1 Literature Review

First, this chapter reviews the core work in sensemaking and visualization. Then, it discusses the literature on analytic provenance with a focus on visualization of provenance data for supporting sensemaking. Finally, it presents visualization techniques of time-oriented and network data because these two data types are heavily involved in the research questions addressed in the thesis.

1.1 Sensemaking

Sensemaking reflects how we make sense of the world so that we can act in it [138]. Sensemaking has been studied in many different contexts, most notably including information science [32], organization [160], human-computer interaction [130] and intelligence analysis [86, 118]. This section reviews the sensemaking research discussed in these contexts.

1.1.1 Gap-Bridging Metaphor

Dervin [32] develops a sensemaking theory focusing on information seeking and use behavior. It underlies the cognitive gap that individuals experience when attempting to make sense of observed data. Figure 1.1 summarizes this *gap-bridging* metaphor. The theory assumes that people move through time-space in some particular context and situation. Sensemaking starts when people encounter a gap that prevents their movement and needs to be overcame such as some unclear or confused problem. To bridge that gap, or to address that problem, 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: whether they help or impede the movement.

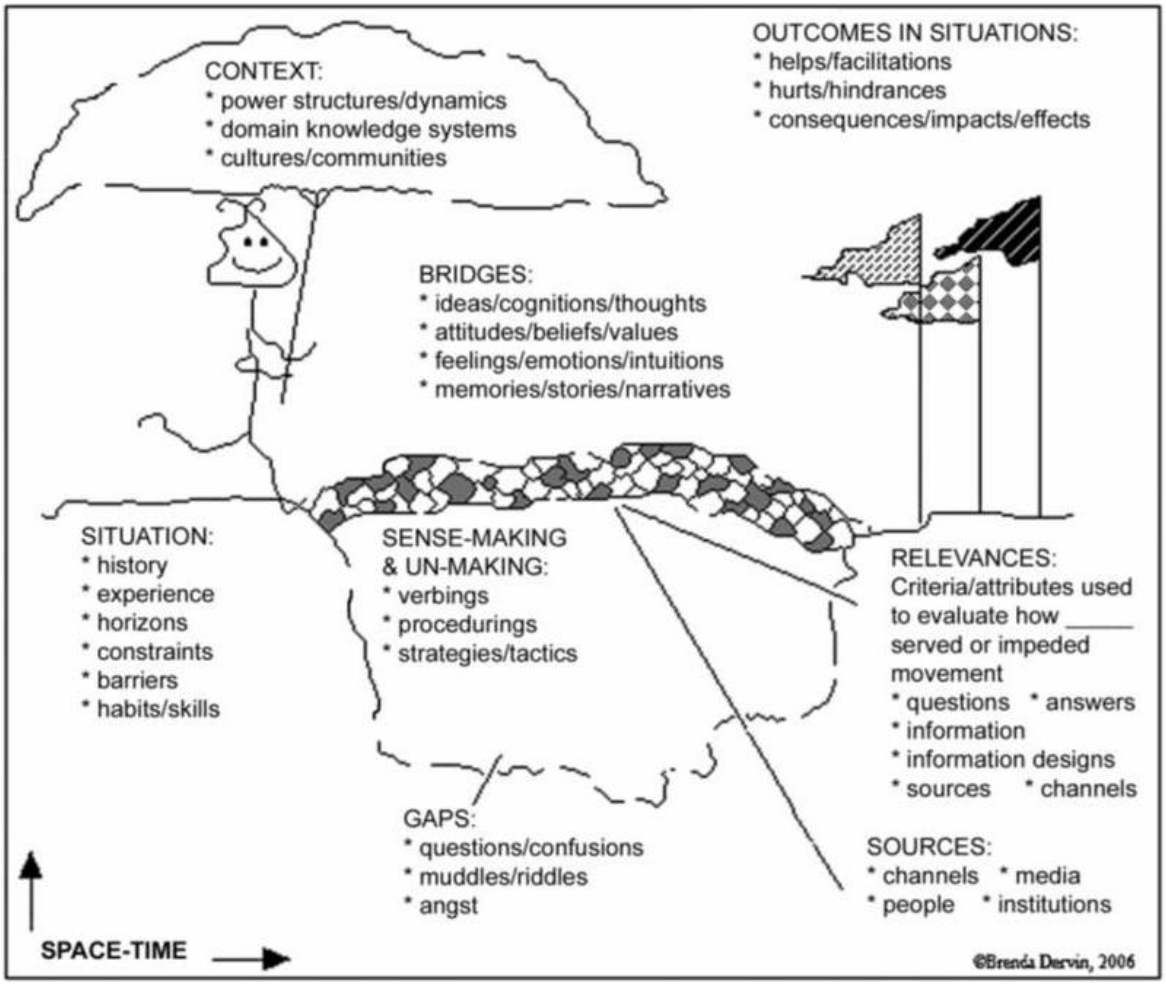


Figure 1.1: The gap-bridging metaphor of sensemaking. People encounter gaps while moving through time-space, then seek for information, evaluate and use it to bridge the gaps. *Image source: [33].*

Dervin also implements the theory into a set of questions that can be used in interview to understand sensemaking within a context [32]. The questions elaborate all parts of the model, aiming to establish an understanding of 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?*).

1.1.2 Sensemaking in Organizations

Different from Dervin who studies sensemaking for individuals, Weick focuses on sensemaking at an organization level [160]. 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 an organization where people interact and socialize with others, and are also influenced by others.
5. *Ongoing*. Sensemaking is continuous because the world and our understanding about it are constantly changing.
6. *Focused on and by extracted cues*. Cues are things 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 focuses on plausibility and sufficiency rather than accuracy and completeness. People tend to stop searching when they find an acceptable solution.

1.1.3 Learning Loop Complex

Russell et al. [130] define sensemaking as the process of searching for a representation and encoding data into that representation to answer task-specific questions. That cyclic process is called the *learning loop complex* as illustrated in Figure 1.2. First, the sensemaker (the person who is making sense of a problem) 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 to produce a more suitable one. This entire learning loop complex is guided by the task with an aim to reduce its cost.

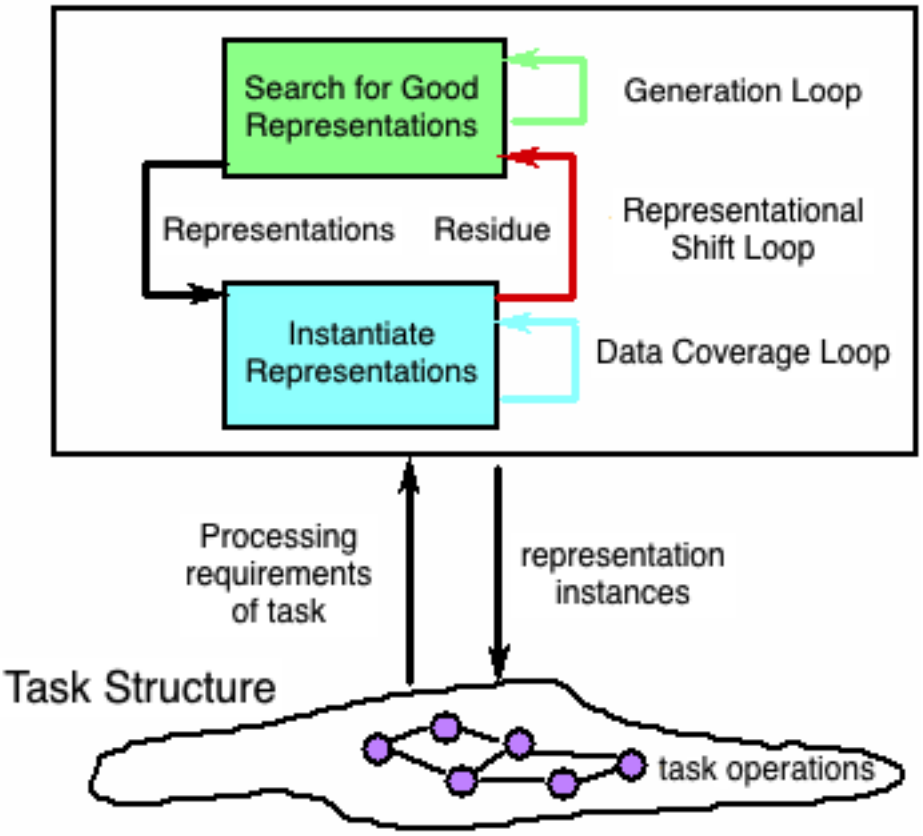


Figure 1.2: The learning loop complex theory of sensemaking. It consists of three iterative loops: searching for a good representation, encoding data into the representation, and adjusting that representation for a better data coverage. These loops are guided by the task and the instantiated representations are then used to implement the task. *Image source: [130].*

1.1.4 A Process Model

Pirolli and Card [118] 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 1.3. 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.

- *Search and filter*. External data sources, such as classified databases or the web, are searched and filtered to retrieve relevant documents to the task.

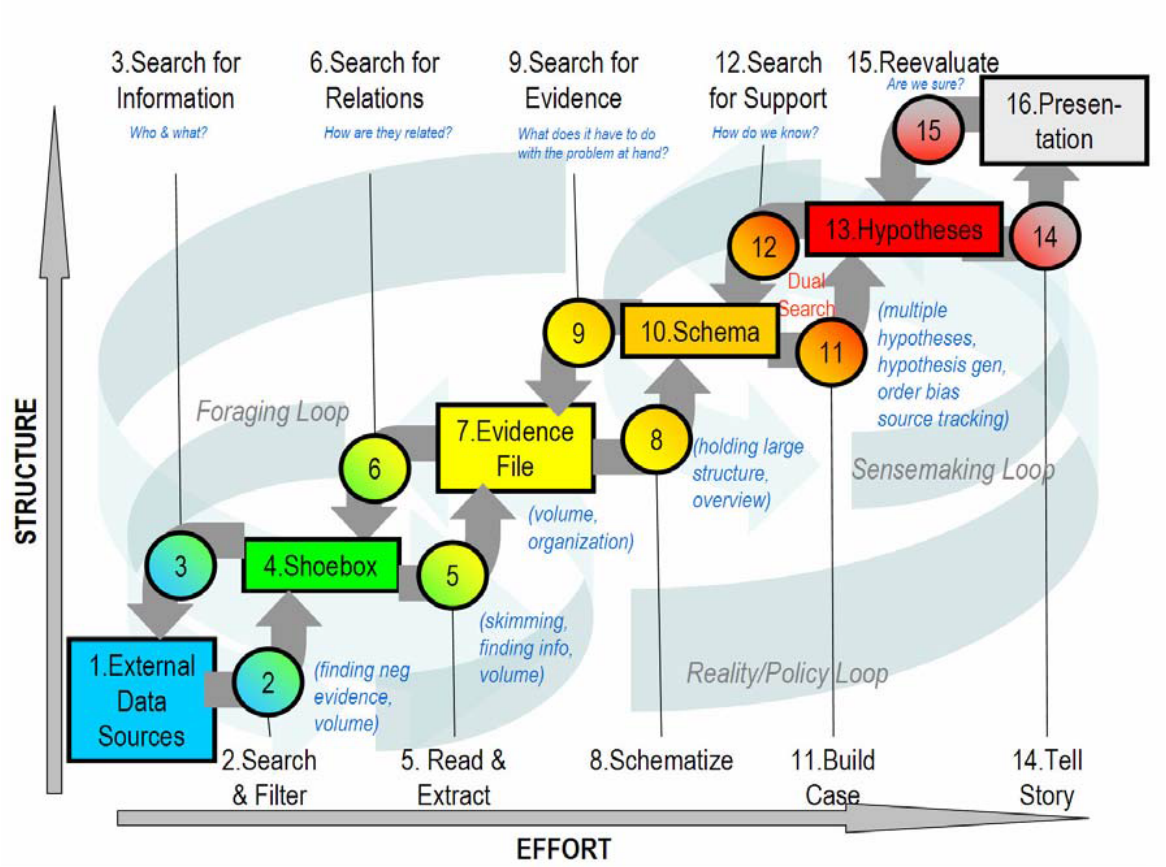


Figure 1.3: A notional model of sensemaking. The sub-processes (numbered circles) and their data input/output (numbered rectangles) are arranged in a two-dimensional space, in which the horizontal axis represents the degree of effort from users, and the vertical axis represents the degree of structure in information representation. *Image source: [118].*

- *Read and extract*. These documents are examined to extract pieces of important information that may be used as evidence later.
- *Schematize*. The collected information is organized in a way that aids the analysis. This organization may be executed implicitly in one's mind, using paper and pen, or with support of a complex computer-based system.
- *Build case*. Multiple hypotheses are generated, and evidence are marshaled to support or disconfirm them.
- *Tell story*. Discovered cases are presented to some audience of interest.

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 [79] also agrees with this suggestion. In their study, all the participants who performed sensemaking tasks well spent considerable time and effort in organizing their collected information. Their organizational schemas 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.

1.1.5 Data–Frame Model

Klein et al. [86] 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 1.4.

- *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 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*. The question 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 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.
- *Reframe*. When encountering inconsistent and contrary data, a person may need to find a replacement to explain all data. Considering discarded data and/or reinterpreting data could facilitate this activity.

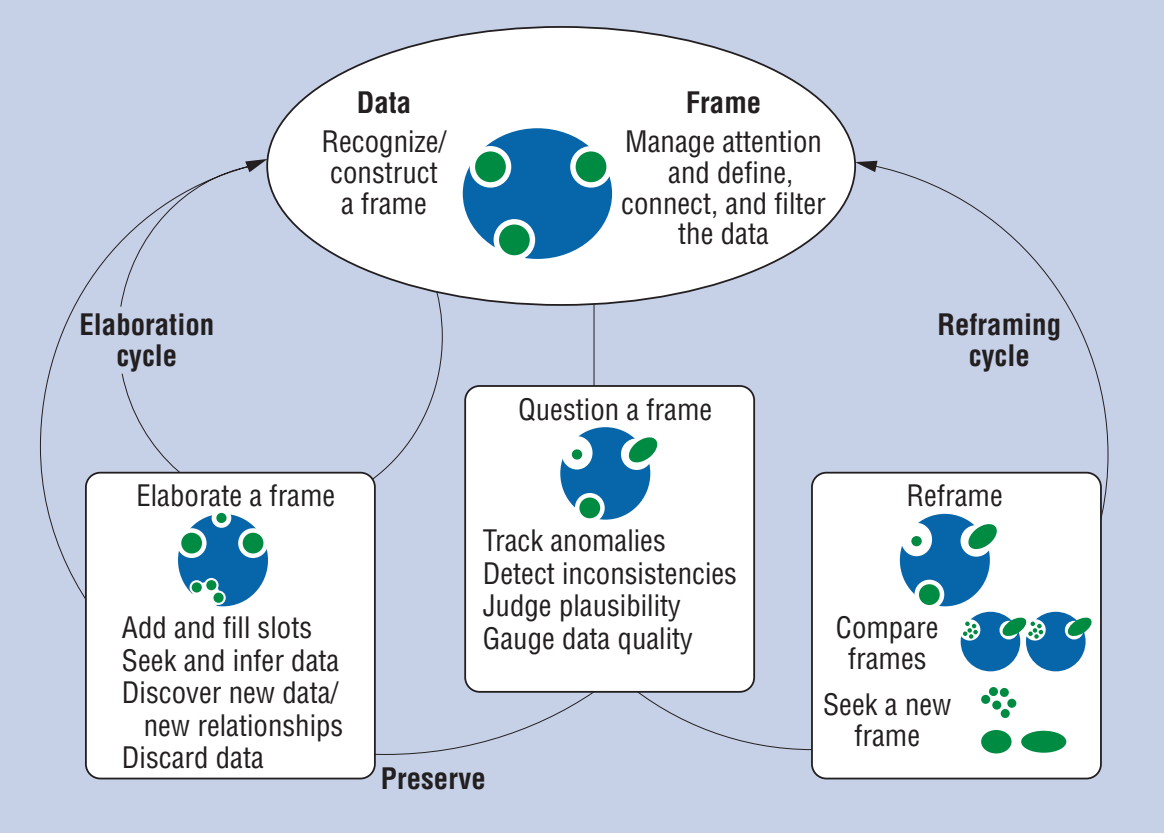


Figure 1.4: The data–frame model of sensemaking. It describes a set of interconnected sensemaking activities centering around data and frame – the explanatory structure of data. *Image source: [86].*

- *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.

1.1.6 Summary

Even though being conducted in different contexts, sensemaking research agrees on many common points. Dervin [32] describes sensemaking as a deliberate effort to bridge the gap of knowledge that a person encounters while solving a problem. More specifically, Russell et al. [130] propose an iterative process of searching for

an appropriate representation of data to bridge such a gap, or to reduce the cost of sensemaking operations. Klein et al. [86] make the representation shift more explicitly by describing seven different sensemaking activities between data and the representation, or frame. More completely, Pirolli and Card [118] suggest a process model of sensemaking that covers searching and filtering relevant information, organizing them, generating hypotheses and presenting the final outcome.

Supporting sensemaking is challenging because the process usually happens implicitly inside a person's head. The aforementioned sensemaking models help unfold this tacit process and provide guidance for improvement. This thesis uses the Pirolli and Card's model and the Data–Frame model as the theoretical foundation for sensemaking because of their comprehensive structure. **??** and **??** focus on the schematization process and support all sensemaking activities in the Data–Frame model. **??** provides sensemaking support based on a simplified version of the Pirolli and Card's model. The support is achieved through the externalization and visualization of an implicit sensemaking process. To provide an overall understanding of the power of visualization, the next section will discuss its core concepts and research.

1.2 Visualization and Visual Analytics

1.2.1 Overview

Computer-based visualization systems provide visual representations of datasets designed to help people carry out tasks more effectively [110]. Card, Mackinlay and Shneiderman, in their seminal *Readings in Information Visualization* book [19] propose major ways in which external visual representations can amplify human cognition. First, it can leverage the limited working memory of human by offloading work from cognitive to perceptual system. Visualization can enhance recognition of patterns and reduce the effort of exhaustive searching for information. Unlike static diagrams, visualization can offer interactive operations to allow exploration of large and complex datasets from different perspectives.

Anscombe's quartet [6] is a classic example for the benefit of displaying all data points in addition to a data summary by descriptive statistics. Figure 1.5 shows scatter plots of four small datasets with identical descriptive statistics including mean, variance, correlation and linear regression line. However, the structures of these datasets are completely different. The top-left plot has a typical structure

for a positive correlation with points gathering around the linear regression line. However, the top-right plot shows a nonlinear pattern in the data. Both datasets at the bottom show how an outlier makes the linear regression line differ from the true pattern with a small change in the left dataset and a complete change in the right one. Again, graphically showing all data points helps us easily see all these patterns that are hidden in the descriptive statistics.

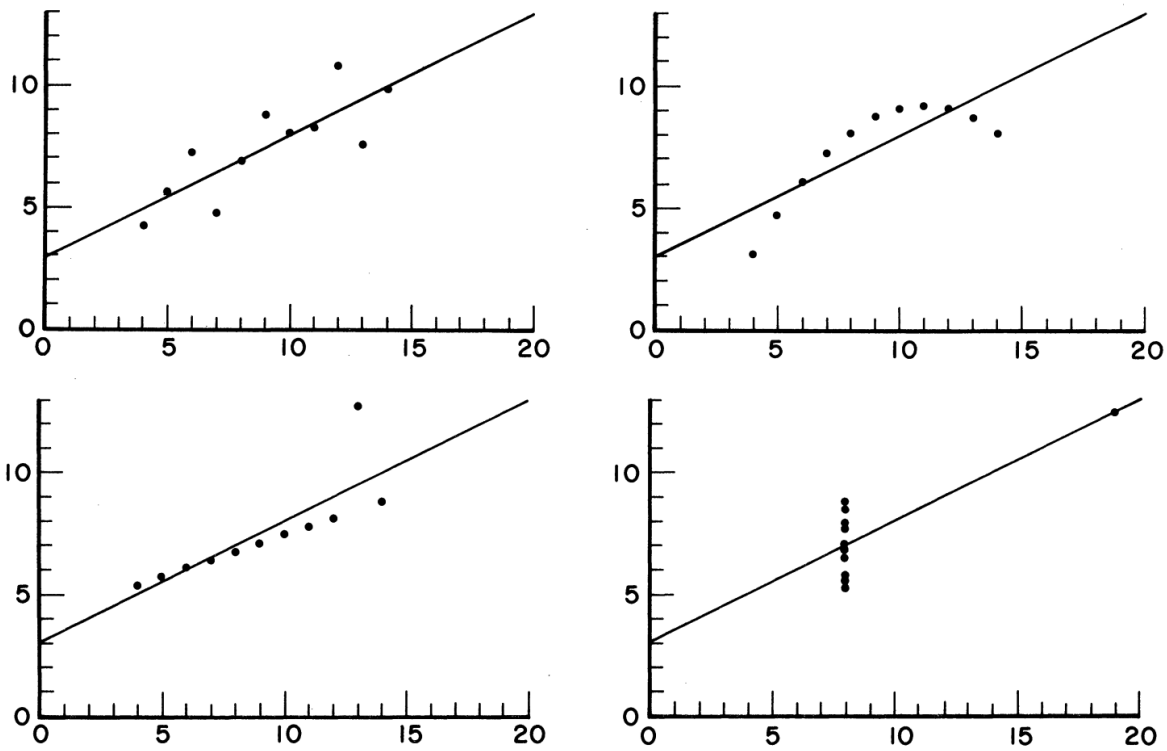


Figure 1.5: Four datasets with identical statistics can have very different structures that are easily seen by using simple graphics. *Image source: [6].*

For a much larger and more complex dataset, a naive visual representation of the entire dataset often leads to a messy and ineffective visualization. In this case, analysis techniques in machine learning and data mining can be complemented such as applying a clustering method to aggregate data into a representative and more manageable set before visualizing it. The combination of interactive visualization and automated data analysis sets the foundation for the field *visual analytics* [81]. In the pioneering book *Illuminating the Path* by Thomas and Cook, with a human sensemaking emphasis, visual analytics is defined as "the science of analytical reasoning facilitated by interactive visual interfaces" [145]. Keim et al. [82] suggest a process model for visual analytics as in Figure 1.6 with interactive visualization

and data analysis model as the two main components to transform data input to knowledge output.

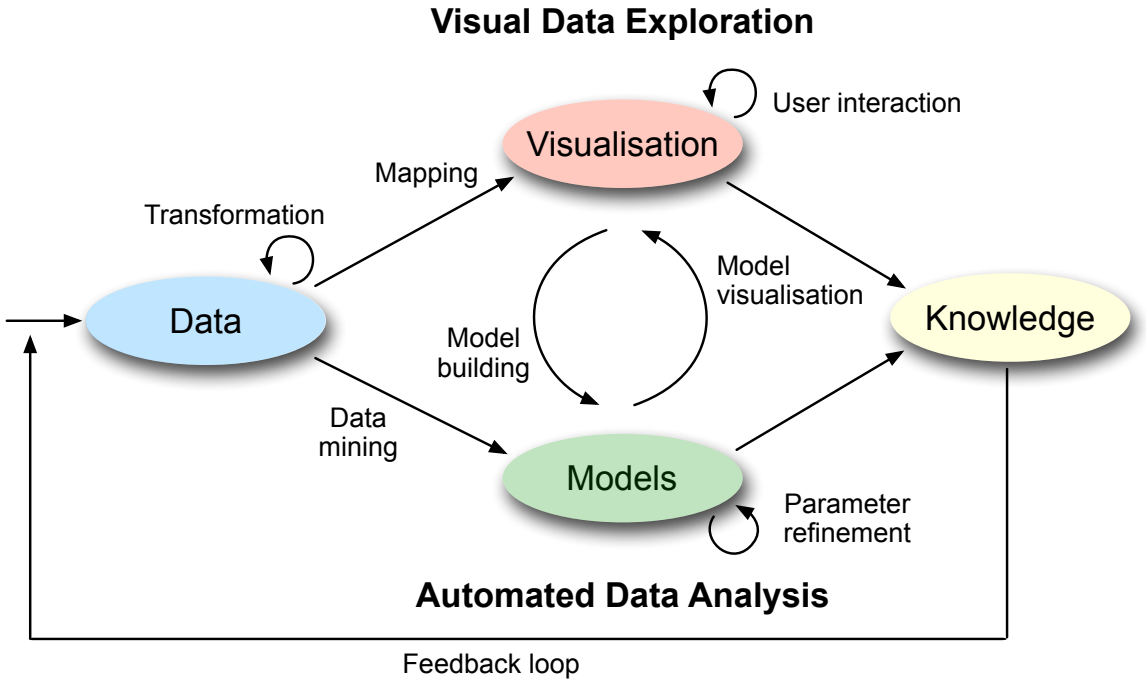


Figure 1.6: An iterative visual analytics process model. It centers around the interaction between data, visualization, models about the data and the users in order to produce knowledge. *Image source: [82].*

Next, we will review the core work in interactive visualization (Section 1.2.2) and automated data analysis (Section 1.2.3). An essential part in every visualization and visual analytics system is to perform validation to check whether the product meets its design purposes and to understand how it helps or hinder users, which will be discussed in Section 1.2.4.

1.2.2 Visualization Design

1.2.2.1 Information Design Principles

Marks and Channels Marks are basic geometric elements that depict items or links, and channels control their appearance [110]. The number of dimensions in item marks can be zero as a *point*, one as a *line*, two as an *area*, and three as a *volume*. Link marks include *connection* showing a pairwise relationship between two items using a line and *containment* showing hierarchical relationship using areas. Figure 1.7 illustrates these marks.

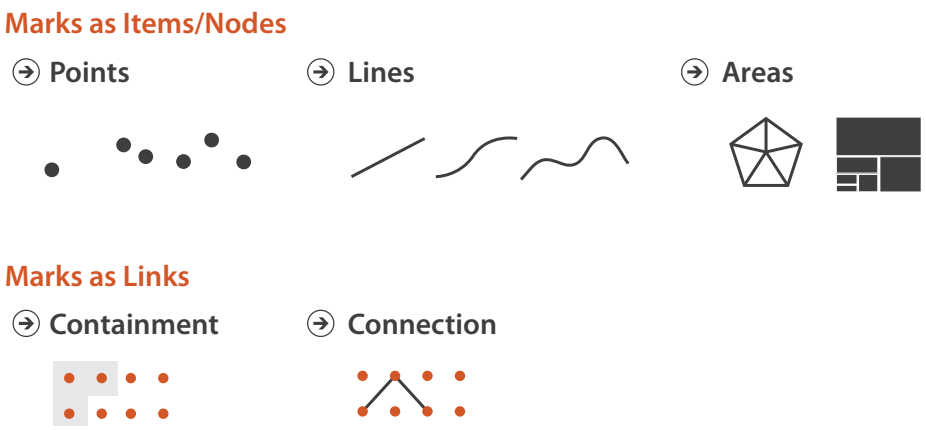


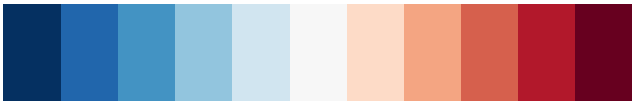
Figure 1.7: Item and link marks as geometric primitives. *Image source: [110].*

A visual channel or graphical attribute controls the appearance of marks such as position, color, shape, angle and size. However, not all channels can be applied to all marks. For instance, an area mark can be used in a geographic map to denote a region. Because the area mark (as a geographic region) is already associated with a shape, it cannot be size-coded to represent another quantitative attribute. 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 non-ordered 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 (non-ordered) attributes should be shown using identity channels, with *spatial region* as the most effective channel and *shape* as the least effective one. Munzer's book [110] describes the detailed ranking of effectiveness of visual channels, which is based on empirical studies such as the work by Cleveland and McGill [24], and by Heer and Bostock [62].

Color is a special channel that can be used for both data attribute types. 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 such an effective colormap is challenging. ColorBrewer [58] is an excellent source for colormap reference, providing color schemes for both ordered and categorical attributes. Human can only distinguish around 12 colors at the same time [110]. Figure 1.8a shows such a categorical colormap with 12 distinguished color hues. Ordered colormaps can be either sequential (Figure 1.8b) or diverging (Figure 1.8c). Diverging colormaps use two different color hues to emphasize values below and above the middle point.

(a) Categorical colormap with distinguishable color hues.

(b) Sequential colormap: a single color hue with different saturation level.

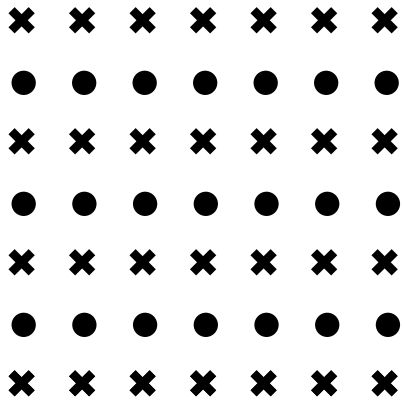


(c) Diverging colormap: two color hues emphasizing positive and negative values.

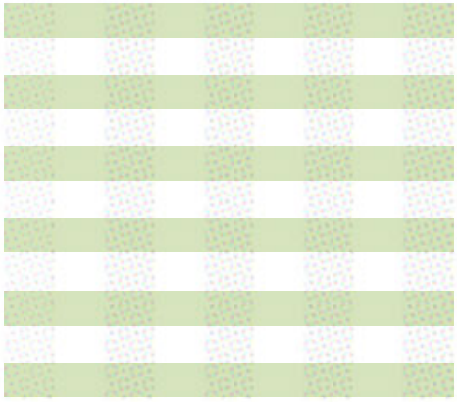
Figure 1.8: Colormaps from ColorBrewer. *Image source: [58].*

Gestalt Principles Gestalt principles describe how people see patterns in visual display [76]. This section reviews three commonly used Gestalt principles.

Similarity Similar elements tend to be perceived as a group. Figure 1.9a shows a matrix of point marks with uniform spacing, but using two different shapes: dot and 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 1.9b, color (green) is used to depict rows, and texture is used to depict columns.



(a) Similarity of shapes distinguishes rows.



(b) Color and texture delineate rows and columns, respectively.

Figure 1.9: Similarity principle: similar elements are perceived as a group. *Image source: [157].*

Proximity Elements that are close together are perceptually grouped together. Figure 1.10a shows two clear groups of dots. Figure 1.10b shows rows of dots. However, with a small change of spacing, these dots are perceived as columns in Figure 1.10c. The application of this principle is straightforward: organizing related information close together. It helps separate groups of unrelated objects and facilitates searching for information.

(a) Two groups of dots. **(b)** Rows of dots. **(c)** Columns of dots.

Figure 1.10: Spatial proximity principle: spatially close elements are perceived as a group. *Image source: [157].*

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 1.11a. This is extensively applied in designing complex graphical user interface: groups of related features are separated by borders. Another way to achieve connectedness is by drawing lines between related elements as in Figure 1.11b. This is the basics of *node-link diagrams* – one of the most common methods of representing relationships between elements.

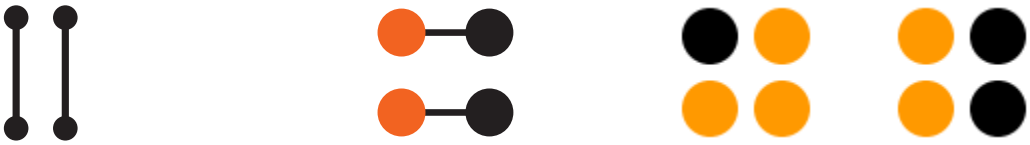
Among these three principles, connectedness has the strongest effect, followed by proximity and then similarity. Figure 1.12 illustrates this comparison. In Figure 1.12a, even though spacing between dots in rows is shorter than spacing between dots in columns, the connected lines make the vertical links has a stronger grouping effect than rows. In Figure 1.12b, the lines also make the horizontal links more notable

(a) Using border to denote a group.

(b) Using lines to denote a group.

Figure 1.11: Connectedness principle: visually connected elements are perceived as a group. *Image source: [157].*

than groups of colored circles. In Figure 1.12c, two spatial groups are more clearly perceived than colored groups.



(a) Links are more clearly perceived than spatial groups.

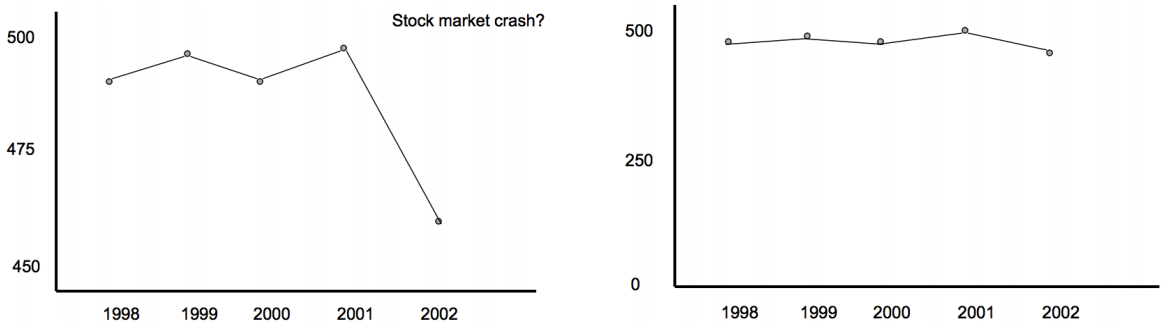
(b) Links are more clearly perceived than colored groups.

(c) Spatial groups are more clearly perceived than colored groups.

Figure 1.12: Comparison of Gestalt principles. Connectedness is stronger than proximity, and proximity is stronger than similarity. *Image source: [157].*

Tufte's Principles Tufte proposes a number of principles in designing effective graphics in his series of books, most notably including *The Visual Display of Quantitative Information* [149] and *Envisioning Information* [150]. This section reviews three 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. Figure 1.13a shows a falsely big drop in stock market value between 2001 and 2002. It is because the chart uses a relative scale with the value range from 450 to 500, causing its height disproportional to the market value. Figure 1.13b corrects this error by using an absolute scale with the value range starting from 0.

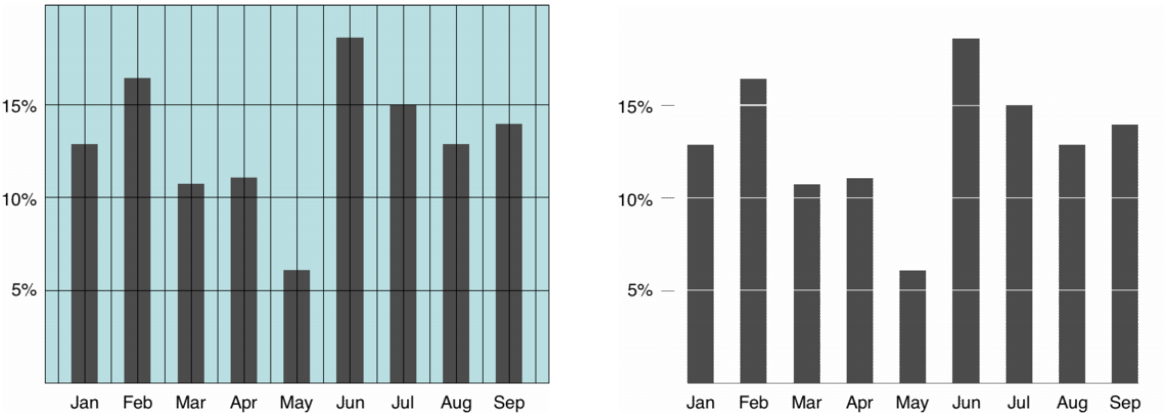


(a) Using a relative value range causes a falsely big drop of stock market value between 2001 and 2002.

(b) Using an absolute value range to depict the data accurately.

Figure 1.13: Graphical integrity principle. The chart should tell the truth about the data.

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 1.14 illustrates this principle.



(a) A bar chart with poor data-ink ratio.

(b) A bar chart with high data-ink ratio by removing background, border and grid lines.

Figure 1.14: 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 before drilling-down closely to examine its individual pieces. Classic stem-and-leaf plot is a great example of this principle

(Figure 1.15). The plot shows all individual data items at an understandable level of detail, and from an overview, it provides the data distribution. The micro/macro principle is extensively applied in interactive visualization, where zooming and panning are made possible.

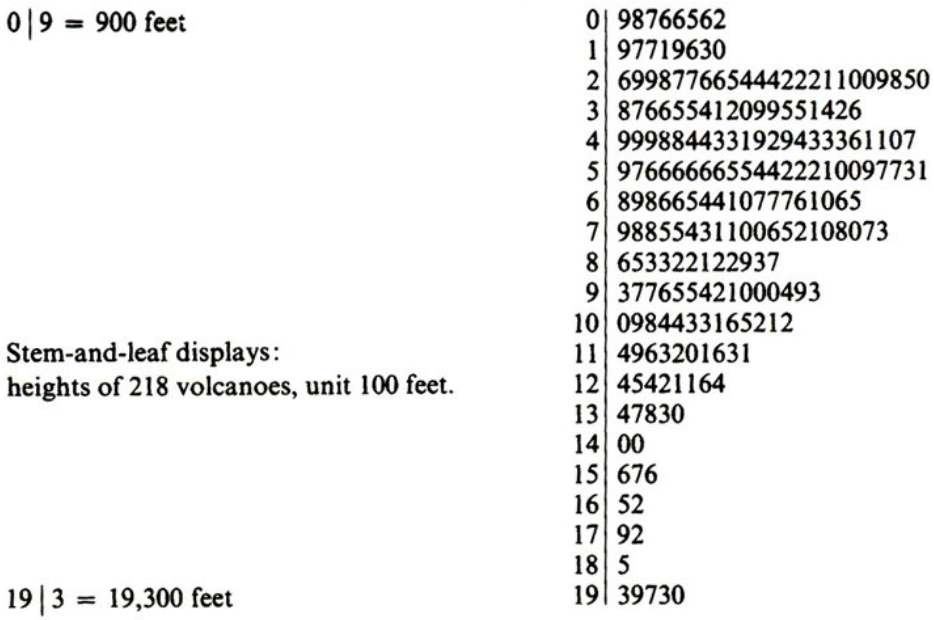


Figure 1.15: Micro/macro principle. A stem-and-leaf plot shows both the data distribution and individual items. *Image source: [149].*

1.2.2.2 Interaction Techniques

Interaction typically refers to the set of controls provided to the user to manipulate an interface [116]. A static visualization can only show one aspect from 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 addressing this problem. 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 commonly used in graphical user interface such as mouse clicking and scrolling, and more visualization specific techniques such as *linking and brushing* [88]. Commonly, a visualization consists of multiple views, each showing an aspect of the dataset. These views should be linked

together to best 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. Besides spatial selection, data can also be selected by data-related similarity with a given point such as all items having the same given attribute value [61]. The data points are brushed in one view and are highlighted in other views, allowing the user to explore them with different perspectives and representations (Figure 1.16).

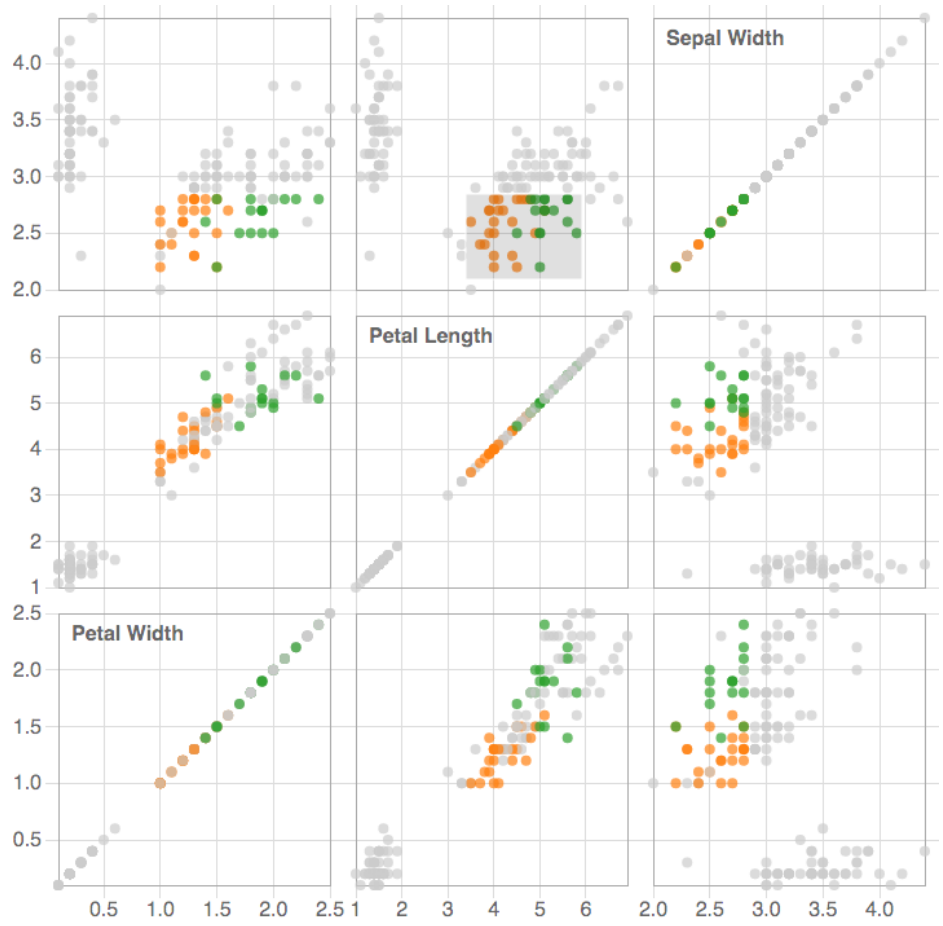


Figure 1.16: Linking and brushing. Data points brushed in one view are linked and highlighted in other views.

Many approaches are proposed to visualize a large amount of information such as overview and detail [26] and semantic zooming [114]. The former method uses two views: one to see the current detail and one to keep track of its global context. This is analogous to the world we can see and its geographic map. The later method uses a single view but with representations for different levels of detail. It lets the user to adjust zoom level to explore the information of interest. Focus+Context is another

technique that brings both the overview (context) and the detailed information (focus) together in one view. Fisheye [44, 45] is one example of this technique: the focal region is magnified and displayed within its surrounding context (Figure 1.17). For all these techniques, interaction is the key factor allowing the user to navigate through a large amount of information and examine the one of interest.

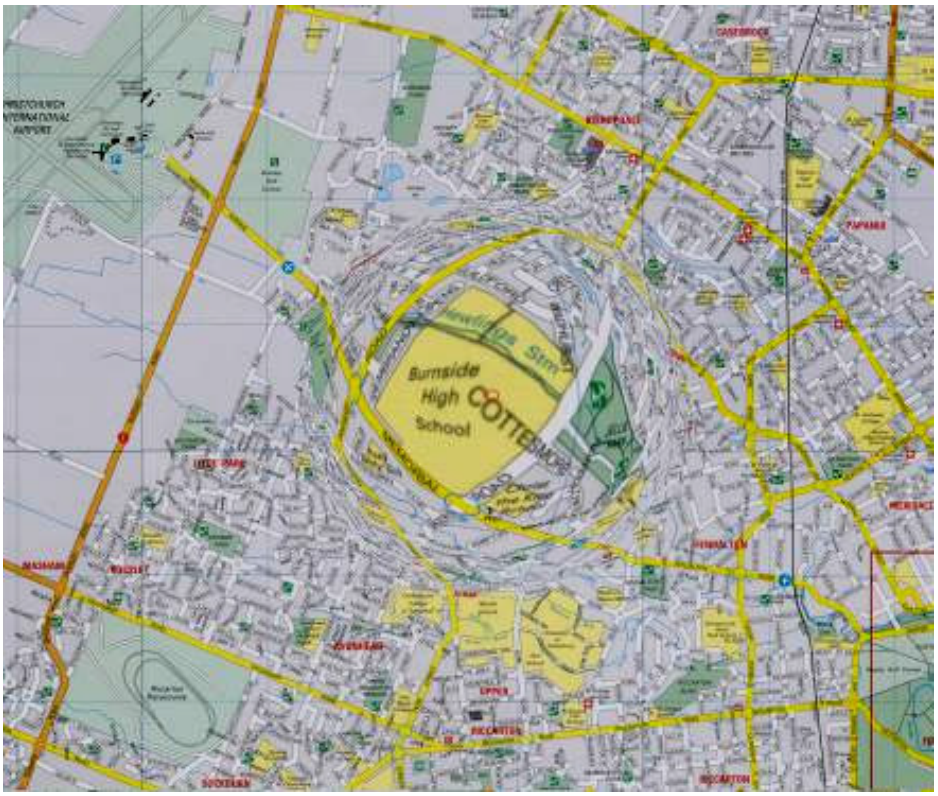


Figure 1.17: Fisheye view for focus+context. Both the overview and the detailed information are displayed in one view.

Taxonomies of interaction techniques by type include the work by Dix and Ellis [34], by Keim [84], and by Wilkinson [162]. Very often, a user performs a particular interaction (or a series of them) to achieve some goal, thus interaction techniques can also be classified based on their intent. Different interaction techniques in different visualizations may actually serve for the same purpose. For example, both drilling-down in a treemap [135] and semantic zooming aim to get more details. Taxonomies of high-level interaction can be found in the work by Yi et al. [168], by Heer and Shneiderman [65], and by Brehmer and Munzner [13]. 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. Its disadvantage is that visual feedback does not appear where the interaction happens, but elsewhere in the visualization. It also takes time for the user to search for the appropriate setting controllers. Direct manipulation [134] of visualization is an approach to address this problem. It enables the user to directly interact with the visual representation and receive immediate feedback. One example is that the axes of a parallel coordinates plot can be reordered by direct dragging and values can be filtered by direct brushing on the axes (Figure 1.18). Surrogate objects can be effective when the data objects are small or distant, thus difficult to manipulate [23]. An example is the use of interactive legends as filtering means [68].

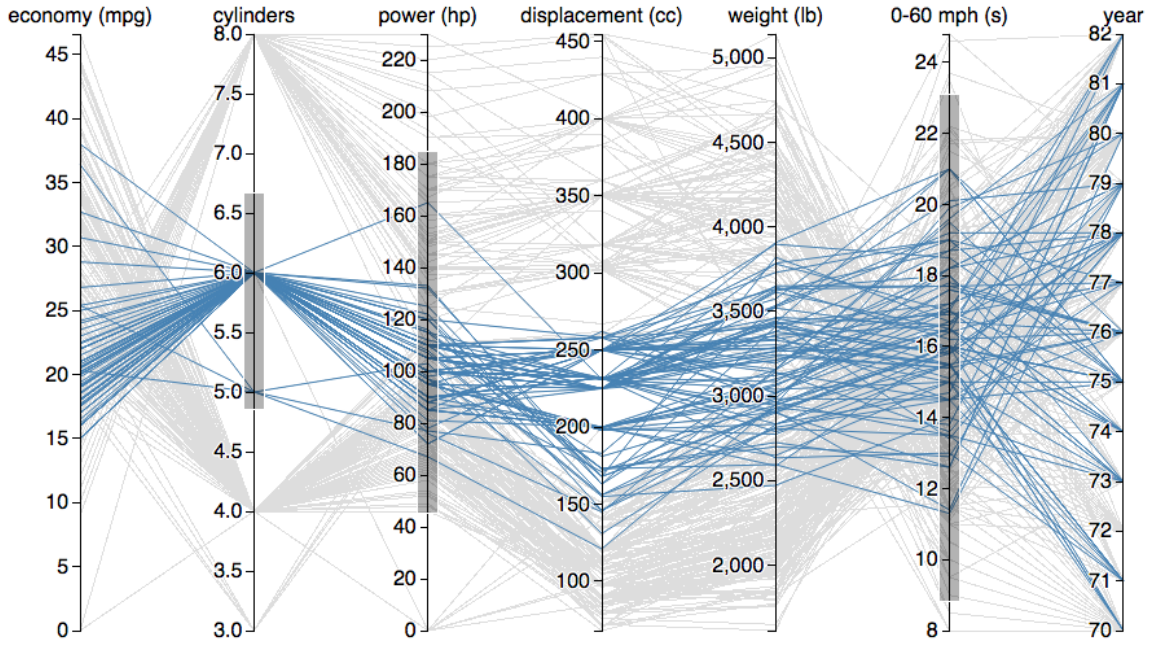


Figure 1.18: Direct manipulation in a parallel coordinates plot with reorder-able and brush-able axes.

Fluid interaction [39] can be applied to improve existing interaction techniques. Besides using direct manipulation as discussed previously, the interaction should produce a smooth animated transition between the state before and the state after an interaction, helping users maintain their mental maps. It also needs to provide immediate visual feedback, allowing users to know what is happening and/or what will happen next.

Interaction techniques are often combined to explore the data or explain a known story. A classic visual information-seeking mantra proposed by Shneiderman [136]

summarizes many information design guidelines and interaction techniques for designing effective information visualizations: *Overview first, zoom and filter, then details-on-demand*. With large datasets, it is challenging to create an overview and provide cues for further exploration. A more suitable approach in this case is *Search, show context, expand on demand* [153].

1.2.3 Automated Analysis Techniques

Data mining is the computational process of extracting patterns in large datasets [141]. 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 summarize the underlying relationships in data. Example tasks 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. Example tasks 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 discuss the clustering and classification tasks in more detail and present how they are applied together with visualization to help users gain deeper insight into the data.

1.2.3.1 Clustering

Overview Cluster analysis finds similarities between data points based on their attributes and groups similar data points into clusters. Clustering is regarded as

unsupervised learning [57] 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* [97]. Given a set of data points $(x_1, x_2, \ldots, x_n)$, k-means clustering aims to partition them into k clusters $(S_1, S_2, \ldots, S_k)$, 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 or centroids in S_i.

The algorithms work as follows. Initially, it partitions data points into k non-empty random subsets. Then, the centroids of the current clusters are computed, and each data point is assigned to the cluster with the nearest centroid. The centroid computation and cluster assignment are repeated 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 and can be found in data mining textbooks [57, 141].

Application Examples Human motion tracking data has been applied in various research fields such as medicine, sports and animation [9]. The data consists of temporal sequences of human poses represented by a set of 3D joint positions (e.g., head, hands, elbows and knees). However, analyzing a large collection of such temporal and high-dimensional datasets is challenging. To gain an overall understanding of the data, MotionFlow [74] applies a k-means clustering method using a simple Euclidean distance of 3D joints as the similarity measurement. A cluster of human poses is represented as a glyph with a stick figure showing the centroid of the cluster and semi-transparent ghosts around the center figure for other similar poses (Figure 1.19).

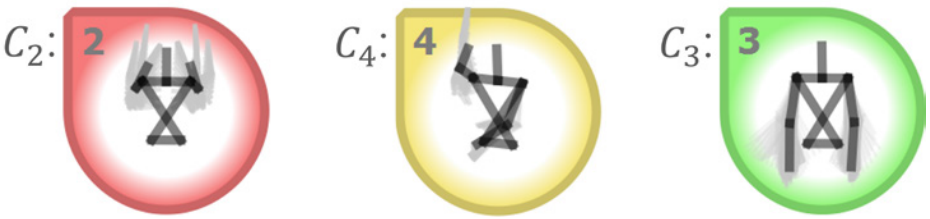


Figure 1.19: Visual representation of human pose clusters. The dark gray stick figure represents the centroid of the cluster, and other light gray ones indicate other poses in the cluster. *Image source: [74].*

MotionFlow then uses a force-directed graph of clusters to illustrate their relationship with node distance mapping to the similarity of pose clusters and edges indicating the directed transition between two clusters. Edges are color coded to represent the transition frequency. A user is allowed to interactively change the number of clusters, enabling exploration of the dataset. However, the re-clustering process may change all existing clusters and make it difficult for the user to keep track. To address this issue, MotionFlow allows the user to select the clusters to be locked or unchanged during the re-clustering process. He or she is also able to interactively merge or split clusters according to his or her own assessment.

Similarly, to provide an overall understanding of human motion tracking data, MotionExplorer [9] applies a hierarchical clustering method [57], which seeks to build a hierarchy of clusters. MotionExplorer takes a divisive approach considering all data items starting within the same cluster and splitting them until a termination condition is met. One of the important decisions in this clustering technique is to determine which cluster to split next. The user is allowed to choose among several splitting strategies such as *maximum standard deviation* to split the most varied cluster first and *highest number of elements* to split the largest cluster first. The hierarchy is visualized as a dendrogram as in Figure 1.20. Clusters are obtained by cutting the dendrogram at the desired vertical level: each connected branch forms a cluster. The vertical axis of the dendrogram can encode different variables depending on the splitting strategy. In Figure 1.20, it shows the standard deviation of each cluster and the user is allowed to slide the cutting value to adjust the resulting clusters.

A different approach in hierarchical clustering is bottom-up or agglomerative clustering. The process starts with singleton clusters before merging the most similar clusters together until a termination condition is met. NewsLab [49] applies such an approach in analysis of large scale broadcast news video collections. It builds a hierarchy of clusters over all keywords extracted from available video captions based on their co-occurrences in the news stories. Therefore, strongly correlated keywords are grouped into the same clusters whereas loosely related keywords are separated in different clusters. Each cluster is visualized as a stream showing the evolution of the keywords within the cluster over time with closely related clusters placed close to each other to allow navigation to different depths of the cluster hierarchy (Figure 1.21).

Understanding people movement patterns in both space and time plays an important role in urban planing. Analysis and presentation of large and complex datasets that contain a number of people in different places and their movement between

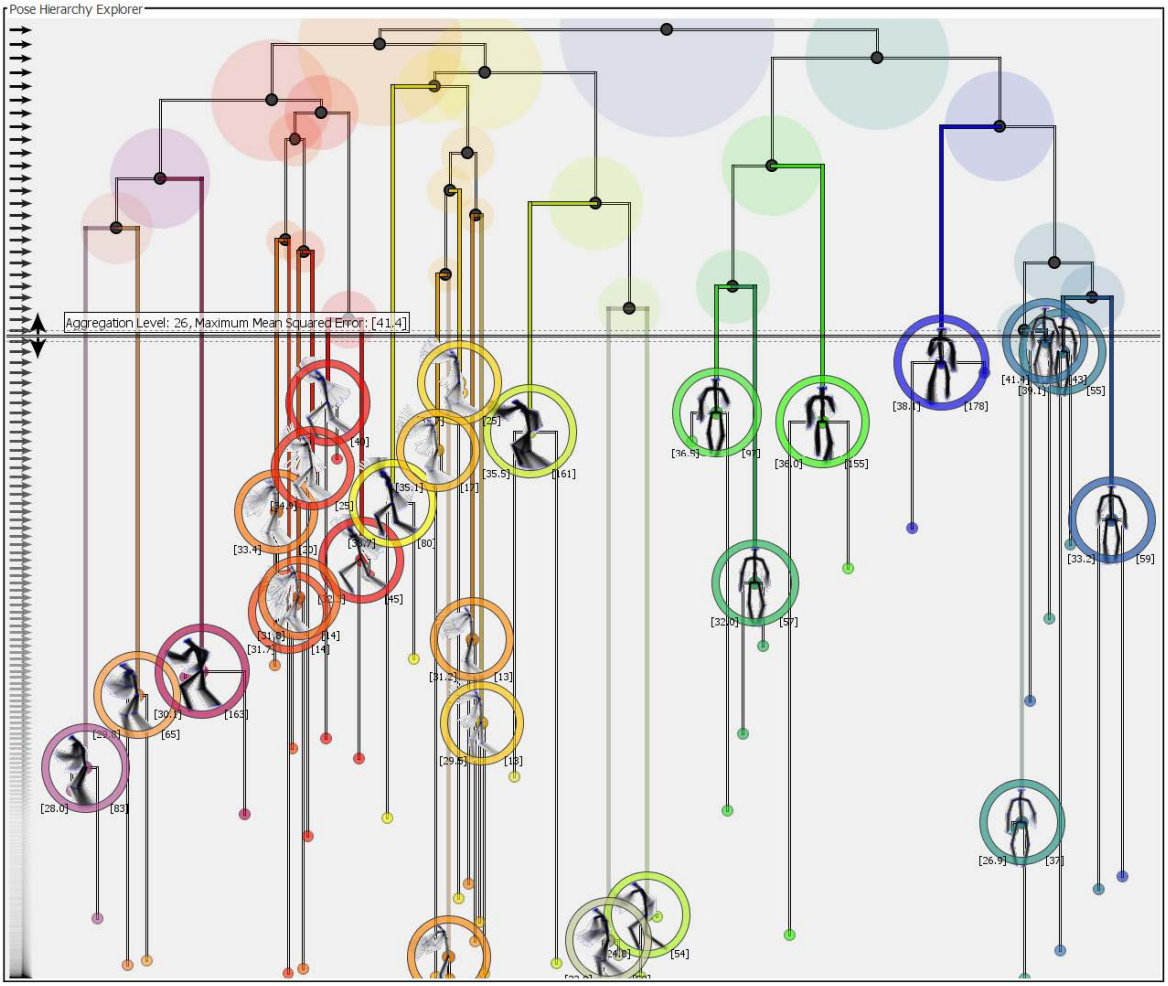


Figure 1.20: A divisive hierarchical clustering of human poses visualized as a dendrogram. Clusters are connected branches after the dendrogram is cut at the desired vertical level. *Image source: [9].*

those places over time are challenging. Mobility Graphs [93] applies cluster analysis to simplify the data in both spatial and temporal dimensions to gain an overall understanding of the datasets. First, it aggregates places using a density-based clustering technique that considers both the density of places and their flow magnitudes so that close and highly connected can be grouped together. Then, a temporal aggregation algorithm groups the time steps by the similarity of those simplified places using a k-means clustering technique. Figure 1.22 shows 7 temporal clusters of simplified places. The clusters are color coded with a calendar view to reveal temporal patterns over a week. In each cluster, a node shows an aggregated place with size corresponding to the total number of people in all individual places and

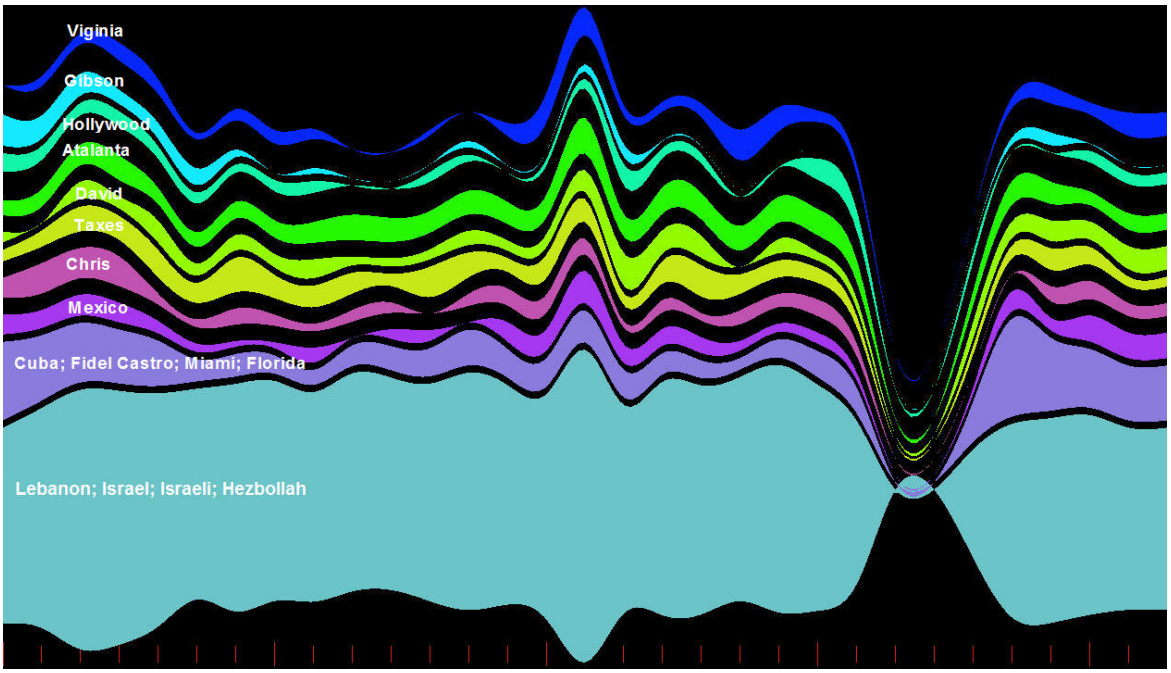


Figure 1.21: An agglomerative hierarchical clustering of video captions visualized as streams. Each cluster is a colored stream, with related clustered placed close together. *Image source: [49].*

arrow widths representing the number of people moving between two aggregated places.

1.2.3.2 Classification

Overview 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 [57]. 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 [124]. Each non-leaf node of the tree 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

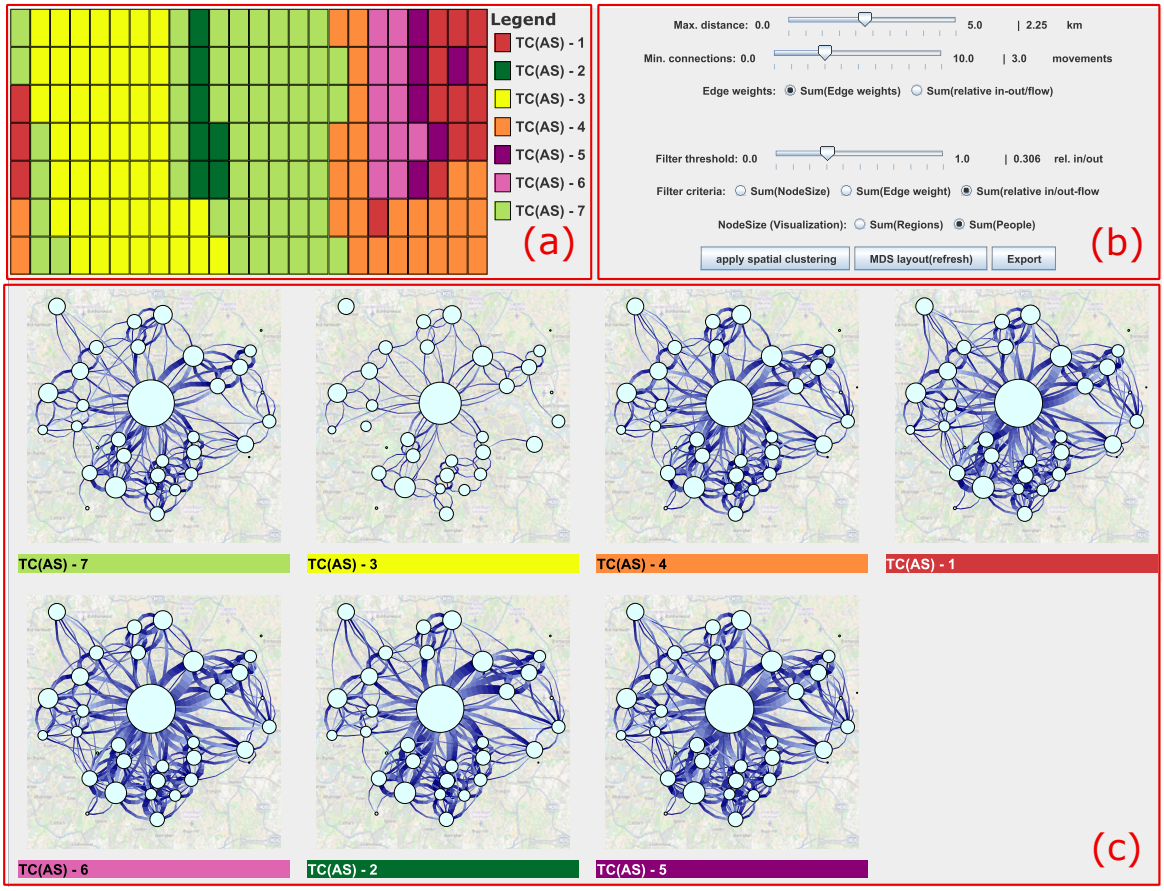


Figure 1.22: Spatial and temporal aggregation of movement data. Seven temporal clusters of simplified places are shown and color coded together with a calendar view to reveal daily pattern. *Image source: [93].*

dataset with 10 records, 5 labeled "true" and 5 labeled "false". Attribute $A1$ splits the set to two subsets: (5 "true", 0 "false") and (0 "true", 5 "false"). Attribute $A2$ splits the set to (3 "true", 2 "false") and (2 "true", 3 "false"). The subsets split by $A1$ is "purer" than the one by $A2$ 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* [141]. More detailed analysis of these measurements and other classification algorithms are out of the scope of this thesis and can be found in data mining textbooks [57, 141].

Application Examples Exploring a large image collection is challenging. Besides the low-level visual features, the semantic contents of images are also effective in searching for relevant ones. Image classification techniques can be used to extract such semantic contents. For example, Fan et al. [40] detect salient objects in images

and associate them with predefined semantic contents according to their perceptual properties. Similar contents are then grouped into a higher level semantic concept; for instance, "sand field", "sea water" and "boat" salient objects construct the concept of "sea world". Visualization can make the output of classification algorithms more interpretable and interactive. To provide an overview of an image collection, Yang et al. [167] shows the extracted semantic contents, with each as a glyph, in a 2D display so that related contents are located close together using a multidimensional dimension scaling method [10]. Similarly, images are also displayed based on their similarity as in Figure 1.23a. Zooming and panning are provided to make the visualization more scalable. When an image is selected, the visualization can be switched to a *rainfall* mode, in which the selected image is shown at the bottom and related images are stacked above it based on their similarity with the selected one (Figure 1.23b). Users are allowed to reassign the contents computed for each image; however, the model does not take into account the changes to improve its accuracy when classifying new images.

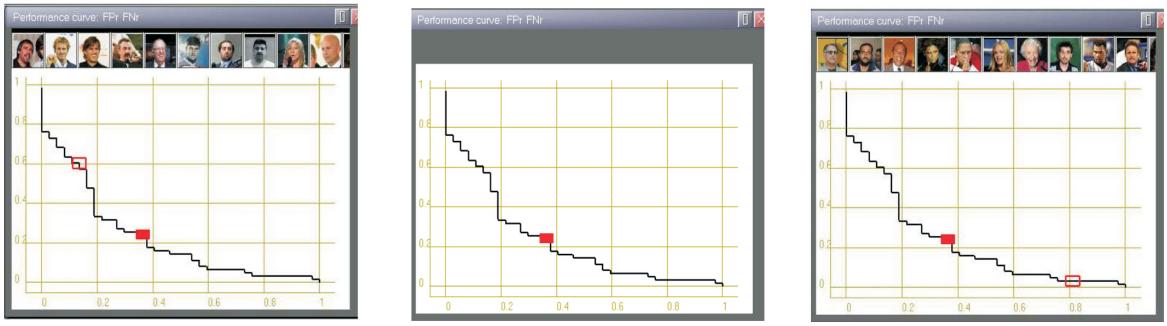
(a) Multidimensional dimension scaling view of images.

(b) Rainfall view of a selected image with highly related images at the bottom.

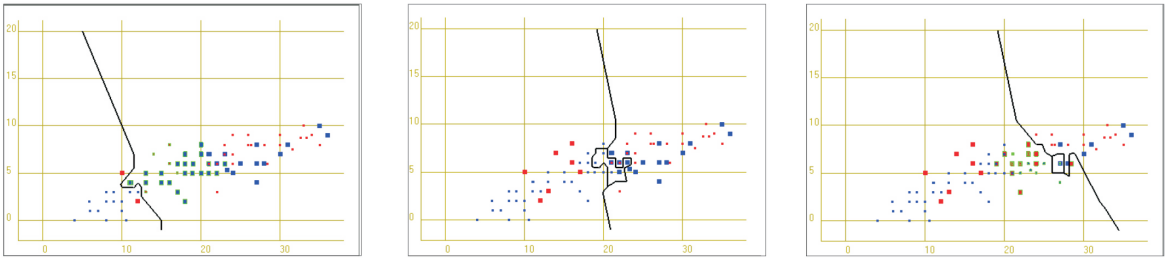
Figure 1.23: Large-scale image browser with semantic content-based image classification. *Image source: [167].*

In a binary classification, the classifier output is either *positive* or *negative*. Two types of error can happen including *false positive* (classified as positive but the actual class is negative) and *false negative* (classified as negative but the actual class is

positive). Depending on a particular domain, the costs of these error types might be considerably different. For example, wrong prediction of a healthy patient with a cancer has a much lower impact than missing a patient with a real cancer. Migut and Worring [104] allow users to adjust the trade-off between these two error types through a visualization of the classification model as shown in Figure 1.24.



(a) Performance curve of the classification model with horizontal axis showing the false positive rate and vertical axis showing the false negative rate.



(b) Data with color indicating original class and size showing classification accuracy.

Figure 1.24: Visualization and interaction of a classification model. Middle figures show the initial state of the system, with initial operating point on the performance curve and corresponding data scatter plot. Left figures show the state of the system when an expert manipulates the operating point to include more false negatives and on the right to include more false positives. *Image source: [104].*

Typically, a receiver operating characteristic curve [41] is used to illustrate the performance of a binary classifier when its discrimination threshold is varied. The curve is composed from a set of true positive rate and false positive rate pairs at various threshold values. Migut and Worring replace the true positive rate with the false negative rate (Figure 1.24a) because their focus is comparing trade-off between the two error rates. Numerical data is visualized in a scatter plot with the decision boundary separating a 2D plane into two regions, each for a class (Figure 1.24b). For each data point, color shows the original class and size indicates the accuracy of the classified class. The current classification setting is shown as a red point on the performance curve, and the user is allowed to move that point along the curve to

change the false positive and false negative rates. The classification reruns with the new threshold and rates and updates on the data scatter plot.

Classification requires training data; however, it can be time-consuming and laborious to produce such a dataset. ScatterBlogs2 [11] includes an interactive classifier that speeds up the training data labeling and classifier construction before applying it in real-time monitoring messages of interest. First, the user can search for relevant messages using a standard keyword query. The system then highlights non-trivial terms that frequently co-occur with the original keywords. The result set of highly relevant messages can be used as *positive* samples, whereas some arbitrary messages not returned in the result set can be used as *negative* ones. After creating an initial classifier, the user can inspect messages to correct and update the classifier through the message visualization. Messages are shown in a map as a colored glyph with color hue indicating class and brightness showing classification confidence (Figure 1.25). They can be filtered by confidence, allowing the user to focus on ones with less certainty, which need human expert to verify.

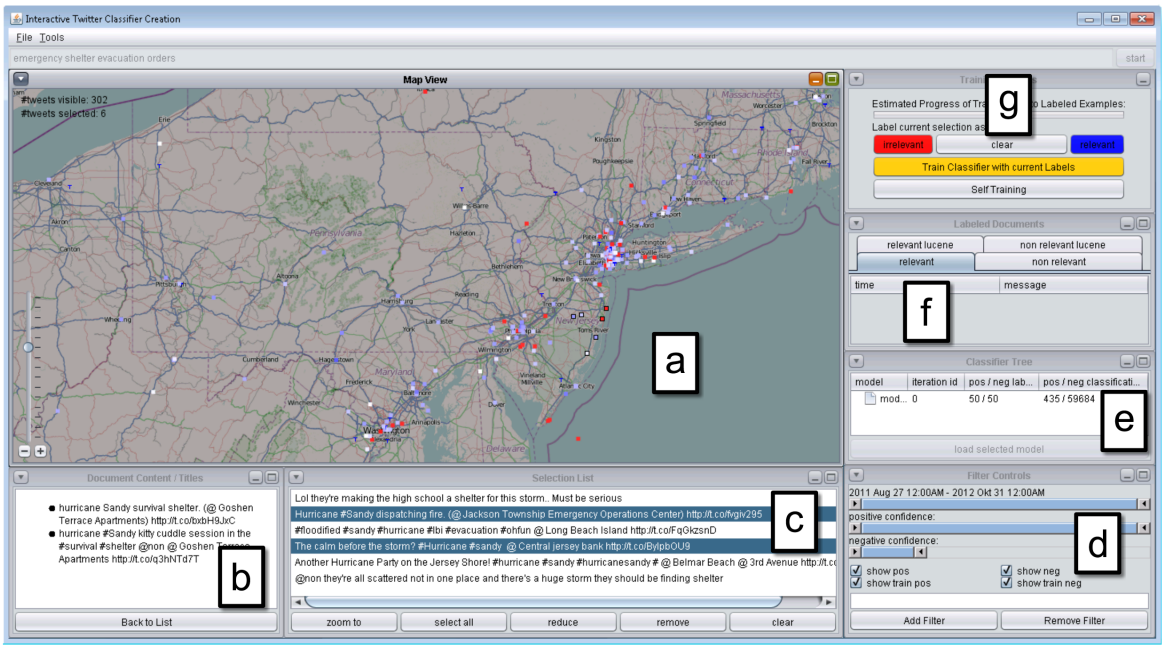


Figure 1.25: ScatterBlogs2 speeding up the creation of a classifier through interactive visualization. *Image source: [11].*

An essential step in classification of high dimensional datasets is feature selection, which selects a subset of relevant features for use in model construction without much loss of information. This step also simplifies the model and reduces training time. INFUSE [89] supports users to explore the predictive power of features in

their models. The system allows comparison of features across four feature selection algorithms. Each feature is shown as a circle glyph divided into four equal quadrants, each for an algorithm (Figure 1.26). A quadrant is further split into 10 slices, each for a cross-validation fold (or random subset of data) to ensure the result robust. The length of a slice indicates the rank of that feature using a given algorithm. Therefore, a glyph can show how its feature performs in different algorithms. To provide an overview of all features, INFUSE shows multiple glyphs in either a sequential layout or a scatter plot, where different options can be used for axes such as average rank of a feature or a more sophisticated importance measurement. It also allows users to explore four classification algorithms by showing the score of all 16 combination of feature selection and classification algorithms. More importantly, users are allowed to build their own model by selecting features besides the ones produced by the four given algorithms. The custom feature set is then included in the classification score comparison.

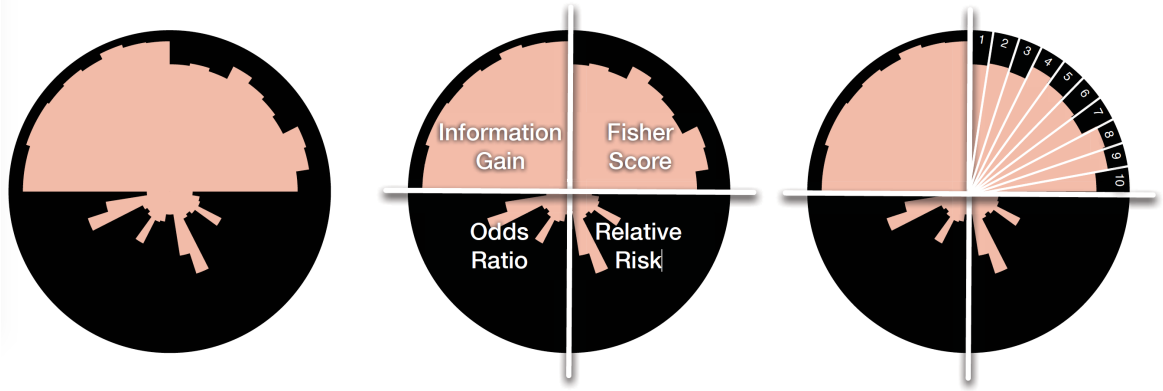


Figure 1.26: INFUSE allowing comparison of feature selection and classification algorithms. The circle glyph is divided into 4 quadrants, each for a feature selection algorithm. Each quadrant is further split into 10 slices, each for a cross-validation fold. *Image source: [89].*

1.2.4 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 [119]. Excellent reviews of visualization evaluation are available with different perspectives such as evaluation techniques [21], scenarios [92] and design process [109].

In this section, we review the evaluation techniques based on the visualization design model by Munzner [109], helping address different concerns separately. The model consists of four levels including explaining the tasks and available data in the vocabulary of the problem domain, abstracting them into domain-independent operations and data types, designing visual encoding and interaction techniques to solve the abstract tasks, and developing 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* approaches requires all inner levels are completed. The threats and evaluation methods for all levels are summarized in Figure 1.27.

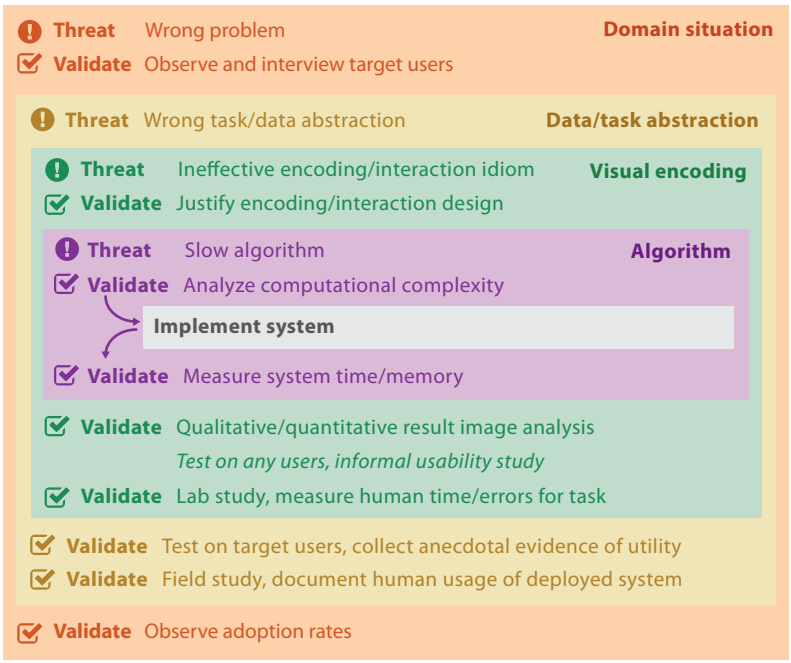


Figure 1.27: Threats and validation at each of the four nested levels of visualization design. *Image source: [110].*

1.2.4.1 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* [21], 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 [72], which allows the investigator to occasionally interview while the user is engaged in the process. One example is the study by Sedlmair et al. [133] on current working behavior and environments of automotive analysis and diagnosis experts.

One downstream form of validation is to report 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 [129] and 6-week field trial of SparTag.us – a tagging system for foraging web content [73].

1.2.4.2 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 [102].

Evaluating visualization 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. [92]. Due to the nature of sensemaking, evaluation is often carried out as case studies [79] with observation and interview, and followed by qualitative data analysis [94]. Attempts also have been made to quantify the insight or knowledge gained during sensemaking [163].

1.2.4.3 Visual Encoding and Interaction Design

At this 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 1.2.2.1, or more comprehensive predefined guidelines as in heuristic evaluation [172]. Asking experts to review the design prototype also provides valuable feedback [147].

A common downstream approach is to conduct a controlled experiment comparing the design with other state-of-the-art alternatives [166]. 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 [42]. 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 [62]. 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 [140].

1.2.4.4 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 [27] 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.

1.2.5 Summary

Visualization helps people carry out tasks more effectively by amplifying human cognition through visual representation and interaction. Visual analytics includes automated analysis techniques to help make sense of complex datasets. This thesis contributes novel visualization techniques and systems to support the sensemaking process. The visualizations are designed based on a number of principles and guidelines described in this chapter such as Gestalt laws, color mapping and fluid interaction. The designs are implemented and evaluated rigorously with suitable methods as discussed previously. The visualizations make use of the provenance data captured during the sensemaking process. Next, we will discuss the literature on capture and visualization of provenance data.

1.3 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 everyday shopping, before purchasing a bottle of fruit juice, a customer would like to know about its origin, ingredients, methods

of collecting, storing and processing fruits, etc. In 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" [108]. It contains information about the input data, the output data, and the configuration of the program used to process the data.

Provenance can be broadly categorized into two groups. The first group, *data provenance*, relies on the previous definition of provenance in computer systems, focusing on the derivation history of data including its source information and the process that produced it. This type of provenance is often emphasized in data-intensive fields such as scientific workflows and databases. The second group, *analytic provenance*, is usually mentioned in the context of visualization and sense-making. It focuses on the interactive data exploration and the reasoning process driven by sensemaking including the provenance of visualizations used, interactions performed, analytical insights found and the conclusion and rationale behind them. Next, we discuss data provenance in different fields that are active in provenance research, including scientific workflows, databases and semantic webs. Then, we will follow with the review of analytic provenance.

1.3.1 Data Provenance

Scientific experiments may consist of thousands of steps, with each step involving distributed data sources and computational data models [50]. Workflows have been used to facilitate the assembly, automation and management of such experiments. Notable scientific workflow systems with provenance enabled include Tarvena [170], Kepler [12] and VisTrails [8]. Provenance plays an important role in scientific workflows, aiming to support data interpretation, reproduction of experiment results, troubleshooting and optimization [105]. Zhao, Wilde and Foster [171] discuss two types of provenance in workflows: *prospective provenance* – focusing on the workflow design, and *retrospective provenance* – focusing on the workflow execution. The provenance of long and complex workflows is huge, thus pose challenges in storing, querying, and making sense of such data [30].

Curated databases are populated and updated with a great deal of human effort, typically published on the web [15]. A well-known example is Wikipedia – a free Internet encyclopedia that allows its users to edit almost all articles. Each record in these curated 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 [22]. *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 produces t [29]. *How*-provenance concerns how the output tuple t is derived from the query [54]. Finally, *where*-provenance describes specific locations, or cells in relational databases, of the input data that contribute to the query output [16]. To compute these types of provenance, two general approaches have been introduced [15]. 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 so far has mainly focused on closed systems, which have full access to the data and its provenance. Modern applications with service-oriented and cloud-computing architectures bring challenges in tracking and exchanging provenance information across systems. The *Open Provenance Model* is designed to address these challenges [108]. 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 the process, managing its execution. Different types of causal relationships can be added between these nodes, forming a *provenance graph* as shown in Figure 1.28.

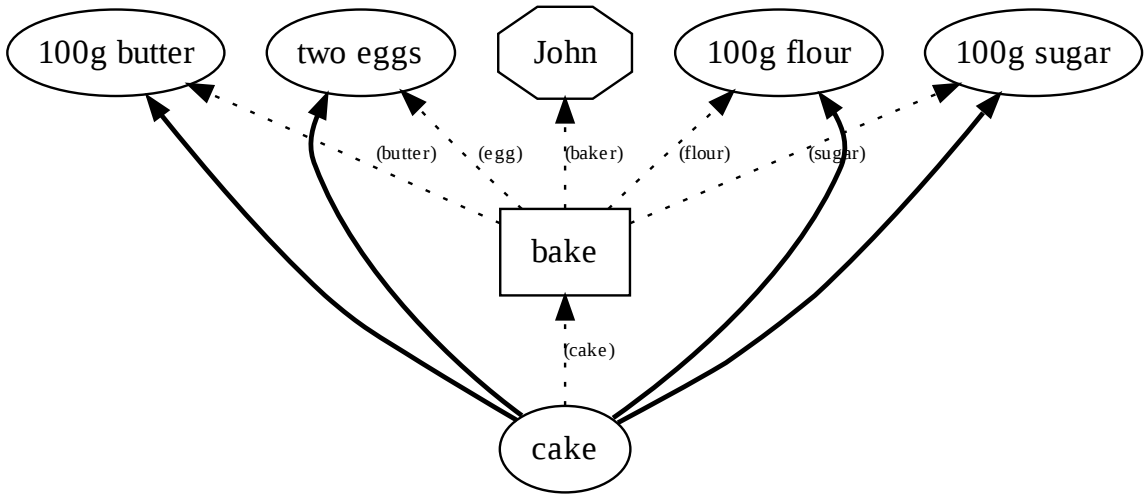


Figure 1.28: A provenance graph for "cake baking" using the Open Provenance Model. The cake (artifact) was baked (the process) by John (the agent) using ingredients (artifacts) including butter, eggs, flour and sugar. *Image source: [108].*

The Open Provenance Model has been implemented in many different scientific workflow systems such as Tarvena [170], Kepler [12] and VisTrails [8]. The model is general and can be extended in both the structure and vocabulary to represent domain-specific problems. *D-profile* [55] describes an extension of the model for representing provenance in distributed systems. ProveML [155] is an extension for recording the provenance of data, analytical process and interpretation in human terrain visual analytics.

The *PROV* set of specifications, produced by the World Wide Web Consortium, is designed to promote the publication of provenance information on the Web and support the interchange of provenance across diverse provenance management systems [107]. The PROV-DM is the conceptual data model that forms a basis of PROV and is based on the aforementioned open provenance model. Besides the core data model, PROV also includes other extensions such as PROV-CONSTRAINTS, a set of constraints applied to the data model to impose rules for consistency and validity of provenance.

1.3.2 Analytic Provenance

Analytic provenance focuses on understanding the sensemaking process of a user through the study of his or her interaction with a visualization [113]. It captures both the interactive data exploration process and the accompanied human reasoning process during sensemaking [165]. This section discusses models for representing analytic provenance data, methods to capture the data and techniques to visualize the captured data for supporting sensemaking.

1.3.2.1 Model

Analytic provenance information can be categorized using a four-layer hierarchical model based on its semantic richness proposed by Gotz and Zhou [53]. Figure 1.29 illustrates this model with the level of semantics increasing from bottom to top. The bottom-level *events* consist of low-level user interaction such as mouse clicks and keystrokes, which contain little semantic meaning. The next level up includes *actions*, which are analytic steps such as querying the database or changing the zooming level of a 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 long term trends.

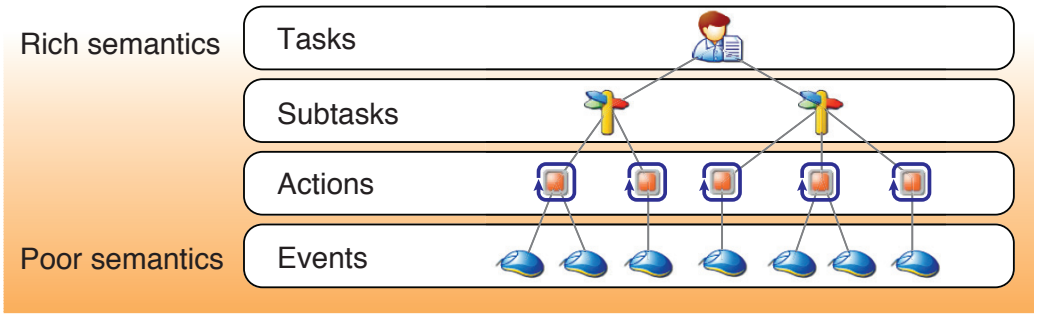


Figure 1.29: The hierarchical analytic provenance model with semantic richness increasing from bottom to top. The bottom layer includes *events* such as key presses and mouse clicks, which have little semantics. The next level up contains *actions* such as the database query and visualization zooming. 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: [53].*

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 elements in each layer is performed to serve for a higher semantic element in the above layer. For example, a database query action consists of several mouse click and key stroke events, and it can be 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 elements they want to capture within each layer for their systems. For example, many existing taxonomies of visualization interaction and tasks [4, 53] can be used to guide the selection in *action* and *sub-task* layers. Low-level analytic activities such as "retrieve value", "sort" and "filter" 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 [168]. A multi-level typology of abstract visualization tasks by Brehmer and Munzner [13] distinguishes why and how a task is performed, as well as what the task input and output are. The *how* part of this typology can be a good candidate for representing the *action* layer, and the *why* part can be suitable for the *sub-task* layer.

A recent taxonomy by Ragan et al. [125] 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 the previous discussion in Section 1.3.1. *Visualization* provenance concerns with the history of graphical views and visualization states, and *interaction* provenance focuses on the history of user actions within a system. The combination of these two types of provenance is equivalent to the *action* layer in Gotz and Zhou's model. *Insight* provenance describes the findings and *rationale* provenance explains the reasoning process that led to these findings. These two types of provenance can be referred to the *sub-task* layer.

The following sections categorize analytic provenance research into four layers of the Gotz and Zhou's model, focusing on the capture and visualization techniques for each layer.

1.3.2.2 Events

Capture There is limited literature on capturing events because it is relatively easy and provides little semantics. A notable example is Glass Box [28], which 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 for later retrieval. Simply capturing these events alone is insufficient to understand their purpose and rationale. For example, even though we know that a *mouse click* happened; 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. In data exploration with a visualization, an analyst often performs many operations and takes different approaches in order to find interesting patterns. 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 are required to be captured together.

Visualization Because low-level events contain little semantics, they may not be able to immediately provide meaningful insight and near real-time support to the users during their sensemaking processes. However, in a post hoc analysis, they can be used to gain a deep understanding about the processes happened. For example, a visualization of users' mouse clicks can reveal patterns of application usage [101] and highlight important usability issues, such as pages where users spent a lot of time and pages where they got lost during the task [158]. User interaction with visual analytics systems can be visually examined to recover their reasoning processes

employed in their analysis tasks such as specific findings and strategies they used to discover them [36, 56]. Interaction logs have also been used to predict user performance of basic visualization tasks like visual search [14].

1.3.2.3 Actions

Capture During the data exploration with a visualization, all user interaction 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, data attributes mapped to visual channels such as spatial positions, color and size. An action could be changing the size mapping. Commonly, undo and 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 actions.

Two common strategies can be applied to automatically capture the exploration process. The first strategy captures the initial state and all the actions so that they can be executed to reproduce all states [78]. This allows reapplication of the analysis process with a different dataset, but could be time-inefficient if the number of actions is high. The second strategy simply captures the visualization state after every action [8]. 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. These browsers often capture data about visited web pages 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 [7, 70, 106]. This results in a hierarchical history rather than a linear list of visited web pages in browser's history feature. Manual capture through bookmarking is also a standard feature of web browsers, allowing users to save web pages for revisitation purpose. Besides bookmarking a whole web page through its URL, a page element such as *table* and *form* HTML tags [73], and a specific fragment of text [35] can also be saved. These more fine-grained captures allow users to record what they want with a higher accuracy.

Visualization We discuss different visual representations of actions and states, and different layouts to position them.

Visual Representation The simplest representation is **text**, commonly used to describe actions and 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 [7] shortens web page titles to accommodate more pages in the view. Within a given width, its truncating method preserves complete words at both ends of the title and trims characters in the middle if necessary. Kaasten, Greenberg and Edwards [77] 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 a medium (60%) recognition, 15 – 20 letters (depending on the truncation method) are required for recognizing web sites, and 28 – 39 letters are required for recognizing exact pages. For longer text such as user notes, machine learning techniques for text summarization could be beneficial [111].

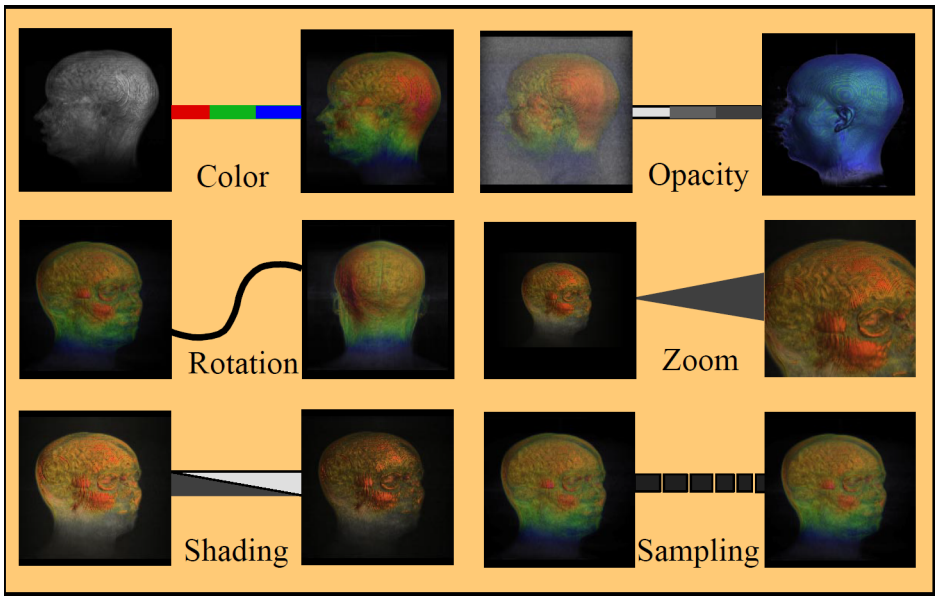
Icon is another popular choice of visual representation for actions and states, enabling users to easily distinguish them. They can be used alone to represent the analysis process when the visual result of each action is out of interest (Figure 1.30a). Alternatively, these icons can be used together with visualization states, connecting the one before and the one after an action (Figure 1.30b).

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 [77]. 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 [25]. For visualization thumbnails, visual encodings and parameters that were used to produce the visualization can also be embedded (Figure 1.31), providing more provenance information to the users.

It may be necessary to apply pre- and post-processing adjustments to make the visualization thumbnails more recognizable [64]. 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 recognize the visualizations they saw 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. [143] propose an automatic method

(a) A sequence of icons representing the analysis process. *Image source: [53].*



(b) Iconic actions connecting two visualization states. *Image source: [99].*

Figure 1.30: Icons as visual representation of actions.

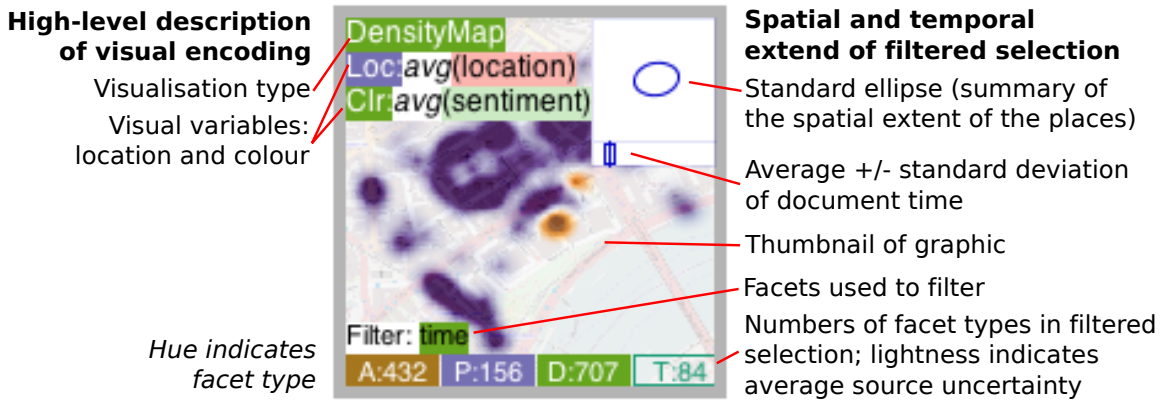


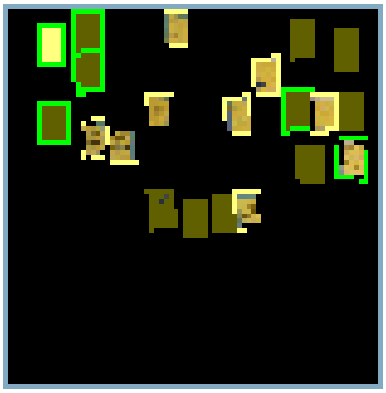
Figure 1.31: Visualization thumbnails with additional information about the filtering used, the characteristics of the filtered subset of data and the visual encoding. *Image source: [155].*

to produce a thumbnail improving its recognition. The thumbnail consists of three components: 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

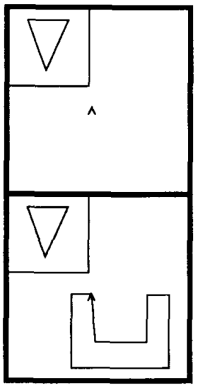
image and the branding logo are chosen from the page using machine learning. Figure 1.32 shows such a thumbnail.

Figure 1.32: An enhanced web thumbnail (right) as a composite of salient text, a salient image and a branding logo. *Image source: [143].*

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 1.33a). The changes may only happen at a small portion of the entire interface. Therefore, showing only that area in both states could help users quickly identify the difference (Figure 1.33b).



(a) Modified items are highlighted with green borders. *Image source: [87].*



(b) Changed area is focused and shown in both states. *Image source: [91].*

Figure 1.33: Techniques improving recognition of state changes.

Layout Linear and branching layouts are commonly used to position actions and states, depending on the structure of data capture.

Linear Layout — Provenance data usually contains an inherent *time* attribute, recording when an action happened. An approach that emphasizes on the order of actions is to show them as a linear sequence of items like a comic strip [91, 103]. This layout facilitates visual scanning of past actions, allowing users to quickly understand the analysis process. Figure 1.34 shows such a layout.

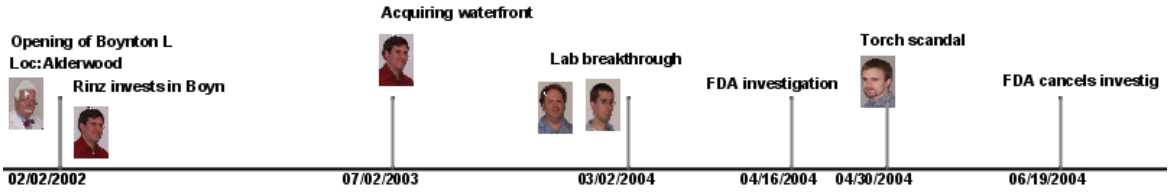
Figure 1.34: Comic strip layout. A sequence of past actions in chronological order. *Image source: [64].*

If the absolute timestamps of actions are of interest, a continuous timeline [31] is a more suitable layout. Actions are positioned along the time axis at when they happen (Figure 1.35a) or during which they last (Figure 1.35b). The time axis can be either horizontal or vertical [139]. The layout algorithms found in these provenance timelines are quite naive. POLESTAR [117] and the timeline from Jigsaw [96] require manual rearrangement of the events from users to solve the overlapping problem. The timeline from nSpace2 Sandbox [139] shows events at the exact time when they happen without considering possible intersection between of them.

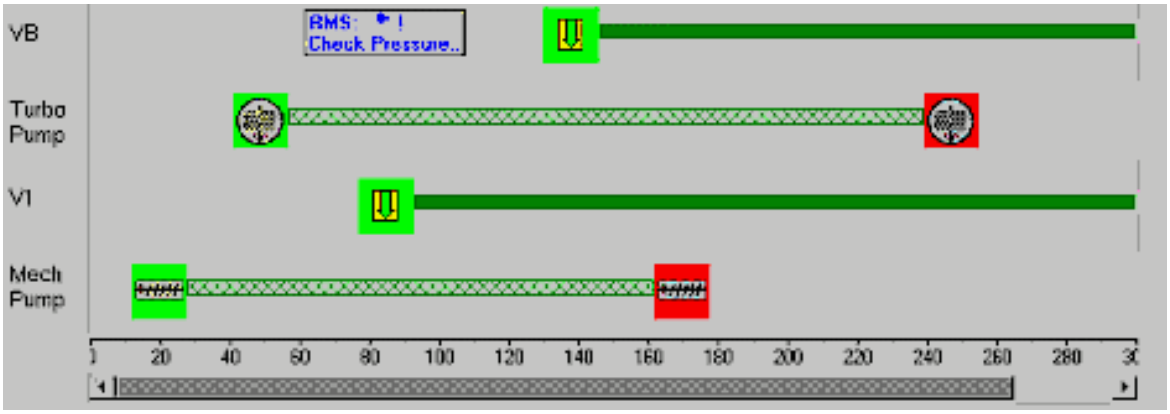
Another approach is to use both the horizontal and vertical axes to represent time. BrowseLine [71] uses the vertical axis for *macro-time* and the horizontal axis for *micro-time*, similar to stem-and-leaf plots. More specifically, a two-dimensional timeline (Figure 1.36) is divided into rows, each for a macro 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.

Visualization techniques of general time-oriented data can be beneficial to the design of temporal visualization of provenance data and are discussed in Section 1.4.

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



(a) Time-point provenance data. User annotations are positioned along a time axis at when their associated events happen. *Image source: [52].*



(b) Interval provenance data. User actions are shown as horizontal bars a long a time axis covering their durations. *Image source: [122].*

Figure 1.35: Timeline layout for visualizing the analysis process.

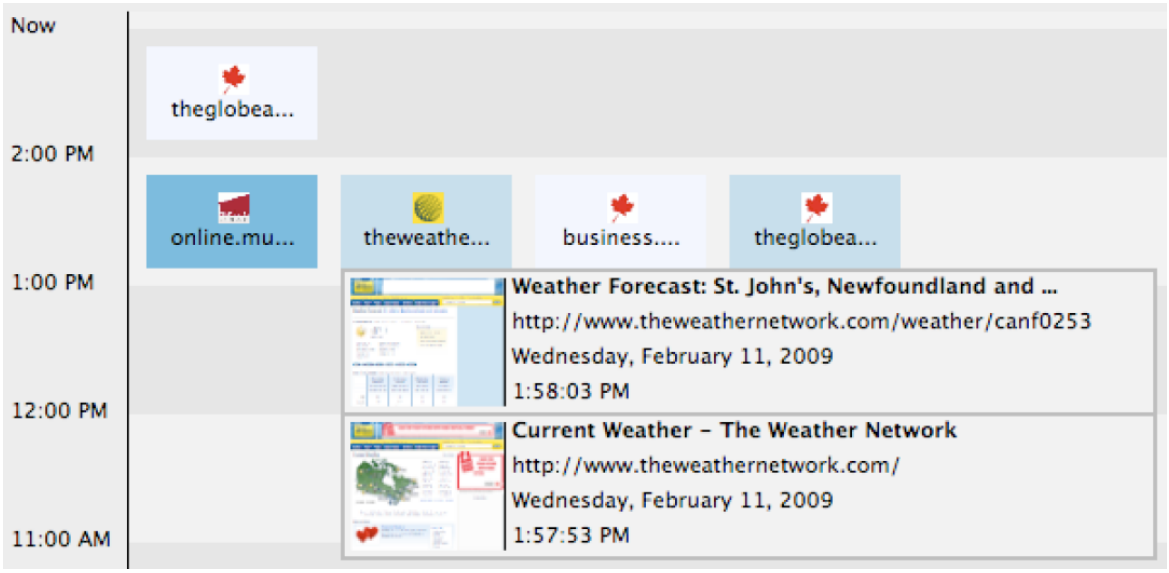


Figure 1.36: 2D timeline. The vertical axis represents macro-time, whereas the horizontal axis represents micro-time. Events within a macro time-slot are positioned in chronological order as a comic strip. *Image source: [71].*

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 many provenance-enabled systems in different fields including scientific visualization [99], information visualization [37], visual analytics [78] and browser history [7]. Figure 1.37 shows such a provenance tree in a scientific visualization.

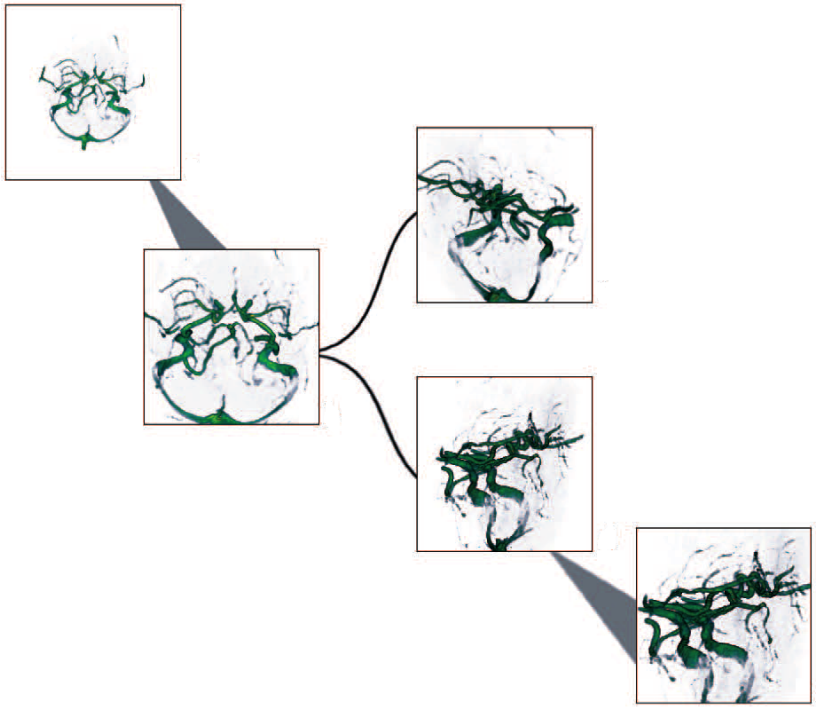
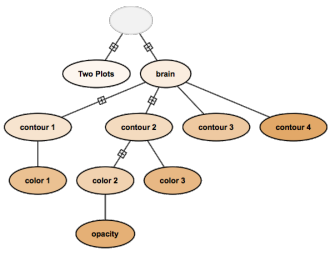


Figure 1.37: Tree visualization for branching analysis process. Nodes are thumbnails of past visualization states and links are transforming actions. *Image source: [75].*

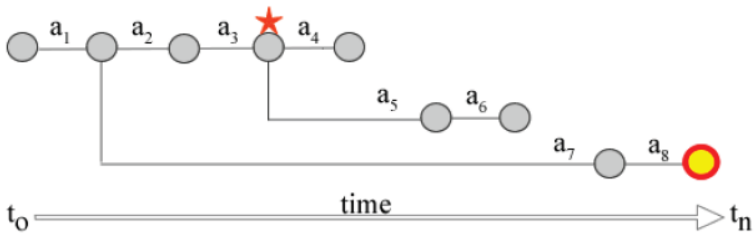
In a tree layout, the order of actions can be inferred through the direction of edges. Moreover, exact time gap between actions can also be visually encoded into the visualization. VisTrails [8] color-codes the background of nodes according to their creation time (Figure 1.38a). Aruvi [137] uses the length of edges to represent the relative time gap between two states (Figure 1.38b).

The diagonal arrangement of tree as in Figure 1.38a may consume a lot of space. One approach is to use only horizontal and vertical edges as in Figure 1.38b. Another approach is to display only the path that led to the currently active visualization [87], and make other paths expanded on demand. Figure 1.39 shows such an example.

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

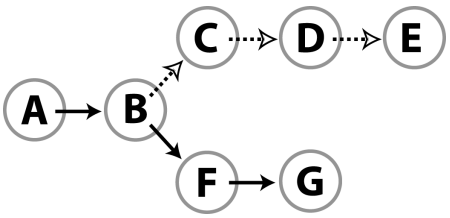


(a) Node backgrounds are color coded based on time. *Image source: [151].*

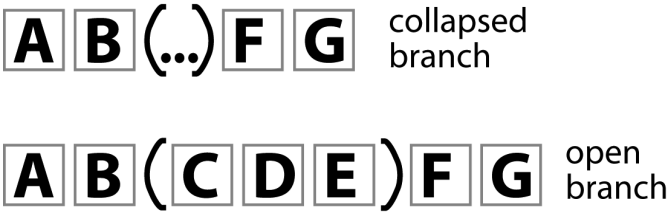


(b) The edge length between two nodes represents the interval between them. *Image source: [137].*

Figure 1.38: Encoding temporal information into provenance trees.



(a) Branched history. The user first performed actions A, B, C, D and E, then undone to D, C and B, then performed new actions F and G.



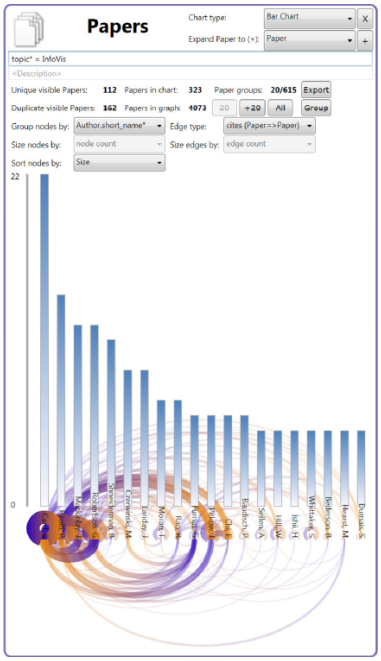
(b) Inline branching history representation. Top: displaying the current path. Bottom: displaying the full history; actions not part of the current path are placed between brackets.

Figure 1.39: Branching layout for tree visualization of the analysis process. *Image source: [87].*

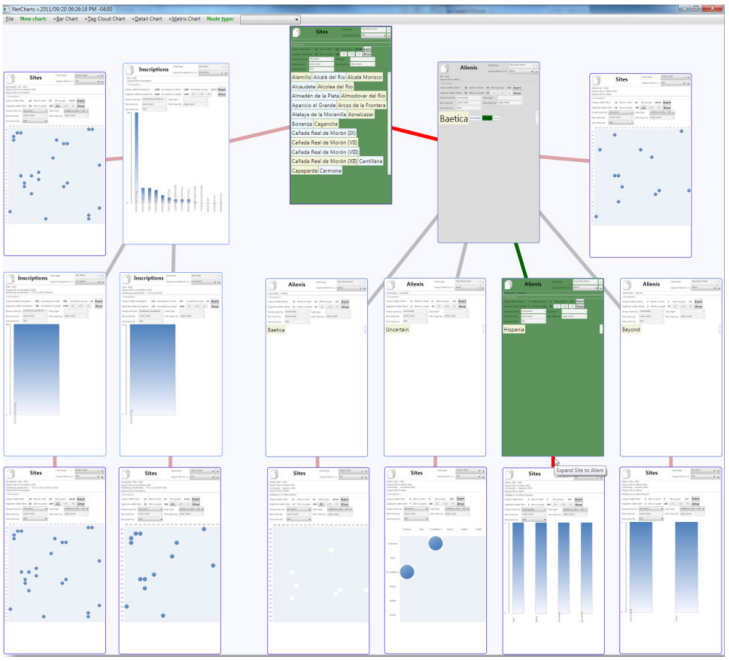
part of interest [37]. Collapsing and expanding branches of the tree on demand can also help reduce its size and avoid distraction [8]. Distortion techniques may help users quickly navigate and focus on more relevant states and actions [103]. Another technique for compressing the provenance tree and improving user understanding is *action chunking* [64]: a sequence of related actions may be better represented as a single higher-level activity. 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 [64, 78, 115, 137]. However the system can also be used as a single item of the provenance view. For instance, in GraphTrail [37], 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. Through zooming and panning, users can switch between

a close examination of individual system states (Figure 1.40a) and an overview of the analysis structure (Figure 1.40b). An extra overview window as in overview-and-detail technique [26] 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 1.40: Unification of the exploration view and the history view. *Image source: [37].*

Visualization techniques of general network-oriented data can be beneficial to the design of a branching layout for provenance data and are discussed in Section 1.5.

1.3.2.4 Sub-Tasks and Tasks

Tasks and sub-tasks provide important clues to the purpose and rationale underlying the sensemaking 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.

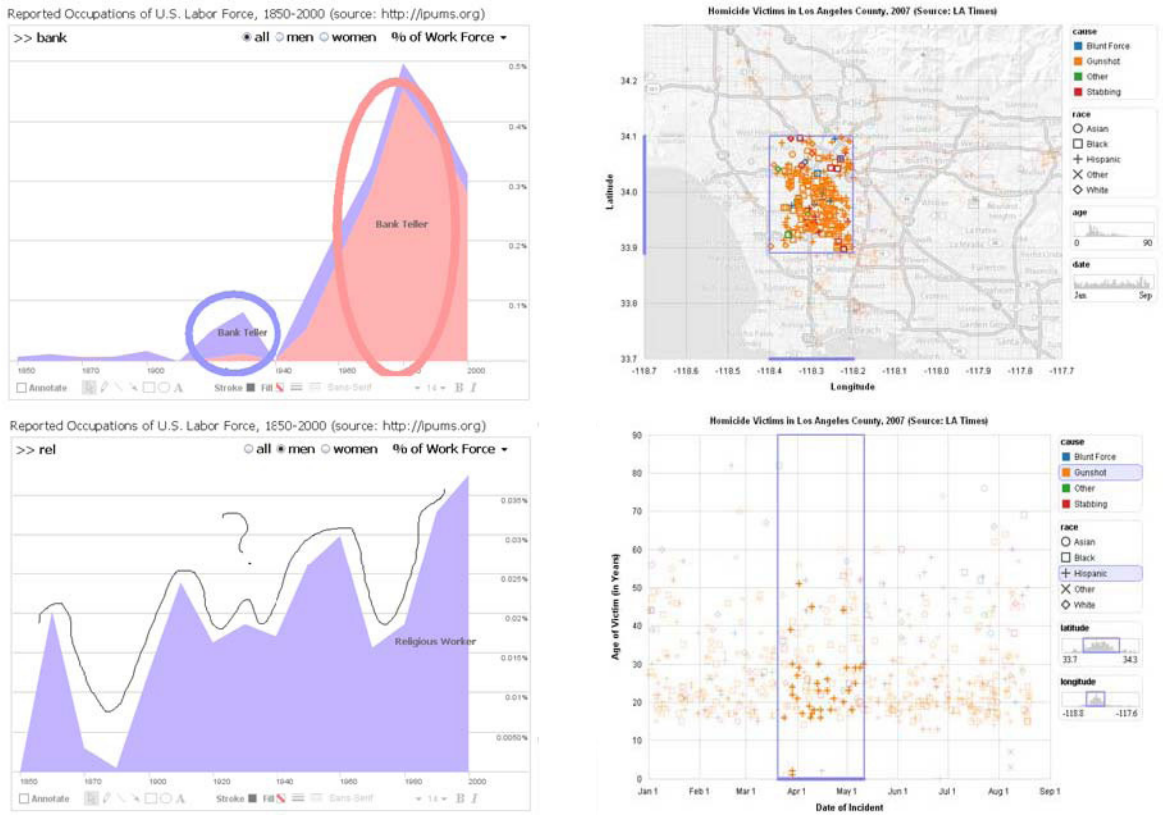
Existing approaches to capture high-level analytic provenance can be broadly categorized into manual and automatic methods. The *manual* methods rely on users recording their analysis processes and sensemaking tasks, whereas the *automatic*

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 [53]. 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 can vary significantly from users to users, even with the same dataset and analysis task.

Manual Tasks and sub-tasks can be captured by allowing the user to externalize them manually. A common form of reasoning externalization is to allow users to write down their thinking. It could be a free note [137] or a description of a user bookmarked visualization [66, 155]. Alternatively, users can annotate directly on the visual bookmark using standard graphical editing features such as circling on interesting patterns (Figure 1.41a – Top) or hand drawing on a suspicious trend (Figure 1.41a – Bottom). To make the annotation more meaningful across multiple views, the selection should be aware of the data involved in it [61]. For example, in Figure 1.41b, 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.

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* [115] allows users to record their discoveries of interesting patterns about the data, then specify their causal relationships by drawing links, and generate hypotheses based on these artifacts. 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. These interactive features are known to help users produce more critical thinking in their analyses [132]. Figure 1.42a shows such a diagram produced with user notes and Figure 1.42b shows such a more formal diagram with different reasoning roles.

More analytical reasoning methods have also been integrated to visual analytics systems to allow users to perform complex analyses with their externalized thoughts.

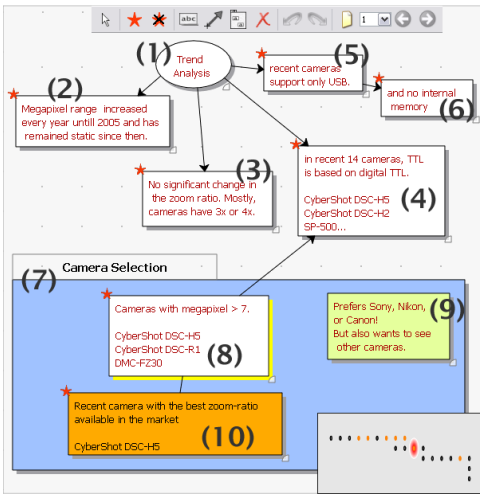


(a) Geometric annotation. **(b)** Data-aware annotation.

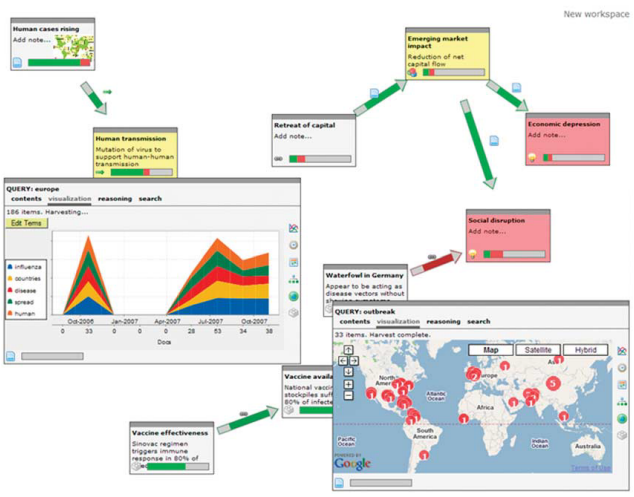
Figure 1.41: User annotation of bookmarked visualization as a form of manual capture of tasks and sub-tasks. *Image source: [65].*

POLESTAR [117] supports argument structuring, based on methods for analyzing legal documents such as the Toulmin model of argument [148]. 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 1.43a shows such a argument tree. Sandbox [164] supports analysis of competing hypotheses – a judgment method that requires careful weighing of alternative explanations [69]. It allows users to add multiple hypotheses, each with supporting or contradictory evidence. These pieces of 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 1.43b shows an example of this support.

Despite all the aforementioned benefits, manual capture is only useful when users are willing to take notes, which can be perceived as distractions sometimes.

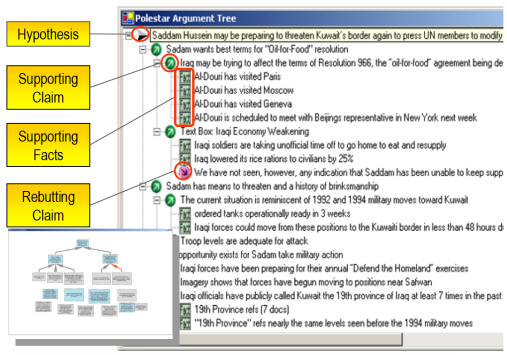


(a) A diagram of user notes showing their thoughts. *Image source: [137].*

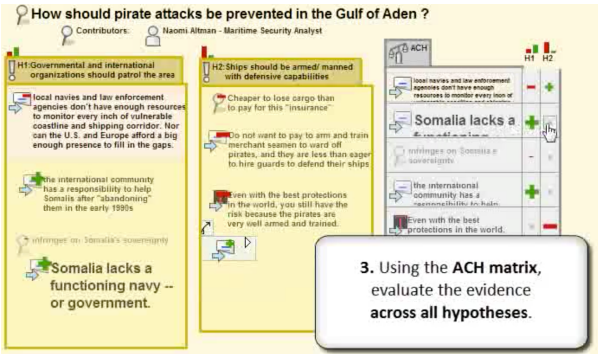


(b) A formal reasoning diagram with nodes having different roles: evidence, casual relationship, assumption and hypothesis. *Image source: [116].*

Figure 1.42: Task and subtask visualization.



(a) Argument tree. Analysts elaborate arguments via a tree structure of claims, sub-claims, and facts. *Image source: [117].*



(b) Analysis of competing hypotheses. The interface allows adding multiple hypotheses, each with weighted supporting/counter evidence, and producing a summary matrix to evaluate them. *Image source: a video snapshot of [164].*

Figure 1.43: Analytical support for the sensemaking process process with externalized thinking artifacts.

Two common strategies to alleviate this include minimizing interruption/cognitive effort [73] and providing immediate benefits to the sensemaking task such as planing exploratory analysis for complex tasks [98]. 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, which are events and actions. 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 [36]. A domain-specific sensemaking task was used for participants to solve, and experts were recruited to analyze the their action logs. Higher-level analytic provenance manually inferred from the logs was 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 recovered correctly. 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 sense-making and the difficulty of replicating expert knowledge 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 can be small.

Given the difficulty of inferring task/sub-task information, a few methods target less semantic-rich provenance. One example is *action chunking*, which identifies a group of actions that are likely to be 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. For instance, one simple heuristic is about the action structure of a sub-task: a series of "exploratory" actions (e.g., sort, filter, zoom, etc) followed by an "insight" action (e.g., bookmark and annotation) [53]. However, it may only work if the user spends time on doing bookmark and annotation, which may be impractical as discussed previously. Another example is to extract sub-tasks from a sequence of actions based on predefined patterns [51]. One such pattern is "scan" defined as a series of "inspect" actions on similar data objects. For instance, in travel planning, when the user opens several hotels to check their price, he or she might want to scan or compare price of those hotels. This method is later extended to monitor user behavior for implicit signals of user intent and uses the information to suggests alternative visualization.

1.3.3 Summary

This section reviews research on analytic provenance, characterized by the four-level model by Gotz and Zhou [53]. It covers the capture and visualization of all levels: event, action, sub-task and task. It is more straightforward to capture events and

actions; however, they contain little semantics compared to sub-tasks and tasks. The later two levels are often manually captured through annotation. This thesis applies both manual capture (**??**) and automatic capture (**??**) of the action level, and supports users to gain understanding of the higher levels through visualization of the captured provenance data. Existing visualizations of provenance data are not specifically designed to support the iterative and dynamic nature of sensemaking. They mainly support users to recall and replicate their analyses, recover their performed actions, and present their final analysis results. This thesis focuses on supporting the ongoing and iterative sensemaking by contributing novel visualization techniques to explore complex temporal (**??** and **??**) and rational (**??** and **??**) relationships hidden in the sensemaking problem. Literature of temporal and network visualization (to help explore rational relationship) will be discussed in the next two sections, respectively.

1.4 Visualization of Time-Oriented Data

Time is an essential aspect of life because 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 temporal relationship of data. One of the oldest documented timelines was created back in 1765 entitled *Chart of Biography* by Joseph Priestley [123] (Figure 1.44). 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.

Since then, many visualization techniques have been developed to effectively reveal the temporal relationship of data. The book by Aigner et al. [3] provides a comprehensive review of this topic. In this section, we focus on different visual mappings of time, or how to represent time.

1.4.1 Horizontal

The most common representation of time is mapping it to a horizontal axis as in the *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.

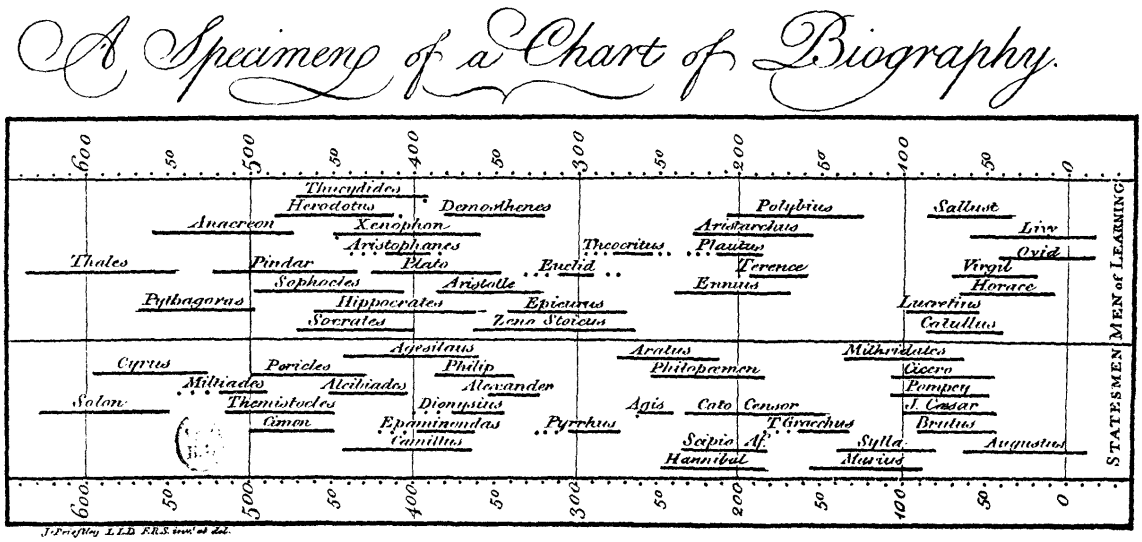
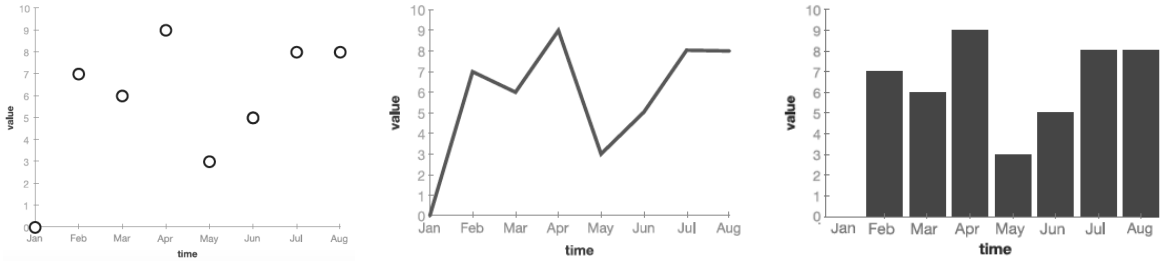


Figure 1.44: Joseph Priestley's Chart of Biography portraying the lifespans of famous persons. Each line segment represents a person, starting at when he/she was born and ending at when he/she died. *Image source: [123].*

1.4.1.1 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 [3]. Figure 1.45 shows an example for each of these three charts.

Horizon graph [126] is an improvement of line chart for visualizing time-series data, designed for a more space-efficient representation 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



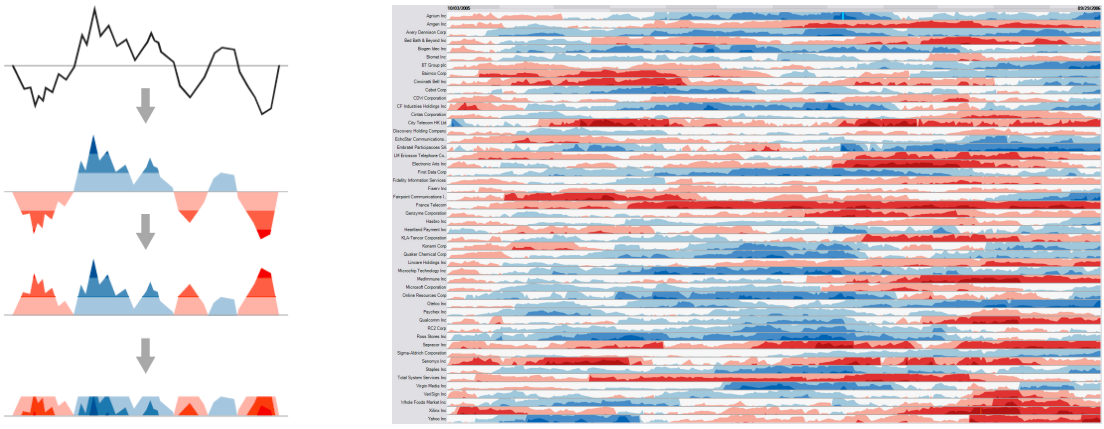
(a) Scatter plot: showing data points as dots.

(b) Line chart: connecting data points with lines.

(c) Bar chart: showing data points as aligned bars.

Figure 1.45: Three charts showing the same time-series dataset with the horizontal axis representing time and the vertical axis representing a numerical value. *Image source: [3].*

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 [131]. Figure 1.46a illustrates these steps, and Figure 1.46b shows prices of 50 stocks over a year.



(a) The construction steps: color → mirror → layer.

(b) A horizon graph showing prices of 50 stocks over a year, each row for a stock.

Figure 1.46: Horizon Graph: a space-efficient visualization of time-series data compared to line chart. *Image source: [126].*

Horizon graph uses space more efficiently than line chart in visualizing time-series data. A study by Heer, Kong and Agrawala [63] 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 [59] 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.

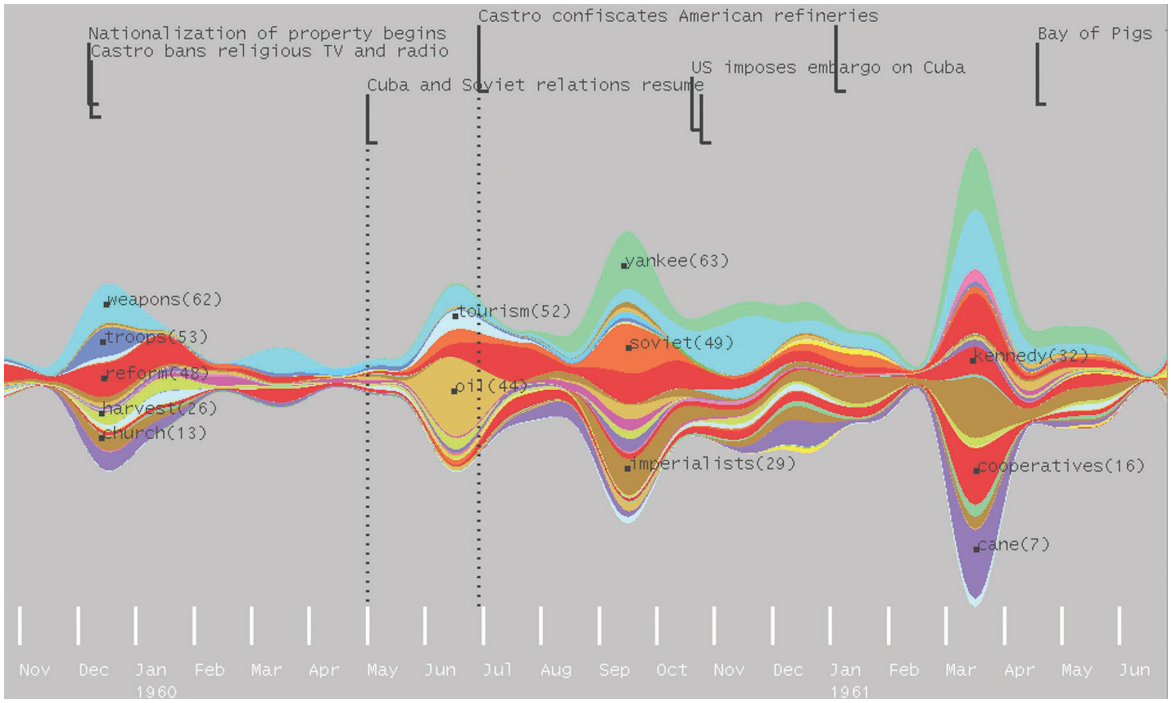
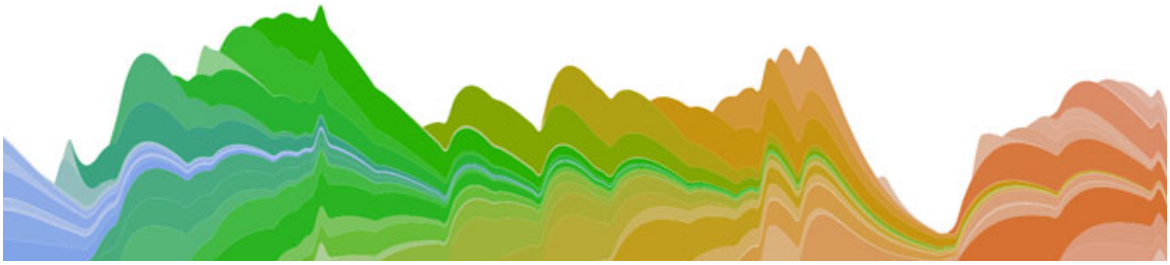
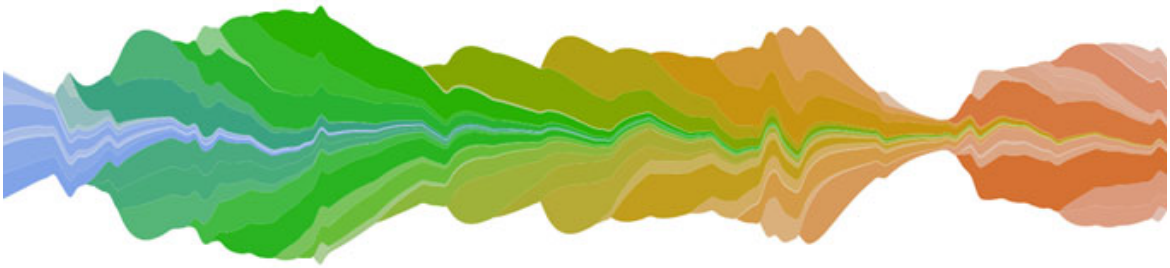


Figure 1.47: ThemeRiver uses a river metaphor to represent themes with each colored current corresponding to a theme. The data is a collection of Fidel Castro's speeches, interviews and articles from the end of 1959 to mid-1961. *Image source: [59].*

Byron and Wattenberg discuss considerations of aesthetics and legibility for designing such stacked graphs. Figure 1.48a shows the traditional stacked area chart, where the bottom of the lowest layer is a horizontal line at 0. Figure 1.48b shows the ThemeRiver version, which is optimized for the symmetry of the entire layout. Figure 1.48c shows the Stream Graph [18] version, which minimizes the number of wiggles of layers.



(a) Traditional stacked graph with the baseline at 0.



(b) ThemeRiver aiming for a symmetric layout around the horizontal xis.

(c) Stream Graph minimizing the number of wiggles of layers.

Figure 1.48: Stacked graphs with different design considerations. *Image source: [3].*

1.4.1.2 Non-Scaled Vertical Axis

Instead of representing another data attribute, the vertical dimension can be used to avoid clutter in producing a compact layout. LifeLines [120, 121], a visualization of patient records, is a good example. It groups these records into different facets such as problems, allergies, diagnosis and medications before vertically stacking them on top of each other. Within each facet, interval-based records are represented as horizontal bars covering their timespans. Because their timespans may overlap, the layout adjusts their vertical coordinates to avoid intersection. Figure 1.49 shows an example of LifeLines.

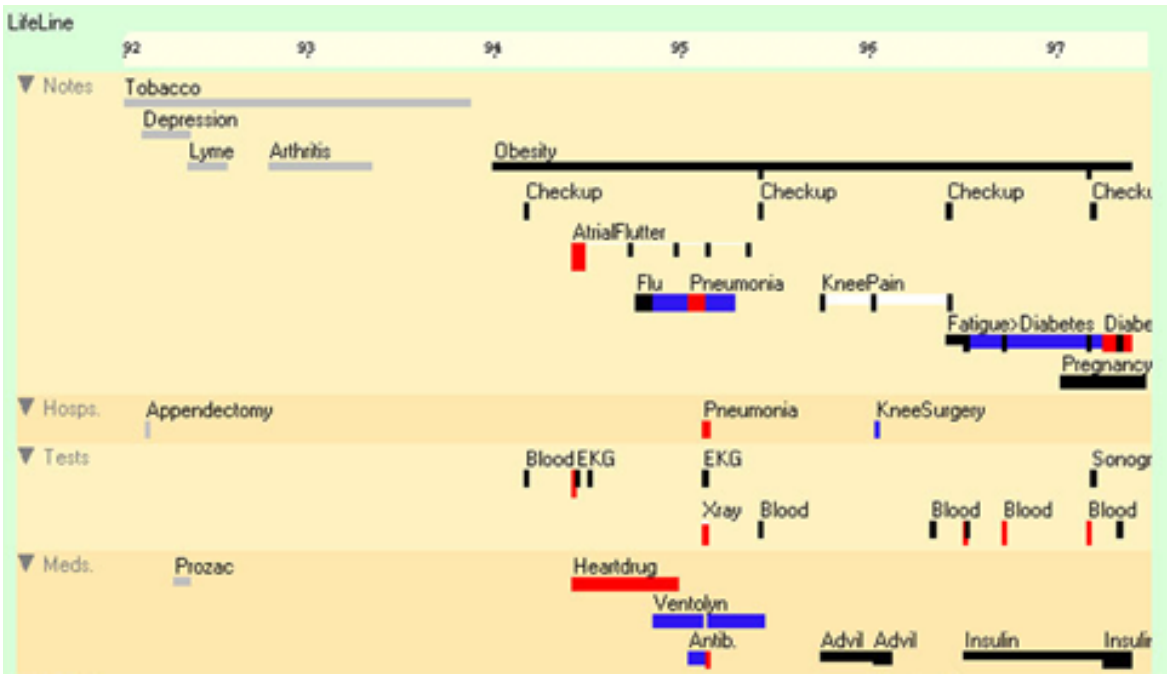


Figure 1.49: LifeLines. The timeline consists of several facets, stacked vertically. Each facet includes health-related records shown as horizontal bars, which may be located in different rows to avoid overlapping. *Image source: [121].*

Another example of timeline with more complex data structure is Continuum [5], which targets hierarchically structured temporal data; for instance, the relationship of era, composers and pieces in the music domain. 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 encodes additional relationship of data items. Gantt chart [46] 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 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 [17] draws an explicit node-link tree on the left part of the timeline to show the hierarchy. Figure 1.50 shows a Gantt chart with hierarchically structured tasks.

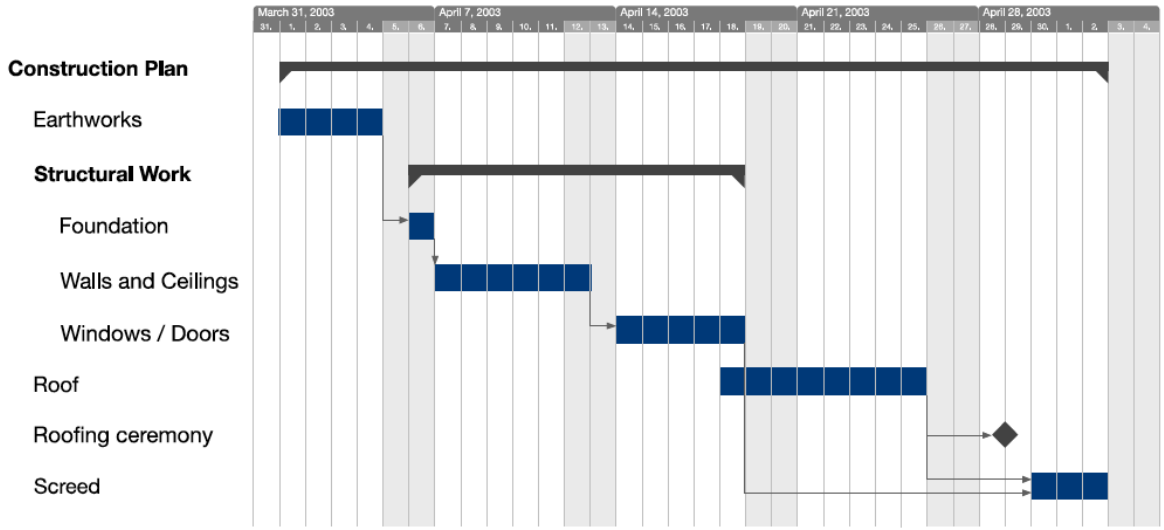


Figure 1.50: Gantt chart. Each task is shown in an individual row with task name on the left and horizontal bar on the right, covering the task time. Arrows show dependency, and label indentation indicates the hierarchy of tasks. *Image source: [3].*

Spatial proximity is another method to represent relationship. 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 [109]. 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 [142] and Liu et al. [95]. Figure 1.51 shows the storyline visualization of the *Star Wars* movie.

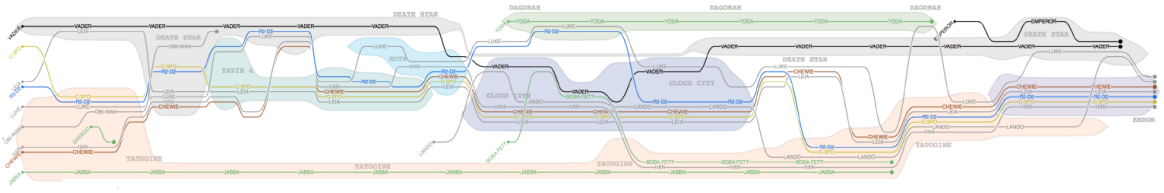


Figure 1.51: Storyline visualization of the movie *Star Wars*. Each line represents a character, and a bundle of lines represents an interaction of those characters. *Image source: [142].*

Similarly, TimeNets [85] also applies *proximity* to visualize relationship 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 1.52 shows a TimeNets example depicting these relationships.

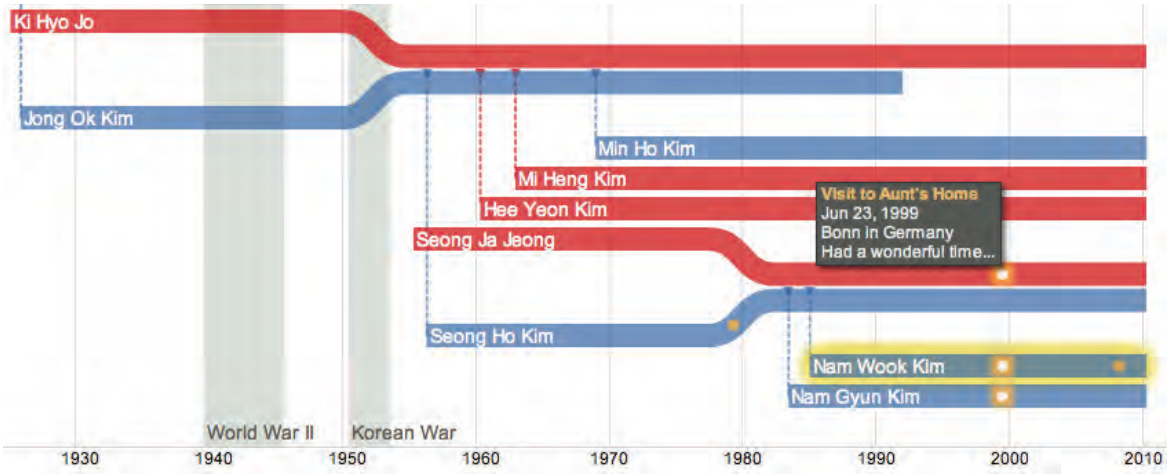


Figure 1.52: TimeNet visualization of genealogical data. Lifelines represent people with converging lines signifying marriage, and dotted lines indicating children. *Image source: [85].*

1.4.2 Spiral

Another representation of time is mapping it to a *spiral* axis, and data items are positioned along that spiral [159]. 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 [20]. Spirals can also be intertwined to show multiple variables. Figure 1.53 shows a spiral graph comparing two variables over time with different color hues.

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 1.54 shows three charts using the same time-series dataset. Figure 1.54a 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 1.54c. The difference between them is the cycle length: 24 days for Figure 1.54b, but 28 days for Figure 1.54c, 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 [146, 159]. Users can manually adjust the cycle length. Alternatively, users can watch an animation of the visualization

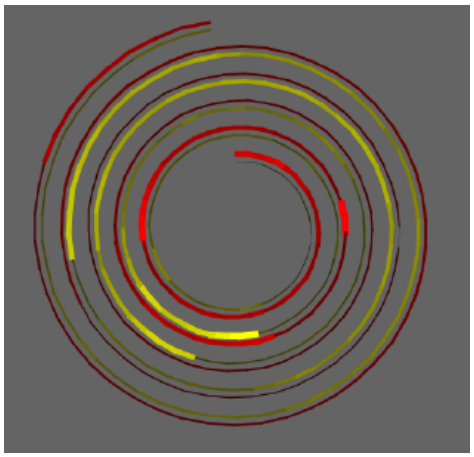
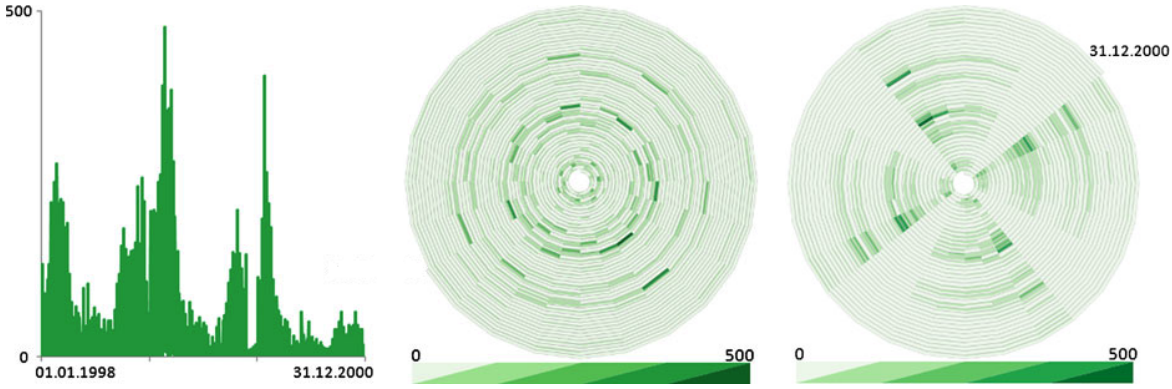


Figure 1.53: Spiral graph. Time is represented along a spiral with color intensity and line thickness are used to indicate time-dependent, numerical values. Two variables are distinguished using different color hue: yellow and red. *Image source: [159].*

through different cycle lengths and stop the animation when they find the pattern of interest.



(a) Bar chart: revealing trends and extreme values.

(b) Spiral graph: no pattern is found with 24-day cycle.

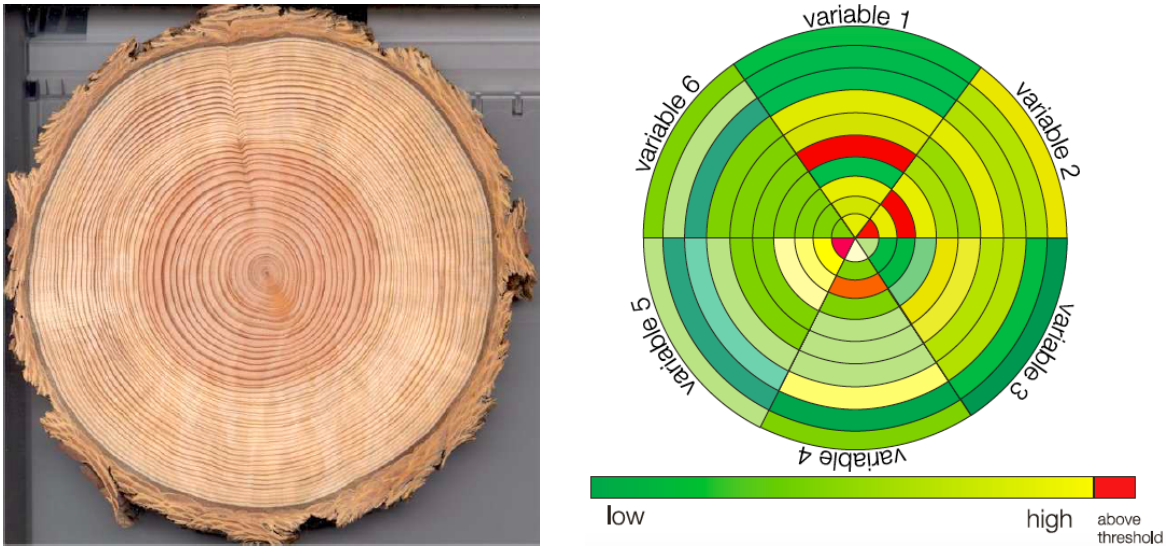
(c) Spiral graph: weekly pattern is clear with 28-day cycle.

Figure 1.54: Different insights can be gained from visual representations depending on whether the linear or cyclic character of the data is emphasized. *Image source: [3].*

1.4.3 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 1.55a). Inspired from this phenomenon, Keim, Schneidewind and Sips [83] 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 1.55b shows such a view with six variables. Color is used to show the (aggregated) data value for the corresponding interval.



(a) Cross section of a Douglas Fir tree showing its tree-rings. *Image source: [144].*

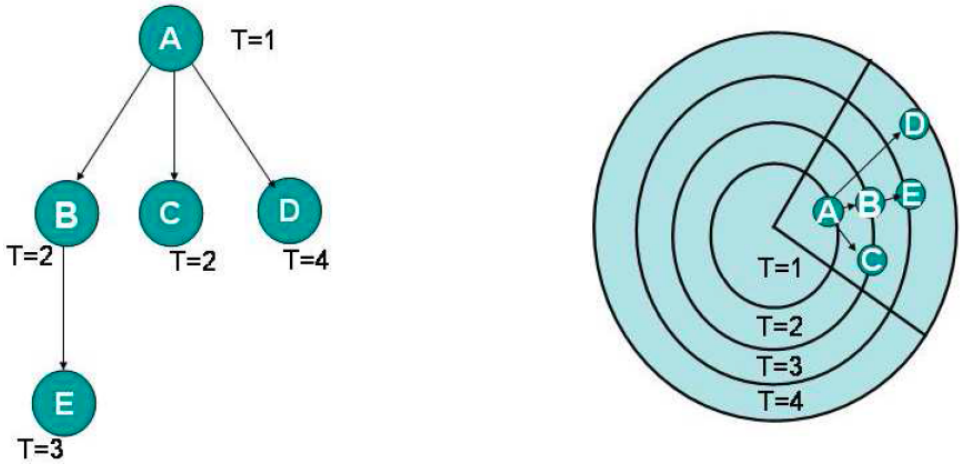
(b) Circle View showing six variables over ten time steps. *Image source: [83].*

Figure 1.55: Tree ring. Concentric rings expanding from the center, each indicating a time step.

Based on tree-rings, Therón [144] develops a technique to show both temporal and hierarchical information. Figure 1.56a shows a simple tree with five nodes, each is 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 1.56b).

1.4.4 Calendar

Another method to represent time is using a popular *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 1.57 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



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

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

Figure 1.56: Tree rings encoding both temporal and hierarchical information at the same time. *Image source: [144].*

some patterns are clear for Dutch people such as on December 5th, employees can leave their office earlier to celebrate Santa Claus.

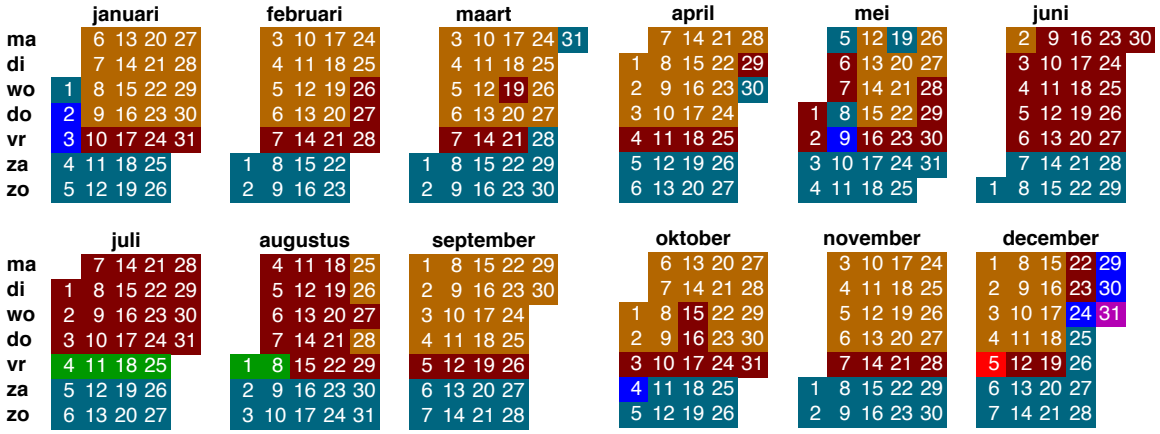


Figure 1.57: Calendar visualization of a one-year data of daily power consumption. A cell for each day is color coded to reveal data patterns. *Image source: [154].*

1.4.5 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 [149]. 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 1.58 shows an example of small multiples of bar charts.

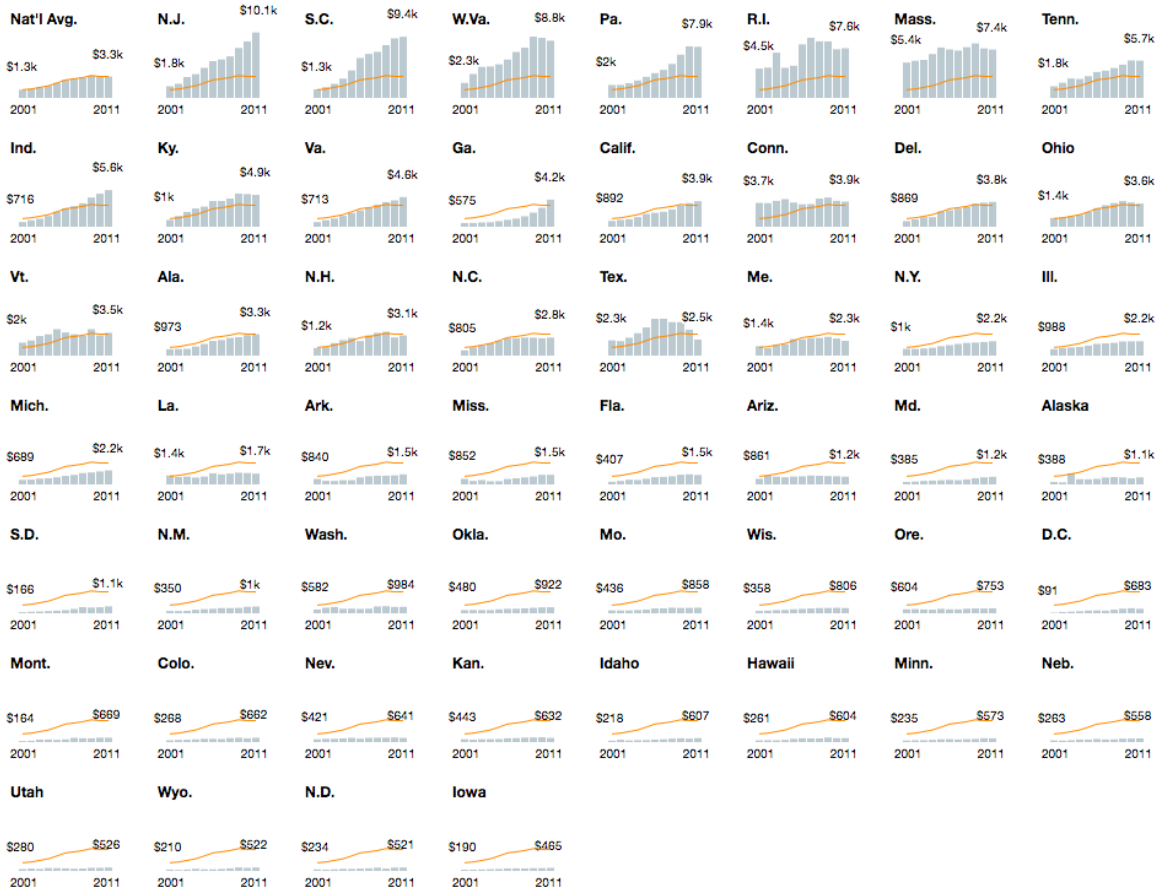


Figure 1.58: Small multiples of bar charts showing average annual Medicare spending on ambulance services per dialysis patient by U.S. state from 2001 to 2011. *Image source: [2].*

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 thumbnails. They are often sortable to reveal the pattern more effectively. For example, bar charts in Figure 1.58 are sorted decreasingly by the value spent on the last year. Alternatively, they can be ordered alphabetically by state names to facilitate lookup. Brushing and linking can also helps compare the subsets of interest. Showing thumbnails at multiple time steps makes the comparison easier; however, the number of visible steps is limited by the thumbnail size.

1.4.6 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 while a visualization smoothly updates from one time step to another. A notable example is Trendalyzer by Gapminder Foundation [1] – 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. [128] shows that animation is both slower and less accurate than small multiples in conveying trends over time.

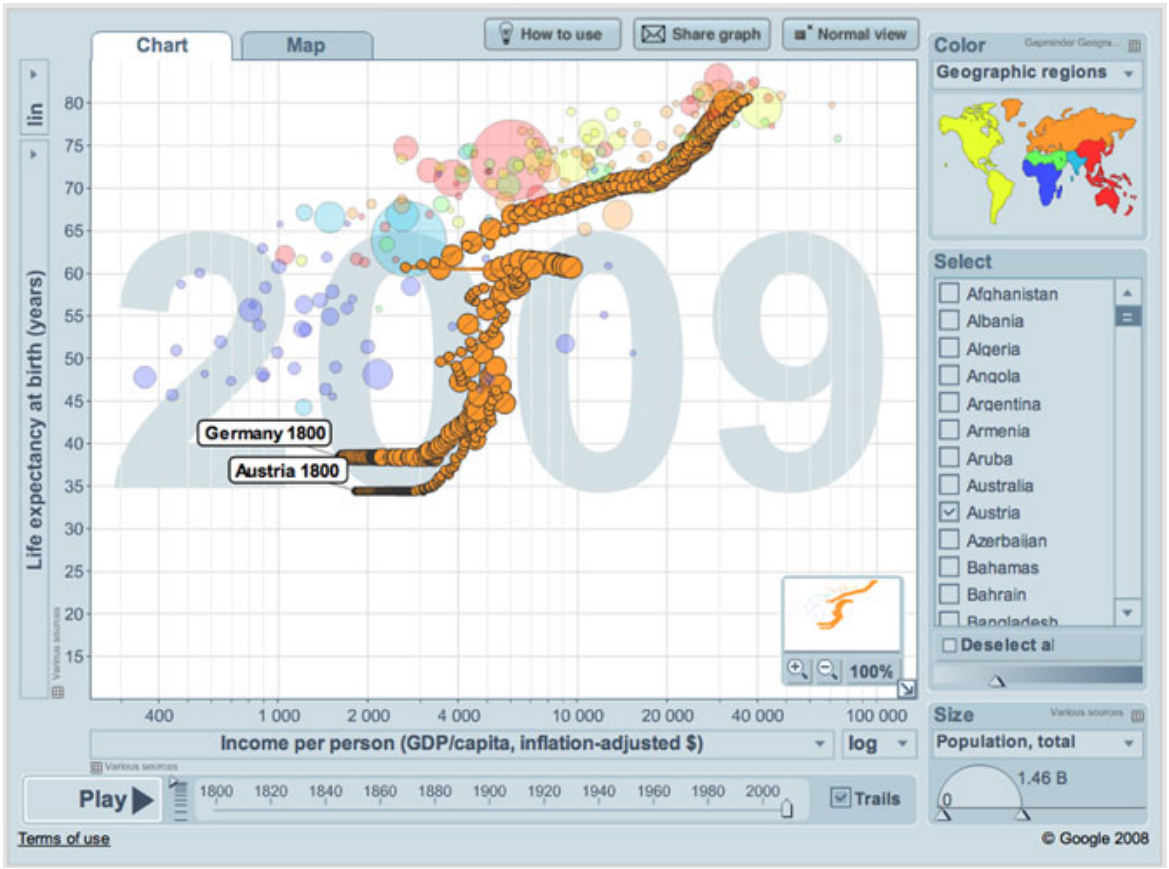


Figure 1.59: Trendalyzer interface. A scatter plot with an animation controller to traverse through time. Additionally, trails are activated for the selected countries, Austria and Germany, which help to preserve the path of a variable in animation. *Image source: [3].*

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 [47]. It helps illustrate computer algorithms step by step and motivate students in approaching complex problems [80]. Animation also allows reproduction of a data exploration process by interpolating visual parameters of key saved visualization steps [99].

1.4.7 Summary

This section reviews visualization of time-oriented data according to how time is visually encoded. None of the time mappings is perfect; some are more effective than others in showing particular characteristics of data. Horizontal time axis is a conventional mapping and easy to understand, whereas spatial time axis is good at detecting cyclic patterns. Calendar is another user-friendly representation and can reveal patterns at different granularities such as daily, weekly and monthly. Animation is engaging and effective in presentation, but is disadvantageous to data comparison at multiple time steps, which is a strength of small multiples. In this thesis, **??** contributes a novel timeline visualization that is inherently designed for sensemaking and based on horizontal time axis mapping. **??** extends the technique to support making sense of complex relationship by showing both temporal and categorical information simultaneously.

1.5 Visualization of Network and Tree Data

1.5.1 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 of graph structure: rooted tree, directed graph and general graph.

1.5.1.1 Tree Layout

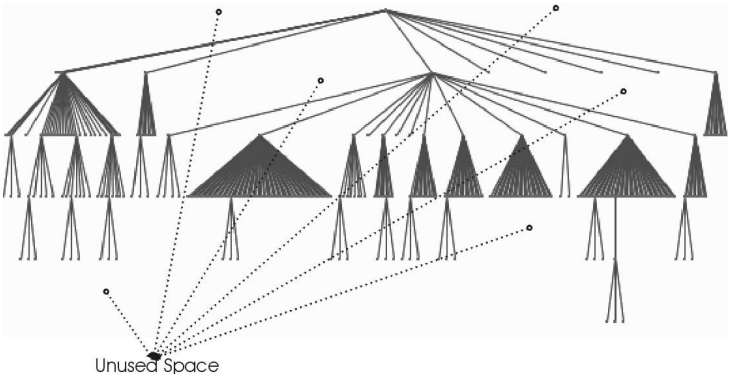
A classic algorithm by Reingold and Tilford [127] produces a tidy tree layout satisfying the following aesthetic rules:

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.

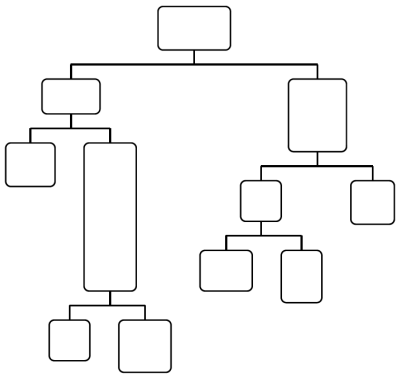
2. A left son should be positioned to the left of its father, and a right son should be positioned to the right of its father.

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 1.60a. Marriott and Sbarski [100] 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 [152] relaxes the layering requirement to produce a shorter tree as shown in Figure 1.60b.



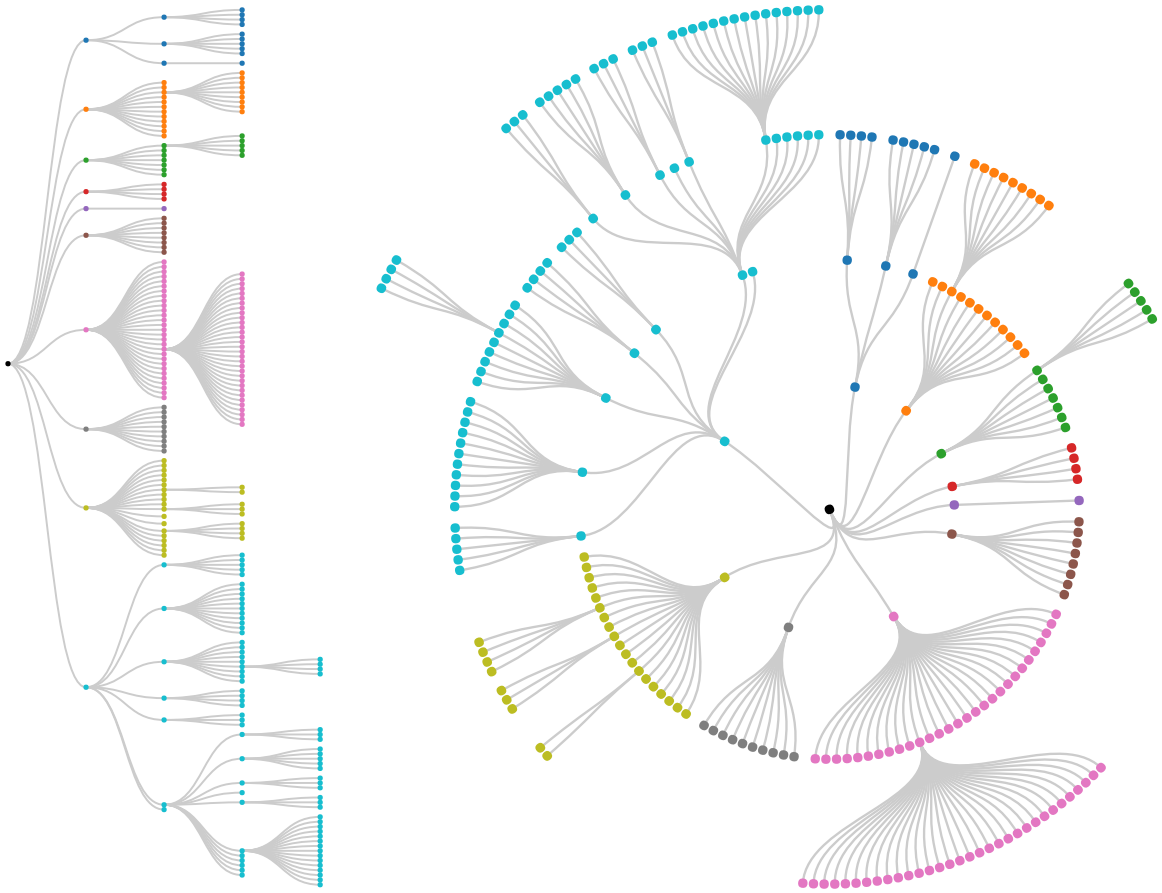
(a) Reingold – Tilford algorithm. *Image source: [112].*



(b) Relaxed layering for shorter layout. *Image source: [152].*

Figure 1.60: Tree layouts.

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. The radial version could be a bit more compact than the linear one and because the outer layer is longer than the inner ones, it has more space to show its nodes. However, it could be easier to see the hierarchical structure with the linear structure because of its familiarity. Figure 1.61 shows the class hierarchy of the *Flare* visualization toolkit [60] 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.



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

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

Figure 1.61: Tree layouts with different orientations. Data is the class hierarchy of the Flare visualization toolkit [60] with color representing the top-level classes.

1.5.1.2 Directed Graphs – Hierarchical Graph Layout

The most popular method of drawing directed graphs is the Sugiyama method [140], which separates nodes into layers to show the hierarchy effectively (Figure 1.62). The method consists of four steps.

1. *Cycle removal*. If the input graph contains directed cycles, the directions of some edges are temporarily reversed to make the graph acyclic.

2. *Layer assignment*. Nodes are assigned to horizontal layers, which determines their y-coordinates.

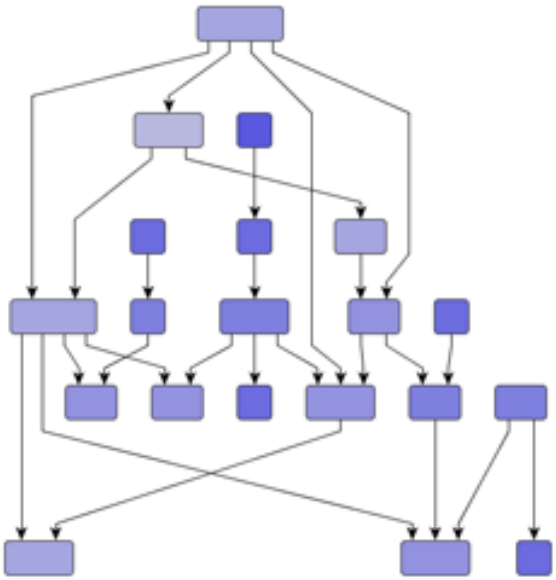


Figure 1.62: A layered graph. Nodes are assigned to horizontal layers. Within each layer, nodes are ordered to minimize edge crossings and assigned x-coordinate to make edges as straight as possible. *Image source: yWorks.*

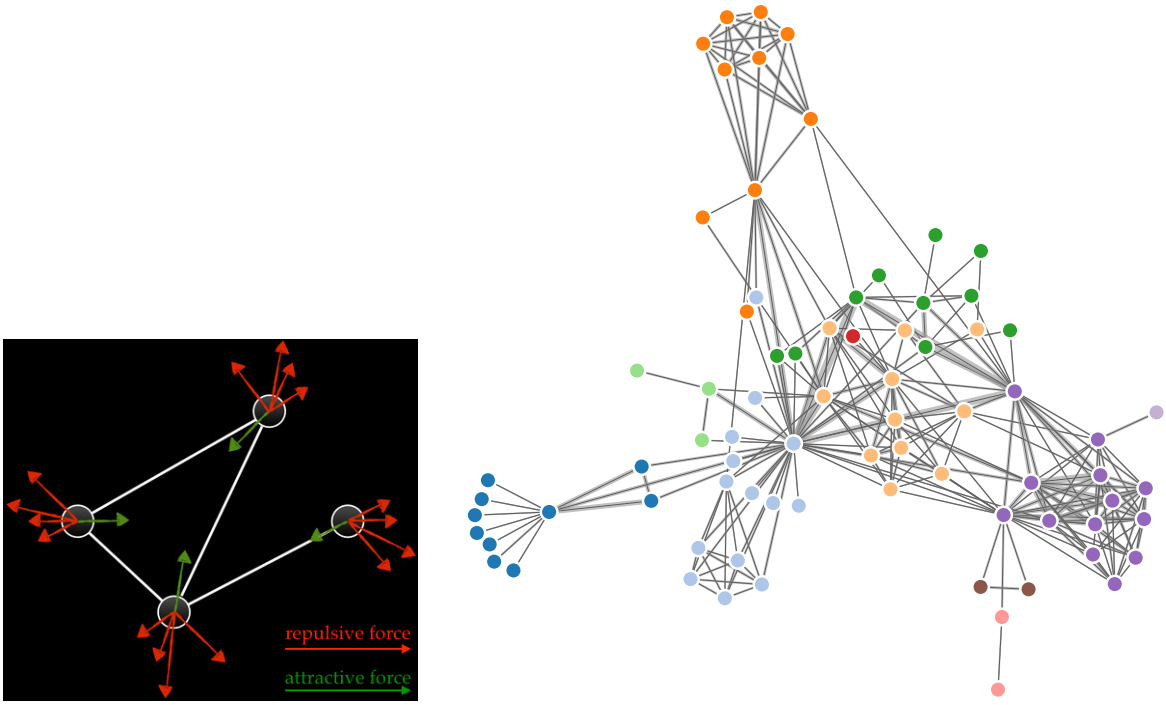
3. *Vertex ordering*. Within each layer, the nodes are ordered to minimize edge crossings between adjacent layers.

4. *Horizontal coordinate assignment*. The x-coordinate of each node is determined, typically aiming to make edges straight.

1.5.1.3 General Graph – Force-Directed Layout

Force-directed layout [38] is a popular method to visualize general graphs using node-link metaphor. 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 simulation stops when it reaches a stable state or a maximum number of iterations. Figure 1.63a illustrates the idea of physical simulation, and Figure 1.63b 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 [43]. However, its major weakness is scalability, in terms of both the visual complexity and the running time [110]. The layout quickly degenerates into a hairball of visual clutter with even a few hundred



(a) A physical simulation. **(b)** An example with color showing node type.

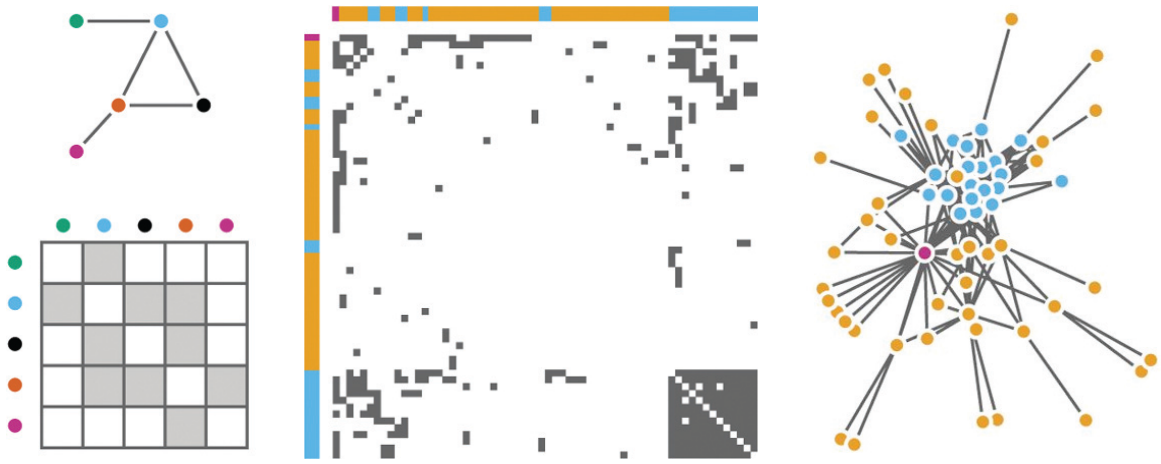
Figure 1.63: Force-directed layout.

nodes. Another limitation is its nondeterministic output: the layout looks different each time it runs, breaking user mental model.

1.5.2 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 1.64 shows examples of matrix views, compared with node-link diagrams of the same datasets.

A major strength of matrix views is the scalability. It can easily show a network with thousands of nodes and millions of edges without suffering from the cluttering issue as in node-link diagrams. Matrix views are also 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 [110]. Nodes, as columns and rows



(a) Comparison with a small network.

(b) Matrix view of larger network.

(c) Force-directed layout of larger network.

Figure 1.64: Comparison of node-link diagrams and matrix views. Gray cells indicate edge connectivity. *Image source: [110].*

in a matrix view, can be reordered, allowing users to reveal outliers, clusters, and patterns of the network [67].

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 [110]. 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 [48].

1.5.3 Space-Filling Techniques

Space-filling techniques can only be applied to tree data and uses *containment* marks to represent hierarchical relationship, placing child nodes within their parent node. Treemap [135] represents a node as a rectangle, which is recursively subdivided into smaller rectangles, each representing a child of the node. The rectangle size is proportional to a quantitative attribute of its 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 1.65 shows such an example of treemap.

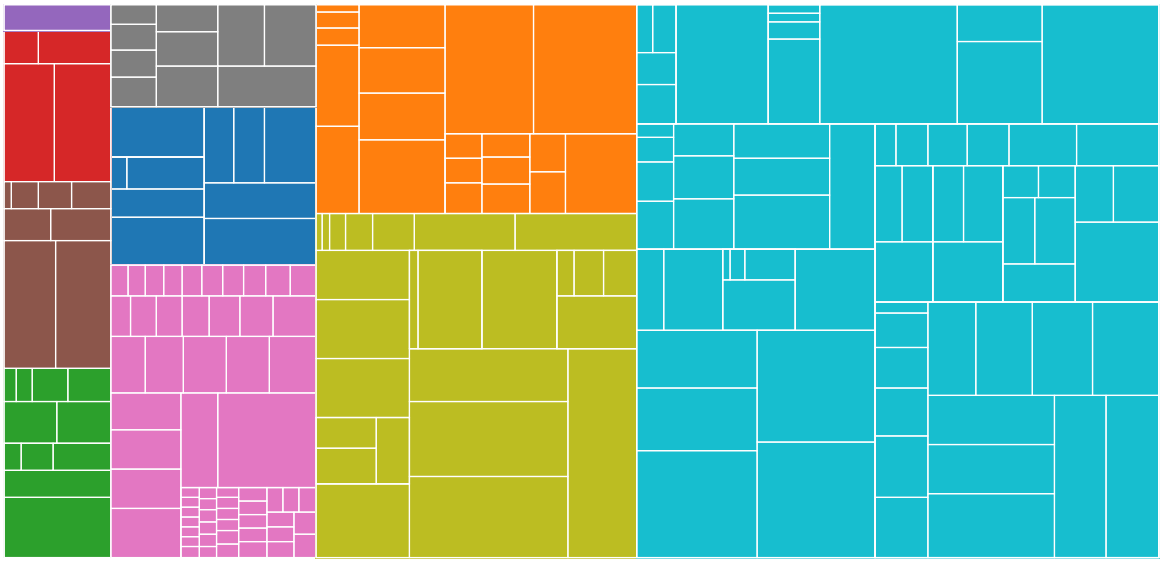


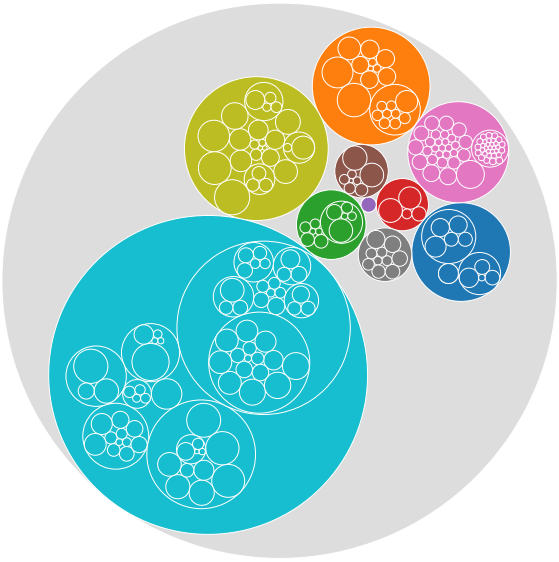
Figure 1.65: Treemap showing the class hierarchy of the Flare visualization toolkit [60]. Area represents class sizes and color represents the top-level classes.

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 can be used to depict the tree structure more strongly [161].

Alternatively, other space-filling techniques have been proposed to represent the hierarchical structure more effectively. Figure 1.66 shows three techniques that will be discussed next using the same dataset as in Figure 1.65 for treemap.

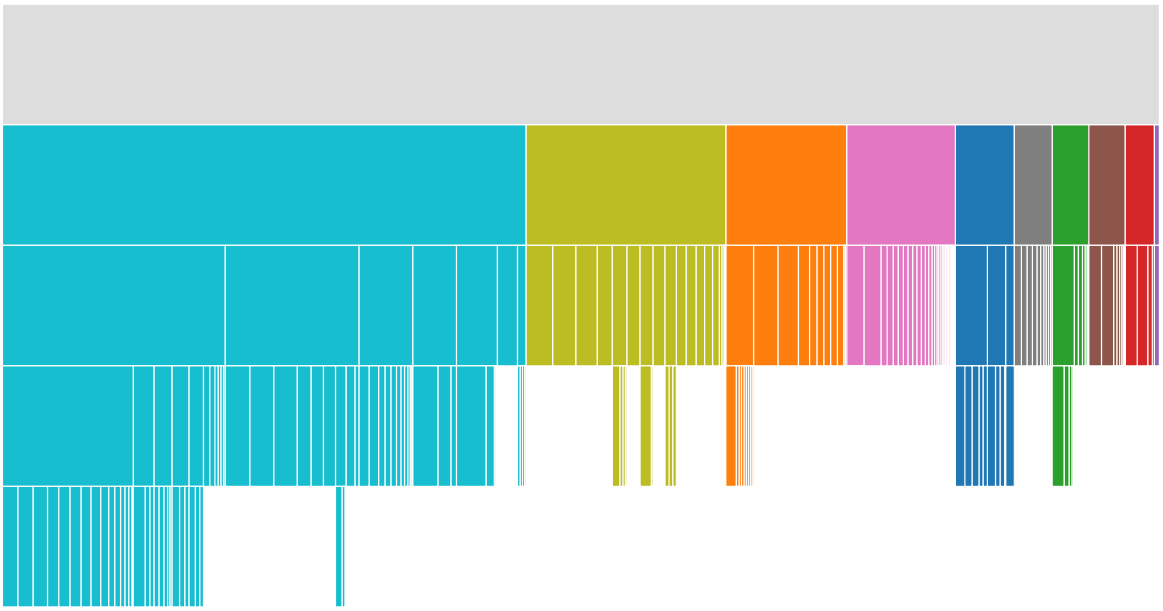
Circle packing [156] also employs containment to represent the hierarchy, but using circles to show nodes. Similar to rectangle size in treemap, the circle size also corresponds to some node value. All child nodes are packed into their parent node so that they are tangent to some of their sibling nodes. This method uses more space than treemap, but actually the additional void space helps convey the hierarchy structure more easily.

Icicle plot [90] 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



(a) Circle packing. Circle containment indicates parent-child relationship. Circle size represents a quantitative attribute.

(b) Sunburst. A curved segment is the parent of its next outer segments. Segment angle represents a quantitative attribute.



(c) Icicle plot. A rectangle is the parent of rectangles under it. Rectangle width represents a quantitative attribute.

Figure 1.66: Other space-filling techniques using the same dataset as in Figure 1.65. Color also represents the top-level classes of the Flare visualization toolkit [60].

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 to interact with.

Sunburst [169] 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.

1.5.4 Summary

This section reviews the most common techniques to visualize network data including node-link diagrams, matrix views and space-filling. Node-link diagrams are user-friendly and suitable for small datasets, whereas matrix views are better at large datasets. Space-filling techniques produce a compact visualization of tree data and are effective in using size to represent a quantitative attribute. **??** also uses network data showing pair-wise relationship between provenance data items. Even though it applies an existing tree layout, the research discussed in this section lays the foundation for making suitable design decision. To address scalability, our tree visualization includes representations for different levels of detail and interactive features such as semantic zooming and panning.

1.6 Chapter Summary

This chapter reviews the background and research related to the problem addressed in this thesis – visualization of analytic provenance for sensemaking. In this section, we briefly preview the relationship between the literature and our designs discussed in subsequent chapters.

Sensemaking support is the ultimate goal of the thesis. We use the Pirolli and Card's model and the Data–Frame model as the theoretical foundation for sensemaking, and design visualizations to support key parts of those models. These visualizations make use of analytic provenance data captured while the user interacts with the sensemaking system to solve his or her problem. Both manual and automatic capture data are used in this thesis. The visualizations are designed based on a number of well-established design principles and interaction techniques such as Gestalt laws for representing grouping information; brushing & linking and fluid interaction for coordinating linked views; and semantic zooming with different lev-

els of detail for scalability. Specific design ideas motivated by the literature include using horizontal axis to represent time (from time-oriented visualization) and using node-link diagrams to show pair-wise relations (from network visualization). The designs are validated rigorously using suitable evaluation methods discussed earlier. Design decisions are justified carefully before implementation and case studies with target audience are conducted to explore the effectiveness of the prototypes in supporting sensemaking. For specific design elements, a lab controlled experiment is used (**??**) to compare user performance in simpler tasks such as readability.

The four subsequent chapters will address the four research questions in the same order described in **??** based on the knowledge summarized here. The next chapter will address the first research question: designing visualization of timestamped provenance data to explore temporal relationship in sensemaking.

References

[1] Gapminder, 2010.

[2] Ambulances for Dialysis Patients on Rise, 2014.

[3] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Data*. Springer, 2011.

[4] 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.

[5] 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.

[6] F. J. Anscombe. Graphs in statistical analysis. *The American Statistician*, 27(1):17–21, 1973.

[7] E. Z. Ayers and J. T. Stasko. Using Graphic History in Browsing the World Wide Web. Technical report, 1995.

[8] 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.

[9] J. Bernard, N. Wilhelm, B. Krüger, T. May, T. Schreck, and J. Kohlhammer. MotionExplorer: exploratory search in human motion capture data based

on hierarchical aggregation. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2257–66, 2013.

[10] I. Borg and P. J. F. Groenen. *Modern multidimensional scaling : theory and applications*. Springer, 2005.

[11] H. Bosch, D. Thom, F. Heimerl, E. Puttmann, S. Koch, R. Kruger, M. Worner, and T. Ertl. ScatterBlogs2: Real-time monitoring of microblog messages through user-guided filtering. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2022–2031, 2013.

[12] 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.

[13] M. Brehmer and T. Munzner. A multi-level typology of abstract visualization tasks. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2376–2385, dec 2013.

[14] 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.

[15] 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.

[16] 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.

[17] 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.

[18] L. Byron and M. Wattenberg. Stacked graphs-geometry & aesthetics. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1245–52, 2008.

[19] S. K. Card, J. D. Mackinlay, and B. Shneiderman. *Readings in information visualization: using vision to think*. Morgan Kaufmann Publishers Inc., jan 1999.

[20] 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.

[21] S. Carpendale. Evaluating Information Visualizations. *Information Visualization*, pages 19–45, 2008.

[22] 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.

[23] B. chul Kwon, W. Javed, N. Elmqvist, 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.

[24] 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.

[25] 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.

[26] 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.

[27] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 3rd edition, 2009.

[28] P. Cowley, J. Haack, R. Littlefield, and E. Hampson. Glass box: capturing, archiving, and retrieving workstation activities. In *ACM Workshop on Continuous Archival and Retrival of Personal Experence*, pages 13–18, New York, New York, USA, oct 2006. ACM Press.

[29] 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.

[30] 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.

[31] 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.

[32] B. Dervin. An overview of sense-making research: Concepts, methods, and results to date, 1983.

[33] 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.

[34] 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.

[35] 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.

[36] 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.

[37] C. Dunne, N. Henry Riche, B. Lee, R. A. Metoyer, and G. G. Robertson. GraphTrail: Analyzing Large Multivariate and Heterogeneous Networks while Supporting Exploration History. In *ACM Conference on Human Factors in Computing Systems*, pages 1663–1672, 2012.

[38] P. Eades. A heuristics for graph drawing. *Congressus numerantium*, 42:146–160, 1984.

[39] N. Elmqvist, 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.

[40] J. Fan, Y. Gao, H. Luo, and G. Xu. Automatic image annotation by using concept-sensitive salient objects for image content representation. *Proceedings of the 27th annual international conference on Research and development in information retrieval - SIGIR '04*, (JANUARY):361, 2004.

[41] T. Fawcett. An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[42] A. P. Field and G. Hole. *How to design and report experiments*. Sage Publications, 2003.

[43] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Software: Practice and Experience*, 21(11):1129–1164, nov 1991.

[44] G. W. Furnas. Generalized fisheye views. In *ACM Conference on Human Factors in Computing Systems*, pages 16–23, 1986.

[45] 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.

[46] H. L. Gantt. *Work, wages, and profits*. Engineering Magazine Co., 1913.

[47] N. Gershon and W. Page. What storytelling can do for information visualization. *Communications of the ACM*, 44(8):31–37, aug 2001.

[48] 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.

[49] M. Ghoniem, D. Luo, J. Yang, and W. Ribarsky. NewsLab: Exploratory Broadcast News Video Analysis. In *2007 IEEE Symposium on Visual Analytics Science and Technology*, pages 123–130, 2007.

[50] 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.

[51] D. Gotz and Z. Wen. Behavior-driven visualization recommendation. In *International Conference on Intelligent User Interfaces*, pages 315–324, New York, New York, USA, 2009. ACM Press.

[52] 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.

[53] D. Gotz and M. X. Zhou. Characterizing users' visual analytic activity for insight provenance. *Information Visualization*, 8(1):42–55, jan 2009.

[54] 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.

[55] P. Groth and L. Moreau. Representing distributed systems using the Open Provenance Model. *Future Generation Computer Systems*, 27(6):757–765, 2011.

[56] 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.

[57] J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[58] 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.

[59] 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.

[60] J. Heer. Flare: Data visualization for the web, 2009.

[61] 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.

[62] 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.

[63] 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.

[64] 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.

[65] J. Heer and B. Shneiderman. Interactive dynamics for visual analysis. *Communications of the ACM*, 10(2):26, apr 2012.

[66] 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.

[67] N. Henry and J.-D. Fekete. MatLink: Enhanced Matrix Visualization for Analyzing Social Networks. In *Human-Computer Interaction – INTERACT*, pages 288–302. 2007.

[68] 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.

[69] R. J. Heuer. Psychology of intelligence analysis. Technical report, 1999.

[70] R. R. Hightower, L. T. Ring, J. I. Helfman, 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.

[71] 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.

[72] K. Holtzblatt and S. Jones. Contextual Inquiry: A Participatory Technique for Systems Design. In *Participatory design: Principles and practices*, pages 177–210. 1993.

[73] 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.

[74] S. Jang, N. Elmqvist, S. Member, and K. Ramani. MotionFlow : Visual Abstraction and Aggregation of Sequential Patterns in Human Motion Tracking Data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):21–30, 2016.

[75] 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.

[76] K. K. *Principles of Gestalt psychology*. Oxford, England: Harcourt, Brace, 1935.

[77] 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.

[78] 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.

[79] 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.

[80] 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.

[81] D. Keim, G. Andrienko, J. D. Fekete, C. Görg, J. Kohlhammer, and G. Melançon. Visual Analytics : Definition , Process , and Challenges. *Information Visualization*, 4950(4):154–175, 2008.

[82] D. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann. *Mastering the Information Age - Solving Problems with Visual Analytics*. 2010.

[83] 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.

[84] D. A. Keim. Information Visualization and Visual Data Mining. *IEEE Transactions on Visualization and Computer Graphics*, 7(1):100–107, 2002.

[85] 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.

[86] 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.

[87] 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.

[88] R. Kosara, H. Hauser, and D. L. Gresh. An Interaction View on Information Visualization. In *State-of-the-Art Report. EuroGraphics.*, 2003.

[89] J. Krause, A. Perer, and E. Bertini. INFUSE: Interactive Feature Selection for Predictive Modeling of High Dimensional Data. *Visualization and Computer Graphics, IEEE Transactions on*, 20(12):1614–1623, 2014.

[90] J. B. Kruskal and J. M. Landwehr. Icicle Plots: Better Displays for Hierarchical Clustering. *The American Statistician*, 37(2):162–168, 1983.

[91] D. Kurlander and S. Feiner. Editable graphical histories. In *IEEE Workshop on Visual Languages*, pages 127–134. IEEE Comput. Soc. Press, 1988.

[92] 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.

[93] T. V. Landesberger, F. Brodkorb, P. Roskosch, and N. Andrienko. Mobility Graphs: Visual Analysis of Mass Mobility Dynamics via Spatio-Temporal Graphs and Clustering. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):11–20, 2016.

[94] J. Lazar, J. H. Feng, and H. Hochheiser. *Research methods in human-computer interaction*. John Wiley & Sons, Ltd., 2010.

[95] 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.

[96] 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.

[97] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, mar 1982.

[98] A. Lunzer. Lightweight Provenance-Driven Exploration. In *International Workshop on Analytic Provenance for Sensemaking*, 2014.

[99] K.-L. Ma. Image graphs - a novel approach to visual data exploration. In *IEEE Conference on Visualization*, pages 81–88, oct 1999.

[100] K. Marriott and P. Sbarski. Compact layout of layered trees. In *Australasian conference on Computer science*, volume 62, pages 7–14, 2007.

[101] 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.

[102] 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.

[103] 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.

[104] M. Migut and M. Worring. Visual exploration of classification models for risk assessment. In *IEEE Symposium on Visual Analytics Science and Technology*, pages 11–18, 2010.

[105] 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.

[106] N. Milic-Frayling, R. Sommerer, and K. Rodden. WebScout: support for revisitation of Web pages within a navigation session. In *International Conference on Web Intelligence*, pages 689–693. IEEE Comput. Soc, 2003.

[107] P. Missier, K. Belhajjame, and J. Cheney. The W3C PROV family of specifications for modelling provenance metadata. In *International Conference on Extending Database Technology*, pages 773–776, New York, New York, USA, 2013. ACM Press.

[108] 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.

[109] R. Munroe. Xkcd #657: Movie narrative charts. \url{http://xkcd.com/657}, 2009.

[110] T. Munzner. *Visualization Analysis and Design*. CRC Press, 2014.

[111] A. Nenkova and K. McKeown. A Survey of Text Summarization Techniques. In *Mining Text Data*, pages 43–76. Springer US, Boston, MA, 2012.

[112] Q. V. Nguyen and M. L. Huang. A space-optimized tree visualization. In *IEEE Symposium on Information Visualization*, pages 85–92, 2002.

[113] 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.

[114] 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.

[115] 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.

[116] W. A. Pike, J. Stasko, R. Chang, and T. A. O 'connell. The science of interaction. *Information Visualization*, 8(4):263–274, 2009.

[117] 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.

[118] 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.

[119] 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.

[120] 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.

[121] 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.

[122] 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.

[123] J. Priestley. *A Chart of Biography*. London: J. Johnson, St. Paul's Church Yard, 1765.

[124] J. R. Quinlan. Induction of Decision Trees. *Machine Learning*, 1(1):81–106, 1986.

[125] 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.

[126] H. Reijner. The development of the horizon graph, 2008.

[127] E. Reingold and J. Tilford. Tidier Drawings of Trees. *IEEE Transactions on Software Engineering*, SE-7(2):223–228, mar 1981.

[128] 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.

[129] D. Russell, R. Jeffries, and L. Irani. Sensemaking for the rest of us. *CHI workshop on Sensemaking*, 2008.

[130] D. M. Russell, M. J. Stefik, 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.

[131] 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.

[132] 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.

[133] 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.

[134] B. Shneiderman. The future of interactive systems and the emergence of direct manipulation. *Behaviour & Information Technology*, 1(3):237–256, 1982.

[135] B. Shneiderman. Tree visualization with tree-maps: 2-d space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99, jan 1992.

[136] 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.

[137] Y. B. Shrinivasan 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.

[138] D. Snowden. Multi-ontology sense making: a new simplicity in decision making. *Journal of Innovation in Health Informatics*, 13(1):45–53, mar 2005.

[139] U. Software. Timelines in the nSpace2 Sandbox, 2012.

[140] 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.

[141] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to data mining*. Pearson Education India, 2006.

[142] 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.

[143] 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.

[144] R. Therón. Hierarchical-temporal data visualization using a tree-ring metaphor. In *Smart Graphics*, pages 70–81. Springer Berlin Heidelberg, 2006.

[145] J. J. Thomas and K. A. Cook. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. National Visualization and Analytics Center, 2005.

[146] C. Tominski and H. Schumann. Enhanced Interactive Spiral Display. In *The Annual SIGRAD Conference Special Theme: Interaction*, pages 53–56, 2008.

[147] M. Tory and T. Moller. Evaluating Visualizations: Do Expert Reviews Work? *IEEE Computer Graphics and Applications*, 25(5):8–11, sep 2005.

[148] S. E. Toulmin. *The uses of argument*. Cambridge University Press, 2003.

[149] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Connecticut, 1983.

[150] E. R. Tufte. *Envisioning Information*. Graphics Press, 1990.

[151] University of Utah. VisTrails Documentation Release 2.0.0. Technical report, 2012.

[152] A. van der Ploeg. Drawing non-layered tidy trees in linear time. *Software: Practice and Experience*, 44(12):1467–1484, dec 2014.

[153] 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.

[154] 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.

[155] 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.

[156] 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.

[157] C. Ware. *Information Visualization: Perception for Design*. Elsevier, 2013.

[158] 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.

[159] M. Weber, M. Alexa, and W. Muller. Visualizing time-series on spirals. In *IEEE Symposium on Information Visualization*, pages 7–13. IEEE, 2001.

[160] K. E. Weick. Sensemaking in organizations. *Sage*, 3, 1995.

[161] J. J. V. Wijk. Cushion Treemaps : Visualization of Hierarchical Information. In *IEEE Symposium on Information Visualization*, pages 73–78, 1999.

[162] L. Wilkinson. *The Grammar of Graphics*. Statistics and Computing. Springer-Verlag, New York, 2005.

[163] M. L. M. J. Wilson and M. L. M. J. 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.

[164] 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.

[165] 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.

[166] 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.

[167] J. Yang, J. Fan, D. Hubball, Y. Gao, H. Luo, W. Ribarsky, and M. Ward. Semantic image browser: Bridging information visualization with automated intelligent image analysis. *IEEE Symposium on Visual Analytics Science and Technology*, pages 191–198, 2006.

[168] 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.

[169] 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.

[170] 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.

[171] Y. Zhao, M. Wilde, and I. Foster. Applying the Virtual Data Provenance Model. In *International Provenance and Annotation Workshop*, pages 148–161. Springer, Berlin, Heidelberg, 2006.

[172] 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.