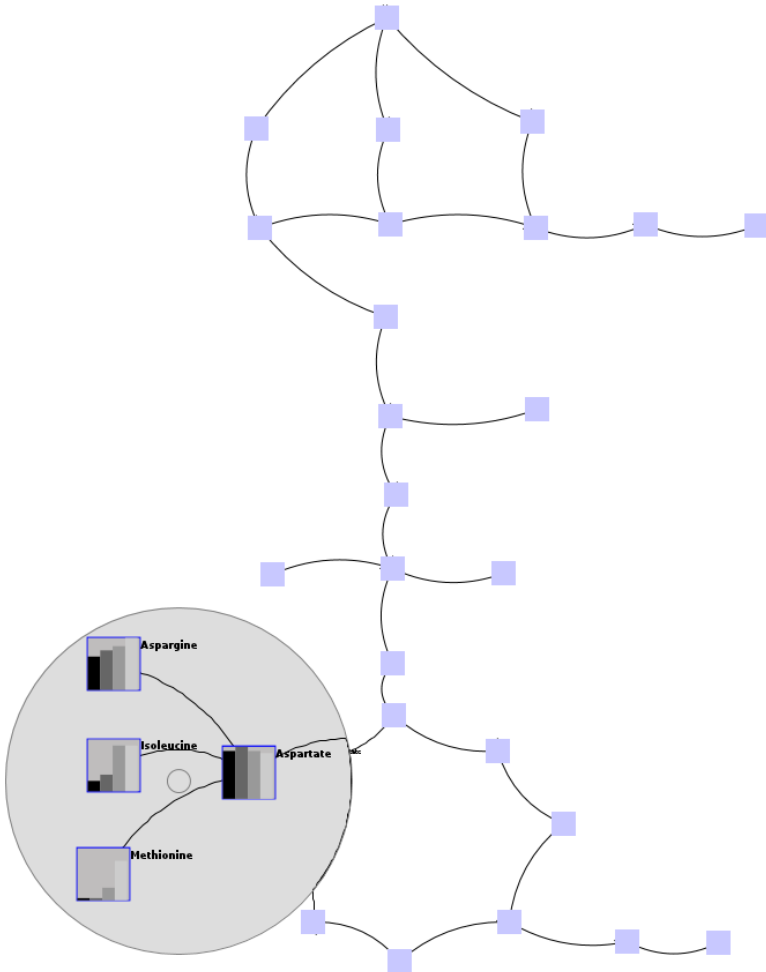


Figure 5.6: Visualization of experimental data by using the Network Lens. Based on the data shown in Figure 3.4.

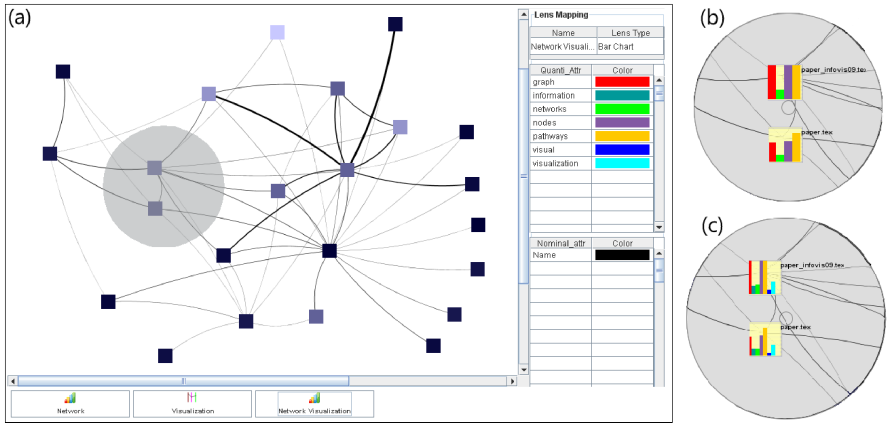


Figure 5.7: The transparent gray circular disk in the network visualization view (a) represents the focus of the lenses discussed in the following. The top right image (b) represents the view of the lens named "Network", while the bottom right image (c) shows the view rendered by the "Network Visualization" lens. © 2010 IEEE.

Let us assume that our user wants to explore these documents without reading them. He/she is interested to find documents that are related to the topic of *algorithms*. Therefore, the user maps the color saturation of the nodes to the value of the attribute *algorithm*. The lighter colors of the nodes represent higher values of the attribute, namely higher occurrences of this word in a document, as shown in our screenshot example. The user can immediately identify several documents with high value, but other documents could discuss *algorithms* in different contexts. Therefore, the exploration is narrowed to those documents with content related to *algorithms* and *networks* (or graphs). At this point, the user creates a new lens and selects the attributes (keywords in this case) such as *networks*, *graphs*, *nodes*, etc., that would give insight to documents related to networks. He/she names this lens "Network" and starts the exploration. The user focuses the lens on the lighter colored nodes (those with content about algorithms). Two documents with relatively high values of specific attributes are discovered, see Figure 5.7(b). A high frequency of the words *graph*, *nodes*, and *pathways* is found in these two documents. This could mean that the documents describe some computational algorithms related to biochemical pathways and might not be related directly to visualization. To verify this, the user loads a previously stored lens named "Visualization". This lens is combined with the current lens ("Network") using the "Union" set operator in order to create a new lens "Network Visualization". The user continues the exploration of those documents again, as he/she created a tool (the new lens) to get more insight about the documents connected to network visualization and algorithms. The new lens reveals that a couple of documents identified earlier fit the criteria since words *visual* and *visualization* are mentioned in those documents as shown in Figure 5.7(c).

5.4 Summary

In this chapter, a novel approach for interactive visualization of multivariate networks was presented. It supports the exploration of such networks by using intuitive visual filtering methods for the local representation of node attributes. Integrated approaches face the issues of readability while multiple coordinated view approaches have to deal with display sizes. Our approach offers a solution while minimizing these side effects. More precisely, our Network Lens combines the advantages of magic lens approaches and integrated graph drawing and reduces the overload issue of the latter technique. Our system offers freedom in terms of attribute filtering and selection of effective visual representations. This makes the approach easy to generalize for different application domains dealing with multivariate networks. Domain experts can use their knowledge and expertise to craft their visual filters in order to gain insight into their relational data sets. The ability to combine the lenses in an intuitive way simplifies the process of creation of new lenses for further analysis of multivariate network data. In the following chapter, a hybrid tool that implements new visualization techniques using the attribute-driven topology approach in combination with multiple coordinated views and integrated approaches is presented.

Chapter 6

A Hybrid Approach for Network Visualization

Challenges and problems of multivariate network visualizations have been mentioned several times by this point. Often users want to understand their data sets, i.e., they need to get an overview about the network structure and how different data values relate to this structure. Answers to the following questions could provide a solution to the problem stated above or even help us in creating new meaningful questions: Is the topology of the underlying network related to the values of particular attributes? Or, are specific network objects similar to each other in context of their attribute values?

As discussed in Section 3.2, hybrid approaches are designed to use the advantages of several approaches. With that in mind, a hybrid tool that uses an attribute-driven topology in conjunction with integrated approaches and multiple coordinated views was developed and is presented in this chapter. Although this tool represents a hybrid approach, its main feature is a new visualization technique based on the attribute-driven topology approaches. It extends traditional force-directed layout algorithms by including the attributes in the process of layering the network. One of the drawbacks of attribute-driven topology approaches is that they often skew the network structure. Our approach presents an improvement in this regard. It relies on user interaction to find a good balance between showing the topology versus the attributes. Additional coordinated views provide support for our main visualization.

The remainder of this chapter highlights our visualization method which is called JauntyNets. Part of the work in this chapter is accepted for publication [67] . The **aim** of this chapter is to provide contributions towards fulfilling the Criterion 2.1 described in Section 1.3 by introducing novel interaction and visualization techniques for multivariate networks that solve existing issues of the attribute-based topology approaches. In Section 6.1, the creation of sample data sets that are used for testing the prototype is described. Afterwards, our extension of the traditional force-based approaches is presented in Section 6.2 together with the most important visualization and interaction techniques. A discussion of the used clustering methods and presentation of additional coordinated views completes the functional descriptions of our approach. We finalize the chapter with the most important implementation aspects and a summary.

6.1 Sample Data

A bunch of scientific papers and articles written by our research group was used to create our multivariate data set D1. Similarly, for creating the data set D2 a larger collection of papers from two different conferences was used. The nodes of the network represent the documents themselves in both of these data sets. These two data sets are created in the same way as the one presented in Section 5.3 (cp."Text Documents"). Each document owns a number of attributes that are calculated based on the occurrences of specific words within the document. Stop words, such as "a" or "the" were not included. The similarity between documents is represented as an undirected edge between the corresponding nodes. It is calculated based on the co-occurrence of the attribute set among the documents. The degree of similarity is calculated as the sum of the minimum values of the attributes and is encoded by the weight of the corresponding edge. An arbitrary threshold for the minimum occurrence of words and for minimum edge weights was used to filter the data.

Additionally, our approach was also applied to the so-called Jigsaw data set D3 containing metadata for every IEEE InfoVis and VAST conference paper from 1995 to 2011 [57]. Here, a special social network was created. The edges between nodes represent co-authorship, i.e, if two papers share an author, then their node representations are connected with an edge. The number of shared authors is shown by the edge weight. Specific metadata named *Concept terms* were identified in titles and abstracts of the papers. They are used as attributes by our visualization tool. The Jigsaw file at hand has the information about the conference of the published papers. This was used as an attribute as well. It is important to know that the individual attributes only have two possible values for D3: one and zero.

6.2 JauntyNets

As mentioned in the introduction of this chapter, the presented visualization tool could be described as a hybrid approach that combines multiple coordinated views and integrated approaches with attribute-driven layout. A number of interactions were developed to aid the exploration of the networks. Users can gain more insight into the data set by changing different parameters of our visualization tool. Multiple coordinated views provide more information about the data or visualize outcomes of data mining techniques such as multidimensional scaling; they complement the analyzes process [80]. Clustering is one additional data mining technique used to facilitate the visual analysis of multivariate networks. In the following, our attribute-driven layout technique is described in detail.

Extending Traditional Force-based Approaches

The main contribution presented in this chapter is an extension of force-based approaches in such a way, that it produces an attribute-driven topology. The approach

is the integral part of our JauntyNets tool used for the interactive visual analysis of multivariate networks. Initially, the input graph G is placed in the center of the main view (cp. Figure 6.1). In contrast to the traditional force-based layout algorithms, the natural edge length l_{uv} is inversely proportional to the edge weight. The reason behind this decision is based on the core notion of the network edge: links between nodes represent their relation, and weights show the strength of such relations. Therefore, the larger the weight, the closer the nodes are positioned together, and as a consequence the relational strength is perceived better. In our concrete case, more similar documents are placed closely together. One of the characteristics of our approach is that attributes A_i are technically considered as graph vertices with special features. These vertices are called *attribute nodes* V_a in the remainder of this chapter. Algorithm 1 shows how the attributes A_i are *transformed* into attribute nodes V_a and appended to the underlying graph G. They are laid out on a circle around the actual network to be visualized. A slider can be used to interactively specify the circle radius. Attribute nodes can be moved angularly only by user interaction within a predefined radius.

A traditional force-based layout is of incremental nature, i.e., it moves the nodes to new positions with each iteration depending on the forces applied on the nodes (cp. Section 2.1) [28]. These forces are not applied on attribute nodes. Additionally, a gravitational force ($F_{gravity(v)}$) is added only when a normal node v crosses the circle perimeter in order to prevent it from leaving the circle due to the repulsion forces between the nodes (this is a so-called *position constraint*). This force is an adaptation of the repulsion forces between two nodes u and v as presented in

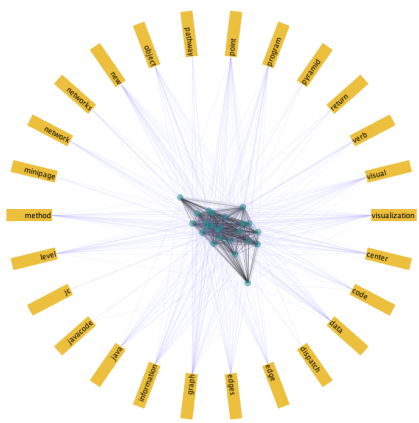


Figure 6.1: This screenshot shows the initial visualization after the data D1 has been loaded. The *orange* blocks placed on a circle represent the attributes of the network. *Green* nodes in the middle represent the text documents.

Algorithm 1 Building Network Structure

Input: network with attributes $N = (G, A)$ where $G = (V, E)$;
Output: extend N so that $V \leftarrow A$

```
    // Add attributes to vertices and mark them as attribute vertices.
 1: for all a in A do
    // Create attribute vertex v_a ∈ V_a such that V_a ⊆ V
 2:   v_a ← a;
 3:   V.add(v_a);

    // Create edges from attribute vertices to graph vertices and compute edge
    weights based on attribute values.
 4:   for all v in N do
 5:     if v.attribute(v_a).hasValue() then
 6:       e = new Edge(v, v_a);
 7:       e.weight(v.attValue(v_a)); // Assign attribute value to the edge weight.
 8:       E.add(e);
 9:     end if
10:   end for
11: end for
```

Equation 2.1. In our case, the forces are applied between a node v and the center of the circle c. In the following equation, r represents the circle radius, while d_{cv} holds the distance value between node v and the center of the circle c:

$$f_{gravity(v)} = \begin{cases} \sum_{(v,c)\in V} \frac{grav_{cv}}{d_{cv}^2} \hat{x}_{cv}, & \text{if } d_{cv} > r \\ 0, & \text{otherwise} \end{cases} \tag{6.1}$$

In contrast to traditional layout, the argument specifying the strength of the electrical repulsion forces rep_{vc} needed to be adapted. Therefore, this argument is given a negative value $grav_{cv} = -rep_{cv}$. Thus, a part of the Equation 2.1 is re-used and adapted to create the gravitational forces $F_{gravity(v)}$. Our extension of the original work of of Eades [28] is shown in detail in Algorithm 2. As previously mentioned, our algorithm does not affect the positioning of the attribute nodes V_a. Therefore, the first modification of the algorithm happens in Line 4 by excluding attribute nodes from the loop so that no forces are applied on them. Another modification ensures that they do not move away as a result of interactions with other nodes. The gravity $F_{gravity(v)}$ force, which pulls the nodes that tend to cross the circle radius, is applied in Lines 6-8.

Edges that connect nodes to attribute nodes are called *node-to-attribute edges*. These edges can be filtered out if necessary, thus that only edges with weights over a user-defined threshold are visible. Users are also able to interactively specify several other filtering thresholds by using the corresponding sliders, such as the stiffness

Algorithm 2 Layout Algorithm

Input: network N such that $V_a \subseteq V$ // cp. Algorithm 1.
Output: vertex placement $p = (p_v)_{v \in V}$

```
// Perform initial placement of vertices.
placeAttributeNodesOnCircle(V_a);
// Place vertices randomly inside the circle.
placeNodesInsideCircle(V − V_a);

// Run force-based layout algorithm. Do not apply forces to attribute nodes positions.
loop
    for all v in (V − V_a) do
        F_v ← F_rep(v) + F_spring(v); // cp. Eq. 2.1
        // If a node is pushed over the circle radius, apply force towards circle center.
        if p_v.isOutsideRadius() then
            F_v ← F_v + F_gravity(v);
        end if
    end for
    for all v in (V − V_a) do
        p_v ← p_v + ε · F_v;
    end for
end loop
```

sti_{uv} of normal edges and node-to-attribute edges. The stiffness control slider for node-to-attribute edges is important as it affects how much the node positioning depends on the attribute values. It is named "Enforce ABL" (Attribute Based Layout), because the higher the value of this parameter, the more the network layout is affected by attribute values. As the name might reveal, the "Enforce graph structure" slider is responsible for enforcing the network topology by controlling the stiffness of the normal (node-to-node) edges. By tweaking the values of these sliders, users can choose to preserve graph topology, gain insight into attribute values, or try to balance both.

The initial view after the sample network D1 has been loaded to the system is shown in Figure 6.1. D1 contains information about 24 research papers in total written by members of our research group. Therefore, the network is highly connected since all papers have similarities. At the current state, the user will only get an overview of the presented network and its relation with specific attributes. For further analysis of the data, the user needs to perform a number of interactions which are described in the following.

Visualization and Interaction

Edges can be filtered out by using a slider if their weight is below a certain threshold as already described in Subsection 6.1. The thickness of the edge corresponds to the edge weight value. Attribute nodes can be dragged and positioned only around the specified circle at a defined diameter. Their movement will relocate the inner nodes as well depending on attribute nodes' locations and the normal nodes' attribute values (connectivity between nodes with attribute nodes). In case some attributes are considered as unnecessary for the analysis process, it is possible to disable them by first selecting the attribute node(s) and pressing the key "A". Furthermore, it is possible to create groups of the semantically related attributes, i.e., they can be subsumed under a superordinate concept. Nodes that correlate with a group of attributes will then move towards that group. However, the positioning will also be affected by the edges between the nodes themselves. Instead of moving the desired attribute nodes individually, it is also possible to grab an entire group and move it around the circle, making the relocation of attributes much easier.

After the initial loading of the D1 data set (cp. Figure 6.1), similar interaction steps as described above have been performed resulting in an image as shown in Figure 6.2. One of the first steps was to filter out the edges of the graph that show weak relations among different documents. To enforce the attribute-driven layout, the stiffness parameter sti_{uv} for node-to-attribute edges was increased by using a slider. This affected the positioning of the nodes that have higher values of particular attributes by moving them towards those attributes. As seen in our screenshot example, three attribute groups have been created: *visualization*, *networks*, and *code*. The users need to select one or multiple attributes with a right-click action and press "G" on the keyboard to create attribute groups. To add more attributes to a desired group, it is necessary to move the attribute inside the group region and press "G" again. The name of the first attribute added to the group is automatically used to label the created group. User can change the label through a context menu later. After creating all the groups, they should be distributed evenly around the circle. In our concrete sample data set in Figure 6.2, we see that the node positions with respect to the groups correspond to the major topics described in the input documents. Thus, insight is gained into the content of these documents. Afterwards, the users might want to continue the explorations of individual documents, as explained in the following.

Nodes are highlighted and a tooltip with the node label is shown after the user places the mouse cursor over the node. This feature is illustrated in the center of Figure 6.2. So far, only the attribute based topology portion of JauntyNets has been described. Beside more interaction features, the following paragraphs describe how our tool implements the integrated approaches concept. Users can get insight into the concrete attribute values of the selected node as their corresponding attribute nodes act as bar charts: here we integrate an attribute visualization into the same view as the underlying network. The direct neighbors of the node are highlighted

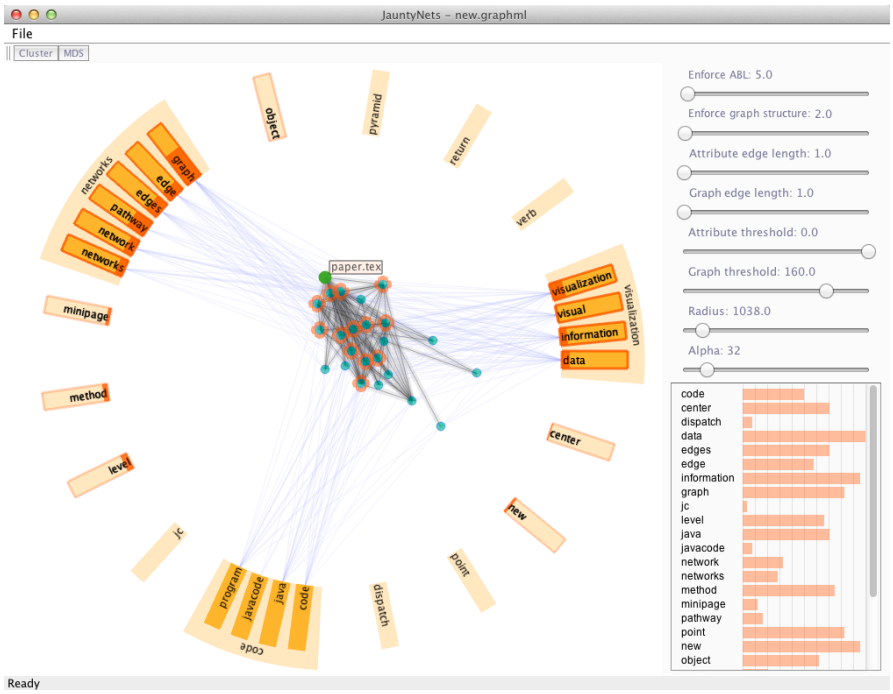


Figure 6.2: The graph view on the left shows three attribute groups (based on data set D1). The green node has been highlighted after a mouse-over action, and a tooltip displays the node label. The first-level neighbors of the selected node are shown with an orange halo. Attribute nodes act as bar charts for the highlighted node, giving insight into the values it has for each attribute. The faded attribute glyphs were disabled by the user. © 2013 IEEE.

as well. Analogous interaction is implemented for attribute nodes too, i.e., all nodes that have a node-to-attribute edge to the selected attribute will be highlighted. This action will reveal all the documents with the value over the specified threshold for the corresponding attribute (keyword in this case). Several other visualization parameters or features could be interactively manipulated, for example, the radius of the attribute circle or the transparency value of the attribute-to-node edges, as well as to zoom-in/-out or to pan the view.

The visualization of the Jigsaw data set D3 is shown in Figure 6.3. JauntyNets offers a possibility to filter out the undesired attributes by deselecting them from the list in the preloading dialog. A number of concept terms have been filtered out by using this feature. Terms related to the field of "geovisualization" have been selected for loading and as such have been grouped together, as seen in the lower part

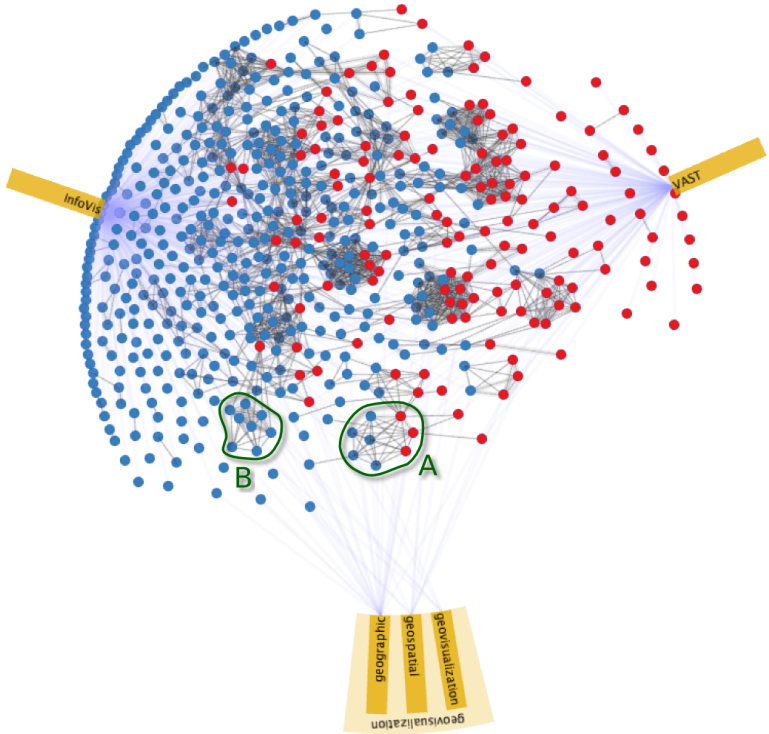


Figure 6.3: The screenshot displays papers from InfoVis and VAST conferences (data set D3). Two cliques marked with A and B are interesting as they are related to the created attribute groups. © 2013 IEEE.

of the screenshot. The two conferences where the papers have been published are the remaining attributes that were not deselected. InfoVis papers are shown by blue nodes, while VAST papers are represented by red nodes. At this point, users can play with the sliders. They might want to enforce the attribute values in which case the nodes will be moved towards attributes for which they have the highest values. They might later choose to enforce the graph topology to see how it affects the visualization. In the example in Figure 6.3, neither topology nor attribute values have a higher coefficient, i.e., the stiffness parameters are balanced. It provides an overview of the paper distribution between the two conferences. Interesting structural features can be noticed immediately, such as a lot of unconnected nodes or subgraphs. Other noticeable structures are a number of cliques. Information about the productivity of a person or group of persons are discovered by such structures. This means that one specific author has written all those papers, or there is a group of authors who

often cooperate. As we were interested in the papers related to geovisualization, one such clique that was placed relatively close to our "geovisualization" group caught our eye (Figure 6.3 marked by "A"). Therefore, further investigation ensued. One common author (MacEachren) was identified after going through the papers from that clique. Only one of his papers was not related to "geovisualization" as none of the attributes of the "geovisualization" group is connected to it. This was found out by using a mouse-over function either on the attributes or the papers as described earlier. Even though the paper content did not correspond to our defined group, it was moved in the group's direction as a result of its connection to other papers related to the group. Another interesting fact about this author is that he published articles in both conferences. Our investigation moved to another clique marked by label "B" with papers published only at the InfoVis conference. Upon further investigation, more than one common authors were identified. Edges that have less than two authors were filtered out, and the clique still remained, although two nodes have been separated. This showed that this clique is strong, namely these authors (Dykes, Slingsby and Wood) have worked closely together, and published their work at the InfoVis conference (Figure 6.4). However, half of their papers are not related to "geovisualization". In contrast to the author of clique "A", the authors of clique "B" are not mainly focused on one domain of information visualization. However, they really seem to prefer the InfoVis conference. This exploratory approach helps us to discover if certain parts of the network structure are related to particular network attributes. Two network structures (cliques) were identified as closely related to the predefined group of attributes.

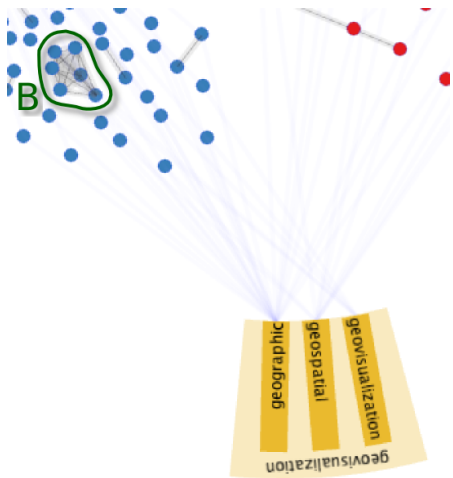


Figure 6.4: A cut-out of the data represented in Figure 6.3. It shows that clique B is still highly connected even after filtering out all edges with weight 1. © 2013 IEEE.

There are different approaches of attribute-driven layouts for multivariate networks (cp. Section 3.2). However, the network layout is drastically affected by most of such approaches. Preserving the graph topology or interfering less with the traditional force-based layout algorithms is important for identifying different network structures. For more comprehensive analyzes of the data, this approach alone would not produce sufficient analysis results. Therefore, to enable a more in depth exploration of the data, several other (standard) views and techniques are added. The particularities of these views and techniques are discussed next.

Clustering

Attribute-based clustering has been implemented to further support the exploration process with JauntyNets. The results of the k-means clustering algorithm [91] performed on the larger network data set (D2) consisting of 421 nodes and 55 attributes is shown in Figure 6.5. Different articles published in the VisWeek[1] and INTERACT[2] conference proceedings from 2009 to 2010 constitute this data set. The user can invoke the "Cluster" option from the toolbar menu that prompts a dialog box where the desired number of clusters is set, after which the clustering process is triggered. Attribute values are then used to calculate the clustering of the nodes (documents). An external library was utilized to perform the clustering (cp. Subsection 6.3). It offers a possibility to choose between several clustering algorithms. Figure 6.5 shows a screenshot where three clusters have been created, hence the color of the nodes corresponds to the cluster they belong to. ColorBrewer's categorical color map [16] has been used for color coding the clusters. Note that those attribute groups (i.e., "interaction" and "mobile") comprising HCI-related keywords attract mainly nodes from the *blue* cluster, which is primarily consisting of nodes (papers) from the INTERACT conference that focuses on HCI.

Automatic attribute group creation is possible through the use of clustering as well. Instead of clustering the nodes based on the attribute values, the opposite is done in this case, i.e., the attributes are clustered based on the node values for a particular attribute. The creation of attribute groups is quicker in this way. However, the results are not optimal. In consequence users might want to shuffle the attributes between the groups to achieve a better grouping. Applying the clustering algorithm in such fashions could also give insight into how different attributes relate to each other for specific data sets.

Further Coordinated Views

In order to provide additional support the process of visual exploration of the multivariate network data, our tool has been extended with a couple of coordinated views

[1] http://visweek.org/
[2] http://www.interact2013.org

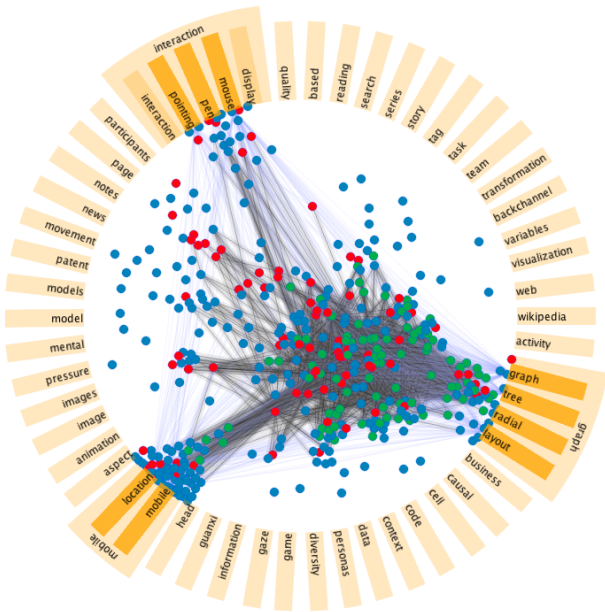


Figure 6.5: The screenshot displays a network of 421 nodes with 55 attributes (data set D2). The nodes are colored differently, because they have been clustered based on their attribute values. © 2013 IEEE.

showing specific data related information. The first additional view shows the number of nodes that a certain attribute has an edge to. This view uses a simple bar chart metaphor as shown in Figure 6.2, down to the right. The view is coordinated with the rest of the tool. If an attribute node is selected in the main visualization view, the corresponding bar will be highlighted. Similarly, the values in the view will be updated in case users change the threshold for node-to-attribute edges. The purpose of the view is to spot outliers, for example attributes with a large number of connections regardless of users' filtering efforts. Such attributes might be too general to be considered useful for exploration with our main approach and could be regarded as "noise" as the majority of nodes would be driven towards it. Users can then disable those attributes to achieve a better positional result as done with "display" in the interaction group of Figure 6.5, for instance.

A multidimensional scaling (MDS) visualization is shown in another view. Here, all nodes' attribute values were used for a projection of the multidimensional data space into the two-dimensional plane. Similarity between nodes is represented as distance, i.e., the closer two nodes are, the more similar their content is. The network data from Figure 6.5 was used to create an MDS visualization shown in Figure 6.6. This view is also coordinated with the rest of JauntyNets' views. It shows similar

interaction possibilities in terms of highlighting, zooming and panning. A selected node in one view will be highlighted in the other view as well enabling the user to track the position of the nodes in all views.

It is interesting to notice that the spatial positioning of the MDS is complemented by the clustering results. A dense area of the nodes in the upper-left corner that belongs to the blue cluster can be noticed. These particular nodes (articles) mainly belong to the INTERACT proceedings. This finding is similar to the one presented in Figure 6.5 as described earlier. Therefore, questions on the similarity of the network objects in context to their attribute values can be answered with the help of MDS view.

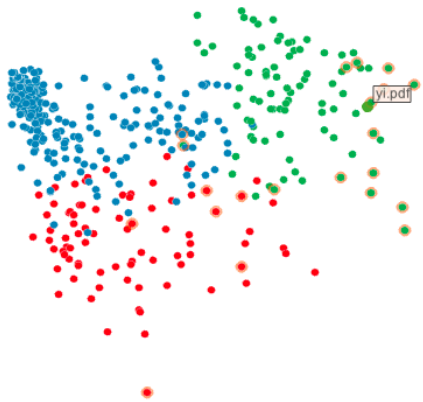


Figure 6.6: MDS view (data set D2). The highlighted node shows a tooltip with the node label "yi.pdf", and the first-level neighbors are displayed with an orange halo. The node colors reflect the results of a previous clustering step. © 2013 IEEE.

6.3 Technical Aspects

The Java programming language was used for implementation of JauntyNets. Additionally, the Processing [101] graphic library that provides method calls to OpenGL was used for implementing various visualization and interaction aspects of our tool. The MDSJ library [90] is the basis for implementing our multidimensional scaling visualization. In order to perform the MDS, it is only required to compute the dissimilarity matrix as input. Clustering has been performed by using the trickl-cluster library that offers several clustering algorithms [130]. In our case, only k-means clustering was used. However, our tool could be easily extended to enable the use of all provided algorithms. All ColorBrewer [16] color maps are provided by the giCentre Utilities [39] libraries which we used for the color coding of our clustering.

The traditional force-based layout algorithm was extended by including (static) nodes as attributes (Algorithm 1), edge weights, and position constraints as presented in Algorithm 2.

6.4 Summary

An extension of force-directed approaches was presented in this chapter. It transforms the traditional force-directed layout into an attribute-driven layout. Various visualization and interaction techniques on which this approach relies are described in detail. The approach could be useful to show if certain parts of the network structure are related to specific network attributes. Clustering, multidimensional scaling and other coordinated views complement the approach to answer questions about the similarity of the network objects in context to their attribute values. JauntyNets represents a hybrid approach consisting of features from integrated approaches, multiple coordinated views and attribute-driven topology approaches. So far, contributions for solving the existing challenges of multivariate network visualization approaches have been presented in Chapters 5 and 6. In the following chapter, a method for visualizing complex data types derived from the multivariate attributes will be described.

Chapter 7

Visualization of Derived Network Attributes

The analysis of transcriptomics and metabolomics data created using high-throughput technologies is important in biology and medicine. Tools such as ontologies and hierarchical clustering are an integral part for studying such data. Statistically overrepresented ontology terms are identified by enriching the ontology terms in the data, giving an overview into important functional modules or biological processes. Hierarchical clustering is used as a standard method to analyze the data in order to find relatively homogeneous clusters of experimental data points. These two methods are usually considered separately, although they focus on the same data set. Therefore, a combined view is desired, namely visualizing a large data set in the context of an ontology under consideration of a clustering of the data. As explained in Subsection 2.3 (Ontologies and Clusterings), hierarchical clustering data is a product of multivariate experimental data. Therefore, visualizing such data may be considered as providing insight into the experimental data.

In this chapter, we present a new method for the task of combining the aforementioned views. Part of the work in this chapter was previously published [65, 64, 78]. Its **aim** is to provide contributions towards fulfilling the Criterion 2.2 described in Section 1.3 which states that a novel approach for visualizing derived cluster data for multivariate networks shall be introduced. The rest of the chapter is structured as follows: the properties of the input data are discussed in Section 7.1. We present our visualization approach of combining the Gene Ontology and cluster tree visualization in Section 7.2. Section 7.3 deals with technical concepts such as development methodology, implementation and scalability issues of the proposed method. Section 7.4 gives a detailed strategy on how to overcome one of the main disadvantages of the presented approach. Finally, we summarize our work in Section 7.5.

7.1 Sample Data

A transcriptomics data set representing different expression levels of genes in *E. coli* was used to demonstrate our approach. However, it is possible to visualize any data set that can be connected to an ontology and used for hierarchical clustering. The Gene Ontology (GO) forms a directed acyclic graph (DAG) as described in

Section 2.3 [7, 124]. Depending on the organism being examined, the number of leaf nodes in the DAG may vary, while there are more than 34,000 inner nodes. Only genes which are significantly up- or down-regulated were considered. This resulted in a reduction of the initial data set to 7,312 genes. Beside these genes, nodes which are on paths between the GO root node and leaf nodes (genes) were considered as well, resulting in the final GO data set made of 10,042 nodes and 24,155 edges. In more detail, the final outcome of the reduction is a DAG consisted of 1 root, 2,729 (non-terminal) nodes and 7,312 other nodes. Note that not all these nodes are leaves of the GO as some of them are unconnected to the rest of the DAG. This happens because not all the genes are assigned to GO terms and therefore do not form a part of the GO DAG. Multivariate data consisting of the expression levels of genes at different time points during the experimental procedure was used to perform the hierarchical clustering. The clustering has been computed based on the distance of these expression levels. A binary tree (called *Cluster Tree* in the following) with 14,623 nodes and 14,622 edges was produced after the cluster analysis of our data. It has 7,311 (non-terminal) nodes and 7,312 leaves (terminal nodes).

Showing both these graphs and their mapping is required to perform the analysis of the data. But, from a developer's point of view, these graphs appear independent from each other. They are two distinct types of graphs (a tree and a DAG) that have different nodes and edge IDs. However, they do "share" a specific part of nodes among each other since they have the same label for terminal nodes (genes). These two data sets can be mapped by using the information provided by the gene labels as presented in Figure 7.1 For additional exploration of the relations between the GO DAG and the Cluster Tree, a corresponding subtree of the Cluster Tree should be calculated and shown from any interactively selected single node in the GO (cp. Figure 7.8). More details about how this calculation is performed can be found in the "Architecture and Implementation" part of this chapter.

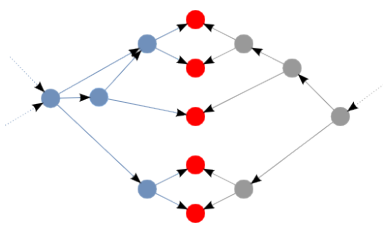


Figure 7.1: The *light-blue* part on the left represents a part of the GO DAG. The *grey* part on the right represents the Cluster Tree, while the *red nodes* in the middle are shared between both of them. Note that this diagram shows an idealized situation, because the common leaves do not need to be neighbored.

7.2 CluMa-GO

Our approach to solve the underlying problem will be discussed in this section. To demonstrate our approach a prototype tool called CluMa-GO (**Clu**ster **Ma**pping of **G**ene **O**ntology) [3] was implemented. GO DAG and the Cluster Tree are represented in two separated and coordinated views due to the complexity and huge amount of data to be visualized [105]. The data is loaded into our tool by using two separate .gml files [50] (one for the GO and one for the clustering) through a standard dialog box.

The complexity and challenges of visualizing huge networks on its own were already discussed in Chapter 2. Our current task is even more complicated as it entails a requirement to visualize and relate two huge data sets of different nature to each other: a DAG and a binary tree. Interaction techniques such as brushing [9, 46] are used to show the mapping between both. Problems such as clutter when visualizing the GO DAG and long or wide cluster trees (depending on the chosen tree drawing algorithm) would appear in case the graphs are drawn by using conventional graph drawing algorithms [42, 69]. One way to overcome the problem in the case of trees is to use scrolling and panning actions [132], because zooming out would not be sufficient in case of the Cluster Tree visualization: traditional tree drawing algorithms produce much unused space and this issue becomes worse with our binary tree as it is highly unbalanced.

In Section 7.1 the mapping of the specific subtree from a selected GO node was briefly introduced and more details about it are presented in the following. This mapping introduces another issue. The computed subtrees or sets of nodes are not sequentially mapped resulting in "gaps", see the red leaf node between the two yellow rectangles in the background of Figure 7.8. The issue here is that these gaps may be too big making the highlighted subtree branches appear too far apart from each other to be shown in a single view. This might be hard to perceive and some important information can be missed.

Different structural characteristics of the GO DAG and Cluster tree were taken into consideration when developing our visualization strategy. Therefore, specific visual representations for both are implemented accordingly. These representations address the aforementioned challenges separately, while interaction is employed to show the mapping between them. Initially, the approaches to visualize both the GO DAG and Cluster Tree are discussed. The supported interaction techniques are described later in order to distinguish between visual representations and interaction concepts. Figure 7.2 shows a complete overview of the GUI of our prototype implementation.

Gene Ontology (DAG) Visualization

As already described in Section 7.1, the selected GO DAG is relatively large even if a subset of the entire GO is used. Without some kind of filtering or aggregation, the

visualization of such a graph would not scale when standard node-link approaches are used. Our challenge was to show all data in one view. The approach presented here is inspired from pixel-based approaches which usually cope with large data sets [71, 72, 73, 74]: colored pixels are used to represent the GO nodes, whereas edges are hidden to avoid clutter. In the remainder of this chapter, these colored pixels are referred to as *node pixels*. Light-blue pixels represent non-terminal nodes, while leafs or unconnected nodes are represented by red pixels.

One of the main characteristics of DAGs is that they are directed and have no cycles. As such, they have a "flow direction" and can be hierarchically layered. Therefore in our approach, nodes are placed into several layers, in order to provide some insight into the topological structure of the GO graph as shown on the left hand side of Figure 7.2. Shneiderman and Aris [116, 6] presented semantic substrates, which are somewhat visually similar to our idea. In their approach, nodes are placed in regions (resembling our layers) based on specific node attributes, while in our approach they are placed in layers solely based on the graph topology. Small horizontal line segments are used to give cue to the spatial area of the particular layers. Additionally, the layers are numbered. Figure 7.2 on the left shows 17 layers marked from 0 to 16. There is a considerable number of unconnected nodes in our GO data set (cp. Section 7.1). A specific case where these unconnected nodes are placed in layer number 0 has been shown on the left of Figure 7.2. This visualization reveals that this level is the most dense layer. Two layering approaches were developed in order to provide more insight into the structure of the data. These approaches mainly differ in the way how leaves and unconnected nodes are positioned. These approaches are discussed in the following two paragraphs.

Levels Layout Depending on their graph-theoretic distance [12] from the source node (root), the leaves (red pixels) and non-terminal nodes (light-blue pixels) are placed into their corresponding layer. This layering approach is named *Levels Layout*. Moreover, non-terminal nodes are arranged on the right part of the layer, leaving the leaf nodes distributed on the left part of the layer. By doing so, additional insight into the topology of a specific layer is gained by acquiring information about the distribution of leaf nodes and non-terminal nodes on that particular layer. An example of this layout strategy is shown in the left hand side of Figure 7.2, namely in the GO view area of our tool. Figure 7.4(a) shows a conceptual diagram of the same layout strategy while Figure 7.3 shows it in the zoomed-in view, but showing only three levels at the same time. It might appear that the resulting visualization is similar to bar charts. However, the number of leaves and/or non-terminal nodes cannot be precisely compared between different layers, because the number of the leaves is proportional to the total number of the nodes in that particular layer, and not proportional to the sum of leaves in each layer. In other words, the density of each specific layer determines the covered area separately meaning that each layer shows the distribution or ratio of terminal and non-terminal nodes within that particular layer. Some form of normalization of area density could be introduced in order to achieve a true bar chart effect, in a future version of our tool. Leaves and

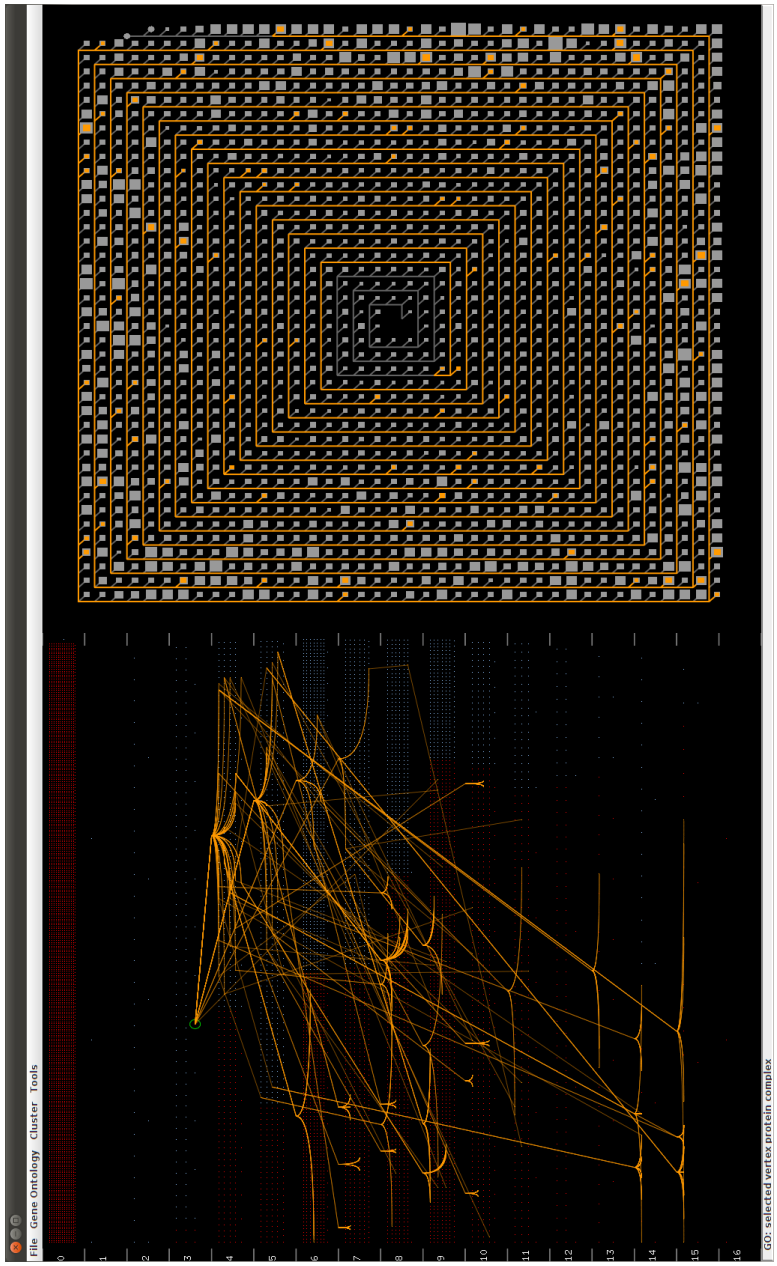


Figure 7.2: GUI of CluMa-GO (rotated by 90°). On the left hand side, the used Gene Ontology is represented in the GO view (Levels Layout). On the right hand side, the Cluster Tree view is located.

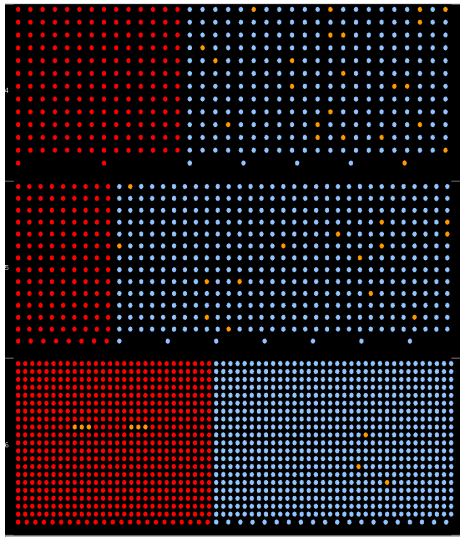
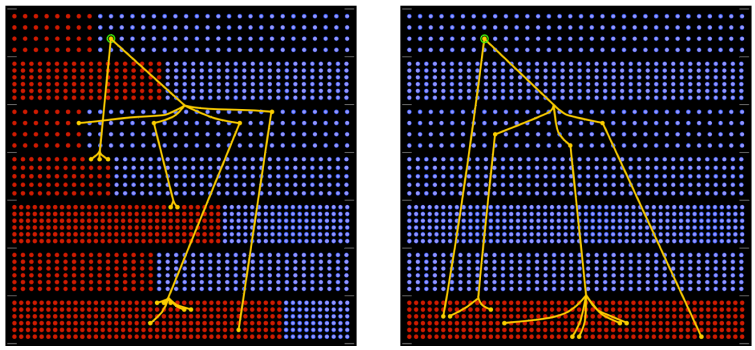


Figure 7.3: Zoomed-in view using the Levels Layout approach. The red nodes represent leaf nodes (e.g., genes); the light-blue nodes represent non-terminal nodes (e.g., terms). This view provides insight into the distribution of leaf nodes in a specific DAG level. The orange nodes represent the calculated subgraph (mapping).

terminal nodes are distributed in separate regions as described above. However, the placement of the pixel nodes inside these regions is arbitrary. Unconnected nodes are placed only in Level 0.

Bottom Layout Our second layering approach shares some similarities with our first approach: nodes are placed into layers depending on their graph-theoretic distance from the source node, and the placement of node pixels within the layers is done randomly. The main difference is that there are no separate regions within the layers as all leaves together with unconnected nodes are placed into one single layer with the highest number which would correspond to the bottom position in the GO view, hence the name *Bottom Layout* (Figure 7.4(b)). Unconnected nodes can be filtered out if necessary. A screenshot of the Bottom Layout with a particular node selected is shown in Figure 7.5. The advantage of this strategy is that it provides insight into the distribution of nodes among different layers without considering the leaves (genes). Additionally, upon selecting a specific node, users can find out more easily how it is connected to the leaves. Is this node connected with direct edges to the leaves? Are there many intermediate nodes and what is their distribution in the GO hierarchy? For instance, in Figure 7.5 there are a lot of intermediate nodes in level 4 that are mainly directly linked to the bottom level (leaves). This feature aids the perception of the graph topology of the DAG.



(a) Levels Layout (b) Bottom Layout

Figure 7.4: Two conceptual diagrams that show the fundamental ideas of both layering approaches.

The visualization of edges is omitted by default in order to avoid clutter. They are only shown in case the user selects a desired GO term (non-terminal node) for further exploration. Keep in mind that in such cases only the edges in the path from the selected node to the leaves are shown. Users could optionally show all the edges, but then the view will be overloaded. To reduce clutter even more, a simple edge bundling algorithm was implemented. The strategy is to bundle only the paths outgoing from a specific node that end up in the same layer. Figure 7.2 on the left shows the edge bundling of the computed subgraph in the GO view based on the Levels Layout approach, while Figure 7.5 shows the bundling applied on the Bottom Layout approach. Beside reducing the visual overload, edge bundles give insight into how different layers are accessed by a specific node. Placing DAG nodes in hierarchical layers ensures that the flow is from lower layers to higher ones, i.e., from top to bottom in our case, and no edge can exist between nodes in the same layer. Therefore, the use of arrows or any other visual cue to show edge direction is unnecessary.

A new challenge was introduced as a result of using a pixel-based approach for visualizing the GO. Choosing a good color scheme that would be optimized for monitors and print was an issue. Our tool provides options to choose different color settings for various elements of the visualization, such as color of the non-terminal and terminal node pixels, background, etc. All graphical elements can be easily distinguished and identified on a computer screen in CluMa-GOs default color scheme. However, finding a good working compromise for both the computer display and for printouts was needed. The ColorBrewer [16] was used to guide us for this task.

Figure 7.5: GO view with visible (bundled) edges based on the Bottom Layout. The green circle in layer 3 highlights the selected GO term.

Cluster Tree Visualization

Similar issues arose when dealing with the Cluster Tree visualization as they did with the GO representation. A traditional type of visualization would not scale for both data sets as they are relatively large. As mentioned earlier, hierarchical clustering of experimental data produces a large binary tree. If this tree is drawn with the help of conventional drawing algorithms, the outcome would be a rather high tree drawing, or a wide one, if we choose a standard dendrogram layout. This meant that a new visual representation for the Cluster Tree had to be developed. One special feature of our data set at hand is that the trees are particularly high and unbalanced, with shallow branches (subtrees). This particular characteristic of our data set emphasizes the disadvantages of traditional node-link layouts even further by consuming the space even more due to the unbalanced structure of the trees. However, what appears as a disadvantage at first could be used to our advantage when designing a special drawing algorithm for large trees of such nature.

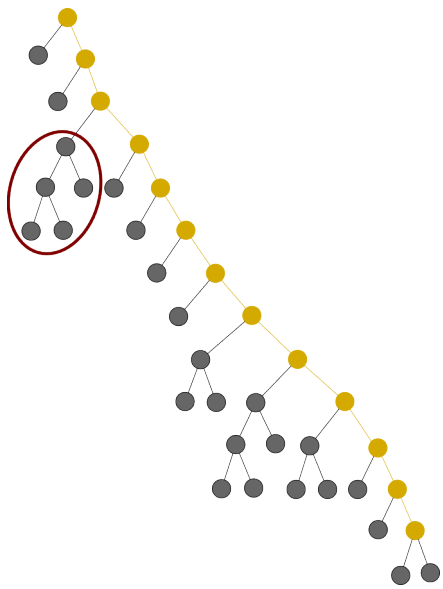


Figure 7.6: Sample cluster tree t. Yellow color represents the calculated backbone.

Figure 7.6 shows how a small part of such a tree might look like. Nodes and edges that form the longest path that connects all branches were used as a "backbone" (cp. the yellow colored part in Figure 7.6) This backbone is laid out as a spiral, thus preserving space and giving us a possibility to show the complete tree in one view. This is the main idea behind our space-filling *Spiral Tree Layout*, which was designed and implemented to deal with large unbalanced binary trees. The need

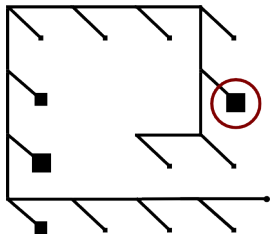


Figure 7.7: Spiral Tree Layout of t. The drawing algorithm was inspired by standard spiral layouts that are mostly used to represent time-series, such as [2, 129].

to perform repetitive scrolling to browse or navigate the elements in such large trees [59, 121] is not present in our approach. The spiral is arranged in a way that the closer the subtrees (see below) are to the center of the spiral the closer to the root they are, i.e., the direction of the flow in the spiral is counter-clockwise from the center towards out. For instance, if our *Spiral Tree Layout* layout is applied on the tree t shown in Figure 7.6, it would result in an image as shown in Figure 7.7.

A certain amount of abstraction is allowed in our visualization approach in order to cope with the size of the data set. The subtrees connected to the backbone are aggregated: each small box glyph in Figure 7.7 corresponds to one subtree branching out from the backbone with an angle of 135° from the vertical. The size of a box glyph is normalized and proportional to the number of nodes of the corresponding subtree. For instance, the size of the box glyph marked with the brown circle in Figure 7.7 is depending on the size of its corresponding subtree marked with the brown ellipse in Figure 7.6. The highlighted box in the spiral is proportionally enlarged by the drawing algorithm as the highlighted subtree with five nodes is one of the largest ones in the tree t. The space between the "spiral arms" of the backbone is constant and not influenced by the size of the subtrees in the current version of CluMa-GO. Therefore, we normalize the size of the box representing the subtree based on the maximum number of elements a particular subtree has.

The Cluster Tree view in Figure 7.2 shows that the largest branches appear far away from the root node of the tree. Other interesting patterns of distributions of subtree branches in the Cluster Tree can be identified by using the Spiral Tree Layout. For a further comprehensive analysis, details of each subgraph visualized in the spiral can be explored by clicking on a box glyph. This will display the subtree visualization widget (Figure 7.11 and 7.12) as described later in this section. The subtree can be drawn using two dendrogram layouts: a radial method and a so-called HV-drawing method.

Brushing techniques are used to show the mapping between the two parts, GO DAG and Cluster Tree respectively. In the following subsection, these and other interaction techniques are described.

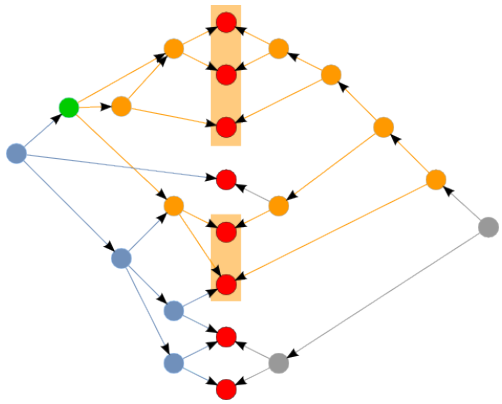


Figure 7.8: The *red nodes* in the middle are shared between both the GO DAG (*light-blue nodes*) and the Cluster Tree (*gray nodes*) (cp. Figure 7.1). The interactively selected node is highlighted in *green*, from which we traverse the graph (*orange nodes*) until we reach all accessible leaves (*red nodes with orange background*). The leaves are used to calculate a subtree of the Cluster Tree (*orange nodes* in the right part of the figure).

Interaction Techniques and Additional Views

Our collaborator from IPK Gatersleben, has explained that biologists explore the data in two ways: by browsing the data set randomly or having a specific GO term in mind. Accordingly, CluMa-GO features a list of terms that can be selected or searched through a dialog box invoked from the menu. Clicking directly on a particular node in the GO view is also possible. A tooltip displaying the name of the node is shown if a mouse-over action is performed on that node enabling the users to select a node for further exploration and to browse the GO. As already explained earlier in this chapter, the GO view displays the nodes as single pixels. Using color coding only makes it pretty hard to perceive a single, highlighted pixel. Therefore, double-coding is introduced by drawing an additional circle around the selected node in the GO view, as seen in the third layer of the GO view in Figure 7.5. This feature is also helpful in identifying the layer that the currently selected node belongs to.

The subgraph consisting of all reachable nodes will be calculated after the node has been selected, as described in Figure 7.8. All nodes belonging to the computed subgraph, will be highlighted in orange in the GO view. The edges of the subgraph will be shown too, using the same color as highlighted nodes. At the same time, reflecting the selection made in the GO view, the corresponding cluster subtree will be highlighted in the Cluster Tree view with the same color. In this way, the user can easily identify the mapping between both views by comparing the orange col-

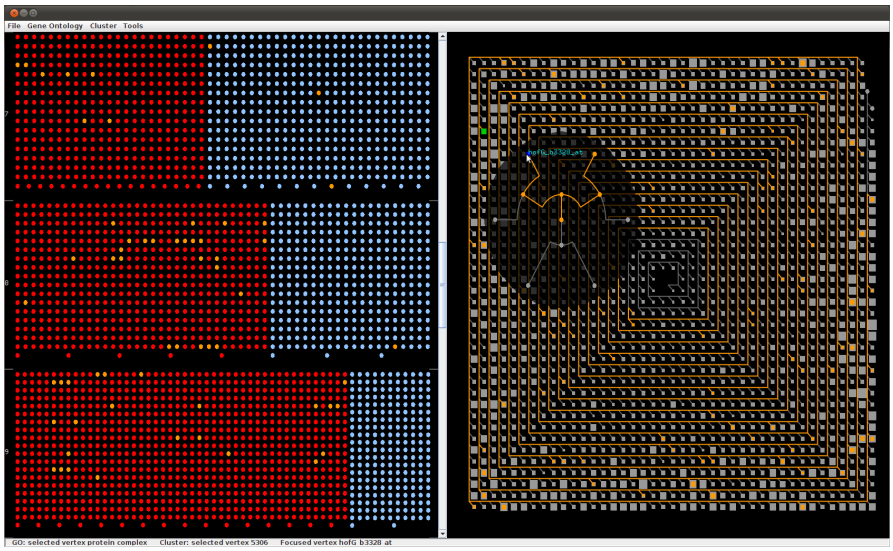


Figure 7.9: This screenshot shows the zoomed-in GO view (with the three layers 7 - 9) on the left hand side and the Cluster Tree view with opened subtree widget on the right hand side.

ored elements. Note that the higher the hierarchical level of the selected node is, the larger the number of nodes can be accessed from that particular node (the root node of the GO DAG, for instance, has access to all nodes of the DAG except the unconnected nodes). The complete DAG will be selected in case the root node pixel is clicked. However, this usually makes no sense for the analyzes purpose. Clutter cannot be avoided in such cases. If necessary, the visualization of edges can be disabled by the user.

Zooming at the specific layer on the GO view is also possible (Figure 7.9 and 7.3). The zoomed layer is usually placed in the center, between its neighbored layers. If the lowest or highest layers are selected for zooming, the three closest neighbors will be displayed, and the selected layer will be placed at the bottom or at the top correspondingly. The user can also scroll up or down between three layers simultaneously. Since a lot of edges from other layers might go through our zoomed layers, the edges are not shown in the zoomed-in view as they will introduce clutter. However, the nodes remain highlighted. It is easier to discover connections in zoomed-in mode than in zoomed-out mode due to the fixed amount of layers and magnified node pixels. As it is easier to select and interact with bigger node representations, this mode is particularly helpful for analyzing different elements of the subgraph.

Beside highlighting the GO subgraph when a specific GO term is selected in the GO view, the calculated subtree is highlighted in orange in the Cluster Tree view

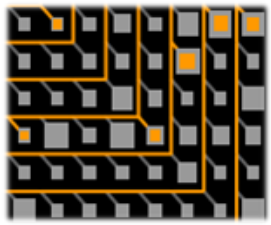


Figure 7.10: Cut-out of a mapping in the Cluster Tree view.

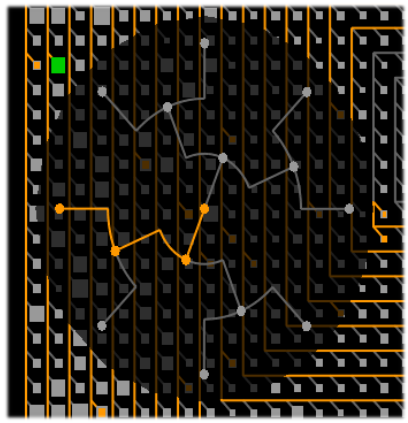


Figure 7.11: Subtree (branch) view. The more detailed view of the selected branch (*green box glyph*) is visualized as a dendrogram.

as well. This can be seen in the right part of Figure 7.2. In this case it can be noticed that this particular GO term has a rather wide cluster subtree, i.e., it covers most of the backbone of the cluster tree. Some subtree box glyphs are only partially highlighted due to the fact that not all nodes in a subtree might be mapped to the selected GO term, while others are not highlighted at all. The area of the highlight is proportional to the number of the nodes mapped in that corresponding subtree. Figure 7.10 displays a cut-out of a Cluster Tree view in order to provide a larger view.

To examine the subtree in detail, users can click on the corresponding subtree box glyph, after which a specific widget is displayed showing the particular subtree in one of two optional layouts that users can select based on their preference. The subtree can be viewed as a radial dendrogram (Figure 7.11) similar to other dendrogram visualizations [125, 108] or in an "explorer view" (Figure 7.12) based on an HV-drawing algorithm [23]. The default display position of the subtree widget is next to the selected subtree box glyph. In order to show the context of the area that

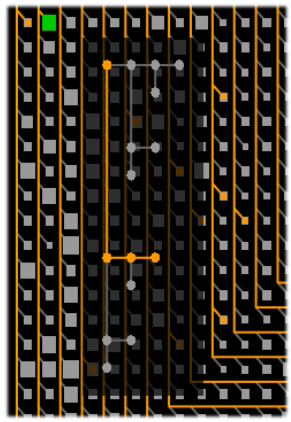


Figure 7.12: Subtree (branch) view. The more detailed view of the selected branch (*green box glyph*) is visualized by following a so-called HV-drawing algorithm.

it covers, slight user-defined transparency is introduced. The widget is movable, meaning if the area covered by the widget is important and interesting the user can move the widget and place it to a specific location. A mouse-over action shows the name of the particular node of the tree through a tool-tip similar to the GO view. Additionally, "reverse mapping" is possible as well by selecting one of the nodes in the widget. This is accomplished by parsing and highlighting the subtree from a selected node until the leaves (genes) are reached and continue parsing and highlighting the GO DAG until a common root in the DAG is reached.

Detailed Mapping View An additional view where the explicit mapping is shown on demand based on the idea presented in Figure 7.1 was implemented as a result of several discussions with domain experts and feedback from visualization experts. It implies the use of traditional graph drawing algorithms and it is called *Detailed Mapping* view. Showing the complete data set with such a view is not possible as described earlier in this chapter. Therefore, the Detailed Mapping view is used for representing only the highlighted subgraph and subtree. However, depending on the selected node the mapping outcome is usually a subgraph and a subtree of considerable size. As most of the selected GO terms produce subtrees with large backbones the issue is almost always present in the cluster subtree. In other words, long strings of backbone nodes are often created. The use of traditional tree drawing algorithms for showing all these nodes might introduce a lot of clutter. In the following a description of our approach to solve these issues is presented.

By selecting *amino acid catabolic process* from the GO view a specific mapping has been created as shown in Figure 7.13. Following the metaphor from Figure 7.8 the genes (red nodes) are placed in the center of both graphs showing the shared no-

des explicitly. However, here only nodes that are part of the mapping (highlighting) are shown. A simple layered based approach is used to draw the GO DAG subgraph. It is placed on the left hand side using a light-blue color. Light-grey nodes represent the cluster subtree drawn as a dendrogram on the right hand side of the screenshot. To correspond with the mapping in the main view, edges are highlighted in orange. However, a number of blue edges can be seen in this screenshot. They represent the long backbones as those nodes are not shown in order to avoid clutter. The number of nodes in a particular part of the backbone is proportional to the length of the corresponding blue edge. This feature gives insight into the number of nodes that have been hidden by the blue edges. This additional view is not shown by default and is activated on user's request. It can enforce the perception of the topology of both subgraphs at the expense of clutter, many edge crossings and increased edge lengths. However, it is an important view as it provides a more direct perspective into the mapping.

7.3 Technical Aspects

CluMa-GO was developed following an iterative process which involved discussions with domain experts and prototype development. Two initial requirements were implemented first. Domain experts were interested in a tool that is able to have a combined visualization of an ontology and hierarchical clustering of one data set in a compact view. They were also interested to search and browse this data. These initial requirements were implemented and new discussions with domain experts were made. These discussion resulted in the definition of further specific requirements that were implemented in CluMa-GO. Some requirements defined specific improvements of the presented methods, such as different representations of subtrees (already implemented by HV-drawings and radial dendrograms), zooming within the GO DAG (already implemented by the zoomed-in view). Other requirements involved creation of new visualization views. One such requirement is introducing ways to visualize a direct mapping between a terminal GO DAG node and a cluster tree leave (already implemented by Detailed Mapping View). One requirement is not implemented in the current state of our tool. However a considerable amount of work has been done for analyzing the problem at hand and some strategies to solve it were proposed. This requirement is to create a different representation of more balanced trees (cf. discussion in Section 7.4)

Architecture and Implementation

The Java programming language was used to develop CluMa-GO with Java OpenGL (JOGL) used to implement the visualization and interaction part of the tool. JOGL is a wrapper library that allows OpenGL to be used in Java [58] and is the reference implementation for Java Bindings to OpenGL (JSR-231). The Java Swing

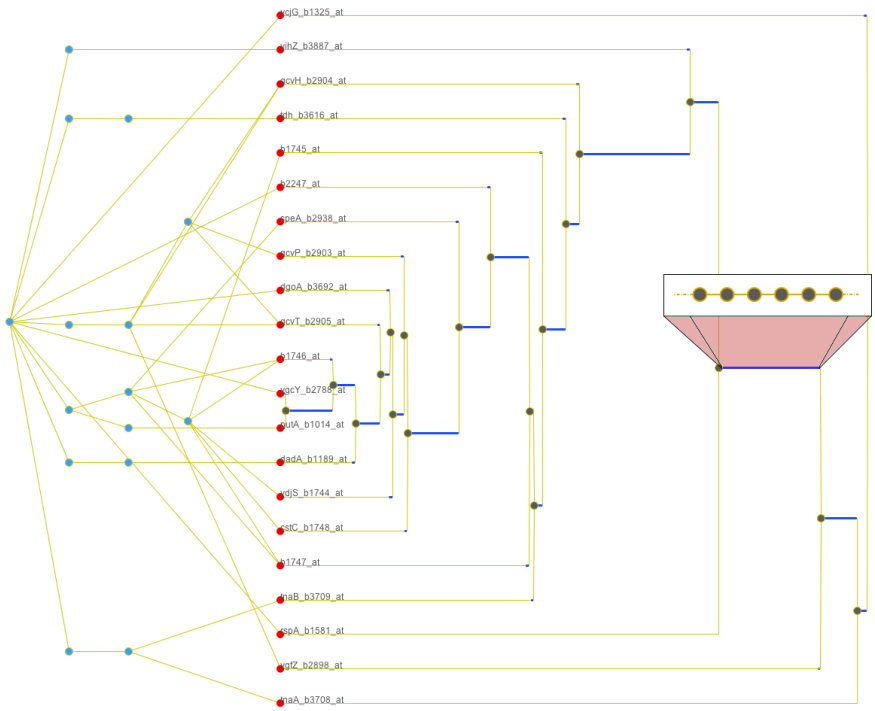


Figure 7.13: This screenshot shows the Detailed Mapping view. On the left hand side, the selected subgraph of the GO DAG is represented; the Cluster Tree is shown on the right hand side using a dendrogram layout. Both are connected with genes: the red nodes in the center. Some edges are thicker and blue. As seen in the cut-out of the screenshot, they represent a lot of backbone nodes which are hidden in order to avoid clutter.

API was used to build the GUI. It emulates the visual appearance of several computer platforms through a native look and feel support. JOGL is just a wrapper that uses corresponding native libraries depending on the platform. This means that builds for all popular platforms, such as Windows 32 and 64 bit versions, Mac OS X 10.6, or similar have to be made for our tool. Native libraries and Java libraries should be contained in every build.

Figure 7.14 presents and overview of the CluMa-GO's architecture. Several modules specialized for various tasks have been implemented. The IO module implements data loading from .gml files. An extended .gml file format is used to store the data. It contains additional properties for nodes, such as the node label. JUNG [94] was used for its various graph algorithms. Its graph model was extended by the Graph Core module in order to fit it to our implementation. The User Interaction module is used to realize the implementation of the Swing GUI and OpenGL user interactions. All the methods for the complete visualization process, including our own layout implementation, primitive drawing abstraction, and program state machine is contained in the Graph Visualization module and its submodules.

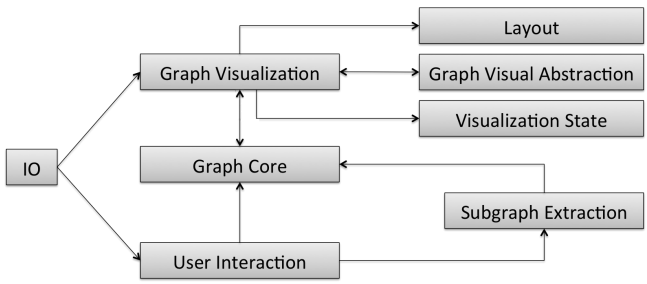


Figure 7.14: Module architecture of CluMa-GO.

Subgraph Extraction is one of the most important modules of CluMa-GO. It contains the implementation of the subgraph/-tree calculation algorithm, see Algorithm 1 for its pseudo-code. The algorithm uses two separate graph data structures as input: a GO graph and a cluster tree and a user-selected vertex within the GO graph. Only the leaves in both graphs have the same label, while every other vertex's label is unique. A GO subgraph and a cluster subtree are the algorithm's final outputs. A non-recursive depth-first search (DFS) approach starting from the user-selected vertex as a root is used to extract the GO subgraph. To continue with the mapping of the cluster tree all leaves of the freshly computed GO subgraph are parsed. At this point labels should be checked on both graphs as only the leaves of both graphs have identical labels. At the same time, all connected vertices from the current leaf up to the cluster tree root are stored in a list. Edges between the vertices are added next. After this process has been repeated for each leaf, a cluster subtree is produced containing a path from each leaf node in the subtree to the root of the cluster tree.

Next, a common subtree root needs to be found and the rest of the vertices (called *root chain* in the pseudo-code) are removed from the cluster tree root to the computed subtree root. Figure 7.8 shows an instantiation of the algorithm on a given small input example. The red leaf node between the two orange rectangles in the background of Figure 7.8 represents "gaps" in the mapping which frequently occur. Our tool and the source code are freely available in a SourceForge repository [118].

Algorithm 3 Subgraph Extraction

Input: *GO_graph*, *Cluster_tree*, and *selected_GO_vertex*
Output: *GO_subgraph* and *Cluster_subtree*

```
1: // extract subgraph using non-recursive DFS starting from selected_GO_vertex as root
2: GO_subgraph = extractSubgraph(GO_graph, selected_GO_vertex);
3: // build a list of all leaves in GO_subgraph
4: listOfLeaves = GO_subgraph.getAllLeaves();
5: // collect all paths to the cluster tree root for all GO leaves
6: for all vertex in listOfLeaves do
7:    // leaf labels are the same for both graphs, but the vertex objects are different
8:    label = GO_graph.getLabel(vertex);
9:    leaf = Cluster_tree.getVertexByLabel(label);
10:   // get all connected vertices from the current tree leaf up to the cluster tree root
11:   connectedVertices = getVerticesFromLeaf(Cluster_tree, leaf);
12:   // add connectedVertices to Cluster_subtree and create edges
13:   addVertices_createEdges(Cluster_subtree, connectedVertices);
14: end for
15: // find lowest common subtree root and
16: // remove the vertices from the cluster tree root to the lowest common root
17: removeRootChain(Cluster_tree, Cluster_subtree);
```

Scalability

As presented in Section 7.1, a number of issues need to be addressed when dealing with large data sets. Showing the complete data set to start the analysis process or providing an overview is one of the main challenges. It is clear that our prototype is able to visualize the complete data set as explained in the previous section. The mapping between the GO subset and Cluster Tree is performed and more insight into the data is gained with the help of the described interaction techniques.

Real time interaction is a must for relatively any kind of visualization environments. Often, visualizing large data sets might affect the system's responsiveness. This was one of our challenges during the development of CluMa-GO. It is extremely important that the system can handle all data and provide the users with real-time interaction possibilities. Approximately three to five seconds are required to load and visualize both input files. However, the complete subgraph and subtree has to be calculated which involves the parsing of almost all nodes from both data sets, when clicking on the GO root term. This action can take up to ten seconds to be calculated on a standard PC (Core 2 Duo Intel processor with 2.53 GHz). This is of course the worst case scenario. Most of the nodes from the lower levels respond immediately when selected. Nevertheless, the process has been speed up significantly once a simple caching strategy has been implemented that results in around one second to highlight the calculated subtree if the root node is chosen. The calculated mapping data are cached once the user has selected a particular GO term. Users will experience an almost immediate highlighting the next time the same node is selected as the previous calculation is stored in memory. A smart map from the open source library Google Guava [41] was used to reduce the memory usage for caching. A limit of stored elements together with a setting of their life times can be specified using this map. Our specification have been set up for a storage of up to 100 subgraphs for the GO and Cluster Tree for about one minute. The oldest map element will be removed if one of these limits is reached. Frequently used elements remain in the cache for a longer time.

7.4 Improvements for Balanced Cluster Trees

Our visualization of the Cluster Tree is based on the premise of visualizing highly unbalanced binary trees as stated in Section 7.2. There is a considerable amount of cases where more balanced trees appear even though unbalanced trees are common. This fact was presented to us by domain experts during the iterative development process. Our approach would not produce useful results with such trees at the current state. The first issue in such circumstances is related to the high ratio of the nodes among different subtrees which would render the normalization of the sizes of the boxes representing the branches (subtrees) ineffective. The second issue is that this would mean abstracting a lot of information which is against our initial goal of showing most of the information in a single view. Therefore, an improved version of our spiral tree metaphor to cope with more balanced binary trees was necessary. Although no implementation has been made yet, in the following possible strategies to solve this problem are presented. One possible solution is to create something we call "Nested Spiral Trees". Only branches under a user defined threshold should be aggregated in this case. Large branches that exceed the threshold should be drawn as smaller nested spirals (cf. Figures 7.15 and 7.16). The problem with such a strategy is that it introduces more unused spaces, making the approach less space-filling.

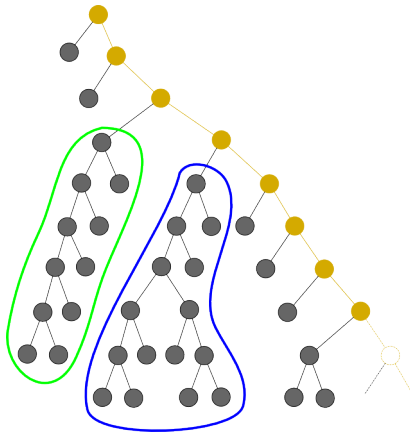


Figure 7.15: In this cut-out of the cluster tree, we assume that the subtrees highlighted in *green* and *blue* are too large to be abstracted into boxes.

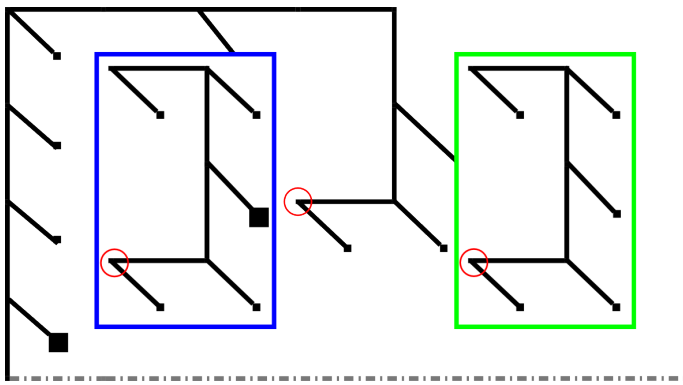


Figure 7.16: Nested Spiral Tree Layout built based on the tree sample in Figure 7.15. The red circles show the roots of the main tree (the circle in the center) and of the branches (i. e., the nested subtrees).

Therefore, other ways to visualize more balanced trees should be investigated due to this conceptual drawback.

Pixel-based approaches were our inspiration again, especially from the recursive pattern metaphor [72, 74]. Keep in mind that only the fundamental concept of a possible solution is presented, as the approach has not been implemented into our tool. The backbone feature will be used again similarly to the original spiral tree metaphor. In this case a snake-like shape layout for the backbone will be employed instead of a spiral, such as the one typically used in a recursive pattern.

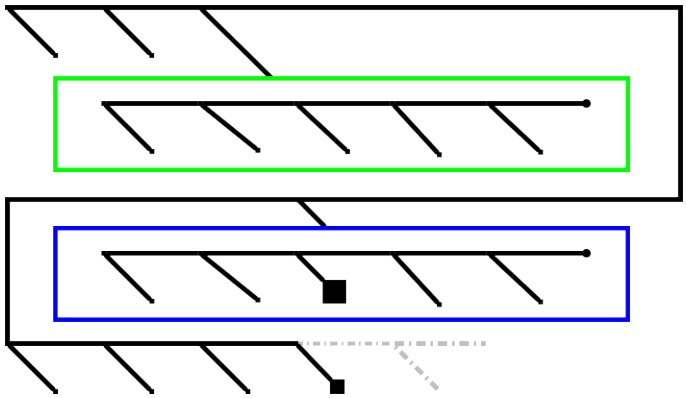


Figure 7.17: A new layout method based on recursive patterns in order to visualize more balanced cluster trees. Here, root nodes are always located in the top-left corner of each container.

Less space-expensive nested subtrees can be created with this approach named "Recursive Pattern Trees". In the following, a detailed description of the approach is presented.

The two subtrees in Figure 7.15, marked in *green* and *blue*, appear disproportionally big compared to other branches. Therefore, we assume that they are too big to be abstracted into a box. In such case, the backbone of these subtrees can be extracted and new spirals can be created. These spirals are then embedded into the general spiral view. As discussed earlier, the drawback of this approach is that it produces a lot of unused space. However, it is possible to lower the space usage by using the recursive pattern metaphor. The basic idea of how such a layout might look like is shown in Figure 7.17. In contrast to the Spiral Tree layout where the tree root is placed in the center, Recursive Pattern Trees places it on the top-left position. Subtrees are aggregated based on their size similarly to the original approach. However, the direction of the backbone resembles a snake-like metaphor, i.e., when the backbone reaches the end of the screen, it goes a step down and turns into the opposite direction and continues until reaching the end of the view. This process continues until the whole tree has been parsed. Similarly to the spiral metaphor, the two initial subtrees (which in this sample tree are single leaf nodes) are aggregated into boxes (see Figure 7.15). One can notice that there are two subtrees connected to the backbone before a bigger green box by following the backbone in Figure 7.17. Some branches will not be aggregated into the boxes. More precisely, larger branches, i. e., those that have more nodes than a predefined threshold will be visualized in a nested fashion. They will be displayed inside a bigger container using the same algorithm as for the entire Cluster Tree. The *blue* branch in Figure 7.17 corresponds to the branch highlighted in *blue* in Figure 7.15. The same

principle is applied for the *green* branch. To avoid the problem of possible clutter when two nested branches appear close to each other as it is the case with our sample tree, the new branch is positioned in a new line/row. This of course introduces some drawbacks, such as loss of space due to the elongated backbone. Users should also be aware that the real distance of the backbone has no meaning. However, this approach makes it easier to follow and to find the location where the nested subtrees appear. The algorithm continues aggregating the rest of the branches that do not pass the threshold after drawing the nested branches, as seen in Figure 7.17

7.5 Summary

A new method for the combined visualization of an ontology (represented as DAG) and a hierarchical clustering (represented as tree) of one data set was presented in this chapter. The proposed method interactively visualizes all the data without scrolling, thereby presenting a complete overview. Inspired by the shape of a spiral, a new metaphor for visualizing highly unbalanced binary trees was presented. Our approach also allows for interactive selection and navigation to explore the data. It was shown that our tool is able to tackle the problem in our research focus, i.e., the visualization and visual mapping of a complex data set derived from multivariate attributes. Additionally, a possible improvement of the approach was discussed at the end of the chapter.

Chapter 8

Conclusion and Future Work

This chapter presents a summary of this thesis. Moreover, our contributions in context of the goal criteria highlighted in Chapter 1 are discussed. Finally, the thesis is concluded with a discussion on future work.

The visualization of large and complex networks is one of the most challenging tasks in the information visualization and graph drawing communities [88]. Such network data sets are produced and/or studied by researchers from different domains of science. Their visualization can be considered as a hot topic for them. Many powerful network visualization tools and approaches have been developed. However, some core issues are still not adequately solved. Standard visualization and interaction techniques for the visual analysis of multivariate networks are introduced and discussed in our "Background Information" chapter.

The visualization of networks and their additional data that are attached to the network nodes and/or edges is the core topic of this thesis. In Chapter 3, a more detailed analysis of techniques and methods for multivariate networks visualization is presented. We defined a number of evaluation criteria and presented a taxonomy for multivariate network visualizations where different approaches are categorized based on the aforementioned criteria, such as the type of the attributes or the domain the tool was used for. Within the groups, common advantages and issues are present. In the following, our final grouping of the approaches is briefly repeated. *Multiple Coordinated Views* use a combination of two or more linked views to represent the network usually in one of the views and the multivariate data in other view(s). The approach allows flexibility in terms of choosing the appropriate visual representations for both, the network and its multivariate data. However, it splits the data across multiple views which acquires more efforts from the users to perceive and interpret this data. *Integrated Approaches* provide a combined visualization where the underlying graph and its attribute data are presented in a single view. But, in order to facilitate the embedding of additional attributes in the network, the size of the nodes or edges in the graph usually needs to be enlarged; this introduces clutter. *Semantic Substrates* place the nodes in non-overlapping regions based on node attributes. The drawback of such approaches is that the underlying graph topology is not completely visible. *Attribute-Driven Topology* approaches use network attribute values to steer the nodes positioning, which might hinder the perception of the network topology. Finally, a combination of two or more groups is the key property of *Hybrid* approaches.

In Chapter 4, the design and results of a task-based study investigating the usability of multiple coordinated views and integrated approaches are discussed. The study compared how effective and efficient they are with respect to multivariate attribute visualization. Here, integrated approaches performed significantly better in terms of efficiency, while in terms of effectiveness no significant difference between both approaches was noticed. The study also investigates possible distinctions in the way the network topology is perceived in both approaches. No significant differences among the approaches were found in terms of effectiveness or efficiency in context of the perception of the topological features.

In Chapter 5, the Network Lens is presented. Here, a focus+context interaction-based approach is proposed for solving the clutter problem introduced by integrated approaches. Our prototype is essentially an extension of the traditional magic lens idea [11, 123] applied to networks that are represented as node-link diagrams. Users can build various lenses by interactively specifying different attributes and selecting different visual representations. New lenses can also be created by combining two existing lenses with the help of set operators. The whole process of creating new lenses and the overall exploration becomes more intuitive by joining the lenses in such way that it follows an optics metaphor. When a particular lens is placed over a node, it will show the user-specified attribute values inside the node by using a user-defined visual representation. Two application scenarios with data from two different domains are provided. They demonstrate that our Network Lens approach could be applied on data from different science domains and be incorporated into other existing visualization systems.

Multivariate network visualization tools that are based on attribute-driven topology approaches usually hinder the perception of different network structures as a result of the attribute-based node placement. A tool called JauntyNets is presented in Chapter 6 that incorporates several approaches for visualizing multivariate networks. Its core visualization embodies an extension of traditional force-directed approaches to an attribute-driven layout technique. Users are able to select attributes or specify groups of semantically related attributes that are used to control the node positioning interactively. Additionally, the approach features interactive sliders to adjust the parameters for enforcing the network topology or providing attribute insight. By using these sliders, users can adjust the trade-off between the attribute or the network overview. A number of other interaction techniques were developed to aid the visual analysis process. This approach facilitates the discovery of interrelations between specific network structures and network attributes. Attribute-based clustering was realized to further support the exploration process. A multidimensional scaling visualization was implemented in a separated view to discover the similarity of network objects in context of their attribute values.

So far, two different tools that deal with the visualization of multivariate networks have been presented. Both approaches are concerned with attributes of standard data types. In Chapter 7, we focus on a slightly different problem. Here, an approach that visualizes complex data types which are derived from multivariate

data is discussed. Ontologies and hierarchical clustering results are important for biological analyzes. Therefore, it is desirable to show them both in the same view. Ontologies are modeled as large DAGs, while hierarchical clusters are a product of multivariate experimental data and form large binary trees. Our approach uses brushing and innovative layout designs to cope with the complexity and amount of data.

First, we present a novel layout approach for drawing the DAG by using a pixel-based techniques. The approach was implemented in two variations whose strengths and weaknesses are discussed separately. Intermediate nodes of the GO DAG are placed into specific hierarchical levels for both variations, and thus they only differ in the way the terminal nodes are placed. In order to emphasize the connections of each node to the terminals, one approach arranges all terminals into the last level. The other one gives an overview about the distribution of the terminals and intermediate nodes in each level by positioning terminal nodes into hierarchical levels accordingly, but on separate regions of the level.

Representing huge unbalanced binary trees by using traditional approaches introduces a lot of issues, thus a new drawing algorithm was developed. A "backbone" represents nodes and edges that form the longest path which connects all branches. The main idea behind our algorithm is to layout the backbone in form of a spiral. It uses the space more effectively for this particular data set, i.e., huge unbalanced binary trees, compared to traditional tree layout techniques. Standard interaction and brushing techniques are used to show the mapping between the GO and cluster tree. As cases of balanced trees appear as well, we proposed a couple of solutions that could cope with such structures.

8.1 Discussion

In this section, the results of the thesis in context to the goals defined in Chapter 1 are discussed. These goals are:

1. Offer a contribution regarding the categorization of the existing approaches and perform a study regarding the usability of the main approaches.

2. Offer a contribution for the visualization challenges presented in Section 1.1 with respect to different categories of approaches for the visualization of multivariate networks and offer a contribution for the visualization of complex data derived from multivariate networks by computational methods.

A set of criteria helped us to focus our research efforts on solving concrete problems that lead to the achievement of the more general goals. They are discussed in the following.

Goal 1

The criteria for the first goal are as follows:

1.1 Provide a survey on the current state of the art on the visualization of multivariate networks in general. As there are several different approaches for visualizing this type of data, there is a need to categorize them and investigate strengths and weakness of different approach categories. Therefore, an important task here is to define the criteria and categories for the categorization of multivariate approaches.

1.2 Convey a usability study regarding the main approaches for the visualization of multivariate networks.

Criterion 1.1 is fulfilled: in Chapter 3, a survey on multivariate network visualization is presented. Different approaches used in practice were investigated and a set of criteria to classify these approaches into different categories were designed. Advantages and disadvantages for each category were identified, which was an important step for our future research efforts. Additionally, the findings of the survey can be used as guidelines for designing multivariate network systems.

Criterion 1.2 is fulfilled: a task-based usability study investigating the effectiveness and the efficiency of multiple coordinated views and integrated approaches was presented in Chapter 4. The results of the study show that integrated approaches are better in terms of time needed to identify particular attribute patterns in a network. However, no significant difference was found in terms of effectiveness for this point. The results of the study could be used as guidelines for designing multivariate network systems as well.

Goal 2

The criteria for the second goal are as follows:

2.1 Different visualization approaches introduce different advantages, but also different disadvantages and challenges. Therefore, introducing novel interaction and visualization techniques for multivariate networks that tackle these challenges is one of the criteria for fulfilling the second goal.

2.2 As explained in Section 1.1, different computational processes may produce additional complex data derived from multivariate data. For instance in biology, different experimental procedures generate multivariate data that are later hierarchically clustered. The clustering usually produces trees which can be considered as additional complex data. Therefore, a novel approach for visualizing such derived cluster data for multivariate networks shall be introduced.

Criterion 2.1 is fulfilled: our first contribution for this criterion is the development of the Network Lens approach which improves the integrated approaches through a focus+context technique. Integrated approaches have the issue of visualizing a

large number of attribute values by enlarging the network components, thus introducing clutter. Our approach facilitates the visualization of such attributes by means of visual filtering. Another contribution for this criterion is our extension of force-directed layouts to form a visualization that belongs to the attribute-driven topology category. This approach relies on multiple coordinated views for some of its visual analysis capabilities. Additionally, the attribute values are interactively visualized inside the main visualization view. Thus, our tool can be considered as hybrid approach. The main contribution here is the ability to interactively enforce either the topology of the underlying network or node positioning based on the multivariate data values. With various interactions, users can achieve a balance between these two requirements, thus improving one of the disadvantages of the attribute-driven topology approaches. Contributions toward semantic substrate approaches are not provided as they are conceptually close to this approach. Multiple coordinated views have a conceptual problem of splitting the data in two or more views, thus requiring more cognitive efforts from the users. Although our hybrid approach relies on multiple coordinated views, no significant contribution is provided for them as well. The main reason for this is that these approaches have the same problem when used with any kind of data or visualization technique (different representations of networks, multivariate data, geographic data, etc.). Therefore, this is considered as a general challenge which is not focused solely on multivariate networks.

Criterion 2.2 is fulfilled: cluster trees are a product of multivariate data. Visualizing such trees in context of the underlying network is a challenging task. A new visualization approach to map these complex data types onto the underlying network was developed. In essence, our approach manages to visualize two conceptually different graphs and shows their mapping through interaction and brushing techniques. To cope with the size and complexity of the data, new layout metaphors were provided for both data sets. The development of the tool involved informal discussions with domains experts.

An overall insight about the challenges regarding multivariate network visualization was acquired by achieving the first goal. Additionally, it helped us in better understanding of these issues and thus focussing our research efforts. Moreover, our survey and usability test results could be considered as guidelines when developing visualizations for multivariate networks. Our contributions towards achieving the second goal mainly focus on the development of new visualization and interaction techniques. For this, three systems were developed and presented, each of them tackling one or more of the existing issues. The first system represents a pure information visualization approach relying on focus+context interactions. The last two approaches show a slight shift in the direction of the research field as already mentioned in the future work section in the licentiate thesis [62]. These approaches rely on heavy use of interaction techniques, especially JauntyNets, and data mining methods besides the introduction of new visualization techniques. These characteristics are related to the field of visual analytics that could be roughly defined as the use of information visualization together with data mining [140, 70]. With the

achievement of both goals, a contribution for the visual analysis of multivariate networks is given in this thesis. In the following, future research efforts are shortly highlighted.

8.2 Future Work

The usability study has shown some interesting results. However, there are some design modifications that could considerably improve its outcome. The sample size was not optimal, and performing the study again with the following improvements on a larger number of participants could overcome its shortcomings. One such improvement is to include a sample video of how the tasks should be performed. This will prepare the user with the interface of the tool and thus soften potential learning effects, where users would perform better by the end of the study process. Even though a figure highlighting the differences between trees and graphs was shown, at least one of the subjects commented that he/she was unsure about what such differences mean. Adding more sample images could help uninformed users to identify such concepts. Increasing the number of attributes which raises the nodes size, could yield different results in terms of efficiency of topology perception. The final improvement is regarding the measuring of the effectiveness of the approaches. Our subjects had virtually unlimited time to identify required objects in the network which led to a high percentage of true answers, i.e., there was no significant disparity between two approaches. Therefore, the tasks that deal with effectiveness of the approaches should be time constrained.

New visual metaphors should be implemented in our Network Lens approach and more logical operands for the lens combination process should be added. Creating lenses for edge attributes and combining them with existing lenses is also part of future work. Visualizing time-dependent data was not part of this thesis. However, this task is in our plans for future work for this approach as well as for JauntyNets.

JauntyNets should be extended by including new views. For instance, parallel coordinates would be suitable for a separate attribute visualization that supports interactive visual filtering of the data set. Different filtering actions can render some nodes unconnected to the rest of the network. In the current state of our tool, these nodes remain inside the circle affected by the repulsion and gravitational forces. To avoid any ambiguity and to possibly optimize the layout algorithm, they should be placed outside the circle in a separate region. At that point, these nodes should not be included in the layout process. In a similar manner the problem of inactive nodes should be solved. They create visual overload, and moving them in a separate region will solve the issue. Of course, users should be able to activate them again if necessary. Another future improvement is related to the further customization of attribute-based layering through the force-directed method. It should be possible to specify the edge stiffness parameter for each attribute or a group of attributes separately. This could allow users to put weights on attributes or a group of attributes.

Then it would be possible to use the tool for simulating data values. This can be achieved by adding small sliders on attribute nodes or group glyphs.

In the following, plans for future improvements of our approach for visualization of multivariate networks and cluster trees are presented (cp. Chapter 7). A direct mapping between a terminal GO DAG node and a cluster tree leaf is not supported currently. A straightforward solution to this problem for a specific node is to highlight the corresponding nodes in the GO view and/or Cluster Tree view by using mouse-over action. Implementing this feature would not require too much efforts. Concrete steps have already been taken to improve our spiral tree metaphor to cope with more balanced trees by creating new visualization designs. Two possible solutions are discussed in detail in Section 7.4. We need to implement these approaches and to perform a comparative evaluation with the existing spiral tree metaphor.

A lot of possibilities for improvements are not directly related to the multivariate network visualization approaches discussed in Chapter 7. Therefore, the following advancements should take place as side projects. As described in Section 7.2, the subgraph in the zoomed-in GO view is displayed only by highlighting the nodes without showing any edges. They are hidden because edges from a higher level might go through the zoomed-in view to nodes in the lower layers. This would introduce clutter. Showing these "transitive" edges makes no sense in a zoomed-in view, as the context is lost at this point. There is no way of knowing from which layer those edges are coming from, nor to which layer they are going to. One solution is to show only edges between the three layers shown in the zoomed-in GO view. Additionally, the edge bundling algorithm could be improved as well.

Several points of our future work are the result of our collaboration with domain experts. During this time, specific requirements were defined which have not been implemented as they require too much effort and do not fit directly into our research aims. They are listed as follows: import of microarray data sets directly into our tool, employment of different clustering algorithms, representing additional information connected to parts of the clustering and implementation of more statistical analysis methods. Another solution is to implement our visualization and interaction techniques as extension to existing tools that already support most of these requirements.

Bibliography

[1] *Streptomycin biosynthesis - Reference pathway*, last accessed: 2013-05-07. http://www.genome.jp/kegg-bin/show_pathway?map00521.

[2] Wolfgang Aigner, Silvia Miksch, Wolfgang Müller, Heidrun Schumann, and Christian Tominski. Visual methods for analyzing time-oriented data. *IEEE Transactions on Visualization and Computer Graphics (TVCG)*, 14(1):47–60, jan.-feb. 2008.

[3] Vladyslav Aleksakhin. Visualization of Gene Ontologies and Cluster Analysis Results. Master's thesis, Linnaeus University, Sweden, 2012.

[4] Daniel Archambault, Tamara Munzner, and David Auber. Grouseflocks: Steerable exploration of graph hierarchy space. *Visualization and Computer Graphics, IEEE Transactions on*, 14(4):900–913, july-aug. 2008.

[5] Daniel Archambault, Tamara Munzner, and David Auber. Tuggraph: Path-preserving hierarchies for browsing proximity and paths in graphs. In *Visualization Symposium, 2009. PacificVis '09. IEEE Pacific*, pages 113–120, april 2009.

[6] Aleks Aris and Ben Shneiderman. Designing semantic substrates for visual network exploration. *Information Visualization*, 6(4):281–300, December 2007.

[7] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, J. Michael Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. Gene ontology: tool for the unification of biology. *Nature Genetics*, 25(1):25–29, 2000.

[8] Patrick Baudisch, Nathaniel Good, Victoria Bellotti, and Pamela Schraedley. Keeping things in context: a comparative evaluation of focus plus context screens, overviews, and zooming. In *CHI '02: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 259–266, New York, NY, USA, 2002. ACM.

[9] Richard A. Becker and William S. Cleveland. Brushing scatterplots. *Technometrics*, 29(2):pp. 127–142, 1987.

[10] Anastasia Bezerianos, Fanny Chevalier, Pierre Dragicevic, Niklas Elmqvist, and Jean-Daniel Fekete. Graphdice: A system for exploring multivariate social networks. *Computer Graphics Forum (Proc. EuroVis 2010)*, 29(3):863–872, 2010.

[11] Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton, and Tony D. DeRose. Toolglass and magic lenses: the see-through interface. In *SIGGRAPH '93: Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pages 73–80, New York, NY, USA, 1993. ACM.

[12] Adrian Bondy and U. S. R. Murty. *Graph Theory*, volume 244 of *Graduate Texts in Mathematics*. Springer, 3rd corrected printing edition, 2008.

[13] 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 Biol*, 5(2):93–102, 2005.

[14] Frederik Börnke. Protein Interaction Networks. In *Analysis of Biological Networks* [61], pages 207–232.

[15] Ulrik Brandes, Markus Eiglsperger, Ivan Herman, Michael Himsolt, and M. Scott Marshall. Graphml progress report (structural layer proposal). In Petra Mutzel, Michael Jünger, and Sebastian Leipert, editors, *Proceedings of the 9th International Symposium on Graph Drawing (GD '01)*, volume 2265 of *LNCS*, pages 501–512. Springer, 2002.

[16] Cynthia A Brewer. *ColorBrewer*, 2nd edition, last accessed: 2013-05-11. http://colorbrewer2.org/.

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

[18] Herman Chernoff. The Use of Faces to Represent Points in K-Dimensional Space Graphically. *Journal of the American Statistical Association*, 68(342):361–368, 1973.

[19] IBM Corp. *IBM SPSS Statistics for Windows, Version 21.0*. IBM Corp., Armonk, NY, 2012.

[20] Richard G. Côté, Philip Jones, Lennart Martens, Rolf Apweiler, and Henning Hermjakob. The ontology lookup service: more data and better tools for controlled vocabulary queries. *Nucleic Acids Research*, 36:W372–W376, 2008.

[21] Paul Craig and Jessie Kennedy. Coordinated graph and scatter-plot views for the visual exploration of microarray time-series data. In *Information Visualization, 2003. INFOVIS 2003. IEEE Symposium on*, pages 173–180, 2003.

[22] Paul Craig, Jessie Kennedy, and Andrew Cumming. Towards visualising temporal features in large scale microarray time-series data. In *Information Visualisation, 2002. Proceedings. Sixth International Conference on*, pages 427–433, 2002.

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

[24] Yang Dingjie. The Network Lens. Master's thesis, Linnaeus University, Sweden, 2010.

[25] Tim Dwyer, Seok-Hee Hong, Dirk Koschützki, Falk Schreiber, and Kai Xu. Visual analysis of network centralities. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation - Volume 60*, APVis '06, pages 189–197, Darlinghurst, Australia, Australia, 2006. Australian Computer Society, Inc.

[26] Tim Dwyer, Kim Marriott, and Michael Wybrow. Integrating edge routing into force-directed layout. In *Proceedings of the 14th international conference on Graph drawing (GD '06)*. Springer, 2007.

[27] Tim Dwyer and Falk Schreiber. Optimal leaf ordering for two and a half dimensional phylogenetic tree visualisation. In *Proceedings of the 2004 Australasian symposium on Information Visualisation - Volume 35*, APVis '04, pages 109–115, Darlinghurst, Australia, 2004. Australian Computer Society, Inc.

[28] Peter Eades. A heuristic for graph drawing. *Congressus Numerantium*, 42:149–160, 1984.

[29] Peter Eades and Fujitsu Laboratories. International Institute for Advanced Study of Social Information Science. *Drawing Free Trees*. IIAS-RR-. International Institute for Advanced Study of Social Information Science, Fujitsu Limited, 1991.

[30] Michael B. Eisen, Paul T. Spellman, Patrick O. Brown, and David Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868, 1998.

[31] Niklas Elmqvist, Pierre Dragicevic, and Jean-Daniel Fekete. Rolling the dice: Multidimensional visual exploration using scatterplot matrix navigation. *IEEE Transactions on Visualization and Computer Graphics (Proc. InfoVis 2008)*, 14(6):1141–1148, 2008.

[32] Ozan Ersoy, Christophe Hurter, Fernando Paulovich, Gabriel Cantareiro, and Alex Telea. Skeleton-based edge bundling for graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17:2364–2373, 2011.

[33] Henning Fernau, Michael Kaufmann, and Mathias Poths. Comparing trees via crossing minimization. In *In Proc. 25th Conf. on Foundations of Software Technology and Theoretical Computer Science, volume 3821 of LNCS*, pages 457–469, 2005.

[34] Akira Funahashi, Mineo Morohashi, and Hiroaki Kitano. CellDesigner: a process diagram editor for gene-regulatory and biochemical networks. *Biosilico*, 1(5):159–162, 2003.

[35] Ursula Gaedke. Ecological Networks. In *Analysis of Biological Networks* [61], pages 283–304.

[36] Michael R. Garey and David S. Johnson. Crossing Number is NP-Complete. *SIAM Journal on Algebraic and Discrete Methods*, 4(3):312–316, 1983.

[37] Birgit Gemeinholzer. Phylogenetic Networks. In *Analysis of Biological Networks* [61], pages 255–281.

[38] Gene ontology, last accessed: 2013-05-11. http://www.geneontology.org/.

[39] giCentre.org. giCentre Utilities, last accessed: 2013-05-11. http://gicentre.org/utils/.

[40] Michael Gleicher, Danielle Albers, Rick Walker, Ilir Jusufi, Charles D. Hansen, and Jonathan C. Roberts. Visual comparison for information visualization. *Information Visualization*, 10(4):289–309, 2011.

[41] Google. Guava: Google Core Libraries for Java 1.5+, last accessed: 2013-05-11. http://code.google.com/p/guava-libraries/.

[42] Carsten Görg, Mathias Pohl, Ermir Qeli, and Kai Xu. Visual Representations. In Kerren et al. [75], pages 163–230.

[43] Martin Graham and Jessie Kennedy. A survey of multiple tree visualisation. *Information Visualization*, 9(4):235–252, 2010.

[44] Carl Gutwin and Chris Fedak. A comparison of fisheye lenses for interactive layout tasks. In *GI '04: Proceedings of Graphics Interface 2004*, pages 213–220, Waterloo, Ontario, Canada, 2004. Canadian Human-Computer Communications Society.

[45] John A. Hartigan. Printer graphics for clustering. *Journal of Statistical Computation and Simulation*, 4(3):187–213, 1975.

[46] Helwig Hauser, Florian Ledermann, and Helmut Doleisch. Angular brushing of extended parallel coordinates. In *Proc. IEEE Symposium on Information Visualization (InfoVis'02)*, page 127, Washington, DC, USA, 2002. IEEE Computer Society.

[47] Jeffrey Heer, Stuart K. Card, and James A. Landay. prefuse: a toolkit for interactive information visualization. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '05, pages 421–430, New York, NY, USA, 2005. ACM.

[48] 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.

[49] Ivan Herman, Guy Melançon, and M. Scott Marshall. Graph Visualization and Navigation in Information Visualization: A Survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, slash 2000.

[50] Michael Himsolt. Gml: A portable graph file format. Technical report, University of Passau, Germany, 1997.

[51] Danny Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12:741–748, 2006.

[52] Danny Holten and Jarke J. Van Wijk. Force-Directed Edge Bundling for Graph Visualization. *Computer Graphics Forum*, 28(3):983–990, June 2009.

[53] Zhenjun Hu, Joe Mellor, Jie Wu, Takuji Yamada, Dustin T. Holloway, and Charles DeLisi. VisANT: data-integrating visual framework for biological networks and modules. 33:W352–W357, 2005.

[54] Victor Hugo. *Les misérables*. Wordsworth Classics, 1994.

[55] Daniel H. Huson, Regula Rupp, and Celine Scornavacca. *Phylogenetic Networks*. Cambridge University Press, 2010.

[56] Alfred Inselberg and B. Dimsdale. Parallel coordinates: A tool for visualizing multi-dimensional geometry. In *IEEE Visualization*, pages 361–378, 1990.

[57] Jigsaw. Visual Analytics for Exploring and Understanding Document Collections, last accessed: 2013-02-15. http://www.cc.gatech.edu/gvu/ii/jigsaw/datafiles.html.

[58] JogAmp. Home of the projects JOGL, JOCL and JOAL, last accessed: 2013-05-11. http://jogamp.org/.

[59] Brian Johnson and Ben Shneiderman. Tree-maps: a space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2nd conference on Visualization '91*, VIS '91, pages 284–291, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.

[60] Björn H. Junker, Christian Klukas, and Falk Schreiber. VANTED: A system for advanced data analysis and visualization in the context of biological networks. *BMC Bioinformatics*, 7:109, 2006.

[61] Björn H. Junker and Falk Schreiber. *Analysis of Biological Networks (Wiley Series in Bioinformatics)*. Wiley-Interscience, March 2008.

[62] Ilir Jusufi. *Towards the Visualization of Multivariate Biochemical Networks*. Licentiate thesis, Linnaeus University, 2012.

[63] Ilir Jusufi, Yang Dingjie, and Andreas Kerren. The Network Lens: Interactive exploration of multivariate networks using visual filtering. *14th International Conference on Information Visualisation (IV '10)*, pages 35–42, 2010.

[64] Ilir Jusufi, Andreas Kerren, Vladyslav Aleksakhin, and Falk Schreiber. Visualization of Mappings between the Gene Ontology and Cluster Trees. In *Proceedings of the SPIE 2012 Conference on Visualization and Data Analysis (VDA '12)*, Burlingame, CA, USA, 2012. IS&T/SPIE.

[65] Ilir Jusufi, Andreas Kerren, and Falk Schreiber. Exploring biological data: Mappings between ontology- and cluster-based representations. *Information Visualization*, 2013.

[66] Ilir Jusufi, Andreas Kerren, and Yuanmao Wang. A new radial space-filling visualization approach for planar st-graphs. In *Poster Abstracts of IEEE VisWeek 2012*, 2012.

[67] Ilir Jusufi, Andreas Kerren, and Björn Zimmer. Multivariate network exploration with jauntynets. *17th International Conference on Information Visualisation (IV '13)*, 2013. (to appear).

[68] Minoru Kanehisa, Susumu Goto, Shuichi Kawashima, and Akihiro Nakaya. The KEGG databases at GenomeNet. 30(1):42–46, 2002.

[69] Michael Kaufmann and Dorothea Wagner. *Drawing Graphs: Methods and Models*, volume 2025 of *Lecture Notes in Computer Science Tutorial*. Springer, 1999.

[70] Daniel Keim, Gennady Andrienko, Jean-Daniel Fekete, Carsten Görg, Jörn Kohlhammer, and Guy Melançon. Visual Analytics: Definition, Process, and Challenges. In *Information Visualization: Human-Centered Issues and Perspectives (Lecture Notes in Computer Science)*, pages 154–175. Springer, August 2008.

[71] Daniel A. Keim. Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):59–78, January 2000.

[72] Daniel A. Keim, Mihael Ankerst, and Hans-Peter Kriegel. Recursive pattern: A technique for visualizing very large amounts of data. In *Proceedings of the 6th conference on Visualization '95*, VIS '95, pages 279–286. IEEE Computer Society, 1995.

[73] Daniel A. Keim, Jörn Schneidewind, and Mike Sips. Circleview: a new approach for visualizing time-related multidimensional data sets. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '04, pages 179–182, New York, NY, USA, 2004. ACM.

[74] Daniel A. Keim, Jörn Schneidewind, and Mike Sips. Scalable pixel based visual data exploration. In *Proceedings of the 1st first visual information expert conference on Pixelization paradigm*, VIEW '06, pages 12–24, Berlin, Heidelberg, 2007. Springer-Verlag.

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

[76] Andreas Kerren and Ilir Jusufi. Novel visual representations for software metrics using 3d and animation. In Jürgen Münch and Peter Liggesmeyer, editors, *Software Engineering 2009 - Workshopband, Fachtagung des GI-Fachbereichs Softwaretechnik*, volume 150 of *LNI*, pages 147–154. GI, 2009.

[77] Andreas Kerren and Ilir Jusufi. 3d kiviat diagrams for the interactive analysis of software metric trends. In *Proceedings of the 5th international symposium on Software visualization*, SOFTVIS '10, pages 203–204, New York, NY, USA, 2010. ACM.

[78] Andreas Kerren, Ilir Jusufi, Vladyslav Aleksakhin, and Falk Schreiber. CluMa-GO: Bring Gene Ontologies and Hierarchical Clusterings Together. In *Extended Abstract, BioVis 11*, Providence, RI, USA, 2011.

[79] Andreas Kerren and Harald Köstinger. Interactive Exploration and Analysis of Network Centralities. In *Interactive Poster, EuroVis 11*, Bergen, Norway, 2011.

[80] 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.

[81] Donald E. Knuth. *The art of computer programming, volume 1: fundamental algorithms*. Addison Wesley, 1970.

[82] Donald E. Knuth. *The Stanford GraphBase: a platform for combinatorial computing*. ACM, New York, NY, USA, 1993.

[83] Jacob Köhler, Jan Baumbach, Jan Taubert, Michael Specht, Andre Skusa, Alexander Rüegg, Chris Rawlings, Paul Verrier, and Stephan Philippi. Graph-based analysis and visualization of experimental results with ONDEX. 22(11):1383–1390, 2006.

[84] Teuvo Kohonen, Manfred R. Schroeder, and Thomas S. Huang, editors. *Self-Organizing Maps*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition, 2001.

[85] Nobuaki Kono, Kazuharu Arakawa, Ryu Ogawa, Nobuhiro Kido, Kazuki Oshita, Keita Ikegami, Satoshi Tamaki, and Masaru Tomita. Pathway projector: Web-based zoomable pathway browser using KEGG atlas and Google maps API. *PLOS One*, 4(11):e7710, 2009.

[86] Olga A. Kulyk, Robert Kosara, Jaime Urquiza-Fuentes, and Ingo H. C. Wassink. Human-centered aspects. In Kerren et al. [75], pages 13–75.

[87] John Lamping, Ramana Rao, and Peter Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95, pages 401–408, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co.

[88] Robert S. Laramee and Robert Kosara. Challenges and Unsolved Problems. In Kerren et al. [75], pages 231–254.

[89] Rafael Messias Martins, Gabriel Faria Andery, Henry Heberle, Fernando Vieira Paulovich, Alneu Andrade Lopes, Hélio Pedrini, and Rosane Minghim. Multidimensional projections for visual analysis of social networks. *Journal of Computer Science and Technology*, 27(4):791–810, 2012.

[90] MDSJ. Java Library for Multidimensional Scaling (Version 0.2), last accessed: 2013-05-11. http://www.inf.uni-konstanz.de/algo/software/mdsj/.

[91] Boris Mirkin. *Clustering for Data Mining: A Data Recovery Approach*. Chapman & Hall/CRC, Boca Raton, FL, USA, 2005.

[92] Tamara Munzner. H3: laying out large directed graphs in 3D hyperbolic space. In *Information Visualization, 1997. Proceedings., IEEE Symposium on*, pages 2–10,, 1997.

[93] Tamara Munzner and Paul Burchard. Visualizing the structure of the world wide web in 3d hyperbolic space. In *Proceedings of VRML '95*, pages 33–38. ACM Press, 1995.

[94] Joshua O'Madadhain, Danyel Fisher, and Tom Nelson. JUNG – Java Universal Network/Graph Framework, 2009.

[95] Christian Partl, Alexander Lex, Marc Streit, Denis Kalkofen, Karl Kashofer, and Dieter Schmalstieg. enRoute: dynamic path extraction from biological pathway maps for In-Depth experimental data analysis. In *Proceedings of the IEEE Symposium on Biological Data Visualization*, Seattle (WA), USA, 2012.

[96] Martin Pinzger, Harald Gall, Michael Fischer, and Michele Lanza. Visualizing Multiple Evolution Metrics. In *Proceedings of the ACM Symposium on Software Visualization (SoftVis '05)*, pages 354–362, St. Louis, Missouri, 2005.

[97] Gary A. Polis and K Winemiller. *Food webs: integration of patterns & dynamics*. Chapman & Hall, 1996.

[98] Anatolij P. Potapov. Signal Transduction and Gene Regulatory Networks. In *Analysis of Biological Networks* [61], pages 183–206.

[99] A. Johannes Pretorius and Jarke J. van Wijk. Bridging the semantic gap: Visualizing transition graphs with user-defined diagrams. *Computer Graphics and Applications, IEEE*, 27(5):58 –66, sept.-oct. 2007.

[100] A. Johannes Pretorius and Jarke J. van Wijk. Visual inspection of multivariate graphs. *Comput. Graph. Forum*, 27(3):967–974, 2008.

[101] Processing.org. Processing, last accessed: 2013-05-11. http://processing.org/.

[102] Helen C. Purchase. *Experimental Human-Computer Interaction: A Practical Guide with Visual Examples*. Cambridge University Press, 2012.

[103] Edward M. Reingold, John, and S. Tilford. Tidier drawing of trees. *IEEE Transactions on Software Engineering*, pages 223–228, 1981.

[104] Jun Rekimoto and Mark Green. The information cube: Using transparency in 3d information visualization. In *Proceedings of the Third Annual Workshop on Information Technologies & Systems (WITS'93*, pages 125–132, 1993.

[105] Jonathan C. Roberts. Exploratory visualization with multiple linked views. In Alan MacEachren, Menno-Jan Kraak, and Jason Dykes, editors, *Exploring Geovisualization*, chapter 8, pages 159–180. Elseviers, 2004.

[106] George G. Robertson, Jock D. Mackinlay, and Stuart K. Card. Cone Trees: animated 3D visualizations of hierarchical information. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching*

through technology, CHI '91, pages 189–194, New York, NY, USA, 1991. ACM.

[107] Markus Rohrschneider, Alexander Ullrich, Andreas Kerren, Peter F. Stadler, and Gerik Scheuermann. Visual network analysis of dynamic metabolic pathways. In *Proceedings of the 6th international conference on Advances in visual computing - Volume Part I*, ISVC'10, pages 316–327, Berlin, Heidelberg, 2010. Springer-Verlag.

[108] Rodrigo Santamaría and Roberto Therón. Treevolution: visual analysis of phylogenetic trees. *Bioinformatics*, 25:1970–1971, August 2009.

[109] Manojit Sarkar and Marc H. Brown. Graphical fisheye views. *Commun. ACM*, 37(12):73–83, December 1994.

[110] Celine Scornavacca, Franziska Zickmann, and Daniel H. Huson. Tanglegrams for rooted phylogenetic trees and networks. *Bioinformatics*, 27(13):i248–i256, July 2011.

[111] David Selassie, Brandon Heller, and Jeffrey Heer. Divided edge bundling for directional network data. *IEEE Transactions on Visualization and Computer Graphics*, 17:2354–2363, 2011.

[112] Jinwook Seo and Ben Shneiderman. Interactively exploring hierarchical clustering results. *Computer*, 35(7):80–86, July 2002.

[113] 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*, 13(11):2498–2504, 2003.

[114] Ross Shannon, Thomas Holland, and Aaron Quigley. Multivariate graph drawing using parallel coordinate visualisations. Technical Report 2008-6, University College Dublin, School of Computer Science and Informatics, 2008.

[115] Zeqian Shen and Kwan-Liu Ma. Mobivis: A visualization system for exploring mobile data. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 175–182. IEEE VGTC, 2008.

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

[117] Márcio Rosa da Silva, Jibin Sun, Hongwu Ma, Feng He, and An-Ping Zeng. Metabolic Networks. In *Analysis of Biological Networks* [61], pages 233–253.

[118] SourceForge. Find, create, and publish open source software for free, last accessed: 2013-05-11. http://sourceforge.net/.

[119] Robert Spence. *Information Visualization: Design for Interaction*. Prentice Hall, 2nd edition, 2007.

[120] John T. Stasko, Carsten Görg, and Zhicheng Liu. Jigsaw: supporting investigative analysis through interactive visualization. *Information Visualization*, 7:118–132.

[121] John T. Stasko and Eugene Zhang. Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *InfoVis '00: Proc. IEEE Symposium on Information Vizualization 2000*, page 57, Washington, USA, 2000. IEEE Computer Society.

[122] Dirk Steinhauser, Leonard Krall, Carsten Müssig, Dirk Büssis, and Björn Usadel. Correlation Networks. In *Analysis of Biological Networks* [61], pages 305–333.

[123] Maureen C. Stone, Ken Fishkin, and Eric A. Bier. The movable filter as a user interface tool. In *CHI '94: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 306–312, New York, NY, USA, 1994. ACM.

[124] The Gene Ontology Consortium. The Gene Ontology project in 2008. *Nucleic Acids Research*, 36(36):D440–D444, 2008.

[125] Roberto Therón. Hierarchical-temporal data visualization using a tree-ring metaphor. In Andreas Butz, Brian Fisher, Antonio Krüger, and Patrick Olivier, editors, *Smart Graphics*, volume 4073 of *Lecture Notes in Computer Science*, pages 70–81. Springer Berlin / Heidelberg, 2006.

[126] Conrad Thiede, Georg Fuchs, and Heidrun Schumann. Smart lenses. In *Proceedings of the 9th international symposium on Smart Graphics*, SG '08, pages 178–189, Berlin, Heidelberg, 2008. Springer-Verlag.

[127] Christian Tominski, James Abello, and Heidrun Schumann. Technical section: Cgv-an interactive graph visualization system. *Comput. Graph.*, 33:660–678, December 2009.

[128] Christian Tominski, James Abello, Frank van Ham, and Heidrun Schumann. Fisheye tree views and lenses for graph visualization. In *Proceedings of the conference on Information Visualization*, IV '06, pages 17–24, Washington, DC, USA, 2006. IEEE Computer Society.

[129] Christian Tominski and Heidrun Schumann. Enhanced interactive spiral display. In *Proceedings of the annual SIGRAD Conference, Special Theme: Interaction*, SIGRAD '08, pages 53–56. Linköping University Electronic Press, 2008.

[130] Trickl. trickl-cluster, last accessed: 2013-05-11. https://github.com/trickl/trickl-cluster.

[131] Martijn van Iersel, Thomas Kelder, Alexander Pico, Kristina Hanspers, Susan Coort, Bruce Conklin, and Chris Evelo. Presenting and exploring biological pathways with PathVisio. *BMC Bioinformatics*, 9:399, 2008.

[132] Jarke J. Van Wijk and Wim A. A. Nuij. Smooth and efficient zooming and panning. In *Proceedings of the Ninth annual IEEE conference on Information visualization*, InfoVis'03, pages 15–22, Washington, DC, USA, 2003. IEEE Computer Society.

[133] Corinna Vehlow, Jan Hasenauer, Andrei Kramer, Julian Heinrich, Nicole Radde, Frank Allgöwer, and Daniel Weiskopf. Uncertainty-aware visual analysis of biochemical reaction networks. In IEEE, editor, *IEEE Symposium on Biological Data Visualization*, volume 2012, pages 91–98, 2012.

[134] Balaji Venkatachalam, Jim Apple, Katherine St. John, and Dan Gusfield. Untangling tanglegrams: Comparing trees by their drawings. In *Proceedings of the 5th International Symposium on Bioinformatics Research and Applications*, ISBRA '09, pages 88–99, Berlin, Heidelberg, 2009. Springer-Verlag.

[135] Matthew O. Ward. A taxonomy of glyph placement strategies for multidimensional data visualization. *Information Visualization*, 1:194–210, 2002.

[136] Colin Ware. Designing with a 2 1/2d attitude. *Information Design Journal*, 10:2001, 2001.

[137] Colin Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers, second edition, 2004.

[138] Martin Wattenberg. Visual exploration of multivariate graphs. In *CHI '06: Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 811–819, New York, NY, USA, 2006. ACM.

[139] Nelson Wong, Sheelagh Carpendale, and Saul Greenberg. Edgelens: An interactive method for managing edge congestion in graphs. In *Proceedinngs of the IEEE Symposium on Information Visualization*, pages 51–58, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[140] Pak Chung Wong and Jim Thomas. Visual analytics. *IEEE Comput. Graph. Appl.*, 24:20–21, September 2004.

[141] Hendley Drew Wood. Narcissus: Visualising information, 1995.

[142] Yingxin Wu and Masahiro Takatsuka. Visualizing multivariate network on the surface of a sphere. In *Asia Pacific Symposium on Information Visualisation*, pages 77–83, 2006.

[143] Ji Soo Yi, Youn ah Kang, John T. Stasko, and Julie Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13:1224–1231, 2007.

[144] yWorks. *yEd Graph Editor*, last accessed: 2013-05-11. http://www.yworks.com/en/products_yed_about.html.

[145] Xiaowei Zhu, Mark Gerstein, and Michael Snyder. Getting connected: analysis and principles of biological networks. *Genes & Development*, 21(9):1010–1024, May 2007.

[146] Björn Zimmer, Ilir Jusufi, and Andreas Kerren. Analyzing multiple network centralities with ViNCent. In *Proceedings of SIGRAD 2012 : Interactive Visual Analysis of Data, November 29-30, 2012, Växjö, Sweden,*, number 81 in Linköping Electronic Conference Proceedings, pages 87–90, 2012.

Appendix A

Usability Study Network Samples

The network samples designed for the usability study discussed in Chapter 4 are presented here. Samples shown in the chapter itself are omitted. Eight different graphs were designed by hand (four sparse and four dense graphs). Each graph has two versions: multiple coordinated views (MCV) and integrated approaches (IA), resulting in 16 different combinations in total. Therefore, both versions for each graph will be shown sequentially. Sparse graphs are shown first.

Subjects of the study had to perform four different tasks related to these graphs. For the first task, they had to identify (by mouse-click) the node of the network with the *lowest* average attribute values. The second task is very similar to the first one. However in this case, the node with the *highest* average attribute values was required. For the third task, subjects were asked to identify if the presented graph is a tree. They could answer the question by choosing "Yes", "No", or "Maybe". For the final tasks, subjects had to grade the readability of the network topology for the given graph with values from 1 to 5 (higher values mean better readability).

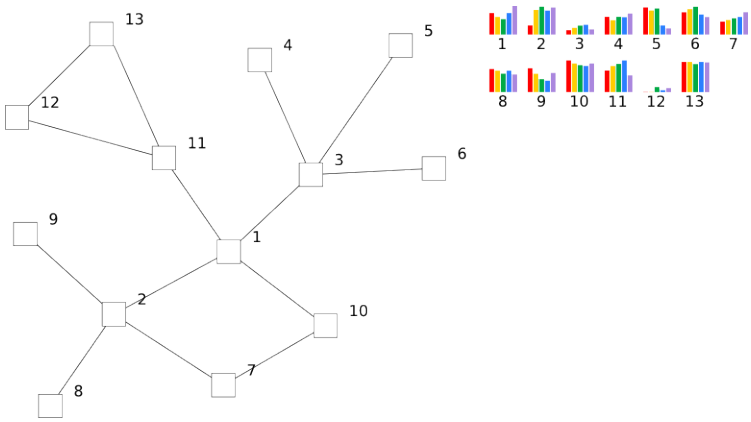


Figure A.1: Showing a sparse graph named SP1 visualized by using a multiple coordinated views approach.

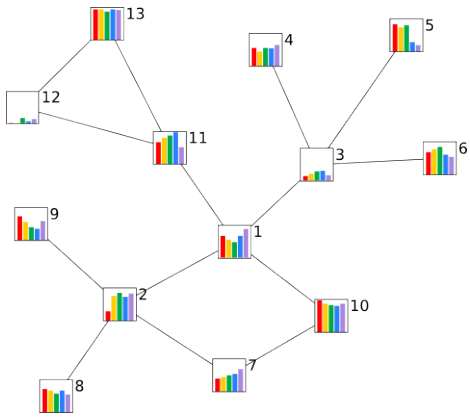


Figure A.2: Showing a sparse graph named SP1 visualized by using an integrated approach.

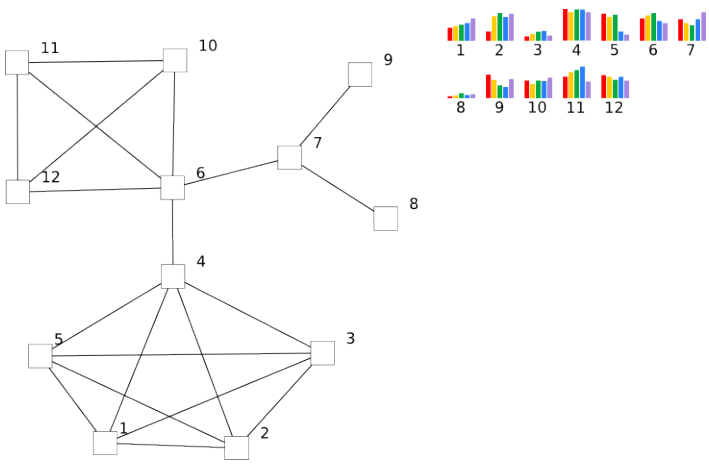


Figure A.3: Showing a sparse graph named SP2 visualized by using a multiple coordinated views approach.

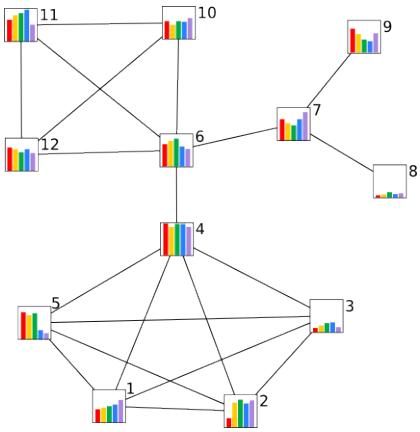


Figure A.4: Showing a sparse graph named SP2 visualized by using an integrated approach.

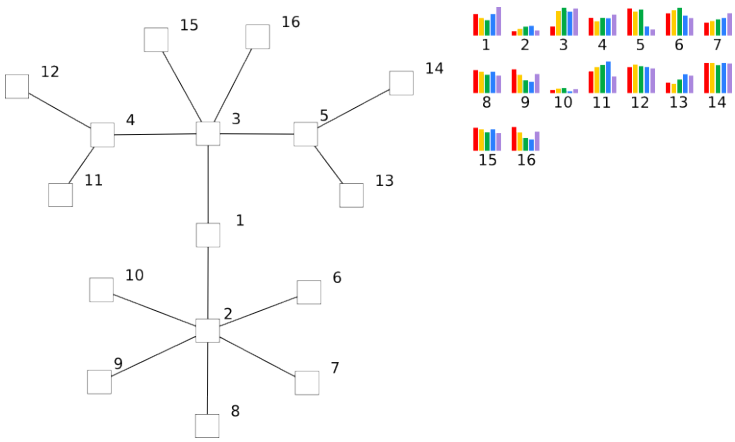


Figure A.5: Showing a sparse graph named SP3 visualized by using a multiple coordinated views approach (cp. Figure 4.1 for its corresponding version using IA).

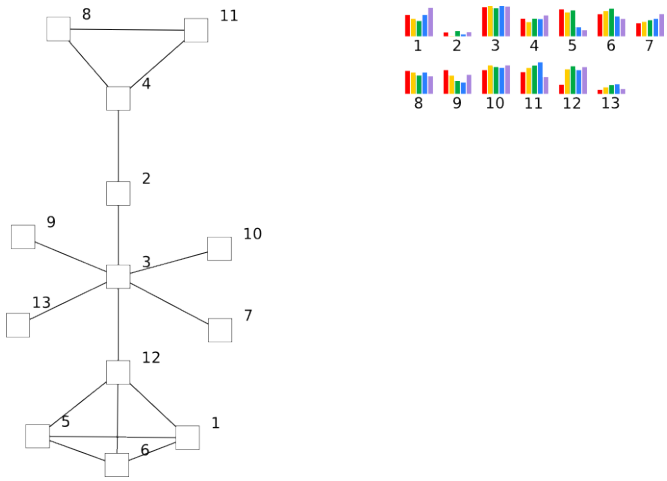


Figure A.6: Showing a sparse graph named SP4 visualized by using a multiple coordinated views approach.

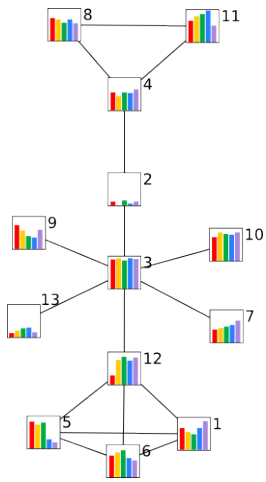


Figure A.7: Showing a sparse graph named SP4 visualized by using an integrated approach.

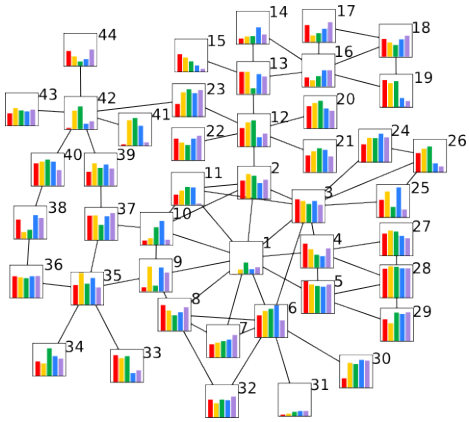


Figure A.8: Showing a dense graph named DE1 visualized by using an integrated approach (cp. Figure 4.1 for its corresponding version using MCV).

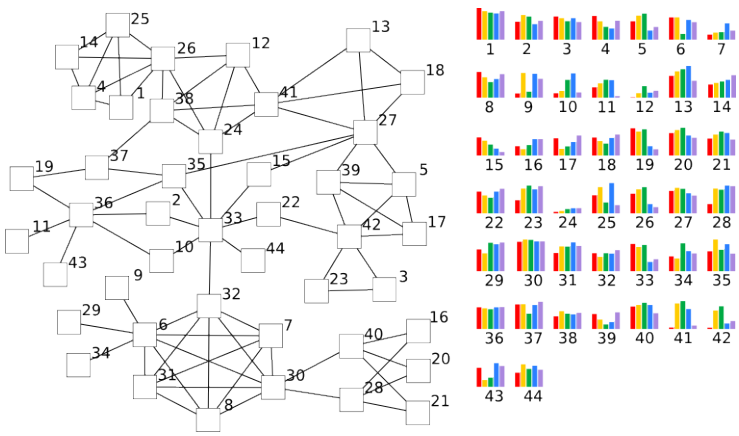


Figure A.9: Showing a dense graph named DE2 visualized by using a multiple coordinated views approach.

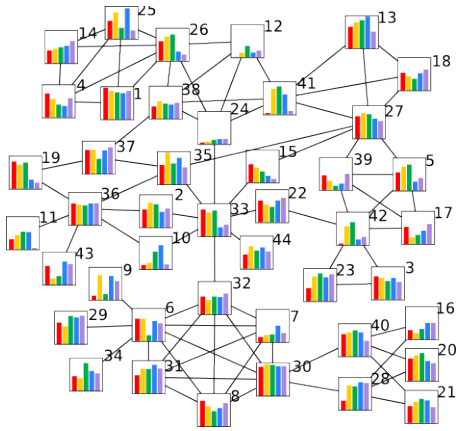


Figure A.10: Showing a dense graph named DE2 visualized by using an integrated approach.

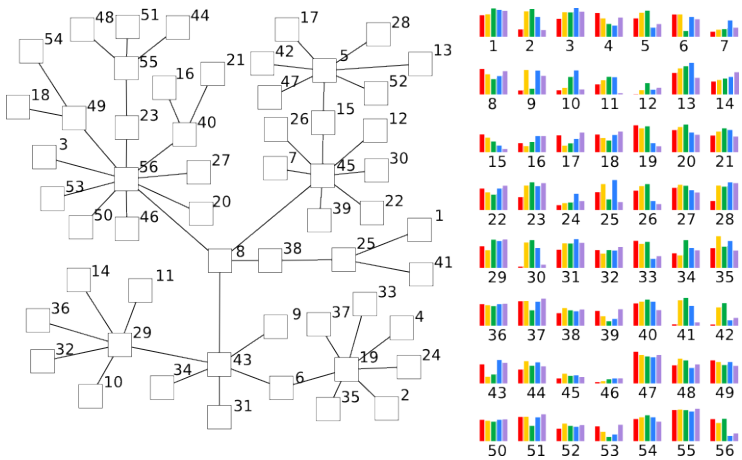


Figure A.11: Showing a dense graph named DE3 visualized by using a multiple coordinated views approach.

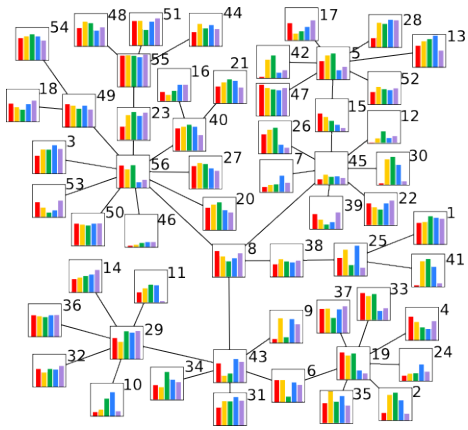


Figure A.12: Showing a dense graph named DE3 visualized by using an integrated approach.

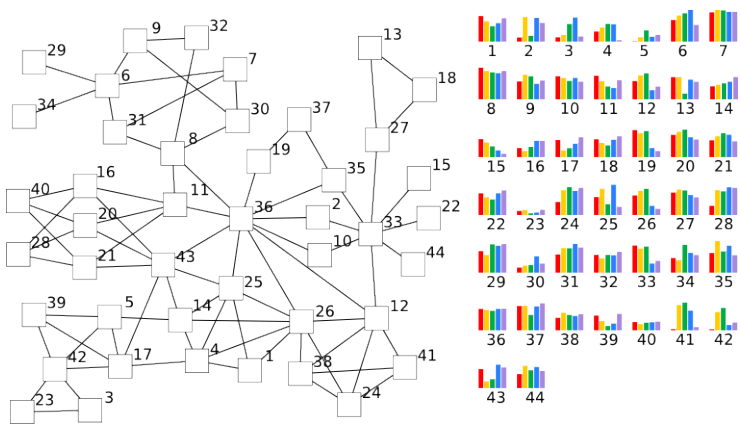


Figure A.13: Showing a dense graph named DE4 visualized by using a multiple coordinated views approach.

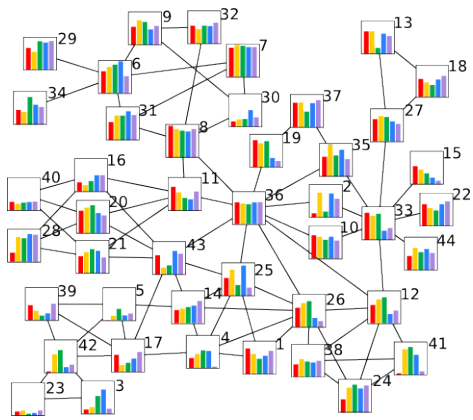


Figure A.14: Showing a dense graph named DE4 visualized by using an integrated approach.