Ilir Jusufi

Towards the Visualization of Multivariate Biochemical Networks

Licentiate Thesis

Computer Science

2012

Linnæus University

A thesis for the Degree of Licentiate of Philosophy in Computer Science.

Towards the Visualization of Multivariate Biochemical Networks
Ilir Jusufi

Linnæus University
School of Computer Science, Physics and Mathematics
SE-351 95 Växjö, Sweden
http://www.lnu.se/

Abstract

Many open challenges exist when dealing with different biological networks. They are crucial for the understanding of living beings. Complete drawings of these typically large networks usually suffer from clutter and visual overload. In order to overcome this issue, the networks are divided into single, hierarchically structured pathways. However, this subdivision makes it harder to navigate and understand the connections between pathways. Another challenge is to visualize ontologies and hierarchical clusterings, which are important tools to study high-throughput data that are automatically generated nowadays. Both of these methods produce different types of large graphs. Although these methods are used to explore the same data set, they are usually considered independently. Therefore, a combined view showing the results of both methods is desired. Additionally, real life data sets, including biological networks, usually have additional attributes related to the considered network. Investigating means to visualize such multivariate data together with the network drawing is also one of the ongoing challenges in biology, but also in other fields.

The aim of this thesis is to lay out the foundations towards defining techniques for the visualization of multivariate biochemical networks. An overall understanding of the problems related to biochemical networks should be acquired to achieve this aim. More importantly, a contribution to the aforementioned challenges is necessary. Two research goals have been defined to accomplish our aim: for the first goal, we should improve shortcomings of the approach of dividing larger biological networks into smaller pieces and contribute to the problem of a visualization of different types of interconnected biological networks. The second goal is a contribution for the visualization of multivariate biological networks.

Initially, a brief survey on techniques to visualize multivariate networks is presented in this thesis. Then, various visualization and interaction techniques are presented that address the challenges in biochemical network analysis. Three different software tools were implemented to demonstrate our research efforts. We discuss all features of our systems in detail, describe the visualization and interaction techniques as well as disadvantages and scalability issues if present.

Keywords: Information Visualization, Biological Networks, Multivariate Networks, Biological Visualization

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. *Information Visualisation, International Conference on,* 0:35 - 42, 2010.

2. Ilir Jusufi, Klukas Christian, Andreas Kerren, and Falk Schreiber. Guiding the Interactive Exploration of Metabolic Pathway Interconnections. *Information Visualization.*, SAGE Publications, 2011.

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

4. Ilir Jusufi, Klukas Christian, Andreas Kerren, and Falk Schreiber. Interactive Navigation in Interconnected Biochemical Pathways. In *Interactive Poster, InfoVis 10.*, Salt Lake City, Utah, USA, 2010.

5. 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.

Related refereed publications which are not part of this thesis:

1. 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.

2. 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 02.-06.03.2009 in Kaiserslautern, volume 150 of LNI, pages 147 - 154. GI, 2009.

Acknowledgments

Writing of this licentiate thesis would not have been possible without the support of many people who were there for me when I needed them the most. First of all, I would like to thank Prof. Dr. Andreas Kerren, who lured me into exciting world of the Information Visualization. I am grateful for his invaluable assistance, support and guidance.

Special thanks goes to Associate Professor Arianit Kurti without whom I would have never started this journey. I must also acknowledge Christian Klukas and Prof. Dr. Falk Schreiber who were extremely helpful and provided me with everything I needed for working in our joint projects. I want to thank Admirim Haliti, Björn Zimmer and Daniel Cernea for their support and for proofreading my thesis. I appreciate all the help you have given me. A lot of appreciation goes to all the colleagues from the DFM department for sharing their time and experience. I am sincerely thankful to all of my friends who gave me support during all this time.

My deepest gratitude goes to my parents and my brothers for their love, support and encouragement throughout my entire life. Last, but not least, I want to thank my wife, for countless hours she spent helping, editing and supporting me in my work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Complex and large data sets that describe relationships between objects are visualized in order to give insight into different patterns between relations of 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 relation or interlink between these elements. Visual analysis tools face issues related to huge, complicated and dynamic data sets and have to cope with different challenges, such as to increase people's understanding of the underlying data or to avoid clutter. A lot of these issues can appear even with smaller data sets as nodes and edges typically overlap each other, and this will make the interpretation of the graph almost impossible (cp. Figure 1.1).

Often, these networks are divided into many smaller parts in order to avoid this visual overload. For instance, if a large data set of all users of a specific social networks is presented, one might not be able to see anything as the data might be too big and highly interconnected, resulting in a over-cluttered image. A common practice is to divide this network into smaller subnetworks based on some determined criteria. This approach could help to understand the data better, as it avoids the overload, but introduces problems, such as loosing the overall picture.

Moreover, the combination of different types of networks is often desired. For instance, if we want to analyze communication patterns among workers in an organization in context of their job position, the we can analyze the email correspondence within the organization. The next step is modeling these data as a network where the employees could be represented as nodes and email correspondence (relationship) between two employees as a edge. By visualizing this network, we might get insight into different patterns of communications between different workers. However, companies have hierarchies (organization charts) based on the job position of the workers. This hierarchy is usually a tree, which is a special case of a graph. If we want to understand the communication of employees in context of the job position, we might want to visualize both networks to get the full context.

Another issue is that each network object may additionally have a number of attributes that are important to be visualized in context of the overall network presentation. Visualization of networks together with additional data is one of the ongoing challenges in various network visualization domains, such as biology, social network analysis or software engineering. If we return to the previous example of

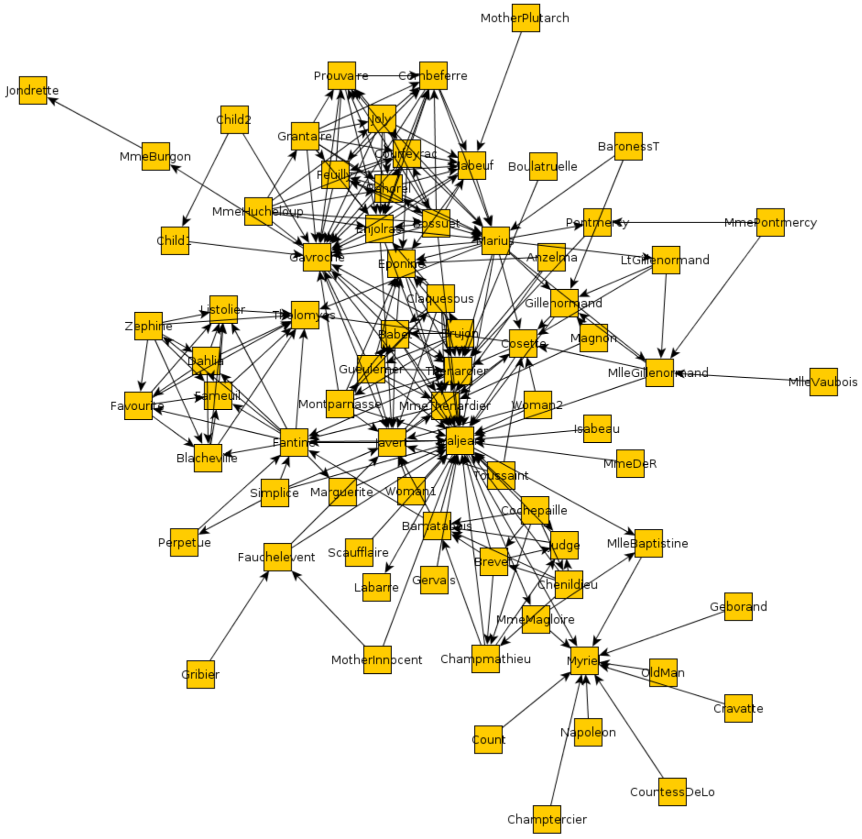


Figure 1.1: This figure represents the coappearance network of characters in the novel "*Les Miserables*" [50, 73]. The clutter appears in many parts of the drawing, where edges cross each other or go over nodes. This network has 77 nodes and 254 edges, while most of biological networks contain thousands of them. The image was produced by the *yEd* tool for graph visualization [133].

the email correspondence network, we will notice that each employee has its salary, 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 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 same gender or age communicate with each other more?

The next section continues with the motivation of this work, focusing on data produced by biologists. It explains the main problems that are addressed by this thesis.

1.1 Motivation

Due to advances in technology, we are facing an increase in biological network data, such as gene regulatory networks or metabolic pathway networks (Section 2.3). Hundreds of new elements are added and/or updated to existing biological data on daily basis. Therefore, online databases were created to help researchers to cope with this data [61, 36]. 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.

A standard practice to overcome the problem is to structure these networks hierarchically and divide them into smaller units as described before. This approach helps in understanding the function of smaller sub-networks and helps to reduce the overall complexity. However, it has the disadvantage that a user loses the context. Users will get insight into the processes of that particular sub-network, but they will not know how various elements are connected to the other sub-networks.

Analyzing biological network data sometimes involves hierarchical clustering of various experimental data that produces huge data sets of tree-like structures. Usually this clustering is performed on a subset of networks elements (nodes). This subset of elements is then "mapped" on a resulting cluster tree, as clustering in a loosest 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 huge network data and 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 case researchers are faced with two huge network visualizations.

Furthermore, these thousands of nodes are not simple one-dimensional data. Each one of the elements may contain many specific attributes. Biologists usually perform different experiments on the obtained data (biological networks in this case). These experiments produce additional data that are often important to be analyzed with respect to the underlying network structure. Therefore, it is important to visualize the additional attributes of the network while preserving the network structure as much as possible. The problem is not trivial as these networks could have a high number of attributes that could be related to their node, edges or different groups or clusters of nodes and/or edges. Even more issues arise if we consider the previous mentioned challenges as the visualization space becomes more expensive due to the complexity and amount of attribute and network data. Additionally, these experiments may produce (multivariate) time-dependent data, as various experimental measures are taken on a number of time intervals while the experiment lasts. These or similar challenges are also prominent in other domains beside biology, such as social network analysis or software engineering. Therefore, other scientific domains could benefit from the solutions of network visualization problems in biology.

1.2 Research Problems and Goals

The aim of this thesis is to prepare a way towards different types and techniques for the visualization of multivariate biochemical networks. In order to achieve this aim, an overall understanding of the problems related to biochemical networks should be acquired, since generally all these networks are multivariate networks. Therefore, besides an extensive literature review, a contribution to a solution of these challenges is necessary. With this point in mind, we define two main research goals:

1. Offer a contribution to the problem of a visualization of huge biologic networks. More specifically, improve shortcomings of the approach of dividing larger biological networks into smaller pieces, and contribute to the problem of a visualization of different types of interconnected biological networks.

2. Offer a contribution for the visualization of multivariate biological networks.

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 Dividing networks into smaller units in order to avoid clutter and complexity when visualizing huge biochemical networks introduces issues of loosing the overview. Nevertheless, this practice is desirable by the biologists. Therefore, an analysis of these issues and a contribution towards solving the problems shall take place.

1.2 Provide an approach to combine two different types of huge networks. As described in the motivation section on Page 3, a visualization of biological networks in context of tree-like data produced by clustering is desirable. Therefore, a tool to visualize both datasets and to show their connection will be developed.

The criteria for fulfilling our second goal are:

2.1 Provide a survey on the current state of the art on the visualization of multivariate networks in general. We shall not only focus on biological networks as the problems and solution from other research domains could be easily adapted for biochemical networks.

2.2 Introduce a novel approach for visualizing multivariate networks.

1.4 Discussion of the Research Approach

Here, we will briefly describe the approach to fulfill the aforementioned criteria, which will lead to achieving our presented goals and aims. Our first goal is more focused on visualization problems related to biochemical networks and not directly on the multivariate nature of the data. Working on this goal helps in understanding the overall complexity of the data and provides a good introduction into the domain problems. The second goal, focuses mainly on the multivariate nature of network data.

For the first criterion, we have decided to focus on an already known tool that uses the idea of splitting huge networks into smaller units [54]. This tool, called VANTED, is being developed since almost nine years and is widely used by biochemists. Other approaches exist to overcome the problem of huge and complex biochemical networks. This will be discussed later in this thesis. We focus on this particular approach as domain researchers understand the metaphor and are used to work in this particular way. The tool should be extended and/or improved to overcome the issues presented by the process of division of the networks. Our approach should provide an insight into the overall interconnectivity of the different networks.

For Criterion 1.2, we shall present an approach to combine two huge and different graphs. In our particular case, we will provide an approach to visualize graphs and binary trees. A prototype tool should be implemented to demonstrate our ideas. The tool should be able to visualize and handle the huge datasets in real-time. With this, we should finish our general contributions in terms of visualizing huge biological networks, respectively we will achieve our first goal. The following criteria will tackle the second goal, namely the issues of multivariate networks in biology.

As explained in the previous section, the state of the art survey about multivariate networks should be done to address Criterion 2.1. The aim of this survey is to provide a good understanding about the underlying problem and to classify different approaches into different categories, based on a number of specific features used in different approaches. This survey will serve as a reference point for fulfilling our last criterion and future work.

For our final criterion, we will present an improvement of an existing technique. The main idea of this technique 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 will be explained later in more detail. We will demonstrate a way to overcome this problem through the use of interaction techniques. Additionally, we will present a novel interactive way to create and use different filters for attributes through our prototype implementation.

1.5 Thesis Outline

So far the motivation behind this work has been described. The aims and goals of the thesis were described together with criteria and methodology to achieve them. The rest of this thesis is organized as follows. Related work briefly presenting the ideas on graph drawing and visualization is described in Chapter 2. Problems related to biological network visualization are discussed in this chapter as well. A brief state of the art survey on multivariate network visualization is presented in the next chapter where different approaches were categorized in different groups based on predefined criteria. The following three chapters describe different tools that were developed to address the problems mentioned in the previous section. A contribution towards solving the problem of the division of the large networks is presented in Chapter 4. A prototype tool that combines two different types of biological networks is described in Chapter 5. In Chapter 6, our contribution to visualize multivariate networks is presented. The thesis work is summarized, discussed and future work is presented in the last chapter of this thesis.

Chapter 2

Background Information

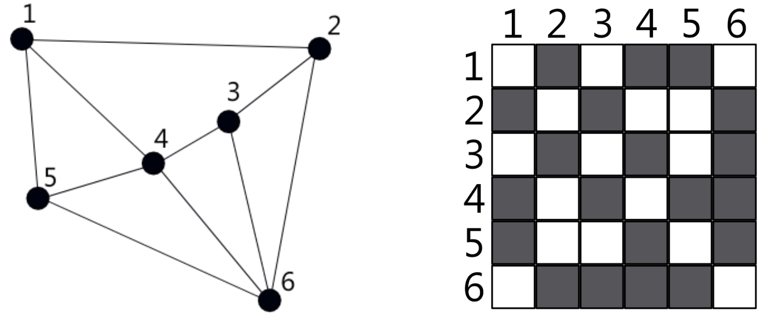
In this chapter, we briefly discuss related work in context of graph and biological network visualization. Initially, we present general ideas regarding the field of Graph Drawing in Section 2.1 and continue with common approaches and techniques for network visualization in context of Information Visualization (Section 2.2). Finally, we present the main challenges of the visualization of biological networks by discussing the current state of the art in dealing with huge and complex networks and presenting the key issues related to ontologies and clustering, two important concepts in life sciences.

2.1 Graph Drawing

Before diving into the exploration of the Graph Drawing discipline, it is worth noting that we distinguish between *graphs* and *networks* in this thesis. A (simple) *graph* $G = (V,E)$ consists of a finite set of vertices (or nodes) V and a set of edges $E \subseteq \{(u,v)|u,v \in V, u \neq v\}$. Whereas, a network consists of an underlying graph G plus additional attributes that are attached to the nodes and/or edges. 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)) [125] or space filling layouts for trees, which are a special case of graphs. 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 [44]. Layout algorithms 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 [20].

The graph readability is affected by quantitative measurements called *aesthetic criteria* [66], such as minimization of edge crossings, displaying the symmetries of the graph drawing, constraining edge lengths, etc. [23]. Thus, graph drawing generally deals with ways of drawing graphs according to the set of predefined aesthetic criteria [20].

Readability is also affected by *layout conventions* and *layout methods*. A layout convention is a basic rule followed by the drawing of graphs, such as *polyline drawing*, *straight-line drawing*, *orthogonal drawing,* etc [23]. Different types of



	1	2	3	4	5	6
1	□	■	□	■	■	□
2	■	□	■	□	□	■
3	□	■	□	■	□	■
4	■	□	■	□	■	■
5	■	□	□	■	□	■
6	□	■	■	■	■	□

(a) Node-link representation of a graph

(b) Matrix-based representation of a graph

Figure 2.1: Different visual representations of the same graph.

graphs usually use different methods to facilitate the drawing with respect to the desired criteria. For instance, if we want to draw *trees*, many layout algorithms are available. One class are *hierarchical layouts*, which place the nodes on horizontal layers according to their distance from the root node. Many algorithms are designed specifically for *directed graphs*, such as *layered drawing*, which produces polyline drawings. *Force-directed* algorithms are relatively simple to code and understand due to their physical analogy, which makes them pretty desirable for *undirected graphs* [66].

A good graph layout algorithm is important when doing any kind of network visualization. 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 [78].

Implementing good graph drawing algorithms is relatively complicated and time consuming. Therefore, quite a number of different open source libraries are available (for example JUNG [86] and Prefuse [42] among others) that deal with graph layout issues.

2.2 Graph Visualization

In this section, we focus on problems of graph visualization from an Information Visualization perspective. They differ in many aspects from those of the traditional Graph Drawing community. A nice overview on this subject is presented in the work of Herman *et al.* [45]. Usually what differs Graph Visualization from traditional Graph Drawing is the use of interaction techniques to overcome problems such as clutter in case of large network data. However sometimes, it is really hard to classify the approaches, especially when less or no interaction is involved.

Visualization Techniques

There are techniques that do not require interaction. For instance, there exist several approaches that try to manage the edge drawing in order to avoid any overlaps or clutter. One such technique is edge routing to avoid edges from overlapping nodes [26] beside the more traditional edge crossing removal algorithms, which are usually NP-hard [34]. Another approach bundles the edges together in a tree layout. It reduces edge clutter while giving insight into the relations of the nodes in the hierarchical data set [47]. The algorithm has been adapted to work on general graphs using force-directed layout algorithms [48]. A couple of other edge bundling approaches have been presented recently. Ersoy *et al* . [31] create edge bundles for general graphs, while Selassie *et al.* [103] present an approach to handle edge directions and weights.

Another technique to deal with huge data sets is to cluster the nodes. There are two main approaches: the first one, a more domain specific, is about clustering the nodes based on their domain specific content. Usually, distance values are calculated based on the node attributes and the nodes are clustered according to this values. This approach will be discussed later in this thesis in more details. The other approach is based on the structure of the graph. For instance, graph components that are strongly inter-linked among them in contrast to other elements of the graph are clustered together.

Certain approaches try to optimize the use of display space. Some tools use the hyperbolic geometry in order to use it more effectively [77, 84, 83]. 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.* [77] 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 interactions technique 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.

Another way to use the display space effectively is to use interactive 3D graphics. The most obvious advantage here is that we will get one more dimension to 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 [95, 130, 93]. After this period, 3D visualization lost a bit of popularity due to some of the issues this approach presents. One such a 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.* [124] presents more in depth analyzes of the 3D visualization problems and suggest that 2D, or 2.5D solutions are better.

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. Several other techniques, such as matrix-based or other space-filling approaches like Treemaps [53] or Sunburst [112], use the space more effectively. However, we will not explain these techniques any further, as we focus on node-link visualizations in this thesis.

Interaction Techniques

Yi *et al.* [132] give a great overview about the role of interaction in Information Visualization. Here, we shortly discuss some of the interaction techniques used in context of graph visualization.

One of the most common interaction techniques in Information Visualization are *dynamic queries*. Techniques such as range sliders could help a lot in reducing the visual clutter by filtering out unimportant data [109]. Most of these techniques could be applied directly to graph visualization. For instance, using a slider to specify the depth of the graph elements from a specified vertice to be shown.

Another common interaction technique that deals with huge data is *zooming and panning*. Large data often cannot fit into the visualization view, therefore 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. The problem with zooming is that the overview is lost when zoomed in. Therefore, several techniques are created to enable users to focus on a specific part of 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 produces an effect similar to that of a fish-eye lens [109, 9, 100].

The magic lens developed by Bier *et al.* [11, 113] is the most closely related work compared to our lens implementation discussed in Chapter 6. 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 authors describe the interactions of their tool as analogous to a real magnifying lens over a newspaper. Beside serving as a simple magnification 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. Thiede *et al.* [116] present a model to overcome this issue.

The magic lens is an approach that was originally not applied in terms of graphs. One example of using lenses in context of graphs, albeit with slightly different aim in mind, is EdgeLens [128]. 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. More recently, various lenses in context of graphs are presented by Tominski *et al.* [117].

A couple of studies suggest that magic lens-based techniques could be usable if combined with a number of other visualization techniques. The results from the work of Baudish *et al.* [9] 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 fisheye view interfaces in context of graph layout tasks was performed by Gutwin *et al.* [40]. Even though there are differences between competing fisheye 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, we briefly discuss different types of biological networks. In general there are six types of biological networks according to Junker *et al.* [55]: *signal transduction and gene regulatory networks*, *protein interaction networks*, *metabolic networks*, *phylogenetic networks*, *ecological networks* and *correlation networks*. Zhu *et al.* [134] present a slightly different classification of biological networks types, but we will bear on the work of Junker *et al.* [55].

Signal transduction and gene regulatory networks play an important role 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 [90].

Proteins are crucial in regulation of the majority of biological processes in living cells. 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 analyses 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].

A network constructed of metabolites and their biochemical reactions in an organism is denoted as metabolic network. 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 [108].

Any graph used to visualize the evolutionary relationship between species, genomes, chromosomes, genes, or nucleotide sequences is defined as phylogenetic network [51]. 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 [35].

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 [89]. Ecological networks show who is present and who affects whom by different interactions, such as feeding [33].

Dealing with Large Biological Networks

Various biological networks such, as metabolic networks or gene regulatory networks, are significant components in biological process analysis. They are essential for an overall understanding of living beings. The constant progress of knowledge and technology has made the process of acquiring these network data fast. 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 [61], which contains hierarchically structured pathways with in total more than 10,000 nodes representing genes, proteins/enzymes, and metabolites.

Several systems and methods have been developed to cope with such large and complex networks. Many tools and databases for visualization and analysis of biological networks are available on-line. Systems such as CellDesigner [32], Cytoscape [104], ONDEX [74], Pathway Projector [76], PathVisio [120], VANTED [54], and VisANT [49] 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.

These approaches are not scalable and introduce a lot of clutter. Therefore, a common practice in life sciences is to brake down the networks into overlapping pathways. Often these pathways are structured hierarchically, for example, in metabolism several pathways representing the synthesis of amino acids are grouped into a super-pathway *amino acid metabolism*. This practice of separating networks into pathways helps in reducing the overall complexity, however it introduces a drawback as the user looses the context of interconnectivity to the network in general. This becomes even more difficult as many network elements are often multiplied several times in different pathways as a result of a division process. An example of such an approach is the Web-interface of KEGG [61], which allows the navigation from one pathway image to another connected pathway image by clicking on a link (Figure 2.2). In this way, following the connections to elements in other pathways is rather hard.

KGML-ED [71] is one of the approaches that tries to improve the navigation between pathways by showing an overview of the top nodes of the hierarchy with an option to zoom into such nodes or by extending the pathway by connected pathways within the same frame. This approach produced the same drawback as the previous one. Namely, too much information (e. g., many or all pathways at once) was shown which affected the readability, or the connections to other pathways were lost or difficult to follow. Another tool that follows a similar approach was presented by Bourqui *et al.* [15]: MetaViz manages to visualize the complete network and specific pathways at the same time without a need to multiply the network nodes. Networks are drawn using one algorithm, while focus pathways are drawn using different drawing algorithms. These sets of focus pathways are drawn independently adhering to domain specific requirements and criteria. Additionally, the links between these pathways are visualized using an orthogonal graph drawing algorithm. It produces long orthogonal edges, which are hard to follow, especially in case of large networks. Therefore, users often need to zoom-in into one or more of the focused pathways to see more details. This brings back the issue of loosing the context of interconnections. To the best of our knowledge, biologists usually have no problem with node redundancies. Recognizing “their” pathways (based on the graph embedding of that pathway) during the exploration process is more interesting to them.

Node duplicates are also used in other domains beside biochemistry. One example of such a use in context of clustered social networks is presented by Henry *et al.* [43]. The aim of duplicates was to avoid clutter in their graph visualization. The authors discuss different cases when and how to duplicate the nodes. A controlled experiment helped them in defining general guidelines for node duplications in clustered graph representations. An example of such a guideline is that the interactive highlighting of duplicated nodes and links is desirable. This particular guideline is important to our work too, and its implementation can be found later in this thesis.

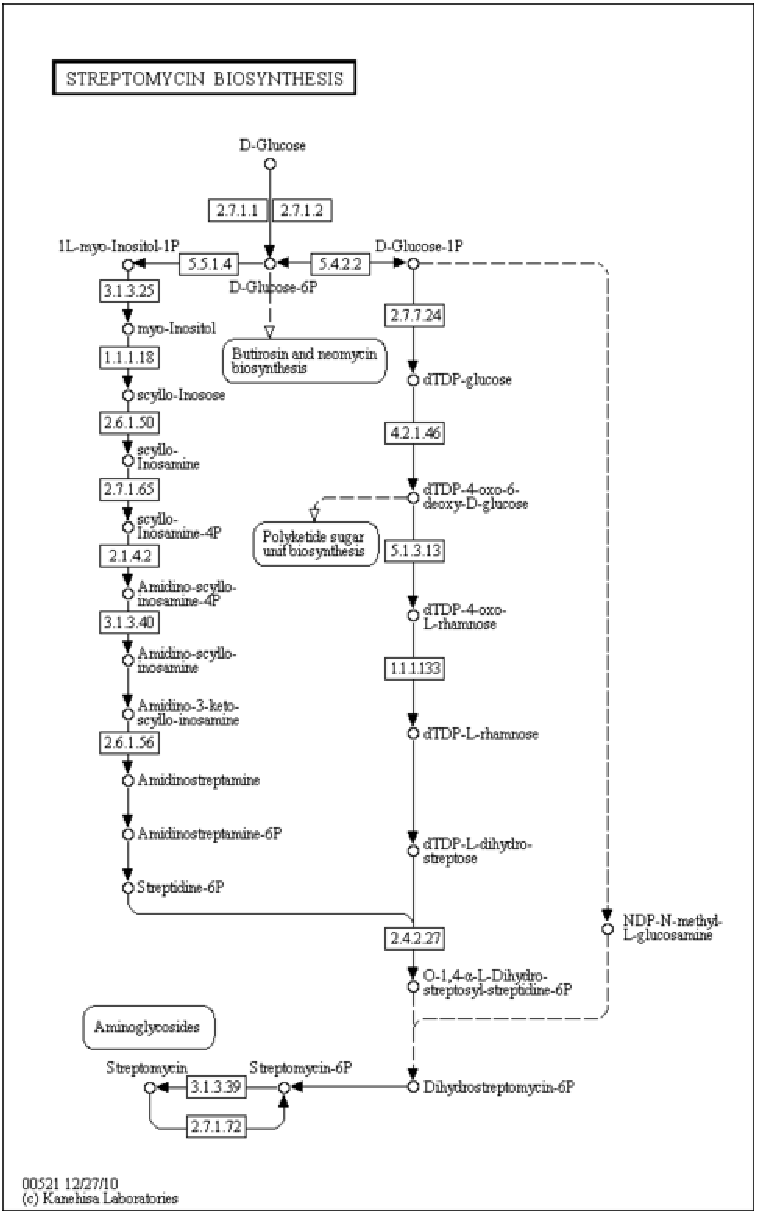


Figure 2.2: The diagram shows an example of a KEGG reference pathway. The image represents the *Streptomycin biosynthesis – Reference pathway* [1].

Current approaches are mostly based on a close interdisciplinary collaboration between visualization or graph drawing researchers and domain experts, such as in biochemistry. Researchers from life sciences have a set of specific criteria in the way the networks should be drawn. A good reference about it could be found in the work of Saraiya *et al.* [99]. The authors performed a series of interviews with domain experts and discussed the findings related to the requirements for biochemical network visualization. The results of the interviews revealed five critical needs that are important to researchers working on pathway analysis and are still not fully realized by existing visualization tools. We will later present a tool that realizes a 2D solution for the need to overview “multiple pathways simultaneously with interconnections between them” [80]. Specifically, “incoming and outgoing visual links could enable users to view how other pathways can potentially affect or be affected by the focus pathway at each node. In a densely populated pathway, it is important to be able to analyze connectivity between components” [99]. The work of Albrecht *et al.* [3, 4] provides a good overview on open problems and challenges in biological network visualization.

A similar project is the Caleydo framework [79] that extends the KEGG pathways drawings into 2.5D, comparable to the work of Kerren [65] or Brandes *et al.* [16], combined with brushing, highlighting, focus + context, and detail on demand. In this way, it supports the interactive exploration and navigation between several interconnected components by using the third dimension. There are some drawbacks of this approach. One drawback is the restricted number of connected pathways that can be displayed because of the bucket-like visualization setup. Another one is the unavoidable clutter/overlaps in 3D if a node-link metaphor is used.

A grid-based visualization approach for metabolic networks supported by a Focus + Context view was recently presented by Rohrschneider *et al.* [96]. This view is based on a Table Lens method [92], which provides multiple foci. Together with the grid-layout, the user’s mental map [81] is also preserved. But even by using these plethora of interaction and navigation possibilities the cognitive overload is still high and the approach follows a classical top-down navigation method. Links to other pathways must be followed successively in case the analyst starts the exploration from a single node. In this thesis, we present a technique which strives to improve this situation. Because it is not the focus of this thesis, we do not give a comprehensive overview of other related tools and approaches and refer to the papers [4, 96, 45] instead. These articles present an extensive list of related work (not only focused on pathway interconnections) and consider navigation and interaction techniques as crucial in network visualization.

Ontologies and Clusterings

Biology and medicine researchers use ontologies to structure biological knowledge. Ontologies can be defined as a set of controlled, relational vocabularies of terms used in a specific area of science. In life sciences, ontologies are used to structure and standardize biological knowledge to support data integration and information exchange. Examples are Gene Ontology (GO – to standardize gene and gene product attributes across species), Molecular Interactions 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) [22], which provides a web service to query multiple ontologies from a single location, providing a unified output format. Often experimental data is analyzed in context of biological ontologies, for example, by means of enrichment of ontology terms to identify statistically over-represented (inner) ontology terms.

In this thesis, we will mainly focus on the Gene Ontology (GO) [36]. GO is an on-line 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” [36]. 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) [6, 114].

Large-scale experimental data in life sciences are often analyzed and visualized using hierarchical clustering [27]. 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 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.

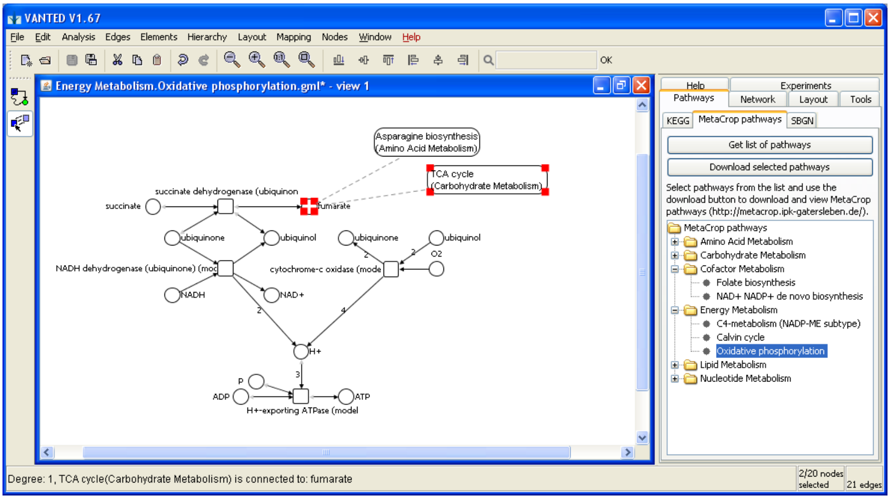
The analysis of molecular-biological data obtained by high-throughput technologies is often supported by ontologies and hierarchical clustering. These high-throughput technologies produce data constantly. This enables the investigation of different biological data or systems under different conditions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

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 analyses 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 that deal with data that are part of two trees or hierarchies or compare two trees. However to our knowledge, no solution exists for visualizing hierarchical clustering (tree) and an ontology (DAG) of the same dataset in one visualization system.

Next, we will discuss a specific framework that deals with visualization of biological networks. We focus on this system as it is central to the development of one of our ideas presented in this thesis.

The VANTED System

VANTED [54, 72] is a flexible system that helps researchers analyze and visualize experimental data in the context of biological networks or pathways. It is based on the extensible graph library and editor Gravisto [8] for graph drawing algorithms. What makes VANTED flexible is that it provides a mechanism for easy extension by plugins. Several plugins were developed, such as a plugin to compute and visualize fluxes in metabolism or to connect databases to VANTED.


VANTED V1.67
File Edit Analysis Edges Elements Hierarchy Layout Mapping Nodes Window Help
OK
Energy Metabolism.Oxidative phosphorylation.gml* - view 1
Asparagine biosynthesis (Amino Acid Metabolism)
TCA cycle (Carbohydrate Metabolism)
succinate dehydrogenase (ubiquinon
succinate
fumarate
ubiquinone
ubiquinol
ubiquinone
ubiquinol
2
2
O2
NADH dehydrogenase (ubiquinone) (mod
cytochrome-c oxidase (mode
NADH
NAD+
2
4
H+
3
P
ADP
ATP
H+-exporting ATPase (model
Help
Experiments
Pathways
Network
Layout
Tools
KEGG
MetaCrop pathways
SBGN
Get list of pathways
Download selected pathways
MetaCrop pathways
Amino Acid Metabolism
Carbohydrate Metabolism
Cofactor Metabolism
Folate biosynthesis
NAD+ NADP+ de novo biosynthesis
Energy Metabolism
C4-metabolism (NADP-ME subtype)
Calvin cycle
Oxidative phosphorylation
Lipid Metabolism
Nucleotide Metabolism
2/20 nodes selected
21 edges

Select pathways from the list and use the download button to download and view MetaCrop pathways (http://metacrop.ipk-gatersleben.de/).

Degree: 1, TCA cycle(Carbohydrate Metabolism) is connected to: fumarate

Figure 2.3: Original navigation approach between pathways provided by VANTED. Shown is a metabolic pathway with circles representing metabolites and rectangles representing reactions. The large rectangles with rounded corners represent links to other pathways. The background coloring to show different cell compartments was switched off to facilitate the identification of the different graphical elements by the reader.

Figure 2.3 shows the standard way of navigating between interconnected hierarchical pathways in VANTED. A pathway hierarchy is displayed on the right hand side in which the *oxidative phosphorylation* pathway is selected and shown on the left. This pathway is a child of the *energy metabolism* super-pathway and relatively small. One node (*fumarate*) within the pathway drawing is in the focus of the user. All links to other pathway nodes connected to *fumarate* can be shown with a mouse click. In this particular example, we may notice two rounded rectangles connected to the focused node using dashed gray lines. Each rounded rectangle represents another pathway. If the user wants to navigate to the *Aspargine Biosynthesis* (part of the *amino acid metabolism*), he/she has to click on the corresponding rounded rectangle to open a new window with the target pathway displayed. All subsequent navigation is performed in similar manner.

Due to the unavoidable clutter if the pathways are large and many nodes are selected, it is easy to understand that this navigation strategy should be improved. Additionally, at this stage, it is not possible to get insight into the overall distribution of the links as well as get an overview of all the pathways linked to the currently selected node(s).

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 on different types of biological networks is discussed followed by the current approaches to visualize such networks. We conclude this chapter with description of the VANTED system, which is crucial to the implementation of one of our approaches presented in Chapter 4. In the next chapter, we will discuss and classify various techniques that deal with multivariate network visualization.

Chapter 3

State of the Art of Multivariate Network Visualization

It is known that the amount of data produced in the world is presenting 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 software engineering data. They can be represented using graph metaphors: graphs help us to understand the structure and topology of relational data. However, these graphs have usually a huge amount of additional attributes related to nodes, edges or even attributes of clusters. The challenge is to show these attributes in context of the underlying graph topology. This problem is becoming more and more important as a number of researchers need solutions for their daily work.

In this chapter, we present a brief survey on multivariate network visualization tools and approaches. We classify the approaches into several categories based on the used visualization techniques and discuss the issues of domain and attribute related issues. We provide a table listing a number of approaches/tools that gives insight on the technique, nature of attributes and the domain the tool is primary used for.

3.1 Multivariate Networks

Multivariate Data

Before describing what a Multivariate Network is, we will make a brief discussion about Multivariate Data Visualization as most of the visualization techniques can be reused for Multivariate Networks. The term multivariate data in Information Visualization is used to describe data that contain more than three attributes or variables [109].

There exist a considerable number of techniques to visualize multivariate data. 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 [41] (cp. Figure 3.1 – B). Coordinate axis-based approaches map the attribute values into different coordinate axes.

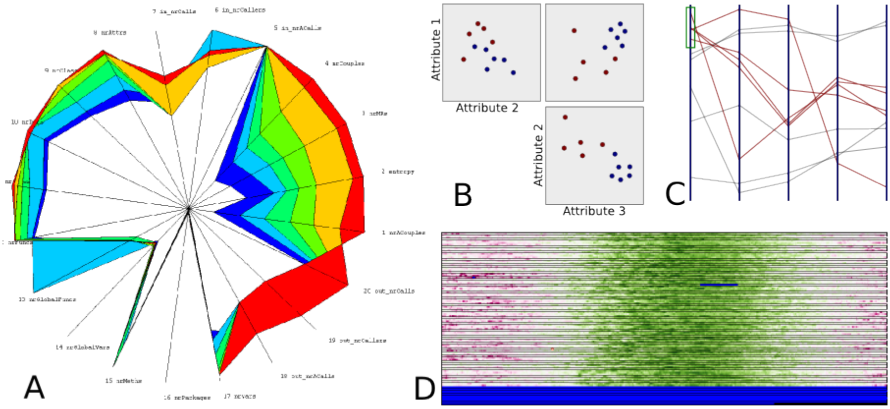


Figure 3.1: Samples of multivariate data visualization techniques. The image marked with label A, is made with a tool presented by Kerren *et al.* [67, 68] based in the original work of Pinzger *et al.* [87]

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 [21]. And last but not least, pixel-based approaches map attribute values to single pixel (cp. Figure 3.1 – D) [109] .

Various interaction techniques provide additional help for coping with 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 dataset. 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

Huge networks present a great challenge to be visualized even without any other additional information. However, real life networks carry additional attributes belonging to the data elements themselves, attributes describing the relation between the data elements and/or attributes related to a cluster of elements.

Let us take an example from the social networking. Each person can be modeled as a node, and the friendship between persons as an edge. 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.

In many fields, such as in Biology or Social Network Analysis (SNA), different algorithms are used to analyze the networks. These algorithms may produce additional data, such as network centrality metrics. These data can be regarded in the same way as node, edge or cluster attributes and visualized using similar approaches. However, some tools are designed to specifically cope with these types of data [25, 70].

3.2 Visualization Approaches

We have grouped several visualization approaches with respect to the way the visualization is made.

(A) Allow use of standard graph drawing algorithms (maximize the perception of network topology)

  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 (hinder the perception of network topology)

  B.1 Nodes are positioned in specific non-overlapping regions

  B.2 Nodes are positioned in specific positions and/or regions

The top level criteria (A and B) for creating the groups focus on the ability of the approaches to show the underlying graph topology. Some 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. Criterion (A) has two sub-criteria, which are related to the way the attributes are visualized; 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). There are also systems that use a combination of several approaches that satisfy more criteria as well as approaches that could be placed in each of the sub-criteria. In the following, we will present and discuss the groups we derived based on these criteria.

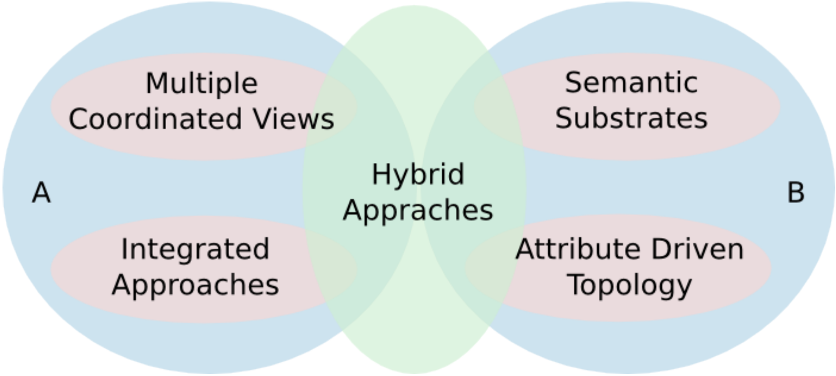


Figure 3.2: Classification of approaches. The left blue circle (A) represents the approaches that are able to represent the underlying network topology, in contrast to the right blue circle (B).

Multiple Coordinated Views use a combination of two or more combined 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. *Semantic Substrates* position the nodes into separate non-overlapping areas based on the node attributes. *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.

We have done an analysis of strengths and weaknesses of each approach separately which is explained in more details later in the chapter. Table 3.1 presents different articles and tools we reviewed, while Figure 3.2 shows how these approaches are used to classify them in respect to the criteria.

Multiple Coordinated Views

The combination of several views into a single system presents one possible solution to the underlying problem. In such cases, the use of brushing and similar techniques is necessary as the users should experience a seamless interaction among different views. 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 a clear advantage using this approach as one may choose the most powerful visualization techniques for each specific view and data set [38, 94]. However, due to the spatial separation of the visual elements, the displayed data will be split in multiple views. In addition, 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.

			Attributes			Approaches						
							Integrated					
Paper / Tool	Domain	Heterogeneous Data	Node	Edge	Cluster	Multiple Coordinated Views	Node	Edge	Cluster	Semantic Substrates	ADT	Hybrid
Multivariate Graph Drawing using PCV [105]	General		●			●						
Network Lens[56]	General	●	●				●					
DBE Information System [13]	Metabolic Networks		●				●					
GraphDice [10]	Social Networks		●	●	●						●	
GeoSOM [131]	General		●								●	
Jigsaw [111]	Document Analysis		●			●						
RelVis [87]	Software		●				●					
NVSS 1.0 [107]	General		●		●					●		
PivotGraph [126]	General		●							●		
Pretorius et al. [91]	General		●	●	●					●		
MobiVis [106]	Mobile data		●		●	●	●		●			●
Dynamic Graph Analysis [97]	Metabolic Pathways		●	●		●	●	●				●

Table 3.1: Multivariate Approaches. The table shows information about different approaches for visualizing multivariate networks. The *Domain* is shown through its column. *Heterogeneous Data* column shows whether the tool visualizes only heterogeneous attributes. There are three sub-columns under *Attributes* column denoting what kind of attributes are visualized by the corresponding tool in respect to the network elements. *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 has the attributes visualization been embedded to.

The work of Shanon *et al.* [105] presents one example of this idea in the network visualization domain. They use two distinct views: one view shows a parallel coordinate approach [52], and the other view displays a node-link drawing of a graph. Their tool offers a variety of visualization and interaction techniques, while maintaining a coordination between the views through brushing and linking [109].

Another example of multiple coordinated views is Jigsaw, a tool for document analysis [111]. Users can model views of relational data of different documents and entities as a graph representation while using additional views to provide more information about the network. Although the tool is primarily designed to visually explore the relationships among different elements of data, it provides possibilities to visualize additional data. For instance, users can view the text of a document in a separate view, or they can view 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 multiple monitor environments or on large, high-resolution monitors.

Integrated Approaches

Contrary to the first approach, 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.” [38]. 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.

Integration of up to four or five attributes in a node is rather straightforward, of course depending on the data being visualized. We can use labels for textual attribute and other different features, such as size, shape, color and stroke for other attributes. As the number of attributes grows, we face more problems as the number of visual features comes to an end. We might need to introduce new visual metaphors to cope with the increasing number of attributes.

One 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 3.3). This approach provides a view to all available information, but with additional costs. The size of the nodes in the graph needed to be enlarged in order to facilitate the employment and the readability of the diagrams. 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 [66]. Thus, it does not scale well.

The work of Pinzger *et al.* [87] is another example of integrating well known multivariate data visualization techniques into nodes. The authors use an improved

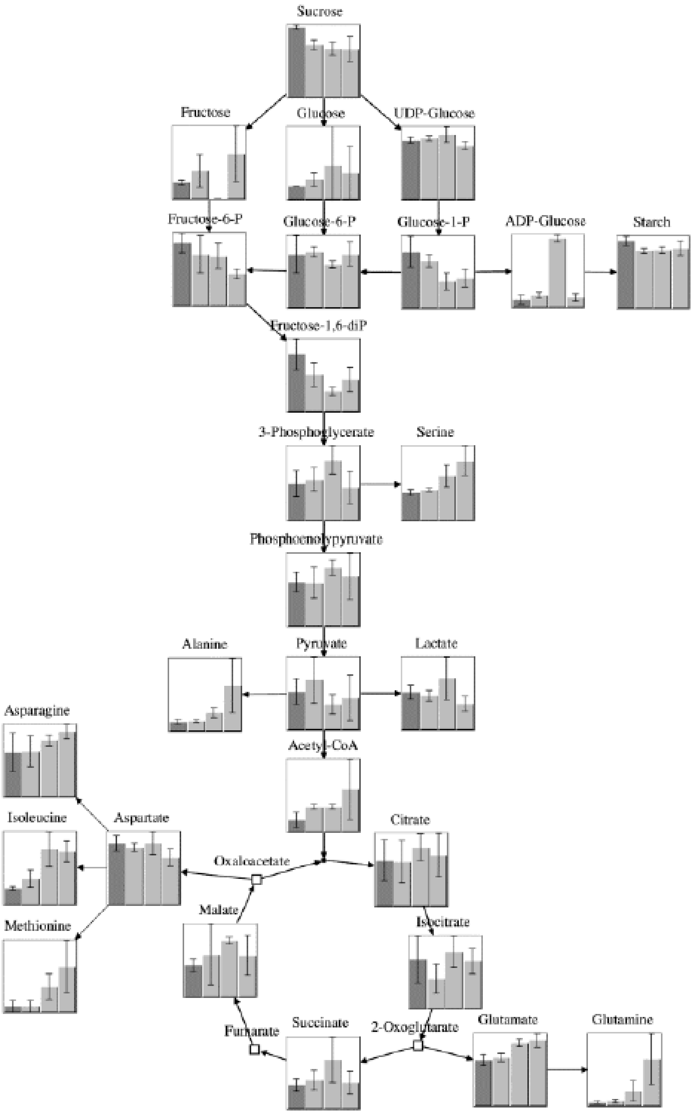


Figure 3.3: An example of Integrated Approach shows experimental data integrated into a biological network. The picture was taken from [13] showing the relative levels of different “Vicia narbonensis” lines integrated into the glycosis and citric acid cycle. “Image courtesy of IPK Gatersleben.”

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 using different focus + context techniques. An example of such an approach visualizes the attributes inside nodes 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 [56].

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 [123].

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” [107]. The nodes were placed in specific substrates (regions) based on specific semantics derived from the nodes' attributes. Additionally, Shneiderman and Aris used sliders to control the edge visibility which ensured the comprehensibility of the edges end nodes.

PivotGraph [126] 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 [91]. They arrange edge labels in a list located in the center of the view and place rectangular regions containing source and target nodes at each side. These regions are recursively partitioned according to the node attributes resulting in specific positioning of nodes based on the created hierarchy. The nodes are then connected via straight lines with corresponding edge labels. One conceptual drawback of these approaches is that the underlying graph topology is not (completely) visible.

Attribute Driven Topology

The graph layout is an important part of network visualization and analyzes, 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 [78]. In cases where network topology is not the key component in the analyses, one could use the network layout to present insight about multivariate data belonging to network elements. This approach is somewhat similar to semantic substrates, however it does not necessary place the nodes into specific regions. It uses some calculations based on the nodes attribute to ”steer” the placement of the node in the graph.

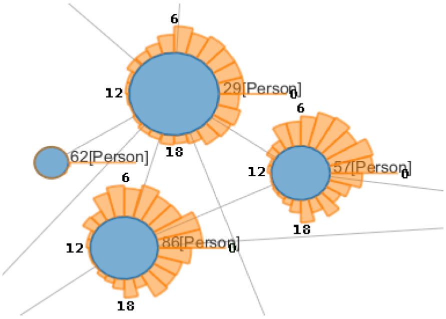


Figure 3.4: An hourly ring of calls shows the frequency of calls every hour. The picture was taken from [106] with permission of the authors.

One example of such a tool is GraphDice [10] which extends the ScatterDice [28] tool used to visualize multidimensional data through multiple scatterplots. The system uses multiple scatterplots made of various node, edge or computed attributes to lay out the graph. Each point in the scatterplot represents a data object and a line 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 lay out the graph on the surface of a sphere. A SOM is an artificial neural network used as dimensionality reduction technique [75]. 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 the use of boundaries as they can not show the relationships between data placed to the borders of the 2D plane [131].

Hybrid Approaches

The approaches belonging to this category combine at least two of the discussed techniques. Hybrid Approaches are designed to harvest the benefits of other approaches. The most common combinations are multiple coordinated views with any of the integrated approaches.

For instance MobiVis [106], 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 3.4). 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 [97]. At the same time, it uses other visualization metaphors creating multiple coordinated views to show the time related data of the network. Users can explore the evolution of the multivariate data as well as of the overall metabolic pathways through series of different interactions.

3.3 Summary

Initially a general multivariate data visualization outside the network visualization context was discussed briefly, as a number of techniques could be applied directly to multivariate network visualizations. The approaches were categorized based on the predefined criteria, which take into account the ability of the approaches to show the topology and the way the additional network attributes are visualized. This criteria helped us to define four distinct groups and a fifth hybrid approach.

Chapter 4

Interconnected Network Visualization

In the last few years, methods for the investigation of biological processes, such as metabolism, the regulation of genes and the interaction of proteins have been of interest. These processes are typically represented as biological networks. They are important for an extensive understanding of living beings. For instance, when modeling a metabolism as biological network, nodes are used to represent metabolites and reactions, and edges connecting metabolites with reactions and reactions with metabolites represent the ways metabolites can be changed into each other via particular reactions. As explained in Section 2.3, the state-of-the-art strategy is to divide a metabolic network into a large set of single (often hierarchically structured) pathways. When visualizing these pathways, additional exploration and navigation problems appear, since the user loses the context of the entire network. Identical complications can emerge in other domains, such as in the visualization of social networks, communication networks, etc.

In this chapter, a general approach for navigating between interconnected subgraphs is presented. We demonstrate the approach in context of biochemical networks, respectively in a set of hierarchically structured pathways (groups and subgroups of pathways). In contrast to common practice, our visualization approach enables the researcher to obtain an overview of pathways, which are connected to the focus pathway via its nodes and edges. This contextual information is provided by using glyphs, brushing, and topological information of the involved pathways. Our interactive visualization tool is able to guide the exploration and navigation process based on this overview. In this way, it can help the analyst to perform the analysis processes more efficiently. We implemented our approach as plugin for the VANTED system, as we wanted to test our approach with real data, current networks and concrete problems. VANTED was originally developed for the visualization and analysis of experimental data in the context of biochemical networks (cp. Section 2.3).

This chapter is based on the two publications [60, 59]. For background information and related work concerning the biological as well as visualization aspects, please refer to Section 2.3. The underlying VANTED system is briefly discussed in Section 2.3. The rest of this chapter is organized as follows: the description of our novel navigation and exploration approach is given in Section 4.1. Here, we also provide an overview of limitations and implementation issues. A use-case scenario is discussed in Section 4.2 followed by a summary.

4.1 Novel Approaches to Guide the Exploration Process

For demonstrating our visualization approach, we use the MetaCrop database [39] that summarizes diverse information about metabolic pathways in crop plants and allows the automatic export of information for the creation of detailed metabolic models. MetaCrop can be directly accessed by VANTED and is based on the freely available Meta-All system [127]. In MetaCrop, metabolites are shown as circular nodes; proteins (enzymes or transporters) are represented as rectangular nodes. Figure 4.3 shows a screenshot of our tool. The background coloring in the navigation window is used to visually represent different cell compartments (different parts within the cell). Additionally, this allows us to restrict links to substances represented in multiple pathways to only those cases, where the substance name and the particular compartment information is matching.

At the time of writing this thesis, our tool fully supports *MetaCrop pathways*, containing six groups *(Amino Acid Metabolism, Carbohydrate Metabolism, Cofactor Metabolism, Energy Metabolism, Lipid Metabolism, and Nucleotide Metabolism)* consisting of 38 pathways in total. Thus, the MetaCrop hierarchy has two levels only: the first level belongs to the groups (super-pathways) and the second level represents the corresponding pathways. There are two important aspects related to this dataset that we used as guide for designing our tool. The first aspect is the relatively small number of pathways, and the second aspect is their more or less distinct graph topology which makes these graphs easy to identify by professionals (depending on the graph layout).

Navigation Glyphs

Initially, we wanted to give insight into the interconnections of each individual element (node) of the currently focused pathway. We have done this by developing small *navigation glyphs*, which are embedded within the network nodes. They show the links of the specific node to other pathways (cp. Figure 4.1). Our rose-shaped diagram resembles *Nightingale's rose diagram* [85] (see also the DataMeadow [29] approach). However, our approach is extended to support the representation of the MetaCrop pathway hierarchy. The work of Shen *et al.* [106] is also similar. But, instead of links, "glyph petals" size shows to the value of particular attributes. Similarly, in the work of Elmqvist *et al.* [30] colored polygons visualize the strength and direction of causal relationships.

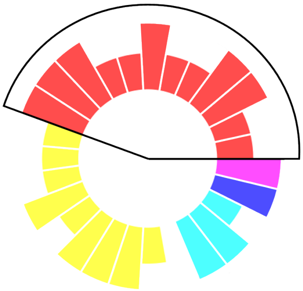


Figure 4.1: Navigation Glyph. The highlighted sector represents all connected pathways belonging to one super-pathway (or pathway group), e.g., *Amino Acid Metabolism*.

In our approach, the size/length of the “glyph petals” shows if there is a link (to the same node in another pathway) or not. The color coding represents the pathway group (Level 1 in the hierarchy). Consider the sample glyph in Figure 4.1 that is embedded into a particular node in a pathway. This specific node is connected to five other super-pathways (pathway groups) represented by five different colors. For color coding the super-pathways, we follow the suggestion of Ware [125, p. 126] for using a subset of standard colors for nominal data. We have manually highlighted a part of the glyph in Figure 4.1. The highlighted area gives us insight about the strength of the connectivity to a specific pathway group (*Amino Acid Metabolism* in this case). This area is divided into 12 petals. They represent all pathways belonging to this group (super-pathway). Six of these petals appear longer/extended. They represent the six other pathways where this element is found as well, i.e., there are links to six sub-pathways of the current group in the second level of the hierarchy. The missing petal (the gap on the lower part of our example) appears if the VANTED system discovers a pathway group that is completely unreferenced by the currently considered node.

The original VANTED approach is improved with the help of navigation glyphs by providing a simple overview of the relationships to connected pathway groups outgoing from a single node. A right click on a node displays a context menu with a list of links to all connected pathways to which users can navigate to (cp. Figure 4.2). The idea is similar to the Bring & Go approach presented by Moscovich *et al.* [82], however without link sliding or smooth animations. In VANTED’s case, a specific window is directly activated after a click on a specific link. This means that the navigation is analogous to the old approach explained earlier, having similar disadvantages, such as losing a direct link to the previous pathway window: in order to return to the previously shown pathway, we need to minimize the newly opened window and find the first one, or directly access the node that has a link to it. Moreover, due to missing interactive mapping of the petals to the connected pathways we have no clear insight into all links of the current pathway.

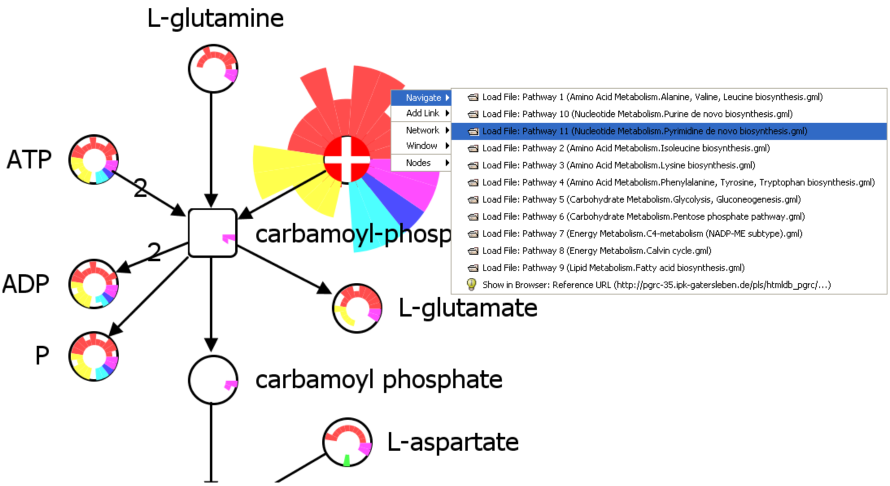

L-glutamine
ATP
2
2
ADP
P
carbamoyl-phosp
L-glutamate
carbamoyl phosphate
L-aspartate
Navigate
Add Link
Network
Window
Nodes
Load File: Pathway 1 (Amino Acid Metabolism.Alanine, Valine, Leucine biosynthesis.gml)
Load File: Pathway 10 (Nucleotide Metabolism.Purine de novo biosynthesis.gml)
Load File: Pathway 11 (Nucleotide Metabolism.Pyrimidine de novo biosynthesis.gml)
Load File: Pathway 2 (Amino Acid Metabolism.Isoleucine biosynthesis.gml)
Load File: Pathway 3 (Amino Acid Metabolism.Lysine biosynthesis.gml)
Load File: Pathway 4 (Amino Acid Metabolism.Phenylalanine, Tyrosine, Tryptophan biosynthesis.gml)
Load File: Pathway 5 (Carbohydrate Metabolism.Glycolysis, Gluconeogenesis.gml)
Load File: Pathway 6 (Carbohydrate Metabolism.Pentose phosphate pathway.gml)
Load File: Pathway 7 (Energy Metabolism.C4-metabolism (NADP-ME subtype).gml)
Load File: Pathway 8 (Energy Metabolism.Calvin cycle.gml)
Load File: Pathway 9 (Lipid Metabolism.Fatty acid biosynthesis.gml)
Show in Browser: Reference URL (http://pgrc-35.ipk-gatersleben.de/pls/htmldb_pgrc/...)

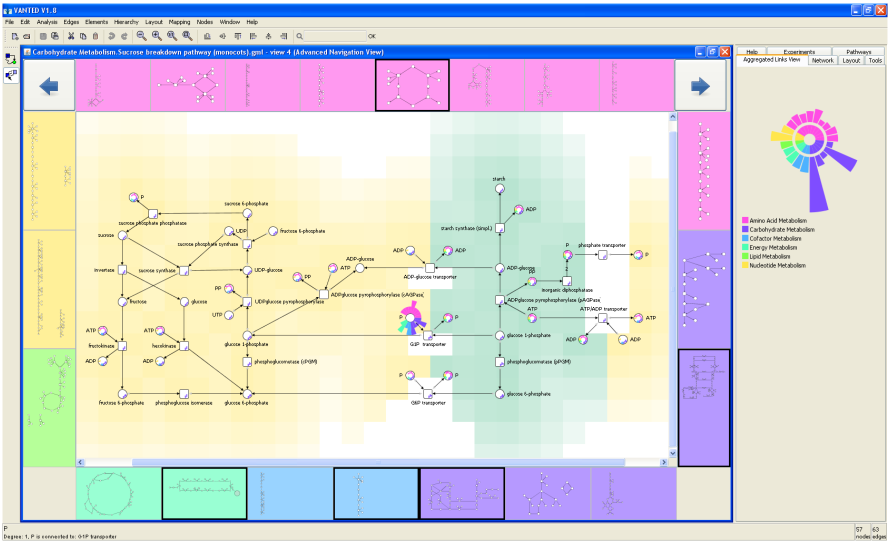
Figure 4.2: Embedding of our navigation glyphs into some nodes of a (focus) pathway. The glyph of the currently selected node is highlighted and magnified by the system. With a right mouse-click the user is able to open a context menu with a link list from which they can jump to connected pathways. The target pathway is then displayed in a new window.

Extended Navigation Approach

Our aim is to overcome the drawbacks of the navigation glyphs and the general VANTED approach presented earlier by

1. providing an overview of all connected pathways of the current focus pathway,
2. showing statistical information about the focus pathway's link distribution, and
3. facilitating the navigation from nodes in the focus pathway to different target pathways.

In order to achieve this, we present new visualization and brushing techniques combined with seamless interaction of various parts of the system. These improvements will make the navigation of different pathways easier and result in a more intuitive exploration experience. Additionally, we have addressed the need to show more insight into the links from a specific pathway by enabling users to compare the total number of references to all connected pathways.



VANTED V1.8

Carbohydrate Metabolism.Sucrose breakdown pathway (monocots).gml - view 4 (Advanced Navigation View)

P
Degree: 1, P is connected to: G1P transporter

57 nodes 63 edges

Figure 4.3: Overview of the new VANTED plugin with the *navigation window* on the left and the *aggregated links view* on the right. The background coloring of the shown focus pathway represents different cell compartments. One node is selected and its navigation glyph is magnified by the system. The glyph petals are directed to the surrounding graph icons. Links to other pathways are double-coded by the longer petals as well as by the highlighted graph icons. The *link rose diagram* in the aggregated links view (on the right) is explained in Section 4.1.

Navigation Window Initially, we introduce *graph icons*, which are minimized images of pathway drawings. The main idea is to lay out graph icons around the main view that contains the currently considered focus pathway, see Figure 4.3. Both together form the *navigation window* of our VANTED plugin. Navigation has become a matter of a single mouse click on a specific graph icon. This action would load the target pathway into the navigation window. Each graph icon's background color has been color mapped according to the color coding in the navigation glyph embedded in the pathway nodes. This makes it easy to distinguish the different pathway groups of the MetaCrop data set. Our graph icons are conceptually similar to the approach presented by Auber *et al.* [7] or to the subtree icons in the SpaceTree tool [88].

Based on the given graph topology in the graph icons and with additional help of the aforementioned color coding, users are able to recognize the pathways connected to the nodes of the focus pathway. As our target dataset is relatively small, users can directly identify well-known pathways with a recognizable and unique layout, such as the *TCA cycle*. The pathway layout can be computed with special algorithms, which take additional pathway information into account [62, 101, 102], or the layout is given (e.g., by information in the database). If certain pathway layouts are similar and hard to identify, a coloring of the groups can help. However, this will not be helpful in case both pathways belong to the same group. Therefore, the user can use tooltips for precise information about the pathways showing the name of the pathway and of the pathway group. A mouse-over event of a specific graph icon will show a tooltip with this information.

The layout of the graph icons is calculated to roughly match the positions of specific petals within the small *navigation glyphs*, which in turn provides additional insight to this existing feature. This layout design was chosen as users already use the navigation glyph metaphor. So, our new approach complements the old one aimed at specific pathway nodes as discussed in Subsection 4.1. An alternative to this would be a simple link list, but that might require scrolling efforts to fit all the links and does not provide a mapping of the pathways with our navigation glyphs. We have also implemented an additional visualization view (the aggregated links view) that uses the same layout and navigation concept and provides an integrated tool for link aggregation to facilitate the guided interactive exploration. We will explain this view later in this chapter.

Graph icons enable the navigation through all related pathways in one window, while providing a good overview of all pathway links. Through a mouse-over, which will highlight the corresponding graph icon, users can easily access all pathways linked from an interesting node, cf. Figure 4.4. This interaction supports users to directly navigate and/or to gain insight into the connectivity of desired pathway components. Additionally, *back and forward buttons* may be used to assist the navigation process. These buttons can be found on the left and right upper corner of the navigation window. They will also highlight a specific graph icon triggered by a mouse-over action, providing an additional insight into the destination at which they would arrive in case of clicking the Back- or the Forward-button, respectively. This feature avoids the need to search for the previous pathway in multiple open windows and helps to keep track of the navigation processes.

Aggregated Links View Above we described our graph icon approach, which gives insight about all pathways linked to the current focus pathway. However, this approach by itself cannot show the frequency a particular target pathway is being linked from the nodes of the focus pathway. Therefore, the visualization of pathway interconnectivity frequency (statistical data) is desired. We decided to adopt a *multiple views* approach that is a widely used technique to visualize different kinds of data simultaneously, instead of integrating the visualization into the main window. One reason for doing this is that we already have glyph icons and graph icons inside the navigation window. Integrating a third visualization would overload the view. Another reason is that the multiple coordinated view approach enables us to choose the most suitable visualization technique for a specific view and data set [38]. Therefore, we introduce a separate interactive diagram called *aggregated*

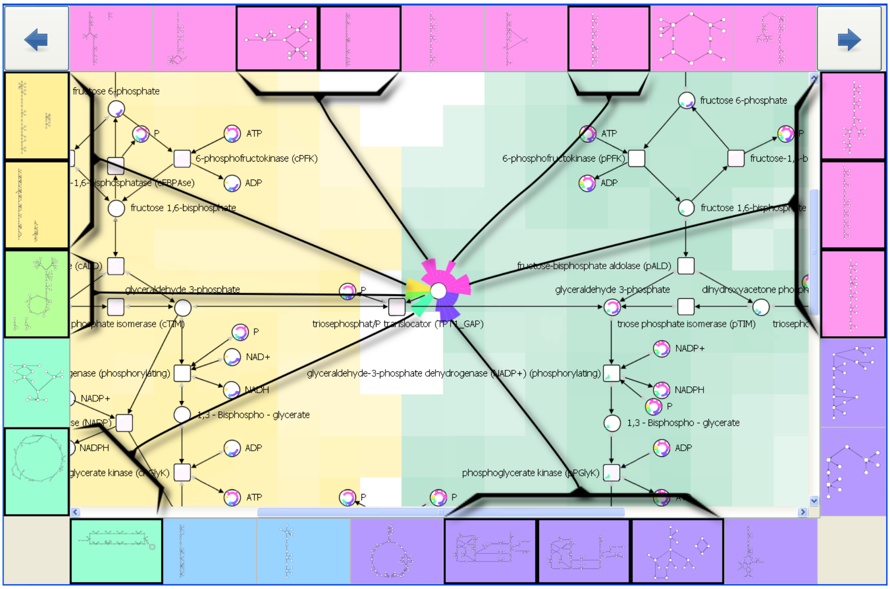

fructose 6-phosphate
P
ATP
6-phosphofructokinase (cPFK)
1,6-bisphosphatase (cFBPAse)
ADP
fructose 1,6-bisphosphate
(cALD)
glyceraldehyde 3-phosphate
P
phosphate isomerase (cTIM)
triosephosphat/P translocator (TPT1_GAP)
NAD+
genase (phosphorylating)
NADH
NADP+
1,3 - Bisphospho - glycerate
se (NADP)
NADPH
ADP
glycerate kinase (cPGlyK)
ATP
fructose 6-phosphate
6-phosphofructokinase (pPFK)
fructose-1,6-b
fructose 1,6-bisphosphate
fructose-bisphosphate aldolase (pALD)
glyceraldehyde 3-phosphate
dihydroxyacetone phosph
triose phosphate isomerase (pTIM)
triosephc
NADP+
glyceraldehyde-3-phosphate dehydrogenase (NADP+) (phosphorylating)
NADPH
1,3 - Bisphospho - glycerate
phosphoglycerate kinase (pPGlyK)

Figure 4.4: A mouse-over action on a specific node highlights the correlated graph icons (black framed) around the focus pathway. Note that the shaded black lines are hand made and are not part of our plugin; they are just drawn to demonstrate the ordering and layout of the approach.

links view to present this statistical data. The diagram was placed into the upper right corner in a tab as seen in Figure 4.3. For a detailed view of the aggregated links view refer to Figure 4.5. We named this interactive diagram *link rose*, because it is able to guide the user similarly to a traditional compass rose metaphor, and it resembles the navigation glyphs used in the focus pathway. It represents statistical data about *all* links to other connected pathways in context of the current focus pathway in comparison to the navigation glyphs that show links of a single specific pathway node. In this way, it supports users to see patterns of strongly and/or weakly connected pathways and gives an overview of all outgoing pathway links connected to the focus pathway.

A tooltip with the name of the corresponding pathway, followed by the number of nodes linked to that specific pathway appears on mouse-over actions on a rose petal. The pathway group name in the second text line is also shown in the tooltip. Thus, a textual dimension of insight into the connectivity of a specific graph to other pathways is provided. A click on a rose petal, besides highlighting the corresponding petal, has also an effect on other parts of our visualization approach. This means that all graph nodes referencing the selected pathway are also highlighted in the navigation window, which follows the idea of several coordinated views [94]. This fea-

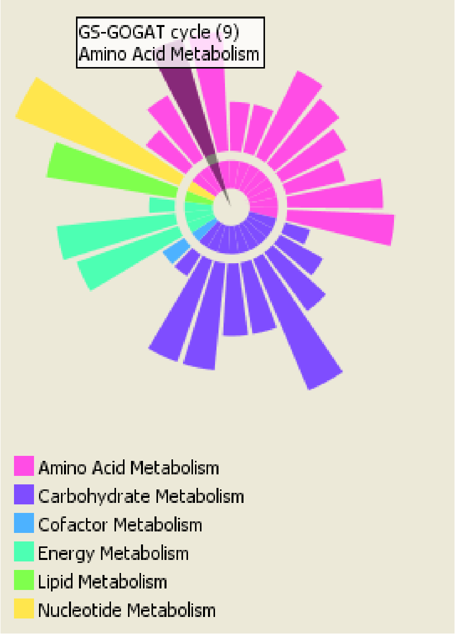


Figure 4.5: Link rose diagram. The *GS-GOGAT cycle* is selected triggering a specific petal to be shadowed in a more saturated color. A mouse-over action shows the name of the pathway in the first line of a tooltip, while the number in brackets reveals the total number of links from the focus pathway to the selected pathway. The second line of the tooltip shows the name of the MetaCrop pathway group to which the selected pathway belongs. Users can instantly notice strongly and/or loosely connected pathways. The legend below helps in identification of different pathway groups. The color-coding of the groups is persistent for each specific group, no matter where we navigate to.

ture enhances the comprehension of interconnections of different graph components and provides a better overview of the entire network structure by supporting users to immediately distinguish components connected to a particular pathway. Moreover, if afterwards we navigate to any target pathway, all shared nodes (clones) will be selected in the target pathway consequently revealing the common components between these pathways. In Section 4.2, we give a more detailed navigation and exploration example. Next, we will briefly discuss scalability issues of our approach.

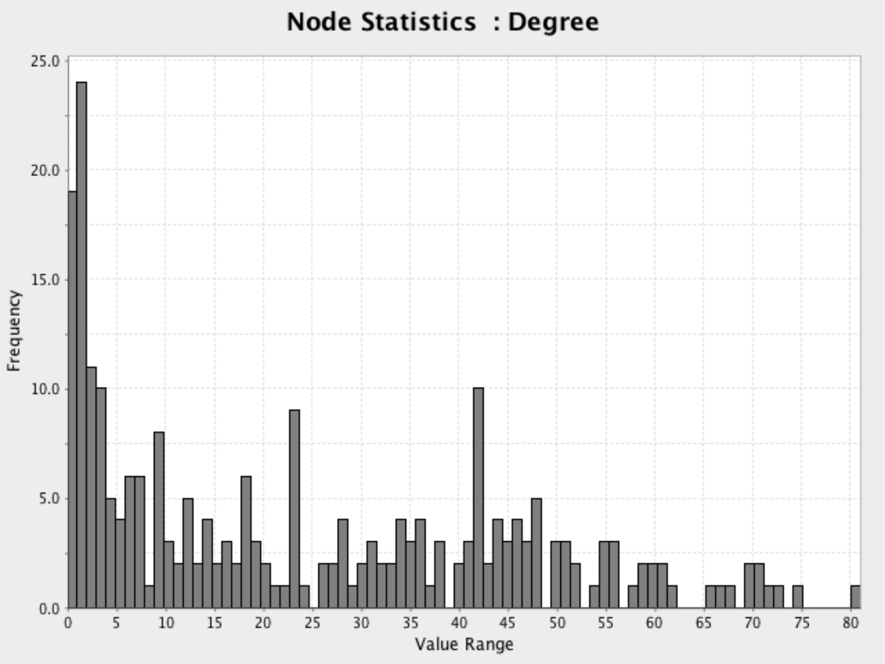


Figure 4.6: The figure shows the number of links to other pathways (same substance in other pathways) for the reference pathways of KEGG ("Frequency" means number of pathways, e.g., there are 10 pathways which have connections to 42 pathways each). The pathway *Inositol phosphate metabolism* has the largest number of links, with substances of this pathway found in 81 other pathways.

Scalability

As mentioned earlier in this chapter, our approach is focused on the MetaCrop database consisted of 38 pathways, and it scales well with this particular data set. Size and resolution of the screen as well as the number of pathway links being visualized undoubtedly have an effect on the scalability of the graph icons discussed in Section 4.1. Theoretically, if the number of pathways is large enough and/or the display size and resolution decreases, our graph icons would look so small that it would be hard to notice anything on them. Several approaches could be used to alleviate this problem. We could reduce the number of displayed graph icons by introducing some prioritization features and/or by providing more filtering. At the current state, users can decide not to download the complete dataset, thus making a kind of pre-filtering step. However, enabling users to filter out uninteresting pathway groups, such as *Amino Acid Metabolism* by simply deselecting them could be useful. In this case, our tool could delete all target graph icons belonging to the deselected pathway group from the current view.

Arranging a fixed rectangular block for each pathway group around the target pathway is another way to address the problem of many pathway links. We could integrate graph icons of different sizes within this block, following a semantic fish-eye metaphor. The size could be proportional to the number of matching nodes between the individual target pathways and the focus pathway. However, we do not need such techniques at the moment as described in the following.

We can demonstrate the scalability of our approach by using a larger data set with our tool. The KEGG [61] dataset contains the largest collection of pathways and also both signaling and metabolic pathways. Therefore, we have chosen KEGG focusing on substances (the most commonly shared elements between different pathways). Our analysis showed that most of the pathways (about 95%) have less than 62 links to other pathways in practical analysis sessions, as shown in Figure 4.6. These findings prove that our approach scales without difficulties for KEGG in the majority of the cases. A pathway with 81 links (extreme case in Figure 4.6) was loaded on a laptop with a screen resolution of 1280x800 pixels. Although the individual nodes and edges in some of the graph icons were less distinguishable, we found that the graph icons scale even in such extreme cases. We do not expect any scalability problems for applications in biology as nowadays standard office displays have even larger resolutions. This also holds for the number of nodes and edges: a typically analyzed (focus) pathway contains at most a few hundred nodes and edges, which can be easily drawn and interactively displayed by the VANTED system.

Although our approach copes relatively well with the interconnectivity of the KEGG dataset, the mapping of KEGG hierarchies/groups is not fully implemented at the moment. There are two potential limitations: deeper hierarchies (i. e., more levels) and/or broader hierarchies (i. e., many different groups). In a perceptual context, the fact that each pathway group has a fixed position around the navigation window is regarded as advantage. Unfortunately, this is not sufficient when dealing with deeper hierarchies. However, in the future work section (Section 7.2) we will discuss possible solutions to this problem. Next we will discuss implementation and performance issues.

Implementation and Performance

The VANTED system supports extensions by implementing different *plugins* as separate projects. This mechanism allows a great deal of flexibility in the implementation process and possibilities for building various improvements as well as the embedding of new (visualization) techniques or tools. Users of the VANTED system are able to customize the functionality according to their needs by downloading and using the desired plugins. Our current visualization tool is separated from the original VANTED core implementation as an individual package, i.e., a JAR file that can be imported into the main application as a plugin. A special loading dialog is available by VANTED in order to enable the integration of different plugins.

Thus, users only need to download and import the plugins through a specific form in the VANTED system. Our plugin has been added to the list of official VANTED plugins recently.

Even if the VANTED system offers access for various core method calls through its plugin mechanism, some interventions on VANTED's core libraries were necessary in order to implement all our desired features. More knowledge of the core VANTED source code was required for the changes in the main view of the navigation window (i.e., where the drawing of the focus pathway is located). VANTED handles the graph layout in the main view together with its background coloring and the navigation glyphs by itself (Figure 4.1). We had to add several frames in order to lay out the small graph icons around the main view. We used simple buttons for representing the graph icons. Methods, such as scaling or changing the background color, which allow the small graph icons to be embedded into the buttons were provided by VANTED. The implementation was simplified considerably by this feature. It speared us of the need to compute a layout of the embedded (target) pathways. Additionally, this approach helped us to effortlessly keep a consistent visual appearance of the specific graphs, whether they are focused in the main view or shown as a graph icon around the main view.

As explained earlier, VANTED also supports a relatively easy access to various data structures and methods to store and retrieve the data as well as their relations to each other. Therefore, we made use of such facilities whenever we needed them. For instance, the implementation of the navigation features was simply done via action events of already existing VANTED methods. Brushing and different color codings were implemented using similar resources. Although the link rose diagram (Figure 4.5) was created in a separate view, it was created mostly using existing VANTED resources besides the drawing algorithm, especially its seamless integration into other parts of the system. The visualization of navigation glyphs (standard and enlarged) and the link rose diagram were implemented straightforwardly and are not based on any specific algorithms.

A Dell Latitude E6400 laptop with a 2.53 GHz Intel Core 2 Duo processor and 3GB of RAM running Windows XP was useed to test our VANTED plugin. The size/number of the desired pathways and the Internet connection speed dictate the time to download the dataset. Users are able to load a focus pathway into the navigation window in 1-2 seconds in worst case once all pathways are completely downloaded. This includes the computation of the link connections to other pathways and the generation of the aggregated links view. In case the focus pathway is strongly connected (40-50 links) to target pathways, it might take at most additional 2-3 seconds to generate the graph icons. We estimate that most interactions will occur immediately based on the connections of the KEGG data set, shown in Figure 4.6. Worst case situations occur within the *unit task* level [19].

4.2 Use-Case Scenario

In this section, we will present the way our plugin is used. Usually, users download the whole MetaCrop data set or the pathways they are interested in. If the number of pathways being explored increases, the navigation and exploration process gets more complicated in standard visualization approaches. At the same time, the contextual information about the interconnectivity of various components is lost due to the reasons explained in previous sections of the chapter.

Users will be able to gain insight into the interconnectivity immediately after activating our plugin. The graph icons arranged around the focus pathway provide an overview of the overall connectivity of the focus pathway (cf. Figure 4.3) at the same time serving as links to particular pathways as well. More specific information about the frequency of the connections to all target pathways is provided by our aggregated links view. These views together with various interaction and brushing techniques will guide the users to interactively explore the pathways. If a particular petal of the link rose is considerably long compared to the others, they might conclude that the corresponding pathway shares a considerable amount of components with the focus pathway currently displayed in the navigation window. We will explain the navigation process by means of Figure 4.7.

Let us assume that our user has navigated to the *Pentose phosphate pathway*. He/she notices that the link rose has a petal that is longer than the others (cp. the upper part of the figure). He/she discovers that the name of the pathway is *Calvin cycle* and that there are 16 links to that particular pathway after a mouse-over event. All the nodes referencing that pathway will be selected after he/she click on the rose petal. This strong interconnection between *Carbohydrate Metabolism* (to which *Pentose phosphate pathway* belongs to) and *Energy Metabolism* (to which *Calvin cycle* belongs to) can be further investigated by selecting the *Calvin cycle*.

The user navigates to the desired target pathway by clicking the corresponding graph icon (see the bottom part of Figure 4.7). Previously selected nodes will remain selected if they exist in the new navigated pathway. Our user will notice that several nodes in the *Calvin cycle* are now selected. This example shows that the plugin makes it rather easy to distinguish the connectivity context of different pathways and supports the navigation between them.

Another investigation could start with the *Sucrose breakdown pathway (monocots.)*. Again, the link rose has a petal that is much longer than the others. The user discovers that the name of the pathway is *Sucrose breakdown pathway (dicots.)* and that there are 54 links to that particular pathway, after a mouse-over event. All the nodes referencing that pathway are selected by clicking on the rose petal. Only three nodes are not selected in this case, which clearly shows that this pathway is strongly interconnected with the selected one. After navigating to the desired target pathway *Sucrose breakdown pathway (dicots.)* by clicking the specific graph icon, a biolo-

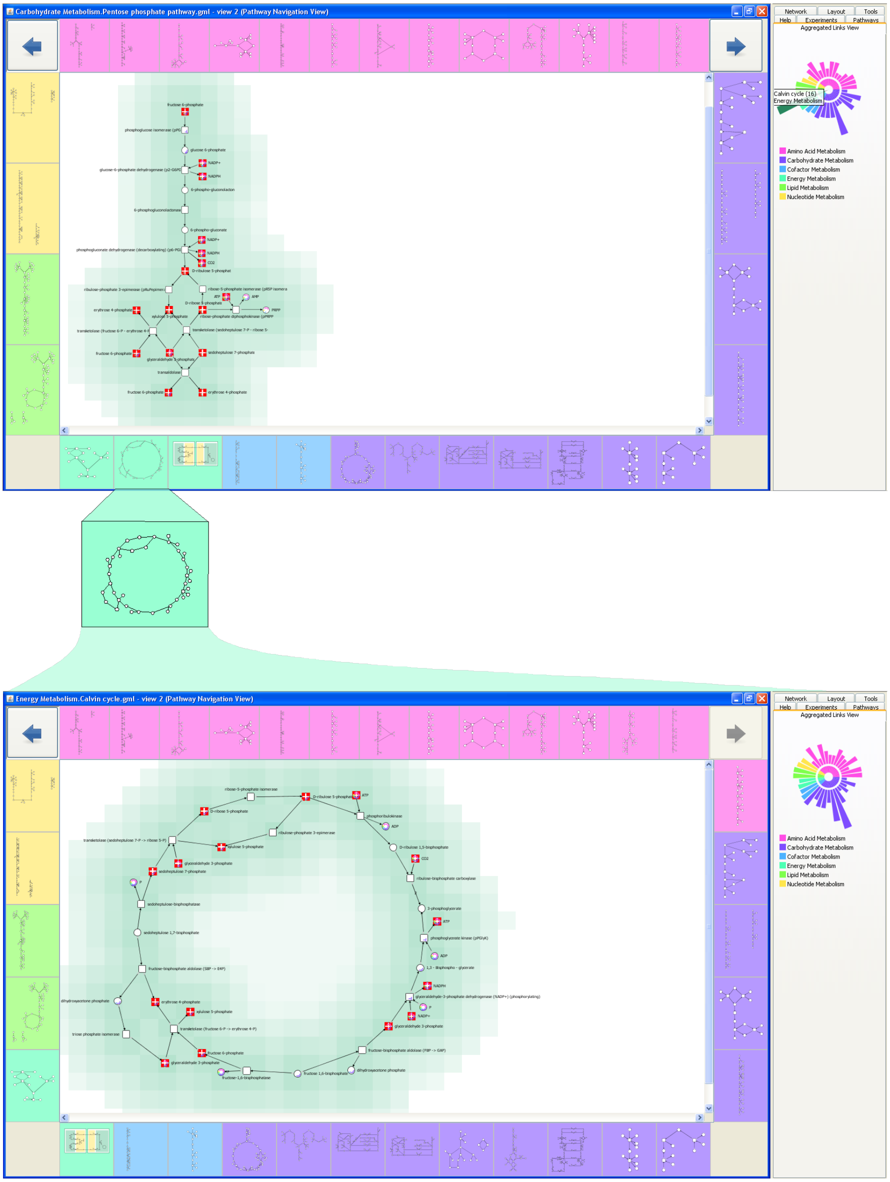


Figure 4.7: Navigation between two pathways. The selection of the nodes is persistent even when navigating in the new pathway. A feature that helps to discover shared components.

gist will immediately recognize that he/she is navigating in our sample through the same metabolic pathway. In this case, it shows the differences between an important pathway in monocotyledons (monocots) and dicotylodons (dicots).

Feedback from Biologists

The plugin was discussed in three user sessions with four domain experts. These experts are biochemists and biologists who are familiar with the VANTED system. They are used to work with metabolic pathways and pathway databases such as MetaCrop and KEGG. As they used VANTED for their work, only a short introduction (less than five minutes) about the navigation idea and the usage of the plugin was made. Afterwards, the experts were asked to work with the plugin to explore the metabolic pathways stored in the MetaCrop system and had to report positive and negative aspects.

Overall, they provided a very positive feedback. The navigation through pathways was described as very helpful and intuitive, even as “cool”. They consider the use of glyph size and colors as appropriate and the speed of navigation (e. g., replacement of one pathway by the next one) mostly as fast enough. The interaction between the the pathway and *link rose* was considered very useful for selecting nodes occurring in another pathway.

However, the domain experts had several suggestions also. One user would prefer to have all other pathways (also currently not connected ones) as graph icons placed around the current pathway. The majority liked the currently implemented concept of presenting only those pathways as graph icons which are connected to the current pathway of interest. Some users recommended to include the name of the pathway in the graph icon (currently the name is only shown after moving the mouse over the icon), even though they are familiar with the MetaCrop pathways and had no problem to navigate to specific pathways (graph icons). Finally, one user recommended optionally restricting linkage of substances and avoiding co-substances (e. g., ATP, H_2O). This would considerably reduce the number of interconnected pathways.

The users found our method to navigate through pathways very helpful. However, they recommended additional functionality to make it more useful for their work such as the ability to map measured data onto nodes or to see changes in fluxes over reactions. It should be noted that these aspects are actually supported by the VANTED system (e. g., fluxes can be represented by different edge width and additional data such as metabolomics data by additional coloring of nodes or including of diagrams into nodes) and are already used by some of the experts for their data analysis. Such functionalities are already possible with our plugin, however, as it is not the focus of this study, we did not extend the discussion with the users into this direction.

4.3 Summary

In this chapter we presented a solution to guide the navigation and exploration process of interconnected pathway visualization and discussed it in the context of biochemical networks. The analyst can obtain an overview of connected pathways while they are working within the currently focused pathway by using our plugin. Although the problem may appear simple, it is still not sufficiently solved in existing visualization tools.

Our main contribution is the development of new and intuitive visualization and interaction techniques to guide the domain expert during the exploration and navigation process and integrate it into a complex system used in practice. To achieve our goal of providing the analyst with more meaningful contextual information in hierarchically structured biochemical pathways, we use well-known techniques such as glyphs [122, 123], brushing [109], and topological information of the target pathways (realized by the graph icons). We published our plugin, called GLIEP (Glyph-based LInk Exploration of Pathways), on SourceForge [58], where users can easily download it. Programmers can use our source code for further improvements or for adding additional features. Additionally, GLIEP is now part of the official VANTED plugin list.

Here we focused on enabling navigation and enhancing perception of interconnectivity between different subnetworks belonging to one bigger network. In the next chapter, we will discuss issues of showing interconnections of two conceptually different, but related networks.

Chapter 5

Visual Analysis of Different Types of Networks in Biology

The study of high throughput data such as transcriptomics and metabolomics data are important in biology and medicine. The analysis of these data is done usually by using tools such as ontologies and hierarchical clustering. Statistically overrepresented ontology terms are identified by enriching the ontology terms in data, giving an overview into important functional modules or biological processes. In order to find relatively homogeneous clusters of experimental data points, hierarchical clustering is used as a standard method to analyze the data. 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.

In this chapter, we present a new method for the task of combining the aforementioned views. It is based on the following publications [57, 69]. The rest of the chapter is structured as follows: the properties of the input data are discussed in Section 5.1. We present our visualization approach of combining Gene Ontology visualization and cluster tree visualization in Section 5.2. Section 5.3 deals with scalability issues of the proposed method. Finally, we summarize our work in Section 5.4.

5.1 Properties of the Input Data

We focus on a transcriptomics data set representing different expression levels of genes in *E. coli*. As mentioned in Section 2.3, the Gene Ontology (GO) forms a directed acyclic graph (DAG) [6, 114]. This large DAG contains more than 34,000 inner nodes and a large amount of leaf nodes depending on the organism being examined. 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 the 7,312 genes, we also consider those nodes which are on paths between the GO root node and leaf nodes (genes), 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. The hierarchical clustering has been computed based on the distance of the expression levels of genes at different time points. 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).

From a developer's point of view, both of 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). Using the information provided by the gene labels, we can map these two data sets as presented in Figure 5.1. 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 5.7).

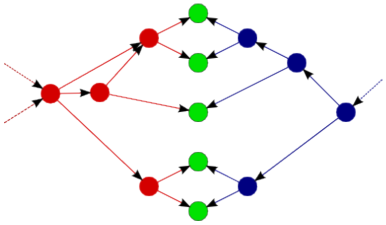


Figure 5.1: The *red* part on the left represents a part of the GO DAG. The *blue* part on the right represents the Cluster Tree, while the *green nodes* in the middle are shared between both of them. Note that this diagram shows an idealized situation, because the common leaves do not need to be neighbored.

5.2 Visualization Approach – CluMa-GO

In this section we will discuss our approach to solve the underlying problem. To demonstrate our approach we have implemented a prototype tool called CluMa-GO (**Clu**ster **Ma**pping of **G**ene **O**ntology) [5]. We visualize the GO DAG and the Cluster Tree in two separated and coordinated views due to the complexity and huge amount of data to be visualized [94]. We load the data into our tool by using two individual .gml files [46] (one for the GO and one for the clustering) through a standard dialog box.

We already discussed the complexity and challenges of the visualization of huge networks on its own in Chapter 2. Our current task is even more complicated as we have 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 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 we draw the graphs by using conventional graph drawing algorithms. One way to overcome the problem in the case of trees is to use scrolling and panning actions [121], 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.

The mapping of the specific subtree from a selected GO node as described in Section 5.1 introduces another issue. These computed subtrees or sets of nodes are not sequentially mapped resulting in “gaps”, see the green leaf node between the two yellow rectangles in the background of Figure 5.7. The issue here is that these gaps may 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.

Our strategy is to take into consideration the different features of the GO DAG and the Cluster Tree. Therefore, specific visual representations for both are implemented accordingly. These representations address the aforementioned challenges separately, while we employ interaction to show the mapping between them. Initially, we discuss the approaches to visualize both the GO DAG and Cluster Tree. The supported interaction techniques are described later in order to distinguish between visual representations and interaction concepts. Figure 5.2 shows a complete overview of the GUI of our prototype implementation.

Gene Ontology (DAG) Visualization

Even if we use a subset of the entire GO, the selected GO DAG is relatively large, as already described in Section 5.1. 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 we present here is inspired from pixel-based approaches which usually cope with large data sets [64]: 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*. Non-terminal nodes are represented by light-blue pixels, while red pixels represent leafs or unconnected nodes.

As there are no cycles in DAGs, 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 5.2. Shneiderman and Aris [107] presented semantic substrates, which are somewhat 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 5.2 on the left shows 17 layers marked from 0 to 16. There is a considerable number of unconnected nodes in our GO dataset. Figure 5.2 on the left shows a specific case where these unconnected nodes are placed in layer number 0. This visualization reveals that this level is the most dense layer. In order to provide more insight into the structure of the data, two layering approaches were developed. These approaches mainly differ in the way how leaves and unconnected nodes are positioned. We discuss these approaches in the following two paragraphs.

Levels Layout The leaves (red pixels) and non-terminal nodes (light-blue pixels) are placed into their corresponding layer depending on their graph-theoretic distance [12] from the source node (root) in the first layering approach named *Levels Layout*. In addition, 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, we gain additional insight into the topology of a specific layer 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 the Figure 5.2, namely in the GO view area of our tool. The same layout strategy is used in the zoomed-in view shown in Figure 5.3, 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. Some form of normalization of area density could be introduced in order to achieve a true bar chart effect, but that is not implemented in the current version of our tool. Leaves and terminal nodes are distributed in separate regions as described above. However, the placement of the pixel nodes inside these regions is random. Level 0 is reserved for the unconnected nodes.

Bottom Layout The second layering approach *Bottom Layout* 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 (Figure 5.4). Unconnected nodes can be filtered out if necessary. Figure 5.4 shows a view of the Bottom Layout with a particular node selected. The strength of this approach is that it provides us with insight into the distribution of nodes among different layers without considering the leaves (genes).

In order to avoid clutter, the visualization of edges is omitted by default. They are only shown in case the user selects a desired GO term (non-terminal node) for further exploration. Users could optionally show all the edges, but then the view will be overloaded. Additionally, 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 5.2 on the left shows the edge bundling of the computed

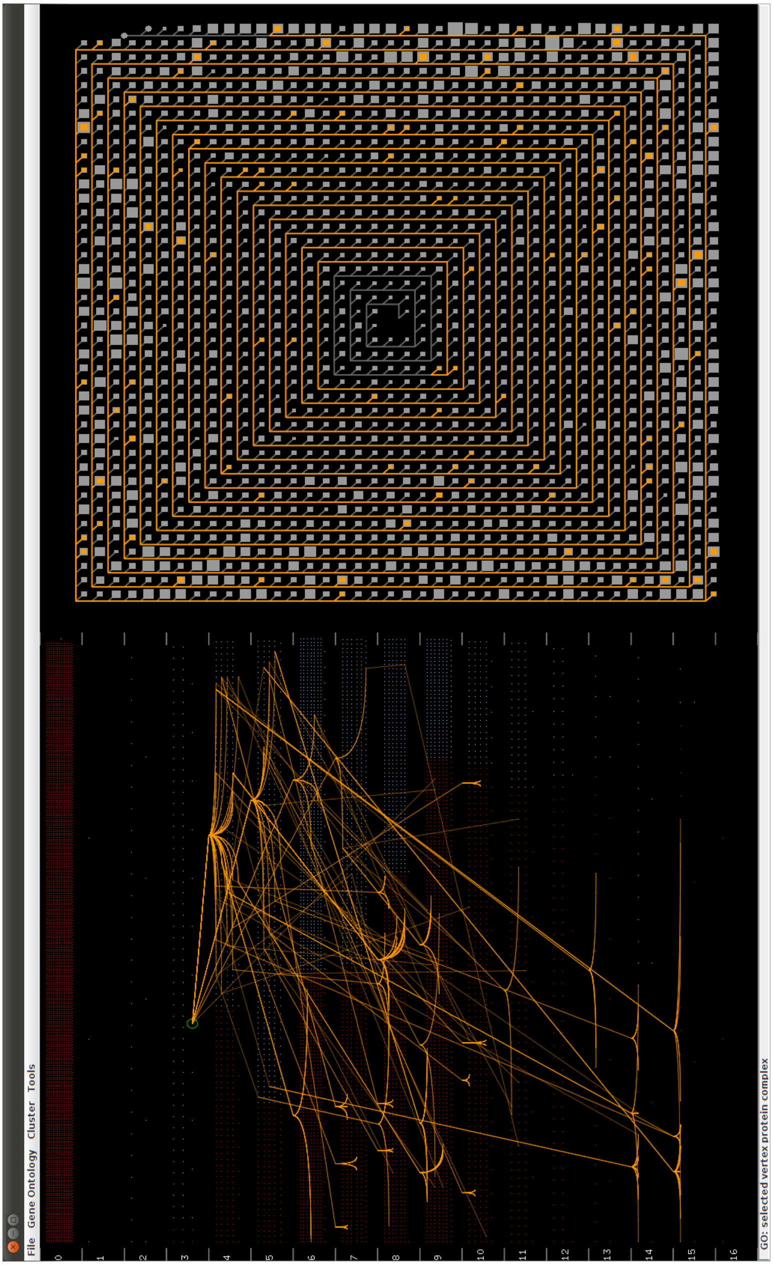


Figure 5.2: GUI of CluMa-GO (rotated by 90°). On the left hand side, the used Gene Ontology is represented in the GO view (Levels Layout). On the right hand side, the Cluster Tree view is located.

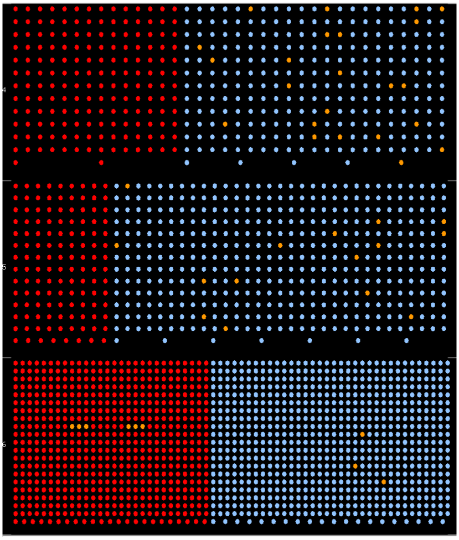
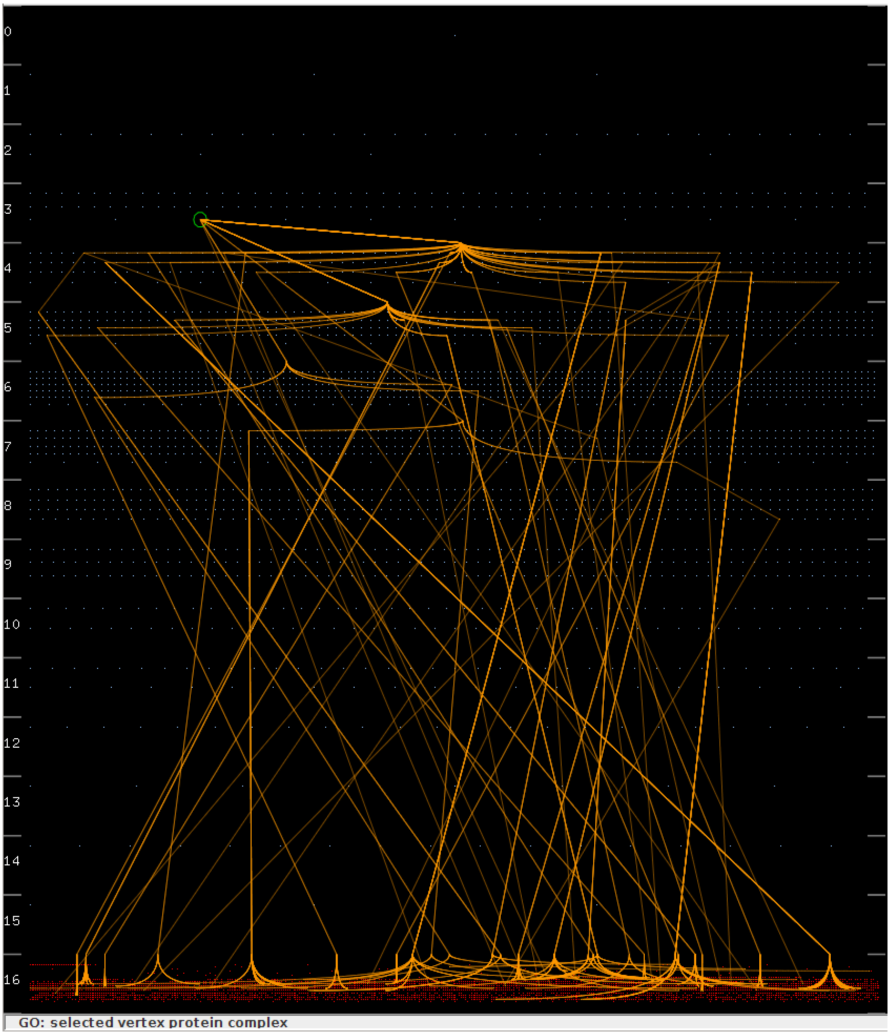


Figure 5.3: Zoomed-in view using the Levels Layout approach. The red nodes represent leaf nodes (e.g., genes); the light-blue nodes represent non-terminal nodes (e.g., terms). This view provides insight into the distribution of leaf nodes in a specific DAG level. The orange nodes represent the calculated subgraph (mapping).

subgraph in the GO view based on the Levels Layout approach, while Figure 5.4 shows the bundling applied on the Bottom Layout approach. Edge bundles reduce the visual overload and give insight into how different layers are accessed by a specific node. The use of arrows to show edge direction is unnecessary, as 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.

The use of pixel-based approaches introduced a new challenge. 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, we needed to find a good working compromise for both the computer display and for printouts. We used ColorBrewer [18] to guide us for this task.



GO: selected vertex protein complex

Figure 5.4: GO view with visible (bundled) edges based on the Bottom Layout. The green circle in leyer 3 highlights the selected GO term.

Cluster Tree Visualization

We faced similar issues when dealing with Cluster Tree visualization as we did with the GO representation. Both data sets are relatively large and any traditional type of visualization would not scale. As mentioned earlier, hierarchical clustering produces a large binary tree. If we draw this tree with the help of conventional drawing algorithms, the outcome would be rather high tree drawing, or a wide one, if we choose a standard dendrogram layout. This issue led us to the development of a new visual representation for the Cluster Tree. One special feature of our data set is that the trees are particularly high and unbalanced, with shallow branches (subtrees). This characteristic enforces the disadvantages of traditional node-link layouts even further by consuming the space even more due to the unbalanced structure of the trees. However, we decided to leverage this feature and to design a special drawing algorithm for large trees of such nature.

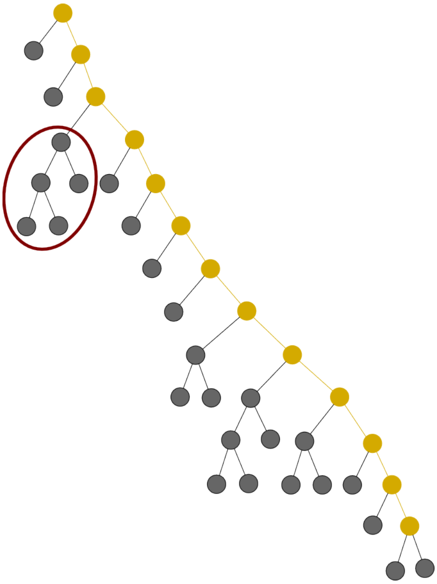


Figure 5.5: Sample cluster tree t.

Figure 5.5 shows how a small part of such a tree might look like. The yellow colored part shows a "backbone" comprised of nodes and edges that form the longest path that connects all branches. 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 we implemented to deal with large unbalanced binary trees. Our approach overcomes the need to perform repetitive scrolling to browse or navigate the elements [53, 112]. 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, Figure 5.6 shows the result of our layout algorithm applied on the tree t, shown in Figure 5.5.

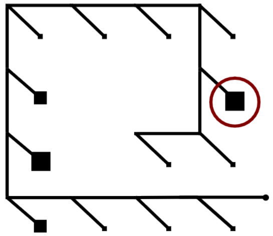


Figure 5.6: Spiral Tree Layout of t. The drawing algorithm was inspired by standard spiral layouts that are mostly used to represent time-series, such as [2, 118].

Due to the size of the data set, the subtrees connected to the backbone are aggregated. This means that a certain amount of abstraction is allowed in our visualization approach: each small box glyph in Figure 5.6 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 5.6 is depending on the size of its corresponding subtree marked with the brown ellipse in Figure 5.5. 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.

Interesting patterns of distributions of subtree branches in the Cluster Tree can be identified by using this approach. For instance, the largest branches appear far away from the root node of the tree if we take a look at the Cluster Tree view in Figure 5.2. 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 5.10 and 5.11) 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 as well.

We use brushing techniques to show the mapping between the two parts, GO DAG and Cluster Tree respectively. In the following subsection, we describe these and other interaction techniques.

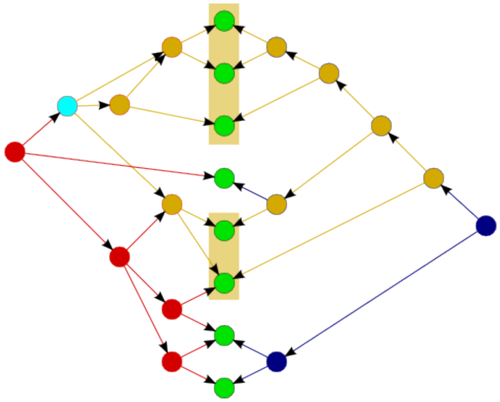
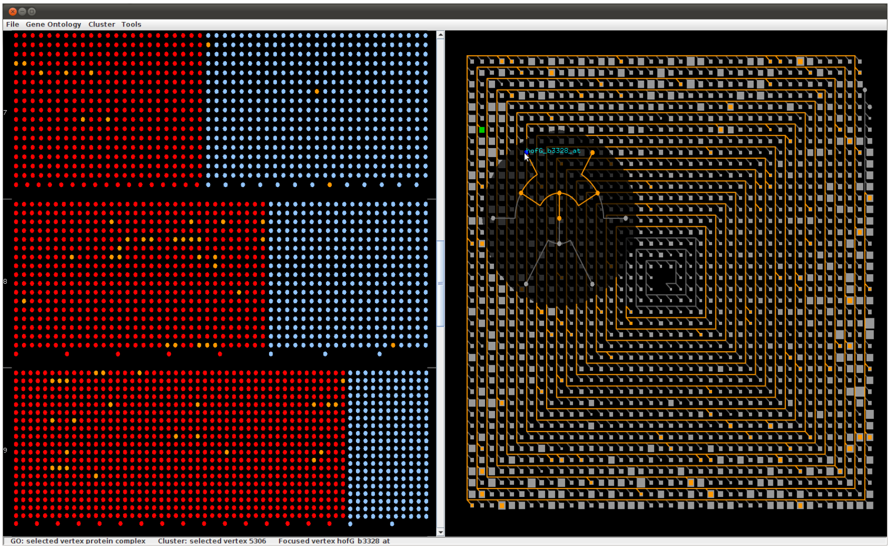


Figure 5.7: The *green nodes* in the middle are shared between both the GO DAG (*red nodes*) and the Cluster Tree (*blue nodes*) (cp. Figure 5.1). The interactively selected node is highlighted in *cyan*, from which we traverse the graph (*yellow nodes*) until we reach all accessible leaves (*green nodes with yellow background*). The leaves are used to calculate a subtree of the Cluster Tree (*yellow nodes* in the right part of the figure).

Interaction Techniques

As described by our collaborator of this project and a domain expert, *Prof. Dr.* Falk Schreiber from IPK Gatersleben, biologists explore the data in two ways: by browsing the data set randomly or have 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. Users could also directly click on a particular node in the GO view. A tooltip displaying the name of the node is shown if a mouse-over action is performed on that node. This enables 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 5.4. In this way, identifying the layer the currently selected node belongs to has been made easier too.

The subgraph consisting of all reachable nodes will be calculated after the node has been selected. All nodes belonging to the computed subgraph, as described in Figure 5.7, will be highlighted in orange in the GO view. The edges of the subgraph will be shown too. 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 colored elements. Note that the higher the hierar-



GO: selected vertex protein complex Cluster: selected vertex 5306 Focused vertex hofG b3328 at

Figure 5.8: This screenshot shows the zoomed-in GO view on the left hand side and the Cluster Tree view with opened subtree widget on the right hand side.

chical level of 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. 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 5.8). The user can also scroll up or down between three layers simultaneously. Since a lot of edges from other layers might go through our focused 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 since we deal with a 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.

When a specific GO term is selected in the GO view, the calculated subtree is highlighted in orange in the Cluster Tree view, as seen in the right part of Figure 5.2. In this case we notice 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 5.9 displays a cut-out of a Cluster Tree view in order to provide a larger view.

]

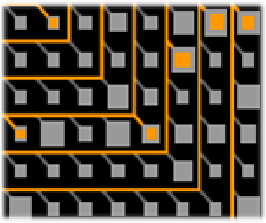


Figure 5.9: Cut-out of a mapping in the Cluster Tree view.

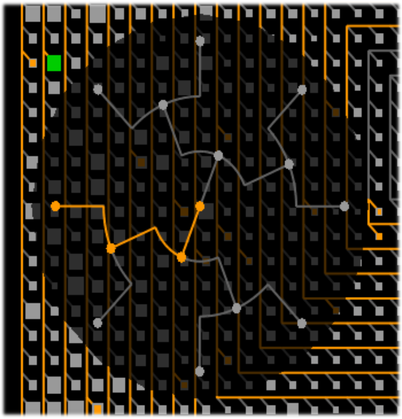


Figure 5.10: Subtree (branch) view. The more detailed view of the selected branch (*green box glyph*) is visualized as a dendrogram.

The subtree can be further examined by clicking 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 5.10) similar to other dendrogram visualizations [115, 98] or in an "explorer view" (Figure 5.11) 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 it covers, slight user-defined transparency is introduced. The user can also grab the widget with the mouse and move it around in case the area covered by the widget is important and interesting. A mouse-over action shows the name of the particular node of the tree through a tool-tip similar to the GO view.

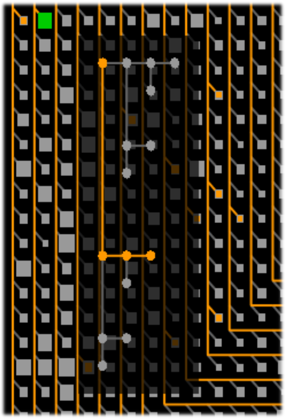


Figure 5.11: Subtree (branch) view. The more detailed view of the selected branch (*green box glyph*) is visualized by following a so-called HV-drawing algorithm.

5.3 Scalability

A number of issues need to be addressed when dealing with large data sets as presented in Section 5.1. Providing an overview or showing the complete data set to start the analysis process is one of the main challenges. We can clearly see that our prototype is able to visualize the complete data set as explained in the previous section. More insight into the data is gained and the mapping between the GO subset and Cluster Tree is performed with the help of the described interaction techniques.

The responsiveness of our prototype was another issue that we faced during our development. It is extremely important that the system can handle all data and provide the users with real-time interaction possibilities. CluMa-GO needs approximately three to five seconds 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, we have managed to speed up the process significantly. 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. We used a smart map from the open source library Google Guava [37] 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.

5.4 Summary

We presented a new method for the combined visualization of an ontology (represented as DAG) and a hierarchical clustering (represented as tree) of one data set. The proposed method interactively visualizes all the data without scrolling, thereby presenting a complete overview. We presented a new metaphor for visualizing highly unbalanced binary trees, inspired by the shape of a spiral. Our approach also allows for interactive selection and navigation to explore the data. We have showed that our tool is able to tackle the problem in our research focus, i.e., the visualization and visual mapping between two huge graphs of different type.

In the last two chapters, we were mainly focusing on visualization of large networks. In the previous chapter, we offered a solution for the visualization and navigation of interconnected pathways. In this chapter, we visualized two different types of networks. The next chapter takes us into another problem faced in almost all scientific domains: we discuss an approach to visualize multivariate network attributes.

Chapter 6

Multivariate Network Visualization Using Magic Lenses

Several approaches that deal with the problem of visualizing multivariate networks were discussed in Chapter 3. In this chapter, we will present the *Network Lens*, 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]. The Network Lens visualizes attributes (or a subset of them) by using small glyphs each time it is applied on a network element, 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. Experimental data measured over time could be attached and represented by the network nodes in form of a time plot, for example. Furthermore, let us assume that a biologist wants to analyze such data at time step t_i. Then, a specific Network Lens instance could be used to show the data at time step t_{i-1} for a set of nodes without having a need to change the current visualization setting. 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 can extend already existing views to show node and edge attributes of the underlying network. It is possible to create lenses for specific exploration processes, to store them for later analyses, and to combine them to analyze related attribute sets.

6.1 The Network Lens

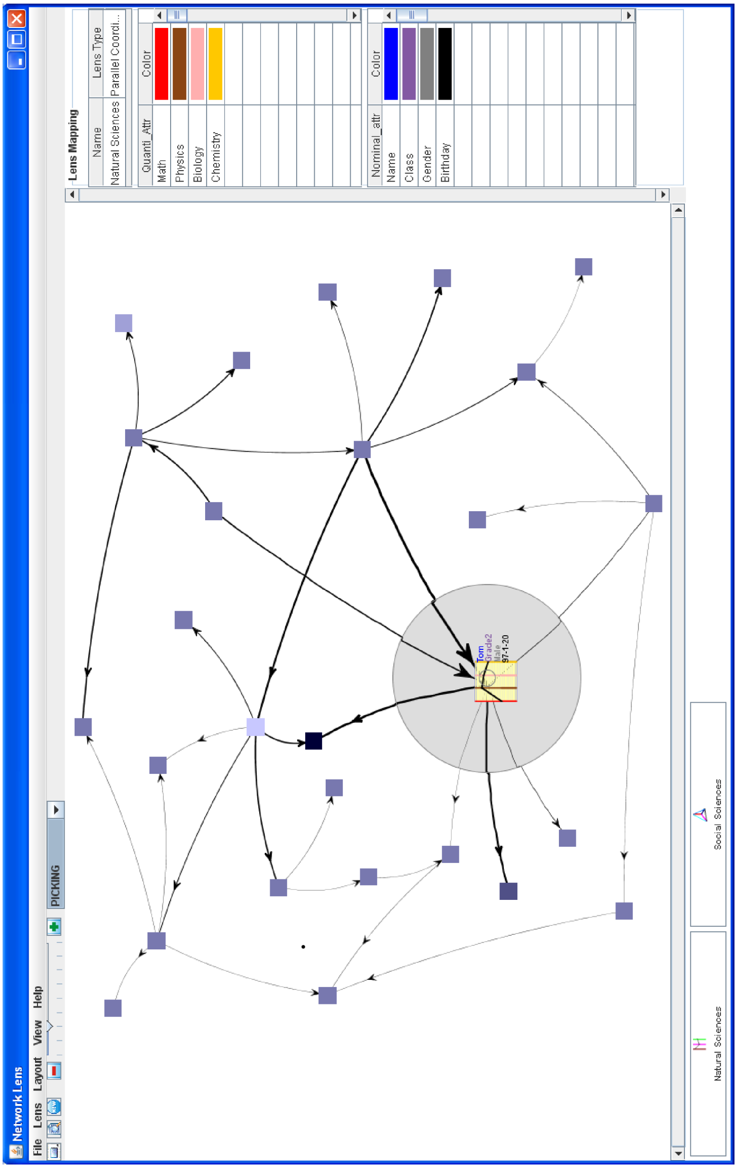
In this section, we will present the Network Lens that supports the interactive analyses of complex networks using visual filtering. It preserves the overall network topology and context and facilitates the visualization of network attributes at the same time. Our approach is independent from the graph layout and 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. By using our approach, users can get more details about *desired* attributes by focusing on specific node(s).

These desired attributes can be chosen from a subset of all available attributes the user is interested in. Afterwards, users can interactively explore the network elements on the basis of the selected attributes. It is conceptually possible to represent the remainder of the attributes inside the nodes if such a feature is desired, although our current implementation does not support that. As discussed in Section 3.2, such approaches have the drawback of space usage and clutter. In the following, we will discuss it and show how our approach copes with such problems. We will also present our software prototype and describe a typical use case scenario.

Approach

The main inspiration for our approach comes from the work of Bier *et al.* [11, 113] as briefly described in Section 2.2. In contrast, our approach is driven by attribute semantics and not focused on pure graphical filters. We extend their idea 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. Depending on the data type each attribute or group of attributes can be represented more or less efficiently by using different visualization approaches. Therefore, having the option to choose the way of visualizing different attributes is important. Moreover, in order to simplify and speed up the process of creating new lenses users should be able to combine different lenses by using drag and drop interaction. Additionally, one should be able to set up and store a number 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 network visualization. Users can then switch between these lenses interactively. The GUI is divided into three parts, as shown in Figure 6.1. A traditional node-link network visualization is placed on the left hand side. This is also the main area of the tool, and occupies the major part of the tool window. After loading the input network (using a GraphML specification [17]), an overview of the entire graph topology is displayed. Ideally, nodes could be drawn in various ways. However, at the time of writing this thesis, nodes are only represented as rectangles. The user can also map the value of an arbitrary attribute to the color saturation of the nodes. The thickness of the edges represent the value of the edge weights if specified in the data set. 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. These features help in identifying interesting parts of the network. They 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.

By looking at the Figure 6.1, an active Network Lens *Natural Sciences* is displayed in the center of the network visualization. It shows a small Parallel Coordinate diagram with four quantitative and four nominal attributes belonging to the



Name	Lens Type
Natural Sciences	Parallel Coordi...

Quanti_Attr	Color
Math	🟥
Physics	🟫
Biology	▨
Chemistry	🟧

Nominal_attr	Color
Name	🟦
Class	🟪
Gender	▨
Birthday	⬛

Figure 6.1: 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).

corresponding node. In the current situation it covers a single node. However it is possible to access other nodes by moving the lens with the mouse or to translate the graph behind the lens. The size of the lens can be changed by simple mouse resizing actions similar to standard windows based GUI's. Additionally, lenses support zoom functions during which node representations are only magnified and undistorted, while the edges are distorted to help following them as the magnification may introduce a "cut effect". In some cases, this distortion cannot compensate enough and the edge flow insight could be lost or distorted too much to make any sense. This side-effect can be avoided either by slightly moving of the lens or by following the edge lines along the lens rim. The legend of the lens attribute color mappings is placed on the right panel (called Lens Mapping). All the lenses created and used in a working session are located in the bottom part of the tool window.

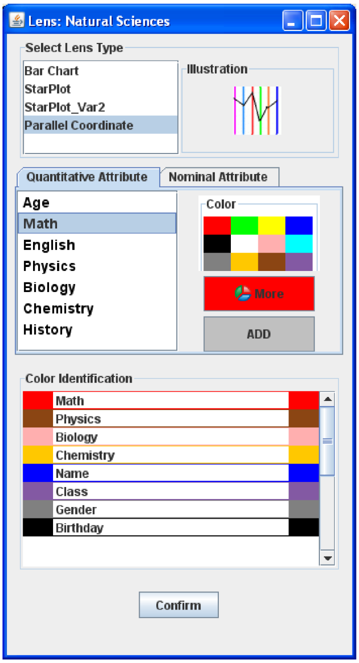


Figure 6.2: A dialog box used to create and edit lenses. Users can specify the type of the lens, select different attributes provided by the input data set and assign their color mapping.

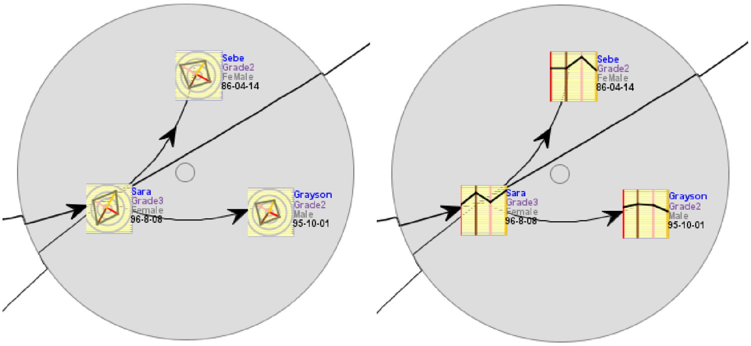
The Lens menu in the main window holds the New Lens option which is used to start the lens creation process. The lens has to be named at first. Users are advised to use self-descriptive names for lenses in order to remember the ”meaning” of the lens in case a lot of them are created and stored. Let us assume our user is analyzing a data set of students relationships in context of their grades in various school subjects (see Figure 6.1). Different students might have different affinities towards certain types of subjects, and exploring their relationships in this context is important to our user. At this point, the user can create different lenses for groups of classes and name them accordingly. For instance, he/she might 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.

After naming the lens, one must specify the visual representation, select the desired attributes to be visualized and specify their color mapping by using the form presented in Figure 6.2. The users 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 supports two variations of star plots, one bar chart, and one parallel coordinate visualization at this time, see Figure 6.3. A sample view of the selected lens type is shown in the Illustration icon at the top of the dialog box (cf. Figure 6.2). We use two different tabs (Quantitative Attribute and Nominal Attribute) to specify quantitative and/or nominal attributes to be visualized by the lens. The attributes are added to the lens by selecting them from the list and clicking the ADD button. Color mapping of the attribute can be specified by the user, otherwise the systems assigns the color automatically. We have created a default color palette comprised of 12 standard colors as suggested by C. Ware [125] to achieve a good visual perception. 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 lenses. New lenses are added in form of buttons to the bottom part of the GUI. One can than 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. The text labels, used to represent nominal attributes, have a transparent background by default. This can be irritating 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. Adding the features of the EdgeLens approach discussed in Section 2.2 would be another way to improve this issue.

So far we presented the procedure for creating lenses. This process can be simplified and extended after at least two lenses were specified. In the following, we discuss the process of combining different lenses to create new ones.

Our approach enables us to combine already created lenses by *laying them one over another*. We were inspired by optics and the work of magic lens which enables such combinations of lenses that have different transformation functions regarding the graphic objects. However, in our case we face different issues. Let us assume we want to combine several lenses with different visualization metaphors for the same set of attributes. Combination makes sense only if the lenses are different in some



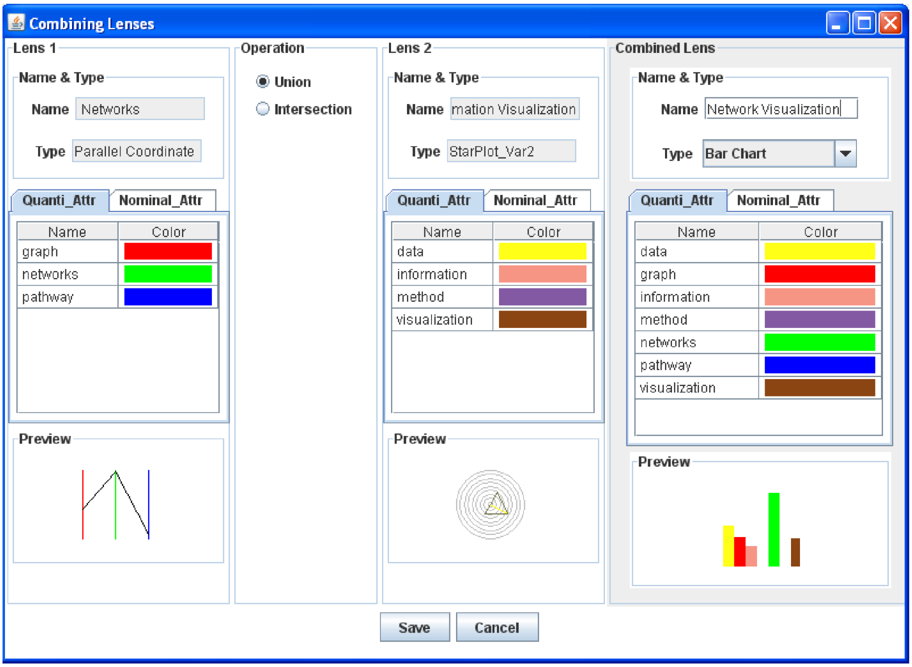
(a) Star plot diagram, 2nd variant

(b) Parallel coordinate representation

Figure 6.3: Two different visual representations used to display the attributes. Attributes can be color coded automatically, or the color can be specified by user. Nominal attributes, such as *Name* and others are represented by text labels on the right hand side of the glyphs.

aspects. As they visualize same attributes, they might use different visualization metaphors for them. Their combination will not be as straightforward as it could look in traditional, graphics-oriented magic lens approaches. As in case of real life optics or magic lens approaches, our users can simply drag and drop one lens button over another one. However, we needed to add another intermediate step where the user can decide the final outcome of the combined lens. Therefore, after this action a new dialog box Combining Lenses appears as shown in Figure 6.4.

Four groups of controls are presented in this dialog box. Information about the lenses that are going to be combined are shown in control groups Lens 1 and Lens 2. These control groups hold information such as lens name and type, the list of attached attributes including their color coding, as well as a preview icon. We use set operations to create a new set of attributes from the given sets of 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. Combined Lens group shows the result of the chosen operation. This way, users can specify the desired attributes in a quicker and simpler way. It has the same GUI layout as Lens 1 and Lens 2. At this point, 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. User can change the type of the lens and attribute colors of the combined lens. A new lens button will appear again at the lower part of the main window after saving the lens, cp. Figure 6.1.



Combining Lenses

Lens 1

Name & Type

Name: Networks

Type: Parallel Coordinate

Quanti_Attr | Nominal_Attr

Name	Color
graph	🟥
networks	🟩
pathway	🟦

Preview

Operation

● Union

○ Intersection

Lens 2

Name & Type

Name: mation Visualization

Type: StarPlot_Var2

Quanti_Attr | Nominal_Attr

Name	Color
data	🟨
information	🟧
method	🟪
visualization	🟫

Preview

Combined Lens

Name & Type

Name: Network Visualization

Type: Bar Chart

Quanti_Attr | Nominal_Attr

Name	Color
data	🟨
graph	🟥
information	🟧
method	🟪
networks	🟩
pathway	🟦
visualization	🟫

Preview

Figure 6.4: A dialog box used to combine different lenses.

Implementation

The prototype was implemented in Java using the JUNG [86] graph drawing library. The developer can use a set of predefined node and edge visualizations as well as implement own visual representations. Beside different graph layout algorithms, it offers a lot of other functions that are already implemented, such as zooming and panning interactions.

JUNG also contains the implementation of a lens, which has the functionality of a normal magnifying and fisheye lens. We appreciated the idea of having an already implemented distortion based lens. Thus, we added the functionality to change the shape of the nodes each time the lens is moving over those. In this way, we only needed to implement the planned visualization metaphor and let the library take care for the distortion.

Our prototype uses GraphML files as data input [17], 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 is not flexible enough. So, we had to implement our own parser due to the need to parse different GraphML files having different attributes, especially for future needs.

6.2 *Application Scenarios*

Biological Data

The main idea behind this prototype is to be used for any type of multivariate network regardless of the application domain. We use a hand made copy of the biological network data used in the work of Borisjuk *et al.* [13]. The visualization of this data using their tool can be seen in Figure 3.3. The figure represents experimental data in context of the metabolic network: glycolysis and citric acid cycle. Each bar inside 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 economically important protein source in food industry. The visualization of these data as shown in Figure 3.3 helps biologists to see the effect of transgenic technology on the plant metabolism.

Figure 6.5 shows that our tool can mimic the standard integrated approaches and can handle the biological data. However as explained in Section 3.2, integrated approaches do not cope well spatially when the number of network elements increases and/or in case of high attribute numbers. In such cases, the use of Network Lens could alleviate the problem as there will be no need to increase the size of the nodes. Figure 6.6 shows the lens applied to a specific part of the network. The rest of the nodes can show different glyphs while we investigate different local parts of the network.

Text Documents

In this subsection we will illustrate how is our prototype tool used to explore multivariate networks. For this purpose, we used a set of research papers (24 text documents) published by our group to build a multivariate network dataset. The documents are represented by nodes in a network. 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 attributes. This degree is represented as the weight of the edge (shown as the thickness of the edge line). We filtered out frequently used words and did not include them 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 6.7(a).

Let us assume we want to explore these documents without reading them. We are interested to find out about the documents that are related to the topic of *algorithms*. Therefore, the color saturation of the nodes is mapped to the value of the attribute *algorithm*. Higher values of the attribute, namely higher occurrences of

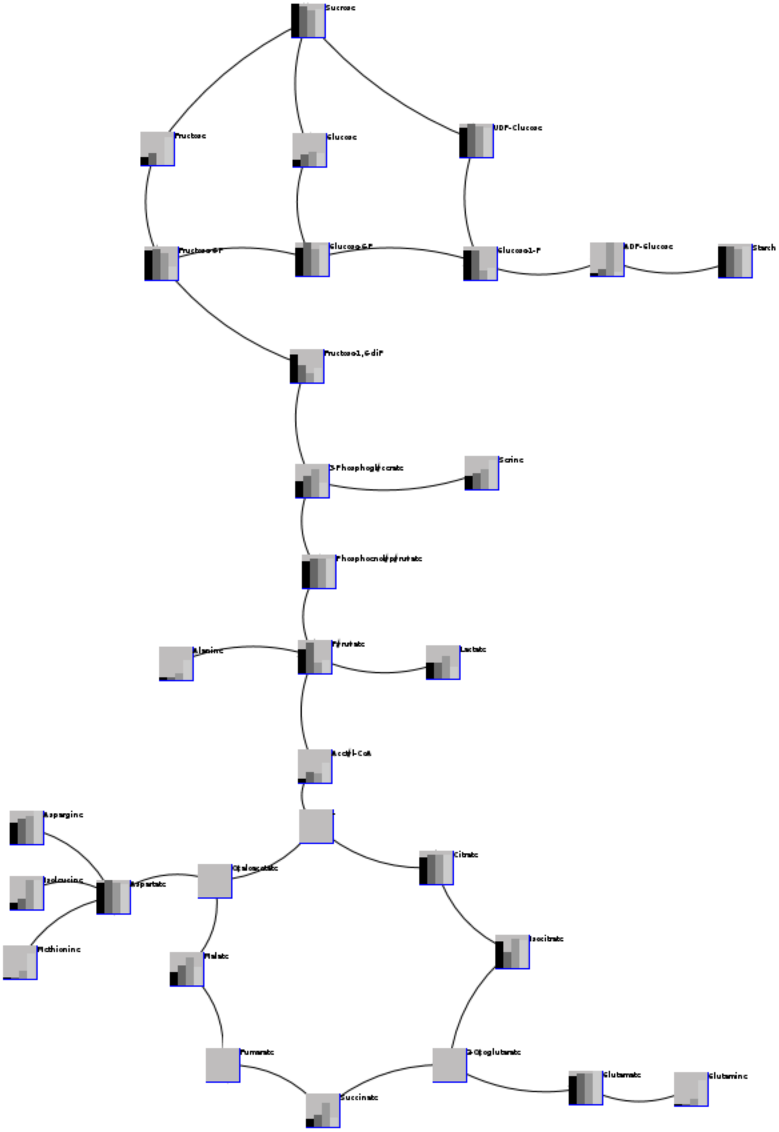


Figure 6.5: Visualization of experimental data integrated into a biological network. Based on the data shown in Figure 3.3.

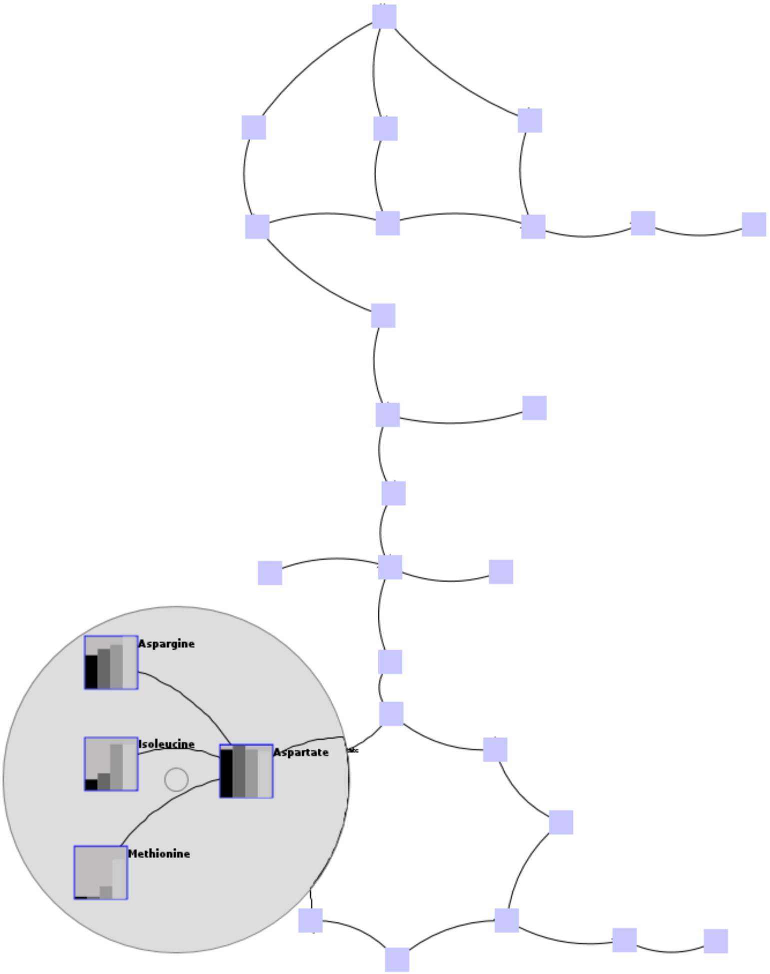
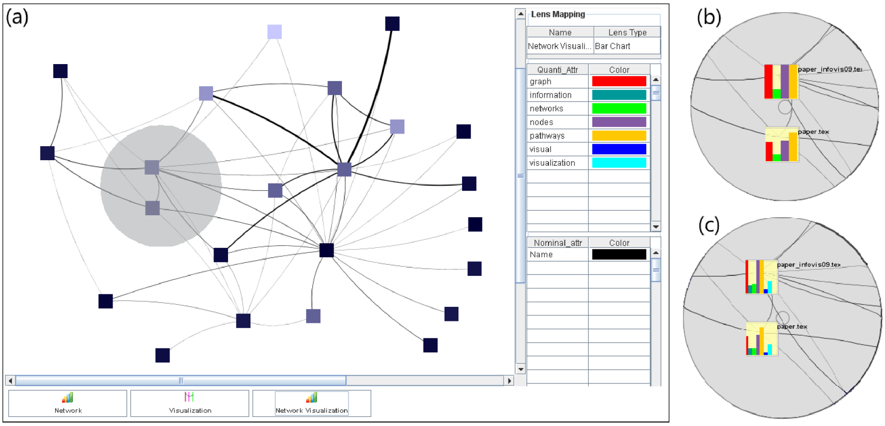


Figure 6.6: Visualization of experimental data by using the Network Lens. Based on the data shown in Figure 3.3.



Lens Mapping

Name	Lens Type
Network Visuali...	Bar Chart

Quanti_Attr	Color
graph	🟥
information	🟦
networks	🟩
nodes	🟪
pathways	🟨
visual	🟦
visualization	🟦

Nominal_attr	Color
Name	⬛

Figure 6.7: 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.

this word in a document are represented by the lighter colors as shown in our screenshot example. We can can immediately identify several documents with high value, but a lot of documents could discuss *algorithms* in different context. Therefore, we narrow our exploration to those documents with content related to *algorithms* and *networks* (or graphs). At this point we create a new lens and select attributes (keywords in this case) such as *networks*, *graphs*, *nodes*, etc., that would give insight to documents related to networks. We name this lens Network and start the exploration. We focus the lens on the lighter colored nodes (those with content about algorithms). Here, we discover two documents with relatively high values of specific attributes, see Figure 6.7(b). We notice a high frequency of the words *graph*, *nodes*, and *pathways* 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, we load a previously stored lens named Visualization. This lens is combined with our current lens (Network) using the Union set operator in order to create a new lens Network Visualization. We continue the exploration of those documents again, as we created a tool (new lens) to get more insight about the documents connected to network visualization and algorithms. The new lens reveals that the couple of documents identified earlier fit our criteria as shown in Figure 6.7(c).

6.3 Summary

In this chapter, we presented a novel approach for interactive visualization of multivariate networks that 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 overloading issue of the latter technique. Our system offers a freedom in terms of attribute filtering and selection of effective visual representations. This makes the approach easy to generalize to 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.

Chapter 7

Conclusion and Future Work

In this chapter, we summarize this thesis and discuss our findings in context of the goal criteria presented in Chapter 1. Finally, we conclude the thesis with a discussion on future work.

One of the most challenging tasks in the Information Visualization and Graph Drawing communities is the visualization of large and complex networks [78]. This, combined with the need to visualize and analyze such data, makes network visualization a hot topic. Recently, a considerable work on graph visualization has been done and powerful tools have been introduced. However, some fundamental problems are still not adequately solved. The visualization of additional data that are attached to the network nodes and/or edges is one important problem. Another example of such problems is related to scalability issues introduced when visualizing huge networks. Drawings of these networks are often visually overloaded (cluttered) and do not scale. To focus on the latter problem, one traditional solution is a hierarchical structuring of the entire network if possible, that is, the complete network is divided into many parts. This is also a common procedure in biochemical network drawings. This approach creates difficulties in navigation and understanding the overall interlink between these pathways. Additionally, several network nodes are duplicated to avoid further clutter. These, and other issues related to biological and network visualization in general, are discussed in Chapter 2.

In Chapter 3, a more detailed analysis of the techniques and methods for multivariate networks is presented. Based on specific visualization techniques, we defined a number of criteria and presented a system for multivariate network visualizations where different approaches are categorized based on the aforementioned criteria and information, such as type of the attributes or the domain the tool was used for.

In Chapter 4, we present an approach to overcome the drawbacks of the aforementioned splitting of large networks into pieces. Our visualization approach enables users to analyze a focus pathway, while providing means to navigate and have an overview of other interlinked pathways. We use glyphs, brushing and pathway topological information to provide meaningful contextual information to the analyst in an intuitive way. The main idea of our approach is to lay out minimized images of pathway drawings around the main view that displays the focus pathway. These graph icons are additionally used to navigate to a specific pathway by mouse clicks, beside giving an overview of the connections between pathways. Mouse-over actions on the graph icons highlights the nodes that are interlinked in the focused pathway, and an additional rose diagram enforces the perception of pathway interconnections even further. Another important aspect of this work is that we have realized our approach as plugin of an already well known and widely used visualization framework.

We discussed that showing the interconnections of pathways of a single biological network is challenging. In Chapter 5, we deal with a slightly different problem as we present an approach to show two different types of large networks and the connections between them. As both ontologies and hierarchical clustering are important for biological analysis, it is desirable to show them both in the same view. Ontologies are modeled as large DAGs, while hierarchical clustering data form large binary trees. Our approach uses brushing and innovative layout designs to cope with the complexity and amount of data.

We present the novel layout approaches for drawing the DAG by using pixel-based approaches. Two variations of this approach are implemented and strengths and weaknesses of each are discussed. Both variations place intermediate nodes of the GO into specific hierarchical levels, and thus differ only in the way terminal nodes are placed. One approach arranges all terminals into the last level, emphasizing the connections of each node to the terminals. The other one positions terminal nodes into hierarchical levels accordingly, but on separate regions of the level, thus giving an overview about the distribution of the terminals and intermediate nodes in each level.

A new drawing algorithm for representing huge unbalanced binary trees is introduced. The main idea behind this algorithm is to use nodes and edges that form the longest path that connects all branches as a “backbone”, which we lay out in form of a spiral, thus forming our *Spiral Tree Layout*. Our layout uses the space more effectively for this particular data set, i.e., huge unbalanced binary trees, then traditional tree layout techniques. Common brushing and interaction techniques are then used to show the interconnections between these two graphs.

At this point in the thesis, we have presented two contributions in context of visualization of large biological networks. The first contribution (Chapter 4) enhances the navigation and gives an overview into the connections of different separated pathways within a larger biological network. In the second work (Chapter 5) , we present new ways to visualize the interrelationship of two huge graphs of different types. Our next contribution deals with the issue of additional attributes connected to the nodes of the network. Therefore in Chapter 6, we present the “Network Lens”, a prototype for the visualization of multivariate networks.

Our prototype is essentially an extension of the traditional magic lens idea applied to networks represented as node-link graphs. By interactively specifying different desired attributes and selecting different visual representations, users can build various lenses. Moreover, the lenses can be combined with each other with the help of set operators to create new lenses. This way of combining lenses makes the whole process of creating new lenses and the overall exploration process more intuitive as it follows an optics metaphor. Each time a lens is applied on a node, it will show the peculiar attributes in particular visual representation depending on the user specification when the lens was created. Moreover, lenses can be saved for later explorations. We have shown an example where we visualize a specific biological pathway by using our prototype.

7.1 Discussion

In this section, we discuss the results of the thesis in context to the goals we have defined in Chapter 1, which are:

1. Offer a contribution to the problem of a visualization of huge biologic networks. More specifically, improve shortcomings of the approach of dividing larger biological networks into smaller pieces, and contribute to the problem of a visualization of different types of interconnected biological networks.
2. Offer a contribution for the visualization of multivariate biological networks.

In the following subsections, we compare our results with the set of criteria for each goals (cp. Chapter 1).

Goal 1

The criteria for the first goal are the following:

1.1 Dividing networks into smaller units in order to avoid clutter and complexity when visualizing huge biochemical networks introduces issues of loosing the overview. Nevertheless, this practice is desirable by the biologists. Therefore, an analysis of these issues and a contribution towards solving the problems shall take place.

1.2 Provide an approach to combine two different types of huge networks. As described in the motivation section on Page 3, a visualization of biological networks in context of tree-like data produced by clustering is desirable. Therefore, a tool to visualize both datasets and to show their connection will be developed.

Criterion 1 is fulfilled: We developed a tool called GLIEP, which uses various interaction and visualization ("Aggregated Links View") techniques in combination with our "Navigation Glyphs" and "Graph Icons" as presented in Section 4.1 in order to guide the interactive navigation and to show the relationship between interconnected pathways. GLIEP is a plugin of a well-known tool (VANTED) for researchers in the domain of biological networks and is now a part of an official list of supported plugins. We have presented the details of this work in Chapter 4.

Criterion 2 is fulfilled: A new visualization approach to combine ontologies and hierarchical clustering data was presented in Chapter 5. Both, ontologies and clustering data are large relational data sets that are conceptually different. We used interaction and brushing techniques to combine them, while providing new layout metaphors to cope with the amount and complexity of both data sets.

Goal 2

The criteria for the second goal are the following:

2.1 Provide a survey on the current state of the art on the visualization of multivariate networks in general. We shall not only focus on biological networks as the problems and solution from other research domains could be easily adapted for biochemical networks.

2.2 Introduce a novel approach for visualizing multivariate networks.

Criterion 1 is fulfilled: In Chapter 3, we present a survey on multivariate network visualization. We investigate different approaches used in practice and developed a set of criteria to classify these approaches into different categories.

Criterion 2 is fulfilled: We extend a general magic lens approach and apply it to explore multivariate networks. Our tool uses the strengths of the integrated approach while avoiding its weaknesses.

7.2 Future Work

So far, we have presented different approaches to deal with huge and complex network data. As future work for the GLIEP plugin presented in Chapter 4, we plan to enhance various features and improve different issues related to the visualization and interaction possibilities. Most of these improvements will be regarded as side projects as they will not directly relate to multivariate network visualization, however they could be used to support or enhance further steps into that direction. Research results in large graph exploration, such as the DOI-approach of van Ham and Perer [119], could be interesting for our system too. We plan to analyze the possibility of integrating such or similar techniques. As mentioned before, data sets with a deeper level of hierarchies exist. At the moment, our plugin supports only two-level hierarchies. This is enough for the MetaCrop data set. Theoretically, our approach could be extended by using the Sunburst approach presented by Stasko and Zhang [112] or Richard *et al.* [110]. The visualization of additional statistical data is the most interesting future work unit related to GLIEP. The Link rose diagram could be extended to support this. The color-coding of the rose diagram and the navigation glyphs could be improved as well, perhaps even including a new view to visualize multivariate network data.

Similarly, CluMa-GO has a lot of possibilities for improvements that are not directly related to the multivariate network visualization (cp. Chapter 5). However, the following improvements should take place as side projects as well. The current state of the prototype does not provide a way to visualize a direct mapping between a terminal GO DAG node and a cluster tree leave. A simple way to overcome this problem for a specific node is to highlight the corresponding nodes in the GO view and/or Cluster Tree view on mouse-over action. This could be easily implemented as a part of our future work. As explained in the Section 5.2, the zoomed-in GO view shows three levels at the same time while displaying the subgraph by highlighting the nodes only. The edges are omitted due to clutter problems that can occur since edges from a higher level might go through the zoomed-in view to nodes in the lower layers. Since we are zoomed in, this does not make sense to show, because we have no insight from which layer those edges are coming from, nor to which layer they are going to. However, an improvement is possible by showing only edges between the three layers shown in the zoomed-in GO view. At the same time, the edge bundling algorithm could also be improved. We are also working on an improved version of our spiral tree metaphor to cope with more balanced binary trees. One possible solution is to create something we call "nested spiral trees". The idea is to draw smaller spirals instead of aggregating larger subtrees that pass over a certain threshold of nodes into box glyphs. This approach will however introduce more unused spaces, making the approach less space-filling.

As a main research direction, we will continue the work on multivariate network visualization. However, we will not focus only on large and complex networks or on the domain of biological networks. As explained in Chapter 1, most of the approaches to visualize multivariate networks can be generalized across different domains. Therefore, the plan is to not limit us in visualizing only the datasets belonging to one domain. We are already working on improvements of the Network Lens. The next step is designing a usability study, which will be used as a guide for further improvements of the tool. We will implement more visual representations, which will allow for even greater flexibility of the tool. Furthermore, we can experiment with time-dependent data and data quality issues.

The following discussion is not related to any of the tools described in this thesis, but should be taken as a general guideline for future work. Even networks of few hundreds of nodes and edges can often be hard to visualize without the use of some interaction or filtering technique. Adding multivariate data makes this process even more challenging. Therefore, moving into the direction of Visual Analytics could be helpful to cope with the problem. Visual Analytics could be roughly defined as use of Information Visualization together with Data Mining [129, 63]. The aim is to develop tools to support multivariate network visualization that would mainly depend on known visualization techniques in combination with Data Mining. These tools should rely on seamless interaction techniques that will facilitate the process of visual analysis. Thus, users should be able to analyze the network and the attached data interactively.

Bibliography

[1] *Streptomycin biosynthesis - Reference pathway*, last accessed: 2011-11-22. 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] Mario Albrecht, Andreas Kerren, Karsten Klein, Oliver Kohlbacher, Petra Mutzel, Wolfgang Paul, Falk Schreiber, and Michael Wybrow. A graph-drawing perspective to some open problems in molecular biology. Technical report, Lehrstuhl XI für Algorithm Engineering, Fakultät für Informatik, Technische Universität Dortmund, 2008.

[4] Mario Albrecht, Andreas Kerren, Karsten Klein, Oliver Kohlbacher, Petra Mutzel, Wolfgang Paul, Falk Schreiber, and Michael Wybrow. On open problems in biological network visualization. In *Proc. International Symposium on Graph Drawing (GD '09)*, volume 5849 of *LNCS*, pages 256–267. Springer, 2010.

[5] Vladyslav Aleksakhin. Visualization of Gene Ontologies and Cluster Analysis Results. Master's thesis, Linnaeus University, Sweden, 2012 (to appear).

[6] 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.

[7] David Auber, Yves Chiricota, Fabien Jourdan, and Guy Melançon. Multi-scale visualization of small world networks. In *IEEE Symposium on Information Visualization (InfoVis '03)*, pages 75–81, Los Alamitos, CA, USA, 2003. IEEE Computer Society.

[8] Christian Bachmaier, Franz-Josef Brandenburg, Michael Forster, Paul Holleis, and Marcus Raitner. Gravisto: Graph visualization toolkit. In J. Pach, editor, *Proc. International Symposium on Graph Drawing (GD'04)*, volume 3383, pages 502–503. Springer, 2005.

[9] 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.

[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* [55], pages 207–232.

[15] Romain Bourqui, Vincent Lacroix, Ludovic Cottret, David Auber, Patrick Mary, Marie-France Sagot, and Fabien Jourdan. Metabolic network visualization eliminating node redundance and preserving metabolic pathways. *BMC Systems Biology*, 1:29, 2007.

[16] Ulrik Brandes, Tim Dwyer, and Falk Schreiber. Visual understanding of metabolic pathways across organisms using layout in two and a half dimensions. *Journal of Integrative Bioinformatics*, 1(1):119–132, 2004.

[17] 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.

[18] Cynthia A Brewer. *ColorBrewer*, 2nd edition, last accessed: 2011-04-29. http://colorbrewer2.org/.

[19] Stuart Card, Jock Mackinlay, and Ben Shneiderman, editors. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, 1999.

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

[21] 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.

[22] 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.

[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] 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.

[28] 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.

[29] Niklas Elmqvist, John T. Stasko, and Philippas Tsigas. DataMeadow: A Visual Canvas for Analysis of Large-Scale Multivariate Data. *2007 IEEE Symposium on Visual Analytics Science and Technology*, pages 187–194, 2007.

[30] Niklas Elmqvist and Philippas Tsigas. Causality visualization using animated growing polygons. *IEEE Symposium on Information Visualization 2003*, 2003:189–196, 2003.

[31] 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.

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

[33] Ursula Gaedke. Ecological Networks. In *Analysis of Biological Networks* [55], pages 283–304.

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

[35] Birgit Gemeinholzer. Phylogenetic Networks. In *Analysis of Biological Networks* [55], pages 255–281.

[36] Gene ontology, last accessed: 2011-04-29. http://www.geneontology.org/.

[37] Google. Guava: Google Core Libraries for Java 1.5+, last accessed: 2011-04-29. http://code.google.com/p/guava-libraries/.

[38] Carsten Görg, Mathias Pohl, Ermir Qeli, and Kai Xu. Visual Representations. In Kerren et al. [66], pages 163–230.

[39] Eva Grafahrend-Belau, Stephan Weise, Dirk Koschützki, Uwe Scholz, Björn H. Junker, and Falk Schreiber. MetaCrop - a detailed database of crop plant metabolism. *Nucleic Acids Research*, 36:D954–D958, 2008.

[40] 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.

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

[42] 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.

[43] Nathalie Henry, Anastasia Bezerianos, and Jean-Daniel Fekete. Improving the readability of clustered social networks using node duplication. *IEEE Transactions on Visualization and Computer Graphics*, 14:1317–1324, 2008.

[44] 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.

[45] 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.

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

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

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

[49] 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.

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

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

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

[53] 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.

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

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

[56] Ilir Jusufi, Yang Dingjie, and Andreas Kerren. The network lens: Interactive exploration of multivariate networks using visual filtering. *Information Visualisation, International Conference on*, 0:35–42, 2010.

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

[58] Ilir Jusufi, Andreas Kerren, Vladyslav Aleksakhin, and Falk Schreiber. *GLIEP Project Homepage*, last accessed: 2011-12-04. http://sourceforge.net/projects/gliep.

[59] Ilir Jusufi, Christian Klukas, Andreas Kerren, and Falk Schreiber. Interactive Navigation in Interconnected Biochemical Pathways. In *Interactive Poster, InfoVis 10.*, Salt Lake City, Utah, USA, 2010.

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

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

[62] Peter D. Karp and Suzanne Paley. Automated drawing of metabolic pathways. In H. Lim, C. Cantor, and R. Bobbins, editors, *Proc. International Conference on Bioinformatics and Genome Research*, pages 225–238, 1994.

[63] 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.

[64] 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.

[65] Andreas Kerren. Interactive visualization and automatic analysis of metabolic networks – a project idea. Technical report, Institute of Computer Graphics and Algorithms, Vienna University of Technology, Austria, 2003.

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

[67] 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 02.-06.03.2009 in Kaiserslautern*, volume 150 of *LNI*, pages 147–154. GI, 2009.

[68] 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.

[69] 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.

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

[71] Christian Klukas and Falk Schreiber. Dynamic exploration and editing of KEGG pathway diagrams. *Bioinformatics*, 23(3):344–350, 2007.

[72] Christian Klukas and Falk Schreiber. Integration of -omics data and networks for biomedical research. *Journal of Integrative Bioinformatics*, 7:112, 2010.

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

[74] 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.

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

[76] 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.

[77] 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.

[78] Robert S. Laramee and Robert Kosara. Challenges and Unsolved Problems. In Kerren et al. [66], pages 231–254.

[79] Alexander Lex, Marc Streit, Ernst Kruijff, and Dieter Schmalstieg. Caleydo: Design and evaluation of a visual analysis framework for gene expression data in its biological context. In *2010 IEEE Pacific Visualization Symposium (PacificVis)*, pages 57–64, Taipei, Taiwan, 2010.

[80] Mircea Lungu and Kai Xu. Biomedical Information Visualization. In Kerren et al. [66], pages 311–342.

[81] Kazuo Misue, Peter Eades, Wei Lai, and Kozo Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages and Computing*, 6:183–210, 1995.

[82] Tomer Moscovich, Fanny Chevalier, Nathalie Henry, Emmanuel Pietriga, and Jean-Daniel Fekete. Topology-aware navigation in large networks. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 2319–2328, New York, USA, 2009. ACM.

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

[84] 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.

[85] Florence Nightingale. *Notes on Matters Affecting the Health, Efficiency, and Hospital Administration of the British Army.* Harrison & Sons., 1858.

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

[87] 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.

[88] Catherine Plaisant, Jesse Grosjean, and Benjamin B. Bederson. Spacetree: supporting exploration in large node link tree, design evolution and empirical evaluation. In *IEEE Symposium on Information Visualization (InfoVis '02)*, pages 57–64, Los Alamitos, CA, USA, 2002. IEEE Computer Society.

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

[90] Anatolij P. Potapov. Signal Transduction and Gene Regulatory Networks. In *Analysis of Biological Networks* [55], pages 183–206.

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

[92] Ramana Rao and Stuart K. Card. The table lens: merging graphical and symbolic representations in an interactive focus+context visualization for tabular information. In *CHI '94: Conference companion on Human factors in computing systems*, page 222. ACM, 1994.

[93] 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.

[94] 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.

[95] 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.

[96] Markus Rohrschneider, Christian Heine, André Reichenbach, Andreas Kerren, and Gerik Scheuermann. A novel grid-based visualization approach for metabolic networks with advanced focus&context view. In *Proc. International Symposium on Graph Drawing (GD '09)*, volume 5849 of *LNCS*, pages 268–279. Springer, 2010.

[97] 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.

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

[99] Purvi Saraiya, Chris North, and Karen Duca. Visualizing biological pathways: requirements analysis, systems evaluation and research agenda. *Information Visualization*, 4(3):191–205, 2005.

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

[101] Falk Schreiber. High quality visualization of biochemical pathways in BioPath. 2(2):59–73, 2002.

[102] Falk Schreiber, Tim Dwyer, Kim Marriott, and Michael Wybrow. A generic algorithm for layout of biological networks. 10:375.1–12, 2009.

[103] 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.

[104] 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.

[105] 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.

[106] 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.

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

[108] Márcio Rosa da Silva, Jibin Sun, Hongwu Ma, Feng He, and An-Ping Zeng. Metabolic Networks. In *Analysis of Biological Networks* [55], pages 233–253.

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

[110] John T. Stasko, Richard Catrambone, Mark Guzdial, and Kevin Mcdonald. An evaluation of space-filling information visualizations for depicting hierarchical structures. *International Journal of Human-Computer Studies*, 53:663–694, 2000.

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

[112] 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.

[113] 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.

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

[115] 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.

[116] 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.

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

[118] 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.

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

[120] 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.

[121] 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.

[122] Matthew O. Ward. Xmdvtool: integrating multiple methods for visualizing multivariate data. In *VIS '94: Proc. Conference on Visualization '94*, pages 326–333, Los Alamitos, USA, 1994. IEEE Computer Society Press.

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

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

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

[126] 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.

[127] Stephan Weise, Ivo Grosse, Christian Klukas, Dirk Koschuetzki, Uwe Scholz, Falk Schreiber, and Bjoern H. Junker. Meta-All: a system for managing metabolic pathway information. *BMC Bioinformatics*, 7:465, 2006.

[128] 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.

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

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

[131] 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.

[132] 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.

[133] yWorks. *yEd Graph Editor*, last accessed: 2011-10-29. http://www.yworks.com/en/products_yed_about.html.

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