Visual Analytics of Dynamic Networks

Paolo Federico

Dissertation
Faculty of Informatics, TU Wien,
May 31, 2017

FAKULTÄT
FÜR !NFORMATIK
Faculty of Informatics

Visual Analytics of Dynamic Networks

DISSERTATION

submitted in partial fulfillment of the requirements for the degree of

Doktor/in der technischen Wissenschaften

by

Paolo Federico
Registration Number 0928613

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Univ.Prof.in Dr.in rer.soc.oec. Silvia Miksch

The dissertation has been reviewed by:

(Ao.Univ.Prof.in Dr.in phil. Margit Pohl)	(Univ.Prof. Dr.rer.nat. Dr.techn.h.c. Dr.-Ing.E.h. Thomas Ertl)

Wien, 25.04.2017

(Paolo Federico)

Acknowledgements

I would like to thank my advisor Silvia Miksch for supporting me during the course of my doctoral study with scientific rigour, friendly communication, and relentless enthusiasm into visual analytic research. Thanks to her and to all my present and past colleagues in the ieg/CVAST group, I could work on my research in an enjoyable, inspiring, and supportive work environment.

In addition, I would like to thank the researchers who collaborated to the projects in whose context I conducted my PhD, namely Wolfgang Aigner, Albert Amor-Amorós, Jürgen Pfeffer, Michael Smuc, Florian Windhager, and Lukas Zenk. Special thanks go to Wolfgang Aigner who provided fruitful suggestions and feedback in the first phases of my PhD. Christian Bors and Markus Bögl helped me with the German abstract.

The research leading to this thesis has received funding from the Austrian Research Promotion Agency (FFG) through ViENA (Visual Enteprise Network Analytics, project number 820928) and Expand (EXploratory visualization of PAtent Network Dynamics, project number 835937) as well as from the Austrian Federal Ministry of Science, Research, and Economy (formerly known as Austrian Federal Ministry of Economy, Family and Youth) through the Laura Bassi Centre for Visual Analytics Science and Technology (CVAST), project number: 822746 (Phase 1).

Abstract

While there are many well-established techniques to analyze and visualize static social networks, visual analysis of dynamic (i.e., time-oriented) network data emerged in recent years as a relevant research topic, facing several open problems. The dynamic nature of this kind of data, indeed, poses the challenge of understanding both its relational aspect (the structure of social interactions) and its temporal aspect (how they change over time).

In this doctoral work, we investigate how a visual analytics approach, integrating automatic analysis, visualization, and user interaction techniques, can support the examination of such dynamic networks. In particular, by focusing on this research problem, we present the following contributions: 1. we propose a set of novel metrics (*change centrality* metrics) to specifically analyse how the network structure changes over time; 2. we combine different visual encodings for the time-oriented aspect of network data, enabling smooth transformations between different views; 3. we introduce novel techniques for user interaction, such as interactive control of dynamic layout stability and the *vertigo zoom*, allowing seamless transitions between relational and temporal perspectives on dynamic network data.

We illustrate our approach by describing a prototypical implementation and demonstrate its utility by introducing a real-world usage scenario. Furthermore, we provide a validation of our approach by reporting findings from expert reviews (involving experts from both the visualization community and the problem domain) as well as from two task-based user-studies, namely a qualitative evaluation and a quantitative controlled experiment.

These findings afford an indication of the overall validity of our approach and allow us to discuss how particular techniques and their combinations can support specific analytical tasks on dynamic network data.

Contents

Part I

The problem

CHAPTER 1

Introduction

1.1 Motivation

Networks are exploited to model diverse phenomena in various domains: social interactions between human beings (sociology), as well as digital connections between electronic devices (communications), relationships between proteins (biology), and interdependencies of industrial sectors or regional markets (economics). Dynamic networks take into account changes over time: they not only model relations between different entities, but also consider the evolution of these relations, i.e. the way and the extent by which they change over time.

Dynamic social networks, in particular, can be useful to model and analyze human relationships in several potential scenarios: the informal social relationships of individuals within a family or a group of friends; the widespread connections through social networking services; the covert activities of small, interconnected terrorist cells; or the structured collaboration of employees within an enterprise.

Visual analysis of dynamic social networks is a topic that has been drawing increasing attention in recent years, not only from different research communities (not limited to social science researchers) but also from the general public. One reason for this interest is the availability of data: nowadays electronic devices often mediate interpersonal interactions or, because of their presence in our every day lives, are in any case able to capture and store social network data. Another reason is the possibility to gain objective insights about social relationships, in order to monitor and understand them, as well as take action to improve or better exploit them.

Whatever application domain we consider, a good visualization of dynamic networks has to support the analysis of the relational aspect (what is the structure of the network) as well as the temporal aspect (how the network evolves over time), and should also enable a seamless switching between the two perspectives.

In this work, we specifically consider the organizational network (i.e. a network consisting of the employees of an organization) of a knowledge intensive enterprise and focused on different kinds of relations, such as communication, collaboration, technical and practical advice, and spreading of new ideas. We aim to support users analyzing the evolution of these relations

as well as some performance indicators, by also considering how they relate to organizational changes: turnover, team restructuring, and other management actions.

Analysis of social, i.e., interpersonal, networks is obviously a sub-field of sociology; it utilizes concepts from graph theory, combined with data models and computational algorithms, to perform automated data analysis. Graph drawing and information visualization provide algorithms and techniques for static and interactive visualization of network data.

In this work, we tackle the problem from a multidisciplinary perspective, abstracting dynamic network data from the specific problem domain (i.e., social networks), and investigating a visual analytics approach, as a combination of automated analysis and interactive visualization.

1.2 Research questions

In particular, our research aims at investigating the following research question:

- *How can a visual analytics approach support the examination of dynamic networks according to specific user tasks?*

  This main research question can be further detailed by three interconnected sub questions:

  - *How can temporal aspects of network data and dynamic underlying phenomena be best visualized?*
  - *How can we integrate analytical methods and visualization techniques to address the complexity of dynamic networks?*
  - *Is it possible to combine analytical methods and interaction techniques to enhance the perception of network dynamics?*

1.3 The visual analytics approach

Visual analytics is defined as "the science of analytical reasoning facilitated by interactive visual interfaces" [367, p.4]; visual analytics tools and techniques are aimed at helping users (i) synthesize and derive insight from complex and dynamic data; (ii) detect the expected and discover the unexpected; (iii) provide timely, motivated, and understandable assessments; and (iv) support decision-making [367]. Visual analytics goes beyond information visualization, defined as "the use of computer-supported, interactive, visual representations of abstract data to amplify cognition" [86, p.7].

Visual analytics has also been defined as a combination of "automated analysis techniques with interactive visualisations for an effective understanding, reasoning and decision making" [227, p.7]. Visual analytics can be described by the interplay of three main components [229]: data analysis, visualization, and interaction (see Figure 1.1). As such, visual analytics combines the strengths of humans and computers, in particular the enormous computational power of machines to manage and process data with the cognitive and perceptual capabilities of humans to direct the analysis and make sense of the data based on priori domain knowledge.

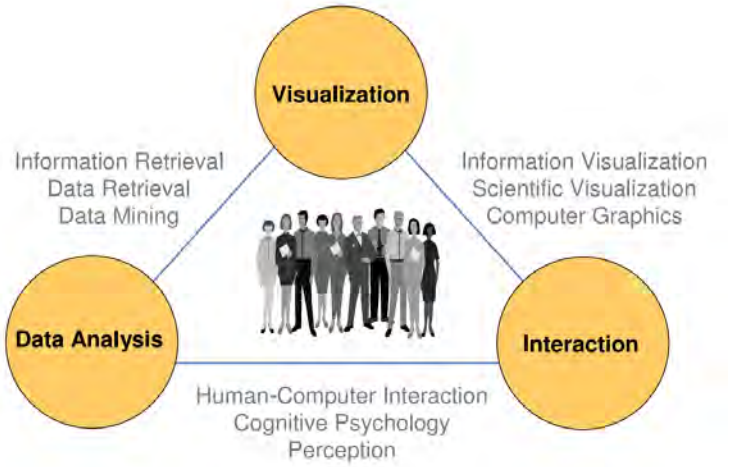


Figure 1.1: Visual analytics is based on the interplay of diverse methods and techniques for data analysis, visualization, and interaction [229]

From an epistemological perspective, visual analytics is an highly interdisciplinary research field, which combines diverse other fields [228] (Figure 1.2). As for data analysis, it integrates methodologies from data management and knowledge management, statistical analysis, knowledge discovery and data mining; as for visualization, it profits from techniques developed in the field of information visualization, scientific visualization, and geospatial visualization; moreover, visual analytics builds upon methods from cognitive and perceptual space and human-machine interaction; it also integrates methodologies for production, presentation, and dissemination of analysis results and insights.

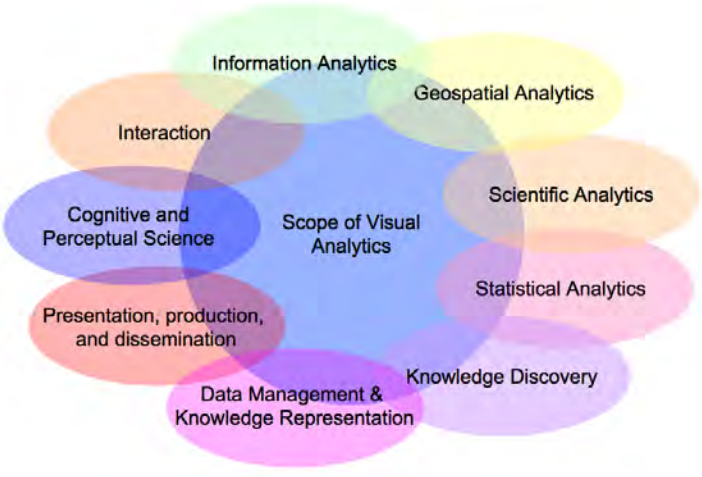


Figure 1.2: Scope of visual analytics [228]

1.4 Research methodology

The methodological foundation of this work is Munzner's nested model for design and validation of visualization [280]. In her work, Munzer identifies four levels for design and validation of visualization (Figure 1.3): (i) domain problem characterization; (ii) data/operation abstraction design; (iii) encoding/interaction techniques design; and (iv) algorithm design.

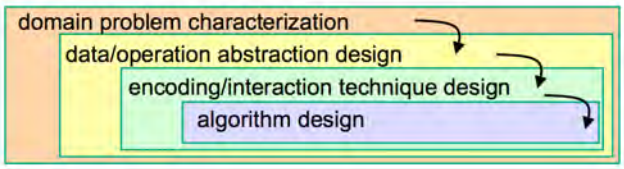


Figure 1.3: The nested model for design and validation of visualization [280]

The first, outermost level is the characterization of problems and data of a particular application domain (i.e., in our case, social network analysis). The second level is the abstraction of domain-specific data and problems into domain-independent data models and tasks. The third level is the design of visual encodings and interaction techniques. The fourth, innermost level, is the design and implementation of algorithms. Since these levels are nested (i.e. the output from an upstream level above is input to the downstream level below, see arrows in Figure 1.3), wrong design decisions cascade from outer levels into inner ones; furthermore, errors at an inner level can affect the validation of outer levels (e.g., the final evaluation of a system might cast doubts that a certain visual encoding does not work properly, but the reason might be a poor algorithmic implementation). Munzer, therefore, suggests two validate each level twice, by an immediate (e.g., upstream) validation as well as by a downstream validation. Level-specific threats and their validation strategies are shown in Figure 1.4.

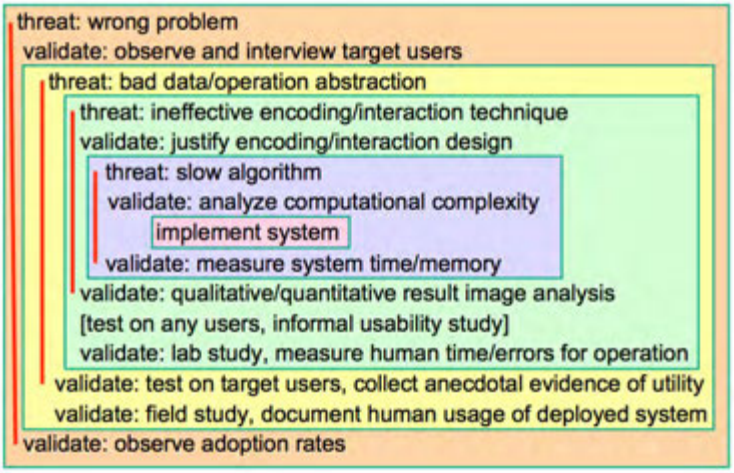


Figure 1.4: Threats and validation in the nested model [280]

Meyer et al. [273] extend the nested model by the introduction of the nested blocks and guideline model (Figure 1.5): blocks are the outcome of the analysis and design process at each level, and guidelines are statements that relate blocks to each other. Guidelines can be either between levels or within levels: the former indicate different possible choices to address situations identified in the outer level, while the latter indicate rankings between those different alternatives. Meyer et al. [273] explain it with an example about network visualization: a between-level guideline states that a node-link diagram is a good visual encoding of small graphs, while a within-level guideline states that one specific force-directed layout algorithm is faster than another.

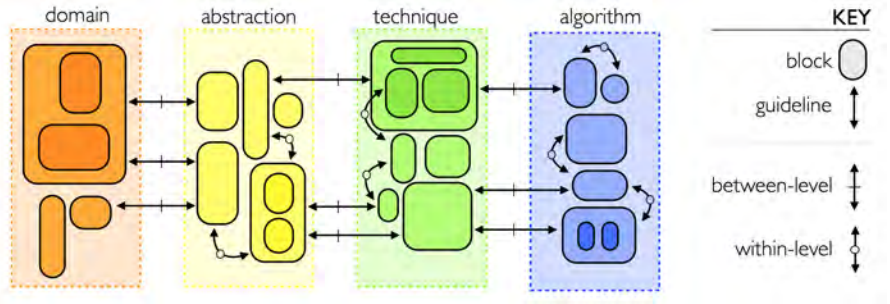


Figure 1.5: The nested blocks and guidelines model (adapted from [273])

The nested workflow model (Figure 1.6) is a further extension of the nested guideline model, aimed at providing designers with an expressive tool for characterizing the problem domain and understanding users' practices and processes [140]. This model enables the description of visual analytics processes at different design levels, in terms of tasks, data, and users, by taking into consideration complex workflow patterns, data and knowledge flows, and user collaboration.

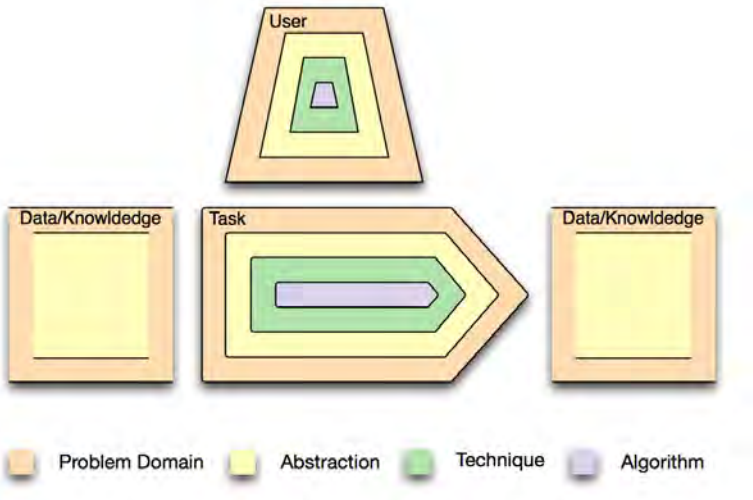


Figure 1.6: The nested workflow model [140]

The main components of the nested workflow model, namely data, tasks, and users, correspond to the vertices of the design triangle for visual analytics of time-oriented data by Miksch and Aigner [274] (Figure 1.7). The triangle provides designers with an orientation in the manifold space of visual analytics solutions by posing three questions: 1. Who are the users of the VA solution(s)? 2. What are the (general) tasks of the users? 3. What kinds of data are the users working with? While its vertices correspond to these questions, the design triangle also provides quality criteria the visual analytics solution should fulfil, namely expressiveness, effectiveness, and appropriateness. A visual analytics solution is expressive if it visualizes exactly the information contained in the data, for nothing more and nothing less must be visualized [265]. It is effective if it properly addresses the perceptual and cognitive capabilities of its user, but also the task at hand, the application background, and other domain- and context-related information [265]. It is appropriate if it is beneficial in terms of cost-value ratio, as defined by van Wijk [387].

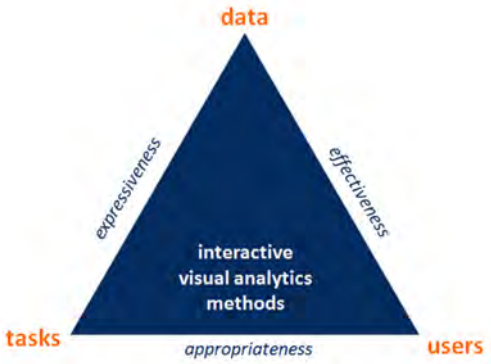


Figure 1.7: Design triangle [274]

Van Wijk [387], indeed, in order to investigate the value of visualization describes the context in which the visualization operates by introducing an operational model (Figure 1.8). The model encompasses three spaces: the data space, the visualization space, and the user space. Within and across these spaces, processes operate: visualization, perception/cognition, and exploration. The visualization if appropriate if it enables the increase of knowledge at a convenient cost, in terms of initial development, user sessions, and cognitive effort.

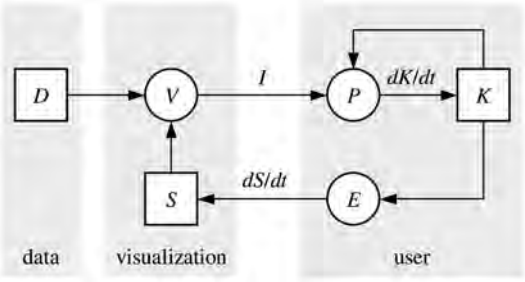


Figure 1.8: The operational model of visualization by van Wijk [387]

Conversely from van Wijk' model, the sense-making loop by Pirolli and Card [303] is more a conceptual model than an operational model, it does not distinguishes between human and computer and rather describes the reasoning process in terms of knowledge artifacts having an increasing structure and requiring increasing user effort (Figure 1.9).

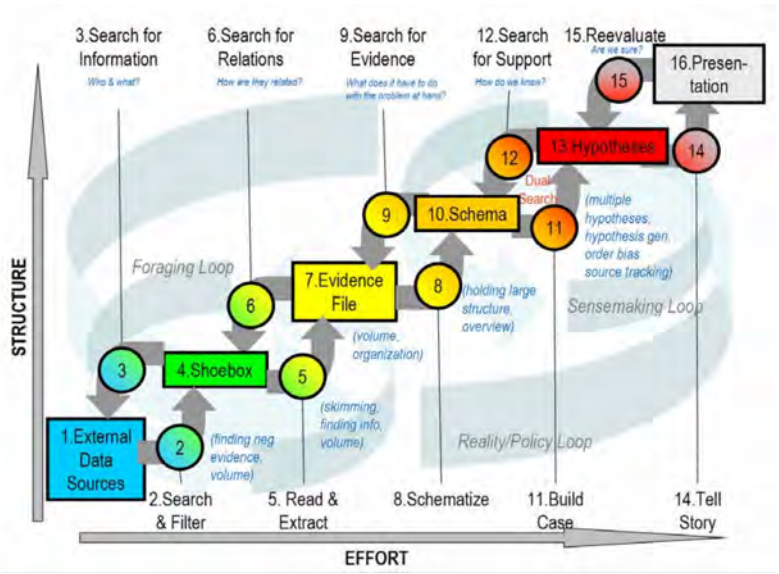


Figure 1.9: The sensemaking loop by Pirolli and Card [303]

Sacha et al. [335] introduce a knowledge generation model for visual analytics. Their model consists of computer and human parts and describes their intertwined collaboration (Figure 1.10). On the human side, the reasoning process is described in terms of three loops: the exploration loop, the verification loop, and the knowledge generation loop. The computer side encompasses two pipelines: the information visualization pipeline [86] and the knowledge-discovery pipeline [134].

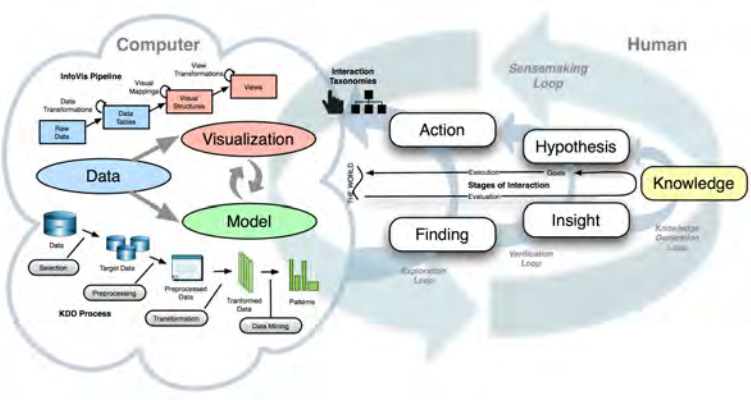


Figure 1.10: The knowledge generation model for visual analytics [335]

However, considering the different models, while the sensemaking loop [303] and the knowledge generation model [335] provide a broad theoretical framework for the comprehension of the visual analytics process, the design triangle [274], the nested model [279], and its derivatives [140,273] provide a methodological framework for conducting and validating our research. Sedlmair et al. [345] extend this framework by complementing the core validation with a precondition validation and a outward facing validation. The core validation consists of four stages (*discover*, *design*, *implement*, *deploy*) corresponding to the four levels of the nested model. The precondition validation comprehends three stages (*learn*, *winnow*, *cast*). The *learn* stage requires to explore the visualization literature (including visual encoding and interaction techniques, design guidelines, and evaluation methods) and the knowledge acquired and systematized during this phase will inform all the other stages. The *winnow* and *cast* stages are to select collaborators and assign roles in order to avoid practical pitfalls that might affect the research (e.g., no real data available, insufficient time available from potential collaborators, wrong domain problem). In the last stages (*reflect* and *write*), the researcher should confirm, refine, reject, or propose guidelines, contributing to extend the knowledge about the topic under investigation.

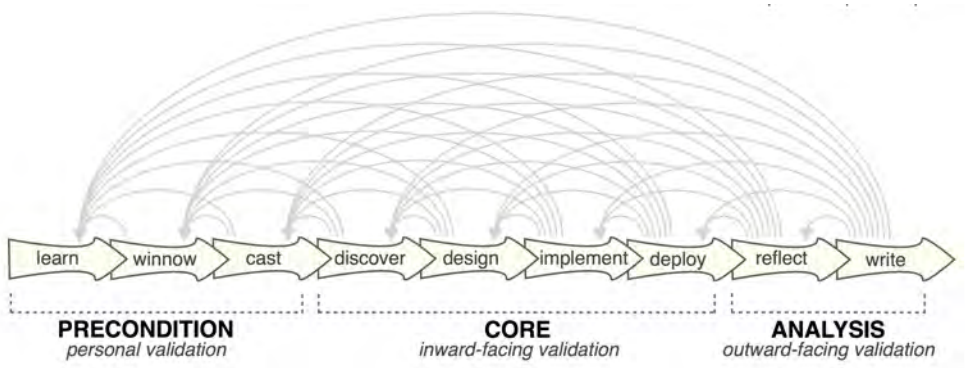


Figure 1.11: The nine-stage design study methodology [345]

The core stages, i.e. the four levels of the nested model, can be validated by a wealth of available evaluation techniques and methods. Most of them (except for the inmost level, which deals with algorithms, their correctness, performance, and computational complexity) adopt a focus-on-user approach as prescribed by the human-centered design concept [247]. According to Stone et al. [363, p.628], human-centered design can be defined as "an approach to user interface design and development that views the knowledge about intended users of a system as a central concern, including, for example, knowledge about user's abilities and needs, their task(s), and the environment(s) within which they work. These users would also be actively involved in the design process."

Ellis and Dix [120] distinguish two facets of validation: reasoned justification versus evaluation. Reasoned justification includes existing published results, own empirical data from prior studies, expert opinion and common sense, and arguments based on all the above. Evaluation include empirical evaluation (e.g., user studies), peer reviews, comparison with previous work. As for empirical evaluation, they distinguish between formative evaluation and summative evaluation. The purpose of summative evaluation is to validate a design, while the purpose of formative

evaluation is to improve a design; it is worht noting that, in the context of iterative development processes, the summative evaluation at the end of one iteration is a formative evaluation for the next one. Ellis and Dix also propose the notion of explorative evaluation, aimed at gaining findings, understang, and knowledge.

Lam et al. [251] elicit seven scenarios for evaluating visualization: Evaluating Environments and Work Practices (EWP), Evaluating Visual Data Analysis and Reasoning (VDAR), Evaluating User Performance (UP), Evaluating User Experience (UE), Automated Evaluation of Visualizations (AEV), Evaluating Communication through Visualization (CTV), Evaluating Collaborative Data Analysis (CDA). Each scenario has its own goals, outputs, and methods. Isenberg et al. [214] extend these scenarios by adding one more scenario, namely Qualitative Result Inspection (QRI), and renaming Automated Evaluation of Visualizations (AEV) as Algorithm Performance (AP). The AP scenario corresponds to the validation of the innermost nested level, involving automated (benchmark) measurements of quality and performance. As for communication (CTV) and collaboration (CDA) scenarios, Isenberg et al. [214] found few to no cases in visualization literature. EWP and VDAR are similar, since they both aim at understanding domain experts' analysis processes and practices; UWP scenarios focus more on understanding the current practices (and therefore is corresponds to the problem domain characterization) while VDAR scenarios involve the assessment of newly introduced tools by a group of domain experts (and therefore can be considered as the downstream validation of the data/operation abstraction as well as the domain characterization level. UP and UE, according to Isenberg et al. [214], are the most common form of evaluation, and correspond to the downstream validation of visual encodings and interaction techniques (when they involve general users / data / tasks) as well as data/domain operations (when they involve domain-specific situations). Qualitative Result Inspection (QRI) is not a form of evaluation in a classical sense, but rather a way to validate a proposed solution, technique or system, by a qualitative discussion of system behavior and interaction concepts or by letting the reader assess visual encodings and image quality.

User studies for characterizing the problem domain are usually based on qualitative methods, such as semi-structured interviews for user-centered design [416], both in-field and in-lab observational studies, as well as contextual inquiries [203] which combine observation and interviews; other methods for pre-design empiricism include [72]: immersive observation by impersonation, screen recording and artifact analysis, and wizard-of-Oz studies.

Expert reviews are also a suitable form of evaluation, and they result useful to assess interface usability as well as to generate valuable feedback on visualization tools; generally, heuristics should guide the process but not dominate it, leaving enough freedom to experts to provide qualitative feedback and discuss their opinions in detail [369]. Expert reviews can provide quick feedback on early implementations, while informal usability studies involving generic users can complement the findings in later development stages.

Quantitative user studies, with either generic or domain-specific users, tasks, and data, are a common evaluation for validating the abstraction as well as the visual encoding and interaction level, respectively. They are usually performed as task-based controlled experiments, i.e., participants are asked to solve given tasks under controlled conditions. Observed variables are usually task completion time and correctness rate. Rigorous statistical methods are utilized to verify if there is a significant effect of the factors upon the observed variables, in other words if different

conditions (e.g, two different visual analytics techniques) affect user performances [326].

Forsell and Cooper [150] provide guidance about how to report salient necessary information from different kinds of evaluation for visualization techniques.

1.5 Structure of this document

In this Chapter 1 we have motivated the problem, formulated our research questions, and presented our research methodology. In Chapter 2 we complement the **problem description** by introducing concepts and definitions and surveying the state of the art in visual analytics of dynamic networks.

Then we illustrate our **proposed solution**. In particular, in Chapter 3 we describe our integration of techniques for automated analysis, comprehending network mining and analysis techniques and a novel graph-theoretic measure for dynamic networks; in Chapter 4 we describe our visualization techniques, with a focus on visual encodings for time and time-oriented network data; in Chapter 5) we describe our interaction techniques, including smooth animated transitions between different visual encodings, an interactive control of graph layout stability as well as a novel interaction named vertigo zoom. In Chapter 6 we provide some details on the prototypical implementation of our approach.

Afterwards, we present the **validation** of our approach. In Chapter 7 we introduce a usage scenario and illustrate how our approach can help us gaining insights about a real-world dataset. In Chapter 8 we present the findings of expert reviews, involving both visulization experts and domain experts. In Chapter 9) we describe a task-based qualitative evaluation, conducted with target users, and discuss usability and utility of our approach. In Chapter 10) we report a comparative evaluation of selected interaction techniques by a controlled task-based quantitative user study. We drive conclusions in Chapter 11, by discussing answers to our research questions as well as future research directions.

CHAPTER 2

Foundations and state of the art

In this chapter we introduce concepts and formal definitions for time-oriented networks and discuss state-of-the-art for modelling, analysing, and visualising this kind of data.

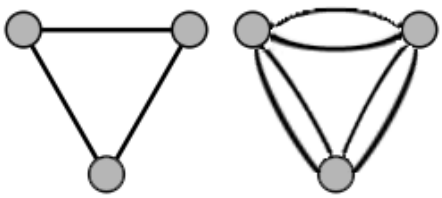
2.1 Concepts and definitions

A **graph** $G := (V, E)$ consists of a finite non-empty set V of elements named **vertices** (or **nodes**) together with a prescribed set $E \subseteq N \times N$ of unordered pairs of elements of V, named **edges** (or **links**) [185].

Each edge $e_{i,j}$ is associated with two vertices i and j by a relation of **incidence**. The edge $e_{i,j}$ is said to be **incident** into the vertices i and j; the vertices i and j are said to be the **ends** of the edge $e_{i,j}$. Moreover, the vertices i and j of the edge $e_{i,j}$ are said to be **adjacent**. The set E of edges induces a relation of **adjacency** on the set V of vertices. An edge $e_{i,i}$, which has the same vertex i as ends, is named a **loop** (Figure 2.3a).

If only one edge is allowed for each pair of vertices and there are no loops, then the graph is a **simple graph**; otherwise, the graph is named a **multigraph**, or a **pseudograph** (see Figure 2.1).

A **directed graph**, or **digraph**, is a graph whose edges are directed (i.e., the edges are ordered pairs of vertices). Directed edges are also named **arcs**. For each arc, we distinguish a **source** (or **origin**, or **tail**) vertex, and a **destination** (or **head**) vertex. The arc is said to **emanate**

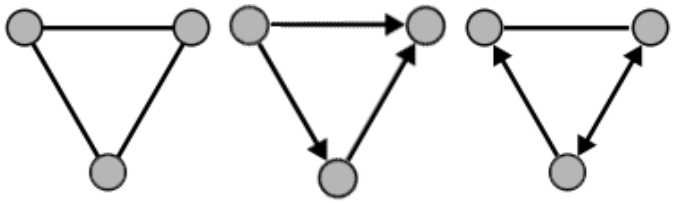


(a) Simple graph (b) Multigraph

Figure 2.1: Different graphs according to edge multiplicity

or **originate** from its source, and to **terminate** into its destination. The arcs emanating from a vertex are referred to as its **outgoing** arcs; the arcs terminating into a vertex are referred to as its **incoming** arcs.

Conversely, a graph properly said (i.e., the edges are unordered pairs of vertices), is also called **undirected**. A graph whose edges can be both directed and undirected is referred to as **mixed** (see Figure 2.2).

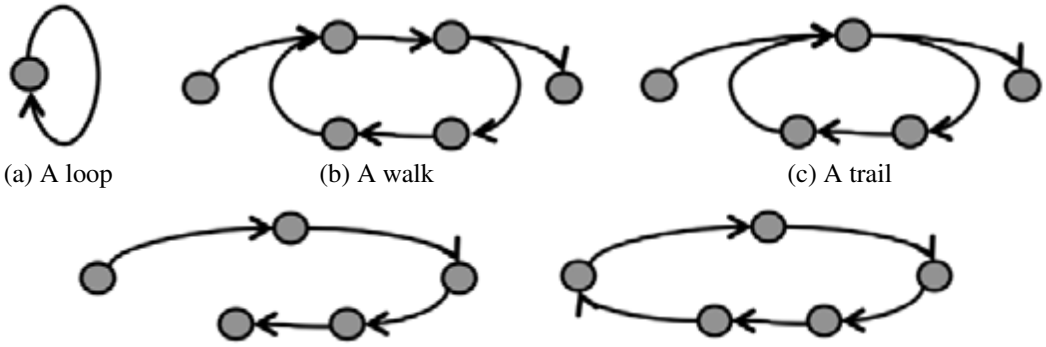


(a) Undirected (b) Directed (c) Mixed

Figure 2.2: Different graphs according to edge direction

A **walk** of **length** k is an alternating sequence of vertices and edges $v_0, e_0, v_1, e_1, ..., v_{k-1}, e_{k-1}, v_k$ (Figure 2.3b). If the graph is undirected, then the endpoints of e_i are v_i and v_{i+1}. If the graph is directed, then e_i is an arc from v_i to v_{i+1}. A **trail** is a walk in which all edges are distinct (i.e., a walk without edge repetitions, Figure 2.3c). A **path** is a trail in which all vertices, except possibly the first and the last, are distinct (Figure 2.3d).

A **cycle** is a path in which the first and the last vertex are not distinct (Figure 2.3e). An **acyclic** graph is a graph without cycles.



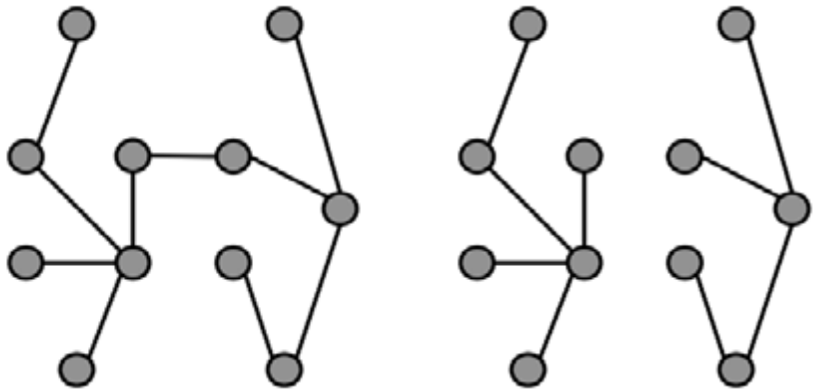
(a) A loop (b) A walk (c) A trail

(d) A path (e) A cycle

Figure 2.3: Graph elements and patterns

Two vertices are said to be **connected** if there exist at least one path between them. Two vertices are said to be connected in k steps if there is a path of length k between them and any other path connecting them has length $l > k$.

A graph is **connected** if there is a path between every pair of vertices; otherwise, the graph is **disconnected**. A **connected component** (or simply component) is a maximal connected subgraph of a graph (Figure 2.4).



(a) A connected graph

(b) A disconnected graph with two connected components

Figure 2.4: Graph connectivity

A **tree** is an acyclic connected graph. A **rooted tree** is a tree in which a given vertex is designated as the **root**. In a rooted tree, edges can be assigned a natural orientation, either towards or away from the root. Rooted trees are a very common class of graphs, which is well suited to model hierarchical data.

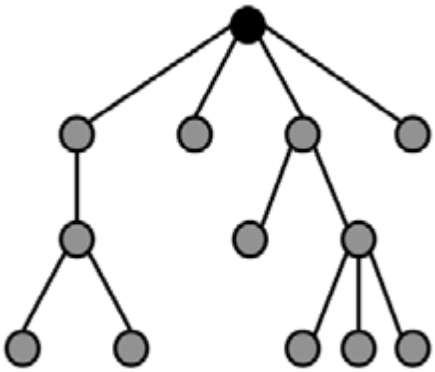
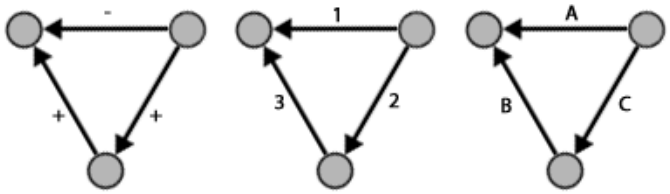


Figure 2.5: A rooted tree

A **network** is defined as a graph or directed graph together with a function which assigns a positive real number (often named **weight**) to each edge, representing one amongst capacity, cost, latency, or other properties, depending on the application [185]. One positive real number may not be enough for representing complex real-world data and, therefore, a network can be generalised by associating several attributes (i.e. key-value pairs) of any type (e.g., sign, integer, floating, string, see Figure 2.6) to vertices and edges; such a network is commonly referred to as a **multivariate** network [309, 404].

The attributes, which are associated to vertices and edges in a multivariate network, can be used to define partitions on them; therefore, we can define **multi-modal** networks (i.e., with

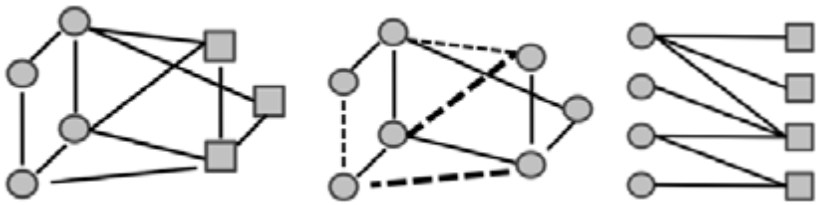


(a) Signed (b) Weighted (c) Labelled

Figure 2.6: Graphs with different edge attributes

several types of vertices, Figure 2.7a) as well as **multi-relational** networks (i.e., with several types of vertices, Figure 2.7b).

In particular, a **bipartite** graph is a graph in which the set of vertices is divided in two partitions and each edge has one end belonging in either partition (Figure 2.7c). A **compound graph** can be seen as a graph in which the set of edges is divided in two partitions, one of which induces a tree structure onto the vertices; in other words, a compound graph is a graph with a hierarchical partition of its vertices.



(a) Multimodal (b) Multirelational (c) Bipartite

Figure 2.7: Networks with partitions of vertices or edges

2.1.1 Time and time-oriented networks

Real-world phenomena are dynamic, since they evolve over time, and a (static) multivariate network does not capture their dynamics; thus, several concepts of **dynamic** (or **temporal**, or **time-oriented**) networks have been introduced. Harary [186] defines five types of dynamic (univariate) networks: in a node-dynamic network the set of vertices varies over time, in an edge-dynamic network the set of edges varies over time, in a vertex-weighted dynamic network also the weights of vertices vary over time, in an in edge-weighted dynamic network also the weights of edges vary over time, in a full dynamic graph the sets of vertices and edges and their weights vary over time.

Carley [87] defines a dynamic network as an overtime sequence of static networks. In both the event-based [31] and the streaming paradigm [427], a dynamic network is seen as a sequence of edges over time.

Casteigts et al. [88] survey different types of temporal networks in several application domains, and formalise different perspectives: the vertex-centred perspective (sequence of vertex properties), the edge-centred perspective (sequence of edges) and the graph-centred perspective

(sequence of networks). Moreover, they propose the time-varying graph (TVG) as a unified representation based on two timing functions: the presence function, which indicates whether a given edge is available at a given time, and the latency function, which indicates the time it takes to cross a given edge if starting at a given time.

Holme et al. [198] draw on the presence function, and distinguish two classes of temporal networks, the contact-sequence class and the interval-graph class, depending on whether the values of the presence function are time instants or time intervals. In both cases, a static network can be obtained as an aggregation over time. Instants and intervals as time primitives will be addressed in the following section.

Archambault et al. [20] define a temporal multivariate network as a multivariate network whose attributes change over time. They identify three temporal aspects: structure (the topology of the underlying network at a given time), behaviour (attribute changes), and evolution (structure changes). Moreover, their model includes a notion of influence, namely a functional relation between network elements and attributes, e.g., an attribute represents a graph-theory property derived from the network structure.

As observed also by Archambault et al. [20], temporal networks can be understood as a particular class of time-oriented data; as such, their analysis and visualization design has to consider the characteristics of time and time-oriented data.

Aigner et al. [14] identify a number of orthogonal design aspects and abstractions for modelling time-oriented data. Design aspects comprehend scale, scope, arrangement, and viewpoint.

The scale can be ordinal, discrete, and continuous, depending on whether time values are mapped to elements of a (partially) ordered set, to integers, or to real numbers. The scope refers to basic elements of the time structure, namely points and intervals; the former have no length, while the latter have a temporal duration. The arrangement can be either linear (time continuously flowing from before to after) or cyclic (recurring time values, such as alternating days and nights or consecutive seasons). Viewpoint can be ordered, branching, or with multiple perspectives: in an ordered time domain, any time value is put before or after another, or eventually in concurrence for partially ordered domains; branching time models admit simultaneous timelines, understood as possible alternatives deriving from decision points or branching events; multiple perspectives arise when data can be mapped to multiple time values representing different viewpoints of the same phenomenon.

Additionally, several abstractions can be used for representing and reasoning about time, namely granularities, calendars, time primitives, and indeterminacy.

Granularities are human abstractions (e.g., hours, days, weeks, months) useful to handle the complexity of time and refer to it in a simpler way. Granularities can by grouped in several ways (e.g., hours are grouped into days, days are grouped into weeks and months), according to a given calendar.

Time primitives are the basic elements to relate data to time, such as instants, intervals, and spans. They are analogous but not to be confused with points and intervals of the time scope: primitives can be understood as an intermediate layer for mapping data to either a point-based or interval-based time domain. Instants and intervals are anchored primitives referring to the time domain, while spans represent unanchored durations.

A time-oriented dataset can have either an internal time or an external time. A time series,

for example, is a dataset that describes the history of a certain phenomenon with respect to an inherent time; conversely, streaming data are changing over time depending on external data.

2.2 Data models

The performances of any algorithm for accessing, analysing, and visualising graph data, as well as any kind of date, can be largely increased when the data are represented with an appropriate data model and managed with efficient data structures. Data structures specifically designed for graph data include incidence matrices, adjacency matrices, adjacency lists, and stars [11, 353].

Let us introduce a graph example (Figure 2.8) in order to illustrate how it can be represented by using different data structures.

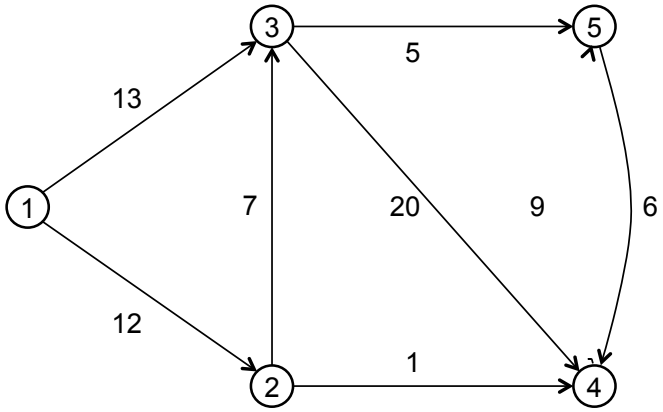


Figure 2.8: A graph example

2.2.1 Incidence matrix

The vertex-edge incidence matrix representation, or simply the incidence representation, stores a graph with n vertices and m edges in a $m \times n$ matrix, in which each row represents a vertex and each column represents an edge.

Each column has only two non-null values, corresponding to the end vertices. Therefore, the incidence matrix is not a space-efficient representation and is hard to manipulate. Moreover, it cannot store edge attributes.

2.2.2 Adjacency matrix

The vertex-vertex adjacency matrix representation, or simply the adjacency matrix representation, stores a graph with n vertices in a $n \times n$ matrix $A = \{a_{i,j}\}$. Each element $a_{i,j}$ of the matrix represents the edge (i, j). Each attribute of the edge can be stored in an additional $n \times n$ matrix.

	(1,2)	(1,3)	(2,3)	(2,4)	(3,4)	(3,5)	(4,5)	(5,4)
1	1	1	0	0	0	0	0	0
2	-1	0	1	1	0	0	0	0
3	0	-1	-1	0	1	1	0	0
4	0	0	0	-1	-1	0	1	-1
5	0	0	0	0	0	-1	-1	1

Figure 2.9: The incidence matrix for our example graph

The **density** $D(G)$ of a graph G is defined as the ratio between the number of edges and the maximal number of edges:

$$D(G) = \frac{2|E|}{|V|(|V|-1)}$$

According to the density, we can distinguish dense graphs (i.e., graphs with many edges) and sparse graphs (i.e., graphs with few edges). The adjacency matrix representation is well-suited for dense graphs but is not space-efficient for sparse graphs. Moreover, it cannot represent multigraphs. Figure 2.10 shows the adjacency matrix representation for the graph we introduced at the beginning of this section.

	1	2	3	4	5
1	0	1	1	0	0
2	0	0	1	1	0
3	0	0	0	1	1
4	0	0	0	0	1
5	0	0	0	1	0

(a)

	1	2	3	4	5
1	0	12	13	0	0
2	0	0	7	1	0
3	0	0	0	20	5
4	0	0	0	0	9
5	0	0	0	6	0

(b)

Figure 2.10: The adjacency matrix (a) and the attribute matrix (b) for our example graph

2.2.3 Adjacency lists

There are two types of adjacency list representations: edge adjacency lists and vertex adjacency lists. The edge adjacency list $A_e(i)$ of a given vertex i is a list of all edges that have i has an end.

$$A_e(i) = \{(i,j)\} \mid j \in V, (i,j) \in E$$

The vertex adjacency list $A_v(i)$ of a given vertex i is the set of vertices adjacent to i,

$$A_v(i) = \{j\} \mid j \in V, (i,j) \in E$$

At level of data structure, both the edge adjacency list and the vertex adjacency list representations can be implemented as linked lists (Figure 2.11), providing an efficient and dynamic resource allocation [284]. These representations, therefore, are efficient for both space occupation and data management, and are well suited for different kinds of graphs.

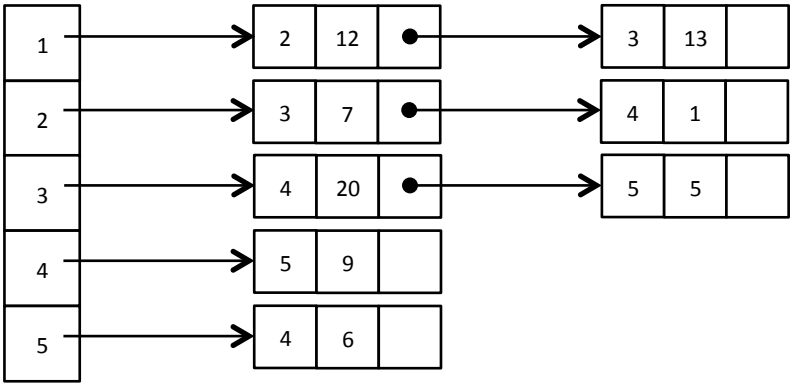


Figure 2.11: The edge adjacency list for our example graph, implemented as a linked-list

2.2.4 Edge list

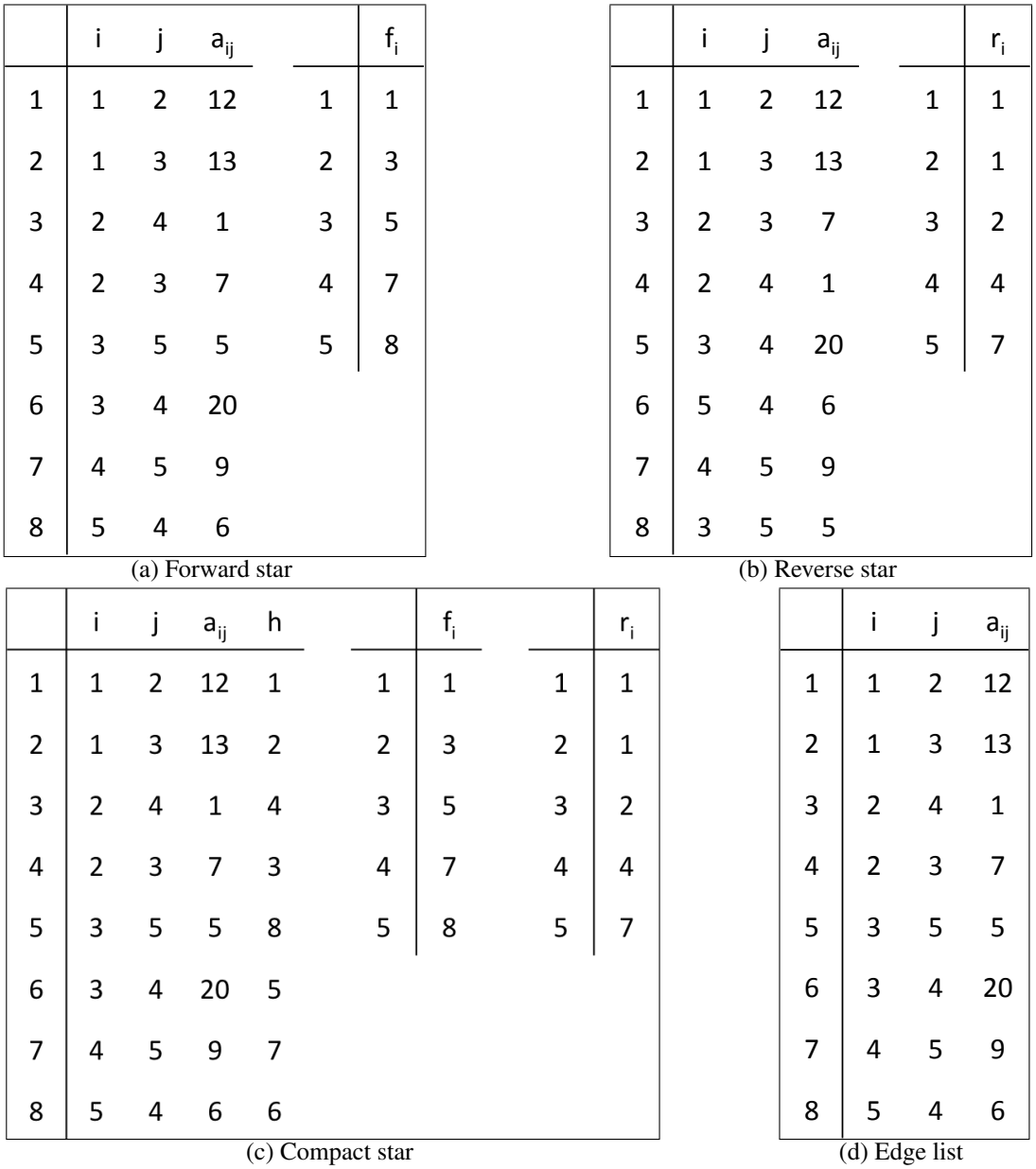
Adjacency matrices and lists are used as both in-memory data representation and storage format; however, edge lists are representations used by common file formats for sharing graph data (such as GraphML, GML, GraphSON, OGML) [329, 330]. An edge list representation is actually a two-dimensional array in which each row represent an edge, and two columns represent the edge ends; additional columns can store edge attributes (Figure 2.12d). Edge lists are an intuitive and space efficient representation of graph data, but they are not efficient to manipulate and do not support efficient computation algorithms.

2.2.5 Forward star, reverse star, and compact star

Star representations are based on edge lists, complemented by index arrays to improve manipulation efficiency. They are designed for directed graphs; however, undirected graphs can be represented as directed graphs through arbitrary assignment of directions to every edge [119].

In particular, a forward star representation (Figure 2.12a) consists of a two-dimension array (i.e., the edge list) and a one-dimension array $F = \{f_i\}$ with an element for each vertex. The element f_i is defined as the row index (in the edge array) of the first arc that emanates from a vertex greater than $i - 1$.

Analogously, the reverse star representation (Figure 2.12b) consists of the two-dimension array and a one-dimension indexing array $R = \{r_i\}$. In this case, each element r_i is defined as the row index of the first arc that terminates in a vertex greater than $i - 1$.



	i	j	a_{ij}
1	1	2	12
2	1	3	13
3	2	4	1
4	2	3	7
5	3	5	5
6	3	4	20
7	4	5	9
8	5	4	6

	f_i
1	1
2	3
3	5
4	7
5	8

(a) Forward star

	i	j	a_{ij}
1	1	2	12
2	1	3	13
3	2	3	7
4	2	4	1
5	3	4	20
6	5	4	6
7	4	5	9
8	3	5	5

	r_i
1	1
2	1
3	2
4	4
5	7

(b) Reverse star

	i	j	a_{ij}	h
1	1	2	12	1
2	1	3	13	2
3	2	4	1	4
4	2	3	7	3
5	3	5	5	8
6	3	4	20	5
7	4	5	9	7
8	5	4	6	6

	f_i
1	1
2	3
3	5
4	7
5	8

	r_i
1	1
2	1
3	2
4	4
5	7

(c) Compact star

	i	j	a_{ij}
1	1	2	12
2	1	3	13
3	2	4	1
4	2	3	7
5	3	5	5
6	3	4	20
7	4	5	9
8	5	4	6

(d) Edge list

Figure 2.12: Edge list and star representations for our example graph

Forward stars and reverse stars are alternatively used by different algorithms. Algorithms needing the indexing of both incoming and outgoing arcs can use a compact star representation, which is a combination of the previous two (Figure 2.12c). The data array is complemented with two index arrays, i.e. the forward array and the reverse array; moreover, the edge-list array has one more column representing the index of the reverse array.

2.3 Automated analysis

Graph theory [185] and network science [39, 285] provide a large number of measures for automated analysis of network data. These measures constitute the mathematical foundations for different specific properties which can be applied in specific domains, spanning from natural sciences to engineering, from life sciences to social sciences.

Network measures can be divided into two main categories, namely local measures and global measures. Global measures are designed to provide an overall characterization of a network, whereas local measures are used to analyse single network elements such as edges and vertices.

2.3.1 Degree centrality

The degree centrality is a common local measures for vertices.

Given a graph $G = (V, E)$, the **degree** (or **valency**) of a vertex $v \in V$ is the number of edges incident to the vertex, with loops counted twice. The degree of v is denoted $deg(v)$ or $degv$. The **degree centrality** of a vertex v is defined as

$$C_D(v) = deg(v)$$

Given a digraph $G = (V, A)$, the **indegree** of a vertex $v \in V$ is the number of adjacent heads ends, while the **outdegree** of v is the number of adjacent tail ends. The indegree of v is denoted $deg^-(v)$ and its outdegree is denoted $deg^+(v)$. The **indegree centrality** of v is defined as

$$C_D^-(v) = deg^-(v)$$

The **outdegree centrality** of v is defined as

$$C_D^+(v) = deg^+(v)$$

2.3.2 Degree distribution

The degree distribution $P(k)$ of a network is the fraction of nodes in the network with degree k. If in a network there are n nodes and n_k of them have degree k, the degree distribution is defined as

$$P(k) = \frac{n_k}{n}$$

It has been observed that the degree distribution of many real-world networks follows a **power law distribution** [95]

$$P(k) \sim k^{-\gamma}$$

where γ is a real number with usually $2 < \gamma < 3$. Such networks are referred to as **scale-free** networks [40]. Their behaviour has been modelled in terms of **preferential attachment**, or **cumulative advantage** [310]: during the network evolution, the vertices with a higher degree attract more edges and, therefore, their degree is increased faster.

2.3.3 Geodesic distance and characteristic path length

Given a graph $G = (V, E)$ and two vertices $u \in V$ and $v \in V$, the **geodesic distance**, or simply the distance, $d(u, v)$ is the length of any shortest path between them. It is worth noting that in a digraph $G = (V, A)$, the geodesic distance is not a metric (since symmetry and non-negativity conditions are not satisfied).
The **characteristic path length** $l(G)$ is the average distance between all pairs of vertices:

$$l(G) = \frac{1}{n(n-1)} \sum_{u \neq v} d(u, v)$$

2.3.4 Eccentricity

The **eccentricity** $\epsilon(v)$ of a vertex v is the greatest geodesic distance between v and any other vertex:

$$\epsilon(v) = \max_{u \in V} d(u, v)$$

The **radius** $r(G)$ of a graph G is the minimum eccentricity of any vertex:

$$r(G) = \min_{v \in V} \epsilon(v)$$

The **diameter** $d(G)$ of a graph G is the maximum eccentricity of any vertex:

$$d(G) = \max_{v \in V} \epsilon(v)$$

A **central vertex** is a vertex with eccentricity equal to the radius:

$$\forall v \mid \epsilon(v) = r$$

A **peripheral vertex** is a vertex with eccentricity equal to the diameter:

$$\forall v \mid \epsilon(v) = d$$

2.3.5 Betweenness centrality

Given a graph $G(V, E)$ and three vertices $u, v, w \in V$, let $n_{u,w}$ be the number of shortest paths between u and w and $n^v_{u,w}$ the number of shortest paths between u and w passing through v. The **betweenness centrality** $C_B(v)$ of the vertex v is defined as [152]:

$$C_B(v) = \sum_{u \neq v \neq w} \frac{n^v_{u,w}}{n_{u,w}}$$

2.3.6 Closeness centrality

Given a graph $G(V, E)$ and a vertex $v \in V$, let $l(v)$ be the average distance of v, defined as:

$$l(v) = \frac{1}{n} \sum_{u \in V} d(u, v)$$

where n is the number of vertices in the graph. The **closeness centrality** $C_C(v)$ of the vertex v is defined as the inverse of its average distance [43]:

$$C_C(v) = \frac{1}{l(v)} = \frac{n}{\sum\limits_{u \in V} d(u, v)}$$

2.3.7 Eigenvector centrality

Let $A = \{a_{ij}\}$ be the adjacency matrix of a given graph (see Section 2.2). The **eigenvector centrality** $C_C(v)$ of a vertex v is the element x_v of the left-hand eigenvector x associated with the eigenvalue λ [255, 346]:

$$C_E(v) = x_v \mid \lambda x = xA$$

2.3.8 Clustering coefficient

The **clustering coefficient** is both a local and a global measure for graphs [405]. Let the **neighbourhood** N_v for a vertex v be defined as the set of vertices connected to v in one step:

$$N_v = \{u \in V : (u, v) \in E\}$$

Let k_v be the number of neighbours of v:

$$k_v = |N_v|$$

Then, there are at most $k_v(k_v - 1)/2$ edges between them (the maximal occurs when all neighbours of v are connected to each other in one step). The clustering coefficient $CC(v)$ of the vertex v is defined as the fraction of these allowable edges that actually exist:

$$CC(v) = \frac{2|\{(u, w) \in E : u, w \in N_v\}|}{k_v(k_v - 1)}$$

The clustering coefficient $CC(G)$ of the graph G is the average clustering coefficient over all vertices:

$$CC(G) = \frac{1}{n} \sum_{v \in V} CC(v)$$

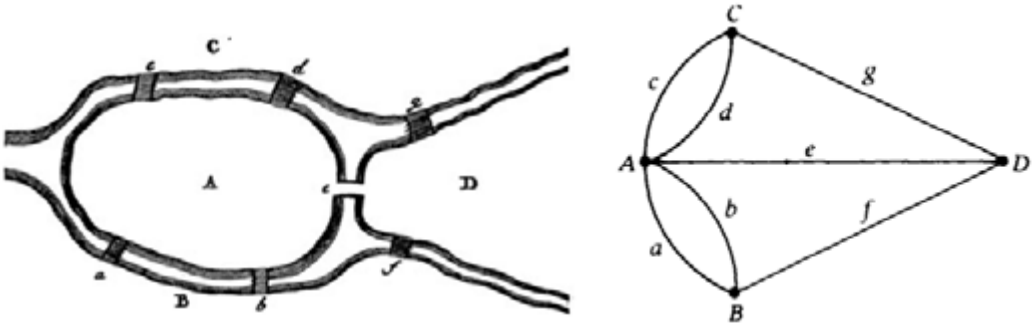
Many real-world networks have a small characteristic path length and a high clustering coefficient [405]; in other words they exhibit a so-called small-world effect [371].

2.3.9 Automated analysis of time-oriented graphs

While several other metrics have been defined and are widely used for automated analysis of static networks [5, 402], dynamic metrics are a relatively new research field. In recent years several extension of existing static metrics to the dynamic case have been proposed, as well as a few inherently dynamic ones.

The time-scale degree centrality has been defined as an extension of the static degree centrality that takes into account both the presence and duration of links [375]. Lerman et al. [256] introduce an attenuation factor to the link duration, and on this basis define a centrality metric for dynamic networks. By modelling social interactions as temporal events and taking into account when they occur, Berger-Wolf and Saia [53] introduce a framework consisting in several metrics for the analysis of dynamic social networks. Other approaches consider the time series of static centrality metrics, measured over subsequent time windows (e.g. daily or monthly), and compute basic statistics [63] [306].

In general, new metrics have been proposed for multidimensional (i.e., multivariate) networks, where the attributes of nodes and links chosen as dimensions are not necessarily referring to the time, but for example to the type of relation in multi-relational networks. Berligerio et al. [54] extend the degree centrality from the monodimensional to the multidimensional case and introduce new metrics, namely the dimension relevance, the dimension connectivity and the dimension correlation. Brodka et al. [73] define a cross-layer degree, that is an aggregated node centrality measure across the relations of a multi-relational network.



(a) The map sketched by Euler in 1736 to illustrate the seven bridges of Königsberg problem [130]

(b) Euler's problem as abstracted and graphically represented by Ball in 1884 [38]

Figure 2.13: Two drawings of one of the most famous graphs

2.4 Historical graph visualization

The earliest publication on graph theory is Euler's 1736 paper on the problem of the seven bridges of Königsberg [130]; since then, a trail which visits every edge exactly once is known as an Eulerian trail. Euler's work is generally recognized [409] the foundational work establishing graph theory as a new discipline (now considered a branch of combinatorics). Nevertheless,

it did not establish graph drawing or graph visualization as well, since Euler only sketched a map of Königsberg (Figure 2.13a) but did not draw any graph representaion of the problem, which was treated in pure mathematical (i.e., non-graphical) terms. It was only in 1884 that Ball [38] published a graphical representation of Euler's problem (Figure 2.13b) in a book about recreational mathematics.

Game and puzzle boards however are probably the earliest examples of graph drawings appeared in human history. In merels boards (i.e., graphs on carved or painted boards to play Morris or Mills games, dating back to the Roman empire, see Figure 2.14), vertices and edges represent allowed positions and moves, respectively [52].

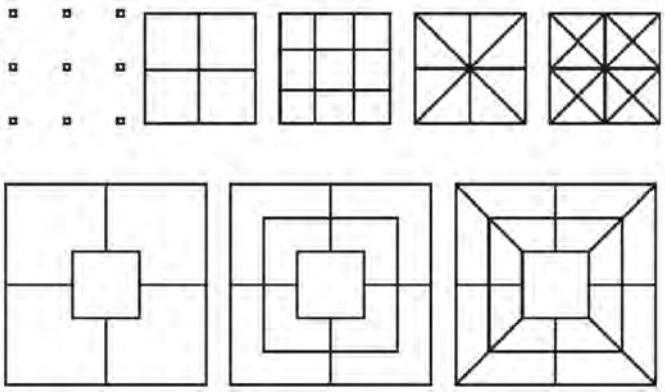


Figure 2.14: Graphs drawings on merels boards dating to Roman empire [52]

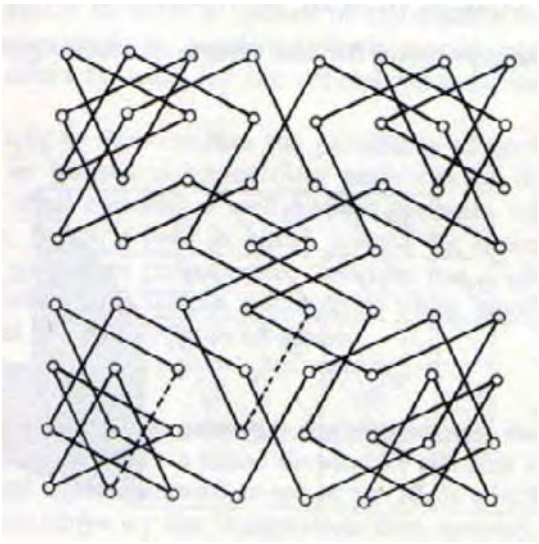
Graphs have been used since the Middle Ages to show relationships between logical propositions, philosophical principles, or concepts. Figure 2.15a shows two diagrams drawn by the Catalan philosopher and writer Ramon Lull: the left-hand graph represents sixteen divine attributes, the right-hand graph represents the seven virtues intertwined with the seven vices [408].

In modern times, graphs have been used to play games and puzzles like the aforementioned Eulerian trail problem, or the analogous problem of finding a Hamiltonian path (i.e., a path that traverses each vertex in a graph exactly once). A similar puzzle, which can be played on a chess board, is the so-called knight's tour, consisting in finding a legal sequence of knight's moves that passes on each square exactly once. This knight's tour problem was first formulated in mathematical terms by Euler in 1756, and then visually represented by a graph drawing by Vandermonde in 1771 (see Figure 2.15b).

Besides recreational mathematics, graph diagrams have been extensively used in natural science and engineering [245], for example to represent electric circuits (Figure 2.16a) and chemical structures (Figure 2.16b).

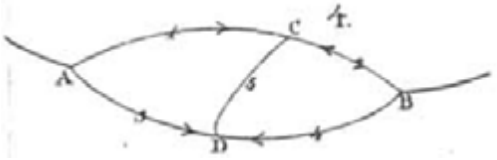
In 1934, the Austrian-American sociologist Jacob L. Moreno introduced graph drawings to visualize social network. His so-called *sociograms* (Figure 2.17) are multi-modal, multi-relational, mixed graph, where nodes represent persons (differentiated by gender), and edge represent social relations, both directed (e.g., attraction, repulsion) and undirected (mutual attraction). Since then, graph drawings have been commonly used to analyze and visualize social networks [50, 151, 402].

(a)

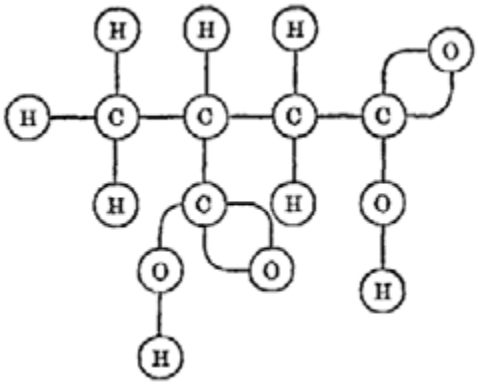


(b)

Figure 2.15: (a) Two graph drawings by Ramon Llull (1280) showing relationships between concepts (from [408]) (b) The graph drawn in 1771 by Vandermonde [388] representing the knight's tour



(a)



(b)

Figure 2.16: Graphs are used in natural sciences and engineering: (a) an electric circuit sketched in 1845 by Kirchhoff [236]; (b) a molecule drawn in 1865 by Brown [75].

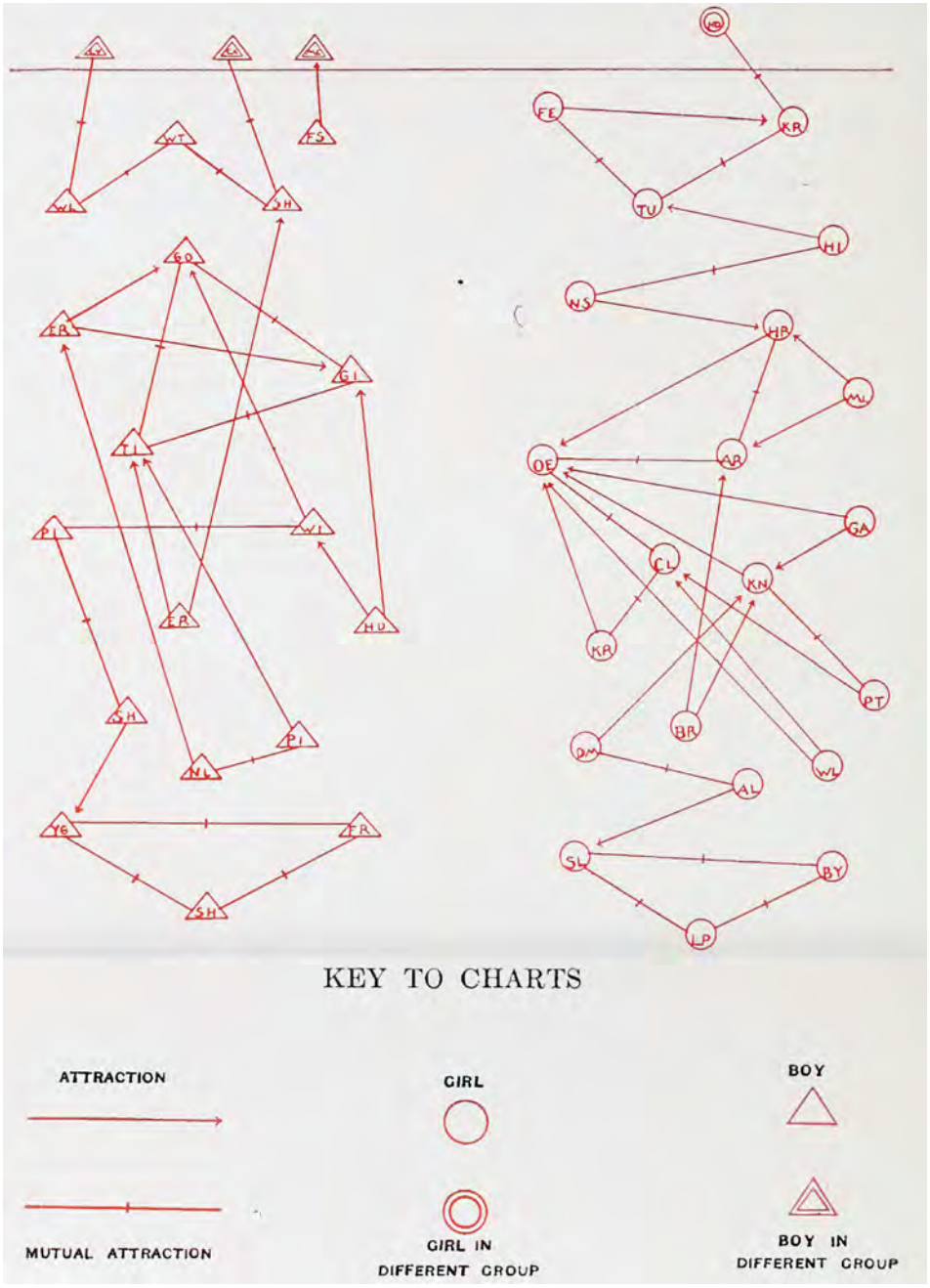


Figure 2.17: Moreno's sociogram [276], the first graph drawing representing a social network

2.5 Graph visualization: surveys, taxonomies, design spaces

After the early approaches that have been developed to address specific problems in diverse application domains, graph visualization has become a relevant research topic that has been systematically explored by specific disciplines, such as graph drawing and information visualization. Di Battista et al. surveyed a large number of algorithms for graph drawing, i.e. "the visual representation of graphs in the plane", where "the vertices are represented by symbols such as circles or boxes, and each edge is represented by a simple open curve between the symbols associated with the vertices" [105]. Their annotated bibliography encompasses algorithms for specific graph classes (e.g., free and rooted trees, planar graphs, directed acyclic graphs), as well as specific graph standards such as polyline, straight-line, orthogonal or visibility representations. In a polyline, straight-line, or orthogonal drawing, each edge is represented by a polygonal chain, a straight-line, or a chain of horizontal and vertical segments, respectively; in a visibility representation, vertices are represented by horizontal segments and edges by vertical segments, so that each edge-segment intersects exactly the vertex-segments associated with its end vertices, and no other vertex-segment. Figure 2.18 shows an overview of algorithms for graph drawing. Other surveys of graphs drawing algorithms appeeared more recently [107, 170, 237].

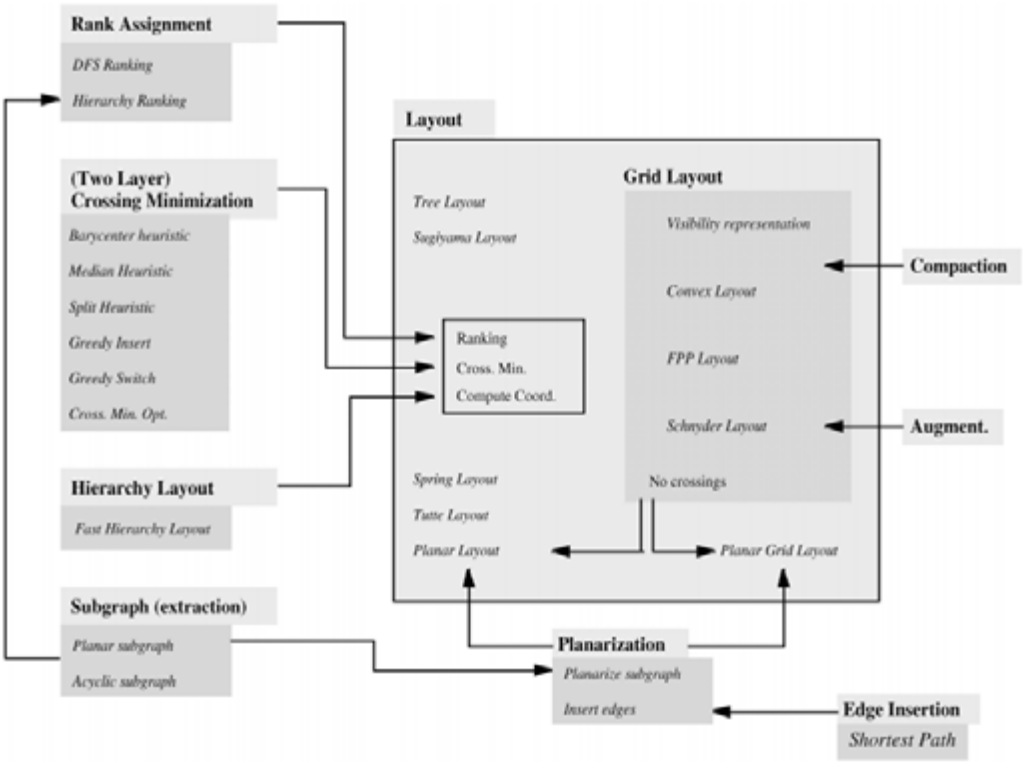


Figure 2.18: Algorithms for graph drawing (in [195], adapted from [281])

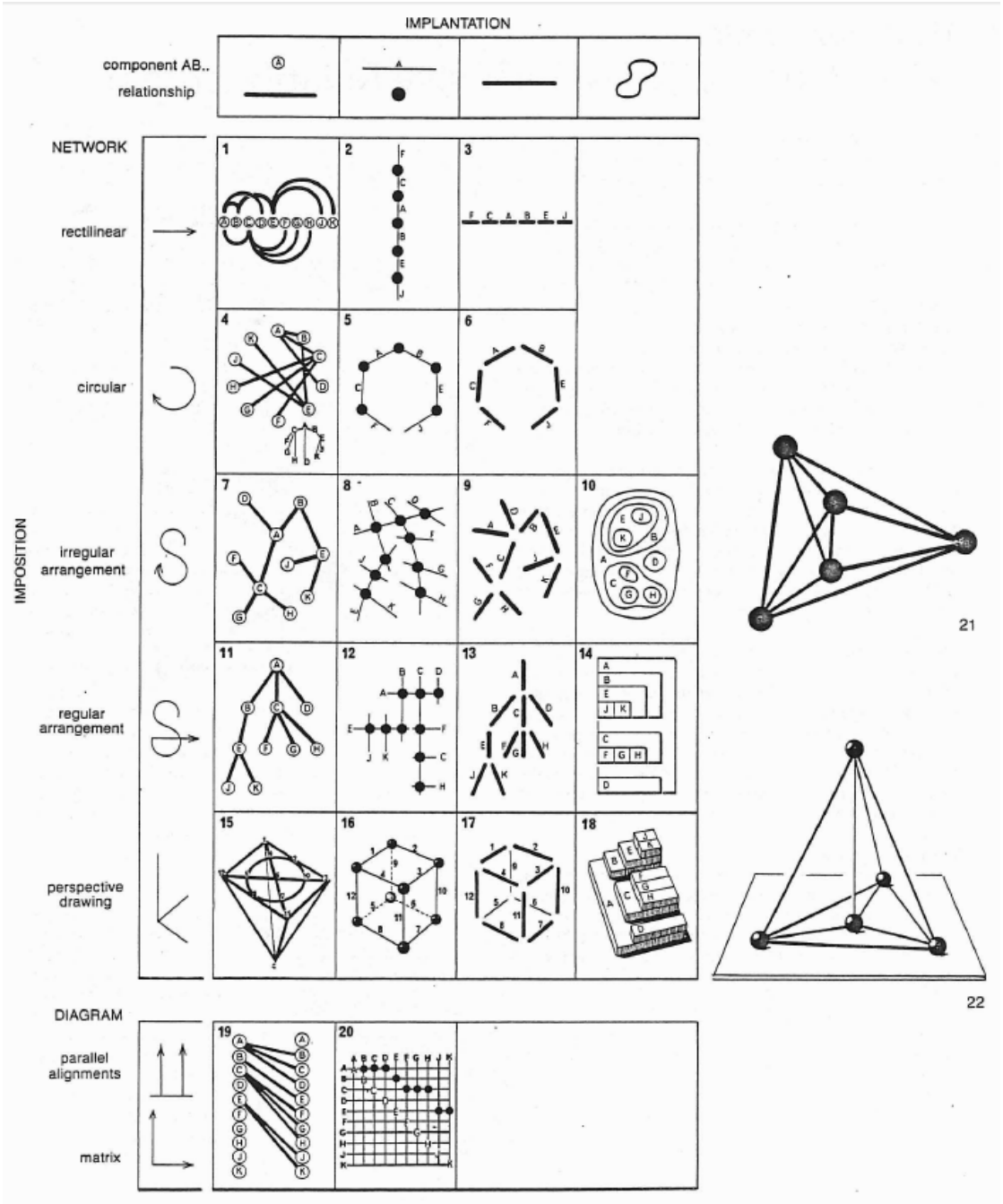


Figure 2.19: Visual encodings for graphs according to Bertin [55]

However, node-link diagrams (e.g., graph drawings where vertices are represented by symbols and edges by a simple open curve between the symbols) are only one of many possible visual encodings for graph data. Bertin [55] lists twenty-two visual encodings, which he divides in network and diagrams. What Bertin calls networks are visual representation in which each vertex and each edge is represented only once (e.g., node-link diagrams). Conversely, Bertin defines a diagram as a visual representation in which each vertex is represented twice, as the two ends of an edge; for example, matrices and parallel alignments (also known as bipartite layouts, to not be confused with bipartite graphs). Bertin categorizes networks in terms of his semiology of graphics (Figure 2.19), by distinguishing between planar variables (namely, implantation and imposition) and retinal variables. Implantations are classes of geometrical primitives: points, lines, and areas. Impositions are arrangements of these geometrical primitives in the space, i.e., types of layout. Retinal variables are six additional features of implantations, such as size, hue, saturation, shape, orientation, and texture. According to implantations, Bertin distinguishes four types of graphics for graphs: points represent vertices and lines represent edges, point represent edges and lines represent vertices, lines represent edges (and vertices are not visualized explicitly), areas represent both edges and vertices. As for impositions, Bertin distinguishes: rectilinear, circular, irregular, regular, and perspective drawings. It is also worth noting that not implantations are well suited for all graphs, and some are specifically designed for graph sunclasses; area inclusions (also known as containment, or space-filling techniques), for example, are commonly used for visualizing hierarchies and other tree structures.

Herman et al. [195] survey graph visualization and navigation in information visualization. Beside encompassing a larger spectrum of visual encodings than node-link diagrams only, their survey emphasizes interaction as a core component of information visualization, comprehending techniques such as zoom and pan, focus+context, incremental exploration, and clustering.

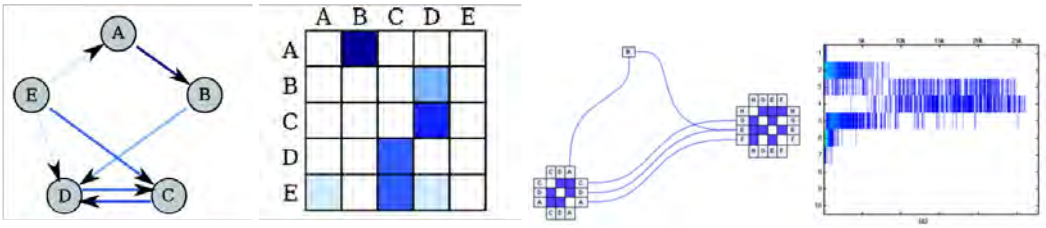
Treevis.net [342] focuses on information visualization techniques tailored for trees and characterizes them along three axis: the layout dimensionality (2D, 3D, and hybrid), the node alignment (free, radial, axis-parallel), and the edge encoding. The latter can be explicit (i.e., each edge is represented by an open curve) or implicit. Schulz et al. [343] explore in detail the design space of implicit tree visualization and identify three types of representation, namely implicit layout by inclusion, implicit layout by adjacency, and implicit layout by overlap.

The survey by von Landesberger et al. [397] lists techniques for visual analysis of large graphs and describes how they address scalability; they report not only about visual graph representation and user interaction, but also about automated graph analysis and the seamless integration of these three different components according to the visual analytics paradigm.

Kerracher et al. [231] map the design space of temporal graph visualization along two axes: structural encoding and temporal encoding. Beck et al. [46] utilize the same dimensions (renamed as time and paradigm, respectively) to survey techniques for visualizing dynamic networks and categorize them into a hierarchical taxonomy. Zaidi et al. [425] provide a short overview about analysis and visualization of dynamic networks. Hadlak et al. [184] survey techniques for multi-faceted visualization, i.e., techniques that are capable of visualizing at once different facets of graph-structured data, such as structure, partitions, attributes, temporal context, and spatial context. Vehlow et al. [390] report on the state of the art of visualizing group structures in graphs. Visualization of multivariate networks is discussed in [233].

2.6 Visual encodings for graphs

In Section 2.5 we have briefly mentioned the wealth of different options for visual encodings of graphs, as explored by Bertin [55]. Nevertheless, if we focus on approaches that are suitable for general graphs (i.e., excluding specific approaches for trees and other graph subclasses), we can distinguish four main categories: node-link diagrams, matrix-based representations, hybrid representations, and other encodings (Figure 2.20).



(a) Node-link (b) Matrix-based (c) Hybrid (d) Other

Figure 2.20: Visual encodings for static (non-temporal) general graphs

2.6.1 Node-link diagrams

A **node-link diagram** is a visual encoding of a graph where vertices are represented by nodes (i.e., points or, better, finite-dimension symbols) and edges are represented by lines (Figure 2.20a). As we have discussed in the previous sections, node-link diagrams are likely the oldest and most common visual encoding for graphs; a node-link diagram is often referred to just as a graph drawing, or even simply a graph. However, in the rest of this work, we will keep to use the word *graph* to refer to the data structure, and the word *node-link diagram* to refer to this specific type of visual encoding. All figures in Section 2.1, as well as Figures 2.13b, 2.14, 2.15a, 2.15b, 2.16a, 2.16b, and 2.17 feature different variations of a node-link diagram. As mentioned in Section 2.5, nodes can be rendered with different retinal properties, to represent vertex attributes, as well as vertex-centric measures (Figure 2.21).

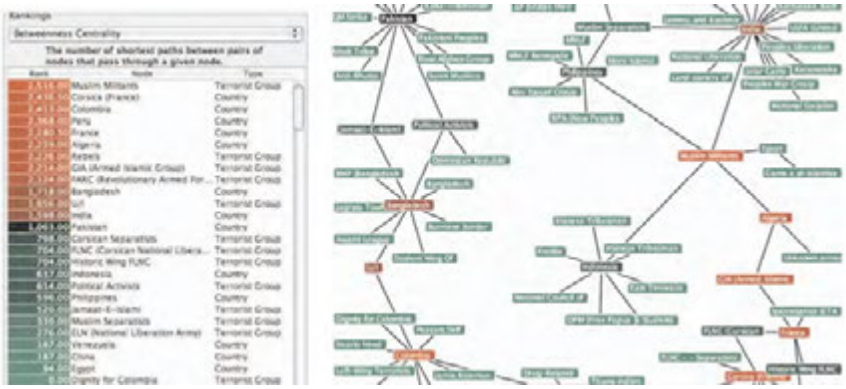


Figure 2.21: A node-link diagram colored according to vertex betweenness centrality [297]

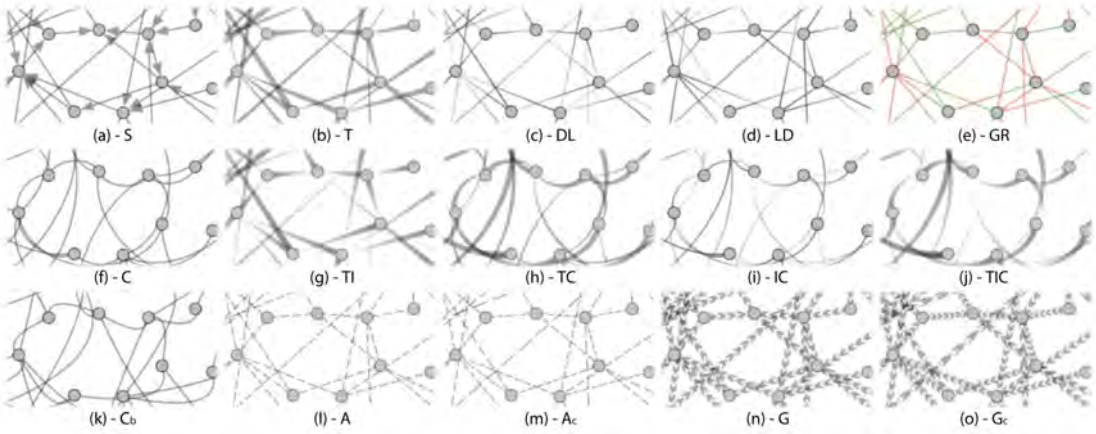
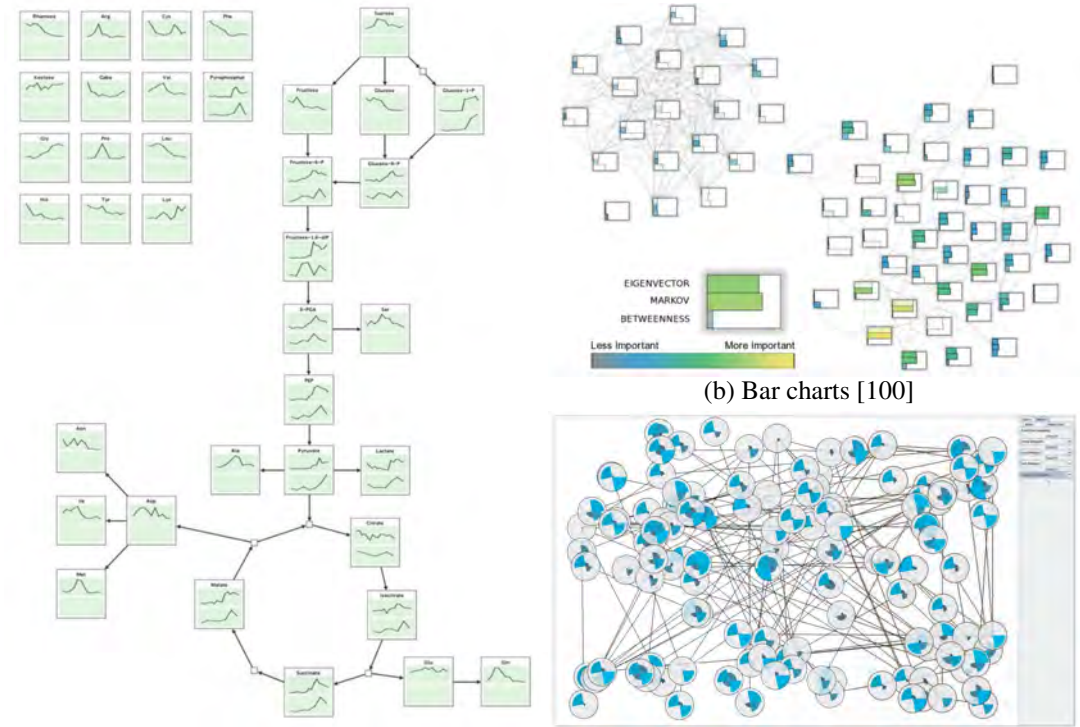


Figure 2.22: Different types of encodings for directed edges in node-link diagrams [200]



(a) Line charts [61]

(b) Bar charts [100]

(c) Polar area charts [271]

Figure 2.23: Multivariate vertex attributes encoded into node-sized diagrams

Furthermore, multivariate data attributes can be visually represented by glyphs or small diagrams, such as line charts (Figure 2.23a), bar charts (Figure 2.23b), or polar area charts (Figure 2.23c). As well as for vertices, there are different possibility for the visual encoding of edges, according to different standards (see also Section 2.5). In graphs with directed edges, in particular, the direction of edges can be represented in several ways. While an arrow shape (featuring an arrow head, an arrow tail, or both) can be considered as the classic representation of a directed edge, alternatives include hue, brightness, curvature, taper, animation, glyphs, as well as several combinations thereof (Figure 2.22).

Besides the choice of the more appropriate visual representation of edges and vertices, two main problems to be addressed in designing node-link diagrams are **layout** and **scalability**.

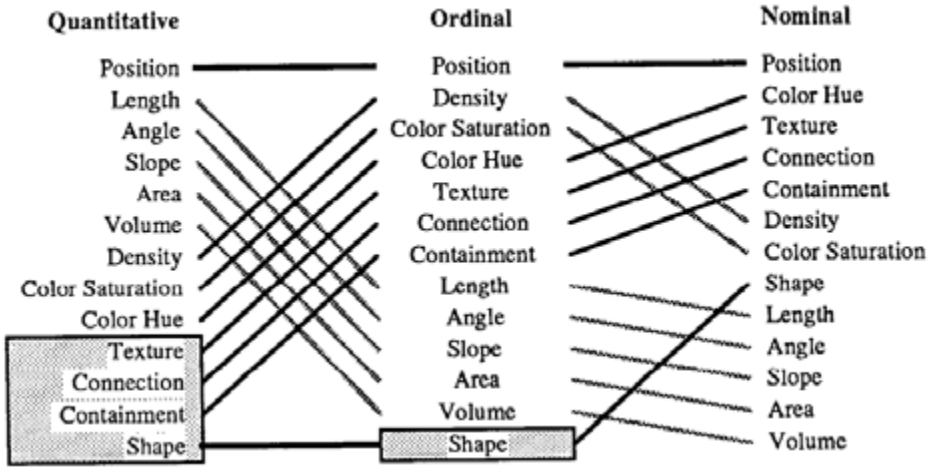


Figure 2.24: Ranking of visual variables according to perceptual accuracy [265]

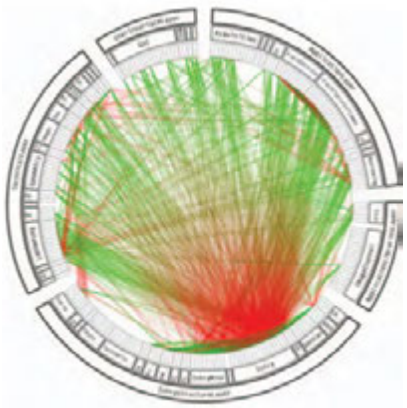
2.6.1.1 Layout

Given a graph, the layout problem consists in assigning positions to its vertices in order to obtain a node-link representation that fulfils specific criteria. This problem is very relevant because position is the visual variable that is perceived with the greatest accuracy. Bertin [55] theorized that position is better suited than his other visual (retinal) variables for encoding all data types (i.e., quantitative, ordinal, and nominal). Cleveland and McGill [96], on the basis of their experimentation with various graph forms, existing psychophysical results, and the theory of psychophysics, hypothesized that position is perceived with more accuracy than length, direction, angle, area, volume, curvature, shading, and color. In particular, by two empirical they provided empirical evidence that human subjects perceive position more accurately than length and angle. Mackinlay [265], building upon existing psychophysical results and analysis of perceptual tasks, extended the rankings by including additional perceptual tasks (e.g., texture), and observed that position is still the most accurate (see Figure 2.24). Therefore, we can certainly state that position of nodes in a node-link diagram is a key factor and affects the overall visualization quality.

The majority of layouts assign position on a plane, e.g. a two-dimensional space. Nevertheless, the simplest layout is the **rectilinear layout**, and it is a one-dimensional layout, where nodes are positioned along a line. Node-link diagrams with a rectilinear layout and curved links (arcs) are known as **arc diagrams** [403] (Figure 2.25c). An arc diagram is well suited for both directed and undirected graphs. For directed graphs, top and bottom arcs can represent the direction (e.g., top arcs represent forward arcs); for undirected graphs, upward concave arcs and downward concave arcs can represent two types of undirected edges. The nodes can be sorted along the line according to a their sequence order (e.g., characters in a string, or notes in a music passage), or any other ordinal attribute.

In a **radial layout**, nodes are positioned along a circle, and links are drawn inside the disc [422]. The position of nodes along the circle can be determined randomly, or according to an additional vertex attribute; in Figure 2.25a, for example, the nodes are grouped along circular arcs according to a hierarchy. A particular radial layout can be used for layered graphs, i.e. graphs where the set of node is divided in ordered partitions. The node are positioned along concentric circles, according to their layer number (Figure 2.25b).

(a) A layered radial layout [422]



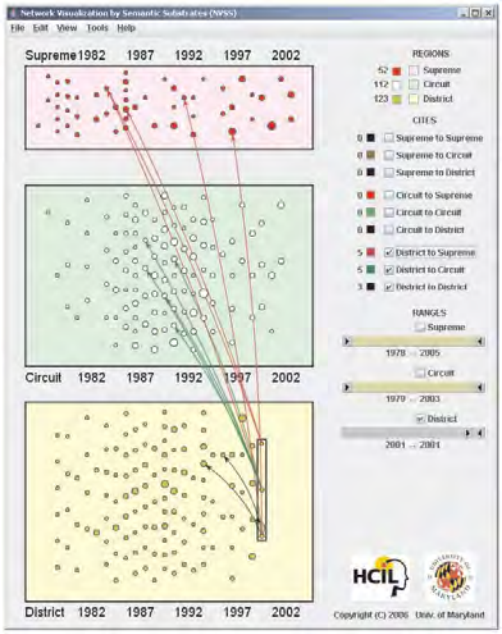
(b) A hierarchical radial layout [200]

(c) A rectilinear layout (arc diagram) [403]

Figure 2.25: Different layouts for node-link diagrams

(a) Geographic map [74]

(b) Scatter plot [298]



(c) Semantic substrates [352]

Figure 2.26: Layouts exploiting inherent spatial information (a), or direct spatial encoding of quantitative values and vertex-centric network measures (b,c)

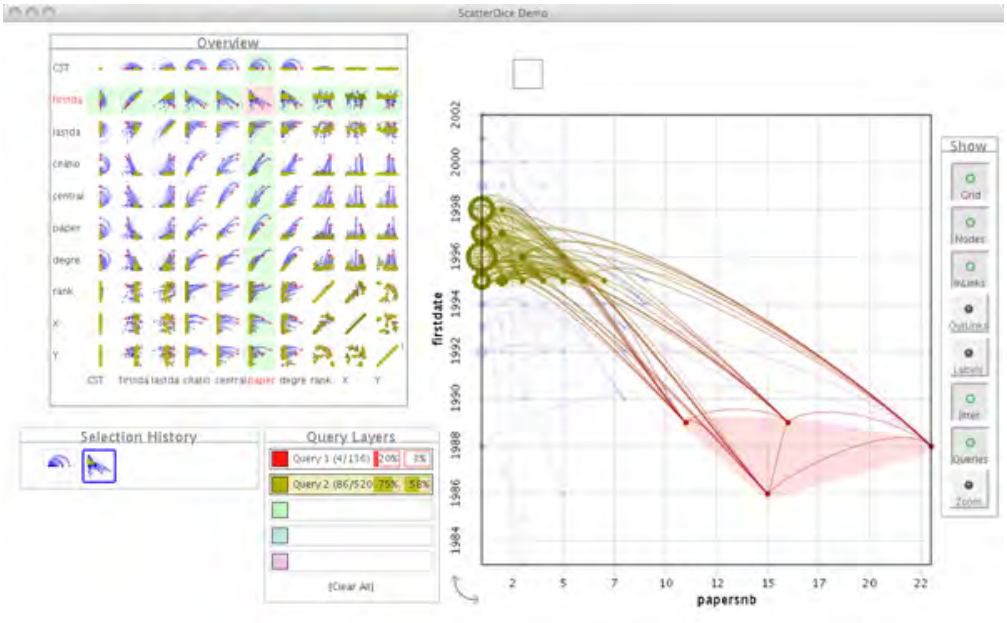


Figure 2.27: In GraphDice, multiple graph measures and attributes are shown by node-link diagrams arranged in a scatter plot layout [58]

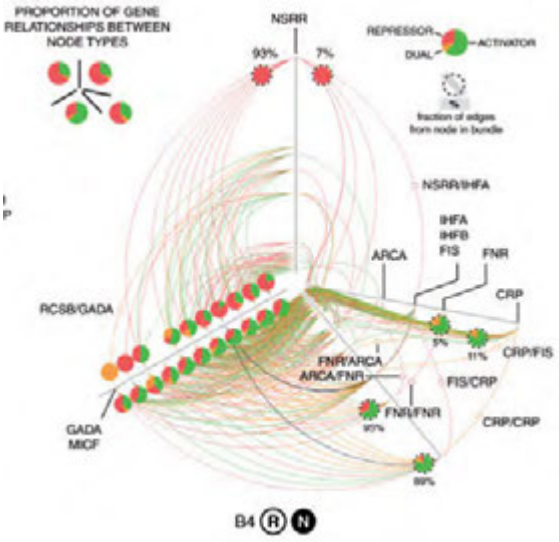


Figure 2.28: In HivePlos, vertex measures and attributes are used to arrange nodes in a parallel coordinates plot with polar coordinates [246]

Special cases of layout occur when vertices have an inherent spatial information that can be used to determine node positions. In a **geographic** layout, nodes are displaced on a map according to their geographic location [74] (Figure 2.26a). Quantitative variables associated to vertices, as well as vertex-centric network measures, can be also used to compute specific layouts, such as scatter plots [298] (Figure 2.26b) or semantic substrates [352] (Figure 2.26c). Bezerianos et al. [58] even state that node-link diagrams can be seen as extensions of scatter plots where data points are connected with links; their GraphDice visualization (Figure 2.27) shows several link-extended scatter plots, each encoding two different network measures. HivePlot [246] utilizes vertex attributes to position the nodes in a parallel coordinates plot with polar coordinates (Figure 2.28).

However, the most interesting layout problem applies to graphs whose vertices do not have any inherent spatial information; therefore, there is the need to find a convenient *spatialization* of the graph, i.e. assigning spatial positions to its vertices. In such cases, the layout can be computed in order to convey information about the relational structure of abstract data. For example, many layouts arrange into the same spatial region those nodes which are connected to each other. This kind of layout exploit the law of *prägnanz*, which is the fundamental law of gestalt perception [361]. In particular, it exploits the principle of grouping by proximity, by which objects which stand near to each other are visually perceived as being connected or forming a group [294]. However, common graph layouts also exploit other grouping principles [239], such as similarity (objects of similar shape or color), continuation (objects forming a continuous pattern), and symmetry (objects forming symmetric patterns).

On a more general level, several criteria have been formulated for drawing better graphs and/or computing better layouts. They are referred to as layout or graph drawing ***aesthetics***. Di Battista et al. [105] state that the fundamental and classical aesthetic is the minimization of

crossings between edges, while the display of symmetries is desirable; however, they note that other criteria depend not only on the graph class of interest and the adopted drawing standard, but also on personal preferences, traditions, and cultures.

Purchase et al. [315] mention symmetry, edge-rossing minimization, and edge bending minimization, while Ware et al. [401] list as factors path continuity, number of crossings, average crossing angles, total geometric line length; however, both Puchase et al. [315] and Ware et al. [401] have disputed these criteria, and have conducted experiments to empirically validate their importance and evaluate their cognitive cost (see Section 2.10).

Coleman and Parker [97] distinguish between syntactic aesthetics and semantic aesthetics; the former involve basic visual principles and the structure of the graph, the latter involve the meaning (i.e., additional attributes) of graph elements. They elicit a long list of rules: "nodes should not be too close together; nodes should not be too far apart; edge lengths should be short; edge lengths should not be too short; edge lengths should be uniform; edge crossings should be avoided; the angle between incident edges should not be too small; nodes should not be too close to edges; nodes should be within a bounding box; nodes should be near the center of a bounding box; nodes should be distributed evenly within a bounding box; the width of the layout should be minimized; inherent symmetry should be reflected by a layout." [97, p.1418]

Bennet et al. [51] survey graph visualization heuristics and distinguish three classes: node placement heuristics, edge placement heuristic, and overall layout heuristics. They list, as node placement heuristics: distributing nodes evenly, minimizing node overlaps, clustering related nodes, keeping nodes from coming too close to edges, and maximizing node orthogonality; as edge placement heuristics: minimizing edge crossings, minimizing edge bends, keeping edge bends uniform, minimizing edge length, keeping edge length uniform, maximizing minimum edge angle, and maximizing edge orthogonality; as overall layout heuristics: maximizing global symmetry, maximizing local symmetry of sub-graphs, minimizing total graph area, maximizing convex faces.

The problem of finding a graph spatialization which fulfils aesthetics criteria can be addressed by energy-based approaches. An **energy-based** approach basically models a graph as a physical system: vertices are modelled as particles and edges as physical interactions between particles. An energy function is then associated to the system, trying to capture some of the aesthetics criteria. Therefore, the problem of finding the best layout is translated into the problem of finding the configuration of the system which minimizes the energy function. An energy-based approach, therefore, consists of two components: of an energy model representing the system, and an optimization algorithm aiming at finding the desired system configuration. The algorithms are usually iterative, i.e. they find the energy minimum by generating a sequence of improving approximations (Figure 2.29).

The first energy-based approach can be considered the **barycentric layout** proposed by Tutte [374] for 3-connected planar graph; it was not described as energy-based by the author, but it is commonly understood as such [105]. The barycentric approach places each node in the center of mass (weighted centroid) of its adjacent nodes, and fulfils the most relevant aesthetics criterion for planar graphs by guaranteeing no edge crossings.

Energy-based approaches can be also referred to as **force-directed**: the physical system is described in terms of forces between particles, and the optimization algorithm aims at finding the

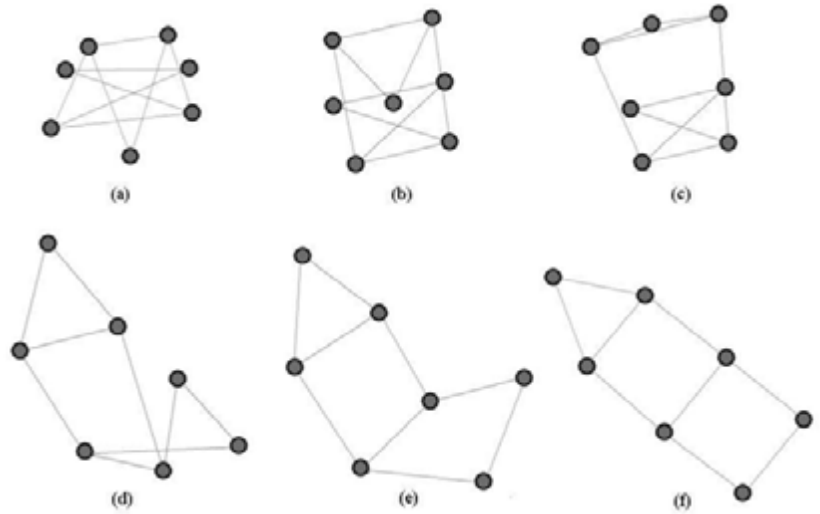


Figure 2.29: A graph drawing through a number of iterations of a force-directed algorithm [317]

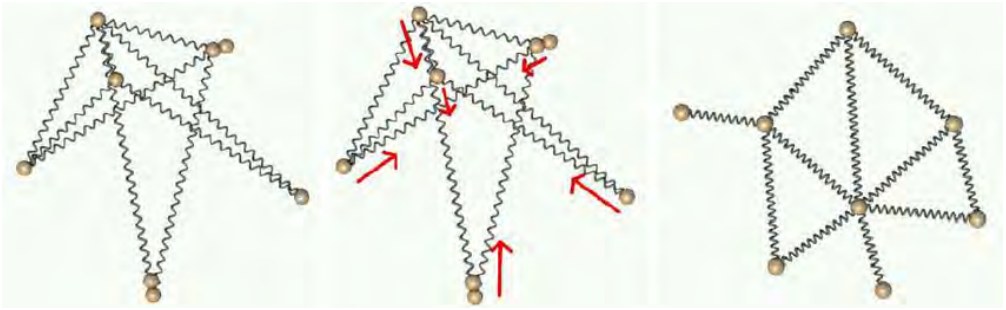


Figure 2.30: The **spring embedder** is a force-directed approach by which vertices are modeled as particles and edges as springs: the layout algorithms aim at the equilibrium of forces [238]

condition where all forces are balanced. In other words, the layout problem is translated into a so-called n-body simulation. Force models are just alternative representation of energy models, since force can be understood as the negative gradient of energy and, therefore, an equilibrium of forces is a local minimum of energy. [287]

The classical force-directed layout is the one proposed by Eades [116]: vertices are modelled as physical particles having an electric charge (then repelling each other), and edges are modelled as elastic springs (Figure 2.30). The optimal layout is obtained at the equilibrium of electric and elastic forces.

Differently from the barycentric layout, which has been proven to guarantee no edge crossings, force-directed layouts are considered well-suited to address the aesthetics criteria, but a formal prove of these fulfilments has been provided in limited cases. Noack [288], for example, observes that layouts with optimal energy are a relaxation of the concept of clustering with optimal modularity; therefore, they can be understood as placing connected nodes close to each other, according to the proximity principle.

Since their introduction, force-directed layouts have been improved in order to tackle the scalability issue, by pursuing three directions [42]: multiscaling, faster approximations, and better energy models.

Multiscaling methods (also knows as multilevel methods) aim at computing the layout of large graphs by subsequent levels of detail [187, 398]. At first, the graph is coarsened by collapsing a set of vertices from one level into one representative vertices in the next level; then, the layouts of all levels are computed, from the coarser levels to the finer ones. In other words, multiscaling approaches compute first an approximate layout of the entire graph which preserves the global structure, and then perform local detailed improvements. These approaches sometimes are also referred to as graph coarsening, even though coarsening is only the first phase of multiscaling.

Another approach to speed up the computation of a layout for large graphs consists in stopping an iterative force-directed algorithm when it has not yet reached the optimum, but is sufficiently close. Fruchterman and Reingold extended the analogy with physical systems by introducing the notion of **layout temperature**: when the system is hot, the particles move faster, when the system is cold, the particles move slower or stop. In the Fruchterman-Reingold layout, therefore, the approximation is controlled by this temperature heuristic [160].

A problem of energy-based algorithms is that the optimization can converge in a local minimum of the energy function (corresponding to an unstable equilibrium of forces). In interactive environments, this unintended deadlock can be solved by the user, who can directly manipulate the layout, for example by dragging a node to untangle it. However, several algorithms try to escape the local minimum by utilizing temperature-related paradigms, such as simulated annealing [104] or adaptive cooling [204].

Another class of layout algorithms aim at faster approximation by **hierarchical space decomposition**. In the Barnes-Hut algorithm [41], the three-dimensional space of the n-body simulation is hierarchically partitioned into cubes (managed by octree data structures); given a particle, forces exerted by particles in nearby cubes are computed individually, while particles in distant cubes are approximated by their centers of mass. Another hierarchical space decomposition method exploits a similar principle and utilizes quadtrees [317].

Spectral drawing denotes a class of layout algorithms which use the eigenvalues and the eigenvectors of the adjacency matrix, or derived matrices (e.g., the Laplacian matrix, which can be understood as the adjacency matrix with the addition of node degrees into the main diagonal). Therefore, such methods they are also known as eigen-projection methods. Different choices of matrices and eigenvectors lead to optimizations according to different aesthetic criteria. In the approach by Koren [243], for example, the energy function corresponds to the barycentric layout, but the algorithm can directly compute the global minimum.

Besides optimization algorithms, different energy models have been also explored to obtain better layouts. Noack [288] observes that many force-directed approaches have attraction and repulsion forces, and these forces are often proportional to a certain power of the distance. Therefore, all attraction-repulsion models can be described by an (a, r) notation, where a and r are the exponents of the distance. Several combinations are possible, given the condition $a > r$ to avoid an indefinite growth of the layout area. For example, since the spring force is proportional to the distance and the electric force is proportional to the square of the distance,

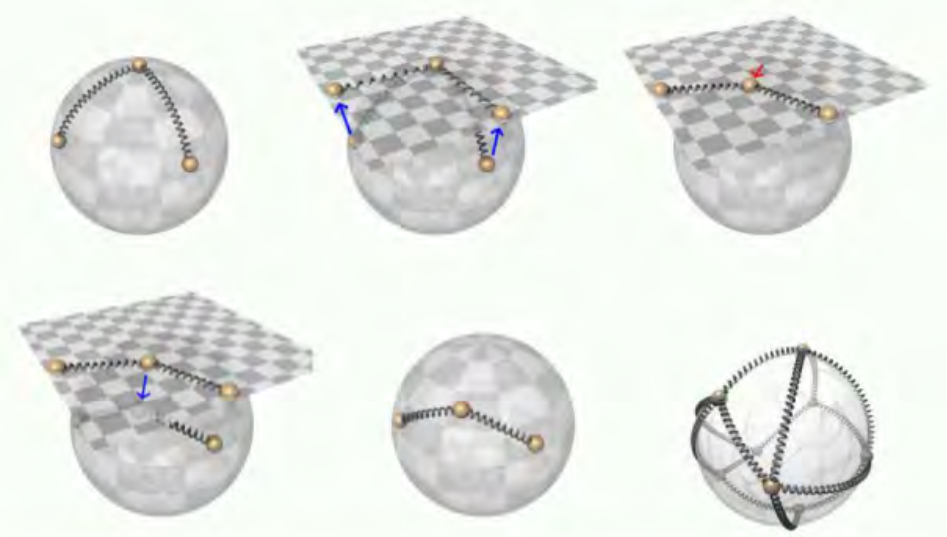


Figure 2.31: A generalized force-directed method for non-Euclidean geometries [238]

Figure 2.32: A three-dimensional hyperbolic layout [279]

the afoementioned spring-electric model can be described as $(1, 2)$. Fruchterman and Reingold use an energy model than can be descibed as $(2, -1)$, Davidson and Harel as $(1, -3)$, the linlog layout [287] as $(0, -1)$.

Kamada and Kawai [223] introduce a force-directed model featuring only springs: each pair of vertices is modelled as a spring, whose rest length is proportional to the length of the shortest path between the two vertices. In this way, at the minimum-energy the euclidean distances between nodes in the layout space are an approximation of the geodesic distances between vertices. A similar paradigm is adopted by **stress minimization** approaches, where the energy function is a stress function, defined as the sum of the squared differences between the euclidean distance and a weighted measure between pair of vertices. The optimization algorithm, therefore, aims at minimizing the stress. This problem can be understood as an application to graph layout of the more general problem of **metric multidimensional scaling (MDS)**. Multidimensional scaling

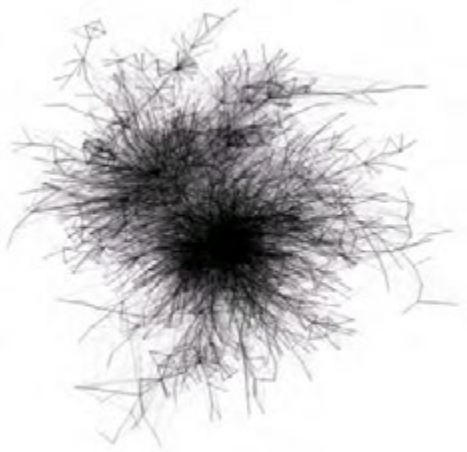
is a technique used to project points of a n-dimensional space onto a k-dimensional space (with $k << n$, usually $k = 2$, i.e. a two-dimensional computer screen), aiming at preseving clusters. A graph with n nodes admits a n-dimensional representation, where each vertex is represented by a vector with n elements (e.g., a row of the adjacency matrix, or the vector of geodesic distances from all other vertices). Therefore, a stress-minimization layout can be computed by a metric MDS algorithm such as, for example, the stress majorization algorithm [166].

With a further advancement, one can think of approximating the geodesic distance not by euclidean distances, but rather by non-euclidean distances, thus projecting the graph into an hyperbolic space or into a elliptic (e.g., spherical) space. Kobourov and Wampler [240] describe a generalized force-directed method for graph layout in Riemannian geometries (Figure 2.31. The H3 system [279] features a three-dimensional hyperbolic layout (Figure 2.32).

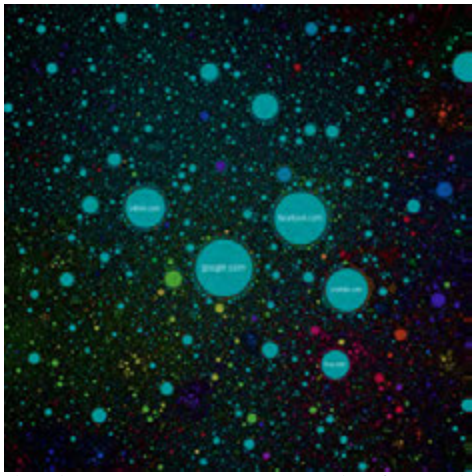
Other techniques for multidimensional scaling make use of artificial neural network and produce a spatialization of data known as Kohonen map, or self-organinzing map [241].

2.6.1.2 Visual scalability

All the aforementioned approaches, and their combinations (e.g., multiscaling and spatial approximation [181]), enable efficient computation of layout for large graphs. Nevertheless, another problem to be addressed is the visual scalability, in order to avoid the so-called hairballs, i.e. graph drawings where visual clutter hinders conveying of information and gaining of insights. A cheap solution might be the deletion of all nodes or, alternatively, of all links. Hurst [212] proposes a visualization of the so-called blogosphere (a graph whose vertices are weblogs, and edges are weblinks between them); he states that by showing only the links (Figure 2.33a, we can get a far better look at the structure than if we include all the nodes. On the contrary, the internet map project [126] exploits links to compute a force-directed layout, but then shows nodes only (Figure 2.33b).



(a) The blogosphere map, an example of a graph diagram with links only [212]



(b) The Internet map, an example of a graph diagram with nodes only [126]

Figure 2.33: Link-only / node-only diagrams

While the choice of showing nodes only, or edges only, might be arbitrary, and its results arguable, there are more sophisticated methods, with clear rationales, for reducing the visual complexity. They have been classified [344] according to the reduction mechanism (e.g., aggregation, pruning), the network elements they reduce (node/vertices or edge/links), or the part of the visualization pipeline they operate onto (data, geometry, or image).

Hierarchical clustering is a reduction technique commonly used to reduce vertices/nodes by aggregation, at both the data level and the geometry level. At the data level it identifies sets of vertices (by similarity, or by topological proximity) and collapses them into aggregate representatives in a pre-processing step (Figure 2.35). At the data level, the identification and collapsing of clusters can be computed in parallel with the layout algorithm, and can also be interactively controlled by the user (Figure 2.34).

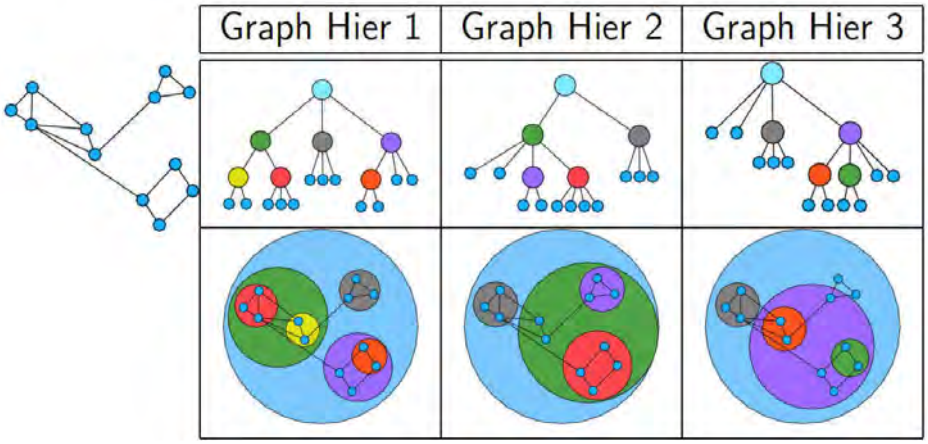


Figure 2.34: Multiple graph hierarchies superimposed on the same graph. [21]

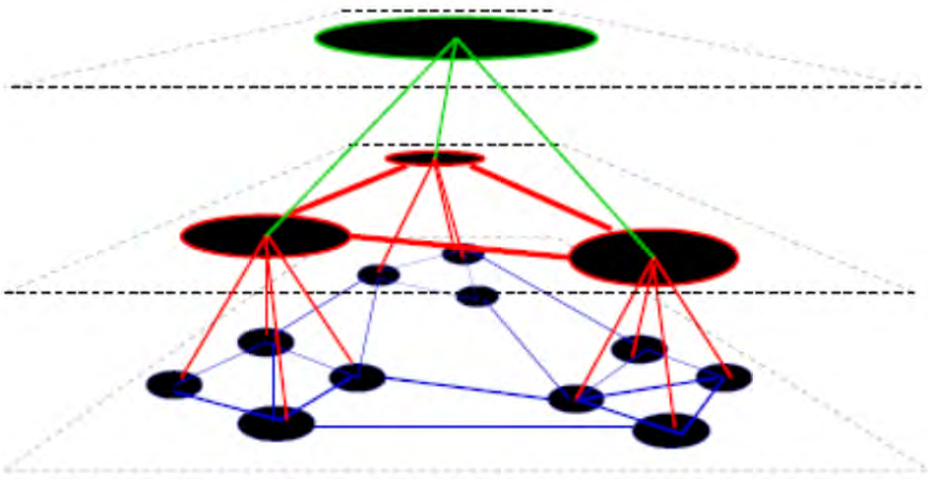


Figure 2.35: An illustration of the concept of hierarchical clustering [117] (in [173])

Several approaches addressing edge scalability rely on pruning at the data level: they aim at deleting edges while preserving the overall structure of the graph. A classical example is edge simplification based on the **minimum spanning tree**. Given a graph G, a spanning tree S can be understood as a tree which is a subgraph of G (i.e., all edges of S are edges of G) and that spans the entire graph (i.e., all vertices of G are vertices of S). A minimum spanning tree is the spanning tree with the minimum length. However, the minimum spanning tree can lead to an excessive lost of information, since a graph with N vertices, having a maximum of $N \times (N-1)$ edges, has a minimum spanning tree with only $N-1$ edges.

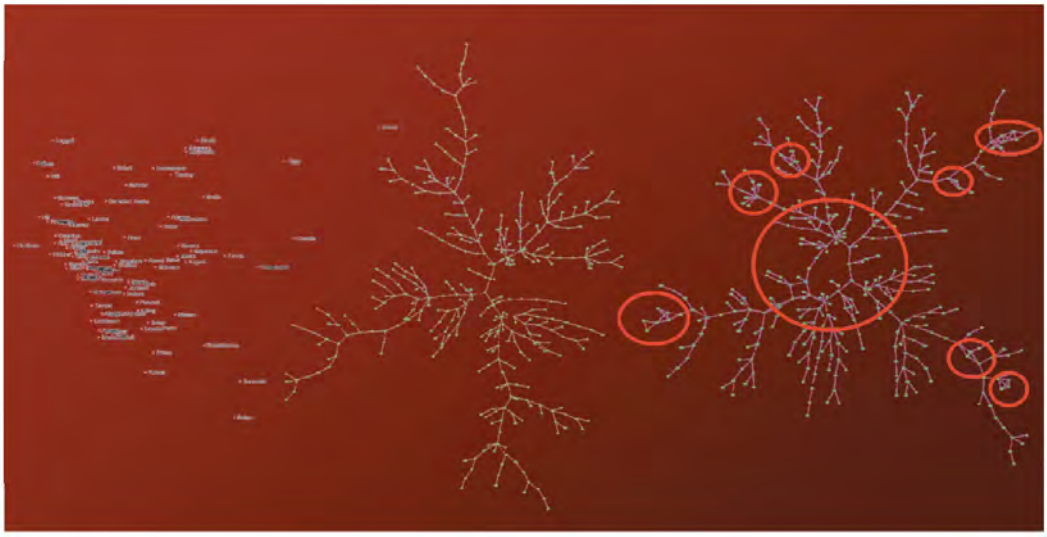


Figure 2.36: Three node-link diagrams of the same graph with different link-reduction approaches: nodes only (left), minimum spanning tree (middle), pathfinder network (right) [93]

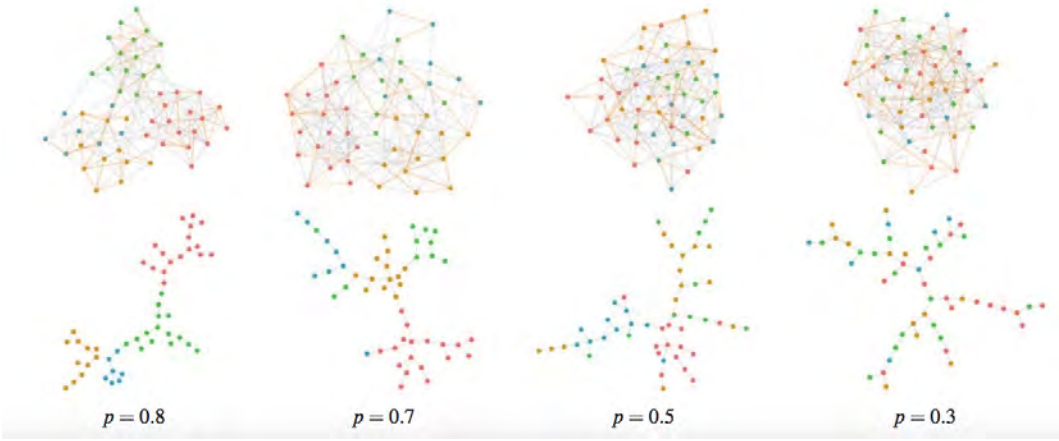
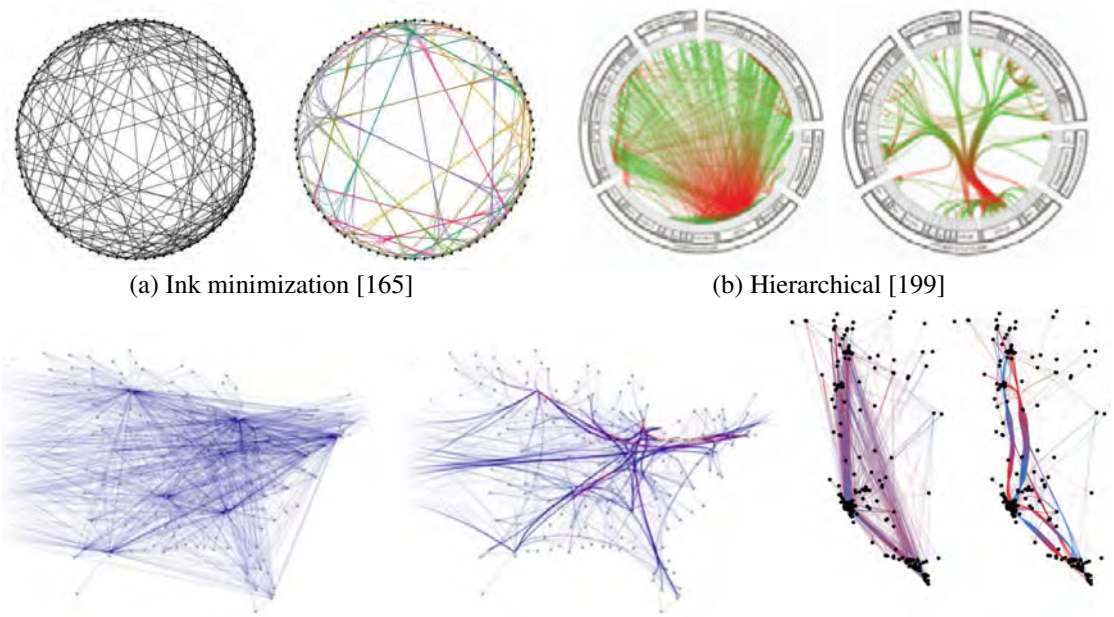


Figure 2.37: Link reduction of four graphs by minimim spanning trees and betweeneess [385]

The **pathfinder network** is a generalization of the minimum spanning tree, where the extent of reduction can be reduced by two parameters r and q. The r parameter is the order of the Minkowski distance, a generalization of the Euclidean distance; the q parameter can be understood as the number of steps which are considered in generating the network. The minimal pathfinder network has $r = -1$ and $q = N - 1$, and contains the edges of all spanning trees. Figure 2.36 shows three visualizations of a network with 367 vertices and 61,175 edges [93]; on the left-hand side, a node-only diagram with a multidimensional scaling layout; in the middle, a minimum spanning tree showing 366 links with a force-directed layout; on the right-hand side, a pathfinder network showing 398 links with a force-directed layout, with cycles highlighted by red circles. Other approaches use different graph-theoretic measures for identifying links that can be deleted while preserving the structure, for example edge betweenness [385] (Figure 2.37).

Edge bundling algorithms [430] reduce edges at the geometry level by aggregation. Some bundling approaches are based on the minimization of a cost function: the ink-minimization algorithm [165] aims at optimizing area utilization by minimizing total edge length (Figure 2.38a), while the energy-minimization approach [201] is similar to a force-directed layout algorithm, but forces are also applied to links (Figure 2.38c). Other approaches exploit additional information: hierarchical bundling [199] exploits a hierarchical partition of vertices (Figure 2.38b), while divided bundling [347] identifies different routes depending on edge direction (Figure 2.38d).

The visual scalability can be also addressed at the image level, by optimizing the rendering algorithms to enhance graph readability. **Splatting** techniques (Figure 2.39) render graphs as vector fields and can be applied to either vertices [386] or edges [76, 83].



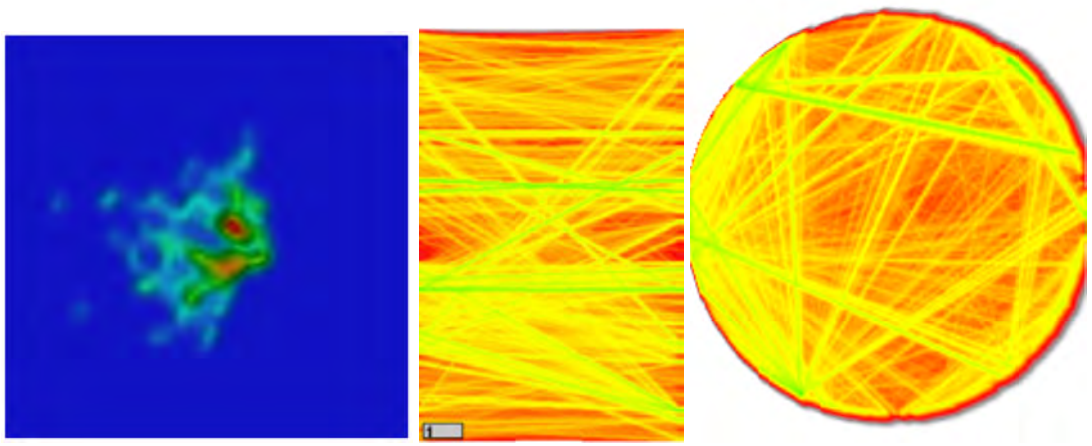
(a) Ink minimization [165]

(b) Hierarchical [199]

(c) Energy minimization [201]

(d) Divided [347]

Figure 2.38: Different approaches to edge bundling



(a) Graph splatting [386] (b) Edge splatting, bipatite layout [83] (c) Edge splatting, radial layout [76]

Figure 2.39: Different approaches to splatting

2.6.2 Matrix-based

As an alternative to node-link diagrams, graphs can also be visualized by matrix-based techniques. These approaches build upon adjacency matrix representation (Figure 2.10), and usually visualize the presence of an edge, or its weight, by the cell color (Figure 2.20b). Since vertices are implicitly encoded in row and columns headers, but are not directly visualized, sometimes matrix-based techniques are also referred to as link-oriented approaches. Figure 2.42 shows a weighted digraph with 50 nodes as a matrix. Analogously to node-link diagrams, matrices can be reordered to show clusters and other possible features (Figure 2.41). Moreover, they can be aggregated at different levels of detail to ensure scalability (Figures 2.40 and 2.43).

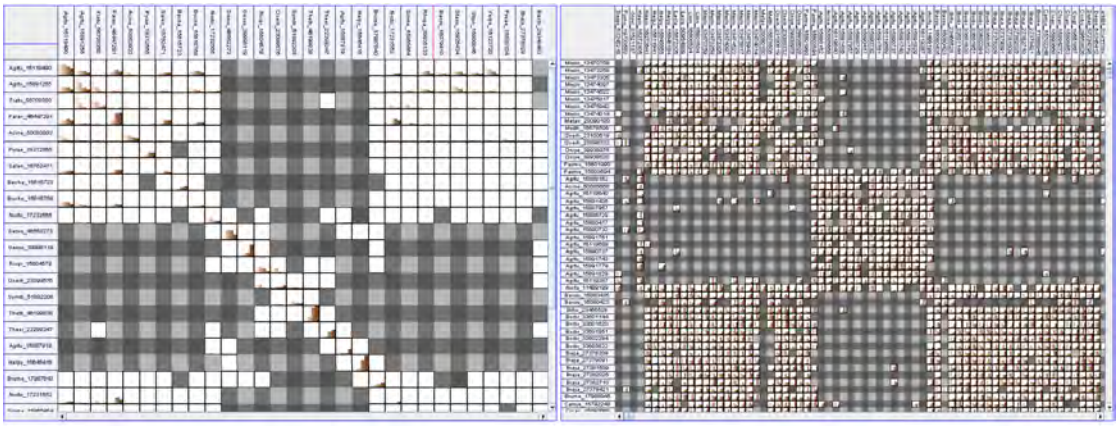


Figure 2.40: A matrix-based visualization at two levels of aggregation [121]

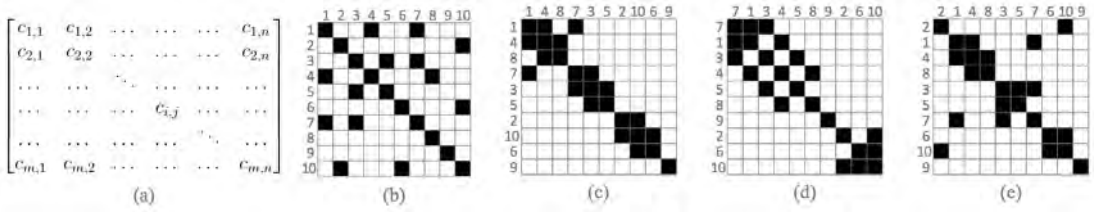


Figure 2.41: An adjacecy matrix and its visualization with different reorderings [49]

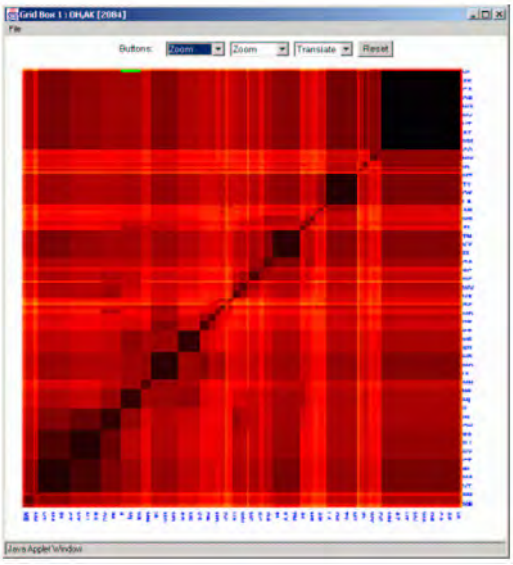


Figure 2.42: A matrix-based visualization [3]

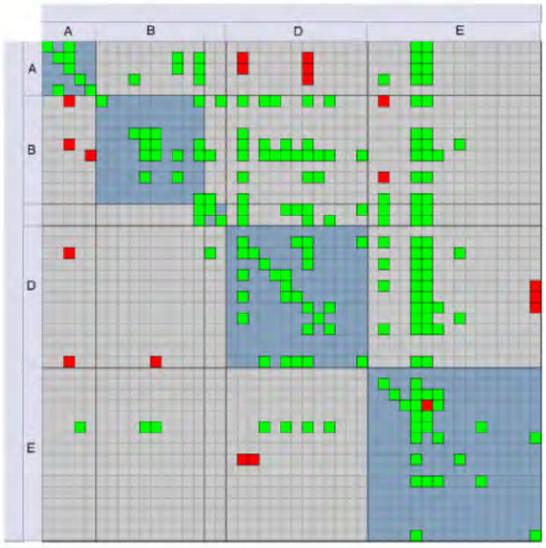
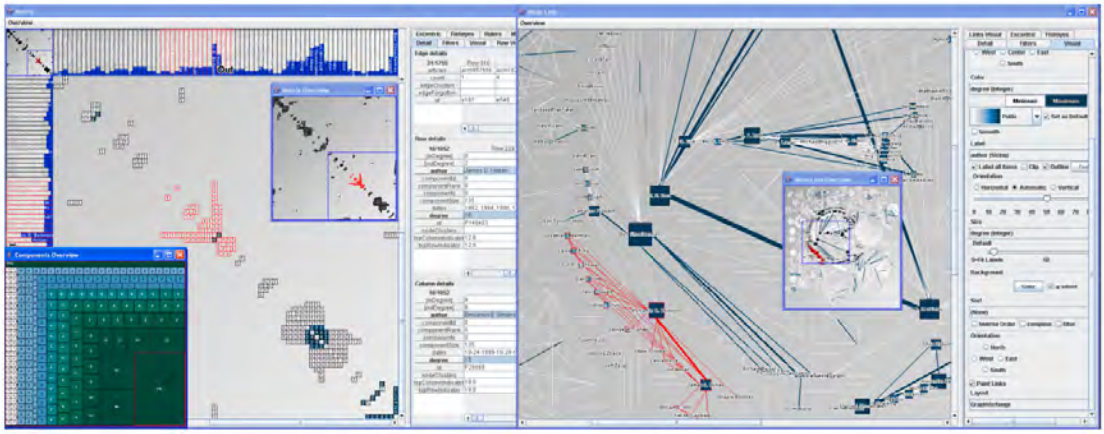


Figure 2.43: A clustered matrix [383]

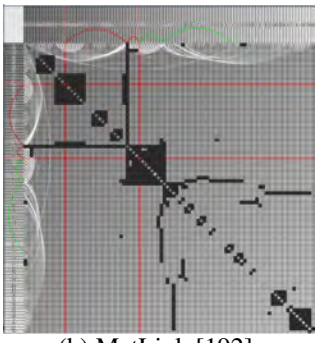
2.6.3 Hybrid visualizations

Hybrid approaches try to combine benefits of node-link diagrams and matrix-based visualizations while avoiding shortcomings (for a discussion of these benefits and shortcomings, see Section 2.10).

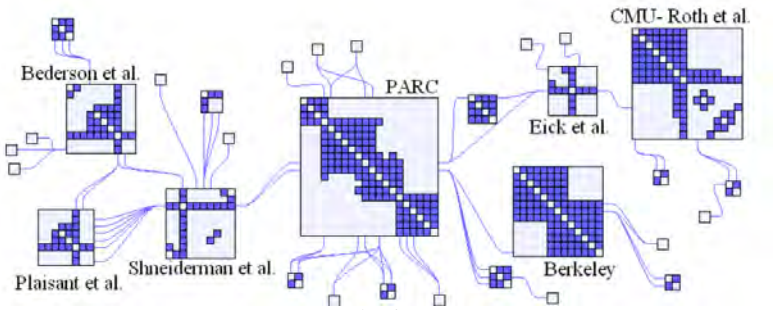
MatrixExplorer uses both paradigms, by putting a node-link and a matrix side by side as coordinated views [192]. A tighter level of integration results in two other techniques: Matlink [193] is a matrix-based visualization complementd by two arc diagams (Figure 2.44b), while Nodetrix [194] is a visualization where intra-cluster relations are shown as matrices and inter-cluster relations as links (Figure 2.44c). TreeMatrix [333] combines adjacency matrices, node-link diagrams, and arc diagrams to visualize compound graphs.



(a) MatrixExplorer [192]



(b) MatLink [192]



(c) Nodetrix [192]

Figure 2.44: Hybrid visualizations

2.6.4 Other visualization

Other visualization techniques for graphs include measure-based visualization and space-filling (or area) visual representation.

2.6.4.1 Measure-based visualization

We have already discussed how graph-theoretic measured can be used to enhance graph visualization; in particular, we have seen how vertex-centric measures can be used to compute layouts of node-link diagrams, e.g., by arranging nodes in a scatter plot (Figure 2.26b). However, vertex-centric measures and their distribution can be used to get alternative visual encodings, besides node-link diagrams and adjacency matrices. Bagrow et al. [36], for example, introduce the B matrix, whose element $B(l, k)$ is defined as the number of nodes from which exactly k nodes are reachable in l steps. Since $B(1, k)$ corresponds to the degree distribution, rows of the B-matrix can be understood as generalizations of the degree distribution. Therefore, the B-matrix can capture graph properties beyond degree distribution, such as small-world characteristics of real networks (Figure 2.45). GraphPrism [222] utilizes further generalizations of a B-matrix, comprehending l-distributions of other vertex-centric measures besides degree (Figure 2.46).

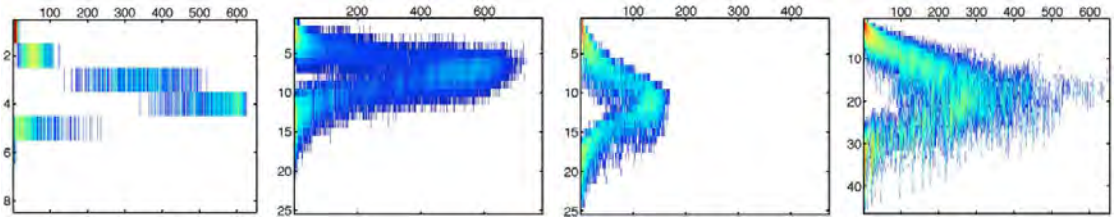


Figure 2.45: Four networks visualized by B-matrix [36]

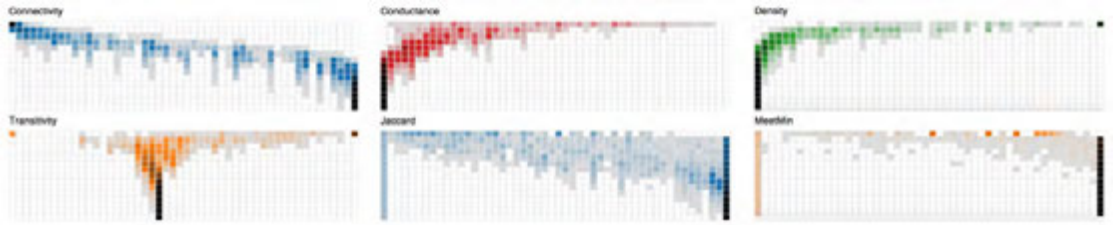


Figure 2.46: GraphPrism is a generalization of a B-matrix [222]

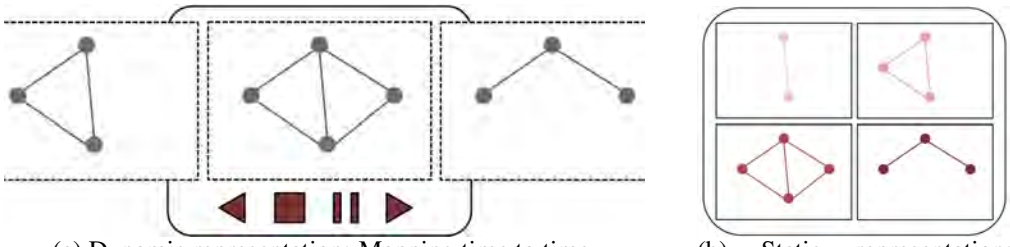
2.6.4.2 Space-filling

Space-filling visual encodings are those encodings which use area as the main geometric primitive, instead of points and lines. They are commonly used to visualize tree and hierarchies (e.g., tree maps [221], see also [342]), and in this case are especially referred to as containment map: child elements are represented by areas contained inside the parent area. However, space-filling encoding, also referred to as maps, can be also used to visualize graphs: in this case, adjacent areas represent adjacent nodes or adjacent clusters [205] (Figure 2.47).

Figure 2.47: GMap is a space-filling encoding to visualize a graph as a map [205]

2.7 Visual encodings for temporal graphs

The problem of visual representation of temporal graphs can be seen as a particular case of the problem of visual representation of time-oriented data, i.e., the choice of a proper encoding of the temporal dimension. There are two main options for mapping time [14]: mapping time to time, and mapping time to space. The former means that the temporal aspect of data is mapped to physical time, thus obtaining a visual representation that changes over time to reflect changes in the data; in other words, mapping time to time leads to **dynamic representations**, such as animations or slide shows (Figure 2.48a). Conversely, mapping time to space leads to **static representations** (Figure 2.48b), where both temporal and non-temporal data aspects are mapped to visual variables, both geometrical (i.e., lines, points, areas, and their positions) and retinal (e.g., color). It is worth noting that in this context the distinction between static and dynamic representations is not related to interaction: both static and dynamic representations support interactive, user-controlled visualization.



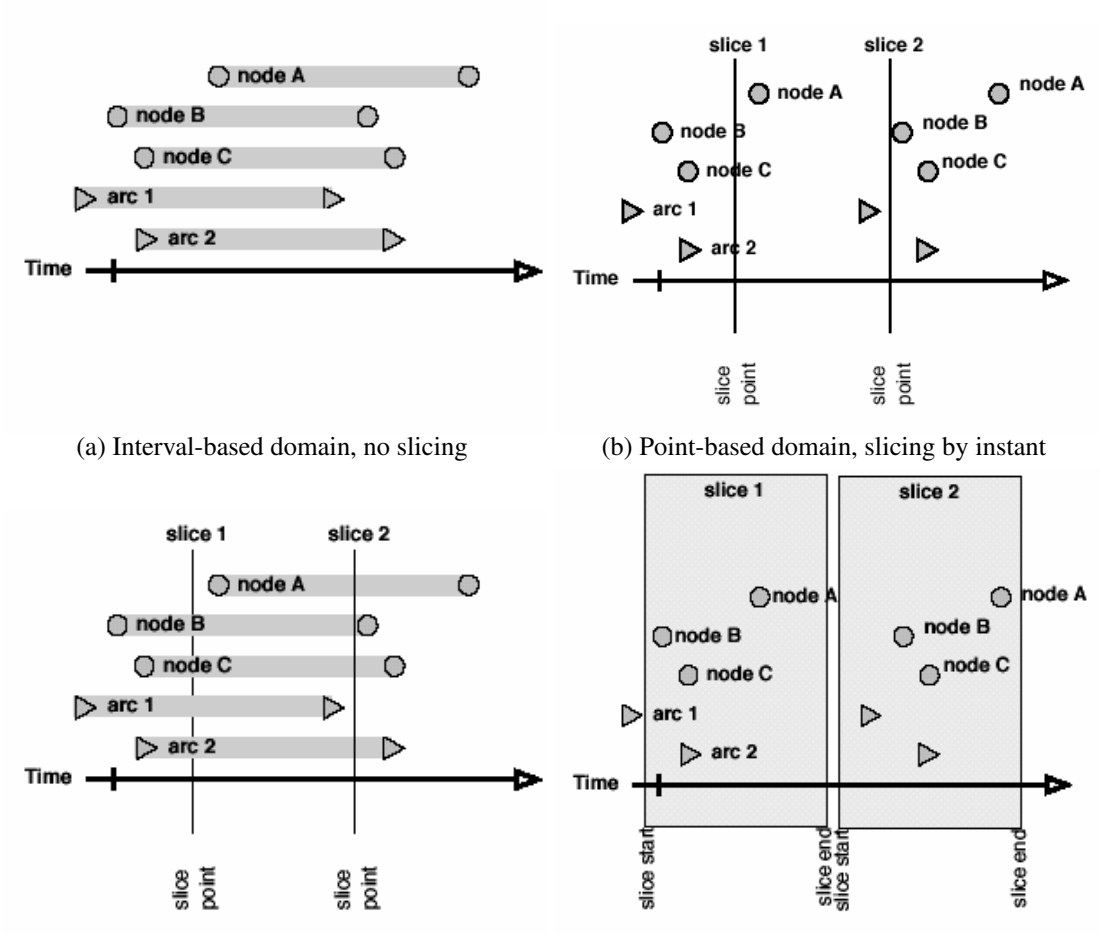
(a) Dynamic representation: Mapping time to time

(b) Static representation: Mapping time to space

Figure 2.48: Visual representation of temporal graphs

An important design aspect for visualization of temporal graphs is the scope of the time domain, which can be point-based or temporal-based (see Section 2.1.1). Point-based time domains can be seen in analogy to geometric points in space, i.e., having an extent equal to zero. Conversely, interval-based time domains relate to subsections of time having a temporal extent greater than zero. In a temporal graph, vertices and edges can be represented as events in a certain time-domain, either point-based or interval-based. However, a temporal graph is generally acquired from a set of events by a process called **time slicing** [50]. Depending on whether the duration of the time slices is finite or null, we distinguish interval slicing from instant slicing, respectively. The sliding process is also necessary to describe a network at any specific point in time: in an instant-based domain, for example, slicing by intervals is necessary to build a non-null network (Figure 2.49b). Moreover, time slicing can be seen as way to transform one time scope into the other, by aggregation (slicing time points by intervals, Figure 2.49d) or by sampling (slicing time intervals by instants, with so-called snapshots as outcome, Figure 2.49c). Additional aggregation over time can be performed to abstract the temporal information up to the intended level of detail, or time granularity. The aggregation can be also performed upon the graph structure, for example by simply collapsing compound graphs, or by clustering and coarsening general graphs; this operation is also referred to as graph granulation [360]. Time granulation or graph granulation enable a lossy reduction of one type of information, endowing

a better visual encoding of the other one.



(a) Interval-based domain, no slicing

(b) Point-based domain, slicing by instant

(c) Interval-based domain, slicing by instants

(d) Point-based domain, slicing by interval

Figure 2.49: Time scope and time slicing (adapted from [50])

2.7.1 Dynamic representations: Mapping time to time

In dynamic representations, since temporal aspects of data are directly mapped to physical time, to a certain extent there is no need to design specific visual encodings: encodings used for static graphs are animated or shown in sequence, in a flip book or slide show fashion. The parameters of the mapping between the domain time in the data and the users' physical time (e.g., animation speed, animation starting and stopping) can be controlled by users by the appropriate interaction techniques (see also Section 2.8).

Traditionally, graph drawing algorithms are categorized as online, or offline, if the dataset bears a dependency from internal time, or external time, respectively (see Section 2.1.1). How-

ever, besides an increased algorithmic difficulty of the online case (since the entire sequence in not known in advance, and the layout has to react to new data in real-time [159, 189, 207]), this distinction does not affect the visual encoding.

In general, in a flip-book or slide-show animation, data is visualized in sequence: when new data is shown, old data disappears. However, especially in the case of node-link diagrams, many approaches recur to morphing: intermediate interpolations are computed to generate smooth animated transitions.

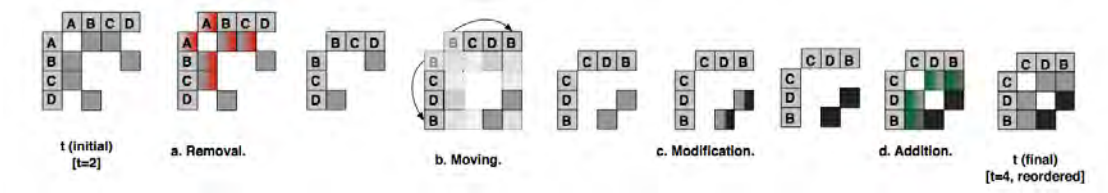


Figure 2.50: A staged animation of a matrix-based dynamic representation [334]

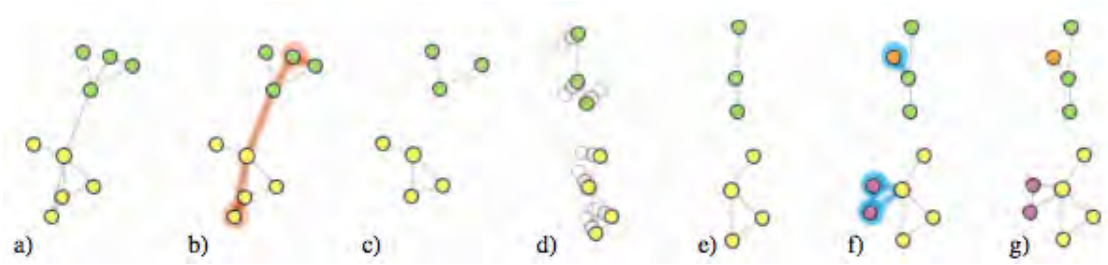


Figure 2.51: Staged transitions with differential highliting [34]

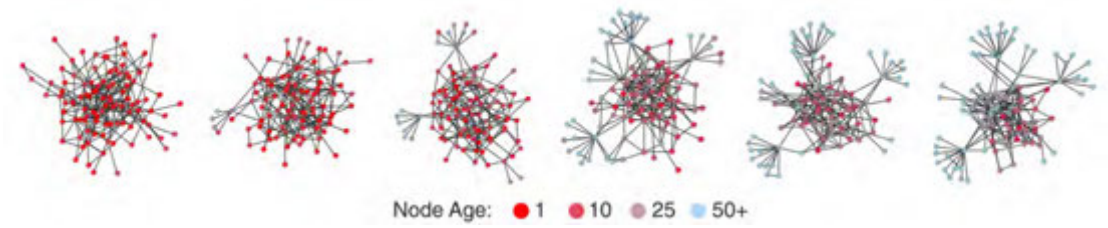


Figure 2.52: Ageing of graph elements [174]

Most of dynamic representations, since the classical examples by Eades [118] and Friederich [154–156], are based on node-link diagrams (Figures 2.51 and 2.52), but some approaches also use adjacency lists [84, 85], area-based encodings [124, 148, 163, 164, 206, 258, 267], and matrix-based encodings [84, 85, 132, 263, 316, 334] (Figure 2.50).

In some cases, the graph structure is fixed and the animation shows only changes of the attributes by changing retinal properties of the visualization [392]. In other cases, the vertices are stable, and edges appear or disappear during the animation [338]. In the more general case,

the entire graph structure changes over time, and changes are reflected by a rearrangement of the layout, which can be radial [291, 292], orthogonal [172], energy-based [108, 109, 248, 254, 289, 305, 395], hyperbolical [89], or spectral [65]. Nesbitt et al. [282] propose an optimization of the dynamic layout problem by Gestalt principles. Frishman & Tal propose a dynamic layout for clustered graphs [157, 158]. Other approaches address the scale and reduce the graph by aggregation (dynamic edge bundling [213]) or filtering [176].

Some authors threat the problem of the visualization stability, or the preservation of the user's mental map, in the context of dynamic representations. However, since the mental map problem affects both static and dynamic representations, we will discuss it in Section 2.10). Indeed, there are other strategies to deal with temporal aspects in both static and dynamic representations; a retinal variable, for example, can be used to emphasize those graph elements there are added or removed (**difference highlighting** [6, 19, 34, 70, 71, 263, 332], see Figure 2.51) or, alternatively, to deemphasize the ones remaining stable (**ageing** [98, 174, 327], see Figure 2.52).

However, further optimization techniques have been proposed to improve the perception of dynamic representation, such as **staged animations**. In a staged animation, changes are divided into steps by type and these steps are reproduced in a given order (e.g., deletions first, then layout rearranging, lastly additions). They have been applied to node-link diagrams [34, 320, 331, 332] as well as matrix-based visualization [334] (Figure 2.50).

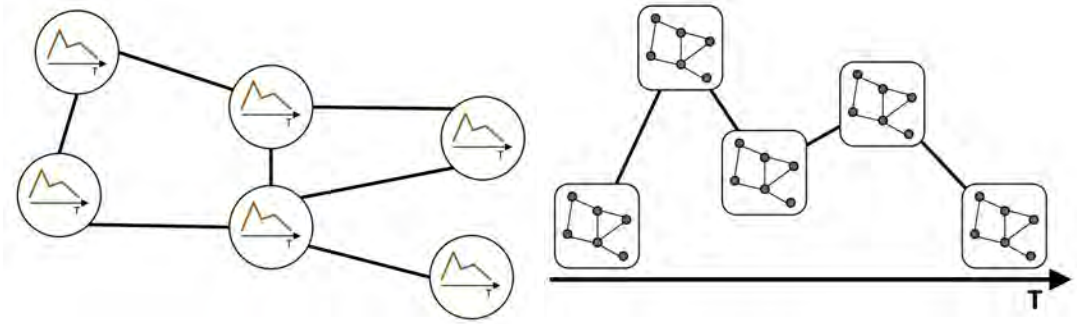
2.7.2 Static representations: Mapping time to space

In static representations, temporal aspects of data as well non-temporal ones are mapped to visual variables, both planar (e.g., geometrical primitives and their position) and retinal (e.g., color, texture). However, the essential nature of a visual mapping is mainly determined by the choice of planar variables (see also Section 2.6.1.1). Therefore, we can categorize static representations of temporal graphs according to the data aspect (either temporal or non-temporal) which is mapped to planar variables and determines the main visualization geometry, while the other one is mapped to a secondary geometry or to retinal variables (e.g., color). In particular (Figure 2.53), we distinguish amongst: time within structure, structure within time, and merged representations.

2.7.2.1 Time within structure

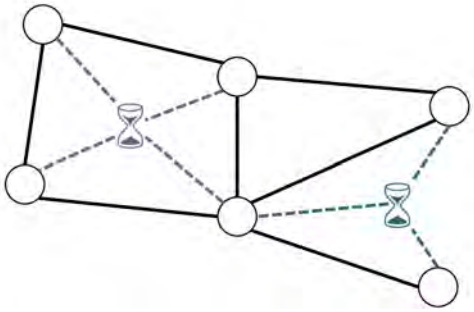
In this type of visualization, the relational information (the graph structure) is mapped to the main geometry, and then time is visually encoded, either by a secondary geometry, or by retinal variables.

Secondary geometry. When time is mapped to the secondary geometry, it is usually visualized as a timeline, arranged along one of the chartesian axes. The secondary visualization can be made smaller and nested within elements of the main geometry. Sparklines [373] are a common choice, both in the line chart and in the bar chart version, but other glyphs are also used, for example *Gestaltlines* [69].



(a) Time within structure

(b) Structure within time



(c) Merged

Figure 2.53: Different types of static representations

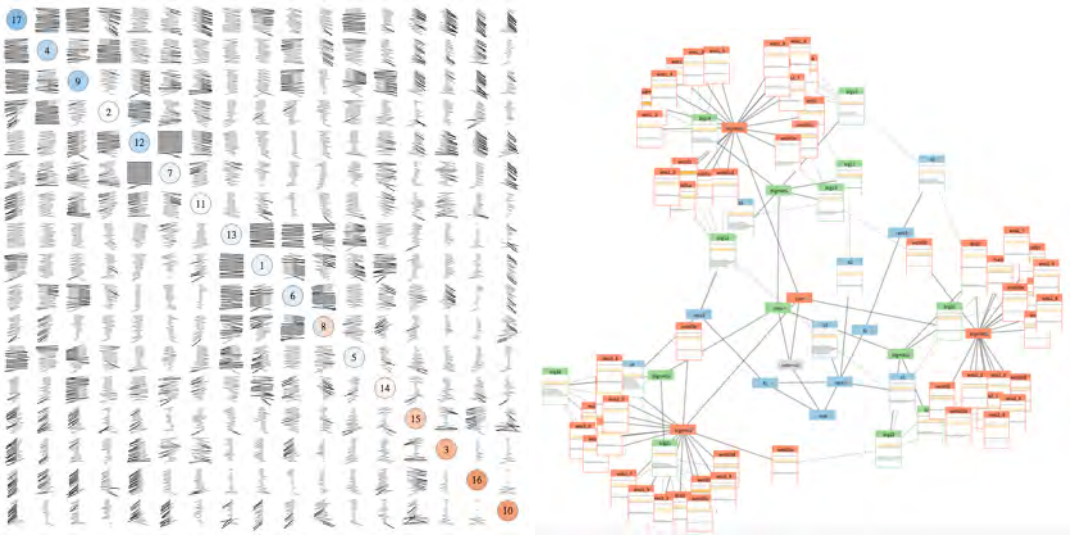
When temporal information is associated with edges and the main geometry is an adjacency matrix, we can have visualizations featuring timelines nested inside matrix cells [69, 82, 424] (Figure 2.54a).

Alternatively, if the temporal information is associated with vertices and the main geometry is a node-link diagram, a timeline-based visualization can be visualized inside each node, for example a line chart [61, 183, 339, 357, 376] (Figure 2.23a), an area chart [362] (Figure 2.54b)), or a bar chart [262, 406]. Liu et al. [259] propose a space-filling visualization, where the graph structure is visualized by adjacent regions in a self-organizing map and the temporal information by concentric containments within each region.

If the temporal information is associated with edges, it can be visualized along each link; Schmauder et al. [340], for example, diveide each link into line segments and only show those segments which correspond to time slices when the edge was present. If temporal information is associated with all graph elements, then timelines can be visualized both around nodes and along edges [242] (Figure 2.55).

It is worth noticing that, even if a Cartesian coordinate is a common choice for visualizing a timeline, there are also node-link diagrams with circular glyphs, where time is mapped to the angular coordinate [242] or the radial coordinate [125, 295].

The levels of geometric mapping can be more than two, like for example in Tarameshloo et



(a) (b)

Figure 2.54: (a) A Gestaltmatrix of temporal asymmetric relations: each cell contains line segments arranged along a vertical timeline; segment slopes represent relative weights of reciprocal edges [69] (b) a node-link diagram with nested line charts: temporal information associated to each node is visualized along a horizontal timeline inside the node [362]

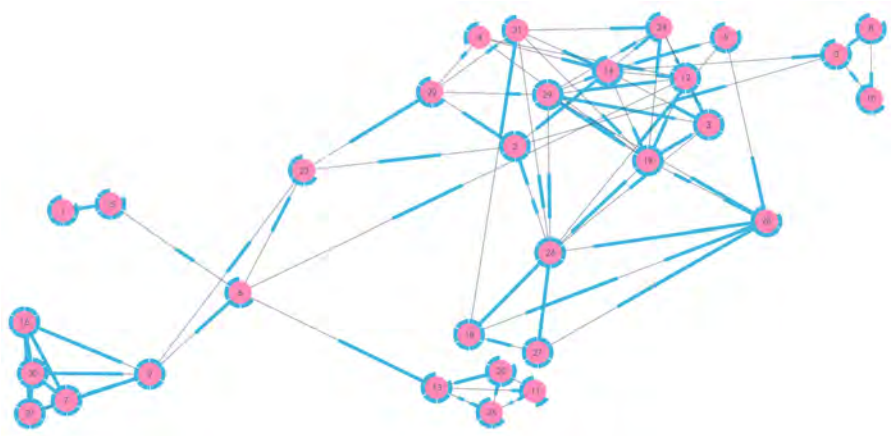


Figure 2.55: Glidgets are interative glyphs showing temporal information (presence) around each node and along each link in a node-link diagram [242]

al. [364] (Figure 2.56): the main geometry is an arc diagram; then, a timeline is associated as a secondary geometry to each node in the arc diagram; finally, at the third level, small node-link diagrams arranged along the timelines represent ego networks of a given node at a given time.

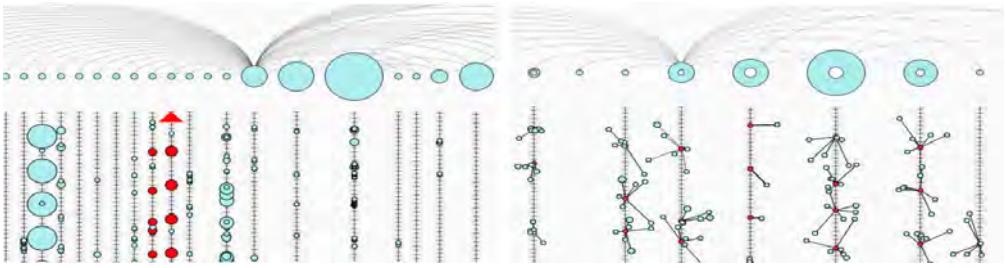


Figure 2.56: A visualization with three levels of geometry, altrnating between structural and temporal information: node-link diagrams, timelines, and arc diagrams [364]

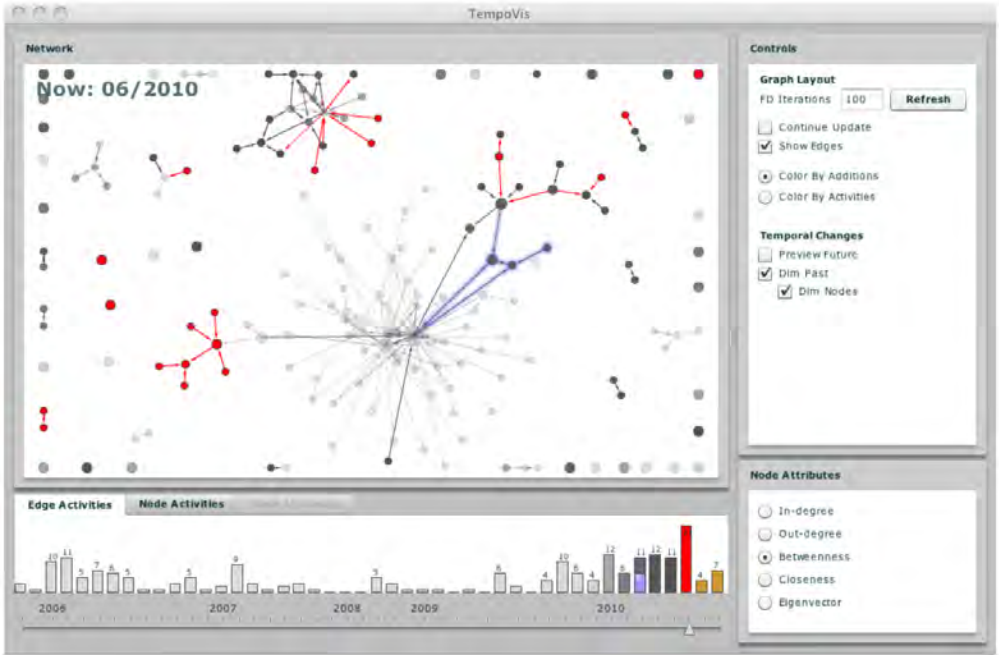


Figure 2.57: Temporal information (specifally, age) associated to vertices and edges is visualized by two retinal variables (hue and brightness) [10]

Retinal variables. When the graph structure is first mapped into geometrical variables, temporal information can be mapped to retinal variables, along a timeline or as an average aggregate.

In node-link diagrams, lifeline-likes visualizations (where temporal information is visually encoded as color along a timeline [304]) can be nested within nodes [339,407] and/or links [319]. In Citespace II [91] each node is divided into rings: the color of each rings represents a time slice, the size of the ring represents the vertex-centric measures computed in that time slice.

If the analysis tasks involve calendars or varying time granularities, then more complex timeline arrangements might be needed, like in pixel-based visualizations, leading to a time matrix nested within each node of a node-link diagram [357], or within each cell of an adjacency matrix [290, 358].

Elements of a node-link diagram can be also colored according to the metaphor of ageing, which we already discussed with regard to animation (Section 2.7.1), but is also applied to static visualization: usually older vertices are shown with lesser emphasis (for example, less saturation, or a fainter color [128]). Multi-hue sequential color scales are also used to visualize the ageing of edges [235], vertices [15, 257], or both [10] (Figure 2.57). Besides ageing, retinal variables can be used to visualize summaries of other temporal attributes, such as time averages [376, 407], differences [249], or temporal patterns of centrality measures [410].

2.7.2.2 Merged

We define merged visualization a visualization where one planar variable is used for temporal information and another one is used for structural information in a balanced fashion, i.e. without any visual prevalence of the one on the other. We distinguish two cases: time is mapped to geometry either by imposition (e.g., a time axis) or implantation (i.e., time as a graph element).

Time as a spatial coordinate (Time imposition) If temporal information is associated with vertices only, a common approach consists in a node-link diagram with a constrained layout: one spatial coordinate is assigned to nodes according to a time axis, the other one is computed by the layout algorithm. Common examples of this timeline-constrained layouts are so-called historiographs [167] and similar techniques to visualize citation networks [268, 382] (see also Figures 2.26c and 2.27). Dörk et al. [110] illustrate the utilization of only one spatial coordinate, by proposing a rectilinear layout (an arc diagram) where nodes are displaced according to timestamps. In EgoNetCloud [260], not single nodes, but rather graph components are arranged in a timeline-constrained layout, complemented with area charts.

When the temporal information is associated to edges, then the structural information can be visualized as an edge list, which is aligned along a Cartesian coordinate, while the other Cartesian coordinate is used as time axis [324]. Van den Elzen et al. [380] propose an efficient reordering of the edge list to optimize the visualization of massive edge sequences. Abello et al. [1] propose an alternative dynamic ranking of edges to encode additional temporal attributes such as recency, persistence, and rate (Figure 2.58a)). Given that the one Cartesian axis is used for the timeline, the other one can accommodate a bipartite layout (or, by using Bertin's words, a diagram with parallel alignment). The visual scalability of the bipartite layout can be increased by image-level edge aggregation (edge splatting [83]), or by data-level temporal aggregation [47]. Gilbert et al. [171] compute dynamic clusters and similarity metrics between clusters at different times, and visualize them in a bipartite layout.

Merged visualization does not necessarily have to exploit two orthogonal axes for encoding temporal and relational information; in other words, the visualization must not necessarily include a timeline. Many approaches, indeed, utilize polar coordinates: Chro-ring [431], for example, uses a node-link diagram with a radial layout, where time is mapped to the angular coordinate. In other approches, time is mapped to the radial coordinate, while the graph structure can be visualized by a space-filling diagram [77, 79, 124], a node-link diagram [78], a node-only diagram [66], a bipartite diagram [76], or an adjacency matrix [391], all with a circular layout.

In many merged visualizations, the graph structure is reduced to vertex-centric measures, which are computed over time and visualized along a time axis, by mapping to a retinal variable

(e.g., lifelines [80], heat maps [175]), or to a geometric primitive (e.g., bar charts [224], stacked area charts [102, 296, 377], line charts [234, 306, 307, 417]. When the geometric primitive which encodes a vertex is a line (a polygonal chain, as well as an open curve), it is also referred to as a storyline. Cui et al. [101] propose an approach to improve the rendering algorithm of such line charts, in order to increase readability at large scales. Dang et al. [103] complement the area chart with additional arc diagrams representing the relational information. The SVEN system [30] computes node rankings over time and visualizes them as slope graphs [372], also complemented with arc diagrams (Figure 2.58b). Zhao et al. [428] utilize a "subway map" metaphor to visualize storylines, while relationships are shown by adjacency matrices (Figure 2.58c).

A large graph can be granulated to display the evolution of graph clusters over time: a common visualization for dynamic graph clusters is the alluvial diagram [318, 328, 365], in which the splitting and merging of clusters resemble confluence and bifurcation of rivers. Muelder et al. [278] use a similar visualization, but encode clusters only by proximity, without merging vertexes into streams. An alluvial diagram can be also complemented with arc diagrams to show relationships between evolving clusters [389]. Beck et al. [47] aggregate the dynamic network temporally, in order to present a series of the most significant bipartite layouts along the time axis.

Time as a graph element (Time implantation) A special class of merged visualization is constructed by considering temporal elements (the references that map graph elements to the time domain by temporal primitives) as if they were graph elements. For example, Thiel et al. [366] display years as vertices into the graph, and utilize a node-link diagram with a force-directed layout to group together all nodes referring to the same year. Analogously, Shi et al. [348, 349] introduce year nodes in a one-and-a-half-dimension layout to visualize dynamic egocentric networks (Figure 2.59a). Hoek et al. [197] utilize arcs connecting data elements (vertices) and temporal elements (Figure 2.59b). Verspoor et al. [393] align temporal elements along axes of a hive plot.

Other cases of visualization, in which time (in particular, ordinal time) is mapped to graph elements, are those visualization techniques featuring vertex trajectories: directed edges connect different instances of the same vertex over time. Oliveira and Gama [293] map two vertex measures to the axes of a scatter plot and then connect vertices over time, thus obtaining a connected scatter plot [188]. Comet plots apply the same mechanism to node-link diagrams with energy-based layout [354].

Van den Elzen et al. [381] propose an approach based on discretization, vectorization and normalization, and dimensionality reduction to reduce a network snapshot into a single point in the space of network spaces; the evolution of the network can then be visualized as a trajectory in this network space.

2.7.2.3 Structure within time

In this type of visualization, the temporal information is mapped to the main geometry, and then the network structure is mapped to a secondary geometry. In the most common case, the main geometry is based on some sort of timeline; then several two-dimension diagrams, representing the graph structure at different time slices, are variously arranged along the timeline. Usual

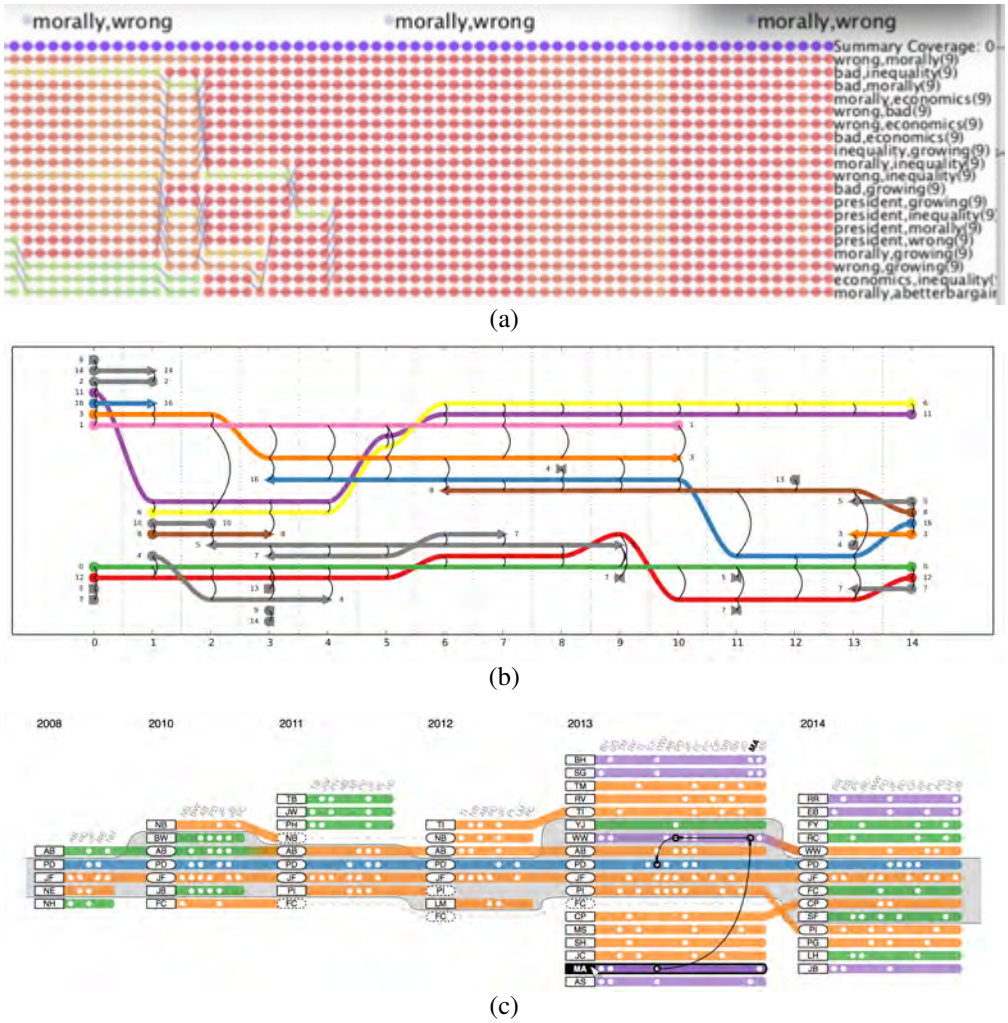


Figure 2.58: In these merged visualizations, the graph structure is mapped to the vertical coordinate, while the timeline is arranged along the horizontal axis: (a) an edge list reordered over time according to dynamic network rankings [1]; (b) a temporal network visualized by a combination of vertex trajectories along a time axis and arc diagrams [30]; (c) combines adjacency matrices with a "subway lines" metaphor [428]

arrangements comprehend **juxtaposition** (the two-dimension diagrams are displaces along a parallel timeline) or **stacking** (the two-dimension diagrams are stacked along an orthogonal timeline, which then constitutes an additional spatial dimension within a prospective view). It is worth noting that an **additional dimension** is only present when the stacked diagrams are kept spatially distinct; if the diagrams are flattened (e.g., the layers are superimposed and merged), then the additional-dimension timeline disappears, and the main geometry does not encode the temporal information any more, but rather the graph structure (as discussed in Section 2.7.2.1),

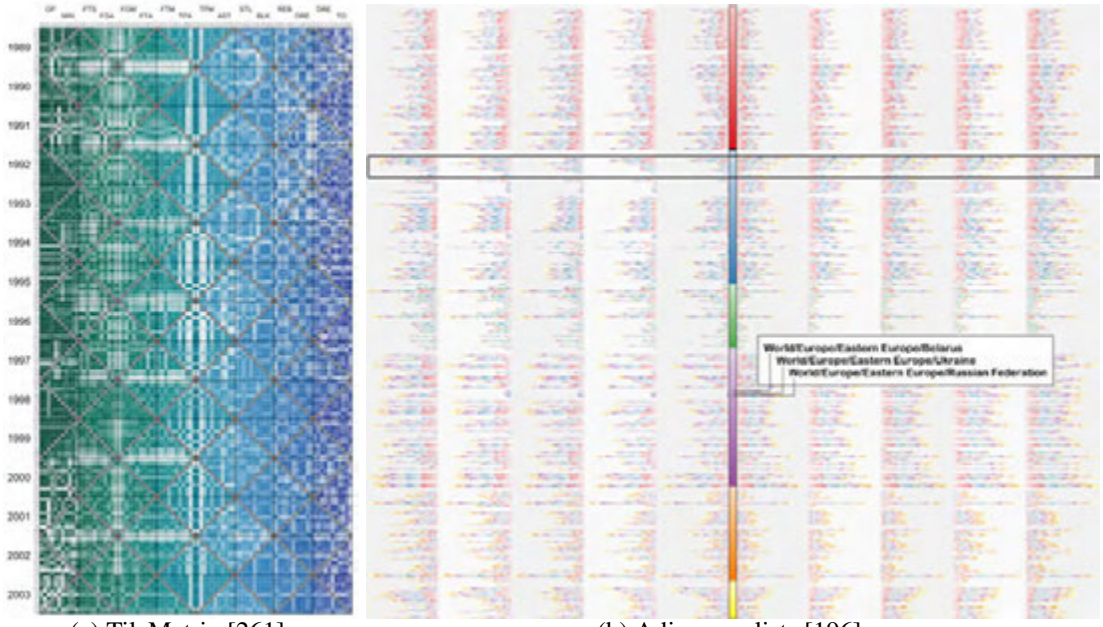
(a) [349] (b) [197]

Figure 2.59: In these merged visualizations, temporal elements are visualized as graph elements: (a) 1.5-dimension egocentric network; (b) parallel arc diagrams

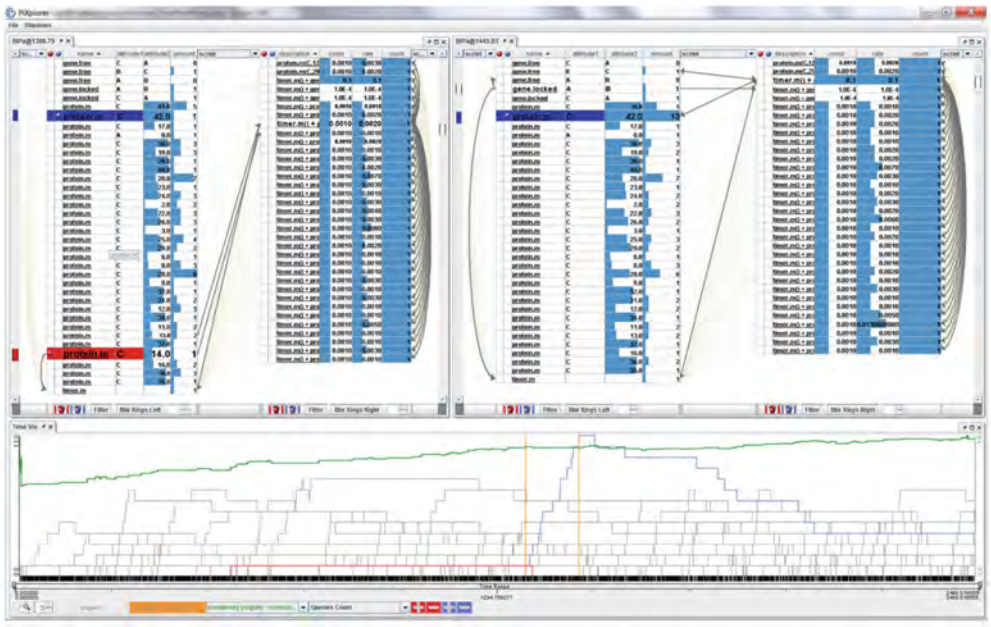
while the temporal information is mapped to retinal variables.

Juxtaposition The juxtaposition paradigm is well suited for time-oriented graphs originally consisting in sequences of graphs, or obtained by time slicing; in this case, each time slice can be visualized separately, and the sequence can be aligned along the time line. The juxtapostion of small multiples has been realized by using different visual endings for graphs. Matrixflow [299], for example, aligns several matrix-based representations. Tilematrix [261] also exploits adjacency matrices but, since adjacency matrices of undirected graphs are diagonal, it utilizes a so-called complementary juxtaposition in order to host multivariate data in a compact arrangement (Figure 2.60a). Other approaches are based on juxtaposition along a time line of visual adjacency lists [196] (Figure 2.60b), also referred to as edge-stacks [81]. The latter technique exploits also color to better distinguish directed edges of different time slices. The most common visual encodings for juxtaposition seems to be the node-link diagram, with an energy-based [370] as well as a radial layout [102, 131]. However, since the juxtaposition paradigm usually dedicates one Cartesian coordinate to the time line, a rectilinear (i.e., one-dimensional) layout the the usual choice; many approaches, therefore, exploit arc diagrams [80, 178]. John et al. [220] propose a juxtaposition of bipartite layouts complemented with arc diagrams; their systems displays only two juxtaposed time slices, selected by the user, while a line chart provides on overview of the entire temporal sequence (Figure 2.60c).

Burch et al. [84, 85] propose a juxtaposition with alternative visual encodings, including adjacency matrices, bipartite layouts, and node-link diagrams; they also integrate an interaction technique by which the user can scroll through time-slices with a flip-book animation. In order to deal with the scale on the temporal dimension, Yang et al. [421] propose an algorithm to detect relevant events (i.e., splitting and merging of clusters) and visualize only significant time slices by juxtaposed node-link diagrams. Manynets [153], conversely, aggregate data from the structural perspective, and visualizes them by juxtaposing several bar charts representing averages and distributions of vertex-centric graph-theoretic measures.



(a) TileMatrix [261] (b) Adjacency lists [196]



(c) Bipartite layouts complemented with arc diagrams [220]

Figure 2.60: Static visualization by juxtaposition

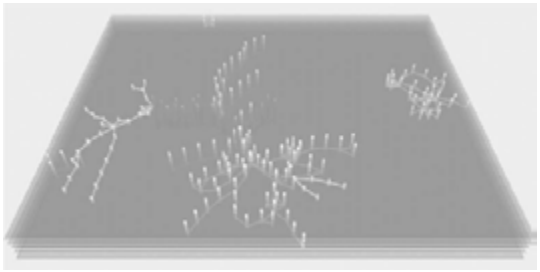
The structure-within-time visualization with a juxtaposition arrangement must not be confused with the merged visualization discussed in Section 2.7.2.2. In the case of streaming edge lists [1,324,380], indeed, at a given point it is not possible to define a network, but only an event or a a set of events; therefore, we cannot talk of juxtaposed time slices. Analogously, in the case of diagrams as defined by Bertin (in which the graph elements are duplicated, such as bipartite layouts [47, 83]), even if the data are sliced, strictly speaking there is no slice juxtaposition, because graph structure is only defined along the axis of replication (coinciding with the time axis), and cannot be defined at a single time point. Conversely, one arc diagram visualizes a full graph structure along a rectilinear layout and, therefore, several arc diagrams aligned along a time axis can be considered a juxtaposition.

Branching time A particular case of juxtaposition occurs when the arrangement of the time domain is not linear and, instead, it contains branches. In this case, the small multiples representing the different time slices are not juxtaposed along a time line, but rather arranged a branching structure. The VANLO system [71], for example, applies this approach to the biology domain, inorder to visualize the evolutive differentiation of protein networks. Another general application is the visualization of graph rewriting [302, 378].

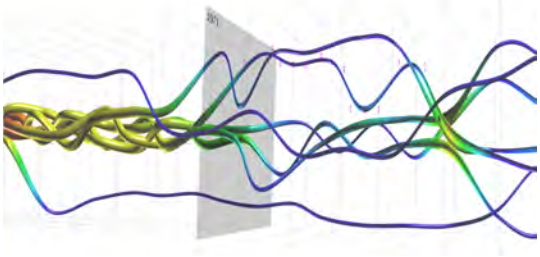
Additional dimension Time-oriented graphs can be visualized in a three-dimensional visualization, where two spatial dimensions are dedicated to the relational information, and a time line is allocated along the third one. In most cases, when the dynamic graph data is partitioned in time slices, each time slice is visualized in a two-dimensional layer, and layers are stacked along the third axes: since data items do not occupy the entire volume, but lay on separate planes, these visualization is generally referred to as a two-and-a-half-dimensional (2.5D) view. Nevertheless, in the more general case, a dynamic graph consisting in a continuous sequence of vertex and edges (without time slicing) leads to full three-dimensional (3D) visualization, like for example in [177]. Moreover, also a time-sliced graph can be visualized in a 3D fashion, for example when data between time slices is interpolated (e.g., [179]).

Most visualization featuring an additional dimension are based on node-link diagrams, but there is also an example of adjacency matrix by Bach et al. [35].

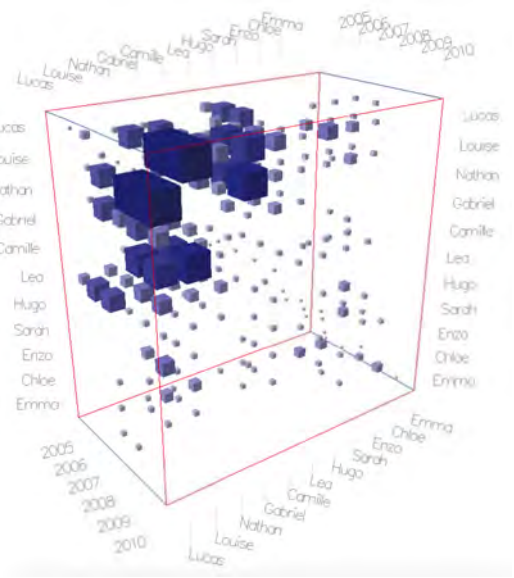
As for node-link diagrams, there are different variants: 2.5D juxtaposition of node-link diagrams with a radial layout [8], an aggregated energy-based layout [311], or a dynamic energy-based layout [149,161]. In some cases, also node trajectories can be shown, i.e. lines connecting node istances between time slices [127, 128, 162]. When the layout is fixed, node trajectories can be visualized as straight lines or tubes, intersecting the time-slicing planes orthogonally [64]. When the layout is dynamic, and trajectories are also interpolated between time slices, then they look like 3D worms [7, 112, 113]; in some cases, edges are not shown, and the evolution of the network structure is represented by these worms moving closer or away from each other [179].



(a) [64]



(b) Dyson [179]



(c) Cubix [35]

Figure 2.61: Static visualization with an additional dimension

2.7.2.4 Combination

Different approaches can be combined in order to exploit the benefits of each of them. In Cubix [35], for example, the user can choose among different arrangements (superimposition, juxtaposition, additional dimension) of matrix-based representations. The Small Multipiles [33] approach combines juxtaposition and a special kind of superimposition, namely piling: each matrix in a pile is reduced into a vector, except for one, which becomes the front cover of the pile. Juxtaposition and two-and-a-half-dimensional view have been combined also in systems based on node-link diagrams [129, 215–217].

Growing polygons [124] are an interesting case, since they combine dynamic and static visualization: the growing polygon is built during the animation by an accumulation metaphor (similar to the sedimentation metaphor [211]), providing details about the current state but also an overview of the past history. A combination of small multiples and flib-book animation is proposed also by Burch et al. [84, 85].

2.7.3 Summary

Figure 2.62 provides an overview on visual encodings for time-oriented graphs, according to the categories introduced and discussed in this section.

		Dynamic	Static															
			Time in Graph					Time and Graph				Graph in Time						
			Geometric		Retinal			Time Imposition		Time implantat.		Juxtaposed		Additional dimension				
			Cartesian	Polar	Timeline	Calendar	Averaged	Cartesian	Polar			Linear	Branching	2.5D	3D			
Adjacency Matrix		11	3			2			1			4		1				
Adjacency List		2										4						
Edge list								3										
Space-filling		7	1					1	2									
Measure-based		1						12		2		1						
Storyline								6						1				
Node-link	Radial	2					1		2			4		1				
	Arc diagram		1					1				3						
	Energy-based	19	10	3	4	1	8	4		5		2	3	11	1			
	Bipartite							3	1			3						
	Other	5								1								

Figure 2.62: A summary of surveyed visual encodings

2.8 Interaction

User interaction is, by definition, a crucial aspect of Information Visualization, defined as "the use of computer-supported, interactive, visual representations of abstract data to amplify cognition." [86, page 6] and as well as Visual Analyicts, defined as "the science of analytical reasoning facilitated by interactive visual interfaces" [367, page 4]. In particular, Thomas and Cook recommend to "develop a new science of interactions that supports the analytical reasoning process" [367, page 9]. Pike et al. have further elaborated on the challenge of establishing a "Science of Interaction" [301]. Nevertheless, the importance of user interaction is generally underestimated visualization of dynamic graphs, as noted by Beck et al. [45]; static visualization is generally considered as a sequence of not-interactive graph drawings, while the most common interaction technique for animation approaches only deals with animation control (e.g., play/pause, or time seeker).

However, user interaction has been investigated in the visualization literature in general. Yi et al. [423] propose a taxonomy of interaction in Information Visualization based on the notion of user intent. Lam [250] introduces a theoretical framework to understand and possibly reduce the costs of interaction. Wybrow et al. [418] review interaction techniques for multivariate graphs and propose a classification based on the Information Visualization Reference Model [86], distinguishing among view level, visual-representation level, and data level. Analogously, von Landesberger et al. [397] categorize interaction techniques for visual analysis of large graphs according to stages in the Information Visualization Reference Model and user actions, following the idea of Elmqvist and Fekete [123] and Bertini and Lalanne [56].

We will discuss interaction techniques for graphs and dynamic graphs according to user intents [423]: 1. *Select*: mark something as interesting 2. *Explore*: show me something else 3. *Reconfigure*: show me a different arrangement 4. *Encode*: show me a different representation 5. *Abstract/Elaborate*: show me more or less detail 6. *Filter*: show me something conditionally 7. *Connect*: show me related items

Yi et al. [423] observe that some other interaction techniques address several user intents and, therefore, it is possible to classify them into multiple categories. In particular, when considering graph data, we notice that the *Connect* user intent is obviously predominant, since it addresses the relational nature of such data and, therefore, several interaction techniques fulfil this intent, either alone or in combination with other intents.

2.8.1 Select

Highlighting, in the stricter sense, is a *brushing* interaction technique, originally developed for scatter plots [48], and then extended and applied to other visualization techniques. Brushing is a "change in the encoding of one or more items essentially immediately following, and in response to, an interaction with another item" [355, p. 235]. In particular, in the case of highlighting, the change may affect hue, brightness, or color. Brushing operates within a view or across multiple views; in the latter case, the interaction technique is better known as *linking and brushing* [226]. Highlighting makes some information stand out from other information; it effectively exploits pre-attentive processing [399], which is the human capability to process visual information prior to, or in the early stage of, focusing conscious attention. Linking and brushing techniques sup-

port two user's intents: *Select* and *Connect*. In the context of graph visualization, highlighting of adjacent nodes upon selection of a certain node (for example, by mouse hovering) is a common interaction technique, also known as *connectivity highlighting* [190]. In the case of static visualization of dynamic graphs, the highlighting technique can be extended in order to fulfil the need of visually linking and synchronizing multiple instances of the same graph entities in different time slices [45], by considering adjacency not only across the graph structure, but also along the time dimension. The connectivity highlighting can be extended further in order to support three user intents: *Select*, *Connect*, and *Encode*. The interaction technique proposed by McGuffin et al. [272], for example, allows the user to select a node in a node-link diagram, define a neighbourhood-radius k in order to extend the selection to the sub-graph of nodes connected in k steps, and then choose a different visual encoding for the selected sub-graph.

2.8.2 Explore

The *Explore* intent corresponds to the will to get additional information. The excentric labeling technique [57] provides additional information about a region of the node-link diagram by showing labels around it, in order to avoid overlapping. Similarly, the in-situ exploration technique by Hadlak et al. [182] shows additional additional data, or additional facets of the same data and fulfils also the *Encode* intent, since the additional information can be shown with a different visual encoding. Pan & zoom is one of the easiest and most common interaction techniques in visualization, and it fulfils the *Explore* user intent. In visualization of graphs, it is combined with the *Connect* intent in order to provide network-aware exploration or, in other words, navigation. The signpost technique [269] utilizes labelled arrows to indicate the direction of the shortest path connecting the visible nodes to off-screen regions of a node-link diagram. The link-sliding technique [277] consists in a guided panning from a selected node to an adjacent one along a visible link. *Explore*, *Connect* and *Reconfigure* are addressed jointly by the Bring & Go technique [277], which after selection of a node brings adjacent nodes closer. The edge lense by Tominsky et al. [368] integrates also a *Filter* intent: besides bringing adjacent vertices closer, it also emphasizes adjacent edges by hiding non-adjacent ones.

2.8.3 Reconfigure

The *Reconfigure* intent is fulfilled by computing a different spatial arrangement of displayed data. It is applied to both matrix-based visualization, by interactive matrix reordering techniques [49, 192] as well as node-link diagrams, by letting users to interactively change or tune the layout algorithm. Dwyer et al. [115], for example, propose means by which users can specify the parameter of constrained layouts. More in general, many graph drawing algorithms allow users to drag-and-drop nodes, and the layout rearranges accordingly, in conformance with the principle of direct manipulation [350]. Besides nodes, also links can be directly manipulated by users; Riche et al. [323] describe the design space of link curvature, identifying four interaction techniques: bundling, fanning, magnets, and legends. GraphDice [58] includes an interaction technique which fulfils both the *Reconfigure* and the *Explore* intent: it allows the user to select a new graph-theoretic index in order to rearrange the scatter plot from one projection to another. In the context of dynamic graphs, Loubier et al. [262] introduce an interactive transition from

the layout of the aggregated graph to the layout of a single time slice; Feng et al. [147] propose a technique to adapt the layout of a dynamic graph in order to increase the stability of nodes of interest, which have been selected by the user.

2.8.4 Encode

The *Encode* intent is addressed by interaction techniques which enable the choice of alternative visual encodings. In NodeTrix [194], for example, the user can trigger animated transitions between a matrix-based visualization and a node-link diagram. Analogously, Zhao et et. [429] provide six animated transitions between a node-link diagram and a space-filling visualization. The color lense [122] is an interaction technique that enables a local adaptation of the color encoding, and has been applied to different visualization techniques including node-link diagrams.

2.8.5 Abstract/Elaborate

The *Abstract/Elaborate* intent corresponds to decreasing or increasing the level of visualized details; in the context of graph visualization, it relates to the control of graph coarsening. This form of semantic zooming has been applied to matrix-based visualization [4, 121, 192] as well as node-link diagrams [21, 22, 106, 415]. Those interaction techniques, which enable users to select different graph-theoretic measures to be visualized as a summary of the network structure [58, 94, 190, 297, 298, 394, 432], can be understood as fulfilling both the *Explore* and the *Abstract/Elaborate* user intents. The same holds for on-demand motif analysis [264, 396], as well as graph-theoretic measures computed over time [106, 306, 307]. These interactions usually operate at the data level but, analogously to the non-interactive case (see Section 2.6.1.2), the abstraction can also operate at the image-rendering level [433].

2.8.6 Filter

In graph visualization, the *Filter* intent is often combined with the *Connect* intent, since filtering criteria are usually based on adjacency and geodesic distance [190], or other degree-of-interest functions based on graph-theoretic metrics for static [384] or dynamic graphs [2].

2.8.7 Connect

Besides all above mentioned interactions, where the *Connect* intent deals with the relational nature of graph data and is fulfilled in combination with other intents, the *Connect* intent is also addressed in the context of coordinated multiple views. In this case, the visualization allows the user to interactively show connections between data items in different views. For example, in NetLens, the user can highlight connection between bar charts for each partition of a bipartite graph [224]; in Detagler [321], highlighting shows connections across two different visual encodings of multiplex networks (named substrate view and catalyst view). In the case of time-oriented graphs, brushing and linking usually operates between a temporal view and a relational view [10, 106]. In VisLink [99], the *Connect* intent is fulfilled not by brushing & linking (i.e., exploiting retinal characteristics to show connected items), but rather by drawing explicit links between connected items.

2.9 Task taxonomies

A profound understanding of analytical tasks is a necessary prerequisite to design novel visualization techniques as well as evaluate existing ones.

In the visualization literature, there is not even a consensus about the terminology, and researchers use several terms with overlapping or conflicting meanings: task, question, problem, objective, activity, action, operation. Rind et al. [32] try to untangle the confusion in terminology, by introducing a conceptual space of tasks along three dimensions: abstraction (domain specific/domain independent), composition (low level/high level), and perspective (how/why).

Shneiderman [351], in his task by data type taxonomy of information visualization, elicits seven types of actions: overview, zoom, filter, details-on-demand, relate, history, extract.

Bertin [55] characterizes analytical question along two dimensions: the question type (which unknown information has to be found) and the reading level (elementary, intermediate, or global, depending on whether the questions involves single elements, group of elements, or the entire dataset, respectively).

The task taxonomy by Andrienko & Andrienko [18] distinguishes two types of tasks: elementary tasks (lookup, comparison) and synoptic (pattern identification and comparison); moreover, they characterize tasks according to involved referrers (space, time, population) and several attribute or pattern types.

Lee et al. [253] introduce a specific task taxonomy for graph visualization. They structure it along three axes: the type of graph entities involved (nodes, links, paths, graphs, connected components, clusters, and groups), the perspective (topology-based, attribute-based, browsing, and overview), and the level of composition (low, medium, and high).

Ahn et al. [9] propose a task taxonomy for dynamic graphs along three different axes: graph entities (node/link, group, or network); temporal features (individual events, shape of changes, or rate of changes); and properties (structural or domain-specific).

Kerracher et al. [232] extend and instantiate the task framework by Andrienko & Andrienko, and define a task taxonomy and task design space for visualisation of time-oriented graphs.

Conversely, Bach et al. [34] build upon the task taxonomy for temporal data in geographic information system by Peuquet [300]; they describe each task can be understood as a question containing references to two dimensions and requiring an answer in the third one. In this way, they distinguish between topological tasks (find a graph structure *where* a given event is present at a given time), temporal tasks (find the time *when* a given entity is present at a given location) and behavioural tasks (find *what* type of event happens at a given time to a given graph structure).

Archambault and Purchase [25] structure their taxonomy for time-oriented graphs along two dimension, mostly aiming at assessing the importance of the mental map. They distinguish between local and global tasks (depending on whether the focus is on single nodes and links, or on the entire graph), and between distinguishable and undistinguishable tasks (depending on whether graph entities need to be distinguished from each other, or can be aggregated).

A general task taxonomy for multivariate networks can be found in [308].

2.10 Evaluation of graph visualization

Most of the techniques for visual analysis of graphs and time-oriented graphs, which we surveyed in previous sections, were validated in some way: graph drawing techniques are usually validated with respect to their computational complexity and performance; many visualization systems are validated by case studies or, more precisely, by usage scenarios, which are performed by the developers themselves and, therefore, are a weaker form of validation [345]; fewer techniques were also validated by user studies involving real users and domain experts. In this section, we focus on works which are mainly focused on evaluation, or whose findings can be generalized and utilized as design guidelines.

As for visual encodings, Ghoniem et al. [169] perform a comparative evaluation between matrix-based representations and node-link diagrams: they found that matrix-based visualization scales better with size and density, but node-link diagrams perform better for path-finding tasks. Keller et al. [230] run a similar user study which confirmed these findings.

With respect to node-link diagrams, Purchase and Ware and colleagues [314, 315, 315, 401] investigate the perception of graph drawings to provide empirical groundings to aesthetics criteria (see Section 2.6.1.1). Besides existing criteria, such as readability and memorability, Nguyen et al. [286] propose faithfulness as a novel criterion for validating node-link diagrams, while Huang et al. [208] focus on cognitive load. Moreover, Huang and colleagues proposed an quality metric combining several aesthtics criteria for both static [209] and dynamic graphs [210]. Other researchers evaluated, by user studies, specific aspects of node-link diagrams, such as directed edges [79, 200, 202, 283], multivariate edges [341], curved edges [419, 420], and node groups [219, 337]. As for matrix-based visulization, the Magnostics framework [49] provides a set of image-based descriptors which can be utilized to evaluate matrix reordering algorithms in terms of resulting visual patterns .

The Mental Map Many existing algorithms for drawing dynamic graphs ensure the layout stability in order to preserve the user's mental map of the graph [275]. This stability can be seen as an additional aesthetic criterion for dynamic graphs [44], prescribing that the placement of nodes should change as little as possible [97].

The mental map has been originally introduced to ensure layout stability in situations in which the user can interactively edit the graph by adding or deleting edges and vertices [275]. Afterwards, the concept has been applied to dynamic graph drawing, in both online and offline cases, and in particular to dynamic visualization (i.e., animation, see Section 2.7.1). However, the problem of the layout stability holds as well for static visualization, for example small multiples [23, 24, 28].

The utility of this dynamic aesthetics has been highly disputed in literature and several evaluations have been conducted, from both the algorithmic and the perceptual perspective. As for the algorithmic evaluation, Brandes & Mader [68] compare three approaches (Figure 2.63): *aggregation* (fixed nodes positions are obtained from the layout of an aggregate of all graphs in the sequence, achieving maximum stability), *anchoring* (nodes are attracted by reference positions), and *linking* (nodes are attracted by instances of themselves in adjacent time slices of the sequence). Their results suggest that the generally preferable approach is linking, that is also

the most computationally demanding; a faster alternative is anchoring to an aggregate layout initialized with the previous one in the sequence. Many user studies have been performed to empirically assess the importance of mental map preservation for readability, memorability, or interpretability of dynamic graphs. Archambault & Purchase [25, 27] review many of them. In an early study about readability of direct acyclic graphs (DAGs), Purchase et al. [312] found that the layout stability is beneficial for several tasks. Conversely, a similar study about readability of DAGs by Zaman et al. [426] found no significant effect of the layout stability. Purchase & Samra [313] tested several interpretation tasks for directed graphs and found that extremes (no stability or maximum stability) are better than a medium stability. Conversely, Saffrey & Purchase [336], by investigating readability and interpretability of directed graphs, found that the layout stability does not provide any advantage and can be even harmful for certain tasks. While all the evaluations mentioned so far were conducted on static visualization, Ghani et al. [168] studied the effects of layout stability in readability of animated node-link diagrams, finding that a fixed layout (maximum stability) outperforms a force-directed layout with no stability. Archmbault & Purchase [26] found that a stable layout helps users with orienting themselves in a dynamically evolving graph, while there is no significant difference between small multiples and animation. By studying the memorability of small-multiples node-link diagrams, Archambault & Purchase [24] also found that maximum layout stability is the best condition.

Evaluating other aspects of dynamic graphs Besides the importance of preserving the mental map in node-link diagrams (Section 2.10), another issue which attracts the interest of researchers is the comparison between dynamic visulization and static visualization for dynamic graphs, the latter being usually based on small multiples in a juxtaposition arrangement. The controlled experiment by Farrugia and Quigley [133] demonstrates that static drawings outperform animation in terms of task completion time. Archambault et al. [23], in an analogous user study, observe that small multiples are generally faster, but more error-prone for certain tasks; moreover, mental map preservation has little influence on both response time and error rate. In a more recent study, Archambault and Purchase [28] find that, if the dynamic graph drawings are not stable and highlighting cannot be used, animation can provide benefits over small multiples. Boyandin et al. [62] also conduct a comparative evaluation of animation versus small multiples in the context of flow maps. They observe that, with animation, users could identify more changes in adjacent time slices, while small multiples are better suited for analyzing long time periods. Moreover, they suggest that switching from one view to the other might lead to an increase in the numbers of findings.Archambault et al. [29] evaluate also the readability of difference maps for dynamic graphs, while Chen & Morris [92] compare approaches relying on minimum spanning trees and pathfinder networks.

Evaluating interaction A few user studies focus also on interaction. An experiment by Ware & Bobrow shows that interactive highlighting can efficiently support visual queries on graphs [400], while Rey & Diehl [322] investigate the effects of two interaction techniques for animated visualization: interactive control of the animation speed and a tooltip showing details on demand. They found that the speed control does not provide a significant benefit, and the tooltip is outperformed by a visualization having labels always visible.

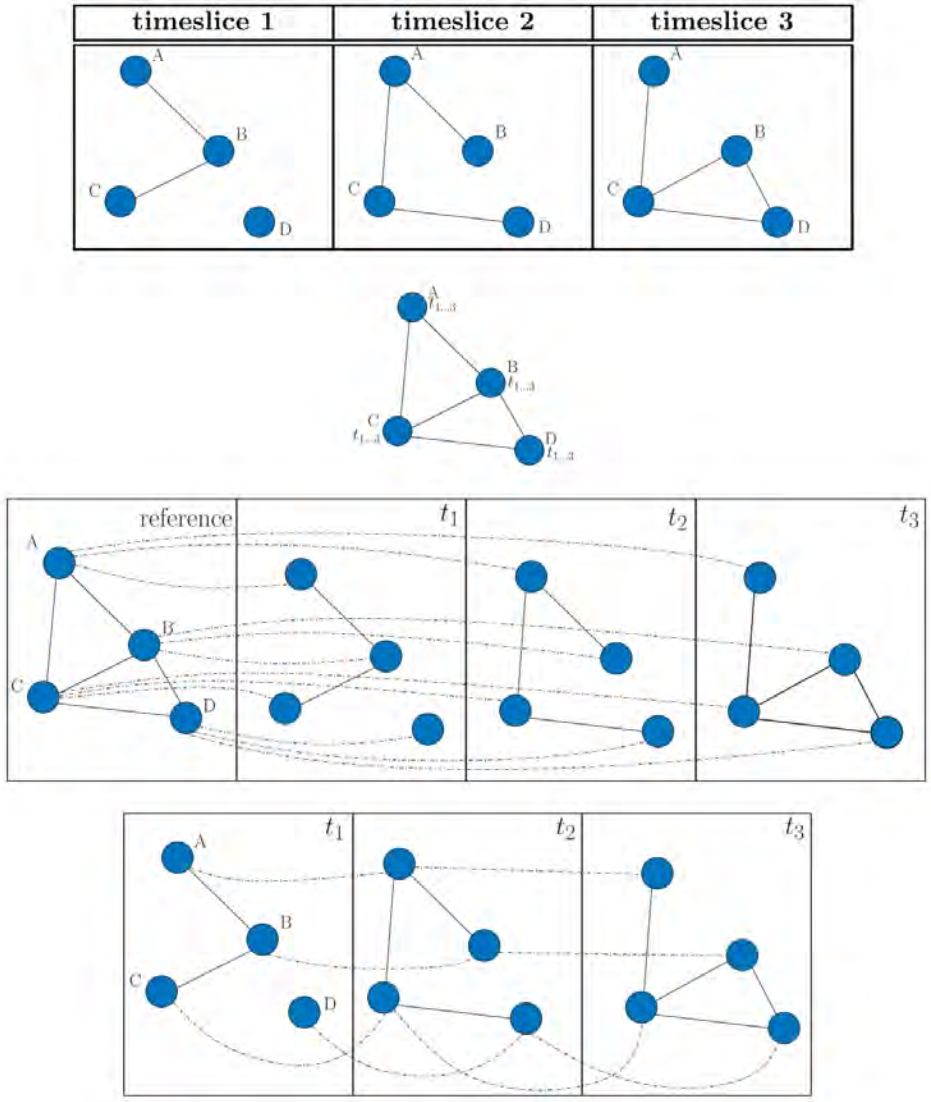


Figure 2.63: A time-oriented graph, and three approaches for mental map preservation: aggregation, anchoring, and linkin [68] (image adapted from [25])

2.11 Limitations of existing approaches and open challanges

In the previous sections, we discussed many existing approaches for visual analysis of time-oriented networks. Most of them have clear benefits and contributed the advancement of this research area. The main limitations of many existing approaches is the lack of a seamless integration of visualization, automated analysis, and user interaction components, which is at the core of the visual analytics paradigm. This lack of integration is probably due to the development of specific techniques, without an unifying multidisciplinary approach, and is reflected by the

literature surveys produced within the single research communities, as well as by the wording adopted therein. In the graph drawing community [105] research has mainly dealt with algorithms for computing static (i.e., non-interactive, lacking a focus on user interaction) drawings; conversely, dynamic graphs have been mainly addressed by dynamic visualization (i.e., animation), without exploring alternative ways of visually encoding time and time-oriented information. Network analysis has developed measures and methods for automated analysis of graph data, but with a minor resort to visual means, limited to standard charts and diagrams. Moreover, most of the automatic analysis attempts to combine relational and temporal aspects exhibit the same shortcoming: these two aspects are considered sequentially during the computation, and not simultaneously. Either a relational feature (e.g., degree) is computed for each time point, and then analysed over time (summed, averaged, compared), or a temporal feature is considered (duration, attenuation), and then used to compute a structural metric. Information visualization has put a stronger emphasis on visual encoding and user interaction [195]. However, especially when it comes to dynamic graphs, there has only been a partial integration of automated analysis methods or specific interaction techniques. The survey on visualization of dynamic graphs by Beck et al. [45], as well as the design space by Kerracher et al. [231] make at most a passing reference to interaction and automated analysis. The survey by von Landesberger et al. [397], which is informed by the the visual analytics paradigm (see Figure 2.64), out of more than hundred surveyd papers, reports only three papers which belong into the intersection of the three areas and apply a fully integrated approach to dynamic networks; two of them [306, 307] utilize line charts to show the evolution of graph measures over time, the third one [396] exploits difference highlighting based on graph motifs. In more recent years, a few other papers have been published tackling visual analysis of dynamic graphs from a multidisciplinary approach, and the research communities have come closer and have mutually benefited from each other, but the challenge of designing, developing, and validating fully integrated visual analytics solutions for time-oriented data is still open.

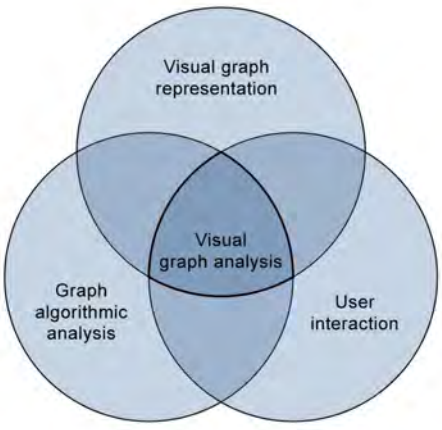


Figure 2.64: Visual graph analysis as a combination of visual graph representation, user interaction, and graph algorithmic analysis [397]

Part II

The proposed solution

CHAPTER 3

Analysis[1]

In this chapter, we introduce techniques for automated analysis of network data, which together with visualization and interaction techniques constitute our integrate visual analytics approach. At first, we describe analytical measures for static networks, derived from graph theory and applied to social network analysis and mining. Secondly, we introduce a novel measure for dynamic networks.

3.1 Automated analysis of static networks

According to a well-established theory of social network analysis [402], the centrality of a person is a measure of that person's prominence, i.e. involvement: prominent persons are persons who are considerably involved in relationships with other persons and, therefore, are more visible to others. Centrality can be considered an estimator of social prestige.

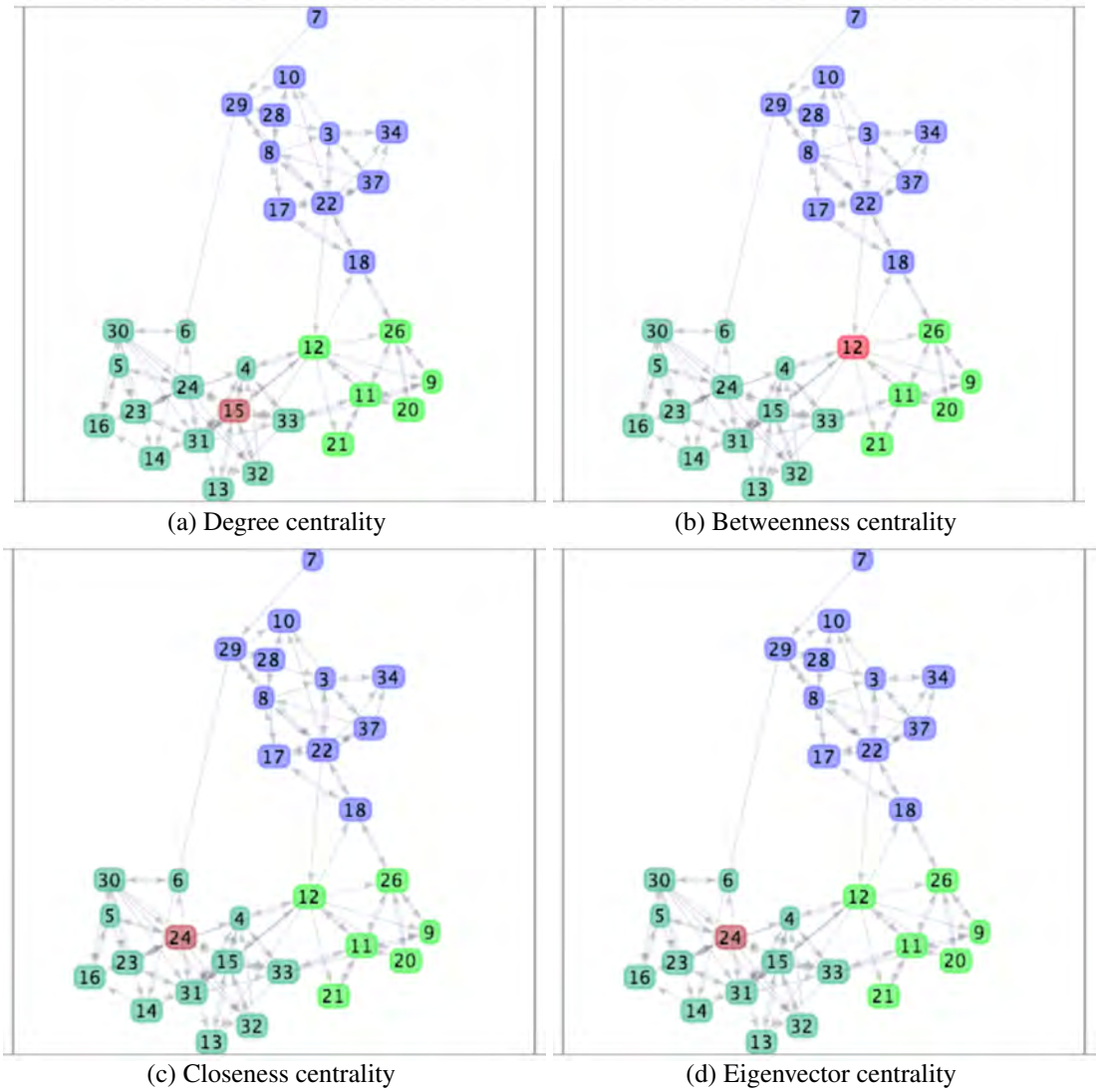
In Section 2.3 we have introduced different definitions of centrality from graph theory. We implemented them all in the automated analysis component of our prototype, since each of them has a slightly different semantics and is commonly used in social network analysis [402].

Degree centrality is the simplest centrality measure; according to its definition, the most central person in a social network is the person with the most ties to other persons [152]. Degree centrality only focuses on direct (i.e., adjacent) relations and, therefore, can be considered a local centrality measure.

Conversely, betweenness centrality is a global centrality measure, based on shortest paths between all actors. The most central person is the middle person, standing between other persons and, therefore, capable of controlling interactions between non-adjacent persons [152].

Closeness centrality is also a global centrality measure, based on shortest paths between all nodes. In this case, the most central person is the one who can quickly interact to all others, by being the closest to them [43, 252].

[1] The main contents of this chapter have been presented at the IEEE/ACM International Conference on Advances in Social Network Analysis and Mining [144]



(a) Degree centrality

(b) Betweenness centrality

(c) Closeness centrality

(d) Eigenvector centrality

Figure 3.1: In this example network, node 15 has the highest degree centrality, node 12 has the highest betweenness centrality, node 24 has the highest closeness centrality as well as the highest eigenvector centrality

Eigenvector centrality is based on the idea that the most prominent person is the one with the most connections to other prominent persons. It can be understood as an indicator of how quickly a person can interact with others as well as how much power a person has over other's interactions [60]. In Figure 3.1, we show an real-world organizational network as a node-link diagram and highlight in red the most prominent person according to the different notions of centrality.

3.2 Automates analysis of dynamic networks

The analysis of dynamic social networks is an emerging research topic. While several methods, both computational and visual, have been proposed for, and applied to, static networks, there is still the lack of a well-established set of methods supporting the analysis of the evolution of a network.

In particular, we might ask ourselves which type of measures are better suited for dynamic networks, and which level of detail/abstraction they shoud have. One possibility is to compute and highlight all single changes of nodes and links between two subsequent time steps; another one is to compare structural metrics and visualize their trend over time. The former option points out each and every single change, with the disadvantage that, for large and dense networks, they might be too many, and hard to be understood from a global perspective; the latter only provides aggregated structural metrics, which lack local details and might hide certain changes. For example, if we only look at the variation of the degree centrality of a certain node, we might not see any change of its relationships if the number of added links is equal to the number of deleted ones; similarly, if we look at the variation of the betweenness centrality or the closeness centrality we might not see anything if changes are symmetric. Conversely, the highlighting of an appearing or disappearing link between two nodes might not help the analyst to understand the impact this change has on the connectivity of the overall structure.

To address the afore-mentioned limitations of existing approaches, we define a novel metric for dynamic networks, discuss its properties and meaning, and provide the algorithm for its computation. We introduce also a set of derived metrics, and show their application to a small exemplary case.

3.2.1 Change centrality

Given a discrete-time dynamic network $\mathcal{G} = G(V, E, T)$, a node $i \in V$ and two time points t_1 and t_2, we define the **1-step change ratio**:

$$r_{t_1,t_2}(i) = \frac{|N_{t_1}(i) \Delta N_{t_2}(i)|}{|N_{t_1}(i) \cup N_{t_2}(i)|}$$

where $N_t(i) = \{\forall j \in V : d_t(i, j) = 1\}$ is the set of nodes connected to node i in 1 step at time t, where $d_t(i, j)$ is the geodesic distance between node i and node j at time t. Thus, the 1-step change ratio of node i at time t_1, t_2 is defined as the ratio between the cardinality of the symmetric difference of the sets of 1-step neighbours of node i at t_1 and t_2 and their union. It can also be seen as the ratio between the number of links added and removed and the number of links added, removed, and remained:

$$r_{t_1,t_2}(i) = \frac{added + removed}{added + removed + remained}$$

Considering the example network of Figure 3.2, node B keeps its links to A and C, but looses its link with D; it looses one of its three connections, then its 1-step change ratio is $\frac{1}{3}$. Node D and E loose all their connections, then their values of 1-step change ratio are both 1. Nodes A and C do not encounter any change, then their values are both zero (see also Table 3.1).

It is worth observing that the 1-step change ratio is the complement to one of the Jaccard similarity index [218], defined as the ratio between the intersection and the union:

$$r_{t_1,t_2}(i) = 1 - J = 1 - \frac{|N_{t_1}(i) \cap N_{t_2}(i)|}{|N_{t_1}(i) \cup N_{t_2}(i)|}$$

We name it change ratio since its value is minimum and equal to zero when there is no change in the 1-step neighbours of the node i from t_1 to t_2, and it is maximum and equal to 1 when all the neighbours change. It is also worth noting that it is symmetric:

$$r_{t_1,t_2}(i) = r_{t_2,t_1}(i)$$

Now, let us generalize it considering the neighbours reachable in n steps. Given a node $i \in V$ and two time points t_1 and t_2, we define the **n-step change ratio** of i between t_1 and t_2:

$$r^n_{t_1,t_2} = \frac{\left|N^n_{t_1}(i) \Delta N^n_{t_2}(i)\right|}{\left|N^n_{t_1}(i) \cup N^n_{t_2}(i)\right|}$$

where $N^n_t(i) = \{\forall j \in V : d_t(i,j) = n\}$ is the set of nodes connected to node i in n steps at time t. In particular, for $n = 0$, by noting that $d_t(i,i) = 0$, we obtain $r^0_t = i$.

Finally, we define the **change centrality** of node i between time points t_1 and t_2 as a linear combination of all the n-step change ratios:

$$C_{t_1,t_2}(i) = \sum_{n=0}^{e_i} a_n R_{t_1,t^n_2}(i)$$

where $e_i = \max_{t \in \{t_1,t_2\}} e(i)$ is the maximum eccentricity of node i and a_n are linear coefficients.

The change centrality of a node is a measure of the change of its connections over time, taking into account its adjacent nodes, the adjacent nodes of the latter and so on. The weight of changes of near and far neighbours depends on the coefficients of the linear combination. By choosing coefficients that decrease with n, the changes of farer neighbours will have smaller weight on the total measure. In particular, if we put $a_n = \left(\frac{1}{2}\right)^{n+1}$, because of the convergence of the sum of the geometric series we will obtain a rational-valued, non-negative, normalized metric:

$$C_{t_1,t_2}(i) = \sum_{n=0}^{e_i} \frac{1}{2^{n+1}} r^n_{t_1,t_2}(i) = \frac{1}{2} \sum_{n=0}^{e_i} \frac{1}{2^n} r^n_{t_1,t_2}(i) \leq 1$$

Thus, the change centrality of a given node will be equal to zero if no changes have occurred in the connected component that node belongs to. It will have a value greater or equal to 0.5 if that node is present in only one of the two time steps considered. It will get closer to 1 if the latter is true and the network diameter gets larger.

Considering the network of Figure 3.2 and the corresponding values of the change centrality (Table 3.1), we observe that the nodes with larger values of change centrality are D (which is present in only one of the two time steps) and E (which looses all of its connections). B has an

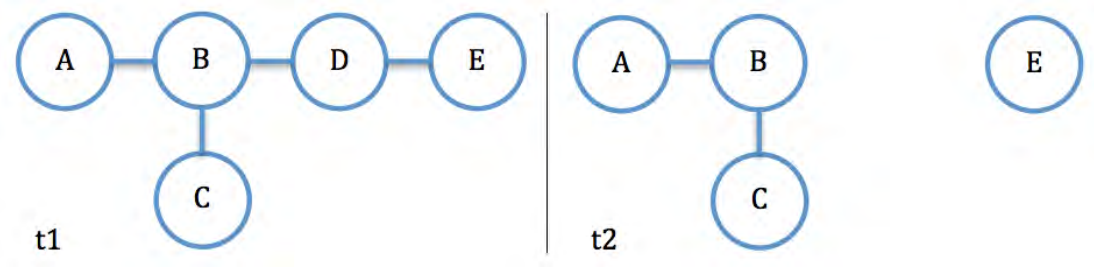


Figure 3.2: A dynamic network at two subsequent time steps.

intermediate value, since it looses one out of its three 1-step neighbours. A and C are the nodes with the lowest change centrality, because they have changes only in their farther neighbours.

The change centrality is a centrality metric in the sense that it measures how central is a node with respect to the network change, taking into account whether the node itself has changed (appeared/disappeared), whether its neighbours have changed, and which is the ratio between stable and changing neighbours. Moreover, it reflects increasing and decreasing connections, but also zero-balance replacements, that might have tiny importance for the overall network structure, but are important when analysing ego networks. Furthermore, it gives a measure of how local changes affect farther and farther nodes. For this aspects, the change centrality is a node-level metric that combines local and global features, from a relational perspective as well as from a dynamic perspective.

3.2.2 Change eccentricity, radius, diameter

Considering the definitions of the previous section, we derive additional measures of change at both the node and the network level. Given a node $i \in V$ and two time points t_1 and t_2 we define the **change eccentricity** of i between t_1 and t_2:

$$E_{t_1,t_2}(i) = \min \left\{ \forall n : r^n_{t_1,t_2}(i) \neq 0 \right\}$$

that can be seen as the geodesic distance between node i and the nearest occurring change. Its value is equal to zero if the node itself has changed (appeared/disappeared), and increases as i gets farer from changes. We put it equal to -1 (to represent infinite distance) if no change has occurred in the set of nodes reachable from i. Analogously to the classic node eccentricity, on the basis of the change eccentricity we define two integer-valued network-level metrics, the **change radius** and the **change diameter**:

$$R_{t_1,t_2} = \min_i E_{t_1,t_2}(i)$$

$$D_{t_1,t_2} = \max_i E_{t_1,t_2}(i)$$

The change radius is equal to zero if at least one node has appeared or disappeared, it is equal to 1 if only links have changed. The change diameter is a measure of the localization of change within the network: the more concentrate is change, the larger is the change diameter. Both

Table 3.1: The values of change centrality, change eccentricity, change diameter, change radius, and stable center for the simple dynamic network of Figure 3.2

	1-step change ratio
$r_{t_1,t_2}(A)$	$0 \left(= \frac{\lvert\emptyset\rvert}{\lvert B\rvert}\right)$
$r_{t_1,t_2}(B)$	$0.\bar{3} \left(= \frac{\lvert D\rvert}{\lvert A+C+D\rvert}\right)$
$r_{t_1,t_2}(C)$	$0 \left(= \frac{\lvert\emptyset\rvert}{\lvert B\rvert}\right)$
$r_{t_1,t_2}(D)$	$1 \left(= \frac{\lvert B+E\rvert}{\lvert B+E\rvert}\right)$
$r_{t_1,t_2}(E)$	$1 \left(= \frac{\lvert D\rvert}{\lvert D\rvert}\right)$
	Change centrality
$C_{t_1,t_2}(A)$	$0.125 \left(= \frac{1}{2}\frac{\lvert\emptyset\rvert}{\lvert A\rvert} + \frac{1}{4}\frac{\lvert\emptyset\rvert}{\lvert B\rvert} + \frac{1}{8}\frac{\lvert D\rvert}{\lvert C+D\rvert} + \frac{1}{16}\frac{\lvert E\rvert}{\lvert E\rvert}\right)$
$C_{t_1,t_2}(B)$	$0.208\bar{3} \left(= \frac{1}{2}\frac{\lvert\emptyset\rvert}{\lvert B\rvert} + \frac{1}{4}\frac{\lvert D\rvert}{\lvert A+C+D\rvert} + \frac{1}{8}\frac{\lvert E\rvert}{\lvert E\rvert}\right)$
$C_{t_1,t_2}(C)$	0.125 (like A)
$C_{t_1,t_2}(D)$	$0.875 \left(= \frac{1}{2}\frac{\lvert D\rvert}{\lvert D\rvert} + \frac{1}{4}\frac{\lvert B+E\rvert}{\lvert B+E\rvert} + \frac{1}{8}\frac{\lvert A+C\rvert}{\lvert A+C\rvert}\right)$
$C_{t_1,t_2}(E)$	$0.4375 \left(= \frac{1}{2}\frac{\lvert\emptyset\rvert}{\lvert E\rvert} + \frac{1}{4}\frac{\lvert D\rvert}{\lvert D\rvert} + \frac{1}{8}\frac{\lvert B\rvert}{\lvert B\rvert} + \frac{1}{16}\frac{\lvert A+C\rvert}{\lvert A+C\rvert}\right)$
	Change eccentricity
$E_{t_1,t_2}(A)$	2
$E_{t_1,t_2}(B)$	1
$E_{t_1,t_2}(C)$	2
$E_{t_1,t_2}(D)$	0
$E_{t_1,t_2}(E)$	1
	Change radius
r_{t_1,t_2}	0
	Change diameter
D_{t_1,t_2}	2
	Stable center
S_{t_1,t_2}	$\{A, C\}$

the change radius and the change diameter are set to -1 if no change has occurred within a connected network.

We can also define a **stable center** for each time step, as the set of all the nodes that have a change eccentricity equal to the change diameter:

$$S_{t_1,t_2} = \{\forall i \in V : E_{t_1,t_2}(i) = D_{t_1,t_2}\}$$

In Table 3.1 we show the values of change eccentricity, change diameter, change radius, and stable center for the simple dynamic network of Figure 3.2. Node D is the node where the change is localized: its change eccentricity, i.e. its distance from the change, is zero, that is also the change radius of the network. Nodes B and E are close to the disappearing node, so their

change eccentricity is equal to 1. Nodes A and C are 2 steps far from the change, so the value for both is equal to 2; it is the maximum value, then it is also the change diameter, and identifies these nodes as the stable center of the network.

3.2.3 Algorithms

The change centrality and the change eccentricity can be computed using the following algorithms:

ChangeCentrality(G_1,G_2)

for $i \in V(G_2)$ **do**
  if $i \in V(G_1)$ **then**
    $CC_i = 0$
  else
    $CC_i = 1$
  end if
  for all $k \in [1..diameter]$ **do**
    $\bigcup_{ik} = V(G_1) \cup V(G_2)\ \forall v \in V : d(i,v) = k$
    $\bigcap_{ik} = V(G_1) \cap V(G_2)\ \forall v \in V : d(i,v) = k$
    if $\bigcup_{ik} \neq \emptyset$ **then**
      $CC_i += (|\bigcup_{ik}| - |\bigcap_{ik}|) \,/\, |\bigcup_{ik}|$
    end if
  end for
end for
return $[CC]$

ChangeEccentricity(G_1,G_2)

for $i \in V(G_2)$ **do**
  $CE_i = -1$
  if $i \notin V(G_1)$ **then**
    $CE_i = 0$
  else
    for $k \in [1..diameter]$ **do**
      $\bigcup_{ik} = V(G_1) \cup V(G_2)\ \forall v \in V : d(i,v) = k$
      $\bigcap_{ik} = V(G_1) \cap V(G_2)\ \forall v \in V : d(i,v) = k$
      if $\bigcup_{ik} - \bigcup_{ik} \neq 0$ **then**
        $CE_i = k$
        $k = diameter + 1$ {break}
      end if
    end for
  end if
end for
return $[CE]$

CHAPTER 4

Visualization[1]

In this chapter we present the visualization components of our visual analytics approach. In particular, we describe the visual encodings for both relational and temporal aspects of dynamic network data. Furthermore, we discuss how the results of automated analytical methods can be utilized to enrich the visualization.

4.1 Visual encoding

The first design decision to be taken when dealing with the visualization of networks concerns the visual encodings of entities and relationships, i.e. their mapping to planar primitives. In our case we chose a node-link diagram because they are the most popular kind of visualization for dynamic networks and consequently they require a shorter learning period to be effectively used than other forms of representation. We also considered matrix-based visualizations, which are popular in different contexts and might also be easy to understand, and scale better with large and dense networks; nevertheless, we excluded them since they are not efficient for visualizing paths and we do not focus on very large networks.

4.2 Dynamic layout

Once we have chosen a specific implantation (a node-link diagram), the next step is the choice of the imposition, i.e. the graph layout. The layout is a very important aspect for designing network visualizations, since position is one of the most prominent visual variables. We aimed for a layout that enhances the perception of both, the relational aspects (the network structure) and the temporal aspects (the network evolution). Here we find the well-know conundrum already introduced in Section 2.10: stability versus consistency. An additional requirement was the high

[1] Some contents of this chapter have been presented at the IEEE/ACM International Conference on Advances in Social Network Analysis and Mining [137] as well as at the International Conference on Knowledge Management and Knowledge Technologies, Special Track on Theory and Applications of Visual Analytics [138]

interactivity of the visualization, like for example the direct manipulation of the networks, which involves the possibility for the user to drag-and-drop single nodes while the layout automatically reacts to these changes and adjusts accordingly. Moreover, we assume that a simple physical metaphor would further enhance the comprehension for non-expert users while algorithms based on more formal mathematical concepts would require longer learning periods. According to this last consideration we adopted a force-directed algorithm using the spring-embedder metaphor: nodes are modelled as repulsing particles while edges are modelled as elastic springs. This physical model roughly ensures that nodes with more connections tend to occupy a more central position, while nodes with fewer connections are pushed towards the periphery; moreover, it clusters connected nodes.Then, we adopted an incremental and continuously running algorithm for the layout that allows the users to directly manipulate the diagram and also enables more sophisticated interaction techniques (see Section 5.3).

In order to ensure stability, we adopted a linking approach. We discarded the use of predetermined positions (they would not allow for free direct manipulation), then we discarded anchors consisting of the positions of nodes in the aggregated graph (which ensure stability but not consistency) and also discarded anchors consisting of the positions of nodes in the previous time slice (because our layout must be computed all at once in order to enable full interactivity). Finally, we adopted a linking mechanism by introducing new edges that link different instances of the same node in different time slices in a chain fashion.

As for computational complexity, the additional inter-time edges do not overly affect the occupation of memory and the speed of computation; in a network with n nodes, there are n inter-time edges for each time slice at maximum, while normal edges can be up to n^2 for each time slice.

4.3 Views

In this section we discuss how adequate visualization techniques for dynamic networks can support their exploration and in particular how time-oriented information can be visually encoded. Focusing on changes over time, we explored three different views, namely, a juxtaposition view, a superimposition view, and a two-and-a-half-dimension view (Figure 4.1). The integration of these three views into a consistent visual environment and interactive transitions between them are discussed in Section 5.2.

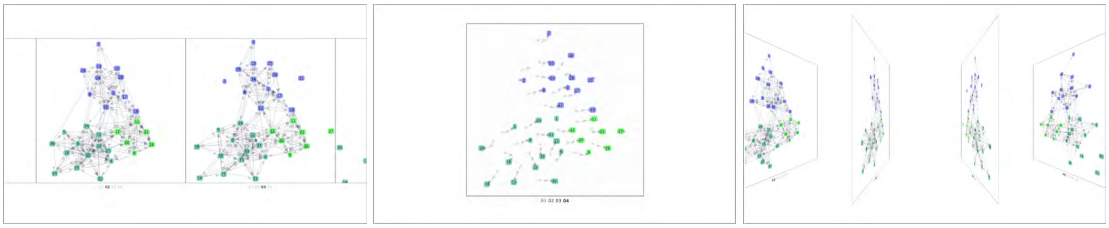


Figure 4.1: The three views

4.3.1 Juxtaposition view

By placing node-link diagrams of different time slices side by side, we obtain a juxtaposition view (Figure 4.2) that we may understand as a mapping of time to space (the horizontal axis, in our case). This view applies the principle of small multiples and allows the reader to compare the time slices and find commonalities and differences. Visual analysis is further facilitated by linking the different frames by interaction features like coordinated zooming & panning and coordinated highlighting, which make exploration and comparison easier. The drawback of juxtaposition is that it takes up more display space: the more time slices we want to visualize, the more display space is needed.

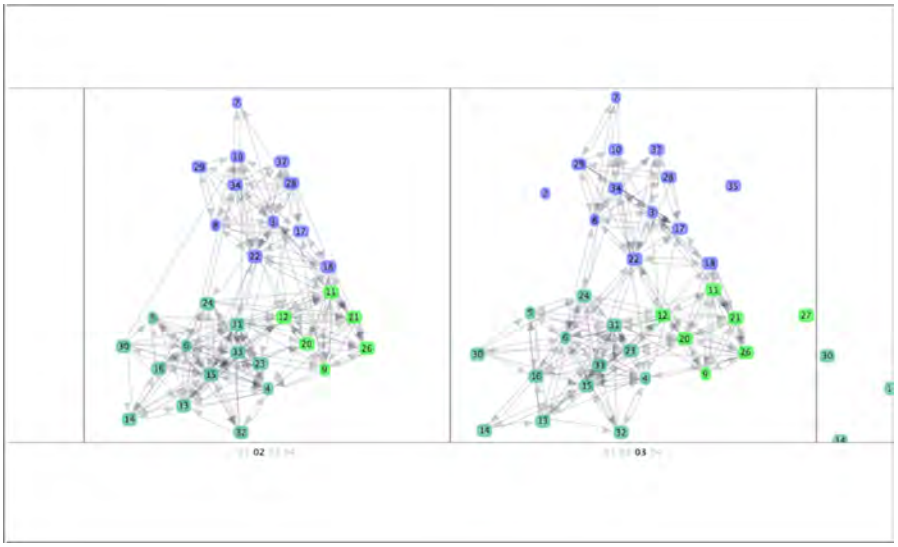


Figure 4.2: The juxtaposition view

4.3.2 Superimposition view

With respect to screen occupancy, we can attain a better performance by superimposing the diagrams (Figure 4.3). In this case, a visual variable must be employed to differentiate between time slices (hence we can refer to superimposition as a mapping of time to visual variable). We used transparency, so that more recent elements are more opaque. Besides the fact that less screen space is used, this has the advantage of reducing the eye movement from one slice to the other compared to juxtaposition and preserves the context. The main disadvantage of this view is the concentration of all edges and nodes within the same diagram with a large number of edge crossings and occlusions that impair readability. In order to reduce visual clutter, we allow users to interactively select the elements (trajectories, nodes or edges) to be shown persistently or by hovering.

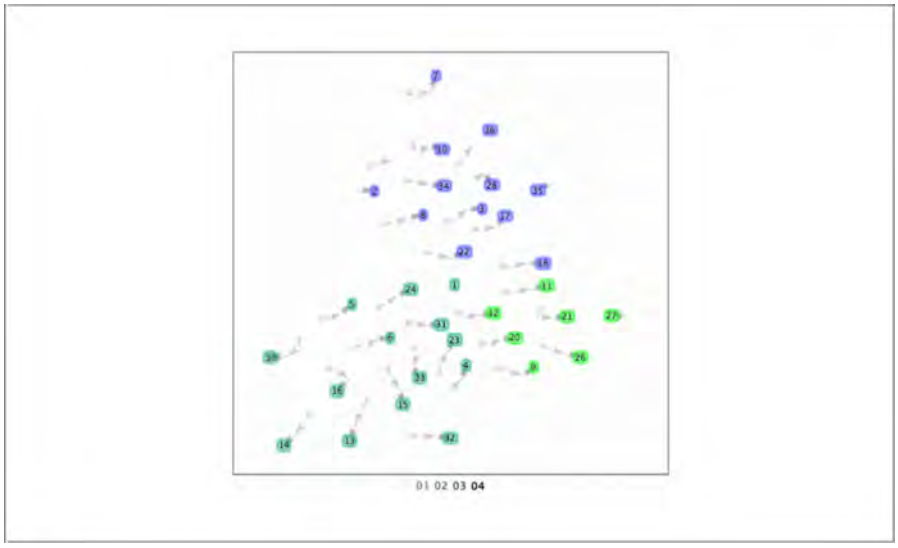


Figure 4.3: The superimposition view

4.3.3 Two-and-a-half-dimensional view

Mapping time to an additional spatial dimension results in a two-and-a-half-dimensional view (Figure 4.4). In such a view, we draw diagrams for each time slice on separate transparent planes, stacked along the horizontal time axis. It combines some of the advantages of the two aforementioned views. Moreover, the added spatial dimension offers us the opportunity to include additional information within this view.

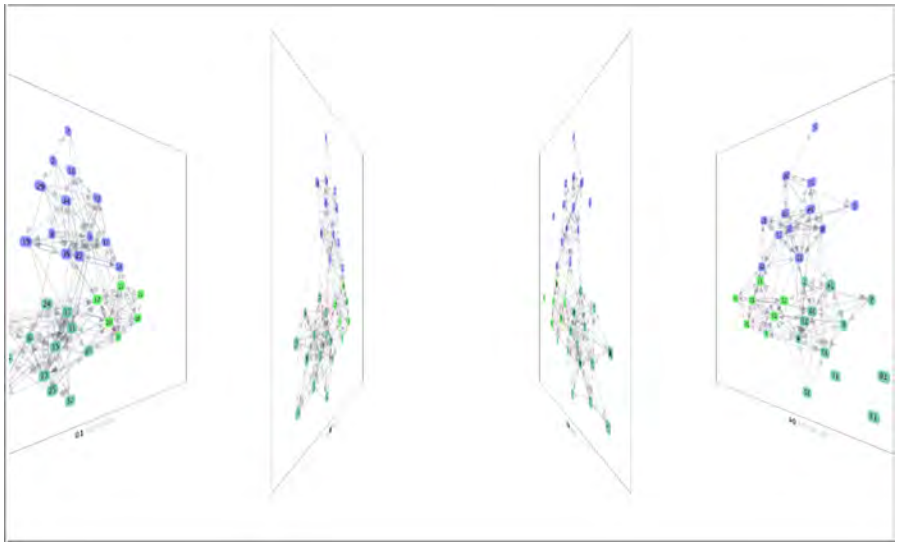


Figure 4.4: The two-and-a-half-dimension view

4.4 Enriching visualization with analysis results

According to our visual analytics approach, the integration of analytical methods with visualization and interaction techniques aims at supporting and enhancing the comprehension of network data. As a first step towards the combination of visual and analytical methods, we considered classic SNA metrics for static (i.e., non-temporal), single-relational networks and integrated their computation into our prototype. In this way, a user can interactively select a certain SNA metric to be computed for a certain type of relation s/he is interested in. The entire temporal multi-relational network is partitioned into as many static single-relational networks as there are time slices and the requested metric is computed for each of them. Then, the resulting values are mapped into retinal variables of the visualization (color, size, etc.) for each time slice. In Figure 4.5, for example, both node sizes and node colors represent clustering coefficients. In general, this integration supports the analysis by enriching the node-link diagram with the values of global and local topological properties.

Figure 4.5: In this diagram, both node sizes and node colors represent clustering coefficients.

4.4.1 Scatter-plot layout

Besides retinal variables such as color, the results of social network analysis algorithms can be also mapped to geometrical variables, e.g. node positions. As also explained in Section 2.6.1.1, the layout of a node-link diagram can be computed on the base of node-centric network measures, resulting in a diagram that is similar to a scatter plot. This type of diagram allows the user to visually analyse pairs of node metrics and correlation between them. Figure 4.6 shows the juxtaposed scatter-plots of two subsequent time frames; the betweenness centrality is mapped to the vertical axis and the eigenvector centrality is mapped to the horizontal axis.

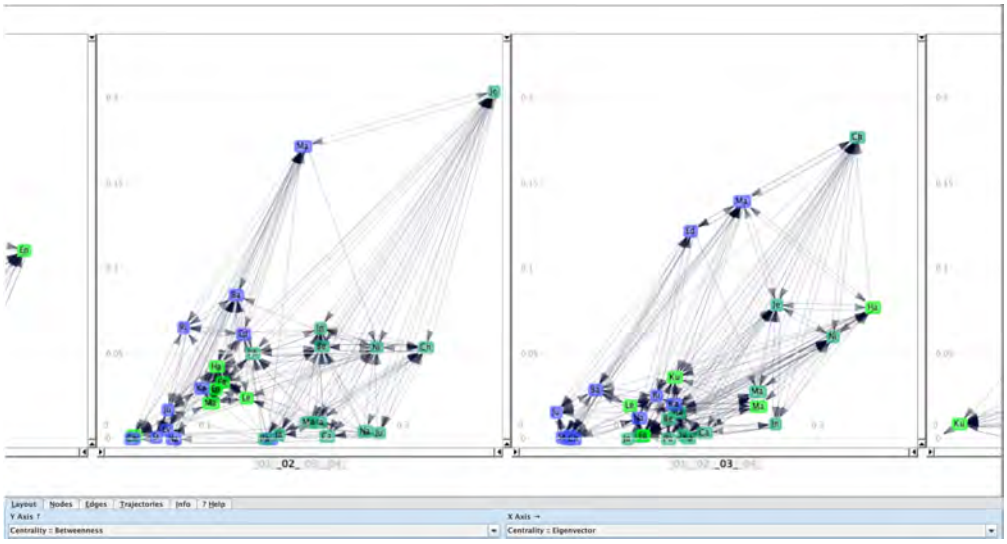


Figure 4.6

4.5 Exploiting change centrality metrics

In this section we discuss how the change centrality and the other metrics we have introduced in Section 3.2 can be exploited visually for identifying changes, comparing subsequent time steps, and detecting temporal trends; in other words, how these metrics can support the visual analysis of dynamic networks.

We will use a research prototype software, ViENA [139], based on a visual analytics paradigm. Visual Analytics can be seen as an integrated approach combining visualization, human factors and data analysis [225]. According to this paradigm, ViENA combines interactive visualizations with a computational kernel, that provides several static and dynamic SNA metrics as well as the change measures we introduce in this paper. As for the visualization, we refer to three views, namely the superimposition, the juxtaposition, and the two-and-a-half-dimensional (2.5D) view, that are all based on node-link diagrams but handle the temporal dimension in different manners. In the juxtaposition view (Figures 4.7, 4.8, and 4.10) the node-link diagrams of different time points are placed side by side, allowing for direct comparison. In the superimposition (Figure 4.11) the diagrams are overlaid using transparency; to avoid clutter, links are shown only on request, while nodes are always visible as well as their trajectories (i.e. lines showing movements of nodes over time). In the 2.5D view (Figure 4.9), the diagrams are drawn upon transparent parallel planes, stacked along the horizontal axis.

The presented change measures can enrich the visualization, supporting the visual reasoning and analysis. In detail, we show how they can be 1. mapped to node size and colour, to facilitate the identification of changes and the comparison of network structures over time; 2. used to interactively filter nodes according to their involvement in changes; 3. taken into account for the computation of an optimized dynamic layout.

4.5.1 Nodes size and colour

A typical way to enrich a node-link diagram by using computed node attributes is mapping them to the sizes and the colours of nodes. Figure 4.7 is a juxtaposition view of node-link diagrams in which nodes are coloured and sized according to their values of change centrality.

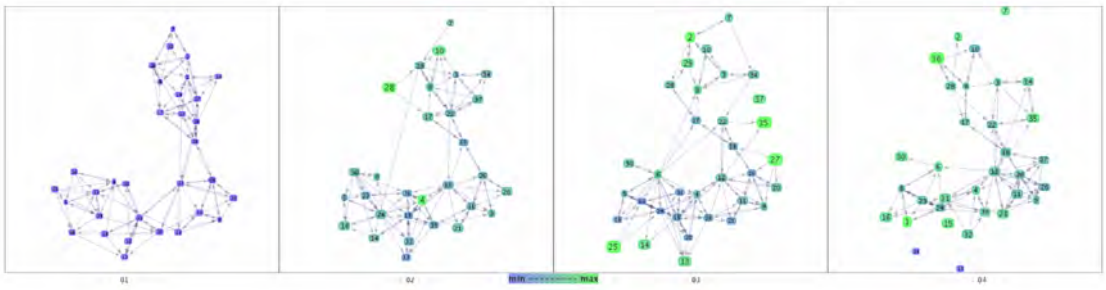


Figure 4.7: Change centrality mapped to node size and colour in a juxtaposition view. Larger green nodes have encountered more changes in their connections from the previous time step. Smaller blue nodes are more stable.

Since the change centrality is computed for each pair of subsequent time steps and is symmetrical, we chose to map its values in the visual features of the second time step of each pair; hence, the visual features of nodes for each time step express what has changed with reference to the previous one, while they are neutral for the first time step of the sequence. In this way nodes that are present in only one time step of a pair are treated differently: new nodes (present only in the second time step of a pair) are highlighted, while dead nodes (present only in the first time step of a pair) are not. For highlighting dead nodes, we can change the perspective and map analytical values to the visual features of the first time step of each pair. In any case, because of our choice of the linear coefficients for the calculation of this metrics, the new/dead nodes are visually prominent. For example Figure 4.7 we easily identify as new the nodes 4 and 28 at t_2, the nodes 25, 27, and 35 at t_3, and the nodes 1 and 36 at t_4. At the same time, having chosen a logarithmic scale for the mapping of analytical measures to visual features, it is also possible to identify smaller differences due to changes in the far neighbours. We can observe that more nodes join the network at t_3 than at other time steps, but at t_3 there is also a group of nodes in the middle-bottom of the diagram that are less affected by change (and appear smaller and blueish). Conversely, at t_4 it seems that all nodes are more affected, in average, by changes.

In Figure 4.8 the same network is visualized with the values of change eccentricity. In this case, the group of nodes that are more distant from changes at t_3 are larger and brighter and, therefore, easier to identify (nodes 23, 24, 15, and 32).

Besides identifying appearing/disappearing nodes and comparing global changes between subsequent time steps, we might be interested to track the amount of change a given node encounters over time. To solve this task, we can use the 2.5D view and map the change centrality to both the colour and the thickness of node trajectories. In Figure 4.9, looking at the colour shading along the trajectory of a selected node, we see how its change centrality varies over time, being maximum at t_3 (green shade).

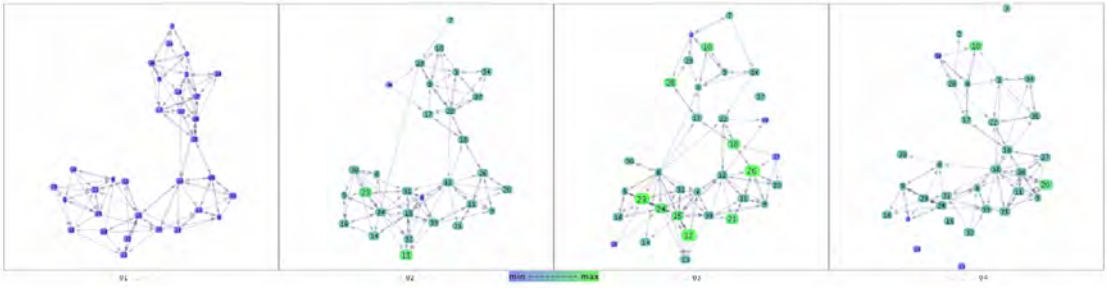


Figure 4.8: Change eccentricity mapped to node size and colour in a juxtaposition view. Larger green nodes are closer to changes, while smaller blue nodes are more distant.

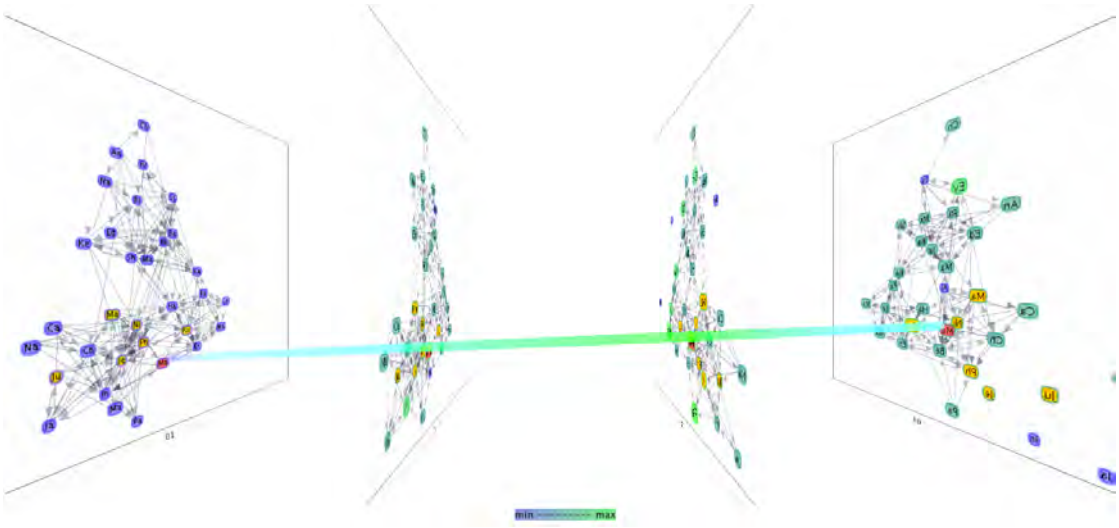


Figure 4.9: Change centrality mapped to thickness and colour of a node trajectory in a two-and-a-half-dimensional view. The selected node has encountered more changes in its connections at time t_3 (trajectory is more green and thicker) than at other time steps.

4.5.2 Node filtering by change eccentricity

Especially when dealing with large and dense networks across several time steps, the problem of visual clutter might occur. Also in the case of smaller networks, we might be interested in focusing only on those parts of the network that undergo major changes, or alternatively minor changes. To support this tasks, change metrics can be used to filter the nodes and only visualize those nodes we want to focus on. In Figure 4.10, for example, nodes are filtered according to change eccentricity, and only the nodes with values equal or greater than 2 are shown. In particular, if we chose for the change eccentricity a threshold equal to the change diameter, only those nodes will be visible which belong to the stable center of the network.

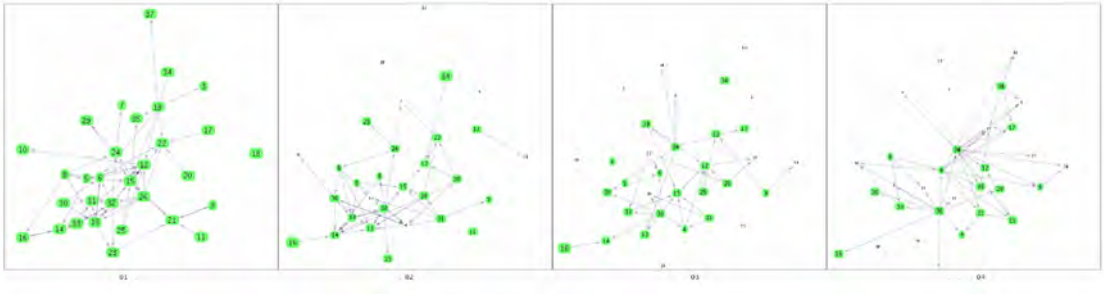


Figure 4.10: A node visibility filter based on change eccentricity: nodes with change eccentricity $E \leq 1$ are filtered out.

4.5.3 Dynamic layout by change centrality

The change centrality metric can be useful to optimize the dynamic layout. As explained in section 4.2, at first we have adopted a linking approach combined with an interactive technique to enable the user to continuously control the stability of the layout. This solution, as well as other methods based on linking and anchoring approaches, has the limitation that the amount of default allowed displacement is the same for all nodes. Even in the more flexible case of dynamic force-directed layout algorithms, the lengths and the elastic forces of the auxiliary springs are the same for all nodes, with the consequence that the amounts of displacement in subsequent time steps do not reflect directly the amounts of relational changes.

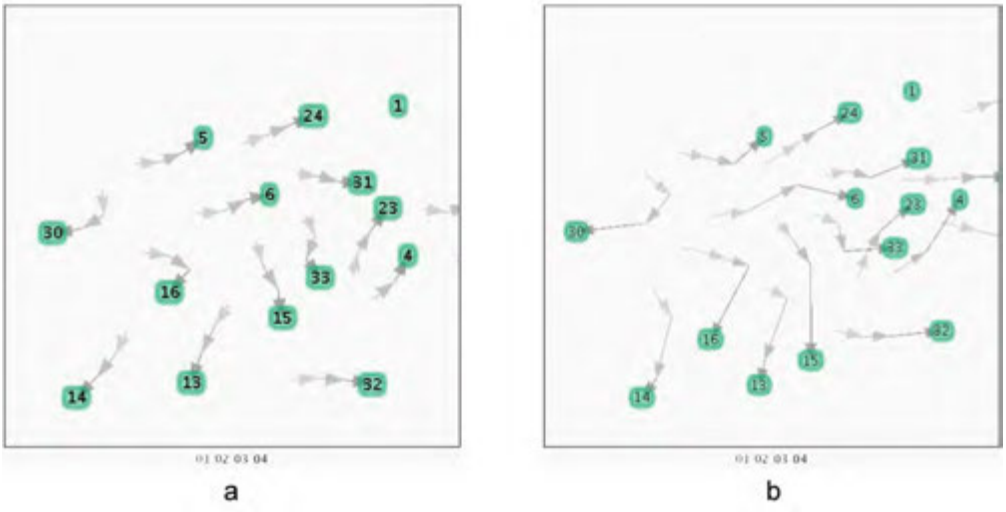


Figure 4.11: Two superimposition views of the same network: a) dynamic layout without change centrality optimization - the extents of node displacements are all the same; b) dynamic layout with change centrality optimization - the extents of node displacements correspond to the amounts of change in their connections.

Conversely, we can use chance centrality values to tune the displacement of nodes between

time steps in our dynamic layout. Since the change centrality is symmetric, we map its values to the lengths and (inversely) to the elastic forces of auxiliary springs connecting the two instances of the same node in subsequent time steps. Analogously, we might map it to the allowed displacement from the initial or the previous position in an anchor-based layout. In both cases, we obtain an optimized dynamic layout, having a flexible linking or anchoring mechanism that actually reflects the relational changes.

All visualizations introduced in the figures above are obtained with this optimized dynamic layout, but its advantages are particularly clear if we consider a superimposition view (Figure 4.11). Looking at the length of the arrows composing node trajectories, we see the displacement of nodes across subsequent time steps. In the dynamic layout computed without the change centrality optimization, they all have the same length. In the one computed with the change centrality optimization, the length are different and reflect the actual relational changes of each node. In this way the layout provides not only a better overview of the network evolution, but also specific details of what nodes encountered larger changes and when.

CHAPTER 5

Interaction[1]

In this chapter, we present the interaction techniques which complement automated analysis and visualization technique in our visual analytics approach. At first, we describe basic interactions for network exploration, as well as smooth animated transitions between views. Then we introduce a novel interaction by which users can control the layout stability. Furthermore, we describe a dual-mode techniques which complements connectivity and continuity highlighting, as well as an interaction for showing node trajectories on demand. Finally, we introduce the vertigo zoom, a novel interactions techniques for two-and-a-half-dimension visualization of dynamic networks.

5.1 Basic interactions

In order to support navigation and exploration of dynamic network, we integrated in our approach basic interaction techniques, as well as more complex and/or novel ones. Basic interactions include (two-dimensional) panning and zooming in the node-link diagram, with both force-directed layout and scatter-plot layout, and independently of the view. The force-directed layout supports also rotation (it would not make sense for the scatter-plot layout) and direct manipulation (i.e., the user can drag and drop nodes, and the layout rearranges). Other interaction techniques are specific to a certain view. In the juxtaposition view only a subset of the time slices are shown at once and, therefore, the user can explore other time slices by panning along the sequence. In the superimposition view, node instances are only visible in the most recent time slice, while in other time slices only node trajectories are visible and edges are not visible; edges and previous node instances are shown on demand, after mouse-over selection. In the 2.5D view, user can seek the most convenient viewpoints by three-dimensional panning, zooming, and rotation. More complex and/or novel interaction techniques are described in the following sections.

[1] Some contents of this chapter have been presented at the International Working Conference on Advanced Visual Interfaces [137] as well as at the International Conference on Knowledge Management and Knowledge Technologies, Special Track on Theory and Applications of Visual Analytics [138]

5.2 Smooth animated transitions between views

Each of the three views introduced in Chapter 4 has advantages and disadvantages and might be efficient for certain data or a particular task. Hence, we exploited them all, and integrated them into our prototype. Moreover, exploring a complex network, the user might happen to switch repeatedly between views. Therefore, we wanted to preserve the user's mental map (in a broader sense than the one we have introduced in Section 2.10) and provide a common context for the interactive exploration of the three views. In order to do so, we designed an interaction metaphor and developed a set of smoothly animated transitions between views (Figure 5.1). According to this metaphor, the planes onto which the diagrams are drawn are rendered as transparent sheets. They are stacked upon each other in the superimposition view, then translated alongside the time axis in juxtaposition view, and finally rotated by 90 degrees around their vertical axes in the two-and-a-half dimensional view.

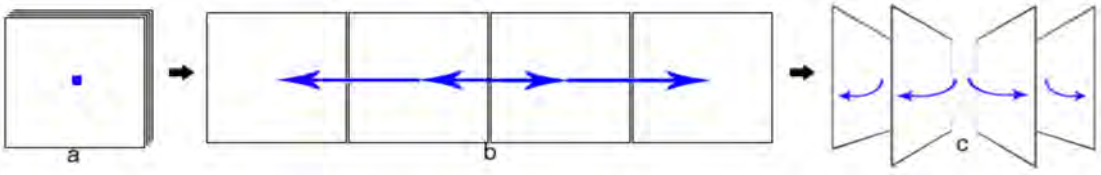


Figure 5.1: Transitions between different views: (a) superimposition, (b) juxtaposition, (c) two-and-a-half-dimensional.

5.3 Interactive control of layout stability

As discussed in Section 2.10, user studies on the problem of mental map preservation have found a broad variability or results about the optimal amount of stability for a dynamic layout. For this reason, we have introduced a novel interaction technique allowing the user to interactively control the layout stability. We exploit the chain of inter-time edges, connecting different instances of nodes across time slices, according to a linking mechanism (see Section 4.2). The length and the force constant of the edges, which compose the chain, can be interactively controlled by the user. The parameters of existing edges that connect different nodes in the same time slices do not vary. A simple slider in the GUI allows the user to select seamlessly between stability and consistency. When the user moves the slider towards consistency, the force constant of the chain decreases and its length increases (Figure 5.2). Hence, the different instances of the same node become almost independent and their positions are computed according only to other nodes and edges in the same time slice. Conversely, when s/he moves the slider towards stability, the force constant of the chain increases and its length decreases. Thus, the different instances of the same node end up at approximately the same position.

In Figure 5.3 we show an example of a social network over two time slices. When the balance of the dynamic layout is set to maximum stability, nodes hold approximately their position and it is easier to locate a certain person and track social relationships over time slices.

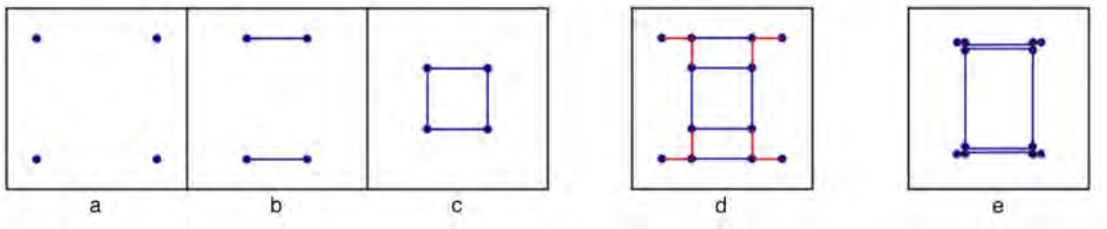


Figure 5.2: Layouts of a network with three time slices: (a, b, c) static layout; (d) dynamic layout, maximum consistency; (e) dynamic layout, maximum stability.

Setting the balance to maximum consistency enables the perception of an overall insight, for example the fact that small groups of five to six persons merge into bigger ones. This integration of graph drawing algorithms with interaction techniques, driven by perceptual principles, aims at supporting the visual analysis of dynamic networks effectively.

5.4 Dual-mode highlighting

The need to combine the analysis of the trajectory of any given node with the analysis of its adjacent nodes led to the design of an additional interaction technique: a dual-mode highlighting. This dual-mode highlighting extends the interaction which highlights adjacent nodes in the network structure (an interation known as *connectivity highlighting* [190]), since it also highlights different instances of a given node across different time slices (an interaction that we name *continuity highlighting*).

The dual mode relates to the way the graph is traversed and adjacent edges and nodes are highlighted on mouse over (see Figure 5.3). In the first mode adjacent edges are traversed first when the user hovers over a node and the nodes that are connected to the selected node in the same time slice are found; then, the trajectories of all these nodes are traversed. Thus, the same set of nodes is highlighted in each time slice, even if they are not adjacent.

The second mode works the other way round: when the user hovers over a node, the graph is traversed with a trajectories-first criterion. First trajectories that lead to other instances of that node in other time slices are found, and then edges. Thus, all node instances that are adjacent to the correspondent instance of the selected node for each time slice are highlighted. By clicking on the selected node, the user can switch from one mode to the other according to her/his needs. The former mode, for example, might be useful to track the temporal evolution of a fixed group of nodes (the neighbors of the hovered one in the hovered time slice).

In the example network (Figure 5.3.a) we see that all persons that are connected to P.K. (the red node) in the first time slice, remain 'near' to her in the bigger group in the second time slice. The latter mode might help to follow the evolution of a single node and to explore the connections it establishes over time. In our example P.K. is directly connected with all of the four persons in her group in the first time slice. In the second time slice, she is directly connected with eight persons, but there are some persons in the bigger group to whom she has no relation (Figure 5.3.b).

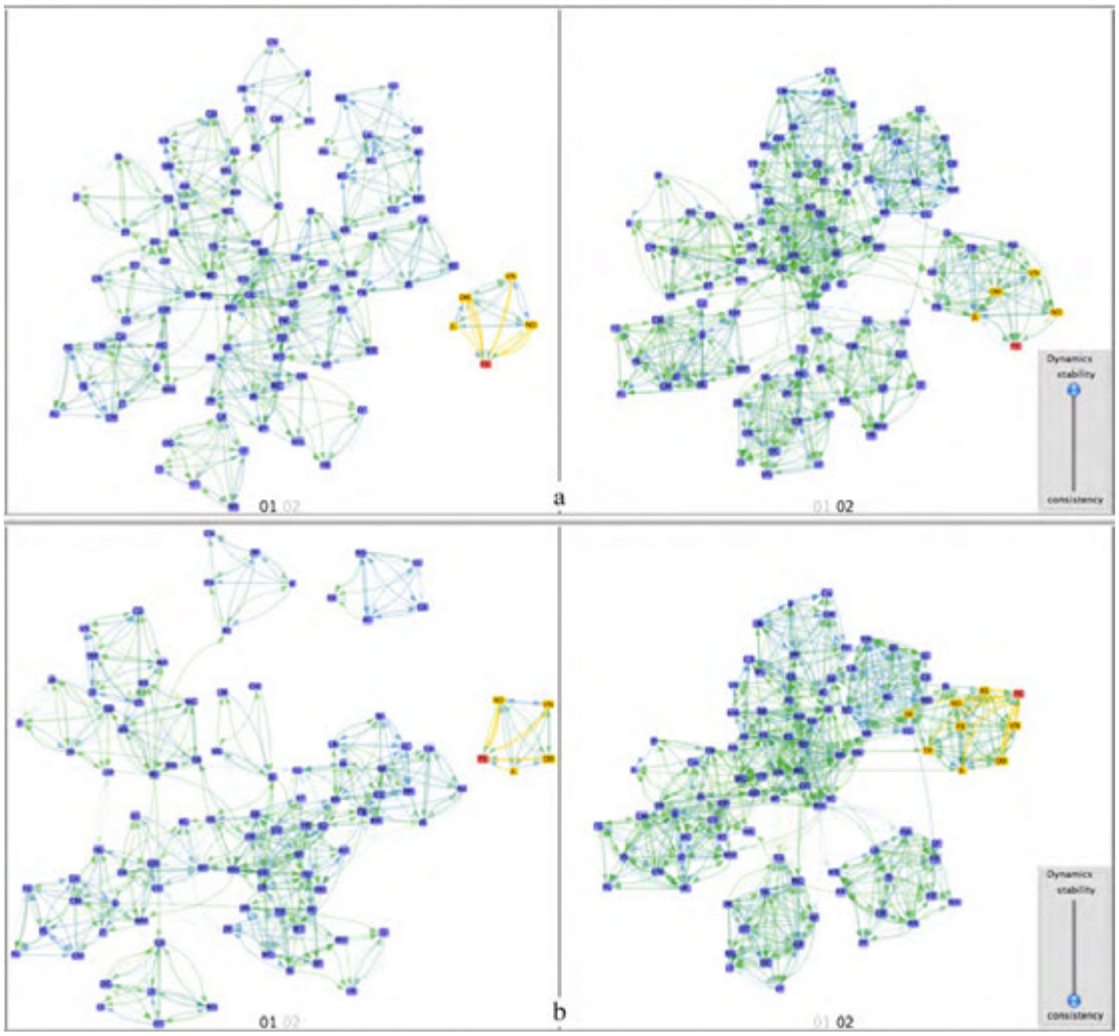


Figure 5.3: Two juxtaposition views of a network, each view showing two time slices. The balance of the dynamic layout is set to maximum stability in (a), enabling the detailed tracking of single nodes since they approximately hold their positions, and to maximum consistency in (b), enabling the perception of an overall insight (small clusters merge into bigger ones). The node highlighting is performed in edges-first mode in (a) and in trajectories-first mode in (b).

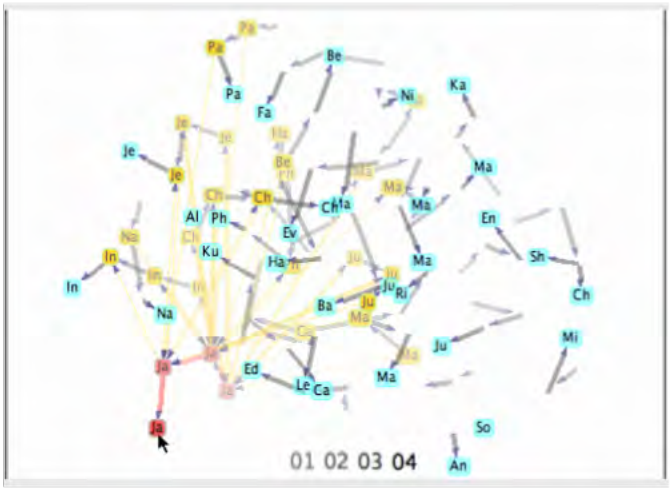


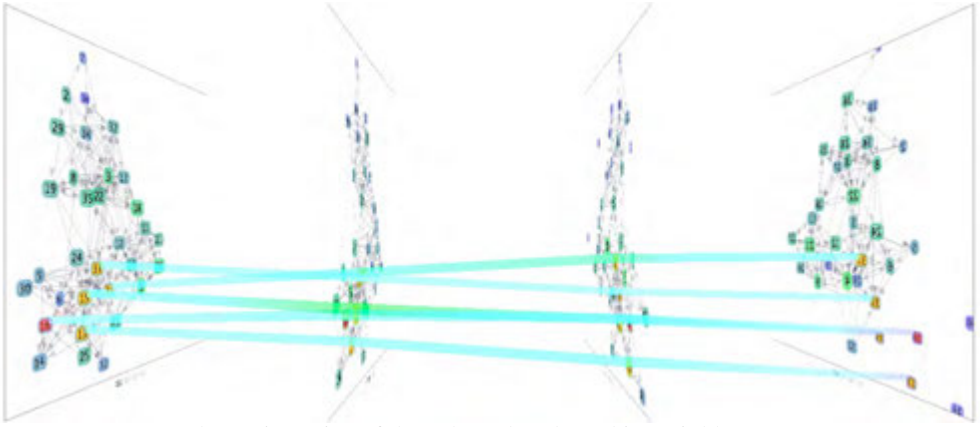
Figure 5.4: The highlighting interaction in the context of the superimposition view.

5.5 Trajectories on demand

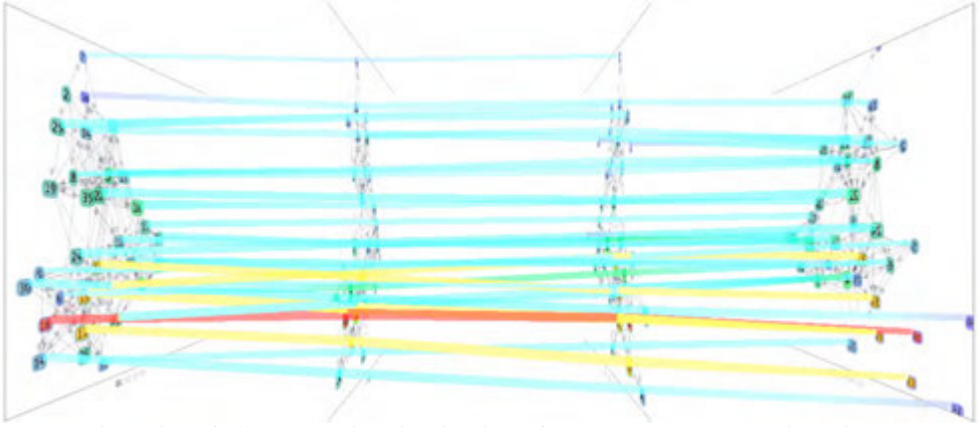
While it is important to analyze and understand the topology of the network, in other words its relational structure (i.e., its edges), nodes are just as important, representing the actors of all the social interaction. Therefore, a good visual analytics approach supporting dynamic network analysis should stress node properties and the changes they undergo over time. In particular, given the importance we give to the layout and the information it conveys, we should provide users with a mechanism to track a node over time and to find its position in each time slice in a more immediate way than by just preserving stability. We accomplished this requirement by showing the node trajectories with an adequate interaction technique for each view.

In the case of the superimposition view, we get an obvious solution from our dynamic chain layout: by simply displaying the special edges connecting the instances of a given node in different time slices, we obtain a polygonal chain that is exactly the trajectory of that node (Figure 5.4). Unfortunately, the superimposition view is already cluttered with nodes and edges of all the time slices, and consequently edge crossing. Moreover, many visual variables that could have been used to differentiate trajectories from nodes are already exploited (color and bending to distinguish different types of relations, transparency to distinguish recent and old time slices). Thus, we allow users to alternatively visualize edges and trajectories or to visualize them only on demand. However, the fact that all nodes lie on the same plane region should provide some sort of a context that facilitates exploration and comparison even if not all edges and trajectories are displayed at the same time. The lack of this proximity context, jointly with the persistence of the crossing of both edges and trajectories, makes it very cumbersome to exploit the trajectories in the juxtaposition view.

(a) Trajectory of a single node shown on demand



(b) Trajectories of the selected node and its neighbours



(c) Trajectories of all nodes, with highlighting of the selected node and its neighbours

Figure 5.5: Node trajectories shown on demand in the 2.5D view.

Conversely, the two-and-a-half-dimensional view is very well suited for the visualization of trajectories, which link node instances in different time slices along the spatial dimension dedicated to time; user can interactively select a node to show its trajectory only (Figure 5.5a), the trajectories of its neighbours (Figure 5.5b), or trajectories of all nodes (Figure 5.5c). Moreover, in the third spatial dimension there are visual variables to which we can map additional information. For instance, we can shade different colors along the trajectory of a given node to show how its values for a certain metric vary over time. In this way, the results of analytics methods are integrated directly into the main visualization of the network, enabling the user to examine its relational and temporal aspects simultaneously without any additional diagram.

5.6 Switching between relational and temporal perspectives

In order to visualize dynamic networks, we have to choose amongst various visual representations (e.g., node-link diagrams, or matrix-based diagrams), and also choose a certain rule for positioning nodes in the space, i.e. a layout, which preserves the user's mental map across evolving network states. An adequate visualization of dynamic network should also provide an effective mapping of the time dimension. In case of discrete-time domain, in particular, we need an effective criterion by which the diagrams referring to each time point are arranged and shown; regarding this issue, we can choose amongst various options, like for example: animation, superimposition, juxtaposition, and two-and-a-half-dimensional (2.5D) view (see Chapter 4).

Aside specific advantages and disadvantages, all these views share the aim of supporting the visual analysis of both the structure and the evolution of the network. During a task-based qualitative user-study, we observed that users alternate and also combine different views, aiming to solve complex synoptic tasks that demand the understanding of both relational and temporal aspects (e.g.: when identifying which employee has joined/left the organization in a certain period, and analysing the organizational consequences). Thus, to better support the visual analysis, the mental map has to be preserved not only across the subsequent states of the network over time, but also across the various views exploited by users to solve their tasks, for example by ensuring smooth animated transitions between views to maintain the exploration context. According to some of the subjects involved in our study, these transitions not only helped them with preserving the mental map, but also served as a live explanation of the visualization metaphors, "saving one from reading many pages of a manual".

Considering in particular the 2.5D view, it has the clear advantage of explicitly mapping time to a specific spatial dimension, that might be used to encode also additional information, like node trajectories visualizing structural changes over time. A node trajectory is visualized as a line connecting different instances of a given node across different time-slices; its slope can show movements from the core to the periphery or vice versa. Moreover, the temporal evolution of selected graph-theoretical measures associated to nodes can be visualized by changing the color and the thickness of trajectories along the time axis. But, the 2.5D view is affected by some problems, like distortion and overlapping. The degrees of distortion and overlapping are correlated to the scene perspective (Figure 5.7), more precisely the camera angle of view or the field of view (FOV). If the FOV is too narrow (Figure 5.6.a), node trajectories drawn along the horizontal axis are visible but we get a foreshortened view of the node-link diagrams with a lot

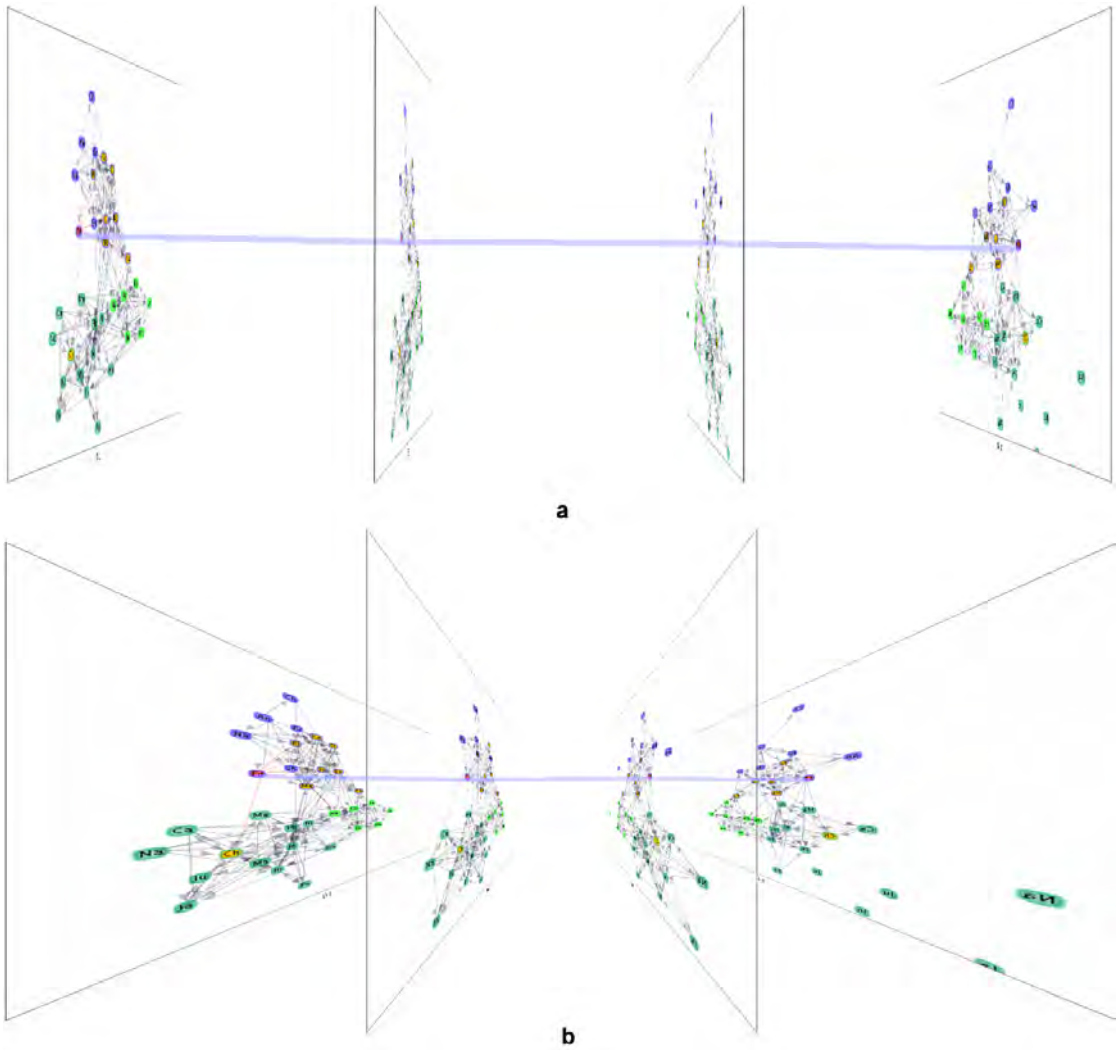


Figure 5.6: Two-and-a-half-dimensional views of a dynamic network with different values of the field of view: a) 30 degrees; b) 120 degrees.

of distortion and compression; if the FOV is too broad (Figure 5.6.b), the node-link diagrams are shown in a better angle but there are more occlusions, more overlapping amongst them, and the added half dimension is less clearly visible. Optimizing the FOV can mean to minimize distortion, or minimize overlapping and occlusion. But, going back to the need of combining temporal and relational perspectives, we also notice that a narrow FOV enables a better analysis of temporal trends (node-link diagrams are parallel to the line of sight and barely visible, while trajectories along the time axis are visually prominent); while a broad FOV eases the inspection of the network topology (diagrams become more orthogonal to the line of sight, and partly overlap trajectories). Thus, we should also find the value of FOV that optimizes the trade-off between these contrasting needs.

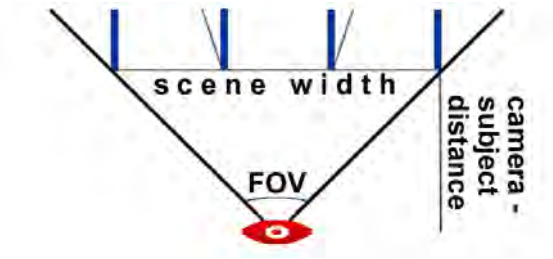


Figure 5.7: The geometric relationship amongst field of view (FOV), scene width, and camera-subject distance.

5.6.1 The Vertigo zoom

Our proposed solution to the above described problems is not based on a method that tries to find the 'best' value for the field of view, but rather on an interaction technique that allows users to interactively change and adapt it to their particular data, tasks, and personal preference. The *vertigo zoom* is a well-know technique in filmmaking, named after Alfred Hitchcock's movie *Vertigo* (1958), also exploited in 3D computer graphics [111]. It is a synchronized combination of a dolly movement and a zoom.

In filmmaking (as well as computer graphics), what we call 'zoom' actually is not performed by changing the FOV, i.e. the focal length of the (virtual) lens, because this would introduce an unwanted distortion of the scene; a zoom-in is realized by a dolly movement: the camera gets closer to the scene, so that the scene is magnified without any perspective distortion. The *vertigo zoom*, conversely, combines a dolly movement and a change of the focal length to produce a change of the perspective without any magnification. The movement of the camera and the change of its FOV have to be synchronized, so that the width of the scene is preserved (Figure 5.7), according to the equation:

$$distance = \frac{width}{2\tan(\frac{1}{2}FOV)} \tag{5.1}$$

While this well-known technique is extensively used in motion pictures as well as in computer games for creating dramatic effects, we propose its adoption for information visualization, as an interaction technique supporting smooth transitions between different perspectives. Considering 2.5D visualizations of dynamic networks, a change of the FOV, controlled by the user by a simple scroll gesture (on either the mouse or the touch-pad), would enable seamless switches between the relational and the temporal perspectives.

The utility of this interaction technique is not limited to visualizations based on force-directed node-link diagrams. Scatter plots using centrality measures as variables have been used to visualize static social and biological networks [114, 244] as well as dynamic networks [293]. We can build a 2.5D visualization by stacking subsequent scatter plots (Figure 5.8a), and then explore it by using our *vertigo zoom* interaction. In case of scatter plots the horizontal and vertical coordinates convey directly the variables they refer to (in the example of Figure 5.8a, eigenvector centrality and clustering coefficient). This is different from force-directed node-link diagrams, where only the relative positions of nodes have a meaning, By using the vertigo zoom

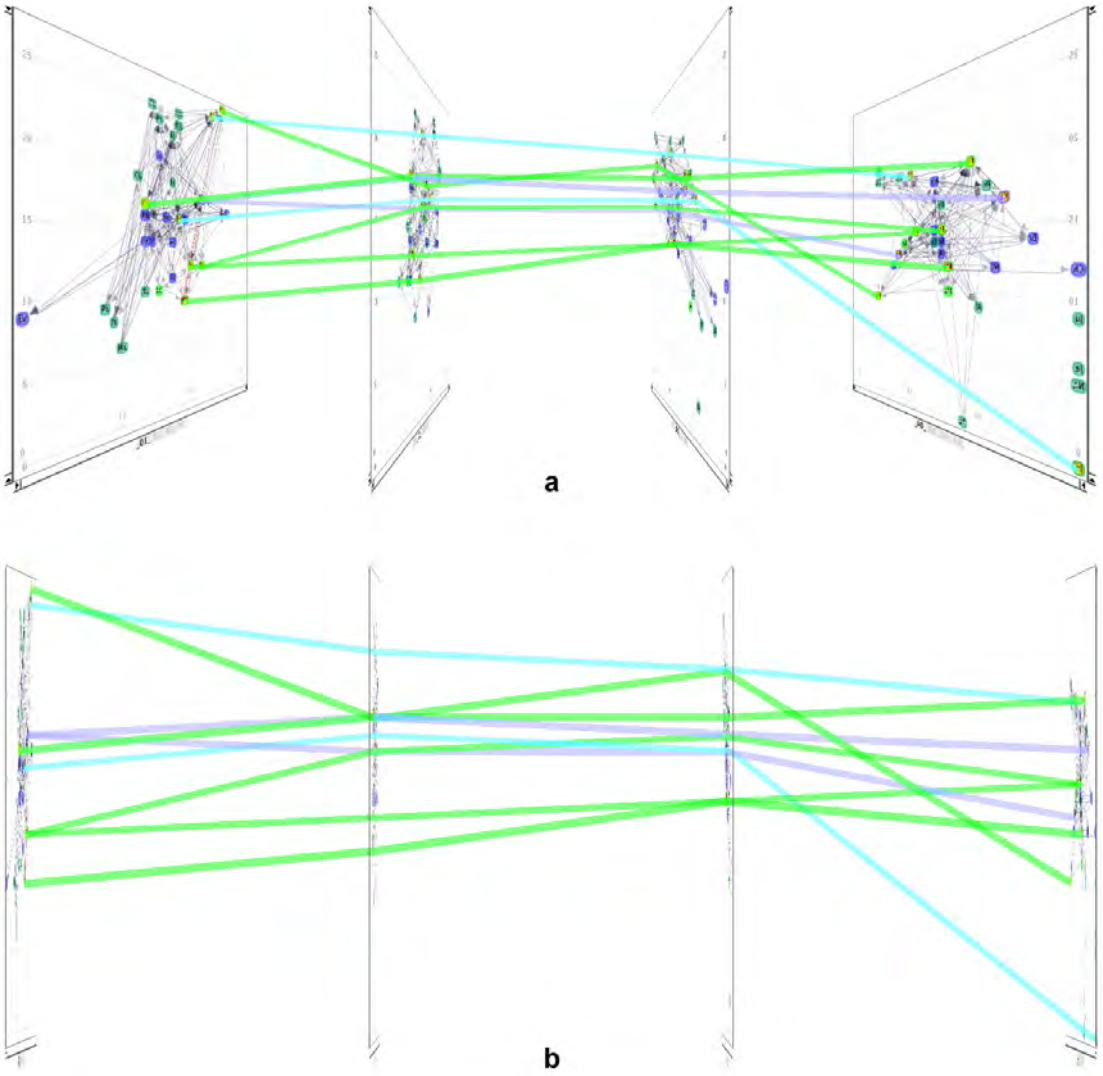


Figure 5.8: a) 2.5D visualization showing scatter plots of network centralities (FOV=60 degrees). b) 2D line chart showing the temporal evolution of network centralities (FOV=5 degrees). The transition between the two views can be controlled interactively by the *vertigo zoom*.

to decrease the field of view and reach the maximum temporal perspective, the 2.5D visualization based on scatter plots is flattened into a line chart, enabling the focused analysis of a single variable for all nodes (FIgure 5.8b). The other variable can be visualized separately (by rotating the view of 90 degrees around the horizontal axis), or jointly (by re-increasing the FOV). In other words, by interactively varying the field of view, the user can focus on a single variable and analyse its temporal evolution without any perspective distortion, or s/he can bear a certain amount of distortion and inspect the temporal evolution as well as the correlation of two variables, and s/he can always smoothly pass from the one end to the other.

A glance at the related work (Chapter 5) shows that although the *vertigo zoom* appears to be interesting and promising, it is only one of several techniques aimed to support the visual analysis of dynamic networks. The switch between the temporal and the relational perspectives can also be attained by using classical interactions, like zoom, pan, and rotation in 3D space, or by switching to a different view through a smooth transition. In order to verify the effectiveness of the *vertigo zoom* in comparison with the other techniques, we plan to perform a comparative evaluation by running a set of controlled experiments. Several aspects which might influence the use and the utility of the *vertigo zoom* should be considered. A factor to be considered is the type of diagrams constituting the 2.5D visualization, either node-link diagrams or scatter plots. Another aspect to take into account is the effect of size and density of the network, as well as the number of time points. Lastly, the initial field of view could also affect users' perception. Steinicke et al. [359] suggest to match the geometric FOV of the visualization with the display FOV, that is the angle of view at which users looks at their screens, and can be calculated on the base of the estimation of the average size of the computer screen as well as the average distance of users' eyes from the screen (they report an average display FOV between 28 and 60 degrees). But it is also worth noting that their suggestion refers to realistic rendering of 3D objects and environments, and its applicability to information visualizations cannot be taken for granted.

CHAPTER 6

Implementation notes

In order to demonstrate and evaluate our approach, described in previous chapters, we developed an inspirational research prototype. For our implementation we used the Java [1] programming language and the *prefuse* [191], a toolkit for interactive information visualization which realizes the information visualization pipeline (Figure 6.1).

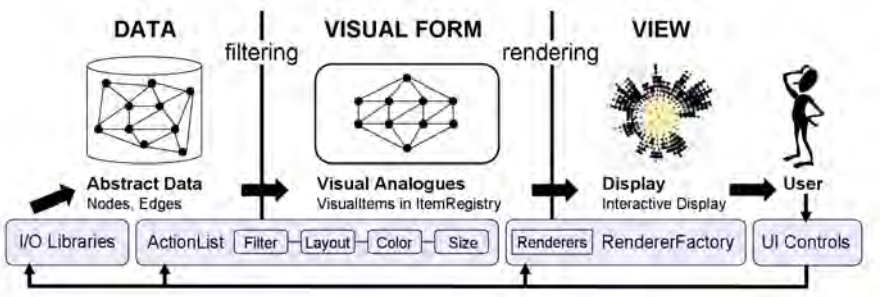


Figure 6.1: The information visualization pipeline as implemented by *prefuse* [191].

We managed graph data by a compact star data model (see Section 2.2.5): an edge list complemented with forward and reverse indexes. We implemented the analytical component in Java, too. We integrated standard SNA measures by using existing algorithms; we described algorithmic details of our newly introduced change centrality metrics in Section 3.2.3.

The three-dimensional scene hosting the three views and the transitions between them has been implemented by using JOGL [2], the Java binding for the OpenGL graphics library. In particular, in the three-dimensional scene we created an OpenGL polygon for each time slice, and mapped the *prefuse* visualization to the polygon two-dimensional texture. Trajectories in the 2.5D view were realized as 3D cylinders by a JOGL renderer extending the *prefuse* edge renderer.

[1] `http://www.java.com`, (accessed on 21/04/2017)
[2] `http://jogamp.org/jogl/www/`, (accessed on 21/04/2017)

Part III

The validation

CHAPTER 7

Usage Scenario[1]

In this chapter, we demonstrate our approach by discussing a real-world usage scenario: we focus on the organizational network of a knowledge-intensive enterprise (referred to as KIE in the following).

The analysed firm focuses on research and public communication in various areas of expertise. We gathered multi-relational network data by surveying the employees 4 times during 14 months. The questions aimed at the analysis of the organizational knowledge communication network and were asked with reference to the last 3-4 months to cover the whole time period between subsequent surveys. Due to generally high response rates, relational data of all relevant and active actors could be collected at each survey. We disregarded probabilistic aspects and multi-modality, by considering only employees as nodes and disregarding skills, tasks and any other organizational data. At first the enterprise was structured into three organizational units, while a restructuring process was altering the initial compartmentalisation at the end of the surveys. The number of the enterprise staff varied between 33 and 34, with a total of 38 persons considering the turnover. To preserve the privacy of the employer and of the employees, we rendered both anonymous.

After having defined the context and the features of data, we conducted preliminary interviews with users to outline their needs and collect a list of requested features. We chose subjects to be interviewed from two groups: business users with a managing function (i.e. non-experts with reference to network analysis), who are our main potential users, and network experts as reference group. We questioned 11 persons by the means of a semi-structured method; then we analyzed the resulting audio recordings, each lasting between one and two hours, and processed them into a list of requirements.

As a result of the user interviews, we elicited a list of tasks, both general (at network level) and specific (at node level), all focused to dynamic aspects. A general task is to understand the structure of the entire network and its evolution over time, for example by identifying large

[1] Main contents of this chapter have been presented at the UK Visual Analytics Consortium International Workshop on Visual Analytics [139]

clusters of employees with tighter relationships and observing how they change over time. Examples of tasks at the node level are the identification of: any occurrence of hires, leaves and resignations; increasing or decreasing levels of involvement; and the presence of key players and their evolution.

7.1 Analysis of network structure

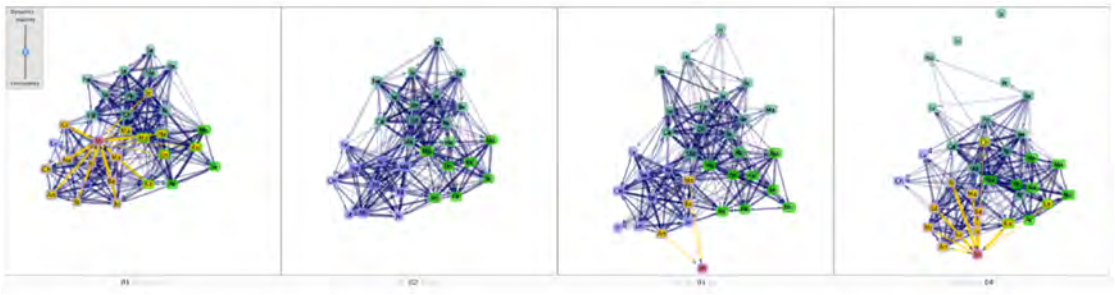


Figure 7.1: Identifying network's structure and its evolution in a juxtaposition view; nodes are coloured according to organizational units; selected node are highlighted in red and its connections in yellow.

The first task consists of identifying the structure of the network and following its evolution. In more detail, we are interested in finding clusters, putting them in relation with organizational units, and observing if and how they vary over time. Several features of our prototype can support this task. The visualization is based on node-link diagrams (which scale properly for networks in our size range). The identification of clusters is ensured by the dynamic layout algorithm, which also is one of the benefits of the integration between visual and computational techniques. We adopted a force-directed layout, ensuring stability by a linking mechanism. The user can interactively control the stability/consistency trade-off: a simple GUI slider allows her/him to select stability or consistency and to pass from one to the other through stepwise transitions. Looking at the KIE dataset, we have found that an intermediate value lets the positions of nodes vary over time according to their connections, while it preserves the positions of clusters (corresponding to organizational units, which remained unchanged until time point 3, when they faced major organizational changes). These clusters and their evolution can be seen in Figure 7.1 and Figure 7.2. The former shows a juxtaposition view, which applies the principle of small multiples and allows the reader to compare the time-slices and find commonalities and differences. Coordinated zooming & panning and coordinated highlighting further facilitate comparison. The latter (Figure 7.2) shows a superimposition view, obtained by superimposing the node-link diagrams. In this case, a visual variable must be employed to differentiate between time-slices: we used transparency, so that more recent elements are more opaque. This view has also the advantage, with reference to juxtaposition, of reducing the eye movement from one slice to the other during comparison, and preserves the context. In both views we see that the dark-green cluster at the top (which also faced the major organizational changes) moves away from other nodes.

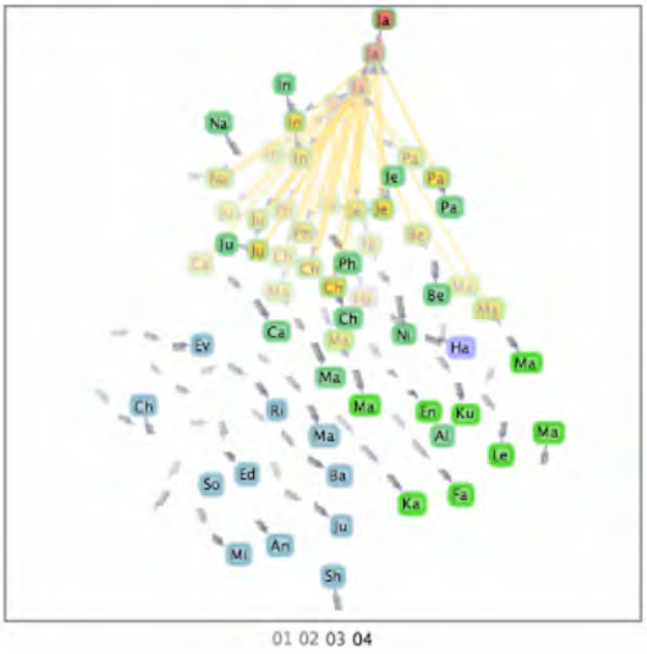


Figure 7.2: Node trajectories in a superimposition view; links (yellow) are only shown for the selected node (red).

7.2 Hires, leaves, and resignations

Passing from overall tasks at the network level to more specific tasks at the node level, the first thing we are interested in is identifying any occurrence of hires, leaves, and registrations. The combination of our dynamic layout algorithm with two interaction techniques supports us with this task. One interaction is the simple highlighting of adjacent nodes. The other consists of showing the node trajectory as a polygonal chain, which connects instances of a given node in subsequent time slices. In the juxtaposition view (Figure 7.1) the node corresponding to Sharon (Sh) is highlighted in red: we observe a leave at time point 2, and a gradual reintroduction at 3 and 4, when she reestablished most of the relations she had. In the superimposition view (Figure 7.2), we allow users to alternatively visualize edges and trajectories, or to visualize them only on demand, to reduce the visual clutter. Trajectories of all nodes are visualized as gray directed polygonal chains. The mouse-hovered node Jack (Ja) is highlighted in red, while its neighbors are highlighted in yellow. The most recent instances are opaque, previous ones are increasingly transparent. Jack has been loosing connections, so he moves from the center to the periphery. Also Nadine (Na) and Ines (In), in the upper left, move from to center to the periphery (and in the case of Nadine this corresponds to a resignation). Sharon (Sh), in the bottom, moves the other way round, since she is coming back to work after a leave.

7.3 The trend of individual performance

Another node-level task is to monitor the trend of individual performance, in the sense of social involvement in the communication/collaboration network. To illustrate how our prototype supports it, we introduce two additional features: a two-and-a-half-dimensional (2.5D) view and the integration with SNA computational methods. We considered metrics for static (i.e. non-temporal) and single-relational networks, like for examples node centralities. In this way the user can interactively select a certain metric to be computed for a certain type of relation; the entire temporal multi-relational network is partitioned into as many static single-relational networks as time-slices are and the requested metric is computed for each of them. Then the resulting values are encoded to visual variables within the visualization (color, size, shape) for each time-slice. We obtained a 2.5D view (Figure 7.3) by mapping time to an additional spatial dimension. In such a view, we draw diagrams for each time-slice on separate transparent planes, stacked along the horizontal time axis. It combines some of the advantages of the two aforementioned views. 3D zooming, rotating and panning controls allow the user to set the best viewpoint. In this view, trajectories run along the spatial dimension dedicated to time. We shade different colors along the trajectory of a given node to show how its values for a certain metric vary over time. In this way, the results of analytics methods are integrated directly into the main visualization of the network, enabling the user to examine its relational and temporal aspects simultaneously without any additional diagram. In the KIE organizational network, we observe that Paul (Pa) has constantly middle importance (bluish trajectory) while Jeff (Je), holding a managing function until the time point 3, had high importance at the beginning but has been loosing it (trajectory shading from green to azure).

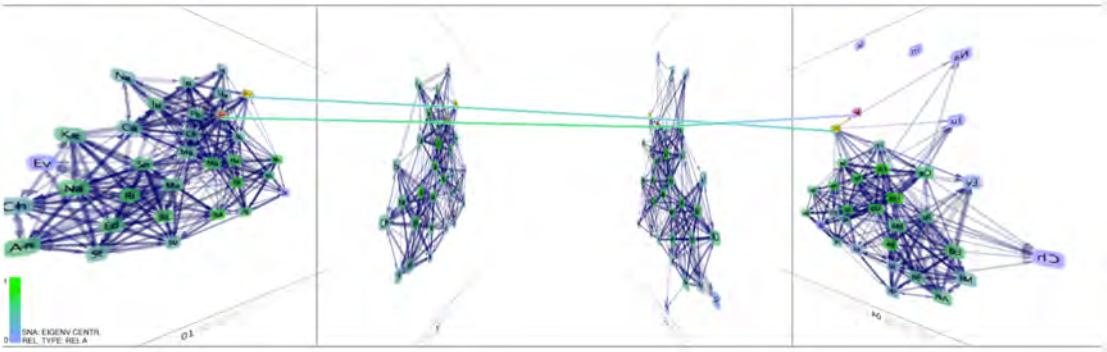


Figure 7.3: Analyzing the trend of individual performances in the two-and-a-half-dimensional view with trajectories.

7.4 Presence of key players and their evolution

The last task we discuss consists in identifying key players and tracking their evolution. Like the previous task, this one is supported by the combination of the SNA computation, the two-and-a-half-dimensional view and the interaction showing trajectories. In Figure 7.4, a zoomed

and rotated 2.5D view, we observe that the betweenness associated to the top manager Hans decreased during the period 1 – 3 (green-azure trajectory), while the betweenness associated to the administrative assistants Christian (Ch) and Judith (Ju) is the highest at the time point 3 (green borders). We might correlate this observation with the managing decision to enforce inter-unit collaboration, taken at time point 1. It gave the expected result, reducing the amount of coordination efforts required to the top manager, but also resulted in a higher charge of coordination to be performed by the administrative assistants.

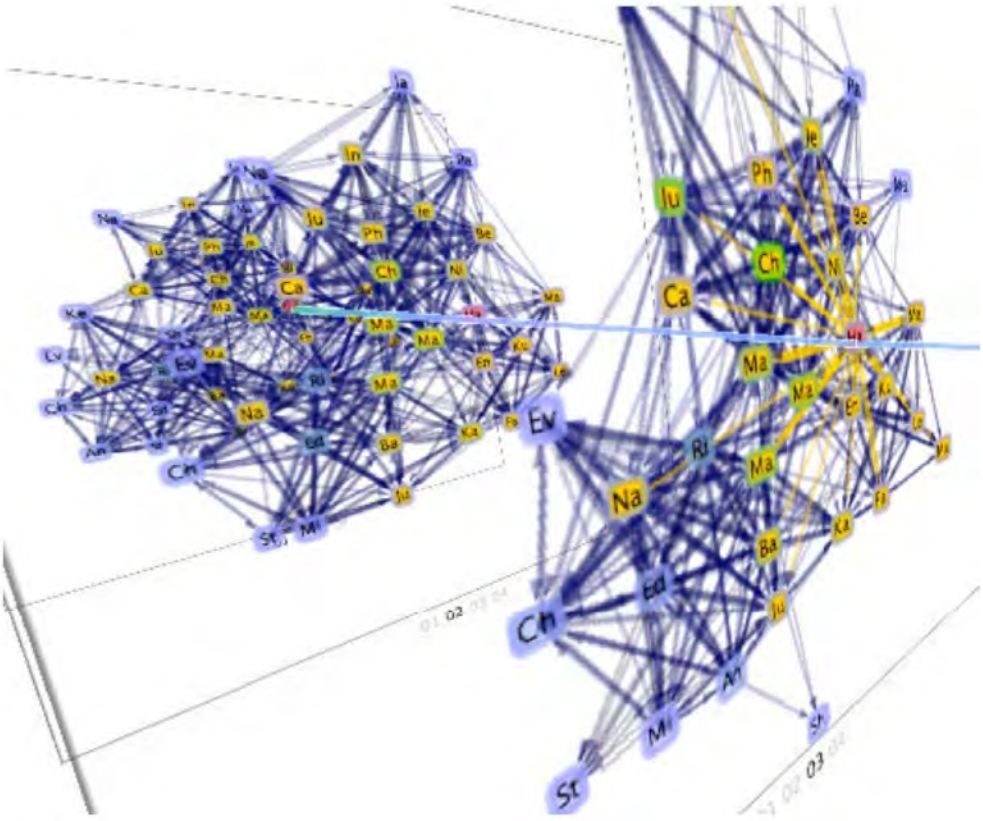


Figure 7.4: The evolution of a key player (red).

CHAPTER 8

Expert Review

According to our methodology (see Section 1.4, in order to avoid threats to validity of our approach, we performed expert reviews of design mock-ups and software prototypes at different stages along the iterative development, involving in the problem domain (social network analysis, organizational network analysis, network science) as well as visualization and human-computer interaction experts.

Domain experts were involved since the early stages of our work, which was performed in collaboration with a multidisciplinary team in the context of a applied research project. As briefly mentioned in Chapter 7, we collected requirements by interviewing domain experts. Besides characterizing the domain problems, we also identified indexes from social network analysis which were reported as relevant and useful by interviewed experts (for example, they referred to the so-called ABCD quartet, i.e., eigenvector, betweenness, closeness, and degree centrality).

We collected additional feedback from domain experts by informal interviews and discussions about mock-ups and early prototypes at specific venues, such as the INSNA Social Network Conference (Sunbelt), the International Conference on Network Science (NetSci), and the Conference on Applications of Social Network Analysis (ASNA). Informal feedback from experts in visualization and human-computer interaction was collected during interviews and discussion sessions at the IEEE Visualization conference (VIS), the EuroGraphics/VGTC Conference on Visualization (EuroVis) as well as during lab visits of international scientists to the Laura Bassi Center for Visual Analytics Science and Technology (CVAST) at TU Wien.

These informal expert reviews included paper mock-ups, videos, software demonstration or direct prototypes hands-on (depending on the development stage). We took down notes, recorded audio and, when feasible, captured screen-casts, for off-line analysis. The outcome of these activity was a collection of suggestions for improvement, comprehending usability issues as well as specific analysis and visualization aspects. These suggestions were taken into consideration for subsequent design and development phases.

For example, as for usability issues, in an early version of the prototype the user was allowed to freely scroll trough time slices in the juxtaposition view, and seeking a specific time slice could be difficult; to improve this interaction, we complemented each time slice with a small list of the

entire sequence, and introduced a click-to-seek control for panning automatically and directly to the selected time slice.

Another minor usability issue emerged from the two-dimensional rotation of the visualization (present in all views). In order to improve the overall perception of the network structure by the force-directed layout, the rotation is not performed around the center of the visualization but rather around the center of mass of the node-link diagram, since the position of the centroid conveys structural information. However, this interaction resulted to be potentially hard to understand to the user; consequently, we made this mechanism cleared by highlighting the center of rotation as soon as the user initiates the interaction.

Several experts identified hidden states affecting the interaction modes that were not explicitly shown to the user like, for example, the two highlighting modes described in Section 5.4. In subsequent development stages, we made sure to externalize all hidden machine states by visual or textual hints, to give the user full control of the interactions.

Minor usability issues emerged about the understandability of text labels on legends, controls, and other elements of the graphical user interface and were easily fixed. Other suggestions went beyond simple usability issues, and encouraged us to improve visual analytics features or introduce novel ones. For example, we found that the different visual encodings and the abrupt switching between views could be confusing; hence, we designed and developed smooth animated transitions. We also found that static social network metrics, even if easily understood and highly appreciated by domain experts, were not sufficient to analysis dynamics and, therefore, we introduced change centrality metrics.

Finally, we got recomendations to redesign the colors used for our dual-mode highlighting; this remarks, in combination with the afore-mentioned need to externalise all inner program states, lead us to the design of a unified, one-mode highlighting interaction (see Chapter 10).

CHAPTER 9

Qualitative User Study[1]

The qualitative evaluation was conducted to assess the prototype's usability as well as the comprehensibility of the different visualization and interaction techniques, and to cover users' exploration process [356]. The subject pool consisted of nine users who work in the field of social network research as pre- and post-docs, mainly as computer scientists or as graph theorists plus another computer scientist from the visual analytics field. None of the participants had prior knowledge about the prototype. In the first phase of this study, the prototype was presented to the participants in an interactive session together with an instructor. Participants were encouraged to explore the functions of the prototype, ask questions, give feedback about the prototype's usability and express their ideas and suggestions for improvement. In the second phase, they had to solve seven tasks, which were derived from the taxonomy by Ahn et al. [9] and were selected on the basis of our experiences with previous mock-up studies. These tasks (see Table 9.1) included lower-level activities like the identification and comparison of the relations of a single node at two time points as well as higher-level activities [266] like the description of structural group changes over time. In the first task participants were allowed to use all prototype functions and views freely. In all other tasks they were compelled to work with a preselected initial view. In the third and last phase, participants were asked to summarize their impressions and give additional feedback. The material consisted of the same real-world dataset described in the usage scenario (see Chapter 7). The verbal comments and a screen cast were recorded during all phases, which lasted about 1.5 hours in average. Notes were taken by an observer during these sessions. These notes were jointly analyzed by a team of three usability experts who were also part of the testing-team. The notes were segmented in single observations, which were categorized and counted as presented in the following section. First we want to present an overview about users' feedback and observed problems during the introduction phase, later we will describe our main insights that derived from task analysis.

[1] Main contents of this chapter have been published as a book chapter [354]

Table 2 List of tasks for the prototype evaluation. The tasks were named according to the scheme proposed by Ahn et al. [1]

Task name	Task description	Predefined settings
T1: Network—Growth	Has the total density of the network increased or decreased from t1 to t2?	Open
T2: Group—Stability	Which groups/clusters do you detect? How do they change?	JX
T3: Node/Link—Growth	Had Leonard (Le) more relations at t1 or at t2?	JX + SNA
T4: Node/Link—Single Occurency	Please identify the outdegree of actor Hans at t3	JX + SNA + tooltip
T5: Node/Link—Growth	Please identify the change (increase/decrease) of Ines' eigenvector centrality (from t1 to t4)	2.5D + SNA + single trajectory
T6: Node/Link—Birth Death	Who has joined / who has left the network (causing relational consequences)?	2.5D + SNA + all trajectories
T7: Node/Link—Peak/Valley	Are there significant shifts of single actors from cores to peripheries or vice versa?	SI

Table 9.1: List of tasks for the prototype evaluation. The tasks were named according to the taxonomy by Ahn et al. [1].

which lasted about 1.5 up to 2 h in total. Notes were taken by an observer during these sessions. These notes were jointly analyzed by a team of three usability experts who were also part of the testing-team. The notes were segmented in single observations, which were categorized and counted as presented in the following section. First we want to present an overview about users' feedback and observed problems during the introduction phase, later we will describe our main insights that derived from task analysis.

9.1 Usability findings

The evaluation results are structured as a matrix, with the main visual, computational and interactive features of the prototype as rows and columns (see Figure 9.1). The feedback was segmented into 255 distinct observations, which were categorized as problems (118), positive feedback (45) and ideas for improvement (109). It has to be noted that similar observations were counted multiple times, so that we could identify 155 unique observations in total. To give an overview we will focus mainly on areas in which many observations have been made, leaving bugs and too specific implementation issues aside. In all views, many participants stated that the transitions are too slow, although the idea to maintain the mental map by transitions yielded consistently positive feedback. In the case of the highlighting feature, participants recommended additional interactions to make comparisons easier by highlighting more than one node at a time. SNA measures of nodes like centralities were always double coded by size and by color in the prototype. Many users expressed the wish to have more freedom in selecting how these measures are displayed, and they preferred to use their favorite color palette. This applies for the relations too, where different types of relations should be visualized by different visual features like color or line style. Concerning the main views, the juxtaposition view was rated as the most comprehensible by users comments and we detected the least problems in this view. In the

3.1 Evaluation Results

The evaluation results are structured as a matrix, with the main visual, computational and interactive features of the prototype as rows and columns (see Fig. 3).

The feedback was segmented into 255 distinct observations, which were categorized as problems (118), positive feedback (45) and ideas for improvement (109).

It has to be noted that similar observations were counted multiple times, so that we could identify 155 unique observations in total.

To give an overview we will focus mainly on areas in which many observations have been made, leaving bugs and too specific implementation issues aside. In all views, many participants stated that the transitions are too slow, although the idea to maintain the mental map by transitions yielded consistently positive feedback. In the

superimposition view, participants mainly struggled following transitions and dealing with visual information overload. For the 2.5D view, users reported navigational problems as being too slow, not responsive enough and they missed an immediate feedback of the prototype when they zoom, pan or rotate. Most users suffered from perspective distortion when comparing node sizes and they mentioned legibility issues since the node labels and tool tips were distorted to a high extent in the two middle layers. Many users also mentioned a visual information overload as soon as many of the (too boldly styled) trajectories were displayed in 2.5D view. Regarding the GUI, most users reported serious problems in understanding some of the labels, especially those of the dynamic views. Seven of eleven users reported (all of them no native English speakers) comprehension problems for the naming of the views (mainly "Superimposition" and "Juxtaposition"), two proposed to use icons instead of names. Only one person made sense of all the chosen view names. When dealing with user feedback seriously, this could be also seen as a hint that the untested transfer of technical terms (here: from the InfoVis community) via a prototype to an audience without that specific domain knowledge could have a negative impact on usability. Aside this summary of problem oriented feedback, all users focused on implementation and in general, nearly all participants expressed a remarkably positive assessment of the prototype in their overall summary.

Figure 9.1: Frequency of observations which feature problems (red, left hand side of each column), positive feedback (green, center of each column) and ideas for improvement (blue, right hand side of each column), which could be identified for all views, for single views (superimposition, juxtaposition, and 2.5D view) and for the GUI itself

9.2 Task Completion Analysis

We used two indicators to assess the effectiveness of our prototype's features in supporting users to solve assigned tasks: correctness and confidence. The correctness is defined as the conformity of user's answer to the answer we obtained by numerical methods and, for certain tasks, also by our previous knowledge of the real-world network at hand. The confidence differentiates between affirmative certain answers, and uncertain answers expressed in vague forms (e.g. "I would say", "I guess", "I am not sure"). We disregarded the task completion time, because we were more interested in the reasoning process, and asked users to think aloud and explain

how they conceived the answer rather than to give the fastest answer. The overall correctness of the answers was 89% (see Table **??**). Half of the incorrect answers were given to task 1, but they might be ascribed to the task openness (without any default settings of the view and other parameters) and to the fact that it was intrinsically hard to solve, demanding the detection of a very slight variation of the network density. As for the confidence, 82% of the correct answers overall were also certain answers, with the highest value for task 3 and task 5, and the lowest also in this case for task 1. As a general conclusion, we observed that most of the users were able to provide correct, complete and confident answers for task 2 to task 7 (see Table **??**), mostly by using the combination of visual, analytical and interactive options we had set, with noticeable exceptions and unexpected behavior that we discussed (see section about multiple problem solving strategies). For some users, their performances on given tasks also affected their initial preference about a given view, for example some users initially were skeptical about the 2.5D and the superimposition views, but changed their mind after they realized they had been able to solve task 6 and task 7 by using them.

9.3 Multiple Problem Solving Strategies

Concerning the collected data of the final prototype tests, we analyzed the thinkaloud audio recordings and the prototype interaction screencasts, using a categorization scheme during observation to extract information how their interaction is related to their insights and categorizing the different solution strategies for every task. This is comparable to the work of Mayr et al. [270], who used a similar procedure with insights by analyzing open tasks. We found interesting empirical results besides correctness and confidence of users' answers: we noticed multiple problem solving strategies spanning both tasks and users, pointing out relevant differences in either the alternative or the combined use of several prototype features.

The first empirical result refers to the integration of visual and analytical methods and their balance. When addressing task 1, that was the toughest to accomplish and registered the lowest correctness, most users looked for or asked for an analytical method directly providing the numeric answer (that was actually missing, since the given SNA component computes only node-level measures so far, and does not provide any network-level measure such as density). Also for task 6, one user said that a binary table would have helped her in tracking nodes' presence more than any visualization. Conversely, we noticed an opposite and unexpected behavior for task 3 and task 4. Task 3 required to compare the degree of a given node over time, and these values are mapped to the color and size of nodes, by default settings; task 4 required to find the out-degree of a given node at a given time, and this value pops out as a numerical tooltip on mouse over, by default settings. By analyzing these tasks, we observed that some users disregarded the analytical hints and preferred to find the (out-)degree just by counting the adjacent nodes, with the help of the highlighting interaction. This observation would lead us to infer that users prefer to visually solve those tasks they think they can manage, and to have recourse to analytical methods for harder tasks. It is worth noting that for task 4 users who counted were as confident as users who looked at the numeric tooltip, but the former were less often correct than the latter. The analysis of task 5 showed us another interesting user behavior: after finding the sign of the variation of the eigenvector centrality by looking at the provided visual

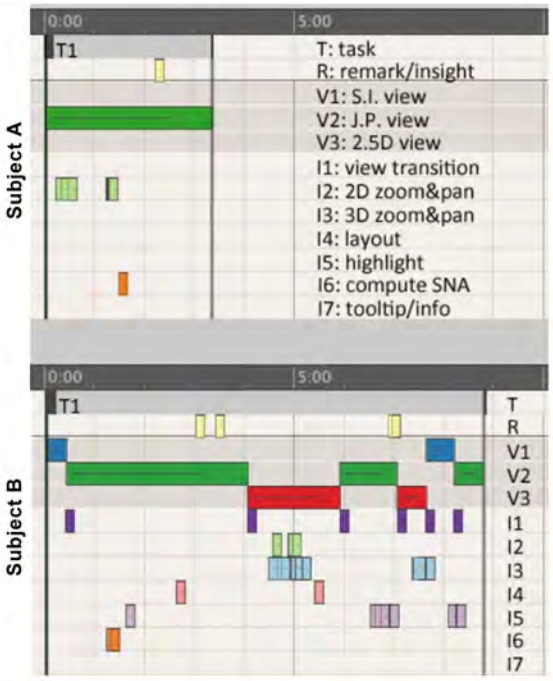


Figure 9.2: Interaction graph for task 1 of the prototype test, by two subjects. While the first two rows show task duration (gray, row *T*) and occurred insights (light yellow, row *R*), rows 3–5 show the usage of the main views (superimposition = blue = row *V1*, juxtaposition = green = row *V2*, 2.5D view = red = row *V3*), whereas the remaining rows show the usage of different interaction and exploration techniques

mapping of the analytical value (color and size of the trajectory), some users rotated the 2.5D view or switched to the juxtaposition view in order to verify whether the network topology was compliant and confirmed the answer they gave.

Another interesting result, which we found from the analysis of the task completion procedures, concerns the recourse to different views to solve the same task. For task 6, for example, we provided a predefined setting with the 2.5D view and all trajectories activated. Most users solved the task looking at the interruptions of trajectories (an interruption on the left side indicates a node who has joined the network, an interruption on the right side indicates a node that has left the network, and an interruption in the middle of two trajectory segments indicates a short leave). In answering the second part of the task, about the relational consequences of these changes, some users switched forth and back to other views, while others kept on using the predefined view and 3D navigation. For task 7, some users switched from the predefined superimposition view to the 2.5D view, where they looked at the slopes of trajectories to investigate movements of nodes.

In Figure 9.2 we can compare the exploration strategies of two subjects dealing with task 1. This figure shows a temporal view of the interaction logs (i.e. which views and which interactions were used or performed and when), as wll as task lengths and insight occurrences.

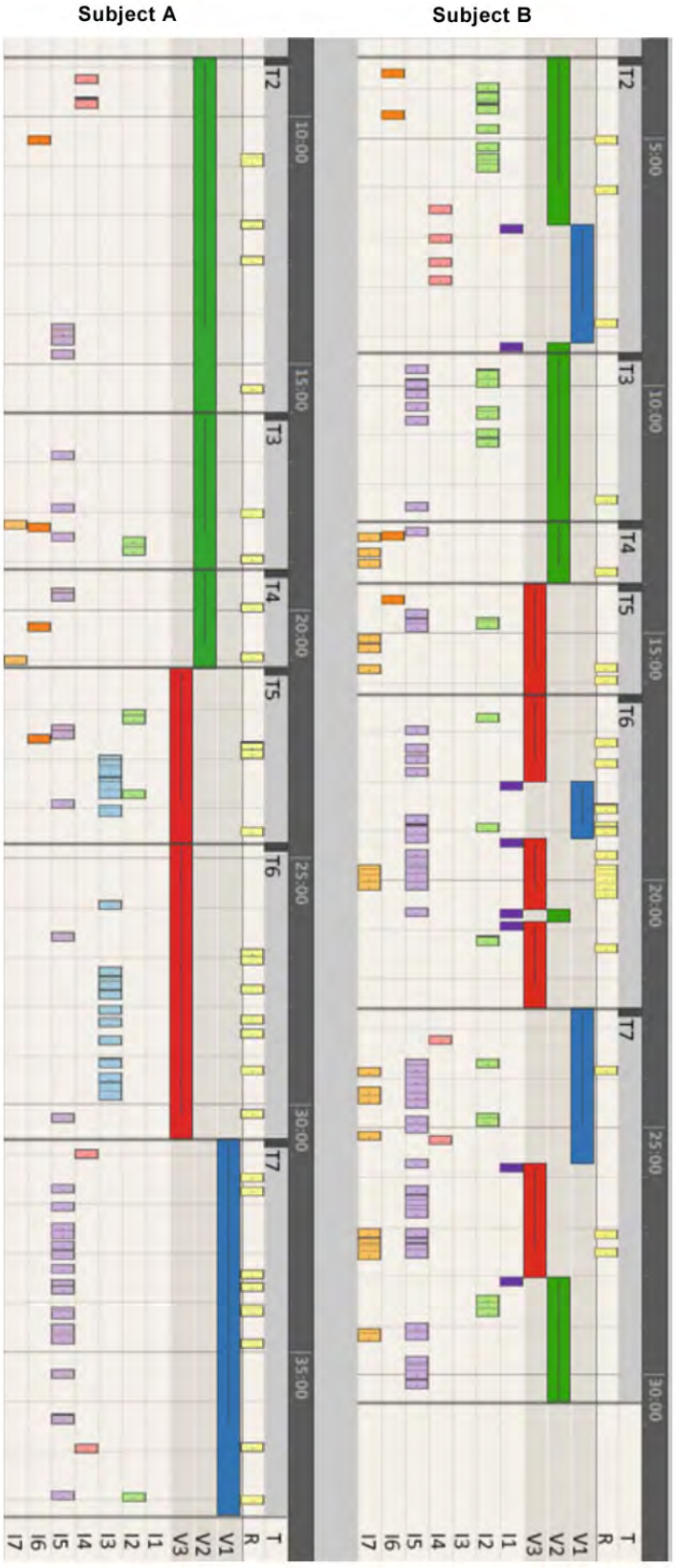


Figure 9.3: 8 Interaction graph for the tasks 2–7 of the prototype test, by two subjects

In this context, an insight is meant as a guess, a partial answer (in the sense of knowledge-building insights [90]), or any additional remark. In particular, the first two rows show the task duration (gray) and the sequence of insights (light yellow). Then, the following three rows correspond to the three views: superimposition (blue), juxtaposition (green), and 2.5D (red). Finally, remaining rows show the sequence of different types of interactions: transition between views (purple), 2D navigation (light green), 3D navigation (light blue), change of layout stability (pink), highlighting nodes and/or their trajectories (light purple), computing an SNA metric (orange), and showing additional details in the info area or in the tooltip (light orange). Comparing the exploration behavior of these two subjects, namely Subject A (top) and Subject B (bottom), we see that Subject A was faster, used as few interactions as possible and solved the task straightforwardly. Conversely, Subject B seemed to exploit task 1 to play with the tool, exploring most of its views and interactions. It is worth noting that both were possible expected behaviors, given the openness of the first task.

In Figure 9.3 we compare the exploration strategies of the same pair of subjects dealing with tasks 2-–7. While the correctness of answers is the same (100%) and completion times are similar, we can identify very different patterns of interaction and few similarities. Overall, we note that subject A switched views quite often, while Subject B never changed the predefined view that was offered by the experiment setup. Moreover, Subject A never performed 3D panning or rotation, which were used quite often by Subject B; conversely, Subject A made an intensive use of the tooltip while Subject B used it very seldom. Nevertheless, there is a similarity in the layout adjustment, which was performed by both users only for task 2 (clusters and their stability) and task 7 (shifts from core to periphery).

Looking in detail at specific tasks, we also find more differences than similarities. When dealing with task 2, for example, Subject A started the analysis at a local level with a lot of 2D zooming and panning and then some layout adjustments, while Subject B set the layout at first and then analyzed the network at a global level, during a long visual reasoning phase without any interaction, besides some highlighting in the end. For task 4 (node outdegree), both subjects had the same interaction pattern, but with a difference: Subject A gave an answer only after reading the SNA value in the tooltip, while Subject B counted highlighted nodes and gave the correct answer, then used the tooltip to confirm it. Tasks 5 and 6, whose predefined view was the 2.5D, also showed differences: Subject A explored the view by highlighting nodes and trajectories, while Subject B navigated it in the 3D space. For task 5, in particular, we can see as both subjects launched an SNA computation, but Subject A looked at the numeric value in the tooltip, while Subject B looked at its visual mapping (as explained above).

In general, considering all subjects of our study, we found that the different views complement each other, and the 'best' view does not depend only on the data and task, but also on users, who might have different strategies even if they belong to a homogeneous group with a common background. Furthermore, even a single user dealing with a single task, might find beneficial switching from one view to another to ensure the correctness and/or the completeness of her insights. Similarly, for many tasks there is no perfect choice between only visualization (node-link diagrams) and only computation (numbers and tables), but the best choice is to integrate both of them to support the visual analytics reasoning process. This approach enables the user to exploit his/her preferred problem solving strategy at best and to gain complex insights from

a multifaceted methods approach to network analysis. A possible disadvantage is that beginners might choose a wrong way, and in more complex cases training might be needed to help them switching to the fastest and most accurate strategy, but in general flexibility and multiple options seem to be beneficial.

CHAPTER 10

Quantitative User Study[1]

We performed a user study to explore interaction in the context of dynamic graphs visualization. In particular, we considered a timeline visualization with the juxtaposition approach, where several node-link diagrams are arranged along a horizontal timeline (Figure 10.1). We evaluated two interaction techniques. The first one is the interactive control of the layout stability, which is executed by the means of a slider control (thus, for the sake of brevity, we will refer to it as the *Slider*). The second interaction is the highlighting of adjacent nodes (briefly, *Highlighting*).

10.1 Study design

We designed our user study as a quantitative controlled task-based evaluation, with two observed dependent variables: time and error. We considered two factors, i.e. independent variables: the presence of the *Slider* interaction (2 levels: off/on), and the presence of the *Highlighting* interaction (2 levels: off/on). In other words, we considered four different interfaces: no-interaction, only *Slider*, only *Highlighting*, and both interactions. We chose this design in order to compare the two interaction with each other and with the non-interactive baseline, and also to assess how the two interactions work together.

We tested 6 *task* types. The full factorial design led to a total amount of $N = Task \times Slider \times Highlighting = 6 \times 2 \times 2 = 24$ conditions. To mitigate the effects of personal skills and preferences, we chose a within-subject design; each subject tested 24 conditions, by solving a different task for each of the six task type and for each of the four interfaces. In order to lower the cognitive effort of switching between different interfaces, we grouped conditions by interface. To mitigate the effects of learning and fatigue, we used a Latin square arrangement of the interfaces and we randomized the order of the tasks within each interface. Moreover, we randomized the initial slider position.

[1] Main contents of this chapter have been presented at the International Symposium on Graph Drawing and Network Visualization [143]

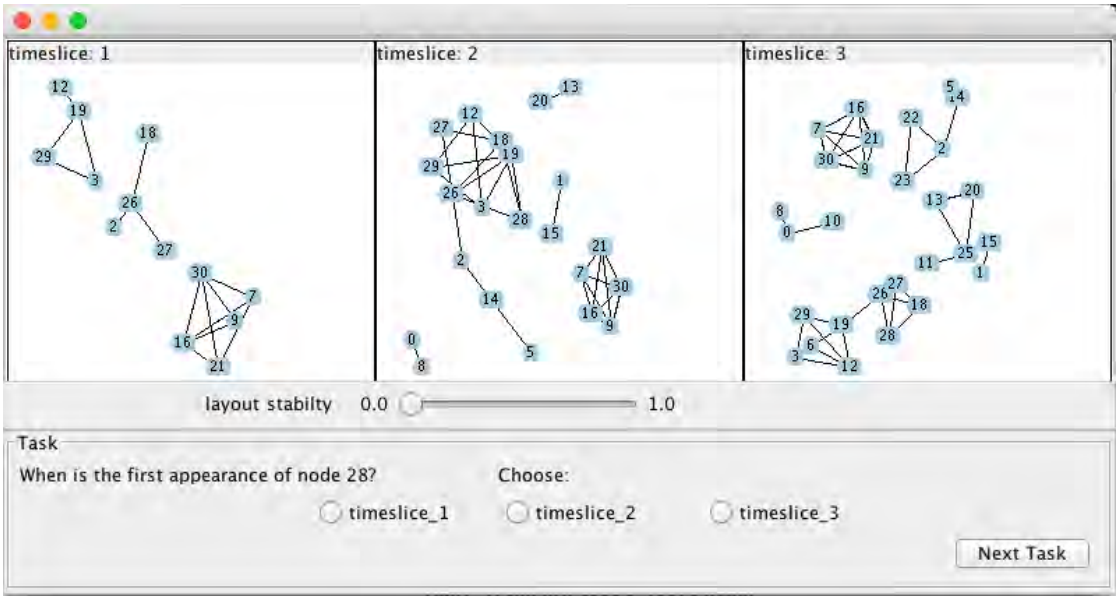


Figure 10.1: The remote evaluation software displays stimuli, provides instructions, and measures time and error.

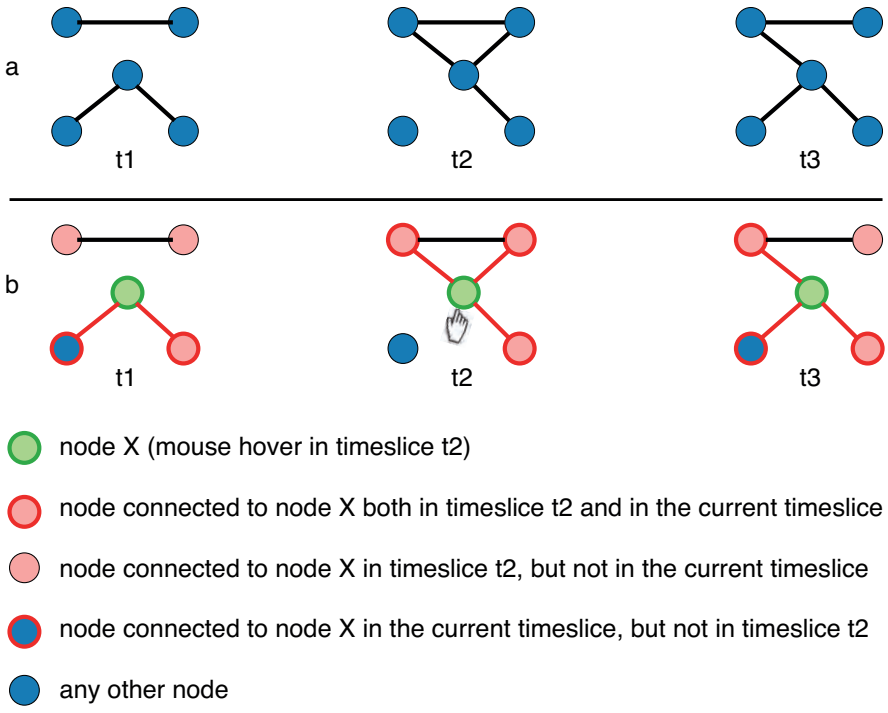


Figure 10.2: a: a dynamic graph over three time slices; a: the same graph highlighted on a mouse-over event.

10.2 Stimuli

We selected as baseline a spring-embedder layout as implemented in the Prefuse visualization toolkit [191](Figure 10.1). According to the *linking* approach [67], we added inter-time links to the graph in order to ensure layout stability. The *Slider* controls the amount of stability by interactively changing the relaxed lengths of inter-time springs. In the implementation of the *Highlighting* technique used for our experiment, when the user moves the mouse pointer over a node, a different combination of fill and stroke colors is used to highlight each different type of "adjacent" graph item, as shown in Figure 10.2. This interaction unifies the dual-mode highlighting introduced in Section 5.4 and combines the connectivity highlighting and the continuity highlighting in a single interaction. For the experiment we used real-world datasets: the dynamic graphs of social relationship between university freshmen collected by van Duijn, consisting of 38 nodes and 5 time points, and the one collected by van de Bunt, consisting of 49 nodes and 7 time points [379]. Through a threshold mechanism we derived two dynamic unweighted (i.e., binary) graphs from the original dynamic weighted graphs. Each task involved only subsets of 3 time slices.

10.3 Tasks

We selected six different types of tasks (Table 10.1). As a criterion for the selection of the tasks, we considered existing studies on the importance of layout stability and tried to elicit a set of similar tasks, in order to have comparable results. Furthermore, we considered the task taxonomy by Ahn et al. [9] in order to have a meaningful and representative set of tasks along its three axes. As for the graph entities, we included tasks referring to all the levels: the *entity* level (nodes, links), the *group* level (paths, components), and the *graph* level (the entire graph).

Table 10.1: Task types, examples, and classifications

Task	Description	by [9]	[25]	[32]
1.NO	Node Occurence	Event/Node	Local/Distinguishable	When
	e.g., *When is the first appearance of node 27?*			
2.LO	Link Occurence	Event/Link	Local/Distinguishable	When
	e.g., *When is the last appearance of link 6–9?*			
3.ND	Node Degree	Event/Group	Local/Indistinguishable	When
	e.g., *When does the smallest degree (number of connections) of node 10 occur?*			
4.SP	Shortest Path	Event/Group	Local/Distinguishable	When
	e.g., *When does the largest geodesic distance between node 7 and node 9 occur?*			
5.CC	Connected Components	Growth/Graph	Global/Indistinguishable	What
	e.g., *Is the number of connected components increasing, decreasing, or stable?*			
6.AL	All Links	Growth/Graph	Global/Indistinguishable	What
	e.g., *Is the total number of edges increasing, decreasing, or stable?*			

As for the properties, we disregarded tasks referring to *domain properties* and only considered tasks referring to *structural properties*, which are specific aspects for graphs. As for the temporal features, we included tasks referring to *individual events* and *contraction & growth*, scoping out more complex tasks, which can be investigated in a follow-up study. In order to better describe the nature of our tasks and to enable a better interpretation of results, we also categorized our tasks according to other existing taxonomies [25, 32], as shown in Table 10.1.

10.4 Subjects' pool and study settings

We conducted the experiment remotely by using the Evalbench toolkit [13] (Figure 10.1). In order to assess the technical setup, the estimated overall length of the evaluation session, and the understandability of textual descriptions of our tasks, we performed two pilot tests with direct observation of subjects, and then we implemented small adjustments before the main remote study. For the main study we recruited 64 volunteer subjects among undergraduate students at the fifth semester of a bachelor programme in Visual Computing. All the subjects had normal or corrected vision. Right after the recruiting, we instructed the subjects with a 15 minute briefing, describing the visualization and the interactions to be evaluated, and recalling the necessary concepts from graph theory (e.g., the notion of geodesic distance as shortest path, or the notion of connected components). The subjects were instructed to be fast and accurate in solving the tasks, without assigning any priority between speed and accuracy. The evaluation software included a training session for each of the four interfaces. During the training sessions, the software does not collect data; it shows the correct answer after completion of each task and allows repetitions until the subject feels confident of having understood the task types and the interface. The test, including the training sessions, had an average duration of 20 minutes.

10.5 Hypotheses

We designed our experiment to test three hypotheses: **A)** the *Slider* reduces error rates at the cost of longer completion times, in comparison with the non-interactive interface; **B)** the *Highlighting* reduces error rates at the cost of longer completion times, in comparison with the non-interactive interface; **C)** the *Highlighting* outperforms the *Slider*.

We hypothesize that each interaction reduces error rates in comparison with the non-interactive interface, because both interactions comply with the *rule of self-evidence* and address the *adjacency task*. The *rule of self-evidence* for multiple views prescribes the use of "perceptual cues to make relationships among multiple views more apparent to the user" [37]. The *Highlighting* complies with this rule, by drawing attention to different instances of the same node across different time slices; the *Slider*also complies with this rule, by allowing the user to select the maximum stability and fix node positions across different time slices. The *adjacency task* (i.e., "Given a node, find its adjacent nodes") has been identified as the only graph-specific task [253]. The *Highlighting* obviously addresses this task, as well as the *Slider* does, by allowing the user to select the minimum stability and exploit the proximity *Gestalt* principle [51].

Conversely, we hypothesize that both interactions increase the task completion time in comparison with the non-interactive interface. We make this hypothesis in analogy with the existing

comparative evaluations between animation and (static) timeline views [133] [23], while we consider interactive timeline views as a middle way. More specifically, in terms of interaction costs [250], the *Highlighting* might increase the completion time because of the physical-motion cost of tracking elements with the mouse, while the *Slider* might imply view-change costs of reinterpreting the perception when the layout rearranges. For both techniques, there might be the decision costs of forming goals, such as deciding whether the available interaction is useful to solve the given task, and how. Moreover, the simple fact that the GUI provides an interactive option might lead users to explore its use, in order to form a solving strategy before solving a task, or to possibly increase the confidence about the solution afterwards.

Furthermore, we hypothesize that the *Highlighting* will have better performance than the *Slider*. We derive this hypothesis from the observation that the *Highlighting* is a common and relatively simple interaction, which at least partially exploits pre-attentive processing, while the *Slider* is based on a novel and complex concept. In other words, while the *Highlighting* directly addresses the issue of connecting entities along two dimensions (time and graph structure), the *Slider* implicitly introduces another dimension, since the stability lies in the parameter space of the layout algorithm.

10.6 Analysis

We preprocessed data collected from 64 subjects in order to assess whether they were eligible for analysis and we had to discard one subject whose logs were corrupted. The analysis was then performed on data from 63 subjects, consisting of 3024 samples in total.

We checked the completion times for normality with the Shapiro-Wilk goodness-of-fit test but the check failed. We then applied a logarithmic transformation to the completion times and checked again the normality with a positive result. The verification of the Gaussian condition assured the applicability of parametric tests; we could perform the analysis of variance through an ANOVA with the subject as a random variable. If the ANOVA found a factor to have a statistically significant effect, we compared the two levels of that factor with a pairwise post-hoc Student's t test; if the ANOVA found the interaction between factors to be statistically significant, we performed an all-pairs Tukey's honestly significant difference (HSD) post-hoc test.

The error can be understood as a dichotomous (i.e. binary) variable, since there are only two possible outcomes for each data sample (correct, not correct). Hence, we analysed the error by logistic regression as a generalized linear model (GLM) with a binomial distribution and a logit transformation as the link function, computing likelihood ratio statistics. If a factor was found to be a significant effect, we analysed the contrast between its levels in terms of pairwise comparisons between estimated marginal means.

10.7 Results

Figure 10.3 shows time and error by *Highlighting* and *Slider*, grouped by *Task*; time is represented by box-plots with first, second (median), and third quartile, while error is represented by bars (mean) and error bars (standard error). Figure 10.4 shows statistically significant differences.

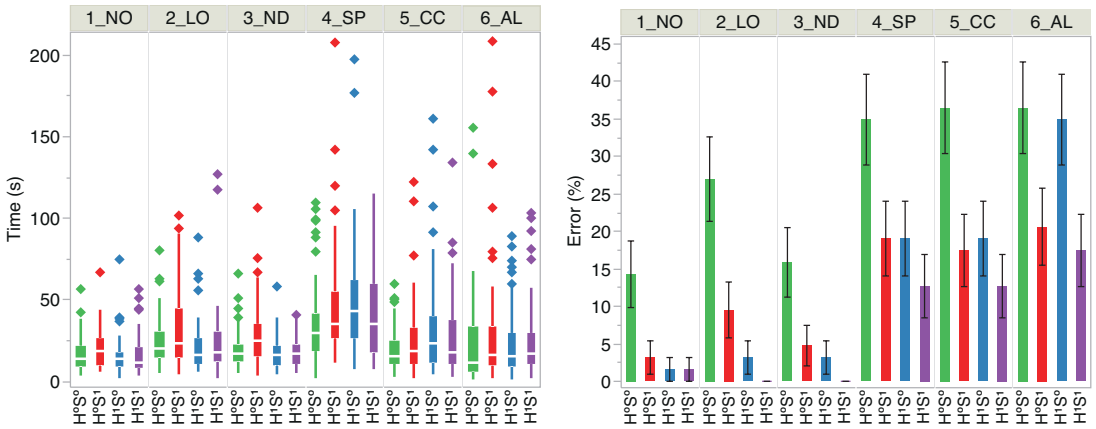


Figure 10.3: Time (left-hand side, as box plots) and error (right-hand side, as bars representing means and error bars representing standard error) by *Highlighting* and *Slider*, grouped by *Task*. H^0S^0 H^0S^1 H^1S^0 H^1S^1

In light of these results, we can verify our hypotheses.

Hypothesis A is partially confirmed. The *Slider* decreases the error rate for all tasks but the easiest one (1.NO), and it increases the completion time for tasks 3.ND and 4.SP only.

Hypothesis B is partially confirmed. The *Highlighting* decreases the error rate for all tasks but the most difficult one (6.AL); it increases completion times for tasks 4.SP and 5.CC, but it reduces it for task 2.LO, and does not affect the remaining tasks.

Hypothesis C is partially confirmed. The *Highlighting* outperforms the *Slider* for tasks 1.NO, 2.LO, and 3.ND. For task 4.SP, the *Highlighting* and the *Slider* score equally: each of them decreases the error rate (by the same amount) and also increases the completion time if used alone, but when used together they do not increase the completion time, showing a desirable effect interaction. For task 5.CC, both factors reduce the error rate, but the *Highlighting* also increases the completion time when used alone. For task 6.AL, the only significant effect is that the *Slider* reduces the error rate.

Besides the verification of our hypothesis, which are mostly confirmed, our user study provides interesting insights about the differences between tasks. First of all, we observe that the differences in error rate and completion time among the tasks are significant, hence we can confirm that in general our tasks have different levels of difficulty. Secondly, we observe that the effectiveness of the tested interaction techniques varies with the levels of difficulty of the tasks. In a very brief but accurate summary we can say that, for easier tasks, the *Highlighting* decreases error rates and in some cases even decreases completion times; conversely, for more difficult tasks, it is the *Slider* that decreases error rates. Moreover, for tasks 3.ND, 4.SP and 5.CC, one technique increases completion times if used alone, but it does not if used in combination with the other one. Looking back at the classification of our tasks (Table 10.1), we can also identify the relevant aspects. We can observe that, for those tasks involving simpler temporal features of distinguishable single entities (1.NO and 2.LO), or indistinguishable

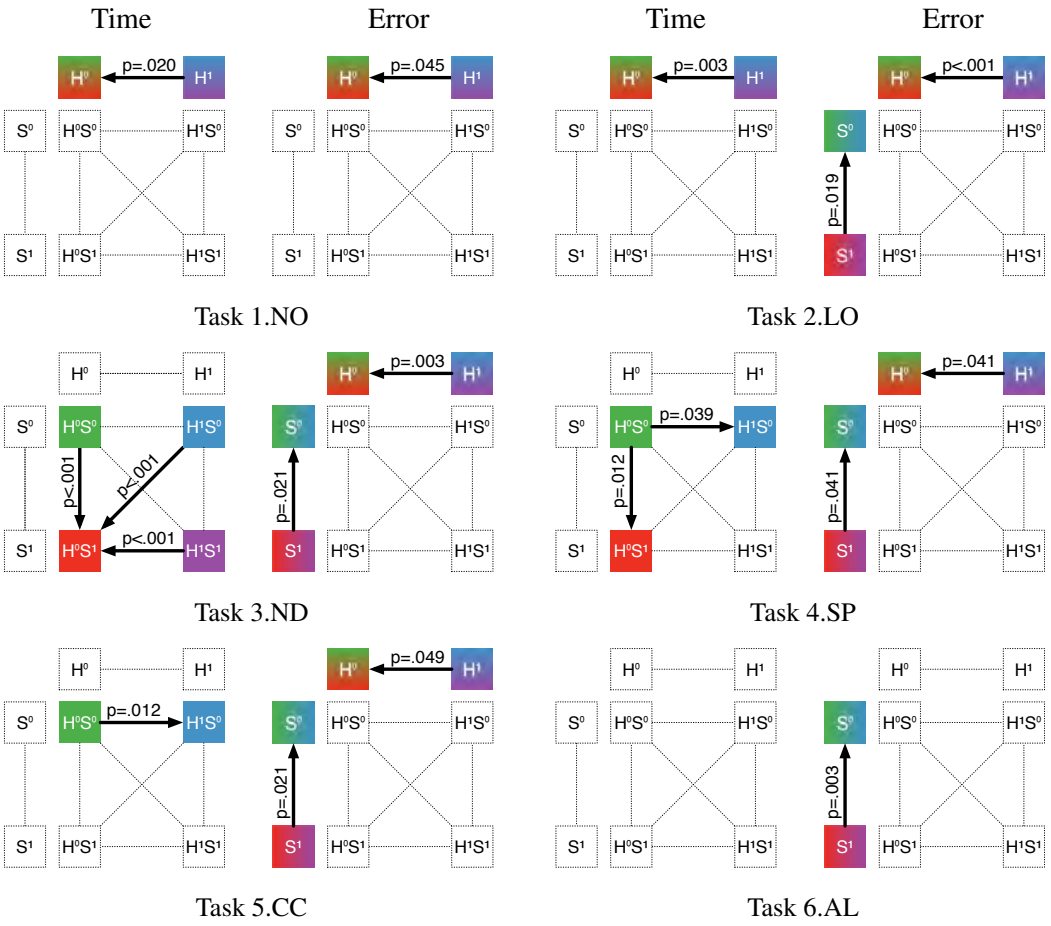


Figure 10.4: Statistically significant differences for time and error, by *Task*. An arrow means that the source is faster or, respectively, more accurate than the destination, with the reported probability. Lines represent all-pairs comparisons between factor combinations (■ H^0S^0, ■ H^0S^1, ■ H^1S^0, and ■ H^1S^1), as well as pairwise comparisons by *Highlighting* (H^0–H^1, top) and *Slider* (S^0–S^1, left).

groups (3.ND), the *Highlighting* is more effective. For those tasks that refer to more complex temporal features at the graph level, even if indistinguishable (5.CC and 6.AL), the *Slider* is more effective. Task 4.SP is about the changes of the geodesic distance between two nodes and requires the distinct identification of several nodes and links along the shortest path. In this case both techniques are equally accurate; if (and only if) they are used together, they do not even slow down the analysis. We can conjecture (by also considering our observations during the pilot experiments) that during the completion of a such complex task, the *Slider* can be used to switch back and forth between the minimum stability (to guess geodesic distances and shortest paths based on Euclidean distances) and the maximum stability (to identify instances of nodes and links across different time slices), while the *Highlighting* helps with tracking objects. As

for task 6.AL, the *Slider* resulted to be effective; we hypothesize (by also considering our pilot observations) that subjects simply set the minimum stability and looked at the total graph area as an estimator of the density. We would have expected the *Highlighting* to be also effective, since the analysis of the degree a few central nodes might provide a good estimator of the graph density, given the power-law distribution of real-world networks. The results show that the test subjects did not exploit this expert strategy.

10.7.1 Design implications

Both the *Slider* and the *Highlighting* are effective interaction techniques for dynamic graph visualization, and their use generally improves user performances. In those circumstances where it might be not possible to include them both (for example, if the color channel is employed to encode attributes of multivariate graphs, or if the GUI is already overloaded with many controls), our evaluation provides an indication to designers according to the user tasks to be supported. Our results suggest that the *Highlighting* is indicated for tasks involving temporal features of distinguishable single entities or indistinguishable groups, while the *Slider* is indicated for tasks involving complex temporal behaviours at the graph level. The joint use of both interactions is beneficial for the most complex task involving temporal behaviours of connectivity paths.

10.7.2 Limitations

We acknowledge the limitations of our experiment, whose findings might not be directly generalizable to large-scale datasets. The highlighting interaction for dynamic graphs is much more complex then the standard connectivity highlighting for static graphs, and may require training to be understood and used effectively. The stability control might have a different effect when combined with 3D visualization and interaction techniques (e.g., the vertigo zoom, see section 5.6.1). However, our study provides preliminary clues for visualization designers who need to choose the most appropriate interaction technique for their users' tasks.

Part IV

Conclusion

CHAPTER 11

Conclusion

In this final chapter we summarize our contributions to the scientific area of visual analytics and illustrate how they address our initial research questions; furthermore, we report how we disseminated the outcomes of our work by scientific publications and presentations; finally, we outline possible follow-ups and future research directions.

11.1 Summary of contributions

We applied a visual analytics approach to the problem of examining dynamic (i.e., time-oriented) network data, in particular organizational social networks. We developed novel techniques for automated analysis, visualization, and user interaction, and demonstrated how they can be tightly combined to support the sense-making process. We summarize our contributions by putting them in context of concepts and state-of-the-art techniques we discussed in Chapter 2:

Data. We focused on a particular type of time-oriented networks. With respect to characteristics of time [14], our data are characterized by a discrete scale, an interval-based scope, a linear arrangement, an ordered viewpoint; the time primitives are instants (obtained by a slicing transformation), with a single granularity.

Automated analysis. We defined a set of novel metrics for this kind of dynamic networks, and designed algorithms to compute them. The *change centrality* metric combines relational and temporal analysis, from both a local and a global perspective: it highlights when and where the network changed, and how local changes affect the overall relational structure.

Visualization. As for the relational aspect, we used the same visual encoding (i.e., a node-link diagram) with two different spatial arrangements: a force-directed layout and a measure-based (i.e., scatter-plot like) layout. As for the visual encoding of the temporal aspect, we combined three different views, which are based on a static approach (i.e., mapping time to space) but

feature diverse encodings. The superimposition view can be understood as a *Time in Graph* encoding: the primary geometry is determined by the relational aspect, while the temporal aspect is mapped to retinal variables (in particular alpha-channel, i.e. transparency). The juxtaposition and the two-and-a-half-dimension view exploit a *Graph in Time* visual encoding: the primary geometry is based on a timeline, and node-link diagrams representing each time slice are placed side by side or, respectively, stacked, along the timeline. The results of automated analysis can be mapped to both retinal and planar variables. In the former case, metrics are mapped to color and size of nodes, or to color and thickness of node trajectories; in the latter case, static SNA metrics are used to compute the scatter-plot layout, and the change centrality is utilized to improve the dynamic layout.

Interaction. We summarize the interaction techniques utilized in out approach according to user intents [423]. We designed smooth animated transitions between views, which fulfil two needs: explaining how the views are constructed and relate to each other, and preserving the exploration context when the user switches from a visual encoding of temporal aspects to another; this interaction supports the *encode* user intent. We introduced a technique for interactive control of layout stability, by which user can adapt the amount of mental map preservation to their preference as well as the data and task at hand. This novel interaction fulfils the *rearrange* user intent. The novel vertigo zoom interaction supports the *encode* and the *rearrange* user intents; it enables a seamless transition between the relational and the temporal perspective in the context of two-and-a-half-dimensional view on dynamic networks. The combined analysis of relational and temporal aspects of dynamic networks is also pursued by an interaction technique that supports *select* and *connect* user intents by combining connectivity highlighting and continuity highlighting. The on-demand computation of network metrics, which are then used to filter or enrich the visualized data, complements the set of supported user intents with *filter* and *abstract/elaborate*.

Validation. We have utilized various methods to validate all the aforementioned techniques, and their combination in a consistent and unitary visual analytics approach. The expert reviews revealed usability issues of design mock-ups and early prototypes, which we resolved in subsequent development iterations. By describing a concrete usage scenario, we illustrate how our approach can be used to analyze and gain insights from real-world data. The qualitative evaluation shows that the presence of multiple and redundant problem-solving strategies helps users address different tasks and improve correctness and confidence of answers. The quantitative evaluation, focused in particular on the highlighting interaction and the layout stability control, indicates that the former is better suited for tasks involving temporal features of distinguishable single entities or indistinguishable groups, while the latter for tasks involving complex temporal behaviours at the graph level; the joint use of both interactions is beneficial for the most complex task involving temporal behaviours of connectivity paths.

11.2 Answers to research questions

After summarizing our work, which consisted design and/or combination of novel techniques, development of a prototypical implementation, and validation by expert reviews and user studies, we can finally discuss the research questions who drove our research and provide well-grounded and empirical answers:

- **Q:** *How can a visual analytics approach support the examination of dynamic networks according to specific user tasks?*

- **A:** Our work has shown that an integrated visual analytics approach can support the examination of dynamic network by seamlessly intertwining automatic analysis, visualization, and interaction techniques, which complement and improve each other. While specific techniques resulted particularly well-suited for given tasks, our qualitative and quantitative observations indicate that their combination is beneficial to improve speed, correctness, and confidence of all tasks, and especially the most complex ones.

  The answer to the main research question can be further detailed by the answer to the three interconnected sub questions:

  - **Q:** *How can temporal aspects of network data and dynamic underlying phenomena be best visualized?*
  - **A:** We proposed the combination of different visualization techniques, which build upon different visual encodings of the time-oriented aspects as well as different relational aspects. As for the temporal aspects, we investigated visual encodings that map them to the primary geometry (i.e., the juxtaposition view and the two-and-a-half-dimension view, belonging both into the *Graph in time* category), as well as visual encodings that map them to retinal variables within a graph-based geometry (the superimposition view, a *Time in graph* visual encoding). As for the relational aspects, we utilized a node-link diagram, alternating an energy-based layout with a measure-based layout (similar to a scatter plot). The empirical evaluation we performed indicates that this combination of visual encodings is effective, especially if supported by an adequate interaction technique which animates transitions between them.
  - **Q:** *How can we integrate analytical methods and visualization techniques to address the complexity of dynamic networks?*
  - **A:** We introduced *change centrality*, a novel metrics for dynamic networks, and a set of metrics deriving from change centrality. We showed how they can be utilized to filter the data (emphasizing either changes or stable cores on-demand), or visualized (e.g., mapped to planar variables to improve the dynamic layout, or to retinal variables to enrich network data and facilitate their interpretation).
  - **Q:** *Is it possible to combine analytical methods and interaction techniques to enhance the perception of network dynamics?*

– **A:** We introduced several novel interaction techniques and showed how they can be combined with results of automated analysis. The interactive control of the layout complements the change centrality metrics, allowing the user to fine tune the layout by selecting the preferred amount of stability. The *vertigo zoom* interaction, enabling smooth transitions between relational and temporal perspectives, is particularly useful in combination with the scatter-plot layout, which is computed on top of network metrics. More in general, we observed that the combination of visual interaction and automated analysis effectively supported users' multiple problem-solving strategies.

11.3 Future directions

We identified three main directions to extend our research on dynamic networks beyond the scope of this dissertation: scale, time, and structure.

Scale In this work, we focused on relatively small (up to 50 vertices) and not very dense (around 500 edges) networks, which is a sufficient scale for our problem domain, i.e. the analysis of interpersonal relationships in organizational contexts. However, other application domains present larger scales, and pose the challenge of adapting existing approaches to visual scalability (reduction by filtering and/or aggregation, see Section 2.6.1.2) for the time-oriented aspects of dynamic networks.

Time In this work, we considered dynamic networks with particular time-oriented characteristics, namely a discrete time scale, only instants as time primitives, and a single time granularity (see Section 2.1.1). Interesting research possibilities open up by considering more complex time characteristics, for example a continuous time scale, both instants and intervals as time primitives, and multiple granularities organized into calendar systems.

Structure The organizational social networks we considered in this work are simple and univariate (see Section 2.1). Other applications domain deal with multi-variate data, which can be modelled as multi-modal networks (featuring vertices of different types) and/or multi-relational networks (featuring edges of different types). Furthermore, this networks can be structured as compound graphs, i.e., their nodes can be grouped in multi-level hierarchies. While this kind of structure provides an inherent aggregation which helps with the visual scalability in the static case, in general it poses the challenge of dealing with its dynamic time-oriented aspects.

11.3.1 Follow-up project

The above-mentioned directions led us to the development of a new research project: we are currently extending and adapting the outcomes of this dissertation to another application domain, namely the visual analysis of patents. Patent data, indeed, present more complex temporal characteristics: a continuous time scale (since they are filed on a daily basis), diverse time primitives (their filing date is represented by an instant, but their validity term is represented by an interval), diverse granularities (they are filed on a daily basis, but they can be analysed by

monthly or yearly aggregation). Patent datasets have very large scales (the European Patent Office dataset contains 90 million patent documents). Patent datasets can be modelled as multi-modal networks (including diverse entities such as patents, inventors, companies, agents) and multi-relational networks (for example citations, co-citations, co-authorship), and comprehend also hierarchies (for example, several patent classification systems). Preliminary results from this follow-up research are reported in the next section.

11.4 Publications and dissemination

The main scientific outcomes of this work contribute to the field of computer science, in general, and visualization, in particular. However, since we conducted our work in the context of a problem-driven applied research, in collaboration with an interdisciplinary team, it also contributed to the development of the problem domain.

11.4.1 Publications about dynamic networks in computer science

- P. Federico, W. Aigner, S. Miksch, F. Windhager, and L. Zenk. A visual analytics approach to dynamic social networks. In *Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies*, i-KNOW 11, pages 47:1–47:8, New York, NY, USA, 2011. ACM

- P. Federico, J. Pfeffer, W. Aigner, S. Miksch, and L. Zenk. Visual analysis of dynamic networks using change centrality. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 179–183, Aug 2012

- P. Federico, W. Aigner, S. Miksch, F. Windhager, and M. Smuc. Vertigo zoom: Combining relational and temporal perspectives on dynamic networks. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 437–440, New York, NY, USA, 2012. ACM

- P. Federico, W. Aigner, S. Miksch, F. Windhager, and L. Zenk. Visual analytics of dynamic networks - a case study. In *The Third International UKVAC Workshop on Visual Analytics (VAW)*, 2011

- M. Smuc, P. Federico, F. Windhager, W. Aigner, L. Zenk, and S. Miksch. How do you connect moving dots? insights from user studies on dynamic network visualizations. In W. Huang, editor, *Handbook of Human Centric Visualization*, pages 623–650. Springer, 2014

- P. Federico and S. Miksch. Evaluation of two interaction techniques for visualization of dynamic graphs. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization: 24th International Symposium, GD 2016, Athens, Greece, September 19-21, 2016, Revised Selected Papers*, pages 557–571. Springer International Publishing, Cham, 2016

- P. Federico. Visual analytics of dynamic social networks. In *Doctoral Colloquium of the IEEE Conference on Visualization*, 2011

- P. Federico, W. Aigner, S. Miksch, J. Pfeffer, M. Smuc, F. Windhager, and L. Zenk. Viena: Visual enterprise network analytics. In K. Matkovic and G. Santucci, editors, *Poster Proceedings of the 3rd International Workshop on Visual Analytics (EuroVA)*, page 12. Eurographics, Eurographics, 2012

Follow-up project We list here preliminary publications in the context of the follow-up project on visual analytics of patent network dynamics, building upon work presented in this dissertation but beyond its core scope:

- P. Federico, F. Heimerl, S. Koch, and S. Miksch. A survey on visual approaches for analyzing scientific literature and patents. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):1–1, 2016

- F. Windhager, A. Amor-Amoros, M. Smuc, P. Federico, L. Zenk, and S. Miksch. A concept for the exploratory visualization of patent network dynamics. In *Proceedings of the 6th International Conference on Information Visualization Theory and Applications*, pages 268–273, 2015

- A. Amor-Amorós, P. Federico, and S. Miksch. Timegraph: A data management framework for visual analytics of large multivariate time-oriented networks. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 217–218, Oct 2014

- A. Amor-Amorós, P. Federico, and S. Miksch. Visually-supported graph traversals for exploratory analysis. In *Poster Proceedings of the IEEE Visualization Conference (VIS)*, 2016

- P. Federico, A. Amor-Amorós, and S. Miksch. A nested workflow model for visual analytics design and validation. In *Proceedings of the Sixth Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization*, BELIV '16, pages 104–111, New York, NY, USA, 2016. ACM

11.4.2 Publications about dynamic networks in other domains

- P. Federico, F. Windhager, L. Zenk, and M. Smuc. Visual analytics of dynamic networks. In *XXX INSNA Sunbelt Social Networks Conference*. International Network for Social Network Analysis, 2010

- F. Windhager, L. Zenk, and P. Federico. Visual enterprise network analytics - visualizing organizational change. *Procedia - Social and Behavioral Sciences*, 22:59 – 68, 2011

- F. Windhager, M. Smuc, L. Zenk, P. Federico, J. Pfeffer, and W. Aigner. On visualizing knowledge flows at a university department. *Procedia - Social and Behavioral Sciences*, 100:127 – 143, 2013

- F. Windhager, M. Smuc, L. Zenk, P. Federico, J. Pfeffer, W. Aigner, and S. Miksch. Visual knowledge networks analytics. In J. Liebowitz, editor, *Knowledge Management Handbook*, page 187–206. CRC Press, 2012

11.4.3 Other publications in computer science

The research project who constitutes the core of this dissertation was conducted in parallel with other research projects in visualization and visual analytics, not directly related to dynamic networks. For completeness, we report the scientific outcome of these parallel projects:

- W. Aigner, P. Federico, T. Gschwandtner, S. Miksch, and A. Rind. Challenges of time-oriented data in visual analytics for healthcare. In J. J. Caban and D. Gotz, editors, *Proceedings of the IEEE VisWeek Workshop on Visual Analytics in Healthcare*, 2012. Vortrag: IEEE VisWeek Workshop on Visual Analytics in Healthcare, Seattle; 2012-10-17

- P. Bodesinsky, P. Federico, and S. Miksch. Visual analysis of compliance with clinical guidelines. In *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies*, i-Know '13, pages 12:1–12:8, New York, NY, USA, 2013. ACM

- P. Federico, S. Hoffmann, A. Rind, W. Aigner, and S. Miksch. Qualizon graphs: Space-efficient time-series visualization with qualitative abstractions. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, AVI '14, pages 273–280, New York, NY, USA, 2014. ACM

- P. Federico, J. Unger, A. Amor-Amorós, L. Sacchi, D. Klimov, and S. Miksch. Gnaeus: Utilizing Clinical Guidelines for Knowledge-assisted Visualisation of EHR Cohorts. In E. Bertini and J. C. Roberts, editors, *EuroVis Workshop on Visual Analytics (EuroVA)*. The Eurographics Association, 2015

- T. Gschwandtner, M. Bögl, P. Federico, and S. Miksch. Visual encodings of temporal uncertainty: A comparative user study. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):539–548, Jan 2016

- A. Rind, P. Federico, T. Gschwandtner, W. Aigner, J. Doppler, and M. Wagner. Visual analytics of electronic health records with a focus on time. In G. Rinaldi, editor, *New Perspectives in Medical Records: Meeting the Needs of Patients and Practitioners*, pages 65–77. Springer International Publishing, Cham, 2017

Bibliography

[1] J. Abello, D. DeSimone, S. Hadlak, H.-J. Schulz, and M. Sumida. Visualizing life in a graph stream. In M. Dehmer, F. Emmert-Streib, S. Pickl, and A. Holzinger, editors, *Big Data of Complex Networks*, pages 293–312. CRC Press, 2016.

[2] J. Abello, S. Hadlak, H. Schumann, and H.-J. Schulz. A modular degree-of-interest specification for the visual analysis of large dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):337–350, 2013.

[3] J. Abello and J. Korn. Mgv: a system for visualizing massive multidigraphs. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):21–38, Jan 2002.

[4] J. Abello and F. van Ham. Matrix zoom: A visual interface to semi-external graphs. In *IEEE Symposium on Information Visualization*, pages 183–190, 2004.

[5] C. C. Aggarwal. *Social Network Data Analytics*. Springer Publishing Company, Inc., 1st edition, 2011.

[6] V. Ahlers, F. Heine, B. Hellmann, C. Kleiner, L. Renners, T. Rossow, and R. Steuerwald. Replicable security monitoring: Visualizing time-variant graphs of network metadata. In *Joint Proceedings of the Fourth International Workshop on Euler Diagrams and the First International Workshop on Graph Visualization in Practice*, GraphViP, pages 32–41. CEUR-WS.org, 2014.

[7] A. Ahmed, T. Dwyer, C. Murray, L. Song, and Y. X. Wu. Wilmascope graph visualisation. In *IEEE Symposium on Information Visualization*, pages r4–r4, Oct 2004.

[8] A. Ahmed, X. Fu, S. H. Hong, Q. H. Nguyen, and K. Xu. Visual analysis of dynamic networks with geological clustering. In *2007 IEEE Symposium on Visual Analytics Science and Technology*, pages 221–222, Oct 2007.

[9] J.-w. Ahn, C. Plaisant, and B. Shneiderman. A task taxonomy for network evolution analysis. *IEEE Transactions on Visualization and Computer Graphics*, 20(3):365–376, Mar. 2013.

[10] J.-w. Ahn, M. Taieb-Maimon, A. Sopan, C. Plaisant, and B. Shneiderman. Temporal visualization of social network dynamics: Prototypes for nation of neighbors. In *Social Computing, Behavioral-Cultural Modeling and Prediction*, SBP, pages 309–316. Springer, 2011.

[11] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993.

[12] W. Aigner, P. Federico, T. Gschwandtner, S. Miksch, and A. Rind. Challenges of time-oriented data in visual analytics for healthcare. In J. J. Caban and D. Gotz, editors, *Proceedings of the IEEE VisWeek Workshop on Visual Analytics in Healthcare*, 2012. Vortrag: IEEE VisWeek Workshop on Visual Analytics in Healthcare, Seattle; 2012-10-17.

[13] W. Aigner, S. Hoffmann, and A. Rind. Evalbench: A software library for visualization evaluation. *Computer Graphics Forum*, 32(3pt1):41–50, 2013.

[14] W. Aigner, S. Miksch, H. Schumann, and C. Tominski. *Visualization of Time-Oriented Data*. Human-Computer Interaction. Springer Verlag, 1st edition, 2011.

[15] A. B. Alencar, K. Börner, F. V. Paulovich, and M. C. F. de Oliveira. Time-aware visualization of document collections. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC, pages 997–1004. ACM, 2012.

[16] A. Amor-Amorós, P. Federico, and S. Miksch. Visually-supported graph traversals for exploratory analysis. In *Poster Proceedings of the IEEE Visualization Conference (VIS)*, 2016.

[17] A. Amor-Amorós, P. Federico, and S. Miksch. Timegraph: A data management framework for visual analytics of large multivariate time-oriented networks. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 217–218, Oct 2014.

[18] N. Andrienko and G. Andrienko. *Tasks*, pages 47–161. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[19] D. Archambault. Structural differences between two graphs through hierarchies. In *Proceedings of Graphics Interface 2009*, GI '09, pages 87–94, Toronto, Ont., Canada, Canada, 2009. Canadian Information Processing Society.

[20] D. Archambault, J. Abello, J. Kennedy, S. Kobourov, K.-L. Ma, S. Miksch, C. Muelder, and A. Telea. Temporal Multivariate Networks. In Kerren et al. [233], chapter 8, pages 151–174.

[21] D. Archambault, T. Munzner, and D. Auber. Grouseflocks: Steerable exploration of graph hierarchy space. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):900–913, July 2008.

[22] D. Archambault, T. Munzner, and D. Auber. Tuggraph: Path-preserving hierarchies for browsing proximity and paths in graphs. In *2009 IEEE Pacific Visualization Symposium*, pages 113–120, April 2009.

[23] D. Archambault, H. Purchase, and B. Pinaud. Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):539–552, Apr. 2011.

[24] D. Archambault and H. C. Purchase. The mental map and memorability in dynamic graphs. In *Proceeding of the IEEE Pacific Visualization Symposium*, PacificVis, pages 89–96, Washington, DC, 2012. IEEE.

[25] D. Archambault and H. C. Purchase. The "Map" in the mental map: Experimental results in dynamic graph drawing. *International Journal of Human-Computer Studies*, 71(11):1044 – 1055, 2013.

[26] D. Archambault and H. C. Purchase. Mental map preservation helps user orientation in dynamic graphs. In *Graph Drawing*, GD, pages 475–486. Springer, 2013.

[27] D. Archambault and H. C. Purchase. On the application of experimental results in dynamic graph drawing. In *Joint Proceedings of the Fourth International Workshop on Euler Diagrams and the First International Workshop on Graph Visualization in Practice*, volume 1244 of *GraphViP*, pages 73–77. CEUR-WS.org, 2014.

[28] D. Archambault and H. C. Purchase. Can animation support the visualisation of dynamic graphs? *Information Sciences*, 330, 2016.

[29] D. Archambault, H. C. Purchase, and B. Pinaud. Difference map readability for dynamic graphs. In *Graph Drawing*, GD, pages 50–61. Springer, 2011.

[30] D. L. Arendt and L. M. Blaha. SVEN: Informative visual representation of complex dynamic structure. *arXiv preprint arXiv:1412.6706*, 2014.

[31] S. Asur, S. Parthasarathy, and D. Ucar. An event-based framework for characterizing the evolutionary behavior of interaction graphs. *ACM Trans. Knowl. Discov. Data*, 3(4):16:1–16:36, Dec. 2009.

[32] B. Bach, P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale. A review of temporal data visualizations based on space-time cube operations. In *EuroVis-STARs*, EuroVis, pages 23–41. The Eurographics Association, 2014.

[33] B. Bach, N. Henry-Riche, T. Dwyer, T. Madhyastha, J.-D. Fekete, and T. Grabowski. Small MultiPiles: Piling time to explore temporal patterns in dynamic networks. *Computer Graphics Forum*, 2015.

[34] B. Bach, E. Pietriga, and J.-D. Fekete. GraphDiaries: Animated transitions and temporal navigation for dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 20(5):740–754, May 2014.

[35] B. Bach, E. Pietriga, and J.-D. Fekete. Visualizing dynamic networks with Matrix Cubes. In *Procceedings of the SICCHI Conference on Human Factors in Computing Systems*, CHI, 2014.

[36] J. P. Bagrow, E. M. Bollt, J. D. Skufca, and D. ben Avraham. Portraits of complex networks. *EPL (Europhysics Letters)*, 81(6):68004, 2008.

[37] M. Q. W. Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, AVI '00, pages 110–119, New York, NY, USA, 2000. ACM.

[38] W. W. R. Ball. *Mathematical Recreations and Essays*. MacMillan and co., Ney York, 2013.

[39] A.-L. Barabási. *Linked: The new science of networks*. Perseus Books Group, 2003.

[40] A.-L. Barabási and R. Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[41] J. Barnes and P. Hut. A hierarchical o (n log n) force-calculation algorithm. *Nature*, 324(6096):446–449, 1986.

[42] G. D. Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph Drawing: Algorithms for the Visualization of Graphs*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition, 1998.

[43] A. Bavelas. Communication patterns in task-oriented groups. *The Journal of the Acoustical Society of America*, 22(6):725–730, 1950.

[44] F. Beck, M. Burch, and S. Diehl. Towards an aesthetic dimensions framework for dynamic graph visualisations. In *Proceedings of the 13th International Conference on Information Visualisation*, IV, pages 592–597. IEEE, 2009.

[45] F. Beck, M. Burch, S. Diehl, and D. Weiskopf. The state of the art in visualizing dynamic graphs. In R. Borgo, R. Maciejewski, and I. Viola, editors, *EuroVis - STARs*, EuroVis, pages 83–103. Eurographics Association, 2014.

[46] F. Beck, M. Burch, S. Diehl, and D. Weiskopf. A taxonomy and survey of dynamic graph visualization. *Computer Graphics Forum*, pages n/a–n/a, 2016.

[47] F. Beck, M. Burch, C. Vehlow, S. Diehl, and D. Weiskopf. Rapid serial visual presentation in dynamic graph visualization. In *Proceedings of the 2012 IEEE Symposium on Visual Languages and Human-Centric Computing*, VL/HCC, pages 185–192. IEEE, 2012.

[48] R. A. Becker and W. S. Cleveland. Brushing scatterplots. *Technometrics*, 29(2):127–142, 1987.

[49] M. Behrisch, B. Bach, N. Henry Riche, T. Schreck, and J.-D. Fekete. Matrix reordering methods for table and network visualization. *Computer Graphics Forum*, 35(3):693–716, 2016.

[50] S. Bender-deMoll and D. McFarland. The art and science of dynamic network visualization. *Journal of Social Structure*, 7(2):2006, 2006.

[51] C. Bennett, J. Ryall, L. Spalteholz, and A. Gooch. The Aesthetics of Graph Visualization. In D. W. Cunningham, G. Meyer, and L. Neumann, editors, *Computational Aesthetics in Graphics, Visualization, and Imaging*. The Eurographics Association, 2007.

[52] F. Berger. From circle and square to the image of the world: a possible interpretation for some petroglyphs of merels boards. *Rock Art Research*, 21(1):11–26, 2004.

[53] T. Y. Berger-Wolf and J. Saia. A framework for analysis of dynamic social networks. In *Proc. of the ACM SIGKDD int. conf. on Knowledge discovery and data mining (KDD)*, pages 523–528, New York, NY, USA, 2006. ACM.

[54] M. Berlingerio, M. Coscia, F. Giannotti, A. Monreale, and D. Pedreschi. Foundations of multidimensional network analysis. In *Int. Conf. on Advances in Social Networks Analysis and Mining (ASONAM)*, 2011.

[55] J. Bertin. *Semiology of Graphics*. University of Wisconsin Press, 1983.

[56] E. Bertini and D. Lalanne. Surveying the complementary role of automatic data analysis and visualization in knowledge discovery. In *Proceedings of the ACM SIGKDD Workshop on Visual Analytics and Knowledge Discovery: Integrating Automated Analysis with Interactive Exploration*, VAKD '09, pages 12–20, New York, NY, USA, 2009. ACM.

[57] E. Bertini, M. Rigamonti, and D. Lalanne. Extended excentric labeling. *Computer Graphics Forum*, 28(3):927–934, 2009.

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

[59] P. Bodesinsky, P. Federico, and S. Miksch. Visual analysis of compliance with clinical guidelines. In *Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies*, i-Know '13, pages 12:1–12:8, New York, NY, USA, 2013. ACM.

[60] P. Bonacich. Factoring and weighting approaches to status scores and clique identification. *The Journal of Mathematical Sociology*, 2(1):113–120, 1972.

[61] L. Borisjuk, M. R. Hajirezaei, C. Klukas, H. Rolletschek, and F. Schreiber. Integrating data from biological experiments into metabolic networks with the DBE information system. *In Silico Biol. (Gedrukt)*, 5(2):93–102, 2005.

[62] I. Boyandin, E. Bertini, and D. Lalanne. A qualitative study on the exploration of temporal changes in flow maps with animation and small-multiples. *Computer Graphics Forum*, 31(3pt2):1005–1014, 2012.

[63] D. Braha and Y. Bar-Yam. From centrality to temporary fame: Dynamic centrality in complex networks. *Complexity*, 12(2):59–63, 2006.

[64] U. Brandes and S. Corman. Visual unrolling of network evolution and the analysis of dynamic discourse. *Information Visualization*, 2(1):40 – 50, 2003.

[65] U. Brandes, D. Fleischer, and T. Puppe. Dynamic spectral layout of small worlds. In *Graph Drawing*, GD, pages 25–36. Springer, 2006.

[66] U. Brandes, M. Hoefer, and C. Pich. Affiliation dynamics with an application to movie-actor biographies. In *Proceedings of the 8th Joint Eurographics / IEEE VGTC Conference on Visualization*, EuroVis, pages 179–186. Eurographics Association, 2006.

[67] U. Brandes, N. Indlekofer, and M. Mader. Visualization methods for longitudinal social networks and stochastic actor oriented modeling. *Social Networks*, 34(3):291–308, 2012.

[68] U. Brandes and M. Mader. A quantitative comparison of stress-minimization approaches for offline dynamic graph drawing. In M. van Kreveld and B. Speckmann, editors, *Graph Drawing*, volume 7034 of *GD*, pages 99–110. Springer, Springer, 2012.

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

[70] S. Brasch, G. Fuellen, and L. Linsen. VENLO: Interactive visual exploration of aligned biological networks and their evolution. In *Visualization in Medicine and Life Sciences II*, pages 229–247. Springer, 2012.

[71] S. Brasch, L. Linsen, and G. Fuellen. Vanlo - interactive visual exploration of aligned biological networks. *BMC Bioinformatics*, 10(1):327, 2009.

[72] M. Brehmer, S. Carpendale, B. Lee, and M. Tory. Pre-design empiricism for information visualization: Scenarios, methods, and challenges. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, BELIV '14, pages 147–151, New York, NY, USA, 2014. ACM.

[73] P. Brodka, K. Skibicki, P. Kazienko, and K. Musial. A degree centrality in multi-layered social network. In *Int. Conf. on Computational Aspects of Social Networks (CASoN)*, pages 237 –242, 2011.

[74] F. Brodkorb, A. Kuijper, G. Andrienko, N. Andrienko, and T. von Landesberger. Overview with details for exploring geo-located graphs on maps. *Information Visualization*, 15(3):214–237, 2016.

[75] A. C. Brown. Xxxvii.-on the theory of isomeric compounds. *J. Chem. Soc.*, 18:230–245, 1865.

[76] M. Burch, F. Beck, and D. Weiskopf. Radial Edge Splatting for visualizing dynamic directed graphs. In *Proceedings of the 4th International Conference on Information Visualization Theory and Applications*, IVAPP, pages 603–612. SciTePress, 2012.

[77] M. Burch and S. Diehl. TimeRadarTrees: Visualizing dynamic compound digraphs. *Computer Graphics Forum*, 27(3):823–830, 2008.

[78] M. Burch, M. Fritz, F. Beck, and S. Diehl. TimeSpiderTrees: A novel visual metaphor for dynamic compound graphs. In *Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing*, VL/HCC, pages 168–175. IEEE, 2010.

[79] M. Burch, M. Höferlin, and D. Weiskopf. Layered TimeRadarTrees. In *Proceedings of the 15th International Conference on Information Visualisation*, IV, pages 18–25. IEEE, 2011.

[80] M. Burch, C. Müller, G. Reina, H. Schmauder, M. Greis, and D. Weiskopf. Visualizing dynamic call graphs. In *Vision, Modeling & Visualization*, VMV, pages 207–214. The Eurographics Association, 2012.

[81] M. Burch, T. Munz, and D. Weiskopf. Edge-stacked timelines for visualizing dynamic weighted digraphs. In *Proceedings of the International Conference on Information Visualization Theory and Applications*, IVAPP, 2015.

[82] M. Burch, B. Schmidt, and D. Weiskopf. A matrix-based visualization for exploring dynamic compound digraphs. In *Proceedings of the 17th International Conference Information Visualisation*, IV, pages 66–73. IEEE, 2013.

[83] M. Burch, C. Vehlow, F. Beck, S. Diehl, and D. Weiskopf. Parallel Edge Splatting for scalable dynamic graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2344–2353, 2011.

[84] M. Burch and D. Weiskopf. A flip-book of edge-splatted small multiples for visualizing dynamic graphs. In *Proceedings of the 7th International Symposium on Visual Information Communication and Interaction*, VINCI, pages 29–38. ACM, 2014.

[85] M. Burch and D. Weiskopf. Flip-book visualization of dynamic graphs. *International Journal of Software and Informatics*, 2015.

[86] S. K. Card, J. D. Mackinlay, and B. Shneiderman, editors. *Readings in Information Visualization: Using Vision to Think*. Morgan Kaufmann, San Francisco, CA, 1999.

[87] K. M. Carley. Dynamic network analysis. In *Dynamic social network modeling and analysis: Workshop summary and papers*. Comittee on Human Factors, National Research Council, 2003.

[88] A. Casteigts, P. Flocchini, W. Quattrociocchi, and N. Santoro. Time-varying graphs and dynamic networks. *International Journal of Parallel, Emergent and Distributed Systems*, 27(5):387–408, 2012.

[89] U. Cengiz Turker and S. Balcisoy. A visualization technique for large temporal social network datasets in hyperbolic space. *Journal of Visual Languages & Computing*, 25(3):227–242, 2013.

[90] R. Chang, C. Ziemkiewicz, T. M. Green, and W. Ribarsky. Defining insight for visual analytics. *IEEE Computer Graphics and Applications*, 29(2):14–17, March 2009.

[91] C. Chen. CiteSpace II: Detecting and visualizing emerging trends and transient patterns in scientific literature. *Journal of the American Society for Information Science and Technology*, 57(3):359–377, 2006.

[92] C. Chen and S. Morris. Visualizing evolving networks: Minimum spanning trees versus pathfinder networks. In *Proceedings of the 2003 IEEE Symposium on Information Visualization*, InfoVis, pages 67–74. IEEE, 2003.

[93] C. Chen and R. J. Paul. Visualizing a knowledge domain's intellectual structure. *Computer*, 34(3):65–71, Mar 2001.

[94] Y. Chiricota, F. Jourdan, and G. Melancon. Metric-based network exploration and multiscale scatterplot. In *IEEE Symposium on Information Visualization*, pages 135–142, 2004.

[95] A. Clauset, C. R. Shalizi, and M. E. J. Newman. Power-law distributions in empirical data. *SIAM Review*, 51(4):661–703, 2009.

[96] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 79(387):531–554, 1984.

[97] M. K. Coleman and D. S. Parker. Aesthetics-based graph layout for human consumption. *Software: Practice and Experience*, 26(12):1415–1438, 1996.

[98] C. Collberg, S. G. Kobourov, J. Nagra, J. Pitts, and K. Wampler. A system for graph-based visualization of the evolution of software. In *Proceedings of the 2003 ACM Symposium on Software Visualization*, SoftVis, pages 77–86. ACM, 2003.

[99] C. Collins and S. Carpendale. VisLink: Revealing relationships amongst visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1192 –1199, nov.-dec. 2007.

[100] C. Correa, T. Crnovrsanin, and K. L. Ma. Visual reasoning about social networks using centrality sensitivity. *IEEE Transactions on Visualization and Computer Graphics*, 18(1):106–120, Jan 2012.

[101] W. Cui, X. Wang, S. Liu, N. H. Riche, T. M. Madhyastha, K. L. Ma, and B. Guo. Let it flow: A static method for exploring dynamic graphs. In *2014 IEEE Pacific Visualization Symposium*, pages 121–128, March 2014.

[102] R. E. Curtis, J. Xiang, A. Parikh, P. Kinnaird, and E. P. Xing. Enabling dynamic network analysis through visualization in TVNViewer. *BMC Bioinformatics*, 13(204):1–13, 2012.

[103] T. N. Dang, N. Pendar, and A. G. Forbes. Timearcs: Visualizing fluctuations in dynamic networks. *Computer Graphics Forum*, 35(3):61–69, 2016.

[104] R. Davidson and D. Harel. Drawing graphs nicely using simulated annealing. *ACM Trans. Graph.*, 15(4):301–331, Oct. 1996.

[105] G. Di Battista, P. Eades, R. Tamassia, and I. G. Tollis. Algorithms for drawing graphs: an annotated bibliography. *Computational Geometry*, 4(5):235 – 282, 1994.

[106] V. Di Donato, M. Patrignani, and C. Squarcella. Netfork: Mapping time to space in network visualization. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '16, pages 92–99, New York, NY, USA, 2016. ACM.

[107] J. Díaz, J. Petit, and M. Serna. A survey of graph layout problems. *ACM Comput. Surv.*, 34(3):313–356, Sept. 2002.

[108] S. Diehl and C. Görg. Graphs, they are changing – dynamic graph drawing for a sequence of graphs. In *Graph Drawing*, GD, pages 23–31. Springer-Verlag, 2002.

[109] S. Diehl, C. Görg, and A. Kerren. Preserving the mental map using foresighted layout. In *Proceedings of the 3rd Joint Eurographics–IEEE TCVG Conference on Visualization*, VisSym, pages 175–184. Eurographics Association, 2001.

[110] M. Dörk, S. Carpendale, and C. Williamson. Visualizing explicit and implicit relations of complex information spaces. *Information Visualization*, 11(1):5–21, 2012.

[111] S. M. Drucker, T. A. Galyean, and D. Zeltzer. Cinema: a system for procedural camera movements. In *Proceedings of the 1992 symposium on Interactive 3D graphics*, I3D '92, pages 67–70, 1992.

[112] T. Dwyer and P. Eades. Visualising a fund manager flow graph with columns and worms. In *Proceedings of the 6th International Conference on Information Visualisation*, IV, pages 147–152. IEEE, 2002.

[113] T. Dwyer and D. R. Gallagher. Visualising changes in fund manager holdings in two and a half-dimensions. *Information Visualization*, 3(4):227–244, 2004.

[114] T. Dwyer, S.-H. Hong, D. Koschützki, F. Schreiber, and K. Xu. Visual analysis of network centralities. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation - Volume 60*, APVis '06, pages 189–197, 2006.

[115] T. Dwyer, K. Marriott, and M. Wybrow. Dunnart: A constraint-based network diagram authoring tool. In I. G. Tollis and M. Patrignani, editors, *Graph Drawing: 16th International Symposium, GD 2008, Heraklion, Crete, Greece, September 21-24, 2008. Revised Papers*, pages 420–431. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

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

[117] P. Eades and M. L. Huang. Navigating clustered graphs using force-directed methods. *Journal of Graph Algorithms and Applications*, 4:157–181, 2000.

[118] P. Eades and W. Lai. Preserving the mental map of a diagram. In *Proc. of the Int. Conf. on Computational Graphics and Visualization Techniques (COMPUGRAPHICS)*, Compugraphics '91, pages 24–33. Elsevier, 1991.

[119] J. Ebert. A versatile data structure for edge-oriented graph algorithms. *Commun. ACM*, 30(6):513–519, June 1987.

[120] G. Ellis and A. Dix. An explorative analysis of user evaluation studies in information visualisation. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization*, BELIV '06, pages 1–7, New York, NY, USA, 2006. ACM.

[121] N. Elmqvist, T. N. Do, H. Goodell, N. Henry, and J. D. Fekete. Zame: Interactive large-scale graph visualization. In *2008 IEEE Pacific Visualization Symposium*, pages 215–222, March 2008.

[122] N. Elmqvist, P. Dragicevic, and J. D. Fekete. Color lens: Adaptive color scale optimization for visual exploration. *IEEE Transactions on Visualization and Computer Graphics*, 17(6):795–807, June 2011.

[123] N. Elmqvist and J. D. Fekete. Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines. *IEEE Transactions on Visualization and Computer Graphics*, 16(3):439–454, May 2010.

[124] N. Elmqvist and P. Tsigas. Causality visualization using animated growing polygons. In *Proceedings of the 2003 IEEE Symposium on Information Visualization*, InfoVis, pages 189–196. IEEE, 2003.

[125] N. Elmqvist and P. Tsigas. Citewiz: A tool for the visualization of scientific citation networks. *Information Visualization*, 6(3):215–232, 2007.

[126] R. Enikeev. The Internet Map. `http://internet-map.net`. (accessed on 21/04/2017).

[127] B. Ens, D. Rea, R. Shpaner, H. Hemmati, J. E. Young, and P. Irani. ChronoTwigger: A visual analytics tool for understanding source and test co-evolution. In *Proceedings of the 2nd IEEE Working Conference on Software Visualization*, VISSOFT, pages 117–126. IEEE, 2014.

[128] C. Erten, P. J. Harding, S. G. Kobourov, K. Wampler, and G. V. Yee. GraphAEL: Graph animations with evolving layouts. In *Graph Drawing*, GD, pages 98–110. Springer, 2004.

[129] C. Erten, S. G. Kobourov, V. Le, and A. Navabi. Simultaneous graph drawing: Layout algorithms and visualization schemes. In *Graph Drawing*, GD, pages 437–449. Springer, 2004.

[130] L. Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140, 1741.

[131] M. Farrugia, N. Hurley, and A. J. Quigley. Exploring temporal ego networks using small multiples and tree-ring layouts. In *Proceedings of the 4th International Conference on Advances in Computer-Human Interactions*, ACHI, pages 79–88, 2011.

[132] M. Farrugia and A. Quigley. Cell phone mini challenge: Node-link animation award animating multivariate dynamic social networks. In *2008 IEEE Symposium on Visual Analytics Science and Technology*, Oct 2008.

[133] M. Farrugia and A. Quigley. Effective temporal graph layout: A comparative study of animation versus static display methods. *Information Visualization*, 10(1):47–64, 2011.

[134] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth. From data mining to knowledge discovery in databases. *AI magazine*, 17(3):37, 1996.

[135] P. Federico. Visual analytics of dynamic social networks. In *Doctoral Colloquium of the IEEE Conference on Visualization*, 2011.

[136] P. Federico, W. Aigner, S. Miksch, J. Pfeffer, M. Smuc, F. Windhager, and L. Zenk. Viena: Visual enterprise network analytics. In K. Matkovic and G. Santucci, editors, *Poster Proceedings of the 3rd International Workshop on Visual Analytics (EuroVA)*, page 12. Eurographics, Eurographics, 2012.

[137] P. Federico, W. Aigner, S. Miksch, F. Windhager, and M. Smuc. Vertigo zoom: Combining relational and temporal perspectives on dynamic networks. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 437–440, New York, NY, USA, 2012. ACM.

[138] P. Federico, W. Aigner, S. Miksch, F. Windhager, and L. Zenk. A visual analytics approach to dynamic social networks. In *Proceedings of the 11th International Conference on Knowledge Management and Knowledge Technologies*, i-KNOW 11, pages 47:1–47:8, New York, NY, USA, 2011. ACM.

[139] P. Federico, W. Aigner, S. Miksch, F. Windhager, and L. Zenk. Visual analytics of dynamic networks - a case study. In *The Third International UKVAC Workshop on Visual Analytics (VAW)*, 2011.

[140] P. Federico, A. Amor-Amorós, and S. Miksch. A nested workflow model for visual analytics design and validation. In *Proceedings of the Sixth Workshop on Beyond Time and Errors on Novel Evaluation Methods for Visualization*, BELIV '16, pages 104–111, New York, NY, USA, 2016. ACM.

[141] P. Federico, F. Heimerl, S. Koch, and S. Miksch. A survey on visual approaches for analyzing scientific literature and patents. *IEEE Transactions on Visualization and Computer Graphics*, PP(99):1–1, 2016.

[142] P. Federico, S. Hoffmann, A. Rind, W. Aigner, and S. Miksch. Qualizon graphs: Space-efficient time-series visualization with qualitative abstractions. In *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces*, AVI '14, pages 273–280, New York, NY, USA, 2014. ACM.

[143] P. Federico and S. Miksch. Evaluation of two interaction techniques for visualization of dynamic graphs. In Y. Hu and M. Nöllenburg, editors, *Graph Drawing and Network Visualization: 24th International Symposium, GD 2016, Athens, Greece, September 19-21, 2016, Revised Selected Papers*, pages 557–571. Springer International Publishing, Cham, 2016.

[144] P. Federico, J. Pfeffer, W. Aigner, S. Miksch, and L. Zenk. Visual analysis of dynamic networks using change centrality. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 179–183, Aug 2012.

[145] P. Federico, J. Unger, A. Amor-Amorós, L. Sacchi, D. Klimov, and S. Miksch. Gnaeus: Utilizing Clinical Guidelines for Knowledge-assisted Visualisation of EHR Cohorts. In E. Bertini and J. C. Roberts, editors, *EuroVis Workshop on Visual Analytics (EuroVA)*. The Eurographics Association, 2015.

[146] P. Federico, F. Windhager, L. Zenk, and M. Smuc. Visual analytics of dynamic networks. In *XXX INSNA Sunbelt Social Networks Conference*. International Network for Social Network Analysis, 2010.

[147] K.-C. Feng, C. Wang, H.-W. Shen, and T.-Y. Lee. Coherent time-varying graph drawing with multifocus+context interaction. *IEEE Transactions on Visualization and Computer Graphics*, 18(8):1330–1342, 2012.

[148] D. Fisher and A. Sud. Animated, dynamic voronoi treemaps. In D. Auber, G. Melançon, T. Munzner, and D. Weiskopf, editors, *Poster Abstracts at Eurographics/ IEEE-VGTC Symposium on Visualization*. The Eurpgraphics Association, 2010.

[149] D. Forrester, S. G. Kobourov, A. Navabi, K. Wampler, and G. V. Yee. graphael: A system for generalized force-directed layouts. In *Graph drawing*, GD, pages 454–464. Springer, 2005.

[150] C. Forsell and M. Cooper. A guide to reporting scientific evaluation in visualization. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 608–611, New York, NY, USA, 2012. ACM.

[151] L. Freeman. Visualizing social networks. *Journal of social structure*, 1(1):4, 2000.

[152] L. C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, 1977.

[153] M. Freire, C. Plaisant, B. Shneiderman, and J. Golbeck. Manynets: An interface for multiple network analysis and visualization. In *Proceedings of the SIGCHI Conference*

on Human Factors in Computing Systems, CHI '10, pages 213–222, New York, NY, USA, 2010. ACM.

[154] C. Friedrich and P. Eades. The Marey graph animation tool demo. In *Graph Drawing*, GD, pages 396–406. Springer, 2001.

[155] C. Friedrich and P. Eades. Graph drawing in motion. *Journal of Graph Algorithms and Applications*, 6(3):353–370, 2002.

[156] C. Friedrich and M. Houle. Graph drawing in motion ii. In P. Mutzel, M. Jünger, and S. Leipert, editors, *Graph Drawing*, volume 2265 of *Lecture Notes in Computer Science*, pages 122–125. Springer Berlin / Heidelberg, 2002.

[157] Y. Frishman and A. Tal. Dynamic drawing of clustered graphs. In *Proceedings of the 2004 IEEE Symposium on Information Visualization*, InfoVis, pages 191–198. IEEE, 2004.

[158] Y. Frishman and A. Tal. MOVIS: A system for visualizing distributed mobile object environments. *Journal of Visual Languages & Computing*, 19(3):303–320, 2008.

[159] Y. Frishman and A. Tal. Online dynamic graph drawing. *IEEE Transactions on Visualization and Computer Graphics*, 14(4):727–740, 2008.

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

[161] X. Fu, S.-H. Hong, N. S. Nikolov, X. Shen, Y. Wu, and K. Xu. Visualization and analysis of email networks. In *Proceedings of the 6th International Asia-Pacific Symposium on Visualization*, APVis, pages 1–8. IEEE, 2007.

[162] M. Gaertler and D. Wagner. A hybrid model for drawing dynamic and evolving graphs. In *Graph Drawing*, GD, pages 189–200. Springer, 2006.

[163] E. R. Gansner, Y. Hu, and S. North. Visualizing streaming text data with dynamic graphs and maps. In W. Didimo and M. Patrignani, editors, *Graph Drawing: 20th International Symposium, GD 2012, Redmond, WA, USA, September 19-21, 2012, Revised Selected Papers*, pages 439–450, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[164] E. R. Gansner, Y. Hu, and S. C. North. Interactive visualization of streaming text data with dynamic maps. *Journal of Graph Algorithms and Applications*, 17(4):515–540, 2013.

[165] E. R. Gansner and Y. Koren. Improved circular layouts. In *Proceedings of the 14th International Conference on Graph Drawing*, GD'06, pages 386–398, Berlin, Heidelberg, 2007. Springer-Verlag.

[166] E. R. Gansner, Y. Koren, and S. North. Graph drawing by stress majorization. In J. Pach, editor, *Graph Drawing: 12th International Symposium, GD 2004, New York, NY, USA, September 29-October 2, 2004, Revised Selected Papers*, pages 239–250. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[167] E. Garfield. Historiographic mapping of knowledge domains literature. *Journal of Information Science*, 30(2):119–145, 2004.

[168] S. Ghani, N. Elmqvist, and J. S. Yi. Perception of animated node-link diagrams for dynamic graphs. *Computer Graphics Forum*, 31(3pt3):1205–1214, 2012.

[169] M. Ghoniem, J.-D. Fekete, and P. Castagliola. 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.

[170] H. Gibson, J. Faith, and P. Vickers. A survey of two-dimensional graph layout techniques for information visualisation. *Information Visualization*, 12(3-4):324–357, 2013.

[171] F. Gilbert, P. Simonetto, F. Zaidi, F. Jourdan, and R. Bourqui. Communities and hierarchical structures in dynamic social networks: analysis and visualization. *Social Network Analysis and Mining*, 1(2):83–95, 2011.

[172] C. Görg, P. Birke, M. Pohl, and S. Diehl. Dynamic graph drawing of sequences of orthogonal and hierarchical graphs. In *Graph Drawing*, GD, pages 228–238. Springer, 2005.

[173] C. Görg, M. Pohl, E. Qeli, and K. Xu. Visual representations. In A. Kerren, A. Ebert, and J. Meyer, editors, *Human-Centered Visualization Environments: GI-Dagstuhl Research Seminar, Dagstuhl Castle, Germany, March 5-8, 2006, Revised Lectures*, pages 163–230. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[174] T. E. Gorochowski, M. di Bernardo, and C. S. Grierson. Using aging to visually uncover evolutionary processes on networks. *IEEE Transactions on Visualization and Computer Graphics*, 18(8):1343–1352, 2012.

[175] R. Gove, N. Gramsky, R. Kirby, E. Sefer, A. Sopan, C. Dunne, B. Shneiderman, and M. Taieb-Maimon. Netvisia: Heat map and matrix visualization of dynamic social network statistics and content. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pages 19–26, Oct 2011.

[176] P. A. Grabowicz, L. M. Aiello, and F. Menczer. Fast filtering and animation of large dynamic networks. *EPJ Data Science*, 3(1):1–16, 2014.

[177] O. Greevy, M. Lanza, and C. Wysseier. Visualizing live software systems in 3D. In *Proceedings of the 2006 ACM Symposium on Software Visualization*, SoftVis, pages 47–56. ACM, 2006.

[178] M. Greilich, M. Burch, and S. Diehl. Visualizing the evolution of compound digraphs with TimeArcTrees. *Computer Graphics Forum*, 28(3):975–982, 2009.

[179] G. Groh, H. Hanstein, and W. Wörndl. Interactively visualizing dynamic social networks with DySoN. In *Proceedings of the 2009 Workshop on Visual Interfaces to the Social and the Semantic Web*, VISSW, 2009.

[180] T. Gschwandtner, M. Bögl, P. Federico, and S. Miksch. Visual encodings of temporal uncertainty: A comparative user study. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):539–548, Jan 2016.

[181] S. Hachul and M. Jünger. Drawing large graphs with a potential-field-based multilevel algorithm. In J. Pach, editor, *Graph Drawing: 12th International Symposium, GD 2004, New York, NY, USA, September 29-October 2, 2004, Revised Selected Papers*, pages 285–295. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[182] S. Hadlak, H.-J. Schulz, and H. Schumann. In situ exploration of large dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2334–2343, dec. 2011.

[183] S. Hadlak, H. Schumann, C. H. Cap, and T. Wollenberg. Supporting the visual analysis of dynamic networks by clustering associated temporal attributes. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2267–2276, Dec 2013.

[184] S. Hadlak, H. Schumann, and H.-J. Schulz. A survey of multi-faceted graph visualization. In *Eurographics Conference on Visualization (EuroVis) - STARs*, EuroVis. The Eurographics Association, 2015.

[185] F. Harary. *Graph theory*. Addison-Wesley, Reading, MA, 1969.

[186] F. Harary and G. Gupta. Dynamic graph models. *Mathematical and Computer Modelling*, 25(7):79 – 87, 1997.

[187] D. Harel and Y. Koren. A fast multi-scale method for drawing large graphs. In J. Marks, editor, *Graph Drawing: 8th International Symposium, GD 2000 Colonial Williamsburg, VA, USA, September 20–23, 2000 Proceedings*, pages 183–196. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[188] S. Haroz, R. Kosara, and S. L. Franconeri. The connected scatterplot for presenting paired time series. *IEEE Transactions on Visualization and Computer Graphics*, 22(9):2174–2186, Sept 2016.

[189] A. Hayashi, T. Matsubayashi, T. Hoshide, and T. Uchiyama. Initial positioning method for online and real-time dynamic graph drawing of time varying data. In *Proceedings of the 17th International Conference on Information Visualisation*, IV, pages 435–444. IEEE, 2013.

[190] J. Heer and D. Boyd. Vizster: visualizing online social networks. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pages 32–39, Oct 2005.

[191] J. Heer, S. K. Card, and J. A. Landay. Prefuse: A toolkit for interactive information visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pages 421–430, New York, NY, USA, 2005. ACM.

[192] N. Henry and J.-D. Fekete. Matrixexplorer: A dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):677–684, Sept. 2006.

[193] N. Henry and J.-D. Fekete. Matlink: Enhanced matrix visualization for analyzing social networks. In C. Baranauskas, P. Palanque, J. Abascal, and S. D. J. Barbosa, editors, *Human-Computer Interaction – INTERACT 2007: 11th IFIP TC 13 International Conference, Rio de Janeiro, Brazil, September 10-14, 2007, Proceedings, Part II*, pages 288–302. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[194] N. Henry, J. D. Fekete, and M. J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, Nov 2007.

[195] I. Herman, G. Melançon, and M. S. Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):24–43, Jan. 2000.

[196] M. Hlawatsch, M. Burch, and D. Weiskopf. Visual adjacency lists for dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, pages 1590–1603, 2014.

[197] P. Hoek. Parallel arc diagrams: Visualizing temporal interactions. *Journal of Social Structure*, 12, 2011.

[198] P. Holme and J. Saramäki. Temporal networks. *Physics Reports*, 519(3):97 – 125, 2012.

[199] D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, Sept 2006.

[200] D. Holten, P. Isenberg, J. van Wijk, and J. Fekete. An extended evaluation of the readability of tapered, animated, and textured directed-edge representations in node-link graphs. In *Proc. IEEE Pacific Visualization Symp. (PacificVis)*, pages 195–202, March 2011.

[201] D. Holten and J. J. Van Wijk. Force-directed edge bundling for graph visualization. *Computer Graphics Forum*, 28(3):983–990, 2009.

[202] D. Holten and J. J. van Wijk. A user study on visualizing directed edges in graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 2299–2308, New York, NY, USA, 2009. ACM.

[203] K. Holtzblatt and S. Jones. Contextual inquiry: A participatory technique for system design. In D. Schuler and A. Namioka, editors, *Participatory design: Principles and practices*, pages 177–210. Lawrence Erlbaum Associates, Hillsdale, NJ, USA, 1993.

[204] Y. Hu. Efficient, high-quality force-directed graph drawing. *mathematica Journal*, 10(1):37–71, 2005.

[205] Y. Hu, E. R. Gansner, and S. Kobourov. Visualizing graphs and clusters as maps. *IEEE Computer Graphics and Applications*, 30(6):54–66, Nov 2010.

[206] Y. Hu, S. G. Kobourov, and S. Veeramoni. Embedding, clustering and coloring for dynamic maps. In *Proceedings of the 2012 IEEE Pacific Visualization Symposium*, PacificVis, pages 33–40. IEEE Computer Society, 2012.

[207] M. L. Huang, P. Eades, and J. Wang. On-line animated visualization of huge graphs using a modified spring algorithm. *Journal of Visual Languages & Computing*, 9(6):623–645, 1998.

[208] W. Huang, P. Eades, and S.-H. Hong. Measuring effectiveness of graph visualizations: A cognitive load perspective. *Information Visualization*, 8(3):139–152, June 2009.

[209] W. Huang, M. L. Huang, and C.-C. Lin. Evaluating overall quality of graph visualizations based on aesthetics aggregation. *Information Sciences*, 330:444 – 454, 2016. SI Visual Info Communication.

[210] W. Huang, M. Zhu, M. L. Huang, and H. B.-L. Duh. Evaluating overall quality of dynamic network visualizations. In Y. Luo, editor, *Cooperative Design, Visualization, and Engineering: 13th International Conference, CDVE 2016, Sydney, NSW, Australia, October 24–27, 2016, Proceedings*, pages 157–162. Springer International Publishing, Cham, 2016.

[211] S. Huron, R. Vuillemot, and J. D. Fekete. Visual sedimentation. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2446–2455, Dec 2013.

[212] M. Hurst. Mapping the Blogosphere. `http://datamining.typepad.com/gallery/blog-map-gallery.html`. (accessed on 21/04/2017).

[213] C. Hurter, O. Ersoy, S. I. Fabrikant, T. Klein, and A. Telea. Bundled visualization of dynamic graph and trail data. *IEEE Transactions on Visualization and Computer Graphics*, 2013.

[214] T. Isenberg, P. Isenberg, J. Chen, M. Sedlmair, and T. Möller. A systematic review on the practice of evaluating visualization. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2818–2827, Dec 2013.

[215] M. Itoh and M. Akaishi. Visualizing for changes in relationships between historical figures in chronicles. In *Proceedings of the 16th International Conference on Information Visualisation*, IV, pages 283–290. IEEE, 2012.

[216] M. Itoh, M. Toyoda, and M. Kitsuregawa. An interactive visualization framework for time-series of web graphs in a 3D environment. In *Proceedings of the 14th International Conference on Information Visualisation*, IV, pages 54–60. IEEE, 2010.

[217] M. Itoh, N. Yoshinaga, M. Toyoda, and M. Kitsuregawa. Analysis and visualization of temporal changes in bloggers' activities and interests. In *Proceeding of the 2012 IEEE Pacific Visualization Symposium*, PacificVis, pages 57–64. IEEE, 2012.

[218] P. Jaccard. The distribution of flora in the alpine zone. *The New Phytologist*, 11(2):37–50, 1912.

[219] R. Jianu, A. Rusu, Y. Hu, and D. Taggart. How to display group information on node-link diagrams: An evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 20(11):1530–1541, Nov 2014.

[220] M. John, H.-J. Schulz, H. Schumann, A. M. Uhrmacher, and A. Unger. Constructing and visualizing chemical reaction networks from pi-calculus models. *Formal Aspects of Computing*, 25(5):723–742, 2013.

[221] B. Johnson and B. Shneiderman. Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the 2Nd Conference on Visualization '91*, VIS '91, pages 284–291, Los Alamitos, CA, USA, 1991. IEEE Computer Society Press.

[222] S. Kairam, D. MacLean, M. Savva, and J. Heer. Graphprism: Compact visualization of network structure. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 498–505, New York, NY, USA, 2012. ACM.

[223] T. Kamada and S. Kawai. An algorithm for drawing general undirected graphs. *Information Processing Letters*, 31(1):7 – 15, 1989.

[224] H. Kang, C. Plaisant, B. Lee, and B. B. Bederson. Netlens: Iterative exploration of content-actor network data. *Information Visualization*, 6(1):18–31, 2007.

[225] D. Keim, F. Mansmann, J. Schneidewind, and H. Ziegler. Challenges in visual data analysis. In *Int. Conf. on Information Visualization*, 2006.

[226] D. A. Keim. Information visualization and visual data mining. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):1–8, Jan 2002.

[227] D. A. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann, editors. *Mastering the Information Age. Solving problems with Visual Analytics.* The Eurographics Association, 2010.

[228] D. A. Keim, F. Mansmann, J. Schneidewind, J. Thomas, and H. Ziegler. Visual analytics: Scope and challenges. In S. J. Simoff, M. H. Böhlen, and A. Mazeika, editors, *Visual Data Mining: Theory, Techniques and Tools for Visual Analytics*, pages 76–90. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[229] D. A. Keim, F. Mansmann, A. Stoffel, and H. Ziegler. Visual analytics. In L. LIU and M. T. ÖZSU, editors, *Encyclopedia of Database Systems*, pages 3341–3346. Springer US, Boston, MA, 2009.

[230] R. Keller, C. M. Eckert, and P. J. Clarkson. Matrices or node-link diagrams: Which visual representation is better for visualising connectivity models? *Information Visualization*, 5(1):62–76, 2006.

[231] N. Kerracher, J. Kennedy, and K. Chalmers. The design space of temporal graph visualisation. In N. Elmqvist, M. Hlawitschka, and J. Kennedy, editors, *EuroVis - Short Papers*, pages 7–11. Eurographics Assocation, 2014.

[232] N. Kerracher, J. Kennedy, and K. Chalmers. A task taxonomy for temporal graph visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 2015.

[233] A. Kerren, H. Purchase, and M. Ward, editors. *Multivariate Network Visualization*, volume 8380 of *LNCS*. Springer, 2014.

[234] U. Khurana and A. Deshpande. Hinge: Enabling temporal network analytics at scale. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, SIGMOD '13, pages 1089–1092, New York, NY, USA, 2013. ACM.

[235] U. Khurana, V. A. Nguyen, H. C. Cheng, J. w. Ahn, X. Chen, and B. Shneiderman. Visual analysis of temporal trends in social networks using edge color coding and metric timelines. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pages 549–554, Oct 2011.

[236] G. Kirchhoff. Ueber den durchgang eines elektrischen stromes durch eine ebene, insbesondere durch eine kreisförmige. *Annalen der Physik*, 140(4):497–514, 1845.

[237] S. G. Kobourov. Force-directed drawing algorithms. In *Handbook of Graph Drawing and Visualization*, pages 383–408. CRC, 2013.

[238] S. G. Kobourov. Force-directed drawing algorithms. In R. Tamassia, editor, *Handbook of graph drawing and visualization*. CRC Press, Taylor & Francis Group, Boca Raton, 2014.

[239] S. G. Kobourov, T. Mchedlidze, and L. Vonessen. Gestalt principles in graph drawing. In *Revised Selected Papers of the 23rd International Symposium on Graph Drawing and Network Visualization - Volume 9411*, GD 2015, pages 558–560, New York, NY, USA, 2015. Springer-Verlag New York, Inc.

[240] S. G. Kobourov and K. Wampler. Non-eeuclidean spring embedders. *IEEE Transactions on Visualization and Computer Graphics*, 11(6):757–767, 2005.

[241] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43(1):59–69, 1982.

[242] B. Kondo, H. Mehta, and C. Collins. Glidgets: Interactive glyphs for exploring dynamic graphs. In *Poster Proc. of IEEE Conf. on Information Visualization (InfoVis)*, 2014.

[243] Y. Koren. On spectral graph drawing. In T. Warnow and B. Zhu, editors, *Computing and Combinatorics: 9th Annual International Conference, COCOON 2003 Big Sky, MT, USA, July 25–28, 2003 Proceedings*, pages 496–508. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[244] D. Koschützki and F. Schreiber. Comparison of centralities for biological networks. In R. Giegerich and J. Stoye, editors, *Proceedings of the German Conference on Bioinformatics (GCB'04)*, volume P53 of *LNI*, pages 199–206. Gesellschaft für Informatik, 2004.

[245] E. Kruja, J. Marks, A. Blair, and R. Waters. A short note on the history of graph drawing. In P. Mutzel, M. Jünger, and S. Leipert, editors, *Graph Drawing: 9th International Symposium, GD 2001 Vienna, Austria, September 23–26, 2001 Revised Papers*, pages 272–286, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[246] M. Krzywinski, I. Birol, S. J. Jones, and M. A. Marra. Hive plots—rational approach to visualizing networks. *Briefings in Bioinformatics*, 13(5):627, 2012.

[247] O. Kulyk, R. Kosara, J. Urquiza, and I. Wassink. Human-centered aspects. In A. Kerren, A. Ebert, and J. Meyer, editors, *Human-Centered Visualization Environments: GI-Dagstuhl Research Seminar, Dagstuhl Castle, Germany, March 5-8, 2006, Revised Lectures*, pages 13–75. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[248] G. Kumar and M. Garland. Visual exploration of complex time-varying graphs. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):805–812, 2006.

[249] M. Lad, D. Massey, and L. Zhang. Visualizing internet routing changes. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1450–1460, 2006.

[250] H. Lam. A framework of interaction costs in information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1149–1156, Nov 2008.

[251] 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, Sept 2012.

[252] H. J. Leavitt. Some effects of certain communication patterns on group performance. *The Journal of Abnormal and Social Psychology*, 46(1):38, 1951.

[253] B. Lee, C. Plaisant, C. S. Parr, J.-D. Fekete, and N. Henry. Task taxonomy for graph visualization. In *Proc. AVI Workshop on Beyond Time and Errors: Novel Evaluation Methods for Information Visualization*, BELIV '06, pages 1–5, New York, NY, 2006. ACM.

[254] Y.-Y. Lee, C.-C. Lin, and H.-C. Yen. Mental map preserving graph drawing using simulated annealing. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation*, APVis, pages 179–188. Australian Computer Society, Inc., 2006.

[255] W. W. Leontief et al. *Structure of American economy, 1919-1929*. Harvard University Press, 1941.

[256] K. Lerman, R. Ghosh, and J. H. Kang. Centrality metric for dynamic networks. In *Proc. of the Workshop on Mining and Learning with Graphs (MLG)*, pages 70–77, New York, NY, USA, 2010. ACM.

[257] Q. Liao, L. Shi, and C. Wang. Visual analysis of large-scale network anomalies. *IBM Journal of Research and Development*, 57(3/4):13:1–13:12, 2013.

[258] Y.-R. Lin, J. Sun, N. Cao, and S. Liu. ContexTour: Contextual contour visual analysis on dynamic multi-relational clustering. In *Proceedings of the 2010 SIAM International Conference on Data Mining*, SDM, pages 418–429, 2010.

[259] H. Liu, P. Eades, and S. H. Hong. Visualizing dynamic trajectories in social networks. In *2012 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, pages 197–204, Sept 2012.

[260] Q. Liu, Y. Hu, L. Shi, X. Mu, Y. Zhang, and J. Tang. Egonetcloud: Event-based egocentric dynamic network visualization. In *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 65–72, Oct 2015.

[261] X. Liu and H.-W. Shen. The effects of representation and juxtaposition on graphical perception of matrix visualization. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 269–278, New York, NY, USA, 2015. ACM.

[262] E. Loubier and B. Dousset. Temporal and relational data representation by graph morphing. *Safety and Reliability for Managing Risk*, 14(02):2008–16, 2008.

[263] C. Ma, R. V. Kenyon, A. G. Forbes, T. Berger-Wolf, B. J. Slater, and D. A. Llano. Visualizing Dynamic Brain Networks Using an Animated Dual-Representation. In *Eurographics Conference on Visualization (EuroVis) - Short Papers*, EuroVis. The Eurographics Association, 2015.

[264] A. Ma'ayan, S. L. Jenkins, R. L. Webb, S. I. Berger, S. P. Purushothaman, N. S. Abul-Husn, J. M. Posner, T. Flores, and R. Iyengar. Snavi: Desktop application for analysis and visualization of large-scale signaling networks. *BMC Systems Biology*, 3(1):10, 2009.

[265] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans. Graph.*, 5(2):110–141, Apr. 1986.

[266] G. Marchionini. Exploratory search: From finding to understanding. *Commun. ACM*, 49(4):41–46, Apr. 2006.

[267] D. Mashima, S. G. Kobourov, and Y. Hu. Visualizing dynamic data with maps. *IEEE Transactions on Visualization and Computer Graphics*, 18(9):1424–1437, 2012.

[268] J. Matejka, T. Grossman, and G. Fitzmaurice. Citeology: Visualizing paper genealogy. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '12, pages 181–190, New York, NY, USA, 2012. ACM.

[269] T. May, M. Steiger, J. Davey, and J. Kohlhammer. Using signposts for navigation in large graphs. *Computer Graphics Forum*, 31(3pt2):985–994, 2012.

[270] E. Mayr, M. Smuc, and H. Risku. Many roads lead to rome: Mapping users problem-solving strategies. *Information Visualization*, 10(3):232–247, 2011.

[271] B. McDonnel and N. Elmqvist. Towards utilizing gpus in information visualization: A model and implementation of image-space operations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1105–1112, Nov 2009.

[272] M. McGuffin and I. Jurisica. Interaction techniques for selecting and manipulating subgraphs in network visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):937–944, Nov 2009.

[273] M. Meyer, M. Sedlmair, P. S. Quinan, and T. Munzner. The nested blocks and guidelines model. *Information Visualization*, 14(3):234–249, 2015.

[274] S. Miksch and W. Aigner. A matter of time: Applying a data-users-tasks design triangle to visual analytics of time-oriented data. *Computers & Graphics*, 38:286 – 290, 2014.

[275] K. Misue, P. Eades, W. Lai, and K. Sugiyama. Layout adjustment and the mental map. *Journal of Visual Languages and Computing*, 6(2):183 – 210, 1995.

[276] J. Moreno. *Who shall survive?: A new approach to the problem of human interrelations*. Nervous and mental disease monograph series. Nervous and mental disease publishing co., Washington D.C., 1934.

[277] T. Moscovich, F. Chevalier, N. Henry, E. Pietriga, and J.-D. Fekete. Topology-aware navigation in large networks. In *Proc. SIGCHI Conf. on Human Factors in Computing Systems*, CHI '09, pages 2319–2328, New York, NY, 2009. ACM.

[278] C. W. Muelder, T. Crnovrsanin, A. Sallaberry, and K. L. Ma. Egocentric storylines for visual analysis of large dynamic graphs. In *2013 IEEE International Conference on Big Data*, pages 56–62, Oct 2013.

[279] T. Munzner. H3: Laying out large directed graphs in 3d hyperbolic space. In *Proceedings of the 1997 IEEE Symposium on Information Visualization (InfoVis '97)*, INFOVIS '97, pages 2–, Washington, DC, USA, 1997. IEEE Computer Society.

[280] T. Munzner. A nested model for visualization design and validation. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):921–928, Nov 2009.

[281] P. Mutzel, C. Gutwenger, R. Brockenauer, S. Fialko, G. Klau, M. Krüger, T. Ziegler, S. Näher, D. Alberts, D. Ambras, G. Koch, M. Jünger, C. Buchheim, and S. Leipert. A library of algorithms for graph drawing. In S. H. Whitesides, editor, *Graph Drawing: 6th International Symposium, GD' 98 Montréal, Canada, August 13–15, 1998 Proceedings*, pages 456–457. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[282] K. V. Nesbitt and C. Friedrich. Applying gestalt principles to animated visualizations of network data. In *Proceedings of the 6th International Conference on Information Visualisation*, IV, pages 737–743. IEEE, 2002.

[283] R. Netzel, M. Burch, and D. Weiskopf. Comparative eye tracking study on node-link visualizations of trajectories. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2221–2230, Dec 2014.

[284] A. Newell and J. C. Shaw. Programming the logic theory machine. In *Papers Presented at the February 26-28, 1957, Western Joint Computer Conference: Techniques for Reliability*, IRE-AIEE-ACM '57 (Western), pages 230–240, New York, NY, USA, 1957. ACM.

[285] M. Newman. *Networks: an introduction*. Oxford University Press, 2010.

[286] Q. Nguyen, P. Eades, and S. H. Hong. On the faithfulness of graph visualizations. In *2013 IEEE Pacific Visualization Symposium (PacificVis)*, pages 209–216, Feb 2013.

[287] A. Noack. Energy models for graph clustering. *J. Graph Algorithms Appl.*, 11(2):453–480, 2007.

[288] A. Noack. Modularity clustering is force-directed layout. *Phys. Rev. E*, 79:026102, Feb 2009.

[289] S. C. North. Incremental layout in DynaDAG. In *Graph Drawing*, GD, pages 409–418. Springer, 1996.

[290] D. Oelke, D. Kokkinakis, and D. A. Keim. Fingerprint Matrices: Uncovering the dynamics of social networks in prose literature. *Computer Graphics Forum*, 32(3pt4):371–380, 2013.

[291] M. Ogawa and K.-L. Ma. StarGate: A unified, interactive visualization of software projects. In *Proceedings of the IEEE Pacific Visualization Symposium*, PacificVis, pages 191–198. IEEE, 2008.

[292] M. Ogawa and K.-L. Ma. code_swarm: A design study in organic software visualization. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1097–1104, 2009.

[293] M. Oliveira and J. Gama. Visualization of evolving social networks using actor-level and community-level trajectories. *Expert Systems*, 30(4):306–319, 2013.

[294] S. E. Palmer. Visual perception of objects. In *Handbook of Psychology*. John Wiley & Sons, Inc., 2003.

[295] J. Pearlman and P. Rheingans. Visualizing network security events using compound glyphs from a service-oriented perspective. In J. R. Goodall, G. Conti, and K.-L. Ma, editors, *VizSEC 2007: Proceedings of the Workshop on Visualization for Computer Security*, pages 131–146. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[296] A. Perer. Using socialaction to uncover structure in social networks over time. In *2008 IEEE Symposium on Visual Analytics Science and Technology*, Oct 2008.

[297] A. Perer and B. Shneiderman. Balancing systematic and flexible exploration of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):693–700, Sept 2006.

[298] A. Perer and B. Shneiderman. Integrating statistics and visualization: case studies of gaining clarity during exploratory data analysis. In *Proceedings of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, CHI 08, pages 265–274, New York, NY, USA, 2008. ACM.

[299] A. Perer and J. Sun. MatrixFlow: Temporal network visual analytics to track symptom evolution during disease progression. In *AMIA Annual Symposium Proceedings*, volume 2012 of *AMIA*, page 716. American Medical Informatics Association, 2012.

[300] D. J. Peuquet. It's about time: A conceptual framework for the representation of temporal dynamics in geographic information systems. *Annals of the Association of American Geographers*, 84(3):441–461, 1994.

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

[302] B. Pinaud, G. Melançon, and J. Dubois. Porgy: A visual graph rewriting environment for complex systems. *Computer Graphics Forum*, 31(3pt4):1265–1274, 2012.

[303] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of the International Conference on Intelligence Analysis*, volume 5, pages 2–4, 2005.

[304] C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. Lifelines: Visualizing personal histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '96, pages 221–227, New York, NY, USA, 1996. ACM.

[305] M. Pohl and P. Birke. Interactive exploration of large dynamic networks. In *Proceedings of the 10th International Conference on Visual Information Systems*, VISUAL, pages 56–67. Springer-Verlag, 2008.

[306] M. Pohl and S. Diehl. What dynamic network metrics can tell us about developer roles. In *Proc. of the int. workshop on Cooperative and human aspects of software engineering (CHASE)*, pages 81–84, New York, NY, USA, 2008. ACM.

[307] M. Pohl, F. Reitz, and P. Birke. As time goes by: integrated visualization and analysis of dynamic networks. In *Proc. of the working conf. on Advanced visual interfaces (AVI)*, pages 372–375, New York, NY, USA, 2008. ACM.

[308] A. Pretorius, H. Purchase, and J. Stasko. Tasks for multivariate network analysis. In Kerren et al. [233], pages 77–95.

[309] A. Pretorius and J. van Wijk. Visual analysis of multivariate state transition graphs. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):685–692, Sept 2006.

[310] D. D. S. Price. A general theory of bibliometric and other cumulative advantage processes. *Journal of the American Society for Information Science*, 27(5):292–306, 1976.

[311] S. Pupyrev and A. Tikhonov. Analyzing conversations with dynamic graph visualization. In *Proceedings of the 10th International Conference on Intelligent Systems Design and Applications*, ISDA, pages 748–753. IEEE, 2010.

[312] H. Purchase, E. Hoggan, and C. Görg. How important is the mental map? An empirical investigation of a dynamic graph layout algorithm. In M. Kaufmann and D. Wagner, editors, *Graph Drawing*, volume 4372 of *LNCS*, pages 184–195. Springer, Berlin, Heidelberg, 2007.

[313] H. Purchase and A. Samra. Extremes are better: Investigating mental map preservation in dynamic graphs. In G. Stapleton, J. Howse, and J. Lee, editors, *Diagrammatic Representation and Inference*, volume 5223 of *LNCS*, pages 60–73. Springer, 2008.

[314] H. C. Purchase. The effects of graph layout. In *Computer Human Interaction Conference, 1998. Proceedings. 1998 Australasian*, pages 80–86. IEEE, 1998.

[315] H. C. Purchase, R. F. Cohen, and M. James. Validating graph drawing aesthetics. In F. J. Brandenburg, editor, *Graph Drawing: Symposium on Graph Drawing, GD '95 Passau, Germany, September 20–22, 1995 Proceedings*, pages 435–446. Springer Berlin Heidelberg, Berlin, Heidelberg, 1996.

[316] J. Qi and Y. Ohsawa. Blocks: Efficient and stable online visualization of dynamic network evolution. *The Review of Socionetwork Strategies*, 10(1):33–51, 2016.

[317] A. Quigley and P. Eades. Fade: Graph drawing, clustering, and visual abstraction. In J. Marks, editor, *Graph Drawing: 8th International Symposium, GD 2000 Colonial Williamsburg, VA, USA, September 20–23, 2000 Proceedings*, pages 197–210. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[318] K. Reda, C. Tantipathananandh, A. Johnson, J. Leigh, and T. Berger-Wolf. Visualizing the evolution of community structures in dynamic social networks. *Computer Graphics Forum*, 30(3):1061–1070, 2011.

[319] F. Reitz. A framework for an ego-centered and time-aware visualization of relations in arbitrary data repositories. *arXiv preprint arXiv:1009.5183*, 2010.

[320] F. Reitz, M. Pohl, and S. Diehl. Focused animation of dynamic compound digraphs. In *Proceedings of the 13th International Conference on Information Visualisation*, IV, pages 679–684. IEEE, 2009.

[321] B. Renoust, G. Melançon, and T. Munzner. Detangler: Visual analytics for multiplex networks. *Computer Graphics Forum*, 34(3):321–330, 2015.

[322] G. D. Rey and S. Diehl. Controlling presentation speed, labels, and tooltips in interactive dynamic graphs. *Journal of Media Psychology: Theories, Methods, and Applications*, 22(4):160–170, 2010.

[323] N. H. Riche, T. Dwyer, B. Lee, and S. Carpendale. Exploring the design space of interactive link curvature in network diagrams. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '12, pages 506–513, New York, NY, USA, 2012. ACM.

[324] N. H. Riche, Y. Riche, N. Roussel, S. Carpendale, T. Madhyastha, and T. J. Grabowski. Linkwave: A visual adjacency list for dynamic weighted networks. In *Proceedings of the 26th Conference on L'Interaction Homme-Machine*, IHM '14, pages 113–122, New York, NY, USA, 2014. ACM.

[325] A. Rind, P. Federico, T. Gschwandtner, W. Aigner, J. Doppler, and M. Wagner. Visual analytics of electronic health records with a focus on time. In G. Rinaldi, editor, *New Perspectives in Medical Records: Meeting the Needs of Patients and Practitioners*, pages 65–77. Springer International Publishing, Cham, 2017.

[326] J. Robertson. *Modern statistical methods for HCI*. Springer, Switzerland, 2016.

[327] M. Rohrschneider, A. Ullrich, A. Kerren, P. F. Stadler, and G. Scheuermann. Visual network analysis of dynamic metabolic pathways. In *Advances in Visual Computing*, ISVC, pages 316–327. Springer, 2010.

[328] M. Rosvall and C. T. Bergstrom. Mapping Change in Large Networks. *PLoS ONE*, 5(1):e8694, 01 2010.

[329] M. Roughan and J. Tuke. The hitchhikers guide to sharing graph data. In *2015 3rd International Conference on Future Internet of Things and Cloud*, pages 435–442, Aug 2015.

[330] M. Roughan and S. J. Tuke. Unravelling graph-exchange file formats. *CoRR*, abs/1503.02781, 2015.

[331] S. Rufiange and C. P. Fuhrman. Visualizing protected variations in evolving software designs. *Journal of Systems and Software*, 88(0):231–249, 2014.

[332] S. Rufiange and M. J. McGuffin. DiffAni: Visualizing dynamic graphs with a hybrid of difference maps and animation. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2556–2565, 2013.

[333] S. Rufiange, M. J. McGuffin, and C. P. Fuhrman. Treematrix: A hybrid visualization of compound graphs. *Computer Graphics Forum*, 31(1):89–101, 2012.

[334] S. Rufiange and G. Melançon. AniMatrix: A matrix-based visualization of software evolution. In *Proceedings of the 2nd IEEE Working Conference on Software Visualization*, VISSOFT, pages 137–146, 2014.

[335] D. Sacha, A. Stoffel, F. Stoffel, B. C. Kwon, G. Ellis, and D. A. Keim. Knowledge generation model for visual analytics. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1604–1613, Dec 2014.

[336] P. Saffrey and H. Purchase. The "mental map" versus "static aesthetic" compromise in dynamic graphs: a user study. In *Proceedings of the ninth conference on Australasian user interface - Volume 76*, AUIC 08, pages 85–93, Darlinghurst, Australia, Australia, 2008. Australian Computer Society, Inc.

[337] B. Saket, P. Simonetto, S. Kobourov, and K. Borner. Node, node-link, and node-link-group diagrams: An evaluation. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2231–2240, Dec 2014.

[338] A. Sallaberry, C. Muelder, and K.-L. Ma. Clustering, visualizing, and navigating for large dynamic graphs. In *Graph Drawing*, GD, pages 487–498. Springer, 2012.

[339] P. Saraiya, P. Lee, and C. North. Visualization of graphs with associated timeseries data. In *Proceedings of the 2005 IEEE Symposium on Information Visualization*, InfoVis, pages 225–232. IEEE Computer Society, 2005.

[340] H. Schmauder, M. Burch, and D. Weiskopf. Visualizing dynamic weighted digraphs with partial links. In *Proceedings of the International Conference on Information Visualization Theory and Application*, IVAPP, 2015.

[341] S. Schöffel, J. Schwank, and A. Ebert. A user study on multivariate edge visualizations for graph-based visual analysis tasks. In *2016 20th International Conference Information Visualisation (IV)*, pages 165–170, July 2016.

[342] H. J. Schulz. Treevis.net: A tree visualization reference. *IEEE Computer Graphics and Applications*, 31(6):11–15, Nov 2011.

[343] H. J. Schulz, S. Hadlak, and H. Schumann. The design space of implicit hierarchy visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):393–411, April 2011.

[344] H.-J. Schulz and C. Hurter. Grooming the hairball-how to tidy up network visualizations? In *INFOVIS 2013, IEEE Information Visualization Conference*, 2013.

[345] M. Sedlmair, M. Meyer, and T. Munzner. Design study methodology: Reflections from the trenches and the stacks. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2431–2440, Dec 2012.

[346] J. R. Seeley. The net of reciprocal influence; a problem in treating sociometric data. *Canadian Journal of Psychology/Revue canadienne de psychologie*, 3(4):234, 1949.

[347] D. Selassie, B. Heller, and J. Heer. Divided edge bundling for directional network data. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2354–2363, Dec 2011.

[348] L. Shi, C. Wang, and Z. Wen. Dynamic network visualization in 1.5D. In *Proceedings of the 2011 IEEE Pacific Visualization Symposium*, PacificVis, pages 179–186. IEEE, 2011.

[349] L. Shi, C. Wang, Z. Wen, H. Qu, C. Lin, and Q. Liao. 1.5D egocentric dynamic network visualization. *IEEE Transactions on Visualization and Computer Graphics*, 21:624–637, 2015.

[350] B. Shneiderman. Direct manipulation: A step beyond programming languages (abstract only). In *Proceedings of the Joint Conference on Easier and More Productive Use of Computer Systems. (Part - II): Human Interface and the User Interface - Volume 1981*, CHI '81, pages 143–, New York, NY, USA, 1981. ACM.

[351] B. Shneiderman. The eyes have it: a task by data type taxonomy for information visualizations. In *Proceedings 1996 IEEE Symposium on Visual Languages*, pages 336–343, Sep 1996.

[352] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, Sept 2006.

[353] D. K. Smith. *Networks and graphs: techniques and computational methods*. Elsevier, 2003.

[354] M. Smuc, P. Federico, F. Windhager, W. Aigner, L. Zenk, and S. Miksch. How do you connect moving dots? insights from user studies on dynamic network visualizations. In W. Huang, editor, *Handbook of Human Centric Visualization*, pages 623–650. Springer, 2014.

[355] R. Spence. *Information visualization : design for interaction*. Addison Wesley, Harlow, England New York, 2007.

[356] R. R. Springmeyer, M. M. Blattner, and N. L. Max. A characterization of the scientific data analysis process. In *Proceedings of the 3rd Conference on Visualization '92*, VIS '92, pages 235–242, Los Alamitos, CA, USA, 1992. IEEE Computer Society Press.

[357] M. Steiger, J. Bernard, S. Mittelstädt, H. Lücke-Tieke, D. Keim, T. May, and J. Kohlhammer. Visual analysis of time-series similarities for anomaly detection in sensor networks. *Computer Graphics Forum*, 33(3):401–410, 2014.

[358] K. Stein, R. Wegener, and C. Schlieder. Pixel-oriented visualization of change in social networks. In *Proceedings of the 2010 International Conference on Advances in Social Networks Analysis and Mining*, ASONAM, pages 233–240. IEEE, 2010.

[359] F. Steinicke, G. Bruder, and S. Kuhl. Realistic perspective projections for virtual objects and environments. *ACM Trans. Graph.*, 30:112:1–112:10, 2011.

[360] J. G. Stell. Granulation for graphs. In C. Freksa and D. M. Mark, editors, *Spatial Information Theory. Cognitive and Computational Foundations of Geographic Information Science: International Conference COSIT'99 Stade, Germany, August 25–29, 1999 Proceedings*, pages 417–432. Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[361] R. Sternberg. *Cognitive psychology*. Wadsworth/Cengage Learning, Belmont, CA, 2012.

[362] H. Stitz, S. Gratzl, M. Krieger, and M. Streit. Cloudgazer: A divide-and-conquer approach to monitoring and optimizing cloud-based networks. In *2015 IEEE Pacific Visualization Symposium (PacificVis)*, pages 175–182, April 2015.

[363] D. Stone, C. Jarrett, M. Woodroffe, and S. Minocha. *User Interface Design and Evaluation (Interactive Technologies)*. Morgan Kaufmann, 2005.

[364] E. Tarameshloo, M. H. Loorak, P. W. Fong, and S. Carpendale. Using visualization to explore original and anonymized lbsn data. *Computer Graphics Forum*, 35(3):291–300, 2016.

[365] A. Telea and D. Auber. Code flows: Visualizing structural evolution of source code. *Computer Graphics Forum*, 27(3):831–838, 2008.

[366] K. Thiel, F. Dill, T. Kotter, and M. R. Berthold. Towards visual exploration of topic shifts. In *2007 IEEE International Conference on Systems, Man and Cybernetics*, pages 522–527, Oct 2007.

[367] J. J. Thomas and K. A. Cook, editors. *Illuminating the path: the research and development agenda for visual analytics*. IEEE Computer Society, 2005.

[368] C. Tominski, J. Abello, F. van Ham, and H. Schumann. Fisheye tree views and lenses for graph visualization. In *Tenth International Conference on Information Visualisation (IV'06)*, pages 17–24, July 2006.

[369] M. Tory and T. Moller. Evaluating visualizations: do expert reviews work? *IEEE Computer Graphics and Applications*, 25(5):8–11, Sept 2005.

[370] M. Toyoda and M. Kitsuregawa. A system for visualizing and analyzing the evolution of the web with a time series of graphs. In *Proceedings of the 16th ACM Conference on Hypertext and Hypermedia*, HYPERTEXT, pages 151–160. ACM, 2005.

[371] J. Travers and S. Milgram. The small world problem. *Phychology Today*, 1:61–67, 1967.

[372] E. Tufte. *The visual display of quantitative information*. Graphics Press, Cheshire, Conn, 2001.

[373] E. Tufte. *Sparklines: Intense, simple, word-sized graphics*, pages 46–63. Graphics Press, 2006.

[374] W. T. Tutte. How to draw a graph. *Proceedings of the London Mathematical Society*, s3-13(1):743–767, 1963.

[375] S. Uddin and L. Hossain. Time scale degree centrality: A time-variant approach to degree centrality measures. In *Int. Conf. on Advances in Social Networks Analysis and Mining (ASONAM),*, pages 520 –524, 2011.

[376] A. Unger and H. Schumann. Visual support for the understanding of simulation processes. In *2009 IEEE Pacific Visualization Symposium*, pages 57–64, April 2009.

[377] P. Valdivia, F. Dias, F. Petronetto, C. T. Silva, and L. G. Nonato. Wavelet-based visualization of time-varying data on graphs. In *2015 IEEE Conference on Visual Analytics Science and Technology (VAST)*, pages 1–8, Oct 2015.

[378] J. Vallet, H. Kirchner, B. Pinaud, and G. Melançon. A visual analytics approach to compare propagation models in social networks. In *Graphs as Models*, volume 181, 2015.

[379] G. G. Van De Bunt, M. A. J. Van Duijn, and T. A. B. Snijders. Friendship networks through time: An actor-oriented dynamic statistical network model. *Comput. Math. Organ. Theory*, 5(2):167–192, July 1999.

[380] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk. Dynamic network visualization with extended massive sequence views. *IEEE Transactions on Visualization and Computer Graphics*, 20(8):1087–1099, 2013.

[381] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk. Reducing snapshots to points: A visual analytics approach to dynamic network exploration. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):1–10, Jan 2016.

[382] N. J. van Eck and L. Waltman. Citnetexplorer: A new software tool for analyzing and visualizing citation networks. *Journal of Informetrics*, 8(4):802 – 823, 2014.

[383] F. van Ham. Using multilevel call matrices in large software projects. In *IEEE Symposium on Information Visualization 2003 (IEEE Cat. No.03TH8714)*, pages 227–232, Oct 2003.

[384] 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.

[385] F. Van Ham and M. Wattenberg. Centrality based visualization of small world graphs. *Computer Graphics Forum*, 27(3):975–982, 2008.

[386] R. van Liere and W. de Leeuw. Graphsplatting: visualizing graphs as continuous fields. *IEEE Transactions on Visualization and Computer Graphics*, 9(2):206–212, April 2003.

[387] J. J. van Wijk. The value of visualization. In *VIS 05. IEEE Visualization, 2005.*, pages 79–86, Oct 2005.

[388] A.-T. Vandermonde. Remarques sur les problèmes de situation. *Mémoires de l'Académie Royale des Sciences (Paris)*, pages 566–574, 1771.

[389] C. Vehlow, F. Beck, P. Auwärter, and D. Weiskopf. Visualizing the evolution of communities in dynamic graphs. *Computer Graphics Forum*, 34(1):277–288, 2014.

[390] C. Vehlow, F. Beck, and D. Weiskopf. The state of the art in visualizing group structures in graphs. In *Eurographics Conference on Visualization (EuroVis) - STARs*, EuroVis. The Eurographics Association, 2015.

[391] C. Vehlow, M. Burch, H. Schmauder, and D. Weiskopf. Radial layered matrix visualization of dynamic graphs. In *Proceedings of the 17th International Conference Information Visualisation*, IV, pages 51–58. IEEE, 2013.

[392] C. Vehlow, J. Hasenauer, A. Kramer, A. Raue, S. Hug, J. Timmer, N. Radde, F. J. Theis, and D. Weiskopf. iVUN: interactive visualization of uncertain biochemical reaction networks. *BMC Bioinformatics*, 14(Suppl 19):S2, 2013.

[393] K. Verspoor, B. Ofoghi, and M. R. Granda. Commviz: Visualization of semantic patterns in large social communication networks. *Information Visualization*, 0(0):1473871617693039, 2017.

[394] C. Viau, M. J. McGuffin, Y. Chiricota, and I. Jurisica. The flowvizmenu and parallel scatterplot matrix: Hybrid multidimensional visualizations for network exploration. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1100–1108, Nov 2010.

[395] T. von Landesberger, S. Diel, S. Bremm, and D. W. Fellner. Visual analysis of contagion in networks. *Information Visualization*, 2013.

[396] T. von Landesberger, M. Görner, R. Rehner, and T. Schreck. A system for interactive visual analysis of large graphs using motifs in graph editing and aggregation. In *Proceedings of Vision Modeling Visualization Workshop*, pages 331–339, 2009.

[397] T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. van Wijk, J.-D. Fekete, and D. Fellner. Visual analysis of large graphs: State-of-the-art and future research challenges. *Computer Graphics Forum*, 30(6):1719–1749, 2011.

[398] C. Walshaw. A multilevel algorithm for force-directed graph drawing. In J. Marks, editor, *Graph Drawing: 8th International Symposium, GD 2000 Colonial Williamsburg, VA, USA, September 20–23, 2000 Proceedings*, pages 171–182. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001.

[399] C. Ware. *Information visualization : perception for design*. Morgan Kaufman, San Francisco, CA, 2004.

[400] C. Ware and R. Bobrow. Supporting visual queries on medium-sized node–link diagrams. *Information Visualization*, 4(1):49–58, 2005.

[401] C. Ware, H. Purchase, L. Colpoys, and M. McGill. Cognitive measurements of graph aesthetics. *Information Visualization*, 1(2):103–110, 2002.

[402] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Number 8 in Structural analysis in the social sciences. Cambridge University Press, 1 edition, 1994.

[403] M. Wattenberg. Arc diagrams: visualizing structure in strings. In *IEEE Symposium on Information Visualization, 2002. INFOVIS 2002.*, pages 110–116, 2002.

[404] M. Wattenberg. Visual exploration of multivariate graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI 06, pages 811–819, New York, NY, USA, 2006. ACM.

[405] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world'networks. *nature*, 393(6684):440–442, 1998.

[406] M. A. Westenberg, S. A. Van Hijum, O. P. Kuipers, and J. B. Roerdink. Visualizing genome expression and regulatory network dynamics in genomic and metabolic context. *Computer Graphics Forum*, 27(3):887–894, 2008.

[407] M. A. Westenberg, S. A. van Hijum, A. T. Lulko, O. P. Kuipers, and J. B. Roerdink. Interactive visualization of gene regulatory networks with associated gene expression time series data. In *Visualization in Medicine and Life Sciences*, pages 293–311. Springer, 2008.

[408] R. Wilson and J. J. Watkins, editors. *Combinatorics : ancient and modern*. Oxford University Press, Oxford, 2013.

[409] R. J. Wilson. An eulerian trail through königsberg. *Journal of Graph Theory*, 10(3):265–275, 1986.

[410] F. Windhager, A. Amor-Amorós, M. Smuc, P. Federico, L. Zenk, and S. Miksch. A concept for the exploratory visualization of patent network dynamics. In *Proceedings of the 6th International Conference on Information Visualization Theory and Applications*, IVAPP, pages 268–273, 2015.

[411] F. Windhager, A. Amor-Amoros, M. Smuc, P. Federico, L. Zenk, and S. Miksch. A concept for the exploratory visualization of patent network dynamics. In *Proceedings of the 6th International Conference on Information Visualization Theory and Applications*, pages 268–273, 2015.

[412] F. Windhager, M. Smuc, L. Zenk, P. Federico, J. Pfeffer, and W. Aigner. On visualizing knowledge flows at a university department. *Procedia - Social and Behavioral Sciences*, 100:127 – 143, 2013.

[413] F. Windhager, M. Smuc, L. Zenk, P. Federico, J. Pfeffer, W. Aigner, and S. Miksch. Visual knowledge networks analytics. In J. Liebowitz, editor, *Knowledge Management Handbook*, page 187–206. CRC Press, 2012.

[414] F. Windhager, L. Zenk, and P. Federico. Visual enterprise network analytics - visualizing organizational change. *Procedia - Social and Behavioral Sciences*, 22:59 – 68, 2011.

[415] P. C. Wong, P. Mackey, K. A. Cook, R. M. Rohrer, H. Foote, and M. A. Whiting. A multi-level middle-out cross-zooming approach for large graph analytics. In *2009 IEEE Symposium on Visual Analytics Science and Technology*, pages 147–154, Oct 2009.

[416] L. E. Wood. Semi-structured interviewing for user-centered design. *interactions*, 4(2):48–61, Mar. 1997.

[417] Y. Wu, N. Pitipornvivat, J. Zhao, S. Yang, G. Huang, and H. Qu. egoslider: Visual analysis of egocentric network evolution. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):260–269, Jan 2016.

[418] M. Wybrow, N. Elmqvist, J.-D. Fekete, T. von Landesberger, J. van Wijk, and B. Zimmer. Interaction in the visualization of multivariate networks. In Kerren et al. [233], pages 97–125.

[419] K. Xu, C. Rooney, P. Passmore, and D.-H. Ham. A user study on curved edges in graph visualisation. In P. Cox, B. Plimmer, and P. Rodgers, editors, *Diagrammatic Representation and Inference: 7th International Conference, Diagrams 2012, Canterbury, UK, July 2-6, 2012. Proceedings*, pages 306–308. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[420] 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.

[421] X. Yang, S. Asur, S. Parthasarathy, and S. Mehta. A visual-analytic toolkit for dynamic interaction graphs. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD, pages 1016–1024. ACM, 2008.

[422] K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst. Animated exploration of dynamic graphs with radial layout. In *Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01)*, INFOVIS '01, pages 43–, Washington, DC, USA, 2001. IEEE Computer Society.

[423] J. S. Yi, Y. ah 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, Nov 2007.

[424] J. S. Yi, N. Elmqvist, and S. Lee. TimeMatrix: Analyzing temporal social networks using interactive matrix-based visualizations. *International Journal of Human-Computer Interaction*, 26(11-12):1031–1051, 2010.

[425] F. Zaidi, C. Muelder, and A. Sallaberry. Analysis and visualization of dynamic networks. In R. Alhajj and J. Rokne, editors, *Encyclopedia of Social Network Analysis and Mining*, pages 37–48. Springer New York, New York, NY, 2014.

[426] L. Zaman, A. Kalra, and W. Stuerzlinger. The effect of animation, dual view, difference layers, and relative re-layout in hierarchical diagram differencing. In *Proceedings of Graphics Interface 2011*, GI, pages 183–190. Canadian Human-Computer Communications Society, 2011.

[427] J. Zhang. A survey on streaming algorithms for massive graphs. In *Managing and Mining Graph Data*, pages 393–420. Springer, 2010.

[428] J. Zhao, M. Glueck, F. Chevalier, Y. Wu, and A. Khan. Egocentric analysis of dynamic networks with egolines. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 5003–5014, New York, NY, USA, 2016. ACM.

[429] S. Zhao, M. J. McGuffin, and M. H. Chignell. Elastic hierarchies: combining treemaps and node-link diagrams. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.*, pages 57–64, Oct 2005.

[430] H. Zhou, P. Xu, X. Yuan, and H. Qu. Edge bundling in information visualization. *Tsinghua Science and Technology*, 18(2):145–156, April 2013.

[431] Y. Zhu, J. Yu, and J. Wu. Chro-ring: a time-oriented visual approach to represent writer's history. *The Visual Computer*, 32(9):1133–1149, 2016.

[432] B. Zimmer, I. Jusufi, and A. Kerren. Analyzing multiple network centralities with vincent. In *Proceedings of SIGRAD 2012; Interactive Visual Analysis of Data*, pages 87–90. Linköping University Electronic Press, 2012.

[433] M. Zinsmaier, U. Brandes, O. Deussen, and H. Strobelt. Interactive level-of-detail rendering of large graphs. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2486–2495, Dec 2012.