

Contents

1	Introduction	1
1.1	Research Problem and Approach	3
1.1.1	Problem	3
1.1.2	Approach	5
1.2	Thesis Contributions	7
1.3	Thesis Outline	10
2	Literature Review	11
2.1	Sensemaking	11
2.1.1	Gap-Bridging Metaphor	11
2.1.2	Learning Loop Complex	13
2.1.3	Sensemaking in Organizations	14
2.1.4	A Process Model	14
2.1.5	Data–Frame Model	16
2.2	Visualization and Visual Analytics	18
2.2.1	Overview	18
2.2.2	Visualization Design	18
2.2.3	Automated Analysis Methods	18
2.3	Provenance	32
2.3.1	Data Provenance	33
2.3.2	Analytic Provenance	34
2.4	Visualization of Provenance Data	40
2.4.1	Visualizing the Analysis Process	40

2.4.2	Visualizing the Reasoning Process	48
2.5	Visualization of Time-Oriented Data	50
2.5.1	Horizontal	52
2.5.2	Spiral	57
2.5.3	Circle	59
2.5.4	Calendar	61
2.5.5	Small Multiples	61
2.5.6	Animation	62
2.6	Visualization of Network and Tree Data	64
2.6.1	Node-Link Diagrams	64
2.6.2	Matrix Views	67
2.6.3	Space-Filling Techniques	69
3	Exploring Temporal Relationship through User Annotations	72
3.1	Introduction	72
3.2	Approach and Requirements	74
3.3	Visual Design	75
3.3.1	Event	75
3.3.2	Schema	76
3.3.3	Interaction	77
3.4	Algorithm	79
3.4.1	SchemaLine Layout	79
3.4.2	Schema Outline	81
3.5	Evaluation	83
3.5.1	Application	83
3.5.2	Case Study	84
3.6	Summary	89
4	Exploring Complex Temporal Relationship	90
4.1	Introduction	90
4.2	Related Work	92
4.2.1	Set Relations in Timelines	92
4.2.2	Set Visualizations	93
4.3	Visual Design	94
4.3.1	Event	94
4.3.2	Set	94
4.3.3	Interaction	99

4.4	Layout	100
4.4.1	Sets Ordering	101
4.4.2	Layer Layout	101
4.4.3	Layers Compacting	103
4.4.4	Layers Balancing	104
4.4.5	Scalability	104
4.5	Evaluation	105
4.5.1	Method	105
4.5.2	Hypotheses	109
4.5.3	Results	110
4.5.4	Discussion	112
4.6	Case Study 1: Publication Data	115
4.7	Case Study 2: VAST Challenge 2014	116
4.8	Summary	118
5	Exploring Rational Relationship through User Actions	119
5.1	Introduction	120
5.2	Related Work on Qualitative Research	121
5.3	Approach	122
5.3.1	Qualitative Analysis of Sensemaking	122
5.3.2	Requirements	124
5.4	Interface Design	125
5.4.1	Approach	125
5.4.2	Overview	126
5.4.3	Provenance Capture	127
5.4.4	Timeline View	128
5.4.5	Browser View	133
5.4.6	Replay View	133
5.4.7	Transcription View	133
5.4.8	Implementation	134
5.5	Evaluation	137
5.5.1	Online Sensemaking Task	138
5.5.2	Evaluation 1	138
5.5.3	Evaluation 2	141
5.6	Summary	143

6 Exploring Complex Rational Relationship	144
6.1 Introduction	145
6.2 Approach	146
6.2.1 Design Research	147
6.2.2 Sensemaking Model	149
6.2.3 Design Workshops	150
6.3 Interface Design	152
6.3.1 Approach	152
6.3.2 Overview	153
6.3.3 Browser View	154
6.3.4 History Map	154
6.3.5 Knowledge Map	158
6.3.6 Communication	160
6.3.7 Implementation	161
6.4 Evaluation	161
6.4.1 Method	161
6.4.2 Data Analysis	163
6.4.3 Discussion	168
6.5 Summary	171
7 Conclusion	172
7.1 Review of Contributions	172
7.1.1 Exploring Temporal Relationship of Sensemaking	172
7.1.2 Exploring Complex Temporal Relationship of Sensemaking	173
7.1.3 Exploring Rational Relationship of Sensemaking	173
7.1.4 Exploring Complex Rational Relationship of Sensemaking	175
7.2 Limitations and Future Research	175
7.2.1 TimeSets	175
7.2.2 SensePath	176
7.2.3 SenseMap	177
7.3 Closing Remarks	178
References	179

1

Introduction

In a broad context, *sensemaking* reflects how we make sense of the world so that we can take further actions in it [162]. More specifically, sensemaking is described as a motivated and continuous effort to understand the relationships among people, places and events around us in order to act more effectively [94]. It is the process of collecting, organizing and representing complex information sets with an emphasis on some problem we need to solve [151]. For example, intelligence analysts may need to examine thousands of reports to establish deep understanding of particular persons or organizations before identifying possible threats from them. In an everyday context such as selecting a smartwatch, a person may search for different models on the Internet, learn unknown technical terminologies, and consider pros and cons between these identified models. The data around the world has been produced more rapidly than ever before, with major sources including sensors, machine logs, public websites and social media. For a single day, 50 million photos are uploaded on Instagram, 500 million tweets are sent on Twitter, 3.5 billion searches are performed on Google and 200 billion emails are sent¹. This data deluge could help us find more relevant data to our problems; however, making sense of such a vast amount of data is a huge challenge.

Visualizations can help people perform tasks more effectively through visual representations of datasets [119]. They can display a large amount of data in a way that quickly reveals hidden patterns. Interaction allows people to further investigate the visualized data in more detail and explore the relationship between identified patterns. Automated techniques in information retrieval and data mining can help speed up the sensemaking process. For instance, named-entity recognition

¹<http://www.internetlivestats.com/>

techniques [120] can identify entities from text and classify them into predefined categories such as persons, organizations and locations. This automation saves analysts a considerable amount of time in their common tasks such as findings all reports mentioning a particular person. *Visual analytics* combines automated analysis techniques and interactive visualizations to effectively make sense of very large and complex datasets [90]. An immediate visualization of all data points of a large dataset could be messy and meaningless. Therefore, it is necessary to analyze the dataset before showing the most relevant aspects of the data.

Research in visual analytics focuses on helping people reveal hidden patterns in the data. However, sensemaking requires support beyond that. Analysts may connect individual patterns, explore the relationships between them, generate possible hypotheses explaining those relationships, and find ways to verify them. Unfortunately, people with limited capacity of working memory cannot hold all of these artifacts simultaneously. They may forget previous findings and relations between them, or remember but fail to retrieve the information needed. They may also forget how those findings were derived, making it more difficult to find similar information. Especially for long and interrupted tasks with big datasets, people often get lost in the task space: they are unable to examine their progresses, unable to synthesize their discoveries, and unable to decide the next step effectively.

Analytic provenance has the potential to address these problems. It is a subfield of visual analytics, focusing on understanding a user's reasoning process through the study of their interactions with the sensemaking system [127]. Analytic provenance captures both the interactive data exploration process and the accompanied reasoning process during sensemaking [195], releasing analysts from a burden of keeping track of their discoveries. Analytic provenance information can be categorized using a multiple layer model based on its semantic richness [61]. Low-level provenance can be captured automatically but contains little semantics. Whereas, higher level provenance primarily has to be captured manually but contains richer semantics. Visualizations of provenance data have been developed for different purposes including recall, replication, action recovery, collaborative communication, presentation and meta-analysis [143]. Most of these visualizations provide post hoc applications of provenance or stop at recalling the process and revisiting past steps. They are not specifically designed to support the ongoing, iterative and dynamic sensemaking process.

1.1 Research Problem and Approach

We propose a cyclic process model, in which analytic provenance can be used to support sensemaking, illustrated in Figure 1.1. The process starts with a *user* employing a *sensemaking system* to solve a task. The sensemaking system could be any computer-based applications, such as a simple visualization tool, a complex visual analytics system and a standard web browser. During the sensemaking process, both the performed low-level actions (e.g., visualization interaction such as sort, filter and zoom) and the produced high-level reasoning artifacts (e.g., findings, assumptions and hypotheses) are captured. They are referred as *provenance data* in the figure. Then, this provenance data should be visualized in a way that can provide support back to the ongoing sensemaking process. In this model, the user can interact with both the sensemaking system and the *provenance visualization* to solve his or her task. The provenance visualization should be able to communicate with the sensemaking system to facilitate the interplay between the user and these two. Technically, it could be part of the sensemaking system by design or a plug-in to the existing system to augment the provenance capability.

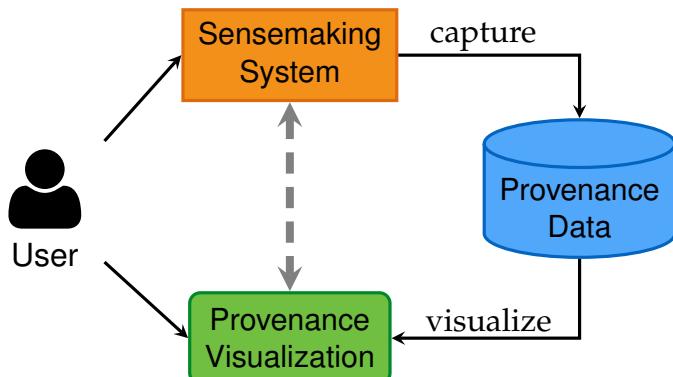


Figure 1.1: A cyclic process model of supporting sensemaking through analytic provenance. While employing a sensemaking system to solve a task, user interaction and reasoning are captured and visualized to provide support back to the sensemaking process. The sensemaking system can be any computer-based applications such as visual analytics systems and web browsers. The provenance visualization should be able to communicate with the sensemaking system to provide support. As a result, the user can interact with both the sensemaking system and the provenance visualization to solve the task.

1.1.1 Problem

The central research problem of this thesis is

How to design interactive visualizations of provenance data for supporting sensemaking?

We elaborate this research problem by discussing what data will be visualized and what task does the visualization aim to support.

1. What **data** will be visualized?

Both low-level actions (e.g., visualization interaction such as sort, filter and zoom) and high-level reasoning artifacts (e.g., findings, assumptions and hypotheses) are considered in this thesis. However, we do not examine bottom-level events such as mouse clicks and keystrokes because they contain very little semantics [61]. One essential attribute associated with every type of provenance data is *time*, providing when a data item is collected. This attribute allows reviewing the sensemaking process in chronological order.

To support complex sensemaking tasks, it could be useful to provide additional relationship to the raw collected data. This could come from an automated process such as by assigning *topics* from a topic modeling technique [14] to the provenance data. This could also come from a manual assessment, in which a user can select relevant data items and assign *pair-wise relations*. More abstractly, the provenance data could contain additional *categorical* attribute or become a *network* dataset, respectively.

2. What **task** does the visualization aim to support?

The characteristics of provenance data suggests the tasks it can support the users. First, the inherently temporal aspect of provenance data provides an opportunity to allow users to identify *temporal* patterns and relationship in the sensemaking task. This could enable users to recall their processes and reveal hidden storylines in the data. Timeline visualizations are often used to show such temporal relationship. However they are not specifically designed to support the iterative and dynamic nature of sensemaking. They are mainly used to present a known story instead of constructing a hidden one.

As discussed earlier, grouping information can be added to help make sense of more complex temporal relationship. It allows users to examine storylines corresponding to individual groups and the interaction between them. However, existing visualizations are unable to effectively show both temporal and categorical information.

After understanding how things happened, exploring why things happened like that is often the next and important step. Understanding how users make sense of tasks is essential to build effective tools to support them. Can provenance data, especially low-level actions, help an analyst understand the *rational* relationship of the sensemaking process?

As discussed earlier, during sensemaking, users can enrich the provenance data by grouping and adding connections. Can the enriched provenance data help users explore more complex rational relationship of sensemaking? Does the benefit that users gain outweigh the overload that users involve in providing extra information richness?

Figure 1.2 summarizes the four research challenges described in this data – task analysis.

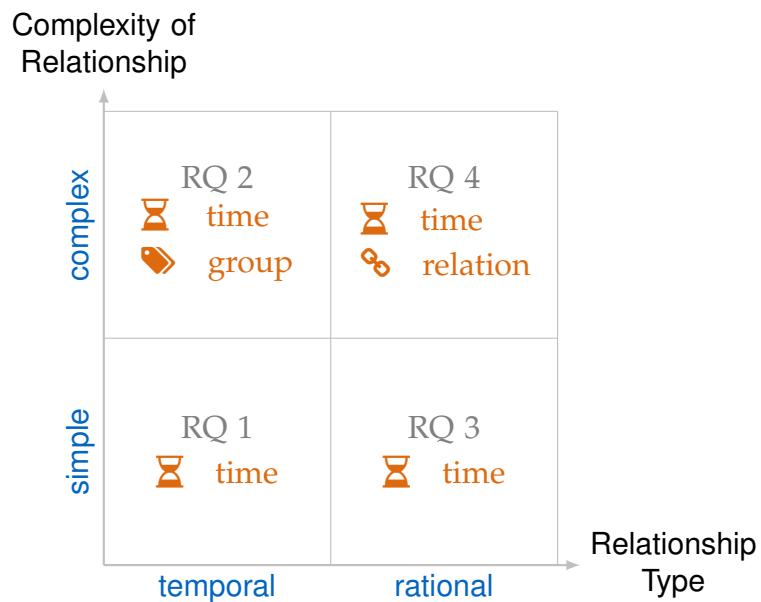


Figure 1.2: Research problem split by **data** and **task**. The horizontal axis represents different tasks and the vertical axis represents the complexity involved in the task. The cells show the characteristics of data will be used to support the task.

1.1.2 Approach

The research problem is broken into the following research questions based on the previous data – task elaboration. They map to the four cells in Figure 1.2. For each question, we explain its context based on the sensemaking-support model described

earlier: the sensemaking system, the user, the task and the provenance data. Note that in our text, we color code the **data** using orange and **task** using blue if we want to emphasize them.

1. How to design interactive visualizations of **time-oriented provenance data** enabling users to explore **temporal relationship of sensemaking**?

The domain we want to focus is *intelligence analysis* – the original domain that set the foundation for visual analytics research [172]. A common task for intelligence analysts is to examine thousands of reports to identify possible threats from particular persons or organizations. Many visual analytics have been designed to facilitate this analysis. These systems often allow users to take notes, recording the users' thought. We aim to augment these systems by a novel timeline visualization that enables users to identify temporal patterns and construct hidden stories from their annotations.

2. How to design interactive visualizations that can utilize both **temporal and categorical provenance data** to reveal **complex temporal relationship of sensemaking**?

This shares the same context with the previous research question. To the best of our knowledge, no existing visualization techniques is designed to show both temporal and categorical information effectively. Therefore, we also aim to design a general timeline visualization that can apply to other domains.

3. How to design interactive visualizations that can exploit **time-oriented provenance data** to enable users to explore **rational relationship of sensemaking**?

To explore rational relationship of sensemaking, analysts often conduct a user study to observe the process, collect relevant data and analyze it. During the qualitative analysis, analysts may need to go through a lengthy process of transcribing the video recordings before coding the transcript with themes and building model based those themes. In this context, the sensemaking system mapped to Figure 1.1 can be considered as a set of tools allowing the analysts to conduct a user study and analyze the collected data. We take a different approach to facilitate the transcription and coding steps. We capture analytic provenance while a user performs a sensemaking task and provide visualizations of the provenance data to enable the analyst explore the rational relationship of the user's sensemaking process. We focus only on browser-

based sensemaking tasks. The provenance data could include information about visited web pages such as page URL, title and visit time.

4. How to design interactive visualizations that can combine both **time-oriented provenance data and user-added relations** to enable users to explore **complex rational relationship of sensemaking?**

This research question targets a very common domain: browser-based online sensemaking. To make sense of everyday tasks such as selecting a camera, people often use standard web browsers – simple sensemaking systems – to search for different models and consider their strengths and weaknesses. The provenance data could include both automatically captured low-level actions as in the previous question and user-added relationship of the automatic capture such as by spatially grouping them or assigning pair-wise connections.

We take a user-centered design approach in seeking solutions to all the research questions. For each question, we elicit the design requirements by conducting a user study and/or drawing from the literature. Visual encoding and interaction are designed to meet those requirements, and the designs are implemented into a working prototype. Finally, an empirical study is conducted to explore how the tool is used by target audience and check whether it provides the intended support.

1.2 Thesis Contributions

Toward the overall goal of supporting users in their sensemaking processes through the visualizations of provenance data, this thesis contributes:

- A timeline visualization technique – *SchemaLine* – that enables users to examine information in chronological order, identify temporal patterns and construct narratives from relevant user annotations. SchemaLine produces a compact but aesthetically pleasing layout and provides a set of fluid interactions allowing users to perform various sensemaking activities described in the Data–Frame model [95]. This is to address Research Question 1, and the work is published as

P. H. Nguyen, K. Xu, R. Walker, and B. L. W. Wong. SchemaLine: Timeline Visualization for Sensemaking. In *International Conference on Information Visualization*, pages 225–233. IEEE, jul 2014.

- A timeline visualization technique – *TimeSets* – that enables users to explore complex temporal relationship by effectively representing both temporal and categorical provenance data. TimeSets visually groups data items that share the same group but still preserves their temporal order. It color codes the backgrounds of the entire groups to distinguish them and uses colored gradient backgrounds for the intersections among those groups. It also adjusts the level of details of each data item dynamically to accommodate more items within a given display estate. This is to address Research Question 2, and the work is published as

P. H. Nguyen, K. Xu, R. Walker, and B. L. W. Wong. TimeSets: Timeline visualization with set relations. *Information Visualization*, 15(3):253–269, jul 2016.

and for the VAST Challenge case study as

- K. Xu, **P. H. Nguyen**, and B. Fields. Visual analysis of streaming data with SAVI and SenseMAP. In *IEEE Conference on Visual Analytics Science and Technology*, pages 389–390. IEEE, oct 2014.
- A visual sensemaking tool – *SensePath* – that enables analysts to explore rational relationship of the sensemaking process. Qualitative research methods are often used in understanding rational relationship of sensemaking. This is a manual and time-consuming process: researchers collect observation data, transcribe screen capture videos and think-aloud recordings, identify recurring patterns, and eventually abstract the sensemaking process into a general model. SensePath offers an alternative and possibly faster approach in performing transcription and coding. In stead of having to transcribe the video, SensePath automatically captures and detects participant’s sensemaking actions, and provides multi-linked visualizations to support further analysis. It visualizes provenance data in a timeline that enables researchers to quickly gain an overview of the sensemaking process and identify recurring sensemaking patterns. It also links with a screen capture video to allow researchers to examine additional context when necessary. Finally, to enable researchers to continue working on later stages of analysis using their normal workflow, SensePath exports its coded transcript in a common format that can be used by other popular qualitative data analysis software packages. This is to address Research Question 3, and the work is published as

P. H. Nguyen, K. Xu, A. Wheat, B. L. W. Wong, S. Attfield, and B. Fields. SensePath: Understanding the Sensemaking Process through Analytic Provenance. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):41–50, jan 2016.

- A visual sensemaking tool – *SenseMap* – that enables users to explore complex rational relationship of sensemaking. It automatically captures and detects sensemaking actions and relationships between these actions before visualizing both of them in a branching history tree. This allows users to examine the rational relationship between the actions they performed and potentially helps them remind of what have been done earlier. SenseMap offers users to assign additional meaning to the automatically collected data by spatially grouping actions or adding rational links between them, in order to help explain complex relationship. Finally, SenseMap allows users to communicate their analysis results at different levels of granularity including a big picture of user-organized findings, a more detailed analysis process and raw provenance data captured. This is to address Research Question 4, and the work is published as

P. H. Nguyen, K. Xu, A. Bardill, S. Betul, K. Herd, and B. L. W. Wong. SenseMap: Supporting Browser-based Online Sensemaking through Analytic Provenance. In *IEEE Conference on Visual Analytics Science and Technology*, 2016.

Besides the main contributions described in this thesis, I also coauthored two other publications, contributing my knowledge in provenance for sensemaking.

- I contributed to the discussion and the design of a framework for provenance in human terrain analysis, and the work was published as

R. Walker, A. Slingsby, J. Dykes, K. Xu, J. Wood, **P. H. Nguyen**, D. Stephens, B. L. W. Wong, and Y. Zheng. An extensible framework for provenance in human terrain visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2139–2148, dec 2013.
- I contributed to the organization of a IEEE VIS workshop on provenance for sensemaking. The discussion at the workshop was published as

K. Xu, S. Attfield, T. J. Jankun-Kelly, A. Wheat, **P. H. Nguyen**, and N. Selvaraj. Analytic provenance for sensemaking: a research agenda. *IEEE Computer Graphics and Applications*, 35(3):56–64, jan 2015.

1.3 Thesis Outline

The remainder of this thesis is organized as follows.

First, Chapter 2 reviews the core work related to sensemaking, analytic provenance and visualization. Then, it emphasizes on the visualization of provenance data for supporting sensemaking. At the end, this chapter presents visualization techniques of general time-oriented and network data because these types of data have similar characteristics as provenance data.

Chapter 3 discusses the SchemaLine timeline visualization of user annotations enabling the users to explore temporal relationship of sensemaking – addressing Research Question 1.

Chapter 4 extends Chapter 3 to present the TimeSets visualization technique that can effectively show both temporal and categorical provenance data in order to reveal complex temporal relationship of sensemaking – addressing Research Question 2.

Chapter 5 discusses the SensePath visualization tool that can exploit time-oriented provenance data, enabling users to explore rational relationship of sensemaking – addressing Research Question 3.

Chapter 6 describes the SenseMap visualization tool that can utilize both temporal and relational provenance data enabling users to explore complex rational relationship of sensemaking – addressing Research Question 4.

Finally, Chapter 7 concludes the thesis with a discussion on its contributions and future research directions triggered from this work.

2

Literature Review

In this chapter, we first discuss the core work in individual fields related to our research problem including sensemaking, visualization and analytic provenance. Then, we focus on the intersection of these three fields: visualization of provenance data for supporting sensemaking. At the end, we present visualization techniques of general time-oriented and network data because these two data types possess similar characteristics as provenance data.

2.1 Sensemaking

Sensemaking reflects how we make sense of the world so that we can act in it [162]. Sensemaking has been studied in different contexts such as information science [41], human-computer interaction [152], organizational studies [188], and intelligence analysis [95, 136]. In this section, we review the sensemaking concepts discussed in these contexts, with an emphasis on the last two sensemaking models that have been highly applied in the visualization community. A recent and comprehensive review of sensemaking can be found in the article by Maitlis and Christianson [109].

2.1.1 Gap-Bridging Metaphor

Dervin develops a sensemaking theory focusing on information seeking and use behaviors [41]. It underlies the cognitive gap that individuals experience when attempting to make sense of observed data. Figure 2.1 summarizes this *gap-bridging* metaphor. The theory assumes that people moves through time-space in some particular context and situation. Sensemaking starts when they encounter a gap that needs to overcome such as something is unclear or confused. To bridge the gap, they

may seek and use information from a variety of sources such as documents, media and other people. These sources are evaluated based on relevant attributes to assess their usefulness.

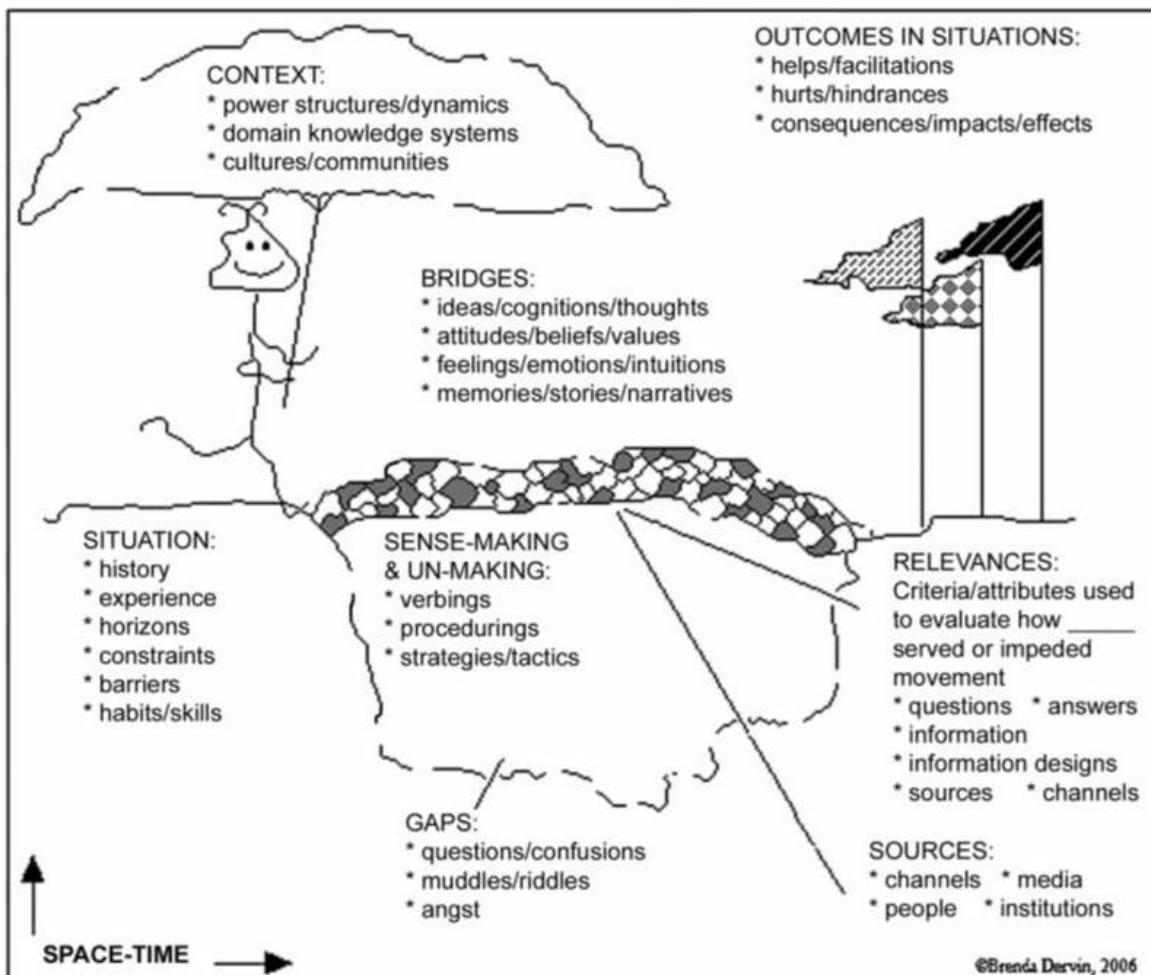


Figure 2.1: The gap-bridging metaphor of sensemaking. People encounter gaps when moving through time-space, then seek for information, evaluate and use it to bridge the gaps. *Image source: [42].*

Dervin also implements the theory into a set of questions that can be used in interview to understand sensemaking within a context [41]. The questions elaborate all parts of the model, aiming to establish an understanding on the situation (*What happened?*), the gap (*What did you struggle with?*), the bridge (*What idea did you come to?*) and the outcome (*How did that help?*).

2.1.2 Learning Loop Complex

In the context of human-computer interaction, Russell et al. [152] defines sensemaking as the process of searching for a representation and encoding data in that representation to answer task-specific questions. That cyclic process is called the *learning loop complex* as illustrated in Figure 2.2. First, the sensemaker searches for a representation to capture salient features of the data (*Generation Loop*). During sensemaking, new information is sought and encoded into this representation (*Data Coverage Loop*). The data unfit to the representation (*residue*) requires the sensemaker to adjust and produces a more suitable one. This entire learning loop complex is guided by the task with an aim to reduce its cost.

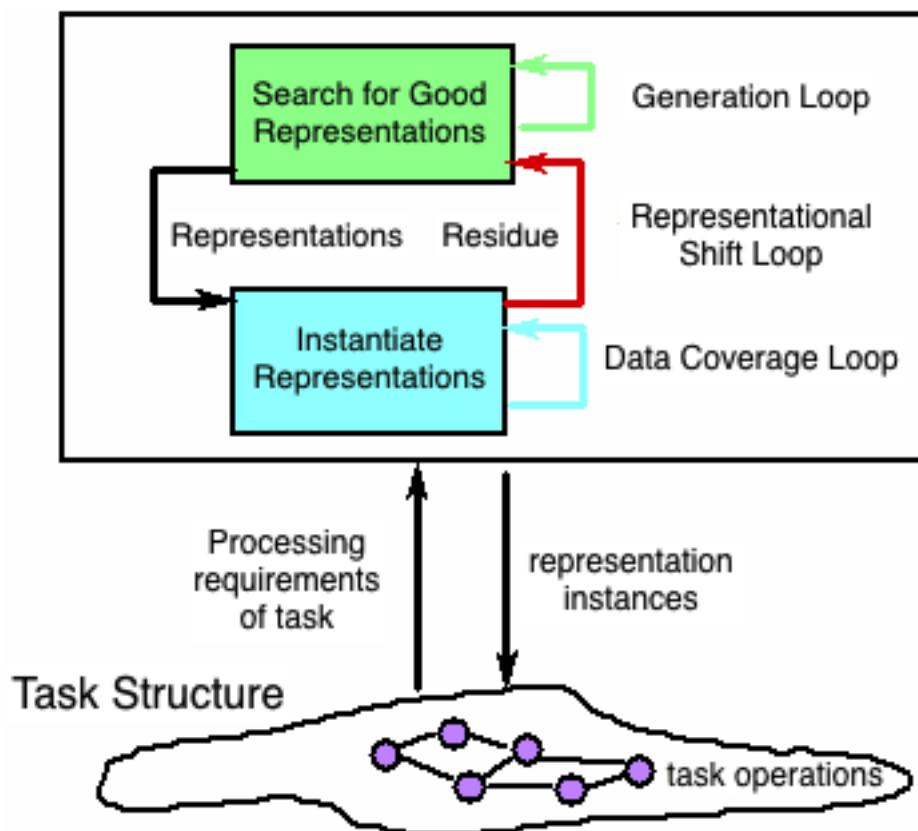


Figure 2.2: The learning loop complex theory of sensemaking. It consists of three iterative loops: searching for a good representation, encoding data to the representation, and adjusting the representation for a better data coverage. *Image source: [152].*

2.1.3 Sensemaking in Organizations

Different from Dervin and Russell who study sensemaking for individuals, Weick focuses on sensemaking at an organization level [188]. He proposes that sensemaking consists of these seven following properties.

1. *Grounded in identity construction.* Who people think they are, both individually and collectively, affect what they interpret and act.
2. *Retrospective.* People look back and make sense from what they have said and what they have done before.
3. *Enactive of sensible environments.* People make sense and contribute to the environments during their sensemaking processes.
4. *Social.* This is an inherent property of sensemaking in organization where people interact and socialize with others, and also are influenced by others.
5. *Ongoing.* Sensemaking is a continuous flow because the world and our understanding about the word are constantly changing.
6. *Focused on and by extracted cues.* Cues are things from the context that people have attention to and may use them to guide further exploration and assessment of the sensemaking problem.
7. *Driven by plausibility rather than accuracy.* Sensemaking is about plausibility and sufficiency rather than accuracy and completeness. People tend to stop searching when they find an acceptable solution.

2.1.4 A Process Model

Pirolli and Card [136] describe sensemaking as an iterative process that gradually transforms raw data into rational knowledge. The process includes two sets of activities: one that cycles around finding relevant information, and another that cycles around making sense of that information, with plenty of interaction between them. They map to the *foraging loop* and the *sensemaking loop* respectively, as shown in Figure 2.3. The sensemaking process can progress upward (from data to knowledge) or downward (from knowledge to data). The steps in the *bottom-up* process are summarized as follows.

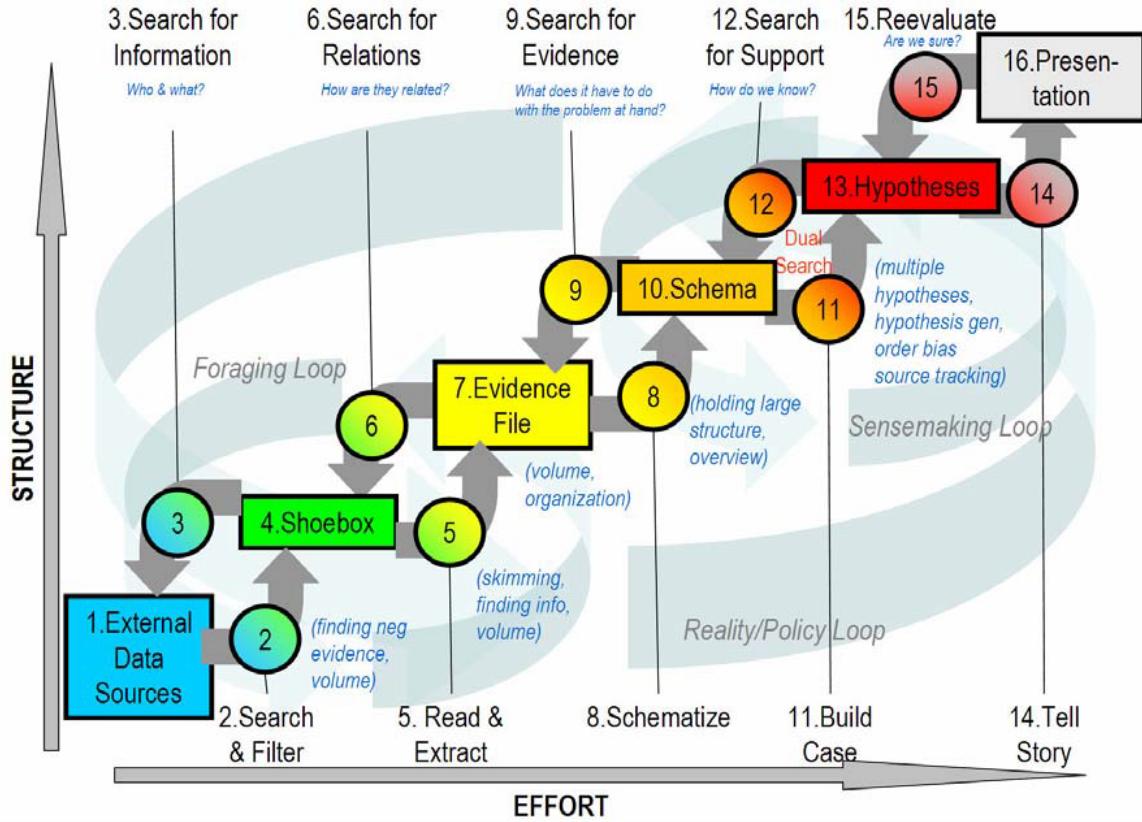


Figure 2.3: A notional model of sensemaking. The sub-processes (numbered circles) and their data input/output (numbered rectangles) are arranged in a two-dimensional space, in which the horizontal axis represents the degree of effort from users, and the vertical axis represents the degree of structure in information representation. *Image source: [136].*

- *Search and filter.* External data sources, such as classified databases or the web, are searched and filtered to retrieve relevant documents to the task.
- *Read and extract.* These documents are examined to extract pieces of information that may be used as evidence later.
- *Schematize.* The collected information is organized in a way that aids the analysis. This may be executed in user mind, using paper and pen, or with a complex computer-based system.
- *Build case.* Multiple hypotheses are generated; evidence are marshaled to support or disconfirm them.
- *Tell story.* Discovered cases are presented to some audience.

In this model, *schematization* plays an important role in converting raw evidence to rational explanations, bridging the foraging and sensemaking loops. A study by Kang, Görg and John Stasko [88] agrees with this observation. In their study, all the participants who performed the sensemaking task well spent considerable time and effort in organizing their collected information. Their organizational schemes were flexible: a *timeline* of related events, a *map* connecting locations that a person has been to, and a *diagram* showing relationships among suspicious targets.

2.1.5 Data–Frame Model

Klein et al. [95] propose a sensemaking model that centers around *data* and *frame*. Data is the information that a person receives or searches for, and frame is the mental structure that organizes and explains the relationship of such data. For instance, a frame can be a *story*, explaining the chronology of events and the causal relationships between them; or a *map*, showing where the events take place and the routes between them. Sensemaking is considered as a deliberate effort to understand an event, starting when a person realizes a gap of their current understanding of that event. Klein and his associates describe seven activities involved in sensemaking and are summarized in Figure 2.4.

- *Connect data and a frame.* A person recognizes relevant pieces of data and constructs an initial frame to explain them. The frame then helps the person to filter and search for new data.
- *Elaborate the frame.* As more is learned about the situation, the frame becomes more elaborate with new data and new relationships.
- *Question the frame.* It happens when a person encounters data that is inconsistent with the existing frame. At this point, the person may be unsure that the frame is incorrect, or the inconsistent data is inaccurate.
- *Preserve the frame.* A person may consider the severity of the inconsistent data, justify why it mismatches the frame, and ignore it.
- *Compare multiple frames.* Depending on experience, a person may think of alternative frames explaining the same set of data. These frames need to be compared to select the most likely one.

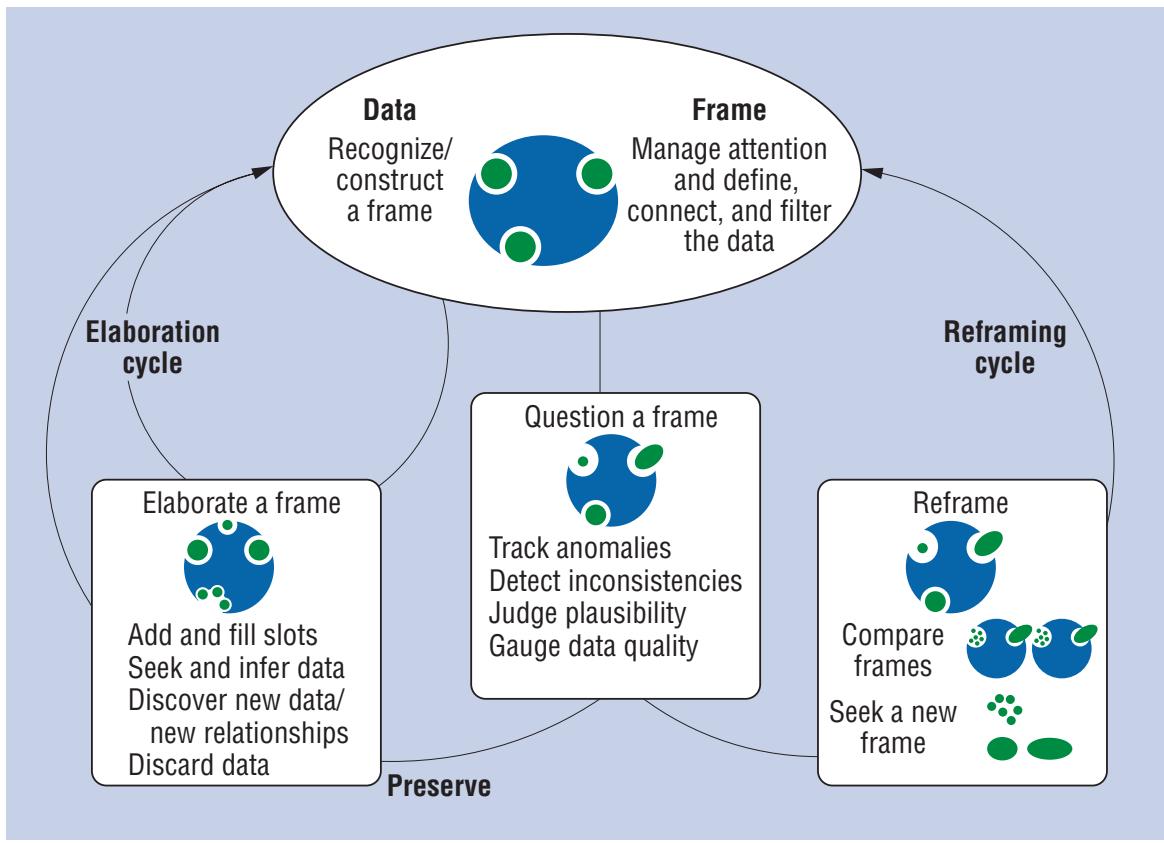


Figure 2.4: The data–frame model of sensemaking. It describes a set of interconnected sensemaking activities centering around data and frame – the explanatory structure of data. *Image source: [95].*

- *Reframe*. When encountering inconsistent and contrary data, the person may need to find a replacement that can explain all data. Considering discarded data and/or reinterpreting data could facilitate this activity.
- *Seek a new frame*. A person may deliberately search for a new frame when encountering plenty of conflicted data. One or two key data elements may serve as *anchors* to help the person to elicit another frame.

The Pirolli and Card's model describes a step-by-step process of sensemaking, in which the analyst collects relevant data and eventually transposes it into rational answers. However, the various sensemaking activities in the Data–Frame model may explain the strategies used by the analyst more comprehensively.

2.2 Visualization and Visual Analytics

Computer-based visualization systems provide visual representations of datasets designed to help people carry out tasks more effectively [119]. Because the design space of possible visual “idioms” is huge, it is challenging to create effective visualizations. Understanding well-established information design principles and interaction techniques could guide designers toward the right direction. Also, every visualization needs to be evaluated to check whether it meets its design purposes and how it helps or hinder users.

2.2.1 Overview

TODO: Overview of vis and VA

2.2.2 Visualization Design

2.2.3 Automated Analysis Methods

Data mining is the computational process of extracting patterns in large datasets [167]. Data mining tasks can be broadly divided into two major categories:

- *Descriptive tasks.* The objective is to derive patterns (correlations, trends, clusters, trajectories and anomalies) that summarizes the underlying relationships in data. Examples include cluster and association analyses. *Clustering* aims to split data items to groups so that items within a group are more similar to each other than those in other groups. For example, clustering can be used to help marketers discover distinct groups of their customers before applying appropriate marketing strategies to those groups. *Association* analysis discovers the connections among a set of data items. For example, it can be used to identify products that customers often purchase together such as bread and milk.
- *Predictive tasks.* The objective is to predict the value of an unknown (target) attribute based on the values of other (explanatory) attributes. Examples include classification and regression. *Classification* predicts discrete target variables whereas, *regression* focuses on continuous ones. For example, predicting whether a customer buys a marketing product is a classification task because the target variable is binary. However, estimating a future house price is a regression task because price is a continuous-valued variable.

Next, we will discuss the clustering and classification tasks in more detail and how they are applied together with visualizations to help users gain deeper insight into the data. We also present some commonly used text mining techniques for exploring large sets of documents, which is essential in visual analytics [172].

2.2.3.1 Clustering

Cluster analysis finds similarities between data points based on the characteristics of data attributes and groups similar data points into clusters. Clustering is regarded as *unsupervised learning* [66] because it can reveal hidden structure of a dataset that does not have *labels* (or groups) defined. The most common clustering algorithm is *k-means* [106]. Given a set of data points (x_1, x_2, \dots, x_n) , k-means clustering aims to partition them into k clusters (S_1, S_2, \dots, S_n) , such that the within-cluster sum of squared distances is minimized:

$$\sum_{i=1}^k \sum_{x \in S_i} \|x - \mu_i\|^2$$

where μ_i is the center of points (i.e., centroids) in S_i .

The algorithms works as follows. Initially, partition data points into k non-empty random subsets. Then, compute the centroids of the current clusters, and assign each data point to the cluster with the nearest centroid. Repeatedly recompute the centroids and reassign the cluster of each data point until no assignment can be done. This k-means clustering algorithm is efficient but often terminates at a local optimal. More detailed analysis of k-means and other clustering algorithms are out of the scope of this thesis; they can be found in data mining textbooks [66, 167].

TODO: more specific clustering techniques/applications and how vis helps improve understanding

2.2.3.2 Classification

Classification predicts the value of a categorical (discrete or nominal) attribute based on the values of other attributes. It builds a model (or *classifier*) based on a labeled training dataset (i.e., *supervised learning*) and applies it in labeling new data [66]. The model needs to not only identify the labels in the training dataset well but also be general enough to predict the labels of new data correctly. One common and intuitive classification algorithm is decision tree induction [142]. Each non-leaf node represents a “test” on an attribute, which splits the node to multiple branches, each

for an outcome of the test. Each leaf node is associated with a class label and is the result of a sequence of tests starting from the root node.

The importance of building a decision tree is choosing which attribute to split at each node. Intuitively, we should choose attributes that can divide nodes into “pure” child nodes so that all data items in a child node belong to a single class and no further splits are needed. For example, in a binary classification, consider a training dataset with 10 records, 5 labeled “true” and 5 labeled “false”. Attribute A_1 splits the set to two subsets: (5 “true”, 0 “false”) and (0 “true”, 5 “false”). Attribute A_2 splits the set to (3 “true”, 2 “false”) and (2 “true”, 3 “false”). The subsets split by A_1 is “purer” than the one by A_2 because they do not contain a mix of “true” and “false”. To achieve this purity, several attribute measurements have been proposed such as *information gain* and *gini index* [167]. More detailed analysis of these measurements and other classification algorithms are out of the scope of this thesis; they can be found in data mining textbooks [66, 167].

TODO: more specific classification techniques/applications and how vis helps improve understanding

2.2.3.3 Information Design Principles

Marks and Channels Marks are basic geometric elements that depict items or links, and channels control their appearance [119]. Item marks can be zero-dimensional as a *point*, one-dimensional as a *line*, two-dimensional as an *area*, and three-dimensional as a *volume*, but rarely used. Link marks include *connection* showing a pairwise relationship between two items using a line and *containment* showing hierarchical relationships using areas. Figure 2.5 illustrates these marks.

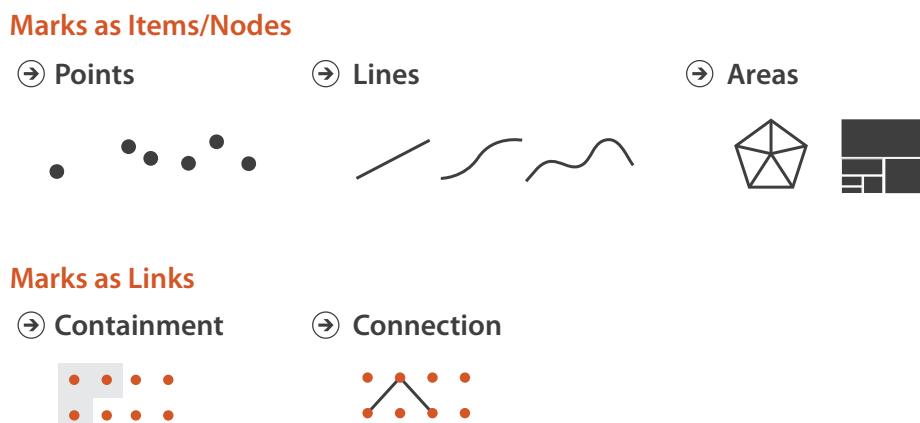


Figure 2.5: Item and link marks as geometric primitives. *Image source: [119].*

A visual channel control the appearance of marks, independent of their dimensionality. Examples include position, color, shape, angle and size. However, not all channels can be applied to all marks. For instance, an area mark is used in a geographic map to denote a region. It is already associated with a shape and size, thus cannot be size coded to represent another quantitative attribute. some visual channels. some cannot encode size, example. Figure 2.6 shows a progression of chart types, with each showing one more data attribute using one more visual channel. Figure 2.6a shows a bar chart representing a single quantitative attribute using the *vertical position* channel. Figure 2.6b shows a scatter plot encoding the second quantitative attribute using the *horizontal position* channel. Figure 2.6c adds the *color* channel to represent a categorical attribute, and Figure 2.6d adds the *size* channel to represent another quantitative attribute. In these examples, each attribute is encoded with a single channel. However, multiple channels can be combined to redundantly encode the same attribute, helping perceive it more easily.

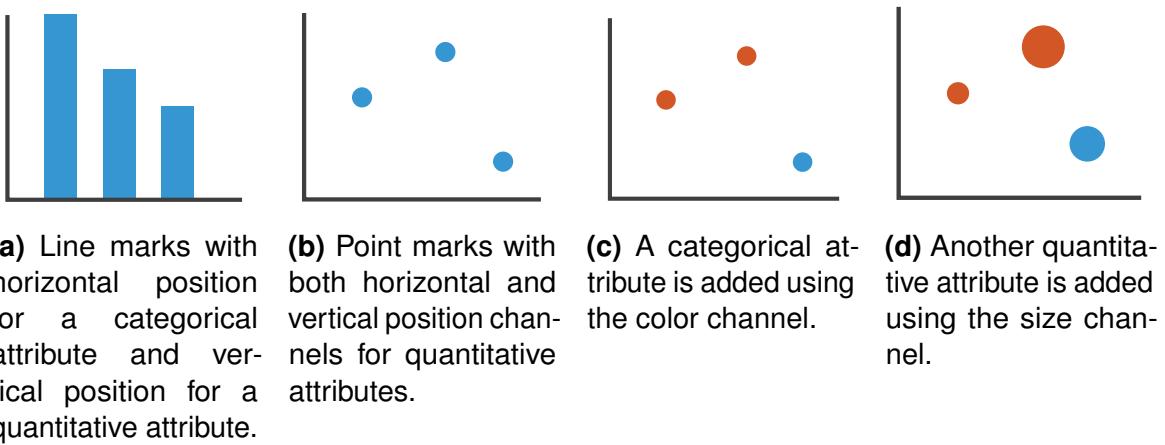


Figure 2.6: Using marks and channels.

All channels are not equal; they are processed and perceived differently by our human visual systems. Also, not all channels are appropriate for encoding both ordered and categorical attributes. Ordered attributes should be shown using magnitude channels, with *aligned spatial position* as the most effective channel and *3D volume* as the least effective one. Categorical attributes should be shown using identity channels, with *spatial region* as the most effective channel and *shape* as the least effective one. Figure 2.7 shows the detailed ranking of effectiveness of many visual channels, separated by the type of attribute. This ranking is documented by Munzner [119], based on many empirical studies such as the work by Cleveland and McGill [30], and by Heer and Bostock [71].

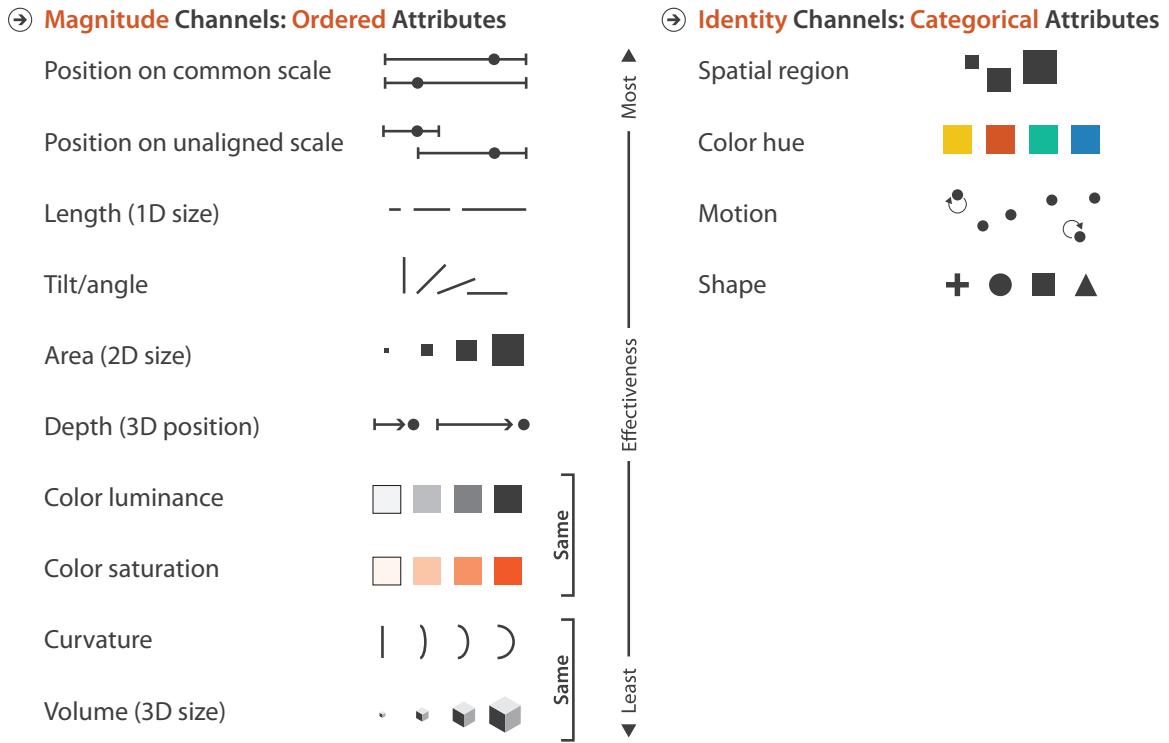


Figure 2.7: Channels ranked by effectiveness according to data and channel type. *Image source: [119].*

Color is a special channel that can be used for both data attributes. As shown in Figure 2.7, color luminance and saturation are used in magnitude channels, and color hue is used in identity channels. A colormap specifies a mapping between colors and data values, and designing an effective colormap is challenging. ColorBrewer [67] is an excellent source for colormap reference, providing color schemes for both categorical and ordered attributes. Human can only distinguish around 12 colors simultaneously [119]. Figure 2.8a shows such a categorical colormap with 12 distinguished color hues. Ordered colormaps can be either sequential (Figure 2.8b) or diverging (Figure 2.8c). Diverging colormaps use two different color hues to emphasize values below and above the middle point.

Gestalt Principles Gestalt principles describe how we see patterns in visual displays [85]. This section reviews three commonly used principles in representing groups of items.

Similarity Similar elements tend to be grouped together. Figure 2.9a shows a matrix of point marks with uniform spacing, but using two different shapes: dot and

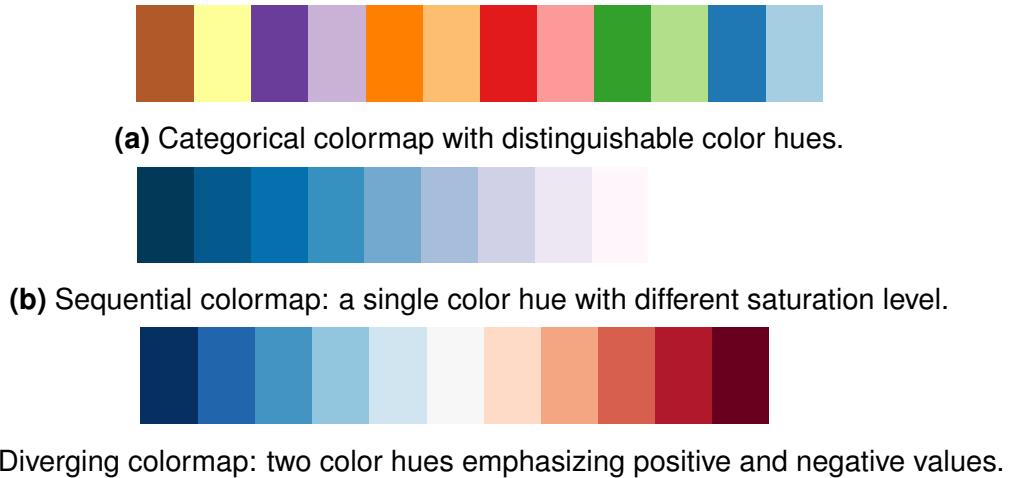


Figure 2.8: Colormaps from ColorBrewer. *Image source: [67].*

cross. The similarity of shapes helps us see the rows more clearly than the columns. Two separable channels can be applied together to reveal patterns by either rows or columns. In Figure 2.9b, green is used to depict rows, and texture is used to depict columns.

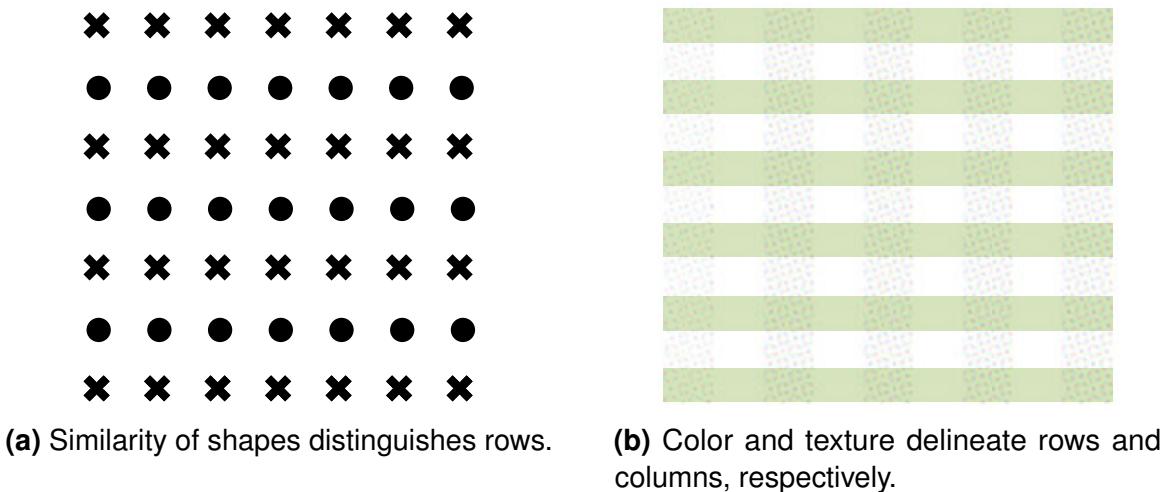


Figure 2.9: Similarity principle: similar elements are perceived as a group. *Image source: [185].*

Proximity Elements that are close together are perceptually grouped together. Figure 2.10a clearly shows two groups of dots. Figure 2.10b shows rows of dots. However, with a small change of spacing, these dots are perceived as columns in Figure 2.10c. The application of this principle is straightforward: organizing

related information close together. It helps separate groups of unrelated objects and facilitates searching for information.

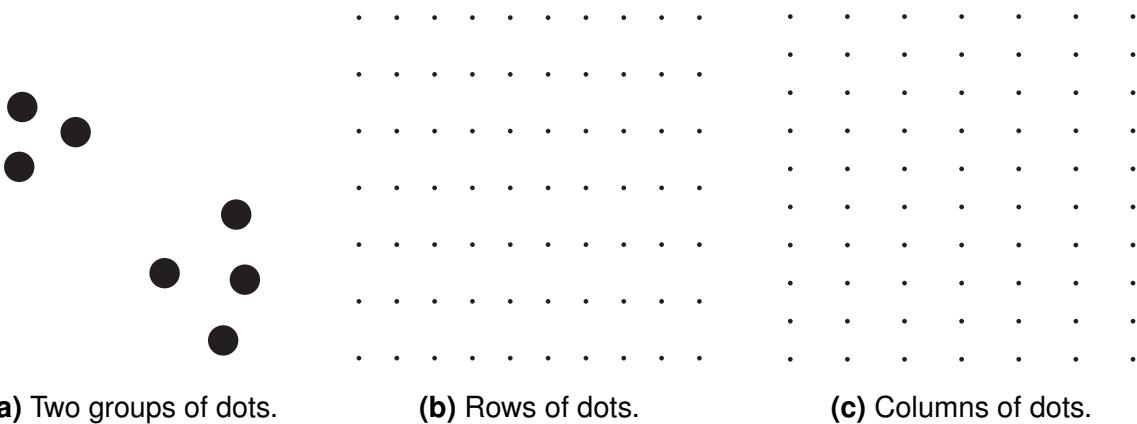


Figure 2.10: Spatial proximity principle: spatially close elements are perceived as a group.
Image source: [185].

Connectedness Elements that are connected by visual properties are perceived as being more related than elements that are not connected. This principle can be achieved simply by drawing a border around a group of elements as in Figure 2.11a. This is extensively applied in designing complex graphical user interface: groups of related features are separated by borders. Another approach to implement connectedness is by drawing lines between related elements as in Figure 2.11b. This is the basics of *node-link diagrams* – one of the most common methods of representing relationships between elements.

Among these three Gestalt principles of representing groups of elements, connectedness has the strongest effect, followed by proximity and then similarity. Figure 2.12 illustrates this comparison. In Figure 2.12a, even though spacing between dots in rows is shorter than spacing between dots in columns, the lines make the vertical links clearer than rows. In Figure 2.12b, the lines also make the horizontal links more notable than groups of colored circles. In Figure 2.12c, two spatial groups are more clearly perceived than colored groups.

Tufte's Principles Tufte proposes a number of principles for a well-designed graphic, documented in his series of books, most notably including *The Visual*

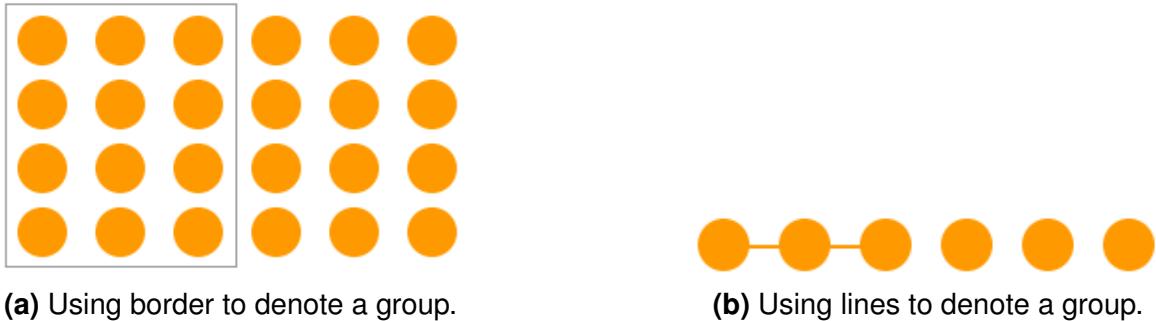


Figure 2.11: Connectedness principle: visually connected elements are perceived as a group. *Image source: [185].*

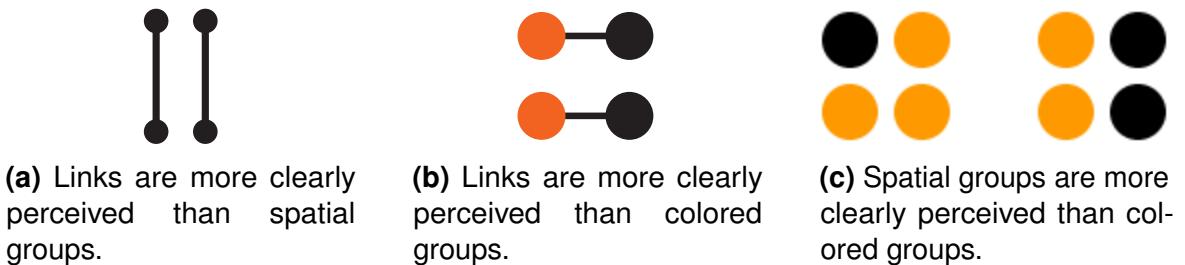


Figure 2.12: Comparison of Gestalt principles. Connectedness is stronger than proximity, and proximity is stronger than similarity. *Image source: [185].*

Display of Quantitative Information [176] and *Envisioning Information* [177]. This section reviews a few principles that have been commonly applied in graphic design and visualization.

Graphical Integrity This principle emphasizes that the graphical representation should tell the truth about the data. Representation of numbers, as physically measured on the surface of the graphic itself, must be directly proportional to the numerical quantities represented [176]. Figure 2.13a shows a falsely big drop in stock market value between 2001 and 2002. It because the chart uses a relative scale with the value range from 450 to 500, causing its height disproportional to the market value. Figure 2.13b corrects this error by using an absolute scale with the value range starting from 0.

Data-Ink Ratio Maximization Data-ink includes the pixels in the graphic that are used for representing the data. Data-ink ratio is defined as the ratio between the data-ink and the total non-background pixels used in the graphic. This principle aims to maximize this ratio by erasing non-data-ink and erasing redundant data-ink.

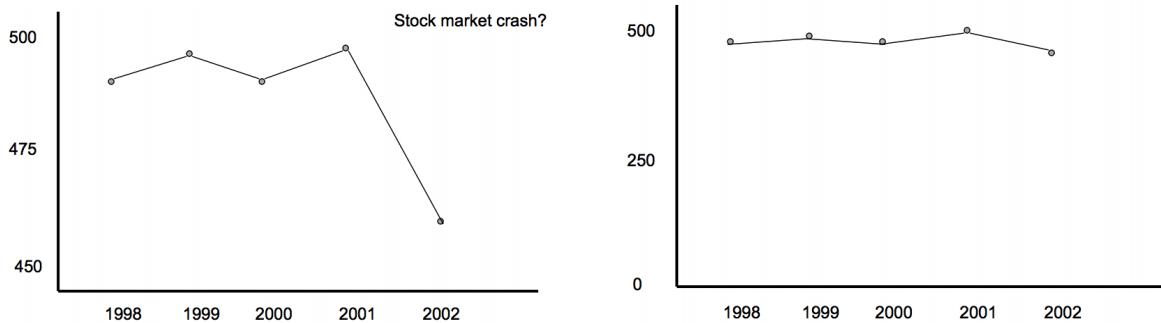


Figure 2.13: Graphical integrity principle. The chart should tell the truth about the data.

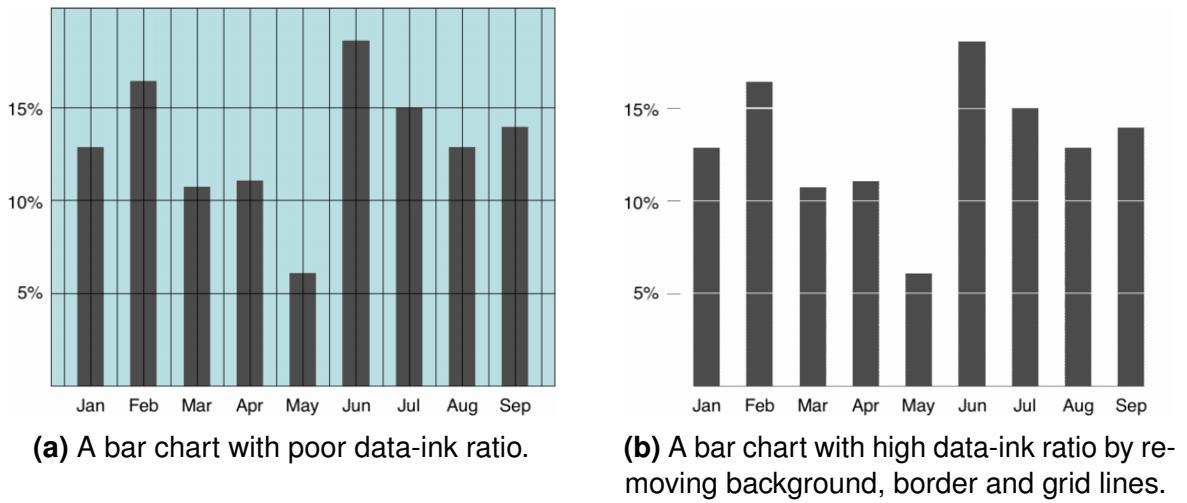


Figure 2.14: Data-ink ratio maximization principle: removing the graphic that does not contribute to the understanding of the data.

Micro/Macro Readings This principle suggests that a graphic can contain both enormous details and an overall pattern. This allows the viewer to glance from a distance to observe the big picture, and later drill-down closely to examine its individual pieces. Classic stem-and-leaf plot is a great example to illustrate this principle (Figure 2.15). The plot shows all individual data items at meaningful level of detail, and provides an understanding of the data distribution. The micro/macro principle is extensively applied in interactive visualization, where zooming and panning are made possible, such as Google Maps. Data items at different scales can be represented with different levels of detail to provide appropriate information based on the allowed display area.

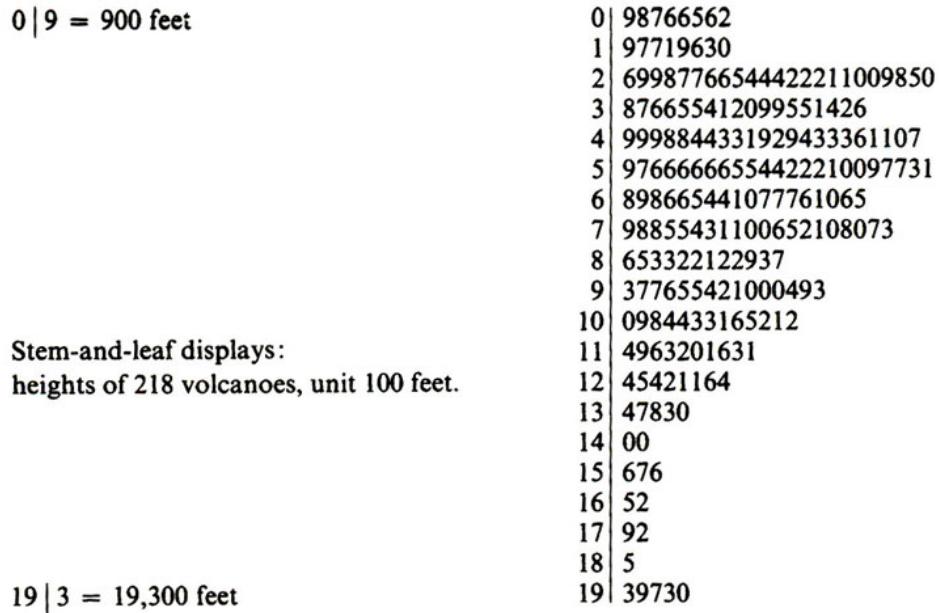


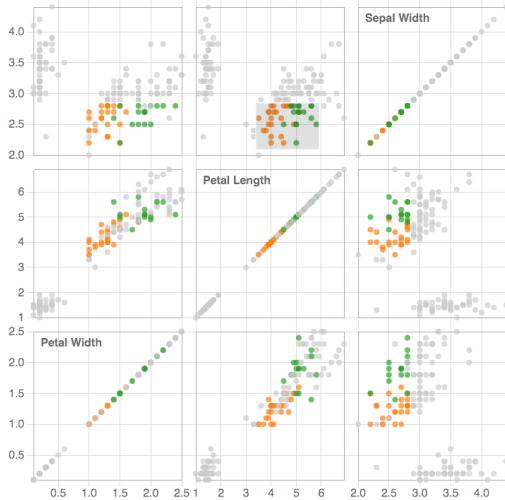
Figure 2.15: Micro/macro principle. A stem-and-leaf plot shows both the data distribution and individual items. *Image source: [176].*

2.2.3.4 Interaction Techniques

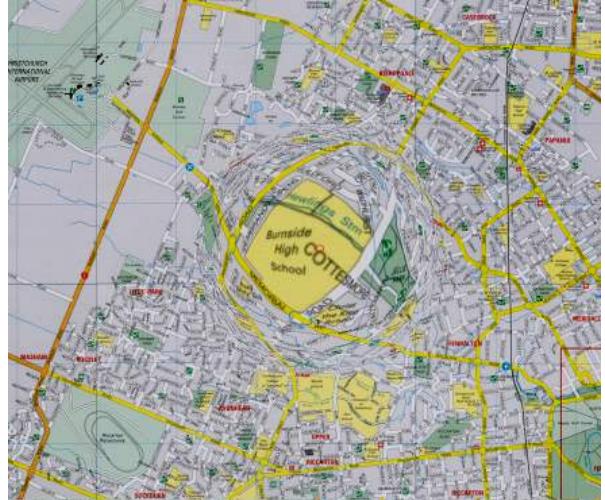
Interaction typically refers to the set of controls provided to the user to manipulate an interface [134]. A static visualization may only show one aspect of a dataset. When the dataset is large enough, showing all the data at once may also make the visualization become cluttered. Interaction plays an important role in solving these problems. It can help explore large datasets at multiple levels of detail, identify patterns through examination of different visual representations and understand the connections between them.

Examples of interaction include standard techniques used in graphical user interface such as mouse clicking and scrolling, and more visualization specific techniques such as *linking and brushing* [97], and *focus+context* [32]. Multiple views in a visualization are often linked together to exploit their strengths. The user can select points of interest using the brushing technique, typically done directly on the visual data representation such as dragging a rectangular area. The points are brushed in one view will be highlighted in other views, allowing the user to explore them with different perspectives and representations (Figure 2.16a). Focus+Context is a technique that brings both the overview (context) and the detailed information (focus) together in

one view. A fisheye view [52, 53] is one example of this technique: the focal region is magnified and displayed within its surrounding context (Figure 2.16b).



(a) Linking and brushing. Data points brushed in one view are linked and highlighted in other views.



(b) Fisheye view for focus+context. Both the overview and the detailed information are displayed in one view.

Figure 2.16: Examples of interaction techniques.

Many other interaction techniques can be found in the taxonomies by Dix and Ellis [43], by Keim [92], and by Wilkinson [190]. Very often, a user performs an interaction to achieve some goal, thus interaction techniques can also be classified based on their intents. Actually, different interaction techniques in different visualizations may serve the same purpose. For example, both drilling-down a treemap [158] and semantic zooming [132] aim to get more details. Taxonomies of high-level interaction can be found in the work by Yi et al. [198], by Heer and Shneiderman [74], and by Brehmer and Munzner [17]. These classifications could help visualization designers select suitable interaction techniques to serve for the capabilities they want to offer to the users.

Traditional graphical user interface widgets are often used to control different settings of a visualization, such as buttons and sliders. The disadvantage is that visual feedback does not appear where the interaction happens, but in some parts of the visualization. It also takes time for the user to search for the appropriate setting controllers. Direct manipulation [157] is an approach to address these problems. It enables the user to directly interact with the visual representation and receive immediate feedback. One example is using mouse scrolling to adjust zoom level while exploring a map instead of clicking on a button in the toolbar. Another

example is parallel coordinates plot with axes can be reordered by direct dragging and values can be filtered by direct brushing on the axes (Figure 2.17). Surrogate objects can be used when the data objects are small or distant [29]. An example is the use of interactive legends as filtering means [78]

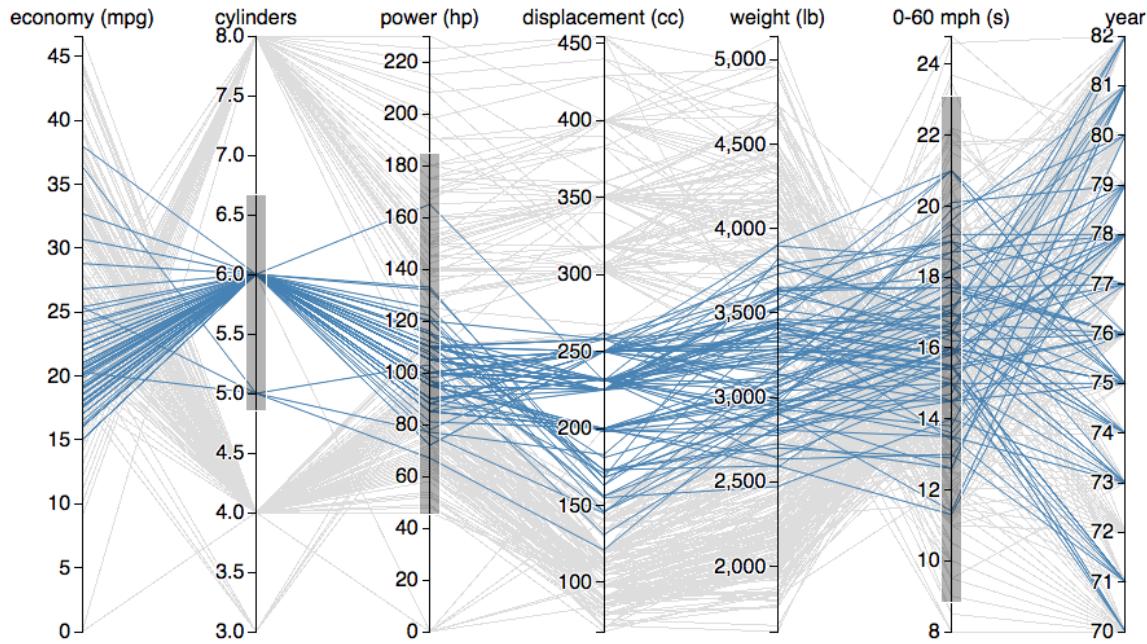


Figure 2.17: Direct manipulation in a parallel coordinates plot with reorderable and brushable axes.

Another concept that can be applied to improve existing interaction techniques is *fluid interaction* [48]. Besides using direct manipulation as discussed previously, the interaction should produce smooth animated transitions between the state before and the state after an interaction, helping users maintain their mental maps; and provide immediate visual feedback, allowing users to know what is happening and/or what will happen next.

Typically, interaction techniques are combined to explore the data or present a known story. A classic visual information-seeking mantra by Shneiderman [159] summarizes many design guidelines and interaction techniques for designing information visualizations: *Overview first, zoom and filter, then details-on-demand*. With large datasets, it is challenging to create an overview and provides cues for further exploration. A more suitable approach in this case is *Search, show context, expand on demand* [180].

2.2.3.5 Evaluation Methods

A visualization, no matter how novel and interesting it is, needs to be evaluated to check whether it meets the design goals and supports the target users to complete the intended tasks. Evaluation has been a research topic in visualization when the field becomes more matured [137]. Excellent reviews of visualization evaluation with different perspectives include evaluation techniques [26], scenarios [101] and design process [118].

In this section, we review the evaluation techniques based on the visualization design model by Munzner [118], helping address different concerns separately. The four levels include: explain the tasks and available data in the vocabulary of the problem domain, abstract them into domain-independent operations and data types, design visual encoding and interaction techniques to solve the abstract tasks, and develop algorithms to execute these techniques efficiently. Each level has its own *threats* to validity and methods to address them. Two types of methods are distinguished: *immediate* approaches can be done before inner levels are implemented, whereas *downstream* requires all inner levels are completed. The threats and evaluation methods are summarized in Figure 2.18.

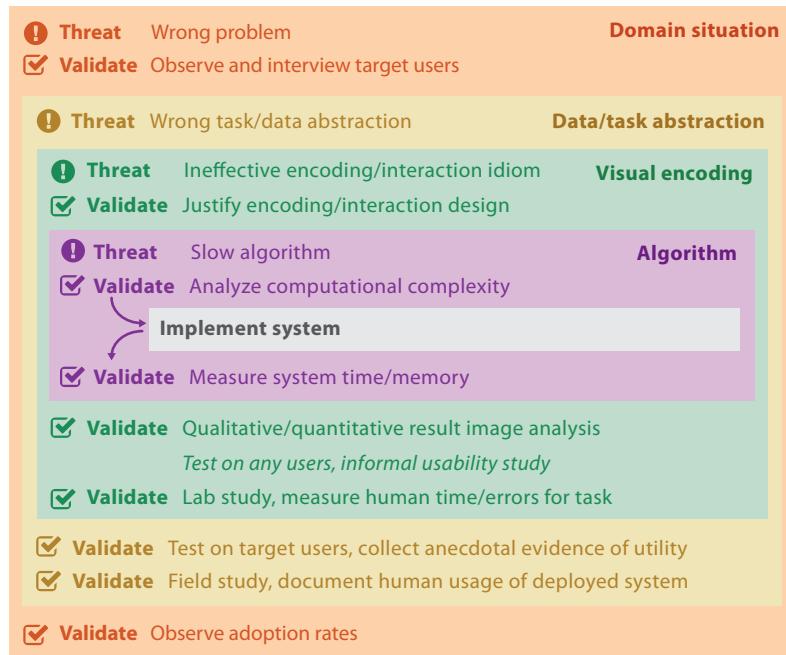


Figure 2.18: Threats and validation at each of the four nested levels of visualization design.
Image source: [119].

Domain Problem and Data Characterization The domain problem of target users is investigated to see if visualization is a potential solution. The primary threat is that the problem is mischaracterized: the users do not really suffer from the identified problem. An immediate form of validation is *field study* [26], where the investigator observes how target users act in real-world settings in order to learn and verify the characterization. Another technique is *contextual inquiry* [82], which allows the investigator to occasionally interview while the user is engaged in the process. One example is the study by Sedlmair et al. [156] on current working behavior and environments of analysis and diagnosis experts in the automotive industry.

One downstream form of validation is to report the rate the adoption rate of the tool by the target users. High effort is required to make the visualization solution reliable and deployable in the real-world environment. Examples include a field study of Google's Notebook product [151] and 6-week field trial of SparTag.us – a tagging system for foraging web content [83].

Operation and Data Type Abstraction The threat at this level is the identified data and task abstraction do not solve the characterized problem. Only downstream approaches can be used to validate the abstraction. The deployed system needs to be used by target users completing their routine tasks in real-world environment. The goal of this evaluation is to collect anecdotal evidence that the solution is in fact useful. The observation and interview need to focus on understanding how the tool is used, and how it helps or hinders the users in performing their tasks. An example is a longitudinal field study of LiveRAC system that supports analysis of system management time-series data [112].

Evaluating the visualizations for supporting sensemaking can be done at this level, as the *evaluating visual data analysis and reasoning* scenario in the taxonomy by Lam et al. [101]. Due to the nature of sensemaking, evaluation is often carried out as case studies [88] with observation and interview, and followed by qualitative data analysis [102]. Attempts also have been made to quantify the insight or knowledge gained during sensemaking [191].

Visual Encoding and Interaction Design At the design level, the threat is the chosen design is ineffective at communicating the desired abstraction to the user. One immediate form of validation is to justify every design decision based on known design principles such as the ones discussed in Section 2.2.3.3, or more

comprehensive predefined guidelines as in heuristic evaluation [201]. Asking experts to review the design prototype also provides valuable feedback [174].

A common downstream approach is to conduct a controlled experiment comparing the design with other state-of-the-art alternatives [197]. A number of participants, depending on the expected size of the experiment, carry out a number of tasks representing real-world cases. Typically, task completion time and accuracy are measured and analyzed using hypothesis testing methods [49]. Post-task interviews are often combined to establish deeper understanding about how the visualization is used. If the experiment can be completed online, crowd-sourcing approach using Amazon's Mechanical Turk service can help largely increase the size of participants [71]. Another downstream approach is the measurement of common aesthetic metrics such as the number of edge crossings and edge bends that have been used in graph visualization [166].

Algorithm Design The primary threat at this level is the algorithm is suboptimal in terms of time or memory performance. In interactive visualization, it is essential to ensure the interaction responsive in real-time. Analyzing the complexity of the algorithm using the standard approaches from the computer science literature [36] is an intermediate form of validation. The complexity can be computed based on the size of dataset or the display screen. Downstream approaches include measuring running time and memory usage for benchmark datasets.

2.3 Provenance

In the Oxford dictionary, *provenance* is defined as “the place of origin or earliest known history of something”. Provenance plays an important role in many aspects of our daily lives. For example, in *food* industry, before purchasing a bottle of fruit juice, it is useful for the customers to know about its origin, ingredients, methods of collecting, storing and processing fruits, etc. In the context of *art*, the provenance information of a painting such as authorship, material, painting time and the story behind the painting greatly decides its value. In computer systems, the provenance of a piece of data is defined as “the process that led to that piece of data” [117]. We broadly categorize provenance into two groups. The first group, *data provenance*, uses the computer systems definition of provenance including the source information of data and the process that produced it. The second group, *analytic provenance*, focuses

on visualization and analysis including the higher level reasoning involved in the analysis process and the interactive data exploration driven by sensemaking.

2.3.1 Data Provenance

Data provenance research has been taken in different fields, notably scientific workflows and databases. Scientific experiments may consist of thousands of steps, with each step involving distributed data sources and computational data models [57]. Workflows have been used to facilitate the assembly, automation and management of such experiments. Notable scientific workflow systems with provenance enabled include Tarvena [200], Kepler [16] and VisTrails [13]. Provenance plays an important role in scientific workflows, aiming to support data interpretation, reproduction of experiment results, troubleshooting and optimization [115]. The provenance of long and complex workflows is huge, thus pose challenges in storing, querying, and making sense of such data [39].

Curated databases are populated and updated with a great deal of human effort, typically published on the web [19]. A well-known example is Wikipedia – a free Internet encyclopedia that allows its users to edit almost any article accessible. Each record in these databases, such as a Wikipedia article, may be edited by many users and referred to other internal and external sources. This produces problems in attribution and provenance: who edited what at when. Research in database provenance can be characterized into a why-where-how framework [27]. *Why*-provenance focuses on the lineage of the output: for each tuple t in the output, the lineage of t is a set of tuples in the input data that helps produce t [38]. *How*-provenance concerns how the output tuple t is derived from the query [62]. Finally, *where*-provenance describes specific locations, or cells in relational databases, of the input data that contribute to the query output [20]. To compute these types of provenance, two general approaches have been introduced [19]. An *eager* approach adjusts the query to pass the extra provenance information to the output. Whereas, a *lazy* approach computes provenance on demand.

Data provenance research in scientific workflows and databases has mainly focused on closed systems, which have full access to the data and its provenance. Modern applications with service-oriented [130] and cloud-computing [23] architectures bring challenges in tracking and exchanging provenance information across systems. The *Open Provenance Model* is designed to address these challenges [117]. It also supports a digital representation of provenance for any objects, whether produced by computer systems or not. Three types of objects are defined in the

model for building this representation. An *artifact* is a state that can be a digital or physical object. A *process* is a series of actions performed on or caused by artifacts, and resulting in new artifacts. An *agent* acts as a catalyst of a process, managing its execution. Different types of causal relationships can be added between these nodes, forming a *provenance graph* as shown in Figure 2.19.

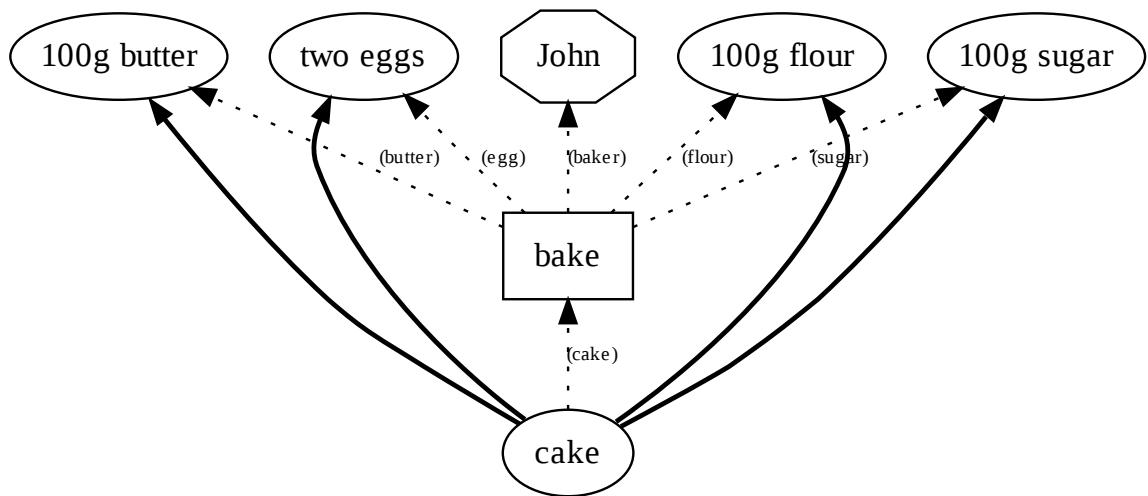


Figure 2.19: A provenance graph for “cake baking” using the Open Provenance Model. The cake (artifact) was baked (the process) by John (the agent) using ingredients including butter, eggs, flour and sugar (artifacts). *Image source: [117].*

The Open Provenance Model has been implemented in many systems, including notable scientific workflows such as Tarvena [200], Kepler [16] and VisTrails [13]. The model is general and can be extended in both the structure and vocabulary to represent domain-specific problems. *D-profile* [63] describes an extension of the model for representing provenance in distributed systems. ProveML [182] is another extension for recording the provenance of data, analytical process and interpretations in human terrain visual analytics.

2.3.2 Analytic Provenance

Analytic provenance focuses on understanding a user’s reasoning process through the study of their interactions with a visualization [127]. It captures both the interactive data exploration process and the accompanied human reasoning process during sensemaking [195]. In this section, we discuss different models for representing analytic provenance data and methods to capture the data.

2.3.2.1 Model

Analytic provenance information can be categorized using a four-layer hierarchical model based on its semantic richness, proposed by Gotz and Zhou [61]. Figure 2.20 illustrates this model with the level of semantics increasing from bottom to top. The bottom-level *events* consists of low-level user interactions such as mouse clicks and keystrokes, which have little semantic meaning. The next level up includes *actions*, which are analytic steps such as querying the database or changing the zooming level of data visualization. The parameters such as data description and visualization settings are also part of the provenance. Further up are the *sub-tasks*, which are the analyses required to achieve the sensemaking goal. Considering stock market analysis as the top-level *task*, examples of sub-tasks could be identifying top performing companies and determining long term trends.

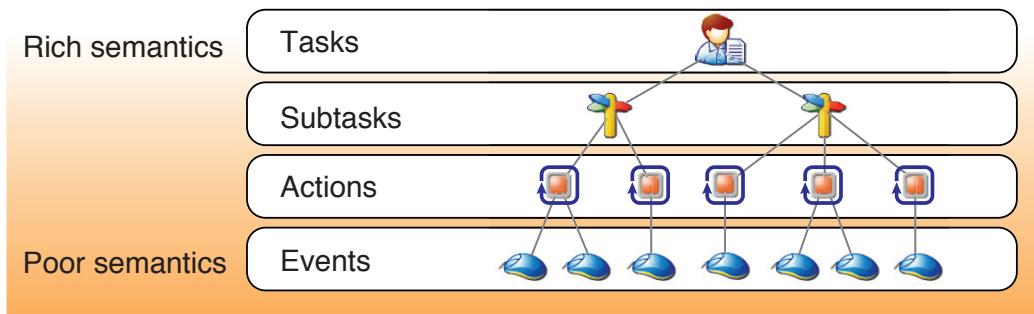


Figure 2.20: The hierarchical analytic provenance model with semantic richness increasing from bottom to top. The bottom layer includes *events* such as key presses and mouse clicks, which have little semantics. The next level up contains *actions* such as the database query and visualization zooming. Further up level consists of *sub-tasks*, which usually are the analyses performed during the sensemaking. The top level *tasks* are the overall sensemaking undertaking. *Image source: [61]*.

Analytic provenance is closely linked both within and across layers. Within a layer, analytic provenance is linked temporally (i.e., one event happens after another) and logically (e.g., one action depends on the two previous actions). A sequence of activities in each layer is performed to serve for a single activity in the next highest layer. For example, a database query action consists of several mouse click and key stroke events, and it is part of a higher level sub-task such as “comparing stock performance”.

This four-layer model is general, allowing the designers to determine the specific activities they want to capture within each layer for their systems. For example, many existing taxonomies of visualization interaction and tasks can be used for

the *action* and *sub-task* layers. Low-level analytic activities such as “retrieve value”, “sort” and “filter” [9, 61] can represent the *action* layer. A taxonomy of interaction techniques based on their intent such as “show me more related items” may stay in-between the *action* layer and *sub-task* layer. A multi-level typology of abstract visualization tasks by Brehmer and Munzner [17] distinguishes why and how a task is performed, as well as what the task inputs and outputs are. The *how* part of this typology is a good candidate for representing the *action* layer, and the *why* part may be suitable for the *sub-task* layer.

A recent taxonomy by Ragan et al. [143] characterizes provenance in visualization and analysis systems based on the type of provenance and the purpose of collecting it. Five “flat” provenance types include *data*, *visualization*, *interaction*, *insight* and *rationale*. The type of data provenance is similar to what we describe previously in Section 2.3.1. Visualization provenance concerns with the history of graphical views and visualization states, and interaction provenance focus on the history of user actions within a system. The combination of these two types of provenance is equivalent to the *action* layer. Insight provenance describes the findings and rationale provenance explains the reasoning process that led to these findings. These two types of provenance are good candidates for the *sub-task* layer.

2.3.2.2 Capture

Analytic provenance capture provides the data for its visualization. We describe capturing methods for each of the four layers in the Gotz and Zhou’s model [61].

Events There is limited literature on capturing events because it is relatively easy and provides little semantics alone. Among these, Glass Box [37] can record a wide range of low-level events such as mouse clicks, key strokes and window events. Its objective is to capture and archive intelligence analysis activities so they can be retrieved later. Simply capturing these events alone is insufficient to understand their purpose and rationale. For example, we know that a *mouse click* is captured; however, what the purpose of that click was (e.g., to sort the data), and why the user performed that click (e.g., to find an interesting pattern from the data) are unknown. Commonly, when analyzing data with a visualization, an analyst needs to perform many operations with trials and errors to find the answer to the problem. In that case, a series of poor-semantic events makes it more difficult for the analyst to recall what has been done. Therefore, more meaningful activities also need to be captured.

Actions During the course of analysis with a visualization, all user interactions can be systematically recorded. The visual exploration process can be modeled using a *graph* metaphor. Nodes in the graph represent *states* of the visualization and edges represent *actions* that transform one state into another state. A state includes all the information required to reconstruct the captured visualization. For example, a state of a *scatter plot* may include the dataset, attributes mapped to spatial positions, color and size. An action while interacting with a scatter plot could be changing attribute mapping of size. Commonly, *undo/redo* features are provided to allow revisiting to previous states. If a new action is performed from a past state, a new branch will be created to store that new line of operations.

Two common strategies can be applied to automatically capture the exploration process. The first strategy is to capture the initial state and all the actions so that they can be executed to reproduce any states [87]. This allows reapplying the analysis process with a different dataset, but could be time-inefficient if the number of actions is high. The second strategy is simply to capture all visualization states after each action [13]. This is easier to implement, but may be memory-expensive if a state contain too much information.

In the context of everyday, online sensemaking, users interact with a standard web browser instead of a visualization system. Visited web pages are automatically captured including visit time, web titles, URLs and favorite icons. Linking relationship between pages such as opening from a web link and using the browser's back button can also be captured [12, 80, 116]. This results in a hierarchical history rather than a linear list of visited web pages in the standard history feature. Manual capture using bookmarks is also a standard feature of web browsers. It allows users to save web pages for revisit purpose. Besides bookmarking a whole web page through its URL, a page element such as *table* and *form* HTML tags [83], and a specific fragment of text [44] can also be bookmarked. These finer-grained captures allow users to record what they want with higher accuracy.

Sub-Tasks and Tasks Tasks and sub-tasks provide important clues to the purpose and rationale underlying the sensemaking. They describe an abstract summary of the task-solving process. However, they are largely part of users' thinking, which a visual analytics system does not have direct access to. Also, the time window to capture such information is very limited. Even the users themselves may forget what they were doing after a while, making it difficult to recover the analytic provenance

information. Therefore, capturing high-level tasks and sub-tasks is one of the biggest challenges in analytic provenance capture.

Existing approaches to capture high-level analytic provenance can be broadly categorized into *manual* and *automatic* methods. The manual methods largely rely on users recording their analysis processes and sensemaking tasks, whereas the automatic methods try to infer the higher level tasks and sub-tasks from lower level events and actions. Even though the manual approaches are usually more accurate, they can distract users from the actual analysis tasks, which may discourage users from recording analytic provenance. Alternatively, the automatic approaches do not introduce interruption to the sensemaking process, but their capability of inferring semantic-rich analytic provenance information is limited [61]. Individual differences also introduce additional challenges to designing a robust algorithm for the inferring process. Users' knowledge and experience have a considerable impact on the way they conduct analyses. As a result, the sensemaking process (i.e. the analytic provenance) can vary significantly from users to users, even with the same dataset and analysis task.

Manual User annotation is one of the most common forms in manual capture. Users create *notes* or *annotations* that are associated with certain data, analysis result, and/or visualization [75, 182]. The content of a “note” is not limited to findings or discoveries; it can also include the thinking that leads to a finding or the relationships between findings. *Data-aware annotation* [70] links a finding and the associated visualization to the underlying data used to produce it, which makes it possible to apply new analysis and visual mapping at a later stage if further investigation is needed.

Even though an individual note only represents a fraction of the analytic provenance, it is possible to provide a reasonably good overview of the sensemaking process if a number of notes and the connections between them are captured. For example, the *Scalable Reasoning System* [134] allows users to record their discoveries of interesting patterns about the data, then specify their causal relationships, and generate a hypothesis based on these artifacts. However, this is only useful when users are willing to take notes, which can be perceived as distractions sometimes. Two common strategies to alleviate this include minimizing interruption/cognitive effort [83] and providing immediate benefits to the sensemaking task such as planning exploratory analysis for complex tasks [107]. However, currently there is a lack of general design guidelines for how to achieve them, and there are few user studies

evaluating how effective they are, in terms of both the benefits they bring and the potential cognitive cost they can introduce.

Automatic One of the main disadvantages of manual capture is the requirement of direct input from users. Automatic approaches try to address this by inferring higher level analytic provenance from what can be automatically captured. As discussed earlier, it is easier to capture analytic provenance at the event and action level. Therefore, most automatic approaches try to infer sub-task and task-level information from event and action provenance.

This turns out to be a difficult challenge. An experiment studied how much of a user's reasoning process can be recovered from user action information [45]. A domain-specific sensemaking task was used and experts were recruited to analyze the user action log. Higher-level analytic provenance manually inferred from the interaction logs were compared with the ground truth obtained through interview. The results showed that 79 percent of the findings, 60 percent of the methods, and 60 percent of the strategies were correctly recovered. The accuracy is not high even in such a constrained setting with domain experts doing the inference. Given the diversity of data and analysis involved in the sensemaking and the difficult of replicating expert knowledge/thinking in a computer system, the chance of having a generic technique that can accurately infer semantic-rich analytic provenance information for a variety of analysis tasks is not high.

Instead, existing methods either constrain the problem/analysis domain or aim for less semantically rich analytic provenance. By limiting the choice of data and analysis/visualization, an inference algorithm has better chance to make the right guess. However, even within a specific domain (such as finance), the types of data and analyses involved are still of very large amount. Also, being limiting on the data and analysis can constrain the system capability, having a negative impact on the sensemaking task.

Given the difficulty of inferring task/sub-task information, a few methods target less semantic-rich provenance. One such example is *action chunking*, i.e., identify a group of actions that are likely to part of the same sub-task, without knowing what the sub-task is. Such approaches apply heuristics to infer patterns from action logs based on repeated occurrence and proximity in data/visualization space or analysis time [61]. Such chunking information can be useful in several ways. For example, the system can prompt user to take a note if such an action usually occurs within a specific sequence. Also, the grouping information can be used for aggregation when

large amount of provenance information is to be visualized. This method is later extended to monitor user behavior for implicit signals of user intent and uses the information to suggest alternative visualization [61]. It is an open research problem to explore similar analytic provenance that can be effectively inferred and provides semantic information that can be used for supporting sensemaking.

2.4 Visualization of Provenance Data

We characterize the visualization of provenance data based on the level of semantics involved in the collected data [61], including event, action, sub-task and task. Because low-level events contain little semantics, they are often visualized and analyzed after users finished their tasks in order to gain deeper understanding about their processes rather than to provide near real-time sensemaking support to the users during their tasks. For example, visualization of users' mouse clicks can reveal patterns of application usage [111] and highlight some important usability issues, such as pages where users spent a lot of time and pages where they got lost during the task [186]. User interactions with visual analytics systems can be visually examined to recover their reasoning processes employed in their analysis tasks such as specific findings they found and strategies they used [45, 65]. Interaction logs have also been used to predict user performance of basic visualization tasks like visual search [18].

Next, we focus on the visualization of levels with richer semantics. Actions and states (the visualization results of the actions) are commonly used to show the analysis process. Sub-tasks and tasks are often documented by users in a graphical reasoning process.

2.4.1 Visualizing the Analysis Process

2.4.1.1 Visual Representation

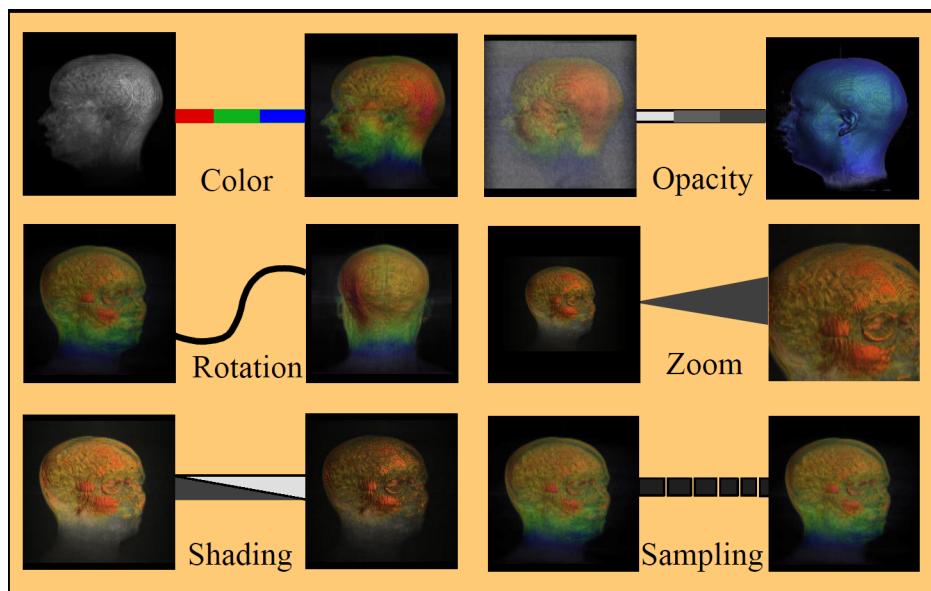
Text Text is commonly used to describe actions or states, such as names of actions, titles of visited web pages and content of user notes. Text can provide accurate information, but long text makes it difficult for users to understand and recognize. A graphical browser history by Ayers and Stasko [12] shortens web page titles to accommodate more pages in the view. Within a given width, its algorithm preserves complete words at both ends of the title and trims characters in the middle if necessary. Kaasten, Greenberg and Edwards [86] compare the recognizability of

titles between various string sizes for all three truncation methods (text is truncated at the beginning, middle or end of the title). The results show that for medium (60%) recognition, we need 15–20 letters (depending on the truncation method) for web sites, and 28–39 letters for exact pages. For longer text such as user notes, machine learning techniques for text summarization could be beneficial [121].

Icon Sensemaking actions can be represented by graphical symbols, allowing users to easily distinguish them. They can be used alone to represent the analysis process when the visual result of each action is out of interest (Figure 2.21a). Alternatively, these icons can be used together with interface states, connecting the one before and the one after an action (Figure 2.21b).



(a) A sequence of icons displaying the analysis process. *Image source: [61].*



(b) Iconic actions connecting two visualization states. *Image source: [108].*

Figure 2.21: Using icons for representing actions.

Thumbnail Thumbnails are commonly used to represent visualization states, aiding users' recognition of previous ones. One study suggests that a thumbnail size of 96 pixels square could provide 60% accurate recognition of a visited web site [86]. For the same accuracy but in recognizing an exact web page, the thumbnail size rises to 144 pixels square. Additional information can be added to a web thumbnail

to improve its recognition such as how often the represented page is revisited and whether that page is bookmarked or not [31]. For visualization thumbnails, visual encodings and parameters that were used to produce the visualization can also be embedded (Figure 2.22), providing more provenance information to the users.

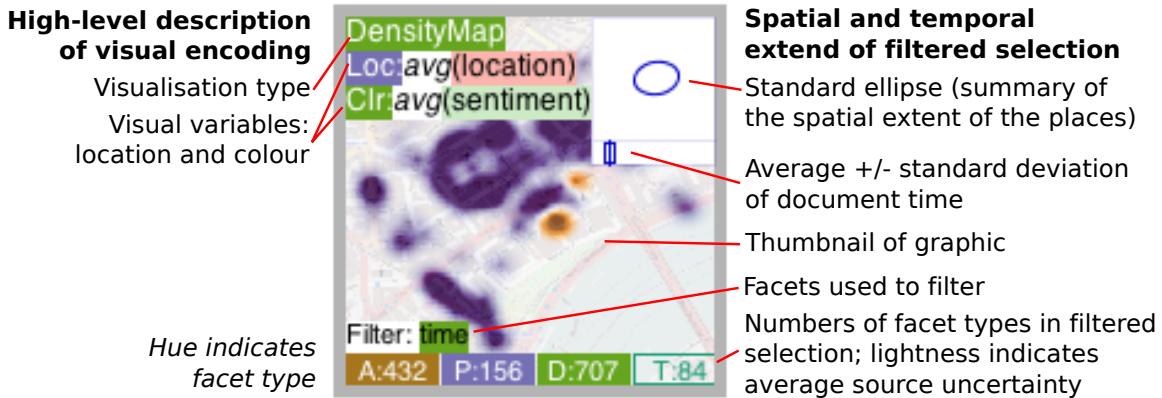


Figure 2.22: Visualization thumbnails with additional information about any filtering used, the characteristics of the filtered subset of data and the visual encoding. *Image source: [182].*

It may be necessary to apply pre- and post-processing adjustments to make the visualization thumbnails more recognizable [73]. For example, high-frequency visual elements that are not helpful in a small size such as gridlines and element borders can be removed to prevent their domination in the resulting thumbnail. As a result of down-sampling techniques, colors of data items may be different from them in the original visualization. Therefore, readjusting the color to match its intended value could help users to recognize the visualizations they analyzed in the past.

The default snapshot may also be an imperfect representation of a web page, especially if it contains a lot of text. Teevan et al. [170] propose an automatic method to produce a thumbnail improving its recognition. It consists of three components: some salient text at the top-left hand corner, a salient image below the text, and a watermarked logo superimposed at the bottom left hand corner of the image. The salient text contains about 20 first characters of the web page title. The salient image and the branding logo are chosen from the page using machine learning. Figure 2.23 shows such a thumbnail.

Besides recognizing previous states, seeing the difference between a state and the one before it is also important in understanding the analysis process. One approach is to highlight the difference between two consecutive states (Figure 2.24a). The changes may only happen at one small portion of the entire interface. Therefore,



Figure 2.23: An enhanced web thumbnail (right) as a composite of some salient text, a salient image and a branding logo. *Image source: [170].*

showing only that area in both states could help users quickly identify the difference (Figure 2.24b).



(a) Modified items are highlighted with green borders. *Image source: [96].*

(b) Changed area is cropped and shown in both states. *Image source: [100].*

Figure 2.24: Techniques improving recognition of state changes.

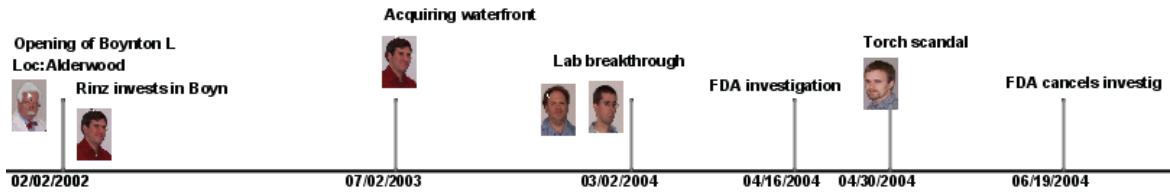
2.4.1.2 Layout

Linear Layout Provenance data usually contains an inherent *time* attribute, recording when an action happened. An approach that emphasizes on the order of completed actions is to show them as a linear sequence of items like a comic strip [100, 113]. This layout facilitates visual scanning of past actions, allowing users to quickly understand the analysis process. Figure 2.25 shows such a layout.

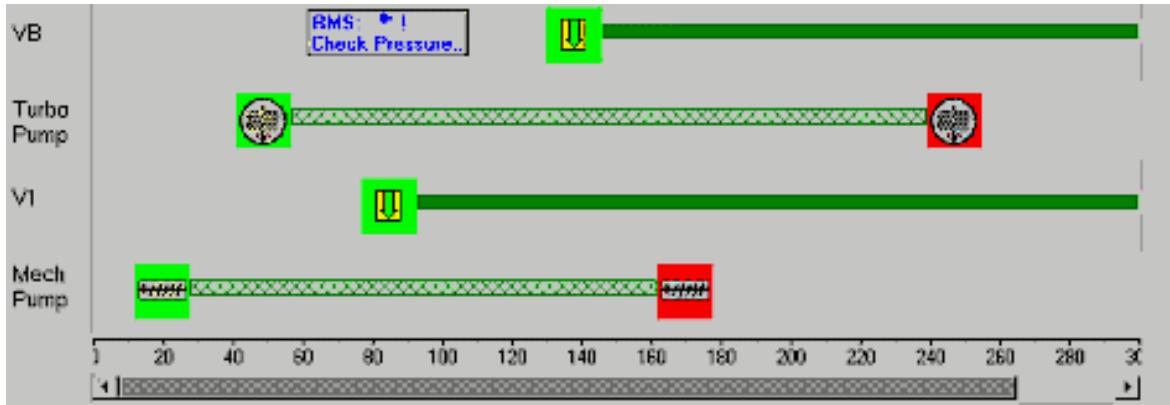


Figure 2.25: Comic strip layout. A sequence of past actions in chronological order. *Image source: [73].*

If the absolute timestamps of actions are of interest, a continuous timeline [40] is a more suitable layout. Actions are positioned along the time axis at when they happen (Figure 2.26a) or during which they last (Figure 2.26b). The time axis can be either horizontal or vertical [163]. The layout algorithms found in these provenance timelines are quite naive. POLESTAR [135] and the timeline from Jigsaw [105] require manual rearrangement of the events from users to solve the overlapping problem. The timeline from nSpace2 Sandbox [163] shows events at the exact time when they happen without considering possible intersection between of them.



(a) Time-point provenance data. User annotations are positioned along a time axis at when their associated events happen. *Image source: [60].*



(b) Interval provenance data. User actions are shown as horizontal bars along a time axis covering their durations. *Image source: [140].*

Figure 2.26: Timeline layout for visualizing the analysis process.

Another approach is to use both the horizontal and vertical axes to represent time. BrowseLine [81] uses the vertical axis for *macro-time* and the horizontal axis

for *micro-time*, similar to stem-and-leaf plots. More specifically, a two-dimensional timeline (Figure 2.27) is divided into rows, each for a big time-slot, such as one hour. In each row, events happening within that hour are positioned along the horizontal axis without considering their absolute timestamps. This design assumes that users may only remember roughly when events happen, thus absolute positioning in the vertical axis facilitate them in recognizing past events. Moreover, relative positioning in the horizontal axis could help pack more events and prevent overlapping.

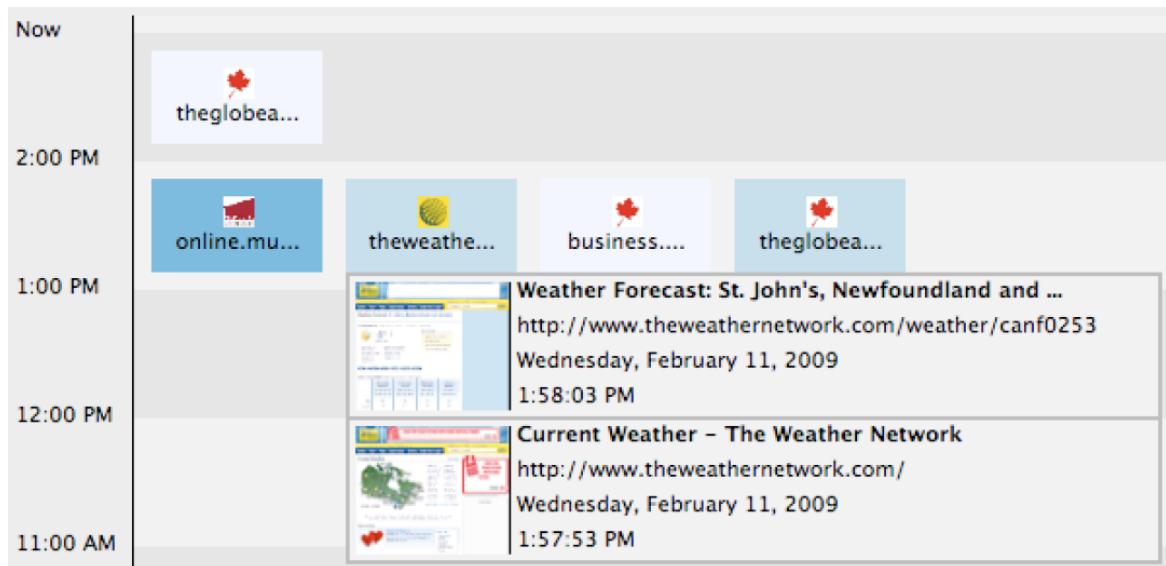


Figure 2.27: 2D timeline. The vertical axis represents macro-time, whereas the horizontal axis represents micro-time. Events within a macro time-slot are positioned in chronological order as a comic strip. *Image source: [81].*

Branching Layout Many sensemaking systems allow users to revisit their past states such as through undo/redo features or backward/forward in web browsers. From a past state, if the user performs a new action, it should be recorded in a new branch forking from that state. This branching history is typically visualized with a tree layout to represent the logic of the analysis process effectively. In such a tree, nodes represent a summary of system states, and edges represent actions that transition the system from one state to another. Examples can be found in provenance-enabled systems in different fields including scientific visualization [108], information visualization [46], visual analytics [87] and browser history [12]. Figure 2.28 shows such a provenance tree.

In a tree layout, the order of actions can be inferred through the direction of edges. Moreover, exact time gap between actions can also be visually encoded into

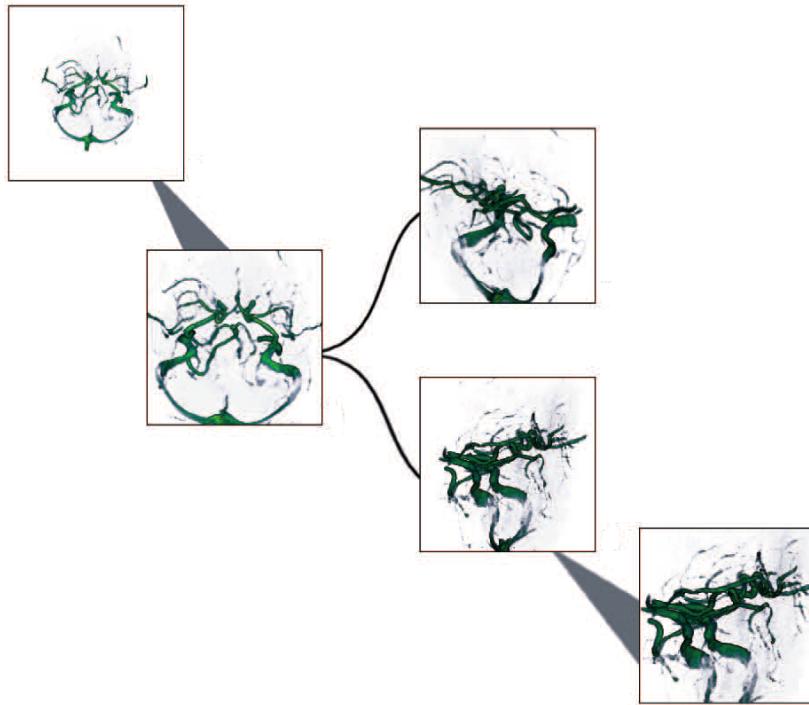


Figure 2.28: Tree visualization for branching analysis process. Nodes are thumbnails of past visualization states and links are transforming actions. *Image source: [84].*

the visualization. VisTrails [13] color-codes the background of nodes according to their creation time (Figure 2.29a). Aruvi [160] uses the length of edges to represent the relative time gap between two states (Figure 2.29b).

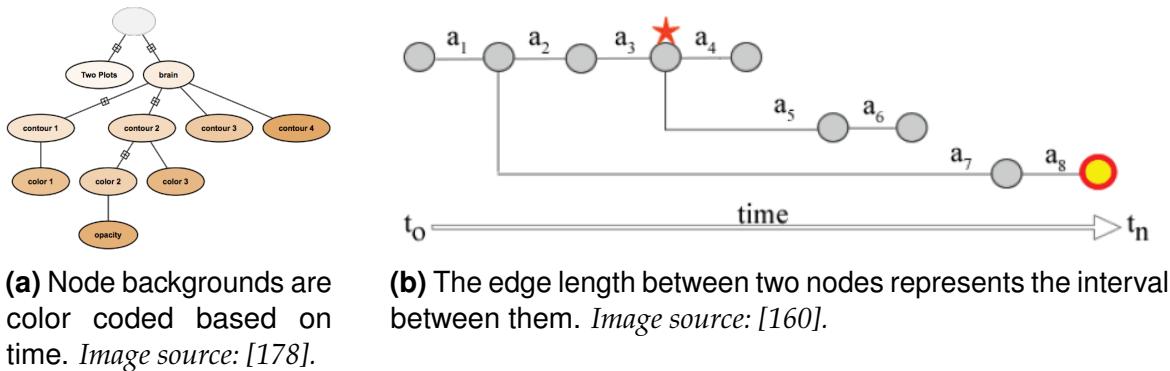


Figure 2.29: Encoding temporal information into provenance trees.

The diagonal arrangement of tree as in Figure 2.52 may consume a lot of space. One approach is to use only horizontal and vertical edges as in Figure 2.29b. Another approach is to display only the path that led to the currently active visualization [96].

Other paths can be expanded on demand. Figure 2.30b shows examples of representations of the active path and the full history.

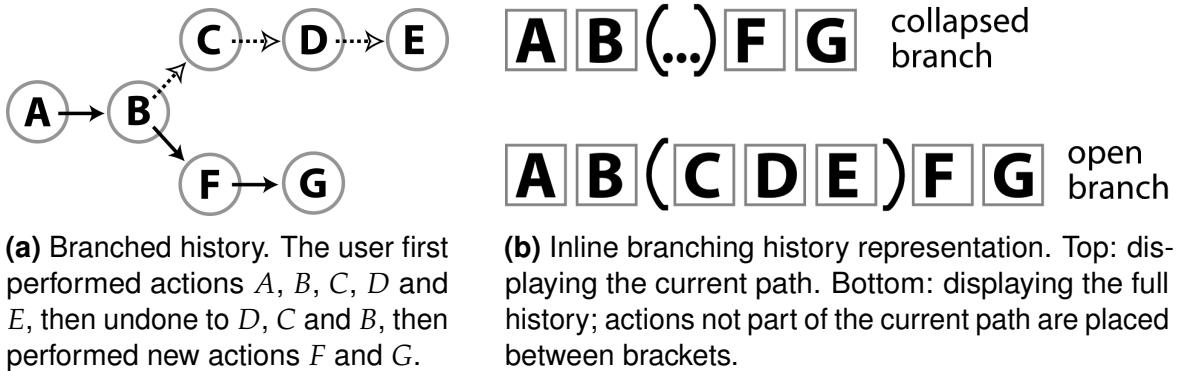
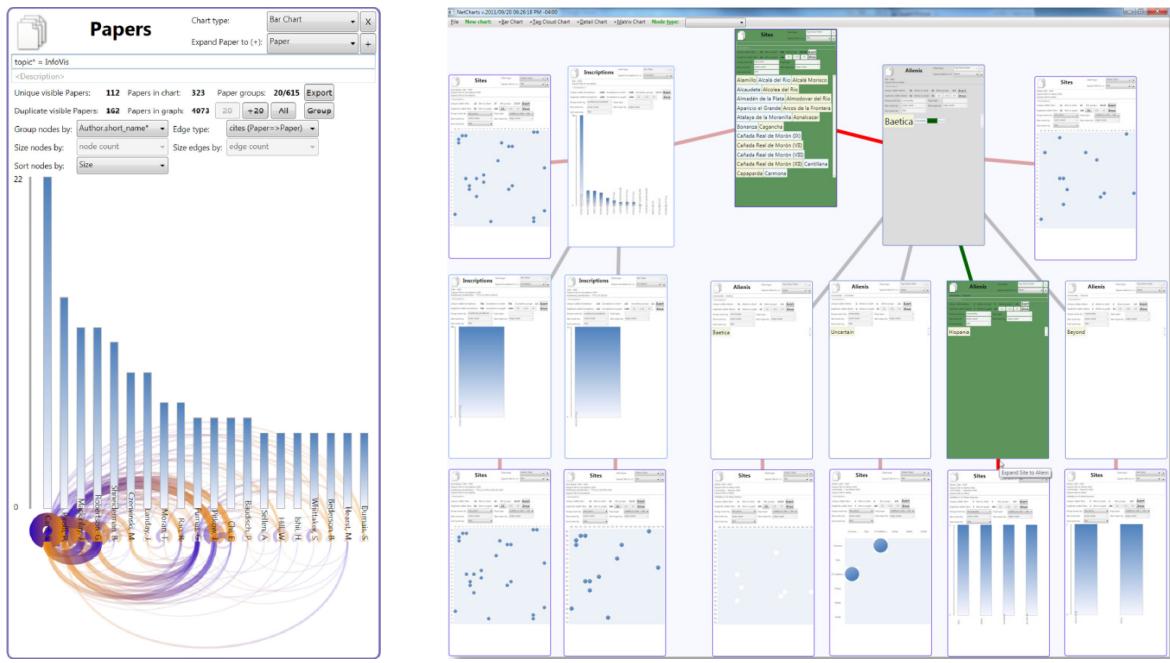


Figure 2.30: Branching layout for tree visualization of the analysis process. *Image source: [96].*

Interaction is also commonly used to address the scalability issue. Zooming and panning allow users to see the overview of the analysis process and navigate to the part of interest [46]. Collapsing and expanding branches of the tree on demand can also help reduce its size and avoid distraction [13]. Distortion techniques may help users to quickly navigate and focus on more relevant states and actions [113]. Another technique for compressing the provenance tree and improving user understanding is *action chunking* [73]: a sequence of related actions may be better represented as a single higher-level action. For example, a quick succession of sort or filter actions likely indicate a multi-step configuration of the view and can be chunked together.

Typically, provenance data is shown in a separate view, linked with other data views of the visualization system [73, 87, 133, 160]. However the system can also be used as a single item of the provenance view. For instance, in GraphTrail [46] – a single-view visualization system, when the visualization is changed (e.g., changing attribute mapping in a bar chart), it creates a new view containing that visualization and links with the current view. This is similar to how normally the provenance view is developed; however, GraphTrail makes the entire exploration space become the provenance view. Moreover, each history item is not a thumbnail, but a fully interactive visualization. By allowing zooming and panning, users can choose between a close examination of individual system states (Figure 2.31a) and an overview of the analysis structure (Figure 2.31b). An extra overview window as in overview-and-detail technique [32] could be useful in search and navigation in a large space.



(a) Zooming in to work on the active visualization.

(b) Zooming out to see an overview as the history of the exploration process.

Figure 2.31: Unification of the exploration view and the history view. *Image source: [46].*

2.4.2 Visualizing the Reasoning Process

The reasoning process reflects user findings, methods and strategies in performing a task. This process mainly happens in the user's mind, and some sensemaking systems allow the user to manually externalize it through user bookmark [182], annotation [75], and manipulation of those externalized items [196]. A common form of reasoning externalization is to allow the user to write down their thinking. It could be a free note [160] or a description of a user bookmarked visualization [182]. Alternatively, users can annotate directly on the visual bookmark using standard graphical editing features such as circling on interesting patterns (Figure 2.32a – Top) or hand drawing on a suspicious trend (Figure 2.32a – Bottom). To make the annotation meaningful across multiple views, the selection should be aware of the data involved in it [70]. For example, in Figure 2.32b, the annotation in the top view also makes the data items in the bottom view get annotated using the same selection query. Data-aware annotations allow users to see the data of interest remained across different views.

A note can simply be any user thought, but it can also take on different roles such as *evidence*, *assumption* and *hypothesis* [133]. They have different visual repre-

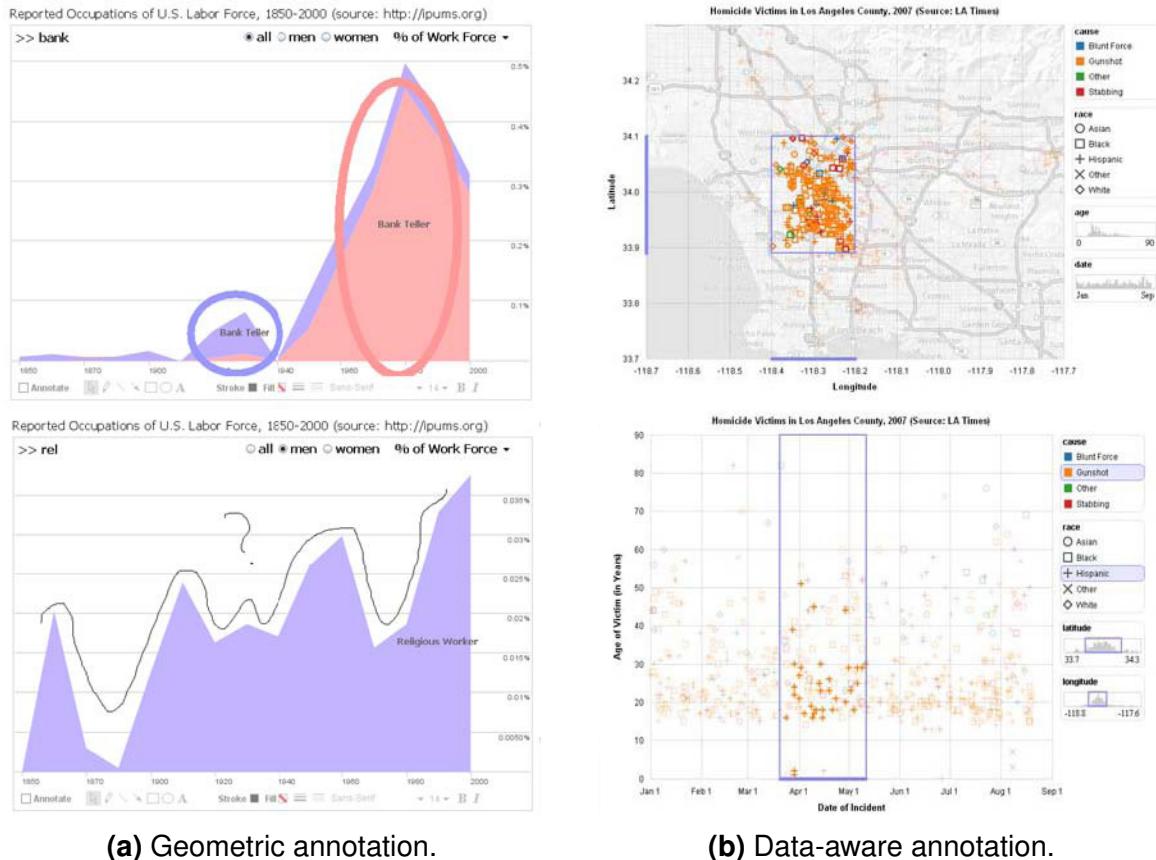
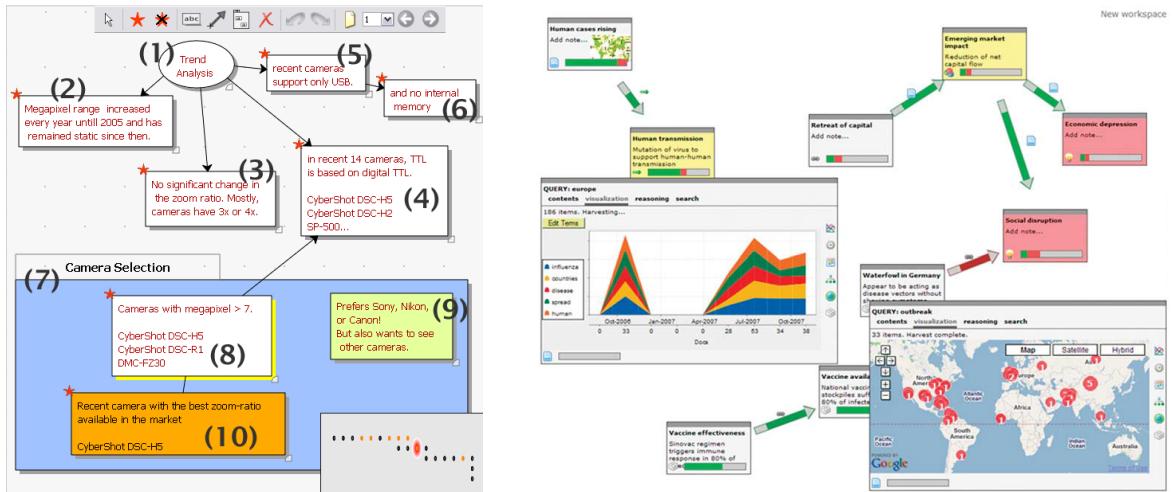


Figure 2.32: User annotation of bookmarked visualization. *Image source: [74].*

sentations, enabling users to create and manage their complex reasoning processes. Typically, users are allowed to move notes freely and draw edges to connect them, enabling spatial grouping of related notes and relationships establishment. These interactive features are known to help users produce more critical thinking in their analyses [155]. For example, users can show connections between a hypothesis and its evidence by drawing links, and spatially separate the evidence into a supporting group and a counter group, facilitating further comparison. Figure 2.33a shows such a diagram produced with user notes and Figure 2.33b shows such a more formal diagram with different reasoning roles.

More analytical reasoning methods have also been applied on the provenance data. The SRS system [133] first allows the user to construct a reasoning diagram with bookmarked visualizations as evidence nodes and causal relationships as links (Figure 2.33b). These evidence nodes and links may lead to a hypothesis. The user then specifies their confidence about the evidence before the system computes the



(a) A diagram of user notes showing their thoughts. *Image source:* [160].

(b) A formal reasoning diagram with nodes having different roles: evidence, causal relationship, assumption and hypothesis. *Image source:* [134].

Figure 2.33: Visualization of reasoning process.

likelihood of the hypothesis using the Dempster–Shafer belief network [154]. The result provides analytical support to the user.

POLESTAR [135] supports argument structuring, based on methods for analyzing legal documents such as the Toulmin model of argument [175]. To help validate a hypothesis, it organizes arguments as a tree structure of claims, each supported by at least one piece of evidence. A claim can have its supporting sub-claims and can also have rebuttals that weaken or restrict their force. Figure 2.34a shows such a argument tree. Sandbox [194] supports analysis of competing hypotheses – a judgment method that requires careful weighing of alternative explanations [79]. It allows users to add multiple hypotheses, each with supporting or contradictory evidence. These pieces evidence are weighted by the users and summarized in a visual matrix, enabling the users to effectively make a decision without having to remember and analyze in their minds. Figure 2.34b shows an example of this support.

2.5 Visualization of Time-Oriented Data

Time is an essential aspect of life; everything contains inherent temporal attributes such as the time when a person was born and the time when an event happens. Long before computers were invented, information graphics have been used to represent



Figure 2.34: Analytical support for the reasoning process.

temporal relationship of data. One of the oldest documented timelines was created back in 1765 entitled *Chart of Biography* by Joseph Priestley (Figure 2.35). It shows the lifespans of two thousand famous names along a horizontal time axis, spanning from 1200 BC to 1800 AD. He uses a horizontal line segment to depict a lifespan, and adds dots to either ends to indicate the uncertainty of the reported values.

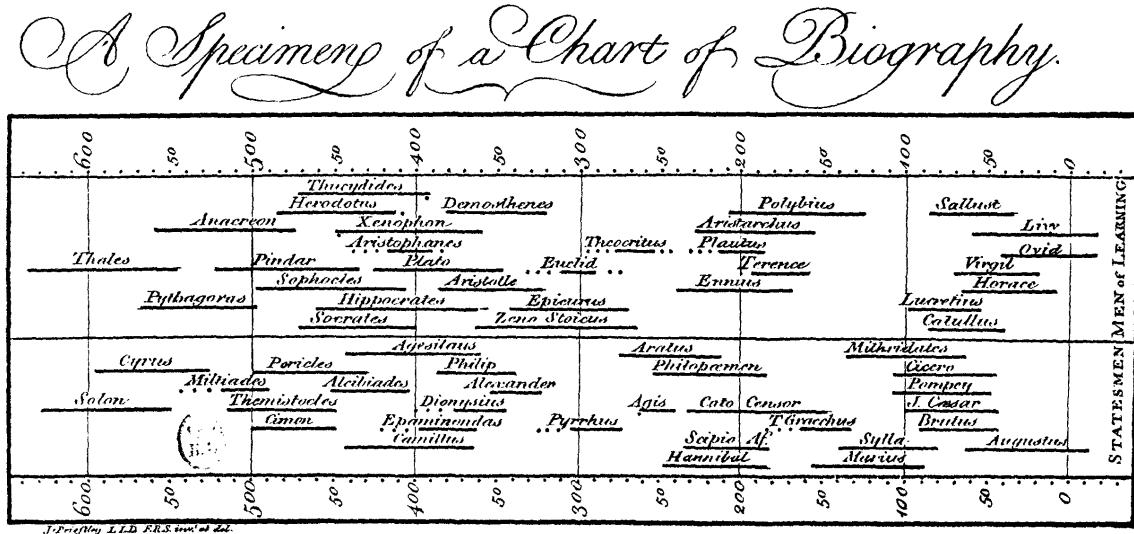


Figure 2.35: Joseph Priestley's Chart of Biography portraying the lifespans of famous historical persons. *Image source: [141].*

Since then, many visualization techniques have been developed to effectively reveal the temporal relationship of data. The book by Aigner et al. [6] provides a

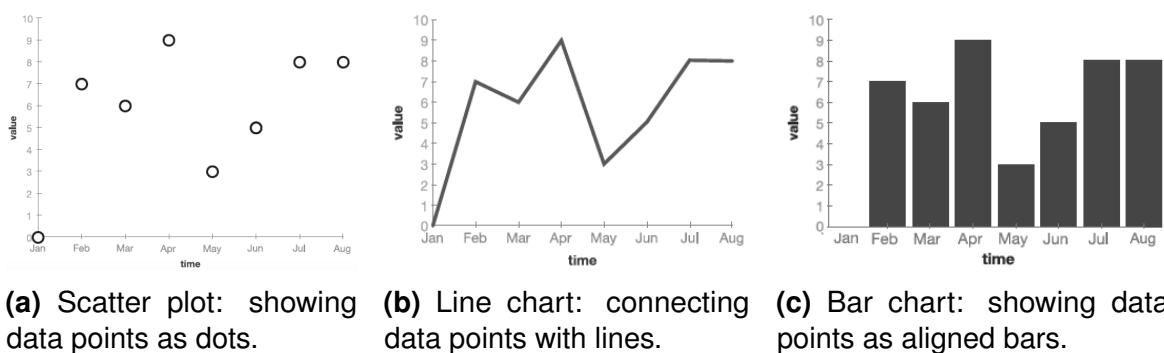
comprehensive review of this topic. In this section, we focus on different visual mappings of time.

2.5.1 Horizontal

The most common representation of time is mapping it to a horizontal axis as in the aforementioned *Chart of Biography*. Data items are positioned along the axis as either a point mark for point-based time or a line mark for interval-based time. Given a two-dimensional space, the vertical axis is available for encoding additional information.

2.5.1.1 Scaled Vertical Axis

Time-series data is a sequence of data points collected at uniform intervals such as population of a country for every year and stock market value for every hour. A time-dependent variable in time-series data is often mapped to a vertical axis. Classic charts such as scatter plot, line chart and bar chart and are all commonly used for this purpose. Scatter plot shows each data point as a point mark, with the x-coordinate mapping to the temporal value and the y-coordinate mapping to the value of the time-dependent variable. Line chart further connects these data points to form a line. Bar chart also shows data points individually like scatter plot, but each data point is represented by a bar, with its height corresponding to its time-dependent value. Line chart is suitable for showing trends of the series, whereas scatter plot and bar chart are good at emphasizing individual data points. With the advantage of visual alignment, bar chart is more effective than scatter plot at comparison of time-dependent values [6]. Figure 2.36 shows an example for each of these three charts.



(a) Scatter plot: showing data points as dots. **(b)** Line chart: connecting data points with lines. **(c)** Bar chart: showing data points as aligned bars.

Figure 2.36: Three charts showing the same time-series dataset with the horizontal axis representing time and the vertical axis representing a numerical value.

Horizon graph [144] is a recent improvement of line chart for visualizing time-series data, designed for a more space-efficient representation in order to facilitate comparison of different series, such as daily prices for one year of multiple stocks. To make a fair comparison, it shows a derived percentage changes from the earliest data point instead of raw values. Starting from a line chart, the value range is divided into equal bands, such as 10% for each band, and color coded using a diverging colormap for positive/negative values with increasing color intensity for greater band values. The colored bands allow more precise value reading. Then, the negative values are mirrored into the positive side, reduced the chart height by half. To save more space, those bands are layered with increasing values atop using the two-tone pseudo coloring technique [153]. Figure 2.37a illustrates these steps, and Figure 2.37b shows prices of 50 stocks over a year.

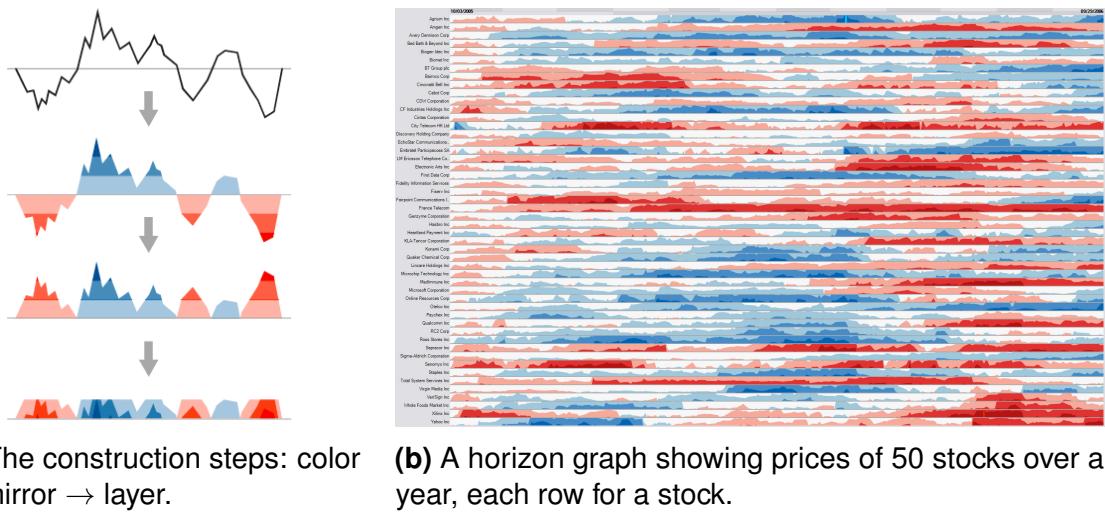


Figure 2.37: Horizon Graph: a space-efficient visualization of time-series data. *Image source: [144].*

Horizon graph uses space more efficiently than line chart in visualizing time-series data. A study by Heer, Kong and Agrawala [72] investigates their performances in value comparison tasks. The results show that mirroring a chart (flipping negative values) does not have negative effects; it neither slowed completion time nor hurt accuracy. Moreover, for charts with small sizes, layered bands are even more effective than line chart.

Stacking multiple area charts on top of each other is a suitable approach to visualize multiple time-dependent variables. ThemeRiver [68] can be considered as a smooth and symmetric version of stacked graph, designed to show thematic variations over time within a large collection of documents. Each theme is displayed

as a colored current flowing through the time, and at any point, the width of the current maps to the strength of the associated theme. The overall river consists of multiple colored currents, providing a good overview of the themes that were important at certain points in time.

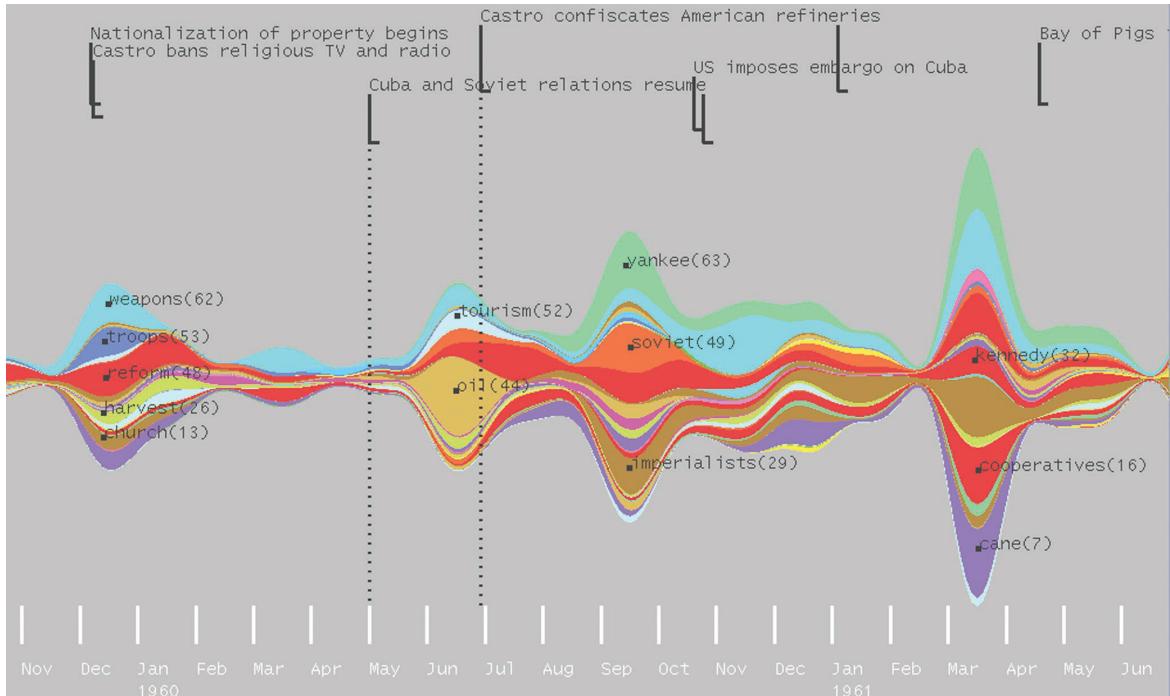


Figure 2.38: ThemeRiver uses a river metaphor to represent themes in a collection of Fidel Castro's speeches, interviews and articles from the end of 1959 to mid-1961. Each colored current corresponds a theme. *Image source: [68].*

Byron and Wattenberg discuss considerations of aesthetics and legibility for designing such stacked graphs. Figure 2.39a shows the traditional stacked area chart, where the bottom of the lowest layer is a horizontal line at 0. Figure 2.39b shows the ThemeRiver version, which is optimized for the symmetry of the entire layout. Figure 2.39c shows the Stream Graph [24] version, which minimizes the number of wiggles of layers.

2.5.1.2 Non-Scaled Vertical Axis

The vertical dimension may be used to avoid clutter in producing a compact layout. LifeLines [138, 139], a visualization system of patient records, is a good example. It groups these records into different facets such as problems, allergies, diagnosis and medications, and vertically stack them on top of each other. Within each facet, interval-based records are represented as horizontal bars covering their timespans.

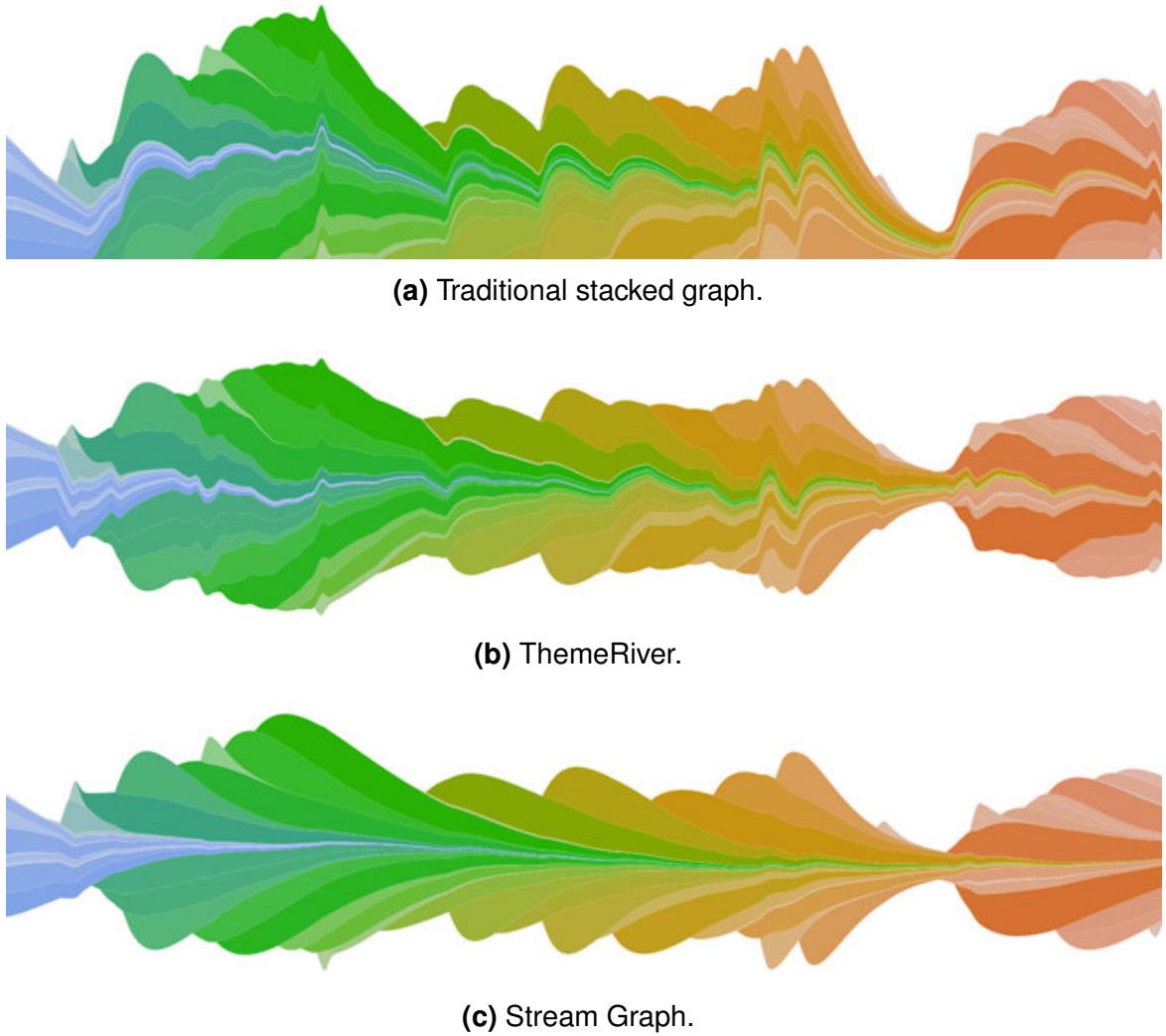


Figure 2.39: Stacked graphs with different design considerations. *Image source: [6].*

Because their timespans may overlap, the layout adjusts their vertical coordinates to avoid intersection. Figure 2.40 shows an example of LifeLines.

Another example is Continuum [10], a timeline visualization of hierarchically structured temporal data such as the relationship of era, composers and pieces. The timeline shows the lifespans of composers as rectangles, where the width represents temporal information, and the height depends on the number of pieces they composed. It also uses vertical position to produce an overlap-free layout.

Temporal visualization often encode additional relationships of data items. Gantt chart [54] is a classic method for displaying planning activities in project management. Each activity is shown as a horizontal bar covering the planning time, and text from the left part of the chart shows activity names. Dependency is a common

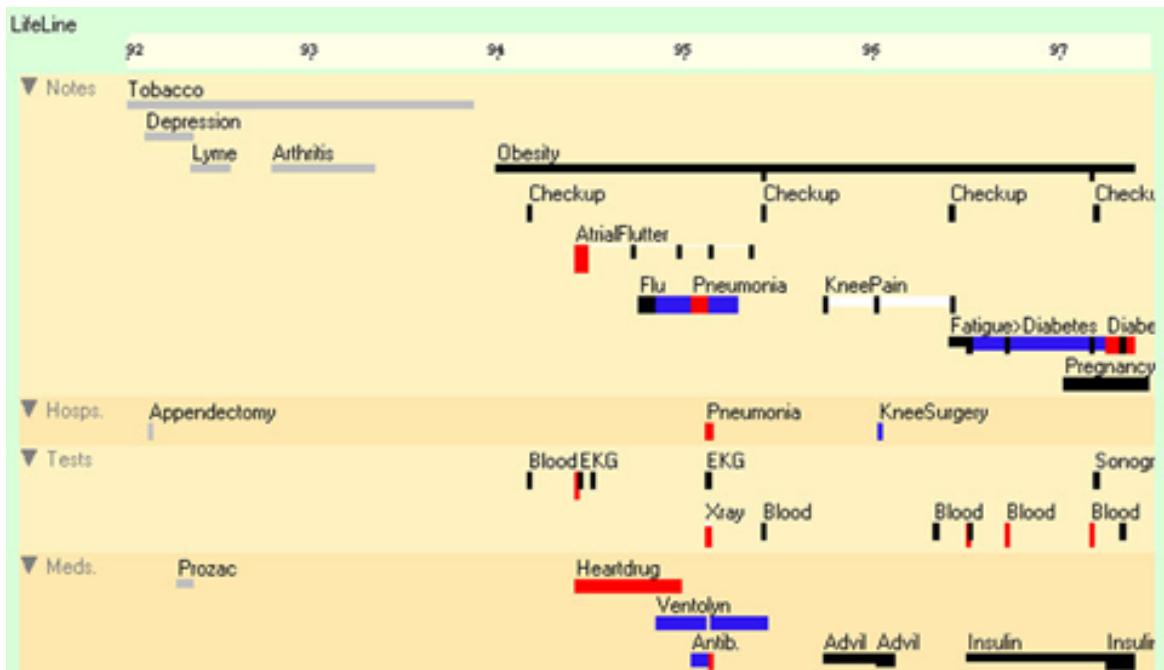


Figure 2.40: LifeLines. The timeline consists of several facets, stacked vertically. Each facet includes health-related records shown as horizontal bars, which may be located in different rows to avoid overlapping. *Image source: [139].*

relationship in planning and can be shown as an arrow. For instance, an arrow pointing from the end of activity *A* to the beginning of activity *B* can indicate that *B* must happen after *A*. Activities are often organized in hierarchical structure such as tasks and sub-tasks. They can be ordered and arranged with indentation to reflect this structure. Timeline tree [21] draws an explicit node-link tree on the left part of the timeline to show the hierarchy. Figure 2.41 shows a Gantt chart with hierarchically structured tasks.

Spatial proximity is another method to represent relationships. This method is applied in storyline visualizations to illustrate the dynamic relationships between characters in a movie, which was first introduced by Munroe with his hand-drawn charts [118]. The visualization depicts each character as a curved line and each scene as a bundle of those character lines. Ideally, all the lines within a scene should be horizontally parallel. A line diverges from its bundle if the character leaves the scene, and conversely, a line converges into a bundle if the character joins that scene. Algorithms have been introduced to automate the drawing process, including work by Tanahashi and Ma [168] and Liu et al. [104]. Figure 2.42 shows the storyline visualization of the *Star Wars* movie.

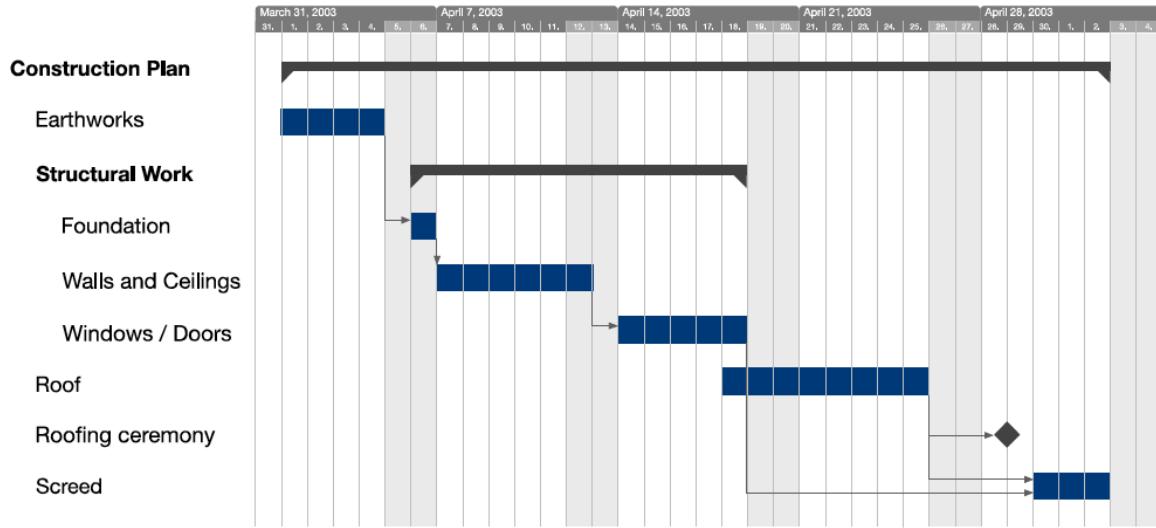


Figure 2.41: Gantt chart. Each task is shown in an individual row with task name on the left and horizontal bar on the right spanning task time. Arrows show dependency, and label indentation indicates the hierarchy of tasks. *Image source: [6].*

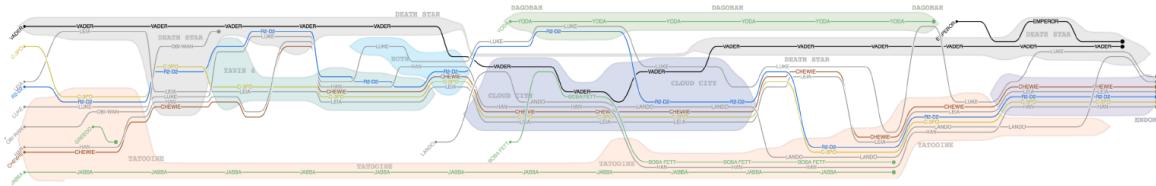


Figure 2.42: Storyline visualization of the movie *Star Wars*. Each line represents a character, and a bundle of lines represents an interaction of those characters. *Image source: [168].*

Similarly, TimeNets [93] also applies *proximity* to visualize relationships of genealogical data. It uses a curved line to represent the lifespan of a person. These lines are located spatially distant if the persons they represent are unrelated. Two lines converge if the represented persons marry and diverge when they divorce. Child lines stay close to their parent lines, and dotted vertical lines are drawn from the parents to the beginning of their child lines to indicate the parent-child relationship. Figure 2.43 shows a TimeNets example depicting these relationships.

2.5.2 Spiral

Another representation of time is mapping it to a *spiral axis*, and data items are positioned along that spiral [187]. Color intensity and line thickness are suitable for encoding an additional quantitative variable. For interval-based data, filled curved segments are aligned with the spiral to indicate the two ends of intervals [25]. Spirals

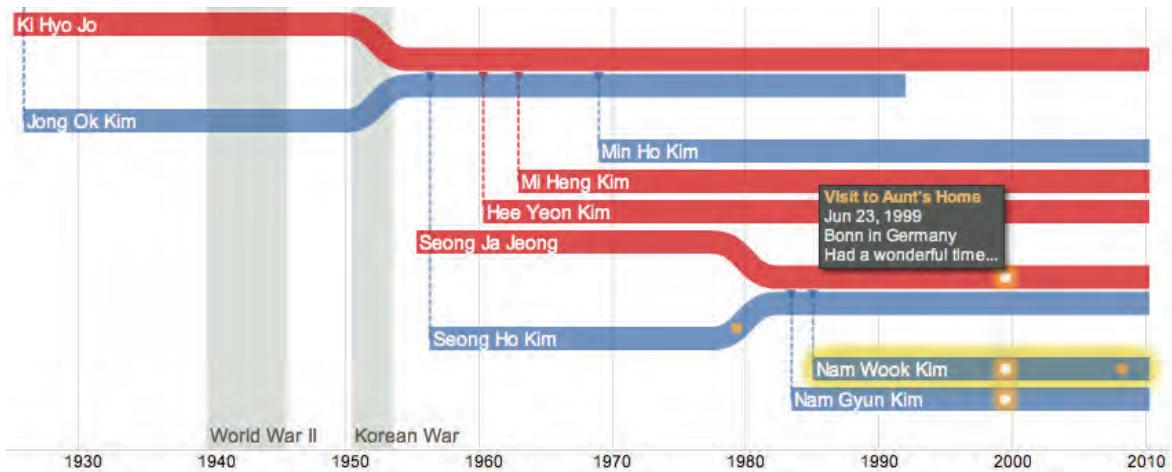


Figure 2.43: TimeNet visualization of genealogical data. Lifelines represent people, converging lines signify marriage, and drop lines indicate children. *Image source: [93].*

can also be intertwined to show multiple variables. Figure 2.44 shows a spiral graph comparing two variables over time with different color hues.

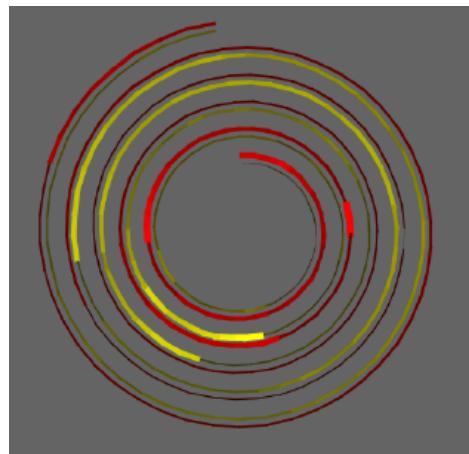


Figure 2.44: Spiral graph. Time is represented along a spiral with color intensity and line thickness are used to indicate time-dependent, numerical values. Two variables are distinguished using different color hue: yellow and red. *Image source: [187].*

Spiral graph is effective at spotting periodic patterns of the data, but it highly depends on the cycle length; i.e., the number of time steps per cycle. Figure 2.45 shows three charts using the same time-series dataset. Figure 2.45a uses a bar chart and clearly reveals trends and extreme values. The other two charts use spiral graphs; however, a cyclic pattern is only visible in Figure 2.45c. The difference between them is the cycle length: 24 days for Figure 2.45b but 28 days for Figure 2.45c, which clearly shows a pattern of four weeks. This pattern can also be revealed if the cycle

length is set to a small multiple of 7 days. Interaction has been proposed to facilitate users in identifying the right cycle length [173, 187]. Users can manually adjust the cycle length. Alternatively, users can watch an animation of the visualization through different cycle lengths and stop the animation when they find the pattern of interest.

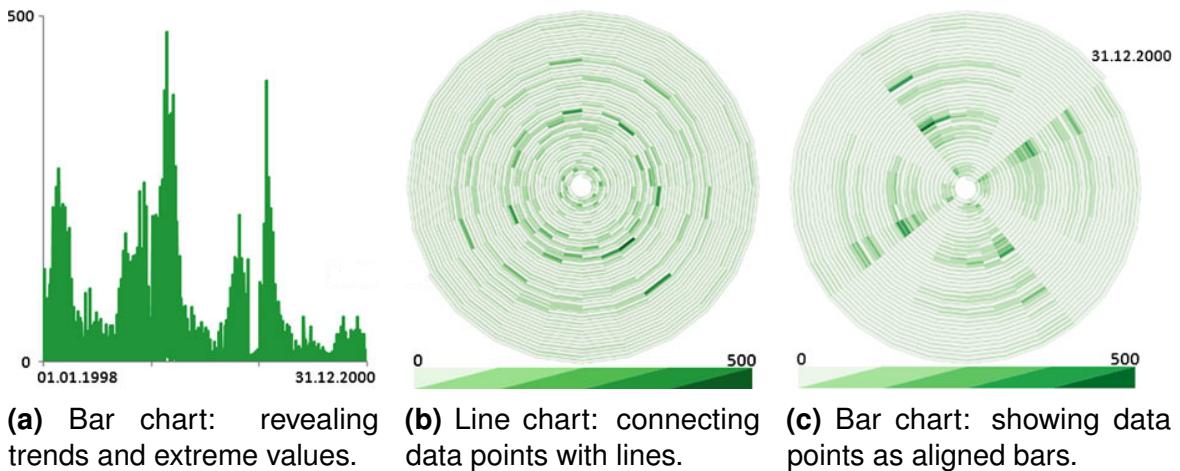


Figure 2.45: Different insights can be gained from visual representations depending on whether the linear or cyclic character of the data is emphasized. *Image source: [6].*

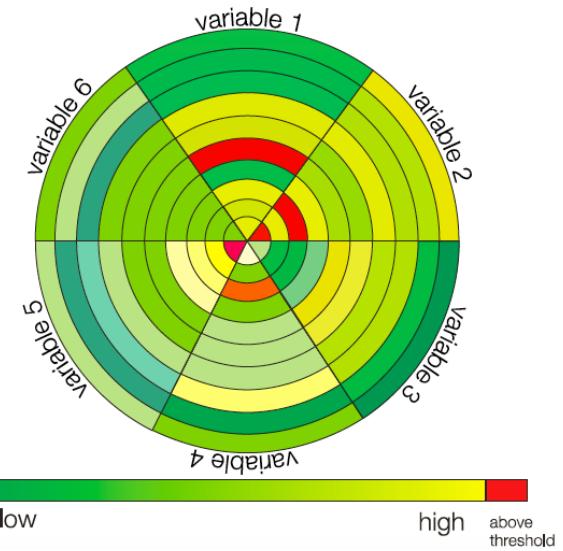
2.5.3 Circle

Time can also be represented using a *tree-ring* metaphor. In a tree, a new layer of wood cells is produced every year, growing out from the center (Figure 2.46a). Inspired from this phenomenon, Keim, Schneidewind and Sips [91] introduce Circle View – an approach to visualize time-related multidimensional datasets. It splits a circle into multiple concentric rings, each for a time step. The circle is divided into a number of segments, each representing a variable. Figure 2.46b shows such a view with six variables. Color is used to show the (aggregated) data value for the corresponding interval.

Based on tree-rings, Therón [171] develops a technique to show both temporal and hierarchical information. Figure 2.47a shows a simple example of such a dataset as tree with five nodes, each associated with a timestamp. These nodes are assigned to the rings based on their temporal values before being positioned along the ring in such a way that arrows can be drawn from parent nodes to child nodes to reflect the hierarchy (Figure 2.47b).

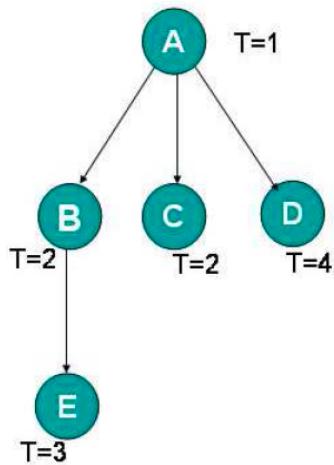


(a) Cross section of Douglas Fir tree showing almost perfect tree-rings. *Image source: [171].*

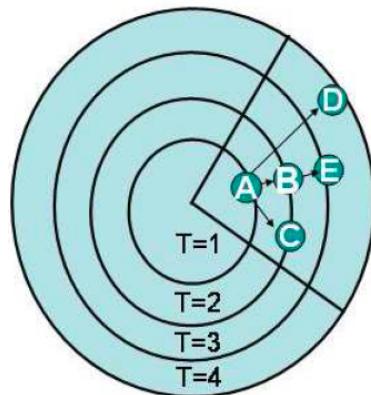


(b) Circle view showing six variables over ten time steps as concentric rings. *Image source: [91].*

Figure 2.46: Tree ring. Concentric rings expanding from the center, each indicating a time step.



(a) Hierarchy is shown as a tree with temporal values are annotated next to the nodes.



(b) Hierarchy is shown as a tree embedded on a circle view with rings indicating temporal values.

Figure 2.47: Visualizing both temporal and hierarchical information using tree rings. *Image source: [171].*

2.5.4 Calendar

Another method to represent time is using a *calendar*. A day is usually color coded based on its value to reveal patterns in the data. A calendar visualization allows users to identify patterns at different temporal granularities such as daily, weekly and monthly. Figure 2.48 visualizes the daily power consumption over a year with color indicating the result of a clustering algorithm. Several patterns can be observed in this figure. First, the energy consumption is low in the summer and on Fridays. Second, it is even lower at weekends and holidays such as Christmas and New Year. Third, this figure shows the energy consumed in Netherlands, thus some patterns are clear for Dutch people such as on December 5th, employees can leave their office earlier to celebrate Santa Claus.

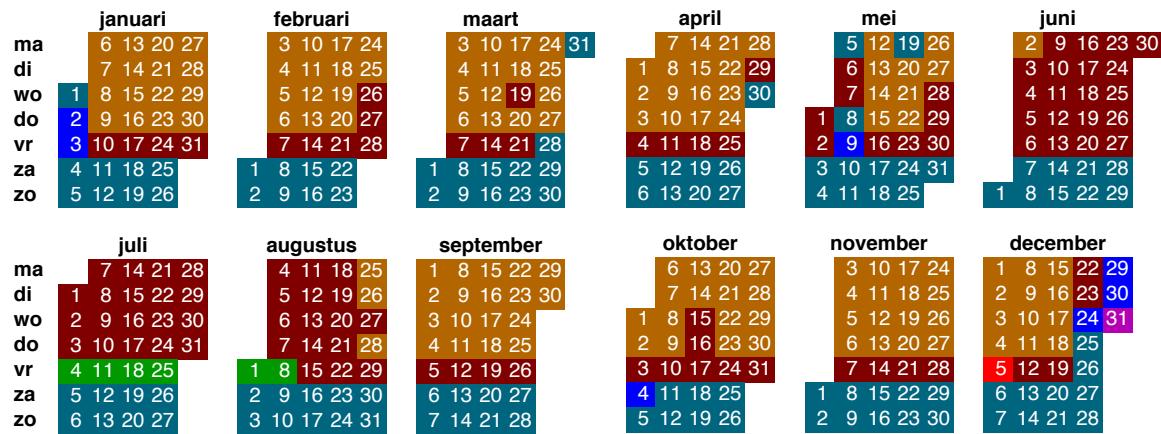


Figure 2.48: Calendar visualization of a one-year data of daily power consumption. A cell for each day is color coded to reveal data patterns. *Image source: [181].*

2.5.5 Small Multiples

Another method to depict changes over time is *small multiples* – a set of miniature visual representations placed next to each other with each showing the visualization at a selected time step [176]. Small multiples provide an overview of the data and allow users to visually compare it at different time points. The concept is general and can be applied to virtually all static visualization techniques because only thumbnails of the visualization for each time step are required. Figure 2.49 shows an example of small multiples of bar charts.

To facilitate the exploration of relationship in the data at multiple time steps, the miniatures should be interactive and linked together, rather than just static

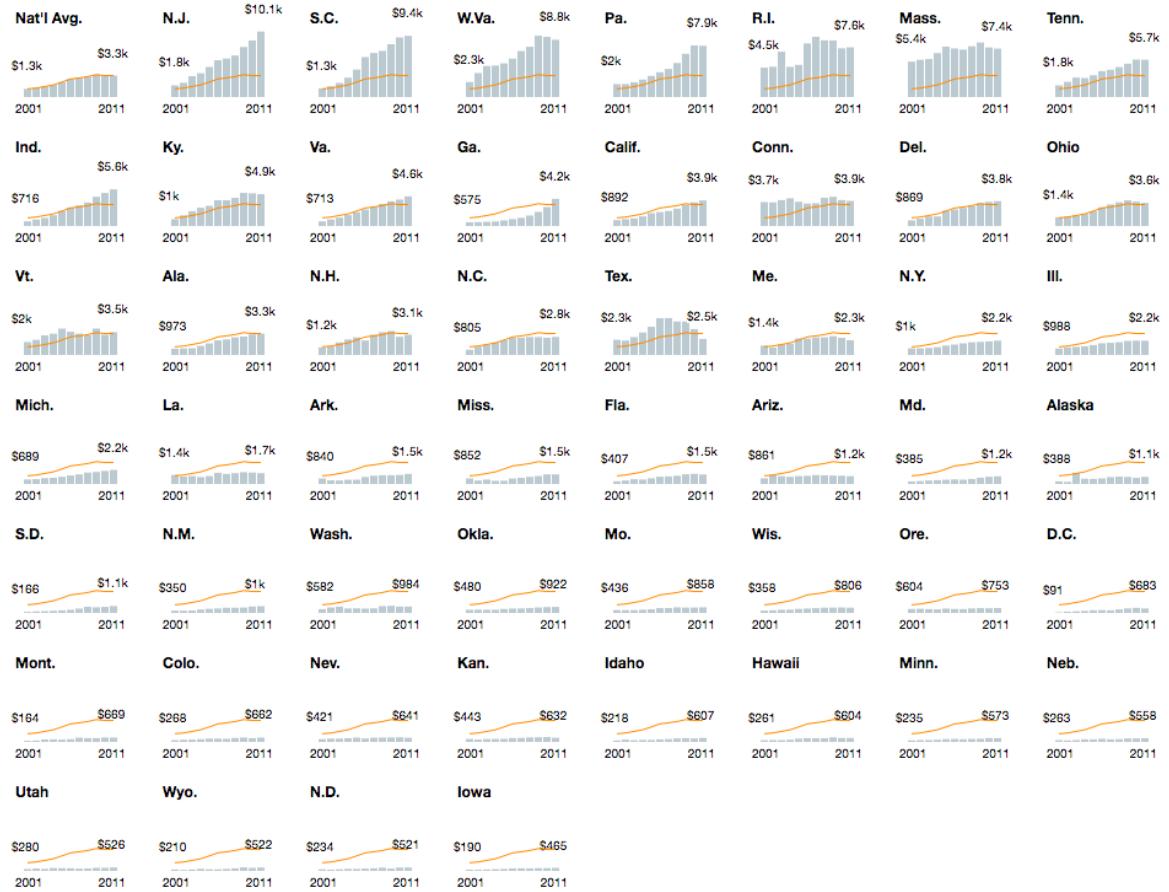


Figure 2.49: Small multiples of bar charts showing average annual Medicare spending on ambulance services per dialysis patient by U.S. state from 2001 to 2011. *Image source: [3].*

thumbnails. Figure 2.49 shows small interactive bar charts sorted decreasingly by the value spent on the last year. Alternatively, they can be ordered alphabetically by state names to facilitate searching. Standard brushing and linking interaction technique also helps compare the subsets of interest. A limitation of this method is scalability. The number of representable time steps are relatively small because of the thumbnail size.

2.5.6 Animation

Animation is another technique to convey time that can be applied to virtually all static visualization techniques. It relies on human perception in perceiving changes when a visualization smoothly updates from one time step to another. The most notable example is Trendalyzer by Gapminder Foundation [2] – an interactive visualization and presentation tool based on scatter plots. The animation is controlled

via a time slider, a play/pause button, and a speed slider. Only a few data items should be animated, and they are often highlighted so that the user can keep track of the changes. Trails may also be displayed to maintain the path of a data item through time. A study by Robertson et al. [146] shows that animation is both slower and less accurate than small multiples in conveying trends over time.

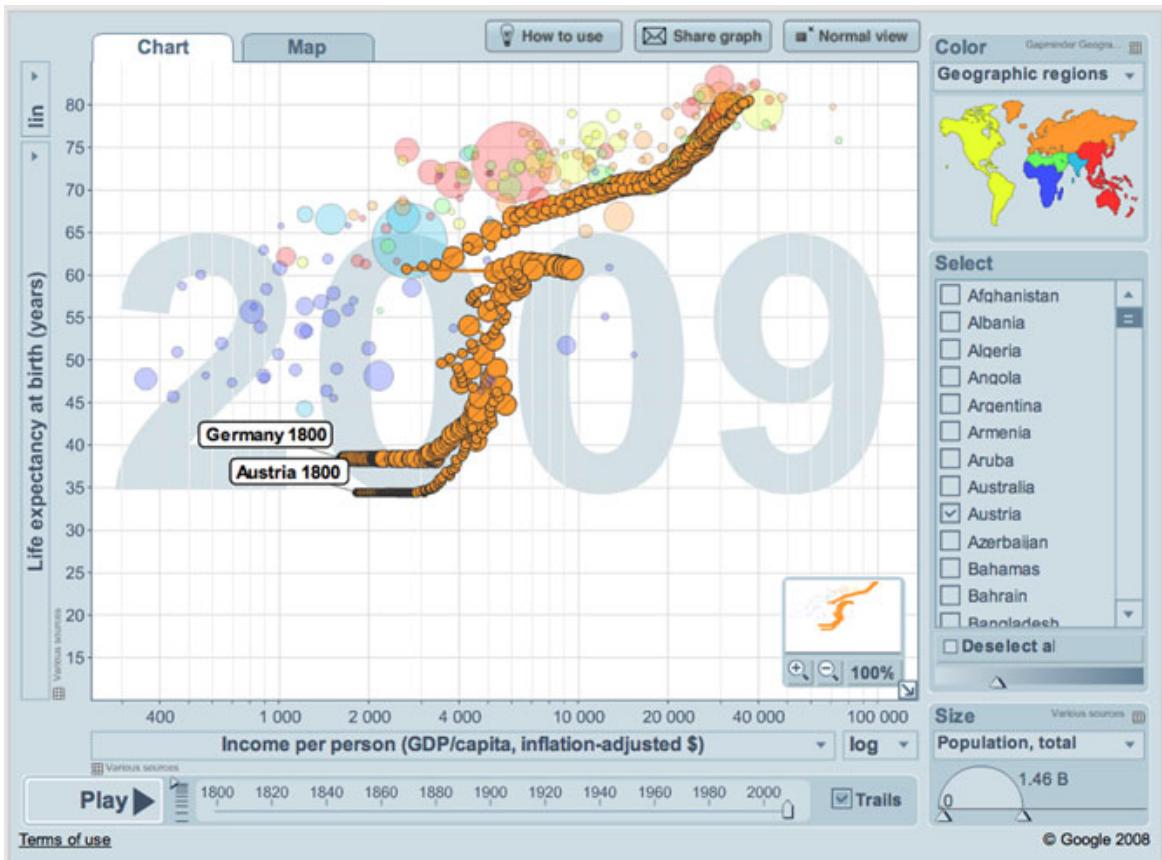


Figure 2.50: Trendalyzer interface. A scatter plot with an animation controller to traverse through time. Additionally, trails are activated for the selected countries, Austria and Germany, which help to preserve the path of a variable in animation. *Image source: [6].*

Besides showing trends of time-series data, animation has also been applied in other temporal datasets. Animation is a powerful and appealing technique in presentation of a known story [55]. It helps illustrate computer algorithms step by step and motivate students in approaching complex problems [89]. Animation also allows reproduction of a data exploration process by interpolating visual parameters of key saved visualization steps [108].

2.6 Visualization of Network and Tree Data

2.6.1 Node-Link Diagrams

The most common visual encoding for network and tree data is *node-link diagrams*, where nodes are represented as point marks and links connecting these nodes are represented as *connection marks*. We further discuss layouts using this representation for network data with different constraints: rooted tree, directed graph and general graph.

2.6.1.1 Tree Layout

A classic algorithm by Reingold and Tilford [145] produces a tidy tree layout satisfying the following aesthetic rules:

1. Nodes at the same level of the tree should lie along a straight line, and the straight lines defining the levels should be parallel.
2. A left son should be positioned to the left of its father, and a right son should be positioned to the right of its father.
3. A father should be centered over its sons.

Even though the layout produced by Reingold and Tilford's algorithm is tidy, it still leaves plenty of void space at the root side of the tree as shown in Figure 2.51a. Marriott and Sbarski [110] relax the rule that a parent must be placed at the center of its children by slightly shifting branches of nodes to produce a narrower tree. In practice, nodes have their sizes rather than just single points, and their heights can also be different. Van der Ploeg [179] relaxes the layering requirement to produce a shorter tree as shown in Figure 2.51b.

Conventionally, tree layouts are rectilinear with children branching from the parent node either from left to right or top to bottom. However, the children nodes can also be arranged radially, along a circular arc of their parent. The depth of a circular tree is encoded as distance away from the center of the circle. Reingold–Tilford algorithm can be modified to show a radial tree. Figure 2.52 shows the class hierarchy of the *Flare* visualization toolkit [69] using both a conventional tree and a radial tree. The toolkit consists of 10 top-level classes, which determine the color of nodes in these figures.

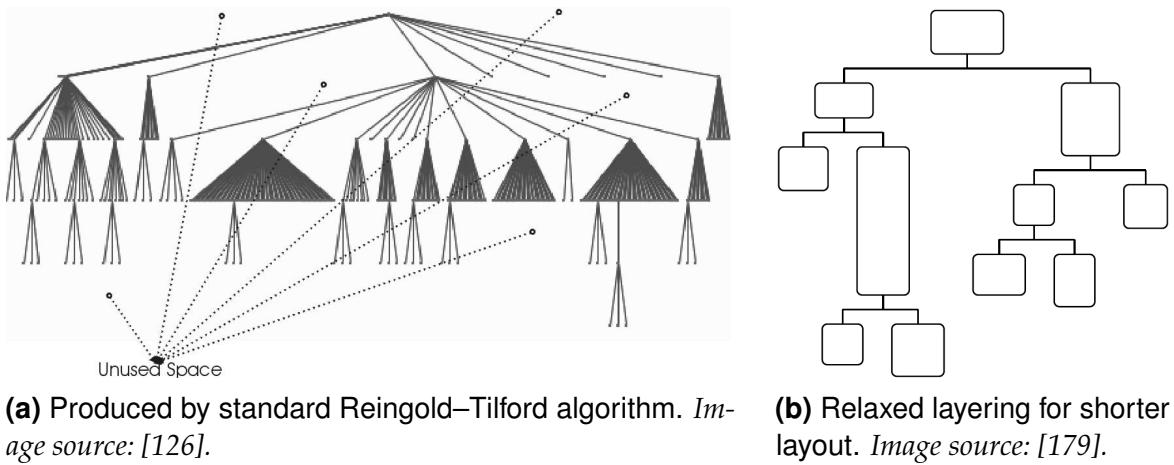


Figure 2.51: Tree layouts.

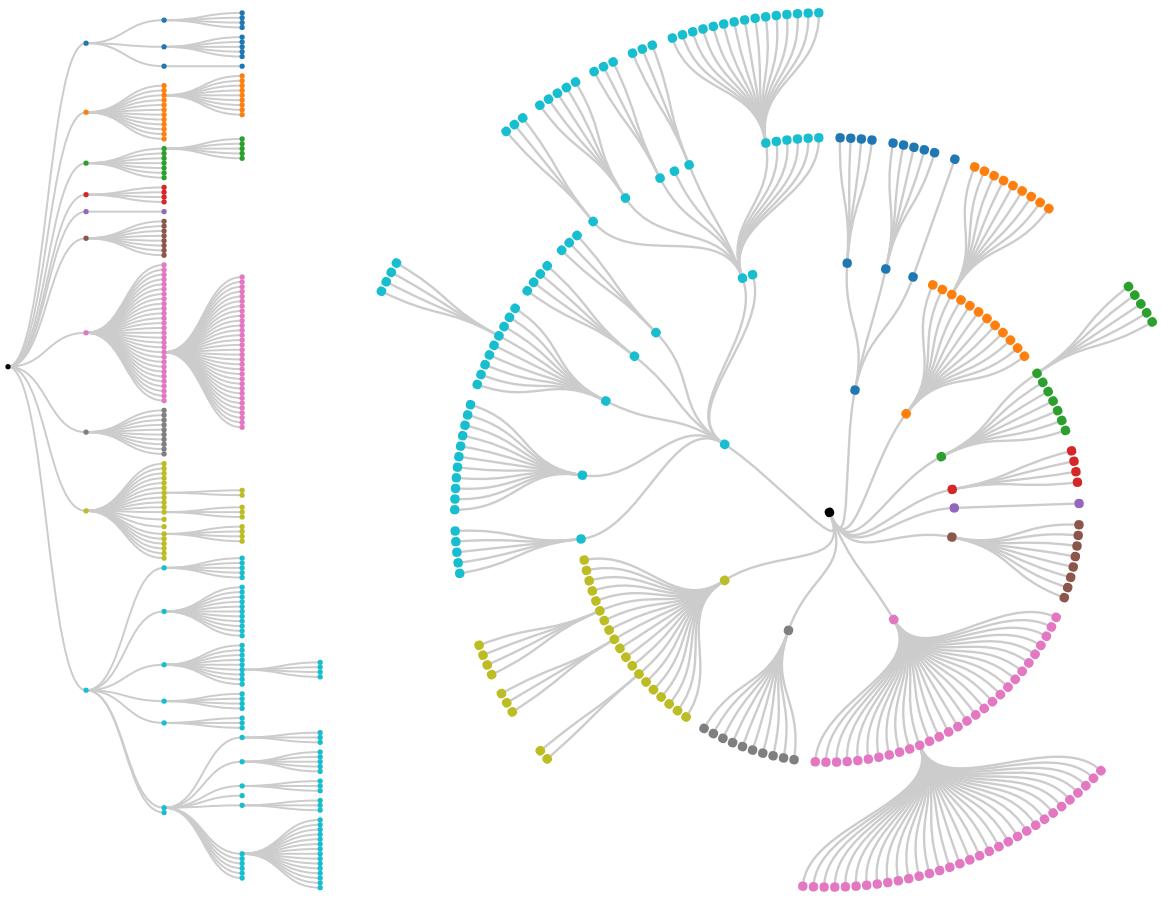
2.6.1.2 Hierarchical Graph Layout

The most popular method of drawing directed graphs is the Sugiyama framework [166], separating nodes into layers to show the hierarchy effectively. Figure 2.53 shows an example of layered graphs. The framework consists of four steps.

1. *Cycle removal.* If the input graph contains directed cycles, temporarily reverse the direction of some edges to make the graph acyclic.
2. *Layer assignment.* Nodes are assigned to horizontal layers, which determines their y-coordinate.
3. *Vertex ordering.* Within each layer, the nodes are ordered to minimize edge crossings between adjacent layers.
4. *Horizontal coordinate assignment.* The x-coordinate of each node is determined, typically aiming to make edges straight.

2.6.1.3 Force-Directed Layout

Force-directed layout is a popular method to visualize general graphs using node-link metaphor [47]. It positions nodes based on a simulation of physical forces: *spring-like attractive* forces attract nodes, and *repulsive* forces like those of electrically charged particles push them away from each other. The layout usually starts with a random arrangement of nodes and then iteratively refines their locations according to the behavior of the simulation. The layout stops when the simulation reaches a stable state or a maximum number of iterations. Figure 2.54a illustrates the idea of



(a) Conventional tree, growing direction as left to right.

(b) Radial tree. Depth is encoded as distance away from the center.

Figure 2.52: Tree layouts with different orientations. Data is the class hierarchy of the Flare visualization toolkit [69]. Color represents the top-level classes.

physical simulation, and Figure 2.54b shows an example with color indicating node type.

Force-directed layout is aesthetically pleasing, aiming to produce uniform edge length, symmetry and even node distribution [51]. However, its major weakness is scalability, in terms of both the visual complexity and the running time [119]. The layout quickly degenerates into a hairball of visual clutter with even a few hundred nodes. Another limitation is its nondeterministic output: the layout looks different each time it runs, breaking user mental model.

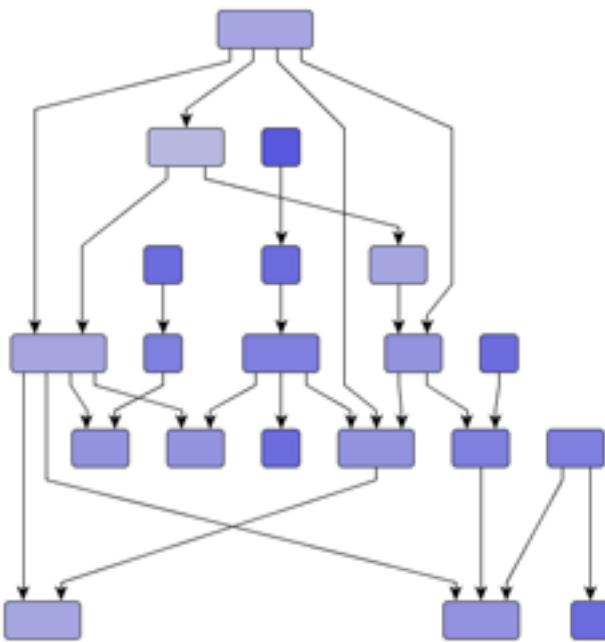
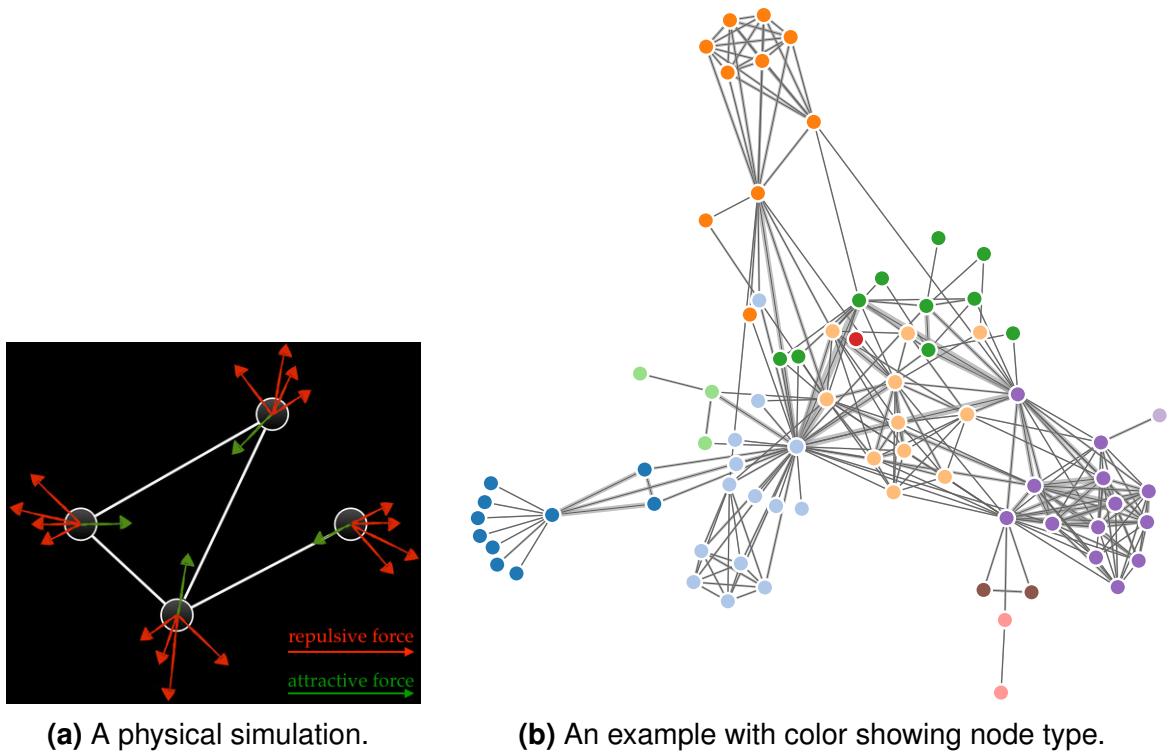
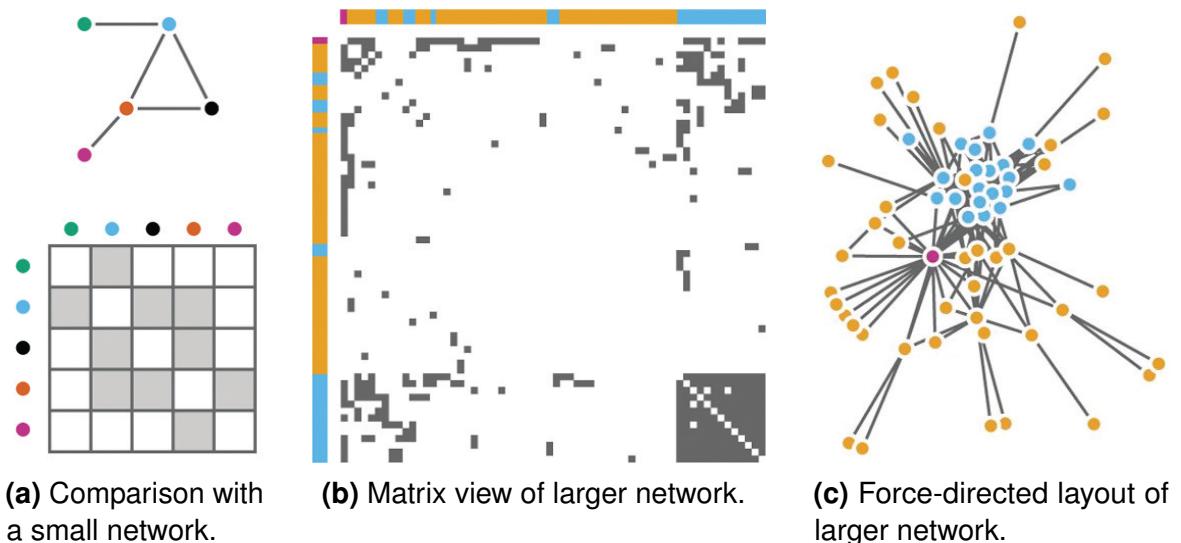


Figure 2.53: A layered graph. Nodes are assigned to horizontal layers. Within each layer, nodes are ordered to minimize edge crossings and assigned x-coordinate to make edges as straight as possible. *Image source: yWorks.*

2.6.2 Matrix Views

A network can be represented by an adjacency matrix. Each row and column of the matrix corresponds to a node, and a cell indicates whether the pair of corresponding nodes is connected in the network. Additional information about edges are often encoded to the visual representation of cells using color, shape and size. Matrix views can also show weighted networks, where each link associates with a quantitative value attribute, by encoding cells with an ordered visual channel such as color luminance or size. Figure 2.55 shows examples of matrix views, compared with node-link diagrams of the same dataset.

A major strength of matrix views is the scalability. It can easily show a networks with thousands of nodes and millions of edges without suffering from the cluttering issue as in node-link diagrams. Matrix views are stable: adding a new node or edge will only cause a small visual change. Whereas, adding a new item in a force-directed view may lead to a major change [119]. Nodes, as columns and rows in a matrix view, can be reordered, allowing users to reveal outliers, clusters, and patterns of the network [76].

**Figure 2.54:** Force-directed layout.**Figure 2.55:** Comparison of node-link diagrams and matrix views. Gray cells indicate edge connectivity. *Image source: [119].*

A major weakness of matrix views is their difficulty in exploring the topological structure of the network, such as path tracing. This because they show links in a more indirect way than the direct connections of node-link diagrams – a trade-off for

their strength in avoiding clutter. Another weakness of matrix views is unfamiliarity: users easily interpret small node-link diagrams but require training to understand matrix views [119]. A study shows that for many low-level abstract network tasks, node-link diagrams are best for small networks, whereas matrix views are best for large networks [56].

2.6.3 Space-Filling Techniques

Space-filling techniques only apply to tree data and uses *containment* marks to represent hierarchical relationship, placing child nodes within their parent node. Treemap [158] represents a node as a rectangle, which is recursively subdivided into smaller rectangles, each represents a child of the node. The rectangle size is proportional to a quantitative attribute of the node. The original motivation of treemap is to analyze the utilization of storage space on a hard disk. Each leaf node represents a computer file, and the node size encodes the file size. The size of a parent node simply maps to the containing folder size. Color is also commonly used to encode additional information to nodes such as file type. Figure 2.56 shows such an example of treemap.

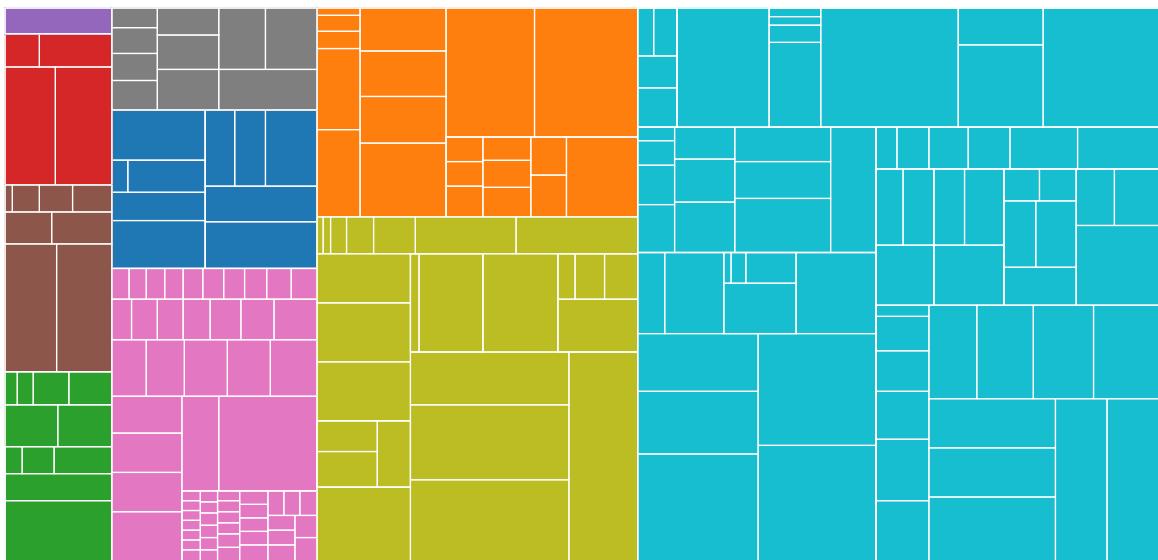


Figure 2.56: Treemap showing the class hierarchy of the Flare visualization toolkit [69]. Area represents class sizes and color represents the top-level classes.

Treemap is very effective when size is the most important feature to be displayed. It easily spots outliers of very large attribute values such as large files. However, containment is not as effective as connection in node-link diagrams for tasks focused

on topological structure. It is difficult to identify the path of a given node, thus suitable for hierarchies with just a few levels. Borders of nodes or shading [189] can be used to depict the tree structure more strongly.

Alternatively, other space-filling techniques have been proposed to better represent the hierarchical structure. Figure 2.57 shows the three techniques that we discuss next using the same dataset as in Figure 2.56 for treemap.

Circle packing [184] also employs containment to represent the hierarchy, but using circles to represent nodes. Circle sizes correspond to some node values. All child nodes are packed into their parent node so that they are tangent to some of their sibling nodes. This method uses more space than treemap but provides better hierarchy structure.

Icicle plot [98] does not strictly use containment; instead, it places child nodes *under* their parent node. Similar to treemap, icicle plot uses rectangles to represent nodes, but they all share the same height. Therefore, node width corresponds to some attribute value. Icicle plot shows parent nodes explicitly, making it more effective in tasks related to the parents such as comparing directories by size. The direct trade-off is space for showing those parent nodes. Icicle plot shows the tree structure and supports path tracing more effectively than treemap. However, small leaf nodes are very thin and difficult to read its label or interact with.

Sunburst [199] is a radial version of icicle plot. Child nodes are located under their parent node in a circular layout. The root node is at the center and deeper levels are further away from it. The angle swept out by a node, or a curved segment, corresponds to its value. Color is also commonly used to represent additional information.

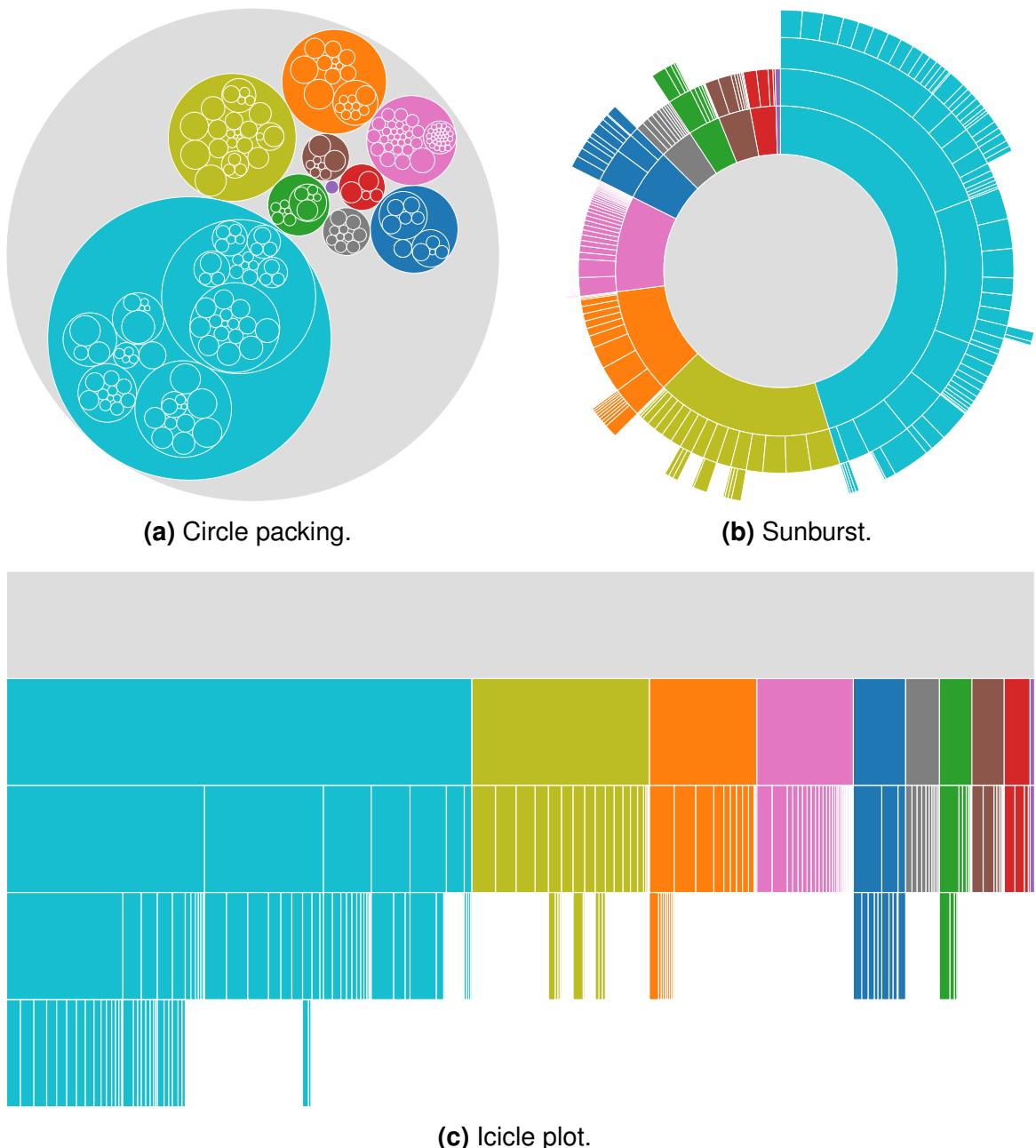


Figure 2.57: Other space-filling techniques. Data is the class hierarchy of the Flare visualization toolkit [69] as in Figure 2.56. Color also represents the top-level classes.

3

Exploring Temporal Relationship through User Annotations

As the first step toward supporting sensemaking using analytic provenance, this chapter discusses how to enable users to explore the hidden **temporal relationship** of sensemaking through the **annotations** that the users made. *Annotation* is a common approach to capture high-level thinking and reasoning of users throughout their sensemaking processes. *Timeline* is a simple yet powerful technique to visualize time-oriented data, enabling users to examine information chronologically and to identify temporal patterns and relationships. However, existing timeline visualization methods are not specifically designed for the dynamic and iterative nature of sensemaking. In this chapter, we introduce a novel timeline visualization – *SchemaLine* – to address these issues. SchemaLine produces a compact but aesthetically pleasing layout and provides a set of fluid interactions allowing users to perform various sensemaking activities. Our user-centered evaluation shows that the participants found the new method easy to learn and use, and its features effective in supporting the sensemaking activities for which it was designed.

3.1 Introduction

Timeline is a simple yet powerful technique to visualize time-oriented data [176], enabling users to explore and identify temporal patterns and relationships in the data. It displays events along the time axis and position them at the time points at which they occur or the time ranges over which they last [138]. Pirolli and

Previously published at the *International Conference on Information Visualization* [123].

Card [136] suggest the usefulness of timeline visualizations in the schematization process of sensemaking. A timeline may help organize collected information based on a chronological order and construct plausible stories. A user study by Kang, Görg and John Stasko [88] also shows that timeline is a common choice of participants for organizing related events in an attempt to make sense of them.

A timeline does not only reveal the temporal relationships of user findings, but also affects how easily they can be understood. A study by Pennington and Hastie [131] suggests that the presentation order of evidence has a significant impact on making decisions in a court trial. The juror constructs stories based on evidence from witnesses, exhibits and arguments before concluding with the most plausible one. The study shows that participants were most likely (78%) to convict when the prosecution evidence was presented in story order and the defense evidence was presented in non-story order; whereas, participants were least likely (31%) to convict when these sets of evidence were presented in the other way round.

Timelines have been applied extensively in visualizing both raw data and analysis findings for supporting sensemaking. POLESTAR [135] and HARVEST [60] allow users to take notes, define new knowledge, and explore them through a timeline visualization. Jigsaw [59] uses timelines to organize extracted named entities such as people, places and organizations. nSpace2 Sandbox [163] allows the creation of multiple sub-timelines for visualizing different types of artifacts. However, these timeline visualizations either lack an automatic layout [135] or use an overly-simplistic linear layout [163]. As a result, the visualization requires significant effort from users to manually arrange the data items, making it difficult for them to detect any temporal patterns of the data.

Sensemaking includes dynamic activities centering around the collected data and its explanation [95]. To support the dynamic nature of sensemaking, timeline visualization allows analysts to interactively create and edit temporal structures. Also, this should be achieved through intuitive and fluid interaction to prevent extra cognitive effort and distraction from users. However, existing timeline visualization techniques are mainly designed for presenting a known story rather than interactively revealing and constructing a hidden one.

In this chapter, we introduce a novel timeline visualization – SchemaLine – to address the aforementioned issues. More specifically, SchemaLine contributes

- A visual design for an interactive timeline that groups user annotations into user-determined schemas.

- An algorithm to automatically generate a compact and aesthetically pleasing visualization of these schemas on the timeline.
- A set of fluid interactions with the timeline to support the sensemaking activities described in the Data–Frame model [95].

3.2 Approach and Requirements

We began by exploring the literature to identify the essential points of the sensemaking process to support. As discussed in the Literature Review chapter, Section 2.1.4, Pirolli and Card describe sensemaking as a cyclic process including two major loops: the foraging loop and the sensemaking loop. In that model, *schematization* serves as a bridge connecting the two loops and plays an important role in converting raw evidence to rational explanations. Pirolli and Card [136] suggest that schematization should be supported by a computer-based tool that coordinates events in the dataset to reveal their relationships, and to reduce the effort from users in memorizing them. Therefore, we decided to support *temporal schematization* through interactive visualization.

Also discussed in the Literature Review chapter, Section 2.1.5, the cognitive processes in schematization are well elaborated through various sensemaking activities centering around *data* and *frame* in the Data–Frame model. For instance, during sensemaking, the user finds some pieces of interesting information. Then, he or she realizes that these pieces mention the same person, thus decides to connect them based on when each event happens to reveal the hidden story. Using the terminology of the Data–Frame model, the user connects data (the pieces of interesting information) to a frame (the story). We decided to enable users to perform all sensemaking activities in the Data–Frame model through intuitive interaction with a timeline visualization.

Note that we use *schema* and *frame* to refer to the same concept: a structure that defines the relationship of data. In this chapter, we focus on the temporal relationship, explaining the chronology of events discovered during the sensemaking process. We use *events* instead of general data to emphasize on their temporal aspect. Therefore, a schema or frame can be defined as a chronological sequence of related events.

We propose the following design requirements for our timeline visualization.

1. **Automatic layout.** Provide an automatic layout to leverage users from manual arrangement of events.
2. **Natural flow.** Easy to follow events within the same schema chronologically.
3. **Data-frame model.** Enable users to perform sensemaking activities described in the Data-frame model through intuitive interaction.

3.3 Visual Design

3.3.1 Event

An event is represented by a rounded rectangle with a short text inside summarizing its content. The width of an event rectangle is constrained by some maximum value and long text is trimmed to fit into its area. The full content of an event will be displayed when it is hovered. Events can be classified into different categories based on a certain criteria. For example, a news article may write about *sport*, *fashion* or both. Small colored badges are added to the left of the text of an event to indicate its categories. The colors are chosen from Qualitative Set 1 of ColorBrewer [67]. Only around 12 colors can be distinguished simultaneously in the human view [119]. Therefore, only the eight most frequently appeared categories are displayed using the selected colormap; whereas, the rest share a different color. Figure 3.1 shows an event with three categories.

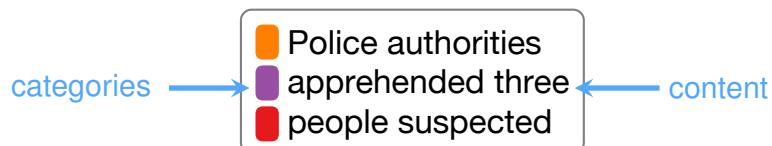


Figure 3.1: Visual representation of event. Text shows the content and colored badges indicate the categories.

An event is left-aligned with its temporal value on the time axis. To reduce cluttering, an event is not visually connected to its corresponding point on the axis. Instead, it only appears when the event is hovered. The time axis is shown as a horizontal line at the bottom of all events. It includes two hierarchical temporal scales, changed dynamically according to the range of the visible events. For example, the time axis in Figure 3.3 shows *month* and *day* but they can be switched to *year* and *month* if needed to cover the range of events.

3.3.2 Schema

As discussed in the Literature Review chapter, Figure 2.2.3.3, Gestalt principles of grouping are commonly used to show relationships between events, most effectively *connectedness* and *proximity*. Therefore, we also apply these two principles in our design: events belonging to the same schema are located close together, and the background of an entire schema is colored to visually connect all of its events. Spatial grouping needs to be achieved through vertical positioning because the horizontal position of each event is already determined by its temporal information. Locating all events within a schema close together also makes it convenient to follow them chronologically (Requirement 2 – natural flow).

Munroe's hand-drawn movie narrative charts [118] show the dynamic interactions of characters throughout the movie. Each character is represented as a curved line along a horizontal time axis; and vertical grouping of lines indicates which characters are together at a given interval. Inspired by this technique, we consider a "schema" as a "character line", connecting all of its events. However, instead of a thin line, we use a thicker path to provide enough space for displaying the content of events and to allow convenient interaction with individual events. For aesthetics, the path is connected rectilinearly, including only horizontal and vertical segments. Also, all events are constrained by the same height to make the width of the path consistent. Figure 3.2 shows two examples of schema.

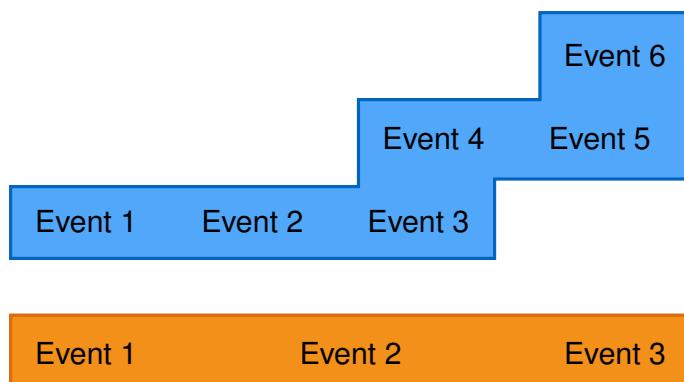


Figure 3.2: Visual representation of schema as a colored stripe. Bottom: a simple rectangle connects events that can display in the same row. Top: a rectilinear path connects events that need to locate in different rows.

Putting it all together, Figure 3.3 shows an example of a complete SchemaLine visualization. The algorithm to produce this is described in Section 3.4.

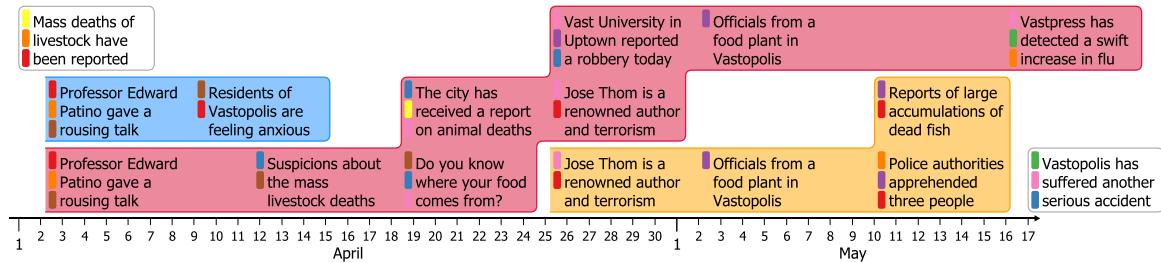


Figure 3.3: SchemaLine visualization of user annotations. Related notes are connected to form schemas.

3.3.3 Interaction

To enable users to intuitively perform sensemaking activities described in the Data-Frame model (Requirement 3), we follow the design guidelines of fluid interaction proposed by Elmquist et al. [48]. More specifically, interactions in SchemaLine

- Produce smooth animated transitions between the state before and the state after an interaction, helping users to maintain their mental maps.
- Provide immediate visual feedback, enabling users to know what is happening and/or what will happen next.
- Manipulate directly on the visual representations of events and frames, instead of using extra menus and buttons.

During sensemaking, when the user recognizes a relationship of events, he or she can group them together and find an account for them (**connect data and a frame**). This activity is performed by dragging one event and dropping it onto another event, resulting a new frame consisting of these two. While dragging over, a *plus* icon and a rectangle with dashed border surrounding the two events are displayed to indicate that a new frame will be created.

The user can also **elaborate the frame** by adding more relevant events. This is simply executed by dropping events onto the colored stripe representing the frame. Conversely, to **preserve the frame**, the user can drag its events and drop them onto void space to remove them from the frame. Appropriate informative feedback is displayed in both cases: a *plus* icon for elaboration and a *minus* icon for preservation. Figure 3.4 shows an example for elaborating the frame.

Questioning the frame occurs when the user encounters inconsistencies in data. The temporal distribution of events in the frame may suggest some concerns about the plausibility and completeness of the frame. For example, if a frame about one

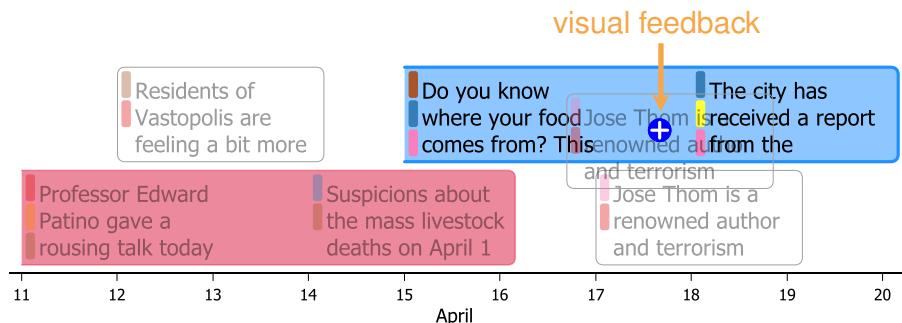


Figure 3.4: Elaborating the frame. Dragging and dropping an event onto the blue stripe to add it to that frame. The plus icon indicates the addition behavior if dropping the event.

person contains many events in January and March, but none in February; it may be inferred that some data might be missed. To highlight a suspected event, the user can double-click on it with the right mouse button. The text of that event will be rendered in red to indicate that it needs more investigation.

Depending on experience, the user can think of multiple explanations for the same set of data. To support **comparing multiple frames**, we enable the user to duplicate events and construct similar frames. By default, dragging an event from one frame and dropping it onto another frame will move it to the new frame. However, holding the *Control* key while dropping an event will instead copy it to the new frame. Also, when two frames are selected, they will be moved vertically next together to facilitate comparison.

The user can remove an event from the timeline by dropping it at the bottom of the time axis. A red *remove* icon is displayed as a visual feedback. While searching for a replacement to account for inconsistent and contrary data (**reframing**), it could be useful to consider discarded data. To enable that, SchemaLine can redisplay events that were removed earlier with half transparency to distinguish them from existing events.

When the user thinks that the existing frame cannot account for its data, he or she may completely discard it and **seek a new frame**. The frame can be removed by dragging it onto void space. However, its events still remain in the timeline, enabling the user to exploit them. Another interaction could be useful is to combine two sets of events together – we call it “merge frames”. This can be performed by dragging one schema and dropping it on top of the other schema.

3.4 Algorithm

This section discusses the algorithm to produce the SchemaLine visualization as in Figure 3.3, consisting of two parts. First, the layout of schemas is generated; and then, their outlines are computed based on the layout information.

3.4.1 SchemaLine Layout

Besides compact and aesthetic, we set these conditions to the layout:

1. **Horizontal position.** Along the time axis, events should be located accurately at when they happen, if possible.
2. **Relative order.** To address scalability, events can be shifted horizontally; however, their relative order should be maintained: $x(e_1) < x(e_2)$ if and only if e_1 happens before e_2 , where $x(e)$ is the horizontal position of event e .
3. **Overlap free.** Events and schemas are not allowed to intersect each other.

To meet these requirements, we design an algorithm consists of the following four steps (Figure 3.5):

1. Order the schemas vertically based on their number of events.
2. For each schema, locate its events satisfying the aforementioned requirements.
3. Compact the schemas following the order computed in the first step.
4. Locate the remaining events that do not belong to any schemas.

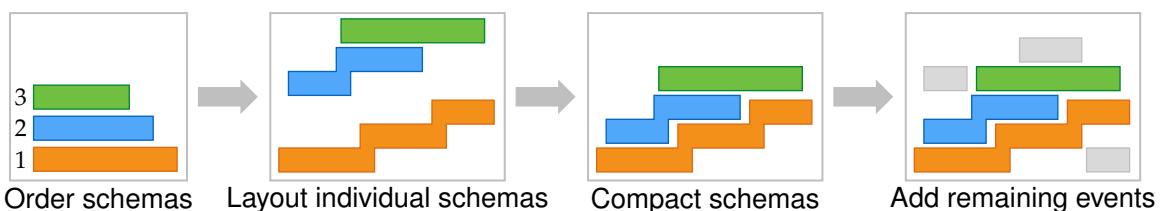


Figure 3.5: SchemaLine layout algorithm. First, the vertical order of schemas is computed. Second, the layout of each schema is generated independently. Third, the schemas are compacted following the computed order. Last, events that do not belong to any schemas are added to the visualization.

3.4.1.1 Order Schemas

As explained in Section 3.3.2, schemas are vertically stacked to apply the *proximity* principle. This step computes the vertical order of all schemas based on their number of events: the schemas with more events are located under the one with less events. This is analogous to that “heavier” schemas are pulled below the “lighter” ones – in terms of the number of events. If two schemas consist of the same number of events, the one with longer time range is located under.

3.4.1.2 Layout Individual Schemas

The second step of the algorithm produces the layout for each schema. Events that are members of multiple schemas are replicated, allowing the layout of each schema to be generated independently. Events within a schema are sorted chronologically and added to the timeline in that order. Because all events have the same height, the layout algorithm only needs to compute the row level and horizontal position of each event.

Initially, an event e_i is located on the same row as the previous one e_{i-1} and at the position proportional to its temporal value (Condition 1). If these two events are separate, the algorithm moves to the next event; or else, two cases will be considered. If e_i happens at the same time as e_{i-1} , it will be located on the upper row and at the same horizontal coordinate as e_{i-1} . Otherwise, e_{i-1} is tentatively shifted to the left to make space for e_i , as discussed next. If the shift is unsuccessful, e_i will be located in the upper row.

Shifting Events To accommodate more events, the accuracy of the horizontal positions of events can be sacrificed. An event can be shifted horizontally to the left to make space for other events. However, an event should not shift too far from its accurate position to avoid misunderstanding from users. We set that shifting limit to the width of the event so that it still covers its time point on the time axis and provides a reasonable indication of its accurate position.

During shifting, it is essential to make sure that events do not overlap each other (Condition 3). Considering an event e_{i-1} is shifted to make space for e_i , if it overlaps with e_{i-2} , that event should be shifted as well. Eventually, all events located on the way of the movement should also be shifted.

It is also necessary to make sure that the relative order between events is still correct after shifting (Condition 2). Otherwise, events with wrong order need to be

shifted as well to reestablish the correct order. Note that if two events happen at the same time, they must be located at the same horizontal position.

An illustration of the layout algorithm for one schema is shown in Figure 3.6.

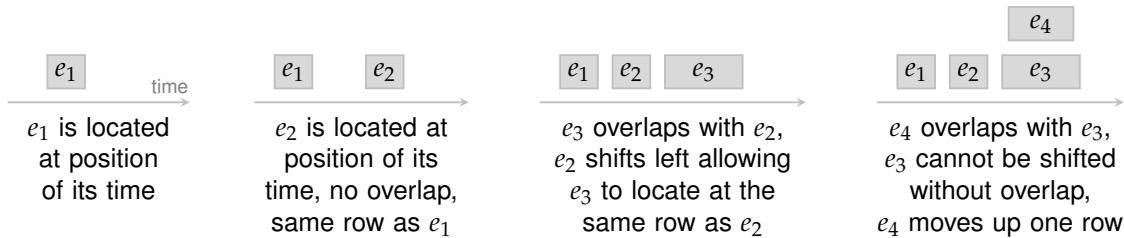


Figure 3.6: Schema layout algorithm. Four events e_1 , e_2 , e_3 and e_4 are added chronologically.

3.4.1.3 Compact Schemas

In this third step, the algorithm stacks schemas in the order computed in the first step to produce a compact and overlap-free visualization. Schemas are stacked upward, starting at the bottom of the display. However, to save vertical space, it is unnecessary to place a schema on top of the previously added schema. A schema can start at the bottom, move upward, and stop when it does not overlap with any other schemas. In the worst case, it will be located at the top of the timeline.

3.4.1.4 Add Remaining Events

This last step allocates events that do not belong to any schemas. Events are sorted chronologically and processed in that order. The ideal horizontal coordinate of an event is the position proportional to its temporal value; however, it can also be shifted using the algorithm described earlier in Section 3.4.1.2. An event begins at the bottom of the display and moves upward until it does not overlap with any other schemas or events after possible shifts.

3.4.2 Schema Outline

In this section, we describe an algorithm to produce a polygonal outline covering all the event rectangles of a schema. We decided to use only horizontal and vertical line segments to keep the outline simple yet aesthetic as in Figure 3.2. The *polygonal path* P_n of a schema that contains n event rectangles R_1, R_2, \dots, R_n , ordered from left

to right, is determined as follows:

$$P_n = \begin{cases} R_1, & n = 1 \\ P_{n-1} \oplus R_n, & n > 1 \end{cases}$$

where \oplus is an operator that *appends* a rectangle to a polygonal path. As described in the layout of individual schema (Section 3.4.1.2), when a new event is added to an existing schema, it either has the same row as the previous event (Figure 3.7a) or one row higher (Figure 3.7b). To produce an aesthetically pleasing path, two other special cases are also considered as described in Figure 3.7c and Figure 3.7d. Technically, a path is represented by a list of vertices and is updated when new events are added. Figure 3.7 illustrates how these vertices are updated, added or unchanged for all four aforementioned cases.

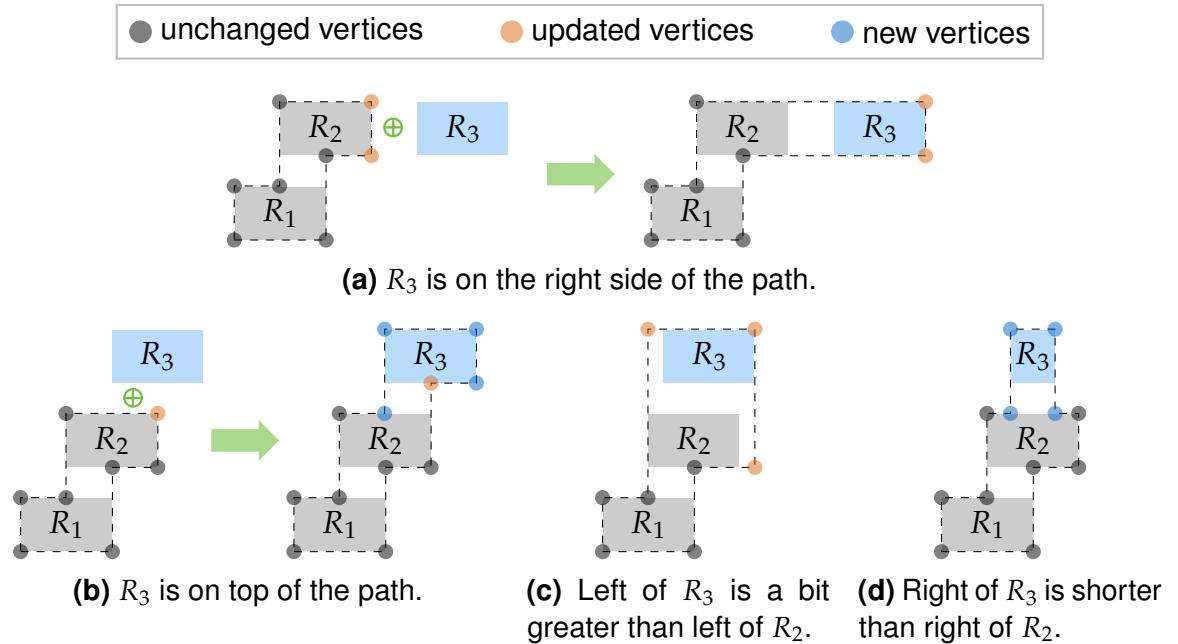


Figure 3.7: Four cases when a new rectangle R_3 is appended \oplus to a polygonal path – representing by a dashed polygon. Vertices of the path are colored coded to describe how they are maintained.

After producing a rectilinear path, all corner bends are made rounded as in Figure 3.3. The path is filled with the same stroke color but less transparency to make the border stand out with a darker hue. The beginning of the path does not have the border to indicate the flow of events within the path.

3.5 Evaluation

We first integrate SchemaLine into an existing visual analytics system, and then evaluate its usefulness in exploring temporal relationship of the sensemaking task.

3.5.1 Application

We integrate SchemaLine into INVISQUE [193] – a visualization system designed for interactive exploration of text documents. INVISQUE provides full-text search and organizes search results in a two-dimensional spatial canvas, with each dimension representing a customized attribute. For example, it may be useful to order academic articles horizontally by publication date and vertically by citation count. Search results from a keyword are shown as a cluster of *index-cards*, each representing a document with selected information such as publication title, date, keywords and authors. Figure 3.8 shows a screenshot of INVISQUE.

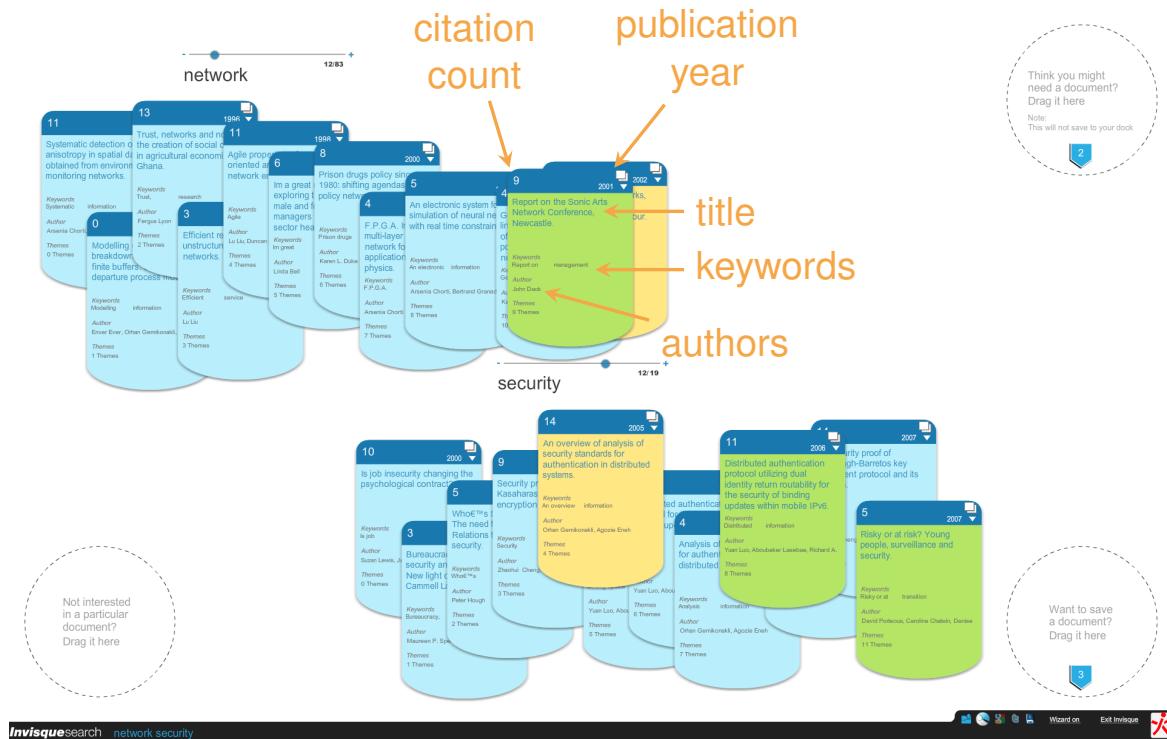


Figure 3.8: INVISQUE interface. It shows two sets of search results for “network” and “security” from a publication dataset.

We add an *annotation* feature enabling users to record their thoughts while reading documents. These annotations are important to users, thus are also displayed on

the index-cards. We choose them as the input *events* of the SchemaLine visualization. SchemaLine is placed at the bottom of INVISQUE. After the user makes a note, or annotation, it is immediately added to SchemaLine as a new event. Double clicking on an event will open the document containing that note as an index-card, enabling the user to quickly reexamine the original information resource.

The *temporal information* of documents, such as “publication date”, is initially assigned to that of events, and can be changed later by users to match the correct meaning. This feature is useful because the report date is not necessarily the same as the date when the event actually occurred. For example, a news article can write about a bomb attack in some city yesterday. The user can perform this change by dragging an event with the right mouse button along the time axis and dropping it at the desired date.

The *label* of an event simply maps to the content of an annotation itself. In INVISQUE, we color code search keywords that contain annotated documents, and use them as *categories* for events. Because a document can be returned from different searches, it can also contain multiple categories. This mapping provides context for the annotations: what did I search for (the original keyword) and what are other related concepts (other search keywords returning the same document)? This context may help users to discover interesting patterns through their annotations. Figure 3.9 shows a screenshot of INVISQUE with SchemaLine integrated.

3.5.2 Case Study

3.5.2.1 Design

Method Evaluating the usefulness of SchemaLine in supporting sensemaking can be categorized as *evaluating visual data analysis and reasoning* – one of the seven scenarios in evaluating information visualization proposed by Lam et al. [101]. The goal of this type of evaluation is to explore whether and how a visualization tool supports participants to make sense and generate relevant knowledge in a domain. During solving sensemaking tasks, the participants may employ various strategies. Their processes and outcomes are highly context-sensitive, making it difficult to quantify and compare their performance. Therefore, sensemaking evaluations are typically case studies with realistic datasets and domain experts as participants. We conducted a case study to explore how SchemaLine supports users to solve a task, focusing on how it enables them to perform sensemaking activities in the Data–Frame model.

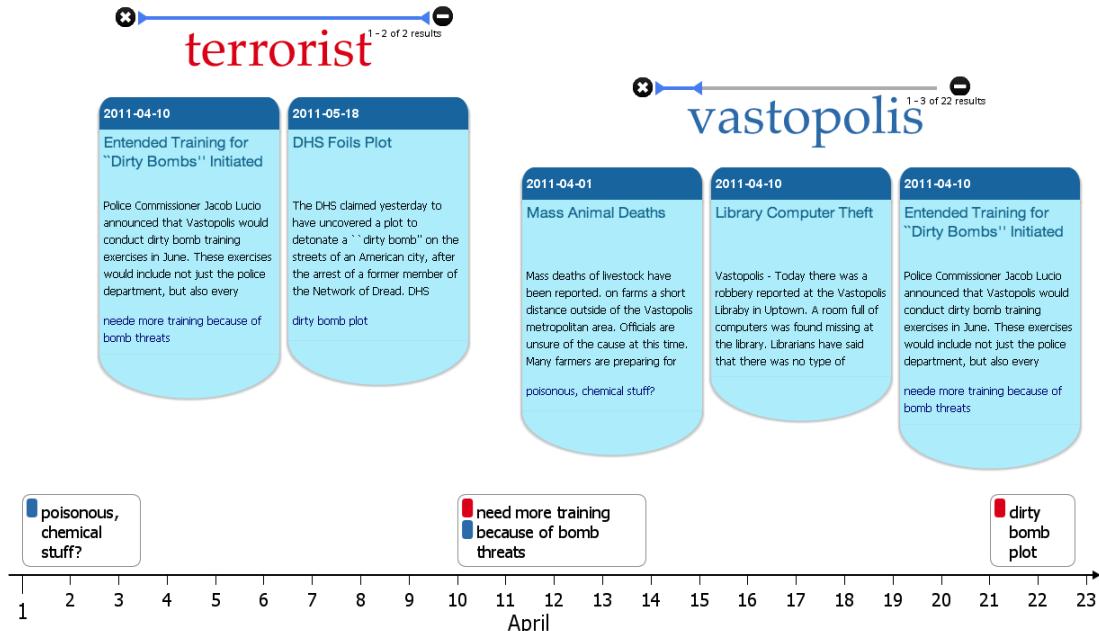


Figure 3.9: INVISQUE with SchemaLine at the bottom. The timeline consists of three events, which are user taken notes. Color coded categories of events indicate keywords that were searched for.

Task We used the task from Mini Challenge 3 of the IEEE VAST Challenge 2011¹, which requires the participants to identify any potential criminal activities from the given dataset. We chose this task because it resembles a real task in a domain (intelligence analysis), and the solution was provided and well-tested by the community. INVISQUE with SchemaLine integrated was used in the study.

Participants and Procedure We were unable to recruit real intelligence analysts for the study. Instead, we recruited three graduate students with different backgrounds: one in visual analytics (surrogate for visualization expert – P1), one in law (surrogate for domain expert – P2), and one in networking (neutral background – P3). After being introduced features of INVISQUE and SchemaLine, participants had a chance to practice with a trial sensemaking task for 15 minutes. The main task was followed and lasts for one hour. The participants were asked to report the criminal activities they had discovered with the supporting evidence. Semi-structured interviews were followed to gain deeper understanding of the processes.

¹<http://hcil2.cs.umd.edu/newvarepository/VASTChallenge2011/challenges/MC3-InvestigationintoTerroristActivity/>

Pilot Tests We conducted two pilot tests to assess the study design. The original dataset contains more than four thousand news reports, many of which have more than 500 words. In the pilots, both participants failed to find any answers after one and a half hours. Most time was spent on reading long but irrelevant documents. The reason could be INVISQUE does not support text-mining features such as entity extraction, which is crucial in analyzing a large document collection. The goal of the evaluation was to assess how SchemaLine can provide additional sensemaking support to INVISQUE rather than assessing INVISQUE itself. Therefore, in the main study, we removed all irrelevant documents that are not part of the ground truth, which actually added by the organizers to challenge contestants. The new dataset contains 36 documents including five criminal activities: food poisoning (13 documents), hacking (3), dirty bomb (6), arms trafficking (4), and money laundering (3). Other documents are isolated cases and are not considered as correct answers. We expected that participants could complete the task within a reasonable amount of time, without affecting the goal of the study.

3.5.2.2 Results and Discussion

We first summarize the three sessions and present our findings next.

Participant 1 P1 began searching for “bomb”, examined the search results, and searched for a refined keyword “dirty bomb”. He took notes in three documents and then linked these notes together (*connect data and a frame*). He then searched for “Network of Dread”, which was mentioned in one document as the author of the dirty bomb attack. He took a note in the new returned document and dropped it onto the “dirty bomb” colored schema (*elaborate the frame*). While investigating, he encountered an article about a man carrying a frozen turkey having wires coming out of it, which was suspected as a bomb. At first, he dropped the “turkey bomb” note onto the “dirty bomb” schema. Then, he wondered whether it is a real bomb. After thinking for a while, he removed it out of the schema (*preserve the frame*). P1 found the “dirty bomb” attack with 4/6 correct pieces of evidence. P1 took many notes in documents related to the “food poisoning” case; however he could not link them together because he said that “I’m not familiar with bio-attack so I couldn’t think of it as a threat”.

Participant 2 P2 took an overview step before searching. He quickly looked at all 36 document titles to have a glimpse of the dataset as well as to detect potential

search keywords. He began searching “animal deaths”, read the results, took notes and grouped them together (*connect data and a frame*). He was happy with the evidence he found for that crime and switched to read another interesting article “Library Computer Left” he came across. From that, he searched for several related terms such as “computer” and “hackers”. He figured out that a group called “F-alliance” stole computers from the library and attempted to hack a bank. He dropped a “computer stolen” note on top of a “bank hacking” note to form a new explanation for the case (*connect data and a frame*). He found another article related to hacking but he said “I won’t drop it to this group because it’s just an announcement from the government about potential threats” (*preserve the frame*). During further investigation, he created another group of notes related to “bioterrorism” and “Prof. Patino”. Then, when figuring out that the reason of the mass deaths is a spore-forming microbe, which is also mentioned in Prof. Patino’s talk, he dropped that new group onto the “animal deaths” group to combine all notes together because he thought that they are related (*merge frames*). Observing the order of events in the new group on the timeline, he said “The equipment of Patino was stolen after the animal deaths report, so they couldn’t be used in that case. This is the group of a potential threat in using bioterrorism” (*elaborate the frame*). **P2** found the “hacking” case with 2/3 correct pieces of evidence and the “food poisoning” case with 9/13 correct pieces of evidence. Figure 4.13 shows the computer screen of **P2** when he reported his findings.

Participant 3 **P3** searched for a few keywords related to criminal activities before examining the search results such as “bomb”, “terrorism”, “money” and “hack”. He took and group notes about “money laundering” together (*connect data and a frame*). Then, he read articles from “terrorism” search results. He followed the article content to search for relevant information such as “Paramurderers of Chaos” – a terrorist group. During further investigation, similar to **P2**, he also combined two groups of notes – “Paramurderers of Chaos” and “food supply” – together when discovering evidence linking the two groups (*merge frames*). When representing his findings, he shared that SchemaLine prompted him to look for missing information. “I noticed the gap between these two events [pointing to the timeline]; then I knew I probably missed something there” (*question the frame*). **P3** found the “food poisoning” case with 6/13 correct pieces of evidence, and a perfect 3/3 pieces of evidence in the “money laundering” case.

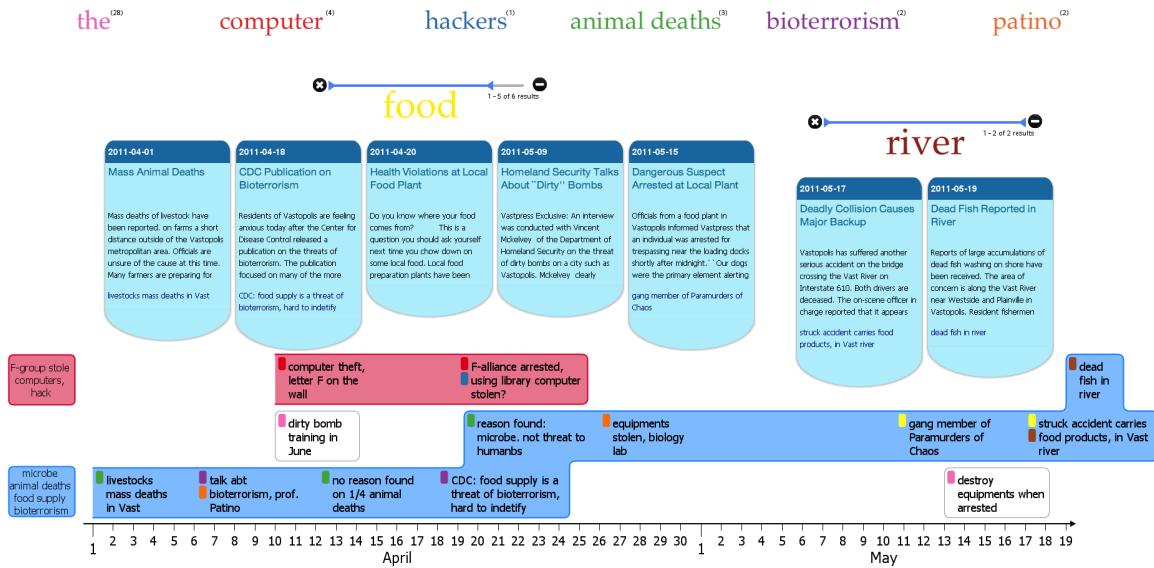


Figure 3.10: Final screen of participant P2. Top: a trail of his keyword searches, collapsed after being read. Middle: search results in index-card metaphor. Bottom: two schemas containing notes as supporting evidence of criminal activities he found.

Discussion Three participants applied different sensemaking strategies. **P1** started with a potential search keyword for criminal activities and kept following the search results. **P2** scanned the titles of all documents to have an overview of the dataset first. **P3** planned ahead what he wanted to search for and sequentially executed it. However, all of them extensively took notes and constructed frames to organize them. These frames presented various forms: a concept (bioterrorism), a criminal activity (dirty bomb), a person (Prof. Patino) and a group of people (Paramurderers of Chaos). All participants also employ a variety of sensemaking activities described in the Data–Frame model, supported through fluid interaction in SchemaLine.

All participants appreciated the automatic addition of user notes to the timeline. **P1** thought that he would have a problem if the system did not support that: “I can remember what happened but it was difficult to remember when they happened”. They found that the chronological order of events helps them to construct the story lines. **P2** shared that he read the news about the robbery at Vastopolis university and the Prof. Patino’s talk about bioterrorism. However, he did not have any insight at that time. When looking at his two notes on the timeline, he thought that the extremely expensive equipment in Prof. Patino’s lab could be the reason of the robbery.

All participants commented that the interaction between data and frame is very intuitive. **P1** even said “I think I don’t even need training and still can figure out

how it works". **P3** liked the transition effect when adding or removing notes because "it helped me to understand what is going on". All participants were confident when presenting their analyses. **P3** even opened the original document (double-clicking on the note) several times to highlight the relevant text. He said that the connection between the note and the containing document enabled him to quickly find the information source when needed.

3.6 Summary

In this chapter, we introduced SchemaLine, a novel timeline visualization, enabling users to explore temporal relationship through their annotations. It supports the schematization process in the Pirolli-Card's sensemaking model and facilitates all sensemaking activities described in the Data–Frame model through fluid interaction. The visual design makes it easy to follow data within a frame, and the algorithm produces simple, compact, but aesthetically pleasing visualization.

We conducted user-centered evaluation to explore how SchemaLine is used in sensemaking. All participants thought that the tool was intuitive to use and provided necessary support to them for solving the task. They extensively took notes, constructed temporal frames to organize them and elaborated the frames when discovering more evidence. They were also confident in presenting and defending their findings using the created frames.

SchemaLine suffers from two limitations. First, events that belong to multiple schemas are currently replicated, which is space-inefficient and may confuse users. More advanced layout needs to be investigated to address this issue. Second is the scalability issue with the relatively small number of events that SchemaLine can show. Suitable aggregation may help lessen this limitation.

4

Exploring Complex Temporal Relationship

Chapter 3 presents a timeline visualization technique, SchemaLine, to explore the temporal relationship of sensemaking through the high-level analytic provenance manually captured by users. However, SchemaLine cannot show events belonging to multiple schemas, or *sets* for generality, limiting users from exploring complex temporal relationship. In this chapter, we introduce a novel technique – *TimeSets* – to address that issue. TimeSets visually groups events that belong to the same set but still preserves their temporal order. It color codes the backgrounds of the entire sets to distinguish them and uses colored gradient backgrounds for the intersections among those sets. It also addresses the scalability issue in SchemaLine by dynamically adjusting the level of detail for each event to suit the amount of information and display estate. A controlled experiment was conducted to evaluate its effectiveness by comparing it to a state-of-the-art method. The results showed significant advantage in accuracy and user preference.

4.1 Introduction

Timelines have been applied in many domains including personal biographies [138], medical records [139], music [10] and historical events [149]. Events in a timeline are commonly categorized into groups or *sets*. For example, academic publications often cover one or many disciplines; similarly, news articles fall into different categories such as politics and sports. Back in 1765, one of the oldest documented timelines

Previously published at the *Information Visualization* journal [124].

produced by Joseph Priestley – the Chart of Biography [141] – already denotes grouping of elements. The timeline includes two thousand famous persons from 1200 BC to 1800 AD, who were classified six categories based on their most well-known achievement. The timeline is divided into six horizontal bands, one for each category, to visualize the set relations.

Timeline visualizations use icons to indicate time-point events [1] and horizontal bars for interval ones [138]. To show set relations, existing methods either color-code the icons or use different shapes [4]. The layouts of these methods simply focus on avoiding overlapping between visual elements such as icons and text [1, 4]. As a result, events in the same set are not always located close to each other. This makes it difficult to follow them chronologically or have an overview of the distribution of events in a set. Another common approach is to visually connect events in the same set [99]. Such a method can introduce extra edges and crossings, which hampers the readability of the timeline.

There has been considerable work on set visualization, which commonly uses closed contours as in Venn or Euler diagrams. Texture and color can be used to depict more complex set relations [185]. However, these cannot be applied in timelines because the horizontal positions of events are fixed. Techniques that visualize set relations of data items with fixed locations could be good alternatives. To connect same-set elements, Bubble Sets [33] draws an iso-contour surrounding them, and LineSets [7] uses a Bézier curve passing through all the elements. KelpFusion employs both lines and areas to connect elements, and has been shown to have a significant advantage in readability tasks when compared to Bubble Sets and LineSets [114]. However, simply applying these methods on top of existing timelines could introduce many crossings between text and visual elements of sets that may reduce readability.

In this chapter, we propose a novel timeline visualization technique – TimeSets – to effectively displays both temporal and set information in data. More specifically, TimeSets

- Clearly shows the events within a set over time and their relationships with other sets.
- Dynamically adjusts the level of details of each event to suit the amount of information and display estate.
- Uses color gradient backgrounds for events belonging to multiple sets and curved set outlines to emphasize its grouping.

To show possible applications of TimeSets, we discuss two case studies with publication data and with intelligence analysis. Also, a controlled experiment was conducted to evaluate the effectiveness of TimeSets. The results showed that TimeSets was significantly more accurate than KelpFusion [114] – a state-of-the-art set visualization method, and was the preferred choice by the participants for aesthetics.

4.2 Related Work

4.2.1 Set Relations in Timelines

As presented in the Literature Review chapter, Figure 2.2.3.3, Gestalt principles are often used to represent set relations among elements. This section discusses the application of those principles in visualizing set relations of events in timelines.

The principle of *similarity* states that objects are perceptually grouped together if they are similar to each other. This principle is extensively applied to show set relations in timelines by using colors and shapes. Time indicators as icons for time-point events [1] and bars for interval events [183] are colored according to event set memberships [1]. Different shapes for icons [4] and bars [139] are also used to distinguish set memberships. It is more challenging to represent multiple set memberships. LineSets [7] uses concentric circles for icons, where each circle is colored to represent one set.

According to the *proximity* principle, objects that are close together are perceived to be more related than objects that are spaced further apart. In the Chart of Biography [141], people within a category are placed in a horizontal band, away from people in other categories. LifeLines [139] splits medical records into different sets, such as *medication* or *diagnosis*, and places them into vertically stacks, which works well if no two sets overlap. Storyline visualizations [104, 168] use curved lines to show interactions among characters within the movie timeline. Character lines converge to a bundle if they appear in the same interaction, and diverge when the it ends. Each line can be considered as a set passing through all of its members, and each interaction is a multi-set event. Thus, this method only works for interval events.

Elements tend to be grouped together if they are visually connected. Following this *uniform connectedness* principle, tmViewer [99] links related entities with line segments. Different line colors, thicknesses, and styles were used to distinguish

set relations. This method can show events with multiple set memberships by connecting them with multiple edges. However, extra edges and crossings may negatively impact the readability of the timeline.

When similarity and proximity are applied together, the later principle dominates [185]. Moreover, uniform connectedness is stronger than proximity [129]. For example, objects with different colors and shapes but located close together are more likely to be perceived as a group, and distant objects but with a closed contour surrounding them also provide a strong sense of grouping. Applying these ideas to visualize set relations for timelines, methods relying on similarity such as colored icons [183] are less effective than spatial grouping methods such as LifeLines [138]. And those, in turn, are less effective than methods using line segments such as tmViewer [99].

4.2.2 Set Visualizations

Sets and their relationships can be visualized using Venn [150] or Euler [147] diagrams. Simonetto et al. [161] proposed a technique to automatically visualize sets that were previously not possible with Euler diagrams. However, the complex shapes it produces may reduce visualization readability. In their controlled study, Henry-Riche and Dwyer [77] showed that for complex set intersections, duplications of shared elements resulted in a better performance in readability tasks than a none-duplicated visualization with more complex shapes. A state-of-the-art report by Alsallakh et al. [8] provides a comprehensive survey of set visualization techniques. In this section, we discuss a few techniques that can be applied atop elements with fixed positions so that they can be used to visualize set relations for timelines.

Techniques without such constraints include Bubble Sets [33], LineSets [7], and KelpFusion [114]. These methods employ the connectedness principle of the Gestalt laws [129] by connecting set elements using extra visual elements. Bubble Sets draws an iso-contour surrounding elements within a set. This iso-contour is filled with a semi-transparent color so that the intersection between sets is shown as an area of blended color. Collins et al. [33] provide an example of applying Bubble Sets to a timeline, in which case a force-directed algorithm is used to adjust the vertical positions of elements while the horizontal position along the time axis is fixed.

LineSets applies a Bézier curve to connect data items. The curve follows the shortest path passing through all elements in the set. Its study showed that LineSets outperforms Bubble Sets in certain readability tasks [7]. KelpFusion, a hybrid technique, uses lines for data-sparse areas and surfaces for data-dense areas. The results

of an evaluation on readability tasks [114] demonstrated that it outperforms Bubble Sets in both accuracy and completion time, and outperforms LineSets in completion time. There has been no reported attempt to apply LineSets or KelpFusion to timeline visualizations. It is expected that crossings between lines, areas and the event text may reduce the timeline readability.

4.3 Visual Design

4.3.1 Event

An event is represented as a line of text – *label* – summarizing its content, and a glyph indicating its temporal information. For a time-point event, a *circle* is shown at the left of the label. For an interval event, a *bar* is shown at the top of the label. When two interval events overlap, their time bars are displayed with half transparency to make the intersection visible. To accommodate a large number of events, labels have three possible levels of detail:

1. **Complete.** The entire label is shown.
2. **Trimmed.** Only the first few words are shown and ended with three dots (...) to indicate that the visible label is incomplete.
3. **Aggregated.** Events are grouped and labeled with the total number of them. A colored border is added to the label to make the aggregate more noticeable. The time bar of an aggregate spans the starting time of its earliest event and the finishing time of its latest event.

Figure 4.1 shows examples these different visual representation of events.

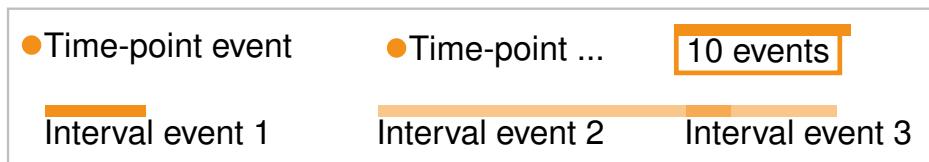


Figure 4.1: Visual representations of events. Top row, left to right: a complete time-point event, a trimmed time-point event, and a aggregate of 10 events. Bottom row, left to right: an interval event, and two overlapping interval events.

4.3.2 Set

4.3.2.1 Design Overview

Similar to SchemaLine (Section 3.3.2), TimeSets also applies the two most powerful Gestalt principles of grouping in its design: *connectedness* and *proximity*. Events belonging to the same set are located close together, and the background of an entire set is colored to make its events visually connected. Spatial grouping needs to be achieved through vertical positioning because the horizontal position of each event is already determined by its temporal information. Sets are stacked vertically, and each set is further divided into a maximum of three *layers*: the top and the bottom layer for events shared with the set above and below respectively (if they exist), and the middle layer for other events in the set. Figure 4.2 shows an example of a layering for three sets.

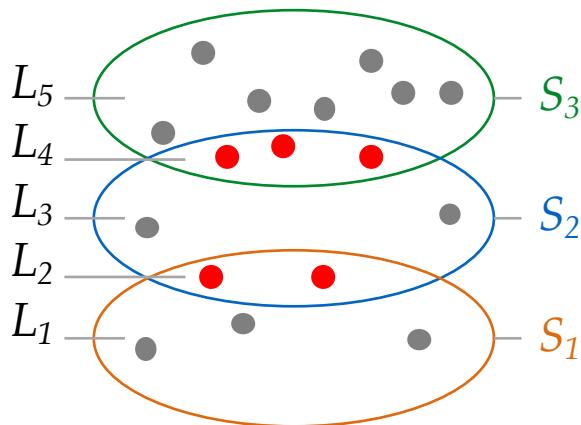


Figure 4.2: Layering for three sets S_1 , S_2 , and S_3 . L_2 includes events shared by S_1 and S_2 , and L_4 includes events shared by S_2 and S_3 . Shared events are shown in red. S_2 consists of events in three layers L_2 , L_3 , and L_4 .

Shared events between two non-neighboring sets can reside in one set and connect to the other set using visual links such as curves [7] or areas [114]. Figure 4.3a shows an approach to connect shared events (red squares) using straight edges and link them to the orange set to indicate that they also belong to that set. An alternative approach is to duplicate shared events in both sets. In Figure 4.3b, red squares are duplicated in both the green and orange sets. Duplication consumes more display space and could make viewers confuse when seeing same events multiple times. However, it ensures all events of the same set being located close together, which makes the visualization compact. Also, a study by Henry-Riche and Dwyer [77] shows that complex set-intersection shapes reduce readability compared to item

duplication. Aiming for a clear visualization, which is crucial for interactive set construction, we decide to duplicate events that belong to non-neighboring sets. Confused duplication and scalability will be addressed later using interaction and layout algorithm respectively.

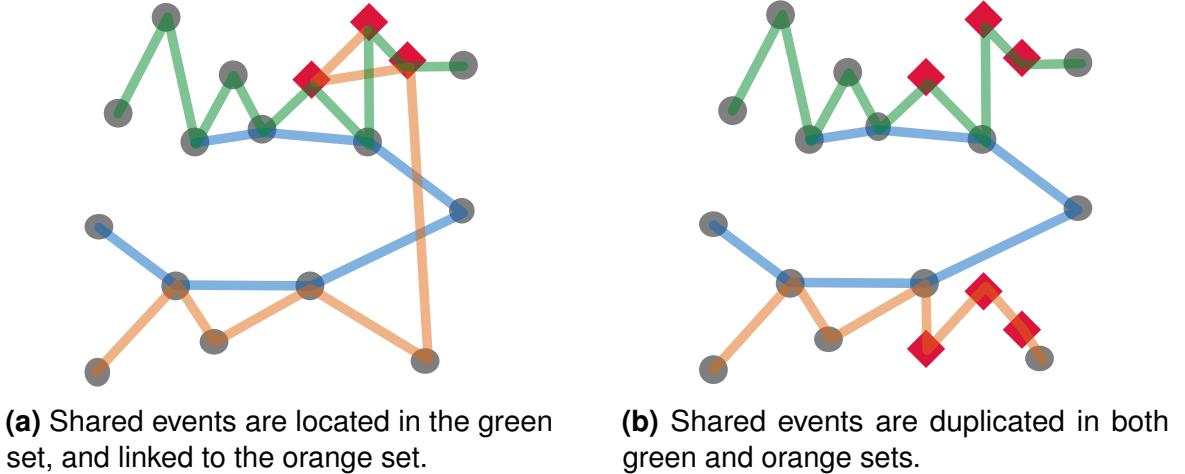


Figure 4.3: Visualizations of shared events between two non-neighboring sets (red squares).

In subsequent sections, we discuss the detail of the set visualization algorithm, which consists of two main steps: generating set shapes, and then coloring them.

4.3.2.2 Shape Generation

This algorithm takes as input a list of bounding-boxes of all events in a set, and generates a closed-curve containing all these boxes. The sizes and positions of the bounding boxes are decided by the layout algorithm described in the next section. A rectilinear shape can be generated using a scan-line algorithm [50], as shown in Figure 4.4a. The number of bends along the border is often used to assess the aesthetics and legibility of visualizations [168]. Even though the generated shape provides the minimal *data-ink* ratio [176], a large number of line bends may reduce its legibility.

To reduce the number of line bends, the top and the bottom sides of the set outline are flattened. The left and right sides are kept unchanged because they indicate the temporal information of events. On both sides, the outline can be “jagged” if two events start or end close to each other. Those close vertical segments are combined to reduce line bends if their horizontal gap is smaller than some threshold. This trades off time accuracy for outline smoothness and can be adjusted by the user. Figure 4.4b shows the result of this simplification.

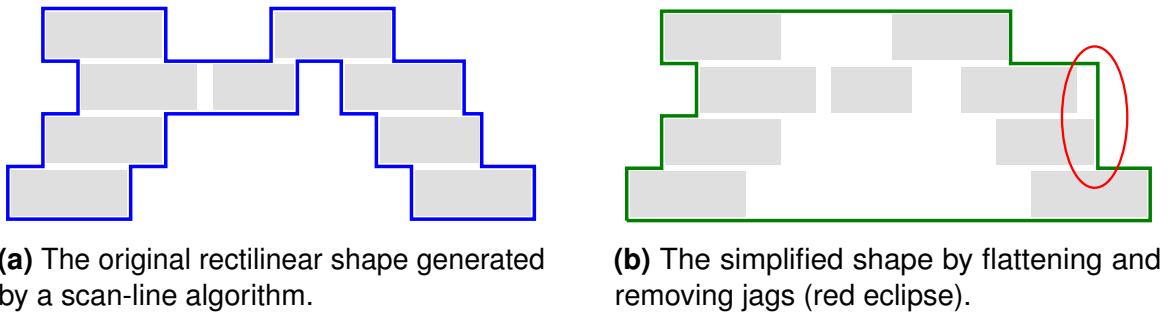


Figure 4.4: Rectilinear shape generation.

To reduce the degree of line bends, vertical segments are converted to diagonal ones wherever possible, such as e_2 and e_3 in Figure 4.5a. Smoother lines are easier for users to follow [93], thus diagonal segments are further converted to Bézier curves, and squared corners are replaced by quadrant arcs as in Figure 4.5b.

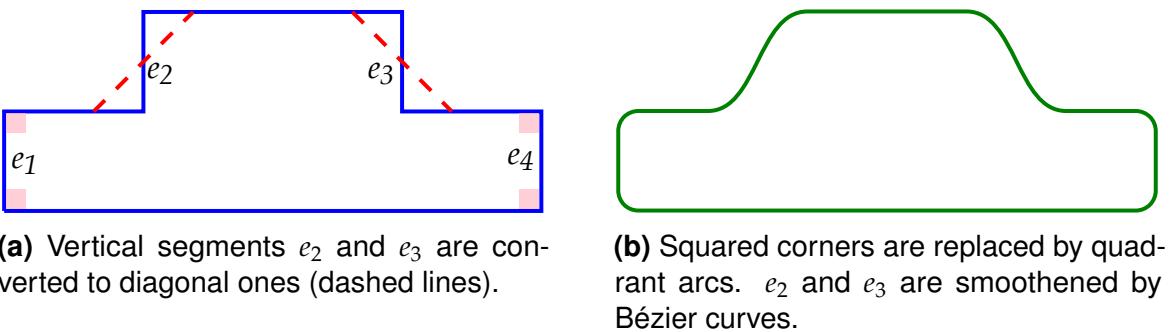


Figure 4.5: Shape smoothening by reducing the degree of line bends.

4.3.2.3 Set Coloring

Each set is filled with a color selected from Qualitative Set 2 of ColorBrewer [67] to make them easily distinguishable. Two color filling options are considered: only the time circle and the entire label. Our design follows uniform connectedness principle requiring visual connection among same-set events. When they are visually connected and only their time circles are filled, intersection between edges and text may reduce the readability of the visualization. Figure 4.13b shows an example of KelpFusion [114] using this approach. In the second option, filling the entire label may produce a false understanding about the temporal information of events. We choose this option and lessen the effect by coloring the gap between events as in Figure 4.13a. It also helps increase the sense of grouping compared to filling only the time circles.

One common coloring method for set intersections is *color blending* as used in Venn diagrams [185]. Color for each set is half-transparent, and alpha blending is applied to produce a new color for the intersection. However, the output color may look irrelevant to the two input colors and may be confused as the color for a new set rather than the intersection part.

To address this issue, we fill the intersection with a linear color gradient changing between the two set colors as in Figure 4.6a. While the gradient provides a smooth transition, it becomes difficult to recognize the two ends of the intersection. For example, it is not clear from Figure 4.6a that the background of the event “Rove’s 4th grand jury appearance” (the second row from top to bottom) is pure yellow or it has a mix of green as well. To solve this problem, multiple color transitions are used instead of a single transition. For instance, in Figure 4.6b, the color transitions between green and yellow are repeated multiple times so that both colors are clearly shown in every row of the intersection.

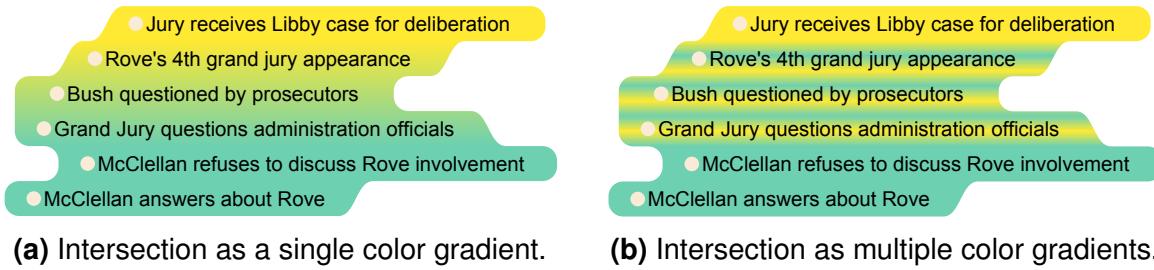


Figure 4.6: Visual representations of set intersections using color gradient.

4.3.2.4 Multiple-set Events

With the vertical layering of sets as discussed earlier, three sets cannot be placed adjacently; therefore, it is unable to visualize intersections among three sets or more. This is also a challenging problem with other state-of-the-art methods [8]. To address this issue, similar to non-neighboring sets, we replicate events for each set that they belong to so that all events in the same set stay close together producing a compact visualization. To provide full set memberships of events, one method is connecting all replicates of the same event using edges. However, this may produce a cluttered visualization with many edge crossings. Another method is to color code the event according to its set memberships. The first option is to color the event time circle using either multiple circles (Figure 4.7a) or concentric rings (Figure 4.7b). The former requires more horizontal space, whereas the latter needs more vertical space.

Another option is to color the background of the event label. Color gradient is used for a smooth color transition as in Figure 4.7c. This visual encoding is consistent with the use of color gradient to show two-set intersections. However, a timeline with many long-label events may produce a too colorful and distracted visualization. Also, limited label height may hamper the detection of color transition. To solve these problems, color is transitioned from left to right, and only run through a first few characters of the event label (Figure 4.7d). Figure 4.16 shows this technique in a visualization of 200 events.

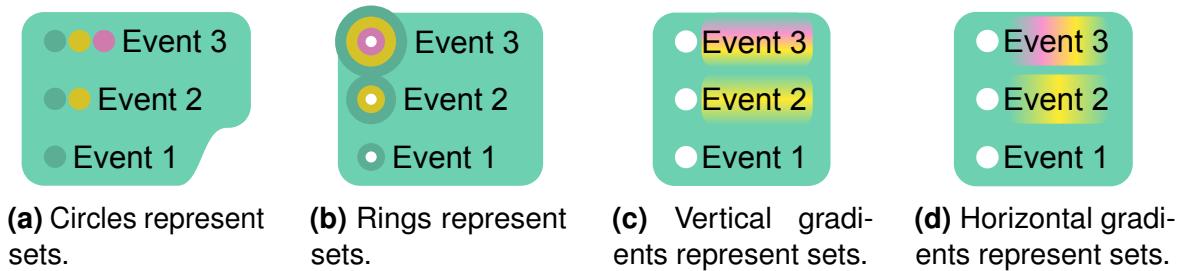


Figure 4.7: Visual representations of multiple-set events. Event 1 is single-set (green). Event 2 is double-set (green, yellow). Event 3 is triple-set (green, yellow, pink).

For interval events, only coloring the background of labels can be used because they do not have time circles, which can be added but at the cost of extra display space. Time bars can be used to show set memberships by dividing into multiple horizontal parts, each color for one set. However, this could be misinterpreted as an event having different set membership in each part of its timespan.

Putting it all together, Figure 4.8 shows a TimeSets visualization with the CIA leak case dataset¹, in which the identity of CIA operative Valerie Plame was made public. Several interesting patterns can be observed in this figure. First, there are many more events related to “White House” compared to other topics. Second, among “Judges, Courts” events, those are related to “White House” are more than those related to “New York Times”. Third, events about “Wilson” only appear at the beginning of the case, while “Judges, Courts” events appear later. The algorithm to produce this visualization is described in Section 4.4.

4.3.3 Interaction

Interactive features are implemented to support timeline exploration. Mouse hovering an event reveals its temporal information and the complete label. When none of

¹<http://www.npr.org/templates/story/story.php?storyId=4764919>



Figure 4.8: TimeSets visualization of the CIA leak case. The timeline contains events that happened from 2002 to 2007, each includes a timestamp, a label, and topics such as “White House”. Events are positioned along the horizontal time axis based on their temporal values, and vertically grouped by colored topics (see the color legend in the bottom right corner).

the multiple-set visualization techniques proposed earlier is used to statically display the full set memberships of an event, it is possible to use interaction to reveal that information. When an event is hovered, all of its replicates are highlighted, allowing easy examination of its full set memberships. This method prevents adding extra ink to the visualization; however, it requires users to discover the set information manually.

TimeSets provides interactive set filtering and focused time window changing via zooming and panning. Clicking on a set in the legend (bottom-right corner in Figure 4.8) toggles its visibility. Time zoom is performed via the mouse-wheel button and pan is controlled by dragging the left mouse button. Users can also interactively modify set ordering by changing the order in the legend through drag-and-drop. A smooth animated transition is provided for all the interactions to help users maintain their mental maps [48].

4.4 Layout

The layout algorithm, determining the positions of sets and events within them, consists of four steps. First, the vertical ordering of sets is computed to ensure that two sets that share events are next to each other wherever possible. Then, sets are further divided into layers, and events are assigned to these layers according to their memberships. After that, the position and length of each event are computed, within the given display space. Finally, layers are compacted to remove any gaps between

them, before their sizes are adjusted to yield a consistent level of detail across all sets.

4.4.1 Sets Ordering

The set ordering algorithm aims to maximize the number of events shared by neighboring sets. This can be mapped to a graph path problem. Given a list of sets $S = \{s_1, s_2, \dots, s_n\}$, an undirected graph $G = (V, E)$ is created with each vertex v_i representing a set $s_i \in S$. Two vertices v_i and v_j are connected if s_i and s_j share an event. The weight of edge e_{ij} is the number of events shared by s_i and s_j . Finding a set order with the maximum number of events shared by neighboring sets is equivalent to finding a path with the maximum weight connecting all vertices in G . This longest path problem is known to be NP-hard. The number of sets we plan to support is constrained by the number of colors that human can easily distinguish when they are shown together; and they are only around 12 colors [119]. Therefore, we decide to use a brute-force approach to find the optimal solution.

4.4.2 Layer Layout

This algorithm positions all the events within a layer. Its inputs include:

- The events belonging to the layer with their *label* and *time* values.
- The maximum width and height of the layer.

The outputs are event locations within the constrained display area, optimizing for the following criteria:

Completeness measuring how much event labels are visible. More specifically, we define the *completeness ratio* as: $\theta = \frac{\alpha \cdot |E_c| + \beta \cdot |E_t|}{|E|}$, where $|E_c|$ is the number of complete events, $|E_t|$ is the number of trimmed events, and $|E|$ is the number of all events. α and β are the coefficients to indicate how strongly complete events and trimmed events contribute to the overall content richness of the layer, respectively. We practically set $\alpha = 1$ and $\beta = 0.5$.

Traceability measuring how easy it is to follow the events within a layer chronologically. Events happened close in time should have small changes in their row levels to facilitate the reading flow. More specifically, we define the *traceability*

ratio as: $\gamma = \frac{\sum_{i=1}^{|E|} (l_{i+1} - l_i)}{|E|-1}$, where $|E|$ is the number of all events within the layer and l_i is the row level of event e_i .

The horizontal position of an event is fixed by its time. The layout algorithm decides on which row to position an event; i.e., vertical position, and the level of detail for its label.

4.4.2.1 Completeness Algorithm

Starting with an empty layer, the algorithm processes events chronologically. An event is located at the possibly lowest row where it does not overlap with any other events. If such a row does not exist, one of the earlier located events is trimmed to make space for it. Among these events, the one with the least text being trimmed is selected. However, if the label space of that event is too short for a single word after trim, it will combine with the current event to form a new aggregated event labeled “2 events”. Aggregated events cannot be trimmed, thus if a new event overlaps with them, it will be added into the existing aggregate. For example, a new event that overlaps with a “2 events” aggregated event will be grouped together producing the “3 events” aggregate.

The completeness algorithm maximizes the number of complete events $|E_c|$ and trimmed events $|E_t|$, thus yielding a maximum completeness ratio θ . However, this algorithm does not optimize traceability because an event is located in the possibly lowest row disregarding the row level of its preceding event.

4.4.2.2 Traceability Algorithm

To improve traceability, the algorithm inserts a new event at the same row as its preceding event. If they overlap, the preceding event is trimmed to make space for the currently adding one. We define the *trim ratio* of an event as the ratio of the remaining text length to its original length. An event can only be trimmed if the resulting trim ratio is greater than a minimum threshold t_{min} , where $0 \leq t_{min} \leq 1$. This value determines how much completeness can be traded for traceability. If the resulting trim ratio is smaller than t_{min} , the event moves up or down, up to r_{max} rows on both sides, to find a satisfied row. r_{max} decides how far, in terms of row level difference, an event can be from the preceding event, which essentially trades traceability for completeness. If no suitable row can be found within $\pm r_{max}$ rows,

the currently adding event returns to the level of its preceding event, which is then trimmed or aggregated with the preceding event as in the completeness algorithm.

Figure 4.9 shows an example of these two algorithms. Both layer layout algorithms run in linear time in terms of the number of events, because during the event insertion, the completeness algorithm checks up to a constant number – the layer height – of times, and the traceability algorithm checks at most $(2 \times r_{max} + 1)$ rows.

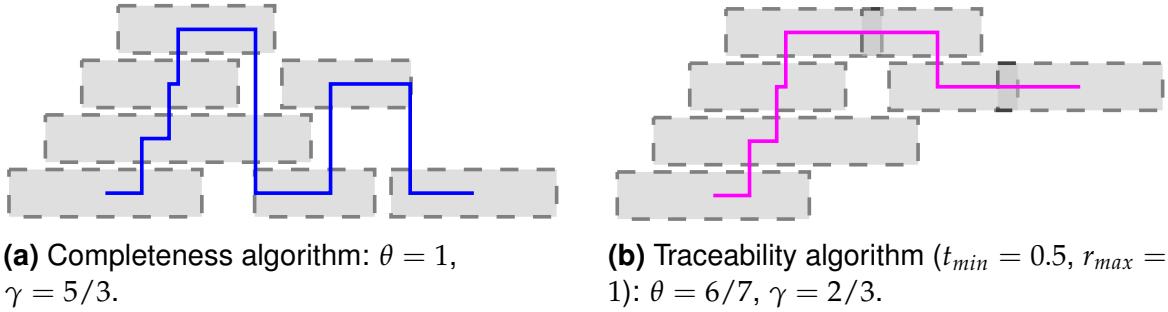


Figure 4.9: Layer layout algorithms. Each rectangle represents an event. The line connecting centers of rectangles illustrates the traceability.

4.4.3 Layers Compacting

The aforementioned layer layout needs a maximum number of rows it can use as an input. Initially, it is assigned proportionally to the number of events within each layer. After the layout of each layer is independently computed, some sets may need to move to fill the new space that appears between layers. This includes moving two layers closer if there is a gap in between, or moving a layer into a newly created space if its set does not share events with any other sets. The freed space is assigned to the layer with the lowest completeness ratio θ . Then, layouts of all layers are recomputed and compacted again. The process repeats until no more space can be saved. Figure 4.10 shows an example of compacting.

4.4.4 Layers Balancing

This last step ensures that all layers have similar levels of detail; i.e., avoiding layers with many complete events and other layers with many aggregated events. This is

achieved by minimizing the variance of completeness ratios of all layers, $\frac{\sum_{i=1}^n (\theta_i - \bar{\theta})^2}{n}$, where n is the number of layers and $\bar{\theta}$ is the mean of all completeness ratios. A brute force approach tests all possible combinations of layer height h_i such that

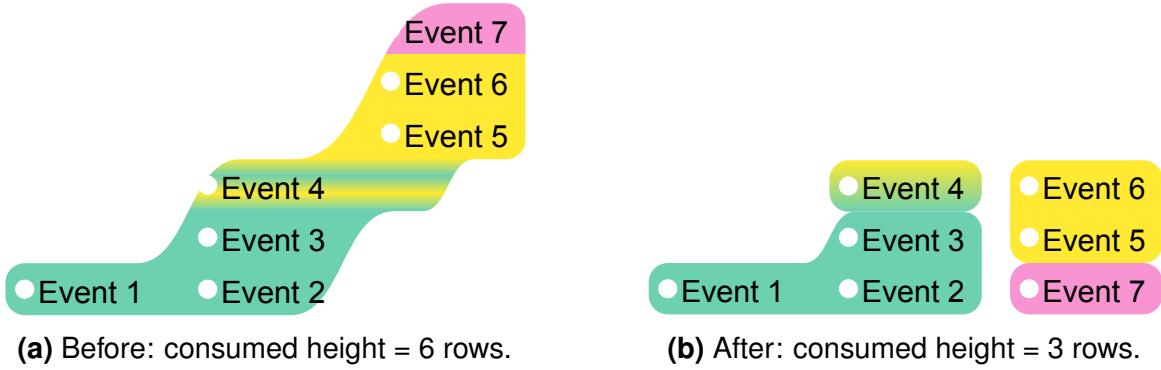


Figure 4.10: Layers compacting.

$\sum_{i=1}^n h_i = H$ for a minimum variance, where H is the height of the display area. However, the number of combinations is an exponential of n . Instead, we apply a heuristic algorithm that relies on a simple observation that the completeness ratio increases with layer height. Therefore, the algorithm reduces the completeness ratio variance by iteratively transferring a row from the layer with the largest ratio to the layer with the smallest one, until the variance no longer decreases. Figure 4.11 shows an example of balancing.

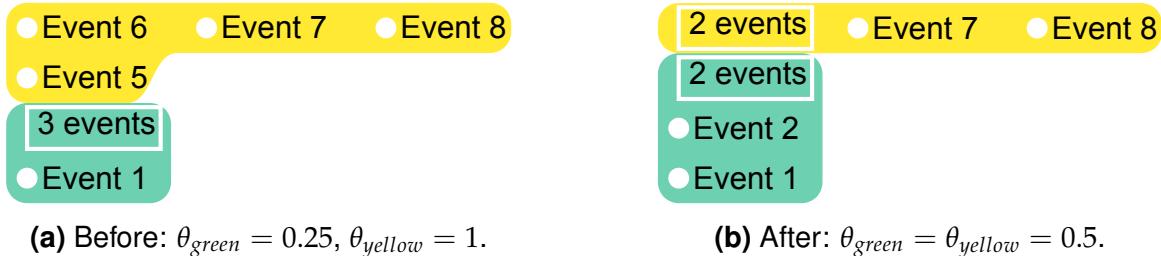


Figure 4.11: Layers balancing.

4.4.5 Scalability

Aggregation enables TimeSets to visualize a large number of events. However, the visual representation of aggregated events is imperfect. For instance, two aggregates “2 events” and “100 events” are displayed exactly the same, except for the total number, whereas their sizes are largely different. We consider four options to address this issue as shown in Figure 4.12.

The first option is to plot each individual event as a dot at when it happens (Figure 4.12b). This also provides a temporal distribution of events rather than just

the total count. When events happen closely, dots are overlapped, which makes it more difficult to see the pattern. Second, the width of the rectangular border can be scaled to indicate the number of events (Figure 4.12b). The full width of an aggregate border is typically small as the length of text “ N events”. Therefore, the difference among aggregate border widths could be subtle and difficult to observe from an overview. Another option is to color code the background of the bounding rectangle using luminance or intensity (Figure 4.12d) according to the number of events. However, when many aggregated events are displayed, their backgrounds could interfere with the set colors and distract users. The last option we propose is to scale the font size of the label based on the number of events (Figure 4.12e). Currently, each event is completely located in one single row with uniform height. Scaling the heights of aggregated events will affect the layout algorithm.

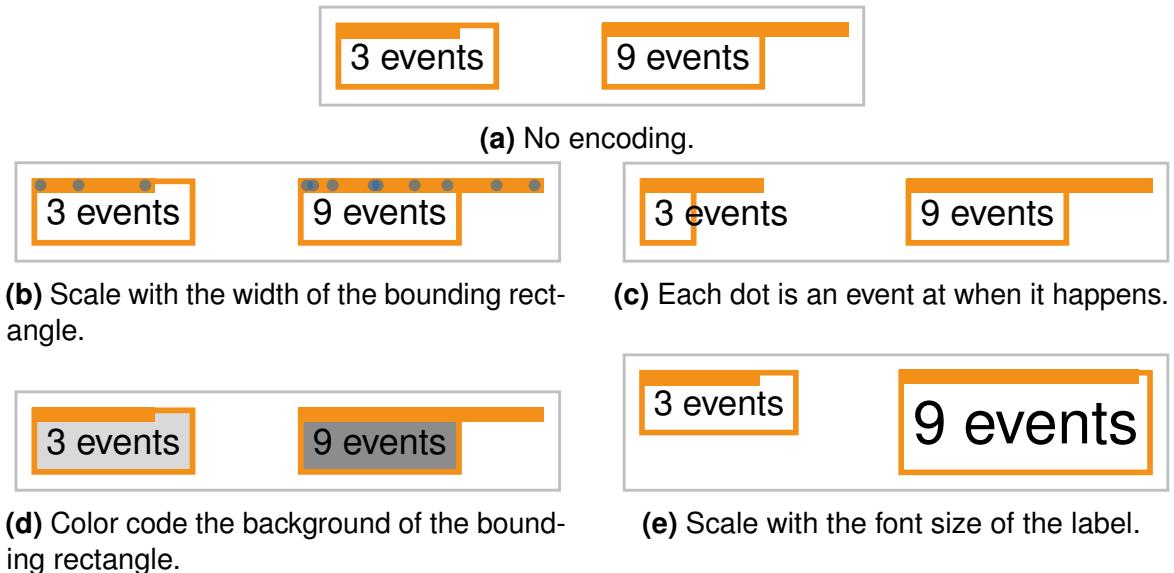


Figure 4.12: Proposed visual representations of aggregates emphasizing the number of its events.

The existing layout is suitable for a small timeline with a few hundreds of events or a detailed view where individual events are of high importance. Figure 4.16 shows 225 publications within 15 years. TimeSets relies on color to distinguish sets, therefore it is constrained by the number of colors that human can differentiate at the same time, which is about 12, according to Munzner’s book [119].

4.5 Evaluation

4.5.1 Method

We conducted a lab controlled experiment to compare task performance between TimeSets and a state-of-the-art visualization technique. However, as discussed in Section 4.2, no existing techniques are designed to show both multiple-set relations and temporal information of events together. Therefore, rather than evaluating both the layout and the set visualization technique of TimeSets, we decided to focus only on the second contribution. We compared TimeSets with a set visualization technique that can apply on top of an existing timeline. We chose KelpFusion [114] because among similar techniques, it has been shown to have the best performance in readability tasks, regarding both accuracy and completion time. We acknowledged that KelpFusion was not specifically designed to work with timelines. However, KelpFusion can work with any given layouts, and it is the best choice for this evaluation. Our experiment followed a within-subject design; accuracy, time and user preference were collected.

4.5.1.1 Datasets

We used generated data for the experiment to avoid that participants might be distracted from their prior knowledge. Only time-point events were used because KelpFusion can only take input as a set of points. The complexity of the dataset was controlled by two parameters: the number of sets and the average number of events per set. Overall, half of the events were multiple-set, the same ratio as in a real-world dataset used in Figure 4.8. The details of the four levels of complexity used in the experiment are shown in Table 4.1.

Table 4.1: Dataset Statistics

Complexity	# Sets	# Events	# Intersections
Level 1	3	30	15
Level 2	3	45	23
Level 3	5	50	25
Level 4	5	75	38

Images of these datasets using the KelpFusion method were generously provided by the method’s author. To avoid bias, our method also used static images instead of interactive visualizations. Colors for both methods were Qualitative Set 2 of ColorBrewer [67]. KelpFusion does not have its own layout; therefore, our layout algorithm was used for both settings. Only one algorithm was used to prevent

adding another factor to the experiment, which doubles the number of trials for participants. The traceability algorithm was chosen because reading comprehension is not required for the tasks. Figure 4.13 shows example images used in the experiment.

4.5.1.2 Tasks

We followed the task design in the KelpFusion technique evaluation [114], including estimation and precise comparison of set sizes, and counting the number of elements in a set. Two time-related tasks were added to evaluate the temporal aspect of the visualization. Therefore, there are five tasks in total. We considered three categories of set readability tasks, relating to: the *set* itself, the *intersection* of two sets, and the *difference* between two sets. However, it was impractical to include all 5×3 task types in the experiment. Therefore, we decided to use a combination of them: two tasks for the set itself, two tasks for the intersection, and one task for the difference. Tasks together with examples are listed in Table 4.2. Each participant would complete a total of 40 questions.

Table 4.2: Tasks used in the experiment.

Task	Example
SetOverview	Roughly estimate which set has more events: A or B (please do NOT count the number of events)?
IntersectionCompare	Which set pair shares more events: A&B or C&D (please count the number of events)?
DifferenceCount	How many events are there that belong to the set A but not its neighboring sets?
SetBiggestYear	In which year does set A have the most events?
IntersectionPattern	During 2002–2004, what is the change pattern in the number of events shared by set A&B?

We used general questions to preserve the external validity of the experiment. It is straightforward to convert them into context-sensitive questions. For example, in the context of *news media*, the last task can be transformed to “What is the trend of news articles related to both science and fashion during the last 3 years?”. We chose to use multiple-choice answers to reduce the completion time, allowing the within-subject comparison to finish within a reasonable time. This reduces the possible effect of boredom or fatigue as confounding factors. It also removes the requirement to consider the typing speed of subjects when evaluating time taken to complete tasks.

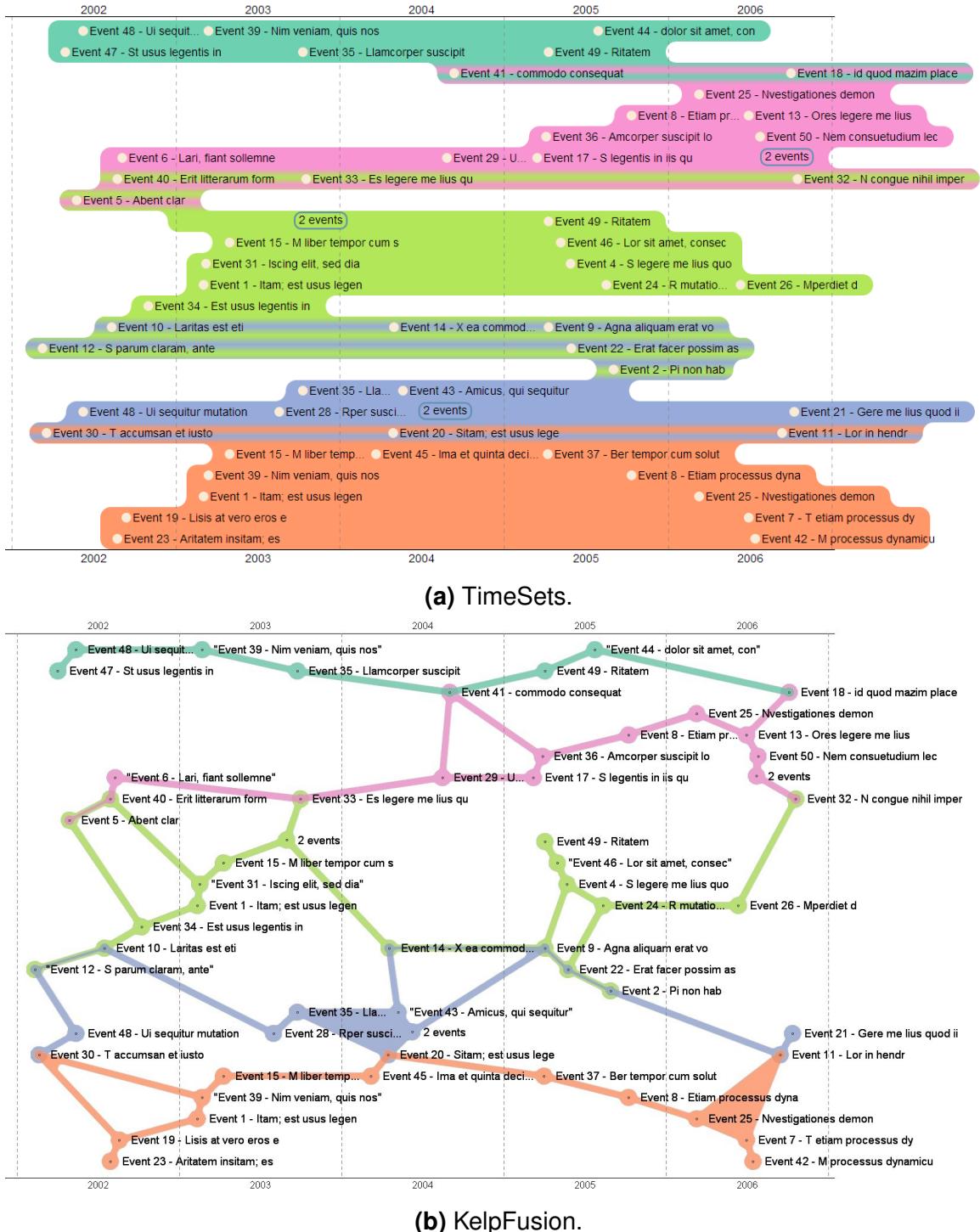


Figure 4.13: Example images used in the experiment.

4.5.1.3 Participants and Apparatus

Thirty students (23 males, 7 females) voluntarily participated in the experiment. They came from various backgrounds including computing, law, and psychology. One participant was under 19, 16 participants were aged between 19–25, 12 were aged between 26–39, and one was aged between 40–60. All participants reported that they can distinguish all colors used in the experiment. Participants completed the experiment using a 23-inch monitor with a resolution of 1920 × 1080.

4.5.1.4 Procedure

The study lasted approximately 45 minutes and consisted of two sessions (one for each visualization technique), followed by a questionnaire. At the beginning of each session, the visualization technique was explained, and participants were shown how to answer each question type using that method. This was followed by five practice questions to familiarize participants with the tasks and the experiment interface. Solutions and explanations were given for these practice questions to help them understand better.

We used two question sets with comparable difficulty and counterbalanced the order of the visualization techniques as well as the order of question sets to reduce learning effects. We fixed the order of task types and the order of difficulty in each type from simple to complex. For each task, the question and all the answer options were displayed without the visualization. Once participants finished reading, they clicked a button to reveal the figure, when the timing started. This is to reduce the effect of individual differences in reading and comprehension speed on the measured time.

4.5.2 Hypotheses

For task performance, we hypothesized that

- H1.** Overall, TimeSets will outperform KelpFusion in both time and accuracy. The colored backgrounds of sets in TimeSets produce a stronger sense of grouping than the connected paths in KelpFusion. Also, in TimeSets, shared events are visually connected, separated from the non-shared ones.
- H2.** For the *SetOverview* task, participants using TimeSets will be faster but less accurate than using KelpFusion. In TimeSets, the colored background makes

sets more noticeable; however, the colored area is not always proportional to the set size, which may lead to a misunderstanding for participants.

- H3.** For both the *IntersectionCompare* and *IntersectionPattern* tasks, TimeSets will outperform KelpFusion in both time and accuracy. In TimeSets, shared events are visually connected in its own layer, whereas in KelpFusion, they are mixed with non-shared events.
- H4.** For the *DifferenceCount* task, TimeSets will outperform KelpFusion in both time and accuracy. In TimeSets, events that do not belong to neighboring sets have their own layer with a unique background color, whereas in KelpFusion, they are mixed with the shared events.
- H5.** For the *SetBiggestYear* task, KelpFusion will outperform TimeSets in both time and accuracy. When focusing on events in each year, connected lines in KelpFusion make it easier to count.

For user preference, we hypothesized that

- H6.** Participants will be more confident with TimeSets because it provides better visual support, especially in intersection and difference tasks.
- H7.** TimeSets will be more aesthetically pleasing than KelpFusion with smooth curves and smooth color changes compared to straight lines and plain colors.
- H8.** TimeSets will be less cluttered than KelpFusion because it uses simple shapes, whereas KelpFusion uses a combination of lines and areas.
- H9.** TimeSets will provide a stronger sense of grouping than KelpFusion because it colors the entire background of a set.

4.5.3 Results

We used a repeated-measure analysis of variance (RM-ANOVA) to analyze the task accuracy and completion time. Accuracy is measured as the percentage of correct answers. The logarithm of completion time is used to normalize its skewed distribution.

4.5.3.1 Accuracy

Figure 4.14a shows the mean accuracy. The RM-ANOVA test revealed a significant main effect of visualization technique ($F(1, 29) = 4.99, p < .05$), showing that accuracy was significantly higher with TimeSets. There was also a significant main effect of task type ($F(4, 116) = 8.89, p < .00001$). No significant effect of the visualization \times task interaction was found ($F(4, 116) = 1.85, p = .12$). Paired t-tests were conducted to investigate the performance difference for each task. A significant effect was found in three tasks: IntersectionCompare ($p < .05$), DifferenceCount ($p < .01$), and IntersectionPattern ($p < .05$), indicating TimeSets was significantly more accurate than KelpFusion in them. Only the task DifferenceCount still had a significant effect with corrected p-value for multiple tests using Bonferroni correction.

4.5.3.2 Time

Figure 4.14b shows the mean completion time. The RM-ANOVA test revealed no significant main effect of visualization technique ($F(1, 29) = .05, p = .82$), indicating that the completion time for TimeSets ($M = 23.87, SD = 9.18$) and KelpFusion ($M = 23.72, SD = 11.38$) were not significantly different. There was a significant main effect of task type ($F(4, 116) = 23.80, p < 10^{-12}$). The visualization \times task interaction was also significant ($F(4, 116) = 3.23, p < .05$), indicating that difference in completion time due to visualization technique was significantly different across tasks. To further investigate this, a paired t-test for each task was conducted. Significant effects were found in the DifferenceCount task ($p < .01$), indicating TimeSets is significantly faster in this task, and the SetBiggestYear task ($p < .01$), indicating KelpFusion is significantly faster in this task. Both tasks still had a significant effect with corrected p-value for multiple tests using Bonferroni correction.

4.5.3.3 User Preference

Participants were asked to rate both methods using a Likert scale from 1 (worst) to 5 (best) after they completed all the tasks. The following four questions were asked for each visualization technique:

- How confident were the participants in answering the questions?
- How aesthetically pleasing were the visualizations?
- How cluttered were the visualizations?

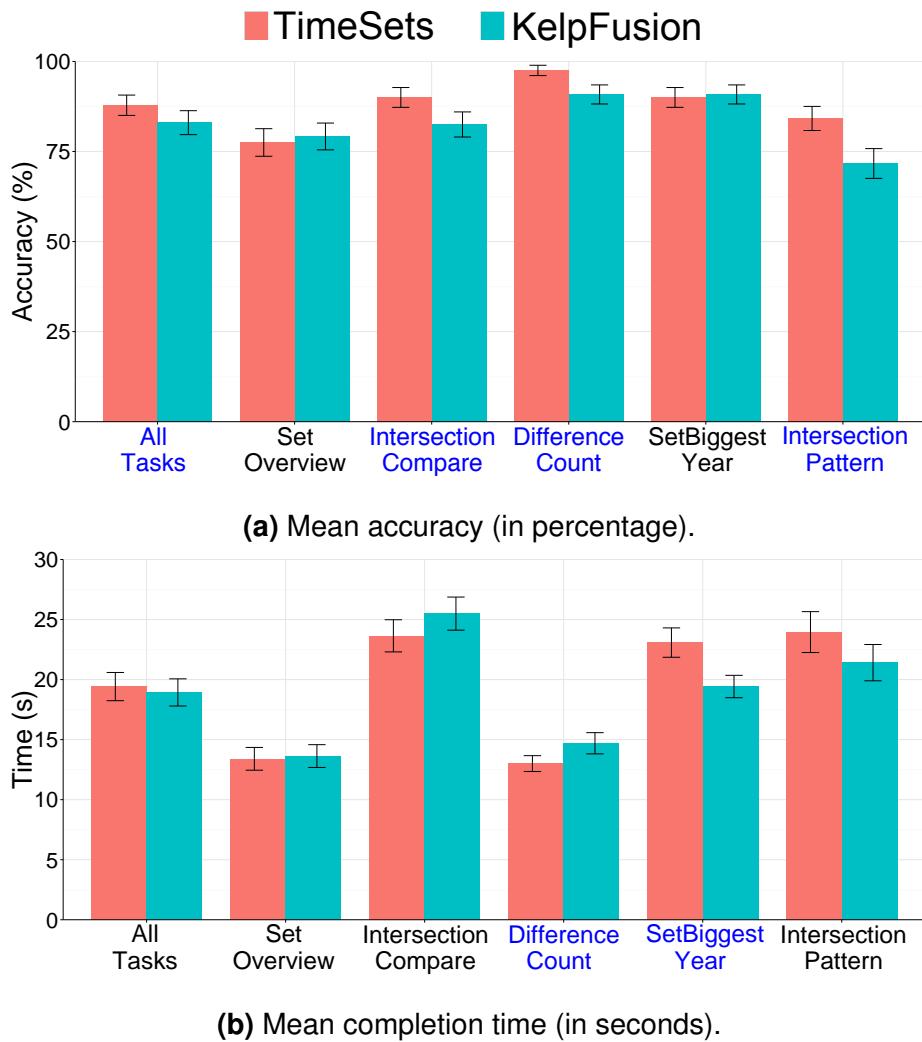


Figure 4.14: Mean accuracy and completion time for each task. Error bars show standard error. Tasks with Significant effect are highlighted.

- How strong was the sense of grouping?

Figure 4.15 shows the summary of user ratings. Fisher's exact tests found significant effects in all questions: Confidence ($p < .01$), Aesthetically Pleasing ($p < .01$), Not Cluttered ($p < .01$), and Sense of Grouping ($p < .0001$); indicating users preferred TimeSets to KelpFusion in those aspects.

4.5.4 Discussion

The results show that overall, TimeSets outperforms KelpFusion in accuracy, but not in completion time. This partly agrees with hypothesis H1.

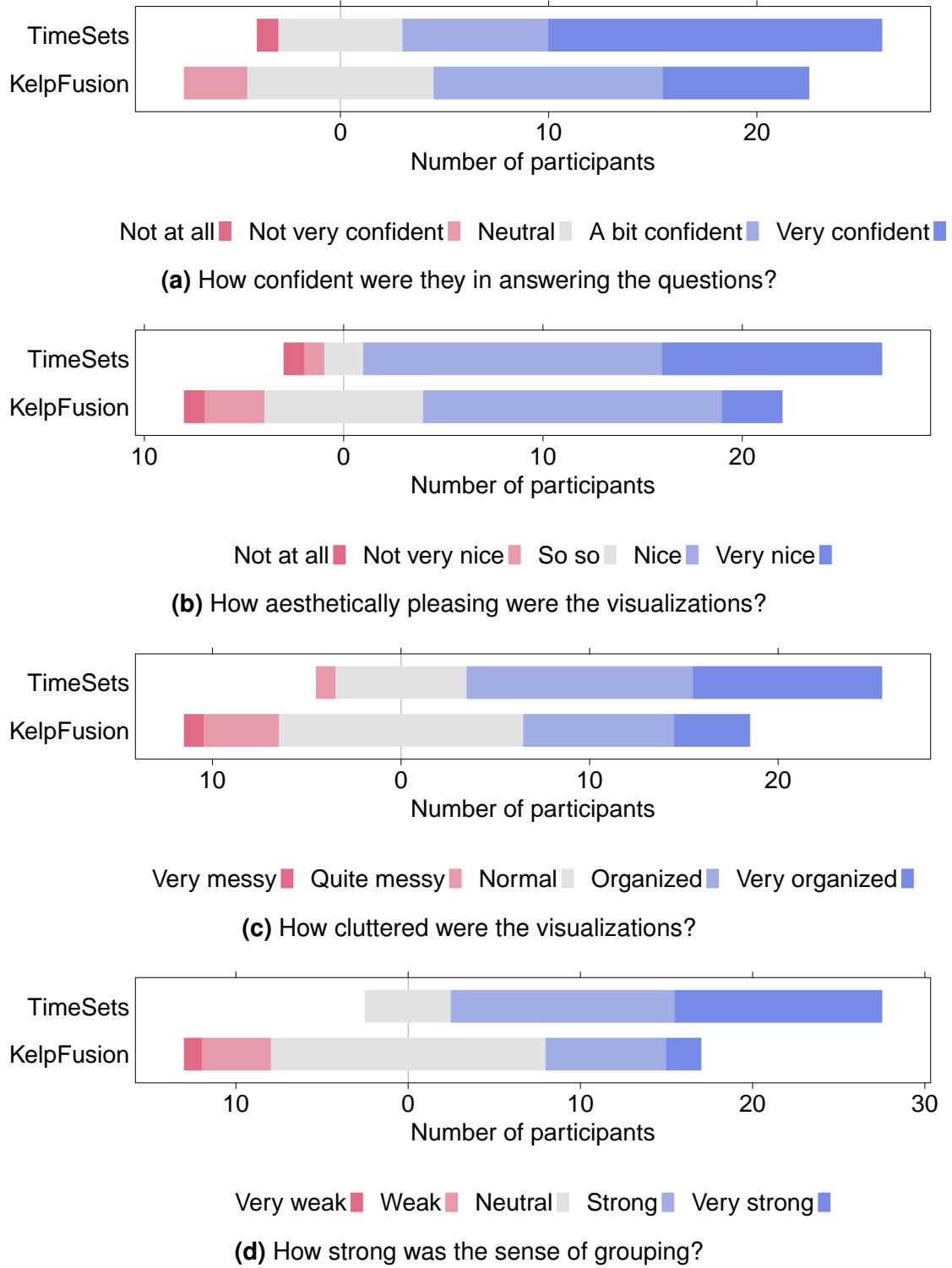


Figure 4.15: Subjective user ratings of each technique for each question. Bar width represents the number of participants selected the corresponding option.

For the *SetOverview* task, there was no significant effect of visualization technique on either accuracy or completion time, which disagrees with hypothesis **H2**. The average accuracy of both methods is relatively low to other tasks in the experiment. Possible causes for TimeSets are discussed earlier in the hypothesis statement, and the edge length in KelpFusion – a prominent visual feature – is probably not either a good size indicator.

For intersection tasks, the results also show that TimeSets has higher accuracy than KelpFusion; however, their completion time performances are not significantly different. This partly confirms hypothesis **H3**. In TimeSets, shared events are highlighted by color gradient, thus participants are less likely to miscount them. In KelpFusion, shared events are horizontally aligned, because it shares the same layout as TimeSets. We observed that some participants tried to trace shared events using this way, which is prone to missing events, thus KelpFusion has similar speed but lower accuracy.

Hypothesis **H4**, about the *DifferenceCount* task, is supported by the results. Events that belong to a single set are clearly shown in TimeSets as a region with a single colored background. This helps improve performance in both accuracy and completion time.

The results show that KelpFusion has faster completion time than TimeSets for the *SetBiggestYear* task, but there is no significant difference in accuracy. This partly agrees with hypothesis **H5**. The vertical lines used to denote year boundaries in this task may have helped, by splitting the visual area into columns. To solve the task, participants count the number of events in each column and pick the highest one. A KelpFusion visualization is quite similar to a network, and edges connecting events within each column can make counting easier. This may explain why participants counted faster with KelpFusion, but had the same accuracy as with TimeSets.

To visualize sets, Bubble Sets [33] uses a similar metaphor as TimeSets – filling the area of same-set events with a unique color. However, KelpFusion outperforms Bubble Sets [114], while TimeSets outperforms KelpFusion in solving similar tasks. One possible explanation is that the irregular shapes generated using iso-contours in Bubble Sets make set memberships difficult to perceive. Also, the layout in TimeSets groups same-set events together, which allows participants to easier count or estimate. Another reason could be that the color gradient in TimeSets may be more effective than color blending in Bubble Sets for visualizing shared events.

The participants preferred TimeSets in all four questions: confidence, aesthetics, readability, and sense of grouping. This supports hypotheses **H6**, **H7**, **H8** and

H9. Half of the participants (15 out of 30) were more confident with TimeSets. Some of them commented that its set background made it easier to count events, especially for the intersections. Only four participants thought that they were more confident with KelpFusion (the other eleven thought they were at the same level of confidence). One said “I can follow the links when counting, so I’m less likely to miss any”. Interestingly, three of these four participants actually had better accuracy with TimeSets. Half of the participants (15 out of 30) thought that TimeSets was more aesthetically pleasing than KelpFusion. Some of them said that they liked the curved boundaries and the smooth changing of colors. Only three participants favored KelpFusion. One of them commented that with TimeSets, his eyes were tired after looking at large areas with bright colors for a long time. More than half of the participants (17 out of 30) rated TimeSets as less cluttered than KelpFusion. One said “TimeSets is more organized. I know event labels aren’t important, but they seem easier to read.”. Three quarters of the participants (22 out of 30) agreed that TimeSets provided a stronger sense of grouping than KelpFusion. Many of them commented that KelpFusion figures looked more like a network than a group.

4.6 Case Study 1: Publication Data

This section demonstrates an application of TimeSets in analyzing publication data. A subset of 200 articles with the most citations from the IEEE InfoVis conference is used [164]. Each publication includes one or many *concepts* such as *network* or *evaluation*. We use this concept as the set attribute to group publications. Figure 4.16 shows the visualization of this dataset. No aggregation is needed when producing the layout; only complete and trimmed labels are used.

First, TimeSets can reveal the temporal distribution of a thematic dataset as in ThemeRiver [68]. A quick glance at the visualization brings us a surprise. There is much void space on the left as opposed to a very dense area on the right indicating that there are many more highly cited papers published in the last ten years than those in the first ten years. This trend also holds for individual concepts: each colored layer starts with a single row and becomes higher towards the end of the timeline. This observation is in contrast with our thought: articles that have been published since longer time ago would receive more citations. One possible explanation is that the IEEE InfoVis conference has accepted more papers over time: in the dataset, 18 articles were published 1995, whereas 37 articles were published in 2013. Another reason could be that articles in the last ten years are really of higher quality.

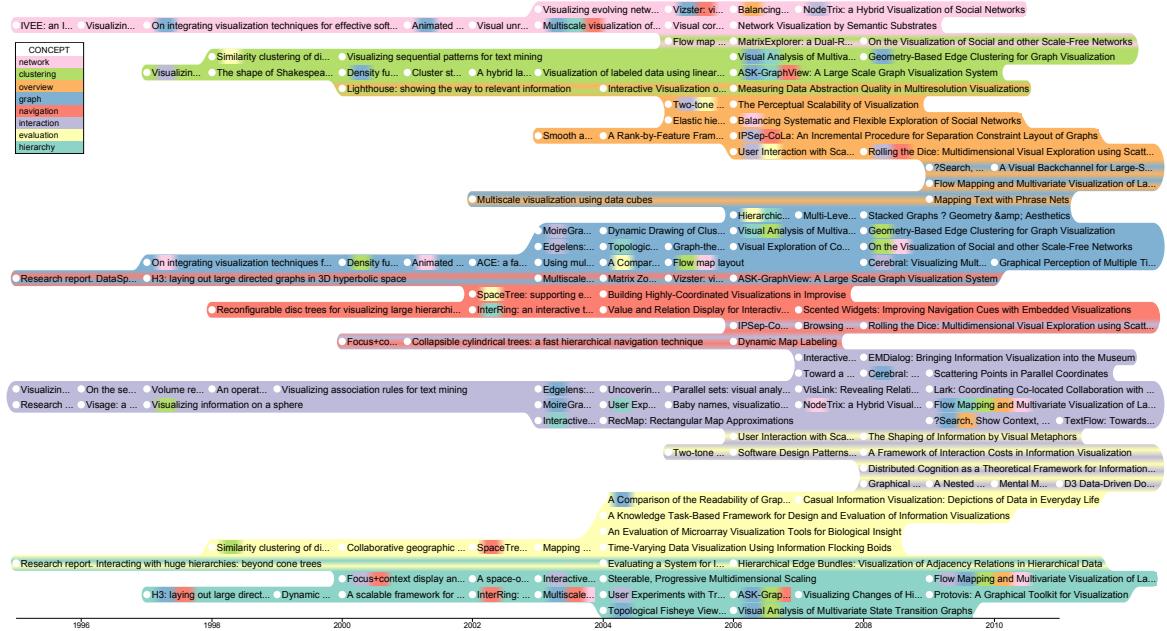


Figure 4.16: TimeSets visualization of publication data. It shows 200 articles with the most citations in the IEEE InfoVis conference from 1995 to 2013. These articles are categorized based on their concepts (see the legend in the top left hand corner).

TimeSets cannot show all intersections among sets; however, its layout maximizes the number of shared elements between two neighboring sets. Therefore, the visible intersections usually include the most elements of all intersections. In the visualization, the most notable gradient area is the intersection between the yellow and the purple sets indicating that many excellent articles focus on both *evaluation* and *interaction*. Also, at the top of the visualization, three concepts: *network*, *clustering* and *overview* with *clustering* is in between the other two. This is expected because clustering techniques are important in visualizing large networks and providing an overview of a large dataset.

In Figure 4.16, TimeSets uses the color gradient method to show full memberships of multi-set elements. For instance, inside the *network* layer, there are quite a few small blue gradients for *graph*. This makes sense because the close relationship between these two concepts, and they may often appear together in the same articles. We also notice an article including the most concepts at the bottom of the visualization: “Flow Mapping and Multivariate Visualization...” (the last article on the third last row) with *hierarchy*, *interaction*, *graph*, *overview* and *network*.

4.7 Case Study 2: VAST Challenge 2014

TimeSets was integrated into the visual analytics system SAVI [196] that won an award in the IEEE VAST Challenge 2014². The challenge is an annual contest with the goal of advancing visual analytics through competition. Commonly, participants are given a synthetic dataset and asked to identify suspicious activities within the dataset. The mini challenge that SAVI participated involves a fictitious company that has several employees have been missing. Participants had to collect and analyze streaming tweets in order to identify five interesting events before presenting hypotheses and evidence about the disappearance of the employees.

The SAVI interface consists of three panels (Figure 4.17). At the bottom is a histogram showing the frequency of tweets over time. On the right hand side is the a collection of named entities identified from the entire dataset. Both components also act as filters. For example, when a user selects an interval of interest from the histogram, only tweets within that interval will be displayed in TimeSets, at the top-left corner of the interface. The user can select several entities to make TimeSets only show tweets mentioning at least one of these entities. TimeSets also uses the entities as *sets*, allowing the user to observe the development of selected entities overtime.

Besides showing tweets as raw data, TimeSets can also be used to construct and present interesting events. An event is described a sequence of related tweets organized in a chronological order. During the analysis, the analyst can create events and add tweets into one or many events. As a result, TimeSets allows the analyst to present an event together with its key elements. Also, by showing many events simultaneously, tweets that belong to multiple events can be identified and suggested further exploration to understand the connection among them.

4.8 Summary

In this chapter, we introduced the TimeSets method that enable users to explore complex temporal relationship by effectively representing both temporal and thematic provenance data. The method is general and can be used to visualize temporal and categorical data in different domains. TimeSets groups temporal events vertically with colored backgrounds according to their set memberships. Events shared by two sets are visualized using layers with a color gradient background. TimeSets also

²<http://www.vacommunity.org/VAST+Challenge+2014>

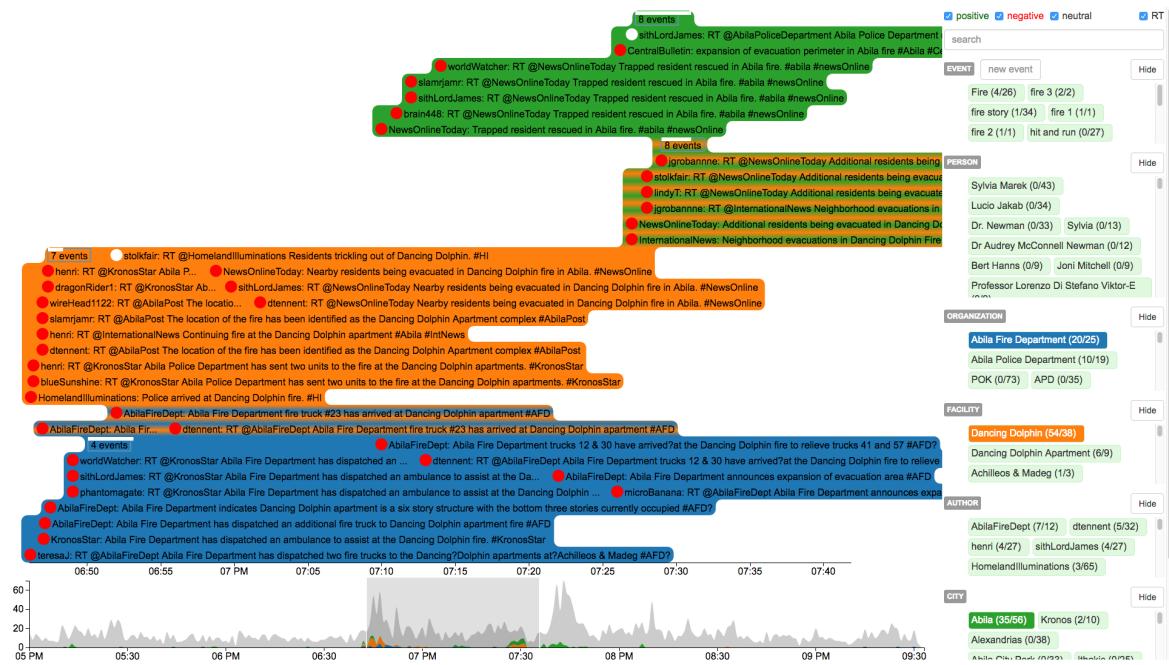


Figure 4.17: SAVI interface. Bottom left: Histogram showing the frequency of tweets. Right: List of extracted named entities. Top left: TimeSets showing detailed tweets.

dynamically adjusts the event labels between three levels of detail to scale with the number of events. The amount of event labels displayed can be traded for ease of following events chronologically using the traceability layout algorithm. The results from the controlled experiment comparing TimeSets to KelpFusion showed that in overall, TimeSets was significantly more accurate and the participants preferred TimeSets for aesthetics and readability.

Many directions can be extended to further improve TimeSets. Currently, duplicated events can only be discovered when mouse hovering. A better visual hint without making the visualization too much cluttered could be useful. We propose different techniques to encode multi-set memberships and to represent aggregated events. However, formal evaluations should be conducted to understand which options are more effective. Also, it would be more beneficial to address the issues identified in the user evaluation, such as the set area not being a reliable indicator of event number and the irritation from bright set colors after a long viewing period.

5

Exploring Rational Relationship through User Actions

Chapter 3 and Chapter 4 enable users to explore complex temporal relationship of sensemaking through interactive timeline visualizations of their annotations. However, the relationship emerged in the sensemaking process could be more complicated than temporal one. Besides understanding how events are connected, why they happened in such order is also important. In this chapter, we investigate how to support users to explore the rational relationship of sensemaking though analytic provenance. Typically, qualitative research methods are used in understanding such relationship. This is a manual and time-consuming process: researchers collect observation data, transcribe screen capture videos and think-aloud recordings, identify recurring patterns, and eventually abstract the sensemaking process into a general model. We contribute a visual sensemaking tool – *SensePath* – that offers an alternative and possibly faster approach in performing *transcription* and *coding*.

The previous two chapters visualize annotations made by users. Even though these annotations may contain high-level thinking of the users, making detailed and frequent notes require a huge amount of time and effort from the users. Also, it may distract them from their sensemaking tasks. In this chapter, we focus on exploring **rational relationship** of sensemaking through both **user annotations** and automatically captured **user actions**. SensePath provides multi-linked visualizations of the captured provenance and interactive features to support their further analysis. Two user-centered evaluation were conducted to explore how the tool would be used and identify any benefits it could provide. The participants found the tool

Previously published at the *IEEE Transactions on Visualization and Computer Graphics* [125].

intuitive and considerably reduced analysis time, allowing better understanding of the sensemaking process.

5.1 Introduction

Sensemaking is described as the process of comprehension, finding meaning and gaining insight from information, producing new knowledge and informing action [136]. Given the rapid increase in data volume and complexity, more tools are needed to support sensemaking, which in many cases remains a slow and laborious process performed by human analysts. The design of such tools requires a deep understanding of the sensemaking process, which is a reoccurring goal of qualitative research conducted by many HCI researchers. Common methods for such qualitative analyses are grounded theory [35] and thematic analysis [64]. Typically, researchers need to design a study, collect observation data, transcribe the screen capture videos and think-aloud recordings, identify interesting patterns, group them into categories, and build a model or theory to explain those findings. Unfortunately, this process largely remains manual and thus very time consuming.

In this chapter, we introduce a visual analysis tool SensePath to help HCI researchers recover user's thinking using analytic provenance information. More specifically, we support thematic analysis of online browser-based sensemaking tasks. We chose this domain because many of the everyday sensemaking tasks such as travel planning, are now performed online [151]. The design of SensePath is based on the observation of a number of sensemaking sessions and the post hoc analyses that researchers performed to recover the sensemaking process. This is followed by a participatory design session with HCI researchers that led to a number of design requirements such as supporting reasonably long sensemaking tasks (up to two hours), integration with existing qualitative analysis workflow, and non-intrusiveness for participants.

As a result, SensePath was designed to target the *transcription* and *coding* phases during which a researcher needs to transcribe the observation data, such as screen capture video and think-aloud recording (transcription), and then identify the common themes of the sensemaking actions within them and assign appropriate names (coding). SensePath consists of two components designed for different stages of thematic analysis: one runs in the background during the observation to automatically capture analytic provenance, which includes sensemaking actions; the other component is designed for post hoc analysis phase and it visualizes the recorded

information in four linked views to help transcription, coding, and identify frequent patterns and high level sense-making process. While some features are tailored for sensemaking, the general approach of understanding user's thinking can be applied to a wider qualitative research. Also, SensePath can be extended to support the analysis of other online activities, not limited to sensemaking tasks.

Two evaluations were conducted to understand how the tool was used by experienced HCI researchers and to discover whether and how SensePath provides any advantages to the analyst compared to traditional analysis methods. The researchers found the tool intuitive and considerably reduces the analysis time, enabling the discovery of underlying sensemaking processes.

In summary, this chapter contributes

- A qualitative study and a participatory design session to understand characteristics of qualitative research on sensemaking.
- A visual analysis tool SensePath enabling researchers to explore rational relationship of a user's sensemaking process. It supports the transcription and coding of the observation data of online sensemaking tasks and can be potentially applied to other qualitative research in HCI beyond sensemaking.
- A qualitative user evaluation that demonstrated the effectiveness of the general approach and the tool SensePath.

5.2 Related Work on Qualitative Research

Qualitative research methodologies [5] are typically used in study of sensemaking. They allow researchers to reveal often complex user experiences and understand issues that are experienced subjectively or collectively [5, 128]. Moreover, sensemaking research is often concerned not with testing an existing theory, but building a new one through the collection and analysis of relevant data, generating new knowledge about users and the usage of technology [148].

Inductive approaches to qualitative research are commonly applied, such as grounded theory [35], content analysis [165], and thematic analysis [64]. They rely on the interpretation of rich textual and multimedia data, through manual processing of data and coding before describing it in the context of categories or themes. Furthermore, in the case of multimedia data, transcription of audio or video data is often also required. Though these approaches lead to important insights, they are

labor intensive, time-consuming and costly in their application [192]. Software packages are designed to provide qualitative researchers useful ways to code and index data, and manage the evolving complexity of the process [103]. However, a qualitative analysis is still a largely manual process requiring a substantial investment of time and resources in leading to insightful findings.

5.3 Approach

5.3.1 Qualitative Analysis of Sensemaking

We conducted two sets of observations to explore the characteristics of qualitative analysis of sensemaking activities. The purpose was twofold:

1. To identify any unique characteristics of qualitative analysis in sensemaking studies differ from that in general human-computer interaction (HCI) research.
2. To understand the process and tools used, and to identify any potential issues that can be addressed using visual analytics.

Each set of observations includes both how a HCI researcher collected observation data when a participant was performing a sensemaking task, and the following data analysis session, during which they used a qualitative method to analyze the collected data and gained a deep understanding of the participant's sensemaking process. We call our observations as *meta observation* to differentiate them from the observations HCI researchers conducted. These HCI researchers are our target users. Figure 5.1 illustrates the meta observation process.



Figure 5.1: Meta observation. A HCI researcher conducts a qualitative study to understand a participant's sensemaking process. This study includes user observation and data analysis. We observe this study to explore its characteristics and identify potential support to HCI researchers – our target users.

In the first meta observation, six participants were recruited and asked to do research online and select the smart watch they would like to purchase. The session stopped when the participant decided on the smart watch model, lasting between 30 to 45 minutes. These sessions were observed by a junior HCI researcher with limited qualitative analysis experience, and he recorded the sensemaking process by making notes on paper and screen recording. Interviews were conducted once the task was completed. Once all six observations were completed, the researcher conducted a *thematic analysis* on the observation data, and the results were summarized in a two-page report. We interviewed the researcher after the report was finished to gain deeper understanding of his qualitative research process.

To ensure the qualitative analysis we observed was not biased, we conducted a second set of meta observations with a more experienced HCI researcher. One participant was tasked to plan a holiday for a fictitious family with particular needs. Two further participants were given the task to select a smart watch as described earlier. This researcher also made notes during the observation, and used thematic analysis to analyze the observation data.

To identify any unique features of qualitative analysis in sensemaking research, we conducted our own thematic analysis on the meta observation data. On completion, we discussed our findings with the two HCI researchers participated in the meta observations. This led to a qualitative research process that aims to understand sensemaking, consisting of the following steps:

1. **Study Design.** Decide the study setup including the sensemaking task, dataset, and data to capture based on the targeted research question.
2. **Data Collection.** Capture the processes while the participants perform their sensemaking tasks. The collected data could include screen captures, think-aloud recordings, video recordings of participants' faces and gestures, and interview notes.
3. **Transcription.** Transcribe video and audio recordings verbatim.
4. **Coding.** Identify common themes in the transcripts and assign appropriate names or codes to them.
5. **Categorization.** Group similar themes into more abstract categories.
6. **Model.** Match the identified themes and categories to existing sensemaking models or design a new one, depending on the research question.

Step 2 to 6 represent a progression on the semantics: each step takes the output from the previous step as input, and produces an outcome with richer semantics. This is similar to the four layers of visual analytic activities in the Gotz and Zhou's model [61], but targeting different aspect: the former focuses on the sensemaking model and theory, whereas the latter focuses on user activities. In summary, we did not discover any unique characteristics of qualitative analysis for sensemaking. This implies the approach and tool we developed for sensemaking are likely to be applicable to qualitative analysis intended for other purposes. Our study did equip us with detailed knowledge about the actual qualitative analysis process, which informed the design of our visual analytics tool.

5.3.2 Requirements

A participatory design session with the two HCI researchers involved in the meta observations was followed up to elicit requirements for the tool that aims to facilitate the existing analysis process. During the session, we discuss possible tool features and designs to address them. The requirements are presented as follows and the design ideas are described in Section 5.4.

1. **Thematic analysis support.** This is the analysis method used in both our meta observations and the two sensemaking observations. It also shares many characteristics with other popular qualitative research analysis methods such as grounded theory.
2. **Transcription and Coding efficiency.** All aforementioned steps from 2 to 6 are time-consuming; however, *transcription* and *coding* are where a visual analytics tool can potentially make the most difference. Their lengths largely depend on the efficiency of the tools employed in these steps. *Data collection* primarily depends on the task's completion time and the number of participants. Also, transcription and coding are not as abstract or semantically rich as *categorization* and *model*, making it more feasible to provide automated support.
3. **Existing workflow integration.** The tool should maintain the way researchers currently work and ideally works together with other software packages already used in the analysis workflow.
4. **Non-intrusiveness.** The tool should not distract participants or affect their behaviors during the sensemaking tasks.

5. **Scalability.** The tool should support reasonably long sensemaking sessions with a duration up to an hour or two.
6. **Lightweight.** The tool should be lightweight and support multiple operating systems.

5.4 Interface Design

5.4.1 Approach

The design process started with a close examination of the analysis steps we plan to support. For transcription, we ruled out the possibility of automatic video or audio transcribing because these are research challenges in their own right and require expertise different from visual analytics. During our meta observation data analysis, we noticed that a large portion of time spent on transcribing video recordings was to identify the sensemaking actions that the participants performed (such as searching) and contextual information associated with these actions (such as the search keyword and its timing). These pieces of information can potentially be captured within the browser, thus considerably reduces transcribing time.

From the aforementioned participatory design session, we found that an important part of the coding process is to understand the sensemaking activities from the video transcripts. For instance, when a participant spent several minutes on a page, he was likely reading through the information. When the participant switched between two pages back and forth, he might be comparing two smart watch models. Understanding the nature of such sensemaking activities; i.e., reading or comparison, is the prerequisite for identifying common themes and naming them. To a certain extent, this is equivalent to inferring “sub-task” from “action” in the Gotz and Zhou’s model [61]. However, this process is difficult to be completely automated [45]. After further discussion with the HCI researchers, we identified three important factors to this process that can be supported by visual analytics:

1. **Seeing the actions before and after the current one.** These provide useful contextual information because an “action” is usually a part of a “sub-task”, which consists of a number of actions. For example, when a participant went through the web page of a number of hotels in succession, they might be comparing these hotels, especially if all these pages are opened from the same hotel booking website. Showing a number of actions together would help a

researcher to identify the connections between them and potentially find an interpretation for all the actions in the sequence as a whole.

2. **Seeing what a participant was looking at.** It may appear obvious, but it can give a researcher the needed context to understand the sensemaking actions. For example, looking at Google Maps may indicate the participant was trying to locate a certain place. This can be particularly useful if the researcher is absent from the observation.
3. **Understanding what a participant was thinking.** Even though this can be partly captured through think-aloud protocol or post hoc interview, another common technique is to enable participants to record their own thinking by providing note taking support.

5.4.2 Overview

Our tool – SensePath – is implemented as a Chrome extension consisting of two components. The first one is a background process running in the participant's browser to automatically capture all the required analytic provenance during the observation stage of the qualitative study. It also offers additional features to add note and highlight text on a web page (Factor 3 – understanding what a participant was thinking). Besides, participants will not notice any difference from their normal sensemaking session (Requirement 4 – non-intrusiveness).

The second component includes a set of four linked visualizations of the captured provenance data (Figure 5.2), designed to facilitate transcription and coding (Requirement 1 and 2).

- A *timeline* view displaying participant's sensemaking actions in their temporal order (Figure 5.2A). This enables researchers to see an action in the context of other actions (Factor 1 – seeing the actions before and after the current one).
- A *browser* view showing the web page where the sensemaking action was performed (Figure 5.2B). This provides the contextual information of sensemaking actions (Factor 2 – seeing what a participant was looking at).
- A *replay* view showing screen capture video (Figure 5.2C). This provides additional contextual information about browser interaction that is missing from the timeline such as scrolling and mouse movement (also Factor 2).

- A *transcription* view detailing selected sensemaking actions (Figure 5.2D). The generated transcript can be exported and then used in popular qualitative data analysis software (Requirement 3 – existing workflow integration).

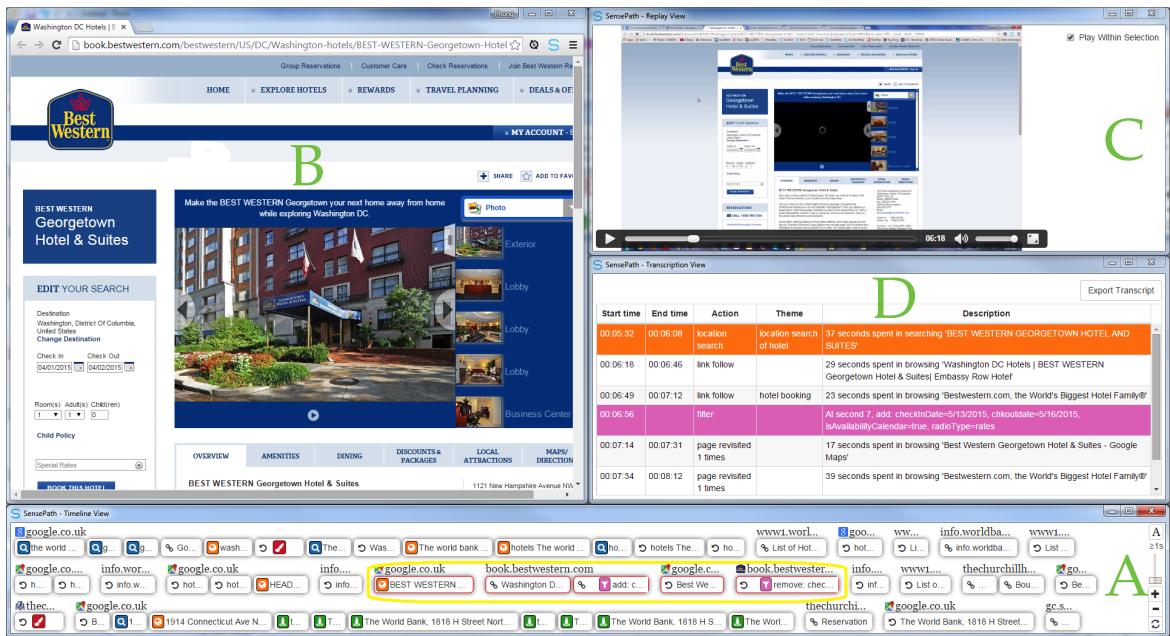


Figure 5.2: Four linked visualizations of SensePath. **A:** The timeline view shows all captured sensemaking actions in temporal order. **B:** The browser view displays the web page where an action was performed. **C:** The replay view shows the screen capture video to provide additional context. **D:** The transcription view details selected actions (highlighted in the timeline) and generates their transcript.

5.4.3 Provenance Capture

During the participatory design session, the HCI researchers suggested to capture rich semantic information that is able to explain the rationale of actions the participant performed. However, this is not always technically feasible. For example, it is possible to detect that a web page has been opened for a long time, but it may be impossible to know whether the participant was reading, thinking, or simply away from the computer screen just by using the information available from the browser alone. Therefore, we agreed to capture the analytic provenance corresponding to the *action* level in the Gotz and Zhou's model [61]. This capture can be done automatically yet still provides reasonable amount of semantics to the researchers. We decided to record the following aspects of actions that were regarded useful for qualitative analysis of sensemaking by the HCI researchers.

- **Type:** When a participant opens a web page, the default type for that action is *browsing*, which lasts until the page becomes inactive. During that period, two common action types are focused: *search & filter* and *reading*. The former type includes *keyword search*, *location search*, *route search* and *filter*. The latter type includes *highlight* and *annotation*, provided for taking notes and capturing part of the participant's thinking.
- **Timing:** This includes the start and end time of an action.
- **Context:** This contextual information provides additional clues for researchers when looking at individual actions. It varies according to its action type such as the “keyword” for search and the “selected text” for highlight. Also, the information common for all action types including title, URL, and a screenshot of the rendered web page are always recorded.
- **Relationship:** This provides how a web page was activated with four cases: *revisit* an already opened page, directly *link* from an existing page, manually *type* a new address, and open from a *bookmark*.

Figure 5.3 summarizes all the action types and relationships captured in SensePath.

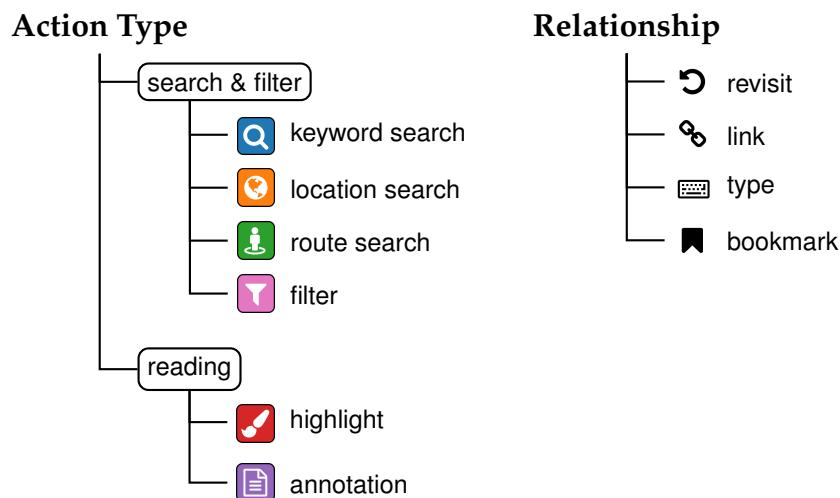


Figure 5.3: All the action types and relationships that SensePath captures, next to the icons representing them.

5.4.4 Timeline View

This view provides an overview of the entire sensemaking process, showing all the captured actions in their temporal order (Figure 5.2A).

5.4.4.1 Visual Representation

An action is represented as either a bar or a tile, presenting all four aspects of provenance information discussed earlier.

Action Bar Figure 5.4 shows an example of an *action bar*. The page URL (context) is displayed atop the bar. In the bar, the first icon shows that this action revisited a previously opened page (relationship). Next is the page title (context); only part of which is shown because of the limited space. This is followed by an icon indicating the type of that action such as a “filter”. Figure 5.3 shows all icons representing action types and relationships in SensePath. Note that action type icons have colored background and a black border to distinguish from relationship icons. The last part is the specialized context for each action type, which is filtering parameters in this figure. The width of the action bar corresponds to the length of time spent in browsing the web page, and the relative position of the action type icon marks when the action happened.



Figure 5.4: An action bar showing all four aspects of provenance information.

Action Title An *action tile* contains similar analytic provenance information but with more details. Figure 5.5 shows the same action as in Figure 5.4 but as a tile. Because more height is given, a tile includes a screenshot, which can help the researcher to recognize the web page more effectively. This can also be useful when the researcher was absent from the observation session because she may get a rough context of what the page was about by looking at its thumbnail. The rest of the provenance information is the same as that in an action bar with more details (e.g., the page title) displayed because of the extra space.

The timeline can be shown with either action bars or tiles, interactively controlled by the user. The former is more compact and scalable, whereas the latter shows

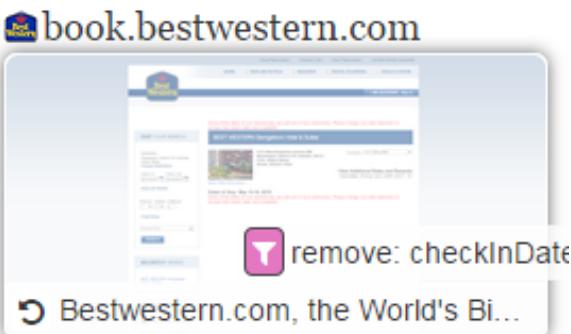


Figure 5.5: An action tile augments the action bar with a page screenshot.

more details and is suitable for a close inspection. Figure 5.6 shows an example of timeline with action tiles.

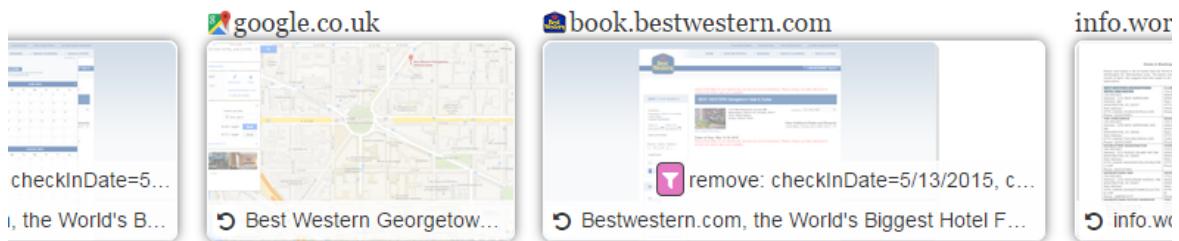


Figure 5.6: A part of the timeline with action tiles.

5.4.4.2 Scalability

Zooming Both action bar and tile can reduce their widths through zooming to accommodate more actions. At the smallest level, only the action type is visible, and more details will become available when zooming in. Figure 5.7 shows three zoom levels of action bars with the details increasing from top to bottom. The top row shows actions with only icons and a few letters. The middle row reveals the last location search is about “Best Western” hotel. The bottom row shows that the first location search is about some “headquarters”, and the second action is revisiting the “World bank” web page. At a low level of detail, only a few letters are shown for each action bar, thus is uninformative. In this case, information richness is sacrificed for scalability. To mitigate this issue, we introduce interactive features such as *selective zooming*, which will be discussed later.

Aggregate Action Instead of showing individual actions, adjacent ones happened on the same web page are merged to save space. It may also help the researcher to

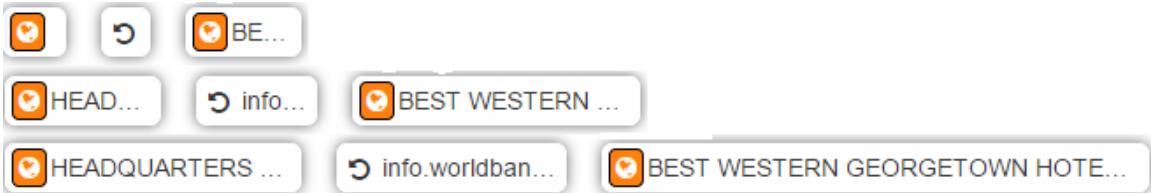


Figure 5.7: Three zoom levels of action bars with the details increasing from top to bottom.

quickly understand the participant's process. Figure 5.8 shows an aggregated action with eight highlights made on the same Google Plus page.

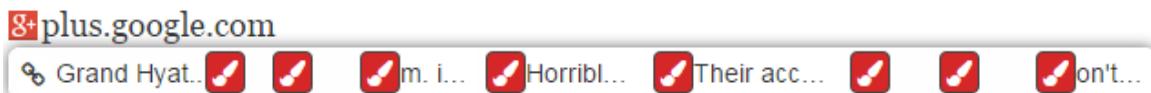


Figure 5.8: An aggregate action bar. It combines eight adjacent highlights made on the same Google Plus page.

Because the action bar is short, a timeline can show multiple rows. This, in combination with aggregation and interaction (described next), enables SensePath to display a reasonably large sensemaking session within a limited space. Figure 5.2A shows about 50 actions out of a total of 70 actions from a 30-minute long session. This addresses Requirement 5 on scalability.

5.4.4.3 Interaction

Mouse Click and Hover SensePath includes a number of interactive features to support the analysis of the sensemaking process. Clicking on an action opens the associated web page in the browser view (Figure 5.2B). This enables researchers to see what the participant was looking at, which is a prerequisite for understanding their thinking. Hovering an action bar highlights other actions happened in the same page with a red border, and brings up a tooltip with additional details such as timing. The example in Figure 5.9 shows that a page was revisited a number of times during a short sensemaking session.

Selective Zooming SensePath implements focus+context technique [?] through *selective zooming*: when a zoom is executed, only a selected set of actions affects. This enables researchers to concentrate on certain actions without losing their context. However, they may forget the difference in zoom levels of actions, thus misunderstand the action lengths indicated by the bar widths. SensePath provides a reset

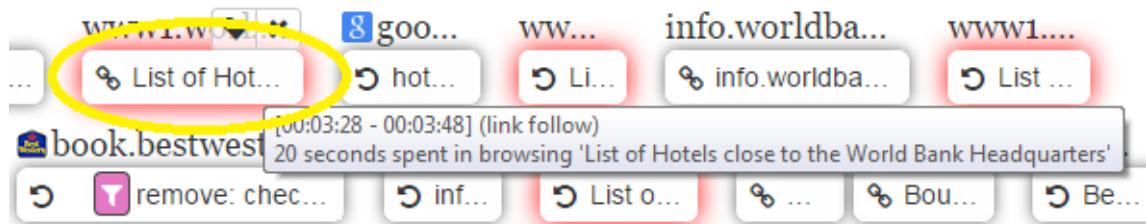


Figure 5.9: Mouse hovering effect. When an action bar is hovered (highlighted with a yellow eclipse), all other actions with the same URL (red borders) are also highlighted, and a tooltip is showing additional information.

button to change the zoom levels of all actions to the default value. Figure 5.10 illustrates this technique.



Figure 5.10: Selective zooming. Selected action bars are with red borders. Top row: before zooming. Bottom row: after zooming – only the selected action has its zoom level changed.

Filtering Researchers can filter actions based on duration, enabling them to focus on the range of actions they want. For example, if researchers think actions that last only a few seconds are trivial, they can be filtered out using a slider (Figure 5.11), which sets a minimal length for visible actions. When the slider moves, actions that will be removed fade out, before disappearing when the slider stops. This enables researchers to preview the effect of filtering.

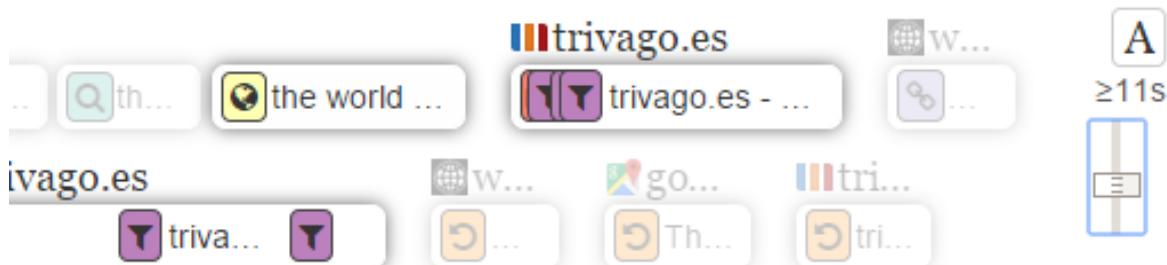


Figure 5.11: Actions filtering. The slider (on the right side) controls the minimal length visible actions. Actions fall below the threshold fade out first before completely disappearing.

Coding In traditional qualitative analysis, researchers analyze transcripts to identify common themes and assign suitable names or codes to them. In SensePath, the timeline view provides a succinct summary of the sensemaking process and allows researchers to drill down to explore more specific actions. Representing action types with icons and visualizing a sequence of actions next together may also help researchers to quickly identify patterns of the data, compared to watching videos or reading transcripts. Coding feature is available through a menu button when hovering an action bar. Researchers can assign a code by simply typing it or selecting from a list of previously entered ones.

5.4.5 Browser View

When an action is selected in the timeline, its associated web page is showed in the browser view (Figure 5.2B). This enables researchers to examine the web page that the participant was looking at when performing a sensemaking action. If the action is an annotation or highlight, the browser view will automatically navigate to the location of the web page where the annotation or highlight was made, informing researchers which part of the page the participant was interested in.

5.4.6 Replay View

SensePath links the timeline to an externally captured screen video to provide additional information about the participant's behavior during the sensemaking session. When a researcher selects an action in the timeline, the replay view automatically jumps to the corresponding part of the screen video when the action is about to start. This avoids manual search within the video, which can be time consuming. After selecting an action in the timeline, a researcher can first check the web page in the browser view and then start the video playback in the replay view if she wants to find out more. The playback automatically stops when it reaches the end of an action, avoiding watching other irrelevant part. Alternatively, the researcher can choose to allow the video to continue; if so, the corresponding action in the timeline will be highlighted as the video progresses.

5.4.7 Transcription View

Detailed information of an action can be revealed by mouse over; however, it is inconvenient to do so for a set of actions. The transcription view addresses this

issue by simultaneously presenting the details for all selected actions, in a tabular format (Figure 5.2D). For each action, this view shows its starting and ending time, action type, assigned themes, and an automatically generated description such as “37 seconds spent in searching Best Western George Town Hotel and Suites”. This description is based on a predefined template for each different action type with advise from the aforementioned participatory design session. The researchers are allowed to edit the description to better reflect what they think. Row backgrounds match the color of action type icons in the timeline view. The design of this view resembles the transcript interface of popular video transcribe software packages to reduce the learning efforts required.

All the information displayed in the transcription view can be exported as a timeline in the SMPTE format¹, which can be imported by many popular qualitative data analysis software packages such as InqScribe² as a transcript. This enables researchers to continue their existing workflows with such software (Requirement 3). Moreover, SMPTE transcript can be used as a subtitle file in popular video players such as VLC³.

5.4.8 Implementation

5.4.8.1 Architecture

SensePath is implemented as a Chrome extension using modern web technologies including HTML5, CSS3, JavaScript and D3.js [15] for visualization. Therefore, it satisfies Requirement 6 about lightweight and support multiple operating systems. Highlight and annotation features require the modification of web page structure, thus they must be implemented as a browser plugin. We decided to target Chrome browser first due to its popularity.

SensePath consists of two components: provenance capture and provenance visualization. The capture component relies on content script injected into a loaded web page (to allow highlight and annotation) and the Chrome extension API (to allow automatic action extraction). Therefore, it always works as long as the Chrome extension is enabled. The captured data is exported as a JSON file, which can then be loaded into the visualization component.

¹ http://en.wikipedia.org/wiki/SMPTE_timecode

² <http://www.inqscribe.com/>

³ http://www.videolan.org/vlc/index.en_GB.html

The four linked visualizations communicate to each other using the *messaging passing* mechanism provided by the Chrome extension API. When an interaction occurs in one view, it sends a message to notify all other views. Each view constantly listens and responds to such messages. For instance, when an action is selected in the timeline view, it broadcasts that selection. The replay view listens and changes the current time frame of the video to when that action was performed. The replay view uses HTML5 video tag⁴ to display the video capture, thus possible to programmatically set the current playback position to a specific point and to start/pause the playback. The replay view also maintains a list of start/end time of all actions, thus when a video is playing, it finds the action that contains the current time frame and sends a message to the timeline view to highlight it.

5.4.8.2 Provenance Capture Techniques

Search Detecting all *search* actions applies URL parsing. When a web page is loaded, its URL is parsed and compared against a set of query templates to check whether a search was performed and to identify its type and parameters. In this prototype, we support automatic detection from many popular web services: keyword search engines (Google, Yahoo, Bing, Ask.com and DuckDuckGo), map search engines (Google Maps, OpenStreetMap and Yahoo Maps), social networking websites (Facebook and Twitter), e-commerce websites (Amazon and ebay) and hotel booking websites (Booking.com and Expedia). All these web services follow their own query templates and expose all necessary parameters in the URLs, allowing extracting the required information from them. The following examples show the query templates and parameters:

- Google keyword search: `https://www.google.com/search?q={keyword}`
- Yahoo Maps route search: `https://maps.yahoo.com/directions/?o={source}&d={destination}`

SensePath uses a structure with three parameters to represent these templates, including a host name (`www.google.`), a path name (`/search`) and a regular expression (`/\wq=([\w%+]*)/i`) for keyword extraction. Adding detection support to other services only requires the knowledge of these three parameters.

⁴https://developer.mozilla.org/en-US/docs/Web/Guide/HTML/Using_HTML5_audio_and_video

Filter Detecting *filter* actions also uses URL parsing as in detecting *search* actions but does not require prior knowledge about query templates. We apply a heuristic that if two consecutive URL requests share the same host name and path name but different parameters, the second page is the result of a filtering action on the first one. More specifically, a URL is compared against its previous URL within the same tab; if they have the same host name and path name, their query strings are parsed to a collection of key/value pairs. For each key, three cases can happen when comparing its value between the previous and the current URL:

- **add:** The key was absent from the previous URL but is added to the current one.
- **remove:** The key was present in the previous URL but is removed from the current one.
- **update:** The key is present in both URLs but their values are different. Note that if their values remain unchanged, it is unnecessary to report.

For example, considering these two consecutive URLs:

1. hotel.com/search?loc=london&guests=1
2. hotel.com/search?loc=london&guests=2&checkIn=2015%2F10%2F24

Following our heuristic of comparing URLs, this filter action can be captured as “`add: {checkIn=2015/10/24}`, `update: {guests=1→2}`”, and may be interpreted as “the participant set a new check-in date and changed the number of guests from 1 to 2”.

Reading Both *highlight* and *annotation* actions are implemented using the content script⁵ in the Chrome extension API, thus all information needed can be saved. To allow revisiting to the exact location where a passage of text is highlighted, the relative location of its DOM element to the root element of the web page is captured. However, when the web page structure is changed, the recorded position might become invalid, leading to inaccurate recovery.

⁵https://developer.chrome.com/extensions/content_scripts

Limitation All the heuristics applied for search and filter actions only work if web services expose their parameters in URL. In other words, our method fail to work if POST or AJAX requests is used. So far, for all services we planed to support, we have never encountered such a case. For example, Google Maps uses AJAX calls to load map tiles, but all the information we want to extract is available in the URL. Also, most popular online services use GET instead of POST requests. It is actually possible to support web services that encode the required information in POST or AJAX requests by monitoring all the communication between the browser and the server, not just the changes in URL. However, this requires considerably more implementation effort and is only possible with open source browsers such as Firefox and Chrome that provide access to all client-server communication.

For GET requests, it is also infeasible to extract the meaning of URL parameters if they are encoded. We encountered one such case, Bing Maps, which encodes query parameters as HEX strings. For example, this lengthy URL, “<https://www.bing.com/maps/#Y3A9NTEuNTkwMTk5fi0wLjIyNTAwMCZsdmw9NiZzdH...>”, is the URL produced by searching for “london”.

To guarantee an exact restoration of a visited web page, it is necessary to save a static copy of the web page when an action happened. This is similar to the *P-Set model* [84] for visualization exploration where the visual transforms, parameters, and results are stored for fully describing and reproducing the performed process. However, for simplicity, we only capture the action type (i.e., visual transforms) and context (i.e., parameters), but not the resulting web page (i.e., results). The browser snapshot and computer screen video are captured to compensate this limitation to a certain extent.

5.5 Evaluation

We conducted two user-centered evaluations: the first one is to investigate how SensePath is used by a domain expert, and the second one is to discover whether SensePath has any potential advantages compared with a traditional method. We first conducted a number of user studies of participants carrying out an online sensemaking task to establish a ground truth dataset. Then, we recruited HCI researchers to analyze the sensemaking processes of these participants. In this section, we regard these HCI researchers as “analysts” and people performing sensemaking tasks as “participants” or “sensemakers”.

5.5.1 Online Sensemaking Task

We recruited two participants to take part in this study: a post-doctoral researcher and a PhD student, both males. Participants were given the same task, which was to use the Chrome browser to find appropriate accommodation for two academics attending a conference at the World Bank headquarters in Washington, D.C. We provided participants with information about the location and dates of the conference, but gave no further details in the scenario to maintain suitable complexity in the task, and to ensure it was as realistic as possible.

Both participants were given 30 minutes to perform the task. Throughout the study, their sensemaking actions were collected within SensePath, and their screens were also recorded using a commercial software. We also encouraged participants to think aloud throughout the study, and finally conducted a semi-structured interview asking them to reveal:

- The rationale behind their choice
- The strategy in approaching the task
- The process they went through in executing that strategy

5.5.2 Evaluation 1

5.5.2.1 Design

The goal of this evaluation is to explore how SensePath is used by a domain expert performing a real-world task. We recruited an analyst with seven years of experience in qualitative research to analyze the actions captured in the sensemaking process outlined earlier using SensePath. We gave the analyst a short tutorial to introduce and explain features of SensePath, before practicing with a trial task. She was provided with a laptop running SensePath, connected to an external monitor, providing a multi-screen setup as in Figure 5.12. During the analysis, we encouraged the analyst to provide feedback through a think-aloud protocol. We recorded her responses and other observations using written notes. At the end of the analysis, we asked the analyst to complete a discovery sheet reporting her findings. A semi-structured interview was followed up to gain deeper understanding of her experience. The discovery sheet included the following questions:

- Identify the sensemaker's strategy in approaching the task and characteristics demonstrating that.

- Identify the steps the sensemaker took in choosing suitable accommodation and provide a name (code/theme) for each step.
- Identify any interesting patterns in the data.

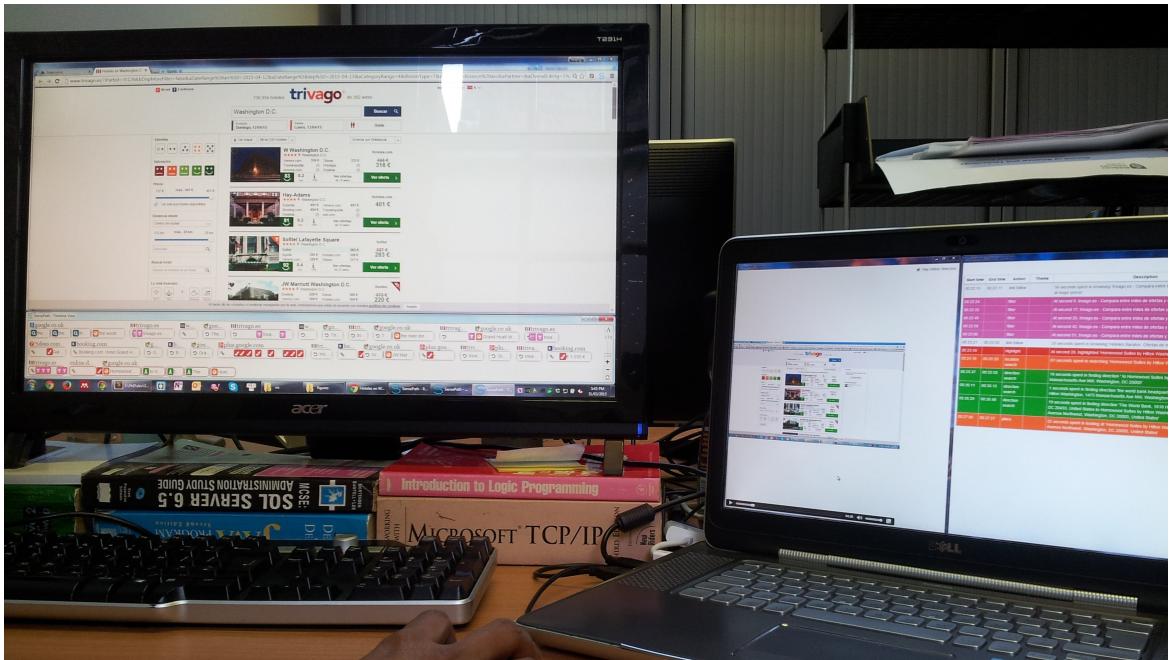


Figure 5.12: The setup of the qualitative analysis with SensePath. The monitor on the left shows the timeline and browser views, and the laptop on the right shows the replay and transcription views.

5.5.2.2 Findings

The analyst took approximately one hour to analyze 30 minutes of study data. This shows a reduction of half the time taken using SensePath compared with traditional methods of video analysis, which would typically take around four hours of analysis for every hour of study data [22].

The analyst initially used the timeline to view sensemaking actions at a low resolution, before focusing on interesting parts of the data in more detail. Because these actions are visualized in a single view, the analyst reported she could quickly make an initial summary assessment of the sensemaker's overall performance of the task, before identifying potentially interesting behaviors in the data. One such example of this is when she saw many highlights on a Google Plus page, next to each other in the timeline, she said "It seems that the guy [the sensemaker] found

interesting information on that [Google Plus] page because he highlighted a lot there". She then moved the mouse over these highlight icons to read the highlighted text shown in the tooltips. Interestingly, she quickly concluded that "He only focused on negative reviews". She clicked on some of those icons to open up the Google Plus page to gain more context. Unfortunately, that page is content-dynamic, thus some highlighted texts failed to be reselected. She switched to watch the video in the replay view and heard that the participant was talking to us about his preference to negative reviews (we used think-aloud protocol), which confirmed her initial judgment. She also mentioned that offering highlight feature to the sensemakers is useful because it enables her to quickly identify their interests.

To understand the whole sensemaking process, the analyst quickly went through all the actions shown in the timeline. The analyst was able to successfully describe the steps taken by the sensemakers in approaching to the task. We confirmed those steps were all correct by re-watching the capture video and verifying with the sensemakers. Figure 5.13 shows a reproduction of a written diagram created by the analyst illustrating the steps she identified. She explained the diagram to us; for instance, "that guy searched for the address of the headquarters, then viewed it in Google Maps to get a sense of where it is".

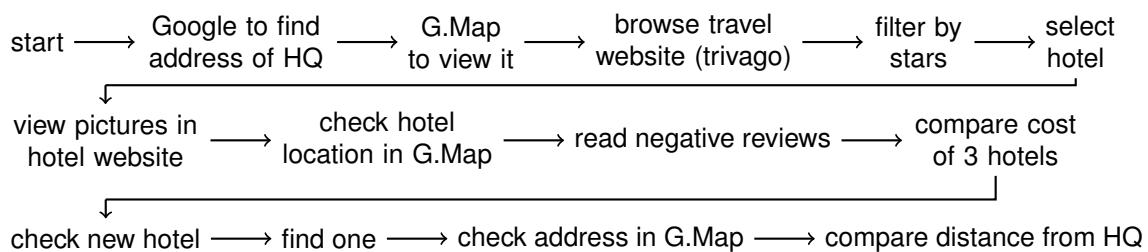


Figure 5.13: Sensemaking steps. A reproduction of diagram written by the analyst during the evaluation illustrating the steps taken by the sensemaker in the task.

The analyst reported that using the timeline view she could easily identify interesting recurring patterns because all actions are shown together. As an example for such patterns, when the sensemaker found a hotel in a booking website, he looked at its pictures first, then searched for its location in Google Maps, and checked its route to the headquarters. The sensemaker repeated this process for several hotels he found.

The analyst managed to find an overall strategy that the sensemaker applied in approaching the task: he targeted reasonably cheap hotels (evidenced by filtering out 5-star ones) and considered hotels close to the headquarters (supported by putting

them in Google Maps for comparison). This confirmed with what the sensemaker told us: as a professional academic, he wanted to save money for the university, but also did not want to stay too far away from the conference venue.

The analyst commented that the video and audio recordings were intrinsic to carry out a fuller, more detailed analysis by providing additional information that was unavailable in the timeline and the browser views such as mouse movement or page scroll interactions. Therefore, the replay view helped her gain further insight into the sensemaker's behaviors. Because the analyst did not have to watch the whole video, she thought that she could save valuable time in the analysis. Moreover, she stated that as clicking on an action in the timeline view skipped to the relevant place in the screen capture, further time was saved in scrubbing through the video, which often happened in her experience of analyzing video data. For example, when seeing a long search action bar, she knew that the sensemaker spent much time in Google Maps, searching for a specific location; however, what exactly he was doing is neither available in the timeline nor the browser view. Thus, she needed to watch the video to get more information.

5.5.3 Evaluation 2

5.5.3.1 Design

The goal of this evaluation is to establish an initial understanding of whether SensePath has any advantages compared with a traditional method. We conducted an experiment with two senior HCI researchers to analyze a short 15-min video (part of the video used in the first evaluation) using thematic analysis, but only for the first two stages: transcription and coding, which SensePath is designed to support. One analyst used SensePath and the other used a transcription software called *Transcription*⁶. This tool shows the transcribing video and a text editor side by side, and provides a set of shortcuts for video manipulation such as pause/play or step forward/backward. This enables users to control the video play while focusing on the editor to type in the transcript, thus accelerates the transcription. The experiment setup and procedure were the same as in the first evaluation. The analysts were asked to produce a transcript of the video, identify the steps the sensemaker took in performing the task, and assign codes for them. Timing for each analysis stage was recorded, and an interview was followed after the analysis to gain deeper understanding of their processes.

⁶<https://code.google.com/p/transcriptions/>

5.5.3.2 Findings

The result showed that SensePath can significantly reduce time for transcription. In the baseline condition, it took the analyst 60 minutes to transcribe the video, about 15 minutes of them spent for the think-aloud audio. In SensePath condition, transcribing was done automatically, and it took the analyst 5 minutes to traverse all actions in the timeline to get a sense of the sensemaker's process.

In regard to accuracy, SensePath identified 39 actions and all are correct (each action is verified by us). In the baseline condition, the analyst identified additional actions that did not make the browser reload such as "click on a pull-down menu to see more options" or "encircle the rating point with mouse movement". However, he missed two actions that were correctly identified by SensePath: "set checkin and checkout date" and "change sorting criteria".

Watching the video helped the analyst transcribe with richer information. For instance, SensePath can only record where a web page came from such as manually typing the address or linking from another page. Whereas in the baseline condition, more context can be added such as which button was clicked to open that new page. However, SensePath can save time from transcribing long text such as for highlights or annotations.

For coding, in the baseline condition, the analyst copied the transcript to Excel and assigned codes in a column next to the text. The analyst using SensePath can directly assign codes to actions in the timeline view. SensePath was only slightly faster than the traditional method (40 minutes vs. 45 minutes). It could because the analyst using SensePath needed to familiarize himself with the data by watching some parts of the video a few times, whereas the other analyst already spent long time watching the video during the transcription stage.

In both conditions, the analysts produced reasonable and comparable codes for the steps they identified. For example, in the baseline condition, the analyst created codes such as "Google search for location", "drill down", "assessment" and "assessing relevance of evidence". The analyst with SensePath produced similar codes including "looking for headquarters location", "locate option", "assess proximity", "assess price" and "assess reviews".

Similar to the analyst in the first evaluation, the analyst with SensePath also quickly went through all sensemaking actions to have an overview of the process, before drilling down to actions of interest. Especially, the analyst often used actions in the timeline as a navigation to the part of the video he wanted to watch. He said he needed to watch some parts of the video several times to understand the

intention of the sensemaker, and clicking on the actions enables him to quickly go to the correct part.

5.6 Summary

In this chapter, we introduce a visual analysis tool SensePath enabling users to explore temporal relationship of the sensemaking process. It facilitates thematic analysis of semi-automatic provenance data for online sensemaking tasks, targeting the transcription and coding phases. The data is visualized in four linked views: timeline, browser, replay and transcription. The timeline provides an overview of the sensemaking process and can support a reasonably long sensemaking session common in qualitative research observations. The browser view shows the web page the participant was looking at when performing a sensemaking action, and is complemented by the replay view with the screen capture video of the action. The transcription view provides all the details for a set of actions and can export the information in a format compatible with popular qualitative analysis softwares such as InqScribe, so that analysts can continue working in their existing workflow.

An evaluation was conducted with an experienced qualitative researcher, who found many features of SensePath helpful for her work, and the data collected from the observation showed that SensePath met most of the requirements it set out to achieve. Another evaluation with two HCI researchers suggested the potential of improvement in completion time for SensePath in the transcription and coding phases.

SensePath can be extended to support exploring rational relationship in other domains beyond online sensemaking tasks. The visualization component can be reused straightforwardly. However, the capture component of SensePath is currently tightly associated with extracting sensemaking actions in a web page, thus needs to be updated. Of course, a discussion with targeted users is required to understand what actions and information are important to capture. Also, to make this more accessible for non-technical users (such as the analysts) in adding automatic detection of “search actions” from new web services, we plan to build a simple GUI, in which they can specify the search templates, thus save the effort of manually modifying the code.

6

Exploring Complex Rational Relationship

Chapter 5 uses timeline visualization of automatically captured user actions to enable researchers to explore rational relationship in the user's sensemaking process. In this last block of the thesis, we investigate whether and how analytic provenance can help users in their own *ongoing* sensemaking processes. Very often, users get lost while solving a complicated task using a big dataset over a long period of exploration and analysis. They may forget what they have done, may be unaware of where they are in the context of the overall task, and may be unsure where to continue. Visualizing the provenance of user actions to provide an overview of the sensemaking process could help address this issue. Also, during sensemaking, users can interact with their provenance information and have opportunities to contribute additional knowledge to the automatic capture such as by grouping and connecting related information. This richly semantic organization could help users understand the relationship of their individual findings better and remembering them more easily.

In this chapter, we introduce a visual analysis tool – *SenseMap* – that enable users to explore **complex rational relationship** of sensemaking through the visualization of such **temporal and relational** provenance information, focusing on the context of *browser-based online sensemaking*. We conducted a semi-structured interview with nine participants to explore their behaviors in online sensemaking with existing browser functionality. A simplified sensemaking model based on Pirolli and Card's model is derived to better represent the behaviors we found: users iteratively *collect*

Previously published at the *IEEE Conference on Visual Analytics Science and Technology* [122].

information sources relevant to the task, *curate* them in a way that makes sense, and finally *communicate* their findings to others. A series of design workshops was followed to derive requirements, discuss designs, implement and test the prototype in an agile setting. SenseMap provides a multi-linked visualizations of analytic provenance and enable users to curate and communicate their findings. To explore how SenseMap is used, we conducted a user study in a naturalistic work setting with five participants completing the same sensemaking task related to their daily work activities. All participants found the visual representation and interaction of the tool intuitive to use. Three of them positively engaged with the tool and produced successful outcomes. It helped them organize information sources, quickly find and navigate to the sources they wanted, and effectively communicate their findings.

6.1 Introduction

People often get lost while solving complicated tasks using big datasets over long periods of exploration and analysis. They may forget what they have done, fail to find the information they have discovered before, and do not know where to continue. One approach is to capture and visualize user interactions in such a way that provides an overview of the sensemaking process to the user. In the World Wide Web context, the aforementioned problem is known as the *disorientation* problem [34]. One approach to address this problem is through a graphical browser history [12, 80, 116]. It visualizes visited web pages and the linking relationships between them to help users to quickly see where they are in the network and to navigate to the page they want. However, when solving a sensemaking task online, which requires gathering, restructuring and reorganizing lots of information to gain insight, the disorientation problem becomes more severe and difficult to address. They do not just get lost in the hypertext space but also get lost in the task space. They may be unable to answer the following questions. What has been done so far? Where am I in the context of the overall task? What information should I search for next?

In this chapter, we introduce a tool, *SenseMap*, to support *browser-based online sensemaking* through analytic provenance. We targeted this domain because many everyday sensemaking tasks such as travel planning are now performed online [151]. We followed a user-centered, iterative design process to address the problem. First, user behaviors in online sensemaking are elicited through interviews. Then, a simplified sensemaking model based on Pirolli and Card's model [136] is derived to better

represent these behaviors: users iteratively *collect* information sources relevant to the task, *curate* them in a way that makes sense, and finally *communicate* their findings to others. A series of design workshops was followed to derive requirements, discuss designs, implement and test the prototype in an agile setting. SenseMap consists of three components: a *browser view* that is a standard web browser with additional sensemaking support, a *history map* that provides an overview of the sensemaking process, and a *knowledge map* that allows users to curate the collected information. Communication support is provided in all three components.

To explore how SenseMap is used, we conducted a user study in a naturalistic work setting with five participants completing the same sensemaking task related to their daily work activities. Both quantitative data about user activities with SenseMap and qualitative data through semi-structured interviews were collected. All participants found the visual representation and interaction of the tool intuitive to use. Three of them positively engaged with the tool and produced successful outcomes.

SenseMap is freely available as a Chrome extension¹. In summary, it contributes

- A user study exploring user behaviors in online sensemaking with existing browser functionality, and a series of workshops followed up to generate requirements and discuss designs.
- A visual sensemaking tool SenseMap supporting browser-based online sensemaking addressing all the derived requirements.
- A user evaluation exploring how SenseMap is used in a naturalistic work setting and a discussion of insights gained and design lessons learned.

6.2 Approach

We followed a user-centered, iterative design process to develop SenseMap as a tool supporting browser-based online sensemaking. First, we identified current user behaviors in sensemaking using existing browser functionality. These behaviors led to the selection and subsequent development of a sensemaking model for user behaviors on the web. We conducted a series of design workshops to derive requirements using these user behaviors and model, then to design, build and test a prototype in an agile setting. Finally, the prototype was evaluated in a naturalistic work setting. Figure 6.1 summarizes this process.

¹<https://chrome.google.com/webstore/detail/sensemapp/agljinpanahlilmpipaeeflmnjkiiecfjb/>

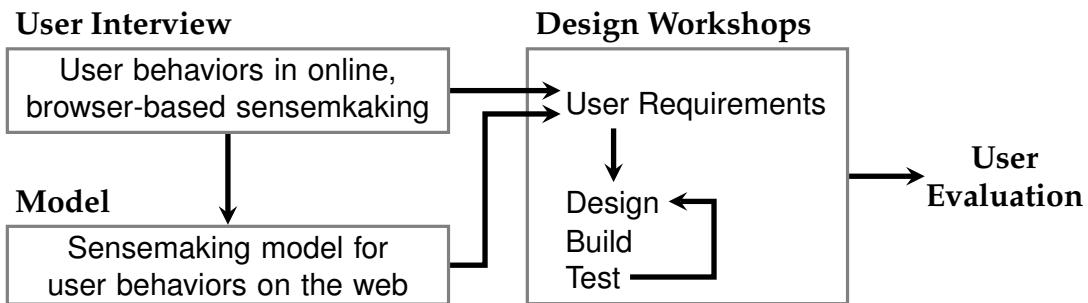


Figure 6.1: Summary of the design process.

6.2.1 Design Research

We conducted a semi-structured interview with nine participants to explore their behaviors in conducting online sensemaking for their daily work activities. The interview happened during a normal working day to access the currently open, in-use browsers of participants, as a representative artifact of their practice. Hence, the participant's browser became the scaffold for the conversation and provided the ongoing probes as the conversation unfolded. This method also ensured that participants talked about what they actually did rather than what they thought they did or should do.

We took video of each interview with the camera showing the interviewee's laptop screen and their hand gestures. We also made screen recordings of each laptop while the interview was taking place. Figure 6.2 shows the combination of hand gesture and laptop screen of a participant. Each interview began with the participant showing their currently open browser windows. Browser choice was discussed and then the ongoing conversations were guided by five browser functions: searching, tabs, windows, bookmarks, and history, with participants illustrating their behaviors using their in-use browsers. These behaviors are summarized as follows.

Starting Searches Opening a new tab preceded most searches. Users spoke about new tabs helping them to manage information, keep things separate and how they could go back to other pages that were relevant to their work activities and ongoing investigations.

Tabs Eight of the nine participants had a number of tabs open and categorized them as either: collections of tabs relating to current investigations or single points of

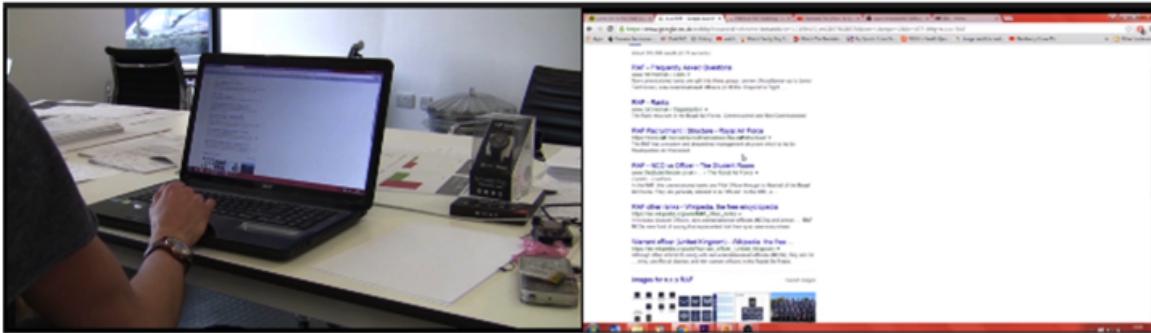


Figure 6.2: Interviewee's hand gesture and laptop screen.

access to commonly accessed services, e.g. social feeds, email etc. In further probing about the tab collections a number of shared behaviors emerged.

1. Opening a new tab if “significant” information is found enabling the page to stay live in the browser.
2. Opening Google search result links in a series of new tabs from one search page. Subsequent tabs were reviewed and then kept or closed based on their significance.
3. Reordering tabs to develop a narrative. In all cases the narrative was described as flowing from left to right. The narrative was used by the participants to make sense of the information found, to develop more refined search strategies and terms where information was lacking, and to communicate their findings to others.
4. All participants expressed anxiety about losing tabs when they were inadvertently closed or lost due to a system error and they all described the same recovery procedure using the recently closed tabs section of the History menu.
5. The number of tabs in browser windows varied greatly across the participants. One participant diligently closed all tabs at the end of each “work episode” although sometimes they kept them open in a non-active window when at home and used a new window for private web browsing. Most described groups of seven to eight tabs that were currently in use for active projects. One user had over fifty tabs open in their main browser and twenty in their second browser, but they gave the same explanation for their presence, use and organization.

Windows Only one user described the use of more than one window in the web browser. Similarly to Behavior 5, this enabled him to keep work-related tabs separate from private browsing.

Bookmarks There was considerable variance in the use of browser bookmarks although most had moved away from using them and relied instead upon tabs to keep relevant information live and accessible. Two participants had no bookmarks at all. One participant saved some bookmarks, but these were not organized into groups, categories or folders. One participant described a behavior where they bookmark the contents of tabs at the conclusion of a project and organize these into named folders. However, they rarely revisited these bookmarks to use them to access information again, instead preferring “Pinterest” or relying on Google to find it again from a search term.

History None of the participants made use of the history menu to revisit pages or to make sense of recorded information. However, all of them used it to reestablish a tab if it had been inadvertently closed.

6.2.2 Sensemaking Model

We considered the relevance of extant sensemaking models to the elicited behaviors, principally Pirolli and Card’s model [136] and Data-frame model [95]. The iterative process of sensemaking described by Pirolli and Card effectively encapsulates the observed tab behaviors:

- The *foraging loop*: behaviors 1 and 2
- The *sensemaking loop*: behavior 3

Behaviors 4 and 5 indicate possible tool features rather than a step described in Pirolli-Card model.

The synthesis of our observed behaviors with the Pirolli and Card’s model indicates a browser-based sensemaking process, during which information sources are held in a *collection* of browser tabs (foraging loop), with each tab containing the provenance for the source. An ongoing *curation* process (sensemaking loop) takes place where tabs are ordered into categories and a narrative sequence unfolds within such categorized groups. These groups and relationships represent the underlying schema. The results of the curation are then used to guide further more refined

searches and, on completion, as a support to *communicate* the findings to others. Figure 6.3 illustrates our refined model.

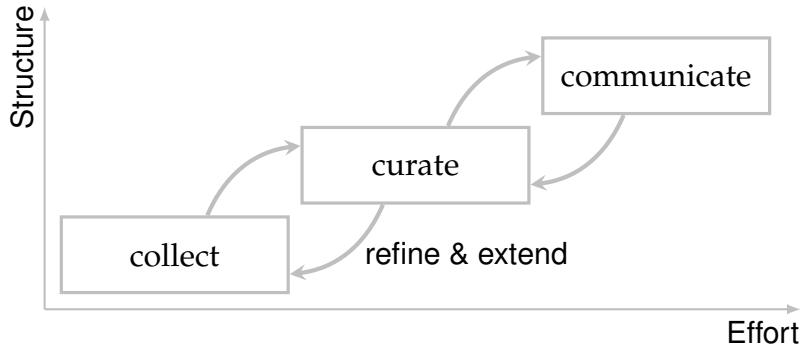


Figure 6.3: Our refined sensemaking model for user behaviors on the web.

6.2.3 Design Workshops

We organized a series of iterative design workshops to derive and satisfy requirements with an overall aim to support and augment current browser-based online sensemaking activities. In the first workshop, an initial design was proposed, detailing visual representation and user interaction. A prototype was built based upon this proposal, and subsequent workshops sought to develop this tool through the ongoing interplay between design, build and test in an agile setting.

We will describe the requirements next and present the interface design in Section 6.3. Some of these requirements link directly to observed behaviors, some are inferred from our sensemaking model, and some are produced during creative design processes set within the constraints of the technology platform chosen.

6.2.3.1 Collection Requirements

1. **Rich provenance:** enrich and make the provenance of information sources more visible to users. Currently, the provenance of tabs is only accessible when they are active and then only by a list of page titles (in Chrome, press and hold the browser's back button), which requires users to build their own schema that is external to the browser.
2. **Easy revisit:** provide quick and easy mean to revisit the information sources needed. Our interviews show that users often revisit their important tabs (Behaviors 1 and 2), but rarely use bookmarks and history. During a

session, they rely on the tab titles, their memories or trial-and-error. However, tab titles are represented by a favorite icon and a truncated page title, which is a poor abstraction from the original source. This abstraction becomes poorer as more tabs are opened, making revisititation difficult when tab collections are large.

3. **Location awareness:** provide an overview of the sensemaking process to address the disorientation problem [34], enabling users to know what they have done so far, where they are in the context of the overall tasks, and potentially guide the next step.
4. **Preparation for curation:** provide highlight and annotation support for users, which can facilitate more elaborate thinking [155], and can serve as a step to assess the relevance of the information sources. The information representation should have different levels of richness depending on the assessed relevance.
5. **Interruption & Separation:** enable task switching without compromising the collection process; for instance, checking email or social feeds should not get recorded as part of the sensemaking process (Behavior 5).

6.2.3.2 Curation Requirements

6. **Rich representation:** provide a rich abstraction of the information source allowing the user to quickly recognize it [169]. This also relates to the “Easy revisititation” and “Location awareness” requirements for Collection.
7. **Spatial organization:** enable users to freely arrange information sources in both x and y dimensions to address the limit of a one-dimensional sequence of tabs being used to visualize multiple narrative threads (Behavior 3). Spatial organization plays an important role in sensemaking [155], especially with a large space [11].
8. **Linking/unlinking:** enable further curation of these sources by establishing links, which is impossible with existing browsers. Linking and unlinking are also known to help users to produce more critical thinking [155].
9. **Formal reasoning:** enable users to apply formal argumentation methods such as Toulmin’s argument [175] or Wigmore’s chart [58]. We think that they may be helpful when solving complex sensemaking tasks analytically.

10. **Collection – Curation:** enable users to see connections between the curated and collected sources, and to use these to inform further searches. This is to support the “refine and extend” direction in our sensemaking model (Figure 6.3).

6.2.3.3 Communication Requirements

11. **Complete picture:** provide a complete picture of the curated sources and the relationships that a user ascribes to them via their curation activity. Currently, it is impossible to see an overall picture of the curated sources and their categorization from the sequence of tabs.
12. **Auditability:** enable users to refer to raw data as evidence supporting their reasoning, which is considered as an important characteristics in analytic presentation [28].
13. **Varied audience:** enable users to customize the curated set of information to suit various needs and backgrounds of the audience. This is also another important characteristic in analytic presentation [28].
14. **Sharing:** enable users to share both raw and curated sets of information with others. This is a first step toward a collaborative environment for online sensemaking.

6.3 Interface Design

6.3.1 Approach

In the initial design session of the workshops, we considered all elicited requirements and agreed that SenseMap needs to:

1. Capture web pages that the user visited, the sensemaking actions that happened there, and how the user arrived at those pages.
2. Visualize the captured information in such a way that the user can understand what they have done, how things are connected, and what else they may do next.
3. Support the user to curate the collected information according to its relevance, facilitate their reasoning, and communicate the findings. Also, this should not

interfere with the original relationship among collected information so that the user can always use it as a reference.

6.3.2 Overview

SenseMap consists of three views:

- A *Browser View* (Figure 6.4A) that is a standard web browser with additional sensemaking support and provenance capture of actions happening there.
- A *History Map* (Figure 6.4B) that shows captured sensemaking actions with their page linking provenance while preserving their temporal order as much as possible to provide an overview of the sensemaking process (Point 2 above).
- A *Knowledge Map* (Figure 6.4C) that allows users to curate the collected information. This map is separate from History Map to preserve the semantic and temporal structure of the captured information (Point 3 above).

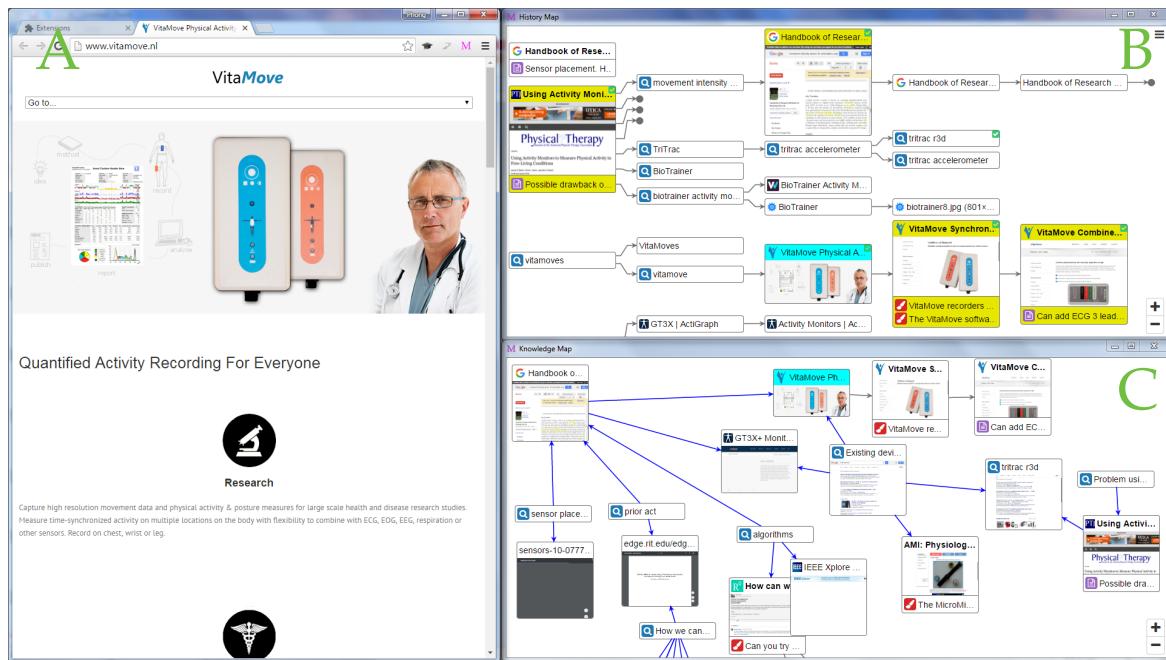


Figure 6.4: Three linked views of SenseMap. **A:** This is the standard browser with additional sensemaking and provenance support. **B:** The history map captures and visualizes user actions to provide an overview of the sensemaking process. **C:** The knowledge map enables users to curate and make sense of the most relevant information to their tasks.

In the next three sections, we will discuss these views and how they address the design requirements. Figure 6.5 summarizes the support that each view provides.

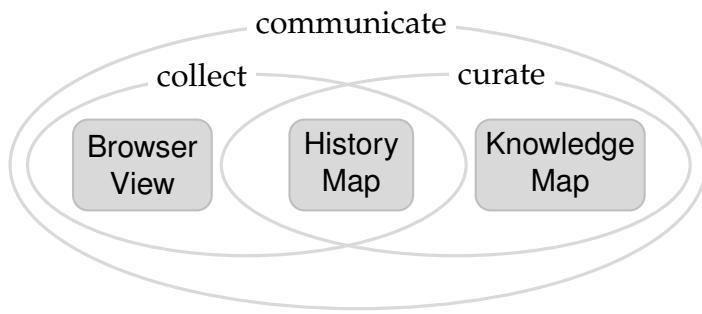


Figure 6.5: Views and the sensemaking subprocesses they support. *Collection* is supported in Browser View and History Map. *Curation* is supported in History Map and Knowledge Map. *Communication* is supported in all three views.

6.3.3 Browser View

This is a standard web browser with provenance capture and additional sensemaking support. The content and method to capture such provenance is the same as in the SensePath tool, Section 5.4.3. Additional features augmented to the browser are described as follows.

Highlighting and annotation are essential editing support. They allow users to mark relevant information and to assign their own interpretation (Requirement 4). A new option “Highlight” is added to the context menu when a passage of text is selected allowing the user to highlight it. That text becomes clickable allowing the user to either write a note or remove the highlight.

When a web page is visited, SenseMap takes a screenshot and uses it to represent the page in the history map (Section 6.3.4). It is intended to help the user to quickly recognize web pages that have been visited (Requirement 2). However, that screenshot may not reflect perfectly the main content of the web page, especially when it contains lots of text. To address this issue, we allow the user to assign a custom representative image to a web page. This can be done by simply right-clicking on any images in the web page and select “Set as Page Image” option in the context menu.

6.3.4 History Map

This map provides an overview of the sensemaking process using the captured actions and their provenance (Figure 6.4B).

6.3.4.1 Visual Representation

An action is represented as a bar with an icon indicating its type and text showing the contextual information. Icons help users to recognize action types faster and we use the same icon set as shown in Figure 5.3. If the action type is the default *browsing*, the favorite icon of its web page is used instead. The contextual text is important to understand what the action is about and it is truncated up to a certain length because of the limited space. Figure 6.6 shows an example of a keyword search action.

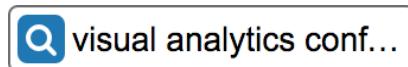


Figure 6.6: Action bar for a keyword search “visual analytics conference”.

Highlights and annotations of the same web page are grouped together as in Figure 6.7. They are located in separate rows below the web page title. By default, just a few highlights and annotations are shown to ensure a reasonable height for the page. All of them can be revealed using a menu available when hovering on any highlight or annotation.

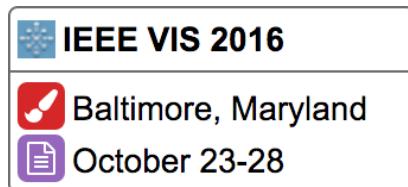


Figure 6.7: A page with one highlight and one note.

To help provide a connection between the history map and the browser view, the action bar corresponding to the active browser tab is highlighted in cyan. Pages that have been opened but have not seen yet (could be the result of opening links in new tabs) are shown with a dashed border, which may help to remind the user on reading them. Figure 6.8 shows an example of pages with these two states.



Figure 6.8: The user is active on a search result page (left bar) and opens a link in a new tab (right bar).

6.3.4.2 Layout

Seeing the provenance of a web page is important to the user (Requirement 1). Currently, it can only be seen if the user presses and holds the browser's back button. This feature is not even available if a page is open in a new tab. In the history map, linking relationships between two pages are always visible and illustrated by an arrow pointing from the source to the target (Figure 6.8). For example, if the user clicks on a link in page *A* yielding to page *B*, an arrow from *A* to *B* will be added to show this relationship. Showing links between pages can reveal branching structures such as when multiple pages are opened in new tabs from the search result page. This provides richer provenance information and easier access for the user compared to a linear list of visited pages as in current browsers (Requirement 1).

Technically, all pages and links in the history map form a *forest*, where tree roots are pages that do not have a parent page such as pages opened by entering the URLs manually. Temporal information of sibling pages are indicated by the order of them: earlier opened pages are placed above later ones. This also helps to maintain the mental model for the user about their process: the order of pages are never changed; and a new page is added either on the right side of the page triggering its opening or at the bottom of the map when such linking does not exist. A virtual node is then added and connected to all tree roots to form a single tree. We use the compacted tree layout in *jgraph* library² to produce the location of pages (Figure 6.4B).

Temporal information shows the order of actions that the user has taken, and the branching and linking relationships reveals their semantics. At a lower level, highlighting the active tab in the layout as described earlier helps the user to know where the page they are focusing is in the context of the overall process. Both these supports address Requirement 3.

6.3.4.3 Preparation for Curation

The history map displays all captured actions; however, probably not all of them are equally important and relevant to the sensemaking task. Therefore, it is necessary to allow users to assess the relevance of the collected information. We use the term *node* to refer to either a simple search action bar or a page containing many highlights – a

²<https://www.jgraph.com/>

node in the tree layout. Three levels of relevance are provided, all through the menu available when hovering a node.

1. If a node is completely irrelevant, the user can *remove* it.
2. If a node is not quite relevant but the user wants to keep it to have a look at some point, they can *minimize* it.
3. If a node is very relevant, the user can *favorite* it.

When a node is removed, it and its links are removed from the map. When a node is minimized, it is collapsed into a small circle. This enables users to focus on other nodes and also save the display space. Favorite nodes are displayed with a yellow background and a thumbnail of the captured screenshot to increase their recognizability. Figure 6.9 shows an example of minimized and favorite nodes.

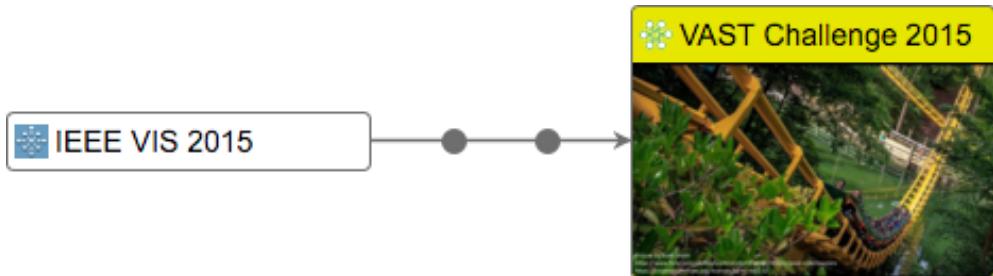


Figure 6.9: Nodes are pre-curated: two irrelevant nodes in the middle are minimized, whereas the last one is set favorite.

6.3.4.4 Scalability

Nodes can reduce their size through zooming to accommodate more nodes within the visible part of the history map. By default, all nodes have the same width and the same maximum height, which allows a few words of the contextual text visible, and a reasonably large thumbnail image, which may help users recognize the visited pages. For each smaller level, both the node width and the number of highlights are reduced. The maximum height should be adjusted so that the ratio between it and the node width remains unchanged. At the smallest level, only the action type icon or a small thumbnail image is shown. Figure 6.10 shows an example of different zoom levels applied onto the same node.

Node zoom level is explicitly controlled by the user using simple plus/minus buttons. When the collection of nodes exceeds the visible area, the user can pan the map to see them.

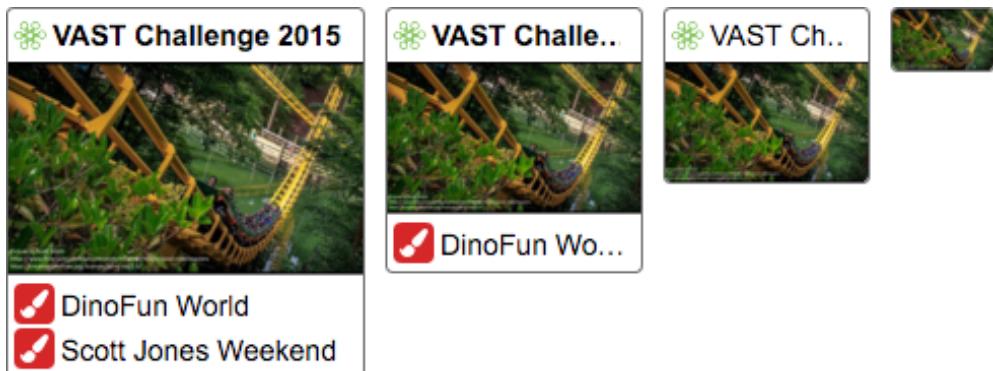


Figure 6.10: The same node with four zoom levels.

6.3.4.5 Revisitation and Interruption

When an action is captured, its web page's URL is also recorded. Clicking on a node opens its associated web page. This releases the user from worrying about losing browser tabs. Moreover, we think that the additional branching and linking structure of the layout will help the user to find information faster than the History feature of the browser (Requirement 2).

To provide a finer grained navigation than the web page URL level, revisiting a captured highlight brings the user to the exact text being highlighted. This is made possible by capturing the relative location of the highlight with respect to the root of the web page.

In the real world environment, the user may have many sensemaking tasks happening at the same time (Requirement 5). Even when working in a single task, the user may do some other things irrelevant to the task such as checking email and social feeds. Therefore, always capturing user actions and putting them into a single place will result a huge mix of unrelated information. To address this issue, we allow the user to create separate collections of information for different tasks. The user can also pause the information capture and resume when needed.

6.3.5 Knowledge Map

This map allows users to curate the information displayed in the history map (Figure 6.4C).

6.3.5.1 Visual Representation

The curation process starts by adding nodes from the history map to the knowledge map. This is done via the *Curate* button in the menu available when hovering over a node. Nodes in the knowledge map have the same visual representation with those in the history map. The only difference is that thumbnail images of curated nodes are always made visible to improve their recognizability (Requirement 6).

6.3.5.2 Spatial Organization

The limit of single dimensional ordering tabs from left to right is addressed in the knowledge map through the spatial organization of nodes (Requirement 7). The user can freely move nodes by simply dragging them around. This enables the user to spatially group nodes and to assign different meanings to them. Figure 6.11 shows an example of a knowledge map with three clear groups based on their locations.

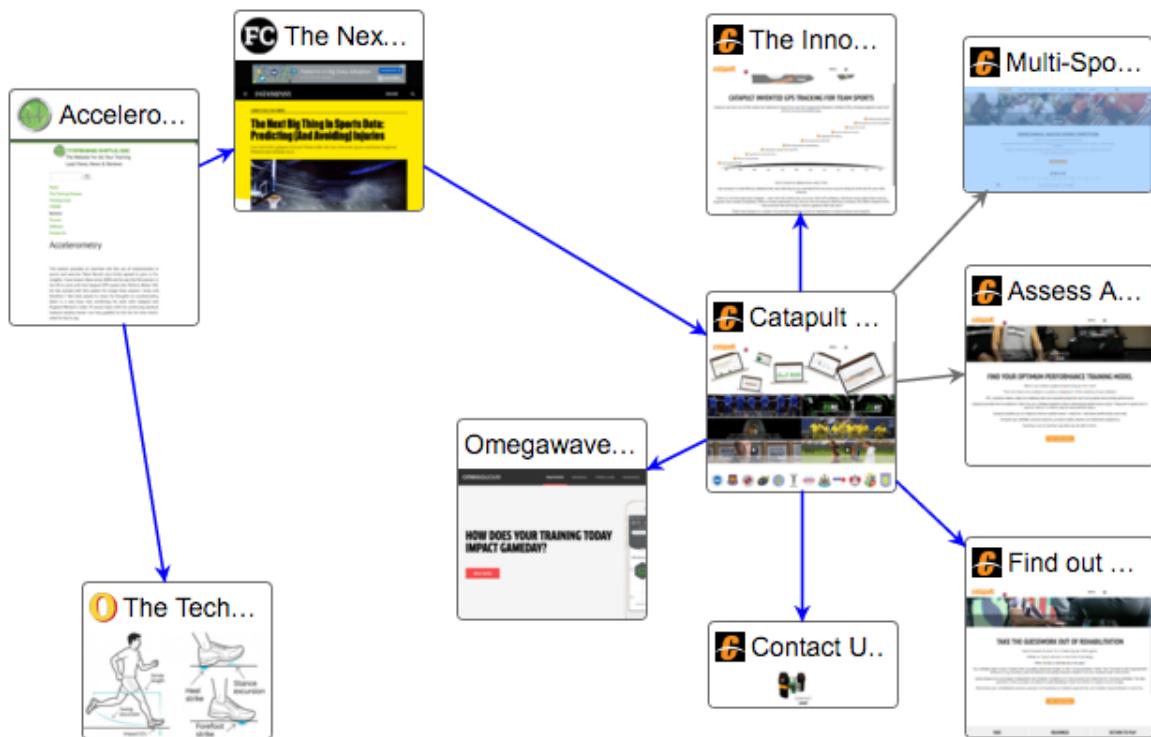


Figure 6.11: A knowledge map with three clear groups of nodes as the result of free movement.

6.3.5.3 Linking/Unlinking

Besides spatial grouping, seeing the causal relationships between collected information is also important to users in supporting sensemaking (Requirement 7). A conventional representation is used to show this relationship: an arrow pointing from the cause to the effect. The user can add a causal relationship by clicking on the “cause node”, holding it for half a second until the cursor changes to an arrow, then releasing the mouse on the “effect node”.

When nodes are added to the history map, the provenance links among them are also copied to the knowledge map to provide an initial understanding of existing relations. Different colors are used to distinguish user-added links from provenance links.

6.3.5.4 Formal Reasoning

Currently, SenseMap does not provide support for any formal argumentation methods. However, we think that the flexibility of spatial organization and relationships establishment can help the user apply their reasoning strategies [155]. For instance, users can draw a link from a “hypothesis” node to its evidence. Then, they can move all supporting evidence nodes to one area and all counter evidence nodes to a different location to distinguish the two groups.

6.3.5.5 Collection – Curation

All nodes in the knowledge map appear in the history map, but the other direction may not be true because only relevant and important nodes may be curated. To help the user quickly recognize which nodes in the history map are already curated, a green “tick” icon is superimposed at the top right hand corner. Also, hovering a node in one map will highlight that node, if it exists, in the other map.

6.3.6 Communication

The final organization of curated information provides a complete picture of solving the sensemaking task, which makes it ideal for the user to present their findings (Requirement 11). If the process is of interest, the history map can be used alongside the knowledge map. Moreover, the user can refer to raw data, via node revisit, to support their presentation (Requirement 12).

Both the history and knowledge maps can be saved as local files and loaded. This allows users to share their maps (Requirement 14). Also, the user can create multiple copies of knowledge maps based on the same history map allowing customizing for various presentation purposes (Requirement 13).

6.3.7 Implementation

SenseMap is implemented as a Chrome extension using D3 [15] for visualization. Chrome was chosen because of its high popularity. This was also confirmed in the initial interviews with seven out of nine participants were using Chrome. Highlighting and annotation require modification of web page structure, thus are implemented as content scripts using Chrome extension API. The detection of “search” actions is based on URL parsing. The web page URL is compared against a set of query templates to extract search type and parameters if it matches some template. Detecting “filtering” actions is based on a heuristic that if two URLs share the same domain and path name but have different query parameters, the second page should be the result of a filtering action on the first one. More details about provenance capture can be found in SensePath [125].

6.4 Evaluation

6.4.1 Method

We conducted a user-centered evaluation of SenseMap to explore its effectiveness in providing the desired support for sensemaking through the collect – curate – communicate process. Evaluating the usefulness of a system in supporting sensemaking is challenging because participants may employ various strategies, and their processes and outcomes are highly context-sensitive, making it difficult to quantify and compare their performance [101]. Therefore, sensemaking evaluations are typically case studies with realistic datasets and domain experts as participants. Similarly, we decided that to conduct a qualitative study in a naturalistic work setting.

We recruited five participants who are all working as junior designers and engineers in an innovation center. The participants were all introduced to the tool, trained in its operation and given thirty minutes to try out the tool with support, before being given the task. Each participant installed the tool on their own device; all participants were using laptops – three Apple Macs and two Windows – and one

participant had a second larger monitor connected. The participants conducted the task in their normal working environment (Figure 6.12) over a two-hour period.



Figure 6.12: The working environment of the participants, which was also used in the evaluation to ensure a naturalistic work setting.

The task was devised to reflect normal work activities for these participants in the early research phases of an innovation project. We focused the task on technology selection and deployment, requiring them to collect and curate information on a variety of interrelated areas and then to communicate their findings. Participants were given the task in written form, and it was discussed to clarify any points of confusion or ambiguity. They were asked to complete the following task while using SenseMap to record and present your findings: “We need to use accelerometers to measure *movement intensity* in ambulatory subjects, in naturalistic settings, for up to 1 week. We need to find out about (in no order of priority): prior art, placement of devices, algorithms, commercial products and APIs, bespoke approaches, and anything else you feel is relevant.”

At the end of the two-hour period we conducted an individual, semi-structured interview with each of the five participants. The participants were asked to present their findings, describe the process that they used to reach these findings using SenseMap, and to reflect upon their experience. We also collected provenance data to explore how participants interacted with SenseMap in their sensemaking activities over time including the timing (when), content (what) and position (where). All sensemaking features supported by SenseMap were captured such as highlight-

ing and annotation in the browser, relevance assessment in the history map, and node movement in the knowledge map. Other standard interaction in the browser including window focus, lost and mouse, keyboard events were also captured.

6.4.2 Data Analysis

6.4.2.1 Quantitative Features

The quantitative data showed two distinct engagement profiles; i.e., how the participants engaged in the sensemaking process and interacted with SenseMap. Figure 6.13 shows the histogram of curation activities for all participants.

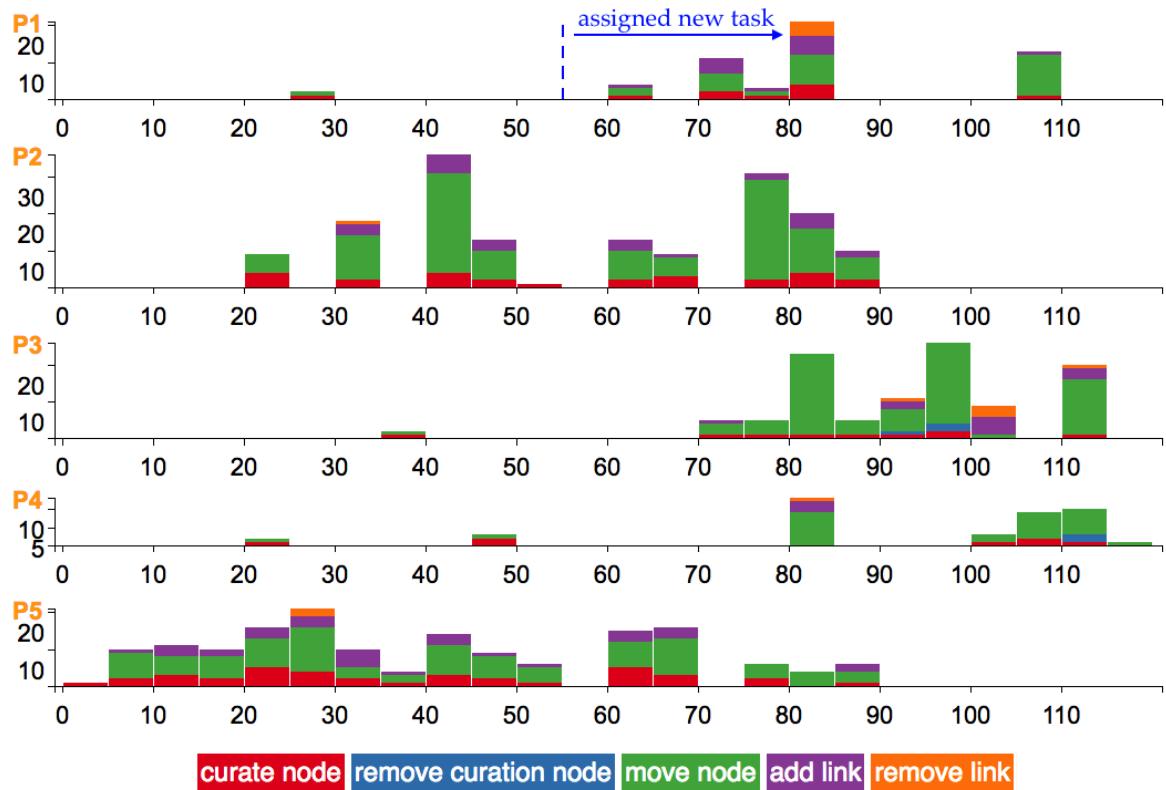


Figure 6.13: Histogram of curation activities for all participants. Each bin is 5 minutes.

High Engagement Table 6.1 shows the number of nodes and links of the knowledge map for all participants. P5 was the first to curate and had the most detailed knowledge map with 35 nodes and 35 links. P2 had a similar profile with early and regular interactions with their window contents. P2 had the second most detailed knowledge map with 26 nodes and 26 links (Figure 6.4C shows part of it). P1 began

the trial with uncertainty due to a lack of technical knowledge of the task. The task was contextualized for P1 helping them to relate it more closely to their expertise. P1 began productive engagement with SenseMap at a later stage resulting in a similar engagement profile to P2 but compressed into a shorter timeframe (starting after around 55 minutes). P1 had the third most detailed knowledge map with 10 nodes and 12 links even though he only spent his second hour for the task. These participants share the same pattern – *curate early, curate often* – and it relates to the interplay between collect and curate in our sensemaking model, through refining searches and extending the schema.

Table 6.1: Knowledge Map produced by participants.

Participant	Number of nodes	Number of links
P1	10	12
P2	26	26
P3	6	7
P4	5	2
P5	35	35

Low Engagement P3 did some minor curation activities early in the sensemaking process, but there was a considerable rise in the last third of the task time. There were only 6 nodes and 7 links in the final knowledge map with an indeterminate linking structure.

P4 did some minor curation activities with a short focus after an hour and more towards the end of the task. P4's interaction profile is notable for long and frequent periods of inactivity. P4 had only 5 nodes and 2 links in their final knowledge map with an indeterminate linking structure (Figure 6.14).

6.4.2.2 Qualitative Features

F1 – Communication Did the participants use the knowledge map to communicate their findings? Had they successfully curated their schema through clusters, linked branches or other coherent structures? Had they constructed a narrative to explain their schema?

Positive: P1, P2, P5. All had arranged their detailed knowledge maps as linked clusters from a key document. P1 and P2 were both able to provide a very coherent narrative about their findings. They confidently used the knowledge map to explain

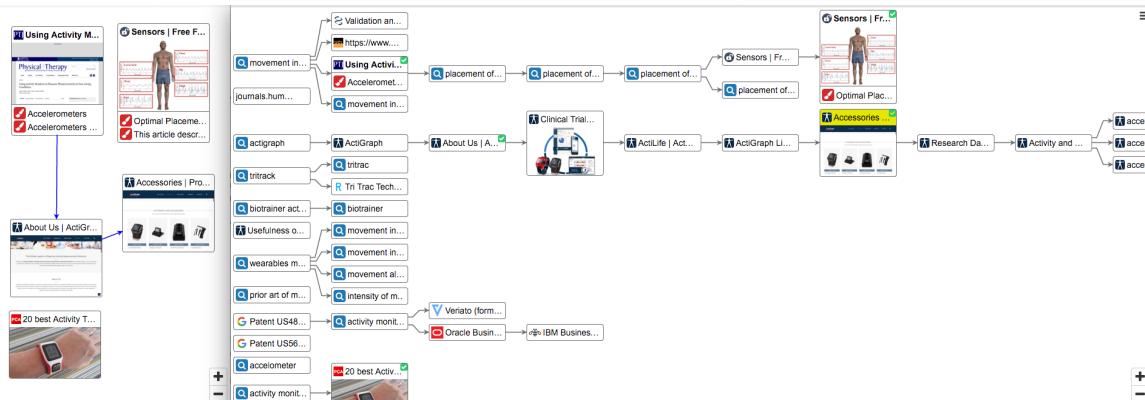


Figure 6.14: P4: Knowledge Map (left) and History Map (right). P4 successfully moved a key document into the curation space, but subsequent schema is scant.

their findings, using links and clusters to explain relationships and recommendations, and clicking on nodes to access the original information sources in the browser view. P2 felt confident that he had completed the task in the time allowed and felt that the tool had helped him to be thorough, systematic and organized. P1 had low confidence in his technical expertise when he began the trial. After the task was recontextualized for him he made confident progress. He was pleased with the visual representation and interaction of the knowledge map, referring to it as a “mind map” – a knowledge mapping process that he often uses. P5 was less able to provide a narrative of his findings even though he had the most detailed knowledge map. He felt that he had not completed the task and was unsure about some of the technical aspects of it, which may have had some bearing on this. He was very positive about the use of the tool.

Negative: P3, P4. Neither of them were able to use their knowledge maps to communicate their findings, referring instead to their history maps. Both saw much the potential in the tool to assist in sensemaking activities, but were less positive about their experience of it.

F2 – Window Display Were participants able to work with their desired browser window size and effectively display or switch between windows during the task?

Positive: P1, P2, P5. P5 had a second monitor connected and was able to work with a full-screen browser window on his laptop as his point of focus. The two maps were arranged on the second monitor, each taking half of the screen, and the monitor was behind his laptop (Figure 6.15). He enjoyed this relationship and referred to the external monitor as his back-up and like having a second-brain. P1 and P2 both

resized windows, but were adept at switching between them, and demonstrated fluidity in this during their interviews and in their engagement profiles. P2 had arranged all three windows to be nearly full-screen and arranged them as an overlapping stack, so he could see the edges of all windows at all times. He also used the three-finger swipe on his mac to minimize and show all windows and to switch between them.

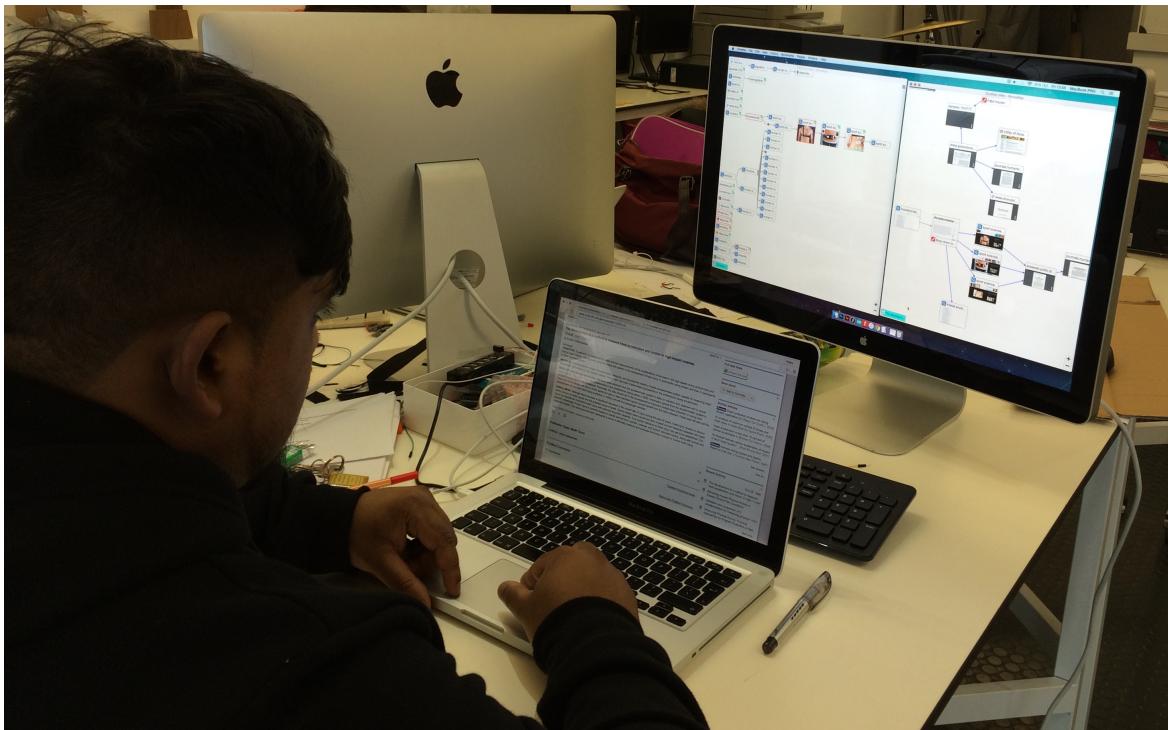


Figure 6.15: P5 with two monitors, comfortably interacting with all three views of SenseMap.

Negative: P3, P4. Both of them reverted to full-screen browser windows and expressed a strong preference for this. P3 ignored the history map after losing reassurance that the tool could support him in his task. P4 reset the browser view to full screen, ignored the history map, and then lost track of the meaning of the collection when she returned to it.

F3 – Keeping Track Did the participant understand and keep track of the building collection in the history map?

Positive: P1, P2, P3, P5. P5 had a second monitor attached so could see the history map at all times. He said that the tool really began to make sense for him when he connected the second monitor and he felt his pace increased. P1 said he had resized the browser to full screen and ignored the history map, but it continued to

make sense to him. He characterized his use as having regular periods of interaction with the history map rather than regular observation of it building up. P3 maintained track in the early phases of the task and did extensive preparation for curation by minimizing nodes to help him keep track and to guide his searches.

Negative: P4. Early in the task P4 decided to reset the browser window to full-screen and to ignore the history map. When she returned to the history map she found she had lost track and did not satisfactorily regain it throughout the rest of the task.

F4 – Trust Did the participant maintain trust in the tool throughout the process? Were they confident that nodes were appearing in the correct relationship and with the correct links to other nodes? Did they understand these relationships when looking at the history map?

Positive: P1, P2, P5. They all expressed feelings of trust in the tool and were able to confidently shift their sensemaking activity focus from collections of tabs to the history and knowledge maps.

Negative: P3, P4. P3 had been managing the history map, but lost trust in the tool as they began to question the position and relationship of nodes generated in the history map to other nodes in that map. P4 lost trust in the tool as they could not understand the relationship of the nodes in the history map.

F5 – Reassurance Was the participant reassured that the tool would provide sufficient support to complete the task? Did the participant believe that the added value in organization of information sources would outweigh the effort?

Positive: P1, P2, P5. P2 referred to the history map and the knowledge map as a good aide memoir allowing him to check completeness and to guide further searches. He felt reassured enough by the growing history map to close browser tabs. In his normal practice, he often created reports based on his tab sets to communicate his findings. He was confident and enthusiastic that SenseMap could remove this burden. P1 saw much potential in the tool and asked if he could use it for a big design project that he would be competing in the following year. He had regular interaction with the history map and used it the most to revisit previous sources of information clicking on nodes in the history map (44% of web browser page reloads). P5 described the history map as “my thinking” and the knowledge map as “a neater view of my thinking”. His reference to having a second brain was clearly reassuring to him.

Negative: P3, P4. P3 was initially reassured by the tool. However, this reassurance diminished over time as he began to feel that the management of the collection was impeding his ability to complete the task and gradually lost interest in the collection view, reverting to his typical sensemaking methods using tabs and a full screen browser window. P4 had lost track and no longer felt reassured that the tool could support her activity.

Table 6.2 summarizes the status of each feature for all participants.

Table 6.2: Qualitative features derived from all participants.

	Communication	Window Display	Keeping Track	Trust	Reassurance
P1	✓	✓	✓	✓	✓
P2	✓	✓	✓	✓	✓
P3			✓		
P4					
P5	✓	✓	✓	✓	✓

6.4.3 Discussion

We roughly assess the quality of the sensemaking outcome for each participant. The metric is based on the number of relevant sources that the participants found and the coherence in the organization of these sources. Both factors are assessed by a senior interaction designer who is the head of the project that the participants are involving. The coherent organization is reflected through both the knowledge map and the explanation of such structure if it was hidden in the participant's mind. Table 6.3 summarizes the assessment result.

Table 6.3: Quality of sensemaking outcome of participants.

Participant	Number of relevant sources	Coherence of schema
P1	6	good
P2	13	very good
P3	7	poor
P4	5	poor
P5	9	satisfactory

The most notable pattern we discovered in both quantitative and qualitative features is a clear division of participants into two groups. One group (P1, P2 and P5) highly engaged with the curation process and were positive in all five qualitative

features. Whereas, the other group (P3 and P4) engaged weakly with curation and were negative in almost all five qualitative features. This division was also true in the quality of the sensemaking outcome: P1, P2 and P5 found more relevant sources than P3 and P4 (note that P1 only spent half of the time that other participants) and structured them in a more coherent schema. This pattern may suggest an almost linear process relating all these features. Users who were able to manage tool windows were also able to keep track of the development of the history map, allowing them to trust that the tool would work properly and reassure that the tool would provide sufficient support to complete the task. Eventually, they were able to communicate their findings effectively and had more successful outcomes.

Next, we will discuss two lessons we learned in this evaluation.

Engagement High engagement with the browser view, the history map and the knowledge map could lead to a positive outcome. All three participants who had positive outcomes achieved this engagement either through having multiple monitors that display all three windows simultaneously (P5) or having the skill and willingness to regularly switch between them (P1 and P2). The challenge is how to design a more space-efficient history map to be displayed side by side with the browser view. Alternatively, the history map could be invisible while users are focusing on the browser view, but provides sufficient feedback to help them keep track of the map construction. Also, a visual summary of what has happened since the last time the history map is active could help users to catch up more quickly.

Another factor could impede user understanding of the history map is the complexity of the map itself. As discussed in Section 6.3.4.2, the history map uses a compact tree layout to produce a tidy visualization. However, as the exploration progresses, the visualization expands and may not fit into the display area, requiring users to manually zoom and pan. To address this issue, we need to ensure the active part of the map be always visible to users by automatically panning the visualization. Another approach is to automatically summarize or condense the inactive part of the map, which could be measured by the spatial or temporal distance to the most active ones. All these actions need to be performed with smooth transition to maintain user awareness.

Trust and Reassurance It is essential to maintain the trust and the reassurance of users with the tool, enabling them to continue curating their collected information and gaining benefit from the curation process. Our initial interviews identified user

anxiety over retaining and organizing tabs. Losing collections of tabs is seen as a serious event. SenseMap requires users to trust that it is recording their browsing activities accurately and in a manner that they can continue to understand throughout the sensemaking process. In essence, users pass control over the collection phase of their sensemaking process to the tool (trust) and curate this collection in ways that aim to provide enhanced ways to understand and present this knowledge (reassurance).

SenseMap is designed to support and augment browser-based online sensemaking, thus requires a change in practice from sensemaking through a collection of browser tabs to sensemaking by engaging with the history and knowledge maps. Our data shows that all participants who had a positive engagement profile were able to make the necessary practice change, were reassured by the tool's ability to support their work and maintained trust in it, and eventually produced successful outcomes. In the negative cases, the two participants were either unable or unwilling to change their practice. This insight suggests that spending time in curation is likely worth the effort; however, users may only curate if they trust and reassure that the tool will help them. The challenge is how to improve trust and reassurance through both the construction process and presentation of our history map. A think-aloud study of user's responses to history map construction would be an obvious next step that could stimulate alternative design proposals.

Opportunities for Further Improvement The evaluation shows that SenseMap provides useful sensemaking support for users in a 2-hour-long session. However, in the real world, a sensemaking task can be split into small chunks and spanned multiple days or even weeks. Because SenseMap is implemented as a Chrome extension, this gives us an opportunity to conduct a longer term and larger scale study to gain a better understanding of SenseMap's use.

Finally, all participants mentioned that they would like to be able to add notes to the knowledge map; one had even invented a way to do this himself using search terms. They would like to be able to label clusters and links to those clusters and also provide explanations about hypotheses and recommendations. This would also help users record their internal knowledge prior to the tasks.

6.5 Summary

In this chapter, we present SenseMap to enable users to explore and express complex rational relationship in browser-based online sensemaking through analytic provenance. SenseMap automatically captures users' sensemaking actions in the browser view and visualizes them in the history map to provide an overview of their sensemaking processes, preventing users from getting lost in the tasks. This enables users to curate the most relevant information into the knowledge map, improving their understanding of the tasks and potentially guide further exploration. At the end, users can communicate their findings using all three views with different levels of detail, including the summary in the knowledge map, the process in the history map, and the raw data in the browser view.

Our evaluation shows that all participants found the visual representation and interaction of the tool intuitive to use. Three of them engaged positively with the tool and produced successful outcomes. It helped them organize information sources, quickly find and navigate to the sources they wanted, and effectively communicate their findings. However, two participants had a negative experience with the tool and were unable to change their practice from sensemaking through collections of browser tabs.

SenseMap shows much potential to provide a powerful approach to online browser-based sensemaking for a wide spectrum of users. In order to meet this potential, it would be useful to improve the following two key areas.

1. More space-efficient visual representations and layouts, and smarter interaction and feedback sets between the browser and two maps allowing users to work on their browsing activities more comfortably.
2. Deeper understanding of how to maximize trust and reassurance of users with the tool, providing design guidelines for developing history and knowledge maps.

7

Conclusion

This chapter reviews the contributions that the thesis makes to support sensemaking through interactive visualizations of provenance data. It also presents the limitations of these contributions and discusses opportunities for future work.

7.1 Review of Contributions

The central problem addressed by this thesis is how to design interactive visualizations of provenance data for supporting sensemaking. The problem was then broken into four research questions based on the type of data that would be visualized and the task it aimed to support. We took a user-centered approach to address these research questions, covering requirements gathering, design, prototype implementation and user evaluation. This section reviews the contributions for each research question.

7.1.1 Exploring Temporal Relationship of Sensemaking

Research Question 1: How to design interactive visualizations of time-oriented provenance data enabling users to explore temporal relationship of sensemaking?

We contributed a novel timeline visualization, SchemaLine, that enables users to explore temporal relationship of sensemaking through the annotations they made during the sensemaking process. SchemaLine helps users examine the events recorded in those annotations in chronological order, identify interesting temporal patterns and construct narratives that account for those patterns. The SchemaLine visualization produces a compact but aesthetically pleasing layout. It also provides

a set of fluid interactions supporting users in performing various sensemaking activities described in the Data–Frame model [95], such as connecting relevant data into an explanatory frame.

Our user-centered evaluation showed that all participants found the SchemaLine visualization intuitive to use and it provided necessary support to them for solving their tasks. They extensively recorded their thoughts by note taking and constructed schemas to organize the notes. The participants were also confident in presenting the stories they found and defending them through the use of SchemaLine.

7.1.2 Exploring Complex Temporal Relationship of Sensemaking

Research Question 2: How to design interactive visualizations that can utilize both temporal and thematic provenance data to reveal complex temporal relationship of sensemaking?

We contributed a novel timeline visualization, TimeSets, that enables users to explore complex temporal relationship of sensemaking by effectively representing both temporal and thematic provenance data. TimeSets visually groups events that shares the same theme but still preserves their temporal order. It color codes the backgrounds of the entire themes to distinguish them and uses colored gradient backgrounds for the intersections among those themes. To handle a large number of events, TimeSets dynamically adjusts the level of detail for each event within a given display estate.

We conducted a controlled experiment to compare the effectiveness of TimeSets in performing temporal and thematic related tasks against a state-of-the-art method – KelpFusion [114]. The results showed that TimeSets was significantly more accurate and was the preferred choice for aesthetics and readability. TimeSets is general and can be applied to visualize any temporal, categorical data besides analytic provenance. We showed two such successful applications. One is the analysis of 200 articles from the IEEE Information Visualization conference. The other one is the visualization of streaming data, and the record and presentation of analysis findings.

7.1.3 Exploring Rational Relationship of Sensemaking

Research Question 3: How to design interactive visualizations that can exploit time-oriented provenance data to enable users to explore rational relationship of sensemaking?

We contributed a novel visualization tool, SensePath, that enables analysts to explore the rational relationship in the user's sensemaking process. To understand such relationship, analysts often carry out a time-consuming qualitative study and analysis including data collection, transcription, coding and theory abstraction. SensePath offers an alternative and possibly faster approach in performing transcription and coding. It detects and captures user's sensemaking actions automatically, thus make it possible to generate an automatic transcript. SensePath also provides multi-linked visualizations of the captured actions, allowing the analysts to gain deep understanding of the rational relationship between these actions, thus facilitate the coding process. More specifically, a timeline view allows analysts to quickly gain an overview of the sensemaking process and identify recurring patterns. It also links with a screen capture video to support a close examination of additional context when necessary. Finally, to enable analysts to continue working on later stages of analysis using their normal workflow, SensePath exports its coded transcript in a common format that can be used by other popular qualitative data analysis software packages such as InqScribe.

We conducted two user-centered evaluations. The first one was to explore how SensePath would be used by a domain expert. A researcher with seven years of experience in qualitative research was recruited to analyze a sensemaking session performed by different user. She found many features of SensePath helpful for her analysis. For example, by looking at the timeline, she quickly gained an initial understanding of the user process, which can only be achieved after watching the entire video if using traditional method. It also helped her to identify recurring patterns, which she used as cues for further examination. She also mentioned that linking between actions in the timeline view and the video capture could save her an enormous amount of time in finding the right moment to watch. The second evaluation was to discover whether SensePath has any potential advantages compared with a traditional method. Two senior HCI researchers were recruited to analyze a shorter sensemaking session. One used SensePath, and one used his standard tool set: a transcribing software and Excel for coding. The analysis results in the two settings had comparable quality, but the researcher with SensePath spent much less time, especially for the transcribing part. Even though this evaluation was conducted with only two participants, it may suggest an improvement in analysis time for SensePath.

7.1.4 Exploring Complex Rational Relationship of Sensemaking

Research Question 4: How to design interactive visualizations that can utilize both temporal and relational provenance data to enable users to explore complex rational relationship of sensemaking?

We contributed a novel visual sensemaking tool, SenseMap, that enables users to explore complex rational relationship of sensemaking. It automatically captures sensemaking actions and linking relationships between these actions before visualizing both of them in a branching history tree. The temporal attribute of actions is also partially reflected in the visualization. This history tree allows users to examine the rational relationship between the actions they performed and potentially helps them remind of what have been done earlier. SenseMap offers users to assign additional meaning to the automatically collected data by spatially grouping actions or adding rational links between them, in order to help explain complex relationship. Finally, SenseMap allows users to communicate their analysis results at different levels of granularity including a big picture of user-organized findings, a more detailed analysis process and raw provenance data captured.

We conducted a user-centered evaluation in a naturalistic setting to explore how SenseMap would be used. The results showed that all participants found the visual representation and interaction of the tool intuitive to use. Three of them engaged positively with the tool and produced successful outcomes. It helped them organize information sources, quickly find and navigate to the sources they wanted, and effectively communicate their findings. However, two participants had a negative experience with the tool and were unable to change their practice from sensemaking through collections of browser tabs.

7.2 Limitations and Future Research

7.2.1 TimeSets

TimeSets (Chapter 4) enables users to explore complex temporal relationship of sensemaking by visualizing both temporal and thematic provenance data. The evaluation showed advantage of TimeSets against a state-of-the-art method; however, it stills suffers some limitations that can be served as future directions.

- *Set intersections.* The major limitation of TimeSets is that it only shows the intersections between vertically neighboring sets, which is a small portion

of all possible combinations of intersections. The set ordering algorithm to maximize the number of shared events and interaction to reorder sets partially helped address this issue. Future research can focus on increasing the number of visible intersections, prioritizing more important intersections based on some metrics, and providing an overview of all intersections to suggest further exploration.

- *Set area encoding.* The area of the representing set seems to be a very strong indicator of its size; i.e., number of events. However, it is not designed like that in TimeSets. Besides the number of events, the set area also depends on the temporal distribution of its events. To address this issue, we could deemphasize the visual appearance of the set area or design a new shape outline algorithm to ensure the set area truly reflect the its size.
- *Multiple-set events.* In Chapter 4, we proposed four different visual representations of multiple-set events (Section 4.3.2.3). However, it is not clear which representation is the most effective and for which tasks. Therefore, a formal evaluation is necessary to examine this aspect.
- *Aggregated events.* In the current TimeSets, there is no extra visual cue to help viewers quickly see the difference between the sizes of the aggregated events: they have to read the label. We proposed four different visual representations to display aggregations (Section 4.4.5). However, similar to the previous point, a formal evaluation needs to be conducted to further investigate this design.

7.2.2 SensePath

SensePath (Chapter 5) enables analysts to explore the rational relationship in the user's sensemaking process. The evaluations suggested that SensePath provided useful features for qualitative analysis and could potentially reduce analysis time. Limitations and corresponding future research are discussed as follows.

- *More coding support.* SensePath supports exploration of rational relationship of captured actions, thus facilitates coding. However, the feature set for coding is limited. SensePath currently supports code assignment and editing. The analyst can either enter a new code or select from used codes. The following coding features can be added to make the tool more practical.
 - Assign a code to a part of an action or a range of actions.

- Assign multiple codes to an action.
 - Support hierarchical codes.
 - Show the codes in the timeline view more effectively.
- *Action capture.* To capture *search actions* in new web services that are unavailable from the current implementation, the search templates for these services need to be added to the code. To make this more accessible for non-technical users (such as the analysts), a user interface that allows specifying the search templates could be useful. More general and challenging is to provide a capture management interface that allows the analysts to choose which actions to capture and to define new actions based on some extraction rules.
 - *Application for other domains.* The approach used in SensePath is general and it can be extended to support exploration of rational relationship in other domains beyond online sensemaking tasks such as usability testing. SensePath consists of two components: capture and visualization. The visualization set can be reused straightforwardly. The capture component is specially designed for capturing online sensemaking actions, thus needs to be changed according to the new domain and system.

7.2.3 SenseMap

SenseMap (Chapter 6) enables users to explore complex rational relationship of sensemaking. This section discuss the limitations and corresponding future research of the tool.

- *Engagement increase.* Our evaluation showed that three participants who engaged with the tool produced positive sensemaking outcome and communicated the findings effectively. However, the other two participants did not engage with the tool. One of the reasons was their computer screen sizes were not large enough to show all three views at the same time, and they were unable to switch between the views comfortably to keep track of the history map construction. One possible improvement is to design a more space-efficient history map so that it can display side by side with the browser view. Another option is to allow users to focus on the browser view, but provide sufficient feedback to help them understand the map development. Also, a visual summary of changes in the map could also help users catch up more quickly.

- *Trust and reassurance increase.* SenseMap requires users to trust that it is recording their browsing activities accurately and in a manner that they can continue to understand throughout the sensemaking process. In essence, users pass control over the collection phase of their sensemaking process to the tool (trust) and curate this collection in ways that aim to provide enhanced ways to understand and present this knowledge (reassurance). However, two participants in our study did not have the trust and reassurance with the tool. To address this issue, first more engineering effort is required to ensure the tool to capture all sensemaking actions and detect the linking relationship between them correctly. Second, more research needs to be conducted to establish understanding of when and how users lost their trust and reassurance. As a result, it could help provide design guidelines for developing history and knowledge maps.
- *Large scale user study.* The evaluation showed that SenseMap provided useful sensemaking support for users in a 2-hour-long session. However, in the real world, a sensemaking task can be split into small chunks and spanned multiple days or even weeks. A larger scale and longer term study is necessary to gain better understanding of SenseMap's use. Because SenseMap is implemented as a Chrome extension, this gives us a good opportunity to conduct such a study.

7.3 Closing Remarks

The data around the world has been produced more rapidly than ever before. It could bring us a good opportunity to collect more relevant data for solving a task, but also challenge us in managing and making sense of a large number of findings. Analytic provenance captures the actions and accompanied reasoning in the sensemaking process. Moreover, visualizations of provenance data help the user recall the process, understand the temporal and rational relationship of the process more deeply, and potentially suggest the next step in sensemaking. A deep understanding about the past could lighten the future. This thesis contributes novel visualizations to support sensemaking and hopes to help people make sense of their tasks more effectively and open more research in this direction.

References

- [1] SIMILE Timeline Widget. \url{http://simile-widgets.org/timeline/}, 2009.
- [2] Gapminder, 2010.
- [3] Ambulances for Dialysis Patients on Rise, 2014.
- [4] TimeGlider. \url{http://timeglider.com/}, 2016.
- [5] A. Adams, P. Lunt, and P. Cairns. A qualitative approach to HCI research. In P. Cairns and A. Cox, editors, *Research Methods for Human-Computer Interaction*, pages 138–157. Cambridge University Press, 2008.
- [6] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Data*. Springer, 2011.
- [7] B. Alper, N. Henry Riche, G. Ramos, and M. Czerwinski. Design study of LineSets, a novel set visualization technique. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2259–2267, dec 2011.
- [8] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers. Visualizing Sets and Set-typed Data: State-of-the-Art and Future Challenges. In R. Borgo, R. Maciejewski, and I. Viola, editors, *Eurographics conference on Visualization (EuroVis)– State of The Art Reports*, pages 1–21. Eurographics, Eurographics, 2014.
- [9] R. Amar, J. Eagan, and J. Stasko. Low-level components of analytic activity in information visualization. In *IEEE Symposium on Information Visualization*, pages 111–117. IEEE, 2005.

- [10] P. André, M. L. Wilson, A. Russell, D. A. Smith, A. Owens, and M. Schraefel. Continuum: designing timelines for hierarchies, relationships and scale. In *ACM Symposium on User Interface Software and Technology*, pages 101–110, New York, New York, USA, oct 2007. ACM Press.
- [11] C. Andrews, A. Endert, and C. North. Space to Think: Large, High-Resolution Displays for Sensemaking. In *ACM Conference on Human Factors in Computing Systems*, pages 55–64, 2010.
- [12] E. Z. Ayers and J. T. Stasko. Using Graphic History in Browsing the World Wide Web. Technical report, 1995.
- [13] L. Bavoil, S. Callahan, P. Crossno, J. Freire, C. Scheidegger, C. Silva, and H. Vo. VisTrails: Enabling Interactive Multiple-View Visualizations. In *IEEE Conference on Visualization*, pages 135–142. IEEE, 2005.
- [14] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, 3(Jan):993–1022, 2003.
- [15] M. Bostock, V. Ogievetsky, and J. Heer. D3: Data-Driven Documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011.
- [16] S. Bowers, T. McPhillips, B. Ludäscher, S. Cohen, and S. B. Davidson. A Model for User-Oriented Data Provenance in Pipelined Scientific Workflows. In *Provenance and Annotation of Data*, pages 133–147. Springer Berlin Heidelberg, 2006.
- [17] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2376–85, dec 2013.
- [18] E. T. Brown, A. Ottley, H. Zhao, Q. Lin, R. Souvenir, A. Endert, and R. Chang. Finding Waldo: Learning about Users from their Interactions. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1663–1672, dec 2014.
- [19] P. Buneman, J. Cheney, W.-C. Tan, and S. Vansummeren. Curated databases. In *ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–12, New York, New York, USA, 2008. ACM Press.
- [20] P. Buneman, S. Khanna, and T. Wang-Chiew. Why and Where: A Characterization of Data Provenance. In *Database Theory — ICDT 2001*, pages 316–330. Springer Berlin Heidelberg, 2001.

- [21] M. Burch, F. Beck, and S. Diehl. Timeline trees: visualizing sequences of transactions in information hierarchies. In *International Working Conference on Advanced Visual Interfaces*, pages 75–82, New York, New York, USA, may 2008. ACM Press.
- [22] B. Burr. VACA: a tool for qualitative video analysis. In *Extended Abstracts on Human Factors in Computing Systems*, pages 622–627, New York, New York, USA, 2006. ACM Press.
- [23] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Generation Computer Systems*, 25(6):599–616, jun 2009.
- [24] L. Byron and M. Wattenberg. Stacked graphs-geometry & aesthetics. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1245–52, 2008.
- [25] J. V. Carlis and J. A. Konstan. Interactive Visualization of Serial Periodic Data. In *ACM Symposium on User Interface Software and Technology*, pages 29–38, 1998.
- [26] S. Carpendale. Evaluating Information Visualizations. *Information Visualization*, pages 19–45, 2008.
- [27] J. Cheney, L. Chiticariu, and W.-C. Tan. Provenance in Databases: Why, How, and Where. *Foundations and Trends in Databases*, 1(4):379–474, apr 2007.
- [28] N. Chinchor and W. a. Pike. The science of analytic reporting. *Information Visualization*, 8(4):286–293, 2009.
- [29] B. chul Kwon, W. Javed, N. Elmquist, and J. S. Yi. Direct manipulation through surrogate objects. In *ACM Conference on Human Factors in Computing Systems*, pages 627–636, New York, New York, USA, 2011. ACM Press.
- [30] W. Cleveland and R. McGill. Graphical Perception and Graphical Methods for Analyzing Scientific Data. *American Association for the Advancement of Science*, 229(4716):828–833, 1985.
- [31] A. Cockburn, S. Greenberg, B. McKenzie, M. Jasonsmith, and S. Kaasten. WebView: A Graphical Aid for Revisiting Web Pages. In *Australian Conference on Human Computer Interaction*, pages 15–22, 1999.

-
- [32] A. Cockburn, A. Karlson, and B. B. Bederson. A Review of Overview+Detail, Zooming, and Focus+Context Interfaces. *ACM Computing Surveys*, 41(1):1–31, 2008.
 - [33] C. Collins, G. Penn, and S. Carpendale. Bubble sets: revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1009–1016, 2009.
 - [34] J. Conklin. Hypertext: An Introduction and Survey. *Computer*, 20(9):17–41, sep 1987.
 - [35] J. Corbin and A. Strauss. Grounded theory methodology. *Handbook of qualitative research*, 17:273–285, 1994.
 - [36] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.
 - [37] P. Cowley, J. Haack, R. Littlefield, and E. Hampson. Glass box: capturing, archiving, and retrieving workstation activities. In *ACM Workshop on Continuous Archival and Retrieval of Personal Experience*, pages 13–18, New York, New York, USA, oct 2006. ACM Press.
 - [38] Y. Cui, J. Widom, and J. L. Wiener. Tracing the lineage of view data in a warehousing environment. *ACM Transactions on Database Systems*, 25(2):179–227, jun 2000.
 - [39] S. Davidson, S. Cohen-Boulakia, A. Eyal, B. Ludascher, T. Mcphillips, S. Bowers, M. K. Anand, and J. Freire. Provenance in Scientific Workflow Systems. *Data Engineering Bulletin*, 30(4):1–7, 2007.
 - [40] M. Derthick and S. F. Roth. Enhancing data exploration with a branching history of user operations. *Knowledge-Based Systems*, 14(1-2):65–74, 2001.
 - [41] B. Dervin. An overview of sense-making research: Concepts, methods, and results to date, 1983.
 - [42] B. Dervin and L. Foreman-Wernet. Sense-Making Methodology as an approach to understanding and designing for campaign audiences. In *Public Communication Campaigns*, pages 147–162. Sage, 2012.

- [43] A. Dix and G. Ellis. Starting simple: adding value to static visualisation through simple interaction. In *International Working Conference on Advanced Visual Interfaces*, pages 124–134, New York, New York, USA, 1998. ACM Press.
- [44] M. Dontcheva, S. M. Drucker, G. Wade, D. Salesin, and M. F. Cohen. Summarizing personal web browsing sessions. In *ACM Symposium on User Interface Software and Technology*, pages 115–124, New York, New York, USA, oct 2006. ACM Press.
- [45] W. Dou, D. H. Jeong, F. Stukes, W. Ribarsky, H. R. Lipford, and R. Chang. Recovering Reasoning Processes from User Interactions. *IEEE Computer Graphics and Applications*, 29(3):52–61, may 2009.
- [46] C. Dunne, N. Henry Riche, B. Lee, R. A. Metoyer, and G. G. Robertson. Graph-Trail: Analyzing Large Multivariate and Heterogeneous Networks while Supporting Exploration History. In *ACM Conference on Human Factors in Computing Systems*, pages 1663–1672, 2012.
- [47] P. Eades. A heuristics for graph drawing. *Congressus numerantium*, 42:146–160, 1984.
- [48] N. Elmquist, A. V. Moere, H.-C. Jetter, D. Cernea, H. Reiterer, and T. Jankun-Kelly. Fluid interaction for information visualization. *Information Visualization*, 10(4):327–340, aug 2011.
- [49] A. P. Field and G. Hole. *How to design and report experiments*. Sage Publications, 2003.
- [50] J. D. Foley, A. V. Dam, S. K. Feiner, J. F. Hughes, and R. L. Phillips. *Introduction to Computer Graphics*. Addison-Wesley, 1997.
- [51] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, nov 1991.
- [52] G. W. Furnas. Generalized fisheye views. In *ACM Conference on Human Factors in Computing Systems*, pages 16–23, 1986.
- [53] G. W. Furnas. A fisheye follow-up: further reflections on focus + context. In *ACM Conference on Human Factors in Computing Systems*, volume 1, pages 999–1008, 2006.

- [54] H. L. Gantt. *Work, wages, and profits*. Engineering Magazine Co., 1913.
- [55] N. Gershon and W. Page. What storytelling can do for information visualization. *Communications of the ACM*, 44(8):31–37, aug 2001.
- [56] M. Ghoniem. On the readability of graphs using node-link and matrix-based representations: a controlled experiment and statistical analysis. *Information Visualization*, 4(2):114–135, 2005.
- [57] Y. Gil, E. Deelman, M. Ellisman, T. Fahringer, G. Fox, D. Gannon, C. Goble, M. Livny, L. Moreau, and J. Myers. Examining the Challenges of Scientific Workflows. *Computer*, 40(12):24–32, dec 2007.
- [58] J. Goodwin. Wigmore’s chart method. *Informal Logic*, 20(3), 2000.
- [59] C. Gorg, Z. Liu, and J. Stasko. Reflections on the evolution of the Jigsaw visual analytics system. *Information Visualization*, jul 2013.
- [60] D. Gotz, M. Zhou, and V. Aggarwal. Interactive Visual Synthesis of Analytic Knowledge. In *IEEE Symposium on Visual Analytics Science And Technology*, pages 51–58. IEEE, oct 2006.
- [61] D. Gotz and M. X. Zhou. Characterizing users’ visual analytic activity for insight provenance. *Information Visualization*, 8(1):42–55, jan 2009.
- [62] T. J. Green, G. Karvounarakis, and V. Tannen. Provenance semirings. In *ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 31–40, New York, New York, USA, 2007. ACM Press.
- [63] P. Groth and L. Moreau. Representing distributed systems using the Open Provenance Model. *Future Generation Computer Systems*, 27(6):757–765, 2011.
- [64] G. Guest, K. M. MacQueen, and E. E. Namey. *Applied thematic analysis*. Sage, 2011.
- [65] H. Guo, S. R. Gomez, C. Ziemkiewicz, and D. H. Laidlaw. A Case Study Using Visualization Interaction Logs and Insight Metrics to Understand How Analysts Arrive at Insights. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):51–60, jan 2016.
- [66] J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

- [67] M. Harrower and C. a. Brewer. ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps. *The Cartographic Journal*, 40(1):27–37, jun 2003.
- [68] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. ThemeRiver: visualizing thematic changes in large document collections. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):9–20, 2002.
- [69] J. Heer. Flare: Data visualization for the web, 2009.
- [70] J. Heer, M. Agrawala, and W. Willett. Generalized selection via interactive query relaxation. In *Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08*, page 959, New York, New York, USA, 2008. ACM Press.
- [71] J. Heer and M. Bostock. Crowdsourcing graphical perception. In *ACM Conference on Human Factors in Computing Systems*, pages 203–212, New York, New York, USA, 2010. ACM Press.
- [72] J. Heer, N. Kong, and M. Agrawala. Sizing the horizon: the effects of chart size and layering on the graphical perception of time series visualizations. In *ACM Conference on Human Factors in Computing Systems*, pages 1303–1312, New York, New York, USA, apr 2009. ACM Press.
- [73] J. Heer, J. Mackinlay, C. Stolte, and M. Agrawala. Graphical histories for visualization: supporting analysis, communication, and evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1189–1196, 2008.
- [74] J. Heer and B. Shneiderman. Interactive dynamics for visual analysis. *Communications of the ACM*, 10(2):26, apr 2012.
- [75] J. Heer, F. B. Viégas, and M. Wattenberg. Voyagers and Voyeurs: Supporting Asynchronous Collaborative Visualization. *Communications of the ACM*, 52(1):87–97, jan 2009.
- [76] N. Henry and J.-D. Fekete. MatLink: Enhanced Matrix Visualization for Analyzing Social Networks. In *Human-Computer Interaction – INTERACT*, pages 288–302. 2007.
- [77] N. Henry Riche and T. Dwyer. Untangling euler diagrams. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1090–1099, jan 2010.

- [78] N. Henry Riche, B. Lee, and C. Plaisant. Understanding Interactive Legends: a Comparative Evaluation with Standard Widgets. *Computer Graphics Forum*, 29(3):1193–1202, aug 2010.
- [79] R. J. Heuer. Psychology of intelligence analysis. Technical report, 1999.
- [80] R. R. Hightower, L. T. Ring, J. I. Halfman, B. B. Bederson, and J. D. Hollan. Graphical Multiscale Web Histories: A Study of PadPrints. In *ACM Symposium on User Interface Software and Technology*, pages 121–122, New York, New York, USA, nov 1998. ACM Press.
- [81] O. Hoeber and J. Gorner. BrowseLine: 2D Timeline Visualization of Web Browsing Histories. In *International Conference on Information Visualisation*, pages 156–161. IEEE, jul 2009.
- [82] K. Holtzblatt and S. Jones. Contextual Inquiry: A Participatory Technique for Systems Design. In *Participatory design: Principles and practices*, pages 177–210. 1993.
- [83] L. Hong, E. H. Chi, R. Budiu, P. Pirolli, and L. Nelson. SparTag.us: a low cost tagging system for foraging of web content. In *International Working Conference on Advanced Visual Interfaces*, pages 65–72, New York, New York, USA, may 2008. ACM Press.
- [84] T. J. Jankun-Kelly, K. L. Ma, and M. Gertz. A model and framework for visualization exploration. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):357–368, 2007.
- [85] K. K. *Principles of Gestalt psychology*. Oxford, England: Harcourt, Brace, 1935.
- [86] S. Kaasten, S. Greenberg, and C. Edwards. How People Recognize Previously Seen Web Pages from Titles, URLs and Thumbnails. In *People and Computers XVI - Memorable Yet Invisible*, pages 247–265. Springer London, 2001.
- [87] N. Kadivar, V. Chen, D. Dunsmuir, E. Lee, C. Qian, J. Dill, C. Shaw, and R. Woodbury. Capturing and supporting the analysis process. In *IEEE Symposium on Visual Analytics Science And Technology*, pages 131–138. IEEE, 2009.
- [88] Y.-A. Kang, C. Görg, and J. Stasko. How Can Visual Analytics Assist Investigative Analysis? Design Implications from an Evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 17(5):570–583, jun 2011.

- [89] C. Kehoe, J. Stasko, and A. Taylor. Rethinking the Evaluation of Algorithm Animations as Learning Aids: An Observational Study. *International Journal of Human-Computer Studies*, 54(2):265–284, 2001.
- [90] D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann. *Mastering the Information Age - Solving Problems with Visual Analytics*. 2010.
- [91] D. Keim, J. Schneidewind, and M. Sips. CircleView: a new approach for visualizing time-related multidimensional data sets. In *International Working Conference on Advanced Visual Interfaces*, pages 179–182, 2004.
- [92] D. A. Keim. Information Visualization and Visual Data Mining. *IEEE Transactions on Visualization and Computer Graphics*, 7(1):100–107, 2002.
- [93] N. W. Kim, S. K. Card, and J. Heer. Tracing genealogical data with TimeNets. In *International Conference on Advanced Visual Interfaces*, pages 241–248, New York, New York, USA, may 2010. ACM Press.
- [94] G. Klein, B. Moon, and R. Hoffman. Making Sense of Sensemaking 1: Alternative Perspectives. *IEEE Intelligent Systems*, 21(4):70–73, jul 2006.
- [95] G. Klein, J. K. Phillips, E. L. Rall, and D. A. Peluso. A Data-Frame Theory of Sensemaking. In R. R. Hoffman, editor, *Expertise out of context: International conference on naturalistic decision making*, pages 113–155. Mahwah, NJ: Lawrence Erlbaum Associates, 2003.
- [96] S. R. Klemmer, M. Thomsen, E. Phelps-Goodman, R. Lee, and J. A. Landay. Where do web sites come from? Capturing and Interacting with Design History. In *ACM Conference on Human Factors in Computing Systems*, pages 1–8, New York, New York, USA, apr 2002. ACM Press.
- [97] R. Kosara, H. Hauser, and D. L. Gresh. An Interaction View on Information Visualization. In *State-of-the-Art Report. EuroGraphics.*, 2003.
- [98] J. B. Kruskal and J. M. Landwehr. Icicle Plots: Better Displays for Hierarchical Clustering. *The American Statistician*, 37(2):162–168, 1983.
- [99] V. Kumar, R. Fur, and R. B. Allen. Metadata visualization for digital libraries: interactive timeline editing and review. In *ACM conference on Digital libraries*, pages 126–133, 1998.

-
- [100] D. Kurlander and S. Feiner. Editable graphical histories. In *IEEE Workshop on Visual Languages*, pages 127–134. IEEE Comput. Soc. Press, 1988.
 - [101] H. Lam, E. Bertini, P. Isenberg, C. Plaisant, and S. Carpendale. Empirical Studies in Information Visualization: Seven Scenarios. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1520–1536, nov 2012.
 - [102] J. Lazar, J. H. Feng, and H. Hochheiser. *Research methods in human-computer interaction*. John Wiley & Sons, Ltd., 2010.
 - [103] A. Lewins and C. Silver. *Using software in qualitative research: A step-by-step guide*. Sage, 2007.
 - [104] S. Liu, Y. Wu, E. Wei, M. Liu, and Y. Liu. StoryFlow: tracking the evolution of stories. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2436–2445, dec 2013.
 - [105] Z. Liu, C. Gorg, J. Kihm, H. Lee, J. Choo, H. Park, and J. Stasko. Data ingestion and evidence marshalling in Jigsaw. In *IEEE Symposium on Visual Analytics Science and Technology*, pages 271–272. IEEE, oct 2010.
 - [106] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, mar 1982.
 - [107] A. Lunzer. Lightweight Provenance-Driven Exploration. In *International Workshop on Analytic Provenance for Sensemaking*, 2014.
 - [108] K.-L. Ma. Image graphs - a novel approach to visual data exploration. In *IEEE Conference on Visualization*, pages 81–88, oct 1999.
 - [109] S. Maitlis and M. Christianson. Sensemaking in Organizations: Taking Stock and Moving Forward. *The Academy of Management Annals*, 8(1):57–125, 2014.
 - [110] K. Marriott and P. Sbarski. Compact layout of layered trees. In *Australasian conference on Computer science*, volume 62, pages 7–14, 2007.
 - [111] J. Matejka, T. Grossman, and G. Fitzmaurice. Patina: Dynamic Heatmaps for Visualizing Application Usage. In *ACM Conference on Human Factors in Computing Systems*, pages 3227–3236, New York, New York, USA, 2013. ACM Press.

- [112] P. McLachlan, T. Munzner, E. Koutsofios, and S. North. LiveRAC: Interactive Visual Exploration of System Management Time-Series Data. In *ACM Conference on Human Factors in Computing Systems*, pages 1483–1492, New York, New York, USA, 2008. ACM Press.
- [113] C. Meng, M. Yasue, A. Imamiya, and M. Xiaoyang. Visualizing histories for selective undo and redo. In *Asian Pacific Computer and Human Interaction*, pages 459–464. IEEE Comput. Soc, 1998.
- [114] W. Meulemans, N. Henry Riche, B. Speckmann, B. Alper, and T. Dwyer. Kelp-Fusion: A Hybrid Set Visualization Technique. *IEEE Transactions on Visualization and Computer Graphics*, 19(11):1846–1858, nov 2013.
- [115] S. Miles, E. Deelman, P. Groth, K. Vahi, G. Mehta, and L. Moreau. Connecting Scientific Data to Scientific Experiments with Provenance. In *IEEE International Conference on e-Science and Grid Computing*, pages 179–186. IEEE, 2007.
- [116] N. Milic-Frayling, R. Sommerer, and K. Rodden. WebScout: support for re-visitation of Web pages within a navigation session. In *International Conference on Web Intelligence*, pages 689–693. IEEE Comput. Soc, 2003.
- [117] L. Moreau, B. Clifford, J. Freire, J. Futrelle, Y. Gil, P. Groth, N. Kwasnikowska, S. Miles, P. Missier, J. Myers, and Others. The open provenance model core specification (v1. 1). *Future Generation Computer Systems*, 27(6):743–756, 2011.
- [118] R. Munroe. Xkcd #657: Movie narrative charts. \url{http://xkcd.com/657}, 2009.
- [119] T. Munzner. *Visualization Analysis and Design*. CRC Press, 2014.
- [120] D. Nadeau and S. Sekine. A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26, 2007.
- [121] A. Nenkova and K. McKeown. A Survey of Text Summarization Techniques. In *Mining Text Data*, pages 43–76. Springer US, Boston, MA, 2012.
- [122] P. H. Nguyen, K. Xu, A. Bardill, S. Betul, K. Herd, and B. L. W. Wong. SenseMap: Supporting Browser-based Online Sensemaking through Analytic Provenance. In *IEEE Conference on Visual Analytics Science and Technology*, 2016.

- [123] P. H. Nguyen, K. Xu, R. Walker, and B. L. W. Wong. SchemaLine: Timeline Visualization for Sensemaking. In *International Conference on Information Visualisation*, pages 225–233. IEEE, jul 2014.
- [124] P. H. Nguyen, K. Xu, R. Walker, and B. L. W. Wong. TimeSets: Timeline visualization with set relations. *Information Visualization*, 15(3):253–269, jul 2016.
- [125] P. H. Nguyen, K. Xu, A. Wheat, B. L. W. Wong, S. Attfield, and B. Fields. SensePath: Understanding the Sensemaking Process through Analytic Provenance. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):41–50, jan 2016.
- [126] Q. V. Nguyen and M. L. Huang. A space-optimized tree visualization. In *IEEE Symposium on Information Visualization*, pages 85–92, 2002.
- [127] C. North, R. Chang, A. Endert, W. Dou, R. May, B. Pike, and G. Fink. Analytic provenance: process+interaction+insight. In *ACM Transactions on Computer-Human Interaction*, pages 33–36. ACM, may 2011.
- [128] S. Pace. A grounded theory of the flow experiences of Web users. *International Journal of Human-Computer Studies*, 60(3):327–363, mar 2004.
- [129] S. Palmer and I. Rock. Rethinking perceptual organization: The role of uniform connectedness. *Psychonomic bulletin & review*, 1(1):29–55, mar 1994.
- [130] M. P. Papazoglou and W.-J. van den Heuvel. Service oriented architectures: approaches, technologies and research issues. *The VLDB Journal*, 16(3):389–415, jul 2007.
- [131] N. Pennington and R. Hastie. Cognitive theory of juror decision making: The story model. *Cardozo L. Rev.*, 13:519, 1991.
- [132] K. Perlin and D. Fox. Pad: an alternative approach to the computer interface. In *ACM Conference on Computer graphics and interactive techniques*, pages 57–64, New York, New York, USA, 1993. ACM Press.
- [133] W. Pike, J. Bruce, B. Baddeley, D. Best, L. Franklin, R. May, D. Rice, R. Riensche, and K. Younkin. The Scalable Reasoning System: Lightweight visualization for distributed analytics. *Information Visualization*, 8(1):71–84, 2009.

- [134] W. A. Pike, J. Stasko, R. Chang, and T. A. O 'connell. The science of interaction. *Information Visualization*, 8(4):263–274, 2009.
- [135] N. J. Pioch and J. O. Everett. POLESTAR - Collaborative Knowledge Management and Sensemaking Tools for Intelligence Analysts. In *ACM International Conference on Information and Knowledge Management*, pages 513–521, New York, New York, USA, nov 2006. ACM Press.
- [136] P. Pirolli and S. Card. The Sensemaking Process and Leverage Points for Analyst Technology as Identified Through Cognitive Task Analysis. In *Conference on Intelligence Analysis*, 2005.
- [137] C. Plaisant. The challenge of information visualization evaluation. In *Working Conference on Advanced Visual Interfaces*, pages 109–116, New York, New York, USA, 2004. ACM Press.
- [138] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. LifeLines: visualizing personal histories. In *ACM Conference on Human Factors in Computing Systems*, pages 221–227, New York, New York, USA, apr 1996. ACM Press.
- [139] C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, and B. Shneiderman. LifeLines: using visualization to enhance navigation and analysis of patient records. *Proceedings of the AMIA Symposium*, 08(98):76–80, 1998.
- [140] C. Plaisant, A. Rose, G. Rubloff, R. Salter, and B. Shneiderman. The design of history mechanisms and their use in collaborative educational simulations. In *Conference on Computer Support for Collaborative Learning*, pages 348–359, dec 1999.
- [141] J. Priestley. *A Chart of Biography*. London: J. Johnson, St. Paul's Church Yard, 1765.
- [142] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.
- [143] E. D. Ragan, A. Endert, J. Sanyal, and J. Chen. Characterizing Provenance in Visualization and Data Analysis: An Organizational Framework of Provenance Types and Purposes. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):31–40, jan 2016.
- [144] H. Reijner. The development of the horizon graph, 2008.

- [145] E. Reingold and J. Tilford. Tidier Drawings of Trees. *IEEE Transactions on Software Engineering*, SE-7(2):223–228, mar 1981.
- [146] G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. Stasko. Effectiveness of animation in trend visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1325–1332, 2008.
- [147] P. Rodgers. A survey of Euler diagrams. *Journal of Visual Languages & Computing*, 25(3):134–155, jun 2014.
- [148] Y. Rogers. HCI Theory: Classical, Modern, and Contemporary. *Synthesis Lectures on Human-Centered Informatics*, 5(2):1–129, 2012.
- [149] D. Rosenberg and A. Grafton. *Cartographies of Time: A History of the Timeline*. Princeton Architectural Press, 2013.
- [150] F. Ruskey and M. Weston. A survey of Venn diagrams. *Electronic Journal of Combinatorics*, 4, 1997.
- [151] D. Russell, R. Jeffries, and L. Irani. Sensemaking for the rest of us. *CHI workshop on Sensemaking*, 2008.
- [152] D. M. Russell, M. J. Stefk, P. Pirolli, and S. K. Card. The cost structure of sensemaking. In *ACM Conference on Human Factors in Computing Systems*, pages 269–276, New York, New York, USA, may 1993. ACM Press.
- [153] T. Saito, H. N. Miyamura, M. Yamamoto, H. Saito, Y. Hoshiya, and T. Kaseda. Two-tone pseudo coloring: Compact visualization for one-dimensional data. In *IEEE Symposium on Information Visualization*, pages 173–180, 2005.
- [154] A. Sanfilippo, B. Baddeley, C. Posse, and P. Whitney. A Layered Dempster-Shafer Approach to Scenario Construction and Analysis. In *IEEE Intelligence and Security Informatics*, pages 95–102. IEEE, may 2007.
- [155] K. Sedig and P. Parsons. Interaction Design for Complex Cognitive Activities with Visual Representations: A Pattern-Based Approach. *Transactions on Human-Computer Interaction*, 5(2):84–133, 2013.
- [156] M. Sedlmair, P. Isenberg, D. Baur, M. Jurmu, M. Oulu, S. Boring, and A. Butz. Requirements for a mde system to support collaborative in-car communication diagnostics. In *CSCW Workshop on Beyond the Laboratory: Supporting Authentic Collaboration with Multiple Displays*, 2008.

- [157] B. Shneiderman. The future of interactive systems and the emergence of direct manipulation. *Behaviour & Information Technology*, 1(3):237–256, 1982.
- [158] B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, jan 1992.
- [159] B. Shneiderman. The Eyes Have It : A Task by Data Type Taxonomy for Information Visualizations. In *IEEE Symposium on Visual Languages*, pages 336–343, 1996.
- [160] Y. B. Srinivasan and J. J. van Wijk. Supporting the Analytical Reasoning Process in Information Visualization. In *ACM Conference on Human Factors in Computing Systems*, pages 1237–1246, New York, New York, USA, apr 2008. ACM Press.
- [161] P. Simonetto, D. Auber, and D. Archambault. Fully Automatic Visualisation of Overlapping Sets. *Computer Graphics Forum*, 28(3):967–974, jun 2009.
- [162] D. Snowden. Multi-ontology sense making: a new simplicity in decision making. *Journal of Innovation in Health Informatics*, 13(1):45–53, mar 2005.
- [163] U. Software. Timelines in the nSpace2 Sandbox, 2012.
- [164] J. Stasko, J. Choo, Y. Han, and M. Hu. Citevis: Exploring conference paper citation data visually. In *Posters of IEEE InfoVis.*, pages 2–3, 2013.
- [165] S. Stemler. An overview of content analysis. *Practical assessment, research & evaluation*, 7(17):137–146, 2001.
- [166] K. Sugiyama, S. Tagawa, and M. Toda. Methods for Visual Understanding of Hierarchical System Structures. *IEEE Transactions on Systems, Man and Cybernetics*, 11(2):109–125, 1981.
- [167] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to data mining*. Pearson Education India, 2006.
- [168] Y. Tanahashi and K.-L. Ma. Design Considerations for Optimizing Storyline Visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2679–2688, dec 2012.

- [169] L. Tauscher and S. Greenberg. How people revisit web pages: empirical findings and implications for the design of history systems. *International Journal of Human-Computer Studies*, 47(1):97–137, 1997.
- [170] J. Teevan, E. Cutrell, D. Fisher, S. M. Drucker, G. Ramos, P. André, and C. Hu. Visual Snippets: Summarizing Web Pages for Search and Revisitation. In *ACM Conference on Human Factors in Computing Systems*, pages 2023–2032, 2009.
- [171] R. Therón. Hierarchical-temporal data visualization using a tree-ring metaphor. In *Smart Graphics*, pages 70–81. Springer Berlin Heidelberg, 2006.
- [172] J. J. Thomas and K. A. Cook. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Center, 2005.
- [173] C. Tominski and H. Schumann. Enhanced Interactive Spiral Display. In *The Annual SIGRAD Conference Special Theme: Interaction*, pages 53–56, 2008.
- [174] M. Tory and T. Moller. Evaluating Visualizations: Do Expert Reviews Work? *IEEE Computer Graphics and Applications*, 25(5):8–11, sep 2005.
- [175] S. E. Toulmin. *The uses of argument*. Cambridge University Press, 2003.
- [176] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Connecticut, 1983.
- [177] E. R. Tufte. *Envisioning Information*. Graphics Press, 1990.
- [178] University of Utah. VisTrails Documentation Release 2.0.0. Technical report, 2012.
- [179] A. van der Ploeg. Drawing non-layered tidy trees in linear time. *Software: Practice and Experience*, 44(12):1467–1484, dec 2014.
- [180] F. van Ham and A. Perer. “Search, Show Context, Expand on Demand”: Supporting Large Graph Exploration with Degree-of-Interest. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):953–960, nov 2009.
- [181] J. Van Wijk and E. Van Selow. Cluster and calendar based visualization of time series data. In *IEEE Symposium on Information Visualization*, pages 4–9. IEEE Comput. Soc, 1999.

- [182] R. Walker, A. Slingsby, J. Dykes, K. Xu, J. Wood, P. H. Nguyen, D. Stephens, B. L. W. Wong, and Y. Zheng. An extensible framework for provenance in human terrain visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2139–2148, dec 2013.
- [183] T. D. Wang, C. Plaisant, A. J. Quinn, R. Stanchak, S. Murphy, and B. Shneiderman. Aligning temporal data by sentinel events: Discovering Patterns in Electronic Health Records. In *ACM Conference on Human Factors in Computing Systems*, pages 457–466, New York, New York, USA, apr 2008. ACM Press.
- [184] W. Wang, H. Wang, G. Dai, and H. Wang. Visualization of large hierarchical data by circle packing. In *ACM Conference on Human Factors in Computing Systems*, page 517520, New York, New York, USA, 2006. ACM Press.
- [185] C. Ware. *Information Visualization: Perception for Design*. Elsevier, 2013.
- [186] S. J. Waterson, J. I. Hong, T. Sohn, J. A. Landay, J. Heer, and T. Matthews. What did they do? understanding clickstreams with the WebQuilt visualization system. In *International Working Conference on Advanced Visual Interfaces*, pages 94–102, New York, New York, USA, may 2002. ACM Press.
- [187] M. Weber, M. Alexa, and W. Muller. Visualizing time-series on spirals. In *IEEE Symposium on Information Visualization*, pages 7–13. IEEE, 2001.
- [188] K. E. Weick. Sensemaking in organizations. *Sage*, 3, 1995.
- [189] J. J. V. Wijk. Cushion Treemaps : Visualization of Hierarchical Information. In *IEEE Symposium on Information Visualization*, pages 73–78, 1999.
- [190] L. Wilkinson. *The Grammar of Graphics*. Statistics and Computing. Springer-Verlag, New York, 2005.
- [191] M. J. Wilson and M. L. Wilson. A comparison of techniques for measuring sensemaking and learning within participant-generated summaries. *Journal of the American Society for Information Science and Technology*, 64(2):291–306, feb 2013.
- [192] B. L. W. Wong and A. E. Blandford. Analysing ambulance dispatcher decision making: trialing emergent themes analysis. 2002.

- [193] W. Wong, R. Chen, N. Kodagoda, C. Rooney, and K. Xu. INVISQUE: intuitive information exploration through interactive visualization. In *Extended Abstracts on Human factors in computing systems*, pages 311–316, New York, New York, USA, may 2011. ACM Press.
- [194] W. Wright, D. Schroh, P. Proulx, A. Skaburskis, and B. Cort. The sandbox for analysis: concepts and methods. In *ACM Conference on Human Factors in Computing Systems*, pages 801–810, New York, New York, USA, apr 2006. ACM Press.
- [195] K. Xu, S. Attfield, T. J. Jankun-Kelly, A. Wheat, P. H. Nguyen, and N. Selvaraj. Analytic provenance for sensemaking: a research agenda. *IEEE Computer Graphics and Applications*, 35(3):56–64, jan 2015.
- [196] K. Xu, P. H. Nguyen, and B. Fields. Visual analysis of streaming data with SAVI and SenseMAP. In *IEEE Conference on Visual Analytics Science and Technology*, pages 389–390. IEEE, oct 2014.
- [197] K. Xu, C. Rooney, P. Passmore, D.-H. Ham, and P. H. Nguyen. A User Study on Curved Edges in Graph Visualization. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2449–2456, dec 2012.
- [198] J. S. Yi, Y. A. Kang, J. Stasko, and J. Jacko. Toward a deeper understanding of the role of interaction in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1224–1231, jan 2007.
- [199] E. Zhang and J. Stasko. Focus+ Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations. In *IEEE Symposium on Information Visualization*, pages 57–65, 2000.
- [200] J. Zhao, C. Goble, R. Stevens, and D. Turi. Mining Taverna’s semantic web of provenance. *Concurrency and Computation: Practice and Experience*, 20(5):463–472, apr 2008.
- [201] T. Zuk, L. Schlesier, P. Neumann, M. S. Hancock, and S. Carpendale. Heuristics for information visualization evaluation. In *AVI workshop on Beyond time and errors: novel evaluation methods for information visualization*, pages 1–6, New York, New York, USA, 2006. ACM Press.