Linnæus University
School of Computer Science, Physics and Mathematics

Degree Project

The Network Lens

Dingjie Yang
2010-06-30
Subject: Software Technology
Level: Master
Course code: 5DV01E

School of Computer Science, Physics and Mathematics

Reports from DFM

The Network Lens

Student: Yang Dingjie

Supervisors: Prof. Dr. Andreas Kerren and Ilir Jusufi

Linnaeus University

Abstract

A complex network graphics may be composed of hundreds and thousands of objects, such as nodes and edges. Each object may hold a large number of attributes that might be difficult to explore in the network visualization. Therefore, many visualization tools and approaches have been developed to gain more information from the network graphics. In this paper, we introduce the concept of the Network Lens, a new widget that assists the users to deal with a complex network. The Network Lens is an interactive tool that combines data visualization to a magic lens. With the help of the Network Lens, users can display hidden information of the elements in the network graphics based on their interest. Moreover, the Network Lens supports a series of interactive functions that give the users flexible options to define their own lenses.

Keywords: magic lenses, node attributes, network visualization, GraphML

Table of Contents

Tables of Figure

List of Tables

1 Introduction

Whereas attributes of the objects in a small network can be directly depicted in the network graph with a node-link metaphor, some complex networks with a large amount of objects and with plenty of attributes cannot be visualized that easily due to space limitation and clutter. How to enable the users to conveniently access the attributes of the arbitrary objects in a network with hundreds and thousands of objects is the topic of several scientific researches.

In this paper, we present a new approach to solve this problem by applying magic lenses [9] and multivariate data visualization.

1.1 Background

Network analysts use one kind of graphic display that consists of nodes to represent objects and edges to represent the ties or relationships between the objects to analyze a network. Figure 1.1 shows a very simple social network. In the figure, it describes the employees' activity in an organization. Each node represents an employee with the color of the node to represent the department she/he is working for. If two employees have talked to each other, there would be a link connecting the two nodes. As such a network is demonstrated to the users, a larger amount of information about the network is not shown in the graphics. In context of the network showed in the figure, each node of the diagram may hold many attributes, such as *age*, *name* and *department*. Whereas, in the following graphics, only the attribute, *name*, is displayed.

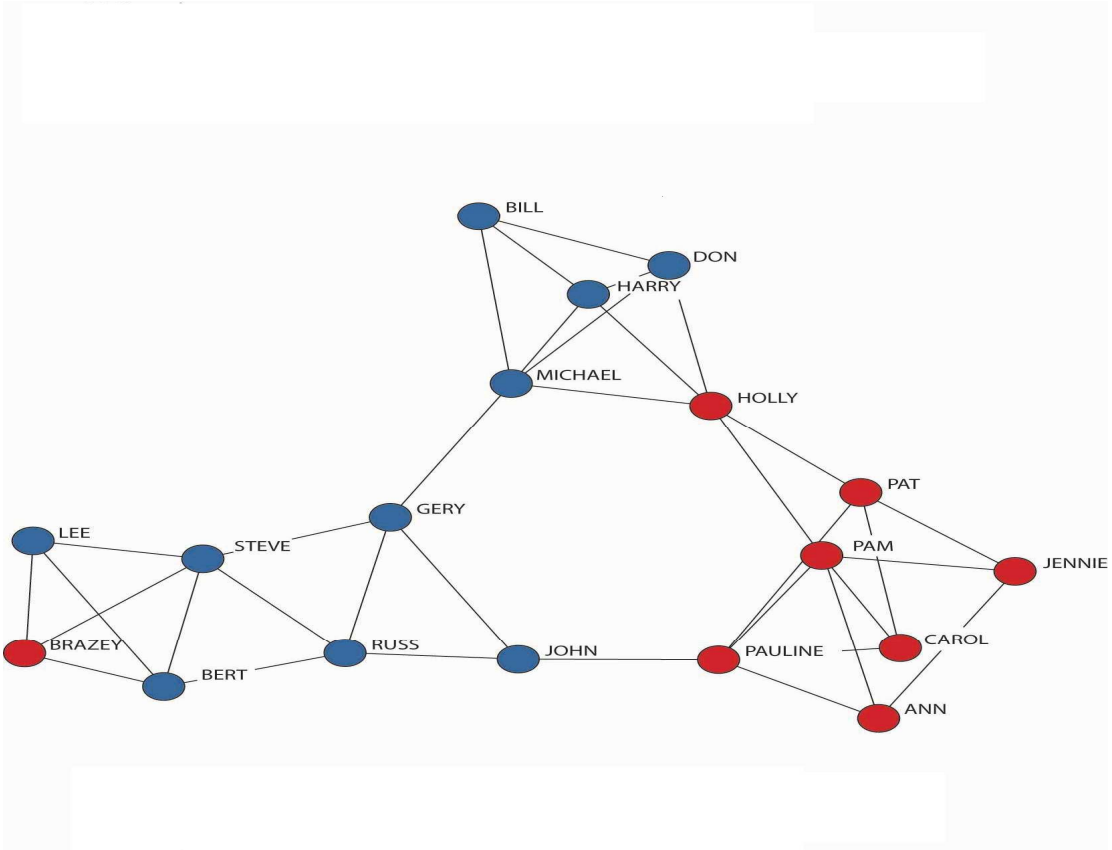

Figure 1.1 A Social Network. Taken from [1]

There are several methods about how to visualize the attributes information of a network. One simple approach is to store and draw the attributes of the nodes in a separate window. The user only needs to open the document and search the required information about the nodes and the relationships between them. However, one disadvantage of this method is that the user has to switch between the displays, which is time consuming and discommodious [53].

The other way is to attach all the attributes of the nodes to the network graphics so that the user can avoid switching between the displays. Unfortunately this method may cause clutter of the diagram since the attributes might overlap each other. An approach that can both save time and avoid clustering will be described later in this paper.

1.2 Goal of this Thesis

In this work, our goal is to develop a new application, called *Network Lens*, which will help users to explore more information of a network without clutter and time-consuming operation. The new application, *Network Lens*, will take advantages of a magic lens [9] to filter information from a graphics; meanwhile, it will add some new features to visualize the information that is filter from the network graphics. In short, our final output of the thesis is a combination of magic lenses and data visualization, where a magic lens is employed to filter the information while data visualization is applied to represent the information that is filtered by the magic lens.

With the help of this application, users are able to explore and analyze information from a network diagram by creating various lenses according to their interest. Moreover, a series of interactive functions are designed for users to make the best of this tool.

1.3 Outline

The chapters followed will present the related work and how our approach is achieved. In Chapter 2, we demonstrate the related expertise that has contributed to our work and made our work possible and easy to accomplish. In Chapter 3, we analyze and discuss our approach by referring to the related expertise and present how we achieve the final object of our work and why our work stands out from others. The concrete method of how to implement the work and how to use this application will be described in Chapter 4. The advantages and limitation of our tool will be discussed in Chapter 5. A conclusion that is composed of a summary of this paper and the further work to make improvement of the current application will be mentioned in Chapter 6.

2 Related Work

Some previous work provides a lot of theory and techniques necessary to implement this *Network Lens* tool. As it is depicted in Chapter 1, the goal of our work is to develop an application of the traditional magic lens to explore the attributes of a network. The related work can be categorized into the following several aspects: Interactive Information Visualization, Graph Drawing, Network Visualization, Magic Lens and Data Visualization. Section 2.1 gives a general idea about the subject of information visualization. The following Section 2.2 will address some important aspects of graph drawing. In Section 2.3, some active methods of network visualization are described and analyzed. Section 2.4 will presents a brief but clear insight of magic lens in order to help reader to set up a good understanding of the *Network Lens*. Section 2.5 will introduce some techniques in data visualization.

2.1 Interactive Information Visualization

2.1.1 Definition and Overview of Information Visualization

"Information visualization is a relatively new research area, which focuses on the use of visualization techniques to help people to understand and analyze data." [17]. Although it is relatively new research area, Information visualization reaches more and more fields in nowadays life. "Information visualization is a rapidly growing field that is emerging from research in human-computer interaction, computer science, graphics, visual design, psychology, and business methods. Information visualization is increasingly applied as a critical component in scientific research, digital libraries, data mining, financial data analysis, market studies, manufacturing production control, and drug discovery." [18].

Many definitions are given about information visualization, "Information visualization (InfoVis) is the communication of abstract data through the use of interactive visual interfaces." [2]. "Information visualizations attempt to efficiently map data variables onto visual dimensions in order to create graphic representations." [41]. "A method of presenting data or information in non-traditional, interactive graphical forms. By using 2D or 3D color graphics and animation, these visualizations can show the structure of information, allowing one to navigate through it, and modifying it with graphical interactions." [42]. No matter how and when the definition about information visualization is proposed, in the description of information visualization, two key words are involved in the definition, one is *abstract data* and the other is *graphic representation*. In other words, the aim of the information visualization is to transfer a set of *abstract data* to a *graphic representation*.

Ed H. Chi developed the information visualization reference model in 1991 [36]. The following. Figure 2.1 is used to depict the model. This model is widely used to develop some information visualization application. This model breaks up the whole information visualization process into a series of discrete steps, which is helpful to enhance the operation ability and readability of the process.

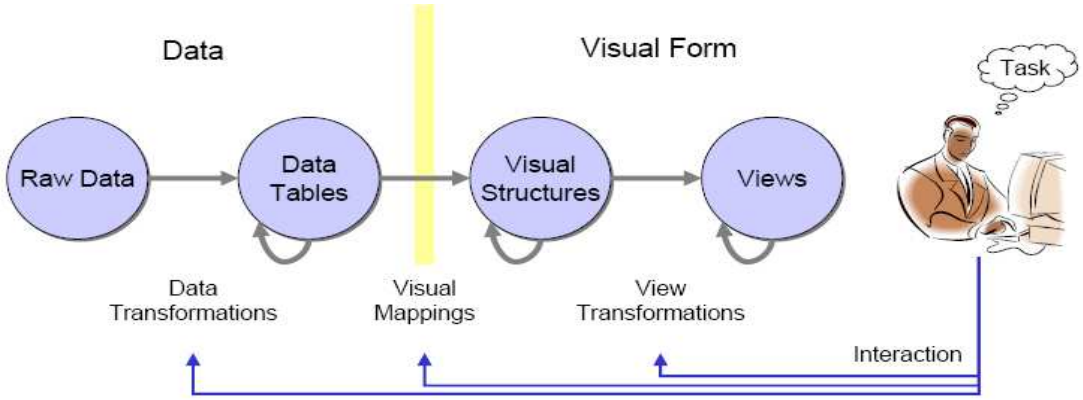

Figure 2.1 Information Visualization Reference Model. Taken from [35]

2.1.2 Important Issues of Information Visualization

Although the model clearly interprets the process or steps to develop information visualization products or tools, considerable issues still remain unresolved, such as the problems like *perception***,** *presentation*, and *interaction*, etc. All these issues are virtually inevitable when one intends to implement application that is connected with information visualization. In the book, *Information Visualization: Perception for design* [3]**,** the author introduces some basic principles in the discipline of information viusualization, such as the cognitive principles. Besides that, a large amount of examples are presented in the book to provide some guidance to the developers. Another excellent book is *Information visualization: design for interaction* [4], the presentation issues, such as scrolling, zooming and panning, overview + detail, Focus + Context and magic lenses are clearly described by the author. Figure 2.2 is an application that applies Overview+ Detail technique.

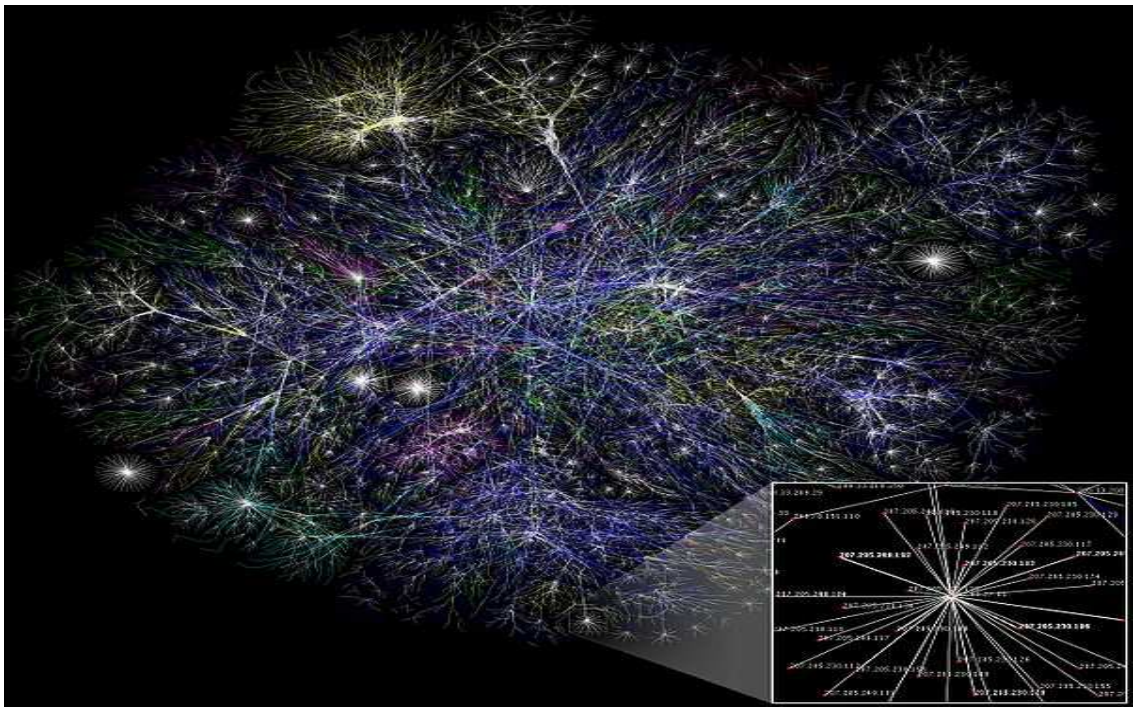

Figure 2.2 Partial map of the Internet early 2005. Taken from [53]

The issues that are discussed above cannot cover all the aspects during the process to develop a visualization tool, but they can give the developer a general direction and reference with respect to a specific application.

2.2 Graph Drawing

2.2.1 Brief Introduction of Graph Drawing

In this work, we distinguish between graph drawing and network visualization. Graph drawing means to display the graph with a set of vertices and edges, while as network visualization means a technique for the presentation of the vast volume of data yielded in such networks. Graph drawing plays a basic role in network visualization and is used to set up the basic topological structure of the network, while network visualization deals with both the attributes of the graph and the graph itself.

The importance of graph drawing leads to various corresponding works that have been studied. There are three basic techniques or approaches for drawing a graph in 2D: connecting with nodes and links, adjacency matrix and converting to tree. “Node-link graphs are simple, powerful, and elegant abstractions that have broad applicability in computer science and many other fields.” [19]. Since a node-link representation is simple and powerful, we will just consider the node-links representation of network in this thesis.

When a graph is drawn by using a node-link metaphor, several challenges may rise, such as spatial layout of graph; scale up to a large graph when the network is huge. In the following part of this subsection the related knowledge about theses work will be presented.

2.2.2 Graph Layout

“Graph drawing or Graph layout, as a branch of graph theory, applies to topology and geometry to derive two-dimensional representations of graphs. A drawing of a graph is basically a pictorial representation of an embedding of the graph in the plane, usually aimed at a convenient visualization of certain properties of the graph in question or of the object modeled by the graph.” [21].

A good layout for a graph will give users a good perception and understanding of the graph. But there is no best layout for drawing graph. Different graph layouts highlight different characteristics [43]. For example, Force Directing layout is very easy to understand since they are based on physical analogies of common objects, like springs, while circular layout can help you find out the critical nodes of your graph in an intuitive way.

However, it is always a challenge to develop a good layout, since it is usually complicated and time consuming and “suffice it to say that most interesting computations on general graphs are NP-hard” [20]. Therefore, many developers and researchers groups have already developed various and abundant algorithms and toolkits for graph drawing, such as Common Graph Layout Algorithms, which contains Circular, Concentric, Layering, Force directed, Clustering, Hierarchical clustering, Geographic, Pre-determined. In Eren’s work [5], he listed some algorithms

and methods for drawing network. By using these algorithms and methods, users can customize their own network drawing to meet the demand.

Besides providing layout algorithms to expedite the process of graph drawing, much more thrilling thing is that many toolkits such as JUNG [6], PREFUSE [7], PAJEK [8], etc even provide a useful visualization library to the users. Most of them are open sources, it is very convenient and handy for a user to refer to the library and invoke the classes and methods in it. The toolkit JUNG is used in this thesis. The following Figure 2.3 is a network graph with a circular layout.

2.2.3 Scalability

If the datasets of graphs are very big and complex, when one tries to display them on the screen, the problems of how to scale up a large graph will emerge since it is too big to display all the view in a screen at once. Eick and Karr defined visual scalability as the ability of visualizations to effectively display large amounts of data. They structured the issue by providing the factors affecting it which include human perception, the visual metaphor and the display as well as algorithms and computation [44]. In the book [4], scrolling, Focus + Context [45], Overview + Details [46], etc, several used approaches are presented to solve this problem. In addition, J. D. Fekete presents some techniques to deal with graph with millions of items [47].

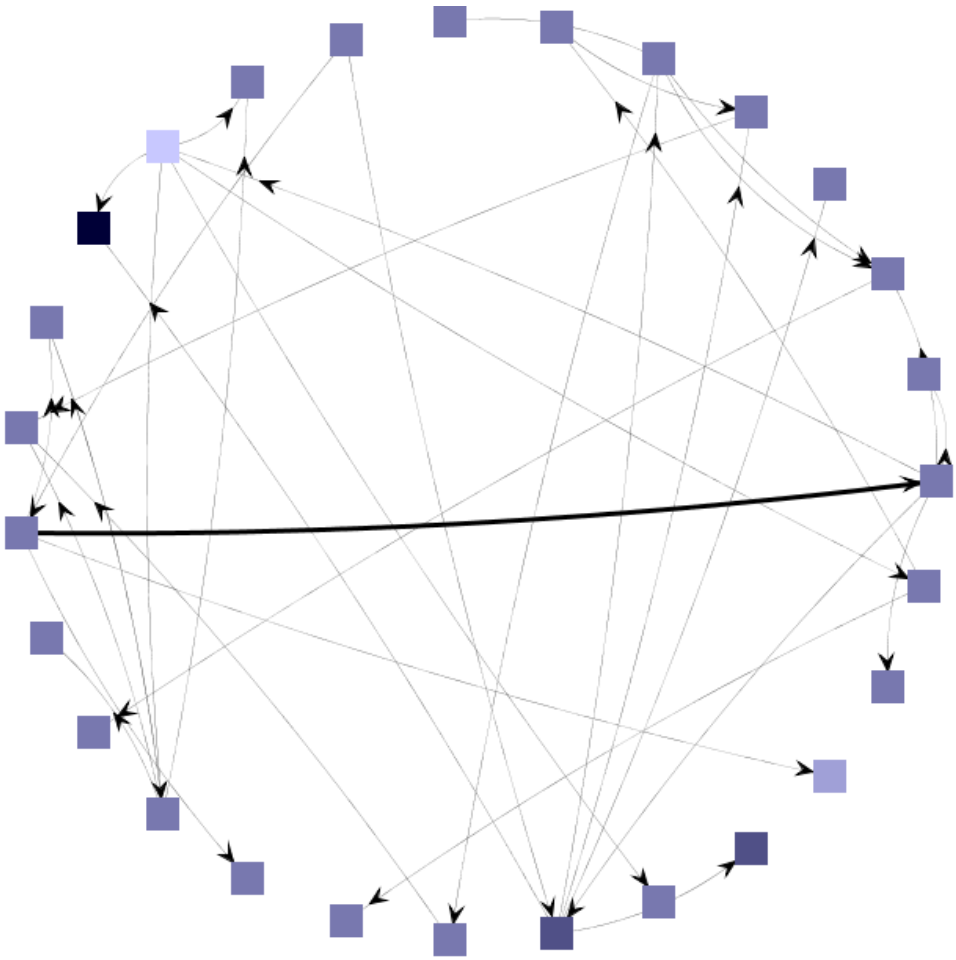

Figure 2.3 A Graph Drawing Based on Circular Layout

2.3 Network Visualization

Network is depicted as a graph consists of nodes, links and spatial information [25]. Compared with graph drawing in the Section 2.2, network obtains one more element, spatial information, namely, the attribute data held by the nodes and edges. Simple graph drawing cannot express the attribute information. Thus, network visualization is applied to complete this task. As it is discussed in Section 2.2, network visualization means a technique for presentation of the vast volume of data yield in such networks.

Network visualization is an established technique for the presentation of the vast volume of data yielded in such networks. Several tools can be found in the previous work that attempt to provide network visualization approach.

2.3.1 Map Attribute Data to the Aesthetic Element

One simple and intuitive way is to map the attribute of the node to the aesthetic element, such as *color, size, shape, pattern, transparency* and *dashing.*

Chris Wilson presents an application by applying this idea. News Dot (shown in Figure 2.4) "visualizes the most recent topics in the news as a giant social network. Subjects—represented by the circles below—are connected to one another if they appear together in at least two stories, and the size of the dot is proportional to the total number of times the subject is mentioned."[33]. As depicted in the Figure 2.3, the mapping tactic is simple but easily understandable.

Size: represents the total numbers of the subject is mentioned

Color: represents different subjects in different domain.

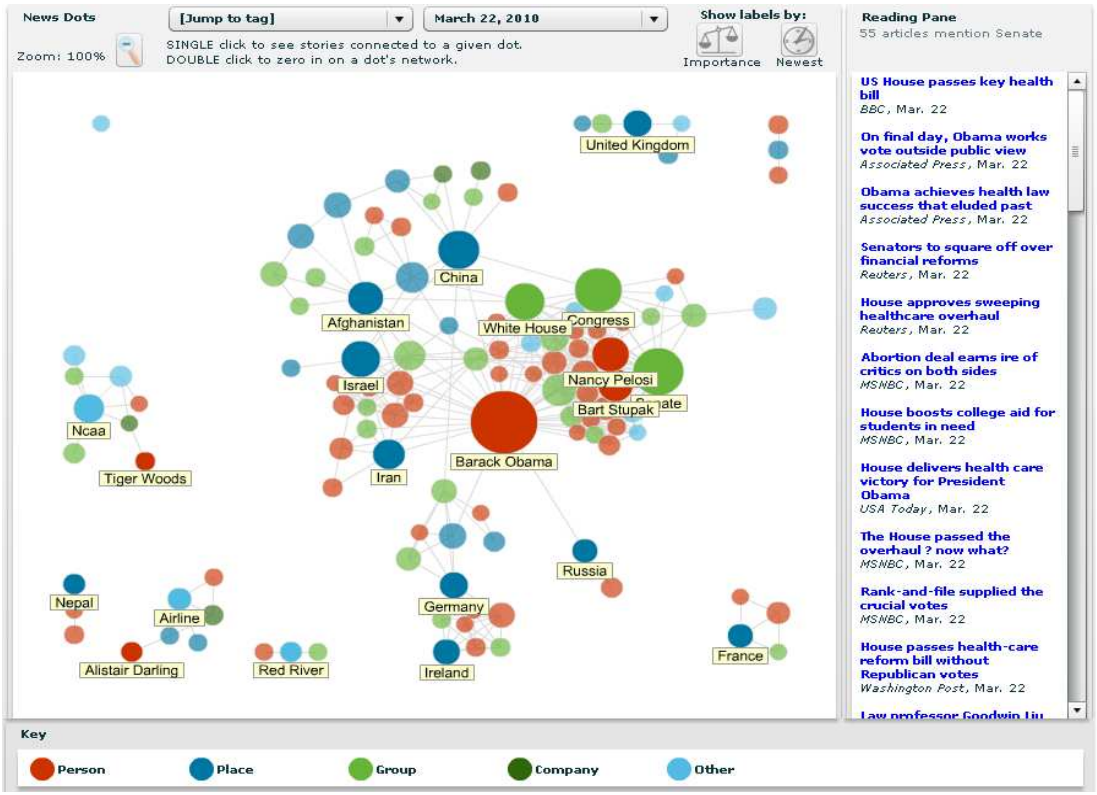

Figure 2.4 News Dot. Taken From [33]

We have to acknowledge that this simple aesthetic mapping method is easy to understand. However, the drawbacks are obvious; one of them is that the aesthetic of the node can just bear limited information. In plain terms, just a small group of attributes can be mapped to the aesthetic elements of the node since there are quite a few aesthetic elements can be used in an efficient way. In the Figure 2.4, aesthetic elements, just *color* and *size* are well used to the presentation，the developers still need recur to the Reading Pane to display more information.

2.3.2 Multiple Views

By separating the node-link drawing from the spatial information representation, Ross Shannon et al. [37], present a visualization technique that combines Parallel Coordinate with graph drawing in order to highly attributed network graph.

In this approach, the graph drawing is used to show the relationship between the nodes, while the attributes of the nodes are visualized by the parallel coordinate. An insight of this technique can be illustrated by Figure 2.5. As shown in the figure, a parallel coordinate view is located on the left side of a node-liking drawing graph. When a node is selected in the node-link graph, the related attributes of this node will be highlighted in the parallel coordinates.

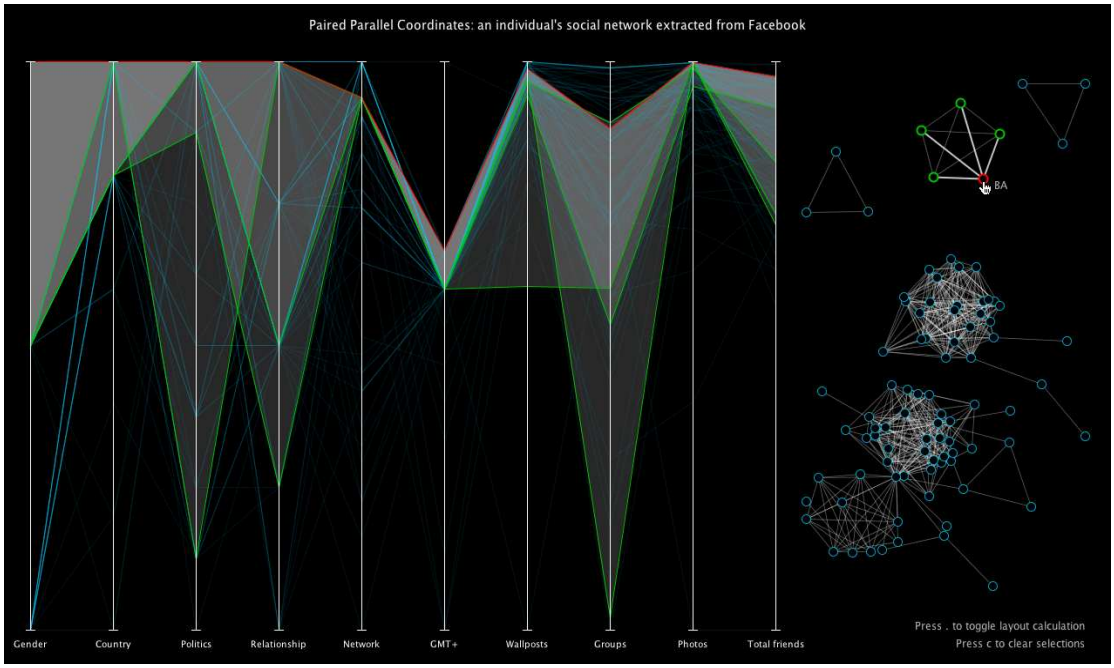

Figure 2.5 Combining Parallel Coordinates with Graph Drawing

In this visualization, a variety of visualization and interaction techniques are implemented in order to explore and analyze the attributes. Moreover, it brings a novel idea to combine the graph drawing and multivariate data visualization (Parallel Coordinate) into network visualization. So far we have found some features which are needed to be improved, such as, scalability problems may come forth due to the space limitation. Another disadvantage is that they break the network visualization into two separate views. [53].

2.3.3 Integrating Data Visualization into Graph Drawing

Compared with the method discussed in Section 2.3.2, the integration method shows the attributes of the nodes and graph in the same view by integrating the attribute information into the graph drawing process.

Borijuks et al. discussed some applications of this method in the work [38]. Figure 2.6 shows visualization of *experimental data in the context of a metabolic network*. Instead of circles or rectangles to represent the nodes, in the view, the nodes are represented with bar charts diagrams, which are used to visualize the attribute of the nodes. By integrating the data into the graph drawing, users have access to the attributes of the node while they observe the relationships of the node. The diagram representation brings users a more intuitive understanding of the attributes of the nodes since the abstract data is represented with an exact diagram. However, the size of the diagrams of the nodes would be very big if the graph is highly attributed, which would cause overlapping between the nodes.

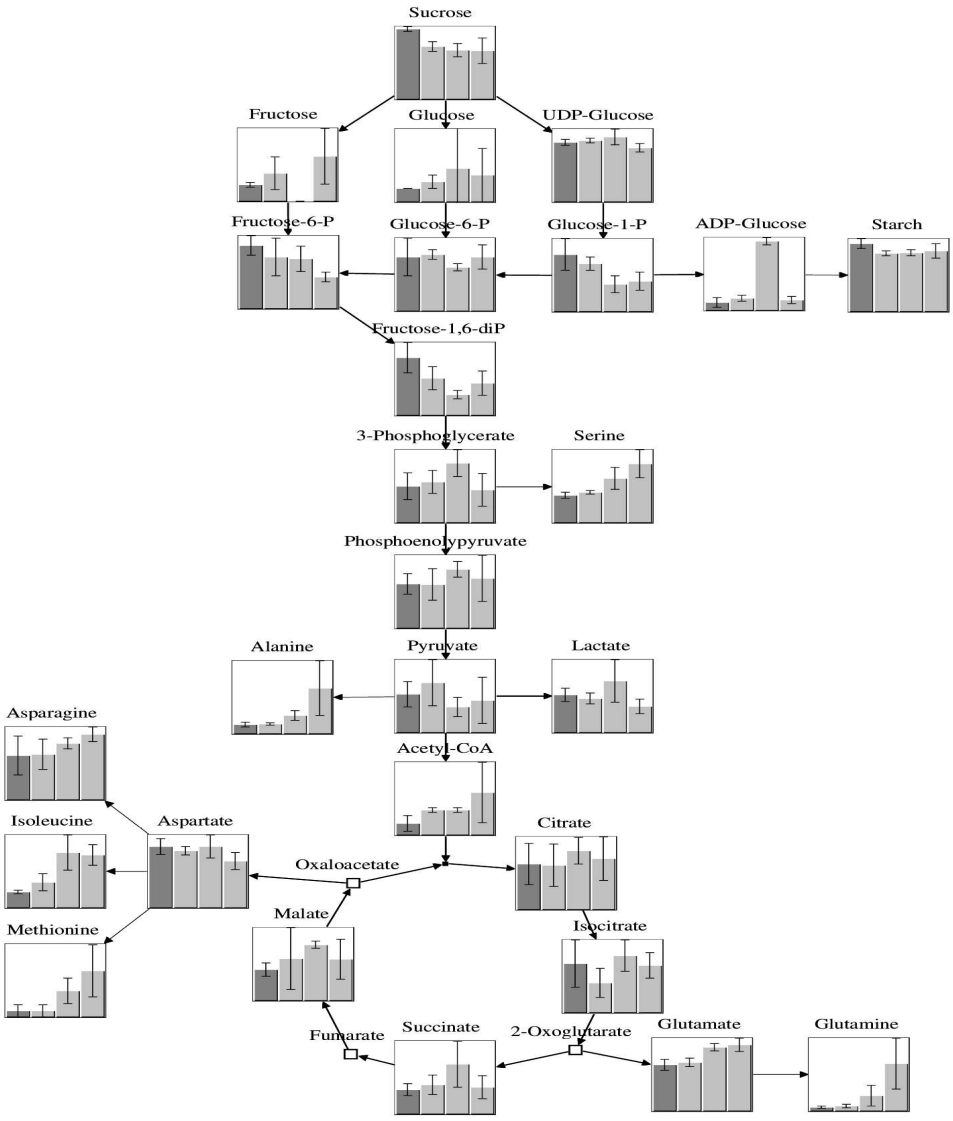

Figure 2.6 Network Visualization Applying an Integrating Method. Taken from [38].

2.3.4 Visualization with Text Notes

In theory, this method can be treated as a hybrid of the Combining Coordinate view with Graph Drawing method in Section 2.3.2 and Integrating Data Visualization into Graph Drawing method in Section 2.3.3. This approach separates the majority attributes from the graph drawing and describes them in the text notes in a separate view area. Vizster [25] is such a tool applying this method.

Vizster is an interactive visualization tool for online social networks, allowing exploration of the community structure of social networking services. As depicted in Figure 2.7, we can easily find that the spatial information (attributes of the node) is separated from the graph drawing. On the left side of the view, a node-ling graph with glyph description is drawn, while on the right hand of the node-link graph are located the profile description of the attributes of the node.

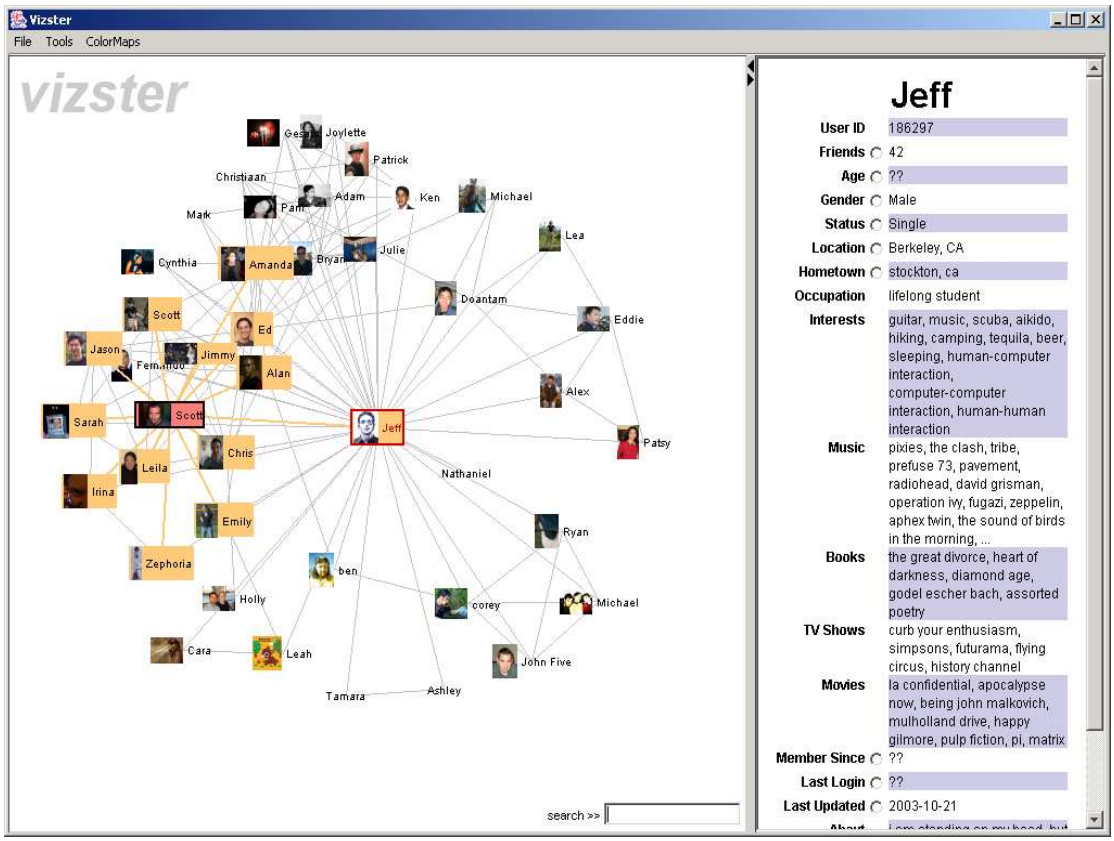

Figure 2.7 Node Highlight Effect of Vizster. Taken from [25]

The motivation to develop this tool is to visualize online social networks. This motivation limits its general applicability to visualize all kinds of network. We must admit that it is a powerful tool to explore the social network, but when it comes to some other types of network, such as metabolic network, as we showed on the Section 2.3.3, the advantage of this tool can not be manifested so well.

Possible solutions about network visualization are far more beyond what are listed above, but these approaches are very typical in the network visualization and they reflect the common methods applied in network visualization.

2.4 Magic Lenses

The notion of a magic lens was firstly proposed in the Eric Bier [9]. A magic lens is a screen region that semantically transforms the content underneath it. Users can move the lens to control what region is affected. In his work [9], Eric introduced what is a magic lens, the principle of operation and implementation of a magic lens. This entire context brings the basic concepts, common traits and construction of a magic lens. "These widgets may incorporate visual filters, called *Magic Lens™ filters*, which modify the presentation of application objects to reveal hidden information, to enhance data of interest, or to suppress distracting information. Together, these tools form a *see-through interface* that offers many advantages over traditional controls."[9]. The description indicates that a magic lens is a good tool to filter and reveal hidden information from some contents.

Erics' work [9] also introduced the basic components of a magic lens. As he listed, the four components are simultaneous use of two hands, movable tools, transparent tools and viewing filters. Each of these four components occupies an important position in the magic lens. In their later work [10], Eric mentioned these four components again, in order to specify the importance of these four components to the users. Some new features are also added in this work. The operations about a magic lens are firstly described in details. The following diagram in Figure 2.8 shows a concise and simple principle of how a magic lens works.

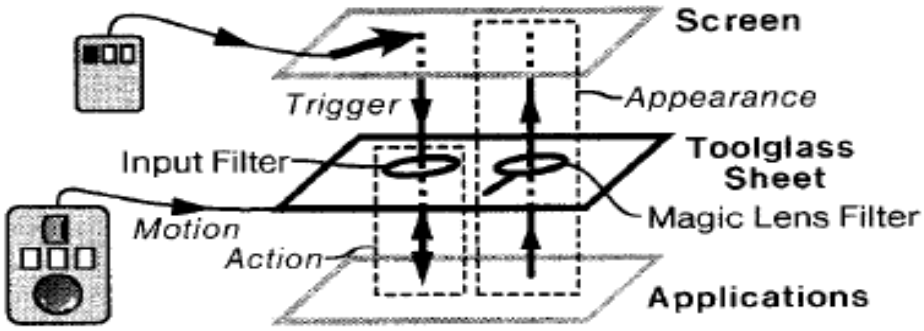

Figure 2.8 Information Flow of a See Through Tool. Taken from [10]

The diagram describes the information flow of a magic lens. It also indicates the operation steps of a magic lens, "receiving a triggering event from the user (Trigger), computing an action based on that event and having an application perform that action (Action), computing the appearance of the tool and the application as seen through it (Appearance), and moving over application objects (Motion), A fifth step, instantiation on a toolglass sheet is not shown." [10]. Thus, we are aware of that there are a series of operations when you try to implement a magic lens. And these operations should be taken into account when we design our own customized *Network Lens.*

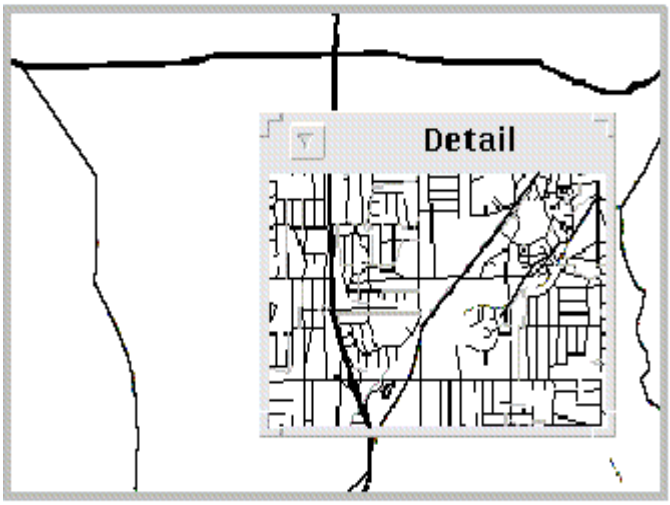

Figure 2.9 An Example of a Magic Lens Application

Almost at the same time, Maureen published a related paper. In this work [11], Maureen concentrated the focus in the magic lens filter. He described a lot of advantages of a magic lens filter and basic principles of operating the lens. In his description, we have a deeper insight of how to use magic lens filter as a tool to explore and modify hidden information.

In the works [9, 10, 11], many examples about magic lenses are demonstrated to show the interested readers some practical applications of a magic lens. Application like *A lens that provides a more detailed roadmap* [11], and *a simple example of the Magic Lens technology* [12] developed at Xerox PARC [13] should bring the users a little closer insight to magic lens.

2.5 Multivariate Data Visualization

Multivariate data visualization is the study of graphical representation of various types of data information. Multivariate data visualization has been long used in many areas. V. Friedman [39] indicates that the main intention of data visualization is to communicate and analyze data in a clear and efficient way. In order to make a clear description of multivariate data visualization, we will discuss it in the following aspects.

2.5.1 Data Types

In the domain of multivariate data visualization, data can be divided into three different categories: *Quantitative*, *Nominal* and *Ordinal*. ***Nominal*** data is a type of data in which there are limited categories but no order [22], such as the name of the nodes in a network diagram. ***Ordinal*** data has an explicit order that allows ranking between items, such as the ID number of the node in a network graph. ***Quantitative*** (or numerical) is a type of data that allows ranking between them and in which the distance between items can be computed.

For different data types, we need think about different visualization techniques to sufficiently to display multivariate data. For example, you can use bar charts to

represent the *age* of the man since *age* is a quantitative data and you can compute it and map it to a bar chart, while you can not apply bar chart to tell the name of a man since it is really pointless to map the name to a bar chart.

2.5.2 Multivariate Data Visualization Representation

Data visualization has been used for a long time. “The earliest table that has been preserved was created in the 2nd century in Egypt to organize astronomical information as tool for navigation. A table is primarily a textual representation of data, but it uses the visual attributes of alignment, white space, and at times rules (vertical or horizontal lines) to arrange data into columns and rows.”[23]. Data visualization can also be described as a process to transform abstract data into another format that can be easily obtained by humans’ eyes.

“Data visualization is graphics representation of information” [24]. Multivariate data visualization offers a good way to understand and process enormous amount of multivariate data quickly since visual presentation brings themes and pattern to the surface. Bar charts, scatter plots are examples of the multivariate data visualization representation that has been used for decades.

In the history of multivariate data visualization, there is a mass of techniques that one can use to visualize the multivariate data. The most commonly used techniques can be included into the follow categories: *chart, graph, plots, maps, images, 3-D surfaces and solids and Isosurface/slices [24]*. In the work [15], Keim and Kriegel sum up all of these techniques into 5 groups: *pixel-oriented, geometric projection, icon-based, hierarchical* and *graph based* techniques. For each group of these techniques, there are some well known representations that are frequently used in visualization field. For instance, in the scope of *geometric projection* technique, the *parallel coordinate* visualization technique and *scatter plot* are widely used visualization techniques. For each visualization techniques, there are some common elements to compose the techniques. We take a figure from Jose F. Rodriguez Mr.’s work [27] to show the components of multivariate data visualization.

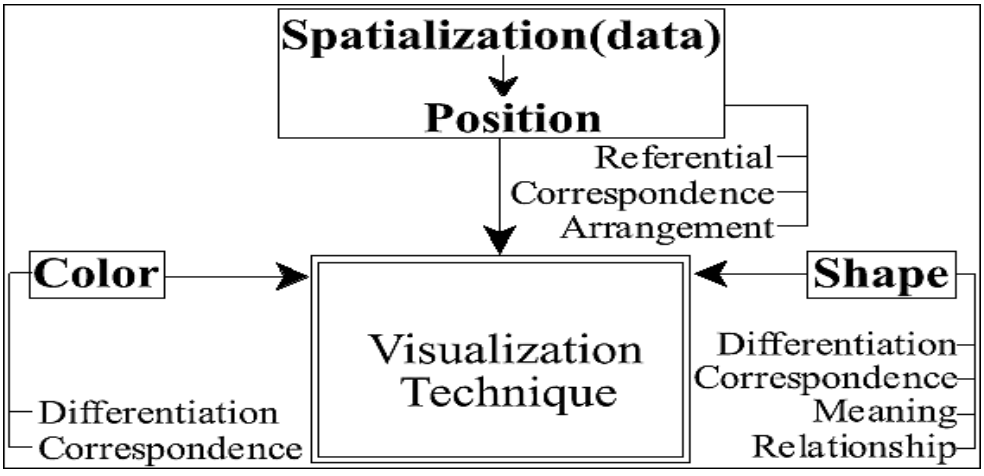

Figure 2.10 Components of Visualization Technique. Taken from [27]

In the figure, the "**Spatialization**" refers to "its transformation from a raw format that is difficult to interpret into a visible spatial format."[26]. Thus, indicated by the Figure 2.10 we conclude that any multivariate data visualization technique is composed by *Spatialization, Color* and *Shape* elements.

Figure 2.11 gives a group of frequently used multivariate data visualization techniques. From top to down, examples of Bar charts, Scatter plot, Parallel Coordinate and Star plot are arranged in the figure.

In order to present a more clear understanding about the three elements: *Spatialization*, *Color* and *Shape* about multivariate data visualization, bar chart is chosen as a sample to be analyzed.

Spatialization: axes(X coordinate and Y coordinate)
Color: discrete differentiation (Different color encoded)
Size: Corresponds Value (The length of the bar chart).

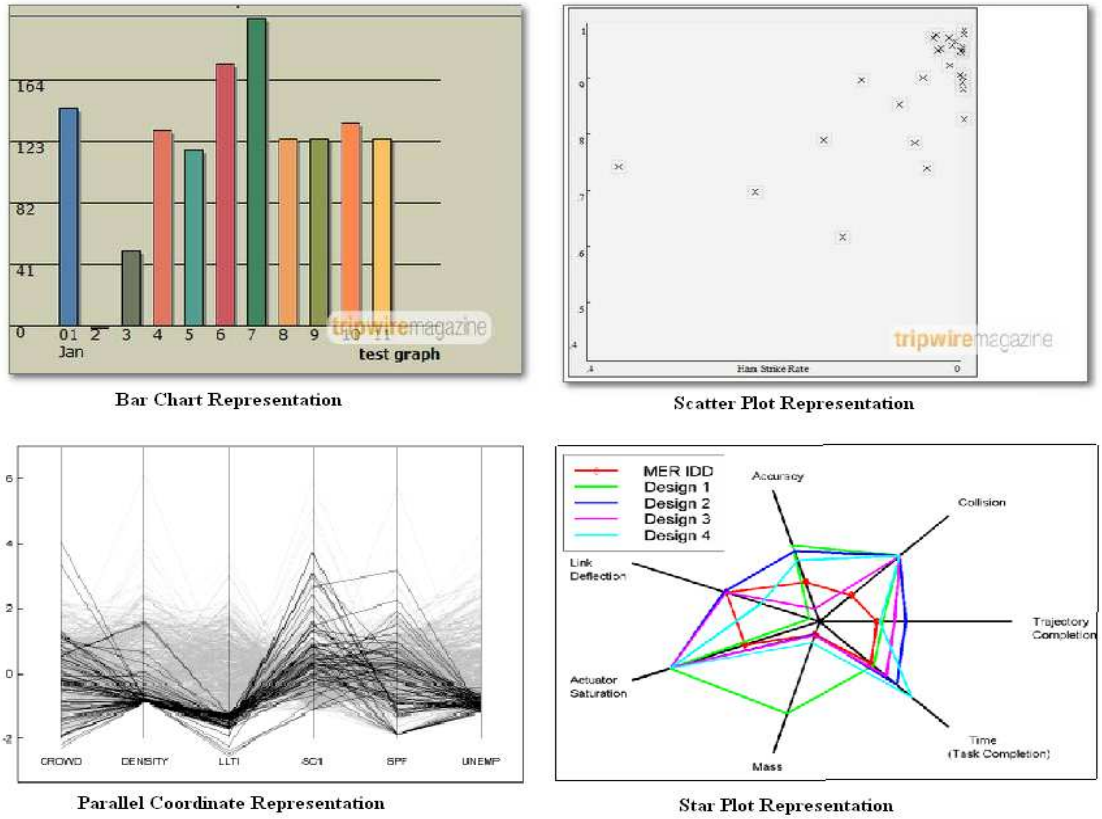

Figure 2.11 Data Visualization Technique Example
(From left to right, and top to down, these four examples are taken from [48], [48] and [49])

2.5.3 Multivariate Data Visualization Process

As it is described from the definition of the data visualization, "Data visualization is graphics representation of Information". Based on this description, we can find that, the input is some raw data and the output is supposed to be visual structure. The following diagram taken from Mehdi Dastani's work [26] can bring us a deep insight of the process of multivariate data visualization:

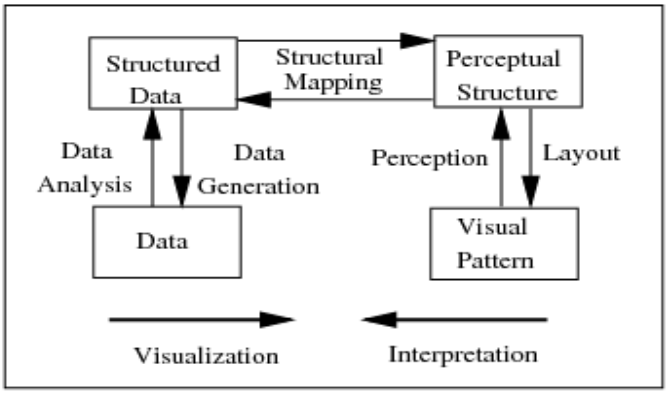

Figure 2.12 A Process Model for Effective Data Visualization. Taken from [26]

Getting inspired from the Figure 2.12, we can divide the process into three stages. The first step is to transform the raw data to the so called structured data, which can be represented in a graphics mean. For example, applying data scale [28] to normalize the raw data. The following step is to map the structured data to a perceptual structure. The perceptual structure extends a spatial medium with visual element and graphical properties in order to encode the data. In this stage, the final stage is to apply some algorithm to render the perceptual structure data into a graph.

3 The Network Lens

As stated in Section 1.2, the objective of this work is to create an application which is a combination of magic lens and data visualization to help user to explore and analyze the hidden information of a network diagram. In this chapter, we will discuss our approach to achieve the aim of this work. Our abstract but clear methods will be provided in this part.

3.1 Overview of the Network Lens

3.1.1 The Network Lens

The Network Lens is an interactive tool to explore and analyze the attribute information of the network by using visual filtering. The idea of this method can be described as following: A network graph is displayed as a node-link metaphor. Each node holds a bunch of attributes. User can explore some desired attribute of the node by setting focus on special node(s). The desire attributes of the nodes are transformed into some graphical representations, such as bar charts or parallel coordinate representation. Then, the traditional node representation of the nodes, such as rectangle or circle representation will be replaced by the new graphical representation that generated from the desired attributes.

Compared with the solutions that mentioned in Section 2.3, our Network Lens is likely to be a hybrid of the *Mutiple Views* and *Integration* Method, because it preserves the node-link graph drawing while integrating the attribute data of the nodes into the graph. The advantages of this method are very obvious: on one hand, there is no scalability problem compared with *Multiple Views* method, on the other hand, it can avoid overlapping problem compared with *Integration* Method.

3.1.2 Approach to Visualize the Network Lens

The main idea of this approach is to bring the magic lens concept mentioned in Section 2.4 and the multivariate data visualization techniques into the node-link network drawing. The development process to build the Network Lens follows the reference model demonstrated in Figure 2.1. At first, the attributes of the nodes and the edges are extracted from the original network format and are constructed in well-organized data tables. Then the users can create a lens by selecting their interested attributes and assign some visualization technique to represent the selected attributes. Additionally, user can implement some interaction function with the lens to explore and analyze the attributes of the nodes. Moreover, user can edit and save the lenses they have created and combine two lenses to form a new lens. The whole process of creating this Network Lens tool can be abstracted into the following process diagram:

As Figure 3.1 shows, the whole process can be divided into three steps. The first step is to draw the network in a node-link metaphor. The second step is to create Network Lens. This step can be broken into four different sub procedures: *lens initiation, data attribute selection, data visualization* and *integration of the data*

visualization and network drawing. The last step is to apply some interaction functions to the network lens to assist user to explore more information. At the end, the following prototype illustrated in Figure 3.2 will be achieved.

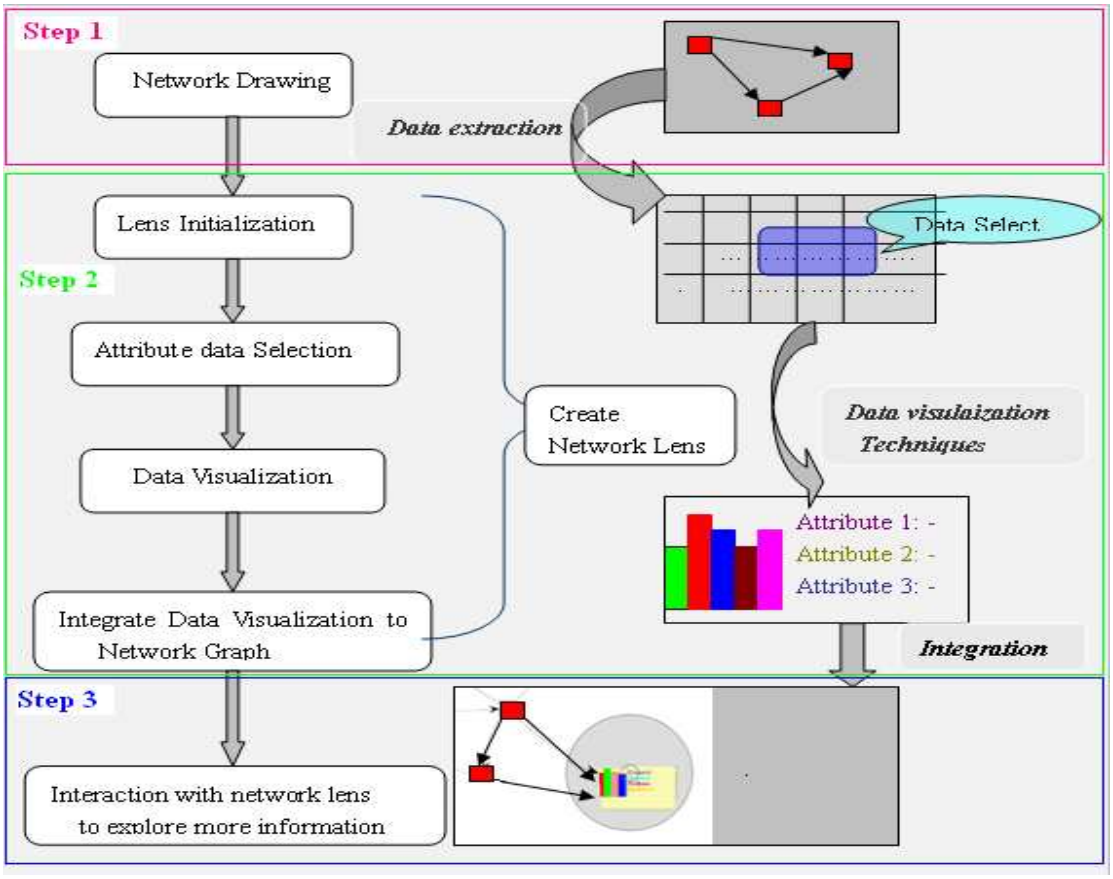

Figure 3.1 Network Lens Visualization Procedure

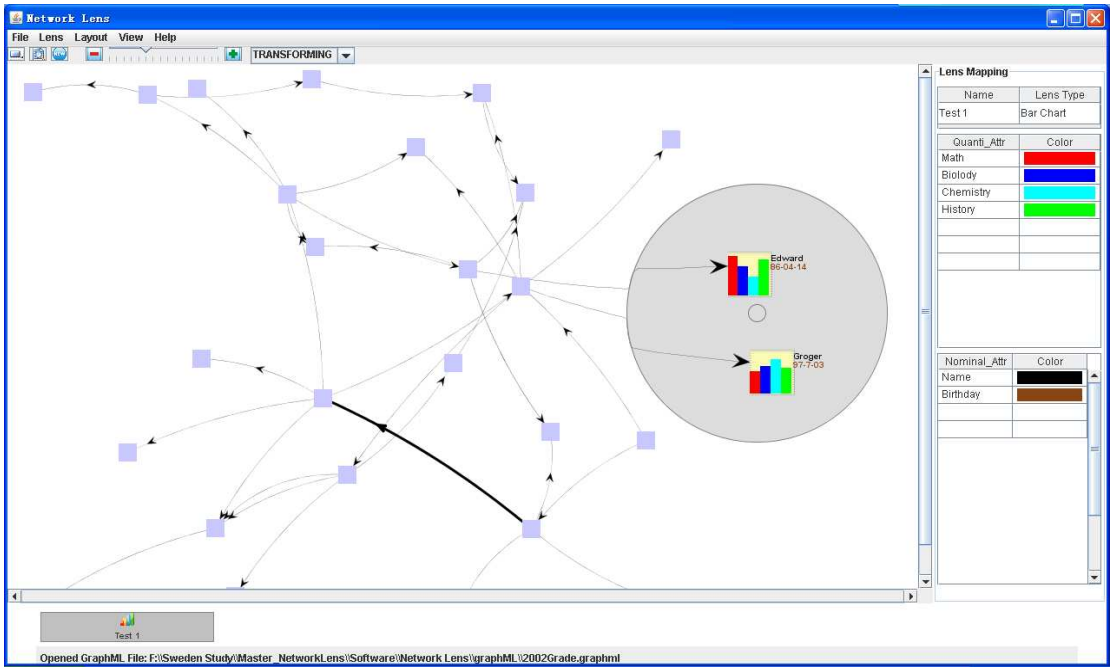

Figure 3.2 Overview of the Network Lens Tool

As Figure 3.2 shows, this tool consists of three parts, the main network visualization area where display a network graph and a lens. The right hand side of the main area is used to demonstrate the attributes that are selected and the colors encoded with the attributes. The bottom part is used to place and preserve all the lenses that have been created.

3.2 Network Graph Drawing

3.2.1 Network Graph Representation

To draw a network graph is our first step to set up this tool. As most networks are draw by applying node-link metaphor, in which node represents the objects in the network and edge represent the relationship between the objects, we apply node-link approach to draw our network graph, too.

The nodes of the network are rendered with a rectangle and the edges are represented with lines with the thickness of the lines to express the weights of the edges. Moreover, inspired by the data visualization, color can be used to carry some information from the network. Therefore, a value of a specific attribute can be mapped to the color saturation. For example, each node in the graph has a numerical attribute; we can apply our predefined color scale technique to create the color expression of the node. Our scale technique can be explained as: the range of the values of the attribute is mapped from light gray to dark, where the smaller the values is, the darker the color is for the node. Figure 3.3 shows an example of this color mapping. In the graph, that darker the node glyph is, the smaller the value of the attribute is.

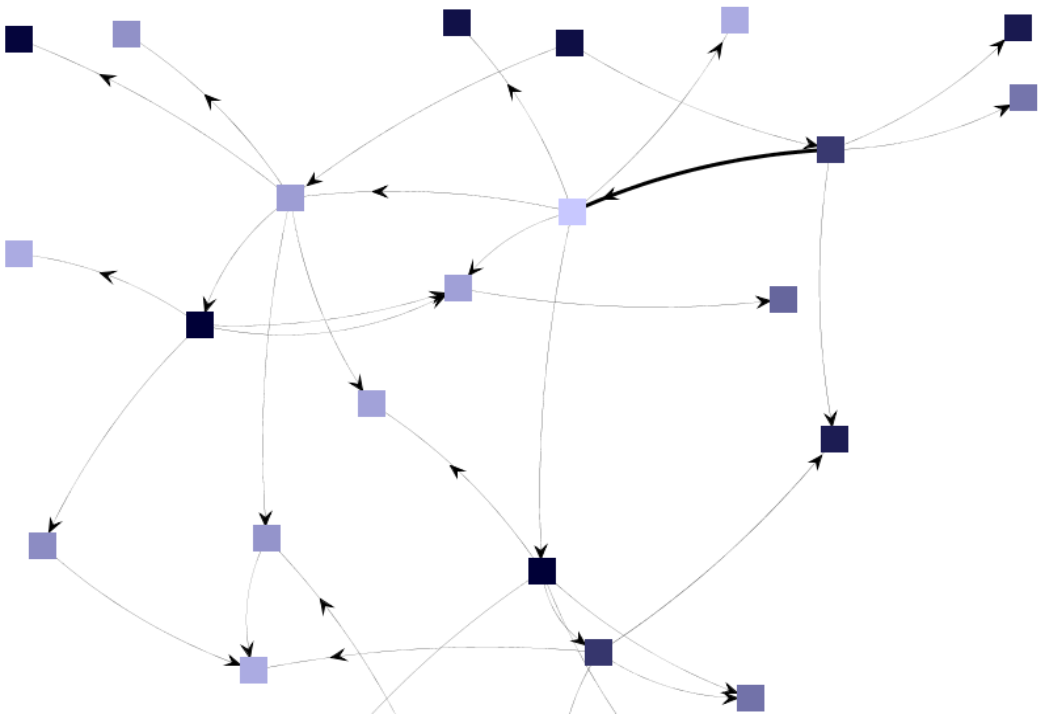

Figure 3.3 Graph Drawing with Color Coding

By applying this color encoding, this process plays a navigation role in the whole process. For example, if the user is just interested in the nodes with low values of a specific attribute, he/she would just focus on the node with dark color coding.

3.2.2 Network Graph Layout

As we know, a good graph layout can enhance the perception of a graph. Since there is no "best" layout for graphs, we are going to provide several options to users in order to satisfy individual requirements of the network representation. The following layouts are supported in our tools: FR Layout (or Fruchterman-Reingold algorithm layout), KK layout (Kamada-Kawai algorithm for node layout), Spring Layout and Circle layout. Following diagram shows four different drawing result of a new network diagram based on four different layouts.

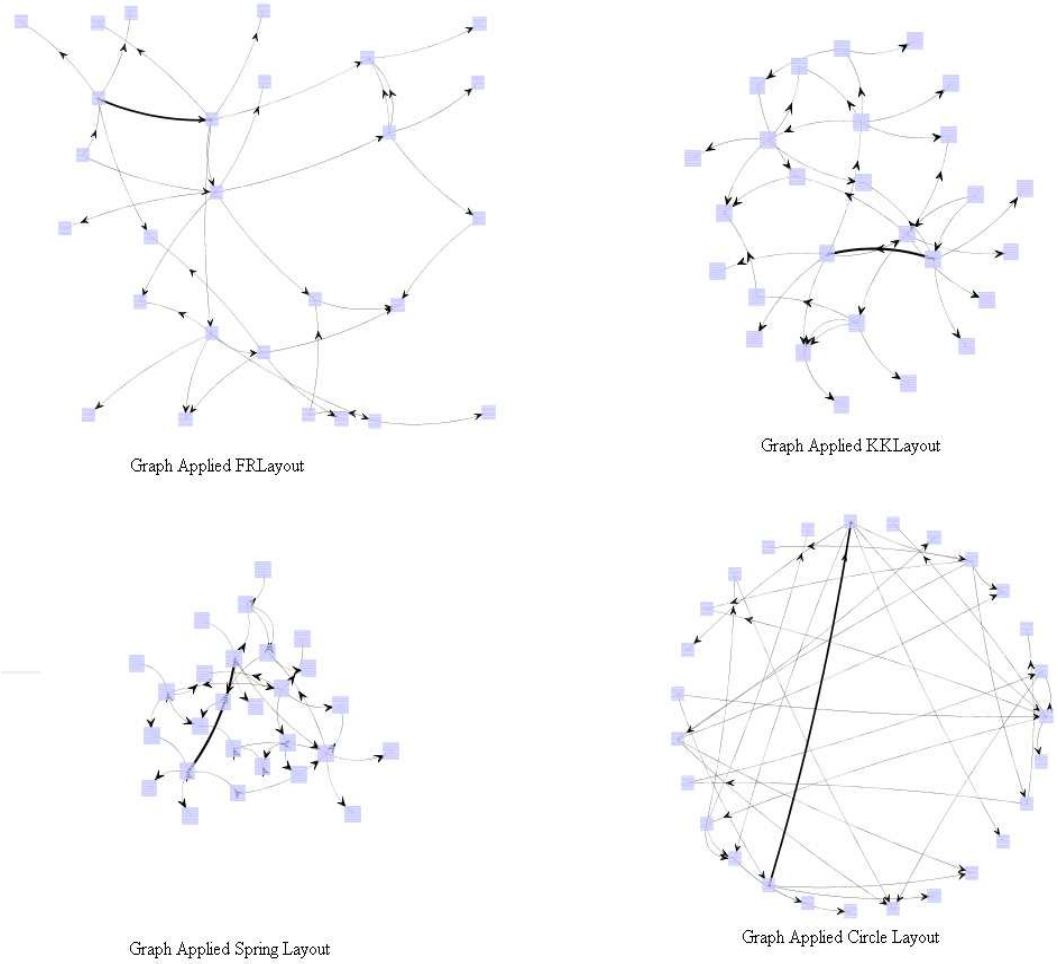

Figure 3.4 A Graph Drawing Based on Four Different Layouts

3.3 Network Lens Creation

After a network diagram is displayed, the process of creating a lens is the following step. As we discussed it in Figure 3.1, this process is composed by four sub processes: *Lens initialization, Attribute data selection and color mapping, Data visualization* and *Integration.* The process *lens initialization* is used to initialize the lens, for example,

define the name of the lens, and choose which kind of visualization technique should be used for the multivariate data visualization. The step, *attribute data selection*, is used to select the desired attribute and attach them to the lens. The following step, data visualization is the process to visualize the data that user has selected in the previous step. After the attribute data is transformed into a graphical representation, the final step is to integrate this glyph representation into the network graph. The main window of Lens Creation is illustrated in Figure 3.5:

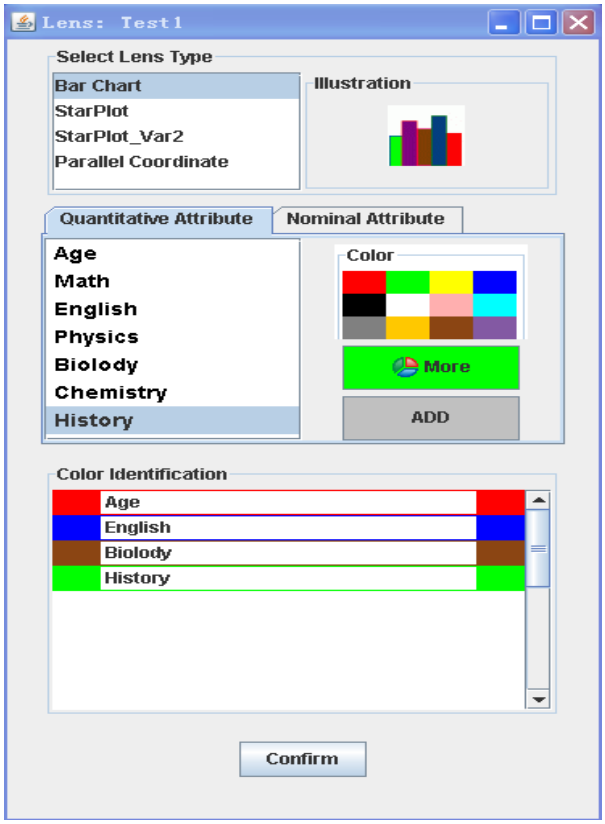

Figure 3.5 The Window to Create and Edit a Lens

3.3.1 Network Lens Initialization

By selecting the New Lens option from the Lens menu in the main window (shown in Figure 3.2), users start the process to create a new lens. The immediately following action is to assign a name to the lens and choose a data visualization technique to visualize the attribute data that you will choose in next step.

However, in some extent, we also take the process of extract and classify attribute data as the initializing action. When the users load a network and display it with node-link metaphor, simultaneously, the attribute data of the network diagram is extracted and classified into different categories based on their types. With regard to the data type in data visualization domain as we mentioned in Section 2.5, the extracted data is classified into two categories: *Quantitative* Attribute Set and *Nominal (incl. Ordinal)* attribute set.

3.3.2 Attribute Data Selection and Color Mapping.

The step *Attribute Data Selection and Color Mapping* is to choose the desired attributes that are attractive to our users. After the attribute data of the network diagram is extracted and classified into two categories. Two tabs Quantitative Attribute and Nominal Attribute *(incl. ordinal attribute)* are used to specify quantitative and nominal attributes data in Len Creation form shown in Figure 3.2.

Users can select the attributes from the list and add them to the lens. When the user selects an attribute from the data, he/she is supposed to map a color to the attribute, because the color will be used as the attribute identification in the data visualization step. In case of the user do not specify the color; the color mapping is done automatically. In order to render a good perception of the data visualization, we have provided 12 standard colors as suggested from C. Ware [31]. Assumed that users have selected a wrong attribute and he/she wants to revoke the operation, he/she can release the attribute form the lens by right mouse clicking on the attribute label listed on the Color Identification panel.

3.3.3 Attribute Data Visual Representation

After the desired attributed are selected and encoded with colors, we are going to visualize the chosen attribute data by applying some visualization techniques. The mechanism that we used to implement the data visualization can be expressed in Figure 3-6.

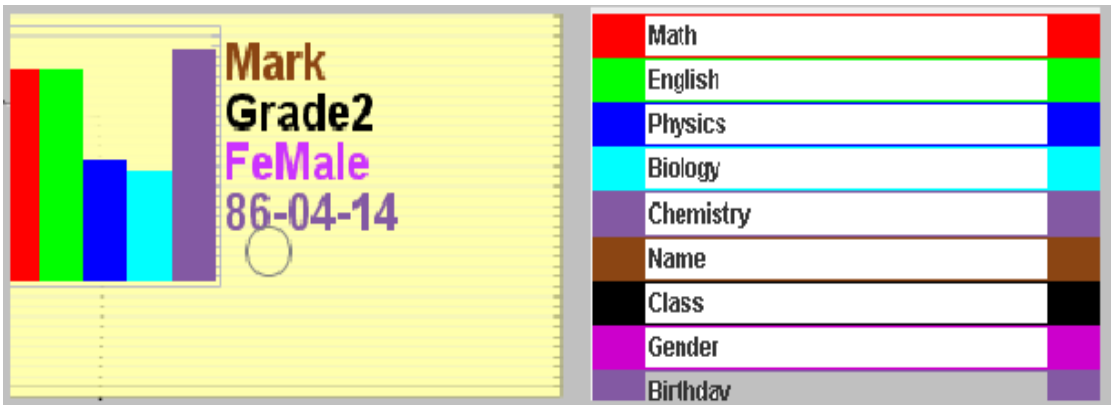

Figure 3.6 Strategy to Visualize the Data

As Figure 3.6 shows, all the attributes the users have selected, regardless of quantitative attributes or nominal (incl. ordinal type) type attributes, they are listed on a panel together with their color mapping, such as, attribute ***Math*** is mapped to ***red*** color. It means that, in the data visualization representation, the bar with red color represents ***Math***. In a further step, the length of the red bar represents the value of the ***Math***. In order to make the data visualization representation much clearer, we employ the following diagram to specify how it works.

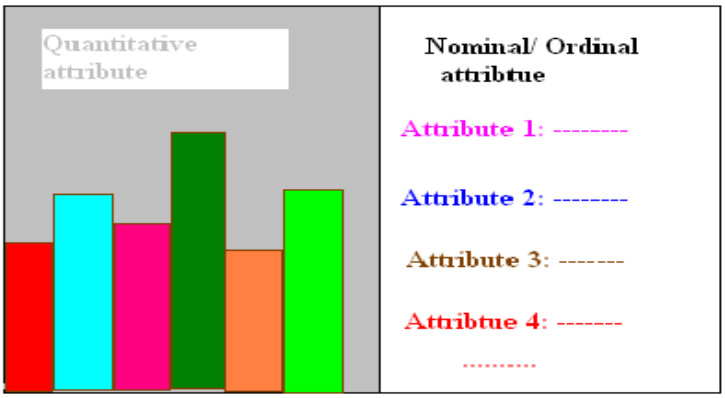

Figure 3.7 Data Visualization Format

The data visualization glyph is divided into two parts as indicated in Figure 3.7. The left parts is used to visualized *quantitative attributes*, there are a large amount of techniques we can apply to visualize the quantitative attributes. In this paper, we provide these four techniques for our users, *Bar Charts, Star plots (two variants)* and *Parallel Coordinates*. The right part of the glyph is allocated to visualize the nominal attributes. In Richard Resnick's work [32], a large amount of advanced visualization techniques have been introduced. Nevertheless, considering the situation that we have, such as space limitation, we just use the simplest way, text representation, to display the nominal attributes.

In this step, nothing particular are mentioned except the data normalization. As explained in the Figure 2.12, we have learned that there is a step called *Data Analysis*, which is used to transform the raw data to a structured data. Actually, this step is to normalize the raw data to some organized data format that would be easily represented in a graphic way. In this paper, the normalization process does not give a perfect result for that data visualization. There is a bunch of reasons. One important reason is that, the range of different attribute varies considerably large. For example, the range of the *Age* attribute of a person is from 1 to 100, whereas, the attribute *Salary* is from 10000 to 50000. For both of these two attributes, we apply the same algorithm to execute the normalization process. This will lead us a not so perceptual result of the visualization. For example, after normalization, the range of *Age* will be the same range with of *Salary*.

3.3.4 Multivariate Data Visualization Technique

Regarding that the users may want to apply several different multivariate data visualization techniques to visualize the selected attributes, the visualization techniques, *Bar Chart, Star Plots* and *Parallel Coordinates* are developed to offer different options to the users. The following diagram shows different views by applying different visualization techniques.

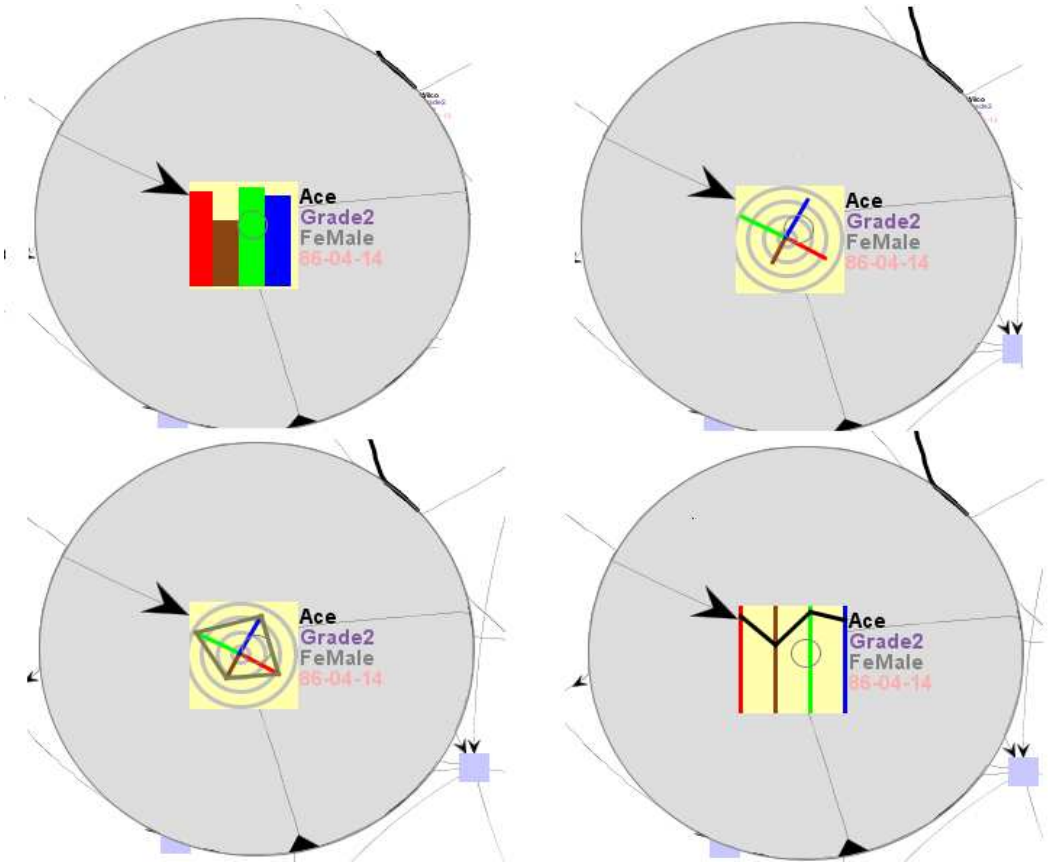

Figure 3.8 Four Different Multivariate Data Visualization Techniques

3.3.5 Integrating the Multivariate Data Visualization into the Graph Diagram

To integrate the multivariate data visualization to the network diagram is considered as the most essential part in this tool. A suitable integration will polish the final visualization result. The strategy that we use in this work is very straightforward. It can be depicted as follow: After the network diagram is displayed successfully. Each node can be attached with several icons. Each node holds two icons, default icon and attribute visualized icon. The default icon is used to render the default representation of the node. The attribute visualized icon is the glyph we presented in Figure 3.8. When users move the lens, if the nodes are beneath the lens, the node will be attached with the attribute visualized icon.

This strategy helps to avoid clutter and overlapping because the data visualization part is located inside the node, the size of the icon will not disturb the view of the graph drawing.

3.4 Interaction Techniques

Interaction between the user and application plays a crucial part in information visualization systems. The flexibility of the application will finally results in the usability of the tool. In Robert and Helwing's work [16], some views on interaction in information visualization are given. They mentioned that interaction is particularly important in InfoVis for exploration, analysis, and presentation of data. Good

interaction designs will give our users more free region to manipulate this tool. The Network Lens is equipped with some powerful interaction techniques, such as Panning, Zooming, Picking and Focus + Context [45].

3.4.1 Navigation

For interest of the user, the interaction design Panning, Zooming and Scalability are supported to navigate the user to explore the useful information.

Scalability: The scroll bar in the horizontal direction and the scroll bar in the vertical direction make the scalability possible. User can drag the scroll bars to obtain the specific display area of the network diagram.

Panning: The interaction panning helps user to get focus on some particular area. In our Network Lens, the panning area is always an ellipse. Panning is performed by dragging the lens. User can drag the lens on the display and locate it on a specific area in order to get focus on some nodes. Moreover, the size of panning area can be changed by adjusting the lens size. Additionally, the whole network layout can be panned if necessary.

Zooming: Zooming and panning always work together in information visualization, in order to give a detail + overall view effects. There are two kinds of zooming actions in our Network Lens tool. One is zooming for the overall overview. The other one is especially for the lens affected area. The previous one can be carried out by scrolling the wheel of the mouse to make zoom in and zoom out of the node-link graph. For the latter, the zooming performance can be finished in two ways: one way is to change the magnification value via slider bar to get the appropriate Focus + Context effect. Another way is to press the ctrl key and scroll the wheel of the button to adjust the zooming effect. Figure 3.9 shows the zooming effect of the lens affected area.

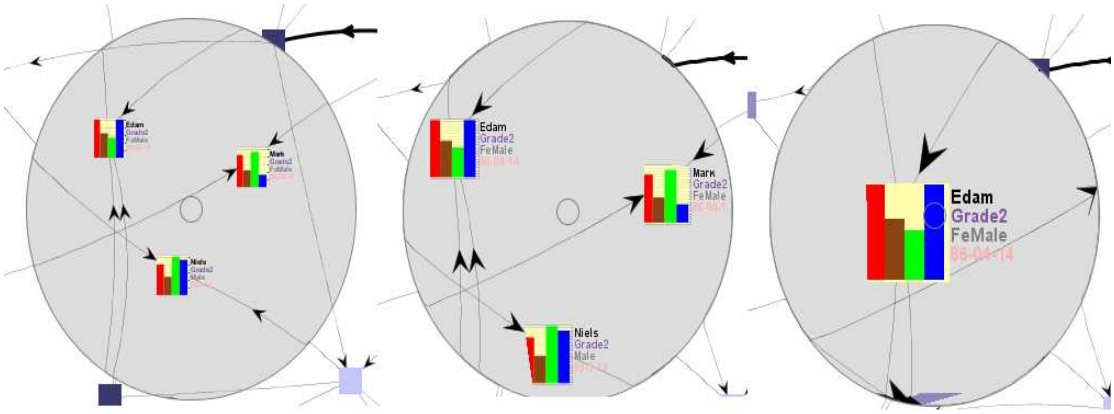

Figure 3.9 Zooming Effect of the Network Lens

3.4.2 Exploration: Picking and Transforming

The Picking and Transforming interactions help user to compare several nodes at one time. The following diagram will enhance one's understanding of the function **Picking**.

Supposed that, one wants to compare several different nodes in the network, such as ***node 1***, ***node 2*** and ***node 3***, shown in the Figure 3.10, the locations of nodes set an obstacle for our user to finish the action at once. One possible solution is to move the lens to the location of the nodes one by one and memory the values of the attributes of each node. But this requires a large amount of effort and contradicts the aim of our approach, which is “to allow users to see, explore, and understand large amounts of information at once” [40].

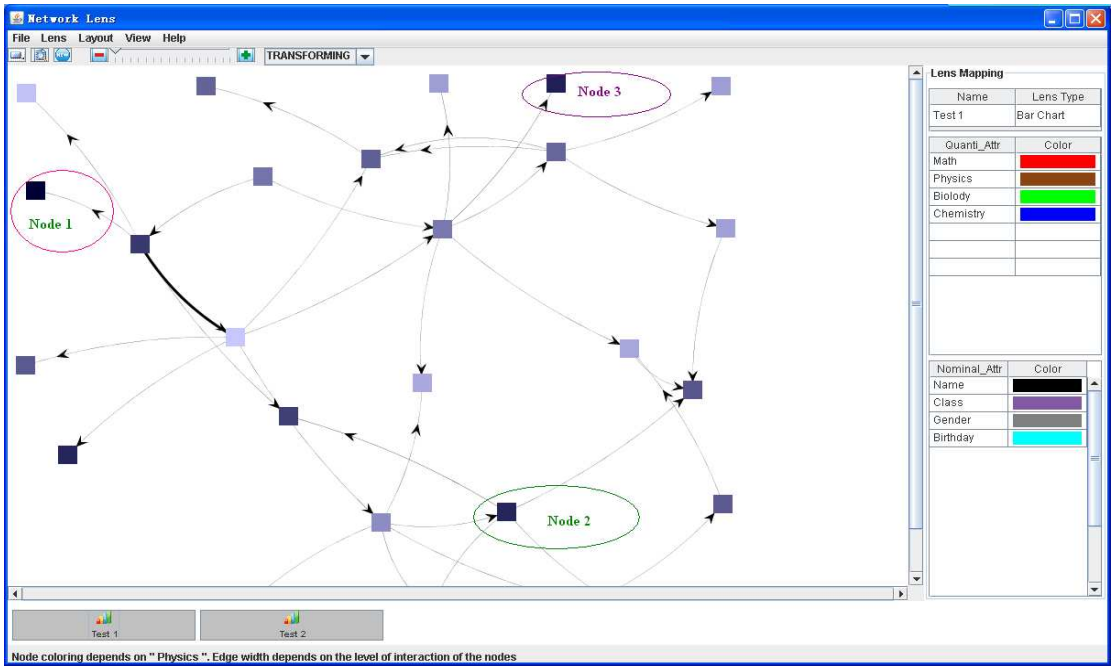

Figure 3.10 Comparing Several Nodes in the Network

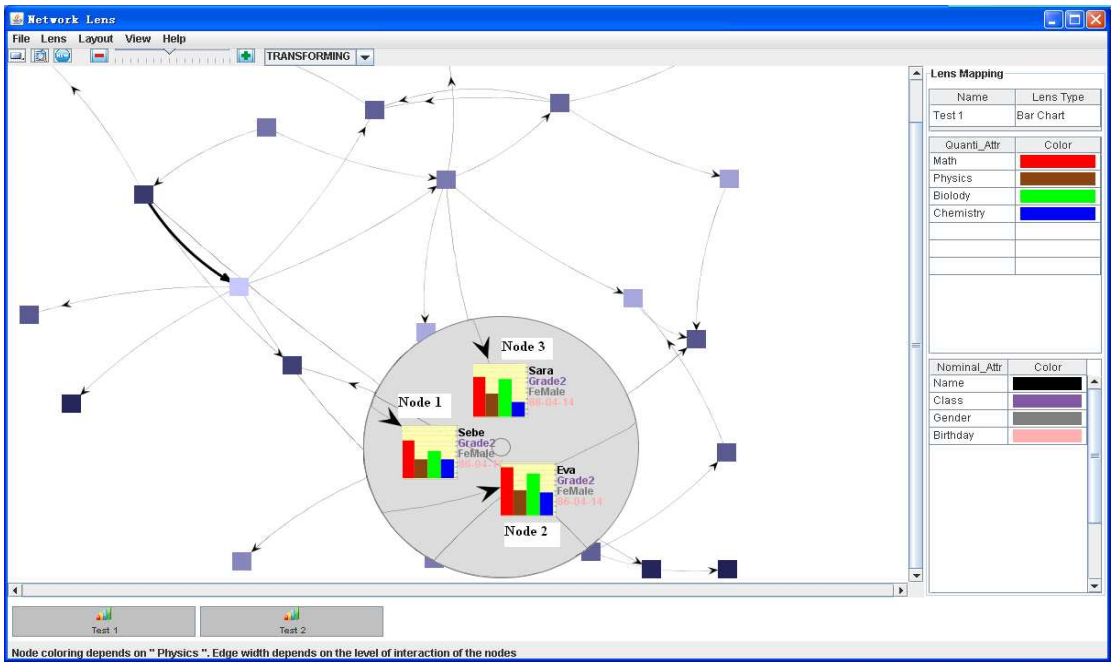

Figure 3.11 Picking and Comparing Different Nodes

Our approach to deal with this problem is simple, by setting the mode in the Combo

Box in the toolbar to PICKING; user can place the three nodes in an area and then move the lens above the area. Then, the attribute of the three nodes can be displayed at the same time. By applying picking function, the Figure 3.10 can be transformed into Figure 3.11.

3.4.3 Save and Load Lenses

In the Section 3.2, we have already introduced the process of creating a Network Lens instance. Our tool also supports users to load a stored lens or lenses (a ground of lenses) that he/she created before. By loading a preserved lens or lenses, it would save the time for the users. By clicking the Load Lens menu item in the Lens menu user can select a file with the suffix name ".lens" or ".glens" to load a lens or a group of lenses.

Additionally, the Network Lens even allows the user to save a lens or lenses. The user can click on the lens buttons, located on the bottom of the window, to save the lens he/she created. Or by clicking the Sava All Lenses menu item of the Lens menu in the toolbar to save all the created lens (lenses), the following diagrams will illustrate how it works.

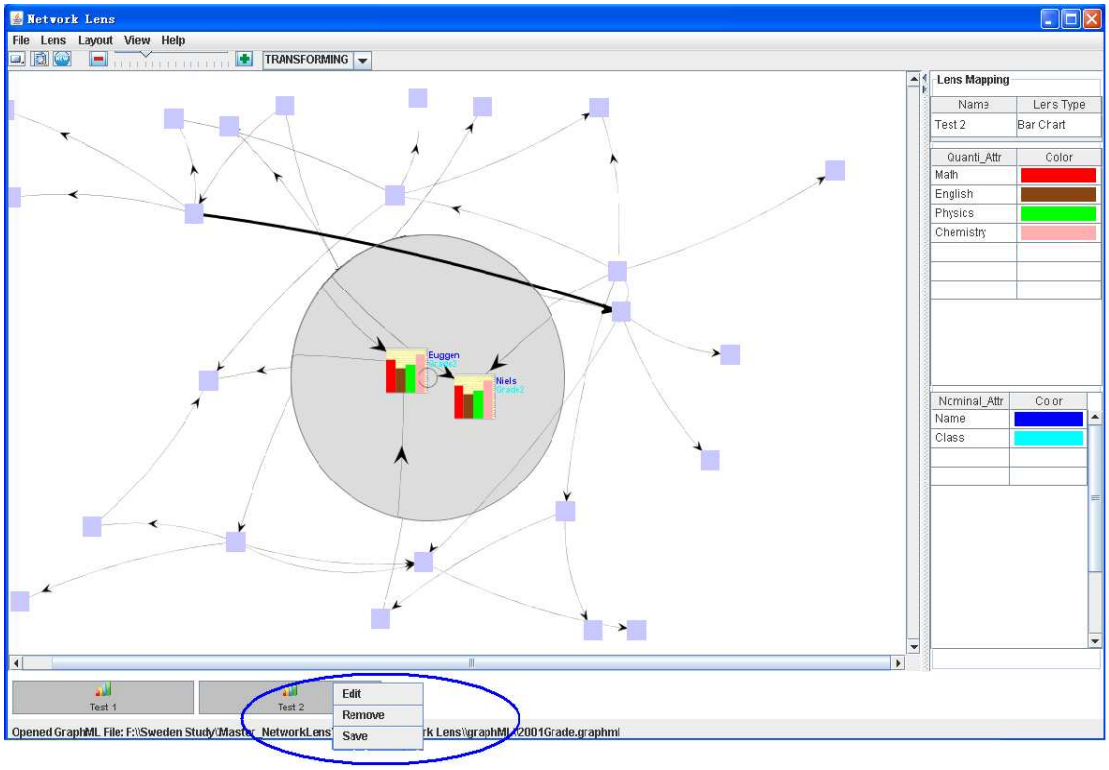

Figure 3.12 Interaction for Save and Edit Lenses

As showed in the blue circle depicted in the Figure 3.12, a pop up menu is displayed when the user right clicks on the lens button. In the pop up menu, there are Edit, Remove and Save menu items. By clicking on the Save menu item, user can save the lens. If one clicks on the Remove menu item, this lens will be removed. While if a user clicks on the Edit menu item, a window as shown in Figure 3.5 will

displayed on the screen. Then, user can edit the lens he/she created, such as, add more attributes to the lens, remove some attributes form the lens or change the encoding color identifications of the selected attributes.

3.4.4 Combining Lenses

Our tool provides a combination of already created lenses by “lying them one over another”. Our approach to combine two different lenses is very straightforward: users can drag and drop one lens button over another one. Then, a new window frame Combing Lenses form shows up as illustrate in the Figure 3.13

This combining Lenses form is comprised by four distinctive panels. Lens1 and Lens 2 are used to specify the information about the already produced lenses that are going to be combined. The Operation Panel is used to list the operation of the Lens 1 and Lens 2. As listed in the panel, our current operations are just **Union** and **Intersection**. If the user chooses the **Union** operation, it means that the attributes that are going to belong to the combined lens would be a union of the attributes of *Lens 1* and *Lens 2*. If **Intersection** operation is selected, the attached attributes for the combined lens will be the intersection of the attributes of the two lenses.

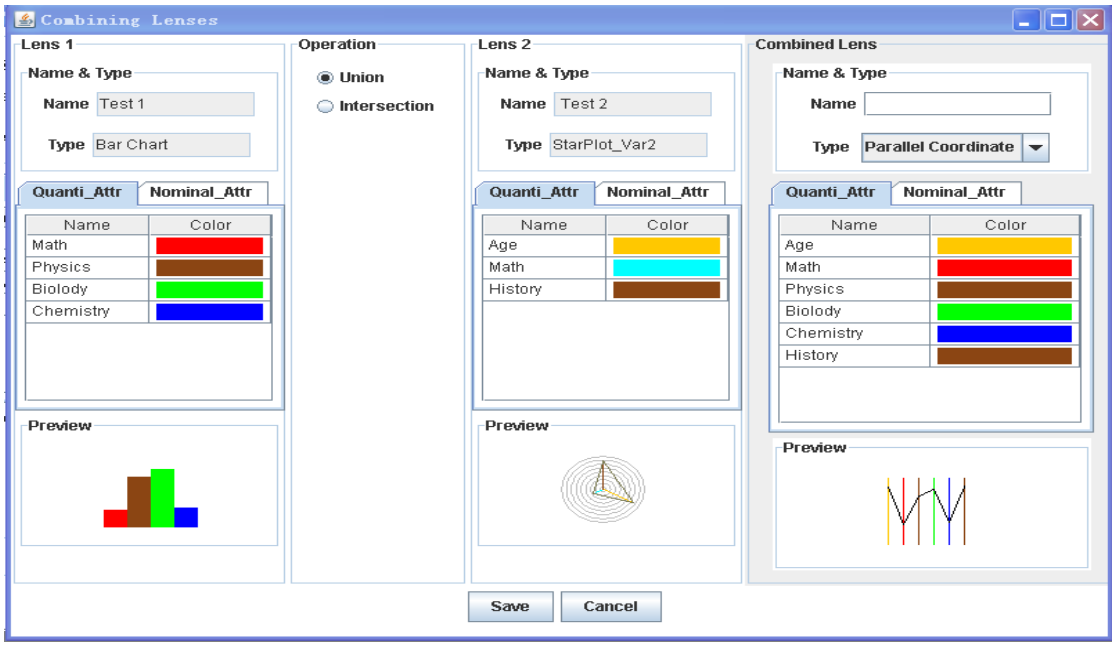

Figure 3.13 Effects of Combining Two Lenses by Union Operation.

By selecting different operations, different results would be shown in the Combined Lens panel. The layout of Combined Lens panel is the same with the Lens 1 and Lens 2. Here, user can define a name for the combined lens and select the visualization technique. Users can also change the color coding for the attribute in case two different attribute will have the same color coding. By clicking on the Color column in the table, user can choose various colors from a color panel to encode the corresponding attribute again.

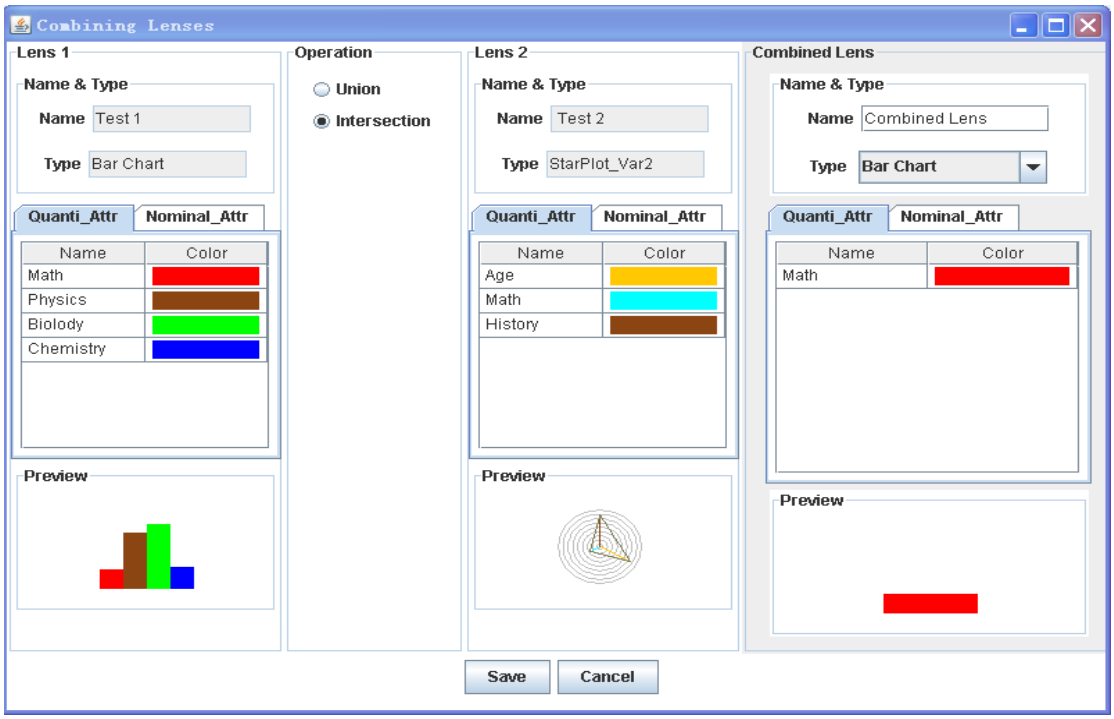

Figure 3.14 Effects of Combining Two Lenses by Intersection Operation.

4 Implementation

In this chapter, we explain the implementation process and issues of our prototype. However, we are not keen to explain the programming codes line by line. Instead of that, we demonstrate some crucial structures and method in the implementation parts. Following the order that we have presented in the Chapter 3, we will show the approach to implement it one by one. The implementation process is divided into the following five parts: network diagram drawing and extract the data from the GraphML file (network format), construct the lens, multivariate data visualization and interactive functions design, such as navigation and exploration.

4.1 The JUNG Library

We have already mentioned JUNG in Section 2.2.2. Due to the use of this library in our project, a more detailed description about how JUNG support the implementation will be illustrated in this part. The benefit that we get from JUNG can be summed up into the following: creating network diagram by loading a file, setting up the basic construction of magic lens and affording interactive functions, such as picking.

4.1.1 A Brief Introduction of JUNG

"JUNG - the Java Universal Network/Graph Framework - is a software library that provides a common and extendible language for the modeling, analysis, and visualization of data that can be represented as a graph or network." [6]. JUNG provides a common frame to analyze and visualize graph and network. It also gives user ability to construct a tool to explore the data of a network. "It provides a mechanism for annotating graphs, entities, and relations with metadata. This facilitates the creation of analytic tools for complex data sets that can examine the relations between entities as well as the metadata attached to each entity and relation." [6]. JUNG is such a powerful library to analyze network that our Network Lens tool borrow several methods from it.

4.1.2 Applying JUNG to Graph Drawing

JUNG provided three means to draw a graph:

- Load from a file, such as Pajek, GraphML
- Build from scratch, (create vertices individually, specify edges)
- Use a graph generator.

Considering the motivation of our application, we adopt the first approach to complete the network graph drawing task for three reasons: one is GraphML, it is easy to learn, the second one is that the GraphML is very flexible to change and the last reason is because the attributes of the network can be easily extracted from the GraphML file.

Our Network Lens allows users to load a GraphML file to draw the network diagram. As presented in the following text description, the structure of our GraphML file can be specified into three parts from top to down: Head Declaration, Declaration of GraphML Attributes and Declaration of Nodes and Edges with assigned attributes

Values. The GraphML file is the source data in the entire visualization process.

```
<?xml version="1.0" encoding="UTF-8"?>
<graphml xmlns="http://graphml.graphdrawing.org/xmlns"
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
     xsi:schemaLocation="http://graphml.graphdrawing.org/xmlns
       http://graphml.graphdrawing.org/xmlns/1.0/graphml.xsd">
  <key id="d0" for="node" attr.name="color" attr.type="string">
    <default>yellow</default>
  </key>
  <key id="d1" for="edge" attr.name="weight" attr.type="double"/>
  <graph id="G" edgedefault="undirected">
    <node id="n0">
      <data key="d0">green</data>
    </node>
    <node id="n1"/>
    <node id="n2">
      <data key="d0">blue</data>
    </node>
    <node id="n3">
      <data key="d0">red</data>
    </node>
    <node id="n4"/>
    <node id="n5">
      <data key="d0">turquoise</data>
    </node>
    <edge id="e0" source="n0" target="n2">
      <data key="d1">1.0</data>
    </edge>
    <edge id="e1" source="n0" target="n1">
      <data key="d1">1.0</data>
    </edge>
    <edge id="e2" source="n1" target="n3">
      <data key="d1">2.0</data>
    </edge>
    <edge id="e3" source="n3" target="n2"/>
    <edge id="e4" source="n2" target="n4"/>
    <edge id="e5" source="n3" target="n5"/>
    <edge id="e6" source="n5" target="n4">
      <data key="d1">1.1</data>
    </edge>
  </graph>
</graphml>
```

Figure 4.1 Example of a GraphML Document with Attributes. Taken from [52]

After a GraphML file is loaded, the JUNG library is ready to display the network graph depicted in the GraphML file. At the same time when JUNG library is loading and displaying the network graph, we are extracting the attributes information of the GraphML file and assign it to the corresponding nodes and edges.

4.1.3 Set up the Construction of the Network Lens.

JUNG provides the users the capability to filter the graph by constructing several different kinds of lenses. Here, two different kinds of lenses are provided, one is named as *Hyperbolic View Lens,* which gives a fish eye perspective; the other one is

named as *Magnified View Lens* which provides a magnified effect. The following figure is an example provided by JUNG library. In this example, the lens is a *Magnified View Lens.*

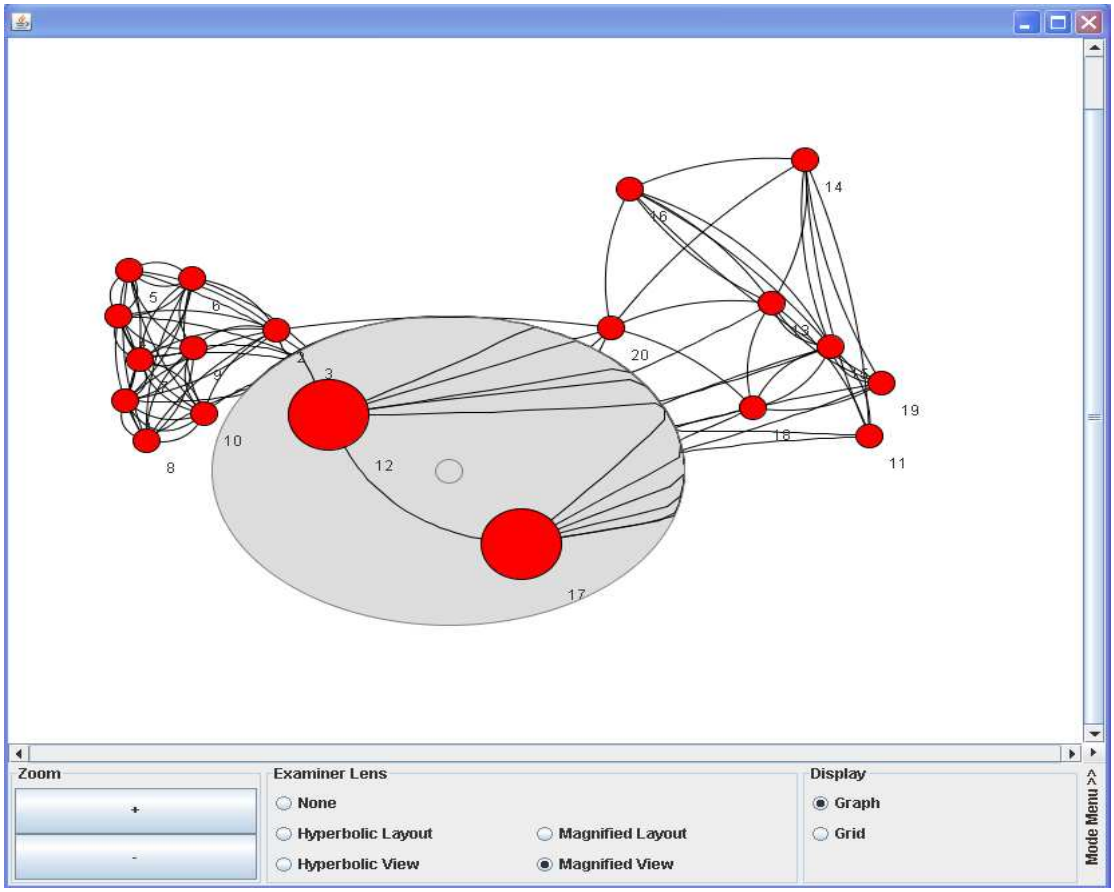

Figure 4.2 A Lens Developed by JUNG

In order to avoid re-inventing the wheel, the basic construction of Network Lens is adopted from *Magnified View Lens* developed by JUNG, so we will not drill down about how the *Magnified View Lens* is implemented. Detailed information of constructing the lens could be found in JUNG library.

After setting up the basic construction of the lens, we come to the next stage, which is to filter the attribute data of the network graph by applying the lens. We have already discussed the mechanism to finish this problem in Section 3.3.5. Now, the remaining tasks for this step are:

1. Get the location of the lens
2. Traverse the nodes and check whether the node is beneath the lens affected area.
3. If the node(s) is overlap by the lens, replace the node representation with the graphical attributes representation node.

The following code is used to implement this idea.

```
 // to traversal all the nodes of the graph and check whether it is affected by the lens
for(final Number Ver:graphLayout.getGraph().getVertices())
{
    location_X=magnifiedViewLens.getLensTransformer().getEllipse().getFrame().getX();
    location_Y=magnifiedViewLens.getLensTransformer().getEllipse().getFrame().getY();
    // Set the nodes with transformed Icon/ graphical attribute Representation.
    networkGraph.getRenderContext().setVertexIconTransformer(new Transformer<Number,Icon>() {
         public Icon transform(final Number Ver)  {
                Icon transformedIcon=getVertexTransformer(Ver)
                 return transformedIcon; }});
  }}}
public Icon getVertexTransformer(Number vertex)      {
     int vertexID=vertex.hashCode();
     Icon tranformedICon;
     //Check Whether this node is overlapped by the MagnifiedView Lens
    Point2D p= new Point2D.Double( ((AbstractLayout<Number, Number>) graphLayout).getX(vertex),
                    ((AbstractLayout<Number, Number>) graphLayout).getY(vertex));
    Point2D vertexLocation =
                networkGraph.getRenderContext().getMultiLayerTransformer().transform(p);
     //Return graphical attribute reprsentation icon if the node is overlappeed by the lens
    if(magnifiedViewLens.getLensTransformer().getEllipse().contains(vertexLocation))
           tranformedICon= new DynamicIcon(vertexID,
                    lensAttribute.doubleAttributeVisible[currentLensIdentification],
                    lensAttribute.doubleAttributeColor[currentLensIdentification],
                    lensAttribute.stringAttributeVisible[currentLensIdentification],
                    lensAttribute.stringAttributeColor[currentLensIdentification],
                     readGML,iconWidth,
                     lensAttribute.lensSelectionType[currentLensIdentification]);
         return tranformedICon;
    }
    //Return default icon if the node is not overlapped by the lens
     else{
           if (NetworkLensApplication.verticeColorAttributeIsSet)
                return new DefaultIcon(vertexID,
                           NetworkLensApplication.verticeColorAttributeID,readGML);
            else
                     return new DefaultIcon();
       }
 }
```

4.1.4 Interactive Functions Provided by JUNG.

Besides the capability to create network graph and setting up the basic construction of Network Lens, JUNG is able to provide some interactive functions to offer users to explore more information of the network. In general, the following interactive

functions are developed by JUNG:
Scalability: JUNG provides a common frame to display the visualization view. Every frame is scalable by adjusting the scale bar of the frame.
Zooming: By adjusting the magnification time of the magnifying effect of the lens, user can zoom in and zoom out on special area. In the implementation part, we set a variable to measure and control the magnification time.

```
public static float lensmagnification=1;
lens.getLensTransformer().setMagnification(lensmagnification);
```

As illustrated in the Section 3.4.1, besides the method providing by JUNG (by executing Ctrl+ Scroll Wheel), a much more intuitive method is implemented as well. That is, a scroll slide is assigned to finish this task. Here is how we implement it: We create a slider named *magnifiedSlider* and add a listener to it. Once the *magnifiedSlider* is changed, the magnification time is reset.

Picking: As we described, the interactive function **picking** makes it possible for our user to compare different nodes of the network diagram. Actually, JUNG also supports this function.

4.2 Multivariate Data Visualization Techniques Implementation

The process to visualize the multivariate data is not very complicated, since we just implement some basic multivariate data visualization techniques, such as bar charts, star plots and parallel coordinate. The data visualization process model depicted in Figure 2.11 is referred to our implementation when these data visualization techniques are performed.

4.2.1 Data Table Structure

The data table structure is the basis for the data visualization technique. In this application, each node holds quantitative attribute and nominal (inc. ordinal) attributes. When we process the node, we just use the data structure, **Array**, to store: quantitative attribute names, quantitative attribute value, nominal attribute names and nominal attribute value correspondingly for each nodes. As shown in the following code, each node possesses four array list, quantiAttrValue, quantiAttrName, nominalAttrValue, nominalAttrName, where quantiAttrValue is to hold the quantitative attribute value; the quantiAttrName is to hold quantitative attribute name, the same to nominal attribute.

```
ArrayList<String>  quantiAttrValue= new ArrayList<String>();
ArrayList<String>  quantiAttrName = new ArrayList<String>();
// These two array list are used to store the nominal type attribute value and name.
ArrayList<String> nominalAttrValue = new ArrayList<String>();
ArrayList<String> nominalAttrName   = new ArrayList<String>();
```

After the data is well organized and store in the data table structure, the next task is to map the structural data into the visual format. As depicted in Section 2.5, the data will be transformed to some graphical representation, which will be used as icons for the nodes. The following code is an implementation segment on how to construct the icons for each node.

```
public DynamicIcon(int nodeID, boolean[] selectedQuanti_Attr,
                   Color[] selectedQuanti_Attr_Color, boolean[] selectedNominal_Attr,
                   Color[] selectedNominal_AttrColor, ReadGraphFile parsedGMLfile,
                   int icon_width, int visualizationType) {
```

The usage of these parameters will be illustrated in the Table 4.1.

Parameter Name	Usage/ Purpose
nodeID	To represent the identification of the node
selectedQuanti_Attr	To mark which quantitative attributes that are selected to show in the icon
selectedQuanti_Attr_color	To mark which colors are chosen to paint the corresponding selected attributes
selectedNominal_Attr	To mark which nominal attributes that are selected to show in the icon
selectedNominal_AttrColor	To mark which colors are chosen to paint the corresponding selected attributes
prasedGMLfile	To have access to get the attribute name and attribute value of the node
icon_Width	To illustrate the width of the icon
visualizationType	To illustrate which data visualization technique is applied to visualize the attribute value of the node.

Table 4.1 Parameters for the Class DynamicIcon Constructor

From Section 3.3.4, we know that the dynamic icon will be rendered with different visual presentation by using different visualization techniques. In order to make the explanation much clearer, suppose that the visualization technique we are using is bar charts. Then, we would draw a bar charts representation on the icon. The constructor for the bar charts is implemented as following

```
public BarGraph(int start_Pos_X, int start_Pos_Y, ReadGraphFile parseGMLFile, int nodeID,
            boolean[] selectedQuanti_Attr,Color[] selectedQuanti_Attr_color,int iconWidth) {
```

The purposes of the some parameters in this constructor have the same intentions as they were illustrated in the Table 4.1. Table 4.2 is used to make some complementarities of the Table 4.1. The following code fragment is to display the implementation of bar charts:

```
for(int i=0;i<barNumber;i++)  //
{
    //Get the value of the selected attribute
   String value=extractedNodeInfo.nodeAttribute[nodeID]
                              .getDoubleAttributeValue(attributeSelection[i]);
     double currentBarLength=Double.parseDouble(value);    // Raw data
     double maxValue=getMaxValue(attributeSelection[i]);   // The upper bound of the range
     double normalizeFactor=maxValue/(double)chartWidth;
     if(normalizeFactor!=0)
          currentBarLength=currentBarLength/normalizeFactor; //Normalized result
     drawLabeledBar((int) currentBarLength,0,barWidth,graphObject,i);
  } }
/** This function is used to draw a bar and the length of the bar is based on the value of the
attribute */
private void drawLabeledBar(int value, int barNumber, int barWidth,Graphics graphObject,int i)
{ // Auto-generated method stub
     int barHeight=value;
     int Y=startY+chartWidth-barHeight;
     int X=startX+i*barWidth;
     graphObject.setColor(attributeColor[attributeSelection[i]]);
     graphObject.fillRect(X,Y,barWidth,barHeight);
}
```

Variables	Purpose/ Representation
start_Pos_X	To represent the start point of X axis of the bar charts
start_Pos_Y	To represent the start point of Y axis of the bar charts
barNumber	To represent the index of the bar in the bar chart
attributeSelection[i]	To store which attributes are ready to draw at the bar index *i*.
barwidth	To represent the width of each bar chart
currentBarLength	To represent the length of the bar chart this is mapped to value of the corresponding attribute.
normilizedFactor	To represent the factor to normalize the value of the attribute to a normal value that can be transformed in a bar charts.

Table 4.2 Parameters for the BarGraph Constructor

By referring to the implementation of bar chart, the parallel coordinates and star plots can be implemented in a similar way. After this step is completed, we are at the end of building up the Network Lens tool.

5 Discussion

In this chapter, the achievement and current limitation of the project will be brought out. By discussing the achievement, we can testify why the Network Lens can stand out as a powerful tool to explore attributes from a network. Meanwhile, the limitation would help us to acknowledge the aspects to improve.

5.1 Achievement

In the Section 2.3, several approaches to visualize the attributes of a network have been presented. Meanwhile, the negative aspects of these approaches have been discussed as well. Our approach, Network Lens improves these negative effects in the following ways.

- The Network Lens integrates the data visualization into the node representation of graph drawing. To some extends, it avoids clutter and optimizes the use of the space. Compared with the *Integration Data Visualization into Graph Drawing* approach discussed in Section 2.3.3, our integration method will not cause overlapping and have a pretty nice Focus + Context effect that the former approach does not support
- The Network Lens provides handy scalability of the graph layout and graph view to overcome the restriction of the space, whereas, the scalability is an inconvenient aspect of the *Multiple Graph Drawing* method.
- The Network Lens provides the function to compare several nodes of the network at the same time in order to bring the users a deeper insight of the attributes of the node by comparison. While Vizster just allows to purchase the attributes of a node at one time.

The Network Lens also provides some other functions to stand out from others in the following aspects:

- The Network Lens supports the saving and loading lens(es) function. Users can preserve some lenses by enforcing saving lens(es) function in order to save the lens(es) they have created. Meanwhile, if users want to use lens(es) they created before, they can just load the lens(es). This function is likely to help users to avoid some repeating work and save time.
- The Network Lens supports the combining lenses function. Users can create a new lens by combining two lenses. Supposed that a user has created two lenses for two different points of emphasis, she/he would like to set up a lens that will embody these two lenses. Then, she/he just need combine the two lenses and select proper operation for the combination.
- The Network Lens provides various multivariate data visualization techniques. In this work, we implemented bar charts, parallel coordinates and two versions of star plots. User can choose different data visualization techniques to gain a deep insight of the attributes of the network.

5.2 Limitations

Although Network Lens is a good tool to visualize the attributes of a network, we appreciate the limitations of it. A non-ignorable one is that the high distortion caused by the lens. The lens applies distortion to magnify the nodes of the graph, whereas, the lines of the graph are bended due to the distortion. The bending of the lines can cause some slide effects in our approach because the bending will make it a hard work to find the relations between nodes sometimes.

6 Conclusions

In the chapter, we will conclude our work with the final outcome by giving a general describtion of the Network Lens in section 6.1. Future work to improve this master thesis is also introduced in this chapter.

6.1 Summary

Equipped with the knowledge and components mentioned in the Section 2 and Section 3, we have created a Network Lens to assist the user to analyze a complex network. All the facilities provided by this Network Lens can meet the requirements proposed at the beginning: our goal is to explore more information of a network without clutter and time-consuming operation. The Network Lens can help users to explore and analyze information from a network without clutter and time-consuming. Meanwhile, as for interactive information visualization, the Network Lens contributes a serial of interactive designs to explore and analyze the information of a network, such as, the features to explore and compare several entities at the same time without clustering, the implementation to combine two different lenses and the several available data visualization techniques to visualize the attributes.

Rather than creating a novel idea, the Network Lens is an experiment to employ some well-known techniques and apply them to construct a user-friendly tool. The topics of data visualization and magic lens have been discussed and studied for quite long time. The whole process that we are getting involved is just to combine this knowledge in an appropriate way. It is a convenient tool to explore and analyze network, at least, the following advantages can highlight this tool.

- **General Usage**. Any network represented by a GraphML file can be loaded by the tool.
- **Easy to Manipulate**. This tool provide a neat graphic interface to navigate the people
- **Various Interactive Techniques**. A good many interactive designs are available to explore and analyze the network. It gives the user a lot of freedom to play with this tool.
- **Visualization of network with multivariable attributes.** In general, the Network Lens is a good interactive tool to explore and analyze the network.

The main idea of this work has been published in a conference paper [51] written together with I. Jusufi and A. Kerren**.** It was also presented at this conference.

6.2 Future Work

Although our visualization tool is capable to visualize the attributes of a network, there are still some factors that can help to improve the usefulness of our tool. Our future work will be focused on designing a usability study, experimenting with time dependent data, implementing more visual representations and data quality issues.

References

[1]: Tony Morden, *Principle of Management* 2nd edition, publised by Ashgate Publishing Limited, Burlinton, VT05041_4405, USA, 2004.

[2]: Daniel A. Keim, Florian Mansmann, Jorn Schneidewind and Hartmut Ziegler, *Challenges in Visual Data Analysis*, Proceedings of Information Visualization (IV 2006), IEEE, p. 9-16, 2006.

[3]: Colin Ware, *Information Visualization: Perception for design*, Morgan Kaufmann publisher, December 31, 1999.

[4]: Robert Spence, *Information visualization: design for interaction*, second edition, published by ACM press 2007.

[5]: C. Erten, S. G. Kobourov, V. Le, and A. Navabi, *Simultaneous Graph Drawing: Layout Algorithms and Visualization Schemes*, University of Arizona, supported by NSF under grant ACR-0222920.

[6]: http://jung.sourceforge.net/
Accessible: April 10th, 2010

[7]: http://prefuse.org/
Accessible: April 10th, 2010

[8]: http://pajek.imfm.si/doku.php
Accessible: April 10th, 2010

[9]: Eric A. Bier, Maureen C. Stone, Ken Pier, William Buxton and Tony D. DeRose *Toolglass and Magic Lenses: The See-Through Interface*. Proceedings of Siggraph 93, Computer Graphics Annual Conference Series, ACM, 1993.

[10]: Eric A. Bier, Maureen C. Stone, Ken Fishkin, William Buxton and Thomas Baudel, *A Taxonomy of See-Through Tools*, ACM CHI94-4/94 Bostan, Massachusetts, USA,1994.

[11]: Maureen C. Stone, Ken Fishkin and Eric A. Bier, *The Movable Filter as a User Interface Tool*, ACM, CHI94-4/94 Bostan, Massachusetts USA, 1994

[12]: http://www2.parc.com/istl/projects/MagicLenses/SimpleDemo.html
Accessible: April 10th, 2010

[13]: http://www.britannica.com/EBchecked/topic/706263/Xerox-PARC
Accessible: April 10th, 2010

[14]: Richard A. Becker, Stephen G. Eick, Allan R. Wilks, *Visualization Network Data*, in the IEEE Transactions on Visualization and Computer Graphics, Vol. 1, No. 1, pages 16-21, March 1995.

[15]: Keim, D. A. and Kriegel, H, *Visualization techniques for mining large databases: A comparison, Transactions on Knowledge and Data Engineering*, Special Issue on Data Mining, 8(6):923–938, 1996.

[16]: Robert Kosara, Helwig Hauser and Donna L. Gresh, *An Interaction View on Information Visualization*, EuroGraphic Association, 2003.

[17]: Andreas Kerren, John T. Stasko, Jean-Daniel Fekete and Chris North, *Information Visualization: Human-Centered Issues and Perspectives*, Volume 4950 of LNCS State-of-the-Art Survey, Springer.

[18]: Benjamin B. Bederson and Ben Shneiderman, *The Craft Information Visualiza-*

tion: Readings and Reflections, Morgan Kaufmann, 2003.

[19]: Tamara Munzner, *Interactive visualization of large graphs and networks*, Stanford University, CA, USA, 2000.

[20]: Franz J. Brandenburg, *Nice Drawings of Graphs are Computationally Hard*. In P. Gorney and M. J. Tauber, *Visualization in Human-Computer Interaction, Lecture Notes in Computer Science 439*, pages 1–15. Springer-Verlag, 1988.

[21]: http://en.wikipedia.org/wiki/Graph_drawing
Accessible: April, 10th, 2010

[22]: Douglas Charles Blood, Virginia P. Studdert and Clive C. Gay, *Saunders Comprehensive Veterinary Dictionary*, 3rd edition. Elsevier Saunders, 2007.

[23]: Stephen Few, *Data Visualization, past, present and future*, 2007.

[24]: Zhao Kaidi, *Data visualization*, National University of Singapore.

[25]: Jeffrey Heer and Danah Boyd, *Vizster*: *Visualizing Online Social Networks*. Proceedings of the 2005 IEEE Symposium on Information Visualization Page: 5, ISBN ~ ISSN:1522-404x , 0-7803-9464-x, 2005.

[26]: Mehdi Dastani, *The Role of Visual Perception in Data Visualization*. Journal of Visual Language & Computing, Pages 601-622, December 2002.

[27]: José F. Rodrigues Jr., Agma J. M. Traina, Maria Cristina F. de Oliveira and Caetano Traina Jr., *Reviewing Data Visualization: an Analytical Taxonomical Study*. Proceedings of the Information Visualization (IV'06), 2006.

[28]: http://www.infovis-wiki.net/index.php?title=Data_Scale
Accessible: March 18th, 2010

[29]: http://en.wikipedia.org/wiki/GraphML
Accessible: March 20th, 2010

[30]: http://en.wikipedia.org/wiki/Java_API_for_XML_Processing
Accessible: March 20th, 2010.

[31]: Colin Ware, *Information Visualization: Perception for Design*,2nd edition, Morgan Kaufmann, 2004.

[32]: Richard Resnick, *Visualizing Nominal Data*, CS563, Advanced Graphics, Feb 25th, 1997.

[33]: http://current.com/1e70c4c Available: March 25th, 2010

[34]: http://graphml.graphdrawing.org/primer/graphml-primer.html
Accessible: April 4th, 2010.

[35]: Andreas Kerren, *Information Visualization I*, course lectures, Växjö University, spring, 2009.

[36]: Ed H. Chi, *A Framework for Information Visualization Spreadsheets*. Ph.D.Thesis. University of Minnesota, Computer Science Department. March, 1999

[37]: R.Shanon, T.Holland, and A.Quigley, *Multivariate graph drawing using parallel coordinate visualization*. Technical report, University College Dubilin, School of computer Science and Informatics, 2008.

[38]: 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.

[39]: V. Friedman, *Data Visualization and Infographics*, January 14, 2008.
[40]: http://en.wikipedia.org/wiki/Information_visualization
Accessible: April 10th, 2010.
[41]: Alexander G. Gee, Min Yu, Hongli Li and Georges G. Grinstein, *Dynamic and Interactive Dimensional Anchors for Spring-Based Visualizations*. Technical Report, Computer Science, University of Massachusetts Lowell.
[42]: University of Illinois at Urbana-Champaign Digital Libraries Initiative, UIUC, DLI Glossary, Created: November 23rd, 1998.
http://dli.grainger.uiuc.edu/glossary.htm
[43]: http://en.wikipedia.org/wiki/Graph_drawing
Accessible: April 24th, 2010.
[44]: S. G. Eick and A. F. Karr, *Visual Scalability*, Journal of Computational & Graphical Statistics, vol. 11, no. 1, pp. 22-43, 2002.
[45]: Stuart K. Card, Jock Machinlay, and Ben Shneiderman, *Readings in Information Visualization: Using Vision to Think*, pp. 1-34, Morgan Kaufmann publishers, San Francisco, Califonia, 1999
[46]: Zhu Bin and Hsinchun Chen. *Information Visualization*. Annual review of information science and technology 39, 1, 139-177, 2005.
[47]: J. D. Fekete and C. Plaisant, *Interactive Information Visualization of a Million Items*. Processings of the IEEE Symposium on Information Visualization, pp. 117-124, 2002.
[48]: http://ajaxfinder.com/9390/
Accessible: April 24th, 2010.
[49]: http://www.evl.uic.edu/aej/526/kyoung/Training-parallelcoordinate.html
Accessible: April 24th, 2010.
[50]: http://commons.wikimedia.org/wiki/Category:Star_plots
Accessible: April 24th, 2010.
[51]: I. Jusufi, Y. Dingjie, and A. Kerren, *The Network Lens: Interactive Exploration of Multivariate Networks Using Visual Filtering*. In Proceedings of the 14th International Conference on Information Visualization (IV '10), London, UK, IEEE Computer Society Press. 2010.
http://cs.msi.vxu.se/isovis/pubs/
Accessible: May 17th, 2010
[52]: http://graphml.graphdrawing.org/primer/graphml-primer.html
Accessible: May 23rd, 2010
[53]: Carsten Görg, Mathias Pohl, Ermir Qeli and Kai Xu, *Visual Representations*, in the book, *Human-centered Visualization Environments*: GI-Dagstuhl Research Seminar, Dagstuhl Castle, Germany, March 5-8, 2006: revised lectures, Andreas Kerren, Achim Ebert, Jörg Meyer (eds.).
[54]: http://en.wikipedia.org/wiki/Information_visualization
Accessible: May 27th, 2010

Linnæus University
School of Computer Science, Physics and Mathematics

SE-351 95 Växjö / SE-391 82 Kalmar
Tel +46-772-28 80 00
dfm@lnu.se
Lnu.se