INTERACTIVE VISUAL DATA EXPLORATION: A MULTI-FOCUS APPROACH

by

Jian Zhao

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto

ABSTRACT

Interactive Visual Data Exploration: A Multi-Focus Approach

Jian Zhao
Doctor of Philosophy
Graduate Department of Computer Science
University of Toronto
2015

Recently, the amount of digital information available in the world has been growing at a tremendous rate. This huge, heterogeneous, and complicated data that we are continuously generating could be an incredible resource for us to seek insights and make informed decisions. For this knowledge extraction to be efficient, *visual exploration* of data is demanded in addition to fully automatic methods, because visual exploration can integrate the creativity, flexibility, and general experience of the human user into the sense-making process through interaction and visualization techniques.

Due to the scale and complexity of data, robust conclusions are usually formed by coordinating many sub-regions in an information space, which leads to the approach of *multi-focus* visual exploration that allows browsing different data segments with multiple views and perspectives simultaneously. While prior research has proposed a myriad of information visualization techniques, there still lacks comprehensive understanding about how visual exploration can be facilitated by multi-focus interactive visualizations. This dissertation investigates issues and techniques of multi-focus visual exploration through five design studies, touching various types of data in a range of application domains.

The first two design studies address the exploration of numerical *data values*. KronoMiner presents a multi-purpose visual tool for exploring time-series based on a dynamic radial hierarchy; and the ChronoLenses system supports exploratory visual analysis of time-series by allowing users to progressively construct advanced analytical pipelines. The third design study focuses on the exploration of logical *data structures*, and presents DAViewer that facilitates computational linguistics researchers to explore and compare rhetorical trees. The last two design studies consider the exploration of heterogeneous *data attributes* (or facets). TimeSlice facilitates the browsing of multi-faceted events timelines by organizing visual queries in a tree structure; and PivotSlice aids the mining of relationships in multi-attributed networks through a dynamic subdivision of data with customized semantics.

This dissertation ends with critical reflections and generalizations of the experiences obtained from the case studies. High-level design considerations, conceptual models, and visualization theories are distilled to inform researchers and practitioners in information visualization for devising effective multi-focus visual interfaces.

ACKNOWLEDGEMENTS

Over the last six years, my doctoral study has been an invaluable experience in my life. It has been a challenging adventure, and I was fortune to receive support from many people. They have deeply inspired my minds and shaped my thinking. I would like to thank all of them, including those who are not mentioned here by name.

My greatest thanks must go to my supervisor, Ravin Balakrishnan. He was the first to bring me into the academic world and open my eyes. I could still remember the excitement when I received a phone call from him to chat about my future graduate school study, and at that time I was only an undergraduate student in the far east China. As a supervisor, Ravin has been a great resource of both inspiration and critique, and a constant support of both research and life. I am deeply grateful for the opportunities to work with him. It was mostly from him that I learned how to be a good researcher and how to explore the world with novel approaches and critical thinking.

Many thanks are also due to my committee members, Khai Truong and Karan Singh, for their valuable time and constructive feedback on my thesis. It was also very delightful to have Fanny as my external advisor from the industry. Moreover, it was a great honor to have John Stasko, a well-respected researcher in the academic community, as my external examiner. Also, thank Gerald Penn and Clifton Forlines for coming to my final oral exam. In addition, I would like to thank Christopher Collins for his great help on many publications included in this thesis, although he is not involved in my committee. Fanny and Chris were the two closest collaborators during my graduate study. They were my invaluable sources for advices on the information visualization side for most of my work.

Industrial internship experience was an essential part of my doctoral study, during which I met many brightest people and had enjoyable time. I have visited Microsoft Research twice, and in particular I would like to thank Steven Drucker, Danyel Fisher, Bongshin Lee, and Saleema Amershi for their unwavering support during our collaborations. Many thanks also go to Donald Brinkman, Mary Czerwinski, and Patrice Simard for allowing me to view research from new perspectives. Moreover, during my internship at Adobe Research, I enthusiastically worked with Mira Dontcheva, Leo Zhicheng Liu, and Aaron Hertzmann who provided me with the greatest inspirations and suggestions. Also, I wish to express my thanks to Michelle Zhou, Fei Wang, and Liang Gou for their persistent help and encouragement during my visit at IBM Research. Lastly, I would like to thank all my colleagues and fellow interns from those internships.

My research has also been benefited from the day-to-day life at the DGP lab at University of Toronto. Special thanks to John Hancock and Ingrid Varga for their energy and effort on keeping everything working, including systems, software, networking, and so forth. I had many helpful discussions with my lab-mates, especially Michael Glueck, Ricardo Jota, Rorik Henrikson, and Haijun Xia. I am thankful for insights and advices from our weekly meetings/brainstorming sessions. At University of Toronto, I have had the opportunities to work with other talented professors, visiting scholars, postdoc researchers, and graduate students. Particularity, I want to thank William Soukoreff, Xiangshi Ren, Daniel Wigdor, and Vanessa Wei Feng. At last, thanks go to all my co-authors during all kinds of collaborations and all the people who surround me over the years; they have greatly influenced my experiences and my views in research.

Finally, there are people who deserve thanks more than others. I am deeply grateful to my parents, Shurong Liu and Jingzhi Zhao, who have always stayed by my side and encouraged me to continue my studies. I owe my sincerest gratitude to my wife, Tianmiao Niu, for her comfort and love as a partner. With her my life at graduate school has been filled with the most happy and joyful memories.

PUBLICATIONS

Many of the ideas and figures contained in this article have been previously published in the following peer-reviewed publications:

- Jian Zhao, Fanny Chevalier, and Ravin Balakrishnan. KronoMiner: Using Multi-Foci Navigation for the Visual Exploration of Time-Series Data. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'11)*, pp. 1737–1746, Vancouver, Canada, 2011. DOI: 10.1145/1978942.1979195. (Chapter 3).

- Jian Zhao, Fanny Chevalier, Emmanuel Pietriga, and Ravin Balakrishnan. Exploratory Analysis of Time-series with ChronoLenses. In *IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis'11)*, 17(12), pp. 2422–2431, Providence, USA, 2011. DOI: 10.1109/TVCG.2011.195. (Chapter 4).

- Jian Zhao, Fanny Chevalier, Christopher Collins, and Ravin Balakrishnan. Facilitating Discourse Analysis with Interactive Visualization. In *IEEE Transactions on Visualization and Computer Graphics (Proceedings of InfoVis'12)*, 18(12), pp. 2639–2648, Seattle, USA, 2012. DOI: 10.1109/TVCG.2012.226. (Chapter 5).

- Jian Zhao, Steven Drucker, Danyel Fisher, and Donald Brinkman. TimeSlice: Interactive Faceted Browsing of Timeline Data. In *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI'12)*, pp. 433–436, Capri Island, Italy, 2012. DOI: 10.1145/2254556.2254639. (Chapter 6).

- Jian Zhao, Christopher Collins, Fanny Chevalier, and Ravin Balakrishnan. Interactive Exploration of Implicit and Explicit Relations in Faceted Datasets. In *IEEE Transactions on Visualization and Computer Graphics (Proceedings of VAST'13)*, 19(12), pp. 2080–2089, Atlanta, USA, 2013. DOI: 10.1109/TVCG.2013.167. (Chapter 7).

CONTENTS

List of Tables

List of Figures

Part I

On Visual Data Exploration

Chapter 1

INTRODUCTION

A picture is a poem without words.

Horace

During the last few decades, advances in computer technologies, such as Internet, database systems, and various electronic sensors, have allowed people to collect and store vast amount of data at an incredible speed. Almost in every discipline or branch of industry, researchers and practitioners aim to turn this raw data into useful knowledge that could inform decision making in the future. Such digital information often has huge scales, heterogeneous contents, and complicated structures; thus people rely on computers to process and filter the data, and further make predications. In correspondence, the power of computers has grown increasingly fast at many aspects, such as computational speed and storage size; at the same time, many advanced mathematical models and algorithms have been developed for automatically discovering hidden rules in data. However, these fully-automated approaches may only work reliably for well-defined and well-understood problems, whereas in many real-world applications, the above process of *knowledge discovery* and *decision making* can be much more challenging because it often differs under various circumstances. Making it worse, many of the analysis problems are *ill-defined*. For example, one situation is that users sometimes only have vague hypotheses or even no hypothesis before analyzing the data, therefore they have no idea on how to choose an appropriate algorithm or where to start the analysis.

The above problem leads to the approach of *exploratory data analysis* [Tukey, 1977]. Different from the analysis that is more goal-directed, exploratory data analysis aims to suggest hypotheses, assess assumptions, and collect insights for the basis of further investigation. However, this process might be tedious by just looking at the raw numbers. Fortunately, computers also allow *visual exploration* of data through interactions and visualizations. Visual exploration empowers human perceptual capabilities with visual interfaces to drive data navigation, actively engaging users' prior-knowledge into the exploration process to make insights discovery much more efficient [Keim, 2002]. In particular, visualizations can augment a user's cognition abilities via different graphical representations of abstract data, and interactions can support dynamic manipulation of visualization objects driven by a user's analytical goals. Fekete et al. [2008] have discussed many benefits of information visualization. In short, interactive visualization techniques are critical for comprehending the growing digital information by integrating human's creativity, flexibility, and general knowledge into the data exploration process.

Further, it is important to note that the datasets we want to analyze today are often large in scales, rich in attributes, and complex in structures. For example, Facebook[1] has billions of users with a variety of personal characteristics; and those users form a large and complicated social network and post vast amount of information everyday including texts, pictures and videos. In order to better support analyses of such data and promote opportunistic findings, different portions of a dataset need to be presented simultaneously with suitable visual encodings, proper levels-of-detail, and coordinated interactions in a coherent schema. This *multi-focus* mechanism can maximize the benefits of visual exploration, allowing users to correlate basic findings in various data parts from different perspectives, thus to form deeper insights. Previous studies have demonstrated the efficiency of this category of techniques, such as multi-view image browsers [Plaisant et al., 1995], multiple coordinated views [Wang Baldonado et al., 2000], multi-point interaction [Shoemaker and Gutwin, 2007], and multi-focus distortion [Elmqvist et al., 2010b]. Here we generally define *multi-focus visual exploration* as the process of leveraging interactive coordination of multiple logically-related data perspectives in a unified visualization to achieve an integral user task as a whole. Detailed discussion of this concept and its relations to the previous work will be described in Section 2.4.

The primary goal of this dissertation is to substantiate our understandings of the possibilities of multi-focus visual exploration enabled by interactive visualizations in supporting effective knowledge discovery and decision making of real-world datasets. In the following chapters, we will approach this objective by reviewing the state of the art, proposing novel visualization techniques in various case studies, and distilling design implications for developing useful multi-focus visual interfaces.

The structure of this chapter is as follows. First, we further elaborate the motivations of this work (Section 1.1), followed by a general introduction of the research areas and basic background related to this dissertation (Section 1.2). After framing the thesis problem and describing our research methodology at a higher level (Section 1.3), we outline the

[1]https://www.facebook.com.

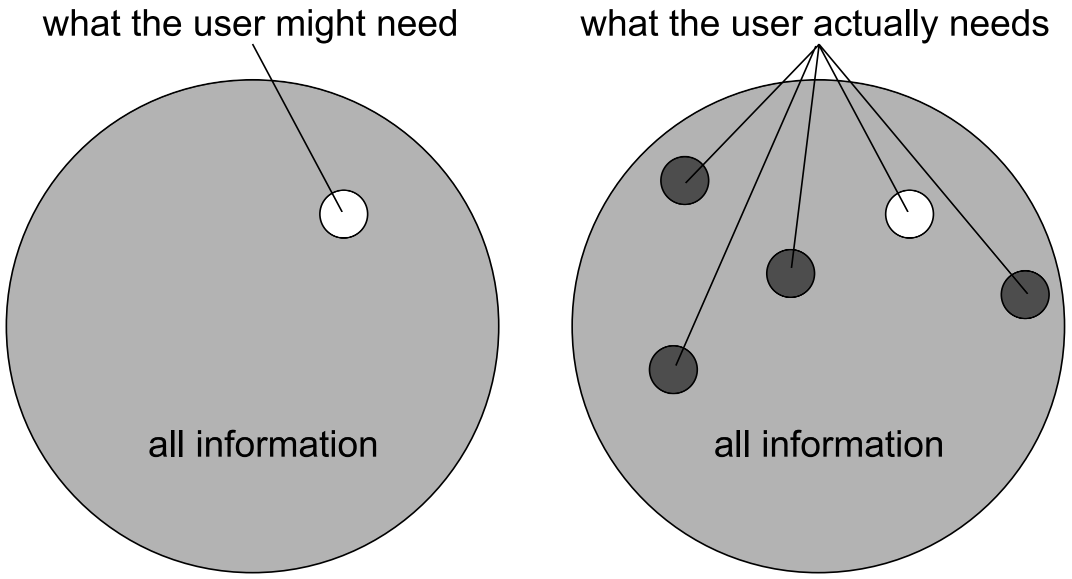


Figure 1.1: Multi-focus visual exploration is needed in sense-making tasks with large amount of information. In the sea of information, it is difficult to identify what the user needs (left). Making it worse, to make a robust conclusion, the user often needs to explore and correlate multiple parts of the data that are initially hidden from the user's expectation (right).

contributions of this research (Section 1.4). Finally, we present an overview of the structure of this dissertation (Section 1.5).

1.1 Motivation

Nowadays, the increasingly fast growth of digital information, both in scale and in complexity, has imposed many challenges for people to find valuable insights and make reliable decisions. Even with the advances of computer technologies, current data management systems can only display quite a small portion of a dataset; and making matters worse, the data is usually presented in raw formats, such as data tables and textual strings. In those cases, the amount of input simply exceeds the cognitive processing capacity of humans which is limited by the bandwidth of their senses and the size of their memory. A user's attention is overwhelmed in such a way of communicating information. Moreover, the recent emergence of *big data* suggests that there exists abundance digital information to exploit for discovering useful opportunities. As the left side of Figure 1.1 shows, it is very difficult to find the right "needle in the sea" of the overwhelming information. This dissertation is driven by the above issues and challenges, situating in the overall increasing trends of research and development in interactive visualization systems for exploring the growing information spaces.

Further, even with means of visualization, presenting the whole dataset at the same time could be overwhelming, thus reducing the saliency of patterns; whereas displaying one single subset of data may constrain the discovery of underlying rules and serendipitous findings. Studies in cognitive science have indicated that human brains recognize an object better when multi-channel *inputs* are presented, such as color, shape, and depth [Myers, 2011]; and people can manipulate things more efficiently when generating multiple *outputs* to the system, such as bi-manual interactions [Buxton and Myers, 1986; Kabbash et al., 1994]. Similarly, for visualizations and interactions to be effective in amplifying the inputs and outputs between the user and the system, we can wisely select multiple important portions of data to be shown with proper visual encodings or visual highlights within the global context. In such way, we are able to best exploit the relative high-bandwidth of our visual system, enabling easy connection and comparison of data patterns in multiple subsets to derive cross-concept relationships. As the right side of Figure 1.1 shows, several parts in data, which may be initially hidden and are out of a user's original expectation, need to be revealed and correlated in order to discover more valuable and substantial insights. While prior research has proposed a myriad of techniques to assist visual data exploration, there exist great opportunities and values in deeply understanding how to better support knowledge discovery and decision making in visual exploration with multi-focus interactive visualizations. Therefore, our research is also motivated by the needs of investigating this category of visualization techniques to maximize the input and output bandwidths between the user and the data.

1.2 Background

This dissertation focuses on investigating interaction and visualization techniques supporting multi-focus visual data exploration. In this section, to set the stage, we discuss the most relevant research areas, and describe the space of available data to be visualized in the real world.

1.2.1 Scope of Thesis Research

The work of this dissertation is mainly related to three overlapping research areas: Information Visualization, Visual Analytics, and Human-Computer Interaction. Detailed overviews of the previous research are provided in Chapter 2. Subsequent chapters will further discuss specific related work when appropriate. Here, we briefly introduce the general background of the three related fields to define the scope of our research.

Information Visualization, or InfoVis, is the research of investigating "the use of computer-supported, interactive, visual representations of abstract data to amply cognition" [Card et al., 1999]. In other words, InfoVis studies computer-based technologies to bridge the external visualization, "diagrams in the world", and the internal visualization, "diagrams in the mind", where their relationships are discussed in Hegarty [2004]'s writings. InfoVis particularly focuses on exploring semantics and meanings of abstract nonspatial data, through novel graphical representations and interaction techniques, which is a source of a user's *external cognition*. External cognition means using external aids to help solve cognitive problems in knowledge crystallization [Card et al., 1999]. One interesting example is how information visualization can support *epistemic actions*, which are external actions that change the nature of our internal mental state, as apposed to *pragmatic actions* that achieve our goals in the physical world [Kirsh and Maglio, 1994]. Ware [2004] listed a number of advantages for effective information visualization: providing an ability to comprehend huge amounts of data, allowing the perception of emergent properties that were not anticipated, enabling problems with the data itself to become immediately apparent, facilitating understanding of both large-scale and small-scale features of data, and assisting hypothesis formation. This dissertation mainly focuses on the research of information visualization, since our primary goal is to devise novel interactive visualization techniques to support effective multi-focus visual data exploration.

Visual Analytics is first defined as "the science of analytical reasoning facilitated by interactive visual interfaces" [Thomas and Cook, 2005]. Although originated from information visualization, visual analytics is more than just visualization, which emphasizes the integral process of sense-making [Russell et al., 1993], decision making, and interactive thinking with data. It seamlessly combines many research domains including visualization, human factors, and data analysis [Keim et al., 2008]. In more details, visual analytics is the formation of abstract visual metaphors in combination with a human information discourse in a systematic way which enables detection of the expected and discovery of the unexpected within massive, dynamically changing information spaces [Cook et al., 2007]. To support analysis that is an iterative and evolutionary process of making judgments, visual analytics helps users identify the strengths and weaknesses of automated algorithms and develop a tightly integrated solution of analytical reasoning tasks with appropriate visualization and interaction. Thus, the greatest difference from information visualization is that visual analytics gives higher priority to data analysis through the whole exploration. Our research is related to the field of visual analytics in supporting sense-making and analytical reasoning tasks based on comparison and correlation of data items in a multi-focus manner.

Human-Computer Interaction (HCI) is concerned with the study, design, implementation, and evaluation of interaction techniques between users and computers, where the term is popularized in the book of Card et al. [1983]. HCI is a broad multi-disciplinary research area involving computer science, engineering, design, cognitive psychology, and many other fields. In a larger context, information visualization can be viewed as a subset of HCI. Hutchins [1995]'s distribution cognition theory suggests that knowledge lies across individuals, artifacts, and the environment. The filed of information visualization studies effective visual metaphors to facilitate communications between users and abstract data, which is a cognitive process distributed across one's internal mind, the external data, and the means presenting and manipulating the data. Also, many of the principles in information visualization are founded by the human cognition and perception frameworks of HCI. Focusing on interactive visual exploration of data, the research of this dissertation is affected by the theory and practice of human-computer interaction at many aspects, for example, the design and evaluation of useful and usable interfaces.

1.2.2 Space of Data in Information Visualization

To design efficient visualizations, we must know characteristics of data to be visualized, since the fundamental strategy of visualization is to transform data into certain visual form that exploits human skills in perception and interaction [Card et al., 1999]. There exist several taxonomies of data types in visualization. Shneiderman [1996] classified the data space into: one-dimensional linear, two-dimensional map, three-dimensional world, multidimensional, temporal, tree, and network data, which is widely accepted in information visualization. In a similar approach, Card et al. [1999] grouped 1D, 2D, and 3D as orthogonal axis composition, and Keim [2002] further included data categories such as text & hypertext and algorithm & software. In summary, the data space taxonomy is generally based on the dimensionality of data points (e.g., 1D, 2D, etc.), the conceptual relationship of data items (e.g., tree and network), and the form of data content (e.g., text and algorithm).

It is important to note that those classification categories are not exclusive, and a complicated dataset can be viewed from multiple data type perspectives. For example, a time-series (temporal) data can be 1D or multidimensional based on the number of data attributes to be concerned, and a flight network is also two-dimensional due to the two geographic coordinates of each node. Moreover, a real-world dataset can be derived into different types of data to visualize, based on the specific goals of the visual exploration. For instance, a text corpus can be visualized as a temporal data like in ThemeRiver [Havre et al., 2002], or a hierarchical data like in DocuBurst [Collins et al., 2009a].

Additionally, Bertin [1977] has suggested that there are two fundamental forms of data: data values and data structures. Data values are objects and variables of interests to be visualized; and data structures define the relations that correlate data items to one another. Spence [2006] has also discussed these two data types and lists some common visual encodings

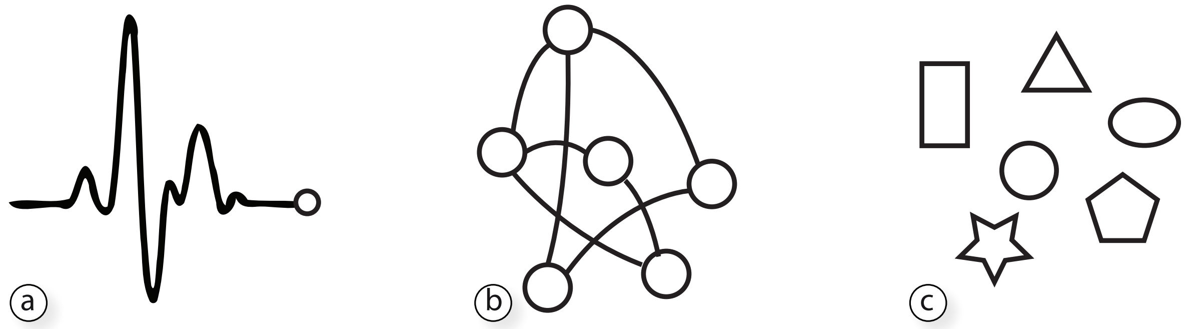


Figure 1.2: Simple illustrations of fundamental data features to explore: (a) data values, (b) data structures, and (c) data attributes.

for them. In addition to data values and data structures, Ware [2004] further added data attributes and described it as a property of some objects and cannot be thought of independently, which is often associated with an entity or relationship. For example, considering a friendship network, a node representing a person in the network is an entity, which is associated with many attributes such as name and gender, and a link representing two-person's relationship may also have some attributes such as strength of connection. Thus, the notion of data attributes is similar to the concept of data values, i.e., variables of interest. For clarification, in this dissertation, we refer data values to simple uniformly formatted data variables (that are often quantitative in many applications, e.g., a series of numbers), and define data attributes as richer and more diverse data values (that are often expressed as metadata of objects including categorical, ordinal, and numerical variables), because the natures of visual exploration of those data values could be dramatically different.

To distinguish, we refer to data types in the first taxonomy (e.g., time-series, tree, and network) as *data formats*, and those in the second one (i.e., data values, structures, and attributes) as *data features* (Figure 1.2). Those two taxonomies are closely correlated. For example, if a user aims to explore how nodes connect with each other in a network, the task can be viewed as data structure related; but if a user cares more about the metadata stored in the nodes, the task becomes data attribute related. The data formats are more concrete that are defined from the perspective of physical characteristics, and the data features are more abstract and fundamental that are extracted from the perspective of user exploration goals. However, as mentioned above, it is difficult to draw clear boundaries between different data formats, whereas each data feature is perceptually distinct. In this dissertation, to adequately investigate multi-focus interactive visualization techniques, we conduct design studies [2] with different data

[2] A design study is a project in which visualization researchers analyze a specific real-world problem faced by domain experts, design a visualization system that supports solving this problem, validate the design, and reflect about lessons learned in order to refine visualization design guidelines [Sedlmair et al., 2012].

forms from various real-world application domains in light of the above two kinds of data type classifications. We choose to frame the entire investigation based on data features to include a complete set of data forms on the basis. At the same time, we strive to touch a wide range of data formats with all the design studies. Details will be discussed in the following section.

1.3 Problem and Approach

The fundamental goal of this dissertation is to investigate critical aspects of multi-focus interactive visualization in facilitating users with the visual exploration, knowledge discovery, and sense-making of data. More specifically, we are interested in answering the following three high-level research questions:

Q1. How would we apply the concept of multi-focus visual exploration in visualizations of different practical datasets?

Q2. How can we design multi-focus interactive visual systems to enable effective data exploration in real-world applications?

Q3. What are implications of multi-focus visualization and interaction techniques for facilitating visual exploratory tasks?

To approach the goals, we first conduct literature reviews to explore important factors for designing interactive visualizations from theories and practices of past research (Chapter 2). Many of the novel techniques proposed in this dissertation are inspired by the related work and grounded by design principles widely accepted in the InfoVis and HCI communities. Specifically, to better address Q1, we survey previous information visualization techniques designed for different data types as well as summarize general themes of interactions for data exploration. In addition, to prepare for answering Q2, we review various design theories, guidelines, and evaluation methods proposed in previous work.

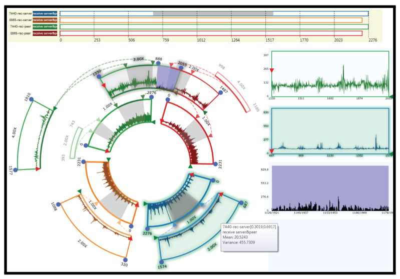
After setting the stage, to further solve Q1 and Q2, we perform five case studies to design, develop, and evaluate multi-focus interactive visualization systems situated in real-world application scenarios (Chapter 3–7). We present these design studies in the aforementioned three important data features for users to explore and analyze — data values, data structures, and data attributes. These data features indicate which aspects of datasets are greatly concerned in practical application domains. More specifically, exploring on data values is concerned with the representation and transformation of low-level numerical values, such as data analysis in finance; the navigation of data structures focuses on the internal conceptual relationships between data items, such as exploring tree structures or graphs; and data attributes exploration

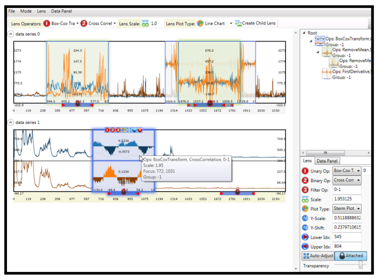
Design Study	Data Feature	Data Format	Application Area(s) Involved
KronoMiner	values	time-series	computer network, climatology, finance, and computational biology
ChronoLenses	values	time-series	computer network, climatology, finance, and astronomy
DAViewer	structures	collections of trees	computational linguistics and machine learning
TimeSlice	attributes	multi-faceted temporal events	general exploration of historical events and flight statistics
PivotSlice	attributes	multidimensional networks	academic literature research

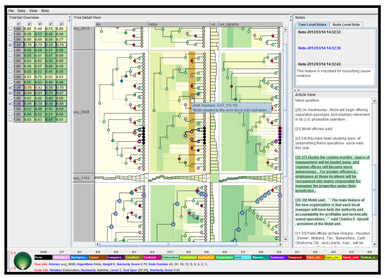
Table 1.1: The space of visual data exploration covered in design studies.

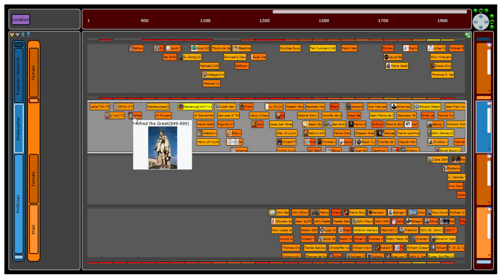
aims to identify semantics, trends, and cross-concept relations expressed with many meta-data properties. We also set these design studies across different application domains, including economics, engineering, and science, and with various data formats, including time-series, multidimensional data, hierarchical structures, and networks. The proposed techniques in design studies, although not exhaustive (in terms of data formats), address most of the data space concerned in information visualization (see Section 1.2.2). Some of them even aim to facilitate the exploration of complicated hybrid data types (e.g., multidimensional networks). Table 1.1 provides the details of the space concerned by the multi-focus interactive visualization technique in each design study. An overall introduction of the prototypes developed in design studies is provided in Section 1.5.

Finally, for answering Q3, we first summarize a set of design considerations for multi-focus interactive visualizations based on the experience learned from the design studies (Chapter 8). We retrospectively apply these high-level design considerations to our case studies, demonstrating how these guidelines can be used for developing multi-focus visual systems in real-world application domains. Further, we extend well-known concepts and frameworks in previous work to versions applied in multi-focus visual exploration, providing fundamental theories and critical implications for designing future multi-focus visualization systems. It is worth noting that those high-level design implications not only address Q3 but also deepen our understandings of issues raised in Q1 and Q2.


(a) KronoMiner


(b) ChronoLenses


(c) DAViewer


(d) TimeSlice

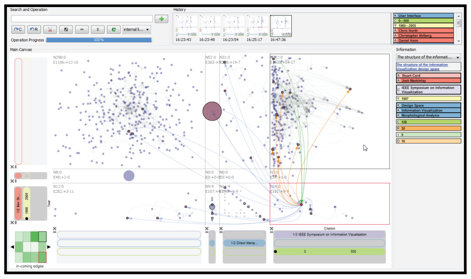

(e) PivotSlice

Figure 1.3: A visual index of research prototypes in design studies. Further video demos can be downloaded through clicking the sub-figure titles.

1.4 Contributions

The main contributions of this dissertation fall into four categories: 1) novel multi-focus interactive visualization techniques, 2) generalizable technical innovations and visualization frameworks, 3) domain-specific design requirements and guidelines, and 4) high-level theoretical implications for designing effective multi-focus visualization.

Firstly, the research prototypes developed in the design studies contribute new multi-focus visualization techniques for interactive data exploration in specific application domains (Figure 1.3):

(a) KronoMiner presents a multi-purpose visualization for exploring time-series based on a dynamic radial hierarchy with flexible interactions (Chapter 3),

(b) ChronoLenses introduce a new method for exploratory visual analysis of time-series that allows users to progressively construct advanced analytical pipelines (Chapter 4),

(c) DAViewer provides an interactive visual tool that facilitates computational linguistics researchers to explore, compare, evaluate, and annotate the outputs of discourse parsers for documents[3] (Chapter 5),

[3]A discourse parser performs analysis of the rhetorical organization of a piece of text by representing it as a tree structure [Marcu, 1999].

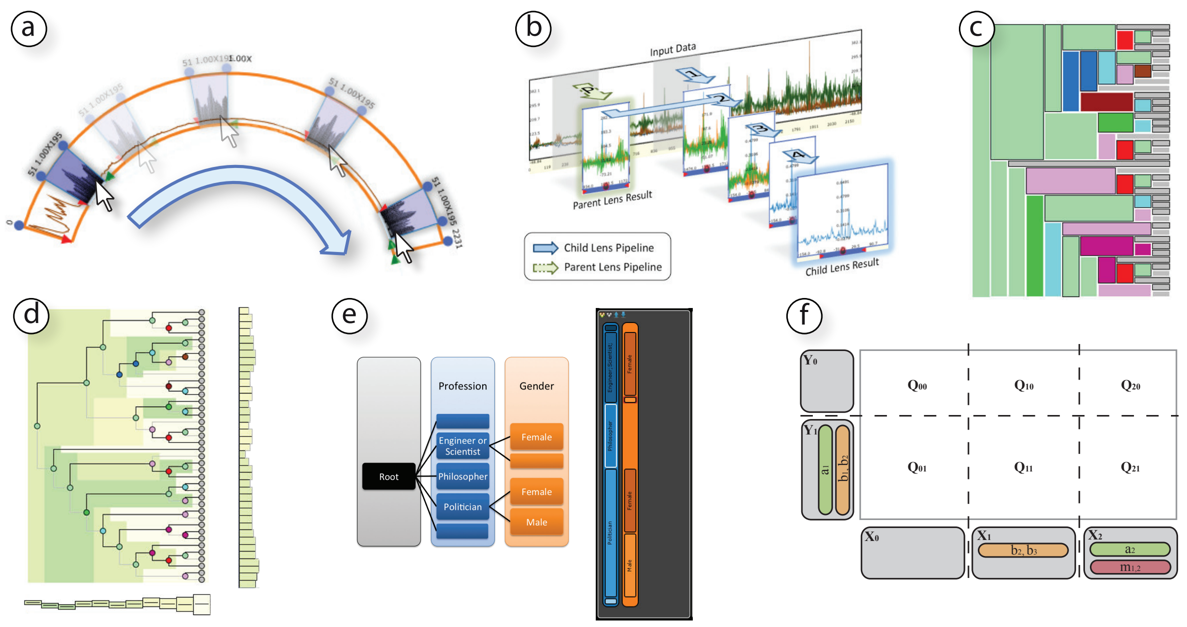


Figure 1.4: A visual index of generalizable visualization techniques and frameworks developed during the design studies: (a) the MagicAnalytics Lens technique in KronoMiner, (b) the lens visualization framework in ChronoLenses, (c) the icicle-dendrogram representation of trees in DAViewer, (d) two compact tree representations (and the original node-link tree) in DAViewer, (e) the dynamic filtering tree in TimeSlice, and (f) the faceted data visual query framework in PivtoSlice. Details are described in the corresponding design study chapters.

(d) TimeSlice contributes a visualization technique for browsing and comparing multi-faceted events timelines via manipulations of visual queries organized in a dynamic tree (Chapter 6), and

(e) PivotSlice addresses the exploration of implicit and explicit relationships in multi-faceted datasets by allowing users to perform visual queries that subdivide the data with customized logics (Chapter 7).

Secondly, a number of generalizable visualization techniques and frameworks are also proposed and explored in the design studies, including:

(a) an interactive visualization technique named MagicAnalytics Lens that dynamically displays correlation plots of time-series in-place on the overlapping region of two visual elements (Section 3.4.3),

(b) a lens-based visualization framework for time-series analysis that enables both single- and cross-data stream computations as well as multi-step transformations by connecting lenses in a hierarchy (Section 4.3),

(c) a novel visual representation of trees that combines the benefits of the dendrogram[4] and the icicle plot [Kruskal and Landwehr, 1983] (Section 5.4.1),

(d) two compact visual representations of trees that summarize their key structural characteristics from the vertical and horizontal perspectives (Section 5.4.2),

(e) a generic visual query[5] structure, called filtering tree, for interactive exploration of faceted temporal events by semantically organizing filters in a hierarchy (Section 6.3), and

(f) a visual query language framework for mining relationships in multi-faceted datasets via a dynamic tabular subdivision of the overall information space (Section 7.2).

Thirdly, the design requirements and guidelines that we derived from the case studies of specific application domains consist of:

(a) design guidelines for multi-purpose time-series visualization (Section 3.2.1),

(b) a design space of multivariate time-series layout (Section 3.2.2),

(c) tasks and design requirements for elaborate exploratory analysis of time-series data (Section 4.2),

(d) design requirements for supporting the research workflow of developing discourse parsers in Natural Language Processing (Section 5.3.2),

(e) design guidelines for effective faceted browsing and querying of multi-attributed datasets (Section 6.2), and

(f) an information seeking mantra for the visual exploration of multi-faceted datasets (Section 7.3.2).

Finally, as a summarization and reflection of all the design studies, we contribute a number of high-level guidelines and theories, providing implications for the future design and development of effective multi-focus visualization systems. Specifically, the contributions include:

(a) a set of four high-level design considerations for developing efficient multi-focus visualizations (Section 8.1),

(b) a multi-focus information visualization pipeline revealing conceptual stages from raw data to final visual representations (Section 8.3.1),

(c) a model of knowledge discovery process in multi-focus visual systems reflecting the relationships among the data, the visualization, and the user (Section 8.3.2), and

[4]A dendrogram is a visual representation of tree structures often used for showing hierarchical clustering, which arranges tree nodes aligned from the lowest level (i.e., the leaves). An example is shown in Figure 5.3-b.

[5]The term "visual query" here is an interactive and interative process in which users manipulate the visualization to perform traditional "database queries" and select data items for visual representations.

(d) an in-depth thinking of multi-focus visual exploration in light of the theory of processing of perceptual structure [Garner, 1974] (Section 8.4).

1.5 Overview of the Dissertation

This dissertation consists of five parts, founded by a series of novel interactive visualization techniques for multi-focus visual exploration with many datasets in a variety of application domains. Following the introduction and background chapters in Part I, the core of this dissertation — five design studies — is divided into three parts (Part II–IV), each focusing on the exploration of a specific data feature, i.e., data values, structures and attributes, as Ware [2004]'s classification discussed in Section 1.2.2. Lastly, Part V addresses high-level design considerations and guidelines of multi-focus visualization techniques, and concludes the dissertation with a summary and future research directions.

1.5.1 On Visual Data Exploration

Following this motivation and introduction chapter, Chapter 2 in Part I reviews the literature related to the research of this dissertation. In that chapter, we first present an overview of the state of the art in information visualization by describing some key visual representation techniques and interaction paradigms of visual exploration. We then discuss several design theories and guidelines widely applied in many visual systems as well as different evaluation methods for information visualization, which both have been influential on the design studies of this dissertation.

1.5.2 Focusing on Data Values

Part II reviews two examples of interactive visualizations facilitating multi-focus visual exploration of data values. We choose time-series analysis as the application domain, because time-series is ubiquitous across many domains and its numerical values are the most critical feature concerned by analysts in practice, for example, in financial analysis, signal processing, weather measurements, and so forth.

Chapter 3 introduces KronoMiner, a multi-purpose time-series exploration tool based on a radial hierarchical layout, allowing users to drill down, align, and compare time-series segments with flexible and straightforward interactions. Chapter 4 outlines a visual interface called ChronoLenses for the exploratory visual analysis of time-series that often contains more elaborate tasks such as visualizing and reusing derived data values. ChronoLenses offer analysts with effective means of interactively constructing customized analytical pipelines through a series of lens-based visual components with different data transformations.

1.5.3 Focusing on Data Structures

Part III presents an interactive visualization system for computational linguists to compare discourse trees, a widely-used representation of the rhetorical structure of documents in Natural Languages Processing, enabling efficient discovery of intuitions and insights for the development of robust discourse parsers (Chapter 5). The proposed technique, DAViewer, although designed particularly for NLP researchers, addresses many general problems in exploring and comparing hierarchical data structures.

1.5.4 Focusing on Data Attributes

Part IV introduces two design studies of developing multi-focus visualization techniques to support the exploration of data with multiple heterogeneous attributes (that form meta-data facets), concerning the needs of dynamically and visually querying, browsing, and comparing data items in multidimensional datasets.

Chapter 6 focuses on multi-faceted temporal events, and presents TimeSlice that allows users to flexibly construct, compare, and manipulate multiple visual queries of data attributes with a dynamic interface called filtering tree. Chapter 7 further promotes and extends the filtering tree idea for the exploration of multidimensional networks. The proposed visualization, PivotSlice, is based on a generalizable visual query language framework for subdividing the entire dataset with customized semantics, revealing various kinds of relationships and trends implied by data attributes.

1.5.5 Closing

In this final part (Part V), we provide critical reflections on designing effective multi-focus visualization techniques and conclude this dissertation with some summaries.

Chapter 8 reviews common fundamental characteristics of all the design studies. By generalizing the experiences obtained from the studies, we present design considerations, theoretical models, and visualization concepts, to inform researchers and practitioners in Information Visualization for devising useful and usable multi-focus visual interfaces. Chapter 9 summarizes the contributions of this dissertation, discusses several limitations of the studies, and outlines several promising directions for future investigation of multi-focus interaction and visualization techniques.

Chapter 2

BACKGROUND

If I have seen further than others, it is by standing upon the shoulders of giants.

Isaac Newton

As introduced in Section 1.2, this dissertation is related to the research of three intersecting fields — information visualization, visual analytics, and human-computer interaction. Based on our primary goals of designing novel interactive visualization techniques, this chapter reviews the literature with the focus on information visualization and its overlapping areas with the other two bodies of research. Where appropriate, additional area-specific and more detailed related work will be discussed in the design study chapters (Chapter 3–7).

The origin of information visualization derives from data graphics for analysis, including maps, diagrams, measurements, and statistical plots [Friendly, 2008]. Among the earliest works of using abstract visual primitives to represent data, Tufte [1983, 1990, 1997] developed a theory of data graphics with the emphasis on maximizing useful information density and a number of graphics design principles. In data statistics, the concept of *exploratory data analysis*, promoted in a book by Tukey [1977], encourages the use of visual methods to provide rapid statistical insights of datasets. In addition to static pictures, dynamic changes and animations are proposed in presenting statistical data [Cleveland and McGill, 1988]. This ability of interaction is the largest distinction between information visualization concerned in this dissertation and the traditional data graphics. In particular, interactive visualization allows human perceptual guidance for manipulating visual representations, thus better supporting the exploration process in which the user gathers information for some purposes, makes sense of it by constructing a representational framework, and then packages it into some form for communication or action [Card et al., 1999].

Following the above theme, in this chapter, we first introduce the "static" component of information visualization, a variety of visual representation techniques classified based on the types of data to visualize (Section 2.1). Then, its "dynamic" aspect — interaction — is discussed, by summarizing the techniques with two main visual exploration paradigms (Section 2.2). Most of the visual representation and interaction techniques are shared in both information visualization and visual analytics communities. Moving forward, we introduce the concept and definition of multi-focus visual exploration, which is the central theme of this dissertation, by leveraging the existing visual representation techniques and interaction paradigms (Section 2.4). Further, we present the design theories and principles that have influenced the research of this dissertation and different methodologies for evaluating interactive visualization (Section 2.3), in which many aspects have roots from the perception, cognition, interface design research in human-computer interaction.

2.1 Visual Representation Techniques

In this section, we review main visual representation techniques closely related to this dissertation in previous work, according to the formats of data to visualize. We introduce the literature based on Shneiderman [1996]'s classification of data in information visualization (Section 1.2.2).

2.1.1 Linear and Temporal Data

A large number of datasets in the world are inherently stored and processed in a linear format, of which time-oriented data is an example that is vital in many application domains and of greatest interests by analysts. Thus, we here focus on discussing visualization techniques that are designed for temporal data. While some literature tends to separate time-series from linear data, these techniques are also applied to any dataset containing items ordered or aligned linearly, even without a temporal dimension. Moreover, time-oriented data can also be multivariate, i.e., consisting of data points with multiple attributes. To distinguish from multidimensional data discussed in the next section, we here focus on data that has a "linear nature", i.e., containing one special dimension such as time, and all other data attributes can be viewed as dependent variables with respect to this particular dimension.

Visual representations of time-series date back to the 18th century with seminal work using line charts by Playfair [Tufte, 1983]. A large body of techniques has been proposed for visualizing temporal data. See [Aigner et al., 2007, 2008; Müller and Schumann, 2003] for relevant surveys. A comprehensive visual index of all the time-oriented data visualizations can be found in the TimeViz Browser[1] tool created by Tominski and Aigner. Most of the work on time-series visualization has focused on the scalability of the more conventional line plot and bar chart representations, either by proposing variations on the original plotting techniques or by enhancing them with advanced interactive visual filtering techniques. For example, CloudLines represents large temporal events in a line of circles with multiple levels of aggregation [Krstajic et al., 2011]. Multi-focus and multi-scale views, discussed later in this chapter, are often incorporated to visualize large-scale time-series with high sampling rates, such as StackZoom [Javed and Elmqvist, 2010], SignalLens [Kincaid, 2010], and LiveRAC [McLachlan et al., 2008]. To express lengths of temporal events, i.e., time intervals, LifeLines visualizes activities of a person's life as rectangular bars anchored along a timeline [Plaisant et al., 1996]. Other innovative variations on the conventional line plots include small multiples and sparklines [Tufte, 1983], horizon graphs [Heer et al., 2009], braided graphs [Javed et al., 2010], and glyphs like music notes [Xie et al., 2014].

In addition to linearity, dependency or branching characteristic of temporal data is another important factor to be considered for visualization design. For example, in project management, PlanningLines uses glyphs to encode critical steps of a project, such as the duration, earliest starting times, and latest finishing times of a project stage [Aigner et al., 2005]. LifeFlow [Wongsuphasawat et al., 2011] and OutFlow [Wongsuphasawat and Gotz, 2012] summarize transitional patterns of large numbers of patients between different departments in a hospital.

When time-oriented data is multivariate, a common approach is to stack properties values along a primary time-axis, such as ThemeRiver [Havre et al., 2002]. Other methods include displaying attributes on multiple axes while emphasizing the time dimension, such as MultiComb and TimeWheel [Tominski et al., 2004]. Dynamic timeline querying based on data properties has also been explored by researchers, such as Progress Multiples [Phan et al., 2007]. DecisionFlow, on the other hand, leverages progressive user-defined queries to discover insights in high-dimensional temporal events [Gotz and Stavropoulos, 2014].

Further, many visualization techniques have been proposed to facilitate the identification of periodic patterns in time-oriented data. For instance, SpiralGraph [Weber et al., 2001] and SpiralView [Bertini et al., 2007] use spiral shaped time-axis to reveal cycles in time-series. CalendarView, on the other hand, employs an aggregation approach to visualize univariate time-series on a calendar on a daily, weekly or monthly basis [Van Wijk and Van Selow, 1999]. Lopez-Hernandez et al. [2010] addressed the same problem by proposing a "drawer" visual metaphor to sort and cluster periodical digital signals. In addition, some other work focuses on discovering more generic patterns in time-series. For example, PatternFinder [Fails et al.,

[1]http://survey.timeviz.net/.

2006] and TimeSearcher [Buono et al., 2005] assist users in detecting customized temporal event patterns in medical records. Along this line, Monroe et al. [2013a,b] explored challenges for interactive temporal pattern query and proposed several visual encodings for simplifying time-series events.

2.1.2 Multidimensional and Multi-Faceted Data

Multidimensional or multi-faceted data is another ubiquitous data format to analyze in real-world application domains. Unlike the multivariate time-series discussed above, where people pay special attention to the time-axis, users usually have equal interests on all the data dimensions. There is no clear distinction between multidimensional and multi-faceted data, although the concept of facets is often categorical and multidimensional data is usually more general which may include continuous numerical dimensions. Real-world data is usually more complicated, containing heterogeneous attributes, i.e., numerical, ordinal, and categorical, therefore in this dissertation, we unify the concepts of these two data types, discussing visualization techniques in a wider scope.

Multidimensional data has brought many challenges to visualization because of its high data dimensionality. See [Wong and Bergeron, 1997] for an early survey. A widely-accepted mental model for multidimensional data is the *data cube* metaphor. For example, consider a 3-dimensional business dataset with time, location, and product as the facets; and the data items can be placed in a 3-dimensional physical space defined by three axes, each representing a facet. However, when the number of dimensions becomes lager, the mapping between the data dimensions to the physical dimensions are impossible or very difficult to comprehend.

One stream of research aims to simultaneously display data attributes across all the dimensions. Parallel Coordinates Plot is a well-known technique for showing multi-dimensional datasets where each data item is represented by a polygonal line that intersects parallel vertical axes [Inselberg and Dimsdale, 1990]. Likewise, Star Coordinates uses a circular layout of attribute axes to determine the position of a data point in the 2D space [Kandogan, 2000], and Claessen and van Wijk [2011] promoted the axis-based idea by allowing users freely manipulate and position the axes. In a different approach, Scatterplot Matrix organizes a collection of scatterplots in a tabular format, displaying all data points across every pair of attributes [Hartigan, 1975]. Im et al. [2013] extended Scatterplot Matrix to a more general form for visualizing datasets with heterogeneous axes. Similar approaches have also been proposed in other fields, such as Prosection Matrix for showing a section of the data space across projections [Furnas and Buja, 1994], and HyperSlice for representing a multivariate scalar function [van Wijk and van Liere, 1993]. However, when the data dimensionality becomes really high, dimension reduction techniques, such as Principal Component Analysis (PCA) [Pearson, 1901], are usually applied first before feeding into the visualization.

Another body of research focuses on leveraging interaction techniques to allow users progressively explore multidimensional data spaces, i.e., interactive faceted browsing. Several early systems, including Interactive Spreadsheet [Chi et al., 1997], FOCUS [Spenke et al., 1996], and InfoZoom [Spenke and Beilken, 2000], allow users to perform dynamic queries by arranging filtered data objects in a large table where the attributes on the vertical facets axis can be pivoted, expanded, and collapsed. Yee et al. [2003] proposed an image searching tool that allows the user to progressively narrow down the query using a faceted meta-data panel displaying item distributions. Several techniques have been proposed to visualize the correlations and trends among faceted data items. Polaris [Stolte et al., 2002] and its datacube concept [Stolte et al., 2003], which later became Tableau [2], provides a tabular representation of relational databases where the columns and rows form a set of dynamic queries with ordinal or numerical facets to divide the whole dataset. Similarly, Ploceus presents connections of data items in node-link graphs by slicing the visualization into a matrix view of subnetworks along user-selected attributes [Liu et al., 2011]. PaperLens [Lee et al., 2005] and FacetLens [Lee et al., 2009] support the exploration of trends and connections for large academic repositories based on multi-faceted queries over properties of publications. Focusing on similar topics, Jigsaw provides a set of rich analytical capabilities for exploring multi-faceted academic databases through multiple views [Gorg et al., 2013]. Recently, Gratzl et al. [2014] proposed a flexible and interactive method to build various types of charts for different subsets of data and systemically connect them to show relationships.

2.1.3 Hierarchical Data and Trees

Hierarchical (tree structure) data is widely studied in many application areas including bioinformatics (e.g., genetic trees), faceted classifications (e.g., organization hierarchies), and software development (e.g., dependency trees). A large body of research has been conducted in information visualization to present hierarchical data, including representations of a single tree structure and visual comparisons of multiple trees. Jürgensmann and Schulz provided a visual survey of a collection of tree visualization techniques[3].

For visualizing a single tree, an intuitive representation is the traditional node-link diagram. There are also several attempts to extend the 2D node-link diagram to radial layout [Carrire and Kazman, 1995; Nguyen and Huang, 2002], hyperbolic space [Lamping and Rao, 1996], and 3D

[2]Tableau Software, http://www.tableausoftware.com/.

[3]http://www.informatik.uni-rostock.de/~hs162/treeposter/poster.html.

space [Robertson et al., 1991]. A second visual representation uses a nested layout indicating the parent-child relationship by positioning child nodes within the boundaries of parent nodes, such as treemap [Johnson and Shneiderman, 1991] and hierarchical circle packing [Wang et al., 2006]. Based on the treemap, a more flexible space filling technique for showing hierarchies is proposed by Duarte et al. [2014]. Apart from placing children inside the parent, a third approach employs an adjacency layout where children are presented next to their parent node, such as icicle plot [Kruskal and Landwehr, 1983] and beam trees [van Ham and van Wijk, 2003]. Sunburst proposes a similar idea to using a radial layout [Stasko and Zhang, 2000]. Moreover, indented list is a popular mean of representing tree structures that are often used in current GUI for showing folder hierarchy. Lastly, trees can be represented in an adjacency matrix format where cells indicate connections between nodes; though it is space efficient, the hierarchical relationship is not very obvious.

Moreover, a breadth of techniques has been proposed for the visual comparison of multiple trees, that can be classified under three main categories: side-by-side views, merged views, and animation. Graham and Kennedy [2010]'s survey provides an exhaustive review. Side-by-side views can leverage visual cues to convey the relationships, such as drawing explicit links between the matched nodes [Holten and van Wijk, 2008]. Also, dynamic queries can be added to allow users to interactively visualize similar subtree structures by highlighting a pattern, such as in [Munzner et al., 2003] and [Bremm et al., 2011]. Second, merged views combine two trees in a single visualization that encodes the differences. For example, Union Tree integrates two treemap representations into one unified view [Tu and Shen, 2007], and Candid Tree extends the traditional node-link view by encoding structural differences between two trees [Lee et al., 2007]. Similarly, van Ham [2003] employed a similarity matrix based approach where the nodes of two trees are placed in the rows and columns. Finally, a third approach is to use animations to convey changes between two different trees [Robertson et al., 2002]. However, users may lose track of the overall difference since the positions of many nodes are varying during the animation [Graham and Kennedy, 2010].

2.1.4 Network Data and Graphs

Network or graph data visualization exists in a wide range of real-world applications. For example, graphs we typically interact with on a daily basis includes social networks, organization diagrams, and web-site maps; in biology and chemistry, network data is applied in representing relations between species, molecular maps, and so on; and other areas in engineering and science can have language semantic networks, knowledge-representation diagrams, and circuit layouts. In general, there are two main types of representations for network data: node-link diagrams and matrix views.

Node-link diagram is an intuitive approach for visualizing a graph, where the placement of nodes is critical for users to obtain an appropriate mental image and conduct efficient exploration of data. Several graph drawing techniques have been proposed, for example, the force-directed layout by simulating attractive force between nodes [Eades, 1984; Fruchterman and Reingold, 1991], multi-scale approaches by refining node positions from coarse to fine levels [Hadany and Harel, 1999; Harel and Koren, 2000], and circular layout by using crisscrossing lines through centers of circles [Huffaker et al., 1999; Yee et al., 2001]. However, a major limitation of the node-link representation is visual clutter of nodes or edges, especially when the network is very large. To address this issue, a number of techniques for grouping edges and nodes have been proposed [Holten, 2006; Holten and van Wijk, 2009; Wattenberg, 2006]. Also, dynamically slicing the entire graph into multiple sub-graphs according to node attributes or topologies can be effective for users to explore large and complex networks, such as in Semantic Substrates [Shneiderman and Aris, 2006] and ManyNets [Freire et al., 2010]. A third approach is to progressively expose sub-regions of the graph based on specific nodes of interests through user interactions [Chau et al., 2011; Dork et al., 2012; van Ham and Perer, 2009]. Those techniques could also be integrated with focus+context views (e.g., Hadlak et al. [2011]) that will be discussed later. On the other hand, van den Elzen and van Wijk [2014] employed an opposite approach to allow users to navigate data from detail to overview using dynamic selections and aggregations.

Matrix view, also known as the graph adjacency matrix first introduced by Bertin [1983] and popularized by Becker et al. [1995], is another essential technique of showing networks, in which each cell of the matrix is used to indicate whether the corresponding nodes represented by the row and column are connected or not. While the matrix representation has no occlusions and is space-efficient, the topology of a graph and inter-connections of nodes are not obvious. Similar in the aspect of layout, one main algorithmic challenge for matrix views is the ordering of rows and columns [Díaz et al., 2002], because certain sub-network patterns (e.g., cliques and communities) may only be revealed when nodes are displayed in a particular order. Combining other visual encoding methods with this matrix or tabular approach may offer more flexibility for users to explore and analyze data. For example, Brandes and Nick [2011] used a matrix of glyphs to show evolving patterns of node relationships in social networks. Moreover, hybrid approaches combining node-link diagrams and matrix representations for dense subgraphs is proposed in MatrixExplorer [Henry and Fekete, 2006] and NodeTrix [Henry et al., 2007]. In the cases of visualizing dynamic networks, i.e., nodes and links evolving along time, the matrix approach can be extended by combining and aggregating multiple adjacency matrices together [Hlawatsch et al., 2014].

There are also several other approaches to represent networks. For example, CiteVis uses the color density of a node to indicate the number of links connected to that node, and further employs user interactions to reveal the node neighbors in the view [Stasko et al., 2013]. Freire et al. [2010] employed a table of charts, such as bars and histograms, to present different graph metrics, such as edge counts and node degrees, over subnetworks segmented from the entire graph. On the other hand, GraphPrism leverages multiple heatmaps to summarize the metrics of individual nodes in a large graph [Kairam et al., 2012].

2.2 Visual Exploration Paradigms

Interaction on visualizations, which allows users to dynamically change visual representations of data according to their exploration objectives, is an essential part of knowledge discovery and decision making. In this section, we review two major interaction paradigms based on the visualization layouts — separated views and distortion views — which are the foundation of the multi-focus visual exploration paradigm concerned in this dissertation. It is worthy to note that these interaction paradigms are independent of visualization techniques and are also widely used for interface design in HCI [Cockburn et al., 2009]. Here, we focus on the application of such interaction paradigms in visualization systems.

2.2.1 Separated Views

Separated views have the benefits to combine different visualization methods to overcome the shortcomings of particular visualization techniques in single views. A common research problem for separated views is design techniques to effectively coordinate different visualizations in all the views. Dynamic brushing and linking techniques, i.e., interactive changes made in one visualization are automatically reflected in the other visualizations [Keim, 2002], are often utilized to indicate corresponding data items in other views while users are interacting within one view. Basically, there are three types of view separations, including overview+detail (i.e., spacial separation), lens-based interfaces (i.e., z-separation), and zooming (i.e., temporal separation). Some researchers have also referred such paradigm to multiple views, especially in the cases of spacial separation and z-separation. To include zooming interfaces that involve temporally separated actions, we use the more general term: separated views.

First, the overview+detail schema is widely applied in many commercial applications, such as Adobe Reader[4] and Microsoft Powerpoint[5], for showing document outlines as well as the part users are working on. Usually, interactions on the detailed view are reflected immediately in the overview with some visual indicator indicating the currently focused region. However, interaction on the overview is loosely coupled to the detailed view. Unless a user moves the visual indicator on the overview, the detailed view does not update, which provides the ability of exploring neighborhood region of the focus without interfering current detailed view window. This kind of overview+detail design has been applied in many visualization systems, such as Continuum for exploring event timelines [André et al., 2007] and GraphDice for showing multidimensional networks [Bezerianos et al., 2010]. In addition, this kind of overview+detail schema can be extended to multiple layers including a series of views at different scales. For example, StackZoom [Javed and Elmqvist, 2010] and PolyZoom [Javed et al., 2012] employ multi-level and multi-scale view windows to visualize large information spaces, which allows a user to progressively drill down the data.

Second, the lens-based approach aims to place views in different layers, which is first proposed as the concept of Toolglass and Magic Lens by [Bier et al., 1993]. Distortion lenses, which blend the context and the focus in a unified presentation [Carpendale and Montagnese, 2001; Carpendale et al., 2004], are sometimes also included when referring to lens-based interfaces. In this dissertation, we differentiate the layered z-separation of views from those distortion techniques that will be discussed in the next section. Interactions based on this lens metaphor usually involve moving the focus area around the overall data space and applying various of transformations to the data below. For example, DragMag allows a user to create multiple zoomed windows layered on top of an image [Ware and Lewis, 1995], and Hadlak et al. [2011] designed an in-situ visualization for networks, allowing users to choose different graph representation techniques for selected sub-network regions. Moreover, some approaches adjust the global visualization according to characteristics of the content below the lens, such as ColorLens that changes the overall luminance of an image when a user explores different local parts of the image [Elmqvist et al., 2010a].

Third, zooming is commonly seen in many of nowadays computer interfaces, allowing users to navigate between the contextual and the focused content by temporal separation. The operations of zooming usually involve using a specific widget or the mouse wheel, and then users interact with the zoomed view using scrollbars or mouse panning. Theoretical studies about zoomable interfaces have been conducted in the past, including the space scale diagram that conceptualizes zoom and pan interactions independent with visual representations [Furnas

[4]https://acrobat.adobe.com/ca/en/products/pdf-reader.html.
[5]http://products.office.com/en-us/powerpoint.

and Bederson, 1995], and the mathematical formulation of a smooth transition between any two positions in this scale space via panning and zooming [van Wijk and Nuij, 2004]. Particularly in visualization, zooming means more than just simply scaling. Semantic zooming, first proposed in the Pad system [Bederson, 1996; Perlin and Fox, 1993], is a widely used technique in visual systems. In contrast to ordinary graphical zoom, semantic zooming also adapts different visual representations when scaling the data space. For example, LiveRAC visualizes large collections of time-series data in a compact tabular view that displays data with various levels of detail in different cells, including color-coded rectangles, coarse line charts without grids, and detailed line charts [McLachlan et al., 2008].

2.2.2 Distorted Views

Distorted views, also called focus+context techniques, aim to seamlessly blend the currently interesting region (i.e., focus) in greater detail and its surrounding content (i.e., context) into a single coherent display through distortions. Such techniques can decrease a user's short-term memory load, and thus improve the abilities of comprehending and manipulating data in large information spaces.

A few theoretical frameworks have been proposed in the past regarding how to integrate information in the focus and the context into a single representation. The first visualization with such concept is the bifocal display, described by Spence and Apperley [1982], which uses a metaphor of a paper stretched with four rollers. Later, Furnas [1986] established the theoretical foundation of focus+context interfaces, called general fisheyes which present objects in information spaces based on a degree of interest (DOI) function. The DOI function addresses what information should be displayed in the focus, whereas most of the subsequent research is focusing on how data should be distorted in the views. By extending the fisheye views, Sarkar and Brown [1994] developed several geometric transformations that can produce continuous view distortions according to the distances between a user's focal point and the coordinates of a data item. Further, Carpendale and her colleagues described a framework for unifying the presentation space based on geometric transformations and presented a set of new distortion functions that can effectively integrate high magnifications of data focus with its context [Carpendale and Montagnese, 2001; Carpendale et al., 2004].

In addition to theories, this distorted view approach has been continuously followed in many visualization system designs to solve problems in real-world applications. For example, some of the earlier systems, including Perspective Wall [Mackinlay et al., 1991] and DocumentLens [Robertson and Mackinlay, 1993], use perspective projection to distort contextual views shown around the focus region, but there are still distinct borders between the focus and the context. On the other hand, some techniques aim to produce smooth distortions, using the metaphor of magnification lenses. For instance, SignalLens applies such approach in supporting efficient analysis of long electronic signal measurements [Kincaid, 2010], and Hadlak et al. [2011] used semantic lenses to embed and integrate different forms of visualizations for graphs. However, it is usually difficult to acquire targets with a distorted lens at a relative high zoom level [Gutwin, 2002]. To address this issue, Pietriga and Appert [2008] explored the design space of lens-based visualization techniques by investigating different aspects such as the time and the content translucence to achieve more efficient lens-movement interactions and user experience.

2.3 Design Influences

The design of efficient visualization systems is not only based on the practices of experimental evidence but also grounded by many well-justified theories and design guidelines. In this section, we explore the work of important theories that have influenced the design of visualization techniques proposed in this dissertation. We also briefly discuss various methodologies for evaluating information visualization techniques, some of which have been adopted in our design studies to assess the effectiveness and usefulness of our approaches in solving real-world problems.

2.3.1 Information Visualization Mantra

Shneiderman [1996]'s information seeking mantra, "*overview first, zoom and filter, detail on demand*", describes a typical visualization usage pattern for exploring data, which provides the basic guidelines for designing effective information visualization systems. More particularly, the system first needs to present an overview of the dataset, where a user can further drill down and define regions of interests via zooming and filtering. As discussed in Section 2.2, zooming of the information space can be geometric (i.e., using consistent visual representation) or semantic (i.e., changing data representation accordingly); filtering can leverage separated view (i.e., creating an overview+detail schema or layered detailed views) or distorted view (i.e., emphasizing content on a focal region with its context). In addition to the mantra, Shneiderman [1996] offers three important visualization tasks, including relate — viewing relationships among items, history — keeping a history of actions to support undo, and extract — allowing extraction of sub-collections and of the query parameters.

Later, Keim et al. [2008] extended this information seeking mantra to the field of visual analytics: "*analyze first, show the important, zoom, filter and analyze further, details on*

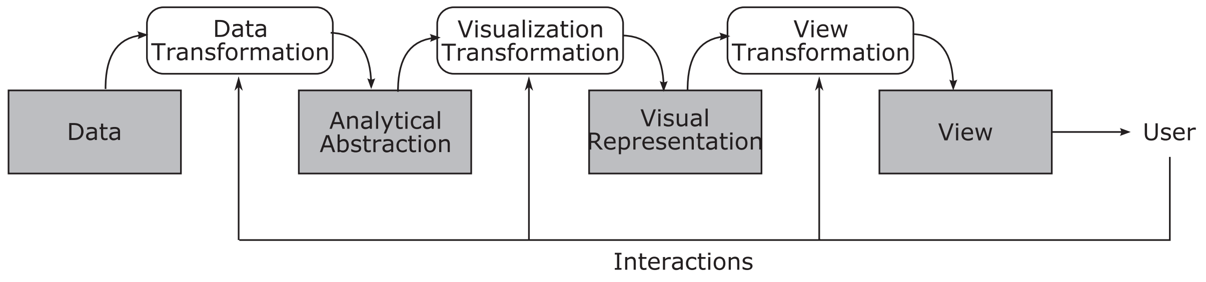


Figure 2.1: Information visualization pipeline, adapted from [Card et al., 1999].

demand". First, this mantra calls for seamless combinations of analytical approaches together with advanced visualization techniques, for example, analysis procedures should be conducted before feeding data into the visualization thus revealing important features and insights for guiding a user to proceed the exploration (i.e., analyze first and show the important). Second, it suggests that a data sense-making process is iterative (i.e., analyze further) so that an efficient visualization system should tightly integrate visual representations, that support human judgment, and interactions, that re-parameterize the visualization and analytical algorithms.

Also, van Ham and Perer [2009] proposed "*search, show context, expand on demand* that is suitable for the cases where users have semi-specific information needs and an overview of the entire database is less achievable due to the huge data amount. Similarly, the philosophy of "*start with what you know, then grow*" has also been suggested when users have some initial familiarity of the dataset [Heer and Boyd, 2005].

2.3.2 Visualization Pipelines and Models

Chi and Riedl [1998] proposed the data state reference model of visualization, which shows a series of logical data stages and operations between two successive steps in a visualization system, forming a pipeline that transforms raw data into final on-screen representations. As indicated in Figure 2.1, Card et al. [1999] also presented this model without the separation of system control (i.e., *data* and *analytical abstraction*) and user control (i.e., *visual representation* and *view*). This was further extended by Carpendale [1999] to include the presentation space — one more *visual presentation* step after a *presentation transformation* from the *visual representation* stage, and indications of user interactions on each step, as the upward arrows shown in Figure 2.1. Additionally, Collins [2010] added the interaction possibility of access/edit/annotate source data.

On a different perspective, Keim et al. [2008] offered a model of knowledge discovery process in visual analytics based on the simple model of visualization by van Wijk [2005]. This

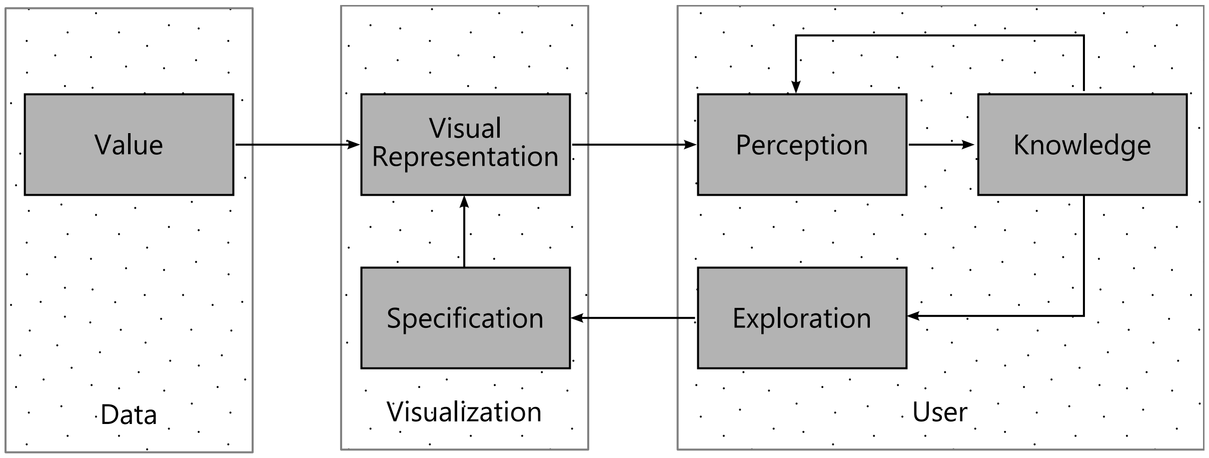


Figure 2.2: Sense-making loop for visual analytics, adapted from [Keim et al., 2008].

model presents the internal stages of the three important parties — data, visualization, and user — and their external relationships in the sense-making loop of using visual analytics systems. As shown in Figure 2.2, after some initial statistical and mathematical transformations for raw data (i.e., value to visual representation), the analytical process enters a loop where the user can gain better knowledge through driving the system towards discovering more insights in data exploration, by specifying parameters to modify visual representations with interactions. Further, Russell et al. [1993] described a model which concurs with this sense-making loop, suggesting that interfaces should guide users to progressively search for a representation and encoding of data to answer their specific questions. Along these lines, Sacha et al. [2014] proposed a knowledge generation model for visual analytics that contains three loops: exploration loop, verification loop, and knowledge generation loop, which encompass different perspectives at different levels.

In addition to leveraging those theoretical models to guide the design of our techniques, we further developed multi-focus versions of the visualization pipeline and the sense-making loop, that have not been noticed in previous work, based on our practice in the design studies, thus offering theoretical foundations for the creation of multi-focus interactive visualization systems (Chapter 8).

2.3.3 Graphics Design

There exist a number of influential theories rooted in graphics design which play significant roles for designing information visualization systems. Bertin [1983] proposed a selection of graphical dimensions, as visual variables, for encoding information with the elementary marks of graphics such as points, lines, areas, surfaces, and volumes. As shown in Figure 2.3, the

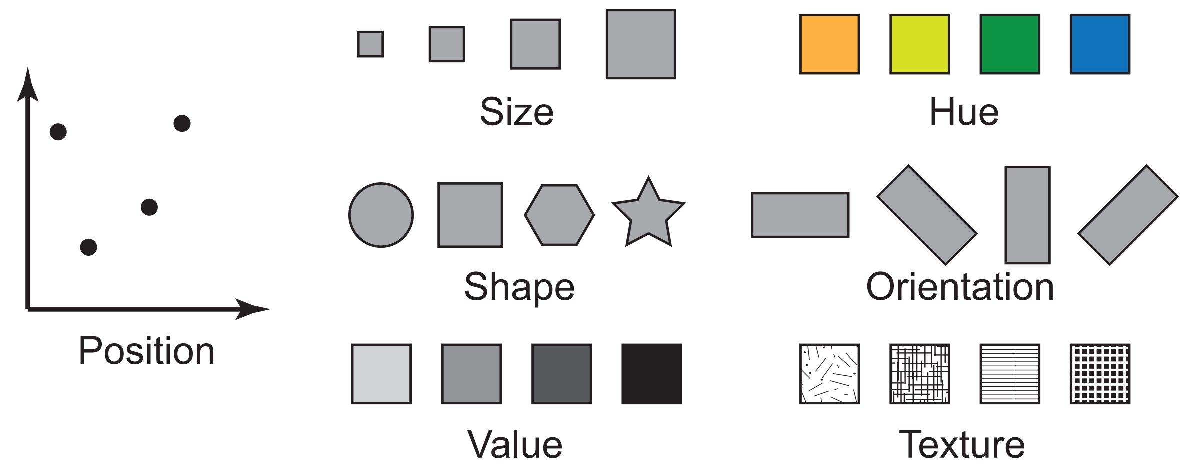


Figure 2.3: Visual variables.

visual variables are: position, size, shape, value, hue, orientation, and texture, which are later extended by Carpendale [2003] to add motion that is especially important for interactive visualization. Each of these visual variables can be characterized by five different features, including:

- Selective – Is it easy to select a changed mark from unchanged ones?
- Associative – Can marks be grouped with the presence of different values?
- Quantitative – Can a numerical value be read from this variable?
- Order – Do changes in this variable support ordered readings?
- Length – How many values can be distinguished and associated in adequate clarity?

Bertin [1983] and Carpendale [2003] provided detailed analysis and usage suggestions of these visual variables based on their levels of expressiveness in these features, which help designers select appropriate visual variables to encode different properties and dimensions of data.

However, one cannot treat all the visual variables (or channels) completely independently when mapping data values to them. Some of those visual variables have dependencies and interactions [Munzner, 2014; Ware, 2004]. The theory of processing of perceptual structure [Garner, 1974] states that the attributes of objects in multidimensional spaces can receive different dominant perceptions, *integral* or *separable*. That is, the way in which the dimensions of the spaces combine perceptually, can affect how human perceives the object. Some attributes of visual objects are viewed perceptually as a whole (integral), where as some remain more distinct and identifiable (separable). For example, lightness and saturation of a color are perceived integrally, while size and position of an object are perceived separably. Figure 2.4 shows some examples about how well two encoding channels can be separated.

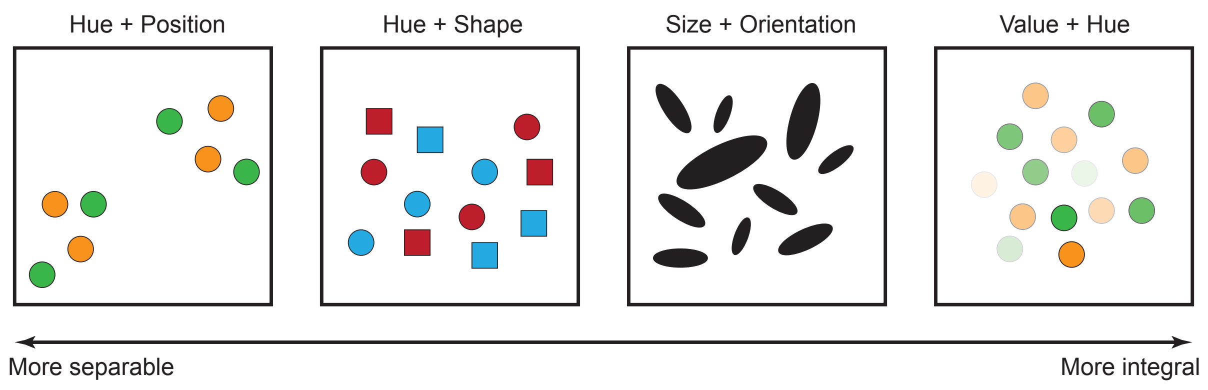


Figure 2.4: Integral and separable perceptual structures of pairs of visual variables.

Jacob et al. [1994] further extended this theory to characterize multidimensional input devices on multidimensional tasks. Along this line, various studies have been pursued by researchers in HCI community, for example, topics about bi-manual manipulation [Leganchuk et al., 1998] and coordination of multiple degrees-of-freedom object manipulation [Zhai and Milgram, 1998].

Another critical principle for graphics design is the Gestalt Theory, founded by Max Wertheimer and popularized by a number of German psychologists in 1920s [King and Wertheimer, 2005]. The central idea of Gestalt psychology is that human mind forms a "unified whole" (i.e., "gestalt" in German) with self-organizing tendencies, which is described by laws in the following.

- Proximity – Objects close to each other are seen as a whole;
- Similarity – Similar objects tend to be grouped together;
- Connectedness – Connecting different objects by lines is a powerful way of expressing relationships between them;
- Continuity – Smooth and continuous lines are perceived before discontinuous ones;
- Closure – A closed contour tends to be seen as an object;
- Symmetry – Symmetrical object pairs are perceived more strongly than parallel pairs;
- Relative size – Smaller components of a pattern tend to be perceived as objects;
- Common fate – Objects moving in the same direction appear as a whole.

These Gestalt laws are presented by Ware [2004] as design principles that offer guidance for information visualization. Ware [2004] also reported other perceptual studies that highlight potential additional Gestalt laws, including perceived groupings based on transparency, perceived direction of movement based on vector illustration, and perceived contours from arrangement of shapes.

Finally, Tufte [1983, 1990, 1997]'s books offer a series of advice on creating clear and concise graphics design, such as avoiding uninformative elements, eliminating misleading visual encodings, and maximizing the data-ink to non-data-ink ratio. Information visualization designers should be aware of these suggestions and principles to best utilize the limited screen real-estate to show appropriate data. Generally, we strive to follow those advice in many places of our design studies.

2.3.4 Evaluation of Information Visualization

There are many known challenges in evaluating information visualization techniques. For example, it is difficult to obtain an appropriate sample of participants, such as domain experts; visual analysis tasks are often complex, exploratory, and ill-defined, such as discovering the unexpected insights, which requires a long term of study; and visualization prototypes being tested are usually incomplete, unlike the software that users are already familiar with, which may affect the visual reasoning process in an experiment. Some of other challenges are common to all the empirical research, such as difficulty to ask the right questions and pick the right focus. An overview of the main challenges is discussed in [Plaisant, 2004]. Carpendale [2008] provided a summarization of major approaches for evaluating information visualization, including laboratory experiments, expert interviews, long-term studies, and so on, most of which are also widely used in the field of HCI. In addition, several other methodologies have been proposed, such as insight-based evaluation [Saraiya et al., 2005], nested model for evaluation methods [Munzner, 2009], visual quality metrics [Bertini and Santucci, 2006], and measuring extrinsic effects such as worker productivity [McNee and Arnette, 2008]. Recently, Lam et al. [2012] discussed information visualization evaluations in seven different concrete scenarios based on study goals and types of research questions. In general, there are two main categories of techniques — quantitative methods and qualitative methods.

Quantitative Approaches

Quantitative evaluation methods are heavily drawn from the HCI community, which have the benefits of relative high precisions, high efficiencies, low cost and the high reliabilities of generalizing the information obtained to large populations. A major approach within the quantitative evaluations of information visualization is known as controlled experiment, which is usually used for comparing different visualization and interaction techniques. Controlled experiment usually involves the process of hypothesis development, user tasks design, independent variables control, observation, and dependent variables measurement. Another evaluation technique within quantitative approaches is usability inspection, which is established on the heuristic evaluation and cognitive walk-through in HCI [Mack and Nielsen, 1995]. As designed for interface testing, these methods are usually used in software engineering to identify usability issues of a prototype system. Zuk et al. [2006] provided a set of heuristics for evaluating information visualization; those approaches are often cheap, fast, and easy, thus appropriate for application during iterations of design. Further, Morse et al. [2000] developed a method via the creation of simple prototypes and task sets based on a visual taxonomy, which allows usability testing of the visualization in isolation from the rest of the system. Recently, many researchers leverage Amazon's Mechanical Turk platform[6] or something similar to conduct studies with a large number of users. But the user tasks that can be evaluated in this approach are usually simple and basic, for example, related to graphical perceptions [Heer and Bostock, 2010].

Many difficulties have been noticed for evaluating visualization systems under these quantitative approaches. In controlled laboratory experiments, it is difficult to tell if there is a true relationship between the independent variables, i.e., usually the visualization techniques, and the dependent variables, i.e., usually completion times and errors, because many factors are involved in analytical reasoning tasks, and these factors usually cannot be controlled completely, such as user's motivation and prior knowledge of the field. In usability inspections, it is unclear how these methods, which are designed for interface evaluation, can be used to examine visualization ideas [Tory and Moller, 2004], because the complexity of the visual analytics tasks makes it difficult to walk through or conduct heuristics evaluations in practice.

Qualitative Approaches

Qualitative methods can achieve richer and deeper understanding of information visualizations by considering the interplaying of many factors in the system as a whole in more realistic work settings. These techniques are often incorporated into both the design and evaluation of visualization systems, to guarantee the robustness and functionalism of the system. One type of qualitative techniques is based on observations that can be conducted in both laboratories and practical fields. The observations could be either passive, in which experimenters document findings by merely observing participants' reactions with the system in a task advocated before, or active, which encourages participants to speak their thoughts while they perform tasks (e.g., the think aloud protocol [Lewis and Rieman, 1993]). Another kind of qualitative methods is driven by interviewing expert users with the goals of developing realistic usage scenarios of the visualization system. The review process is based on ideas and opinions interchanged between the visualization designer and the user. Consequently many interview skills developed

[6]https://www.mturk.com/mturk/welcome.

in the HCI community are applicable in this process. Also, notes, sketches, and audios can be recorded for further development of scenarios. A third main qualitative approach uses longitudinal studies or long-term case studies [Shneiderman and Plaisant, 2006], which usually requires the deployment of a visualization system to participants for using it in their actual working environment for a longer time. System logs, notes, diaries, and video recordings generated by participants can be used for analyzing the effectiveness of the visualization system. After the study, interviews and questionnaires are often conducted to collect users' feedback and to further understand the process of participants using the system in the wild. Recently, Stasko [2014] proposed a new methodology for evaluating visualization driven by its value that consists of time efficiency of completing actions, ability to discover insights, overall essence, and ability to generate confidence about data. This value-driven approach clearly states what types of questions we should address over the course of evaluation.

While the qualitative methods overcome several issues of the quantitative approaches, there are still many known challenges. The first issue is that the sample size of participants, i.e., domain experts, is usually small in many cases of evaluating information visualization systems, because they are difficult to recruit. Although in some situations it is possible to recruit enough participants, for example, by allowing users to download the software freely online, the depth of study is usually decreased, thus revealing fewer insights in results. The second issue is the subjectivity: for instance, in some observational studies, findings are collected by a second person, i.e., the experimenter. Even for results reported from the first person, i.e., participants, there are still many subjective matters, such as their relationships to the experimenter, personal preferences, and knowledge of the field, which reduce the generality of extending such design to other scenarios. Further, not only being imprecise and time-consuming, but also qualitative methods introduce difficulties of analyzing and coding the data collected (e.g., notes, video recordings, and audio tapes).

A Methodological Note

In practice, the above two main categories of methodologies can be used together to enhance our understanding of user behaviors and system performances for real-world analysis tasks. Stasko [2014]'s value-driven approach has actually indicated such combination of methods, for example, time efficiency might be better evaluated with quantitative methods and other value components might be more suitable for qualitative methods. One key question is how to choose appropriate methods for a particular case. Munzner [2009]'s model provides a general idea about this by splitting the visualization design process into four nested levels, and thus suggesting particular evaluation methodologies (including both quantitative and qualitative methods) based on the threats at each level (Figure 2.5). In the design studies of this dissertation

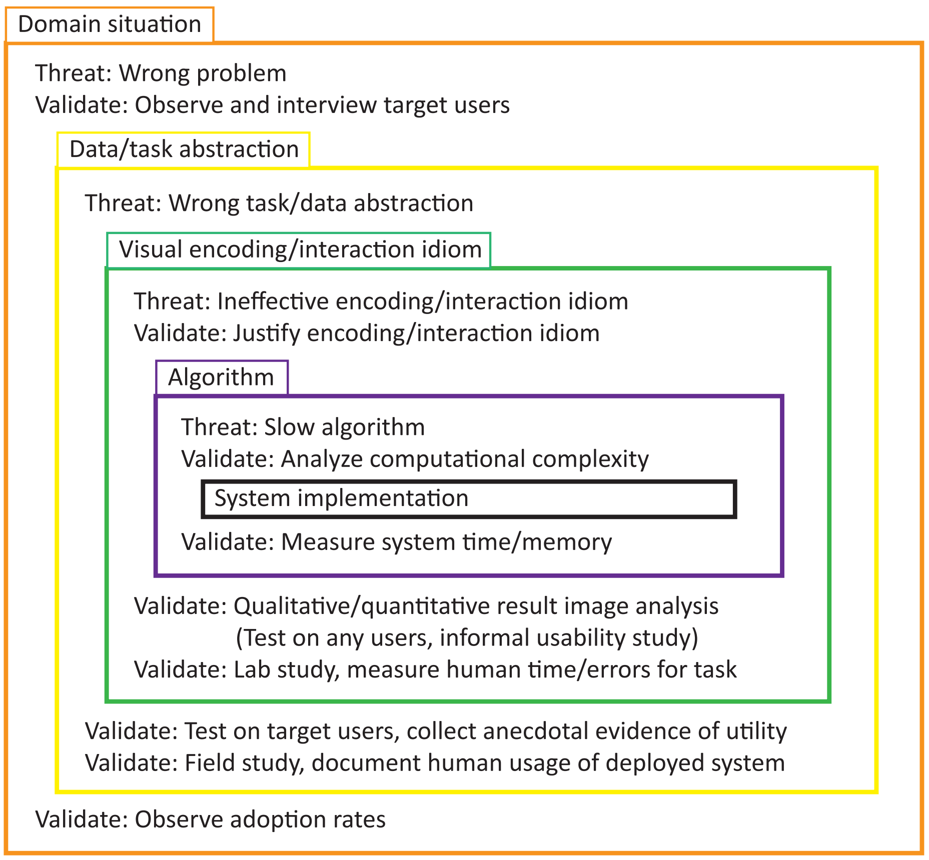

Domain situation
Threat: Wrong problem
Validate: Observe and interview target users
Data/task abstraction
Threat: Wrong task/data abstraction
Visual encoding/interaction idiom
Threat: Ineffective encoding/interaction idiom
Validate: Justify encoding/interaction idiom
Algorithm
Threat: Slow algorithm
Validate: Analyze computational complexity
System implementation
Validate: Measure system time/memory
Validate: Qualitative/quantitative result image analysis
(Test on any users, informal usability study)
Validate: Lab study, measure human time/errors for task
Validate: Test on target users, collect anecdotal evidence of utility
Validate: Field study, document human usage of deployed system
Validate: Observe adoption rates

Figure 2.5: Nested model for visualization design and validation, adapted from [Munzner, 2009].

(Chapter 3-7), we adopted different methods for evaluating our visualization prototypes in light of experiences and guidelines discussed in the previous work. As the main contributions are not algorithms, our design practices generally fall into the first three levels. Accordingly, we conducted observations and interviews with domain experts in each design study to identify domain-specific problems, and we carried out controlled laboratory experiments, participatory interviews, case studies, and field studies where appropriate, in order to validate our designs. Further, since visualizations developed in our design studies are mostly early prototypes and have few comparable techniques (perhaps none with published evaluations), we chose to explore a broader space of multi-focus visualization by employing primarily qualitative evaluation methods.

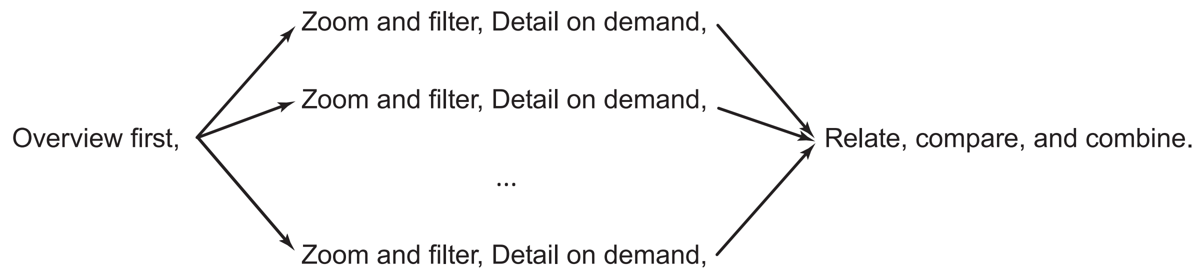

Overview first,
Zoom and filter, Detail on demand,
Zoom and filter, Detail on demand,
...
Zoom and filter, Detail on demand,
Relate, compare, and combine.

Figure 2.6: Multi-focus information seeking mantra in light of Shneiderman [1996]'s original information seeking mantra.

2.4 Towards Multi-Focus Visual Exploration

Broadly, the concept of multi-focus based techniques are rooted in some early studies about bi-manual manipulation in the HCI community, i.e., using two hands to complete interaction tasks [Buxton and Myers, 1986; Kabbash et al., 1994]. By generalizing bi-manual manipulation, Shoemaker and Gutwin [2007] proposed multi-point interaction which involves the operation of several mutually-dependent control points in a visual workspace, for example, adjusting a selection rectangle in a paint program. Shoemaker and Gutwin [2007] also suggested four important design requirements of multi-point interaction: control point visibility, scale adequacy, guaranteed manipulability, and visibility of intermediate regions. Elmqvist et al. [2010b] further generalized this concept in their multi-focus interaction model used in a distortion-based focus+context visualization technique called Mélange. Moreover, they formulated a number of design goals, including guaranteed visibility of focus and context as well as awareness of intervening context and distance, which share a similar spirit of Shoemaker and Gutwin [2007]'s requirements. There also exist numerous visualization techniques that employ this multi-focus mechanism to support effective exploration of large information spaces, such as TableLens [Rao and Card, 1994], Line Graph Explorer [Kincaid, 2006], StackZoom [Javed and Elmqvist, 2010] and PolyZoom [Javed et al., 2012].

While prior research has proposed a myriad of visualization techniques and associated interactions to assist visual data exploration, including those reviewed previously in this chapter (Section 2.1 and Section 2.2) and the ones discussed above, there still lacks of a deep understanding of how the multi-focus concept can be used effectively in visualization systems to better support sense-making of data. Thus, in this dissertation, we propose the concept of *multi-focus visual exploration*, based on the intuitions of multi-point and multi-focus interaction models [Elmqvist et al., 2010b; Shoemaker and Gutwin, 2007]. By extending ideas in the previous works, we define multi-focus visual exploration in a general form and in the context of information visualization — *an integrated knowledge discovery* *process in data which leverages dynamic coordination of multiple logically related data views (separated or distorted), each of which displays part of data with appropriated visual representation techniques, in a unified visualization*. In a similar language of Shneiderman [1996]'s information seeking mantra, the concept of multi-focus visual exploration can be expressed as "*overview first*", then multiple "*zoom and filter, detail on demand*" processes interleaved and executed in parallel, and "*relate, compare, and combine*" as an integral step to discover deeper insights. In this dissertation, we elaborate the concept of multi-focus visual exploration in both practices with design studies (Chapter 3-7) and theories that are distilled from the studies (Chapter 8).

2.5 Summary

This chapter reviews the literature related to various aspects of this dissertation research. Based on our primary focus that is to design novel interactive visualization techniques, this chapter discusses the large body of previous work on information visualization, along with its intersections of other related areas such as Human-Computer Interaction, Visual Analytics, and Design. More specifically, we first discuss the "static" component of information visualization — visual representation techniques — classified by the various types of data to present described in Section 1.2.2. We then describe the "dynamic" component — interaction — by introducing two main visual exploration paradigms (i.e., separated views and distorted views). This chapter also reviews many facets in the literature that has influenced the design of visualization techniques in this dissertation, including different conceptual models and theories of InfoVis, graphics design principles, and methodologies for conducting evaluations. Based on the previous work, we further define the concept of *multi-focus visual exploration* — the central theme of this dissertation.

In the upcoming chapters, we present the five design studies of developing highly interactive multi-focus visualization techniques in a range of real-world application domains and data formats. The proposed techniques were inspired by many aspects in the literature reviewed in this chapter, and we employed some of the general design guidelines, theories, and methodologies in practice, at the same time contributing new ideas. We classify the design study by what the data feature (i.e., data values, data structures, and data attributes) is primarily concerned and targeted, as fitted in the space of data to visualize described in Section 1.2.2. Where appropriate, chapter-specific related work and background will be discussed. In the end, we think retrospectively in Chapter 8 to generalize a set of key design considerations and theories commonly existing across all case studies, and compare their different approaches to fulfill these general guidelines.

Part II

Focusing on Data Values

Chapter 3

KronoMiner: Exploring Time-Series with Fluid Interactions

Time and tide wait for no man.

Geoffrey Chaucer

This chapter[1] presents the first design study with the focus on visual exploration of data values. We set our application context as time-series analysis, because characteristics existing in numerical data values of time-series are most concerned in many real-world domains, such as the analyses of climate measurements, computer system signals, and stock market prices. Time is considered uniform and absolute, and thus often treated as a special variable, in terms of how data is presented to users. Hence, most of the studies about time-series focus on other (dependent) variables, i.e., data values concerned here, with respect to time, for example, exploring large-scale time-series, identifying trends and recurring patterns, establishing correlations, and possibly predicting the future based on past and current behavior.

Although numerous tools have been developed for time-series processing and visualization, as reviewed in Section 2.1.1, there still lack techniques that are generic, thus not restricted in specific domains, and at the same time flexible enough to promote opportunistic explorations. To fill this gap, in this design study we developed KronoMiner, a multi-purpose time-series visualization framework equipped with fluid interactions and basic analytical capabilities (Figure 3.1). In a style of direct manipulation (i.e., operating on objects in-place and directly in the view), KronoMiner allows users to simultaneously browse and compare many time segments organized in a dynamic radial hierarchical structure, supporting visual exploration in a multi-focus manner. During the design study, the prototype system was iteratively refined

[1]Main portions of this chapter were previously published in Zhao et al. [2011a]. Thus any use of "we" in this chapter refers to Jian Zhao, Fanny Chevalier, and Ravin Balakrishnan.

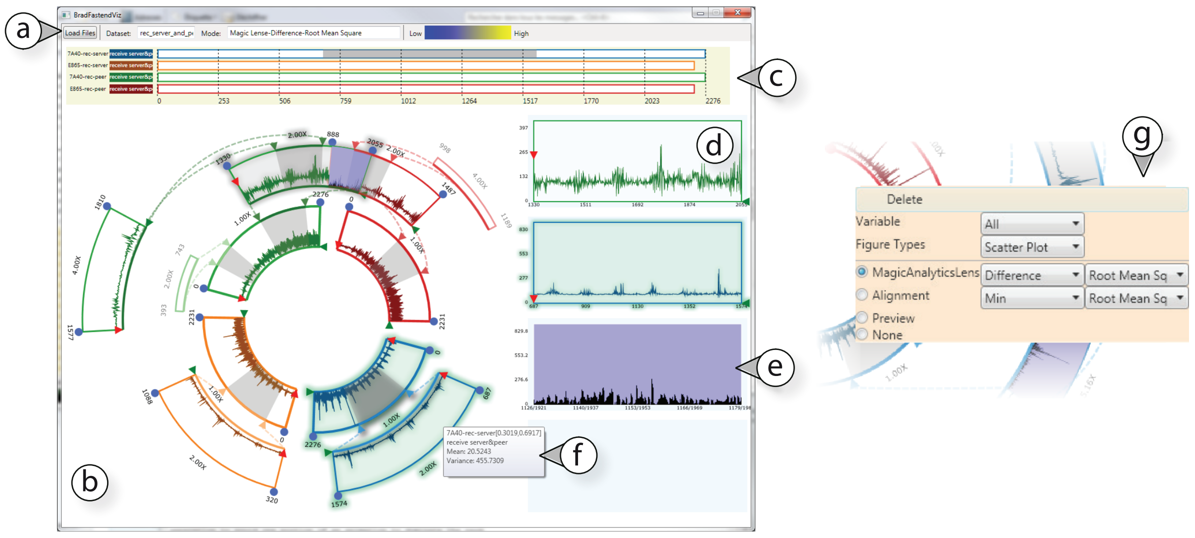


Figure 3.1: Exploration of four stock market datasets using KronoMiner. The visualization window of KronoMiner including a (a) status bar, (b) main circular tree window, (c) timeline overview, and (d) four plots comparison area (displaying (e) a MagicAnalytics Lens detailed view). Additional information is provided on demand with (f) a tooltip for each segment. Both general and context-based operations are available by invoking (g) a context menu. A video demo of the system can be downloaded here.

based on feedback from expert users. A preliminary qualitative evaluation with an expert who was not involved in the design process indicates that KronoMiner is useful and promising for exploring and analyzing time-series data in his everyday work.

3.1 Motivation

The need to analyze time-series data is prevalent in various application domains ranging from science, engineering, and economics. Many interactive visualization techniques have been proposed in the literature and demonstrated as effective means to analyze time-series in various real-world scenarios (see Section 2.1.1). In general, the existing time-series visualization systems are either domain-specific or multi-purpose.

Domain-specific tools are typically designed with expert users and are rich in specialized functionalities. For example, LifeLines aims to assist doctors in diagnosis and monitoring patient conditions [Plaisant et al., 1996], and LiveRAC [McLachlan et al., 2008] is designed to help computer network engineers optimize bandwidth usage. They are powerful for answering precise questions but cannot be easily applied to other domains. This confines users into predefined tasks and procedures, thus offering less flexibility in the exploration process and leaving little room for more opportunistic discoveries. Also, most of visual pattern mining techniques, such as PatternFinder [Fails et al., 2006], require prior knowledge of the domain (e.g., what characterizes an interesting pattern?), which cannot be easily applied in the context of domain-independent exploration. Nevertheless, combining visualization with general analytical methods, such as the difference plot of two variables, remains useful as an indication for tentatively identifying regions of interest.

In contrast, multi-purpose tools are not restricted to a specific domain by targeting a wider audience. However, they usually support a small limited number of tasks, and thus require other tools to be used in conjunction for complementary but necessary exploration capabilities. For example, SpiralGraph [Weber et al., 2001] and SpiralView [Bertini et al., 2007] require time-series to exhibit regularity for discovering patterns, and therefore are of a little utility for data without periodicity. Some techniques attempt to address the flexibility of data exploration by leveraging multiple dynamic separated or distorted views (see Section 2.2), such as StackZoom [Javed and Elmqvist, 2010] and PolyZoom [Javed et al., 2012]. However, their multi-focus views cannot be freely rearranged and basic analytical functions are missing, making it difficult to compare and discover relationships across regions of interest in time-series data.

Therefore, there is a need for interactive time-series visualization systems that are functional enough to support various kinds of basic time-series analysis tasks as well as are flexible enough for promoting opportunistic exploration in a domain-independent manner. Not only does domain-independence make a tool accessible to a wider audience, but also allows cross-examination of data from multiple fields, opening new possibilities for collaboration between analysts with complementary expertise. To address those concerns, we designed KronoMiner, a multi-purpose time-series exploration tool providing rich data navigation capabilities and necessary analytical support (Figure 3.1). With fluid direct manipulations, the visualization design of KronoMiner is based on a dynamic hierarchy of time-series segments that display data in a radial layout. Thus users can simultaneously explore, compare, and rearrange many pieces of data with different levels of details in a multi-focus manner.

3.2 Design Guidelines

To design KronoMiner as a general tool, we conducted informal interview studies with many domain experts to understand what features users need and how they generally carry out the exploration of time-series data. KronoMiner has been iteratively refined through 4 participatory design sessions with 7 experts from different domains at various stages of the development, including Computer Network (1 expert), Medical Science (1 expert), Weather Forecast (3 experts), and Finances (2 experts). We sampled those 4 domains ranging from science, engineering, and economics, and deliberately selected experts whose daily work involves much time in exploratory analysis of time-series rather than only numerical analysis. In the design process, each interview session lasted 4 to 5 hours, during which we first tried to identify their tasks, then introduced sketches of our early ideas, and gathered feedback and requirements.

We also performed extensive literature research to identify problems that have not been addressed in previous time-series visualization tools. One of the inherent limitations of traditional time-series interactive visualizations lies in regarding time as being an immutable and uniform dimension. Depending on the tool, visual exploration is usually constrained to at least one of the following: 1) *synchronism*: different attributes are usually time-aligned, making it difficult to compare delayed sequences; 2) *chronology*: time-series data is often considered as ordered data, for which successive pieces cannot be rearranged; 3) *time scale consistency*: comparing time-series simultaneously at different time scales is rarely supported. For example, squishing or stretching the time scale of server logs independently makes it possible to adjust the visualization according to response-time capacity.

In the rest of this section, we describe the design requirements of multi-purpose time-series visualization tools, and discuss the design space of layouts for visualizing time-series data.

3.2.1 Design Requirements

Based on comments from interviews with our expert users and the knowledge we gained through our literature survey, we derived the following design requirements.

R1 Multivariate, multiple and heterogeneous datasets. Dealing with as varied input data as possible is important for supporting multi-purpose use.

(a) *Multiple heterogeneous datasets:* supporting time-series from various data sources to facilitate cross-concept comparison and refrain from being domain-dependent (e.g., one may consider the correlations between housing prices and weather data that are from distinct sources);

(b) *Multivariate data:* visualization and manipulation of both coupled (e.g., velocities of an object along three axes) and dissociated attributes (e.g., temperatures of two different locations);

(c) *Variety of chart representations:* providing different chart plotting choices of the same time-series facilitates the data analysis from different perspectives (e.g., line charts for stock analysis, and amplitude chart for audio data).

R2 Multiple coordinated views. When user actions receive immediate visual feedback, multiple views aid visual exploration by providing diverse perspectives of the data [Aigner et al., 2008].

(a) *Large-scale overview:* acquiring general context at any exploration state through an overview of the entire datasets help maintain a mental map of the data [Shneiderman, 1996];
(b) *Multi-focus, temporal hierarchies:* showing several levels of detail simultaneously in coordinated views can aid in effective browsing of the data [Javed and Elmqvist, 2010] (e.g., users may want to access hourly, daily, and weekly precipitation from various time ranges to conduct in-depth data comparison);
(c) *Side-by-side comparison:* juxtaposition of visualization facilitates data comparison than visually spaced-out charts or animations that requires remembering previously-seen views [McLachlan et al., 2008].

R3 Fluid interaction. Giving users as much freedom as possible to manipulate and query the data is crucial for visual exploration [Aigner et al., 2008].

(a) *Direct manipulation:* physically manipulating graphical representations of objects has the benefit of being easy to use and learn, and allows for greater system comprehension [Shneiderman, 1983];
(b) *Overview first, zoom & filter, details on demand:* the widely followed Shneiderman [1996]'s principle is recognized as effective for dealing with large datasets;
(c) *Dynamic multi-focus, temporal hierarchies:* drilling down into multiple regions of interest by time brushing (i.e., manually select data on an interactive data display) eases comparison [Javed and Elmqvist, 2010];
(d) *Aggregation for data abstraction:* grouping several regions of interest into a single entity allows for user-defined clustering and data abstraction (e.g., users may save the customized data group and reuse it for future analysis);
(f) *History:* maintaining the history of the exploration process is important to help users revisit views and keep track of their exploration [Heer et al., 2008];
(e) *Annotation:* annotations aid for organizing and lower memory load as they act as labels and reminders.

R4 Analytical methods. All our expert users echoed that semi-guided exploration based on statistical analysis should be considered [Aigner et al., 2008].

(a) *Cross-concept relationships:* displaying the relations (e.g., links, and similarity scores) resulting from the computation of classical domain-independent analytical methods provides important clues for visual exploration, for instance, showing correlation scores between unit time segments (e.g., a week of stock prices) of different time-series can help users know where to dive in;

(b) *Matching time-series:* finding the best match between data pieces and their matching score allows for pattern mining [Buono et al., 2005] (e.g., a user may wonder which part of today's network traffic can be best aligned to yesterday's traffic for similarity analysis);

(c) *Dynamic comparison:* displaying on-the-fly computed information (e.g., difference plot) facilitates identifying regions of interest if immediate visual update is supported while manipulating the data, because users need to effectively decide which part of data worth further exploration;

(d) *Customization:* allowing for user-defined operators avoids a multi-purpose tool to be too limited for specialized domain-dependent analysis, because a wide variety of time-series analysis methods are applied by users in data exploration such as cross-correlations and moving averages.

3.2.2 Design Space of Time-Series Layouts

Here we review different layouts that can be used to represent time-series data as a complement to Meyer et al. [2009]'s taxonomy. Figure 3.2 depicts the possible layouts for multivariate (a, b) and multiple (c-f) time-series datasets. Note that an exhaustive taxonomy would include spiral layouts previously described, but these ones are suited for data that exhibits a periodicity, which is out of the scope of this design study aiming at developing general-purpose visualization tools. Let us consider a daily temperature data of three locations as an example in the following.

We first distinguish between multivariate time-series, which implies the plots of all dependent variables (with respect to time) to be coupled, for example, the highest, the lowest, and the average daily temperatures, and multiple independent datasets (that could be multivariate or not), for example, average daily temperatures of three locations. Multivariate data can be overlaid in a single plot (Figure 3.2-a), but visual clutter might be overwhelming; an alternative is to stack multiple charts (Figure 3.2-b). Different chart types can be set for each individual plot independently without overlap, but this requires more screen real estate and might split the user's visual attention.

For plotting more than one time-series dataset at a time, one can choose between a linear and a circular layout, and use stacked or sequential plots (Figure 3.2-c to -f). For example, each rectangular or circular bar in Figure 3.2-c to -f can represent one of the three daily temperature measurements, or of the three locations. Linear layouts are more familiar to users and do not distort the data. Circular layouts are more appropriate for visualizing hierarchical structures

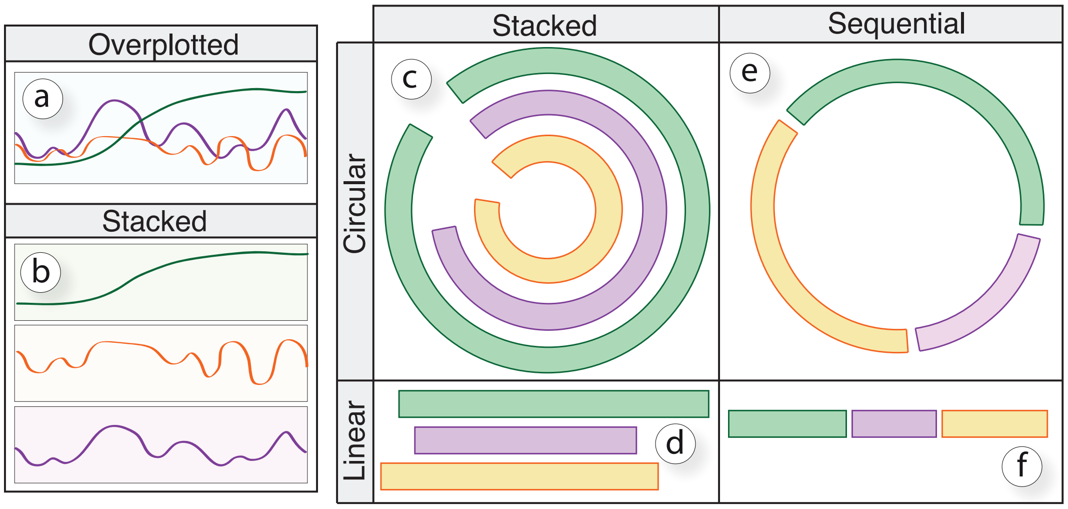


Figure 3.2: Design space of layouts for multiple multivariate time-series: (a)-(b) multivariate layouts, and (c)-(f) multiple datasets layouts.

and relationships [Draper et al., 2009]. They also successfully apply to periodic data when spirally-shaped, for example, when displaying long-term climate data, but this raises issues when considering multiple series with different periodicity and time interval as the series have to be piled up in the same spiral. The sequential linear view (Figure 3.2-f) makes it difficult to compare time-aligned data, especially when the plots are spaced out and cannot be rearranged. For example, users want to compare daily temperatures of three locations in the same month, thus need to align them in time. The circular stacked layout (Figure 3.2-c) preserves time alignment, but the same data plotted on different rings is perceived differently, which may cause misinterpretation. For example, the same one-month data is displayed longer when placed in an outer ring. As described next, the KronoMiner interface combines both circular (Figure 3.2-e) and linear (Figure 3.2-d) layouts.

3.3 Visualization of Time-Series Hierarchy

The main visualization component of KronoMiner, as shown in Figure 3.1-b, is based on a dynamic hierarchical structure to assist a multi-focus inspection of data at different scales and locations (R2-b). The design aims to ease the identification and manipulation of multiple sub-pieces of interest for finer analysis, rather than providing a single global view of numerous long time-series to be analyzed as a whole. We have chosen a circular sequential layout for the display of the multi-focus time-series hierarchy mainly because of its compact representation of tree structures [Draper et al., 2009], but also putting selectable data within easy reach of the user. Each of the tree nodes corresponds to data points within a specific range of a parent series, and is displayed as a concentric arc segment on which the data is plotted. The central ring contains all the available full datasets (R1-a,b). Segments belonging to different datasets can be easily distinguished by the color of their outlines. Detailed information such as timestamp bounds and time scale is displayed besides each arc segment.

A parent-child relationship of two segments is explicitly visualized as a pair of dashed curves that link one segment to the corresponding region of interest (ROI) on its parent. Data points belonging to the ROI are duplicated in the associated segment, as a new and more detailed view of the same initial dataset. To visually reinforce the hierarchical structure, ROIs are filled in with light gray, and arcs on both of the child and parent segments are color-coded to give awareness of which portion of this data piece resides in the series, where a distinguishable color hue is mapped to each dataset. The dashed links as well as the lower boundary of each child segment are colored with the same hue as its parent. The luminance conveys the position in the parent series (the brighter the sooner in the parent). Moreover, when hovering over a segment, its ancestors and descendants in the hierarchy and the associated ROI in the parent are emphasized by glowing effects (Figure 3.1-b). We also indicate the hovered segment's location in the timeline overview (R2-a, Figure 3.1-c), and a tooltip showing detailed information is popped-up (R3-b, Figure 3.1-f). Duplicated data is thereby clearly pointed out by providing immediate feedback on the different coordinated views as the user interacts (R2).

3.4 KronoMiner Interface

The design of the entire KronoMiner interface relies on the aforementioned guidelines, and has been iteratively refined based on experts feedback gathered at different development stages. As shown in Figure 3.1, KronoMiner combines several coordinated visualization components, taking full advantage of the individual strengths of each layout to effectively support different tasks. In this section, we describe various aspects of the visualization tool, including its flexible interactions with the time-series hierarchy supporting multi-focus visual exploration and comparison of data, as well as its dynamic analytical methods facilitating the discovery of deeper insights.

3.4.1 Direct Manipulation on Time-Series Hierarchy

As introduced in Section 3.3, the core of this KronoMiner visualization is the dynamic time-series hierarchy. The whole mechanism of creating and adjusting time segments in the hierarchy relies on the direct manipulation principle (R3-a) to support a quick and effective way of drilling down into multiple locations of the datasets (R3-c). To enable quick learning

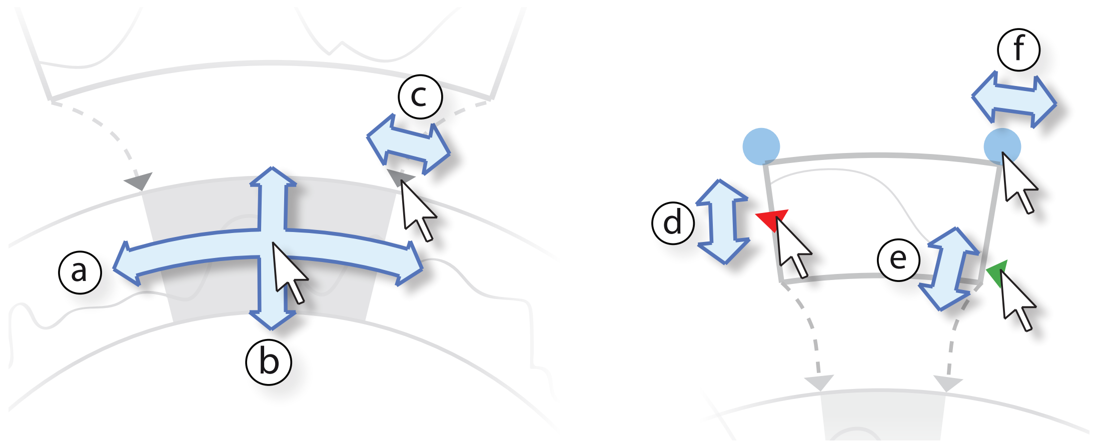


Figure 3.3: Direct manipulation on the dynamic time hierarchy: a region of interest can be (a) moved and (b,c) resized, and a time segment can be operated for (d) changing y-axis amplitude, (e) shifting y-axis, and (f) stretching or squishing the time dimension.

of the interface, all the operations for editing the hierarchy are performed on the main canvas through simple mouse interactions and modifier keys.

Building the Hierarchy

A new segment can be created by directly brushing (i.e., dynamic selection in the interactive view) a region of interest (ROI) on any existing series in the hierarchy (R3-a,c). A user is required to right click on the desired starting point — a new segment appears on the ring located one level outside from the currently queried data piece, and the ROI is colored with light gray. The ROI and its associated segment are dynamically adjusted while a user drags the cursor, until the mouse button is released at the intended ending point, finalizing the creation. When a user needs to focus on a specific time frame, she can perform the above time brushing (while holding Shift) on a collection of segments belonging to a ring at once to simultaneously create many child sequences.

Figure 3.3 depicts how the multi-focus time intervals can be adjusted by direct manipulation on the view (R3-a). A user can look at earlier or later data points in the parent series by dragging the ROI along the ring, keeping the same time frame length but at different time locations (Figure 3.3-a). Dragging the cursor on the radial direction towards the center (Figure 3.3-b) reduces the interval. Conversely, a longer interval is obtained by dragging in the opposite direction. This interaction is inspired by the OthoZoom scrollbar [Appert and Fekete, 2006]. Arrows ending the dashed curves that link a segment to its corresponding ROI are also draggable and act as sliders to adjust the ROI size (Figure 3.3-c). Each time segment can also be modified using the round handles placed on the extremities of the visualization object (Figure 3.3-f), which allows for stretching or squishing the data representation, thus affecting the time scale used for rendering. For more precise manipulation, a user can also set up specific parameters by invoking the segment setting pop-up menu (Figure 3.1-g). Two more controllers (Figure 3.3-d,e) explained later, are provided to the user for data plot visual settings.

Editing the Hierarchy

In general, the time-series hierarchy mimics a user's exploration history (R3-f), since a new segment is created each time when a user drills down into a specific data part (with all the existing segments remain in place). The hierarchy can be edited freely with direct manipulation at any time, thus supporting an effective navigation of the data along the exploration history.

To avoid the view being too crowded because of the presence of too many segments, a user can collapse a segment to display it in a reduced height. Hence, visual clutter is reduced, making it possible to focus on a small subset of visually detailed segments without losing the mental map of the current hierarchy. Figure 3.1-b shows examples of both collapsed and detailed segments.

Moreover, when a user wants to focus on the deepest levels of the hierarchy because they correspond to the finest details, she does not require the whole hierarchy to be displayed all the time. In our radial representation, the deepest levels are located on the outer rings. To release space and focus visual attention, a user can shrink the inner rings by dragging a ring towards the center of the circular view — the ring interaction described later (Figure 3.5-c). Figure 3.4-b shows an example of the resulting dynamic time-series hierarchy after inner rings shrunk.

Keeping the whole history of exploration is sometimes not relevant when intermediate stages are not important for the analysis. For example, when a sequence of interest has been identified, a user might no longer need its ancestors in the hierarchy. To get rid of unused segments, a user can delete either a single segment or a segment and its whole sub-hierarchy by entering a *Deletion* mode through a context menu (discussed later). The deleted pieces disappear in a smooth fade-out, avoiding abrupt changes in the visualization.

Context-Based Interaction

We introduce a novel context-based interaction language for facilitating manipulation of the time-series hierarchy. We exploit the current position of the cursor to determine whether a user intended to perform an action on (a) a single segment, (b) a segment and its whole children branch, (c) a single ring, or (d) the whole view. Figure 3.5 shows a summary of

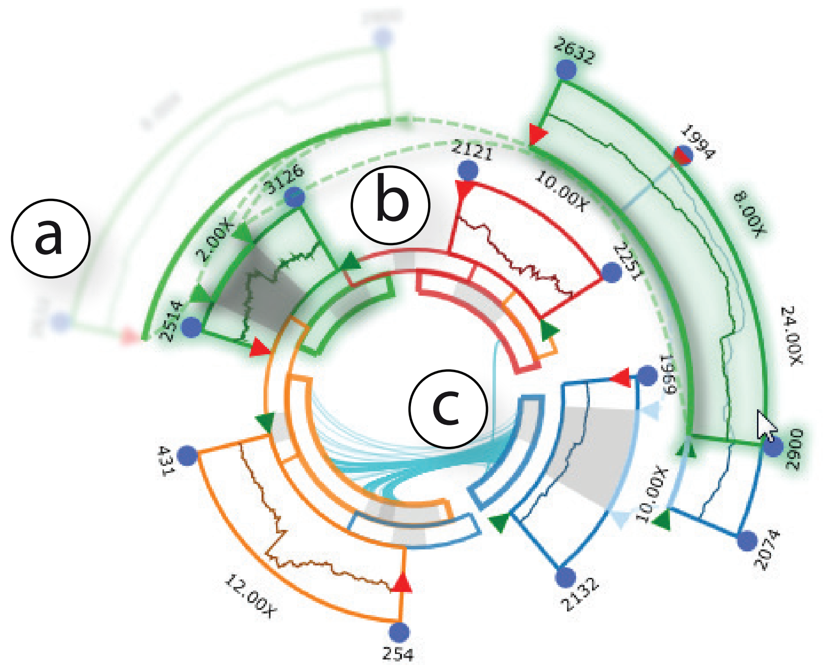
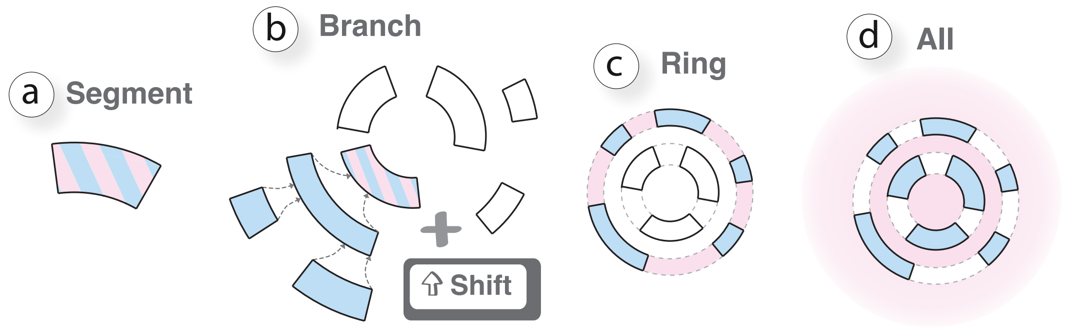


Figure 3.4: Exploring the NYSE exchange daily open price (1970-2010) with KronoMiner: (a) a phantom of the manipulated segment remains at the initial position as it is temporary overlaid on another segment for trend comparison in the Preview Mode, (b) the inner rings have been shrunk to save screen real estate for the deepest levels of the hierarchy, and (c) the relationships (matching subsequences) between the datasets are shown in the center as links.

the four different selectable groups, and the available operations for each group using the mouse buttons and the modifier key. The pink areas correspond to all the cursor positions that determine an object group, which is indicated in blue. For example, by holding the Shift key and dragging a segment, a user may performs a branch moving operation that rotates a segment and all its descendants together (Figure 3.5-b); and the whole hierarchy can be manipulated in a similar way when the cursor in the empty space of the view (Figure 3.5-d). This context-based interaction allows a user to adopt different strategies for visual exploration and optimize the manipulation. For example, the rings can be used to categorize the data; each ring being dedicated to a specific group that a user wants to keep related (R3-d). By performing actions in the ring-mode (Figure 3.5-c), a user can apply the same changes to all the elements of the group in a single operation while keeping consistency within the group.

Keeping consistency while designing direct manipulation is possible although limited: there is a finite number of ways of binding operations to the combination of mouse buttons, mouse wheel, and modifier keys. Other interaction techniques such as context menu must be integrated as a complement to direct manipulation to offer more complex operations and capabilities. Following the same context-based principle just described (Figure 3.5), additional



		2	Drag	Drag	Wheel
a	Select	Collapse Extend	Rotate Ring jump	Create and adjust a new segment	
b					
c				Shrink inner rings	
d			Rotate	Pan	Zoom

Figure 3.5: Context-based interaction on the time-series hierarchy. The activation area is showed in pink, and the affected elements in blue. Interaction can be performed on (a) a single segment, (b) a segment and its children branch, (c) a whole ring, and (d) all rings at the same time.

functionalities are available on a context menu (Figure 3.1-g). For example, a user can perform deletion operations that affect a targeted set of segments, depending on the cursor position where the menu is invoked. Similarly, previews and visual settings of a group of related segments are also accessible on the context menu, which will be described later.

3.4.2 Flexible Visual Comparison

Visually comparing and correlating different parts in data is critical in the visual exploration process to promote insight discoveries. In addition to comparison of multiple time segments that is inherently enabled in the dynamic time-series hierarchy, KronoMiner also supports data comparison in a traditional manner with side windows (Figure 3.1-d).

In-Place Comparison with Time-Series Hierarchy

In the time-series hierarchy, two segments can be spaced out, sometimes showed upside down because of the radial layout, which make the comparison more difficult than looking at side-by-side pieces (R2-c). To facilitate comparison, we provide a user with the full freedom of rearranging times segments in the view: a segment can be rotated anywhere, and ring jumps are also supported. Thus, segments can be freely aligned either on a stacked configuration by putting them on different rings, or overlaid to obtain overplotted data (Figure 3.2-a) as they are semi-transparent.

Looking for the unexpected is difficult and tedious, which usually requires going through a lot of operations before serendipitously discovering relevant insights in data. To avoid a user messing up the hierarchy layout during the exploration, we allow for a *Preview* mode in which the current layout is maintained while she performs a temporary analysis (e.g., dragging a segment on top of another). A phantom of the segment remains in place at its initial position, and is restored as she releases the mouse button (Figure 3.4).

KronoMiner is designed to be as free as possible in terms of comparing multiple sub-pieces in several datasets: segments from any loaded dataset, at any location in data, and rendered at any time scale, can be placed anywhere in the view. Easy access to these operations through the direct manipulation techniques allows for a quick and smooth exploration of data with immediate visual feedback (R3). However, giving too much freedom to the user also has its drawbacks. The hierarchy structure might not always be easy to read after reorganizing the segments. Visual cues have to be integrated to help the user maintain her mental map. To this end, we use explicit dashed links, color coding, and brushing & linking of related objects in a branch (ancestors and children in the hierarchy). All these techniques are designed to provide as many visual cues as possible to keep awareness of the overall tree structure.

Detail Comparison Window

Radial layouts suffer from distorting the data which might be constraining for a more precise analysis. To balance the drawbacks, we designed a dedicated Detailed Comparison Window, located on the right of the interface (Figure 3.1-d), where a number of selected segments can be displayed and compared in a traditional linear stacked layout. To facilitate the navigation of time-series, the comparison window is enriched by interactive capabilities: a user can pan the plot for further exploration or alignment of time-series, and the corresponding ROI and its associated segment in the main view are updated at the same time. Similar to Figure 3.3-d and -e, Two controllers are also available to set up y-axis amplitude and shift. A full coordination between the two views is supported so that a user can easily keep track of the impact of one operation on the other view. Moreover, both the time segment and its copy in the Detailed Comparison Window are highlighted while hovering over one or the other to make it clear which data is common in coordinated views (R2).

3.4.3 Dynamic Analytical Methods

Analytical methods can help gain insight into time-series by automatically detecting periodicity, recurrent patterns and self-similarity [Aigner et al., 2008]. When a user has no idea or only vague idea of the patterns she is looking for, precise analytical computations answering specific questions are of little use. In this case, people rely on visual exploration in order to find out regions of interest in data. KronoMiner enriches its visual exploration capabilities by integrating analytical analysis support in three different ways: 1) visualization of the relationships within and between datasets, 2) the MagicAnalytics Lens: on-the-fly computation and display of relational plots of two data segments, and 3) the Best Match mode: dynamic computation and display of the best matched sub-sequences of two time-series.

Showing Dataset Relationships

As an start, KronoMiner computes cross-dataset and within-dataset relationships that indicate matchings between different data sub-sequences, using a simple brute-force similarity mining algorithm. But any other algorithms or predefined relationships can be applied. Similarly to Meyer et al. [2009], the relationships are displayed at the center of the radio layout as links connecting the data items on the main ring. The links intensity is mapped to the matching score (the darker, the higher). To make trends visually obvious, we employed the edge bundling technique described in Holten [2006] with alpha blending resulting from overlapping semi-transparent links (Figure 3.4-c). In this way, a user has a constant access to relationship information for visually mining regions of interest (R4-a), thereby guiding visual exploration.

MagicAnalytics Lens

Detailed pairwise comparison is one of the main tasks when analyzing time-series data. It usually involves computing statistical measures, such as similarity scores and cross-correlations, between two series (or subsequences of them), sometimes at a different time scale. Traditional statistical tools, such as Matlab[2], allow for a fine analysis of the data: a wide variety of predefined analytical methods are available, as well as the possibility of defining new functions. However, in such tools the user is required to specify exact instructions (e.g., the bounds of the interval to focus on) using a command line before visualizing the result. In the case that a user does not know exactly what to look for, she might generate a lot of plots before finding something relevant. Not only is the process repetitive, haphazard, and time-consuming, it is also distracting as a user splits her attention focus back and forth between the command

[2] http://www.mathworks.com/products/matlab/.

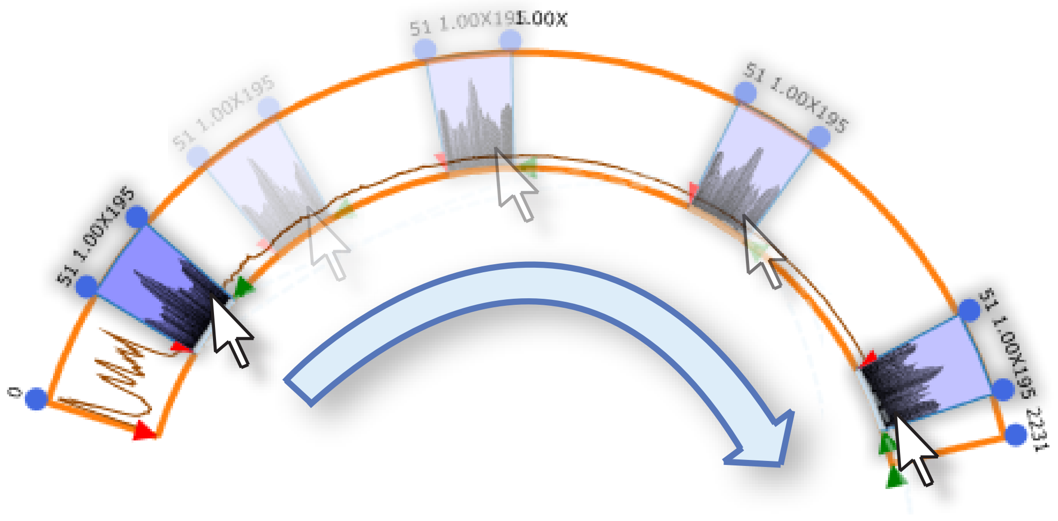


Figure 3.6: Example of using the MagicAnalytics Lens on peer-to-peer network activity logs. A segment that roughly corresponds to one day of activity is used as a lens. The lens displays the cross-correlation plot, and the background color conveys the overall value difference by the root mean square operator on a yellow-to-blue scale, in this case showing that a daily pattern is repeated as a user drags the lens along the series.

interface and the visualizations. Moreover, these tools usually require some basic programming skills, which narrows the target audience.

To avoid breaking the flow of the exploration process and facilitate opportunistic discovery, we introduce the *MagicAnalytics Lens*, an interactive visualization technique that computes the result of a function involving two time-series in real-time, and immediately displays its result (R4-c). MagicAnalytics Lens is inspired by Magic Lens [Bier et al., 1993] that affects pixels of an image inside the bounds of a shape defining the lens. In contrast to Magic Lens that uses a dedicated window as a lens applying a fixed predefined operator, MagicAnalytics Lens considers all data pieces as potential lenses, as the displayed result depends on both the data points in the window used as a lens, and the data points under it. In Figure 3.6 the lens shows the cross-correlation plot between two series using the data contained in the overlapped interval. The plot is automatically updated as a user moves or stretches the lens segment. The background color of the lens is mapped to a computed global measure (root mean square in Figure 3.6). In this way, a user can quickly assess if the current alignment contains closely related time-series. Any segment in the view can be used as a MagicAnalytics Lens, making it possible to compare data between both the original series and data at different time scales (as segments cna be stretched and squished). A MagicAnalytics Lens can also be plotted in the synchronized Detailed Comparison Window described earlier (Figure 3.1-e).

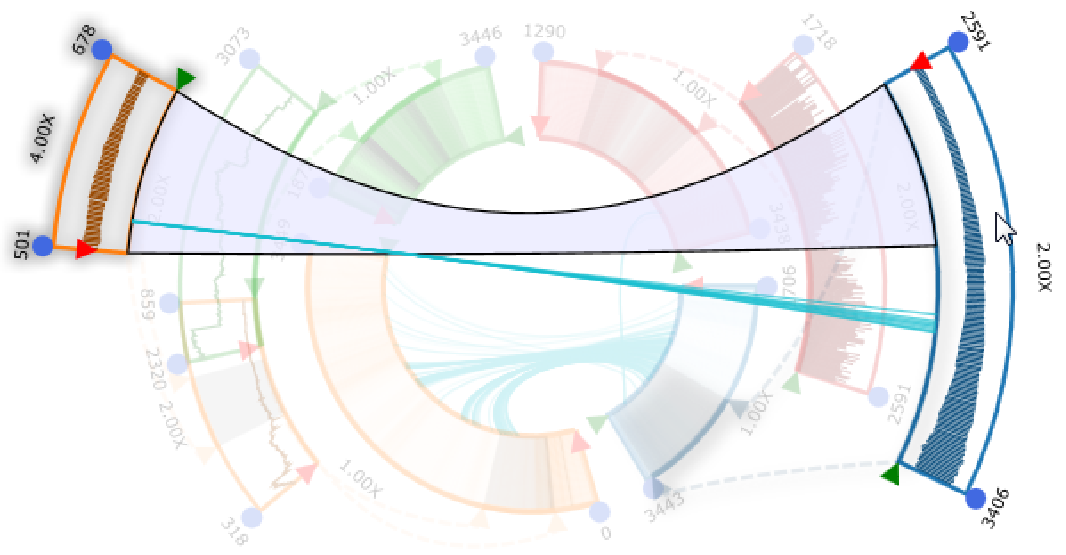


Figure 3.7: Example of finding the best match between two segments of NYSE exchange daily open price (1970-2010) by minimizing the root mean square. An arch conveys the best matched location, and its background color is mapped to the matching score on a yellow-to-blue scale. The precomputed relational links are replicated between the two sequences. In this example, the relationships between the segments surprisingly do not reflect the best match location, which might be worth exploring further.

The MagicAnalytics Lens is available through the contextual menu (Figure 3.1-g). When entering the mode, all the overlaid areas on segments are replaced by results of selected analytical functions. The current prototype integrates general functions such as the difference plot, the minimum, or maximum plots for lens operators, and inner product or root mean square for the global measure shown as the background. Ideally, users should be able to define their own functions and customize the tool by introducing new analytical methods on demand (R4-d).

Finding Best Matched Time-Series

Pattern discovery is an important task for analyzing time-series data, as it is a preliminary step to discover new ways of defining what could characterize relationships. KronoMiner integrates a *Best Match* mode (R4-b), where the current selected segment defines the querying pattern, and hovering over a segment defines the matching target. The best match (based on a selected similarity measure) is dynamically computed and displayed: an arch links the querying segment to its best matched position within the target segment, and the rest of the main view is faded out to avoid visual clutter (Figure 3.7). The background color of the arch shape is mapped to the matching score. To provide context, we also replicate the same precomputed

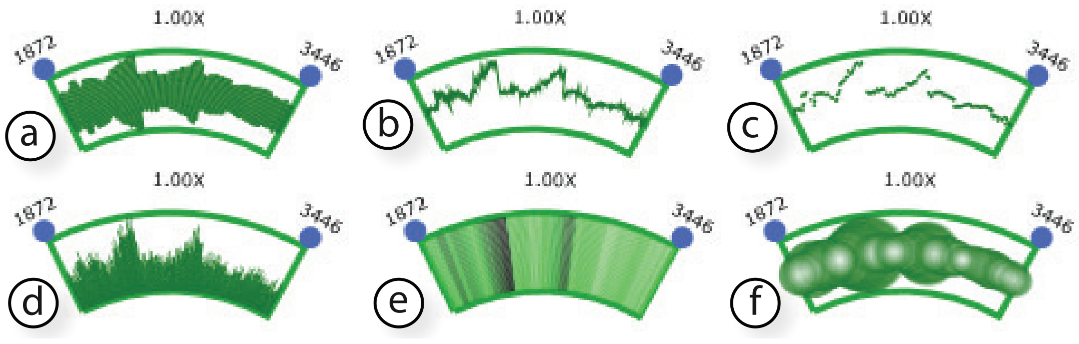


Figure 3.8: Different plot styles available in KronoMiner: (a) amplitude plot, (b) line chart, (c) scatter plot, (d) histogram, (e) lines plot, and (f) bubble plot.

relationships as showed in the center of the radial display. By hovering over the arch, a user can access the detailed information, such as the precise score value and properties of the matched segments. Similarly to the MagicAnalytics Lens, the Best Match operation also allows for comparing sequences at a different time scales.

3.4.4 Visual Settings of Plots

To support various analytical purposes, KronoMiner provides six different plots (Figure 3.8), however other types such as horizon graphs [Heer et al., 2009] or heatmaps could be easily added. Note that the amplitude and bubble plots particularly suit for radial layout as their perception is less affected by the circular deformation. KronoMiner also supports data display adjustments by allowing a user to alter the y-axis amplitude and shift, using the two draggable handles on each side of a segment in both the main view and the comparison window (Figure 3.1-b,d). More specifically, dragging up and down the left handle respectively increases and decreases the amplitude, and moving the right handle up and down shifts the chart display along the y-axis.

One might have to deal with multiple variables while keeping them coupled, for example, coordinates of a moving object, min/max temperatures, and opening/closing price of the stock market. KronoMiner supports multivariate datasets by piling up multiple views as shown in Figure 3.9. In this example, the MagicAnalytics Lens reveals a repeated pattern, but the trend is observed at a different time scale. The room price and the CPI evolve in a similar way at different time periods, but twice as fast in the latter time period.

Although it is possible to make rings larger to fit more than three variables at a time, KronoMiner does not scale to larger dimensions because of visual clutter. Design improvements (e.g., dynamic reordering and stacking) are needed to support high-dimensional

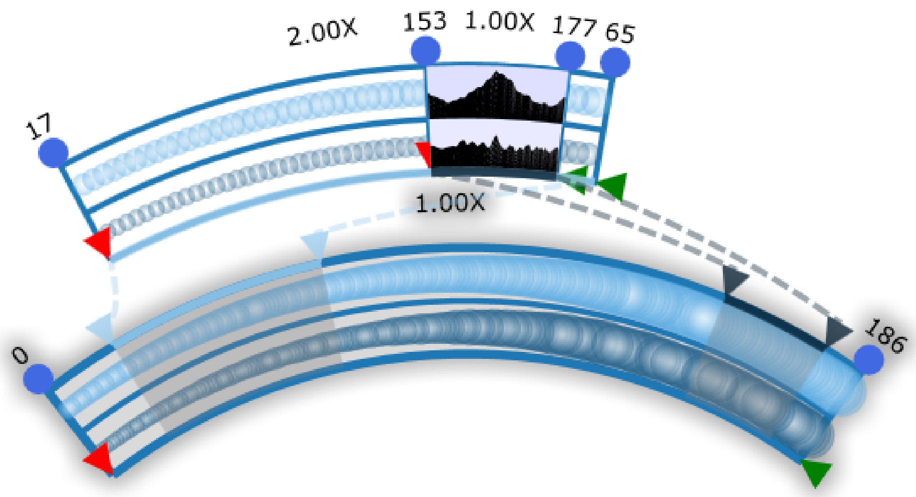


Figure 3.9: Example of using the MagicAnalytics Lens on a multivariate dataset. Top layer plot: monthly average cost of a night's accommodation in Victoria; bottom layer plot: Consumer Price Index in Melbourne (1980-1995). The correlation plot and the background color (root means square) indicate common trends between the two segments.

data. Space-efficient alternative visualizations such as the braided graphs [Javed et al., 2010] would partially solve the problem but may imply issues for the MagicAnalytics Lens design.

3.5 Preliminary Case Study

An expert user who was not involved in the design process of KronoMiner has tested the final prototype with his own data in 2 working sessions (3 hours each). The participant is a researcher studying large scale peer-to-peer traffic data on video-on-demand websites, and usually uses Matlab as means of analyzing the data. He has worked in this area of research for 4 years. After an introduction of the KronoMiner interface and functionalities, the user was asked to explore his data using the think-aloud protocol.

We first observed that the user was clearly more comfortable with the interface during the second interview, whereas he complained *"it is difficult to remember, I have to learn"* when testing the earlier prototype. At the end of the sessions, he compared the learning process: *"It took me weeks, even months to learn how to use Matlab because I had first to learn code. I can do a lot with your tool, and I don't even have to think about it"*. For example, the participant mentioned that it took a lot of time to master the Matlab language and be familiar with the function calls: extracting subsequences of time-series, interpolating and scaling the data, and finally plotting the chart was not that easy. We strove to make the interaction in KronoMiner as direct as possible so that not breaking the exploration flow. Context-based interaction requires training, but one cannot benefit from complex actions while always keeping straightforward interactions. We believe a novice user to be able to quickly get familiar with the different functionalities but this remains to be validated in further studies.

Our expert's dataset consisted of a month of the visiting population of two different channels, sampled from a server every 10 minutes. He first wanted to test the hypothesis that the sub-sequences of everyday's data are similar within and between channels. To do so, he brushed a region of interest on one channel corresponding to what he described as *"the first day, roughly"* (he was able to identify it because of the characteristic shape of the plot). To refine his selection, he displayed the segment in the comparison window, changed the plot type to gain a better view and barely adjusted the time frame, pointing that *"it doesn't have to fit exactly, I just want a quick look"*.

The user panned the plot in the comparison window to get an overall idea of day-to-day data, and possibly spot anomalies or interesting behavior that he would not see in the radial representation. He adjusted the scale and the shift of the y-axis as the amplitude of the plot decreased. At some point, he mentioned that *"the plot is too small now, I can't say if I see a pattern, or if it is noise. So far I've been able to barely identify a similar pattern, but now it's not possible anymore, and increasing the amplitude wouldn't help"*. He thus decided to use the MagicAnalytics Lens as a support to analyze the cross-correlation. As the expert dragged the *"first day"* segment along the sequence, he observed a constant peak on the center of the computed plot (Figure 3.6), indicating a high correlation at a zero delay. This justified part of his hypothesis and he thought that the lens dragging was "*very handy"*.

Next, the participant roughly selected the *"last day"* data points, and dragged both days segments to outer rings to gain a better view, and made them overlap, still in the MagicAnalytics Lens mode. He then dragged the first segment to sequentially compare the data from the latest day to the rest of the sequence, and validated his hypotheses as they revealed being correlated. While doing so, our user emphasized that *"In Matlab you have to generate tons of graphs. Even if you have enough space to display them, you don't have enough brain. Here, all you have to do is to simply drag [the segment]"*.

Then the user started to explore the data on the secondary channel. He found the daily cross-correlation to be high between the two datasets, but not as strong as within the same series. After extensive navigation and digging into the data, the work space turned to be messy. Our user started to rearrange the segments, commenting that "*I need to organize*": shrunk inner rings, and collapsed some remaining segments, as they were not of interest at this time, saying that *"I don't need this guy [a time segment he just explored] for now..."*. When cleaning his space, he realized he looked at many plots without noticing: "*This saves me a lot of time. Normally I have to eyeball all the data first because I don't want to go into every pieces because* *it is tedious"*. Further, he gathered unexpected findings as he dynamically adjusted the range of the ROI to time intervals less than a day: *"The data around midnight is similar to that around noon! It's interesting. I would have expected it is the case at other rush hours, but people should sleep at midnight. It is worthwhile to study deeper"*.

Our user was enthusiastic about analytical capabilities in KronoMiner and the fact of direct manipulations on visualization objects. He said that he wanted to explore more datasets with more variables and longer time periods using our prototype, as it would be too time-consuming with traditional tools. He mentioned that for analyzing the peer-to-peer traffic data, he would require to freely "*stretch the time*" as netowrk servers likely have different hardware. Hence, the performance plots have to be adjusted for a fair comparison. The Best Match mode would also be helpful as the need for mining guidance increases with the number and size of the time-series.

3.6 Discussion

The KronoMiner interface contributes a first step toward multi-purpose time-series exploration tools seamlessly integrating fluid interactions and analytical supports in a dynamic visual representation of data. In this section, we discuss several insights that we identified during the development and evaluation of KronoMiner which provides some implication for designing general visual exploration tools.

Look: provocative or serious? When analytical tools do not present users with a sober interface that exhibits several menus, control panels, and numerous settings, they are often not taken seriously as not looking professional. People may also not know what to start with the absence of menus. Hence, these tools are usually admitted to be nice-looking and pleasant to "play" with, but their appearance has a negative impact on users' reliance. A tool that feels too easy to use might suffer from its apparent simplicity. Paradoxically, users highly demand for more interactivity and user-friendlier interfaces, and thus it creates a tension between apparent professionalism and accessibility. KronoMiner demonstrates that it remains possible to perform effective analysis while offering high interactivity in a seamless integration. There is a strong need to convince users to accept that the absence of menus and settings panels does not mean a tool is not powerful. Our user who was baffled at the beginning *"I've never seen that before"*, ended up asking us for a release of the prototype to use it for his research.

Back to the roots. The previous observations raise the question of breaking habits. Integrating familiar interaction styles and representations seems to be essential for users to feel confident and consider the tool to be potentially useful. Surprisingly, our expert user has not been able to start any constructive exploration during the first meeting, as he *"[couldn't]* *work without seeing somewhere some plots stacked in a linear view"*. However, he did not use the Detailed Comparison Window as much as expected during the second interview. To his own surprise, he admitted that it was reassuring to be able to get back to the traditional representations: *"I know it is there, so I feel comfortable using the tool now"*. It is difficult to make an interface feel familiar since people have different working habits, especially when targeting a wide audience. We strove to design KronoMinerto be as flexible as possible for multipurpose analysis through participatory design studies deliberately involving researchers from disparate domains, as a guarantee not to specialize to a domain and an attempt to cover common needs. By doing so, we thought our tool could be adapted to a wide audience. In return, it is still too general to fully support deeper analysis, as our user quickly missed analytical operations that he was used to apply: *"It would be great if you can add more operators to the lens"*. Allowing users to customize the tool has revealed to be crucial for functionalities extension purposes. We also believe that this would help them accept more easily new ways of interactions, as they could personalize the tool to fit with their needs.

Towards more elaborate data analysis. Based on the case study with our domain expert, we also identified several limitations of the current KronoMiner system, which needs to be improved to support more elaborate time-series analysis. Although KronoMiner incorporates highly flexible interactions allowing users to effectively explore data in a multi-scale and multi-focus approach, in most of the cases, users can only browse the raw time-series (with time-axis scaling). However, in the practice of exploratory time-series analysis, our expert pointed out that it is important to see derived data by applying some basic transformations (e.g., derivatives, moving average, etc.) to original values. This would help users verify more complicated hypotheses on-the-fly. Although the MagicAnalytics Lens shows data relationships derived from two time segments when they overlap, a variety of data transformations like this is needed to support deeper visual analysis of time-series. Moreover, the expert user mentioned that it would be very helpful that the system could support combining a series of basic data transformations together to form an operation pipeline in a similar spirit of the smooth interaction offered by KronoMiner. Lastly, a seamless integration of automatic analysis methods and user-guided visual exploration is also critical. Currently, KronoMiner only supports the detection of similar time-series motifs before hand and shows the matching sequences with links (Figure 3.4-c). A more dynamic integration is demanded with real-time computation as users interact with the visualization. For example, the system can suggest the most interesting ROIs of a time segment based on where the user starts to brush the data and how long the time-series is brushed.

3.7 Summary

In this chapter, by situating the application domain as time-series analysis, we have presented a design study on multi-focus visual exploration of data values, which approaches the challenges arising from promoting the discovery of opportunistic insights and cross-concept relationships in generic datasets. The research prototype, KronoMiner, integrates novel visualization metaphors with direct and context-based interactions, providing users with much freedom and flexibility to ease the comparison and analysis of multiple data segments of interest simultaneously. KronoMiner illustrates how multi-focus navigation enriched with basic analytical operations can effectively help users perform exploratory analysis on time-series data. A preliminary case study with an expert from computer networking research has shown the effectiveness of KronoMiner. In this design study, we have made the following contributions:

- a taxonomy of design requirements for multi-purpose time-series visualizations gathered from experts with various technical backgrounds,
- a review of the design space of layouts for visualizing multiple multivariate time-series,
- a novel technique called MagicAnalytics Lens that performs in-place, on-the-fly visualization of correlations between two data inputs in overlapped regions, and
- a flexible time-series exploration tool, KronoMiner, based on a dynamic hierarchy visualization seamlessly integrated with fluid interactions.

This chapter has focused on the field of time-series analysis because temporal data widely exist in real-world applications where the data values are most concerned. Although KronoMiner was designed based on this particular domain, many aspects of its visualization and interaction are generalizable to other fields when focusing the exploration of data values. For example, if a dataset does not has the time dimension, as long as it can be expressed linearly (with fixed or flexible orders of data points), the proposed dynamic hierarchy idea (Section 3.3) is applicable to multi-scale and multi-focus exploration of such data. So do the layouts described in Section 3.2.2. Similarly, the proposed dynamic analytical methods (Section 3.4.3) can be extended to discover cross-concept relationships between two data pieces.

In the upcoming chapter, we will further explore multi-focus visualization techniques for supporting visual exploration of data values, also situated in this time-series analysis domain. Following the spirits of enabling multi-scale and multi-focus display of large datasets with direct manipulations, we will go a step deeper in developing advanced analytical functions to support more elaborate visual analysis tasks of time-series data, as pointed out as the limitations of KronoMiner in Section 3.6.

Chapter 4

ChronoLenses: Analyzing Time-Series with Interactive Pipelines

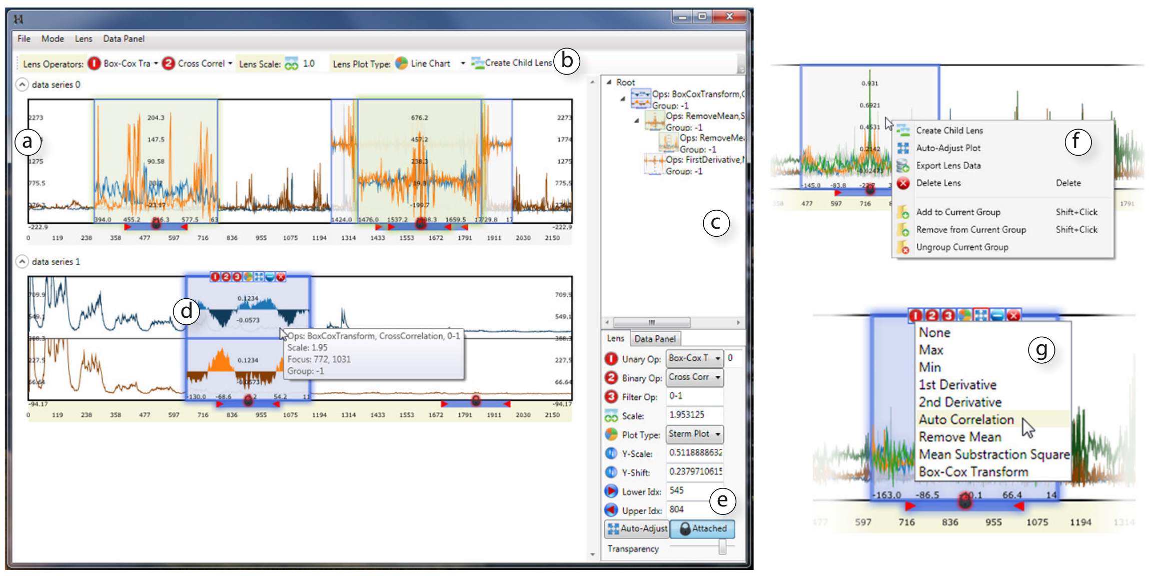
Time changes everything except something within us which is always surprised by change.

Thomas Hardy

In this chapter[1], we present the second design study on visual exploration of data values in the application domain of time-series analysis. Many previous studies have focused on the visual representation of time-series, such as revealing the dependency, branching or periodical characteristic of certain temporal data (see Section 2.1.1), and enabling multi-scale and multi-focus display of large datasets with interactions (see Section 2.2). There has been comparatively little research on how to support tasks in visual exploratory analysis of time-series at a more sophisticated level, e.g., visualizing transformed values, identifying correlations, or discovering anomalies beyond obvious outliers. Chapter 3 has touched this aspect by introducing a visualization system with two basic analytical functions for pairwise comparison of time-series: real-time rendering correlation results when two data segments are overlapped (MagicAnalytics Lens), and reveal of the best matched parts across two time-series based on a similarity measure.

However, there is a further need for analytical methods supporting more elaborate time-series analysis tasks, which often demand reusing or modifying transformed data values

[1]Main portions of this chapter were previously published in Zhao et al. [2011b] © 2011 IEEE. Reprinted, with permission, from: Jian Zhao, Fanny Chevalier, Emmanuel Pietriga, and Ravin Balakrishnan, Exploratory Analysis of Time-Series with ChronoLenses, IEEE Transactions on Visualization and Computer Graphics. Thus any use of "we" in this chapter refers to Jian Zhao, Fanny Chevalier, Emmanuel Pietriga, and Ravin Balakrishnan.



Unary Op: Box-Cox T 0
Binary Op: Cross Corr
Filter Op: 0-1
Scale: 1.953125
Plot Type: Stem Plot
Y-Scale: 0.5118888632
Y-Shift: 0.2379710615
Lower Idx: 545
Upper Idx: 804
Auto-Adjust
Attached
Transparency

Figure 4.1: The ChronoLenses interface includes (a) a charts panel showing the time-series; (b) a lens creation toolbar; (c) a lens analysis pipeline view of (d) the currently selected lens; (e) a property panel showing details of the currently selected lens. A context menu (f) can be invoked to perform lens-based operations, and (g) the lens toolbar allows quick access to a lens' parameters. A video demo of the system can be downloaded here.

across different temporal data segments. Such tasks typically require deriving new time-series from the original data, and trying different functions and parameters in an iterative manner. Similar requirements were briefly discussed in Section 3.6 based on the KronoMiner case study. In this chapter, we introduce a novel visualization technique called ChronoLenses (Figure 4.1), that performs on-the-fly transformations of data in multiple focus regions, shown with a metaphor of magnification lenses to support focus+context exploration. ChronoLenses allow users to progressively construct advanced analytical pipelines by creating lenses with different basic operators to form a dynamic lens hierarchy. Each lens in the hierarchy derives data based on the output of its child lens and/or the original time-series, propagating results in customized analytical pipelines. In the design study of ChronoLenses, our domain expert users from astronomy and computer networking were able to efficiently monitor and analyze complex time-series data.

4.1 Motivation

Visual representations of time-series take advantage of people's innate perceptual abilities to process information and detect structure, making it significantly easier for users to discover trends and patterns at different scales, but also to identify anomalies in the data [Buono et al., 2005; McLachlan et al., 2008]. However, basic time-series visualizations using line plots do not scale well, as time-series data are often very large, featuring multiple, possibly heterogeneous, dependent variables measured for long periods of time and/or at high sampling rates. As discussed in Chapter 2, many interactive visualization tools have been developed to address this scalability problem, mainly along the line of applying focus+context lens-based [Furnas, 1986] or similar visualization techniques, such as StackZoom [Javed and Elmqvist, 2010], SignalLens [Kincaid, 2010], and Line Graph Explorer [Kincaid, 2006].

However, there has been comparatively little research on how to support the more elaborate tasks typically associated with the visual exploration and analysis of time-series, which require deriving new data points (e.g., computing the derivatives and cross-correlations), visualizing those new time-series, and relating them to the original data plots. Visual exploration takes advantage of human abilities to drive the data browsing process, and is especially useful for undirected searches, when users know little about the data or have only vague exploration goals [Keim, 2002; Shneiderman, 1996]. As emphasized by Keim et al. [2008]'s visual analytics mantra — *"Analyze first, Show the important, Zoom, Filter and analyze further, Details on demand"*, this process is iterative. For it to be efficient, visual representations, that support human judgment, and interactions, that re-parameterize the visual representation, should be tightly integrated, enabling users to quickly choose and refine parameter values that best suit the task at hand [Aigner et al., 2008]. New plots derived from the original data should be put in context and made easy to relate both to the original data and to other plots that have been derived as part of the exploratory process.

Therefore, we must go beyond the simple transformations offered in current focus+context lens-based visualization techniques, by offering multi-step transformations of one or more pieces of time-series to support the above iterative visual exploratory analysis of data. Also, various types of time-series transformations should be flexibly integrated in a customized analysis pipeline; these can be purely visual transformations (e.g., scaling and magnifying data), or transformations that change data values (e.g., 1st derivative, point-wise subtraction).

To that end, we introduce a novel, domain-independent time-series visualization technique called ChronoLenses (Figure 4.1), aimed at addressing the above concerns. As many other types of lens-based techniques, ChronoLenses delimit a region of interest in data to be put in focus through magnification [Cockburn et al., 2009; Pietriga et al., 2009], visual filtering [Furnas, 1986] or other arbitrary transformations of the underlying content [Bier et al., 1993]. Based on the metaphor of direct manipulation, ChronoLenses perform on-the-fly transformation of the data points in their focus areas, tightly integrating exploratory visual analysis with interaction. Users can build pipelines composed of lenses performing various transformations on data, effectively creating flexible and reusable time-series visual analysis interfaces. At any moment, users can change the parameters of already created lenses, with the modifications instantaneously propagating down through the pipeline, providing immediate visual feedback that supports the iterative exploration process.

4.2 Tasks and Design Requirements

To better understand the needs for exploratory analysis of time-series data with interactive visualizations, we further consolidated feedback collected from domain expert interviews described in the previous design study (Chapter 3). Based on their comments and the literature [Andrienko and Andrienko, 2006], we identified low-level tasks that users typically perform to carry out time-series analyses. From these tasks, we derived design requirements for the ChronoLenses technique, as introduced in this section.

The analysis of time-series datasets typically implies feature extraction and data comparison by transforming one or more time-series into another. Such computations usually require performing one, or a combination of, the following low-level tasks:

T1 Single-data stream transformation. Transform each data point of a series by applying an operator. Examples are data alteration, e.g., Fourier transforms[2] and Box-Cox transforms[3]; bias reduction, e.g., remove means and remove trends; repeated pattern identification, e.g., auto-correlation; and operations related to delays and time lags, e.g., differencing and seasonal differencing.

T2 Cross-data stream analysis. Compute a new time-series from two or more input time-series. Examples are data comparison, e.g., point-wise subtraction; similarity examination, e.g., inner product; relationship discovery, e.g., cross-correlation; and series aggregation, e.g., point-wise maximum value.

However, exploring time-series often implies going through a more elaborate analysis pipeline that combines various tasks [Keim et al., 2008]. Practically speaking, this implies that users should be able to make compound queries on the data by iteratively performing sequences of low-level tasks T1 and T2, combining and modifying the transformation parameters as part of the visual exploration process. From these observations we derive the following requirements:

R1 Dynamic transformations. Low-level tasks T1 and T2 are the core components of the visual analysis process. The transformations that support them should be easy to perform

[2]http://mathworld.wolfram.com/FourierTransform.html.
[3]http://onlinestatbook.com/2/transformations/box-cox.html.

through operators applied to the input time-series data. Visualizations and interactions that re-parameterize these transformations should be tightly integrated so as to facilitate exploratory analysis.

(a) *Dynamic selection of input region of interest:* users should be able to dynamically select and modify what timespan(s) in the input data are to be processed through the operators.
(b) *Dynamic transformation parameters:* users should be able to easily configure and edit transformation parameters.
(c) *Immediate visual feedback:* the system should provide instant visual appearance update to reflect these changes.

R2 Dynamic analysis pipeline. Enabling the easy combination of operators makes it possible to progressively build and refine the analysis pipeline, thereby helping formulate complex queries [Keim et al., 2008].

(a) *Dynamic composition:* users should be able to build the analysis pipeline iteratively, combining basic transformations through the incremental composition of operators that take as input arbitrary combinations of time-series from the original dataset and time-series that result from earlier transformation steps upstream in the pipeline.
(b) *Reuse of intermediate results and easy backtracking:* the above requirement entails that all intermediate time-series transformation steps in the pipeline should be reusable as input to downstream operators, enabling users to branch out and explore, and possibly compare, alternatives at any point while sharing the data transformation steps that come earlier in the analysis process.
(c) *Visual representation of the pipeline:* the system should provide an overview of the analysis pipeline, helping users maintain a mental map of the transformation steps and keep track of the exploration history.

We also rely on general principles for visual exploration as listed in the previous design study (see Section 4.2), including: direct manipulation; overview first, zoom & filter, details on demand; and support for dynamic multi-focus exploration.

4.3 ChronoLenses Visualization Framework

ChronoLenses is an interactive visualization technique for the exploratory analysis of time-series data, whose design was driven by the above requirements. It computes on-the-fly transformations of data points and displays the result of those transformations in place, using the metaphor of lenses. The MagicAnalytics Lens technique introduced in Chapter 3 relies on the Magic Lens concept [Bier et al., 1993] and forms one of the basic building blocks for ChronoLenses. While MagicAnalytics lenses enabled users to apply for single-step basic transformations to exactly two input time-series, ChronoLenses extends the concept, allowing for an arbitrary number of input time-series and introducing several additional operators. But most importantly ChronoLenses enable multi-step transformations to be specified, where the result of time-series transformed through a given lens can serve as input to another lens, and where multiple pipelines can be branched out to explore multiple alternative visualizations simultaneously, easing the formulation of complex queries.

4.3.1 Lens Parameters

We define a lens $\mathcal{L}$ as the transformation of an input time-series in the focus region of that lens into a resulting time-series. Time-series can be seen as streams of data that get transformed through the lenses that constitute an analysis pipeline structured as a data-flow. Transformations are computed on-the-fly according to a set of parameters that can be dynamically adjusted[4]. To support tasks T1 and T2, $\mathcal{L}$ can either transform a single-datastream, or perform cross-data stream operations. Data streams can be univariate or multivariate. Each lens $\mathcal{L}$ is defined by four transformations, all optional:

Unary Operator: $\mathcal{L}_{unary}(\cdot)$ defines the transformation that applies to a single-datastream in the focus region of the lens (T1);

Binary Operator: $\mathcal{L}_{binary}(\cdot,\cdot)$ defines the transformation that takes the data in the focus region of the lens (processed by $\mathcal{L}_{unary}$ if set to anything else than the identity transform) as the first operand, and the output data stream resulting from the parent lens in the hierarchy (detailed later), if any, as the second[5] operand (T2);

Filter: $\mathcal{L}_{filter}(\cdot,\theta)$ defines visual filters that hide time-series specified by parameter θ (applies to multivariate data streams only);

Scaling: $\mathcal{L}_{scale}(\cdot,s)$ determines the magnification factor s applied to the data rendered in the lens' focus, thus facilitating a multi-scale exploration of data.

Operators $\mathcal{L}_{unary}$ and $\mathcal{L}_{binary}$ perform actual computations on the input data and can take some data points outside the lens' time span, such as when computing the 1st derivative. On

[4]ChronoLenses support the representation of stacked multivariate time-series. In that case, each individual series is processed separately using the $\mathcal{L}$ transformation, and is then stacked in the lens' frame.

[5]Most of the operations that consider more than two input data streams can usually be simulated by cascading a series of binary operations.

the contrary, $\mathcal{L}_{filter}$ and $\mathcal{L}_{scale}$ only affect the visual representation of the processed data within that timespan. They do not need to access data outside it.

4.3.2 Pipelines of Lenses

As mentioned earlier, the output of a lens, i.e., the time-series resulting from the transformation of an input time-series, can be fed to one or more lenses. In other words, lenses can be piped, effectively creating a lens-based data flow pipeline that can be used to progressively build elaborate visual analysis interfaces. The hierarchical combination of lenses relies both on a layering system and on cross-data stream parent-child relationships between lenses. The tree structure not only helps users keep track of the sequence of exploration steps, but also makes it possible to *backtrack*, that is, adjust the intermediate processing stages iteratively, the system providing immediate visual feedback of the consequences of these parameter adjustments by propagating them downwards in the hierarchy. In the following, we describe the steps involved in rendering data seen through a lens, and then describe how lenses can be combined to build a full analysis pipeline.

Rendering Pipeline

Figure 4.2 illustrates the steps of the rendering pipeline applied to some multivariate input data, eventually leading to the visualization of the resulting time-series through lens $\mathcal{L}$, that gets part of its input from a parent lens.

More formally, the rendering pipeline can be described as follows. Let $\vec{D} = (d_1, d_2, \ldots, d_m)^T$ be the m-dimensional set of data points defined by the focus region of lens $\mathcal{L}$. We denote $\mathcal{L}_{op}(\vec{D_i})$ as the result of applying one of the transformations steps to the data, where $op \in \{unary, binary, filter, scale\}$. The final result $\mathcal{L}(\vec{D})$ is obtained by performing the following four computational steps:

C1 Apply unary operator $\mathcal{L}_{unary}$, such that $\vec{D_1} = \mathcal{L}_{unary}(\vec{D})$ (Figure 4.2-1);

C2 Apply binary operator $\mathcal{L}_{binary}$, such that $\vec{D_2} = \mathcal{L}_{binary}(\vec{D_1}, \vec{D_p})$ (Figure 4.2-2), where $\vec{D_p}$ is the set of data points resulting from applying the parent lens to the same or to another set of input data points (Figure 4.2-P);

C3 Apply visual filter operator $\mathcal{L}_{filter}$ such that $\vec{D_3} = \mathcal{L}_{filter}(\vec{D_2}, \theta)$, where θ defines the subset of time-dependent variables to be rendered in case of multivariate input (Figure 4.2-3);

C4 Apply scaling operator $\mathcal{L}_{scale}$ such that $\vec{D_4} = \mathcal{L}_{scale}(\vec{D_3}, s)$, where s defines the magnification factor (Figure 4.2-4). We note $\mathcal{L}(\vec{D}) = \vec{D_4}$ the final result.

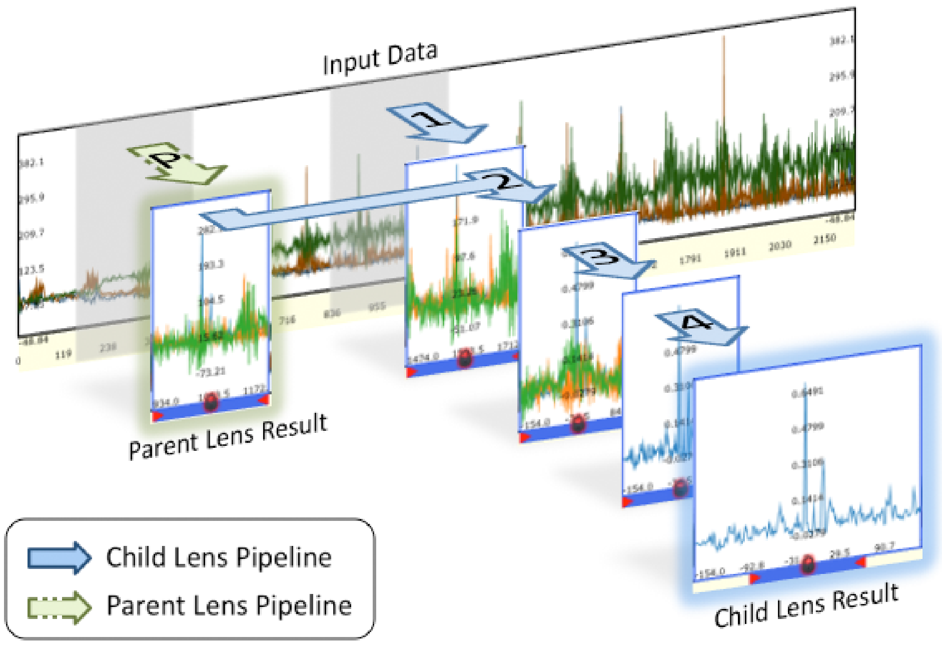


Figure 4.2: ChronoLenses rendering pipeline for lens $\mathcal{L}$. (P) processing pipeline in parent lens; (1) applying *Unary Operator* $\mathcal{L}_{unary}$; (2) applying *Binary Operator* $\mathcal{L}_{binary}$; (3) applying *Filter Operator* $\mathcal{L}_{filter}$; (4) applying *Scaling Operator* $\mathcal{L}_{scale}$.

Any of the four operators can be set to $null$, in which case the associated step is equivalent to an identity transform. As mentioned earlier, operators $\mathcal{L}_{unary}$ and $\mathcal{L}_{binary}$ perform actual computations on the input data while $\mathcal{L}_{filter}$ and $\mathcal{L}_{scale}$ only affect the visual representation of the data. Thus, a simple magnifying lens can be obtained by only setting $\mathcal{L}_{scale}$. And a lens showing only one time-series in a multivariate plot can be obtained by only setting $\mathcal{L}_{filter}$ with θ limited to that particular series. The following section focuses on the more complex computational steps C1 and C2, and describes complete analysis pipelines involving multiple lenses.

Analysis pipeline

Users can specify elaborate transformations by combining multiple lenses. Two lenses $\mathcal{L}_1$ and $\mathcal{L}_2$ can be combined either by overlaying them in the chart, in which case the operators defining the lenses get applied sequentially to the input time-series data points (first those of $\mathcal{L}_1$, then those of $\mathcal{L}_2$); or by declaring $\mathcal{L}_1$ as the *parent* lens of $\mathcal{L}_2$, the transformed output of $\mathcal{L}_1$ being an operand of $\mathcal{L}_2$'s binary operator. Figure 4.3-I gives a schematic representation of these two combination mechanisms. Basic transformations that apply to a single-datastream only require a lens with a unary operator (Figure 4.3-a) as defined in C1. Cross-datastream computations require the definition of a child lens (Figure 4.3-b), that takes as input a data stream resulting

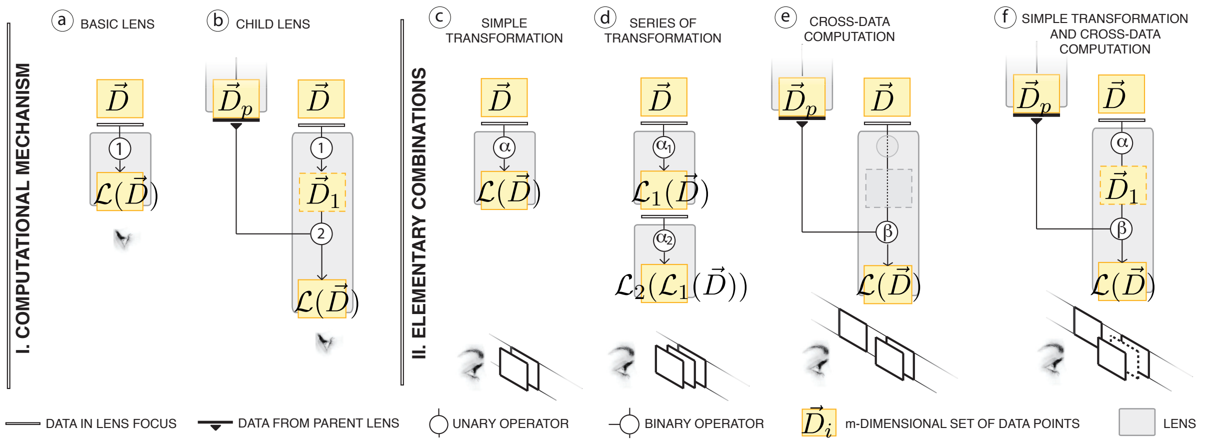


Figure 4.3: Schematic representation of the ChronoLenses combination mechanism (I), and all possible elementary combinations (II).

from a parent lens and the data in the child lens' focus. The two streams are processed through the child lens' binary operator (C2), with the unary operator (C1) preprocessing the data points in the child lens' focus, if set. Figure 4.3-II illustrates all possible elementary lens combinations:

E1 *Simple transformation* (Figure 4.3-c): the most basic lens, where $\mathcal{L}_{unary}$ is set and $\mathcal{L}_{binary}$ is $null$. It is conceptually similar to a Magic Lens [Bier et al., 1993], with the unary operator (noted α) transforming the data in the lens' focus region. For example, one can apply a lens with a Gaussian filtering operator to smooth the original time-series.

E2 *Series of transformations* (Figure 4.3-d): two lenses that are piled up (they observe the same timespan) on separate layers as in traditional graphics editors. The first lens processes the input time-series data points that fall into its focus through unary operator α_1. The resulting data is fed to the second lens and gets processed by unary operator α_2. It is possible to pile up an arbitrary number of lenses, as with Fishkin and Stone [1995]'s Movable Filters. For example, a user can pipe another lens with a derivative operator on top of the Gaussian filtering lens, in order to compute the derivative of the smoothed data as in E1.

E3 *Cross-data computation* (Figure 4.3-e): cross-data computations are achieved by creating a parent-child relationship between two lenses. Output data from the parent lens and time-series data points in the child lens' focus region are the operands given to binary operator β. The unary operator of the child lens is set to $null$ (identity transform). A straightforward example would be computing the point-wise difference between the result of the parent lens and the input of the child lens, in order to known how much those two time-series differ.

E4 *Simple transformation and cross-data computation* (Figure 4.3-f): both the unary operator α and the binary operator β are set. The lens first applies transformation α to the time-series data points in its focus. The result of this transformation, $\vec{D_1}$, is then fed to the binary operator β along with the output of the parent lens' transformation, $\vec{D_P}$. An example is that the parent lens outputs the result of the Gaussian filtering as in E1, and the child lens first applies Gaussian filtering of the original time-series and then performs point-wise difference as in E3: this allows users to see the difference between two data segments after the filtering (or smoothing).

More elaborate analysis pipelines can be built by creating hierarchies consisting of the above elementary combinations (see Section 4.5).

4.4 ChronoLenses Interface

We developed a proof-of-concept, full-featured interactive visualization tool implementing the ChronoLenses technique for the visual exploration of multivariate time-series. The interface (Figure 4.1) consists of five main components: (a) the *chart panel* displaying the different time-series in a classical stacked view; (b) a *lens creation toolbar*; (c) a *lens analysis pipeline view* displaying all ancestors of the currently selected lens (d); and (e) a *property panel* showing the latter lens' settings. Following the design requirements listed in Section 4.2, the interactive visualization was designed to allow rapid exploration of time-series and construction of analytical pipelines. This section describes the interactive visual interface of our prototype. Concrete examples of use are introduced in the following section.

Figure 4.4-I shows the user interface of a lens $\mathcal{L}$. The region of interest (time span $\vec{D}$ in the lens focus) is visually represented as a blue *focus bar* on the time axis (Figure 4.4-c). The time-series $\mathcal{L}(\vec{D})$ is displayed inside the lens' frame (Figure 4.4-b). A toolbar located on the top border of the lens provides quick access to the lens' main parameters (Figure 4.4-a). This toolbar is visible only when the cursor is hovering over the lens, so as to minimize visual clutter. Detailed information of a lens is also provided in a tooltip when hovering.

The user can modify the visual representation of plots, offering different perspectives on the data. Our prototype supports classical plots such as line charts and dot plots, but also statistical plots including histograms and Q-Q normal plots for comparing data distributions.

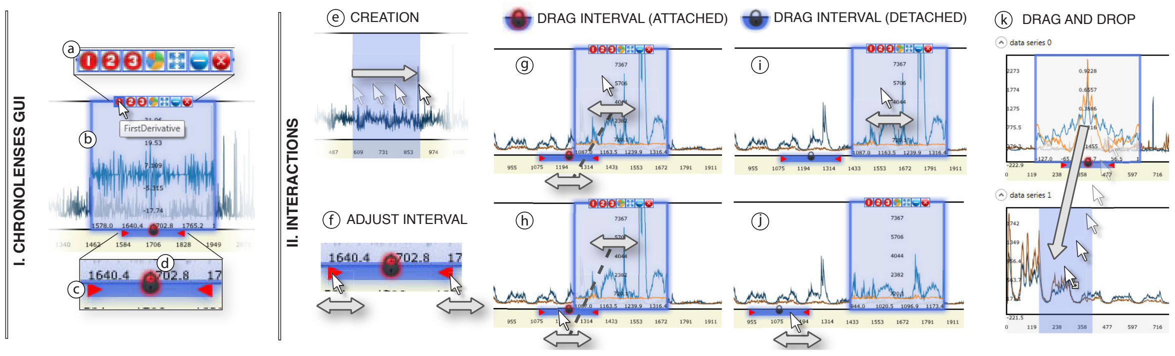


Figure 4.4: User interactions with a ChronoLens. (I) The interface of a ChronoLens: (a) toolbar enabling adjustment of the lens' main parameters, (b) tooltip (pops-up when hovering), (c) focus bar with resizing handles, (d) lock icon for detaching/reattaching the lens' frame from/to the focus bar; (II) the associated user interactions.

4.4.1 Creating an Analysis Pipeline

Advanced analysis pipelines are typically built by creating and progressively combining multiple lenses corresponding to the desired elementary operations E1-E4 defined in Section 4.3.2.

To create a basic lens, a user first specifies its parameters through the creation toolbar, for instance, choosing a *unary operator* such as *remove means*, and setting the *binary operator* to *null*. She then selects the input data to the lens by initiating a rubber-band selection on any time-series loaded in the charts panel. This selection, achieved thanks to a simple mouse drag (Figure 4.4-e), defines the time span of interest — the lens' focus.

Lenses that enable cross-data stream analyses obviously accept two input streams [6]. One stream is defined by the lens' focus, set as explained above. The second stream is provided as the output of another lens. This lens is considered as the *parent lens*; it feeds data to the *child lens*, that performs the cross-datastream analysis. The parent lens must exist in order to create a child lens.

The user can derive a child lens from the currently selected lens, by clicking on the *Create Child Lens* button in the creation toolbar or on the equivalent contextual menu item (Figure 4.1), after having set the desired parameters. The time span of a child lens must match that of the parent lens. This is to guarantee that input streams contain the same number of data points for binary operations, and thus make sure that cross-data stream operations are consistent. However, it would be possible to support unequal time spans between parent and

[6]As explained in Section 4.3.2, these streams can contain multivariate time-series. The binary nature of the transformation does not mean that input time-series are restricted to two variables. See Section 4.4.3 for more information about handling multivariate time-series.

child lenses, using time-series transformations that would make the number of data points between both streams consistent using, e.g., linear interpolation methods for time-series.

The created lenses are z-ordered using layers and can be piled up according to this ordering. Piling up lenses entails piping their operators as described in Section 4.3.2. In the current implementation, two lenses get actually piped when the upper lens' time span is fully contained within that of the lens underneath: the result of the latter then becomes the input of the former, as when composing Movable Filters by overlapping them [Fishkin and Stone, 1995].

To help the user keep awareness of the analysis pipeline, a pipeline (hierarchical) view is provided, depicting the currently selected lens' ancestors (Figure 4.1-c). The tree structure displays dynamic miniatures of the different lens branches, alongside labels detailing the operators associated with each lens in the pipeline. The current lens is visually emphasized with a thick blue border, and always corresponds to the top-left element in the pipeline view. By hovering over a lens either in the pipeline view or in the main chart view, the user can get an overall preview of all its ancestors. All lenses upstream are visually identified using an outer-glowing effect varying from green (closest ancestor) to yellow (furthest ancestor) in both views. The hovered lens is also emphasized with a blue glow effect (Figure 4.1-d).

4.4.2 Visual Exploration through Direct Manipulation

ChronoLenses, by virtue of their progressive analysis pipeline building process, are designed to facilitate step-by-step exploration through direct manipulation, while giving users freedom to edit intermediate steps and to browse different regions of a dataset simultaneously. Changes to a lens in the hierarchy are automatically propagated to its descendants, and all affected plots are visually updated accordingly.

Modifying Lens Parameters

To facilitate opportunistic discovery of regions of interest, we make it possible to redefine the focus region of a lens by simply dragging the lens frame (Figure 4.4-g) or the associated focus bar (Figure 4.4-h). The immediate redisplay of the newly computed result in both the chart view and the pipeline view enables quick access to different regions of the data stream, and thereby effectively supports visual exploration. The red arrow controllers placed on the focus bar (Figure 4.4-f) make it possible to adjust the lens' time span. The time span of all lenses downstream is adjusted accordingly so as to keep the number of data points consistent, as explained earlier. The user can also decide to move a lens to a different chart to further her analysis on another dataset, as illustrated in Figure 4.4-k.

The lens frame's width can be increased using the mouse wheel. The time span considered does not change, which means that resizing affects the $\mathcal{L}_{scale}$ parameter: the lens behaves as a magnifying glass[7]. Resizing on the vertical axis works similarly, except that the lens frame's height never changes and is always equal to the underlying chart's bounding box height. The y-axis origin can be adjusted by dragging the plot up and down inside the frame. An auto-adjust function is also provided, that automatically adjusts the y-axis' origin and scale so as to make the best use of available screen real-estate within the lens' frame as the user drags it.

Grouping Lenses

As mentioned earlier, the z-ordering of lenses that belong to the same analysis pipeline implicitly defines in what order transformation operations are applied to the data. The z-ordering and relative position of lenses that make a given analysis pipeline is important and should be kept constant during visual exploration via direct manipulation (when brushing the time-series plots with the lenses). Having to move each lens would be extremely cumbersome and would greatly impede the interactive visual exploration process. To avoid this, lenses can be grouped. All lenses that belong to the same group move synchronously as the user drags any one of them. The user can therefore create a compound lens set in which lenses are piled up in a specific order, thus creating a reusable analysis pipeline.

Grouping lenses can also be useful when analyzing multiple charts at a time, or when looking for seasonality. Such tasks require the interval between lenses (time lag) in the data stream to remain constant. By defining a group consisting of the different lenses to be moved synchronously, the user can keep the time lag between these constant.

A lens can be added to the currently selected group through the contextual menu (Figure 4.1-f). All lenses belonging to the same group as the currently selected one are outlined with a green stroke. Group membership is also reflected in the tree view, where the group number is explicitly specified alongside the lens miniatures.

Dealing with Visual Clutter

The instantiation of many lenses can quickly lead to visual clutter. Lenses that are part of the same analysis pipeline will often overlay each other, either partially or fully occluding one another. This can be a problem when the user wants to visualize intermediate computational steps of the pipeline. Occlusion problems can also arise when magnification lenses (any lens with $\mathcal{L}_{scale} > 1$) look at time spans that are disjoint but close to one another.

[7]The above-mentioned blue bar still shows the original region of interest.

To address these issues, we introduce a mechanism that enables the user to detach the lens frame from its associated focus bar, thereby giving her control over where to display the lens' output, independently of the actual input region of interest (focus bar location). Figures 4.4-g to -j illustrate this mechanism. To detach a lens from its focus bar, the user simply clicks on the lock icon (Figure 4.4-d) to unlock the lens from the focus bar. In that mode, dragging the lens frame does not affect the focus interval (Figure 4.4-i), making it possible to space out lenses or, on the contrary, move them closer to facilitate comparison. In the same manner, dragging the focus bar only affects the input time span, the lens frame remaining in place (Figure 4.4-j). Wherever a lens and its focus bar are positioned, locking them again will keep the position offset constant (Figure 4.4-g,h). When hovering over a lens frame or a focus bar, both are emphasized with a glow effect making it easier to identify what focus bar is associated with what lens, and conversely.

Other mechanisms that help manage visual clutter include *delete* and *minimize* buttons in the toolbar (Figure 4.4-a). The former deletes the lens, the latter hides the lens' frame, but not its focus bar. The bar remains visible so as to keep the user aware of its existence. At all times, double-clicking it makes the lens' frame visible again.

Multi-Focus Exploration

As described in the previous design study (Chapter 3), time-series are amenable to multi-focus and multi-scale navigation, where a user might be interested in behaviors and event patterns that occur in different data segments at various time scales. Magnification lenses, a common practice to facilitate such exploration [Elmqvist et al., 2010b], can easily be instantiated in ChronoLenses: any lens with $\mathcal{L}_{scale} > 1$ is a magnification lens that enables drilling down into the data. However, magnification lenses pose some problems from an interaction perspective [Appert et al., 2010; Pietriga et al., 2009]. In-place magnification of the focus region means that the immediate surroundings of the region of interest will be occluded if the lens is simply overlaid on the original data, hiding potentially valuable information and hindering navigation. Smooth transition between the magnified focus and surrounding context can be achieved using spatial distortion [Kincaid, 2010], but the introduced deformation has a cost in terms of legibility and interpretation of the visualization.

An alternative drill-down method is to stack multiple multi-scale views as a user explores the data, such as [Javed and Elmqvist, 2010; Javed et al., 2012]. The technique can also be seamlessly integrated with ChronoLenses. A user can create a new empty chart panel in the main view. Then, dragging-and-dropping an existing lens into this empty panel duplicates the data within the span of the lens and displays it in this new panel. The visual representation is stretched horizontally to fill the panel, thus providing a magnified view of the lens' content that

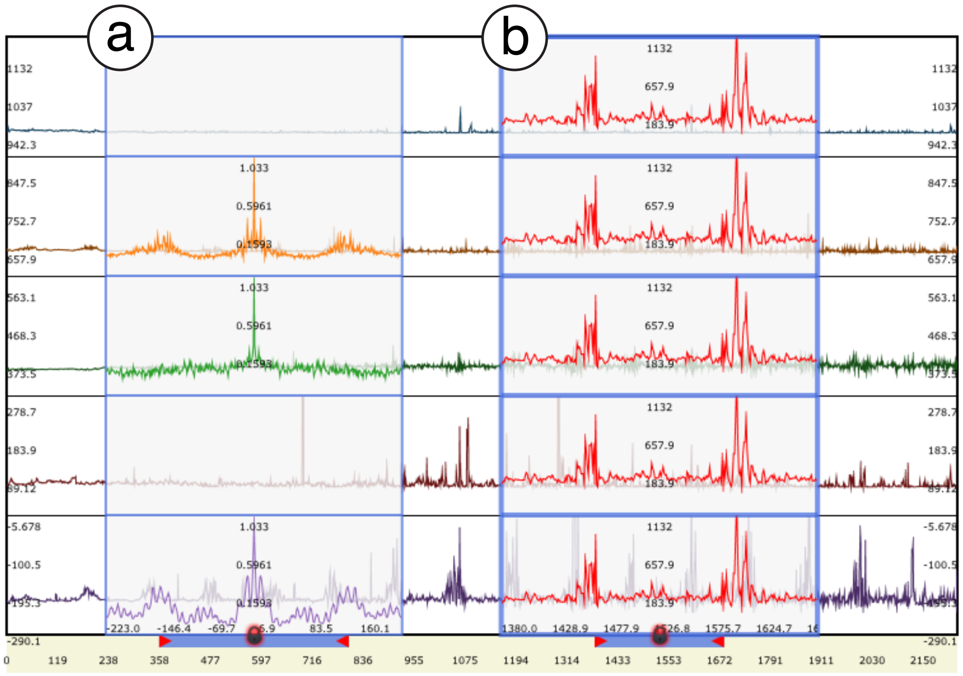


Figure 4.5: ChronoLenses applied to multivariate data: (a) auto-correlation for three series out of five; (b) point-wise aggregated maximum showing the resulting plot (common to all series) overlaid on top of each variable's plot.

is dynamically updated whenever that lens moves. Figure 4.6 illustrates this technique, with chart (b) providing a magnified version of the region seen through the lens in chart (a). Lenses can be instantiated on this new panel, enabling users to drill-down recursively and build a truly multi-scale view hierarchy in which lenses at any level can be freely adjusted. Lenses that form this hierarchy are of course not limited to simple magnification and can apply to the data any of the elaborate transformations enabled by the ChronoLenses operators.

4.4.3 Multivariate Time-Series

In ChronoLenses, multivariate datasets can be visualized either overplotted in the same chart (Figure 4.1, top chart) or separately, stacked on top of one another (Figure 4.1, bottom chart). When applying a lens to a multivariate data stream, each series gets processed independently, except for some operators that aggregate the data, such as *point-wise minimum* or *maximum*. In that case, the output is a univariate stream, that can be duplicated and overlaid in all charts. We distinguish aggregated operators from other operators by color-coding in red all the replicas of the unique plot resulting from data aggregation (Figure 4.5-b) as opposed to the one-to-one color mapping used when the different variables are processed separately (Figure 4.5-a).

To focus on a subset of variables, the user can filter out series that she does not want to be considered in the computational pipeline by setting the filter operator $\mathcal{L}_{filter}$ accordingly. To do so, she can either select all the series to be taken into account in the corresponding menu accessible from the lens toolbar, or specify the list in the property panel (Figure 4.1-e), by entering the corresponding textual expression using a very simple syntax. When filtered out, streams are neither computed nor rendered, as the first and fourth streams in Figure 4.5-a. Note that when a lens defines an aggregated operator, filtering out a stream has an impact on the result, as the stream is no longer considered as an operand. A full control over what data gets processed is offered to users, making the analysis process highly flexible and customizable.

4.5 Use Case Scenarios

In this section, we illustrate how ChronoLenses can be used with two use cases involving real datasets. Those use case scenarios were developed based on our interviews with two domain experts: one was from astronomy and the other was from computer networking. Both experts have worked in their fields for many years, and conducted extensive time-series analysis and modeling as their daily working activities. During the interview session, we first introduced the ChronoLenses visualization tool, then asked them to use it for exploring their own datasets, and finally solicited their comments. Based on their feedback and our observations, we derive the following use case scenarios to best reflect the actual usage of ChronoLenses in a realistic environment.

4.5.1 ALMA Observatory Usage Scenario

The Atacama Large Millimeter/submillimeter Array[8] (ALMA) is a single telescope (under construction in the Chilean Andes) that will eventually be composed of 66 high-precision antennas. Observations are based on the principle of interferometry: a source in the sky is observed by at least two antennas; the signals (radio waves) captured by each antenna are then combined by a central computer called the correlator to form images suitable for performing scientific analysis. Time-series visualization are used by both operators and astronomers for a variety of tasks, ranging from checking some of the thousands of monitor points in the system to performing scientific data quality assurance during observations. In the following, we focus on one example where ChronoLenses can help users in their daily task.

When combining the signals coming from the different antennas taking part in a given observation, the correlator must know the length of the path traveled by the signal through the

[8] http://www.almaobservatory.org.

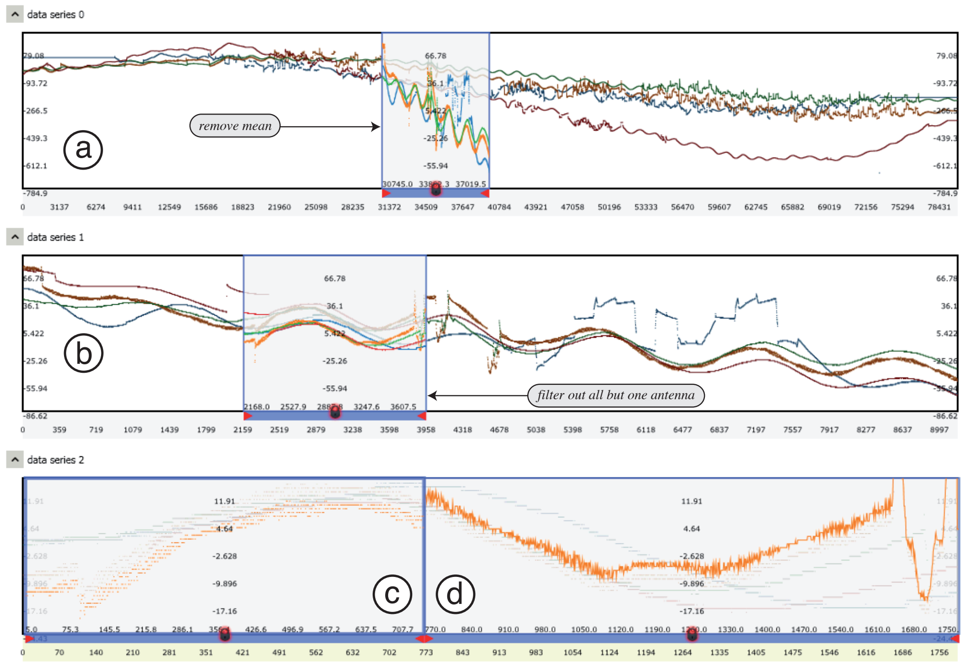


Figure 4.6: Exploring ALMA Line Length Correction Stretcher Voltage plots for four antennas: (a) two-day overview at a sampling rate of one second; (b) magnification of the 5 hours seen through the *remove mean* lens applied in (a); magnification of the 50 minutes for a single antenna seen through a filtering lens, rendered in scatterplot mode (c) and line-plot mode (d).

fiber optics cables with an accuracy of hundredths of a millimeter. A round-trip laser signal gets sent to all antennas in order to continuously monitor the length of the optical fibers, as the latter can expand and contract due to temperature variations. These changes in path length must be compensated in real-time. This is achieved by the Line Length Correction (LLC) system, which ensures that path lengths are all stabilized to about 1 micron. Operators are interested in monitoring the LLC stretcher voltage for each antenna, and observing potential deviation of an antenna's LLC stretcher voltage compared to others.

From a time-series visualization perspective, different things can be happening at different time scales: from several days to a few minutes. Efficient multi-scale and multi-focus visualization, as enabled by the $\mathcal{L}_{scale}$ operator and the variation on StackZoom [Javed and Elmqvist, 2010] implemented in ChronoLenses, is thus an essential feature. Figure 4.6-a plots voltage against time for 2 days at a sampling rate of 1 second.

Starting from this 2-day overview, the operator is first interested in finding out whether all antennas are behaving normally or if one or more are somehow deviating. All antennas are expected to behave more or less similarly. Direct visual comparison between the plots, overlaid or stacked, is sufficient to see the deviation of one antenna during the second day (Figure 4.6-a).

To better see the smaller and shorter fluctuations, the operator creates a lens spanning a 5-hour period, with $\mathcal{L}_{unary}$ operator *remove mean* (Figure 4.6-a). The lens is dragged and dropped to create panel 4.6-b. The content of that panel is a stretched version of what is seen through 4.6-a's lens. The operator can clearly see that the fluctuations are in phase, which implies that they all come from a single source. She could then plot various monitoring points simultaneously based on system or environmental components likely to cause these fluctuations and create lenses defining a cross-correlation $\mathcal{L}_{binary}$ operator, eventually tracing the source to temperature fluctuations in the local oscillator room.

Finally, zooming in further to display just 50 minutes (Figure 4.6-c), the operator can see some odd features in the signal for one of the antennas. She filters out antennas that behave normally using a lens defining the appropriate $\mathcal{L}_{filter}$. She also switches from a scatterplot to a line-plot rendering inside the lens focus (Figure 4.6-d), revealing fast oscillations at a much lower scale that occur on top of the slow ones observed earlier for this particular antenna. This antenna-specific issue eventually gets traced to a device repeatedly turning on and off in the antenna.

4.5.2 Peer-to-peer Network Fluctuation Analysis

We consider a network system manager who is trying to optimize bandwidth usage on a centralized peer-to-peer system. She uses line graphs of the visiting population of two different video-on-demand channels. The relative evolution through time of the two channels, temporal distances, and the activity load before and after peak events are important clues that the analyst can rely upon, assisting her in the decision making process. For instance, identifying regular activity patterns helps better predict future fluctuations and therefore better pinpoint the needs for server pool optimization. Being able to spot anomalies such as overloads in their context also helps identify the potential causes, and plan for technical solutions.

Comparing the Evolution of Two Time-series

Our dataset consists of the visiting population of two different channels over a month, sampled from the server every 10 minutes (Figure 4.7). The overall yearly or monthly trends of the channels might be different. For instance, one might be rising much faster than the other, or one might fall while the other is rising a bit. In this first scenario, the user is rather interested in the smaller scale fluctuations, comparing evolution and identifying potential correlations at this scale.

The user first focuses on a single channel, looking for repeating patterns. To do so, she compares differentiated data at different times by performing the following steps: 1)

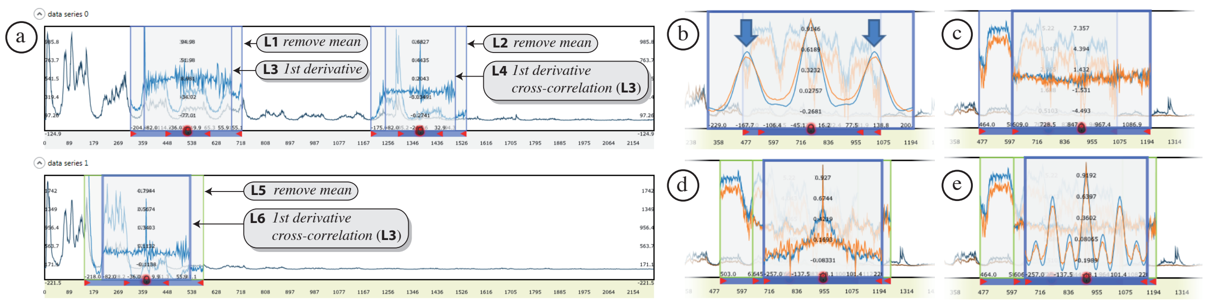


Figure 4.7: Analysis of a month of the visiting population of two P2P video-on-demand channels: (a) analysis of the evolution of the correlation within and between the channels (see Section 4.5.2); (b) looking for seasonal pattern lag value, the blue arrows show a strong correlation at a delay of 1440s (=24h) and (c)-(d) comparing the delayed time spans (see Section 4.5.2).

remove mean, to make the data more stationary and thus more amenable to comparison for our purposes; 2) compute the first derivative (or differencing) of those; and 3) perform the cross-correlation between the two for comparison. The pipeline that supports this type of analysis, described below, is composed of four lenses, as shown in Figure 4.8.

The user creates lenses L1 and L2, with the $\mathcal{L}_{unary}$ operator set to *remove mean* (step 1). She then creates L3 ($\mathcal{L}_{unary}$ operator set to *1st derivative*) and places it on top of L1 (step 2). Its child lens L4 gets positioned on top of L2 with operators $\mathcal{L}_{unary}$ and $\mathcal{L}_{binary}$ set to *1st derivative and cross-correlation* respectively (steps 2-3). In the end, L4 outputs the correlation of the data below it (1st derivative of L2) and its parent's output (L3). The discovery of recurring patterns usually requires performing computations on data with parameterized-yet-fixed lags in time. Here, we use lens groups — that make all member lenses move synchronously — to build dynamically-parameterized operators enabling efficient exploration of the effect of different time lags and time spans via direct manipulation.

Grouping L1 and L3 on the one hand (L1-3), and L2 and L4 on the other hand (L2-4) makes it possible to drag one or the other focused intervals along time and look for strong correlation by observing the result of L4. When such a correlation is identified, as in Figure 4.7-a, the user can group L1-3 and L2-4 altogether and drag them. If L4 exhibits a stable plot while dragging, then a recurrent trend is identified. If on the contrary the plot varies, direct manipulation with immediate feedback helps better understand the correlation variation before and after peaks of interest, whereas it might be more difficult to apprehend when computing automatic pattern detection.

When the analyst has located a pattern of interest, she can focus on the second channel to look for correlations. She creates pipeline L5-6 similar to group L2-4 using the data from the

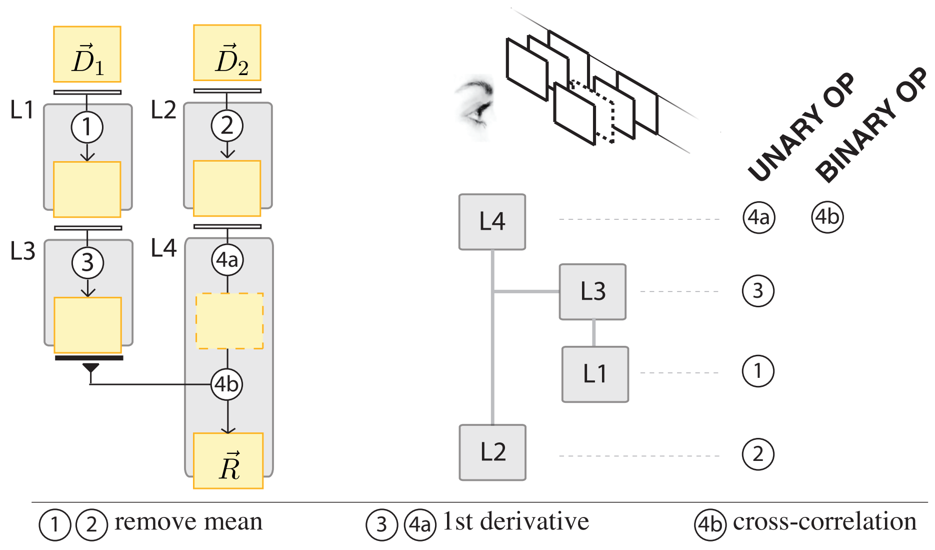


Figure 4.8: Comparing the evolution of two time-series using ChronoLenses. Left: the analysis pipeline; right: the corresponding lens hierarchy.

second channel as input: L5 (remove mean) and L6 (L3's child placed on top of L5 with 1st derivative and cross-correlation operators), as depicted in Figure 4.7-a.

Different analysis tasks can easily be performed with the current set of lenses: (1) drag the group L5-6 until the plot is like, or the opposite of, L2-4 to identify correlations between the two series (see Figure 4.7-a); (2) group L2-4 and L5-6 so that they preserve their relative distance while dragging one or the other in order to compare the correlation fluctuations by simultaneously looking at the results of lenses L4 and L6. If those behave the same, then a similar evolution is identified and the relative lag is known. If they always behave the opposite, there is an inverse causality; and (3) by extending the group to L1-3 and dragging the whole, the analyst can also explore if a pattern is preserved in time. The user can also change the lenses' operators at any time during exploration, as for example changing L3, L4 and L6's $\mathcal{L}_{unary}$ operator to *2nd derivative* and looking for insights when more lag is involved (20 minutes in our example).

Comparing different Time Spans in the same Series

Discovering seasonality is another important task in time-series data analysis. It helps better predict future fluctuations. In this second part of the scenario, our user is interested in finding seasonal differencing on the two channels simultaneously, in order to gain a better understanding of what characterizes the typical data streaming cycle, if such a cycle exists. Gaining this knowledge will eventually help develop strategies to optimize bandwidth usage, but also assist in spotting abnormal behavior when it occurs.

Box-Cox Transforms can be used to perform a variance stabilizing transformation. Seasonal differencing parameterized with a time lag can then be applied to find out if the non-stationarity of the original time-series is removed [Madsen, 2007], but this requires to know the lag value in advance. The identification of an appropriate lag value to perform seasonal differencing can be made easier by building the following pipeline: an *auto-correlation* lens (L1) on top of a *Box-Cox transform* lens (L2). A strong correlation is observed at a delay of approximately 1440 minutes = 24 hours (see Figure 4.7-b).

The analyst deletes L2 and furthers her exploration as she creates L2's child lens L3 with *Box-Cox transform* and *subtraction*, synchronized so that L3 is always 24 hours ahead of its parent (Figure 4.7-c). An *auto-correlation* lens is then applied on top (after the differencing operation) to evaluate stability of the series. At this position in the original time-series (Figure 4.7-d), there does not seem to be any significant seasonal pattern, as high correlation values exist only at zero-delay. But moving the lens group along the timeline, one pattern is eventually discovered (Figure 4.7-e), calling for further exploration.

4.6 Discussion

The ChronoLenses system aims at facilitating the creation of customized analysis pipelines for easy exploration and navigation through time-series datasets. Our initial prototype, although already offering a large set of functionalities, still has limitations. It could be improved in several ways, as discussed in this section.

In its current implementation, the ChronoLenses interface does not support effective layer management for the z-ordering of lenses that pile up. Although the grouping of lenses keeps the z-ordering constant, when not grouped, selecting a lens puts it on the topmost layer. This might be annoying when dealing with complex pipelines as clicking on a lens only to get information about it or even by mistake can alter the pipeline. Making layer management more stable and more easily configurable through a dedicated synchronized view would reduce the need for advanced planning when building the pipeline and would aid keep an accurate mental map of the data flow.

Similarly, the lens hierarchy treeview could be enhanced. As is, the hierarchical view dynamically changes as the user selects a different lens. This might be distracting and difficult to interpret. Moreover it is not interactive. There is an opportunity for improvement here, as an additional and complementary representation of the analysis pipeline could be used as an alternative means of performing exploration tasks if enriched with interactive capabilities. In particular, allowing for the creation of lenses, duplication of analysis branches, or other changes to the analysis pipeline from this alternate view would make it easier to build up and manage complex pipelines.

Another limitation of the current implementation is related to the positioning of lenses. When displaying an overview of large time-series, ChronoLenses only draws a subset of all points (subsampling). Magnifying the data through a lens enables the display of more detail in context. However, as the lens' magnification factor increases, small movements of the lens translate to larger jumps in terms of time span observed in the lens' focus. For instance, given a magnification factor of 10x, moving the lens by 10 pixels will move the time span observed through the lens by 100 equivalent pixels at that zoom factor. The problem also exists when controlling the time span seen in a plot through a lens observing a lower scale version of that plot. Adapting one of the high-precision magnification lens techniques introduced in Appert et al. [2010] would solve this issue, enabling both fast repositioning and precise selection within the lens focus.

A related issue is that our lenses occlude the area immediately surrounding the region of interest, as basic magnifying lenses do. SignalLenses [Kincaid, 2010] address this issue by distorting the representation in a bounded transition region between the lens focus and the context. We chose not to rely on spatial distortion because of the issues it raises in terms of making accurate analytical comparison and interpretation. However, there are other solutions to this problem, such as speed-coupled Sigma Lens focus targeting techniques [Pietriga et al., 2009]. For instance, the Speed-coupled Blending Lens technique consists in coupling the lens focus' opacity to its speed, smoothly fading out the lens as speed increases and smoothly fading it back in when it comes to a stop, having reached its target position. The technique addresses the problems of visual occlusion and would be relatively easy to implement in ChronoLenses as we already support translucency in lens renderings. However, potential issues related to visual interference between the layers during lens repositioning would have to be investigated.

4.7 Summary

This chapter has presented the second design study with the focus on visual exploration of data values, more specifically, in numerical time-series datasets. We have approached the challenges in preforming advanced time-series analysis tasks that require dynamically deriving new data values and trying different functions and parameters to reuse and modify existing values in an iterative manner. The proposed visualization technique, ChronoLenses, relies on the metaphor of lenses that compute on-the-fly data transformations in place. ChronoLenses allow users to progressively build elaborate analysis pipelines by interactively compounding elementary operations, thus supporting complex user-defined exploratory analysis tasks. For example, in one of the use cases, the user could perform fluctuation analysis and identify correlations at this scale by simply manipulating different lenses and connecting them for advanced transformations. In summary, this chapter has made the following contributions:

- tasks and design requirements for preforming elaborate time-series analysis and exploration,
- a generic lens-based visualization framework for constructing advanced analytical pipelines containing a series of data transformations with customized operators, and
- a design of ChronoLenses as an implementation of the framework enhanced with direct manipulations and a range of other features, to demonstrate its usefulness and effectiveness in visual analysis of time-series.

It is worth noting that the concept of ChronoLenses visualization framework is generalizable to other types of data. Single-datastream transformation, cross-datastream computation, and their combinations to generate advanced operations are widely used for data analysis in almost every domain, even beyond the exploration of numerical data values. For example, in the analysis of temporal media such as video, image processing techniques can be applied on a single frame (e.g., color correction filters and sharpening) or across frames (e.g., feature matching and similarity measuring) in the video sequence. More importantly, the dynamic and flexible features of the user-generated analysis pipelines facilitate the fundamental iterative exploratory behaviors of users analyzing datasets (as summarized in Keim et al. [2008]'s visual analytics mantra), which are commonly demanded in real-world applications.

In the next chapter, we will move away from the focus on data values to explore multi-focus visualizations that assist the visual exploration and comparison of data structures, that logically relate data entities to one another, which is the second data feature we have concerned and discussed in the space of data for information visualization (see Section 1.2.2).

Part III

Focusing on Data Structures

Chapter 5

DAVIEWER: FACILITATING VISUAL COMPARISON OF DISCOURSE TREES

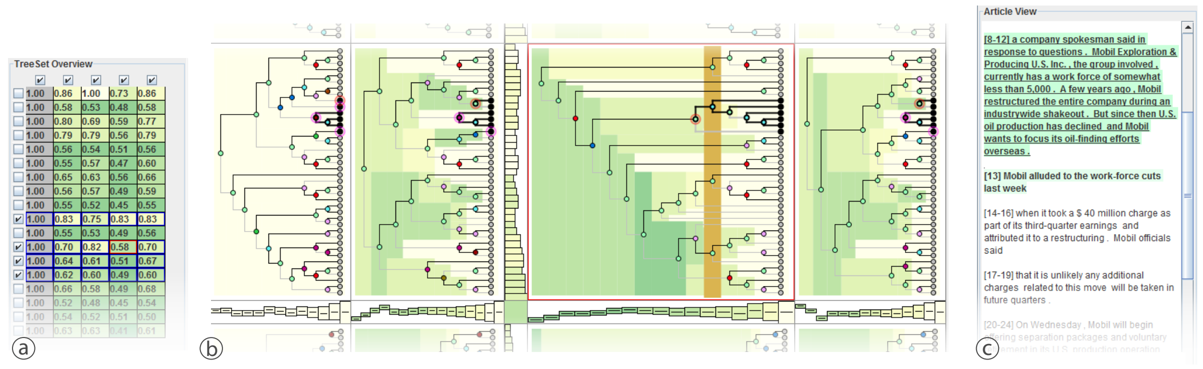
Language is the dress of thought.

Samuel Johnson

Part I (Chapter 3 and Chapter 4) presented two design studies of developing interactive visualizations supporting the exploration of data values. Moving forward, this chapter[1] describes a design study on techniques facilitating the visual exploration of data structures, i.e., logical relationships that connect individual data items. More specifically, we focus on visual comparison of a collection of hierarchical structures, by situating the design study in the application context of visualizing discourse trees from Natural Language Processing (NLP). A discourse tree is the output of a discourse parser, which can represent the organization of a document based on a rhetorical tree. The discourse structures are the foundation of many algorithms in computational linguistics, such as text summarization [Marcu, 1999], question answering [Chai and Jin, 2004], and dialog generation [Prendinger et al., 2007].

However, accurate discourse analysis remains a challenge due to the complex and nuanced nature of language. Computational linguistics researchers currently rely on manually exploring and comparing the discourse trees to get intuitions for improving parsing algorithms. This process is often very tedious based on their current practice using the textual outputs of trees (Figure 5.2-b). In this chapter, we present DAViewer (Figure 5.1), an interactive visualization tool for assisting computational linguistics researchers to browse, compare, evaluate and

[1]Main portions of this chapter were previously published in Zhao et al. [2012a] © 2012 IEEE. Reprinted, with permission, from: Jian Zhao, Fanny Chevalier, Christopher Collins, and Ravin Balakrishnan, Facilitating Discourse Analysis with Interactive Visualization, IEEE Transactions on Visualization and Computer Graphics. Thus any use of "we" in this chapter refers to Jian Zhao, Fanny Chevalier, Christopher Collins, and Ravin Balakrishnan.



	✓	✓	✓	✓	✓
☐	1.00	0.86	1.00	0.73	0.86
☐	1.00	0.58	0.53	0.48	0.58
☐	1.00	0.80	0.69	0.59	0.77
☐	1.00	0.79	0.79	0.56	0.79
☐	1.00	0.56	0.54	0.51	0.56
☐	1.00	0.55	0.57	0.47	0.60
☐	1.00	0.65	0.63	0.56	0.66
☐	1.00	0.56	0.57	0.49	0.59
☐	1.00	0.55	0.52	0.45	0.55
✓	1.00	0.83	0.75	0.83	0.83
☐	1.00	0.55	0.53	0.49	0.56
✓	1.00	0.70	0.82	0.58	0.70
✓	1.00	0.64	0.61	0.51	0.67
✓	1.00	0.62	0.60	0.49	0.60
☐	1.00	0.66	0.58	0.49	0.68
☐	1.00	0.52	0.48	0.45	0.54
☐	1.00	0.54	0.52	0.51	0.50
☐	1.00	0.63	0.63	0.41	0.61

[8-12] a company spokesman said in response to questions . Mobil Exploration & Producing U.S. Inc. , the group involved , currently has a work force of somewhat less than 5,000 . A few years ago , Mobil restructured the entire company during an industrywide shakeout . But since then U.S. oil production has declined and Mobil wants to focus its oil-finding efforts overseas .

[13] Mobil alluded to the work-force cuts last week

[14-16] when it took a $ 40 million charge as part of its third-quarter earnings and attributed it to a restructuring . Mobil officials said

[17-19] that it is unlikely any additional charges related to this move will be taken in future quarters .

[20-24] On Wednesday , Mobil will begin offering separation packages and voluntary retirement in its U.S. production operation

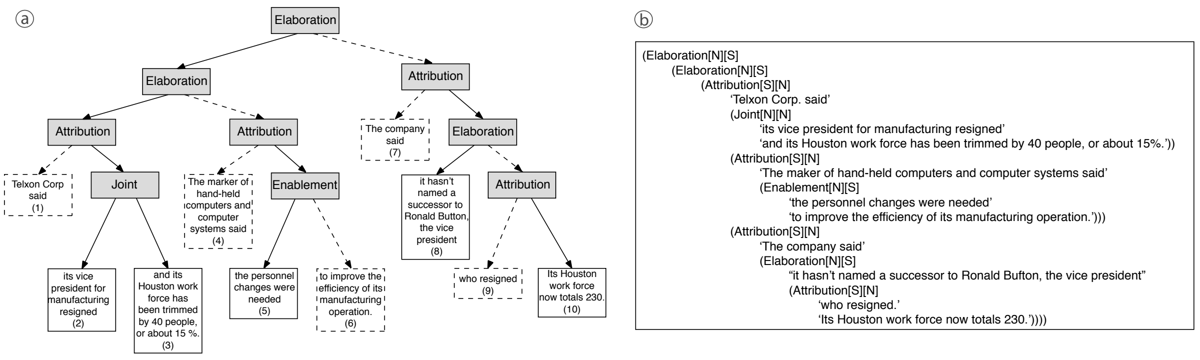
Figure 5.1: The DAViewer, a visualization system for discourse analysis: (a) The overview panel shows similarity statistics about a collection of discourse trees; (b) the detail panel shows the full structure of the discourse trees; (c) the text panel is coordinated with the visualizations to allow detailed analysis of the correspondence of rhetorical structure and text content. A video demo of the system can be downloaded here.

annotate the results of discourse parsers through multi-focus visual exploration. DAViewer has been developed through an iterative user-centered design process with domain experts. A preliminary deployment study of the prototype was conducted with an expert user over a period of four weeks in realistic work settings. The results indicate that DAViewer dramatically sped up some comparison and analysis tasks which were otherwise difficult to carry out by the user in the past.

5.1 Motivation

Natural Language Processing has become a vitally important area of computer science research, as the results of research in this field are quickly put to wide use in systems such as text categorization, automatic translation, topic extraction, and summarization. A large portion of current efforts in this area involves computational linguistics researchers working to improve statistical parsing algorithms to gain ever higher accuracy and recall. Of particular interest is the subfield of automated discourse analysis, which aims to analyze the semantic structure and relationships within a text document. While parsing a sentence for its grammatical structure may be familiar to many readers, discourse parsing crosses sentence boundaries, extracting relationships within an entire document.

In order to develop robust discourse parsers, researchers typically rely on a corpus — a large collection of human labeled documents — as a reference (called a *gold standard*) for training and evaluating algorithms. Several techniques have been proposed to make discourse parsers under a widely accepted framework, Rhetorical Structure Theory (RST) [Mann


ⓐ
Elaboration
Elaboration
Attribution
Attribution
Attribution
The company said (7)
Elaboration
Telxon Corp said (1)
Joint
The marker of hand-held computers and computer systems said (4)
Enablement
it hasn't named a successor to Ronald Button, the vice president (8)
Attribution
its vice president for manufacturing resigned (2)
and its Houston work force has been trimmed by 40 people, or about 15 %. (3)
the personnel changes were needed (5)
to improve the efficiency of its manufacturing operation. (6)
who resigned (9)
Its Houston work force now totals 230. (10)
ⓑ

```
(Elaboration[N][S]
        (Elaboration[N][S]
                (Attribution[S][N]
                        'Telxon Corp. said'
                        (Joint[N][N]
                                'its vice president for manufacturing resigned'
                                'and its Houston work force has been trimmed by 40 people, or about 15%.'))
                        (Attribution[S][N]
                                'The maker of hand-held computers and computer systems said'
                                (Enablement[N][S]
                                        'the personnel changes were needed'
                                        'to improve the efficiency of its manufacturing operation.')))
                        (Attribution[S][N]
                                'The company said'
                                (Elaboration[N][S]
                                        "it hasn't named a successor to Ronald Button, the vice president"
                                        (Attribution[S][N]
                                                'who resigned.'
                                                'Its Houston work force now totals 230.'))))
```

Figure 5.2: Example of a typical discourse tree structure: (a) node-link representation of the hierarchical binary tree and (b) indented text format.

and Thompson, 1988], which represents the organization of text as a tree structure after dividing it into non-overlapping text chunks (Figure 5.2-a). While such an abstracted representation offers helpful support for analysis, there are no adequate visual exploration tools to assist NLP researchers in discourse studies: in practice, researchers display or print out static representations of the discourse tree structures in the form of indented text chunks (Figure 5.2-b). This makes the exploration and comparison process tedious, and particularly inefficient for the task of comparing the outputs of several variations of an algorithm.

The visualization tool proposed in this chapter, DAViewer (Figure 5.1), is designed as an interactive tool to augment the manual analysis process, supporting the verification of hypotheses and discovery of insights about existing parsers in order to inspire the development of improved parsing algorithms. In close collaboration with computational linguistics experts at every stage of the development, we implemented and iteratively refined a functional prototype to address our target users' needs. The resulting interface is built around a table of discourse tree visualizations, interactively coordinated with other visualization components including the texts under analysis and detailed information about selected objects. DAViewer is designed in such a way that computational linguistics researchers can actively integrate the visual exploration of intermediate results into their research process and use these results to further develop and refine parsing algorithms in discourse studies.

5.2 Background

In general, the work presented in this chapter is related to visualization of tree structures, more particularly, the visual comparison of trees. Previous techniques on those topics have been introduced in Section 2.1.3. In this section, we discuss literature that is specifically related to this chapter, including the background of discourse analysis and visualization techniques applied in computational linguistics.

5.2.1 Discourse Analysis

Discourse analysis in the NLP community is mostly driven by the RST framework [Mann and Thompson, 1988]. In RST, the discourse structure is a binary tree [2] built from non-overlapping text chunks called elementary discourse units (EDUs). The EDUs are segmented from the text document, and serve as the leaves of the discourse tree. Then the discourse parsing algorithm successively combines EDUs to create internal nodes, each of which corresponds to a rhetorical relation between its branches (e.g., *Attribution*, *Cause*, etc.). Hernault et al. [2010] describe the 18 relations which are used in the parsers tested in this research, and widely used by the discourse analysis community. Under each relation, the children can be either *nucleus* or *satellite*. With respect to the parent relation, the text associated with a nucleus branch is considered more prominent or important than the text associated with a satellite branch.

There are two ways to combine the segmentation and parsing algorithms. They can be coupled, so that the segmentation of the text into EDUs and the creation of the parse tree are interdependent and co-optimized. Or, the segmentation can take place in a separate, initial step, and the parsing algorithm is then forced to build the tree from these EDUs. The second (decoupled) method is more common and is used in this work. By forcing the EDUs to be the same across parsing algorithms, it allows for easier structural comparison of the trees generated by different parsers.

A popular corpus used in this area of research is the RST Discourse Treebank (RST-DT) [Carlson et al., 2001], which consists of 385 documents transcribed from the *Wall Street Journal* and annotated under the RST framework. Figure 5.2-a shows a typical rendering of a discourse tree, generated over the text of an article, with rhetorical relations shown in gray-filled boxes, and EDUs depicted with white boxes labeled with sequence numbers (i.e. the EDU's position as it appears in the text). Solid or dashed lines indicate whether the branch (or EDU) is nucleus or satellite respectively.

Several attempts have been made to develop discourse parsers using the RST framework. Many discourse parsers rely on a bottom-up approach for the tree building, i.e., linking EDUs and internal nodes with a parent relation, level by level until encountering the single top root node. The HILDA discourse parser [duVerle and Prendinger, 2009], further improved by Feng and Hirst [2012], is an instantiation of this approach to parsing, and is used in this work. One

[2] While the original tree is not necessarily binary, a widely accepted practice in discourse analysis consists of converting an n-ary tree into a binary tree [Hernault et al., 2010].

of our domain experts focus on improving parsers built with the HILDA parser as a starting point.

Despite the efforts of computational linguists, the generation of highly accurate discourse structures over a text document remains an open research question in NLP. In order to get a better understanding of the flaws and strengths of existing and newly created algorithms, the common practice consists of a close analysis of the discourse trees generated by different parsers: the comparison of a generated tree with the gold standard reveals how the obtained result differs from the optimal solution; and the comparison between generated trees helps analysts to identify the impact of tuned parameters of the same algorithm, or differing performances of multiple algorithms. In particular, linguists are interested in answering the following questions:

Q1. At a given intermediate level in the discourse tree, is the text separated into meaningful groups (chunks)? Do the nucleus branches capture the prominent content of the text?

Q2. Where are the errors in the generated discourse tree? Do errors at low levels propagate to upper levels of the tree structure?

Q3. What are the structural differences between branches generated by two parsing algorithms over the same EDUs?

Q4. Does the algorithm generate common parsing structures (or errors) across all the documents in the corpus?

Q5. How consistently does each algorithm perform across documents in the corpus? Which types of documents are more problematic?

With the lack of efficient visualization systems to address their needs, linguists struggle to answer such questions effectively, as they are currently working with a collection of indented text encoding the tree structure, as shown in Figure 5.2-b.

5.2.2 Visualization and Computational Linguistics

There is a growing body of research in the area of text visualization, with most efforts aimed at content analysis — revealing keywords in a document, topics in a corpus, and changes in streaming text data. In this work, we are interested in a specific subset of language-related visualizations: those works whose aim is to improve our understanding of linguistic phenomena or computational linguistic algorithms.

Visualization has been used to answer fundamental questions in linguistics. For example, the Dichronlex diagram is used to reveal changes in language constructs over time [Therón et al., 2011]. Pilz et al. [2008] use multidimensional scaling to plot the similarities of spelling variants to understand the propagation of spelling changes through time and geography. Structured Parallel Coordinates can be used to understand common patterns in corpus linguistics [Culy et al., 2011]. Finally, Constellation is a graph-based visualization targeted at helping refine algorithms that determine semantic networks [Munzner et al., 1999].

More relevant to this work are visualizations designed to better understand, and even to improve, computational linguistic algorithms. The DerivTool is designed for computational linguistic researchers to interactively correct problems in a translation system by directly editing the translation model learned from training data [DeNeefe et al., 2005]. The lattice visualization of Collins et al. [2007] and the Chinese Room visualization of Albrecht et al. [2009] both use visualization to reveal a collection of closely ranked translation hypotheses considered by an automated translation algorithm, and allow a user to select the most reasonable alternative. Finally, the Bubble Sets visualization was designed to support machine translation researchers, using a participatory approach similar to the one we adopted here [Collins et al., 2009b]. Bubble Sets are overlays on parse trees to improve their usefulness in the task of diagnosing translation errors in order to improve translation algorithms. Similarly, the discourse tree visualizations of DAViewer are designed to support the discovery of errors in discourse trees in order to inspire improvements to discourse parsing algorithms.

5.3 Design Process

To design and develop DAViewer, we employed a user-centered approach over the course of four months. Here, we describe our methodology and the design requirements.

5.3.1 Methodology

We followed a standard user-centered design process by involving two experts: an expert in both computational linguistics and visual design, and a researcher (technical expert) from a university computational linguistics group whose focus is discourse analysis. The first expert has conducted research in the intersecting filed of both NLP and visualization for over 6 years; and the second expert has studied computational linguistics and machine learning techniques for 4 years.

Through regular consultations with our experts, we gathered and refined a list of requirements, and built a series of prototypes: over the course of four months, we maintained a weekly meeting with the experts, during which they were presented with the latest prototypes for feedback on further requirements using several methods including interviews, meetings, observations, exchanging emails, and phone calls. A series of prototypes were deployed, including sketches, paper prototypes, an initial implementation running on a small dataset, a refined high-fidelity prototype with full functionality, and a deployable system. Details of the four stages of our design process follow.

Problem identification and quick prototyping. The first step was aimed at identifying the problems and challenges faced by the target users, along with the exploration of possible design alternatives to address their needs. Through a series of interviews focusing on the frustrations and issues NLP analysts encounter in daily research life with their current workflow, we derived a set of basic functionalities and design requirements for a visual exploration tool (Q1-5 in Section 5.2.1, and Section 5.3.2). Several design alternatives were then explored, critiqued, and refined with the help of the experts using sketches and paper prototypes.

Initial implementation with core functionality. After agreeing on a general design concept and priorities, the second phase consisted of implementing, iteratively testing, and refining an initial interactive prototype that included a number of key functions. We conducted this iterative process over a period of six weeks, using a small sample of the whole data provided by the experts for testing: the annotated gold standard and the results of the HILDA parser over six documents. During this phase, several design problems were identified regarding the color coding of discourse trees, improper visual cues and interaction issues preventing a fluid exploration. Our experts also requested additional functions such as filtering and querying the tree structures and displaying summary statistical information about the discourse trees.

Prototype refinement. At this stage, a full version of the tool with a refined interface and complete set of functions was released to the technical expert for use in her real work setting, using a journaling technique to record usage and potential areas for improvement. This two-week long testing phase allowed us to identify a number of minor bugs. More importantly, our expert reported that she sometimes felt that the visualizations failed at providing adequate information about how the text was separated into groups under a level of the tree (Q1). In response, we designed and added an alternative representation of the tree structure based on the icicle plot [Kruskal and Landwehr, 1983]. Since icicle plots are space-filling, the extent of EDU nodes at each level is readily apparent.

System release and field test. This final phase served as a field formative study: a final system, including logging instrumentation of user actions for reliable quantitative usage analysis, was deployed to the expert user for a longer time period (several weeks). We report on this last phase in more detail in Section 5.6.

5.3.2 Design Requirements

Through the iterative process with our expert users, we learned a lot about the process of research to improve discourse analysis. Gathered from and validated by our observations, and in response to the questions enumerated in Section 5.2.1, we defined the following specific design requirements for analytic tools to support discourse analysis:

R1 **Discourse tree representation.** While several approaches for representing the discourse trees could be considered, our experts explicitly requested that the core visualization resemble the representation they currently use for analysis of the parsing structure (see Figure 5.2-a). In particular, immutable constraints are the presentation of leaf nodes in the same order as the associated text chunks appear in the document, and the explicit visual encoding of nodes' type (satellite or nucleus) and relation. More importantly, the visualization should facilitate the identification of clusters at any level of the tree, a flaw of the node-link diagram currently used in practice. Finally, our users mentioned that it was naturally preferable that the different trees that share the same set of EDUs are aligned to facilitate structure comparison (Q3).

R2 **Errors, distributions and statistical information.** In order to gain better intuitions for designing new algorithms, linguists must discover the pros and cons of different parsers. Statistical information, such as the distribution of relationship types appearing in a tree or the similarity scores assigned to branches of the tree should be readily available, as they provide an important overview of the discourse structure and performance of the parsers (Q2).

R3 **Article views and textual contexts.** In order to observe whether the parsing algorithm groups EDUs into meaningful units when building the discourse tree, it is important to be able to see how EDUs are grouped under a relation (Q1). At any level of a discourse tree, the tool should support separating the source text into proper chunks (groups of EDUs) according to the branching pattern. It is also important to be able to view the text of individual EDUs in isolation. Finally, to support close reading of the documents under analysis, a standard paragraph-based layout of the document should be supported.

While the above requirements are described in the context of the specific NLP application domain, we note that the general ideas behind (R1) and (R2) are not exclusive to linguists. Indeed, representing the tree structure and node details in a similar fashion to that traditionally used by the target users is likely to be valid in other domains involving tree comparison. Similarly, the analysis of distributions, statistical information and errors are recurrent themes for systems that operate under uncertainty. However, the general tree visualizations tools that could support these requirements would still need to be adapted to effectively satisfy the specific domain constraints. To best support R1-3, we propose a highly tuned design for representing the discourse tree structures, the core visualization component of the DAViewer system, that we describe in Section 5.4.

In addition to the above requirements, the visual exploration tool should support general requirements including: dynamic queries, such as filtering and querying, to allow for a focused attention on specific types of relations or structural patterns (R4); visual consistency across the different visualization components for a fluid and effective exploration (R5); flexible data management, to assist researchers to do long-term progressive research during which the datasets may expand (R6); and finally, annotation, to support documentation of the findings for future reference (R7).

5.4 Visual Representations of Discourse Trees

In this section, we describe the core visualizations of discourse trees used in DAViewer. Those designs are relatively independent to the application domain, and can be generalized to visualize any hierarchical structure, especially when the visual comparison of tree structures are concerned, as we later discuss in Section 5.7.

5.4.1 Expanded Views

We designed two main types of representations to visualize the discourse trees: an adapted version of the icicle plot [Kruskal and Landwehr, 1983] (Figure 5.3-a), and a dendrogram — a branching node-link diagram particularly suited to reflect relationships (Figure 5.3-b), that resembles the traditional representation as used by our target users (Figure 5.2-a).

Tree structure and node details. In both representations, the nodes are color-coded according to the assigned relation label from the 18 described by Hernault et al. [2010]. The specific hues were selected from 18 of the 22 most distinguishable colors in Green-Armytage [2010]'s color alphabet. The hue of the links between the nodes and their parent relation indicates the nuclearity of the children nodes in the node-link dendrogram: black indicates a nucleus (the most prominent nodes), and grey indicates a satellite. In the icicle-dendrogram, the color of the outline of the children node indicates this property. These representations are interactively linked to the text panel which displays the content of the document (see Section 5.5.2), thus allowing for easy access of the text chunks associated to the nodes of current focus (R3).

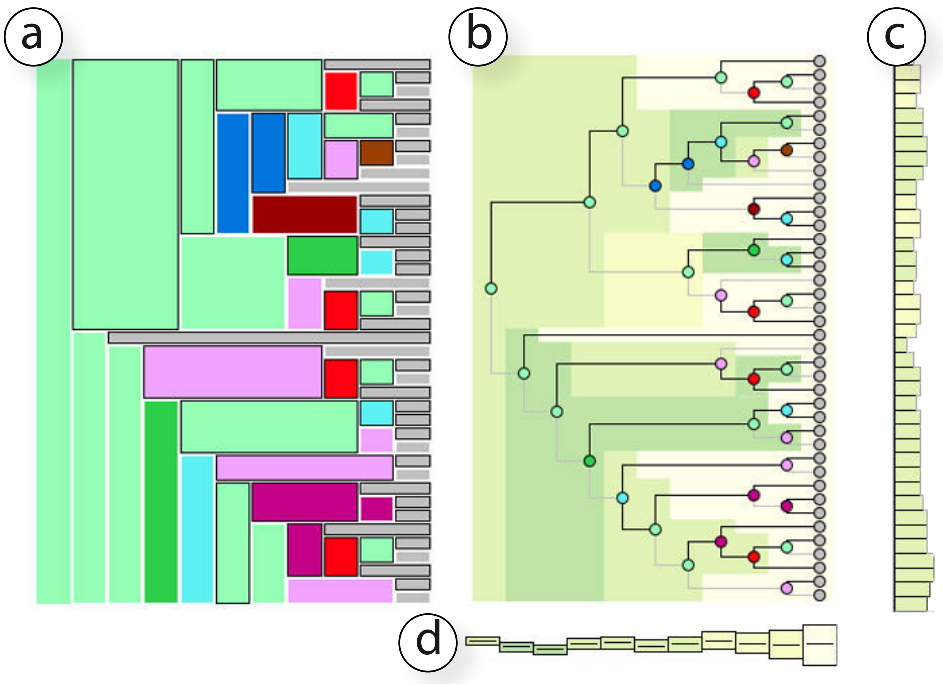


Figure 5.3: Visual representations of discourse trees: (a) icicle plot view, (b) dendrogram view, (c) vertical compact view, and (d) horizontal compact view.

Representing the clusters. We added the icicle-based plot after our experts reported that it was difficult to see the clustering of the EDUs at a specific level of the tree (Q1). With the node-link dendrogram, one has to follow the lines to accurately rebuild the different groupings. This can be challenging with large or high trees. While it is well accepted that the dendrogram eases the readability of hierarchical structures when the leaves must be aligned, the traditional icicle plot is generally better than node-link diagrams for the task of identifying clusters [Kruskal and Landwehr, 1983]. Our icicle visualization is a hybrid representation: we display nodes in the form of rectangles as in the traditional icicle, to make the embedding relation apparent. However, our layout mimics the dendrogram in that we align all the leaves at the same rightmost level, and we expand each rectangle's width up to the level where the corresponding node is grouped in turn. In this way, when looking at the icicle-dendrogram in columns, one can clearly see the clustering of EDUs at each intermediary stage of the bottom-up grouping process.

Showing the errors. In order to inspire the development of improved parsing algorithms, it is essential for linguists to develop a good understanding of the specific flaws of the existing algorithms compared to the gold standard. In particular, our experts are interested in identifying precisely which step(s) of the greedy bottom-up process fail, and to what extent such errors have an impact on the steps that follow (Q2). To facilitate such analysis, we build an icicle-dendrogram where nodes are color-coded according to the similarity scores of the internal nodes using a yellow-to-green palette[3]. This dendrogram is used as a background on top of which the node-link dendrogram is overlaid (Figure 5.3-b), thus serving as a heatmap where errors are made more salient because of the darker color.

5.4.2 Compact Views

When many trees have to be compared, as it is the case for our users, space limitation can become an issue. We designed two new compact representations of the discourse trees. In a method similar to how Table Lens [Rao and Card, 1994] allows for compact graphical representations of symbolic data for space efficiency, we propose compacted rows and columns, while providing a vertical (Figure 5.3-c) or horizontal (Figure 5.3-d) informative representation that summarizes important characteristics of the compacted discourse trees (R2).

Vertical compact view. In this view, each bin corresponds to a leaf node, i.e., an EDU. The width of a bin encodes the depth of its associated leaf in the tree, i.e., the number of intermediate clusters the EDU belongs to on the path to the root. The color of a bin is mapped to the average similarity score of its corresponding leaf node and all nodes on the path from the leaf to the root. Hence, the compact representation conveys the average error an EDU contributes across the levels. For example, a wide dark bin indicates an early grouped EDU whose initial error strongly impacts all its upper levels.

Horizontal compact view. In this view, each bin corresponds to a level of the compacted tree. The height of a bin encodes the number of nodes at the associated level. The color is mapped to the average similarity score of the set of nodes. Hence, the compact representation allows a viewer to identify at which stage of the clustering process the algorithm starts to fail and whether such errors propagate to upper levels or not. We also use the vertical axis to represent the centroid vertical position of the nodes belonging to the level, and center the bins accordingly. A horizontal black mark in the center of the bin is used as a visual cue of the centroid location. This gives an indication of whether the EDUs are clustered evenly across the document (reflected by a mark close the center), or if the groups of EDUs are more concentrated on one or the other side (the higher the bin, the more independent clusters at the beginning of the document).

The two compact representations shown in Figure 5.3-c,d can be viewed as two marginal perspectives of the node link view shown in Figure 5.3-b along the vertical and horizontal dimensions. While they have initially been designed to help answer question Q2 through the

[3]The color scheme is adopted from Colorbrewer, http://www.colorbrewer.org.

presentation of trees at different levels of abstraction, the proposed compact representations, as a design concept, can be generalized for hierarchical tree structures in general (see Section 5.7).

5.5 DAViewer Interface

We developed DAViewer, a complete interactive visualization tool for computational linguists to explore, compare, evaluate, and annotate the results of parsers in discourse studies. We designed DAViewer around the core discourse tree visualizations introduced above, and refined the interface through the four discrete stages of the user-centered design methodology as described in Section 5.3.1.

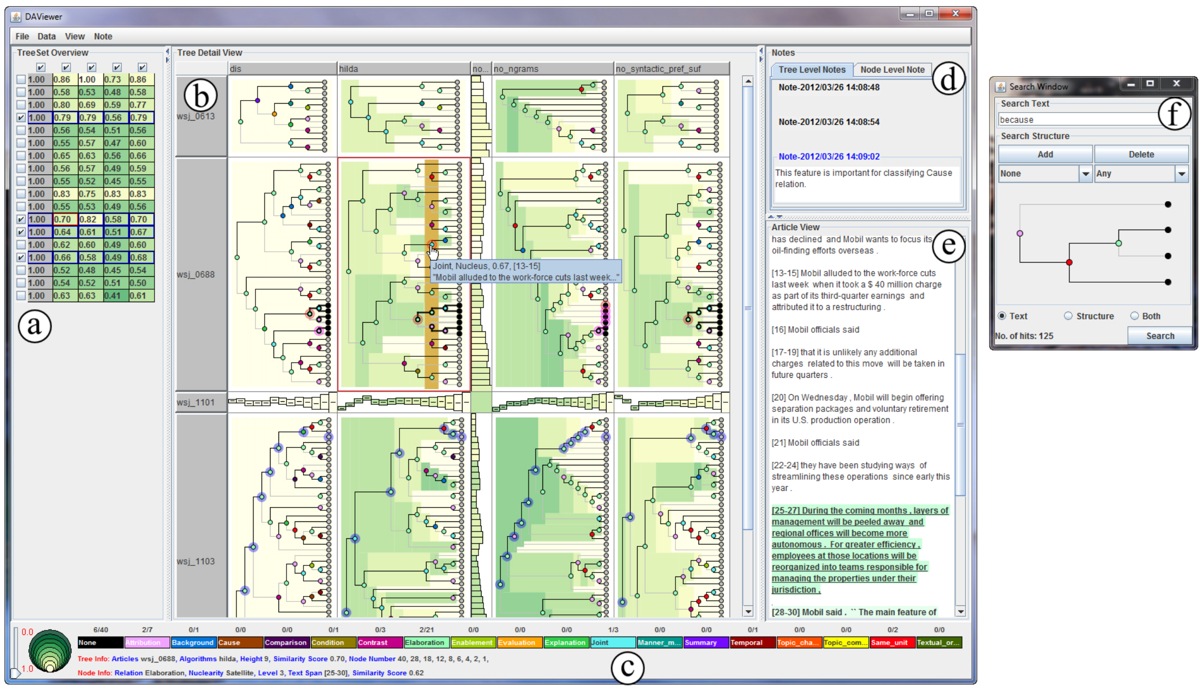
The DAViewer interface is composed of five interactively coordinated views (Figure 5.4) including (a) an overview panel of the entire dataset, (b) a detail panel, (c) a status panel, (d) an annotation panel, (e) an article panel, and (f) a search window. Dynamic brushing and linking techniques [Keim, 2002] are applied for interactively coordinating information displayed in the different panels (R5).

5.5.1 Overview: Seeing the Whole Dataset

The overview panel (Figure 5.4-a) consists of a matrix view representing the entire available dataset— a collection of discourse trees, each tree resulting from the computation of a given parsing algorithm (columns in the matrix) applied to a given document (rows in the matrix).

The overview is useful to determine at a glance how the different algorithms perform on the same document — which amounts to comparing the results along a row of the matrix, and how the same algorithm performs across a set of documents — which amounts to comparing the results along a column of the matrix (Q5), as compared to the reference algorithm (i.e., the gold standard, or otherwise user-defined). One column is set as the reference and shown in grey. In Figure 5.4-a, the leftmost column serves as the reference. Each non-reference cell is color-coded according to a similarity score between the tree in that cell, and the current reference tree in that row (i.e., we compare trees for the same document). The numeric scores are displayed in each cell, and also mapped to a yellow-to-green scale. Therefore, the matrix plays the role of a heatmap, that a user can use to quickly identify the trees that differ the most from the current referent, and thus requiring deeper investigation to identify the problems.

In our current implementation, the similarity measure relies on the element-based measure proposed by Bremm et al. [2011], but other similarity measures could be considered. The overview displays the scores associated with the root nodes, conveying the similarity scores between the whole tree structures. Low similarity scores generally indicate parsing errors. For



	✓	✓	✓	✓	✓
☐	1.00	0.86	1.00	0.73	0.86
☐	1.00	0.58	0.53	0.48	0.58
☐	1.00	0.80	0.69	0.59	0.77
✓	1.00	0.79	0.79	0.56	0.79
☐	1.00	0.56	0.54	0.51	0.56
☐	1.00	0.55	0.57	0.47	0.60
☐	1.00	0.65	0.63	0.56	0.66
☐	1.00	0.56	0.57	0.49	0.59
☐	1.00	0.55	0.52	0.45	0.55
☐	1.00	0.83	0.75	0.83	0.83
☐	1.00	0.55	0.53	0.49	0.56
✓	1.00	0.70	0.82	0.58	0.70
✓	1.00	0.64	0.61	0.51	0.67
☐	1.00	0.62	0.60	0.49	0.60
✓	1.00	0.66	0.58	0.49	0.68
☐	1.00	0.52	0.48	0.45	0.54
☐	1.00	0.54	0.52	0.51	0.50
☐	1.00	0.63	0.63	0.41	0.61

Joint, Nucleus, 0.67, [13-15]
"Mobil alluded to the work-force cuts last week..."

Note-2012/03/26 14:08:48

Note-2012/03/26 14:08:54

Note-2012/03/26 14:09:02
This feature is important for classifying Cause relation.

has declined and Mobil wants to focus its oil-finding efforts overseas .

[13-15] Mobil alluded to the work-force cuts last week when it took a $ 40 million charge as part of its third-quarter earnings and attributed it to a restructuring .

[16] Mobil officials said

[17-19] that it is unlikely any additional charges related to this move will be taken in future quarters .

[20] On Wednesday , Mobil will begin offering separation packages and voluntary retirement in its U.S. production operation .

[21] Mobil officials said

[22-24] they have been studying ways of streamlining these operations since early this year .

[25-27] During the coming months , layers of management will be peeled away and regional offices will become more autonomous . For greater efficiency , employees at those locations will be reorganized into teams responsible for managing the properties under their jurisdiction .

[28-30] Mobil said . `` The main feature of

Search Text: because

No. of hits: 125

6/40	2/7	0/1	0/0	0/0	0/1	0/3	2/21	0/0	0/0	0/0	1/3	0/0	0/0	0/1	0/0	0/0	0/2	0/0
None	Attribution	Background	Cause	Comparison	Condition	Contrast	Elaboration	Enablement	Evaluation	Explanation	Joint	Manner_m...	Summary	Temporal	Topic_cha...	Topic_com...	Same_unit	Textual_or...

0.0
1.0

Tree Info: Articles wsj_0688, Algorithms hilda, Height 9, Similarity Score 0.70, Node Number 40, 28, 18, 12, 8, 6, 4, 2, 1,

Node Info: Relation Elaboration, Nuclearity Satellite, Level 3, Text Span [25-30], Similarity Score 0.62

Figure 5.4: The DAViewer system's user interface is comprised of: (a) an overview which displays the overall performance of all the parsers over all the documents in the dataset, (b) a detail panel which visualizes the discourse tree structures of the focused algorithms and documents as node-link or icicle dendrograms, (c) a status panel which provides the basic properties of the currently selected items as well as an interactive legend for filtering operations, (d) an annotation panel which allows users to edit notes, (e) a text panel which shows the content of the active document as parsed by the focused algorithm, and (f) a search window which for querying based on keyword and structure over the data in the detail panel.

example, one can easily tell from Figure 5.4-a, that the last document (last row) is generally problematic, and that the third algorithm (third column) under performs the others on this specific document.

In practice, as the research goes, linguists develop several refined versions of the same algorithm or propose new algorithms. They may also decide to extend their gold standard by adding more annotated documents. To support the progressive research process of our users, DAViewer provides convenient ways of adding and/or removing entries to workspace (R6).

5.5.2 Drilling Down: Further Exploration of a Set of Trees

When a user has identified problematic algorithms and documents, she can drill down into the data and access an expanded view of a set of trees, as described in Section 5.4.

Choosing the Trees of Interest

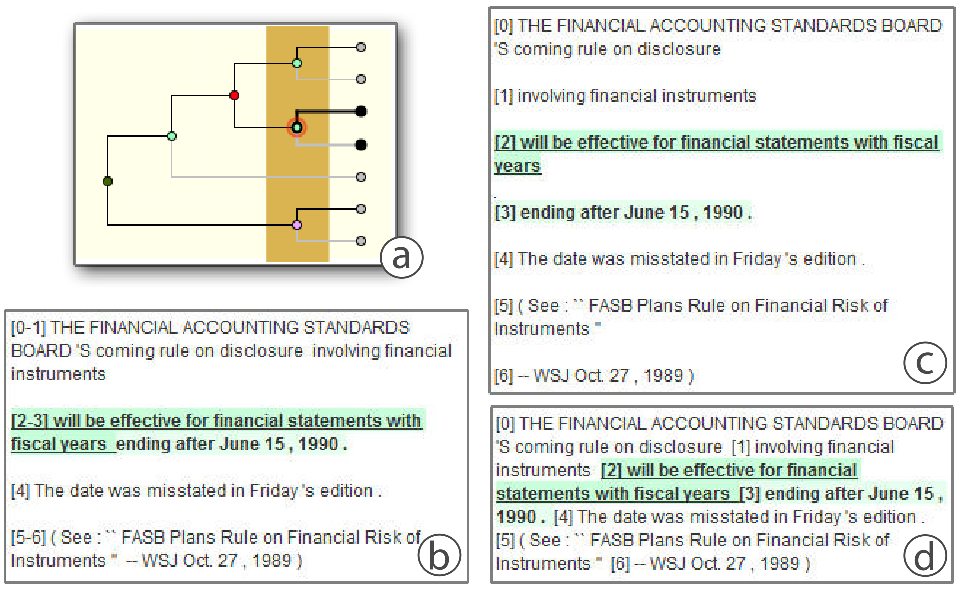
In order to preserve a consistent tabular representation across the views and maintain the alignment of EDUs across algorithms (R1), the selection is algorithm- and document-based, in that the user selects the set of trees of interest by checking row (document) and column (algorithm) headers in the overview. All the cells at the intersection of the selected rows and columns are loaded in the detail panel, as shown in Figure 5.4-b. The currently selected cells are outlined in blue in the matrix overview.

Depending on her current task, the user can interactively choose between the icicle plot view, the dendrogram view, or decide to reduce some trees to their compact representations to devote more space to the algorithms and documents of immediate interest.

Exploring the Discourse Trees

DAViewer provides fluid interactions for a multi-focus visual exploration of discourse trees. A user can select a node to access detailed information of the node and its subtree in the status panel (Figure 5.4-c). This also triggers the emphasis of the node and the branch under it, using thicker edges. In other trees in the same row, the branches corresponding to the same EDUs (leaf nodes) are also emphasized (see middle row, Figure 5.4-b) so as to facilitate the comparison of internal structures (Q4). The associated text of the EDUs under the selected node is loaded into the text panel (Figure 5.4-e), and emphasized in several ways (R3): (1) we apply a bold font to the selected node's content, (2) the text of the first child node is underlined, and (3) the background is colored according to the main relation's hue, with a dark or light tone whether the text belongs to a nucleus or a satellite branch respectively (Figure 5.5).

As the user moves the cursor in the active table cell, a semi-transparent brown ribbon indicates the tree level where the cursor resides, as an aid to identify the nodes on the same tree level (Figure 5.5-a). The text can be formatted in three ways. First, clicking on an empty space within a level (as opposed to a node) selects that level, which rearranges the text into a layout we call *hybrid display*: the text is split into paragraphs according to the EDUs branching at that level (Figure 5.5-b). This text display is particularly useful when the user wants to analyze the intermediate stages of the relation grouping (Q1, R1). The second text layout is the *separated display* (Figure 5.5-c) which presents each EDU as a separate paragraph. Finally the *continuous display* (Figure 5.5-d) displays the article as a single paragraph. The separated display is suited to the matching EDUs to leaves in the tree, as the EDUs remains clearly separated, while the continuous display aims to offer a more comfortable way of reading text in a continuous flow. The separated display is equivalent to choosing the leaf level in the hybrid display; the continuous display is equivalent to selecting the root level in the hybrid display.



[0-1] THE FINANCIAL ACCOUNTING STANDARDS BOARD 'S coming rule on disclosure involving financial instruments

[2-3] will be effective for financial statements with fiscal years ending after June 15 , 1990 .

[4] The date was misstated in Friday 's edition .

[5-6] (See : `` FASB Plans Rule on Financial Risk of Instruments '' -- WSJ Oct. 27 , 1989)

[0] THE FINANCIAL ACCOUNTING STANDARDS BOARD 'S coming rule on disclosure

[1] involving financial instruments

[2] will be effective for financial statements with fiscal years

[3] ending after June 15 , 1990 .

[4] The date was misstated in Friday 's edition .

[5] (See : `` FASB Plans Rule on Financial Risk of Instruments ''

[6] -- WSJ Oct. 27 , 1989)

[0] THE FINANCIAL ACCOUNTING STANDARDS BOARD 'S coming rule on disclosure [1] involving financial instruments [2] will be effective for financial statements with fiscal years [3] ending after June 15 , 1990 . [4] The date was misstated in Friday 's edition . [5] (See : `` FASB Plans Rule on Financial Risk of Instruments '' [6] -- WSJ Oct. 27 , 1989)

Figure 5.5: Different text display formats: (a) in the discourse tree, the currently active level is indicated by a dark brown ribbon. In the text panel, the content display can be set to (b) hybrid, to reveal the EDUs grouping at that level, (c) separated, to show individual EDUs, or (d) continuous to present the text as a single paragraph.

DAViewer also supports traditional tree operations such as branch collapsing when the user double clicks on a node. It is also possible to collapse branches through batch operations using a context menu (e.g., collapse all branches under a node with similarity score below a certain threshold, or all branches up to a specific tree level).

The rich interactions in DAViewer allows for a flexible exploration of the trees. This is particularly useful for a close analysis of the relations and the text associated to a problematic node, that can help identify the causes of a mislabeled relation (e.g., a keyword causing ambiguity). When comparing how a set of EDUs are clustered across several algorithms, the user can also explore in details of parsing differences, allowing her to infer hypotheses on the impact of parameter settings on the correct identification of specific relations, as illustrated in Section 5.6.4.

5.5.3 Filtering and Querying: Importance Revealed

The status panel acts as an interactive legend for filtering operations (R4), including the similarity score legend (Figure 5.6-a), to filter out nodes with specific similarity scores, the relation legend (Figure 5.6-b), where each label is a modal button for dynamic filtering. DAViewer also incorporates querying functions for node and branch searching (R4), including keyword search, structure-based querying, and the combination of the two (i.e., all branches

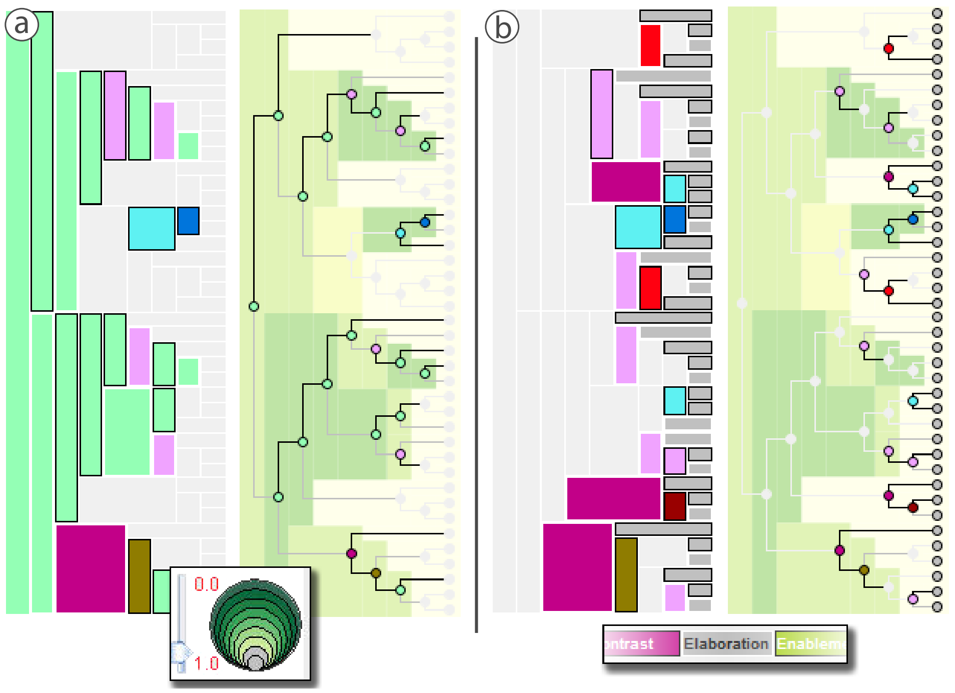


Figure 5.6: Filtering operations of interactive legends: (a) filtering out nodes above score 0.8 and (b) filtering out nodes with the *Elaboration* relation.

that satisfy both queries) through a separate dialog (Figure 5.4-f). Keywords can be entered in the text area, triggering the highlighting of all the nodes which descendant EDUs contain such a keyword. To perform a structural search, the user builds her pattern of interest either from scratch, or by editing a structure copied from the detailed view. The user can specify or omit the relation and nuclearity types associated to the nodes of the query pattern. When no value is specified, the engine searches for *any* value of such nodes. All the structures that match the query are highlighted in the detail panel with a blue halo. The structure-based query function has been explicitly requested by our users as a critical function to locate recurrent error patterns, because this task is extremely difficult to perform with their current analysis method (Q4).

5.5.4 Annotation: Writing a Research Journal

The analysis of parsing results is a continuous and progressive process as the research progresses. To allow NLP researchers to record insights and exploration notes, DAViewer supports annotations on trees or nodes, and groups of trees or nodes through the panel as shown in Figure 5.4-d (R7). Notes are presented as a collapsible list of items titled with the timestamp of the last addition and can be saved together with the workspace status. The note panel is fully coordinated with the other views: when a note is selected, all the corresponding items in the overview and detailed view are visually emphasized.

5.6 Preliminary Deployment Study

We conducted a preliminary longitudinal study with our technical expert to evaluate DAViewer in a realistic work environment for three weeks. This section describes the methodology, summarizes the findings, and presents two example use case scenarios.

5.6.1 Methodology

DAViewer is designed to be tailored to the needs of computational linguists in their research process. Our research process depends on many factors including the user's motivation, previous knowledge and particular work settings, which are not easy to evaluate in a laboratory study. To better understand the strengths and flaws of our tool in a realistic work environment, we adopted a formative study approach with our expert user over a three-week period. The expert participated in our study was the technical expert who had been involved in the design process of DAViewer. She has worked in the field of computational linguistics research for 4 years, and her current focus was to develop robust parsers for discourse analysis of text at the document level.

The goal of the study was threefold. First, we wanted to evaluate the overall acceptance and usability of DAViewer as an analytical tool assisting computational linguists in their research process. Second, we were interested in investigating the patterns of use of the different interaction and visualization components, as a valuable guide for future improvements of our system. Third, we were hoping DAViewer would allow our user to discover new insights, and were therefore interested in collecting use case scenarios.

We released DAViewer to the technical expert, who was free to use the tool at will on her personal computer. Since DAViewer supports the standard file formats used by linguists, our user did not encounter any difficulty generating and loading the additional datasets to her workspace. For the study, we added a logging mechanism to collect occurrences of a set of 39 operations (e.g., select a node, set icicle view, etc.) that we classified into eight higher-level categories (e.g., dataset management, detail view manipulation, querying, etc.). Operations and categories are detailed in Figure 5.7-b. Each operation was recorded alongside its time stamp. We also requested our user to keep an analysis diary, and maintained regular contact via email and several short informal interviews at regular intervals during the study. At the conclusion, we also conducted an additional interview to collect her feedback after the evaluation period.

5.6.2 Usability and Satisfaction

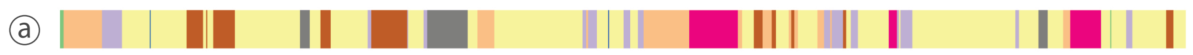
Over the course of the study, our expert user had been developing effective and robust discourse parsers for analyzing linguistic structures of articles mainly from the Wall Street Journal. To research on the nature of discourse analysis, she had tried different linguistic features extracted from the text as the input of the parsers, thus to study the effect of certain features and sensitivity of different parser frameworks, which had involved lots of comparison analysis of structures of the resulting discourse trees.

During the final interview concluding the formative study, our expert user reported that the visualizations offered significant benefits for analyzing the outputs of discourse parsing algorithms: *"It is great to have all the [discourse] trees available and see them visually. I can print [trees] in text files and look at them all day long, but never found it could be that clear and easy with the visual representations."* She added that *"The collapsed views are very handy, because sometimes I just want an overall comparison, not the deep deep details,"* and reported to be highly satisfied with the GUI design and interactions that she found to be *"handy and straightforward"*.

DAViewer proved to be very efficient to our user as it greatly increased her productivity: *"I used to draw trees by hand according to the output text files, so I could only compare two or three trees at the time and they are basically simple trees around 5 levels. With DAViewer I can compare many large trees efficiently. All trees are nicely aligned and the interactions allow me to focus on subtrees easily."* The user emphasized that her past experience of drawing trees of about 20 EDUs was "awful" (each tree taking her more than 10min to draw) and highly error-prone due to the manual process. Then she mentioned: *"For the same data, now I can spot the [tree structure] differences in 2 seconds by observing the [similarity] score heatmap."* For comparing very large trees, which was almost impossible to do in the past, the expert also found the row and column compact tree views useful for identifying the parts to focus on first.

The visual representations of discourse trees were favored by the user because they were efficient in showing the parsing process and comparing tree structures, though they were slightly different from the manually produced graphs. The user said, *"I know I want the tree levels aligned from the bottom, but I can't draw such layout from the output file in one round since the tree nodes are indented in the depth-first order."* Moreover, she commented that the data visualization was flexible because the separation of text and tree structure allowed her to focus on different aspects of the dataset and the interactions such as tooltips and highlights of text provided the connections conveniently.

Other valuable functions for our user were the searching and filtering capabilities, especially the structure-based query. She said, *"I used to insert debug code inside the parsers, for a verbose output, but it is tedious and almost impossible to find patterns. Now, I can do*



b

dataset management	open treeset in plain text format, import a tree or a treeset, save all workspace, load a workspace, remove row/column from the dataset, append row/column to the dataset
overview and tree subset	change reference column, load selected trees in the detail panel, add trees to the detail panel, remove trees from the detail panel
detail panel manipulation	select a row in the detail panel, select a column in the detail panel, select a tree cell, compact a tree cell, uncompact a tree cell
tree exploration	set dendrogram view, set icicle view, collapse branches up to a level, collapse branches above a score, expand all nodes, expand a branch, collapse/expand a branch (double click), select a node
text display	set continuous display, set separated display, set hybrid display
annotation	add a tree-level note, add a node-level note, delete a note
querying	add a note to a pattern, select a node in the pattern, edit pattern, search keywords, search structure, search both
interactive legend filtering	activate/deactivate a relation, change score filter threshold

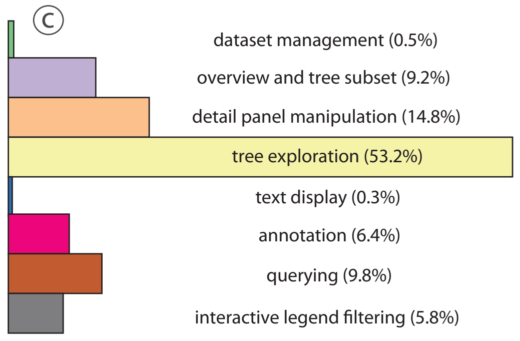


Figure 5.7: Results of the logging gathered during a longitudinal study: (a) shows the break up of a typical session of DAViewer into categories of operation as detailed in (b), and (c) summarizes the overall usage of the different types of operations.

it visually by just creating what I need and clicking a button." She indicated that a further improvement of the querying function with boolean operations would greatly enhance the tool.

In summary, our expert was enthusiastic about DAViewer, as reflected by the numerous extended sessions she conducted during the evaluation. She would like to continue using the tool in her research.

5.6.3 Patterns of Usage

In addition to the interview notes, user's diary, and emails, we collected 17 log files (449 minutes in total), from which we discarded 5 log files that corresponded to sessions of less than 5 minutes long. The remaining logs corresponded to use of 432 minutes, in session ranging from 15 to 56 minutes long ($\mu = 36, \sigma = 13$). Altogether, the remaining log files contain a total of 3048 operations, with a minimum of 55 and a maximum of 456 operations per session ($\mu = 254, \sigma = 124$). By looking into the temporal aspect of the logs, we found that the expert used DAViewer for 3 days in the first week (in total 97 minutes), only for 2 days in the second week (in total 86 minutes), and for 5 days in the third week (in total 249 minutes). Our expert mentioned that she used DAViewer for some short general explorations during the first week, then discovered something interesting in the second week and thus switched to working on developing discourse analysis algorithms. In the third week, she largely relied on the tool to compare different results generated by different methods with which she had experimented.

Figure 5.7 summarizes the results of the operations logging during the formative study, including (a) a color-coded time diagram of a typical working session, (b) the classification of the operations into categories, and (c) the overall distribution for each category use.

Not surprisingly, the user spent most of her time on tree exploration, of which node selections were predominant (92% of tree exploration operations). We found out in the final interview that collapsing nodes was not desirable, as it does not preserve alignment of the EDUs and it disturbs the global *"mental representation of the tree."* A design implication is that future versions should support synchronized node collapsing and coupled panning to preserve EDU alignments.

Since the detail panel was her main focus, the user naturally spent a fair amount of time adjusting the treeset shown in this panel. Nearly one third of the operations consisted of compacting and uncompacting rows and columns. The expert commented that she preferred to look at the trees first in these compact views to get a general idea about the performance of the parsers, especially when the structure was large. This reveals that our design of compact views was useful for analysis, in addition to saving screen real estate.

The dendrogram was largely preferred over the icicle plot (79%). Our user explained that for most of the time her purposes were to identify structural differences among trees, that is easier to do with the dendrogram. However, she said that the icicle plot was more useful and more salient when observing the behaviors of parsers, i.e., how nodes are merged from bottom to top and how the whole article is partitioned at each tree level of processing.

The user also performed a large amount of queries (299 in total) and filtering operation (177 in total). Among the querying operations, 18% were keyword searches, 55% were structure-based and 27% were both, confirming the importance of structural queries in our domain of application. Moreover, the expert mentioned that this significantly increased the efficiency of spotting desired structural dissimilarities by interactively combining operations of both querying and filtering.

Surprisingly, the user rarely changed the text display format, although it was an important functional requirement. In the end, the user explained that her most favored display format, the hybrid display, was flexible enough to easily access both other displays by simply selecting the root or the leaves levels. She however did choose to keep the display fixed in rare occasions.

Finally, we analyzed individual log files to gain knowledge of the user's exploration flow when using DAViewer. We plotted each session as a color-coded time diagram. An example in Figure 5.7-a represents a typical working session. We observed a recurrent pattern across all the sessions: after loading the dataset, the user adjusts the documents and algorithms of interest in the overview. Then a long period of time is dedicated to detailed exploration: tree manipulations, interleaved with querying and filtering, and eventually the creation of a note about insights. This pattern is then repeated several times on the same treeset, until the user decides to adjust the latter for a new investigation on different trees. This usage exhibits a typical visual exploration process for hypothesis verification and discovery, that DAViewer successfully supports with its coherent coordination between the panels presenting the data at different levels of abstraction.

5.6.4 Use Case Scenarios

In this section, we present two use case scenarios based on the user diaries and interaction logs that we discussed with the expert user during the interviews. We tried to restore the images of her normal usage of DAViewer in the realistic working environment. The dataset she used as a gold standard was a set of 20 annotated documents from the RST-DT set [Carlson et al., 2001]. The parsing algorithms included the HILDA discourse parser [Hernault et al., 2010] and several algorithms of her own.

The HILDA Parser versus the Gold Standard

In this scenario, a user wants to investigate the flaws of the HILDA parser, a popular algorithm in the domain. She starts by glancing at the overview which presents the similarity scores, and finds that overall, the parser is performing fairly well. She identifies the two documents with the highest (0.86 and 0.83) and lowest (0.52 and 0.54) scores and selects them for deeper analysis. The gold standard, and the HILDA output for these four documents are loaded in the detail panel.

The user immediately finds out that the documents causing errors are much longer than the other ones, which is reasonable, as typically, the more content, the more challenging the parsing. Likewise, the discourse trees are large and difficult to read as a whole structure. In order to get an overall idea of where the algorithm fails, the user reduces the trees to their compact representations to observe the distribution of scores, groups and so forth. From the vertical compact view, she observes that while the distribution of nodes into groups is similar to the gold standard, the scores of HILDA are very low (Figure 5.8-a). Next she expands the tree views, and at the same time compacts the short documents since they are not the focus a the moment.

With the help of the heatmap background of the dendrogram, the user identifies where the error first occurs: HILDA groups EDUs 16 and 17 as early as the second level whereas the gold standard keeps the branches separated up to level 17 (Figure 5.8-b). Thus she found that this first error, which propagates to the root node, is a major problem that strongly affects the overall parsing. By looking at the text, she finds that the EDU 17 says “individual prosperity inevitably would result” where the keyword “result” is a critical indicator of the *Cause* relation. However, the HILDA parser groups this node under an *Elaboration* relation.

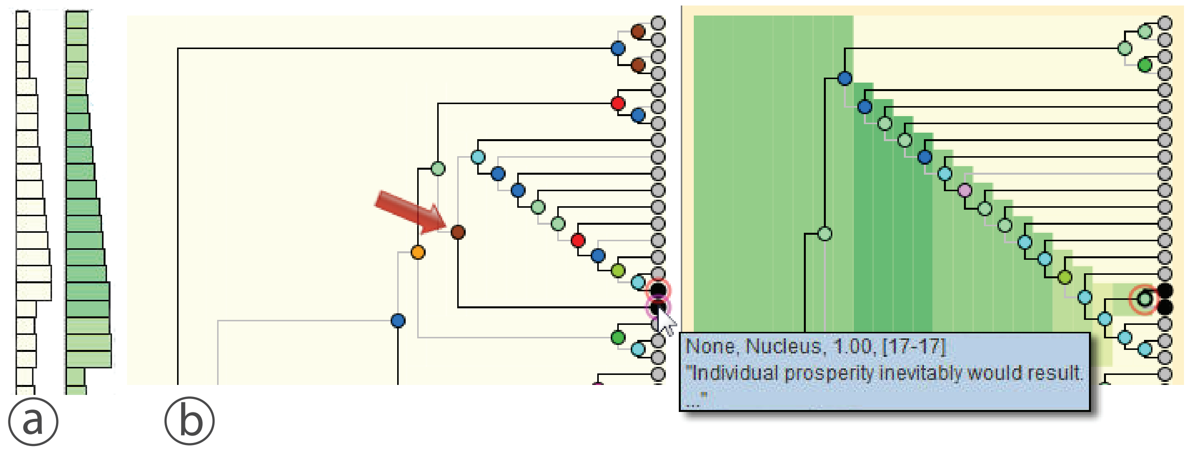
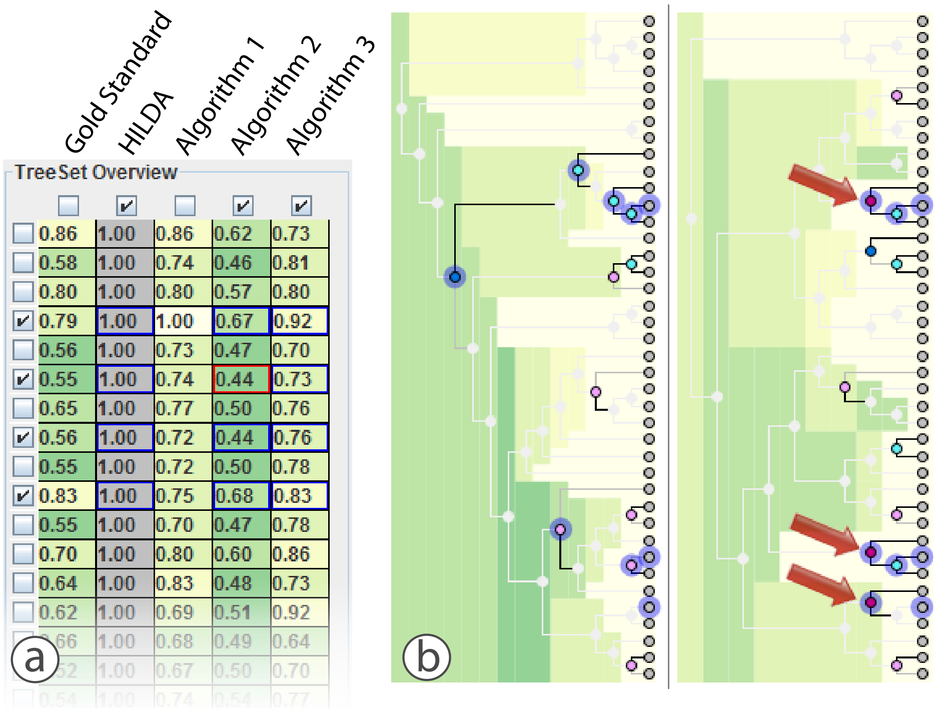

None, Nucleus, 1.00, [17-17]
"Individual prosperity inevitably would result.
..."
a
b

Figure 5.8: Comparing HILDA with the gold standard: (a) the compact vertical views allow for the localization of the error in HILDA: (b) the incorrectly grouped nodes are shown in red outlines, (c) the red arrow points at the correct *Cause* relation.

To get some context, the user switches to the continuous text display to comfortably read the sentence with the problematic EDU. She finds that EDUs 7-16 as a whole are the summary of previous content, that should be grouped together in a branch under the *Cause* relation, with EDU 17, as indicated by the gold standard. The user thus selects the group of nodes and comments on her finding on the annotation panel. Meanwhile, she wonders if such errors happen elsewhere. She adds more trees with low scores to the detail panel, and through the query panel, looks for other *Causal* relation structures with branches containing the keywords "because" and "as a result". A close examination of the results reveals that HILDA incorrectly groups the nodes or mislabels the relation (*Elaboration* or *Explanation*) and adds notes each time she finds such error in the dataset for further consultation. Indeed, the above findings provide hints for improving parsers, by taking more careful consideration of the content under a *Cause* relation.

Comparison of the Performance of Different Algorithms

After identifying several issues in the HILDA parser, the user wants to investigate if and how tuning different parameters affects the outputs of her own parsers. She appends the result of three variations of her algorithm to the overview matrix (referred to as algorithms A1, A2 and A3). In this scenario, she sets HILDA as the reference column, since she wants to compare where the algorithms differ in performance. Adopting a similar approach as that of the previous scenario, she first glances at the overview and finds out that the fourth column (A2, corresponding to the condition "no N-grams feature") provides the most differing results, and that the rightmost column (A3, condition "no syntactic prefix and suffix") provides a



	☐	✓	☐	✓	✓
☐	0.86	1.00	0.86	0.62	0.73
☐	0.58	1.00	0.74	0.46	0.81
☐	0.80	1.00	0.80	0.57	0.80
✓	0.79	1.00	1.00	0.67	0.92
☐	0.56	1.00	0.73	0.47	0.70
✓	0.55	1.00	0.74	0.44	0.73
☐	0.65	1.00	0.77	0.50	0.76
✓	0.56	1.00	0.72	0.44	0.76
☐	0.55	1.00	0.72	0.50	0.78
✓	0.83	1.00	0.75	0.68	0.83
☐	0.55	1.00	0.70	0.47	0.78
☐	0.70	1.00	0.80	0.60	0.86
☐	0.64	1.00	0.83	0.48	0.73
☐	0.62	1.00	0.69	0.51	0.92
☐	0.66	1.00	0.68	0.49	0.64
☐	0.52	1.00	0.67	0.50	0.70
☐	0.54	1.00	0.74	0.54	0.77

Figure 5.9: Comparing different discourse analysis algorithms: (a) HILDA parser is set as the reference; (b) after filtering out *Elaboration* and *Same Unit* relations, the tree of algorithm 2 (left) is almost all faded out. A keyword search for "but" is applied (blue highlighted nodes). When compared with the tree of algorithm 3 (right) algorithm 2 cannot detect *Contrast* relations containing "but" whereas algorithm 3 does (red arrows).

very similar parsing as that of HILDA (Figure 5.9-a). She selects a subset of four rows (two documents with the most similar and most differing results) and three columns (HILDA, A2 and A3) for further analysis in the detail panel.

By looking at the different tree representations (i.e., compact views, dendrogram and icicle), the user discovers that the trees generated by A2 usually contain many more levels and are more skewed, indicating that the classifier cannot find clear grouping pivots. To make the differences more visible, the user decides to fade out the branches similar to those of HILDA by filtering them out through the use of the interactive similarity score legend, and switches to the icicle view to analyze the relation types. She observes that most of the remaining nodes are labeled with *Elaboration* or *Same Unit* by A2, which is unsurprising, since the two relations are the most common ones in that specific corpus. After deactivating the latter in the relation legend, she clearly observes that A2 hardly identifies any other relations, indicating that the N-grams feature, deactivated in A2, is essential for the relation classifier.

Our user wants next to compare the performance of a particular relation: *Contrast*. She looks for phrases indicative of this relation: "but," "on the contrast," etc., coupled with a structure-based query around the *Contrast* relation. She observes that A2 fails at identifying such relations, while it is usually well labeled by discourse parsers (Figure 5.9-b). She further comments on her findings by adding a note that considering a certain number of EDUs as a whole (as N-grams does) is a good parser feature for identifying the real rhetorical relations.

5.7 Discussion

The core visualization component of DAViewer is the tabular tree detail panel (Figure 5.4-b) with its icicle-dendrogram and compact representations of discourse trees which were revealed to be an important contribution of this paper (Figure 5.3). The icicle-dendrogram, when visually browsed vertically, reveals key features of hierarchical clustering as the traditional icicle plot does, which in our case reflects the greedy process of merging of text chunks from the original EDUs. Taking the benefits of dendrogram, nodes that remain unmerged across many levels, are shown saliently as rectangles with larger horizontal widths, making it easier to identify the nodes that reside on the levels covered by the width of a specific node, than when using a traditional dendrogram. In some studies of clustering algorithms such as classifiers in machine learning applications, this is useful for checking why a specific node is not combined with others in the algorithm as well as spotting the anomalies. In addition, when an icicle plot encoding the similarity score is displayed as the background of the dendrogram representation (Figure 5.3-b), the analyst can more easily locate the structural differences, as large color-coded areas are easy to spot at a glance. Our expert user extensively relied on this background heatmap to quickly find the very first merging anomaly propagated to the top levels when comparing parsers. This visualization, as a combination of two representations of the same tree, is general enough to be applied to any other hierarchical structure, whether it is for comparison purposes, e.g., studying the difference of evolutionary relationships between organisms in phylogenetic trees, or to simply augment a dendrogram with the visualization of an additional attribute associated to the nodes, e.g., displaying the space that files and folders take on the hard drive on the background of a file system browser.

The row and column compact representations of trees, on the other hand, offer a legible summary of both the structural information of the tree and aggregate attributes of contained nodes. While these visualizations were designed for discourse analysis, the concept is also generalizable to exploration of other types of hierarchical structure, and their new ways of abstracting tree structures could be useful for common tasks in other domains, of which finding structural change and identifying co-evolutions in phylogenetic trees are an example. In particular, when projected to the vertical axis, the view compresses the tree along its leaves by showing the depths similar to the icicle-plot, which indicates how species generally evolve and which branches are most active; and when projected to the horizontal axis, the view compresses the tree along its levels by showing its skewness, which shows the trends of the evolution to a common ancestor.

It is worth recalling that in our study, the trees corresponding to the same document are built from an identical set of EDUs, which is motivated by the unique features of discourse analysis. Yet, the tool proposed in this paper, and all its respective components, can be extended to the comparison of trees with different leaf nodes. In our specific application domain, this means that the similarity measure needs to be adjusted accordingly. The synchronized selection would also have to be adapted, by looking into the contents of the selected text, rather than EDU identifier numbers, as is done in our current implementation. In this more general case, the alignment of EDUs in trees within the same row of our detail view would not be meaningful anymore, as the EDU contents could differ.

5.8 Summary

In this chapter, we have described a design study in the domain of Natural Languages Processing, focusing on assisting computational linguistic researchers with the exploration and comparison of tree structures generated by discourse parsers. This design study has approached the challenges of visually exploring logical data structures expressed in a large number of data items. By situating it in the application domain of discourse analysis, our prototype system, DAViewer, was iteratively developed in a four-stage user-centered design process with expert users, based on incremental collection of user feedback and refinement of design requirements. The contributions in this chapter include:

- a set of domain specific design requirements for facilitating NLP researchers with the analysis of discourse parsers,
- a novel interactive visualization system, DAViewer consisting of multiple coordinated components particularly tailored for the research workflow in discourse analysis,
- two compact visual representations summarizing tree structures in horizontal and vertical perspectives, and
- a novel visualization of trees combining the dendrogram and the icicle plot.

As discussed in details above (see Section 5.7), the icicle-dendrogram and compact representations of trees together with the DAViewer's core interface component, i.e., the tabular view for visual comparisons, are generalizable to other application domains where the exploration of tree structures are interested, such as in biology. Although we focused on visualizing discourse trees (in the dendrogram layout), the proposed compact representations of trees can be used for summarizing normal hierarchical layouts (where leaves are not aligned). In addition to trees, there exists another common type of logical data structures — graphs — where there is no parent-child relationships but connections between data items. However, in some applications, such as directed acyclic graphs (DAG) used in project planning, nodes (e.g., key phases or steps of the project) are drawn in layers to express their sequential orders clearly, and links are used to show dependencies between steps. This is like a special tree (level based) layout but one "child" node may have multiple "parents". In this case, the compact visual representations in horizontal and vertical directions still apply using similar histogram-based aggregation ideas. Those views can provide overviews of this special graph structures from different perspectives.

In the upcoming two chapters, we will move forward to focus on developing multi-focus visualization techniques for facilitating the visual exploration of data attributes, the third data feature concerned in the space of data to visualize (see Section 1.2.2). It is worth noting that data attributes are properties of some objects that cannot be thought of independently and are often associated with data values or structures [Ware, 2004]. Thus design studies in the following chapters also have underlying connections to those introduced in Part II and Part III, although the following two chapters put special emphasis on visual exploration driven by the attributes on data items. For example, Chapter 6 considers multi-attributed temporal events, so that techniques introduced in Chapter 3 and Chapter 4 can be used for enhancing the interface; and Chapter 7 investigates multidimensional graphs, which is a complement to this chapter's study, i.e., covering both trees and graphs for data structures.

Part IV

Focusing on Data Attributes

Chapter 6

TIMESLICE: INTERACTIVE FACETED BROWSING OF TEMPORAL EVENT DATA

History is the version of past events that people have decided to agree upon.

Napoleon Bonaparte

The previous three chapters (Chapter 3–5) investigated techniques assisting visual exploration of data values and data structures. In this chapter[1], we focus on the third data feature concerned in this thesis — data attributes. More specifically, we are interested in visual exploration of datasets with multiple facets: A facet is a group of related attributes representing logical classifications of data. For example, "color" is a facet of an object, and "red", "yellow", "blue", and etc., are potential attributes that belong to the "color" facet. In additional to categorical attributed facets, a facet can be ordinal and numerical. For instance, the "grade" facet of a student record database may include ordinal attributes: A, B, C, D, and F, and the "weight" facet of a good inventory is numerical, i.e., expressed in numbers. This type of faceted information exists in many real-world applications, indicating the way how data is structured and represented. For example, faceted browsing is often used in navigating large collection of objects on commercial websites such as Walmart and Amazon.

Most of the previous work in this area mainly focuses on exploring collections of discrete data items that only contain categorical faceted attributes (see Section 2.1.2). Of particular interest, in this chapter, we further consider multi-faceted datasets that also have continuous time dimensions, i.e., temporal events associated with various categorical meta-data attributes, since time-oriented data is ubiquitous across many different domains as discussed in previous

[1]Main portions of this chapter were previously published in Zhao et al. [2012b]. Thus any use of "we" in this chapter refers to Jian Zhao, Steven Drucker, Danyel Fisher, and Donald Brinkman.

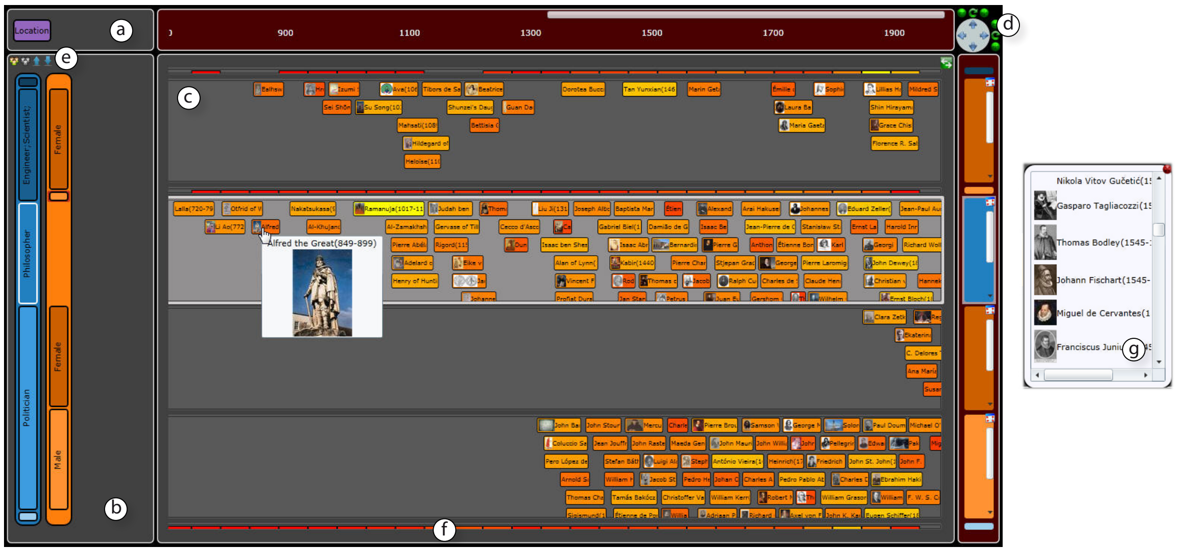


Figure 6.1: Exploration of lives of famous historical people using TimeSlice. The user is comparing female engineers and scientists, all philosophers, and politicians of each gender. Occupations such as philosophers are popular in early time. But people are actively involved in politics later (about 1300AD) in which females are few and involved even later (about 1900AD). Also, there seems to be fewer female engineers and scientists between 1300AD and 1700AD, which requires further exploration of the history. A video demo of the system can be downloaded here.

chapters. In this design study, we propose a novel visualization technique called TimeSlice (Figure 6.1), for visual comparison of multiple event timelines that are dynamically generated from raw data through visual queries in the facet space. A visual query is an abstraction of the traditional database query, such as using SQL, in a visual form so that users can execute it interactively and intuitively with visualizations. These queries are visualized in a hierarchical structure, named filtering tree, with shared constraints (on querying data attributes) grouped in the same tree node, which facilitates the discovery of insights in a multi-focus manner to promote opportunistic findings. A preliminary qualitative laboratory study with two real-world datasets indicates the effectiveness of TimeSlice in exploring multi-faceted temporal events.

6.1 Motivation

The analysis of temporal events, common in many domains including science, engineering and humanity, allows people to discover new trends and patterns. Interactive visualizations have proven efficient in exploring event data in many formats, such as network logs [Phan et al., 2007], medical records [Plaisant et al., 1996], and musical artist history [André et al., 2007]. These temporal events are often multi-dimensional and multi-faceted. For example, in addition to the timestamp, a medical record could include other information such as hospital department, type of disease, patient gender and age, and so on. Faceted browsers enable the viewing of datasets from different perspectives, allowing users to create queries to compare events along distinct attributes. For example, in [Phan et al., 2007], users working with network logs were able to compare temporal sequences of packets filtered by IP addresses or port numbers.

As reviewed in Chapter 2, although there exist numerous visualization tools for assisting navigation of multi-faceted data (e.g., [Lee et al., 2005, 2009; Spenke and Beilken, 2000; Spenke et al., 1996]) and temporal events (e.g., [Plaisant et al., 1996; Wongsuphasawat and Gotz, 2012; Wongsuphasawat et al., 2011]), few has the abilities of both supporting effective filtering and browsing over faceted data attributes as well as flexible comparison and exploration of data along its temporal aspect. Further, when datasets are large and users may only have vague hypotheses, they often have no ideas about which facet or attribute to filter and drill down. Thus the discovery of serendipitous findings requires multi-focus visualization which simultaneously presents several parts of data with different querying criteria while preserving the context of queries (i.e., queries sharing the same attribute constraint on one facet but different on another).

To address these needs, we designed TimeSlice (Figure 6.1), an interactive faceted browsing tool for temporal events, which provides a flexible approach for constructing, comparing, and manipulating multiple queries over the facet space of data. A dynamic filtering tree visualization is proposed to systemically organize and represent all the queries created by users. Through multi-focus interaction, the filtering tree allows users to easily browse timelines of both the queried events and those that are closely related, which is important for the discovery of serendipitous findings. Tree nodes (representing attributes) and tree levels (representing facets) can be manipulated directly, which offers efficient navigation across different perspectives of data.

6.2 Design Considerations

In this design study, we focused on facilitating the visual exploration of temporal event repositories by event attributes, i.e., faceted browsing. In order to understand what a user needs in exploring both the temporal and faceted space of data, we conducted multiple sessions of interviews with a professor at Earth and Planetary Science department of a university. He has conducted research and taught classes in this field for over 30 years. His research usually involves analyzing and correlating a huge collection of events in the histories of earth (e.g., specific changes of the environment), life (e.g., the emerging of certain species), and humanity (e.g., the beginning and ending of important reigns, the birth and death of critical people, etc.) spreading in a large time scale (e.g., millions of years). Those temporal events are often characterized by a number of attributes. For example, in studying critical people (e.g., famous scientists, artists, and philosophers) in human societies, they could have different meta-data attributes such as nationality, profession, gender, and so on. Browsing timelines of specific groups of people could provide insights and generate thinking of our history, such as "what types of people are likely emerging during what kinds of society?"

Based on our literature research (see Section 2.1.1 and Section 2.1.2) and the feedback collected from our interviews, we identified the following design requirements for the aspect of faceted browsing in multi-attributed temporal events exploration.

R1 Visual representation of queries. Users often drill down temporal events through specifying constraints of data attributes facet by facet, creating dynamic queries to filter data. The system should provide intuitive and flexible visual representations of the queries, rather than a number of abstract conditions.

R2 Systematic organization of queries. When users create many queries along with the exploration of data by making ad hoc decisions, the system should offer mechanisms to organize those queries effectively, as well as to allow users go back and forth in each exploratory querying progress.

R3 Contexts of queries and filtred data. When users have no or only vague hypotheses, they need to explore the faceted attribute space of data by trying many similar queries. To promote opportunistic findings in this visual exploration process, the system should provide the context of data currently focused, i.e., data items that can be presented by performing similar queries but are filtered out in this particular subset of data.

R4 Logical operations. Some logical manipulations on data attributes within a facet, such as *AND* or *OR*, need to be supported to create advanced data queries based on elementary constraints on only particular attributes.

Moreover, built upon the user requirements distilled from our previous design studies focusing on the exploration of time-series values (Chapter 3 and Chapter 4), we found that many of the more general design guidelines were also aligned with our interview results here, including: direct manipulation of visual objects; multi-focus and multi-scale exploration of timelines; side-by-side comparisons; overview first, zoom & filter, details on demand; and clear visual feedback (e.g., animated transitions).

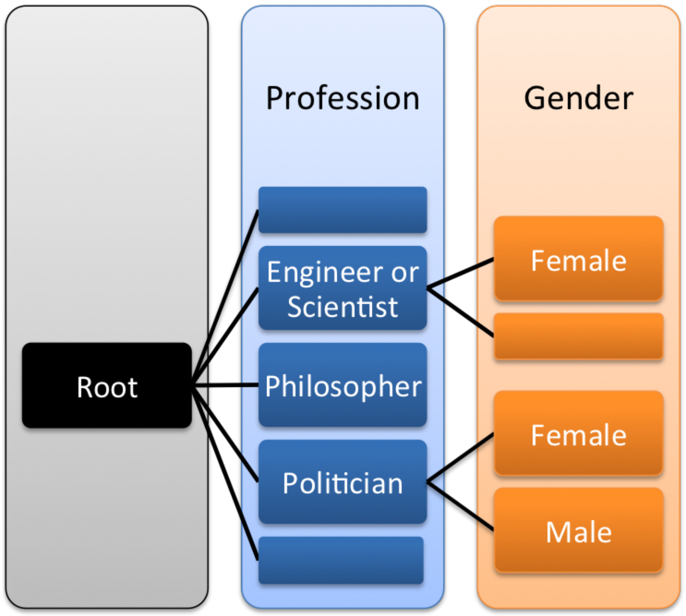


Figure 6.2: The abstract logical representation of the filtering tree visualization in TimeSlice: each tree level represents a facet, and each tree node represents a constraint of the attributes on that facet. The tree structure and color-codings correspond to the visualization in Figure 6.1-b.

6.3 Hierarchical Query Organization

In light of the aforementioned design guidelines, we designed a novel visualization called *filtering tree* (Figure 6.1-b), which is the core component of the TimeSlice interface. The filtering tree visually present and organize data queries in a hierarchical structure by grouping of queries with the same prefix in terms of attribute constraints on facets (R1, R2). These queries determine which temporal events should be displayed in the corresponding event timelines (Figure 6.1-c).

Figure 6.2 depicts an abstract logical representation of the filtering tree visualization in Figure 6.1-b, where each tree level denotes a facet and each tree node represents a querying constraint of data attributes on that facet. Thus, every path from the root node to a leaf node represents a single querying process by walking along the path while performing filtering operations with the constraints. In this way, the filtering tree can not only reveal the exploration history of user drilling down the data (R2), but also provide the information contexts of queried data in a compact manner (R3), for example, the tree nodes without attribute labels indicate the non-focused queries but having similar constraints. Tree nodes can be dynamically combined to support logical operations of data attributes (R4), which indicates a logical *OR* among grouped attributes. Logical *AND* is not supported in this prototype because we assume orthogonal faceted classification of the data (i.e., each event only has one attribute on one facet). Thus, a logical *AND* of attributes on the same facet would generate an empty set. Other direct manipulations on the filtering tree, such as reordering tree nodes and tree levels, will be discussed in details later.

6.4 TimeSlice Interface

The TimeSlice visualization system is a web application developed with the Microsoft Silverlight platform[2]. Its interface contains the following components (Figure 6.1): a) a facet container showing the available facets that can be added to the filtering tree, b) a filtering panel showing the current filtering tree for querying the events, c) a main window for browsing events timelines that are dynamically updated according to the filtering tree, and d) a navigation control that supports multi-scale exploration of the temporal events.

6.4.1 Manipulating the Filtering Tree

The exploration of the dataset usually starts with an empty tree in the filtering panel (Figure 6.1-b), which presents an overview of the entire dataset in the main window. To drill down, a user can add a facet to the filtering tree by selecting it from the facet container (Figure 6.1-a), and the whole dataset are sliced into multiple timelines with items having the corresponding attributes on that facet. This automatic and simple manner of adding all the attributes on the selected facet and slicing the entire datasets according to them do not require users to explicitly specify a constraint on the facet. It is especially effective for visual exploration with vague or no hypotheses of the data, because it displays all possible queries on the facet of interest, so that the user just needs to visually compare the queried timelines and identify possible regions of interests to explore further. As successive facets are added, the height of the filtering tree is increased, which makes it possible to further slice the exploration space. For example, in Figure 6.3-a, the user has added the orange *Gender* and purple *Region* facets. Next the user can choose to selectively subdivide the original event timelines with the attributes of each newly added facet, like expanding a tree node, in order to build new queries based on existing ones.

A user can expand an attribute node by clicking the blue round plus button that appears at the top-left corner of each facet. In Figure 6.3-a, the user expands the *Male* value of the *Gender* facet to the *Region* facet in the next tree level that contains five distinct attribute values. (Figure 6.3-b). Similarly, a node can be collapsed by using the minus button at the same place, which allows users to go back along their historical exploration path of creating the queries.

[2]http://www.microsoft.com/silverlight/.

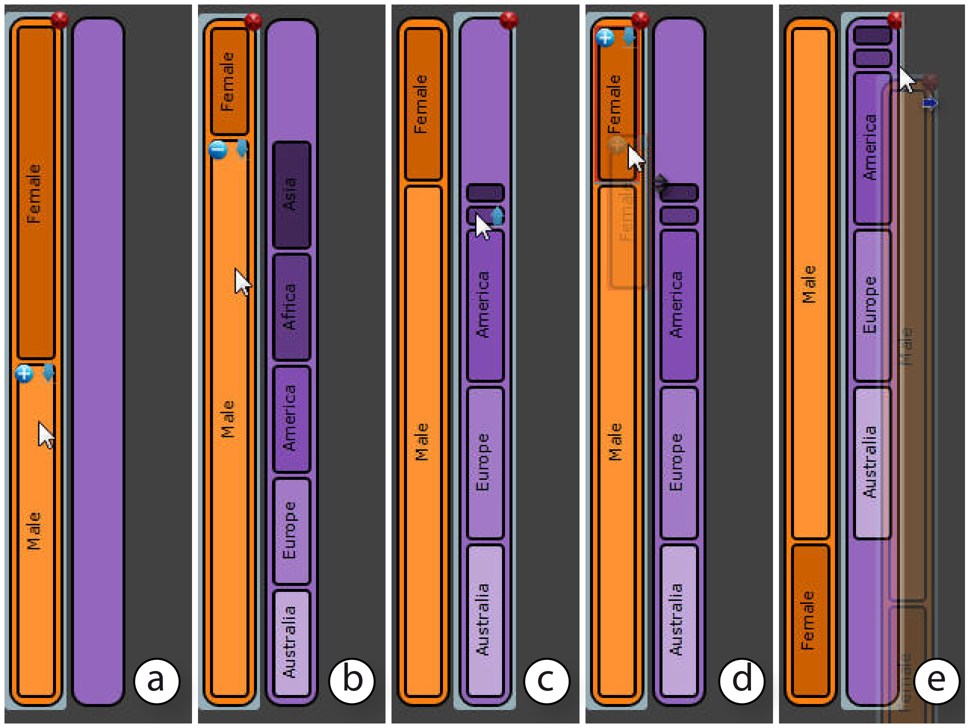


Figure 6.3: Direct manipulations on the filtering tree for creating dynamic visual queries.

When a user is not interested in certain attribute values, she can minimize the corresponding node in the filtering tree. For example, Figure 6.3-c indicates the filtering tree after the user has minimized the *Asia* and *Africa* attributes on the *Region* facet. When a tree node is minimized, the system displays a compact representation of event timelines as shown in Figure 6.1-f. In TimeSlice, we implemented this compact form of timelines as color-coded heat-map bars. These minimized timelines save the screen real estate for displaying the timelines of greater interest, while retaining the access to other timelines. The minimizing operation allows users to both keep an interaction history and easily get back to previous queries, thus supporting serendipitous discoveries with a manner of multi-focus visual exploration. To further facilitate the visual exploration, TimeSlice also allows users to reorder attributes within a facet (Figure 6.3-d), and switch facets in the filtering tree (Figure 6.3-e), using “drag-and-drop” interactions, providing a fluid user experience using direct manipulations.

In addition to comparisons of queried event timelines that are created by specifying only one attribute on each facet, TimeSlice allows users to group multiple attributes on the same facet, thus creating a filtering tree with queries that are constrained by more than one attribute value of specific facets. To do so, the user needs to first select the attribute nodes then press the group button in Figure 6.1-g. For example, the first maximized timeline groups *Engineers* or *Scientists* on the *Profession* facet (blue) as shown in Figure 6.1-b and Figure 6.2. This grouping function enables a logical *OR* operation on querying attributes, thus supporting many cross-concept comparisons of event timelines with advanced queries.

6.4.2 Browsing and Comparing Timelines

TimeSlice provides basic timeline operations such as zooming and panning the whole timeline canvas in both x and y directions or each individually. These operations are achieved by using the navigation control (Figure 6.1-d) or directly manipulating the canvas with the mouse. When the content of an event timeline requires more vertical space, users can use scrollbars to interact with it individually without scaling the whole canvas vertically. The scale of the whole canvas can also be reset at any time. Though only basic browsing functions are implemented in this prototype since we focused more on the faceted browsing aspect, other more advanced multi-scale exploration techniques, such as stretching and squishing of time (Chapter 3) and the lens-based visualization (Chapter 4) presented in previous design studies, as well as other techniques in the literature (e.g., StackZoom [Javed and Elmqvist, 2010]), can be easily integrated to facilitate the navigation along the time dimension of data.

Once event timelines are created based on the dynamic filtering tree, users can interact with them in several ways. First, users can change the view types of displaying the event items in the timeline, thus supporting the comparison of temporal events in multiple analytical purposes. For the famous people dataset, we implemented a histogram view which shows the distribution of people lives (Figure 6.4-a), a lifespan view which displays each person as a horizontal bar (Figure 6.4-c), and a mini-lifespan view with highly aggregated visual representations of the lifespan view (Figure 6.4-b). The lifespan-based views use the width of a rectangle to encode the length of the person's life, which is also color-coded as the background of the rectangle in a red-yellow scale (where shorter lives are more red). The lifespan rectangular bars are carefully aligned without overlaps on the timeline.

Second, when a user hovers over an item in a timeline, a tooltip is displayed with context-based information such as a figure and some description of the person (Figure 6.1). In addition, when the mouse is hovered inside a timeline, its associated query path in the filtering tree is highlighted, allowing users to examine how the event timeline is constructed and what the common attributes of the temporal events are.

Third, to get more details, a user can click an event item, and then a popup is initiated to view appropriate additional information, such as a list of all the people inside a bin (Figure 6.1-g) when in the histogram view, or a web search results of the selected person when in the lifespan view.

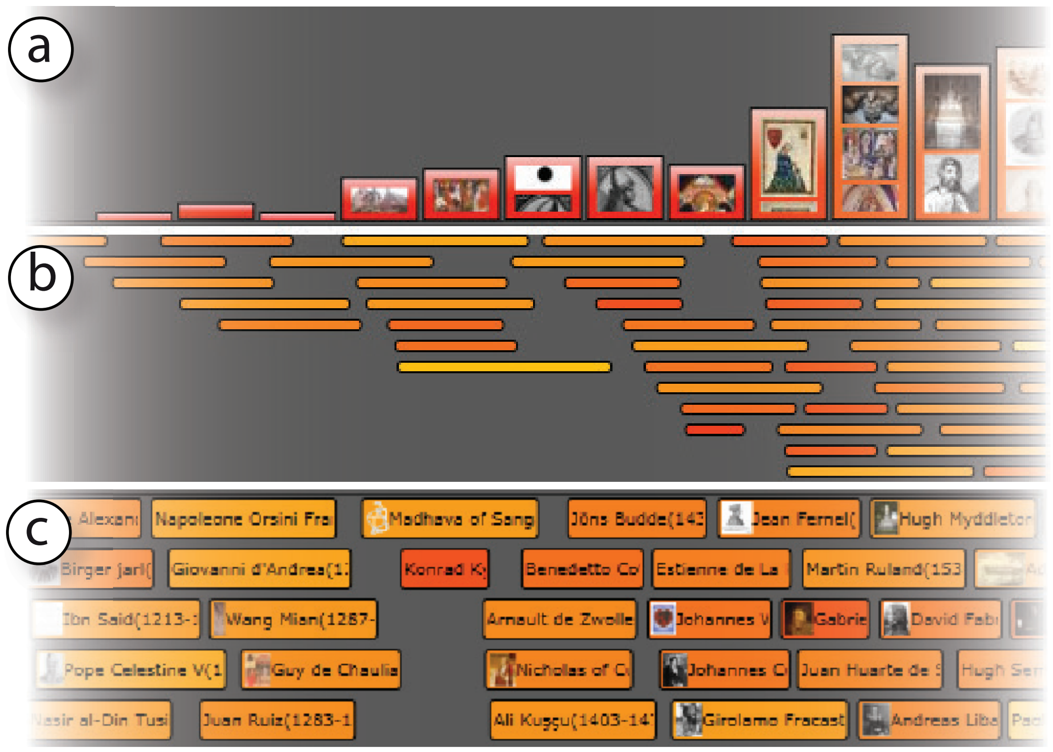


Figure 6.4: Different visualizations of the life spans of historical people on a timeline: (a) a histogram view showing the distributions of people's lives, (b) a mini-lifespan view to highly aggregate each person's life in time with a rectangular bar, (c) a lifespan view that is a expanded version of (b) with a thumbnail and a text label of the person represented by each bar.

It is important to note that the framework of the TimeSlice filtering tree structure is independent of the representations of temporal events. For multi-faceted event datasets with less rich information, such as network logs where the number of events within a time-span is the important quantity, other types of timeline views can be included, for example, bar charts, line charts, and bubble charts. Moreover, for timelines with combined querying attributes in the filtering tree, overlays of multiple plots of different attributes can be displayed for more accurate comparisons. Similarly, the compact form of event timelines is independent of the mechanism, and can be selected appropriately to include other types of small multiples.

6.5 Preliminary Qualitative Study

In this section, we describe a qualitative laboratory study to assess whether the design of TimeSlice is helpful for exploring and comparing multi-faceted temporal events.

6.5.1 Datasets

Two real-world datasets, daily US flight delays (1989-1991), and famous historical people, were used in the study. The first dataset contained airline delay events classified into three orthogonal facets: *airline region* (east, mid-west or west), *flight elapsed time* (short or long), and *arrival delay* (early, on time, within 1 hour or greater than 1 hour). The second dataset contained richer information for each temporal event item, i.e., a person, in which the facets include: *gender* (male or female), *profession* (artist, writer, scientist, politician or philosopher), and *region* (Asia, America, Africa, Australia, Europe, or America). In total, the first dataset had around 500,000 flight delay events, and the second dataset had 2845 people.

6.5.2 Participants and Apparatus

We recruited eight volunteers, six males and two females, aged from 22 to 30, who were university graduate students. All participants were generally familiar with visual interfaces and temporal data. We performed the study using IE 9.0 web browser on a 13.4-inch display (resolution 1280$\times$800 pixels) laptop driven by Windows 7.

6.5.3 Tasks and Procedure

Before the study, participants were given a brief demonstration about the major features of TimeSlice, including operations on the filtering tree, multi-scale exploration of timelines, and visual representation settings. After that, they were encouraged to play with the system a bit and asked the experimenter any questions they might have. During the study (detailed below), participants were asked to address a series of structured questions, and then they were given unstructured time to explore data with the system further.

During the structured time of the experiment, the daily US flight delays dataset was used, so that each queried timeline showed a series of delay events in the histogram view (see Figure 6.5). Participants were asked to perform two types of tasks with this dataset – identification tasks and comparison tasks. With each task group, the difficulty of tasks were increased gradually in terms of the number of facets or attributes involved, and the complexity of operations required, such as grouping of attributes. Table 6.1 shows the detailed questions of all the tasks, where there were seven identification tasks and six comparison tasks in total. The identification tasks allow the user to get familiar with the TimeSlice system and be prepared for the comparison tasks later (that were more complicated). For each task, participants were presented with the question, and were asked to find the correct answer as quickly as possible. Once participants felt confident about their answers, they reported to the experimenter. If the answer was not correct, some hints or guidance were given until participants got it right in order to discover real issues of the interface and make sure they fully understand the question.

	Identification Tasks
I.1	Create a timeline of short elapsed time and midwest region.
I.2	Create a timeline of long elapsed time and early arrival delay.
I.3	Create a timeline of long and short elapsed time, midwest region, and within-one-hour arrival delay.
I.4	Find flight delays in Feb 1990 of the timeline of long and short elapsed time and early arrival delay.
I.5	Find flight delays on March 2, 1991 of the timeline of midwest region and early arrival delay
I.6	Identify the quarter with the highest number of delays in timeline of long elapsed time, midwest region, and within-one-hour delay.
I.7	Identify the quarter with the lowest number of delays in timeline of long elapsed time and east region.
	Comparison Tasks
C.1	Compare timeline of long elapsed time and midwest region with timeline of short elapsed time and east region, are they similar overall?
C.2	Among all timelines of east region with different arrival delays, which one has the highest number of delays?
C.3	For east region and midwest & west region flights with either short or long elapsed time, which timeline has the lowest number of delays overall?
C.4	Among early flights with different regions, find the timeline that you think is most similar to the timeline of ontime arrival delay and midwest region.
C.5	Compare flight delays in Quarter 1, 1989 in the timeline of early arrival delay and midwest region with that in the timeline of greaterhour arrival delay and east region, which one is larger?
C.6	Compare flight delays in Feb 1991 in the timeline of early arrival delay and east region with that in the timeline of greaterhour arrival delay and all the regions, which one is larger?

Table 6.1: User tasks performed in the structured time of the study.

During the unstructured (exploratory) time of the study, the people dataset with richer information of the event items was used. Participants were encouraged to explore the dataset freely by using all the functions of the system for around 15 min. They were instructed to "think aloud" so that the experimenter could monitor how TimeSlice was used to visually explore the data and construct hypotheses.

In the end, we followed with a semi-structured interview to discover participants' general perceptions about the system design and its strengths and weakness. The whole session of study lasted about 45 min for each participant.

6.5.4 Experimental Results

Here we report our observations and user feedback obtained from the above experiment.

Overall Feedback

Participants were able to get familiar with the TimeSlice system quickly; both the filtering tree representations and operations seemed straightforward to users. Overall, all participants considered that the multi-focus and multi-scale visualization of timelines with different attributes in TimeSlice made comparisons very convenient. One commented: "*I really like to open up an attribute and you get all the related queries along the way of drilling down into the data*", and another commented that "*You don't need to make tons of almost identical queries and add the results one by one for comparison.* They also found logically combining faceted attributes were important for creating relatively complicated queries. One participant said that "*It would be better to have other types of set operations in addition to this union operation.* Users especially liked the direct manipulations of the filtering tree which made the operations"*obvious and simple*. Moreover, all the participants thought the TimeSlice interface was aesthetically pleasing and the animations were intuitive.

Structured Time Results

Each identification task or comparison task was accurately completed in less than a minute. On average, participants spent a little longer on tasks regarding the number of delays in a specific time period. This may be because users had to zoom and pan the timelines to find particular data points. Participants sometimes relied on the order of facets stated in the questions. After completing some of the tasks, they realized that adding facets to the filtering tree in another order would reduce a lot of clicking operations as three participants made comments about: "*I should do the facet reordering first*". In general, little assistance was needed for participants to complete all the tasks. However, 3 users were reminded to use the grouping operation of attribute nodes or the reordering interaction of facets during C6.

There were some unexpected insights discovered by participants outside the requirements of the tasks. Particularly, in one task a participant found that for busy seasons like Christmas time there were much larger numbers of flight delays for greater than 1 hour compared to the sum of all other attributes on the facet arrival delay across all three regions (parts in the yellow box in Figure 6.5), and especially in Dec 1990 the delays of the mid-west region were extremely high (the third timeline in Figure 6.5).

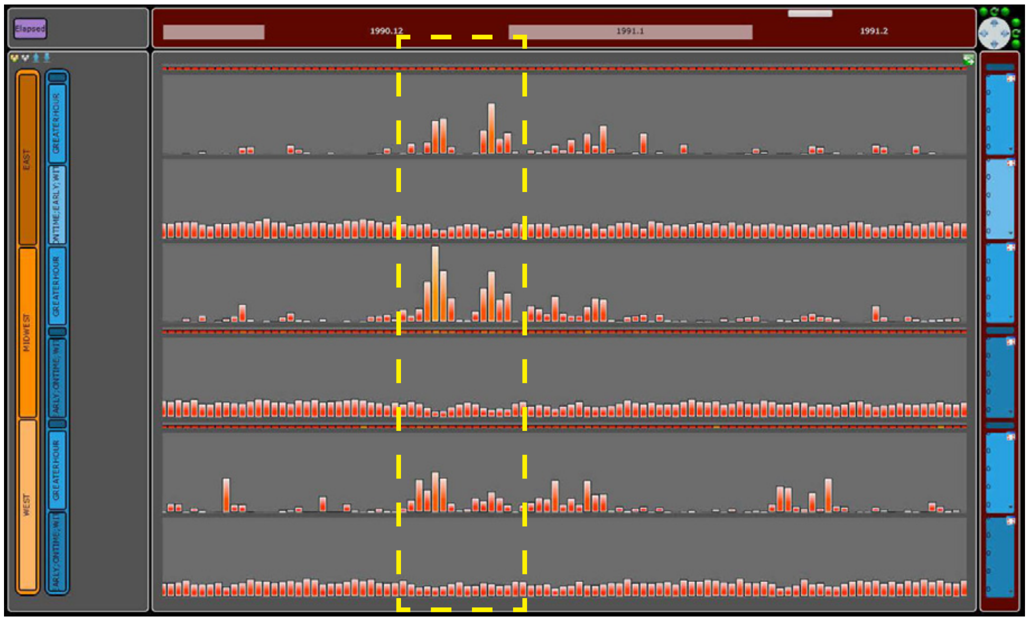


Figure 6.5: A participant is exploring the US daily flight delay dataset with TimeSlice.

Unstructured Time Results

During the unstructured time, we found that some participants were able to make creative comparisons and obtain insights about the data. For example, one compared writers and artists versus scientists and engineers from different continents. The participant's rationale was that people in those two groups of professions did jobs in two distinct ways – subjectively or objectively. To do so, he first added the *profession* facet, then grouped occupations into the above two categories, and further expanded the grouped attribute node into a new *region* facet. He found that these two classes of people had similar trends in time – the overall lives distributions in Europe were longer and started earlier, whereas those in America were shorter and started later. Another participant identified that there was a burst of the number of artists around 1900AD, after browsing the sub-timeline for the artist attribute on the *profession* facet. He then wanted to explore the gender distribution of those artists by expanding this node to a new *gender* facet, and found that males were dominant all the time. Next the participant added the *region* facet and tried to examine where the male artists were from, and he found that most of them were male American artists. Two of other participants were interested in exploring great philosophers in the history. By adding the *profession* and *region* facets, they observed that big clusters existed in Asia and Europe for early years and in America for recent years, which matched their expectations. Similar patterns were also found for writers, which implied "*great thoughts emerged in the history*". Another two participants browsed famous scientists in the history, and found that peaks were in Asia and Europe for early ages and in

America for recent time, after displaying the sub-timeline for each continent. Another thing they observed was that there was a huge drop of the number of famous scientists between 1250AD and 1450AD in Europe. One participant mentioned that it might be related to the known European crisis during the late middle age. Also, significant burst of the numbers was detected after this period, which reflected the Renaissance.

As for the visual representation aspects of the people dataset, six participants thought the lifespan views were very attractive. They were also excited with the function of searching a specific person online by simply clicking the lifespan bar in the view. But one participant mentioned it might introduce bias by putting bars of unrelated people together. Another participant commented that it would be better to enable users to drag the lifespan bars vertically in order to bring people of interests on top.

Usability

We also identified several usability issues of the current TimeSlice interface. The first one is the labeling of the time-axis. The system only has tick marks on top of the main window, which makes it difficult to identify a data point at a specific timestamp for timelines aligned at the bottom. One participant suggested adding a vertical line that followed the mouse cursor across the main window and showing the values of data points passing the line with tooltips. Second, participants complained that once the timeline canvas was scaled only along the y-axis, they lost track of the associated queries although the corresponding nodes in the filtering tree were highlighted — "*Probably zooming the filtering tree along with timelines is better*. Third, participants found the current side-by-side layout of timeline comparison was not adequate for detailed comparisons of values of data points, thus other plot layouts such as overlaying of timelines should be supported. Fourth, participants were "lazy in minimizing timelines unless it was necessary and they said it required too many clicks to leave only the wanted timelines maximized. Therefore some common operations should be provided such as expanding an attribute node with only the selected sub-timelines maximized.

6.6 Discussion

The TimeSlice system is designed to facilitate multi-focus faceted browsing and querying of temporal events with multidimensional attributes, especially to promote opportunistic findings when users are less clear about their exploration goals and hypotheses. The design study, although demonstrate that TimeSlice is powerful and flexible in supporting various important tasks in visually exploring multi-faceted datasets, reveals several limitations of the prototype that could be further improved.

First, the functionalities of TimeSlice are limited in multi-scale exploration of the temporal dimension of datasets. In addition to the complexity of faceted attribute structures, that is the major focus in this design study, many real-world data often expresses large time scales, such as the data concerned by our expert here. A number of existing timeline browsing techniques can be easily integrated into TimeSlice to address those issues. For example, focus+context interfaces with magnification lenses [Furnas, 1986] are possible to be built into the current prototype. On the other hand, timelines can be stacked in different scales to mimic the StackZoom technique [Javed and Elmqvist, 2010]. More advanced techniques used in the previous design studies (e.g., Chapter 3 and Chapter 4) can also be considered for complicated visual analysis tasks that involve the temporal aspect. For example, the squishing and expanding interactions of time scale and the MagicAnalytics Lens technique in KronoMiner can be integrated into TimeSlice to support temporal analysis of events.

Moreover, in this design study, we only considered one type of facet with categorical attributes when developing TimeSlice. But real-world datasets usually contain multiple heterogeneous facets, which could be categorical, ordinal, and numerical. The extension to supporting ordinal facets is straightforward, in which we can restrict the attribute reordering operation on those facets and at the same time introduce sorting, providing an awareness of attribute orders to users. The integration of numerical facets is tricky but also doable. For example, when opening a numerical facet in the filtering tree, instead of showing all possible attribute values as the categorical facets, the system could offer users options to create a number of thresholds on the associated tree nodes, thus discretizing the continuous value range on numerical facets. Those thresholds can be later adjusted dynamically to trigger re-querying of temporal events and form new timelines.

The third limitation is that TimeSlice assumes that each event item can only have one attribute in each facet. In some faceted classification systems, an object can belong to multiple attributes within the same facet, for example, when exploring academic publications, a paper can contain multiple keywords (i.e., keyword is the facet used to classify papers here). A simple and intuitive way of solving this problem is to duplicate the items that would appear in results of multiple queries represented by the filtering tree. Further, brushing and linking techniques should be incorporated to highlight associated items in the view.

6.7 Summary

This chapter has presented a design study on developing dynamic visual exploration techniques for multi-faceted temporal events, which has approached the challenges arising from browsing the large and complex semantic space expressed by data attributes. The core component of the research prototype, TimeSlice, is a general visualization framework for querying faceted datasets, i.e., the filtering tree. It allows users to dynamically create and systemically organize data attribute queries with direct manipulations, promoting the discovery of opportunistic insights with multi-focus visual exploration. In the preliminary qualitative lab study, participants were able to easily complete the tasks and identify interesting patterns in the famous historical people dataset. Specific contributions have been made in this chapter include:

- a set of design guidelines for interactive querying and browsing of multi-attributed datasets,
- a generic visual query framework, dynamic filtering tree, that facilitates faceted exploration of data by representing the processes and contexts of queries in semantics, and
- a visualization system, TimeSlice, that implements the query framework in the context of supporting navigation and comparison of multi-faceted temporal events.

The datasets concerned in this design study contain two important aspects: the faceted attributes and the temporal dimension. Although our focus in this chapter is the visual exploration of data attributes that construct the information space of facets, the visualization prototype also implemented several basic visual encodings and interactions to facilitate the temporal navigation. However, to fully support a variety of analysis tasks in multi-faceted temporal events, techniques and ideas discussed in Chapter 3 and Chapter 4 need to be integrated for a more powerful and complete visual analytics system (as mentioned in Section 6.6).

In the upcoming chapter, we will present a second design study for further investigating multi-focus visualization techniques that support interactive exploration of data attributes (and their faceted space). As we have only explored categorical facets in this chapter, the following design study extends the basic concept of the filtering tree framework to accommodate dynamic visual querying of more generic information space: multi-faceted dataset consisting of data points can be freely arranged (i.e., without a special temporal dimension), and with heterogeneous facets (i.e., including categorical, ordinal, and numerical attributes).

Chapter 7

PIVOTSLICE: REVEALING IMPLICIT AND EXPLICIT RELATIONS IN MULTI-FACETED DATA

Nothing is perfect. Life is messy. Relationships are complex. Outcomes are uncertain. People are irrational.

Hugh Mackay

Chapter 6 introduced the first design study on multi-focus visual exploration of data attributes, where the proposed prototype, TimeSlice, only focuses on temporal events with categorical faceted attributes. However, as mentioned in the beginning of Chapter 6, in many real-world applications, a multi-faceted dataset could be heterogeneous, containing categorical, ordinal, and numerical attributes. Thus, in this chapter[1], we further investigate multi-focus visualization techniques supporting visual exploration of a wider range of data attributes with different natures. Values of attributes on those heterogeneous facets often reveal implicit relationships between data items, e.g., how one attribute is affected by another. There often exists another type of relationships in data, which is explicit, such as relational references (i.e., links) between objects in a network. While extensive research has been done in either of the fields for exploring explicit (graph and tree visualization techniques; see Section 2.1.3 and 2.1.4) and implicit (faceted browsing techniques; see Section 2.1.2) data relationships, few addresses both aspects at the same time. However, the analysis of datasets consisting of both explicit and implicit relationships is critical in many real-world applications

[1]Main portions of this chapter were previously published in Zhao et al. [2013] © 2013 IEEE. Reprinted, with permission, from: Jian Zhao, Christopher Collins, Fanny Chevalier, and Ravin Balakrishnan, Interactive Exploration of Implicit and Explicit Relations in Faceted Datasets, IEEE Transactions on Visualization and Computer Graphics. Thus any use of "we" in this chapter refers to Jian Zhao, Christopher Collins, Fanny Chevalier, and Ravin Balakrishnan.

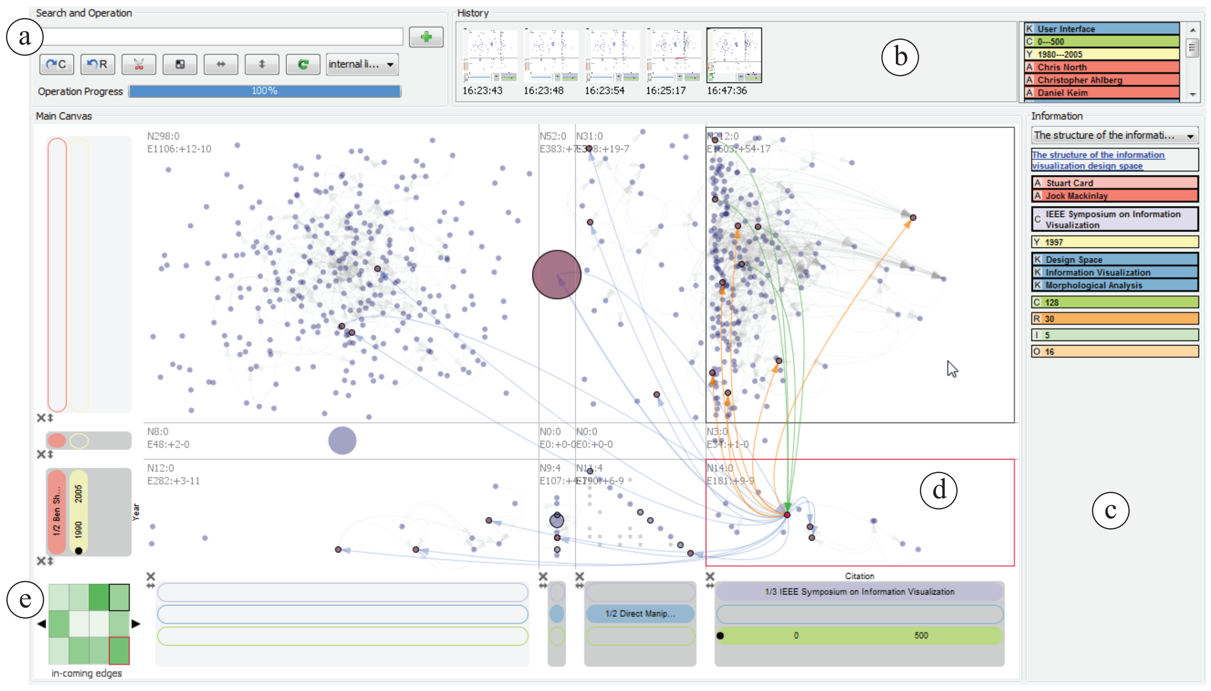


Figure 7.1: The main PivotSlice interface: (a) Search and Operation Panel, (b) History Panel, (c) Information Panel, (d) Main Canvas, and (e) Cell Relation Panel. The user is conducting faceted exploration by sub-dividing the entire data into 3-by-4 regions and discovering relationships among different sub-datasets, using the a table of queries constructed by two sets of filters on both axes. A video demo of the system can be downloaded here.

to discover cross-concept insights. One example type of data commonly seen is multi-faceted and multidimensional networks, such as user friendship graphs on social media and citation graphs in scientific literature collections.

In this chapter, we present a design study situated in the context of exploring academic publication databases. These datasets consist of a large number of data items (e.g., publications) and a variety of attributes associated with each publication (e.g., authors, keywords, and years). References and citations make up the explicit relations between articles, while implicit relations can be formulated by queries on specific properties, such as “share the same author(s) and be published before 2000”, “contain the same keyword(s) and be widely referenced after a few years”. Using the academic publication domain as our testbed, we designed PivotSlice (Figure 7.1), an interactive visualization technique which provides efficient faceted browsing as well as flexible capabilities to discover data relationships. With the metaphor of direct manipulations, PivotSlice allows users to visually and logically construct a series of dynamic queries over the data based on a multi-focus tabular view that subdivides the entire data with customized semantics. PivotSlice further facilitates the visual exploration process through features including live search and integration of online data, graphical interaction histories, and smoothly animated transitions. This chapter also describes the evaluation of PivotSlice through a preliminary qualitative lab study with university researchers and reports the findings from our observations and interviews.

7.1 Motivation

In many application domains, the datasets people want to explore and analyze are becoming not only larger in size but also richer in structure and semantics. Datasets can have explicit relations between entities, consisting of links between data items that are intrinsically defined (e.g., friendship between actors in a social network, or railways between cities in a transportation network); but also implicit relations, which are correlations between entities that can be derived from their faceted attributes (e.g., actors in a social network that share the same language and profession, cities in a transportation network that have more than one airport and exhibit a population increase).

The size scale of this type of data suggests a manner of multi-focus and multi-scale visual exploration which provides progressive, dynamic, and flexible access of various parts of data at different levels of detail [Stolte et al., 2002; van Ham and Perer, 2009; Wattenberg, 2006]. Moreover, the structural aspect of datasets, that are expressed with heterogeneous attributes associated with data items, indicates that facet-based exploration is necessary in addition to the direct and navigational search, allowing users to freely narrow down and view data across various dimensions [Dork et al., 2012; Lee et al., 2009; Yee et al., 2003].

Particularly in the application domain — exploring scientific literature — concerned in this design study, we focus on supporting visual analytics of datasets which include two types of relationships: the explicit referential relationships of the core data items (e.g., paper citations), and the implicit correlations and trends between different subsets or certain attributes of data (e.g., publications by authors versus years). Improved interfaces for analysis and exploration of academic publication data could lead to better discovery of relevant background research, a holistic understanding of emerging topics in research to inform policy and funding, and the establishment of new collaborations between researchers.

However, few existing visualization systems have adequately addressed all the above issues. There are two main large bodies of research — faceted classification systems (see Section 2.1.2) and multivariate network visualizations (see Section 2.1.4). The former focuses on mechanisms of pivoting and dynamic filtering but lacks visualization of the structural overview and explicit relationships in datasets. The latter provides many efficient means of displaying and manipulating the topological layout of data, but they usually have limited exploration abilities for implicit relations, where only small numbers of attributes or a single type of facet (e.g., numerical) are allowed, often through visual encoding (e.g., coloring) the nodes of a graph.

To fill this gap, we designed a novel visualization system called PivotSlice (Figure 7.1) that provides effective faceted exploration capabilities while revealing various kinds of data relationships and correlations. Our core visualization design is based on a dynamic tabular view which represents a set of logical visual queries performed by a user progressively filtering and slicing the information space, which facilitates a multi-focus and multi-scale exploration of the whole dataset. With direct manipulations, individual or groups of data attributes can be dragged, split, combined, and pivoted to construct a customized tabular sub-division of the whole dataset, supporting a freeform faceted exploration with heterogeneous attributes. Based on the node-link view and flexible graph layout, PivotSlice provides the visual representations of both explicit referential relationships across all data items and implicit trends and correlations between different facet attributes.

7.2 Visual Query Language

In the design process of PivotSlice, we conducted a 2-hour interview and discussion session with a group of 14 university researchers (including graduate students, post-docs and professors) about how they explore and organize scientific publications and find interesting topics in their fields, in order to determine the essential elements of interactive faceted browsing tools for their tasks. We found that many of their comments were aligned with the design guidelines introduced in the previous design study (see Section 6.2). Although Chapter 6 focused on the exploration of multi-faceted temporal events, the design considerations on how users perform faceted browsing remained similar.

We thus developed a visual query language to facilitate the visual exploration of more general multi-faceted datasets, by extending the original filtering tree idea in Chapter 6. As Figure 7.1 shows, in PivotSlice, implicit relations, such as trends and correlations, are revealed through the layout of data items based on user-defined queries; and explicit relations, such as cross-referencing, are revealed by directed edges drawn between items in the view. As depicted in Figures 7.1 and 7.2, PivotSlice logically organizes a set of user-defined queries in a *Query Table* view (Figure 7.2-a) made of a matrix of *Query Cells* (Figure 7.2-c). The Query Table layout is built according to the two groups of *Data Filters*, shown as gray rectangles (Figure 7.2-b), on the horizontal and vertical *Query Axes* (Figure 7.2-d). A Data Filter contains several *Facet Panels* (Figure 7.2-e), displayed as color-coded bars, each of which composed of several attribute constraints.

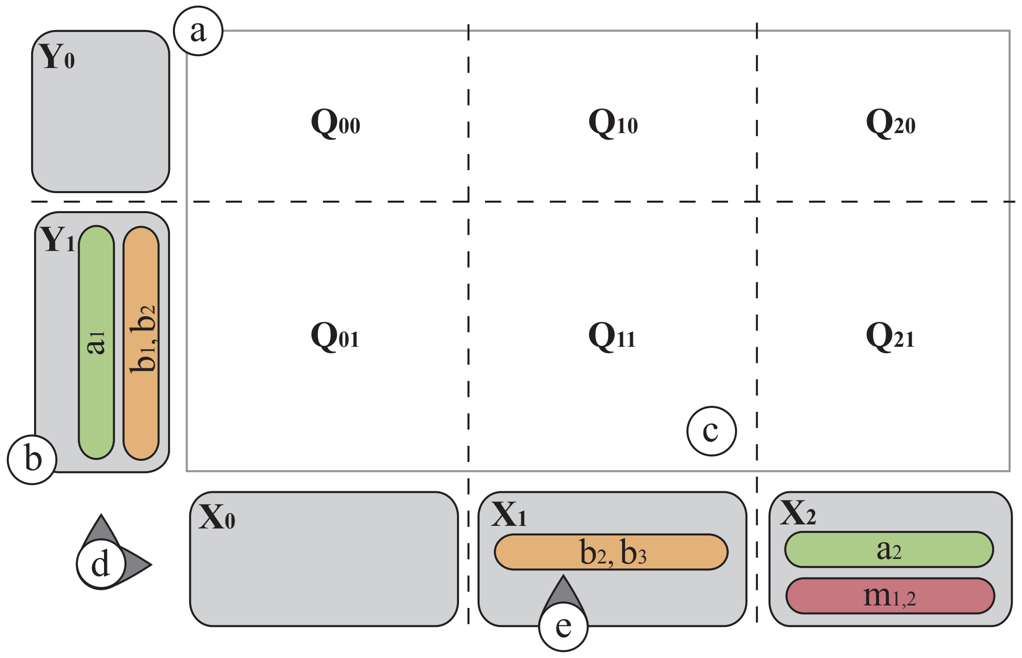


Figure 7.2: An illustration of the visual query logic in PivotSlice.

Let us consider a multi-faceted dataset $\mathcal{D}$ of data entries d. We note the categorical facets as $A, B, \ldots$, and a_i the ith attribute value of the facet A. For numerical facets, we use the notation $M, N, \ldots$, and $m_{i,j}$ as the interval of values within $[m_i, m_j]$ on facet M. Further, let $A(d)$ be the attribute value of facet A for the entry d. Based on the universal relational model — which states that "one can place all data attributes into a table, which may then be decomposed into smaller tables as needed" [Hawryszkiewycz, 1984], the visual query language in PivotSlice subdivides a logical map of all data entries into a dynamic table according to the criteria (i.e., Data Filters) defined by the user. Figure 7.2 shows such an example of dividing the information space into a 2-by-3 grid constructed with filters $\{Y_0, Y_1\} \times \{X_0, X_1, X_2\}$.

The data query associated to each cell is evaluated in three steps:

S1 Each Facet Panel corresponds to a logical *OR* operation on the selected attribute values. For example, the Facet Panel B of Data Filter X_1 corresponds to $\{b_2 \cup b_3\} = \{d \in \mathcal{D}, B(d) \in \{b_2, b_3\}\}$. The numerical Facet Panel M in X_2 corresponds to $\{m_{1,2}\} = \{d \in \mathcal{D}, M(d) \in [m_1, m_2]\}$. We chose the *OR* operation for attributes on a Facet Panel to make it conceptually consistent between numerical and categorical facets: since a query on a numerical facet corresponds to a range of values equivalent to a *OR* relation over all values in the interval, we thus use the same operation for categorical facets.

S2 Each Data Filter corresponds to a logical *AND* operations on the basic queries of its Facet Panels. For instance, the Data Filter Y_1 forms the query $\{a_1 \cap (b_1 \cup b_2)\} = \{d \in$

$\mathcal{D}, A(d) = a_1$ and $B(d) \in \{b_2, b_3\}\}$. This logical *AND* operation was inspired by the filtering tree structure introduced in TimeSlice (chapter 6), so users can combine attributed specifications on different facets to form a compound query.

S3 Each Query Cell corresponds to a compound query $Q_{i,j}$ with a logical *AND* operation over the two resulting queries of the corresponding Data Filters X_i and Y_j, such as $Q_{2,1} = \{(a_1 \cap (b_1 \cup b_2)) \cap (a_2 \cap m_{1,2})\}$.

Following this process, the whole dataset is sliced into multiple subsets filled with items satisfying the constraints of every Query Cell. When query results overlap, data items are duplicated in multiple cells.

The top-most row and left-most column correspond to the "others" or "remaining" filters, that represent the complement query to all other queries on the corresponding Query Axis. For example, $Y_0 = \mathcal{D} \setminus Y_1 = \{\overline{a_1 \cap (b_1 \cup b_2)}\}$ and $Q_{1,0} = \{\overline{a_1 \cap (b_1 \cup b_2)} \cap (b_2 \cup b_3)\}$. In particular, the top-left cell contains data items that are not compatible with any of the other Query Cells. This allows a better understanding of the Query Table structure since the location of the cells remains consistent when the user dynamically adds or removes Data Filters.

With the above visual query language, a user is able to frame data from a range of perspectives with customized semantics. Multiple different facets can be examined simultaneously by specifying faceted queries in Data Filters on the two axes, because those queries are flexible and independent with each other. In other words, a user can create as many Data Filters as possible and place any Facet Panel inside to create queries as she likes.

7.3 PivotSlice Interface

The main user interface of PivotSlice is composed of five interactively coordinated views (Figure 7.1): (a) the *Search and Operation Panel* which provides the meta-data searching abilities and essential functions on data management and view configuration; (b) the *History Panel* which shows the recent states of the visualization, allowing users to undo or redo operations; (c) the *Information Panel* presenting the detailed attributes of visualization objects that the user is currently interacting with; (d) the *Main Canvas* displaying the dynamic visual representations of the dataset based on user interactions; and (e) the *Cell Relation Panel* which summarizes the overall relationships among different subsets of the data on the Main Canvas.

Based on the results of our interviews as well as the knowledge we gained from our literature survey and the previous design study (Chapter 6), we developed the interface of PivotSlice based on a set of design rationales as detailed later. These can be also taken as general guidelines for designing visualization systems of faceted dataset exploration.

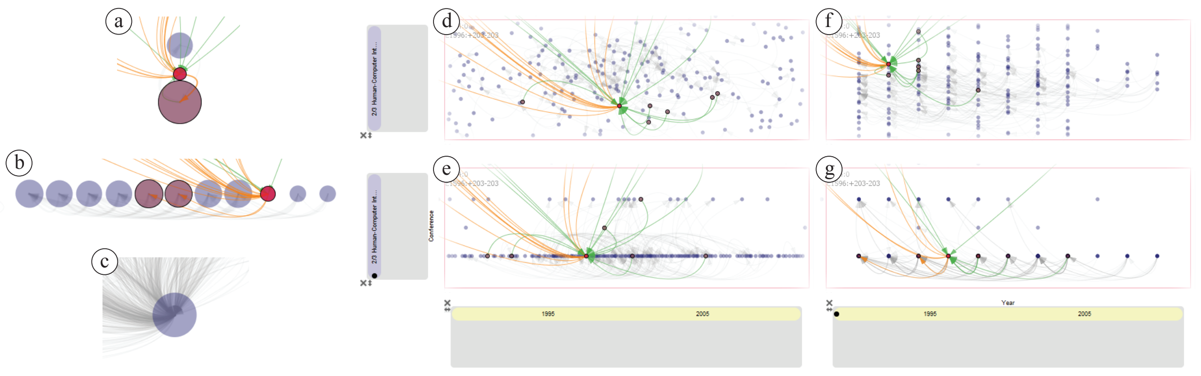


Figure 7.3: Different visual layouts of nodes within a cell: (a)-(c) aggregated, (d)-(f) force directed, and (e)-(g) facet aligned layouts (the vertical axis is the *Conference* facet with 3 attributes and the horizontal axis is the *Year* facet with range 1990–2010).

7.3.1 Multi-Focus and Multi-Scale Exploration

Numerous previous studies have indicated that multi-focus visualization, which allows the user to browse several parts of a dataset simultaneously at higher levels of detail while maintaining the context, is an efficient way to explore graphs, relational databases, faceted datasets and other forms of data [Hadlak et al., 2011; Stolte et al., 2002; van Ham and Perer, 2009; Wattenberg, 2006]. In particular for large scale datasets, integrating multi-focus views and level-of-detail representations of data objects has been shown to facilitate navigation.

PivotSlice supports the multi-focus exploration of faceted datasets by allowing users to systematically divide the dataset into meaningful subsets via dynamic queries in a tabular form. This enables convenient comparisons of different parts of a dataset across user-defined semantics by viewing the Query Cells along columns or rows. To make best utilization of the screen real estate, the size of each table row or column in the Main Canvas is allocated dynamically based on the number of nodes contained in it.

To view the subsets of data at different scales, a user can minimize rows or columns of the tabular view by clicking the “arrow” button at the corner of each filter, which results in aggregating the nodes in the corresponding cells (see the 2nd column and the 2nd row in Figure 7.1). Additionally, the aggregation views of a cell can be initiated by using the horizontal and vertical “arrow” buttons on the Search and Operation Panel, which does not minimize the entire row or column.

The form of node aggregation is determined by the current cell layout method (i.e., nodes can be aligned with one of the facets on each axis; see Section 7.3.2 for details) and its visual status (i.e., minimized or not). Specifically, if a facet on the axis to be minimized is configured as the layout-aligning facet, the nodes will be aggregated into buckets corresponding to a single attribute on that facet. For example, along the vertical *Conference* facet in Figure 7.3-a, and along the horizontal *Year* facet in Figure 7.3-b; otherwise, all the nodes will be grouped as one aggregated node (Figure 7.3-c). The size of an aggregated node is mapped to the number of actual publications in it. Depending on the node aggregations, the directed edges connecting the same nodes are aggregated as well, where the line thickness is mapped to the number of merged links.

7.3.2 Information Seeking with Direct Manipulation

Our design of PivotSlice follows the principles of creating efficient visualization systems suggested in many fundamental works including the well-known visual information seeking mantra [Keim et al., 2008; Russell et al., 1993; Shneiderman, 1996]. Particularly, we further extend Shneiderman [1996]'s seven general abstract tasks to the following essential elements — *Overview First, Pivot and Slice, Relate and Extract, Details on Demand, and History Available* — supporting the iterative knowledge discovery process in faceted data exploration. Moreover, we design the interaction of these features in PivotSlice with direct manipulations and smooth animations which can greatly ease the use, learnability, and understanding of the system [Bezerianos et al., 2010; Shneiderman, 1983], aligned with the design practices in previous chapters.

Overview First

PivotSlice provides a flexible overview of the entire dataset all the time with the multi-focus and multi-scale Query Table, where data items are placed dynamically according to the user's exploration interests in the implicit relations (facets) in the data. The global topology of network data referential patterns (i.e., the explicit relations) is presented by combining the visualizations in every Query Cell with: 1) either default node-link views by applying a force-directed layout algorithm [Fruchterman and Reingold, 1991] or 2) matrix-style views with nodes aligned diagonally (the 3rd cell in the bottom row in Figure 7.1). The user can selectively display the network edges using the options on the Search and Operation Panel, including "no links", "cell-internal links", "cell-crossing links" and "all links". At the top-left corner of each cell, a summary of the sub-network information such as the numbers of total nodes and edges is displayed (Figure 7.1). Further, the aforementioned various forms of node aggregations, such as the node grouping and the minimization of rows and columns, allow the easy access of the overall data at different levels of details.

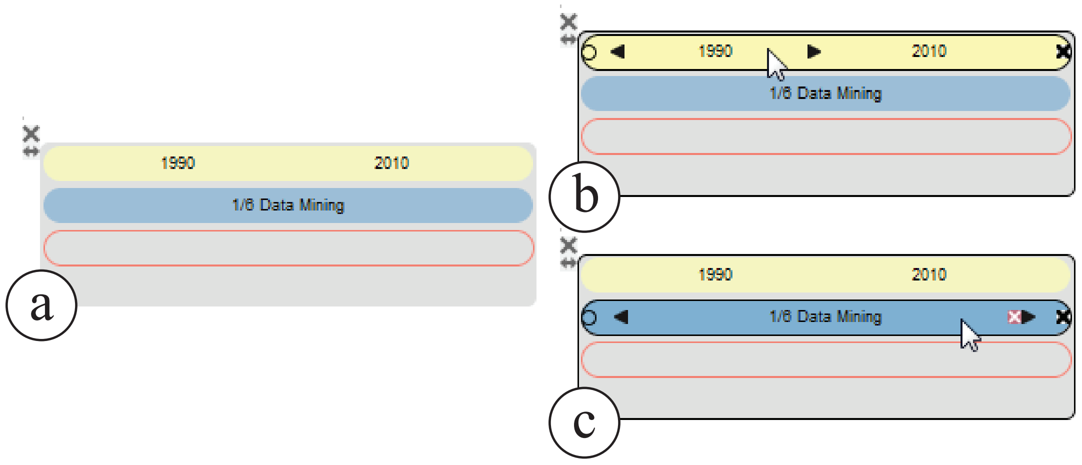


Figure 7.4: The visual encodings of a filter (with a year facet of 1990–2010, a keyword facet of 6 attributes, and an empty author facet): (a) the default view, (b) hovering over a numerical facet, and (c) hovering over a categorical facet.

Pivot and Slice

Pivoting is a widely-used browsing method in many faceted visualizations [Dork et al., 2012; Dunne et al., 2012; Lee et al., 2009]. PivotSlice extends the traditional pivoting concept into a more general and flexible filtering technique by systemically organizing a set of dynamic queries in a table. As in the aforementioned visual query language, the filters composing the query support complicated logical operations on multiple attributes and facets as well as categorical and numerical values. With the interactive slicing of a dataset in the Query Table, a user can explore and compare the data by building her own semantics as described later in Section 7.5.

Visual Query Encodings. In PivotSlice, as shown in Figure 7.1 and Figure 7.4, a Data Filter is represented as a gray round rectangle and a Facet Panel is displayed as a color-coded rectangular bar inside the filter[2], in which a hollow bar with only the outline indicates no attribute associated with that facet in the filter. For categorical facets, attributes are rendered as textual labels on the Facet Panel. For numerical facets, a continuous range of values is shown by two separated labels for the starting and ending numbers.

Slicing Interactions. The user can progressively slice the dataset by interactively adding filters to the column or the row on the Query Axes. A new Data Filter can be created if an attribute is drag-and-dropped onto the "others" filter representing the complement query (always the first one on each Query Axis). This newly-created filter adds one more column or row to the existing tabular division of the data. To drill down the data, the user can keep placing attributes of any facet onto existing filters (except the "others" filter), which updates

[2]The color scheme is derived from ColorBrewer, http://colorbrewer2.org/.

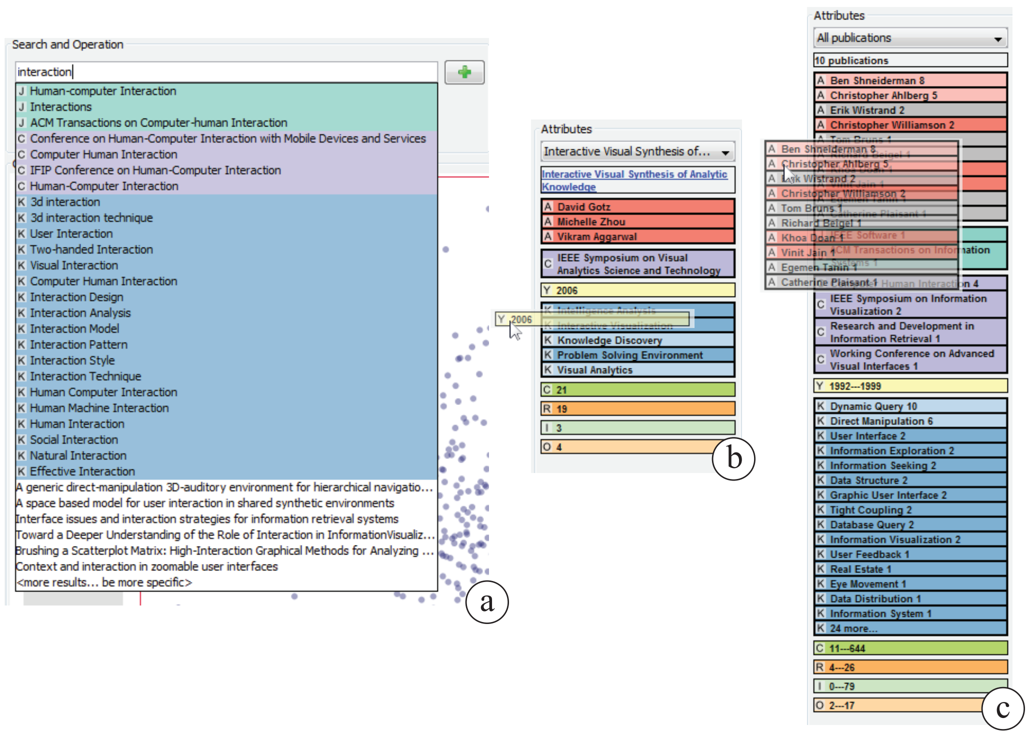

Search and Operation
interaction
J Human-computer Interaction
J Interactions
J ACM Transactions on Computer-human Interaction
C Conference on Human-Computer Interaction with Mobile Devices and Services
C Computer Human Interaction
C IFIP Conference on Human-Computer Interaction
C Human-Computer Interaction
K 3d interaction
K 3d interaction technique
K User Interaction
K Two-handed Interaction
K Visual Interaction
K Computer Human Interaction
K Interaction Design
K Interaction Analysis
K Interaction Model
K Interaction Pattern
K Interaction Style
K Interaction Technique
K Human Computer Interaction
K Human Machine Interaction
K Human Interaction
K Social Interaction
K Natural Interaction
K Effective Interaction
A generic direct-manipulation 3D-auditory environment for hierarchical navigatio…
A space based model for user interaction in shared synthetic environments
Interface issues and interaction strategies for information retrieval systems
Toward a Deeper Understanding of the Role of Interaction in InformationVisualiz…
Brushing a Scatterplot Matrix: High-Interaction Graphical Methods for Analyzing …
Context and interaction in zoomable user interfaces
<more results… be more specific>
a
Attributes
Interactive Visual Synthesis of…
Interactive Visual Synthesis of Analytic Knowledge
A David Gotz
A Michelle Zhou
A Vikram Aggarwal
C IEEE Symposium on Visual Analytics Science and Technology
Y 2006
K Intelligence Analysis
K Interactive Visualization
K Knowledge Discovery
K Problem Solving Environment
K Visual Analytics
C 21
R 19
I 3
O 4
Y 2006
b
Attributes
All publications
10 publications
A Ben Shneiderman 8
A Christopher Ahlberg 5
A Erik Wistrand 2
A Christopher Williamson 2
A Tom Bruns 1
A Richard Beigel 1
A Khoa Doan 1
A Vinit Jain 1
A Egemen Tanin 1
A Catherine Plaisant 1
J IEEE Software 1
J ACM Transactions on Information Systems 1
C Computer Human Interaction 4
C IEEE Symposium on Information Visualization 2
C Research and Development in Information Retrieval 1
C Working Conference on Advanced Visual Interfaces 1
Y 1992---1999
K Dynamic Query 10
K Direct Manipulation 6
K User Interface 2
K Information Exploration 2
K Information Seeking 2
K Data Structure 2
K Graphic User Interface 2
K Tight Coupling 2
K Database Query 2
K Information Visualization 2
K User Feedback 1
K Real Estate 1
K Eye Movement 1
K Data Distribution 1
K Information System 1
K 24 more…
C 11---644
R 4---26
I 0---79
O 2---17
c

Figure 7.5: Adding new attributes in PivotSlice via (a) searching on the Search and Operation Panel and (b)(c) dragging from the Information Panel.

all the queries and data items in cells of the corresponding row or column. Also, when the user double-clicks a Facet Panel, the Data Filter will be separated into several filters with each containing one attribute of that facet (for numerical facets, each separated filter will contain one unit in the attribute range, such as 1 year).

Pivoting Interactions. There are two ways of adding a new attribute to a filter in PivotSlice to further pivot the dataset. First, the user can type keywords in the multi-functional search box to open a drop-down list of matching categorical attributes, which can be selected and added onto a filter by pressing or dragging across the green "plus" button (Figure 7.5-a). A prefix of the keyword string, such as "/author Mark", can be specified to restrict the searching in meta-data on one specific facet. For numerical facet, a range of values can be added via similar interactions by first typing the specification strings, such as "/year 2005-2010". Second, as shown in Figure 7.5-b, the user can grab single attribute labels from the Information Panel. Multiple categorical attributes can be selected through standard multiple selection interaction then dragging the group and dropping it on the Query Axis to form an OR query between all selected attribute values (Figure 7.5-c).

Relate and Extract

The ability to view relationships among data objects is a key function in visualization applications to support visual analytics [Keim et al., 2008; Shneiderman, 1996]. Especially for faceted datasets, not only does the topology of explicit referential relationships of data items need to be presented, but also the implicit cross-concept relations and trends between different facets.

Revealing Within-Cell Relationships. For one particular subset of data in a Query Cell, PivotSlice supports the discovery of trends and relations between different attributes by allowing flexible layout of data items. As Figure 7.3-d indicates, the default view is produced by force-directed layout [Fruchterman and Reingold, 1991]. When the user sets a layout-aligning facet for only one axis (by toggling the black circle on the Facet Panel), the nodes split into rows or columns where each contains the data instances of one specific attribute value (Figure 7.3-e,f), similar to the "attribute-driven graph layout" [Shneiderman and Aris, 2006; Wattenberg, 2006] Node positions in the "free" dimension are still determined using the force-directed layout. When both of axes have been specified with a layout-aligning facet (Figure 7.3-g), the view becomes a regular scatter-plot. Together with the aggregation views (Figure 7.3-a to -c) introduced in Section 7.3.2, PivotSlice provides flexible and powerful analytical abilities of data correlations in categorical-to-categorical, numerical-to-numerical, and categorical-to-numerical facets.

Revealing Between-Cell Relationships. PivotSlice assists the user with discovering relations at the cross-cell level in several ways. First, the Cell Relation Panel displays the inter-connection measurements between the selected table cell (highlighted in the red outline) and every other cell via color-mapping the strength of cell-to-cell connections in four different metrics (Figure 7.1-e). The connection strength values are computed by simply normalizing the numbers of total crossing edges, in-coming edges, out-going edges and overlapping nodes. Other network metrics could be easily integrated into PivotSlice. Second, the user can read the exact numbers of overlapping nodes, in-coming and out-going edges at the top-left corner of each cell. Third, when a cell is selected and another cell is hovered over, only the links between these cells are highlighted in green (out-going) and orange (in-coming), relative to the selected nodes; other links are displayed in a less saturated color (Figure 7.1).

Extracting and Modifying Queries. PivotSlice offers the ability to modify query parameters through direct manipulations at multiple interface levels, including filters, facets, and attributes. First, the entire Data Filter can be dragged: if dropped on to another filter, the queries of these two filters are merged; if dropped in the empty area between filters, the filter is inserted in place; otherwise, the dragged filter is removed. Second, at the facet level, all the attributes on a Facet Panel can be manipulated as a group in a similar manner. But alternatively, the user can delete a facet by clicking the black cross button (Figure 7.4-bc). Third, the user can pull-out one attribute by dragging the label and place it onto another filter. Other attribute-level modifications are supported: for numerical facets, the range of values can be adjusted by using the black forward and back arrows (Figure 7.4-b) or double-clicking to enter the number; for categorical facets, these arrows are used for attributes navigation and clicking the red cross button removes one (Figure 7.4-c). Moreover, the user can extract part of the visual query at the above three interaction levels without modifying the original one by performing the drag-and-drop operations while holding a modifier key.

Details on Demand

In PivotSlice, information details are presented in many forms when the user selects or hovers over a visualization object. For example, tooltips are presented for nodes, filters, and Facet Panels. Moreover, as in Figure 7.1, when a node is selected or hovered over, which is emphasized in red color, all its neighbor nodes are highlighted in lighter red and the edges connected to them are shown in green or orange (i.e., references or citations). Meanwhile, all the meta-data of the selected node appears on the Information Panel: if the node represents a single publication, its title, URL, and all attributes on the facets will be displayed; if the node contains a collection of publications (e.g., an aggregated node), the panel will show a similar summary view containing the top 15 attributes (also with the number of occurrences) of each categorical facet ranked by frequencies as well as the attribute ranges of every numerical facet in these publications (Figure 7.5-c), and the user can further select and browse an individual publication. In addition to single-node selections, the user can choose a group of nodes using a polygon selection tool, or via the context menu to make special selections, such as all the nodes in a cell and neighboring nodes with the citation/reference relation.

History Available

History mechanisms, which allow the user to undo or “time travel”, play a very important role in interface design [Shneiderman, 1996]; especially for visual analytics systems, graphical histories of representation states can effectively facilitate the analysis process [Heer et al., 2008]. PivotSlice maintains two types of interaction history (Figure 7.1). First, every time when there is a change to the visualizations on the Main Canvas, such as adding and removing a filter or a number of attributes, the state of the canvas is recorded and shown as thumbnails on the History Panel. Then the user can navigate back in time and restore to previous visualizations by simply clicking the thumbnails. Second, PivotSlice also stores a list of recently added or removed attributes on the right side of the History Panel, which can be added back to the existing filters via drag-and-drop interactions. The history mechanisms benefit the user in many ways, such as when the user accidentally deletes an object and wants to construct queries with the same attributes.

7.3.3 Dynamic Dataset

As noted in many studies [Dork et al., 2012; Hadlak et al., 2011; van Ham and Perer, 2009], the data we want to analyze today is often very large but the user may be only interested in part of it. Moreover, the dataset are distributed and continuously evolving. For example, it would be very difficult to collect all the publications into a single academic database for exploration and the data collections themselves are updated all the time. Often, research tasks involving academic papers require up-to-date data.

PivotSlice attempts to support the above nature of datasets by providing several dynamic data management functions. The user can reduce the dataset according to her needs with the "prune data" button on the Search and Operation Panel, which prunes away all data items except selected items or items within a selected table cell. On the other hand, with the integration of web search, the citations or references of selected papers can be retrieved online and added to the current dataset. Furthermore, with the Search Box, the user can initiate a web search with the desired keywords and choose a list of publications in the results to insert into the data. To highlight new data points, recently added publications are displayed in red on the Main Canvas.

7.4 Preliminary Qualitative Study

We conducted a laboratory study to access how well people can use PivotSlice to explore, make sense and discover findings of faceted datasets as well as to find any usability issues for improving the system. We used the InfoVis 2004 Contest dataset[3], consisting of a directed network of 591 nodes (publications) and 1866 edges (citations and references). The metadata of publications (gathered using the Microsoft Academic Search API[4]), consists of titles, URLs and a range of facets including authors, keywords, venues, publication years, reference numbers and citation counts. The scenario is derived from the interview results and feedback of participants in the process of our design and study.

[3]http://www.cs.umd.edu/hcil/iv04contest.
[4]http://academic.research.microsoft.com/About/Help.htm.

7.4.1 Participants and Apparatus

We recruited six participants (two females), aged 24–33, from our university network. All participants were either graduate students or research staff who had computer science or related backgrounds. We pre-screened participants to ensure that they all had at least two years research experience and had ever conducted literature searches in their fields. In order to balance the user expertise levels of our experimental dataset, we selected two participants whose research interests were information visualization and interaction in general, and others were from completely different research areas. The experiment was conducted on a desktop which was running on Windows 7 and connected with a 24-inch LCD display (resolution 1920×1200). Participants used a mouse and keyboard to interact with PivotSlice during the whole study.

7.4.2 Tasks and Procedure

Based on the experiments conducted in previous studies with similar datasets [Dunne et al., 2012; Kang et al., 2006; Lee et al., 2005, 2009] and our own experience of doing literature surveys, we developed a list of tasks for evaluating PivotSlice which was guided by the 10 low level components of analytic activities in information visualization proposed by Amar et al. [2005] (T1–T20 in Table 7.1). Although classified in those themes, some tasks required the combination and interleaving of multiple low-level analytical components and might be followed with open-ended questions. In general, we presented these tasks to the participants from simple to complicated ones. We also designed three high-level exploratory tasks (T21–T23), where the attributes were not mentioned in the low-level tasks, to evaluate PivotSlice in more general situations.

The study began with a brief introduction and demonstration (about 10 minutes) of the features of PivotSlice. Then participants were asked to complete a series of tasks shown in Table 7.1 by exploring the provided dataset. During the process, participants were encouraged to articulate their intentions of performing each operation ("think aloud"). Meanwhile, interaction logs were recorded by the software. Different approaches of finding the answers to the task questions were allowed. Participants could ask for further explanations of features of PivotSlice when necessary. In the end of study, we gave participants questionnaires to rate PivotSlice and conducted semi-structured interviews (about 30 minutes) to collect their feedback. The whole study lasted around 1.5 hours for each participant.

Retrieve Value	T1. Who are the authors of Cone trees: Animated 3D visualization of hierarchical information? T2. What are the papers cited by paper H3: Laying out large directed graph in 3D hyperbolic space?
Filter	T3. What are the papers coauthored by John Stasko and Robert Amar? T4. What are the papers with keyword Information Visualization in 2000–2002? How do papers in these years cite each other?
Determine Range	T5. What are the publication years of author Catherine Plaisant? T6. What is the range of citation counts for papers written by Chris North?
Find Extremum	T7. What are the most popular keywords of papers in conferences excluding Computer Human Interaction? T8. Which author publishes the most papers in conference IEEE Symposium on Information Visualization?
Compute Derived Value	T9. Find all the citations of the papers with keyword Information Visualization in conference Computer Human Interaction. T10. Who are the collaborators of author Stuart Card? Are there any patterns in his coauthors?
Sort	T11. Order papers with keywords Information Space or Dynamic Query, by year. Tell us one insight about the citation patterns among papers in different years. T12. Rank papers written by author George Robertson with and without Jock Mackinlay by citation count. What is the most cited paper and how do other papers cite it?
Characterize Distribution	T13. What is the distribution of Stuart Card's papers by conference? How do papers in those conferences refer to each other? T14. Compare the distributions of papers by year between conference Computer Human Interaction and IEEE Symposium on Information Visualization. What can you identify?
Find Anomalies	T15. Are there any exceptions for the conference IEEE Symposium on Information Visualization in terms of paper year trends? T16. Observe papers with keywords Information Seeking or Dynamic Query in this dataset. Is there any incomplete data in terms of citations and references? What are the papers?
Cluster	T17. Identify groups of papers in conference IEEE Symposium on Information Visualization, in terms of similar year and citation count attribute values. T18. Find clusters of papers written by Stuart Card, George Robertson, or Jock Mackinlay. Can you tell more about the collaboration patterns between them?
Correlate	T19. Compare papers containing both keywords Information Visualization and User Interface, and the ones containing neither of them. Who are the authors that publish the most in the two categories? What are the popular keywords in addition to the two above? T20. Compare papers published in years 1990–2000 and 2001–2010. Of keywords Three Dimensional, Information Space and World Wide Web, which one is more popular in the year ranges? How many papers exactly? Can you tell more about the citations across those paper categories?
High Level Tasks	T21. Tell three facts about the author Ben Shneiderman. T22. Give three comments on the conference User Interface Software and Technology. T23. Name three interesting findings for the Direct Manipulation research keyword.

Table 7.1: Experimental tasks for evaluating PivotSlice, organized by the 10 low level components of analytic activities in information visualization proposed by Amar et al. [2005], plus three high level open-ended tasks.

7.4.3 Experimental Results

Based on the interviews conducted during the design of PivotSlice and the results in our study, we derived a usage scenario for demonstrating the application of PivotSlice in practice, which will be described in Section 7.5. In this section, we report the other findings resulting from the experiment.

Questionnaire

In the post-study survey, participants completed a questionnaire of 10 questions using a 1–7 Likert scale (from strongly-disagree to strongly-agree). Figure 7.6 indicates the questions and average user ratings. Based on the results of Q1 and Q2, PivotSlice seemed to be easy to use and learn, which was fairly encouraging as novice users were able to conduct complicated querying and filtering operations in PivotSlice with a very brief introduction and minimal assistance during the experiment. For the faceted browsing and analytical capabilities (Q3–Q6), participants ranked PivotSlice with very high scores and the small variances further indicated their agreement on the effectiveness and usefulness of these PivotSlice functions. For other features, participants felt that the animated transitions between two different layouts and the abilities of importing online search results were useful (Q7 and Q9); but the graphical history component received more diverging opinions (Q8). Some participants thought the thumbnails of the historical views on the History Panel looked too similar to identify useful information about which step to go back. They suggested that "*adding textual labels of the interactions performed under each thumbnail*" and "*associating the thumbnails with the list of historical operations on the right of the panel*" would be helpful. Overall, participants ranked their desire of using PivotSlice to explore faceted data as 6 out of 7, which was very encouraging given that it was only a prototype system.

Observations

Participants were able to complete the low level tasks (T1–T20 in Table 7.1) relatively quickly, where the average time was about 1.5 minutes per task. Simple tasks that require only a few steps of operations, such as T1–T8, were accomplished roughly within half a minute. Tasks with follow-up open-ended questions took longer. Participants spent much more time (around 2–3 minutes) on complicated tasks consisting of many analytical reasoning operations and comparisons, such as T17–T20. Assistance from the experimenter was rarely needed when participants conducted T1–T15. For T16–T20, on average 1.8 (out of 6) participants were helped for each task. One common issue was that participants forgot the function of aligning data points based on an attribute, which resulted in frustrations of finding the answer to the

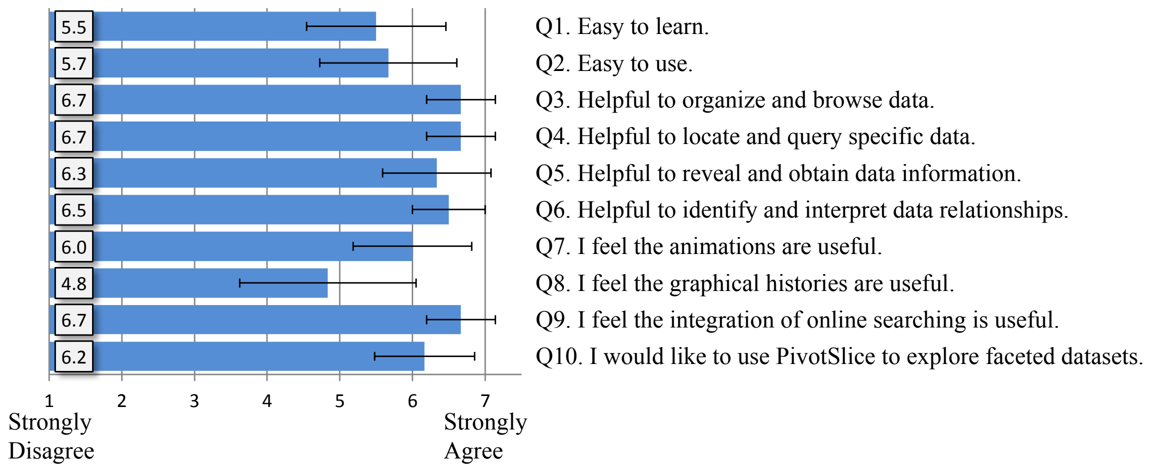


Figure 7.6: User satisfaction results from the post-study questionnaire.

task question. For example, placing and aligning nodes with *Citation* and *Year* on different axes could solve T17 easily; but they suddenly realized so with a bit more instructions, commenting in the end "*Awesome! I never thought it [PivotSlice] could do that!*". For some tasks demanding complex semantics in facet attribute layout and alignment, participants sometimes needed time for planning to solve the problem or hints on the features they should use. For example, some participants did not think of using the *Reference* and *Out-degree* facets (or *Citation* and *In-degree*) as a way to identify the incompleteness of data in T16. Participants demonstrated various approaches to complete the tasks through pivoting attributes in filters to slice the data and obtain the desired queries, especially for T19 and T20. Participants were impressed by the flexibility and effectiveness of PivotSlice. Some interesting insights could be easily identified by participants in tasks with open questions. For example in T11, one participant found that there was a trend toward fewer cross references between papers with the two keywords *Information Space* and *Dynamic Query*.

For the high level tasks (T21–T23), participants were more familiar with PivotSlice and primed to explore more deeply and use different functions to find insights. The total time spending on these three tasks was around 10–15 minutes for each participant. Generally in T21, participants looked for the year, keyword and citation distributions of *B. Shneiderman*'s publications as well as his collaborators. Participants sometimes imported papers with the online search feature, to further explore and confirm the data patterns. In T22, even after fetching more data, some participants found that more citations appeared from papers of *InfoVis* to those of *UIST* but fewer in the other direction, although both conferences contained similarly frequent keywords. When conducting the last task, some participants sought deeper findings based on previous layouts, such as investigating publications of the *Direct Manipulation* keyword for different conferences and authors. Other interesting insights found in T23 included: *Dynamic Query* was the most related keyword, and papers of such keyword tended to cite those of *Direct Manipulation* more often than the other way around.

Initial User Feedback

During the post-study interview, all participants indicated that the PivotSlice design was aesthetically pleasing and user-friendly, although a bit of learning time to understand the visual query logic was necessary. They agreed that PivotSlice was capable of flexible, efficient and adequate analytical functionalities for faceted data exploration and relation discovery. One mentioned that PivotSlice "*makes it easy for organizing complicated filters and searches*". Another said that "*it is extremely helpful in uncovering trends and distributions between data attributes [...] relating different parts of data is easy by using the connections and different alignments*". One participant who works in social network modeling research indicated that PivotSlice could be very useful in mining relationships in multivariate social networks and was eager to apply PivotSlice to his data in the future. Some of them thought PivotSlice's flexibility allowed conducting various basic analytical tasks in a unified interface, including correlation analysis, anomaly detection and similarity clustering. However, a few participants commented that too much flexibility made the PivotSlice interface complicated and difficult to learn, since "*it is sometimes confusing that you can perform the same task in many different ways*". But we argue that maintaining useful and adaptable functionalities is more critical in visual analytics.

While some participants mentioned that in the beginning they were confused about whether the logical *AND* or *OR* relations were computed for attributes of a filter, they could easily understand the rules of the query logic after a few manipulations of the dataset. One mentioned that it was difficult to remember the *AND*/*OR* operations forming the queries, suggesting that "*using similar visual cues from the filed of logical circuit design*". We also identified some usability issues during the study. One participant suggested that functions of merging adjacent cells and freely adjusting cell spaces should be implemented. Another thought that integrating short-cuts for common filtering operations, such as "*add facet and order by it*", as well as for all the drag-and-drop interactions, could speed up the data exploration process. Some commented that text labels should be displayed when the nodes were aligned or aggregated along a facet.

7.5 A Usage Scenario: Exploring Visualization Research

We here introduce the use of PivotSlice in visual biographic analytics with a usage scenario by exploring a collection of scientific literature, which was derived from our interviews in the design process and results of the laboratory experiment.

Suppose that Sally, a graduate student from the data mining group, is interested in visualization as a future area of research. But she knows very little about the field, so she wants to gain more knowledge about the literature in order to find research opportunities, discover foundational readings, and get a feel for the influential authors. Sally loads the InfoVis dataset into PivotSlice, which presents the entire citation network on the Main Canvas.

7.5.1 Uncovering Entry Points for Exploration

Sally starts by looking for papers that relate to her research expertise. Just after she types "data" in the Search box, a list of matched items drops down, including journals, conferences, keywords and so on. Within the list, she identifies some familiar conferences such as *Knowledge Discovery and Data Mining* as well as the keyword *Data Mining*. She drags the latter keyword onto the horizontal Query Axis, which results in a split of the dataset into a table of two Query Cells with smooth animation: papers containing the keyword move to the right column, and the remaining papers move to the left column. However, there are only a few papers associated to the keyword *Data Mining*. She thinks of other keywords and starts to type "visualization" in the Search Box. This time the drop-down panel suggests a list of attribute values, many of which are unfamiliar. A common keyword *Data Visualization* catches her attention. She drags this keyword onto the existing Data Filter with *Data Mining* (resulting on a *OR* query of the two keywords), and more papers move back from the left column to the Query Cell that is interested by the user.

By hovering over these nodes, Sally accesses the details on the Information Panel, and finds that these articles focus on specific domain applications. So she decides to change her exploration strategy by using more common keywords. With the selection of all the papers in this Query Cell via the context menu, she obtains a rank of frequently mentioned keywords on the Information Panel. In addition to the two keywords she just dragged, she identifies *Information Visualization*, *User Interface*, and *Visualization Technique* to be interesting general keywords for a broader exploration.

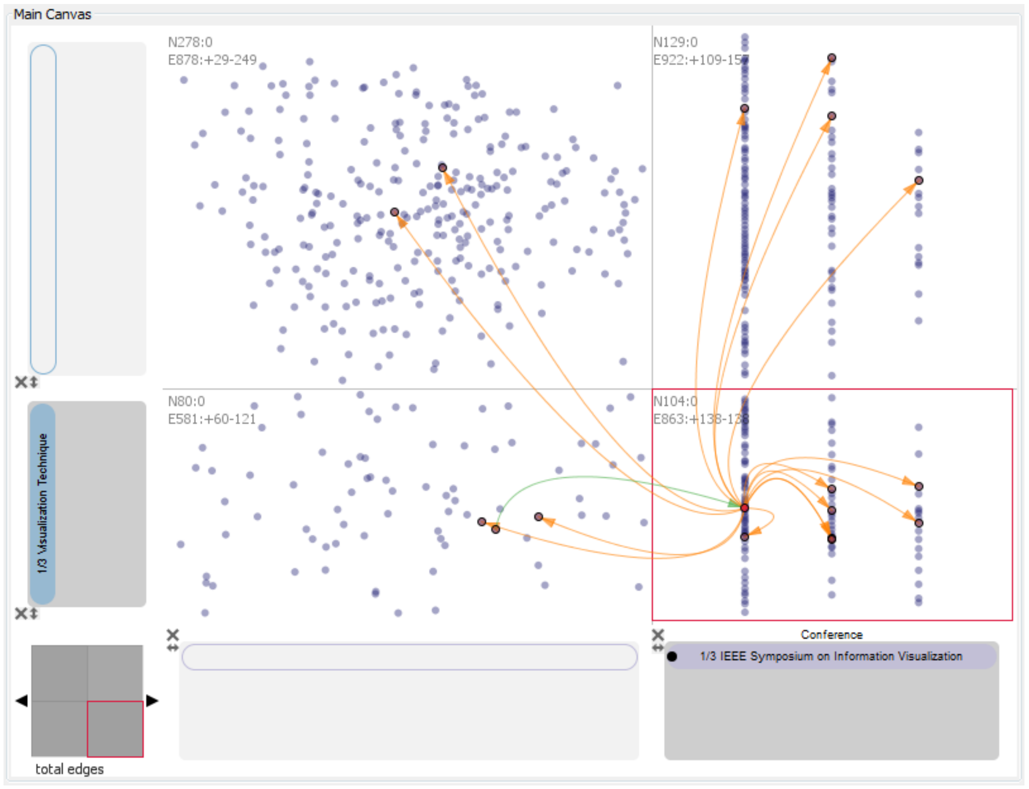


Figure 7.7: A 2-by-2 division of the dataset with queries of keywords *Information Visualization*, *User Interface*, and *Visualization Technique* versus conferences *InfoVis*, *CHI* and *UIST*, where the conference facet in the Data Filter is used for aligning the nodes.

7.5.2 Investigating Top Venues

Sally drags all three keywords onto the horizontal Query Axis to create a new Data Filter, and in the meantime clears the previous filter. To obtain the summarized information, she selects all the newly filtered nodes and looks at the ranked metadata attributes associated with these papers on the Information Panel. She finds that *InfoVis*, *CHI* and *UIST* are the top three popular conferences for the selected topics. Similarly, Sally drags these purple conference attributes to create a new Data Filter on the other Query Axis, which further divides the canvas into a 2-by-2 Query Table. To get a better idea of paper distributions over the three conferences, she aligns the nodes according to the *Conference* facet in the focused Query Cell (see Figure 7.7).

Sally observes that most of the papers matching her current interest are published at *InfoVis*. Wanting to ensure she spends time investigating important papers, she is curious about the citation patterns between the three conferences for the selected keywords. By double-clicking the *Conference* Facet Panel, she splits the original filter into three separate queries, with each conference alone as a the filtering attribute. Using the Cell Relation Panel, which indicates the overall connection relations between the selected cell and other cells, she gets a rough idea

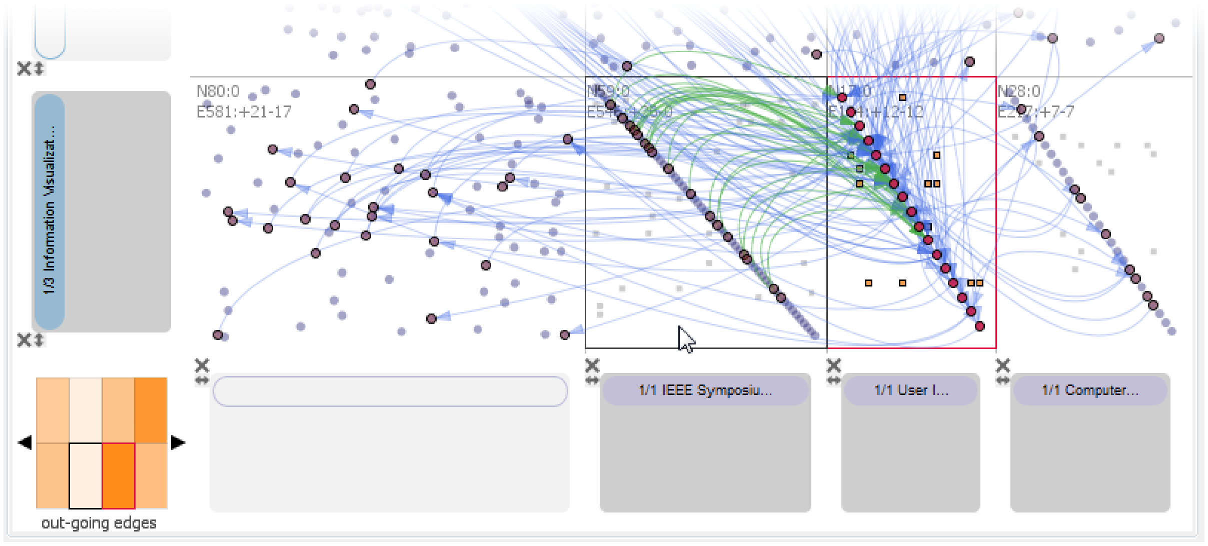


Figure 7.8: Relationships of citations between conferences of interest. After selecting the all papers from the *UIST* conference, hovering over the *InfoVis* cell indicates a lot of citations (green) from *InfoVis* and no references to *InfoVis* (orange). This can be further confirmed with the colormap of out-going edges on the Cell Relation Panel).

of the citation patterns among the three conferences. For further verification, she changes the layout of the three cells to the Matrix-Style View for more clarity, since it aligns all the nodes diagonally and declutters internal directed links by showing them as tiny squares (Figure 7.8). Sally conducts some observations of the crossing links by selecting and hovering over several papers. An intriguing insight she uncovers is that *InfoVis* cites a lot of papers from *CHI* and *UIST* but the relationship is not reciprocal, at least in this dataset.

7.5.3 Revealing Specific Hot Keywords and Trends

Next Sally wants to narrow down her exploration by looking for more specific popular topics, so she first goes back to the original 2-by-2 Query Table using the History Panel. By selecting all the nodes in the top-right and bottom-right Query Cell, which are all the papers from the three conferences, Sally identifies a range of top-mentioned keywords, in which *Direct Manipulation* and *Dynamic Query* seem to be interesting. Thus she deletes the original general keywords, and adds the two new keywords to the Data Filter.

Sally now wonders what are the trends for publications of selected topics along time. So she adds a yellow *Year* facet to the Data Filter on conferences and aligns the nodes with this facet, which results in a distribution of papers over time. To get a better view of the trends, Sally aggregates the nodes vertically in this Query Cell, where she finds that those topics are more active in years 1994 and 1995 (Figure 7.9).

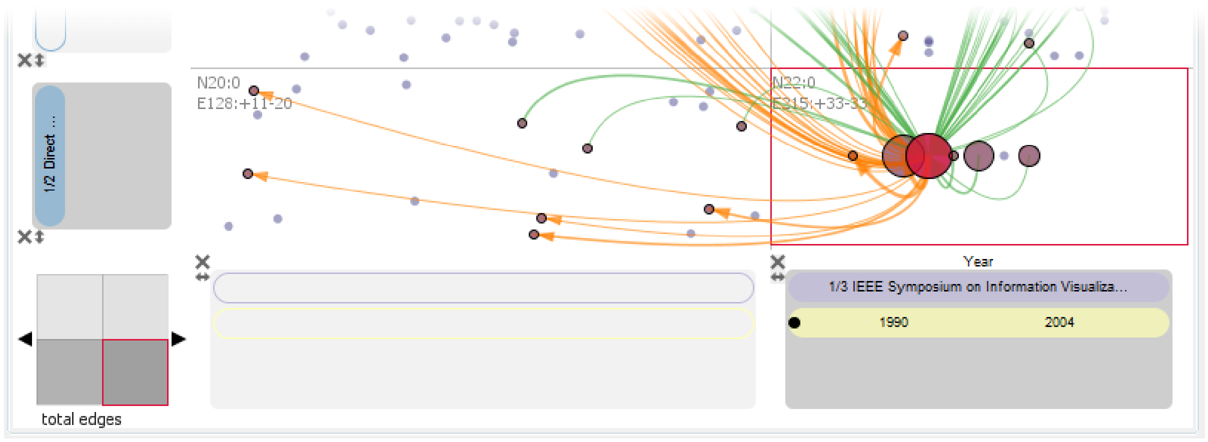


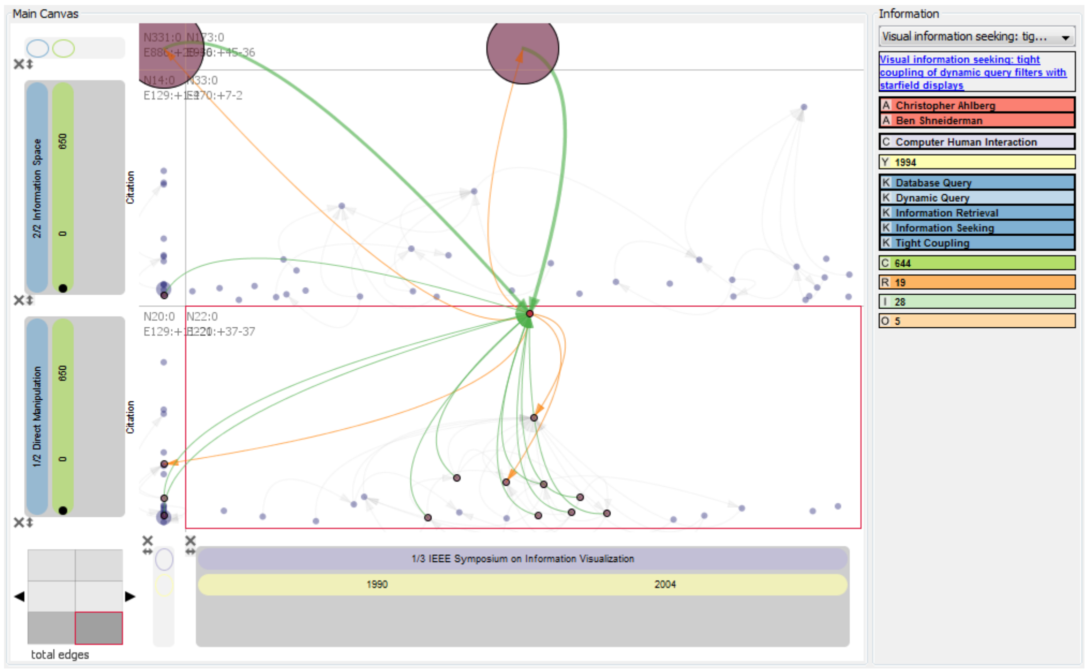
Figure 7.9: Viewing the temporal trend of papers using the keywords *Direct Manipulation* or *Dynamic Query* from conferences *InfoVis*, *CHI* and *UIST*.

To search for other more recent research topics, Sally moves her attention to the top-right cell, which contains publications in these conferences but without the two selected keywords. She selects all the papers after year 1996 in that cell, and from the list of top-mentioned keywords, she identifies two more interesting keywords: *Information Space* and *World Wide Web*. She then adds these keywords into a new Data Filter next to the original one on the vertical Query Axis, since she is not sure about which group of keywords to explore further.

7.5.4 Studying Influential Papers and Authors

To gain more knowledge about the selected topics, Sally wants to find the most influential papers in both keyword groups. Hence, she appends the green *Citation Count* facet to the Data Filters and organizes the nodes by this facet, so that more highly cited papers appear higher in the cell. The distribution of nodes clearly indicates the most cited papers. However, Sally realizes that the Main Canvas is a bit crowded as more Query Cells are generated, thus she minimizes the rows and columns that are less important (Figure 7.10).

Next Sally decides to identify the dominant authors in these fields by selecting and hovering over a number of highly-cited papers and accessing the paper author attributes on the Information Panel. After a short time of observation, she finds that *B. Shneiderman* coauthors almost all of the influential papers with keywords *Direct Manipulation* or *Dynamic Query*, and similarly *S. Card* is an influential author for the other topic about *Information Space* or *World Wide Web*. So Sally determines to further study the work of these two authors by adding them separately on the horizontal Query Axis, which allows her easily compare their work on the selected keywords.



Visual information seeking: tig...

Visual information seeking: tight coupling of dynamic query filters with starfield displays

A Christopher Ahlberg
A Ben Shneiderman
C Computer Human Interaction
Y 1994
K Database Query
K Dynamic Query
K Information Retrieval
K Information Seeking
K Tight Coupling
C 644
R 19
I 28
O 5

Figure 7.10: Identifying highly-cited publications with keywords of *Direct Manipulation* and *Dynamic Query* as well as *Information Space* and *World Wide Web*. The “others” row and column are minimized.

At the first glance, Sally finds that both authors publish papers in the two selected topics, but *B. Shneiderman* seems to mainly focus on *Direct Manipulation* and *Dynamic Query*, as she identifies only one paper he published on the other topic — *Dynamic Query for Visual Information Seeking* — which is actually a duplicated node (highlighted with a black outline in PivotSlice) that exists in the other Query Cell as well (Figure 7.11). However, Sally notices that the values of citation and reference counts for a lot of the articles do not equal the actual in-degree and out-degree attributes of the network, indicating that the dataset is quite incomplete.

7.5.5 Digging into Authors of Interest

In order to get better knowledge of the work of these two authors, Sally starts to build her own specific but more complete dataset. She first prunes the data to crop all the rest of the network except their papers, then selects a number of interesting and highly-cited publications and fetches their references and citations into the current dataset with the online search ability of PivotSlice.

After a while, Sally starts to explore this more coherent and thorough data. By placing the two authors in filters on different axes, Sally finds a book coauthored by them: *Readings in Information Visualization: Using Vision to Think*, which might be worthy looking into in

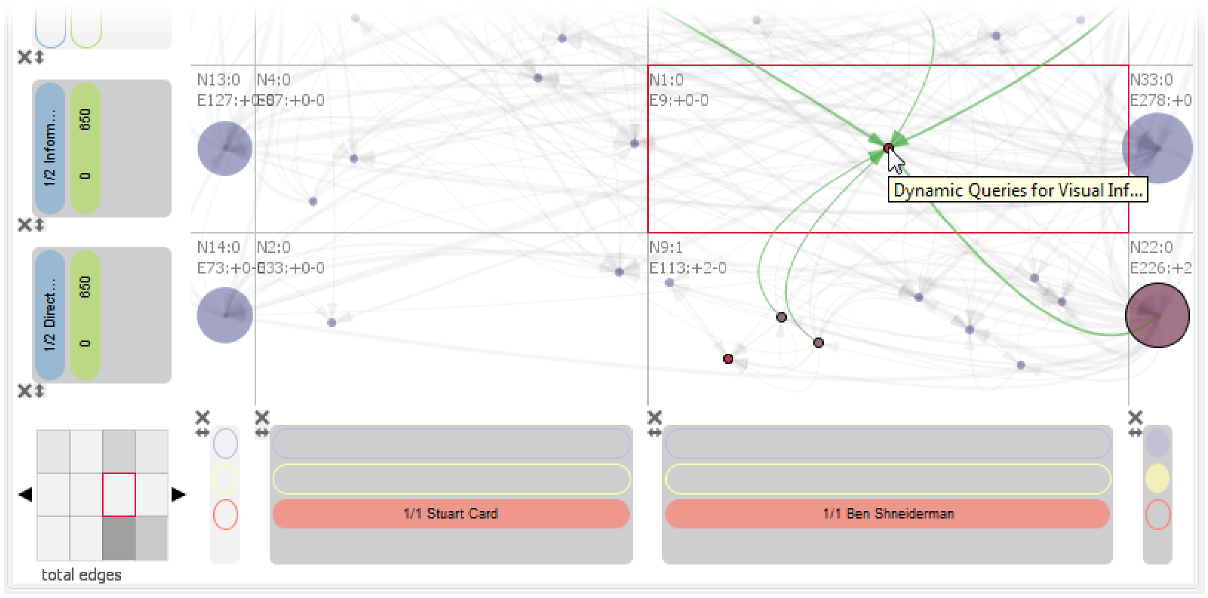


Figure 7.11: Investigating publications of authors *B. Shneiderman* and *S. Card* versus two collections of keywords.

the future. To compare the authors, she keeps both author filters side-by-side, then adds the *Year* facet to the other Query Axis and further aligns and aggregates papers by it. She identifies that *B. Shneiderman* has more active publishing records than *S. Card* in recent years (Figure 7.12). In a similar way, Sally compares their publications by adding the three main conferences explored (i.e., *InfoVis*, *CHI* and *UIST*), and an interesting pattern is that *S. Card* publishes at all of the three venues but *B. Shneiderman* seems to rarely publish at *UIST*. In the end, Sally quits PivotSlice and saves her data into a new file for future reference.

7.6 Discussion

The core interface component of PivotSlice is the Main Canvas, in which the user can construct dynamic queries organized in a table, thus to build customized semantics of subdividing the whole dataset. This visualization provides not only an efficient multi-focus and multi-scale browsing of the explicit data relation (i.e., the topology), but also a flexible manner of correlating data items based on their facets to reveal implicit trends. PivotSlice supports these kinds of exploration for faceted datasets with both categorical and numerical attributes, which can be further extended to include more general data formats.

In particular, the ordinal facet is a special type of categorical facet, which can be easily supported in PivotSlice by restricting the attribute sequence in a Facet Panel. Although we only demonstrated PivotSlice with numerical facets of integer values (e.g., years and citation

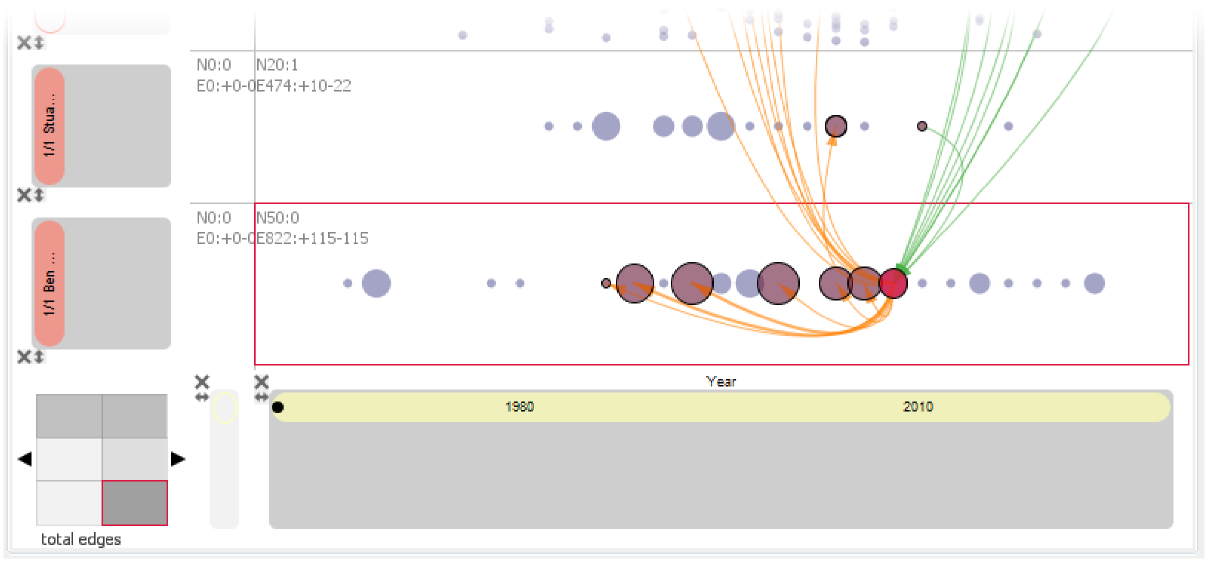


Figure 7.12: Comparing publication trends between *B. Shneiderman* and *S. Card* by year.

counts), it can further support real-value numerical attributes by simply including continuous adjustments of attribute ranges, such as using a slider, and customized binning thresholds when aligning the data items with this facet (i.e., the current binning threshold is always unit one for integer attributes). Although the research prototype was developed with certain data models tied in the Microsoft Academic Search APIs, the concept of PivotSlice design is applicable to many other types of faceted datasets, even for ones not containing referential links, where a different node layout algorithm using dimension reductions, such as the multidimensional scaling (MDS) [Borg and Groenen, 1997], could be used instead of force-directed layout. This kind of dimension reduction algorithms can reflect relationships of data items in the original high-dimensional space other than the dimensions that are currently browsed on the x-axis and y-axis, thus providing hints for users to explore other dimensions. Therefore, we believe that PivotSlice is flexible and general enough to support various kinds of faceted data existing many application domains.

Some participants found that the History Panel in PivotSlice although limited and basic, was useful since operations of slicing the workspace could be traced. This is critical in exploratory data analysis because users attempt to try a variety of approaches and then determine what to do next on the fly. The history mechanism was implemented as the common undo-redo stack in most of the software systems. But this multi-focus visual exploration enabled in PivotSlice actually reflects multiple browsing histories in parallel for different focus views. The ability to undo-redo operations for each focus individually would be an interesting extension to PivotSlice, which can be further enhanced by allowing users to bookmark particular states of the view. However, a user may not know whether the visualization in a focus is useful or not in the future. Thus, as a user explores the data, some kinds of correlation searching methods can be employed to examine the whole parallel exploration histories and suggest potential insights across past and current views.

The ability to analyze large scale datasets is a perennial challenge for designing visual analytics systems. Although PivotSlice was not evaluated with datasets containing a very large number of data items, its facet attribute space is considerably rich: 5 numerical and 4 categorical facets, consisting of 1169 authors, 1163 keywords, 50 conferences, and 63 journals. We designed PivotSlice with several scalability considerations. First, the visualization is based on a multi-focus and multi-scale exploration of the data topology, relationships and trends, combined with different aggregation views and data item layout mechanisms. When the number of nodes is extremely large, the visual clutter of nodes and links will become an issue. However, PivotSlice can be easily extended with more advanced graph layout algorithms and visualization techniques such as edge bundling [Holten and van Wijk, 2009] and focus+context graph views [Hadlak et al., 2011]. Second, the flexible data management functions in PivotSlice, such as data cropping and resource extension with online searches, make it capable of visualizing large and dynamic datasets. Users can progressively modify and load data based on their evolving interests during exploration. Lastly, PivotSlice displays summarizations of selected objects on the Information Panel, which is very helpful in identifying points of interests within large scale data.

However, to further support very large datasets and advanced analytic capabilities, PivotSlice should be augmented with smart representation of data in each cell by well-suited informative charts and visualization techniques, such as tag clouds, linked bar charts [Dunne et al., 2012], heatmaps, and linked treemaps [Fekete et al., 2003]. In this way, the advantages of each visualization can be balanced in the flexible dynamic query table thus to overcome disadvantages of specific techniques, for example, occlusion problems of node-link views when the dataset is large. Moreover, proper analytical algorithms such as MDS can be integrated in processing the data before visualizing them in the cell, in order to guide the user's exploration in large scale data following the principle of "Analyze First" in Keim et al. [2008]'s visual analytics mantra.

Although PivotSlice is designed primarily for exploring multi-faceted data, of particular interest in this chapter is the academic publication collection, which is similar to data concerned in many existing tools such as CiteVis [Stasko et al., 2013] and Jigsaw [Gorg et al., 2013]. Despite the fact that almost all the tasks listed in Table 7.1 can be achieved in CiteVis and Jigsaw, some of them are easier in PivotSlice due to the multi-focus faceted semantic subdivision of data. For example, faceted exploration tasks are difficult in CiteVis because it lacks of flexible data query interactions. Tasks for comparison and correlation of different faceted querying results are cumbersome in the two systems because at any time a user can only explore along one data query across facets. In addition, Jigsaw requires users switching between different views (e.g., the facets list view and node-link graph view), whereas PivotSlice combines them in a single semantic canvas thus having less divided attention. However, Jigsaw is equipped with rich text analytics functions such as document clustering, relevance ranking, and text summarization, which are not supported by PivotSlice. But some components of CiteVis and Jigsaw can be integrated into PivotSlice to enhance its capabilities. For example, the visual encoding of graphs in CiteVis can be added to one of the visual representations in each table cell; and the text mining functions in Jigsaw can be borrowed to address the large dataset issue discussed above.

7.7 Summary

In this chapter, we have presented a design study of visual exploration techniques assisting the discovery of explicit and implicit relationships in multi-faceted networks. This design study has approached the challenges from the growing complexity of the attribute space of data due to its multiple heterogeneous facets and rich semantics. The proposed visual analysis tool, PivotSlice, not only provides flexible faceted browsing capabilities but also assists the user with revealing various data relationships, by allowing for a systematic and logical management of dynamic queries in a multi-focus and multi-scale tabular view. A preliminary qualitative evaluation with university researchers has shown the usefulness of PivotSlice in exploring complicated citation networks. In summary, the contributions we have made in this chapter include:

- a generic visual query framework for constructing a tabular division of the entire information space of multi-faceted data with customized semantics,
- an information seeking mantra of multi-faceted datasets based on Shneiderman [1996]'s to imply future visualization designs, and
- an interactive and functional tool, PivotSlice, for visually mining data relationships in multidimensional networks, as a demonstration of the visual query framework and an implementation of the information seeking mantra.

It is worth noting that the visual query framework is not restricted to the domain of multi-attributed networks, where all other kinds of multi-faceted datasets can be manipulated and queried in this way to form a logical tabular subdivision, in order to compare multiple subsets of data simultaneously and derive cross-concept relationships. As discussed above (Section 7.6), the framework is also general in the sense of supporting heterogeneous facets, including categorical, ordinal, and numerical attributes. Although developed in the context of multi-faceted data, the proposed information seeking mantra — *Overview first, Pivot and slice, Relate and extract, Details on demand, and History available* — is universal as well in guiding the design of visual exploration techniques of analyzing data relationships in large and complicated information space.

We have presented our last design study in this dissertation. In the upcoming chapters, we will think retrospectively about common characteristics of all the five design studies of multi-focus visualization techniques in various data formats and application domains (Chapter 3–7). We will discuss high-level design considerations, theoretical concepts, and implications based on the practical experience gathered through the five design studies.

Part V

Closing

Chapter 8

REFLECTIONS FROM DESIGN STUDIES

The only source of knowledge is experience.

Albert Einstein

In the preceding three parts (Part II–IV, Chapter 3–7), five design studies have been described to explore different aspects of visual exploration with multi-focus interactive visualization techniques, covering a complete set of fundamental data features (data values, structures, and attributes) in real-world [Bertin, 1977; Ware, 2004]. Those design studies have also touched various kinds of basic data formats (according to Shneiderman [1996]'s classification), situated in many application domains (e.g., economics, computer network analysis, astronomy, computational linguistics, etc.). In this chapter, we take a step back to think in-depth about all the design studies, and further generalize those practical experiences, by providing guidelines, theories, and implications to the future design and study of multi-focus information visualization systems.

This chapter is organized as follows. We first develop four key design considerations for multi-focus interactive visualization that summarize and reflect the common issues derived from the case studies (Section 8.1). Then, we apply these principles retrospectively to those studies by comparing and analyzing their detail approaches to achieve the design goals (Section 8.2). After, by generalizing all the resulting prototype systems at an abstract level, we propose several conceptual models for multi-focus interactive visualization based on some existing models in the previous work (see Section 2.3.2), which offers theoretical foundations for developing multi-focus visual systems (Section 8.3). Finally, inspired from the relevant research conducted in cognitive science about perceptual structures [Garner, 1974], we discuss several insights of the integral and separable characteristics in multi-focus visual exploration tasks to further distill the design studies at the theoretical level (Section 8.4).

8.1 Design Considerations

Based on the practices and experiences of design studies described from Chapter 3 to Chapter 7, we derive the following four high-level principles for developing effective multi-focus interactive visualization techniques.

Diverse Focus Representation. Multi-focus visualization provides the abilities of displaying multiple subregions of a dataset, and user may be interested in data from different perspectives during different stages of exploration. Thus, a visualization system should support various visual representations for each focus region, which can be dynamically adjusted in the exploration process. Moreover, the focus regions that are less interested by a user should be minimized or de-emphasized on the display to provide more visual resources, such as the screen real-estate, for presenting more important information in other foci. The diversity of data focus representations, possibly displayed in multi-scale or multi-level forms, adds several intermediate layers (i.e., different scales) for the visual exploration from various perspectives (i.e., different representations). This is in contrast to the general background context that is always available in the visualization, such as the overview. A user can flexibly manipulate the visual representations for different subsets of data to effectively explore points of interests while maintaining the multi-focus visualization context.

Clear Focus Relationship. In multi-focus visual systems, a user is able to display and manipulate many visualizations, where each represents different data parts with various transformations or encodings. Thus, a clear organization of those visualization objects is a critical issue for enhancing the visual exploration process. The system should support adequate means for users to clearly understand the logical and semantic relationships between various data focus regions in the whole information space as well as to effectively manage them. That is because interpreting those interface relationships is the basis of performing higher-level knowledge discovery tasks. Important properties of focus regions should be indicated from the visualization to allow easy comprehension of the current data navigation status. For example, a user's concerns may include where the data is extracted from the dataset, how the data is transformed, and what the relations are from one data part to others.

Easy Focus Comparison. One of the main benefits of multi-focus interactive visualization is to present various sources of information in a dataset in parallel. This can effectively support the discovery of serendipitous findings, which is achieved by correlating different data subsets from multiple focus regions. However, the inherent and basic side-by-side comparison in multi-focus visualizations may not be adequate for complicated datasets or elaborate exploration tasks. Thus, the visualization system should promote this feature by providing ways for easy visual comparison of data objects. In addition to the abilities for identifying insights in each individual data part, interactions and visual encodings improving the cross-focus data comparison, such as dynamic pattern matching and data differences highlighting, could significantly facilitate the sense-making and analytical reasoning of data.

Smooth Focus Manipulation. Multi-focus interactive visualizations allow a user to explore different data portions simultaneously and view them as a single big picture to make connections and correlations. Each individual focus region reflects one specific exploration path within the entire dataset, which should be dynamically created, modified, and removed with flexible and straightforward interactions. To better assist data navigation along those multiple parallel paths, the system should provide techniques for easily understanding, tracking, and operating the visual and data state changes in the simultaneous explorations within different focus regions, for example, smooth state animations and direct manipulations of visualization objects. More specifically, when the user advances the navigation stage in one data focus, its visual representation should be updated smoothly, reflecting changes of the exploration path situated in the whole visualization context. At the same time, visual features indicating the connections and relations to other focus regions should also be modified accordingly in a similar dynamic manner.

8.2 Comparison and Discussion of Design Studies

The previous section introduces four high-level principles that one should consider during the design of multi-focus interactive visualization systems, based on our experiences obtained from the case studies in this dissertation. In this section, we conduct in-depth analysis of each resulting prototype system to reveal how these four design considerations are implemented in practice and how they affect the visual exploration of data in different real-world application domains and data formats. Table 8.1 presents a brief summary of how we approached the design considerations in each case study. To discuss details, in the following, we refer to the names of the research prototypes developed in the design studies, namely KronoMiner (Chapter 3), ChronoLenses (Chapter 4), DAViewer (Chapter 5), TimeSlice (Chapter 6), and PivotSlice (Chapter 7).

8.2.1 Approaches for Diverse Focus Representation

All the study prototypes were designed with a number of visual representations to display a data subset in the focus, including at least a full-size normal view that shows all the information, and a compact view that shows only important or summarized attributes of the data.

Design Considerations	**High-level Approaches**	**Design Studies (*data feature*, *data format*)**				
		KronoMiner (data values, time-series)	ChronoLenses (data values, time-series)	DAViewer (data structures, collection of trees)	TimeSlice (data attributes, multi-faceted temporal events)	PivotSlice (data attributes, multidimensional networks)
Diverse focus representation	normal visualization	✓ line chart, amplitude plot, bar chart, etc.	✓ line chart, histogram, stem plot, etc.	✓ dendrogram, icicle-dendrogram	✓ histogram, lifespan view, mini-lifespan	✓ node-link diagram, matrix representation
	compact visualization	✓ empty wedge placeholder	✓ empty bar placeholder	✓ horizontal & vertical aggregations	✓ color-coded tiny bars	✓ aggregated nodes
Clear focus relationship	interactive linking	✓ highlighting time segment branches, etc.	✓ highlighting lens ancestors, groups, etc.	✓ highlighting related branches, etc.	✓ highlighting querying paths, etc.	✓ highlighting connected nodes, etc.
	separated view		✓ dynamic tree view	✓ data overview	✓ filtering tree visual	
	spatial placement	✓ flexible hierarchy		✓ tabular organization based on data nature	✓ stacked organization based on queries	✓ tabular organization based on queries
Easy focus comparison	juxtaposition	✓ flexible movement anywhere	✓ detached lens and input focus	✓ rows & columns in the table	✓ flexible movement in the vertical stack	✓ rows & columns in the table
	superposition	✓ MagicAnalytics Lens, semi-transparent overlay	✓ semi-transparent overlay			
	explicit encoding	✓ MagicAnalytics Lens, best-matching arch		✓ similarity heatmap background		✓ cell relation view
	special function	✓ dedicated comparison windows	✓ linking lens to empty panel	✓ text and pattern search		
Smooth focus manipulation	direct manipulation	✓ drag, brush to create, zoom, etc.	✓ drag, brush to create, zoom, etc.	✓ collapse & expand, etc.	✓ drag & drop, etc.	✓ drag & drop, etc.
	contextual menu	✓ pop-up menu	✓ in-place lens toolbar	✓ pop-up menu		✓ pop-up menu
	separated menu/panel		✓ drop-down menu, side panel	✓ drop-down menu		✓ toolbar, drop-down menu, side panel

Table 8.1: A summary of approaches to the design considerations in each case study. Sample implementations in the visualization prototypes are briefly described.

In particular, KronoMiner offers six types of charts to display a time-series segment, each has its own benefits, which serves for a variety of analysis goals (Figure 3.8). For example, the amplitude plot (Figure 3.8-a) is suitable for displaying audio data; the most common line chart (Figure 3.8-b) is used for showing trends; and the histogram (Figure 3.8-d) is good for seeing distributions. The chart type can be dynamically modified through a context menu, and the scale and shift of the y-axis can be adjusted through direct manipulations (Figure 3.3). In addition, a minimized version of the time-series segment is incorporated, which displays the data in a reduced height placeholder, making it possible to focus on a small subset of visually detailed segments without losing the mental map of overall context (Figure 3.4).

Similarly, in ChornoLenses, the time-series within a lens can be displayed in multiple types of charts, such as line chart, scatter plot, stem plot, and histogram, for different browsing purposes. For example, the scatter plot can clearly display data points of the time-series, which is helpful when the sampling rate is not constant; and it is easier to see signs of the values (positive and negative) in the stem plot. Multivariate time-series are also supported by either stacking all the variables or overlaying them (Figure 4.5). Moreover, like KronoMiner, a minimized version of the lens representation is supported, which can be further detached from and moved synchronously or asynchronously with the actual lens panel (Figure 4.4).

The implementations of the compact views in the above two prototypes, though providing the abilities to save the screen real-estate and preserve the context, do not display information about actual data values in the focus. In other words, those compact views are like empty placeholders in a user's mental map. To make those compact visualizations also informative, in the following design studies, we strove to design alternative minimized views that can visually summarize the data in a small physical space.

For example, in DAViewer, two compact tree representations (Figure 5.3-c,d) are designed to reflect the node structural information and the similarity score distributions long the vertical and horizontal dimensions. Those views are in addition to two full-size tree visualizations (Figure 5.3-a,b) based on the dendrogram and the icicle plot [Kruskal and Landwehr, 1983]. This variety of visualizations in representing tree structures promotes the effectiveness of exploring and comparing multiple data regions of interest in the tabular view of DAviewer (Figure 5.4). TimeSlice also supports several visual representations of a temporal event timeline, including a histogram view, a life-span view, and a mini-life span view (Figure 6.4). The entire timeline can also be visualized in minimized form, using colors to encode volumes of events along time (Figure 6.1-f). The PivotSlice system employs a similar idea as DAViewer to display data in its tabular view with multiple levels of scales: rows and columns can be collapsed to show data (graph nodes) in an aggregated manner (Figure 7.1). In addition, for full-size views, a node-link view and a matrix representation of graphs are provided for viewing data in individual table cells for different exploration purposes (Figure 7.8).

8.2.2 Approaches for Clear Focus Relationship

In the design studies, we developed many visual and interactive features for each prototype, in order to assist users with understanding the logical relationships among different data focus regions being explored.

One of the common approaches was to incorporate brushing & linking techniques [Keim, 2002] to connect visualization objects that are far apart in the view. More specifically, a brushing interaction, i.e., directly selecting a subset of the data items, is used in conjunction with a linking effect, i.e., highlighting related data items in different visualizations simultaneously. Brushing & linking can be used in combination with some explicit visual cues to represent the relationship. For example, KronoMiner employs a dynamic time-series hierarchy to maintain the multi-focus exploration history (Section 3.3). The radial tree structure represents how a user drills down the data, allowing users to backtrack along their navigation paths. The parent-child relationship is explicitly visualized as a pair of dashed curves that link one time segment to the corresponding region of interest (ROI) on its parent, and the ROI is shown as a light-gray background to provide visual awareness (Figure 3.1-b). Those explicit visual cues have advantages for the case that time segments can be placed at any locations in the time-series hierarchy.

Another method is to design an interactively coordinated overview to describe the relationships between focus regions. For instance, in ChronoLenses, the relationship between individual lenses is shown in a tree view (Figure 4.1-c), displaying dynamic miniatures of the lens branches alongside labels detailing the operators. When a lens is selected, the tree view shows the analysis pipeline from the current lens upstream to its ancestors, indicating how data flow is processed from the original input. However, a global view of relationships among all the lenses is not included, which is a limitation of ChronoLenses as discussed in Section 4.6. Also, TimeSlice employs the filtering tree component (Figure 6.3) as a separated view to organize a group of temporal event timelines that are formed by performing dynamic queries on faceted attributes of events. Similar to KronoMiner, the filtering tree is built progressively as a user dives into the data that reflects the exploration process.

A third approach is to constrain the spatial locations of all focus regions, thus indicating their relationships in a logical form mapped to a user's mental model. Specifically, both DAViewer and PivotSlice leverage a tabular layout to achieve such goal. The columns of the DAViewer visualization represent different discourse parsers, and the rows are a range of articles, so each cell in this table view displays the resulting discourse tree for a particular parser operating on a particular article. This matrix-like layout eases a user with locating data of interest and mentally managing the focus relationships. Similarly, PivotSlice uses a more dynamic tabular view that is constructed by progressively adding visual queries on the two axes. Thus the visualization indicates a subdivision of the entire information space with customized semantics, allowing for a more flexible sense-making process with multi-focus visual exploration.

8.2.3 Approaches for Easy Focus Comparison

There are mainly three categories of methods for visual comparison of data: juxtaposition, superposition, and explicit encoding [Gleicher et al., 2011]. Side-by-side comparison (i.e., juxtaposition) of various data focus regions is inherently available in multi-focus visualization systems. For example, users can freely move time segments in KronoMiner to compare them. However, in ChronoLenses, the position of a lens defines its region of input, so it cannot be moved without affecting the output visualization. To support a similar kind of side-by-side comparison, a user can detach the display panel of a lens from its focus bar (that defines the input range), thus moving it freely along the timeline (Figure 4.4-d).

Another common practice existing in our design studies is to leverage a logical organization of all the focus regions, enabling this side-by-side data comparison in semantics. One example is the tabular layout of foci in DAViewer and PivotSlice (Figure 5-b and Figure 7.1-d); since the columns and rows hold logical meanings, a user can conduct straightforward comparisons. Particularly, in DAViewer, comparing trees in a row reveals nuances of various parsers with the same document, and comparing trees in a column indicates how the same parser performs over different documents; in PivotSlice, exploring data in the same row or column reflects the comparison of items sharing common data attributes that are defined by the faceted queries on the axes. Another example is the filtering tree visualization in TimeSlice, which semantically organizes similar data queries by grouping the same constraints into a tree node (Figure 6.3). This empowers the inherent side-by-side comparison of the corresponding temporal events since similar timelines are placed closer in the view (Figure 6.1-c).

In addition to juxtaposition, overlaying two focus views (i.e., superposition) is another way to facilitate data comparison. For example, time segments in KronoMiner is semi-transparent and can be freely arranged on top of each other, allowing users to compare time-series in a see-through manner (Figure 3.4). Similarly, the transparency value of lenses in ChronoLenses can be adjusted, and with the detaching function users can also do the comparison with superposition.

A third approach is to compute data relations, e.g., based on a similarity measurement, and explicitly visualize them in the view. For instance, the MagicAnalytics Lens (Figure 3.6) and the feature of finding the best-matched data (Figure 3.7) are two analytical functions that significantly facilitate the comparison of time-series segments in KronoMiner. Both of the methods displays on-the-fly computations while a user manipulates the visual elements. In DAViewer, the background of the tree visualization is color-coded according to the similarity scores of the content and structural differences with respect to a referencing tree (Figure 5.3), so that a user can easily spot which branches of a tree is more error prone. Likewise, PivotSlice integrates a cell relation view to depict how each sub-graph visualized in the tabular view relates to each other in terms of crossing links (Figure 7.1-e).

Some other methods are also employed in the designs studies to help users compare data. For example, the Kronominer interface contains several detail comparison windows where a user can temporally store data of interest and compare them in a more traditional way (Figure 3.1-d). The ChronoLenses system allows users to create empty chart panels and add resulting time-series of a lens to compare (see Section 4.4.2). In DAViewer, when a user selects a tree branch in a cell in the tabular view, corresponding parts in all other trees in the same row are also highlighted to facilitate the comparison of sub-tree structures (Figure 5.1-b). In addition, the text and pattern searching abilities in DAViewer further promote the finding of similarities in data (Figure 5.4-f). Last, the variety of charts and visual representations of data introduced previously in all prototypes can assist users with data comparisons in different perspectives.

8.2.4 Approaches for Smooth Focus Manipulation

In all the design studies, direct manipulation is one of the central techniques to empower users to explore data with multi-focus visualizations in an easy and flexible manner. Each study prototype was designed to support fluid and straightforward interactions on the visualization objects, sometimes coupled with smooth animations, allowing for an effective management of the multi-focus navigation paths as well as easy learning and understanding of the interface. The following lists some example interaction techniques that we used.

Fluid interactions are one of the strengths for the KronoMiner system, where almost all major exploration functions can be achieved by directly manipulating the visual objects. Users can move a time segment or a ring of segments to any locations of the time-series hierarchy in the view. Squishing and stretching the time scale add another layer of flexibility in visual exploration of time-series data. To allow users easily remember and understand those interactions, we designed KronoMiner with a novel context-based interaction language

(Figure 3.5). The ChronoLenses interface also integrates direct manipulation techniques with the lenses, such as moving and scaling a lens, modifying lens parameters, and changing lens chart types (Figure 4.4). Additionally, any updates on one lens propagate the effects along the analytical pipeline. Moreover, DAViewer embeds some direct tree operations (e.g., expanding and collapsing tree branches), straightforward interactions that coordinate various views (e.g., selecting trees in the overview to show in the detailed view), and layout manipulation of display texts associated with discourse trees (see Section 5.5.2). Finally in TimeSlice and PivtoSlice, visual queries can be constructed and modified by drag-and-drop operations, and smooth animations are played when visual states are changed, which assists a user with understanding the visualization (see Section 6.4.1 and Section 7.3.2).

In addition to direct manipulation, some other common approaches for configuring objects in user interfaces, e.g., contextual menus and control panels, are integrated to some of prototype systems. For example, in KronoMiner users can trigger a menu to precisely set up parameters by hitting the space bar (Figure 3.1-g). There are side panels available in ChronoLenses (Figure 4.1-e) and PivotSlice (Figure 7.1-c) that allow users to obtain with some information of the currently selected visual elements and configure some of values.

8.2.5 A Summary Note for Design in Practice

The above describes different methods implemented in each design study to approach the four general design considerations. Table 8.1 outlines the major high-level strategies to fulfill those design considerations in practice and lists examples in the case studies. This can be used as a broad guideline for people to design effective multi-focus visualization given specific data types and tasks. But the design studies in this dissertation have only explored limited numbers of data forms and application domains. Further investigations are needed to generate a more complete design space about how to approach the design considerations under different situations. Nevertheless, even though the actual application domains are sometimes different, as long as the data types and users' tasks are similar to our design studies, one can leverage the example methods listed in the table as starting points.

Additionally, we note that the case studies fulfilled the four design considerations to different extents. Some implemented more features in one design consideration, but supported less in another. Thus we suggest that visualization researchers and practitioners allocate different amount of effort to each design consideration in practice, based on the specific cases encountered in real-world. The fulfillments of those design considerations may depend on the domain, the data types, and the user's tasks. For example, to develop a general visual analysis tool for time-series data (as in KronoMiner and ChronoLenses), one may pay more attention to diverse focus representation, since different domains have their own commonly-used charts for time-series, and also easy focus comparison, since most of insights are derived from data comparison in time-series analysis. For multi-faceted datasets, the approach to diverse representation depends on the type of data elements containing the faceted attributes, such as temporal events in TimeSlice and networks in PivotSlice. Further, in these two case studies, the major tasks are sense-making of data relationships in attribute space, so diverse focus representation is at a less important place than clear focus relationship. That is because the information space is complicated where users often get lost. If major user tasks in a design study are related to data comparison, a rich set of methods should be implemented to address easy focus comparison. For instance, DAViewer incorporated a complex algorithm to compute tree branch similarities, in addition to the side-by-side comparison that is naturally supported in multi-focus visualization. In cases where the visualization design is complicated, smooth focus manipulation, especially direct manipulation techniques, should be seriously considered to ease a user's understanding and learning of the tool.

Thinking from the opposite direction, we can also use the above two insights to guide the selection of visualization tools given specific problems and tasks. For example, if one happens to have the datasets and problems that are similar to those in one design study, he can assess the effectiveness of different systems based on how they implement the list of example features shown in Table 8.1. At a higher level, according to a user's exploration goals, the richness of a visualization system's functionalities towards each design consideration can also reflect the usefulness of the tool. For instance, if the task is to visually compare multiple elements, the default support of side-by-side comparison in multi-focus visualization may not be adequate, therefore other features, such as explicit encodings of the data difference, should be incorporated.

8.3 Models of Multi-Focus Interactive Visualization

In this section, we take a step further to generalize the experiences learned from the design studies. Based on the previous work described in Section 2.3.2, we propose two conceptual models of multi-focus interactive visualization, which provides a fundamental understanding of the structure and process of visual data exploration with multi-focus visualization systems.

8.3.1 Multi-Focus Information Visualization Pipeline

As shown in Figure 8.1, we introduce a multi-focus information visualization pipeline by extending the original data state reference model [Chi and Riedl, 1998] and its variations [Card

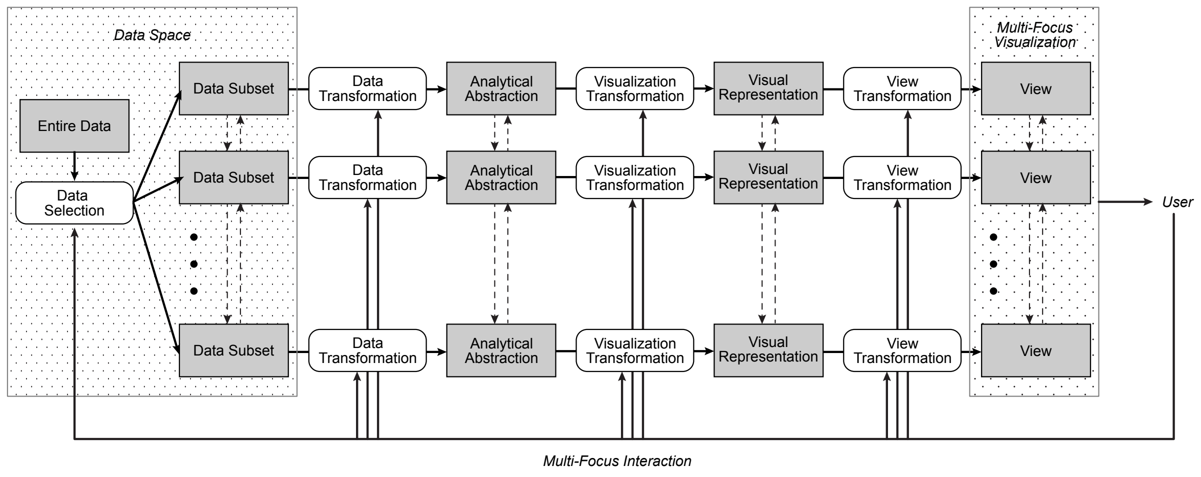


Figure 8.1: Multi-focus information visualization pipeline.

et al., 1999; Carpendale, 1999; Collins, 2010]. Similar to the old model, the pipeline starts at the raw data, diverges into multiple data subsets, and then goes through several stages including analytical abstractions, visual representations, and graphical views that are the end-product of the pipeline to be presented to users. Four types of operations, including data selection, data transformation, visualization transformation, and view transformation, are performed to process the information between two consecutive stages.

The largest distinctions in this multi-focus version include: 1) the whole pipeline is composed of multiple component pipelines operating on particular subsets of data simultaneously, and 2) there is a data selection step to extract multiple data subsets from the entire dataset. All the component pipelines are systematically treated as a whole and communicate with each other, rather than just separately processing their own data. At any of the stages, information from each component pipeline can be coordinated to maximize the capabilities of knowledge discovery (as shown with the dash-line arrows in Figure 8.1). Our design studies have provided several evidence of this concept. For example, at the first raw data stage, the MagicAnalytics Lens of KronoMiner (Figure 3.6) and each individual lens of ChronoLenses can combine the raw time-series segments from multiple focus regions to derive correlations. At the analytical abstraction stage, DAViewer computes similarity scores between the referencing tree and all other trees in the same row of a tabular view (Figure 5.4-b). For visual representations, the visual and interactive coordination of different foci, such as dynamic brushing and linking techniques described in previous sections, is widely applied in all five prototypes. Finally, at the last stage, various views, as the outputs of all component pipelines, form a multi-focus visualization of the whole dataset perceived by users, allowing easy correlations and manipulations of the views to discover insights.

Moreover, with multi-focus interactions, users have the capabilities to 1) create, adjust, and remove data subsets that initiate the component pipelines, and 2) modify the three functional operators (data, visualization, and view transformations) in each component pipeline, as shown with the upward arrows in Figure 8.1. Thus users can manipulate the multi-focus visualization according to their specific navigation goals. The dynamic management of multiple foci is interactively enabled in all the design study prototypes. Examples of the view transformations include the squeezing and rotating of segments in KronoMiner (Figure 3.3), scaling of lenses in ChronoLenses (Figure 4.4), and minimizing and maximizing of visualizations in all study prototypes. Additionally, these systems enable various kinds of visualization transformations in the pipeline, interactively managed by the user, through different visual encoding methods and filtering mechanisms for each focus. As for data transformations, an example is that ChronoLenses support numerous transformation techniques of time-series data for different analytical purposes, such as cross-correlation, differentiation, and point-by-point subtraction (see Section 4.3).

Although this multi-focus visualization pipeline is mainly developed based on the notation proposed by Card et al. [1999], it can be further extended by adding a visual presentation stage as Carpendale [1999] did, or having one more user interaction access point to the initial raw data which is similar to Collins [2010]'s model.

8.3.2 Knowledge Discovery with Multi-Focus Interactive Visualization

By extending the basic visualization model proposed by van Wijk [2005], which is later applied in representing the visual analytics process [Keim et al., 2008], we propose a new model for interpreting knowledge discovery in multi-focus visual exploration with interactive visualizations (Figure 8.2).

Following similar information flow of the original model, initially, different portions of data are analyzed with proper techniques and displayed to a user with adequate visual representations and interactions in the multi-focus form. The process then enters to a loop of the user gradually gaining knowledge from the visualizations and driving the system towards discovering more insights. The multi-focus views of the dataset are simultaneously perceived by the user to build knowledge piece by piece both in individual focus and the visualization, for example, through comparisons and correlations of different data subsets. Once enough insights are identified, the user can make further hypotheses, and move forward to explore and analyze the data in different focus regions by commanding the system with corresponding specifications, or configure (i.e., create, delete, and modify) the focus regions through interactions with the data subsets. This multi-focus interaction updates the visual

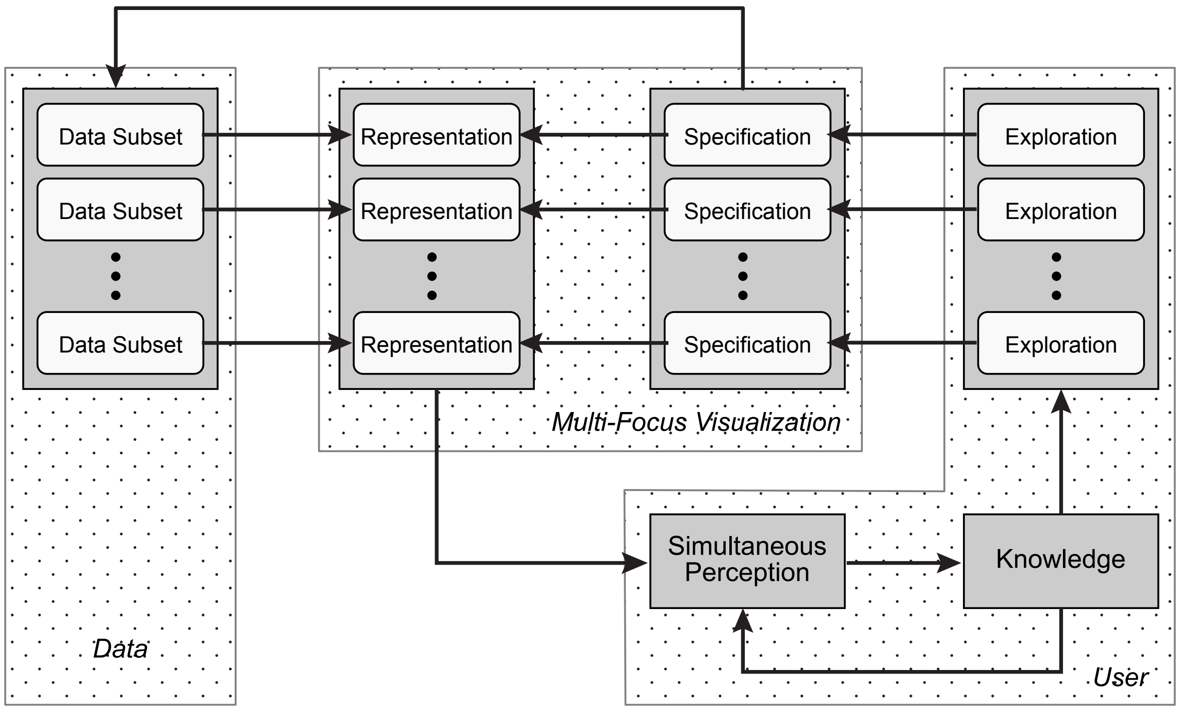


Figure 8.2: A model of knowledge discovery process with multi-focus interactive visualization.

representations of data dynamically, allowing the discovery of new knowledge by continuing the loop. The key difference in this model is that multiple subsets of a dataset are processed, rendered, and manipulated in a concurrent manner as shown in the data and multi-focus visualization components (Figure 8.2); and in the user component, one interprets the visual representations as a whole to make mutually dependent decisions for further directing the exploration in each focus.

This knowledge discovery process model is a superset of the previous information visualization pipeline (Figure 8.1). The process of transforming data subsets into multi-focus representations in this model is in coordination with the information visualization pipeline, but omitting the internal stages — analytical abstraction and visual representation, and the operations — data selection, data transformation, visualization transformation, and view transformation. Moreover, the specifications and management of multiple foci made by users according to their mental exploration goals are interpreted as the multi-focus interactions in the above pipeline. Additionally, this model provides a detailed demonstration of how useful knowledge is formed inside a user's mind by gaining insights from the visualizations progressively and how the overall knowledge could further drive the explorations in the focus regions (see the user component in Figure 8.2).

8.3.3 Design Implications and Discussion

The above multi-focus visualization pipeline and model can help designers better understand the nature of multi-focus visual exploration supported by interactive visualizations. There are several design implications that can be derived from the pipeline and model.

First, each focus view in a multi-focus visualization is not absolutely independent, as indicated with the "dashed lines" in Figure 8.1. As stated in earlier in the introduction of the definition of multi-focus exploration (see Section 2.4), the multiple foci should be "logically related". This also aligns with the previous design consideration "clear focus relationship". A visualization with several completely independent focus windows may impose difficulty for users to discover knowledge. Hence, when design a multi-focus visualization, one should consider facilitating the correlation of foci by allowing cross-focus communications at difficult stages of the visual system.

Second, users perceive a multi-focus visualization as a whole but can react differently with individual focus views. The multi-focus visualization pipeline indicates such fact with multi-focus user interactions, and the model further shows a user's internal processing of knowledge. This is related to distribution cognition theory mentioned earlier in the dissertation (see Section 1.2), which suggests that knowledge lies across individuals, artifacts, and the environment [Hutchins, 1995]. By going one step further, Figure 8.1 and Figure 8.2 indicate that the cognitive process lies across multiple foci within the same visualization system.

Third, it is critical to facilitate the dynamics of creating, modifying, and removing focus regions within the entire information space. One large distinction of extending the basic models to those multi-focus versions is to add an interactive mean of manipulating the foci, for example, the data selection in Figure 8.1 and the direct arrow from the user to the data in Figure 8.2. This is because the multi-focus visual exploration is an iterative process with dynamic focus windows rather than a set of static views restricted with predefined input. Thus, flexible multi-focus interaction is a key factor to consider when designing such visualization systems in practice.

8.4 Integrality and Separability of Visual Data Exploration

Information visualization is a powerful tool to turn abstract data into easily perceivable visual objects by meaningfully encoding various data attributes into different visual features, such as the visual variables proposed by Bertin [1983]. When people perceive a multidimensional encoding of data attributes with the visual features, they can be integral (for example, x and y positions of a glyph) or separable (for example, shape and color of a visual object), as introduced in the theory of processing of perceptual structure [Garner, 1974]. With integral displays, many attributes of an object are viewed holistically, and with separable representations, people tend to have independent judgment about each dimension [Ware, 2004], which critically affects the accuracy of identifying data characteristics and the efficiency of data exploration. However, there are tons of ways to map a multidimensional data point to a multidimensional visual representation. So how would the theory of perceptual structure imply on visualization design? Should we encode certain data attributes with an integral perceptual structure to enable coordination and correlation, or make them distinct and separable to minimize misleading? We can raise many questions like those.

Thus, in the following part of this section, we discuss the integrality and separability issues in visual data exploration, by extending and generalizing the perceptual structure processing theory (see Section 2.3.3) in the information visualization context. We conduct in-depth analyses of the problems by making connections with our design studies, providing critical implications for multi-focus interactive visualization system design.

8.4.1 Levels of Integrality and Separability

The purpose of information visualization is to amplify a user's cognition by representing abstract data visually, also with interactions. Based on the theory of processing of perceptual structure, we can facilitate the insight discovery process by properly presenting data attributes to certain visual forms with integral or separable perceptual structures. Thus, we here propose the idea of integrality and separability levels in information visualization design, including *basic encoding level*, *data presentation level*, and *visual exploration level*, ordered from low to high levels of data abstraction. In coordination with the Gestalt principles [King and Wertheimer, 2005; Ware, 2004], we discuss an extension of the original concept of integral and separable perceptual structures, in which whether the visualization techniques are designed for showing data attributes with easy connections or distinctions is interested.

The basic encoding level is concerned with visual mappings of low-level data items, such as multivariate nodes in a graph, to primitive visual objects, such as circles and glyphs. In this level, we try to define an optimum way of visually conveying the data by carefully choosing which attribute should be associated to which visual variable. Ware [2004] listed a number of display dimension pairs in order of integral to separable (e.g., Figure 2.4). Some practice has been found in our design studies. For example, in representing time-series in KronoMiner and ChronoLenses, we enhance the integrality by encoding the time dimension and the dependent data value with x and y positions. On the other hand, in DAViewer, we use color and size, respectively in the dendrogram version icicle plot (Figure 5.3), to make the distinctions between node categories and the structural information of the node. Moreover, it is important to note that we can dual-encode a data attribute with separable visual variables to emphasize that attribute, such as the use of both length and color to represent a person's lifespan in TimeSlice (Figure 6.4).

An the data presentation level, we focus on the organization of primitive visual objects with semantics, in order to present a series of data items with a meaningful visualization for finding insights. In other words, we need to wisely choose the layout of the basic visualization objects to reveal important relationships and trends among data items. For example, PivotSlice allows the alignment of data items along different facets through the flexible pivoting operation on both axes, and thus those attributes can be perceived integrally to enable the detection of correlations (Figure 7.2). In contrast, KronoMiner suggests that different components of multivariate time-series can be shown in a stacked manner in one focus segment or placed onto multiple segments (Figure 3.9), which indicates the integral or separable arrangements of basic data representations for different exploration purposes. In a similar way, ChronoLenses support both a combined view and a stacked view of time-series data on the lens panels (Figure 4.5).

Finally, the highest level — exploration level — deals with the coordination of views or visual representations of different data pieces to support sense-making and knowledge discovery. This level is closely associated with the nature of multi-focus visualization about how to organize multiple views systemically to enable effective data exploration. For emphasizing the integrality, KronoMiner supports the overlapping of two focus regions to display a combined and derived visualization for presenting two sources of information as a whole (Figure 3.6); Chronolenses extend this idea of overlapping lenses by further adding the lens grouping function (Figure 4.7-d). Moreover, in DAViewer and PivotSlice, a tabular organization of the focus regions is employed to enable an integral perception of each view (i.e., through the column and row positions) within the context and relationships of the overall visualization (Figure 5.4-b and Figure 7.1-d). In contrast, all research prototypes provide the abilities of minimizing any focus regions to display data in compact views, enhancing the perception of separability between the views that are currently interested by the user and those that are not.

8.4.2 Dynamic Integrality and Separability

As a user's goals are continuously changing in data exploration, it is difficult to have a perfect visual mapping of data throughout the whole process. The preceding discussion also indicates a trade-off between integrality and separability for visual representations of data attributes. Thus, we introduce the concept of dynamic integrality and separability of information visualization through user interactions. More specifically, the visualization system should be able to adapt different analytical purposes in the exploration process with dynamic modifications of data visual mappings. This interactive feature of integrality and separability in visual data exploration appears in all the three levels mentioned above.

At the lowest basic encoding level, for example, KronoMiner provides different types of visual encodings for time-series, including line chart, bubble chart, colored strips and some other type of charts, for different encoding requirements of integrality and separability of data attributes (Figure 3.8). Similarly, in DAViewer, each node of a discourse tree can be represented as a small circle or a rectangle according to specific exploration goals, enabling the combination and separation of tree depths and tree-level ranges of internal nodes in a dendrogram (Figure 5.3). As for the data presentation level, PivotSlice supports dynamic organization and division of all data items within a semantic tabular view through interactive visual queries (Figure 7.2). Further in each cell of the tabular view, data points can be displayed with a force-directed layout, a matrix view, or alignments with different faceted attributes, serving for various analytical purposes (Figure 7.3). With those interactions, the integrality and separability of data presentation layouts can be flexibly changed according to a user's needs. Finally, in the exploration level, which reflects the main benefits of interactions in multi-focus visualization, all prototypes in the design studies provide some forms of selectively collapsing the views. Also, columns and rows of the tabular views in DAViewer and PivotSlice can be rearranged in order to associate or separate the visualizations of certain views in the context of customized semantics (Figure 5.4-b and Figure 7.1-d). In some other systems, such as KronoMiner and ChronoLenses, the user has much of the freedom for repositioning focus regions to interactively manipulate integrality and separability of the multi-focus visualizations based on their spatial proximities.

Therefore, interaction, which sets information visualization away from traditional static data graphics, is governed by a user's exploration goals to dynamically and perceptually associate data attributes that indicate useful knowledge (i.e., supporting integrality) as well as divide ones that convey less meanings (i.e., supporting separability). This further articulates the means of how information visualization can amplify human cognition of data, i.e., through the flexible dynamic perceptual integrality & separability trade-off of representing data attributes, based on the widely accepted definition of information visualization by Card et al. [1999].

8.4.3 Design Implications and Discussion

Although the above discussion about extending the perceptual structure processing theory in the context of visual exploration is preliminary, we can derive a number of useful implications for designing multi-focus visualization techniques in practice.

First, designers should be aware of the levels of integrality and separability existed in multi-focus visual exploration, and develop suitable visualizations to maximize its abilities for communicating and discovering insights. Several principles have been developed at the first level — visual encodings — to understand the perceptual structure processing. But it is more complicated in multi-focus visual exploration, in which the two higher levels — data representation and visual exploration — play important roles in the process. It is difficult to derive a set of comprehensive guidelines at the higher levels, but the general rules of thumb are to encourage the integral perception for visual objects that convey more meaningful insights, otherwise separate them to avoid confusion. Designers should consider those points when making design choices in studies. However, in practice, it is usually unknown in advance that which parts of visualization can express more integral knowledge. Thus working with a group of domain experts may provide some basic guidelines.

Second, interaction is the key to accommodate different exploration goals and user needs by manipulating the extents of integrality and separability on each level. All the design studies and the derived theory have indicated the importance of user interaction in multi-focus visual exploration. The above discussion has provided some examples of using interaction to achieve the dynamic nature of integrality and separability. A general design goal is to integrate flexible and intuitive interaction techniques with multi-focus visualizations, allowing modifications of visual representations of data at the three aforementioned perception levels. However, one questions is to what extent of the freedom of interaction the system should provide. If it is too free-formed, the intended integrality and separability setup by design would be impaired; otherwise, the ability to discover serendipitous knowledge would be restricted. The design studies in this dissertation have demonstrated different levels of freedom in user interactions (e.g., KronoMiner is more free-formed and DAViewer is more restricted). Those issues should be seriously considered by designers when developing multi-focus interactive visualizations.

8.5 Summary

This chapter has summarized all the five design studies described from Chapter 3 through Chapter 7. We have conducted in-depth thinking about the practices of developing highly interactive multi-focus visualization systems to solve real-world problems existing in the exploration of various data formats from a range of application domains. This chapter has presented many aspects of our critical thinking that generalizes the experiences in all design studies, including design guidelines, theories, and implications, which create a broader impact on the future design of multi-focus visualizations. Specifically, the contributions have been made in this chapter include:

- a set of high-level design considerations for multi-focus interactive visual systems —- *diverse focus representation, clear focus relationship, easy focus comparison, and smooth focus manipulation*,
- a multi-focus information visualization pipeline reflecting conceptual stages of processing data inside visual systems, based on the original versions [Card et al., 1999; Carpendale, 1999; Chi and Riedl, 1998; Collins, 2010],
- a model of knowledge discovery process in multi-focus visual interfaces by extending Keim et al. [2008]'s notation, and
- an in-depth discussion about the phenomena of perceptual integrality and separability existing in multi-focus visual exploration, proposing the concepts of integral-separable levels and dynamics.

For each of the above theoretical foundations we have derived, this chapter has described concrete examples in our design studies reflecting the high-level abstractions and summarizations. For example, Section 8.2 has listed how each design study approached the four design considerations in the development of the research prototype, and has also summarized common methods used in the practices, which can inspire the future design. Similarly, in introducing other thoughts and theories, we have referred to the detailed design choices made in the studies to illustrate how they can be applied in practice.

Nevertheless, as theoretical concepts, some of the discussions are undoubtedly simplistic, especially for the notion of integrality and separability in multi-focus visualization. Eventually, we expect that there will emerge a more complete body of theory to account for different kinds of information and applications; and this is the same for the multi-focus visualization pipeline and the knowledge discovery model. Although the five design studies in this dissertation have covered a relatively wide space of information people want to explore, there still need a large number of experiments to discover deeper insights of these theories and to address many of the challenges arising from design practices. However, the abstract summarizations described in this chapter have suggested a rich set of useful guidelines to imply and inspire the future design of interactive visualizations in order to support effective multi-focus visual exploration of data.

Chapter 9

CONCLUSIONS

Computer science is no more about computers than astronomy is about telescopes.

Edsger Dijkstra

This dissertation has investigated issues and techniques for supporting effective visual exploration of data with multi-focus interactive visualizations. Our work has been motivated by the needs for efficiently discovering deeper knowledge and making informed decisions from the abundant information of the world that is quickly growing in both scale and complexity. The preceding three parts (Part II–IV, consisting of Chapter 3–7) have described five case studies of designing, developing, and evaluating various multi-focus interactive visualization techniques. Each part has focused on the visual exploration driven by one particular aspect of the information, including data values, data structures, and data attributes, which covers a complete set of fundamental data features to study in real-world [Bertin, 1977; Ware, 2004]. These design studies were also situated in a wide range of practical application domains, solving real-life problems emerged from users exploring various forms of datasets (e.g., time-series, trees, networks, and multi-faceted data). Moreover, in the previous chapter (Chapter 8), we have conducted critical thinking about the five design studies to summarize the experiences we have learned at a theoretical level, developing vital perspectives about designing effective multi-focus visualization techniques.

In this final chapter, we first summarize the contributions we have made towards addressing the objectives and research questions posed at the beginning of this dissertation (Section 9.1). Next we discuss a number of limitations of the design studies (Section 9.2). We then describe several promising future research directions, based on new challenges and questions arose from the previous chapters (Section 9.3). Finally, we conclude this chapter and the dissertation with some high-level closing remarks (Section 9.4).

9.1 Summary of Contributions

The primary goal of this dissertation is to substantially investigate the issues and techniques of multi-focus visual exploration in helping users make decisions, discover knowledge, and communicate insights within information spaces that are growing at a tremendous rate. Through conducting five design studies on developing multi-focus visual systems for exploring different types of data in a range of application domains (Table 1.1), we have approached to address the following three high-level research questions that were introduced at the beginning:

Q1. How would we apply the concept of multi-focus visual exploration in visualizations of different practical datasets?

Q2. How can we design multi-focus interactive visual systems to enable effective data exploration in real-world applications?

Q3. What are implications of multi-focus visualization and interaction techniques for facilitating visual exploratory tasks?

In the rest of this section, we summarize the contributions of this dissertation led by those questions, which have deepened our understanding about the effects of multi-focus visualization techniques on exploring real-world data.

9.1.1 Novel Interactive Visualization Systems

To address Q1, through five design studies, we have contributed new multi-focus visualization techniques to provide effective visual exploration of various kinds of data.

KronoMiner presents a generic visualization tool for time-series exploration based on a dynamic radial hierarchy with fluid direct manipulations (Chapter 3);

ChronoLenses introduce a flexible visualization technique for exploratory analysis of time-series through progressive building of advanced analytical pipelines (Chapter 4),

DAViewer provides an interactive visual system that facilitates researchers in Natural Language Processing with the exploration and comparison of parsing results in discourse analysis (Chapter 5),

TimeSlice contributes a visual query technique for browsing multi-faceted temporal events based on a dynamic hierarchical organization of data filers (Chapter 6), and

PivotSlice addresses the discovery of various relationships in multi-faceted datasets through dynamic visual queries that subdivide the information space with customized semantics (Chapter 7).

9.1.2 General Visualization Techniques and Frameworks

During the development and implementation of the five design studies, we have contributed the following technical innovations to better answer Q1 in a general sense.

MagicAnalytics Lens: an interactive visualization technique that dynamically displays in-place data relationship plots on the overlap of two visual elements (Section 3.4.3),

ChronoLens Framework: a lens-based visualization concept that supports both single and dual data inputs as well as multi-step transformations by linking lenses in a hierarchy, enabling elaborate time-series analysis tasks with customized pipelines (Section 4.3),

Icicle-Dendrogram: a new visual representation of trees that combines the benefits of the dendrogram and the icicle plot (Section 5.4.1),

Compact Tree Visualizations: two aggregated visual representations of trees that shows their structural features from the vertical and horizontal perspectives (Section 5.4.2),

Dynamic Filtering Tree: a generic visual query structure that organizes filters logically in a hierarchy, supporting interactive faceted exploration of multi-attributed temporal events (Section 6.3), and

Multi-Faceted Data Query Framework: a visual query language for exploring relationships in datasets with heterogeneous faceted attributes with a dynamic matrix-like subdivision of the entire information space in customized semantics (Section 7.2).

9.1.3 Domain-Specific Design Requirements and Guidelines

In order to understand how to design effective multi-focus visualization in real-world applications (Q2), we have derived a number of design requirements and guidelines from the practices of our design studies. Although those summarizations were distilled from specific application domains, they could help us understand the critical aspects that users demand in performing certain types of tasks, such as time-series analysis, tree comparison, and faceted browsing, which are more generalizable.

- design guidelines for multi-purpose time-series visual exploration tools (Section 3.2.1),
- a design space of multivariate time-series visualization layout (Section 3.2.2),

- tasks and design requirements for visual exploratory analysis of time-series data (Section 4.2),
- design requirements for supporting the workflow of comparing and developing discourse parsers by computational linguistics researchers (Section 5.3.2),
- design guidelines for effective browsing and querying of datasets with multiple faceted attributes (Section 6.2), and
- an information seeking mantra for the visual exploration of multi-faceted datasets (Section 7.3.2).

9.1.4 High-Level Implications and Concepts

To answer Q3, we have conducted critical thinking and reflection about all the design studies, which generalizes our experiences with several high-level theoretical implications and concepts. Also, those insights further address the previous two questions (Q1 and Q2) in a more general and abstract approach.

- four high-level design considerations for devising efficient multi-focus interactive visualizations (Section 8.1),
- a multi-focus information visualization pipeline reflecting conceptual stages inside visual systems (Section 8.3.1),
- a knowledge discovery model for multi-focus interactive visualizations revealing the relationships among the dataset, the visual representation, and the user (Section 8.3.2), and
- a critical thinking about multi-focus visual exploration inspired by the theory of processing of perceptual structure [Garner, 1974] (Section 8.4).

9.2 Limitations

This dissertation has conducted comprehensive investigations on interaction and visualization techniques to support multi-focus visual exploration of various forms of data. However, there still exist several limitations with the design studies.

First, the validations of visual systems developed in the case studies are mostly preliminary evaluations. Hence, it is unknown that if those visualization systems could provide longer-term benefits to users in the specific application domains. In referring to [Munzner, 2009]'s nested visualization design and evaluation model introduced in Section 2.3.4 (Figure 2.5), design studies in this dissertation have not gone to the step of "observe adoption rates" in the domain situation level. Therefore, development studies of the visualizations with longer time and more realistic working environment are needed to further validate the design.

Second, most of the visualizations created in the design studies fall in the category of early prototypes, which may have usability problems on the interfaces even though general design guidelines were followed. Some of those issues have been discovered during the evaluation phase (e.g., in the TimeSlice study described in Section 6.5). This further places obstacles for the long-term evaluation discussed above. More importantly, it may affect users' satisfactions and opinions to the actual visualization design that we care about, even in a preliminary evaluation. Although not all of the problems can be full addressed before the development of visual systems, basic interface usability issues could be minimized by running low-cost heuristic evaluations for several times [Nielsen, 1994]. We thus recommend that visualization researchers should pay more attention to this aspect when conducting design studies.

Third, almost all the visualization techniques presented in the previous chapters largely rely on pure user-guided visual exploration to discover knowledge and insights in data, although some basic analytical capabilities were incorporated (e.g., links connecting pre-detected motifs between time-series segments in KronoMiner as shown in Figure 3.4-c). Advanced analysis in the community of machine learning, artificial intelligence, and data mining has not been explored in the design studies. Although the main focus of this dissertation is visual data exploration techniques that should heavily involve human in the loop, the proposed systems might be limited when the scale of data becomes huge. We argue that fully automatic methods are only reliable and effective for well-understood problems, but it does not necessarily mean that analysis results of machines are not useful in the context of visual data exploration authored by users. Those results can be potential suggestions for helping users find insights faster, and effective approaches for preprocessing or filtering large and noisy data to make the following exploration tasks conducted in visualization easier.

9.3 Future Directions

This dissertation has made considerable progress on answering the three high-level questions about multi-focus interactive visualization, through the contributions summarized in the previous section. However, there are sill many new challenges, ideas, and problems arose in the design studies, which are interesting to pursue as future work. In the following, we outline three promising research directions to further substantiate our understanding of issues and techniques on multi-focus visual exploration.

9.3.1 Towards More Intelligent Visual Systems

Many of the visualization prototypes in this dissertation were designed to give as much freedom as possible to users for data exploration. For example, KronoMiner allows people to manipulate multi-scale data segments of time-series in a freeform, such as moving time segments anywhere in the dynamic hierarchy, and stretching & squishing the time scale. Also, ChronoLenses support building unconstrained analytical pipelines with a series of different basic transformations on time-series. Those kinds of freestyle data exploration provide much flexibility that accommodates different visual analysis tasks and data with various characteristics, which is essential for discovering opportunistic and cross-concept insights.

However, these visual interfaces merely accept commands and specifications from users, and rely on people's judgment of the visualization outputs to proceed the data exploration. Thus, the relationship between the user and the system is one-directional, where the system passively receives the user's operations. It can be enhanced by further enabling a bi-directional communication between the user and the system by making the visualization smarter with proper suggestions, such as parts of data to dive in and specific parameters to use. A simple example would be that enabling a kind of "snapping" behavior to indicate strong correlations when a user is moving a MagicAnalytics Lens in KronoMiner or a component lens in ChronoLenses. The ultimate goal along this direction is to develop intelligent visualization systems that can flexibly adjust and recommend several settings based on the dynamics of the current analysis objective, the user's experience, and the dataset. Proper system suggestions can be learned from historical interactions of many users. Since users have various levels of expertise and experiences about analyzing different types of data, a novice user can benefit from the approaches generalized from experienced users by the system when encountering similar analytical problems. Multi-focus interaction and visualization techniques can promote the above intelligent feature as the user-system communication is "multi-channeled", allowing for a coherent and systemic knowledge base to be developed through the visual exploration of data. But, at the same time, it demands for more complicated approaches to address those bigger challenges of enabling effective multi-channeled bi-directional communications between users and systems.

9.3.2 Towards More Integrated Analysis and Visualization

Some of the design studies presented in the previous chapters have touched problems requiring a combination of visualization and data analytics to discover insights. For example, KronoMiner provides the MagicAnalytics Lens technique and the feature of finding the best-matched time-series; and ChronoLenses offers a set of basic data transformations for constructing advanced analytics pipelines. Those techniques demonstrate the effectiveness of integrating analytical functionalities with interactive visualization. However, those data analytics methods are still very fundamental, mainly some basic numerical transformations of time-series. There exists a huge range of analytical models that are difficult and abstract for people to comprehend, such as some advanced machine learning techniques used for analyzing complex real-world problems. Effectively integrating those models and visualizations in a seamless mode remains challenging.

The design study of DAViewer has made some effort in approaching this question to attempt to facilitate discourse analysis in computational linguistics. The discourse parser concerned in DAViewer is a complicated analysis process that first breaks a document into text chunks and then hierarchically clusters them into a rhetorical tree with a bottom-up manner. The DAViwer visualization displays the parsing results and the clustering processes that are in tree structures. It allows effective comparison among those outputs between different algorithms, which helps computational linguistics researchers better understand the mechanisms of the discourse parsers. However, DAViewer merely visualizes the ending results and roughly links the discourse trees with the original inputs (i.e., the documents), thus omitting many aspects involved in the discourse analysis process, such as parameters and specifications of the algorithms. Without such information, it can be difficult for a user to interpret the analyzed results or trust them because automatic methods are not always perfect.

Therefore, there remain big challenges and unsolved problems for tightly integrating advanced data analysis and interactive visualizations. On one hand, it is beneficial for end-users to perform efficient data exploration promoted by various kinds of data analytics methods in a fluid integration. On the other hand, it would be interesting to leverage visual exploration techniques for debugging analytical models, tuning algorithm parameters, and explaining phenomena of results. In both contexts, multi-focus visualization techniques are critical to be incorporated for supporting advanced tasks, such as comparing different analytical methods and evaluating different phases of an algorithm, which is promising to pursue as future work.

9.3.3 Towards More Effective Data Query and Exploration

Data querying plays a vital role in the identification of insights and rules, converting raw information into useful knowledge. Thus, the design studies on all of the three data features (i.e., data values, data structures, and data attributes) concerned in this dissertation support some mechanisms of dynamic visual data queries to different extents. For example, KronoMiner's ability of finding the best matched parts in two time-series is a query of similar patterns existing in data; DAViewer supports dynamic query of customized template trees with both structures and node attributes specified by a user; and one of the main contributions in TimeSlice and PivotSlice is about visual query frameworks of faceted data attributes. Those examples further indicate that queries can be performed on a range of data characteristics in visual exploration.

Although those design studies have investigated a number of aspects on effective data queries, there still exist many challenges, especially when datasets become larger and more complicated. In this "big data" era, the scale of data makes it difficult to be consumed by the system or the user without effective query methods. All study prototypes developed in this dissertation have the limitations to visualize extremely large datasets, which indicates promising future research directions on developing novel data management techniques as the middleware between the visualization and the raw database, as well as efficient visual representations for summarizing large amounts of data. Moreover, as queries become more complicated when richer semantics are contained in data, the creation, management, and understanding of those queries is another big problem to users. The design studies, especially TimeSlice and PivotSlice, have shown evidences that multi-focus interactive visualizations are beneficial for those tasks. However, they focus on performing queries on single data features, e.g., only data attributes. Real-world data analysis may require data query along several perspectives, for example, placing constraints on both temporal dimension and meta-data attributes in the TimeSlice design study. Therefore, it would be interesting to explore how multi-focus visualization can facilitate visual data exploration driven by dynamic queries on heterogeneous information aspects.

9.4 Closing Remarks

As the quantity and complexity of digital information continue to grow, so do new challenges and demands for developing effective computational and perceptual aids to help people explore, comprehend, and analyze the data. Terms such as *Information Overload* and *Big Data* are becoming more common in our daily vocabulary. As their effects are turning into as an integral part of our culture, research on how to interpret and represent the digital information is undoubtedly critical. This dissertation research is situated in the intersection of three broad areas — Information Visualization, Human-Computer Interaction, and Visual Analytics, which indicates that the bond of those fields will be even closer in the future to yield more intuitive, flexible, and intelligent visual systems in an interdisciplinary approach. We have investigated the possibility that multi-focus interactive visualization techniques, grounded by design studies in various practical domains about the exploration of different data characteristics, may significantly facilitate the discovery, correlation, and communication of insights. However, our work is only a start of the emerging research of devising, evaluating, and generalizing new visual interfaces to assist people with making sense of large and diverse data. The word “big” in front of “data” describes not only its capacity but also its potential, which is full of knowledge and imagination as mankind is exploring the universe.

BIBLIOGRAPHY

Aigner, W., Miksch, S., Müller, W., Schumann, H., and Tominski, C. (2007). Visualizing time-oriented data-a systematic view. *Computer Graphics*, 31(3):401–409. *On page 18.*

Aigner, W., Miksch, S., Müller, W., Schumann, H., and Tominski, C. (2008). Visual methods for analyzing time-oriented data. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):47–60. *On pages 18, 45, 54, 65.*

Aigner, W., Miksch, S., Thurnher, B., and Biffl, S. (2005). Planninglines: novel glyphs for representing temporal uncertainties and their evaluation. In *Proceedings of the 9th International Conference on Information Visualisation*, pages 457–463. *On page 19.*

Albrecht, J., Hwa, R., and Marai, G. E. (2009). The Chinese Room: Visualization and Interaction to Understand and Correct Ambiguous Machine Translation. *Computer Graphics Forum*, 28(3):1047–1054. *On page 94.*

Amar, R., Eagan, J., and Stasko, J. (2005). Low-level components of analytic activity in information visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 111–117. *On pages 146, 147.*

André, P., Wilson, M. L., Russell, A., Smith, D. A., Owens, A., and schraefel, m. (2007). Continuum: Designing timelines for hierarchies, relationships and scale. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, pages 101–110. *On pages 25, 118.*

Andrienko, N. and Andrienko, G. (2006). *Exploratory Analysis of Spatial and Temporal Data*. Springer. *On page 66.*

Appert, C., Chapuis, O., and Pietriga, E. (2010). High-precision magnification lenses. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 273–282. *On pages 76, 84.*

Appert, C. and Fekete, J.-D. (2006). Orthozoom scroller: 1d multi-scale navigation. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 21–30. *On page 49.*

Becker, R., Eick, S., and Wilks, A. (1995). Visualizing network data. *IEEE Transactions on Visualization and Computer Graphics*, 1(1):16–28. *On page 23.*

Bederson, B. (1996). Pad++: A zoomable graphical sketchpad for exploring alternate interface physics. *Journal of Visual Languages & Computing*, 7(1):3–32. *On page 26.*

Bertin, J. (1977). *Graphics and Graphic Information Processing*. de Gruyter Press, Berlin. *On pages 8, 163, 183.*

Bertin, J. (1983). *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press. *On pages 23, 29, 30, 176.*

Bertini, E., Hertzog, P., and Lalanne, D. (2007). Spiralview: Towards security policies assessment through visual correlation of network resources with evolution of alarms. In *Proceedings of IEEE Symposium on Visual Analytics Science and Technology*, pages 139–146. *On pages 19, 43.*

Bertini, E. and Santucci, G. (2006). Visual quality metrics. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*, BELIV '06, pages 1–5. *On page 32.*

Bezerianos, A., Chevalier, F., Dragicevic, P., Elmqvist, N., and Fekete, J. (2010). Graphdice: A system for exploring multivariate social networks. *Computer Graphics Forum*, 29(3):863–872. *On pages 25, 140.*

Bier, E. A., Stone, M. C., Pier, K., Buxton, W., and DeRose, T. D. (1993). Toolglass and magic lenses: the see-through interface. In *Proceedings of the conference on computer graphics and interactive techniques*, SIGGRAPH '93, pages 73–80. *On pages 25, 55, 65, 68, 71.*

Borg, I. and Groenen, P. (1997). *Modern Multidimensional Scaling: Theory and Applications*. Springer. *On page 157.*

Brandes, U. and Nick, B. (2011). Asymmetric relations in longitudinal social networks. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2283–2290. *On page 23.*

Bremm, S., von Landesberger, T., and Heb, M. (2011). Interactive visual comparison of multiple trees. In *Proceedings of IEEE Conference on Visual Analytics Science and Technology*, pages 29–38. *On pages 22, 100.*

Buono, P., Aris, A., Plaisant, C., Khella, A., and Shneiderman, B. (2005). Interactive pattern search in time series. In *Proceedings of Visualization and Data Analysis*, pages 175–186. *On pages 20, 46, 64.*

Buxton, W. and Myers, B. (1986). A study in two-handed input. In *Proceedings of the ACM conference on Human factors in computing systems*, pages 321–326. *On pages 6, 36.*

Card, S. K., Mackinlay, J. D., and Shneiderman, B., editors (1999). *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann. *On pages xv, 6, 7, 8, 17, 28, 172, 174, 179, 181.*

Card, S. K., Moran, T. P., and Newell, A. (1983). *Psychology of Human-Computer Interaction*. Lawrence Erlbaum Associates. *On page 7.*

Carlson, L., Marcu, D., and Okurowski, M. E. (2001). Building a discourse-tagged corpus in the framework of Rhetorical Structure Theory. In *Proc. of the Second SIGdial Workshop on Discourse and Dialogue*, pages 1–10. *On pages 92, 109.*

Carpendale, M. S. T. (2003). Considering visual variables as a basis for information visualisation. Technical Report 2001-693-16, Department of Computer Science, University of Calgary. *On page 30.*

Carpendale, M. S. T. and Montagnese, C. (2001). A framework for unifying presentation space. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, UIST '01, pages 61–70. *On pages 25, 26.*

Carpendale, S. (1999). *A Framework for Elastic Presentation Space*. PhD thesis, Simon Fraser University. *On pages 28, 173, 174, 181.*

Carpendale, S. (2008). Evaluating information visualizations. In Kerren, A., Stasko, J., Fekete, J.-D., and North, C., editors, *Information Visualization*, volume 4950 of *Lecture Notes in Computer Science*, pages 19–45. Springer Berlin / Heidelberg. *On page 32.*

Carpendale, S., Ligh, J., and Pattison, E. (2004). Achieving higher magnification in context. In *Proceedings of the 17th annual ACM symposium on User interface software and technology*, UIST '04, pages 71–80. *On pages 25, 26.*

Carrire, J. and Kazman, R. (1995). Interacting with huge hierarchies: Beyond cone trees. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 74–81. *On page 21.*

Chai, J. Y. and Jin, R. (2004). Discourse structure for context question answering. In *Proceedings of HLT-NAACL Workshop on Pragmatics in Question Answering*, pages 23–30. *On page 89.*

Chau, D. H., Kittur, A., Hong, J. I., and Faloutsos, C. (2011). Apolo: making sense of large network data by combining rich user interaction and machine learning. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 167–176. *On page 23.*

Chi, E.-H., Barry, P., Riedl, J., and Konstan, J. (1997). A spreadsheet approach to information visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 17–24. *On page 21.*

Chi, E. H.-h. and Riedl, J. (1998). An operator interaction framework for visualization systems. In *Proceedings of the IEEE Symposium on Information Visualization*, INFOVIS '98, pages 63–70. *On pages 28, 172, 181.*

Claessen, J. and van Wijk, J. (2011). Flexible linked axes for multivariate data visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2310–2316. *On page 20.*

Cleveland, W. C. and McGill, M. E. (1988). *Dynamic Graphics for Statistics*. CRC Press, Inc., 1st edition. *On page 17.*

Cockburn, A., Karlson, A., and Bederson, B. B. (2009). A review of overview+detail, zooming, and focus+context interfaces. *ACM Computer Survey*, 41:2:1–2:31. *On pages 24, 65.*

Collins, C. (2010). *Interactive Visualizations of Natural Language*. PhD thesis, University of Toronto. *On pages 28, 173, 174, 181.*

Collins, C., Carpendale, S., and Penn, G. (2007). Visualization of uncertainty in lattices to support decision-making. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization*, EUROVIS'07, pages 51–58. *On page 94.*

Collins, C., Carpendale, S., and Penn, G. (2009a). Docuburst: Visualizing document content using language structure. *Computer Graphics Forum*, 28(3):1039–1046. *On page 8.*

Collins, C., Penn, G., and Carpendale, S. (2009b). Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1009–1016. *On page 94.*

Cook, K., Earnshaw, R., and Stasko, J. (2007). Guest editors' introduction: Discovering the unexpected. *IEEE Computer Graphics and Applications*, 27(5):15–19. *On page 7.*

Culy, C., Lyding, V., and Dittmann, H. (2011). Structured parallel coordinates: a visualization for analyzing structured language data. In *Proceedings of the International Conference on Corpus Linguistics*, pages 485–493. *On page 94.*

DeNeefe, S., Knight, K., and Chan, H. H. (2005). Interactively exploring a machine translation model. In *Proceedings of the Annual Meeting of the Assocation for Computational Linguistics, Poster Session*. *On page 94.*

Díaz, J., Petit, J., and Serna, M. (2002). A survey of graph layout problems. *ACM Computer Survey*, 34(3):313–356. *On page 23.*

Dork, M., Riche, N., Ramos, G., and Dumais, S. (2012). Pivotpaths: Strolling through faceted information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2709–2718. *On pages 23, 135, 141, 145.*

Draper, G., Livnat, Y., and Riesenfeld, R. (2009). A survey of radial methods for information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(5):759–776. *On page 47.*

Duarte, F., Sikansi, F., Fatore, F., Fadel, S., and Paulovich, F. (2014). Nmap: A novel neighborhood preservation space-filling algorithm. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2063–2071. *On page 22.*

Dunne, C., Henry Riche, N., Lee, B., Metoyer, R., and Robertson, G. (2012). Graphtrail: analyzing large multivariate, heterogeneous networks while supporting exploration history. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 1663–1672. *On pages 141, 146, 158.*

duVerle, D. A. and Prendinger, H. (2009). A novel discourse parser based on support vector machine classification. In *Proc. of the Joint Conf. of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing*, ACL '09, pages 665–673. *On page 92.*

Eades, P. A. (1984). A heuristic for graph drawing. In *Congressus Numerantium*, volume 42, pages 149–160. *On page 23.*

Elmqvist, N., Dragicevic, P., and Fekete, J. (2010a). Color lens: Adaptive color scale optimization for visual exploration. *IEEE Transactions on Visualization and Computer Graphics*, 17(6):795–807. *On page 25.*

Elmqvist, N., Riche, Y., Riche, N. H., and Fekete, J.-D. (2010b). Melange: Space folding for visual exploration. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):468–483. *On pages 4, 36, 76.*

Fails, J., Karlson, A., Shahamat, L., and Shneiderman, B. (2006). A visual interface for multivariate temporal data: Finding patterns of events across multiple histories. In *Proceedings of IEEE Symposium On Visual Analytics Science And Technology*, pages 167–174. *On pages 19, 43.*

Fekete, J.-D., van Wijk, J. J., Stasko, J. T., and North, C. (2008). The value of information visualization. In Kerren, A., Stasko, J., Fekete, J.-D., and North, C., editors, *Information Visualization: Human-Centered Issues and Perspectives*, volume 4950 of *Lecture Notes in Computer Science*, chapter 1, pages 1–18. Springer Berlin Heidelberg. *On page 4.*

Fekete, J.-D., Wang, D., Dang, N., Aris, A., and Plaisant, C. (2003). Overlaying graph links on treemaps. In *Posters compendium of InfoVis*, pages 82–83. *On page 158.*

Feng, V. W. and Hirst, G. (2012). Text-level discourse parsing with rich linguistic features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, pages 60–68. *On page 92.*

Fishkin, K. and Stone, M. C. (1995). Enhanced dynamic queries via movable filters. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 415–420. *On pages 71, 74.*

Freire, M., Plaisant, C., Shneiderman, B., and Golbeck, J. (2010). Manynets: an interface for multiple network analysis and visualization. In *Proceedings of the 28th international conference on Human factors in computing systems*, CHI '10, pages 213–222. *On pages 23, 24.*

Friendly, M. (2008). A brief history of data visualization. In Chen, Chun-houh abd Härdle, W. K. and Unwin, A., editors, *Handbook of Data Visualization*, pages 15–56. Springer. *On page 17.*

Fruchterman, T. M. J. and Reingold, E. M. (1991). Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21:1129–1164. *On pages 23, 140, 143.*

Furnas, G. W. (1986). Generalized fisheye views. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '86, pages 16–23. *On pages 26, 65, 131.*

Furnas, G. W. and Bederson, B. B. (1995). Space-scale diagrams: understanding multiscale interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI '95, pages 234–241. *On page 25.*

Furnas, G. W. and Buja, A. (1994). Prosection views: Dimensional inference through sections and projections. *Journal of Computational and Graphical Statistics*, 3:323–385. *On page 20.*

Garner, W. R. (1974). *The Processing of Information and Structure*. Lawrence Erlbaum Associates. *On pages 15, 30, 163, 177, 186.*

Gleicher, M., Albers, D., Walker, R., Jusufi, I., Hansen, C. D., and Roberts, J. C. (2011). Visual comparison for information visualization. *Information Visualization*, 10(4):289–309. *On page 169.*

Gorg, C., Liu, Z., Kihm, J., Choo, J., Park, H., and Stasko, J. (2013). Combining computational analyses and interactive visualization for document exploration and sensemaking in jigsaw. *IEEE Transactions on Visualization and Computer Graphics*, 19(10):1646–1663. *On pages 21, 158.*

Gotz, D. and Stavropoulos, H. (2014). Decisionflow: Visual analytics for high-dimensional temporal event sequence data. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1783–1792. *On page 19.*

Graham, M. and Kennedy, J. (2010). A survey of multiple tree visualisation. *Information Visualization*, 9(4):235–252. *On page 22.*

Gratzl, S., Gehlenborg, N., Lex, A., Pfister, H., and Streit, M. (2014). Domino: Extracting, comparing, and manipulating subsets across multiple tabular datasets. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2023–2032. *On page 21.*

Green-Armytage, P. (2010). A colour alphabet and the limits of colour coding. *Colour: Design and Creativity*, 5(10):1–23. *On page 97.*

Gutwin, C. (2002). Improving focus targeting in interactive fisheye views. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves*, CHI '02, pages 267–274. *On page 27.*

Hadany, R. and Harel, D. (1999). A multi-scale algorithm for drawing graphs nicely. In *Proceedings of the 25th International Workshop on Graph-Theoretic Concepts in Computer Science*, WG '99, pages 262–277. *On page 23.*

Hadlak, S., Schulz, H.-J., and Schumann, H. (2011). In situ exploration of large dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2334–2343. *On pages 23, 25, 27, 139, 145, 158.*

Harel, D. and Koren, Y. (2000). A fast multi-scale method for drawing large graphs. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '00, pages 282–285. *On page 23.*

Hartigan, J. A. (1975). Printer graphics for clustering. *Journal of Statistical Computation and Simulation*, 4:187–213. *On page 20.*

Havre, S., Hetzler, E., Whitney, P., and Nowell, L. (2002). Themeriver: Visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8:9–20. *On pages 8, 19.*

Hawryszkiewycz, I. T. (1984). *Database Analysis and Design*. Science Research Associates. *On page 137.*

Heer, J. and Bostock, M. (2010). Crowdsourcing graphical perception: Using mechanical turk to assess visualization design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 203–212. *On page 33.*

Heer, J. and Boyd, D. (2005). Vizster: visualizing online social networks. In *IEEE Symposium on Information Visualization*, pages 32–39. *On page 28.*

Heer, J., Kong, N., and Agrawala, M. (2009). Sizing the horizon: The effects of chart size and layering on the graphical perception of time series visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 1303–1312. *On pages 19, 57.*

Heer, J., Mackinlay, J., Stolte, C., and Agrawala, M. (2008). Graphical histories for visualization: Supporting analysis, communication, and evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 14:1189–1196. *On pages 45, 144.*

Hegarty, M. (2004). Diagrams in the mind and in the world: Relations between internal and external visualizations. In Blackwell, A., Marriott, K., and Shimojima, A., editors, *Diagrammatic Representation and Inference*, volume 2980 of *Lecture Notes in Computer Science*, pages 1–13. Springer Berlin Heidelberg. *On page 7.*

Henry, N. and Fekete, J.-D. (2006). Matrixexplorer: A dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):677–684. *On page 23.*

Henry, N., Fekete, J.-D., and McGuffin, M. J. (2007). Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309. *On page 23.*

Hernault, H., Prendinger, H., duVerle, D. A., and Ishizuka, M. (2010). Hilda: A discourse parser using support vector machine classification. *Dialogue and Discourse*, 1(3):1–33. *On pages 92, 97, 109.*

Hlawatsch, M., Burch, M., and Weiskopf, D. (2014). Visual adjacency lists for dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, 20(11):1590–1603. *On page 24.*

Holten, D. (2006). Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *Transactions on Visualization and Computer Graphics*, 12(5):741–748. *On pages 23, 54.*

Holten, D. and van Wijk, J. J. (2008). Visual comparison of hierarchically organized data. *Computer Graphics Forum*, 27(3):759–766. *On page 22.*

Holten, D. and van Wijk, J. J. (2009). Force-directed edge bundling for graph visualization. *Computer Graphics Forum*, 28(3):983–990. *On pages 23, 158.*

Huffaker, B., Nemeth, E., and claffy, k. (1999). Otter: A general-purpose network visualization tool. In *Proceedings of the International Networking Conference*. *On page 23.*

Hutchins, E. (1995). *Cognition in the Wild*. MIT Press. *On pages 8, 176.*

Im, J.-F., McGuffin, M., and Leung, R. (2013). Gplom: The generalized plot matrix for visualizing multidimensional multivariate data. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2606=–2614. *On page 20.*

Inselberg, A. and Dimsdale, B. (1990). Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *Proceedings of the 1st conference on Visualization*, VIS '90, pages 361–378. *On page 20.*

Jacob, R. J. K., Sibert, L. E., McFarlane, D. C., and Mullen, Jr., M. P. (1994). Integrality and separability of input devices. *ACM Transactions Computer-Human Interaction*, 1(1):3–26. *On page 31.*

Javed, W. and Elmqvist, N. (2010). Stack zooming for multi-focus interaction in time-series data visualization. In *Proceedings of the IEEE Pacific Visualization Symposium*, pages 33–40. *On pages 19, 25, 36, 43, 45, 65, 76, 79, 124, 131.*

Javed, W., Ghani, S., and Elmqvist, N. (2012). Polyzoom: multiscale and multifocus exploration in 2d visual spaces. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 287–296. *On pages 25, 36, 43, 76.*

Javed, W., McDonnel, B., and Elmqvist, N. (2010). Graphical perception of multiple time series. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):927–934. *On pages 19, 58.*

Johnson, B. and Shneiderman, B. (1991). Tree-maps: a space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2nd conference on Visualization*, VIS '91, pages 284–291. *On page 22.*

Kabbash, P., Buxton, W., and Sellen, A. (1994). Two-handed input in a compound task. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '94, pages 417–423, New York, NY, USA. ACM. *On pages 6, 36.*

Kairam, S., MacLean, D., Savva, M., and Heer, J. (2012). Graphprism: compact visualization of network structure. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 498–505. *On page 24.*

Kandogan, E. (2000). Star coordinates: A multi-dimensional visualization technique with uniform treatment of dimensions. In *Proceedings of the IEEE Information Visualization Symposium*, pages 9–12. *On page 20.*

Kang, H., Plaisant, C., Lee, B., and Bederson, B. (2006). NetLens: Iterative exploration of content-actor network data. In *Proceedings of the IEEE Symposium on Visual Analytics Science And Technology*, pages 91–98. *On page 146.*

Keim, D., Andrienko, G., Fekete, J.-D., Grg, C., Kohlhammer, J., and Melanon, G. (2008). Visual analytics: Definition, process, and challenges. In Kerren, A., Stasko, J., Fekete, J.-D., and North, C., editors, *Information Visualization: Human-Centered Issues and Perspectives*, volume 4950 of *Lecture Notes in Computer Science*, chapter 7, pages 154–175. Springer Berlin Heidelberg. *On pages xv, 7, 27, 28, 29, 65, 66, 67, 85, 140, 143, 158, 174, 181.*

Keim, D. A. (2002). Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8. *On pages 4, 8, 24, 65, 100, 168.*

Kincaid, R. (2006). Line graph explorer: scalable display of line graphs using focus+context. In *Working Conference on Advanced Visual interfaces*, pages 404–411. *On pages 36, 65.*

Kincaid, R. (2010). Signallens: Focus+context applied to electronic time series. *IEEE Transactions on Visualization and Computer Graphics*, 16:900–907. *On pages 19, 27, 65, 76, 84.*

King, D. B. and Wertheimer, M. (2005). *Max Wertheimer and Gestalt Theory*. Transaction. *On pages 31, 177.*

Kirsh, D. and Maglio, P. P. (1994). On distinguishing epistemic from pragmatic action. *Cognitive Science*, 18(4):513–549. *On page 7.*

Krstajic, M., Bertini, E., and Keim, D. (2011). Cloudlines: Compact display of event episodes in multiple time-series. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2432–2439. *On page 19.*

Kruskal, J. B. and Landwehr, J. M. (1983). Icicle plots: Better displays for hierarchical clustering. *The American Statistician*, 37(2):162–168. *On pages 14, 22, 95, 97, 98, 167.*

Lam, H., Bertini, E., Isenberg, P., Plaisant, C., and Carpendale, S. (2012). Empirical studies in information visualization: Seven scenarios. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1520–1536. *On page 32.*

Lamping, J. and Rao, R. (1996). The hyperbolic browser: A Focus+Context technique for visualizing large hierarchies. *Journal of Visual Languages & Computing*, 7(1):33–55. *On page 21.*

Lee, B., Czerwinski, M., Robertson, G., and Bederson, B. B. (2005). Understanding research trends in conferences using PaperLens. In *CHI '05 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '05, pages 1969–1972. *On pages 21, 119, 146.*

Lee, B., Robertson, G. G., Czerwinski, M., and Parr, C. S. (2007). Candidtree: visualizing structural uncertainty in similar hierarchies. *Information Visualization*, 6(3):233–246. *On page 22.*

Lee, B., Smith, G., Robertson, G. G., Czerwinski, M., and Tan, D. S. (2009). Facetlens: exposing trends and relationships to support sensemaking within faceted datasets. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1293–1302. *On pages 21, 119, 135, 141, 146.*

Leganchuk, A., Zhai, S., and Buxton, W. (1998). Manual and cognitive benefits of two-handed input: an experimental study. *ACM Transactions on Computer-Human Interaction*, 5:326–359. *On page 31.*

Lewis, J. and Rieman, J. (1993). *Task-Centered User Interface Design: A Practical Introduction*. Department of Computer Science, University of Colorado, Boulder. *On page 33.*

Liu, Z., Navathe, S. B., and Stasko, J. T. (2011). Network-based visual analysis of tabular data. In *Proceedings of the IEEE Symposium on Visual Analytics Science and Technology*, pages 39–48. *On page 21.*

Lopez-Hernandez, R., Guilmaine, D., McGuffin, M., and Barford, L. (2010). A layer-oriented interface for visualizing time-series data from oscilloscopes. In *Proceedings of IEEE Pacific Visualization Symposium*, pages 41–48. *On page 19.*

Mack, R. L. and Nielsen, J. (1995). *Human-computer interaction*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. *On page 33.*

Mackinlay, J. D., Robertson, G. G., and Card, S. K. (1991). The perspective wall: detail and context smoothly integrated. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, CHI '91, pages 173–176. *On page 26.*

Madsen, H. (2007). *Time Series Analysis*. Chapman & Hall/CRC. *On page 83.*

Mann, W. C. and Thompson, S. A. (1988). Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281. *On pages 90, 92.*

Marcu, D. (1999). Discourse trees are good indicators of importance in text. In *Proceedings of th Advances in Automatic Text Summarization*, pages 123–136. *On pages 12, 89.*

McLachlan, P., Munzner, T., Koutsofios, E., and North, S. (2008). Liverac: Interactive visual exploration of system management time-series data. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1483–1492. *On pages 19, 26, 42, 45, 65.*

McNee, S. M. and Arnette, B. (2008). Productivity as a metric for visual analytics: Reflections on e-discovery. In *Proceedings of the 2008 Workshop on BEyond Time and Errors: Novel evaLuation Methods for Information Visualization*, BELIV '08, pages 1:1–1:6. *On page 32.*

Meyer, M. D., Munzner, T., and Pfister, H. (2009). Mizbee: A multiscale synteny browser. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):897–904. *On pages 46, 54.*

Monroe, M., Lan, R., Lee, H., Plaisant, C., and Shneiderman, B. (2013a). Temporal event sequence simplification. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2227–2236. *On page 20.*

Monroe, M., Lan, R., Morales del Olmo, J., Shneiderman, B., Plaisant, C., and Millstein, J. (2013b). The challenges of specifying intervals and absences in temporal queries: A graphical language approach. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 2349–2358. *On page 20.*

Morse, E., Lewis, M., and Olsen, K. A. (2000). Evaluating visualizations: Using a taxonomic guide. *International Journal of Human-Computer Studies*, 53(5):637–662. *On page 33.*

Müller, W. and Schumann, H. (2003). Visualization methods for time-dependant data-an overview. In *Proceedings of the 2003 Winter Simulation Conference*, pages 737–745. *On page 18.*

Munzner, T. (2009). A nested model for visualization design and validation. *Visualization and Computer Graphics, IEEE Transactions on*, 15(6):921–928. *On pages xv, 32, 34, 35, 186.*

Munzner, T. (2014). *Visualization Analysis and Design*. CRC Press. *On page 30.*

Munzner, T., Guimbretiere, F., and Robertson, G. (1999). Constellation: A visualization tool for linguistic queries from mindnet. In *Proceedings of the IEEE Symposium on Information Visualization*, INFOVIS '99, pages 132–135. *On page 94.*

Munzner, T., Guimbretière, F., Tasiran, S., Zhang, L., and Zhou, Y. (2003). Treejuxtaposer: scalable tree comparison using focus+context with guaranteed visibility. *ACM Transactions on Graphics*, 22:453–462. *On page 22.*

Myers, D. G. (2011). *Psychology*. Worth Publishers. *On page 6*.

Nguyen, Q. V. and Huang, M. L. (2002). A space-optimized tree visualization. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 85–92. *On page 21*.

Nielsen, J. (1994). *Usability Inspection Methods*. John Wiley & Sons. *On page 187*.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine*, 2(11):559–572. *On page 21*.

Perlin, K. and Fox, D. (1993). Pad: an alternative approach to the computer interface. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, SIGGRAPH '93, pages 57–64. *On page 26*.

Phan, D., Paepcke, A., and Winograd, T. (2007). Progressive multiples for communication-minded visualization. In *Proceedings of Graphics Interface 2007*, GI '07, pages 225–232. *On pages 19, 118, 119*.

Pietriga, E. and Appert, C. (2008). Sigma lenses: focus-context transitions combining space, time and translucence. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI '08, pages 1343–1352. *On page 27*.

Pietriga, E., Bau, O., and Appert, C. (2009). Representation-independent in-place magnification with sigma lenses. *IEEE Transactions on Visualization and Computer Graphics*, 16(03):455–467. *On pages 65, 76, 84*.

Pilz, T., Luther, W., and Ammon, U. (2008). Retrieval of spelling variants in nonstandard texts – automated support and visualization. *SKY Journal of Linguistics*, 21:155–200. *On page 93*.

Plaisant, C. (2004). The challenge of information visualization evaluation. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '04, pages 109–116. *On page 32*.

Plaisant, C., Carr, D., and Shneiderman, B. (1995). Image-browser taxonomy and guidelines for designers. *IEEE Software*, 12(2):21–32. *On page 4*.

Plaisant, C., Milash, B., Rose, A., Widoff, S., and Shneiderman, B. (1996). Lifelines: visualizing personal histories. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 221–227. *On pages 19, 42, 118, 119*.

Prendinger, H., Piwek, P., and Ishizuka, M. (2007). Automatic generation of multi-modal dialogue from text based on discourse structure analysis. In *Proceedings of International Conference on Semantic Computing*, pages 27–36. *On page 89.*

Rao, R. and Card, S. K. (1994). The table lens: merging graphical and symbolic representations in an interactive focus + context visualization for tabular information. In *Proceedings of the SIGCHI conference on Human factors in computing systems: celebrating interdependence*, CHI '94, pages 318–322. *On pages 36, 99.*

Robertson, G., Cameron, K., Czerwinski, M., and Robbins, D. (2002). Animated visualization of multiple intersecting hierarchies. *Information Visualization*, 1(1):50–65. *On page 22.*

Robertson, G. G. and Mackinlay, J. D. (1993). The document lens. In *Proceedings of the 6th annual ACM symposium on User interface software and technology*, UIST '93, pages 101–108. *On page 26.*

Robertson, G. G., Mackinlay, J. D., and Card, S. K. (1991). Cone trees: animated 3d visualizations of hierarchical information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, CHI'91, pages 189–194. *On page 22.*

Russell, D. M., Stefik, M. J., Pirolli, P., and Card, S. K. (1993). The cost structure of sensemaking. In *Proceedings of the INTERACT '93 and CHI '93 Conference on Human Factors in Computing Systems*, CHI '93, pages 269–276. *On pages 7, 29, 140.*

Sacha, D., Stoffel, A., Stoffel, F., Kwon, B. C., Ellis, G., and Keim, D. (2014). Knowledge generation model for visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1604–1613. *On page 29.*

Saraiya, P., North, C., and Duca, K. (2005). An insight-based methodology for evaluating bioinformatics visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 11(4):443–456. *On page 32.*

Sarkar, M. and Brown, M. H. (1994). Graphical fisheye views. *ACM Communication*, 37:73–83. *On page 26.*

Sedlmair, M., Meyer, M., and Munzner, T. (2012). Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440. *On page 9.*

Shneiderman, B. (1983). Direct manipulation: A step beyond programming languages. *Computer*, 16:57–69. *On pages 45, 140.*

Shneiderman, B. (1996). The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings of IEEE Symposium on Visual Languages*, pages 336–343. *On pages xv, 8, 18, 27, 36, 37, 45, 65, 140, 143, 144, 159, 163.*

Shneiderman, B. and Aris, A. (2006). Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740. *On pages 23, 143.*

Shneiderman, B. and Plaisant, C. (2006). Strategies for evaluating information visualization tools: Multi-dimensional in-depth long-term case studies. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*, BELIV '06, pages 1–7. *On page 34.*

Shoemaker, G. and Gutwin, C. (2007). Supporting multi-point interaction in visual workspaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pages 999–1008. *On pages 4, 36.*

Spence, R. (2006). *Information Visualization: Design for Interaction.* Pearson, 2 edition. *On page 8.*

Spence, R. and Apperley, M. (1982). Data base navigation: An office environment for the professional. *Behaviour and Information Technology*, 1(1):43–54. *On page 26.*

Spenke, M. and Beilken, C. (2000). Infozoom - analysing formula one racing results with an interactive data mining and visualisation tool. In *Proceedings of Data Mining*, pages 455–464. *On pages 21, 119.*

Spenke, M., Beilken, C., and Berlage, T. (1996). Focus: the interactive table for product comparison and selection. In *Proceedings of the 9th annual ACM symposium on User interface software and technology*, UIST '96, pages 41–50. *On pages 21, 119.*

Stasko, J. (2014). Value-driven evaluation of visualizations. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, BELIV '14, pages 46–53, New York, NY, USA. ACM. *On page 34.*

Stasko, J., Choo, J., Han, Y., Hu, M., Pileggi, H., Sadana, R., and Stolper, C. D. (2013). Citevis: Exploring conference paper citation data visually. In *IEEE conference on information visualization (poster)*. *On pages 24, 158.*

Stasko, J. and Zhang, E. (2000). Focus+context display and navigation techniques for enhancing radial, space-filling hierarchy visualizations. In *IEEE Symposium on Information Visualization*, pages 57–65. *On page 22.*

Stolte, C., Tang, D., and Hanrahan, P. (2002). Polaris: a system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):52–65. *On pages 21, 135, 139.*

Stolte, C., Tang, D., and Hanrahan, P. (2003). Multiscale visualization using data cubes. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):176–187. *On page 21.*

Therón, R., Fontanillo, L. F., Marcos, A. E., and Herrero, C. S. (2011). Visual analytics: A novel approach in corpus linguistics and the Nuevo Diccionario Histórico del Español. In *Proceedings of III Congreso Internacional de Lingüística de Corpus*. *On page 93.*

Thomas, J. J. and Cook, K. A., editors (2005). *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Press. *On page 7.*

Tominski, C., Abello, J., and Schumann, H. (2004). Axes-based visualizations with radial layouts. In *Proceedings of the 2004 ACM Symposium on Applied Computing*, SAC '04, pages 1242–1247. *On page 19.*

Tory, M. and Moller, T. (2004). Human factors in visualization research. *IEEE Transactions on Visualization and Computer Graphics*, 10(1):72–84. *On page 33.*

Tu, Y. and Shen, H.-W. (2007). Visualizing changes of hierarchical data using treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 13:1286–1293. *On page 22.*

Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Graphics Press. *On pages 17, 18, 19, 32.*

Tufte, E. R. (1990). *Envisioning Information*. Graphics Press. *On pages 17, 32.*

Tufte, E. R. (1997). *Visual Explanations: Images and Quantities, Evidence and Narrative*. Graphics Press. *On pages 17, 32.*

Tukey, J. W. (1977). *Exploratory Data Analysis*. Pearson. *On pages 3, 17.*

van den Elzen, S. and van Wijk, J. (2014). Multivariate network exploration and presentation: From detail to overview via selections and aggregations. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2310–2319. *On page 23.*

van Ham, F. (2003). Using multilevel call matrices in large software projects. In *Proceedings of the IEEE Symposium on Information Visualization*, INFOVIS'03, pages 227–232. *On page 22.*

van Ham, F. and Perer, A. (2009). ”search, show context, expand on demand”: Supporting large graph exploration with degree-of-interest. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):953–960. *On pages 23, 28, 135, 139, 145.*

van Ham, F. and van Wijk, J. J. (2003). Beamtrees: compact visualization of large hierarchies. *Information Visualization*, 2:31–39. *On page 22.*

van Wijk, J. (2005). The value of visualization. In *IEEE Visualization*, pages 79–86. *On pages 28, 174.*

van Wijk, J. and Nuij, W. (2004). A model for smooth viewing and navigation of large 2d information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):447–458. *On page 26.*

van Wijk, J. J. and van Liere, R. (1993). Hyperslice: visualization of scalar functions of many variables. In *Proceedings of the 4th conference on Visualization*, VIS ’93, pages 119–125. *On page 20.*

Van Wijk, J. J. and Van Selow, E. R. (1999). Cluster and calendar based visualization of time series data. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 4–9. *On page 19.*

Wang, W., Wang, H., Dai, G., and Wang, H. (2006). Visualization of large hierarchical data by circle packing. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI ’06, pages 517–520. *On page 22.*

Wang Baldonado, M. Q., Woodruff, A., and Kuchinsky, A. (2000). Guidelines for using multiple views in information visualization. In *Proceedings of the working conference on Advanced visual interfaces*, AVI ’00, pages 110–119. *On page 4.*

Ware, C. (2004). *Information Visualization: Perception for Design*. Morgan Kaufmann. *On pages 7, 9, 15, 30, 31, 114, 163, 177, 183.*

Ware, C. and Lewis, M. (1995). The dragmag image magnifier. In *Proceedings of Conference Companion on Human Factors in Computing Systems*, CHI ’95, pages 407–408. *On page 25.*

Wattenberg, M. (2006). Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI ’06, pages 811–819. *On pages 23, 135, 139, 143.*

Weber, M., Alexa, M., and Muller, W. (2001). Visualizing time-series on spirals. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 7–13. *On pages 19, 43.*

Wong, P. C. and Bergeron, R. D. (1997). 30 years of multidimensional multivariate visualization. In *Scientific Visualization, Overviews, Methodologies, and Techniques*, pages 3–33. *On page 20.*

Wongsuphasawat, K. and Gotz, D. (2012). Exploring flow, factors, and outcomes of temporal event sequences with the outflow visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2659–2668. *On pages 19, 119.*

Wongsuphasawat, K., Guerra Gómez, J. A., Plaisant, C., Wang, T. D., Taieb-Maimon, M., and Shneiderman, B. (2011). Lifeflow: visualizing an overview of event sequences. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 1747–1756. *On pages 19, 119.*

Xie, C., Chen, W., Huang, X., Hu, Y., Barlowe, S., and Yang, J. (2014). Vaet: A visual analytics approach for e-transactions time-series. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1743–1752. *On page 19.*

Yee, K.-P., Fisher, D., Dhamija, R., and Hearst, M. (2001). Animated exploration of dynamic graphs with radial layout. In *Proceedings of the IEEE Symposium on Information Visualization*, pages 43–50. *On page 23.*

Yee, K.-P., Swearingen, K., Li, K., and Hearst, M. (2003). Faceted metadata for image search and browsing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '03, pages 401–408. *On pages 21, 135.*

Zhai, S. and Milgram, P. (1998). Quantifying coordination in multiple dof movement and its application to evaluating 6 dof input devices. In *Proceedings of the ACM conference on Human factors in computing systems*, pages 320–327. *On page 31.*

Zhao, J., Chevalier, F., and Balakrishnan, R. (2011a). Kronominer: using multi-foci navigation for the visual exploration of time-series data. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 1737–1746. *On page 41.*

Zhao, J., Chevalier, F., Collins, C., and Balakrishnan, R. (2012a). Facilitating discourse analysis with interactive visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2639–2648. *On page 89.*

Zhao, J., Chevalier, F., Pietriga, E., and Balakrishnan, R. (2011b). Exploratory analysis of time-series with chronolenses. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2422–2431. *On page 63*.

Zhao, J., Collins, C., Chevalier, F., and Balakrishnan, R. (2013). Interactive exploration of implicit and explicit relations in faceted datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2080–2089. *On page 133*.

Zhao, J., Drucker, S. M., Fisher, D., and Brinkman, D. (2012b). Timeslice: interactive faceted browsing of timeline data. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 433–436. *On page 117*.

Zuk, T., Schlesier, L., Neumann, P., Hancock, M. S., and Carpendale, S. (2006). Heuristics for information visualization evaluation. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*, BELIV '06, pages 1–6. *On page 33*.