EUROVIS 2019
R. S. Laramee, S. Oeltze, and M. Sedlmair
(Guest Editors)

Volume 38 (2019), Number 3
STAR – State of The Art Report

The State of the Art in Visualizing Multivariate Networks

C. Nobre[1], M. Meyer[1], M. Streit[2], and A. Lex[1]

[1]University of Utah, Utah, USA
[2]Johannes Kepler University Linz, Austria

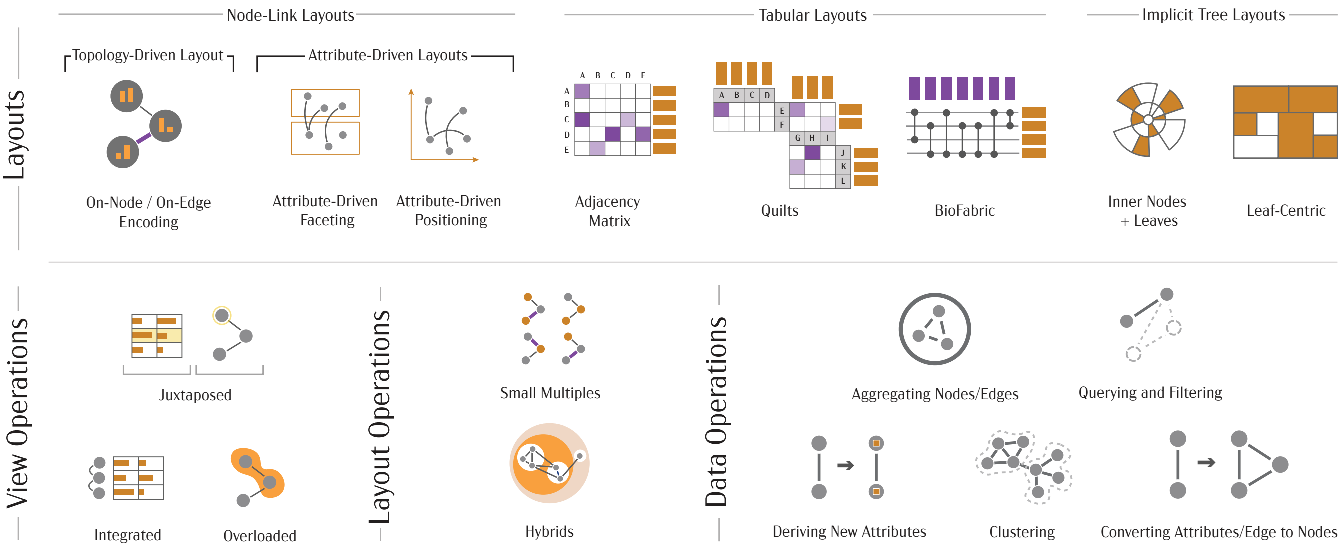


Figure 1: *A typology of operations and layouts used in multivariate network visualization.* ***Layouts*** *describe the fundamental choices for encoding multivariate networks.* ***View Operations*** *capture how topology and attribute focused visualizations can be combined.* ***Layout Operations*** *are applied to basic layouts to create specific visualization techniques.* ***Data Operations*** *are used to transform a network or derive attributes before visualizations. The colors reflect node attributes (orange), edge attributes (purple), and topology (grey).*

Abstract

Multivariate networks are made up of nodes and their relationships (links), but also data about those nodes and links as attributes. Most real-world networks are associated with several attributes, and many analysis tasks depend on analyzing both, relationships and attributes. Visualization of multivariate networks, however, is challenging, especially when both the topology of the network and the attributes need to be considered concurrently. In this state-of-the-art report, we analyze current practices and classify techniques along four axes: layouts, view operations, layout operations, and data operations. We also provide an analysis of tasks specific to multivariate networks and give recommendations for which technique to use in which scenario. Finally, we survey application areas and evaluation methodologies.

1. Introduction

Networks are ubiquitous in today's society. A network, also called a graph, consists of nodes (or vertices) and links (or edges) connecting these nodes. Examples range from social networks, to physical networks such as power grids, to networks modeling cellular processes in biology, to trees describing evolutionary relationships between species. Nodes and links can represent people, locations, genes, or ultimately any entity that has a connection with other entities. Most real-world networks consist of nodes and links that are also associated with attributes. Attributes can range from minimal information, such as node labels, to complex attribute vectors, such as the age, skills, and gender of a person, or the activity levels of genes from hundreds of samples. Networks that also contain attributes are commonly referred to as multivariate networks (MVNs) [KPW14]. When the structure (the topology) of the network needs to be analyzed together with these node or edge attributes, we employ *multivariate network visualization techniques*, which are the topic of this state-of-the-art report.

Visualizing MVNs is particularly challenging, as both the *topology* of the network (also called *structure* of the network), and *attributes* associated with the nodes and links need to be shown. When visualizing topology and attributes in the same view, choosing efficient encodings for one aspect often interferes with the ability to effectively visualize the other. For example, when attempting to encode multiple node attributes directly on the node in a node-link diagram, a visualization designer has to make compromises with regard to the node size, the network size, and the number of attributes shown. Conversely, when visualizing attributes and topology in separate views, an analyst's ability to judge interactions between nodes/links and attributes is limited. Most of the published techniques on multivariate networks attempt to mitigate the trade-off between visualizing topology and visualizing attributes well by introducing novel encodings for certain data types or analysis tasks.

In this paper, we introduce new typologies for multivariate network tasks and multivariate network techniques. Our *task typology* (Section 4) is based on two fundamental MVN tasks: identifying a topological structure based on attributes or characterizing the attributes of a topological structure. We apply these tasks to a series of relevant topological structures, which results in ten MVN tasks.

The *visualization technique typology* (Section 5), shown in Figure 1, consists of four dimensions: layouts, view operations, layout operations, and data operations. The layouts category includes the eight unique multivariate network layouts. View operations describe how multiple views can be used for multivariate network visualization. Layout operations either combine multiple layouts or multiply layouts based on attributes. Data operations describe possible network wrangling operations such as aggregation and computing new attributes. We also provide explicit guidelines on recommended usage of each technique, as well as on how to select an appropriate technique given a network type, attribute characteristics, and a set of tasks the user aims to support. A companion website, available at `https://vdl.sci.utah.edu/mvnv/`, provides an overview of the taxonomy and a wizard to select a technique for a specific data and task combination.

Finally, we discuss our methodology (Section 6), common application areas of MVNs (Section 7), and strategies for evaluating multivariate networks (Section 8).

2. Related STAR Reports

The body of literature in the space of visualizing multivariate networks is consistently growing, but a comprehensive survey of the techniques used has not yet been done. STARs exist on related topics such as visualizing large graphs [vLKS*11], trees [Sch11], implicit hierarchies [SHS11], group structures in graphs [VBW15], dynamic graphs [BBDW14], edge bundling techniques [LHT17], among others. However, no survey paper introducing a typology of multivariate network visualization techniques exists.

The survey on multifaceted graph visualization by Hadlak et al. [HSS15] covers the visualization of four facets of graph visualization, two of which are relevant to our work—partitions and attributes. Their work distinguishes three overarching categories of MVN visualizations—visualizations with a topology-driven layout, an attribute-driven layout, and coordinated views—but is not exhaustive in describing MVN techniques, including tabular techniques and hybrid approaches. Most related to the proposed STAR is a survey book edited by Kerren et al. based on a Dagstuhl Seminar [KPW14] that reports on selected topics in the field but does not introduce a taxonomy of existing techniques.

This review paper aims to fill that gap by describing the design space of techniques used to encode multivariate networks, classifying these techniques into a formal typology, and surveying existing evaluation methods. We only briefly discuss application areas for MVNs, as these are covered in depth in the survey book by Kerren et al. [KPW14].

3. Network and Attribute Characteristics

Here we introduce definitions and describe different types of networks. We consider multivariate networks with nodes and edges where either the nodes, the edges, or both, have attributes associated with them. We refer to the nodes and edges as the network's *topology*. Attributes can be of varying data types, such as textual, quantitative, ordinal, nominal, or set data. Attributes can also be computed based on topological properties. For example, the degree of a node, or its centrality, are common derived attributes. Analogously, topological structures can be derived from attributes. For example, two nodes can be connected if they share an attribute.

Both nodes and edges can be of different types. We consider nodes and edges of a type to be of a semantic unit that has a different set of attributes associated with it. For example, a node type *actor* could have a name, current residence, age, etc. associated with it. Another node type in the same network could be *movies*, with attributes such as movie title, budget, revenue, genres, etc.

In this survey, we discuss the properties of a visualization technique on a number of dimensions, including network size, network type, node and edge attribute size and type, and their suitability for tasks. Here we discuss the data properties, while the tasks are described in Section 4.

For *size*, we distinguish between three categories of networks: small (less than 100 nodes), medium (100-1,000 nodes), and large (more than 1,000 nodes). For our evaluation, we consider the number of nodes simultaneously displayed, not the number of nodes a technique can handle through querying or filtering.

We distinguish between the following network types:

- **Complex networks**, i.e., networks with nontrivial topological features, such as a diverse degree distribution (the distribution of the number of links per node), clusters, hubs, etc. [WC03]. Among the complex networks are scale-free networks, with a degree distribution that follows the power law distribution. These networks are characterized by a small number of nodes with a very high degree that exhibit preferential attachment, i.e., that nodes added to the network are likely to attach to these central hubs. An example is the world wide web, where major websites are linked to much more frequently than others. Another type of complex network is a **small world network**, with a degree distribution that peaks around the average value. These networks exhibit significant clustering and are characterized by a short average path distance (hence small world). Social networks are a typical example.

- **Layered and K-Partite networks** are networks where the nodes can be partitioned into discrete sets and where links are mostly (in the case of layered networks) or exclusively (in the case of k-partite networks) between the different sets. Layered networks are ordered (e.g., as in genealogies, where there is an order of generations that is represented as layers) and have links only between successive layers, whereas k-partite networks can have links between multiple partitions and are not necessarily ordered.
- **Trees** are acyclic networks, commonly with a defined root, where each node is either a leaf or the root of a subtree. Trees have exactly one path between two nodes and have a low average degree.

For complex networks, we also distinguish between *sparse* networks, with a low average degree, and *dense* networks, with a high average node degree. Layered networks, k-partite networks, and trees are sparse by their nature.

For *node attributes*, we distinguish between *few (less than five)* and *many (more than five)*, but also give more detailed ranges when discussing a specific technique. We considered a category for a high number of attributes (e.g., hundreds), but realized that the same set of techniques that support five or more attributes well also support larger numbers of attributes. We also discuss data type properties when appropriate. We further distinguish between *homogeneous networks with respect to node types*, i.e., all nodes are of the same type (e.g., only actors), and *heterogeneous networks*, i.e., the network has more than one node type (e.g., actors and movies).

We categorize analogously for *edge attributes*, albeit we consider few to be less than three and many to be three or more attributes. Finally, we distinguish between homogeneous and heterogeneous edges.

4. Multivariate Network Analysis Tasks

Here we introduce a taxonomy of tasks focusing specifically on multivariate network analysis. Our taxonomy is designed to evaluate MVN techniques for their suitability to address a task and is complementary to general network task taxonomies. This task analysis, combined with considerations of network type and network scale described in the previous section, form the basis of our assessment of individual visualization techniques and of our guidelines for MVN visualization (Section 5.1).

There are various data-type independent visualization task taxonomies. The works by Brehmer and Munzner [BM13] and Schulz et al. [SNHS13] are examples of recent, generic visualization task taxonomies. An example of a data-type-specific task taxonomy is the work by Valiati et al. [VPF06] which covers multidimensional visualization techniques.

Specialized task taxonomies for networks also exist. The well-known taxonomy of network tasks by Lee et al. [LPP*06] does not cover MVN tasks in detail. In addition to topology-focused tasks, the taxonomy includes selected attribute-based tasks. For node attributes, these tasks are *finding nodes having a specific attribute value* and *reviewing the set of nodes*. For edges, the two tasks are about identifying neighbors of a node connected by an edge with a specific property (edge type, specific edge value). Shneiderman and Aris [SA06] also discuss examples of MVN tasks but do not take a systematic approach to the topic.

The most comprehensive analysis of tasks for multivariate networks was done by Pretorius et al. [PPS14]. The authors generalize the tasks proposed by Lee et al. to cater to cases where the network's attributes are of interest to the user. This approach is useful in that it provides a systematic and comprehensive approach to MVN tasks. They list 25 tasks in the categories of structure-based (topology-based), attribute-based, browsing, and estimation tasks. They also discuss how these tasks are composed of lower level analytical tasks proposed by Valiati et al. (identify, determine, relocate, compare) [VPF06].

We believe that the task taxonomy by Pretorious et al. is comprehensive and refer to it for a detailed review. Pretorius et al. also discuss that in the abstract, a task is an analytical activity performed on an entity and a property of an entity, which is called “actions” applied to “targets” by Brehmer and Munzner [BM13]. For the purpose of this STAR, we believe that a slightly simplified approach is useful for both characterizing existing techniques and recommending techniques based on tasks specified by analysts.

In our taxonomy, any task can be expressed as a combination of two fundamental tasks, as applied to different topological structures of a network. These two basic tasks are:

- *analyzing topology for given attributes (TgA)*, which are aimed at identifying, characterizing, or comparing topological structures that have certain attributes, and
- *analyzing attributes for a given topological structure (AgT)*, which are aimed at identifying, characterizing, or comparing the attributes of a given topological structure.

We do not further distinguish between more detailed tasks, such as identifying, summarizing, or searching, because we believe that in most practical scenarios these go hand in hand, and because most visualization techniques that support one of these also support the others. A notable exception is comparison, which can often be achieved without special considerations, but can benefit from dedicated support by a visualization technique in certain situations.

The targets in the context of network tasks are the *topological structures*. We consider major types of topological structures, shown in Figure 2, but recognize that others, such as cliques or spanning trees, could be relevant for specific analysis questions.

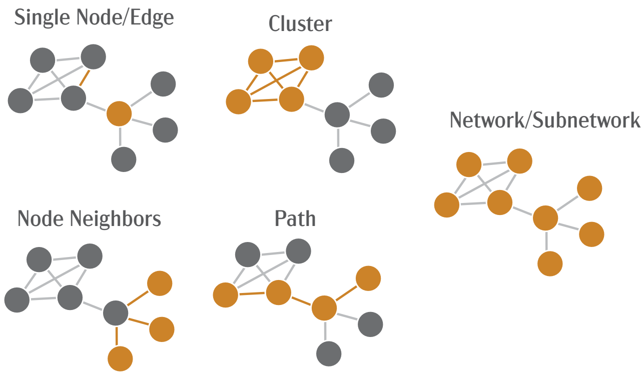


Figure 2: *Topological structures we consider as targets for tasks.*

The structures include individual *nodes and edges*, *paths* of sequential nodes and edges, *neighborhoods* of a particular node or edge, *clusters* of nodes and edges, and larger *networks or subnetworks* that can be considered as complex networks by themselves. We use the term "cluster" to refer to a set of nodes that are well-connected to each other, but less so with other nodes with regards to the topology of the network. A cluster is also often called a community, especially in the context of social networks. Conversely, a "subnetwork" is a subset of a network that is not limited to a specific structure.

We selected these specific structures based on our literature analysis of MVN visualization techniques, where we identified many published techniques that are optimized for performing MVN tasks on specific topological structures. For example, approaches such as Pathfinder [PGS*16] and Entourage [LPK*13] are explicitly developed for path-based MVN tasks. GraphCharter [TS13], on the other hand, is optimized for analysis on node neighbors.

A comprehensive analysis of multivariate network tasks can thus be described as a combination of general network tasks (e.g., as described by Lee et al. [LPP*06]) and our MVN tasks. By reducing MVN tasks to a combination of two basic tasks and a topological structure, we are able to decompose complex tasks into a simple combination of tasks and targets. The task "Estimate the average value of a specific attribute of all nodes connected to a specific node" (a concrete example: what's the average income of University of Utah graduates in a social network) can be deconstructed into three basic tasks: (1) find the node with label "University of Utah" (analyzing topology for given attributes, on target "network"), followed by (2) Lee et al.'s adjacency task (find the set of nodes adjacent to a node), followed by (3) estimate the average value of an attribute of a node neighborhood (analyzing attributes for a given topolgy on target "neighbors").

Table 1 gives examples of the two basic MVN analysis tasks as applied to the set of topological structures listed in Figure 2. For each task, we also provide an example for a domain-specific task from the literature.

Although we observed both TgA and AgT tasks in our corpus, we saw a preference for one of the task types in certain application areas. Techniques developed for social network analysis, for example, tend to first focus on topology-only tasks, followed by attribute tasks on the selected topological structures (AgT). For example, a commonly described task is to identify particular structures, such as communities, followed by an analysis of the attributes in these structures [HF07].

We also observed that identifying a topological structure for given attributes (TgA) is frequently supported by query or filter operations, instead of being a purely visual task. For example, an analyst may want to identify a fraudulent bank account by querying for patterns of transactions, and then expand a network of interactions between that account and others [vHP09]. Similarly, Juniper [NSL19], Pathfinder [PGS*16], and Enroute [PLS*12] use queries or filters to isolate a subnetwork or path of interest, followed by inspection of the attributes in that structure.

5. Multivariate Network Visualization Typology

The primary contribution of this paper is a typology of multivariate network visualization techniques, illustrated in Figure 1. The typology is made up of four components: a taxonomy of MVN layouts, and three types of operations that can be applied to these layouts. Every technique and tool discussed in this paper can be described by selecting (at least) one layout, and optionally applying operations to it.

The *layout taxonomy* encompasses core approaches to encoding the topology and attributes of multivariate networks and trees. The taxonomy of layouts contains three classes: node-link, tabular, and implicit layouts, with various layouts within these classes.

The *view operations* capture different ways of combining multiple views with each other. For example, view operations include juxtaposition for placing an attribute and a topology visualization side-by-side.

The *layout operations* allow for different combinations of the layouts to produce hybrids and small multiples.

The *data operations* category contains methods that can be applied to the data either as a preprocessing step or during the analysis. Example data operations are aggregating nodes into supernodes or filtering parts of a network.

Toplogical Structure	Task Type	Example	Domain-Specific Example
Node/Edge	TgA	Which node/edge has the given attribute value?	Who is the oldest person in a social network? [HB05]
	AgT	What is the attribute value of a given node/edge?	What house is Jon Snow in? [NSL19]
Neighbors	TgA	Which neighbors have a given attribute value?	Who is Mary's tallest friend? [LPB*06]
	AgT	What are the attribute values of a node's neighbors?	What are the zodiacs of Joe's friends? [TS13]
Paths	TgA	Which path between two nodes has the lowest cost?	Which path between two genes is most frequently cited in pathway databases? [PGS*16]
	AgT	What are the attribute values along a path?	How does a metabolite increase along a pathway? [MWS*10]
Clusters	TgA	Which cluster has certain attribute characteristics?	Which cluster has the most educated people? [HF06]
	AgT	What are the attribute characteristics within a cluster?	What is the average education level in Mary's cluster? [HF06]
Network	TgA	Find the subnetwork of nodes with a specific attribute.	Which subnetwork contains only people with a PhD? [HF06]
	AgT	What are the attribute values in this network?	What is the average age in this network? [vHSD*09]

Table 1: *List of generic and domain-specific examples of MVN analysis tasks composed of our two basic tasks and the described topological structures. Each domain-specific example references a paper that describes that task.*

		Size			Type				Node Attributes				Edge Attributes				Topolog. Structures				
		Small <100 nodes)	Medium (<1,000)	Large (>1,000 nodes)	Complex (sparse)	Complex (dense)	Layered/K-Partite	Trees	Few (<5)	Several (≥5)	Homog. (1 type)	Hetero. (>1 type)	Few (<3)	Several (≥3)	Homog. (1 type)	Hetero. (>1 type)	Single node/edge	Neighbors	Paths	Clusters	Entire/sub network
Node-Link Layouts	On-node/edge encoding	3	2	1	3	1	3	3	2	1	3	2	2	1	3	1	3	3	2	2	2
	Attr.-driven faceting	3	1	1	3	1	3	1	3	1	3	3	2	1	2	1	3	2	1	1	1
	Attr.-driven positioning	3	1	1	3	1	1	1	3	1	3	1	2	1	2	1	3	2	1	1	2
Tabular Layouts	Adjacency matrix	3	1	1	2	3	2	1	2	3	3	2	3	2	3	2	3	3	1	3	2
	Quilts	3	1	1	3	1	3	3	3	3	3	3	3	3	3	2	3	3	2	2	2
	BioFabric	3	1	1	3	3	2	1	3	3	3	3	3	3	3	3	3	1	1	1	2
Implicit	Inner nodes & leaves	3	2	1	0	0	0	3	3	1	3	1	0	0	0	0	3	3	3	0	3
	Leaves	3	2	2	0	0	0	3	3	1	3	1	0	0	0	0	3	2	1	0	3
View Operations	Juxtaposed	3	2	1	3	1	3	3	3	3	3	3	3	3	3	3	2	1	1	2	2
	Integrated	3	2	1	3	1	3	3	3	3	3	3	2	2	3	3	3	3	3	1	2
	Overloaded	3	2	1	3	1	3	3	3	1	3	1	1	1	1	1	3	3	2	3	2

Table 2: *Scores for how well each technique performs on different network types and different multivariate network tasks.* ***0*** *means* ***does not support at all****,* ***1*** *is* ***supports poorly****,* ***2*** *encodes* ***supported but with limitations / may require interaction****, and* ***3*** *means* ***well supported****.*

In describing MVN visualization techniques along these four orthogonal dimensions, we not only cover the techniques observed in the set of papers surveyed, but also provide a design space in which to decompose and analyze any MVN visualization technique.

When describing and discussing these approaches, e.g., with respect to scalability, we assume a static visualization technique. However, we call out opportunities for *interaction* where appropriate. Certain types of interactions, such as zooming and panning, or visualizing attributes such as labels in tool tips, can be applied to most techniques. Other, more sophisticated interactions, such as linking and brushing, sorting, changing the layout algorithms, or manually adjusting a layout, depend on the visualization technique used. The *data operations* we describe can frequently be invoked interactively, and especially operations such as filtering or aggregating can drastically change how much and what types of raw data can be visualized with a basic layout technique.

We scored each of the layouts and view operations along 20 dimensions, which include network characteristics such as size and number of attributes, as well as different types of MVN analysis tasks. Table 2 shows the scores for each type. Our methodology for developing these scores is described in Section 6.4. We use the following scores:

0	This technique **cannot** support this data type or task.
1	This technique supports this data type or task very poorly.
2	This technique can support this data type or task, but is not ideal and may require interaction to achieve it.
3	This technique supports this data type or task very well.

5.1. Layouts

We distinguish between three approaches to network visualization: Node-link layouts, tabular layouts, and implicit tree layouts. The class of tabular layouts contains the widely used matrix layout and two variants: quilts and biofabric. Implicit tree layouts encode the relationships between nodes only by their relative positions, hence edges are only implicitly represented. In practice, the lack of explicit edges means that implicit layouts are useful only for trees.

5.1.1. Node-Link Layouts

Node-link diagrams are the most common graphical representation of networks. In a node-link diagram, the nodes are drawn as point marks and the links as line/curve marks connecting the nodes. Node link layouts are the subject of their own field of study—graph drawing—and countless algorithms for node-link layouts have been developed [BETT98]. Schulz and Schumann [SS06] distinguish between *free layouts*, where the node layout is not restricted, an example of which is the force-directed placement; *styled layouts*, where the node positions are determined by a predefined scheme, such as a grid, or a radial layout; and *fixed layouts* where the position of the node is determined by an attribute, such as latitude and longitude on a map. In fixed layouts, the only degree of freedom is conferred to the drawing of the edges connecting the fixed nodes.

We use this distinction, but for the purposes of multivariate network visualization, we group free and styled layouts into one larger category of *topology-driven layouts* and take a slightly relaxed view on fixed layouts and describe it as *attribute-driven layouts*.

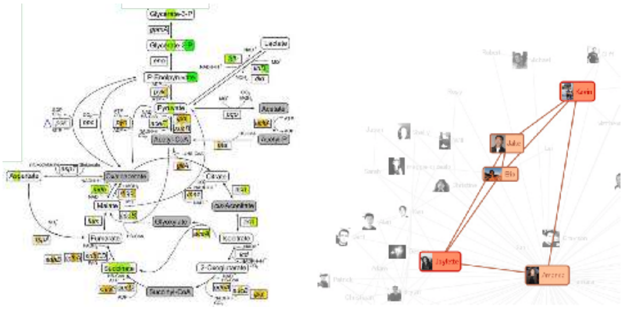
Topology-driven layouts prioritize the topology of the network over the attributes of the nodes and edges. The most common node-link layouts, such as force-directed layouts, spectral layouts, or orthogonal layouts, fall into this class, or more specifically into the free layouts as described by Schulz and Schumann. Trees are also commonly drawn as topology-driven node-link layouts. Styled layouts position nodes based on a pattern, such as along a circle or in a grid, and then connect them with edges. Styled layouts quickly lead to clutter, which can be partially mitigated by optimizing the ordering of nodes, e.g., by placing connected nodes close to each other. As topology-driven layouts are optimized to visualize the network structure, position, the most powerful visual channel, is not available to encode attributes of the network. Hence, the only layout strategy to visualize attributes is to alter the appearance of the nodes and links through on-node or on-edge encoding.

Attribute-Driven Layouts use node position to encode attributes. We distinguish two cases: attribute-driven faceting, which places the nodes in regions corresponding to a categorical attribute, but the exact node position within that region is determined in another way, or attribute-driven positioning, which places the nodes exactly by an (often numerical) attribute value.

5.1.1.1. On-Node/On-Edge Encoding.

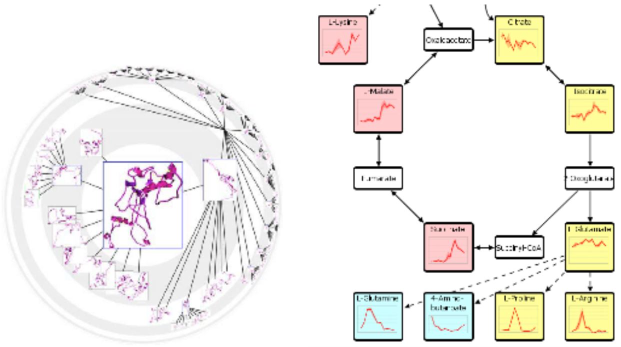
On-node and on-edge encoding refers to modifying the visual appearance (size, color) of a node or an edge or embedding marks (bar charts, line charts, etc.) in a node or an edge in a node-link diagram.

For *nodes*, labels are often shown as text, whereas color coding is a common choice to encode numerical or categorical data values. Other common encodings for node types are shapes or icons. Gehlenborg et al. [GOB*10] review techniques used in systems biology for visualizing multivariate networks, many of which make use of on-node encoding by means of embedded charts, such as line charts, box plots, etc. An example of a systems biology use of on-node encoding is shown in Figure 3(a), where metabolic pathways are overlaid with node attributes on metabolite concentration [NPA*09]. Auber et al. [ACJM03] use on-node encoding on social networks, displaying small network representations for previously computed topological features of interest. Also in the realm of social media, Vizster [HB05] uses photo and labels on the nodes. Dunne and Shneiderman [DS13] use glyphs to encode aggregate nodes representing topological patterns in the network such as cliques and fans. On-node encoding is also widely supported by common network visualization tools such as Cytoscape [SOR*11] and Gephi [BHJ09].



(a) (b)

Figure 3: *On-node encoding. (a) Multiple attributes (metabolite concentrations) are encoded directly on the nodes using color [NPA*09]. Figure licensed under CC BY 2.0. (b) Photos and labels are encoded on the nodes of a social network [HB05].* *© 2005 IEEE, reprinted with permission.*

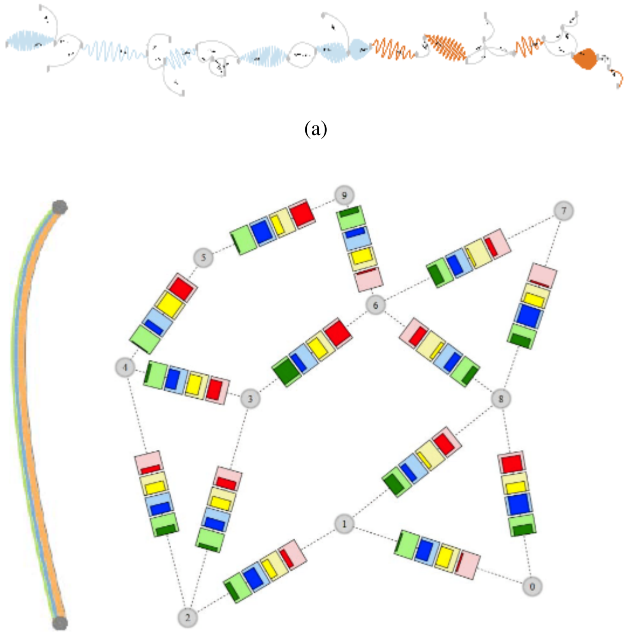


(a) (b)

Figure 4: *Complex on-node encodings. (a) A protein interaction network where nodes contain visualizations of protein configurations [JM03].* *© 2003 IEEE, reprinted with permission. (b) Line charts show metabolite concentrations over time in a pathway network [JKS06]. Figure licensed under CC BY 2.0.*

Ghani and Elmqvist [GE11] study the efficiency of different channels for on-node encoding for revisitation tasks. The authors compare the use of color, size, and color combined with size, but did not find significant difference in the performance of their tasks.

On-node encoding can also be realized using more complex visualizations within the nodes. Examples of these encodings include line-charts, or a “business card” layout for a person, with name, picture, age, etc.



(b) (c)

Figure 5: *On-edge encoding. (a) A numerical attribute is encoded by the overall length of the edge curve. To make the distance between connected nodes of approximately uniform length, wiggles are used [NJBJec]. © 2009 IEEE, reprinted with permission. (b) Several numerical attributes are encoded by the thickness of the colored segments [KAW*14]. © 2014 IEEE, reprinted with permission. (c) Colored bars encoding four quantitative edge attributes [SSE16]. © 2016 IEEE, reprinted with permission.*

Examples for techniques using complex on-node encodings are MoireGraphs [JM03] (see Figure 4(a)) and Network Lens [JDK10]. Network Lens allows the users to enlarge nodes and encodes their attributes with approaches such as bar charts and parallel coordinates plots. Other examples from the biology domain include embedded bar charts with error bars [WvHKR08] and line charts [JKS06].

On-edge encoding shows edge attributes by modifying the channel of the line mark used for the edge. Common approaches are modifying line width [KWS*13, Guo09], line color [Guo09], stippling and dashes [JKS06], curvature [XRP*12], or blurring [SSSE16]. Multiple attributes are occasionally encoded with bar charts overlaid on the edges [SSE16] (Figure 5(c)) or with multicolored segments with different line width [KAW*14] (Figure 5(b)).

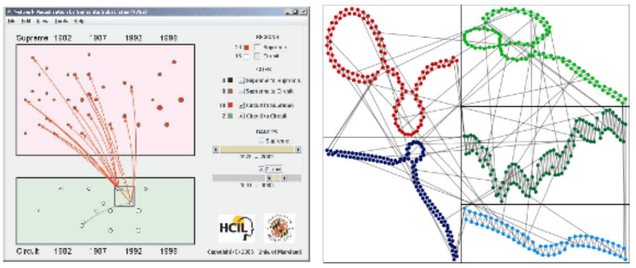
Abyss Explorer [NJBJec], shown in Figure 5(a), encodes edge attributes by edge length. However, as modifying edge length has a negative impact on the freedom to place nodes, the authors “wiggle” the edges, so that the attribute is also encoded by the frequency of the wiggles.

Edge directionality is commonly visualized with arrows at the ends but tapering or gradients are other approaches [HIvWF11].

Recommended Usage. On-ode/on-edge encoding supports the integration of topology- and attribute-based tasks well and supports all kinds of MVN tasks on all structures (see Table 2). On-node/on-edge encoding is easily understood by most users, and works well for sparse complex networks, layered networks, and trees. However, it comes with scalability trade-offs. Even for a modest number of nodes in a node-link layout, node size has to be limited; hence little space is available to encode attributes. When details about nodes are shown, as for example in MoireGraphs [JM03], the number of nodes that can be displayed simultaneously is limited. We recommend on-node layouts when only a few (usually under five) attributes on the nodes are shown, or in combination with a zooming/filtering strategy. Complex on-node encodings can also be embedded on top of aggregated nodes, to summarize the content of the aggregates, in which case their larger size is often justified [vdEvW14]. On-node encoding generally works well for networks with different node types. On-edge encoding is even more limited than on-node encoding. First, most node-link layouts guarantee that nodes do not overlap, however, edges commonly cross even in sparse networks, interfering with on-edge encoding. Second, the nature of the link mark as a slim line limits the discriminability of any modulation of the visual channel. We recommend on-edge encoding for a single numerical or categorical attribute.

5.1.1.2. Attribute-Driven Faceting.

Attribute-driven faceting groups nodes according to one or more attributes and places the elements of a group in a shared region. A prominent example is Semantic Substrates [SA06], Figure 6(a), which facet a network based on a categorical attribute and let analysts choose whether to show links between or within facets, or both. This approach is also commonly used in biological networks to visualize nodes within spatially segregated cell compartments [BMGK08], or to lay out k-partite or multityped graphs [SJUS08, PLS*14, PvW08, GKL*13, ABF*07]. There are alternatives to faceting by sets or node types: Figure 6(b) shows faceting based on a hierarchical clustering algorithm where the clusters are laid out in a treemap [RMS*11] and nodes are shown within the treemap cells. A key choice after faceting is how to place the nodes within the assigned region, which can be done using any network layout technique. Semantic Substrates [SA06,AS07], for example, place nodes according to other attribute values, whereas Cerebral [BMGK08]



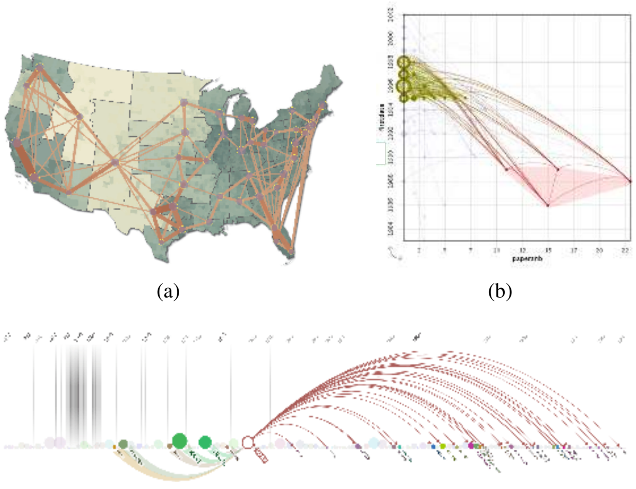
(a) (b)

Figure 6: *Attribute faceting. (a) Nodes are faceted by node type or a set attribute and placed in corresponding regions. The horizontal position within the facets is driven by an attribute [SA06]. © 2006 IEEE, reprinted with permission. (b) Nodes are faceted based on a clustering algorithm. The clusters are arranged using a treemap layout [RMS*11]. © 2011 IEEE, reprinted with permission.*

uses a layout optimized for topology. A series of other technique use a linear layout [PvW08, SJUS08, PLS*14], which is amenable to attribute visualization.

Recommended Usage. Attribute-driven faceting is well suited for networks with different node types or with an important categorical or set-like attribute. Such faceting is especially useful when the separation into groups and the study of the interaction within and between the groups are the subject of the analysis, which is commonly the case in k-partite and layered networks (see Table 2). Due to restrictions on the layout, it is slightly less scalable with respect to the number of nodes and network density than node-link layouts. Other attributes can be visualized independently of the basic principle of faceting, so that the scalability with respect to other attributes depends on these choices. Edge attributes are not supported by faceting and have to rely on a secondary encoding. Neighborhoods, paths, and clusters are not easily visible if they span different facets. We recommend attribute faceting for cases where nodes can be separated into groups easily and where these groups are central to the analysis.

5.1.1.3. Attribute-Driven Positioning. Attribute-driven positioning (fixed layouts) assigns node or edge positions according to one or more attribute values. Spatial networks, such as networks overlaid on maps [vdEvW14, Guo09] (Figure 7(a)), are the most common example of attribute-based node positioning. However, attributes other than geographic coordinates have also been used. In 1D, genomic coordinates are a common example [KSB*09, MMP09], as are networks with a time component [DCW11], shown in Figure 7(c). Multiple nonspatial attributes can be encoded in two dimensions [SA06, BCD*10, VMCJ10] (Figure 7(b)). This principle has also been extended to visualize multiple attribute pairs [BCD*10, VMCJ10] in a small multiple/scatterplot matrix display. The attributes used for positioning can also be derived from a larger attribute vector. For example, Bonabeau [Bon02] uses self-organizing maps, and Doerk et al. [DCW11] use dimensionality reduction based on attributes to compute placement of the nodes in space.


(a)

(b)

(c)

Figure 7: *Attribute-driven node positioning. (a) Nodes are positioned according to their latitude and longitude on a map [Guo09]. © 2009 IEEE, reprinted with permission. (b) Nodes are positioned according to nonspatial attributes [BCD*10]. © 2010 John Wiley and Sons, reprinted with permission. (c) Nodes are positioned in 1D, time in this case [DCW11]. Image courtesy of the authors.*

Spatial positioning is also sometimes combined with an aggregation operation (see Section 5.4). For example, Wattenberg [Wat06] leverages attribute-based positioning to generate an overview of an aggregated network along two user-defined attributes.

Using spatial positioning for edges is less straightforward, as they need to connect nodes at different positions. One approach is edge bundling, which routes edges that are related according to some metric of similarity in close proximity [Hol06, HvW09] (see the recent survey by Llhuillier et al [LHT17] for details on edge bundling). The similarity metric is frequently based on the topology of the network [KRM*17], so that edges that have a similar region of origin and destination are bundled together. Edge bundling reduces clutter and makes it easier to detect connectivity patterns in the network [PHT15]. A similarity metric can also consider attributes of the edges [PHT15]. Similar edges are then bundled only if they share directionality or specific attributes. Edge bundling methods do not alter the endpoints of an edge, they only alter the routing of the edge between source and target nodes [LHT17].

Figure 8 illustrates the usage of edge bundling in a dataset of flight paths (edges) between hubs (nodes) in France [PHT15]. The routing of similar edges based on their source and target, as well as their direction attribute, simplifies the interpretation of flight paths across the country.

Recommended Usage. Attribute-driven positioning is well suited for cases where the value of a single node attribute or the relationships between two node attributes are the most important feature in a network dataset, but it does not lend itself well to visualizing the topology of the network. Even simple structures such as neighborhoods can be difficult to spot. Complex structures such as paths or clusters can be hidden completely (see Table 2). Unlike attribute-driven faceting, the technique is well suited for quantitative attributes. The technique works mostly for homogeneous net-

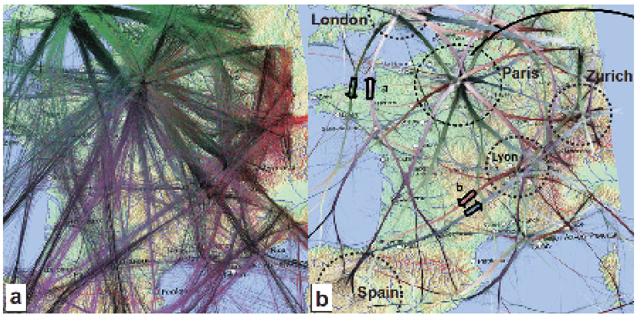
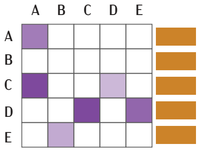


Figure 8: *Attribute-driven edge positioning via edge bundling [PHT15]. (a) Raw edges, colored according to directionality. (b) Edges bundled while taking direction into account. © 2015 IEEE, reprinted with permission.*

works since it relies on common node attributes for positioning. Due to the placement driven by attributes, nodes can occlude each other (although jitter was suggested to address that [BCD*10]), and edge crossings are much more likely than, e.g., in a force-directed layout. Hence, attribute-driven positioning is not well suited for dense networks or for visualizing edge attributes. We recommend attribute-driven positioning for smaller, sparse networks where relationships between node attributes are paramount to the analysis task, and topological features only provide context.

5.1.2. Tabular Layouts

Tabular layouts encompass approaches where nodes and/or links are represented as columns and/or rows of a table. The most well-known of these approaches is the adjacency matrix, but this category also includes Quilts [BDF*10] and BioFabric [Lon12]. A beneficial property of all tabular layouts with respect to multivariate networks analysis is that nodes, and in some cases edges, are in dedicated rows or columns, which can also be used to visualize attributes.

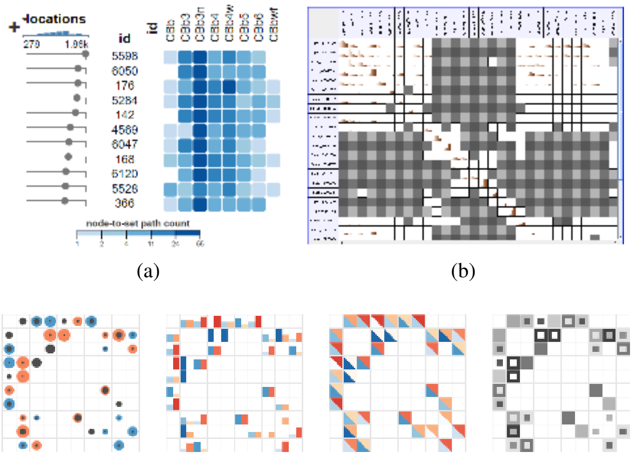


5.1.2.1. Adjacency Matrices. Adjacency matrices encode nodes as rows and columns, whereas the presence/absence of an edge between two nodes is encoded in the cell where the nodes rows and columns intersect. Matrices work for both directed networks, in which case the rows are the source of the edge and the columns the targets (or vice versa), and undirected networks, in which case the encoding is redundant above and below the diagonal of the matrix. Matrices have both favorable and unfavorable properties compared to node-link layouts when analyzing topological features [GFC05, OJK18, OJK17]: matrices are widely acknowledged to be well suited to analyze neighborhoods and clusters (given a suitable matrix seriation) but perform poorly when analyzing paths. For attribute visualization, adjacency matrices have favorable properties for both node and edge attribute visualization.

For *node attributes*, it is easy to juxtapose attribute visualizations with the rows or columns of the matrix [KLS*17, YDGM17], as shown in Figure 9(a). Attribute visualizations can be aligned in a tabular layout so that comparisons between nodes on the same scale are possible. Operations, such as sorting or filtering based on attributes can also be integrated conveniently.

Edge attributes can be encoded in the matrix cells, with a wide range of choices, similar in complexity and expressiveness to on-node encoding in node-link diagrams. The simplest form, beyond a presence/absence mark, is to use color saturation/luminance to show a numerical attribute, such as an edge weight, or to use color hue for a categorical value, such as an edge type [RMF12]. Alternatives encodings include the size of a glyph such as a bar or a circle [DHRL*12]. Multiple attributes can be encoded with more complicated glyphs [EDG*08, ABHR*13], as shown in Figures 9(b) and 9(c). However, the small space available for matrix cells limits how much can be encoded.

One of the key factors influencing the efficacy of understanding the topology of a network is its seriation, i.e., the ordering of the rows and columns. There has been a lot of research on reordering algorithms [BGJ01, DPS02, Lii10, DPS02, MML07, HF06, Fek15]. Different algorithms vary in terms of runtime performance and the types of patterns emphasized [BBHR*16]. Edge and node attributes could conceivably be considered when calculating the distance matrix underlying the seriation algorithms.



(a)

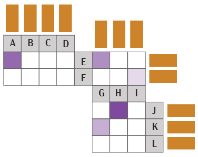
(b)

(c)

Figure 9: *Adjacency matrix approaches. (a) Connectivity Matrix approach where cells represent total path counts between nodes [KLS*17]. Image courtesy of Ethan Kerzner. (b) Aggregated matrix represented by an adjacency matrix where aggregated edge attributes are displayed in the cells [EDG*08]. © 2008 IEEE, reprinted with permission. (c) Comparison of on-edge encoding to adjacency matrix edge attribute encodings [ABHR*13]. © 2013 ACM, reprinted with permission.*

Recommended Usage. Adjacency matrices are one of the most versatile approaches with regard to visualizing multiple attributes for nodes and edges (see Table 2). Alper et al. [ABHR*13] studied the efficacy of edge-attribute encodings by comparing edge-weight encodings on node-link diagrams to different edge-weight encodings in the cells of adjacency matrices. They conclude that in-cell encoding in adjacency matrices outperformed on-edge encoding on node-link diagrams for effectively displaying edge weights for their study subjects.

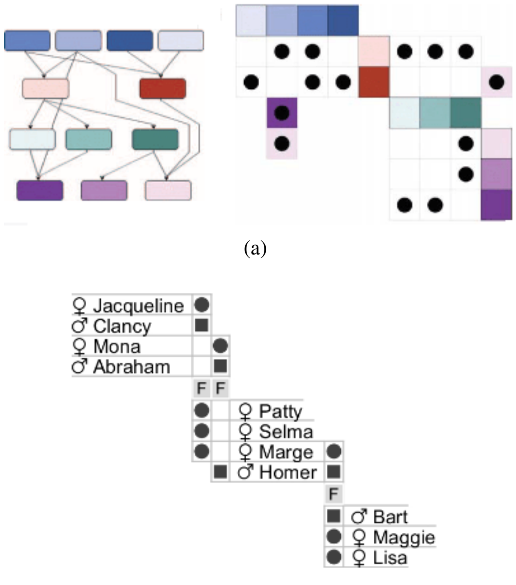
Adjacency matrices require quadratic screen space with respect to the number of nodes; hence, the size of the network that can be visualized without aggregation is limited. Matrices reserve space for every possible edge, and, thus, dense and even completely connected networks are an ideal fit for matrices. Another challenge of matrices is the complexity of choosing the right reordering algorithm, as different algorithms are best suited for revealing different types of patterns. Matrices are well suited for tasks involving all the topological structures we discuss, except for paths, assuming an appropriate seriation method was applied. Overloaded approaches such as visually superimposing the paths directly on the adjacency matrix can aid in path-related tasks [SM07]. Trees and layered networks can technically be visualized with an adjacency matrix, but the sparsity of these networks suggests that they are not a good fit. Overall, we recommend adjacency matrices for smaller, complex and dense networks with rich node and/or edge attributes, for all tasks except for those involving paths.



5.1.2.2. Quilts. A quilt is a tabular layout optimized for layered networks [BW11]. Quilts are similar to an adjacency matrix in that nodes are represented as either rows or columns, and edges are shown in the cells at the intersection of the source and target nodes. The main difference between quilts and adjacency matrices is that nodes are assumed to be partitioned into layers, and no links exist within a given layer. Hence, nodes are not duplicated between rows and columns. Figure 10(a) shows an illustration of quilts. Quilts alternate rows and column-wise arrangement for nodes for successive layers to optimize the use of space. This design works for strictly layered networks, yet in practice, layered datasets often contain some links to nonadjacent layers, known as skip-links, which are difficult to encode for layers separated by an odd number of intermediate layers, as these layers have the same orientation. Watson et al. [WBS*08] address skip links by encoding the targets of skip links using color and labels. An area of application where quilts have been used is genealogies. GeneaQuilts [BDF*10] treats each generation as one layer, and the families founded by that generation as the successive layer.

Quilts have properties similar to matrices with respect to node and edge attribute visualization. Node attribute visualizations can be easily juxtaposed with nodes. For example, GeneaQuilts juxtaposes labels and attributes for the sex of the nodes. Unlike in matrices, nodes are either in rows or columns, and hence comparison between node attributes cannot rely on an aligned scale. Edge attributes in quilts can be visualized using the same methods as for adjacency matrices.

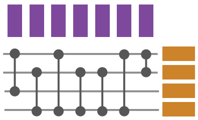
Recommended Usage. Quilts are well suited for layered networks or k-partite networks where all partitions have connections to at most two layers. For these kinds of networks, quilts require less screen-space than matrices and have similar favorable properties in terms of attributes (see Table 2). Links between nonconsecutive layers, however, can be problematic to integrate. Albeit the class of networks suitable for quilts is small, they support all relevant tasks on these well.



(a)

(b)

Figure 10: *The quilts technique. (a) Showing how a small layered network (left) is represented using quilts (right). Links that connect nodes in nonconsecutive layers are represented by colored cells with black dots in them. (b) Simpson family as represented by a quilt [BDF* 10]. © 2010 IEEE, reprinted with permission.*



5.1.2.3. BioFabric. Biofabric is a tabular layout that places each node in a row of the table and draws edges between the nodes in columns [Lon12]. BioFabric has, to our knowledge, not been used to encode attributes, yet the layout of both nodes and edges in unique rows or columns makes it suitable to visualize both node and edge attributes by juxtapositoning attribute visualizations, as for adjacency matrices, but in this case for both edges and nodes.

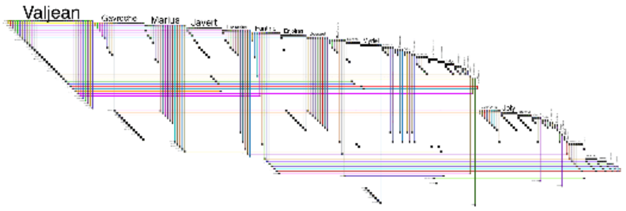


Figure 11: *BioFabric places nodes in rows and edges in columns [Lon12]. Image courtesy of the author.*

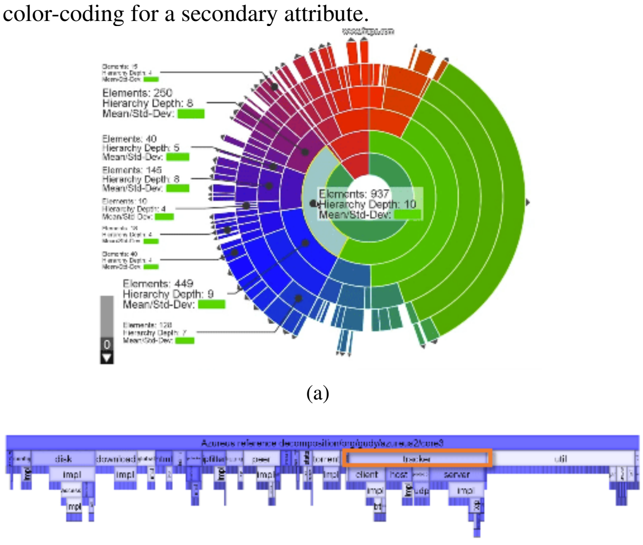
Recommended Usage. Biofabric is unique in that it can be used to visualize rich edge attributes and node attributes at the same time, while also making it possible to align these attribute visualizations on the same scale. It therefore has the potential to visualize large attribute datasets and also heterogeneous node types. Biofabric is about as scalable as an adjacency matrix in terms of the number of nodes, but less scalable with respect to the density of the network. Biofabric is not well studied with respect to users' ability to detect topological features, but it is likely slightly more difficult to discovering neighbors and clusters in Biofabric compared to matrices. Overall, we recommend BioFabric for small, sparse networks with many nodes and rich edge attributes.

5.1.3. Implicit Tree Layouts

Implicit hierarchical layouts are techniques for visualizing trees that rely on node positioning to encode edges. Well-known examples are TreeMaps [JS91, vWvdW99], SunBursts [AH98, SZ00], or Icicle Plots [KL83], but many variations are possible [SHS11]. These layouts excel at visualizing a numerical node attribute as node size, and an additional node attribute as color, but are usually not amenable to encode multiple attributes simultaneously. Since edges are only implicitly encoded, edge attributes cannot be shown.

We distinguish between two types of implicit hierarchical layouts: those that show the whole tree, i.e., the inner nodes and the leaves, and those that show only the leaves of the tree.

5.1.3.1. Implicit Tree Layouts for Inner Nodes and Leaves. Implicit layouts such as Sunburst [AH98, SZ00] and Icicle Plots [KL83, BWB*14] show both the backbone of the tree and the leaves. The hierarchy is encoded by adjacency [SHS11]: a child node is adjacent to its root. The root-child relationship is encoded by the order (inner-to-outer for Sunburst, for example). One numerical attribute is used to encode a size parameter, such as the angle in Sunburst, and the width in Icicle Plots. A typical example used to demonstrate implicit layouts are file systems, where a root folder contains (sub)folders and (nested) files. Files, the leaves, are assigned a width or angle corresponding to their size. Folders are sized according to the sum of the sizes of their subfolders and files. These techniques assume that the size of the primary attribute in the intermediate nodes corresponds to the sum of the size of their leaves, as this is necessary to ensure proper nesting in the layout. Both, intermediate nodes and leaves can use color-coding for a secondary attribute.



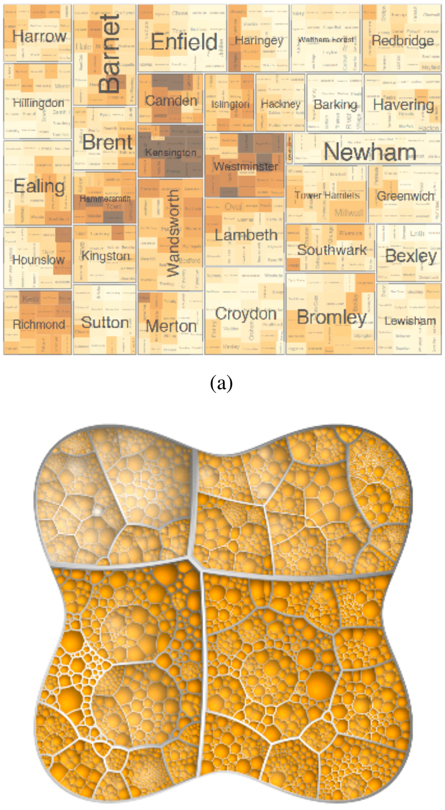
(a)

(b)

Figure 12: *Implicit tree layouts for inner nodes and leaves. (a) The Sunburst layout encodes tree topology by adjacency and a primary (numerical) attribute by angle, a secondary attribute by color [SZ00]. Image courtesy of Alexander Lex. (b) Icicle Plots also encode topology by adjacency, a primary numerical attribute by node size, and a secondary attribute by color [BWB*14]. Image courtesy of the authors.*

Recommended Usage. Implicit tree layouts for inner nodes and leaves are well suited for tree datasets with numerical attributes at the leaves and potentially a secondary attribute at inner nodes and/or leaves. In contrast to leaves-only layouts, these layouts support finding neighbors and paths between nodes well (see Table 2). The layouts scale well with regard to the number of nodes; however, deep hierarchies tend to use a lot of space. Various intuitive interactions are available, such as successively revealing leaves on demand, to deal with deeper trees. Overall, we recommend implicit tree layouts for inner nodes and leaves for tree datasets where one numerical leaf attribute is dominant, and the tree topology plays an important role.

5.1.3.2. Leaf-Centric Implicit Tree Layouts. Leaf-centric implicit layouts are those that only or predominantly allocate screen space for the leaves of a multivariate tree and encode the hierarchy by inclusion/nesting [SHS11]. The classical example of this type of layout is the Treemap, originally proposed by Johnson and Shneiderman [JS91]. Treemaps use the size of a shape to represent a numerical attribute of a leaf node, as shown in Figure 13(a). The leaves belonging to one parent are arranged spatially so that their parent is implicitly represented by a discernible shape, usually a rectangle. Variations of Treemap algorithms exist to either represent only inner nodes implicitly by the arrangement of the nodes, or to draw a border outline for inner nodes. A series of different layout algorithms has been developed to improve the readability of the size of nodes and of the topology [BHvW00, BD05]. A secondary attribute can be encoded as color hue or value/saturation, but other encodings also exist, including glyphs [TS07] or approximate position [SDW09]. Depending on whether inner nodes are shown or not, color or labels can be used to encode an attribute for them. However, in practice, inner nodes are assigned little space, often only a border, making salient encoding of attributes difficult.



(a)

(b)

Figure 13: *Treemap examples. (a) A Treemap layout showing housing data for different boroughs in London. Attributes are encoded by size, color, labels, and approximate position [SDW09]. © 2009 IEEE, reprinted with permission. (b) A voronoi Treemap with thousands of leaves and explicit borders to indicate inner nodes [BD05]. © 2005 IEEE, reprinted with permission.*

Recommended Usage. Leaf-centric implicit layouts are well suited for analyzing trees with an important numerical attribute on the leaves. Because this layout assigns most, if not all, display space to the leaves of a tree, and the hierarchical structure groups small elements, it is more scalable than implicit layouts that also visualize inner nodes (see Table 2). Treemaps have been used to visualize up to a million items [FP02]. Path-related tasks can be difficult because the tree structure is often only implied.

5.2. View Operations

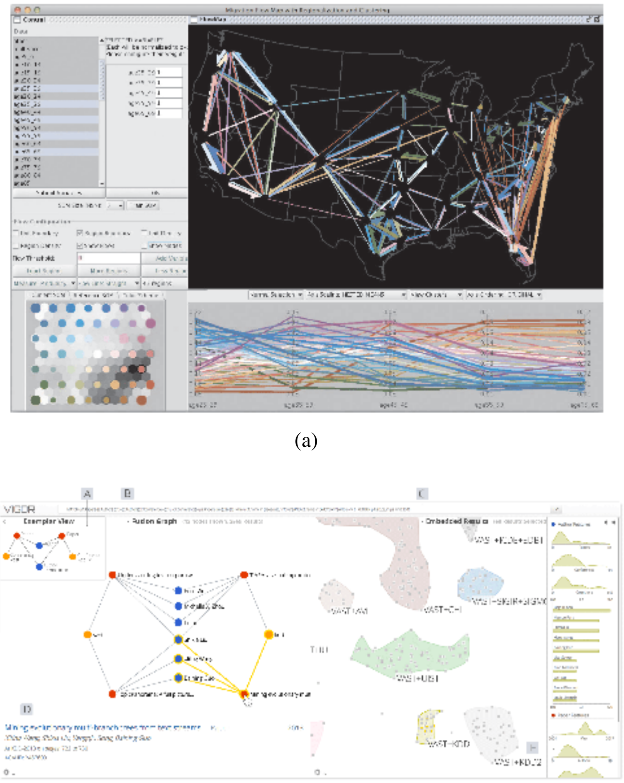
View operations combine existing techniques using several views, a technique commonly referred to as multiple coordinated views (MCVs) [Rob00, WBWK00]. These approaches use separate, dedicated views for the attributes and the topology. Common examples are combinations of node-link diagrams with multidimensional data visualization techniques [SHQ08, LSKS10] or providing a detail view for individual nodes [HB05, TS13].

We distinguish three types of MCVs: juxtaposed, integrated, and overloaded. The juxtaposed and overloaded categories are adopted from the design space of composite visualizations described by Javed and Elmqvist [JE12]. We consider their category of nested views as a variant of our on-node/on-edge encoding category within layouts. We introduce the separate category "integrated", which could be treated as a subtype of juxtaposed, yet is more tightly coupled than typical juxtaposed views, since matches between nodes or edges and their attributes are encoded by their alignment.

5.2.1. Juxtaposed Views

In the context of MVN visualization, juxtaposed views separate the topology visualization from the attribute visualization into two or more views. Relationships between the topology and the attributes are not explicitly encoded and typically are revealed through interaction by linking and brushing. The key benefit of juxtaposed views is that each view can do what it does best: visualize either topology or multivariate data. Another benefit is that standard techniques can be employed. Juxtaposed views are widely used in the MVN visualization literature [AvH04, HB05, SGL08, Guo09, BCD*10, VMCJ10, KAW*14]. Juxtaposed views are also one of the few MVN techniques, other than on-node encoding, supported by the major network visualization tools Cytoscape [SOR*11] and Gephi [BHJ09]. Plaisant et al. [PSM98] showed that while a well-designed juxtaposed layout increases user performance, designing effective juxtaposed views can be a challenging task. Figure 14(a) shows a juxtaposition of a geospatial node-link view to encode the topology of the network, with a parallel coordinates plot to encode attributes [Guo09]. VIGOR [PHE*18], shown in Figure14(b), uses juxtaposition to show two topology-related views and two attribute views.

Recommend Usage. Juxtaposed MCVs are recommended for large networks and/or very large numbers or heterogeneous types of node and link attributes (see Table 2). Since each view can optimize for either topology or attributes without concern for the other, the independent analysis of attributes or topology is generally well supported. Linking and brushing can reintroduce the connection, but only for selected items, and even then matches between specific items in a large brushed set are difficult to identify. Consequently, juxtaposed views do not support the tasks on our topological structures well.



(a)

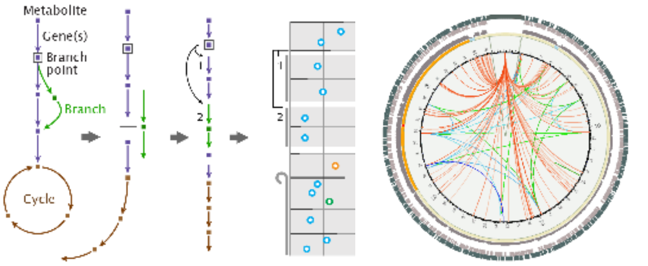
(b)

Figure 14: *Examples of juxtaposed views. (a) A geospatial node-link diagram juxtaposed with two attribute views: a parallel coordinates plot and a self-organizing map [Guo09]. © 2009 IEEE, reprinted with permission. (b) VIGOR [PHE*18] displays two views for the topology of the network along with a juxtaposed attribute view that shows attribute-specific distributions across the network. © 2018 IEEE, reprinted with permission.*

5.2.2. Integrated Views

Unlike juxtaposed views, in integrated views the topology and the attribute visualizations are laid out with the other view in mind. Typically, integrated MCVs have an unambiguous spatial relationship between the topological features and their attributes. This spatial relationship is easily achieved in tabular views, such as adjacency matrices [KLS*17]: since nodes are in rows and columns, tabular attribute visualizations can be aligned to the rows or columns (see Figure 9(a)). Other examples are linear or circular genome views, where edges between genomic regions indicate a variety of relationships [KSB*09, MMP09] (Figure 15(b)).

However, integrated views can also use a node-link layout for visualizing the network topology [MWS*10, PLS*12, DHRL*12, PLS*13, PGS*16, NSL19]. These approaches use a variety of



(a) Pathline (b)

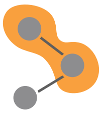
Figure 15: *Integrated Views. (a) Pathline introduces dedicated encodings for cycles and branches to linearize a network and juxtapose it with attribute visualizations [MWS*10]. Image courtesy of Miriah Meyer. (b) Circos plot of structural genome variations from sequencing data [NdTS14]. Figure licensed under CC BY 4.0.*

strategies to make sure attributes and nodes can be juxtaposed. For example, simple networks can be linearized with special encodings [MWS*10] (shown in Figure 15(a)). For more complex networks, one strategy is to ask users to query for paths and integrate attribute views only with these paths [PLS*12, PLS*13, PGS*16], or use interactively extracted spanning trees [NSL19] (see Figure 20).

Integrated views also work well for trees represented as node link diagrams. These views are commonly used for leaf-only attributes, for example in heat maps juxtaposed with a dendrogram [ESBB98, SS02, LNR*08], but integrated layouts that also visualize intermediate nodes also exist [NGCL19].

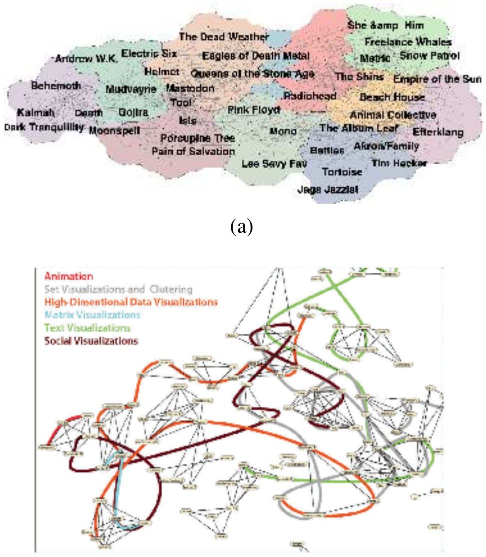
Recommended Usage. Integrated view approaches are exceptionally good at integrating complex attribute vectors of various types with topology, if the topology can be represented sensibly in a linear layout. Integration is easily achieved for tabular approaches such as adjacency matrices, and for specific types of datasets, such as trees and datasets with a natural linear ordering, such as when using genome coordinates. For general networks and node link approaches, integrated views can usually not visualize complex topology, but they can be very useful if the network can be linearized, e.g., using spanning trees or user-selected paths. Compared to juxtaposed views, integrated views excel at tasks related to paths, neighborhoods, and when used with matrices, clusters (see Table 2). One drawback of integrated views is scalability with respect to the number of nodes and density.

5.2.3. Overloaded Views



Overloaded views are those that display two encodings on top of each other by encoding shared properties of nodes by overlaying visual features on the whole view. A classical example, shown in Figure 16(a), is to show a hull around a cluster of nodes (as opposed to coloring the nodes by their cluster membership, which would be on-node encoding). Set memberships are also commonly encoded in a similar way [CPC09, GHK10, ARRC11, HB05, BCD*10, SSK14]. Alternatives to the predominant "hull" approach are curves [ARRC11], as seen in Figure 16(b).

Recommended Usage. Overloaded views are well suited for displaying sets, groupings, or clusters on top of an existing representation of the topology of the network. Overloading works best if the grouped elements are also in spatial proximity in the underlying representation, which is commonly the case when visualizing cluster membership. The major limitation of this approach is the limited number of concurrent attributes it supports. While encoding one or two attributes simultaneously is possible, encoding more than two attributes with overloading can lead to clutter (see Table 2). We hence recommend overloading for the particular use case of visualizing set-memberships or clusters on top of node-link diagrams.



(a)

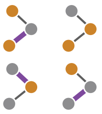
(b)

Figure 16: *Overloaded Views. (a) Group membership encoded on top of the network topology [GHK10]. Image courtesy of S. Kobourov. (b) Group membership encoded as curves [ARRC11]. © 2011 IEEE, reprinted with permission.*

5.3. Layout Operations

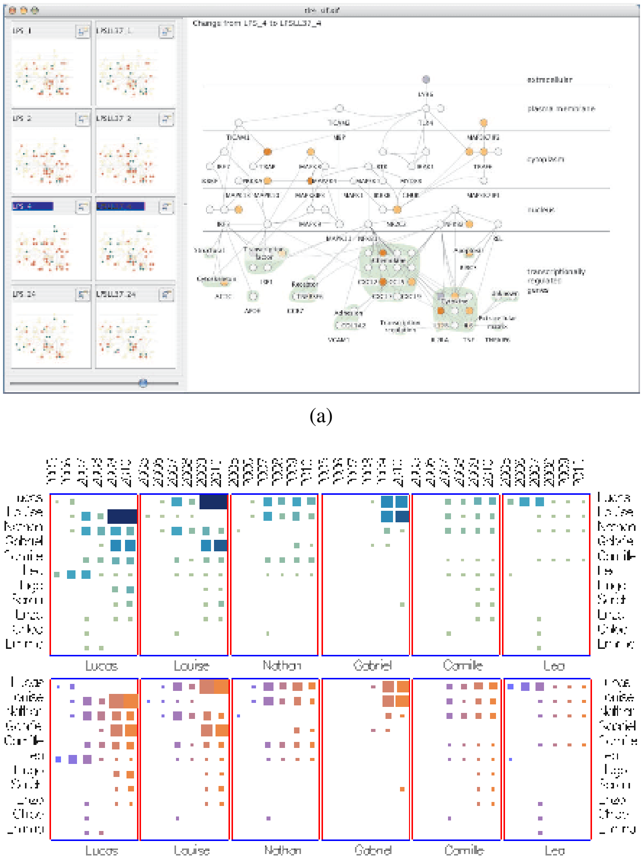
Layout operations combine layouts to create new multivariate network representations. They are distinct from view operations because they are either integrated in a single view or show different facets of the same view using the same layout.

5.3.1. Small Multiples



Small multiples show multiple instances of the same layout under different conditions. Small multiples are commonly used to encode a different attribute for each of the multiple views using on-node/on-edge encoding in node link diagrams [BMGK08, ?, Guo09, vdEvW13] (see Figure 17(a)) or adjacency matrices [BPF14] (Figure 17(b)). Individual views in small multiples can be difficult to see in detail due to their size. A common strategy to avoid this problem is to combine the small multiples display with a large focus view [BMGK08, ?].

Recommended Usage. Small multiples are well suited for comparing several attributes of a small network. The use of a common layout makes it easy to look for attribute variations in specific topological features, such as clusters or paths. A disadvantage of



(a)

(b)

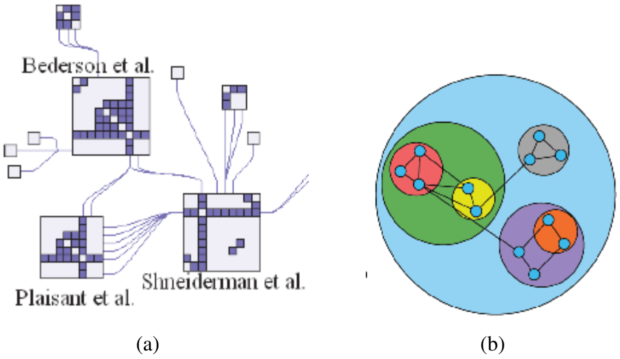
Figure 17: *Small multiples. (a) Small multiples showing eight experimental conditions using on-node encoding (left) with a focus view showing the network enlarged (right) [BMGK08]. © 2008 IEEE, reprinted with permission. (b) Small multiples of a time-varying multivariate network in an adjacency matrix [BPF14]. Image courtesy of the authors.*

small multiples is scalability: by rendering a network layout multiple times, a lot of screen real estate is used even for small networks. Also, attribute comparison between different views can be difficult and require memorization.

5.3.2. Hybrid Layouts

Hybrid layouts combined multiple approaches to laying out network topology. Unlike the techniques using *view operations*, these layouts are characterized by combining different encodings for different portions of the topology of a network. Hybrid approaches are used to either represent networks that have different topological characteristics in different portions of the network [HFM07], or to highlight a certain topological aspect of the graph, such as a hierarchy embedded in a network [FWD*03, AMA08].

NodeTrix [HFM07] is a hybrid node-link/matrix approach (Figure 18(a)). The idea is to use matrices for dense subsets of the network but node-link layouts to connect these subsets or connect to



(a)

(b)

Figure 18: *Hybrid approaches. (a) Hybrid node-link layout and adjacency matrices [HFM07]. © 2007 IEEE, reprinted with permission. (b) Hybrid of a circular Treemap and a node-link layout [AMA08]. © 2008 IEEE, reprinted with permission.*

outliers. Various techniques combine a Treemap encoding for the hierarchical structure of the network with other network encodings [ZMC05, AMA08] (see Figure 18(b)).

Hybrid techniques can also be used to encode multivariate attributes. For example, NodeTrix [HBF08] supports on-node encoding for the node-link/matrix hybrid.

Recommended Usage. Hybrid layouts can be useful for networks with an irregular degree distribution, such as small-world networks. However, the benefits of using an optimal layout for each part of the network has to be weighed against the cost associated with understanding and reading a visualization that combines multiple visualization techniques. Hybrid techniques are not well suited for visualizing multivariate networks, as also encoding attributes increases the complexity of the visualization technique further, and a different encoding schema likely has to be used for each of the techniques employed.

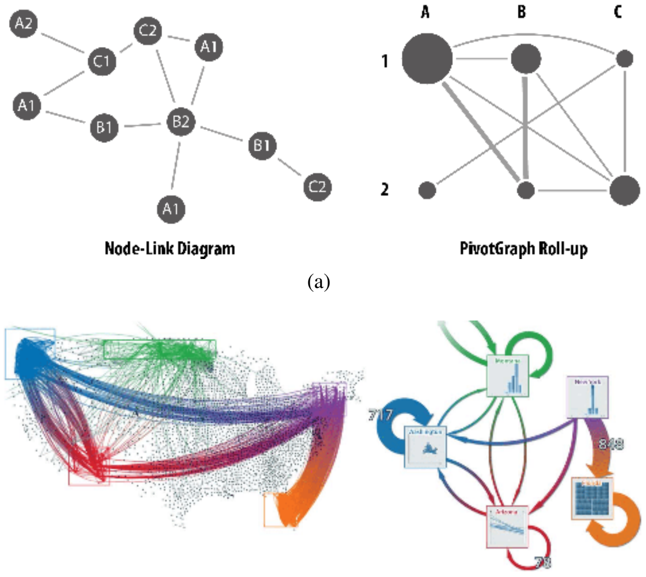
5.4. Data Operations

Multivariate network visualization techniques often rely on data wrangling operations invoked either as a preprocessing step or interactively during the analysis. Here we introduce five types of data wrangling operations that we frequently encountered in the surveyed papers. However, many other operations are conceivable. For a more comprehensive discussion of network wrangling operations, refer to the work by Bigelow et al. [BNML18].

5.4.1. Aggregating Nodes/Edges

Aggregating nodes and edges is a common strategy to increase the scalability of a visualization technique and to summarize nodes and edges with shared characteristic. Aggregation also allows the visualization to more easily support overview tasks on the network. Techniques such as PivotGraph [Wat06], Zame [EDG*08], and Motif Simplications [DS13] employ aggregation strategies to succinctly visualize important properties of large networks.

Aggregation can be based on topological features, such as clusters or motifs [DS13], but also be driven by network attributes.



(a)

(b)

Figure 19: *Aggregation operations. (a) Original node-link network on the left, a network aggregated based on attributes on the right. Figure based on PivotGraph [Wat06]. (b) Geospatial network on the left, an aggregated version with on-node encoding on the right [vdEvW14]. © 2014 IEEE, reprinted with permission.*

PivotGraph, for example, aggregates the nodes by a categorical attribute (node type) and then lays out the aggregate nodes according to their attribute value on a grid-like node-link layout (Figure 19(a)). Van den Elzen and van Wijk [vdEvW14] aggregate nodes based on an attribute, such as a geographic region, and then display the aggregated nodes, their attribute properties, and the relationships within and between the aggregates (Figure 19(b)). Zame [EDG*08] aggregates the nodes as well as the attributes, which are then displayed in an adjacency matrix. An interactive approach to aggregation is used by GraphCharter [TS13], which allows a user to select an attribute of interest and generates an aggregate node with all unique values for that attribute. Graffinity [KLS*17] introduces a novel approach to connectivity exploration by aggregating certain paths that connect pairs of nodes in the network.

Recommended Usage. Aggregation is a key operation to ensure the scalability of MVN visualization and is especially useful when high-level overviews are desired. Aggregation by attributes can reveal interesting relationships between attribute properties and topological features. Aggregation can be achieved in a preprocessing step, yet it is most powerful when invoked interactively.

5.4.2. Querying and Filtering

Filtering a network or querying for a subnetwork allows users to focus on nodes and edges of interest, reducing visual clutter from other nodes and edges. Querying or filtering enables analysts to work with much larger networks than can be reasonably displayed otherwise. Filters and queries can be defined based on topological structures, attributes, or combinations of both [KZA10, PHE*18]. Querying first and expanding a network from a seed is a hallmark approach of many large graph MVN visualization approaches [LPB*06, vHP09, PGS*16, NSL19]. These approaches are often combined with degree-of-interest functions [Fur86, HC04] to further manage network size (see Figure 20).

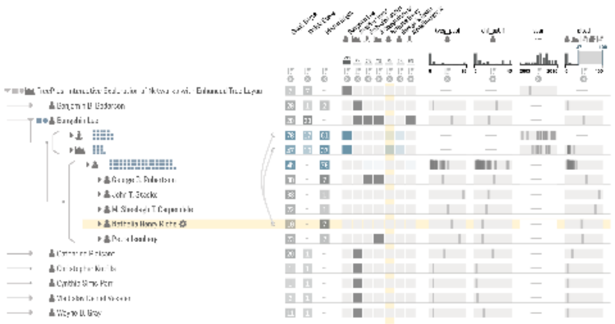


Figure 20: *A query-first approach to network visualization: visualizing a spanning tree originating from a node of interest, combined with a degree-of-interest function to manage large node sets [NSL19]. Image courtesy of the authors.*

Filtering edges is a valuable operation, especially in dense networks. EdgeMaps [DCW11], for example, reveals edges only for selected nodes, whereas the technique by van den Elzen and van Wijk [vdEvW14] displays only edges that connect selected nodes, as shown in Figure 19(b).

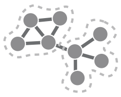
Recommended Usage. Although interactive filtering based on topological and attribute features can and should be available in every MVN visualization approach, query-first strategies follow a different analysis paradigm. The benefit of the latter is that very large networks that cannot be reasonably visualized at once can be investigated with this approach. We recommend a query-first strategy for cases where analysis questions target specific nodes and relationships, rather than overall network patterns.

5.4.3. Deriving New Attributes

Node and edge attributes can be derived either from topological features (e.g., node degree, clusters) or from other attributes. Derived measures can be helpful in, e.g., finding the most connected node in a network or quickly identifying clusters of interest. Once these derived attributes have been computed, they can be visualized just as other attributes. Examples of techniques that rely on derived attributes are Juniper [NSL19], which visualizes node degrees for various subsets (Figure 20), and EdgeMaps [DCW11], which uses a precomputed similarity metric to position the nodes in an attribute-driven node link view.

Recommended Usage. Attribute derivation can be useful to answer specific analysis questions. Derivations that are local to a node, such as determining degree or deriving an attribute based on existing attributes, are easy to realize. However, derivations that rely on topological features, such as clustering, or deriving attributes based on connectivity patterns, require more sophisticated methods.

5.4.4. Clustering

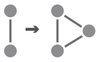


Network clustering is used to group nodes based on similar features. In most cases, clustering of networks is based on network topology. Social network analysis, for example, often employs hierarchical clustering in order to find community structures in the network. Clustering can be used to highlight groups on existing networks, i.e., to create a derived attribute, or as a precursory step to aggregation. Techniques such as Zame [EDG*08], Honeycomb [vHSD*09], and TreeMatrix [RMF12] all rely on hierarchical clustering on the network before visualization. Zame and Honeycomb visualize the clustered network with zoomable adjacency matrices, while TreeMatrix displays the results in a hybrid approach using both Treemaps and adjacency matrices.

Hierarchical algorithms applied to tabular datasets also generate trees encoding the cluster similarity, which is a frequent data type in multivariate tree visualization [ESBB98, SS02].

Recommended Usage. Clustering can be used to create derived attributes, highlighting topological features, to improve layouts, e.g., in matrix seriation or to create groups for faceted layouts, or as a precursor step to aggregation. It is a widely used and versatile operation in multivariate network visualization.

5.4.5. Converting Attributes/Edge to Nodes



Converting attributes or edges to nodes is a useful operation in various scenarios. Many visualization techniques are better suited to visualize node attributes than edge attributes, hence converting an edge to a node with new edges connecting the original nodes is a convenient way to visualize edge attributes [PvW08]. Making an attribute into a node can aid in the analysis of which nodes share that specific attribute [JKZ13].

Recommended Usage. Converting attributes or edges to nodes is a powerful operation in the right circumstances but is also a significant reshaping operation, which risks confusing analysts. Hence, it should be applied with care. We believe that it is most useful as a preprocessing operation in dedicated graph wrangling tools [BNML18].

6. Methodology

This survey is focused on techniques for visualizing multivariate networks. We analyzed these techniques through the study of a corpus of literature composed of two types of papers: those that propose a specific encoding to enable visualization of multivariate networks, which we refer to as *technique* papers; and those that focus on evaluation and application of these techniques.

We developed a typology of MVN visualization techniques through our analysis of the corpus, following the analysis method used by Vehlow et al. [VBW15]. After outlining the scope of techniques we consider, our core analysis activities were to compile a corpus of relevant papers and code each paper according to its described technique, application area, and evaluation method. This section describes our method in detail and provides a list of the categories used to derive the proposed taxonomy.

6.1. Scope

We limit ourselves to techniques that specifically aim to visualize multivariate networks. Techniques that focus on visualizing either only the network topology or attributes are not within the scope for this survey. We consider techniques for all network types, including general (complex) networks, layered networks, k/bi-partite networks, and trees.

6.2. Corpus

We identified candidate papers for the corpus using three sources: (1) all papers published in the VIS and EuroVis conference proceedings since their respective inceptions; (2) papers in the IEEE Digital Library identified with a search for the keywords *network, graph, visualization, multivariate, attributes, faceted, and multidimensional*; and (3) papers that cite, or are cited by, the papers identified in steps 1 and 2, according to Google Scholar. For each of these candidate papers, we manually surveyed its title, keywords, and abstract, and included the paper in the corpus if it proposed a technique, evaluation, or application for visualizing multivariate networks. This process resulted in a corpus of 210 papers published between 1991 and 2018.

6.3. Coding

The first author completed two rounds of coding on the corpus using the MAXQDA Qualitative Data Analysis Software [VER18]. In the first round, we coded papers for their application area and evaluation method, and in the second round we focused on coding the visualization techniques. The application areas were coded using an open-coding approach. The evaluation methods, however, were coded with a closed set derived from the seven guiding scenarios identified by Lam et al. [LBI*11]. This first coding pass allowed us to review the entire body of work and prepared us for the second round of exclusively coding the visualization technique used in each paper.

The aim of the second round of coding was to identify the specific visual encodings employed by each multivariate network technique. We used the open coding method described by Beck et al. [BBDW14], and coded papers for the specific visual encodings used to visualize network structure and associated attributes.

In the second round, we did not code the entire corpus, instead stopping when we reached saturation [OK14]: i.e., we no longer encountered new concepts in any of the categories we were coding for. More specifically, the categories we coded for were visualization techniques and encoding methods. We refer to *techniques* as a visualization idiom such as node-link diagram, adjacency matrix, tree-like layout, and others. Nuances of the technique used were also captured, including encoding methods such as the use of color, a particular network layout algorithm, and ordering for matrices.

In addition to textual codes, the first author also developed a set of *visual codes* that summarized the encountered approaches to encoding both the topology and the attributes of a network. These representations did not aim to faithfully reproduce every nuance of the techniques, but instead supplemented the assigned textual codes with a visual summary of each technique. The need for visual codes arose from the limited ability of textual codes to efficiently capture the ways in which different techniques encoded the topology and the attributes of a network. As a result, the visual codes emerged as a way to capture these aspects and were critical in our analysis and in the development of the multivariate network typology.

The first author categorized the textual and visual codes into 33 distinct techniques for visualizing multivariate networks, grouped by their underlying layout approach: node-link, tabular, implicit, or coordinated views. The first author and one of the senior authors then performed an analysis of these techniques, aided by feedback from a second senior author. This analysis revealed core common encodings among a set of techniques. For example, although PivotGraph [Wat06], GraphDice [BCD*10], and FlowVizMenu [VMCJ10] appear to have unique approaches, they build on a common backbone of positioning the nodes according to attribute values. Similarly, Circos plots [KSB*09] and force-directed layouts share the basics of node-link layouts with edges connecting items.

This analysis process provided us with two insights regarding the similarities and differences of the techniques. First, we found a useful distinction between the layout technique used by an approach, and the data transformations applied to the network prior to visualization. The second insight is that complex layouts are often the result of combining two or more core layouts into a single design. Guided by these insights, we used the technique analysis to develop a typology with four dimensions: layouts, view operations, layout operations, and data operations. The layouts category includes the eight core layout approaches observed in the literature (Figure 1). View operations are those that combine the basic layouts into multiple coordinated views. Layout operations refers to the process of either multiplying a given layout or combining components of two or more layouts into a hybrid layout, such as the NodeTrix design [HFM07]. The last dimension, data operations, encompasses any data processing that is done to the data prior to visualization, such as aggregation and computing new attributes. All four dimensions are discussed in more detail in Section 5.

6.4. Developing the Numerical Guidelines

The process of scoring the techniques (see Table 2) for how well they support certain data types and tasks was done with a 'peer review'-like system among all four authors of the paper. The first phase involved independent scoring by each author for each of the techniques along the 20 dimensions. Once all authors submitted their scores, each author identified their outlier scores and wrote a short justification for that score or modified the score when a mistake was made. A discussion phase allowed us to reach a consensus score for every combination in the table.

The rating of techniques that fall within the view operations, such as overlays and juxtaposed views, required some assumptions in order to grade them consistently across authors. For example, when rating juxtaposed views, we considered an optimized encoding for the topology and attributes in order to focus the score on the juxtaposed aspect of this technique, and not on the choice of layout for the topology or the attributes. The same applied to the overlay category, where the score was given to account for the overlaid aspect of the views, regardless of the specific layout technique used for each component.

7. Applications

MVNs are a prevalent data type in various areas. Our corpus of papers spans applications in several domains, including digital humanities [MLCM16], oceanography [NL15], system analysis [PvW08], and engineering [SIB*11]. The most common application areas we observed were those of social network analysis, biological applications, and software engineering. We discuss the latter three areas below and give a brief overview of others. We do not, however, claim an exhaustive review of application areas since the range of journals and conferences where application papers can be published is too broad for the scope of this survey. A survey of application areas for MVNs, also focusing on social networks, biology, and software engineering, can be found in the book edited by Kerren et al. [KPW14].

7.1. Social Network Analysis

Social networks are a popular and widely available source of data. Social networks are defined as a network whose vertices are social entities such as people or organizations, with edges capturing relationships between them. The field of social network analysis (SNA) is concerned with finding structural properties in the network and analyzing the associated attributes of the nodes and edges [WF94, Sco17]. Social networks often contain a multitude of attributes on both nodes and edges. This large number of node and edge attributes represents one of the major challenges with visualizing social networks. Additional attributes are frequently derived using SNA algorithms, which compute additional topology-related attributes such as degree, centrality, and clustering coefficients.

Visual analytics systems in this area can leverage multiple coordinated views, one for each aspect of the network. Since social networks are commonly categorized as a small world network [Sco17], several systems in this field use adjacency matrices for their ability to support dense and highly connected portions of the network [HF07, HF06, vHSD*09, YEL10]. Examples of such approaches include the work by Henry et al. in both MatLink [HF07] and MatrixExplorer [HF06], which use adjacency matrices in either hybrid or juxtaposed MCV layouts. MatLink leverages an overloaded node-link and matrix layout to address the tradeoffs between using these two opposing layouts for locally dense structures such as those found in social networks. Their overloaded layout consists of an adjacency matrix with edges connecting the nodes overlaid on top to allow for easier path following. This modification is particularly useful in addressing path-related topology tasks, a shortcoming of tabular approaches.

Similar to MatLink, NodeTrix [HFM07] leverages two topology encodings, but in a hybrid layout. Their design displays the network with a node-link layout while replacing dense and highly connected areas with embedded adjacency matrices. NodeTrix also allows the user to encode network attributes for both nodes and edges using the following visual channels: color, transparency, shape size, filled area of the shape, border color, width, and labels.

Despite the prevalence of adjacency matrices for SNA visualization, several MVN approaches use node-link diagrams to represent the topology of the network [HB05, VD04, BCD*10]. GraphDice [BCD*10] displays a grid of small multiples where each multiple contains an attribute-driven node-link with two attributes, thereby giving the user an overview of how several attributes vary throughout the network. Vizster [HB05] displays a node-link view for the topology of the network, linked with a juxtaposed attribute panel that shows details on demand for nodes of interest.

Combining multiple ways of encoding attributes in a single view can also be effective, an example of which can be found in the paper by Ghani et al. [GKL*13]. The system uses coordinated views, one of which is an instance of *parallel node link bands*. This view leverages attributes in three ways: a faceted node-link diagram partitions the nodes into bands according to their attribute values for categories of interest; within each band, the user can order the nodes according to either node or edge attributes; and on-node encoding is used to represent an additional attribute, such as the previously computed degree centrality of each node.

7.2. Biological Applications

Biological networks are another common area of applications for MVN visualizations. Networks in this space are most often characterized by nodes and edges with complex attributes, with several examples in fields such as genetics, cancer research, and systems biology.

Unlike the prevalence of adjacency matrices in social network visualizations, MVNs in biological applications rely heavily on node-link layouts [OFH*11,GOB*10,NGCL19,LPK*13,PLS*12]. These layouts are often optimized for displaying paths, since several tasks in this area relate to understanding how attributes vary along biologically meaningful cascades. As a result, work such as that done by Schreiber et al. [SDMW09] and Karp and Paley [KP94] focus exclusively on providing layout algorithms for node-link diagrams of biological networks that focus on displaying paths in an intuitive manner. These optimized layouts are particularly useful for MVN tasks that target either topology-driven or attribute exploration of biological pathways.

Techniques that focus on path-related tasks include Entourage [LPK*13], Pathline [MWS*10], Pathfinder [PGS*16], and Enroute [PLS*12]. Entourage and Enroute allow the user to highlight a path of interest in a network, which is then juxtaposed with an attribute view to create a single integrated view. In a similar vein, Pathline linearizes a pathway network, which is then juxtaposed with an attribute table.

In a survey of existing techniques for visualizing omics data in systems biology, Gehlenborg et al. [GOB*10] highlight the prevalent use of node-link diagrams, often with on-node encoding or attribute-driven layouts. Among the examples given in their survey is Cerebral [BMGK08], which uses both, small multiples and multiple coordinated views to display the topology and attributes of the network. The node-link view facets the nodes according to their locations in the cell and is linked to a small multiples view and a parallel coordinate view to display the associated attributes. In Lineage [NGCL19], Nobre et al. linearize a node-link representation of genealogies and juxtapose an attribute table to create a single integrated view of the topology and attributes of the network.

New et al. describe an example of linked views that includes an adjacency matrix instead of the more commonly used node-link diagram [NKHC08]. The authors introduce a tilted adjacency matrix that permutes rows and columns of a matrix in such a way as to cluster nonzero elements in blocks along the diagonal.

7.3. Software Engineering

Visualization of MVNs in software engineering is part of the broader area of software visualization [Die07]. Software visualization refers to visual depictions of any component of the software lifecycle, from the source code itself to the associated documentation, mental models, and output data [DT14]. The survey by Diehl and Telea [DT14] gives an overview of multivariate network visualization as applied to software engineering.

Within the context of multivariate networks, software visualization often models networks with nodes representing entities such as files, classes, functions, or other components of software. Edges encode the relationships between these entities and can either represent a hierarchical relationship, such as files in a folder or functions in a class, or simply reflect the association between nodes, such as function calls or data flow between nodes [DT14]. Attributes can be diverse and include computed software metrics such as lines of codes, numbers of classes, number of calls, or runtime in a specific module or function, which make up the multivariate node and edge attributes of these networks [JDK10].

Similar to MVNs in the field of biology, networks in software engineering are often very large and contain several attributes, and therefore requires particular attention to scalability [DT14]. Another defining characteristic of networks in this field is the large variety of attribute types, since nodes and edges can represent diverse entities software components, data, people, files, etc.

Visualizing software engineering data has been done in a myriad of ways including with UML diagrams [SW05], treemaps [RVET14], tabular visualizations [ESS92], icicle plots [CCeT*12], and multiple coordinated views [CCeT*12, RVET14]. Adjacency matrices, though less common, have been used to compare the hierarchies of two different systems [BD13].

UML diagrams are node-link layouts with relevant attributes encoded directly on the node marks in the form of text, glyphs, or small embedded visualizations. The often hierarchical nature of structure data is well suited to the combined use of node-link layout and overloading techniques, an approach optimized to encode group membership or hierarchical relations between elements [BT09].

Treemaps are often used to convey the hierarchical nature of the data, e.g., from packages to classes, to functions, to lower-level control structures. The size of the node represents attributes such as the lines of code underlying the structure. Other attributes in treemaps are usually encoded using color on the treemap nodes, but 3D objects have also been used [WL07].

Another network datatype in the context of software engineering is versioning data. Visualization approaches are used to track the evolution of software components (the nodes) between different versions [TA08].

7.4. Other Application Areas

Other areas where MVN visualization techniques are used include transportation networks, communication networks, and security.

Transportation Networks. In transportation networks, nodes often represent locations, such as countries, cities, or intersections, and edges represent either connectivity between these nodes (roads of flight routes), or actual movement of people or goods between the nodes. Node attributes are commonly properties of the locations, such as the number of inhabitants of a city, edge attributes are frequently distance, travel time, or number of people traveling between locations. Edge attributes are typically encoded using color [HTC09] Due to their geospatial nature, transportation networks are often visualized on maps in the form of node-link diagrams, resulting in a fixed layout of the nodes. This fixed layout exacerbates the problem of edge clutter. Edge bundling techniques can improve readability by aggregating edges with common sources and targets, thus making higher-level flow patterns visible [PHT15]. By aggregating areas, it is possible to reduce the scale of the network and display sophisticated visualizations embedded on the nodes [vdEvW14]. Certain transportation networks, such as subway maps, often do not use precise geolocations and instead preserve only approximate positions and the relative ordering of nodes [Wol07]. An example of attribute-driven positioning used in transportation networks are "isotime flow maps", where distances between nodes encode the travel times between them [ZFA*14].

Communication Networks. Communication networks are concerned with the flow of information between devices or people. Similar to transportation networks, the most prevalent form of MVN visualization within communication networks are node-link diagrams [BFN04, OGK06]. While geolocated node-link representations are common in cases where the precise location of the nodes is of interest [BEW95, Eic96], many communication networks are laid out differently, for example using attribute driven positioning to group similar nodes [BFN04].

Security Networks. The field of security visualization is frequently concerned with the analysis and visualization of security networks, i.e networks aimed at detecting anomalous patterns that may indicate vulnerability and attacks [SSG12]. These networks are often communication networks. The difference to generic communication networks lies mostly in the types of attributes and analysis tasks, which are geared towards capturing unexpected behavior and traffic patterns. The types of visualizations used are varied and include force-directed node-link layouts [PR07, MMK08], node-link layouts as parallel axis [YYT*04], and matrices [KOK05].

In general, we noticed that social network analysis, biological and life science applications, and software engineering use a variety of different encodings, but other domains rely heavily on node-link diagrams. We speculate that this is because visualization research papers are often case studies for the former domains and therefore see a wider variety of techniques.

8. Evaluation

All papers in the corpus were coded for the evaluation methods described in Section 6. An in-depth analysis of how MVN visualization techniques are evaluated is beyond the scope of this survey. Coding evaluation methods, however, allowed us to understand how multivariate network techniques are currently being evaluated, and to, we hope, open up a conversation on whether these methods of evaluation are sufficient and appropriate.

The closed set of evaluation codes we used, and our interpretation of each method are as follows:

- **Use cases** are informal demonstrations of usage, without the involvement of a domain expert or any quantitative measure of the tool's validity.
- **Controlled experiments** are those that control some of the analysis process and include crowd-sourced studies.
- **Case studies** are a form of evaluation that reports in detail on domain experts' use of the proposed tool in the wild.
- **Theoretical evaluation** argues for the efficacy of a proposed technique based on existing visualization principles and a comparison with similar work.
- **Usability evaluation** is aimed at observing how users interact with a tool, either through a set of predefined usability tasks or through observations of free-form usage.
- **Algorithmic evaluation** measures computational performance of a technique.
- **Heuristics evaluations** entail inspecting a system according to a chosen set of relevant heuristics or guidelines.

For each paper, we coded all the evaluation methods used, which ranged from a single approach to up to three different evaluation methods. Our analysis found that the majority of technique papers in our corpus used use cases (63%), followed by controlled experiments (28%), case studies (15%), and usability studies (15%). Less frequently, we found examples of theoretical evaluation (8%) or algorithmic evaluation (4%). We did not encounter heuristic evaluation in our corpus. Also, likely due to our inclusive definition of evaluation, we found that all papers included at least one of these methods.

Use cases were by far the most common approach to evaluation, likely due to the simplicity of conducting them and their usefulness for explaining a new technique. When done well, use cases leveraged real-world data to walk the reader through examples of how a tool can be helpful for data analysis and exploration. A common problem with this approach is the choice of data. Sample datasets such as movie actors and movies, or interactions between the characters in Les Miserables, are readily available and explainable without extensive domain knowledge, but do not provide evidence that the technique would scale well to real-world applications and datasets. With more complex, real-world datasets, use cases provide stronger evidence that the proposed technique could be effective in the wild.

Controlled experiments were also common and were most often conducted in one of two scenarios: (1) comparing a novel MVN approach with existing techniques [LPB*06, JRA09, ARRC11, YDGM17]; or (2) comparing variations of a given design for visualizing MVNs [SSE16, BW11].

Other common types of evaluation encountered were case studies and usability evaluations. The systems evaluated with case studies were often developed in close collaboration with domain experts. These systems are frequently complex and unique so that they cannot be easily compared to other tools.

Usability studies were often used in conjunction with another evaluation approach, such as a usage scenario or a case study. Theoretical evaluations were uncommon, likely due to the challenge of convincing readers of the superiority of the proposed tool through theoretical arguments. Only 4% of the papers were coded as algorithmic evaluations, which is unsurprisingly low given the types of contributions—novel encodings as opposed to novel algorithms—of papers in this survey.

9. General Guidelines for Visualizing Multivariate Networks

In this section, we provide general guidelines on visualizing multivariate networks, complementing our technique specific recommendations given in Section 5.1 and our recommender system on the accompanying website (https://vdl.sci.utah.edu/mvnv/).

One of the major challenges that MVN visualization techniques face is scaling to large networks (Table 2) and large attribute datasets. For large *trees*, treemaps are very well suited, with the caveat that they are tailored to displaying only a few attributes and only for leaves. For other types of networks, MVN techniques typically do not scale to networks with over 1,000 nodes, especially when attempting to show both attributes and network topology. *Data operations* can be very useful to either reduce the subset of the network the user is looking at, through a query-first or filter approach [NSL19, TS13], or to aggregate the network into a more manageable size [Wat06, EDG*08]. These data operations can be done through either preprocessing of the network or interactive operations directly in the visualization.

Juxtaposed views scale to large networks and many attributes, since they allow for a topology-only view that can optimize the layout and an attribute only view well suited to display multivariate data. However, having separate views for topology and attributes comes at the cost that these systems often perform poorly on tasks that require integration between attributes and topology.

A scalable technique with regard to the number of node and edge attributes and node/edge heterogeneity are *integrated views* that align a node/edge with a tabular representation. These techniques are easily combined with all tabular methods or can be combined with node link layouts when leveraging interaction. Along with juxtaposed views, integrated views are suitable for networks with several attributes. The tight visual coupling of the topology and attribute of the network make integrated networks the highest rated techniques for performing MVN tasks on non-tree networks. We recommend adjacency matrices with integrated attribute views for dense networks with many attributes, and node-link networks, combined with integrated attribute views and interactive data operations, for small networks.

Interaction is useful for most techniques, but it is particularly important for node-link representations when combined with juxtaposed or integrated coordinated views. Juxtaposed views rely heavily on interaction to link the components in the topology and attribute views through linking and brushing. Integrated views, on the other hand, are based on a linear ordering of the nodes in the network. This linear ordering can be the product of user selection by picking subsets of the networks, such as spanning trees or paths. For these subsets, integrated views provide excellent integration of topological features and attributes.

10. Conclusions and Future Work

In this paper, we presented the state of the art in visualizing multivariate networks. We provide a data and task abstraction for MVNs, along with a typology of techniques for visualizing this complex data type. The typology was generated through a qualitative analysis of 210 papers published within the visualization research community and lays out the four dimensions along which MVN visualizations can be characterized. Additionally, we provide an overview of common application areas targeted by these published techniques and the evaluation methods used to test them. We believe the typology will be useful for practitioners to understand and choose appropriate multivariate network visualization techniques, and for researchers to identify areas that would benefit from further development.

MVNs are becoming ubiquitous in a broad range of domains, and thus the demand for efficient ways of visualizing them is growing. The use of node-link diagrams to portray networks is by far the most common approach, but the limitation on the number of attributes that can easily be encoded on these structures has led to the popularization of coordinated views and other techniques that are more amenable to encoding multiple attributes, most notably, tabular approaches.

This overview of the field of MVN visualization techniques highlights areas that are ripe for further development, as well as challenges encountered by current techniques.

Tabular Techniques. Node-link layouts have received a lot of attention, but tabular techniques such as adjacency matrices have not been studied in depth in the context of attribute visualization. Tabular approaches have significant potential for visualizing MVNs due to their linear layout of nodes, which allows for trivial juxtaposition of both node and edge attributes.

Edge Attributes. Many solutions are available for visualizing node attributes, but the currently known techniques for visualizing edge attributes are limited. Even matrices, which are considered the best solution for visualizing edge attributes, can only show a handful of edge attributes. Visualizing edge attributes comes with its own set of challenges. Hence, dedicated research in the area is necessary. We identified tabular and integrated approaches as potentially fruitful areas of investigation for edge attributes. BioFabric provides a unique but so far underexplored opportunity for better edge attribute visualization, as edges are represented as columns in BioFabric, which could be leveraged for integration with edge attribute visualizations. Similarly, interactive integrated techniques provide opportunities for better edge attribute visualizations.

Interaction. With the increasing use of web-based systems, interaction has become a ubiquitous element in multivariate network exploration systems, allowing users to quickly and effortlessly filter and analyze the network, as well as modify the encodings used on demand. The study of interaction in MVN visualization techniques is an open opportunity.

User Studies. No studies have rigorously investigated the benefits and trade-offs of different MVN techniques with respect to certain tasks and datasets. We believe that our taxonomy can guide more formal evaluation approaches.

Tools. Hundreds of papers have addressed multivariate network visualization techniques, but the few robust, well-documented, and maintained open-source software packages provide only basic multivariate attribute visualization capabilities. The scientific community needs new and better tools that integrate the scientific state of the art in well-maintained software.

We hope this review allows visualization designers to more easily compare existing approaches to MVN visualization, as well as to leverage our guidelines on which techniques are best suited for given network types and specific exploration tasks. MVN visualization is a constantly evolving field providing visualization researchers rich and interesting open problems.

11. Acknowledgements

The authors thank members of the Visualization Design Lab for their feedback. This work is funded by the National Science Foundation (OAC 1835904, IIS 1751238, and IIS 1350896), the State of Upper Austria (FFG 851460), the Austrian Science Fund (FWF P27975-NBL), and by the Pro2Future Competence Center for Products and Productions of the Future. Pro2Future is funded within the Austrian COMET Program Competence Centers for Excellent Technologies—under the auspices of the Austrian Federal Ministry of Transport, Innovation and Technology, the Austrian Federal Ministry of Science, Research and Economy and of the Provinces of Upper Austria and Styria. COMET is managed by the Austrian Research Promotion Agency FFG.

References

[ABF*07] Ahmed A., Batagelj V., Fu X., Hong S., Merrick D., Mrvar A.: Visualisation and analysis of the internet movie database. In *2007 6th International Asia-Pacific Symposium on Visualization* (Sydney, NSW, Australia, 2007), IEEE, pp. 17–24. doi:10.1109/APVIS.2007.329304. 7

[ABHR*13] Alper B., Bach B., Henry Riche N., Isenberg T., Fekete J.-D.: Weighted graph comparison techniques for brain connectivity analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13* (Paris, France, 2013), ACM Press, p. 483. doi:10.1145/2470654.2470724. 9

[ACJM03] Auber D., Chiricota Y., Jourdan F., Melancon G.: Multiscale visualization of small world networks. In *IEEE Symposium on Information Visualization 2003 (IEEE Cat. No.03TH8714)* (Seattle, WA, USA, 2003), IEEE, pp. 75–81. doi:10.1109/INFVIS.2003.1249011. 6

[AH98] Andrews K., Heidegger H.: Information slices: Visualising and exploring large hierarchies using cascading, semicircular discs. In *InfoVis'98: Proc. IEEE Information Visualization Symposium, Carolina, USA* (1998), pp. 9–12. 10, 11

[AMA08] Archambault D., Munzner T., Auber D.: Grouse-Flocks: Steerable Exploration of Graph Hierarchy Space. *IEEE Transactions on Visualization and Computer Graphics 14*, 4 (2008), 900–913. doi:10.1109/TVCG.2008.34. 14

[ARRC11] Alper B., Riche N., Ramos G., Czerwinski M.: Design Study of LineSets, a Novel Set Visualization Technique. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '11) 17*, 12 (2011), 2259–2267. doi:10.1109/TVCG.2011.186. 13, 19

[AS07] Aris A., Shneiderman B.: Designing Semantic Substrates for Visual Network Exploration. *Information Visualization 6*, 4 (2007), 281–300. doi:10.1057/palgrave.ivs.9500162. 7

[AvH04] Abello J., van Ham F.: Matrix Zoom: A Visual Interface to Semi-External Graphs. In *IEEE Symposium on Information Visualization* (2004), pp. 183–190. doi:10.1109/INFVIS.2004.46. 12

[BBDW14] Beck F., Burch M., Diehl S., Weiskopf D.: The State of the Art in Visualizing Dynamic Graphs. In *Proceedings of the Eurographics Conference on Visualization (EuroVis '14) – State of The Art Reports* (2014). 2, 16

[BBHR*16] Behrisch M., Bach B., Henry Riche N., Schreck T., Fekete J.-D.: Matrix Reordering Methods for Table and Network Visualization. *Computer Graphics Forum 35*, 3 (2016), 693–716. doi:10.1111/cgf.12935. 9

[BCD*10] Bezerianos A., Chevalier F., Dragicevic P., Elmqvist N., Fekete J. D.: GraphDice: A System for Exploring Multivariate Social Networks. *Computer Graphics Forum (EuroVis '10) 29*, 3 (2010), 863–872. doi:10.1111/j.1467-8659.2009.01687.x. 8, 9, 12, 13, 17

[BD05] Balzer M., Deussen O.: Voronoi treemaps. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005.* (2005), pp. 49–56. doi:10.1109/INFVIS.2005.1532128. 11

[BD13] Beck F., Diehl S.: Visual comparison of software architectures. *Information Visualization 12*, 2 (2013), 178–199. doi:10.1177/1473871612455983. 18

[BDF*10] Bezerianos A., Dragicevic P., Fekete J., Bae J., Watson B.: GeneaQuilts: A System for Exploring Large Genealogies. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (2010), 1073–1081. doi:10.1109/TVCG.2010.159. 9, 10

[BETT98] Battista G. D., Eades P., Tamassia R., Tollis I. G.: *Graph Drawing: Algorithms for the Visualization of Graphs*, 1st ed. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1998. 6

[BEW95] Becker R. A., Eick S. G., Wilks A. R.: Visualizing Network Data. *IEEE Transactions on Visualization and Computer Graphics 1*, 1 (1995), 16–28. doi:10.1109/2945.468391. 19

[BFN04] Ball R., Fink G. A., North C.: Home-Centric Visualization of Network Traffic for Security Administration. In *In VizSEC/DMSEC '04: Proceedings of the 2004 ACM Workshop on Visualization And* (2004), ACM Press, pp. 55–64. 19

[BGJ01] Bar-Joseph Z., Gifford D. K., Jaakkola T. S.: Fast optimal leaf ordering for hierarchical clustering. *Bioinformatics 17*, Suppl 1 (2001), S22–S29. doi:10.1093/bioinformatics/17.suppl_1.S22. 9

[BHJ09] Bastian M., Heymann S., Jacomy M.: Gephi: An Open Source Software for Exploring and Manipulating Networks. In *Third International AAAI Conference on Weblogs and Social Media* (2009). 6, 12

[BHvW00] Bruls M., Huizing K., van Wijk J. J.: Squarified Treemaps. In *Data Visualization 2000* (2000), de Leeuw W. C., van Liere R., (Eds.), Eurographics, Springer Vienna, pp. 33–42. 11

[BM13] Brehmer M., Munzner T.: A Multi-Level Typology of Abstract Visualization Tasks. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '13) 19*, 12 (2013), 2376–2385. doi:10.1109/TVCG.2013.124. 3

[BMGK08] Barsky A., Munzner T., Gardy J., Kincaid R.: Cerebral: Visualizing Multiple Experimental Conditions on a Graph with Biological Context. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '08) 14*, 6 (2008), 1253–1260. doi:10.1109/TVCG.2008.117. 7, 13, 14, 18

[BNML18] Bigelow A., Nobre C., Meyer M., Lex A.: Origraph: Interactive Network Wrangling. *arXiv:1812.06337 [physics]* (2018). arXiv:1812.06337. 14, 16

[Bon02] Bonabeau E.: Graph multidimensional scaling with self-organizing maps. *Information Sciences 143*, 1-4 (2002), 159–180. doi:10.1016/S0020-0255(02)00191-3. 8

[BPF14] Bach B., Pietriga E., Fekete J.-D.: Visualizing dynamic networks with matrix cubes. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems - CHI '14* (Toronto, Ontario, Canada, 2014), ACM Press, pp. 877–886. doi:10.1145/2556288.2557010. 13, 14

[BT09] Byelas H., Telea A.: Visualizing Multivariate Attributes on Software Diagrams. In *2009 13th European Conference on Software Maintenance and Reengineering* (2009), pp. 335–338. doi:10.1109/CSMR.2009.17. 18

[BW11] Bae J., Watson B.: Developing and Evaluating Quilts for the Depiction of Large Layered Graphs. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2268–2275. doi:10.1109/TVCG.2011.187. 10, 19

[BWB*14] Beck F., Wiszniewsky F.-J., Burch M., Diehl S., Weiskopf D.: Asymmetric Visual Hierarchy Comparison with Nested Icicle Plots. *Proceedings of the Workshop on Graph Visualization in Practice (GraphVIP '14)* (2014). 11

[CCeT*12] Carr H., Czanner (editors S., Trümper J., Telea A., Döllner J.: ViewFusion: Correlating Structure and Activity Views for Execution Traces. In *Theory and Practice of Computer Graphics* (2012). 18

[CPC09] Collins C., Penn G., Carpendale S.: Bubble Sets: Revealing Set Relations with Isocontours over Existing Visualizations. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '09) 15*, 6 (2009), 1009–1016. doi:10.1109/TVCG.2009.122. 13

[DCW11] Dörk M., Carpendale S., Williamson C.: EdgeMaps: Visualizing explicit and implicit relations. In *IS&T/SPIE Electronic Imaging* (San Francisco Airport, California, USA, 2011), p. 78680G. doi:10.1117/12.872578. 8, 15

[DHRL*12] Dunne C., Henry Riche N., Lee B., Metoyer R., Robertson G.: GraphTrail: Analyzing Large Multivariate, Heterogeneous Networks While Supporting Exploration History. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '12)* (2012), ACM, pp. 1663–1672. doi:10.1145/2207676.2208293. 9, 12

[Die07] Diehl S.: *Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software*. Springer-Verlag, Berlin Heidelberg, 2007. 18

[DPS02] Díaz J., Petit J., Serna M.: A Survey of Graph Layout Problems. *ACM Comput. Surv. 34*, 3 (2002), 313–356. doi:10.1145/568522.568523. 9

[DS13] Dunne C., Shneiderman B.: Motif simplification: Improving network visualization readability with fan, connector, and clique glyphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '13* (Paris, France, 2013), ACM Press, p. 3247. doi:10.1145/2470654.2466444. 6, 14

[DT14] Diehl S., Telea A. C.: Multivariate Networks in Software Engineering. In *Multivariate Network Visualization*, Kerren A., Purchase H. C., Ward M. O., (Eds.), no. 8380 in Lecture Notes in Computer Science. Springer International Publishing, 2014, pp. 13–36. 18

[EDG*08] Elmqvist N., Do T.-N., Goodell H., Henry N., Fekete J.: ZAME: Interactive Large-Scale Graph Visualization. In *Visualization Symposium, 2008. PacificVIS '08. IEEE Pacific* (2008), pp. 215–222. doi:10.1109/PACIFICVIS.2008.4475479. 9, 14, 15, 16, 20

[Eic96] Eick S. T.: Aspects of network visualization. *IEEE Computer Graphics and Applications 16*, 2 (1996), 69–72. doi:10.1109/38.486685. 19

[ESBB98] Eisen M. B., Spellman P. T., Brown P. O., Botstein D.: Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences USA 95*, 25 (1998), 14863–14868. doi:10.1073/pnas.95.25.14863. 13, 16

[ESS92] Eick S., Steffen J., Sumner E.E. J.: Seesoft-a tool for visualizing line oriented software statistics. *IEEE Transactions on Software Engineering 18*, 11 (1992), 957 –968. doi:10.1109/32.177365. 18

[Fek15] Fekete J.-D.: Reorder.js: A JavaScript Library to Reorder Tables and Networks. In *IEEE Symposium on Information Visualization* (Chicago, United States, Oct. 2015), p. 3. 9

[FP02] Fekete J. D., Plaisant C.: Interactive information visualization of a million items. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis '02)* (2002), pp. 117–124. doi:10.1109/INFVIS.2002.1173156. 12

[Fur86] Furnas G. W.: Generalized Fisheye Views. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Boston, MA, USA, 1986), CHI '86, ACM, pp. 16–23. doi:10.1145/22339.22342. 15

[FWD*03] Fekete J.-D., Wang D., Dang N., Aris A., Plaisant C.: Interactive Poster: Overlaying Graph Links on Treemaps. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis '03)* (2003), IEEE, pp. 82–83. 14

[GE11] Ghani S., Elmqvist N.: Improving revisitation in graphs through static spatial features. p. 8. 6

[GFC05] Ghoniem M., Fekete J.-D., Castagliola P.: On the Readability of Graphs Using Node-Link and Matrix-Based Representations: A Controlled Experiment and Statistical Analysis. *Information Visualization 4*, 2 (June 2005), 114 –135. doi:10.1057/palgrave.ivs.9500092. 9

[GHK10] Gansner E. R., Hu Y., Kobourov S.: GMap: Visualizing graphs and clusters as maps. In *2010 IEEE Pacific Visualization Symposium (PacificVis)* (Taipei, Taiwan, Mar. 2010), IEEE, pp. 201–208. doi:10.1109/PACIFICVIS.2010.5429590. 13

[GKL*13] Ghani S., Kwon B. C., Lee S., Yi J. S., Elmqvist N.: Visual Analytics for Multimodal Social Network Analysis: A Design Study with Social Scientists. *IEEE Transactions on Visualization and Computer Graphics (VAST '13) 19*, 12 (2013), 2032–2041. doi:10.1109/TVCG.2013.223. 7, 18

[GOB*10] Gehlenborg N., O'Donoghue S. I., Baliga N. S., Goesmann A., Hibbs M. A., Kitano H., Kohlbacher O., Neuweger H., Schneider R., Tenenbaum D., Gavin A.-C.: Visualization of omics data for systems biology. *Nature Methods 7*, 3 (2010), 56–68. doi:10.1038/nmeth.1436. 6, 18

[Guo09] Guo D.: Flow Mapping and Multivariate Visualization of Large Spatial Interaction Data. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov. 2009), 1041–1048. doi:10.1109/TVCG.2009.143. 7, 8, 12, 13

[HB05] Heer J., Boyd D.: Vizster: Visualizing online social networks. In *IEEE Symposium on Information Visualization, 2005. INFOVIS 2005* (Oct. 2005), pp. 32–39. doi:10.1109/INFVIS.2005.1532126. 4, 6, 12, 13, 17, 18

[HBF08] Henry N., Bezerianos A., Fekete J.: Improving the Readability of Clustered Social Networks using Node Duplication. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (Nov. 2008), 1317–1324. doi:10.1109/TVCG.2008.141. 14

[HC04] Heer J., Card S. K.: DOITrees Revisited: Scalable, Space-constrained Visualization of Hierarchical Data. In *Proceedings of the Working Conference on Advanced Visual Interfaces* (New York, NY, USA, 2004), AVI '04, ACM, pp. 421–424. doi:10.1145/989863.989941. 15

[HF06] Henry N., Fekete J.-D.: MatrixExplorer: A dual-representation system to explore social networks. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 677–684. 4, 9, 17

[HF07] Henry N., Fekete J.-D.: MatLink: Enhanced Matrix Visualization for Analyzing Social Networks. In *Human-Computer Interaction – INTERACT 2007*, Baranauskas C., Palanque P., Abascal J., Barbosa S. D. J., (Eds.), no. 4663 in Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2007, pp. 288–302. 4, 17

[HFM07] Henry N., Fekete J. D., McGuffin M. J.: NodeTrix: A Hybrid Visualization of Social Networks. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '07) 13*, 6 (2007), 1302–1309. doi:10.1109/TVCG.2007.70582. 14, 17

[HIvWF11] Holten D., Isenberg P., van Wijk J. J., Fekete J.: An extended evaluation of the readability of tapered, animated, and textured directed-edge representations in node-link graphs. In *2011 IEEE Pacific Visualization Symposium (PacificVis)* (Mar. 2011), pp. 195–202. doi:10.1109/PACIFICVIS.2011.5742390. 7

[Hol06] Holten D.: Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '06) 12*, 5 (2006), 741–748. doi:10.1109/TVCG.2006.147. 8

[HSS15] Hadlak S., Schumann H., Schulz H.-J.: A Survey of Multi-Faceted Graph Visualization. In *Eurographics Conference on Visualization (EuroVis) - STARs* (2015), Borgo R., Ganovelli F., Viola I., (Eds.), The Eurographics Association. doi:10.2312/eurovisstar.20151109. 2

[HTC09] Hurter C., Tissoires B., Conversy S.: Fromdady: Spreading aircraft trajectories across views to support iterative queries. *Visualization and Computer Graphics, IEEE Transactions on 15*, 6 (2009), 1017–1024. 19

[HvW09] Holten D., van Wijk J. J.: Force-Directed Edge Bundling for Graph Visualization. *Computer Graphics Forum (EuroVis '09) 28*, 3 (2009), 983–990. doi:10.1111/j.1467-8659.2009.01450.x. 8

[JDK10] Jusufi I., Dingjie Y., Kerren A.: The Network Lens: Interactive Exploration of Multivariate Networks Using Visual Filtering. In *Proceedings of the Conference on Information Visualisation* (2010), pp. 35–42. doi:10.1109/IV.2010.15. 7, 18

[JE12] Javed W., Elmqvist N.: Exploring the design space of composite visualization. In *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis '12)* (Mar. 2012), IEEE, pp. 1 –8. doi:10.1109/PacificVis.2012.6183556. 12

[JKS06] Junker B. H., Klukas C., Schreiber F.: VANTED: A system for advanced data analysis and visualization in the context of biological networks. *BMC Bioinformatics 7*, 1 (Mar. 2006), 109. doi:10.1186/1471-2105-7-109. 6, 7

[JKZ13] Jusufi I., Kerren A., Zimmer B.: Multivariate Network Exploration with JauntyNets. In *2013 17th International Conference on Information Visualisation* (July 2013), pp. 19–27. doi:10.1109/IV.2013.3. 16

[JM03] Jankun-Kelly T. J., Ma K.-L.: MoireGraphs: Radial focus+context visualization and interaction for graphs with visual nodes. In *IEEE Symposium on Information Visualization 2003 (IEEE Cat. No.03TH8714)* (Oct. 2003), pp. 59–66. doi:10.1109/INFVIS.2003.1249009. 6, 7

[JRA09] Jern M., Rogstadius J., Åström T.: Treemaps and Choropleth Maps Applied to Regional Hierarchical Statistical Data. In *2009 13th International Conference Information Visualisation* (Barcelona, Spain, July 2009), IEEE, pp. 403–410. doi:10.1109/IV.2009.97. 19

[JS91] Johnson B., Shneiderman B.: Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In *Proceedings of the IEEE Conference on Visualization (Vis '91)* (1991), pp. 284–291. doi:10.1109/VISUAL.1991.175815. 10, 11

[KAW*14] Ko S., Afzal S., Walton S., Yang Y., Chae J., Malik A., Jang Y., Chen M., Ebert D.: Analyzing high-dimensional multivaríate network links with integrated anomaly detection, highlighting and exploration. In *2014 IEEE Conference on Visual Analytics Science and Technology (VAST)* (Oct. 2014), pp. 83–92. doi:10.1109/VAST.2014.7042484. 7, 12

[KL83] Kruskal J. B., Landwehr J. M.: Icicle Plots: Better Displays for Hierarchical Clustering. *The American Statistician 37*, 2 (1983), 162. doi:10.2307/2685881. 10, 11

[KLS*17] Kerzner E., Lex A., Sigulinsky C., Urness T., Jones B., Marc R., Meyer M.: Graffinity: Visualizing Connectivity in Large Graphs. *Computer Graphics Forum 36*, 3 (June 2017), 251–260. doi:10.1111/cgf.13184. 9, 12, 15

[KOK05] Koike H., Ohno K., Koizumi K.: Visualizing cyber attacks using IP matrix. In *IEEE Workshop on Visualization for Computer Security, 2005. (VizSEC 05).* (Oct. 2005), pp. 91–98. doi:10.1109/VIZSEC.2005.1532070. 19

[KP94] Karp P. D., Paley S. M.: Automated Drawing of Metabolic Pathways. In *Processdings of the Conference on Bioinformatics and Genome Research* (1994). 18

[KPW14] Kerren A., Purchase H. C., Ward M. (Eds.): *Multivariate Network Visualization*. No. 8380 in Lecture Notes in Computer Science. Springer, 2014. 1, 2, 17

[KRM*17] Kruiger J. F., Rauber P. E., Martins R. M., Kerren A., Kobourov S., Telea A. C.: Graph Layouts by t-SNE. *Computer Graphics Forum 36*, 3 (June 2017), 283–294. doi:10.1111/cgf.13187. 8

[KSB*09] Krzywinski M., Schein J., Birol I., Connors J., Gascoyne R., Horsman D., Jones S. J., Marra M. A.: Circos: An information aesthetic for comparative genomics. *Genome Research 19*, 9 (2009), 1639–1645. doi:10.1101/gr.092759.109. 8, 12, 17

[KWS*13] Katz Y., Wang E. T., Silterra J., Schwartz S., Wong B., Mesirov J. P., Airoldi E. M., Burge C. B.: Sashimi plots: Quantitative visualization of RNA sequencing read alignments. *arXiv:1306.3466 [q-bio]* (2013). arXiv:1306.3466. 7

[KZA10] Koenig P.-Y., Zaidi F., Archambault D.: Interactive searching and visualization of patterns in attributed graphs. p. 8. 15

[LBI*11] Lam H., Bertini E., Isenberg P., Plaisant C., Carpendale S.: Seven Guiding Scenarios for Information Visualization Evaluation. p. 19. 16

[LHT17] Lhuillier A., Hurter C., Telea A.: State of the Art in Edge and Trail Bundling Techniques. *Computer Graphics Forum 36*, 3 (June 2017), 619–645. doi:10.1111/cgf.13213. 2, 8

[Lii10] Liiv I.: Seriation and Matrix Reordering Methods: An Historical Overview. *Statistical Analysis and Data Mining: The ASA Data Science Journal 3*, 2 (2010), 70–91. doi:10.1002/sam.10071. 9

[LNR*08] Lee B., Nachmanson L., Robertson G., Carlson J., Heckerman D.: Det. (Distance Encoded Tree): A Scalable Visualization Tool for Mapping Multiple Traits to Large Evolutionary Trees. p. 8. 13

[Lon12] Longabaugh W. J.: Combing the hairball with BioFabric: A new approach for visualization of large networks. *BMC Bioinformatics 13*, 1 (Oct. 2012), 275. doi:10.1186/1471-2105-13-275. 9, 10

[LPB*06] Lee B., Parr C., Bederson B., Veksler V., Gray W., Kotfila C., Kotfila C.: TreePlus: Interactive Exploration of Networks with Enhanced Tree Layouts. *IEEE Transactions on Visualization and Computer Graphics 12*, 6 (Nov. 2006), 1414–1426. doi:10.1109/TVCG.2006.106. 4, 15, 19

[LPK*13] Lex A., Partl C., Kalkofen D., Streit M., Gratzl S., Wassermann A. M., Schmalstieg D., Pfister H.: Entourage: Visualizing Relationships between Biological Pathways using Contextual Subsets. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '13) 19*, 12 (2013), 2536–2545. doi:10.1109/TVCG.2013.154. 4, 18

[LPP*06] Lee B., Plaisant C., Parr C. S., Fekete J.-D., Henry N.: Task Taxonomy for Graph Visualization. In *Proceedings of the AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization (BELIV '06)* (2006), pp. 1–5. doi:10.1145/1168149.1168168. 3, 4

[LSKS10] Lex A., Streit M., Kruijff E., Schmalstieg D.: Caleydo: Design and Evaluation of a Visual Analysis Framework for Gene Expression Data in its Biological Context. In *Proceedings of the IEEE Symposium on Pacific Visualization (PacificVis '10)* (2010), IEEE, pp. 57–64. doi:10.1109/PACIFICVIS.2010.5429609. 12

[MLCM16] McCurdy N., Lein J., Coles K., Meyer M.: Poemage: Visualizing the Sonic Topology of a Poem. *IEEE Transactions on Visualization and Computer Graphics 22*, 1 (Jan. 2016), 439–448. doi:10.1109/TVCG.2015.2467811. 17

[MMK08] Mansman F., Meier L., Keim D. A.: Visualization of Host Behavior for Network Security. In *VizSEC 2007*, Goodall J. R., Conti G., Ma K.-L., (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 187–202. doi:10.1007/978-3-540-78243-8_13. 19

[MML07] Mueller C., Martin B., Lumsdaine A.: A comparison of vertex ordering algorithms for large graph visualization. In *2007 6th International Asia-Pacific Symposium on Visualization* (Feb. 2007), pp. 141–148. doi:10.1109/APVIS.2007.329289. 9

[MMP09] Meyer M., Munzner T., Pfister H.: MizBee: A Multiscale Synteny Browser. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '09) 15*, 6 (2009), 897–904. doi:10.1109/TVCG.2009.167. 8, 12

[MWS*10] Meyer M., Wong B., Styczynski M., Munzner T., Pfister H.: Pathline: A Tool For Comparative Functional Genomics. *Computer Graphics Forum (EuroVis '10) 29*, 3 (2010), 1043–1052. doi:10.1111/j.1467-8659.2009.01710.x. 4, 12, 13, 18

[NdTS14] Naquin D., d'Aubenton-Carafa Y., Thermes C., Silvain M.: CIRCUS: A package for Circos display of structural genome variations from paired-end and mate-pair sequencing data. *BMC Bioinformatics 15*, 1 (Dec. 2014). doi:10.1186/1471-2105-15-198. 13

[NGCL19] Nobre C., Gehlenborg N., Coon H., Lex A.: Lineage: Visualizing Multivariate Clinical Data in Genealogy Graphs. *Transaction on Visualization and Computer Graphics 25*, 3 (2019), 1543 – 1558. doi:10.1109/TVCG.2018.2811488. 13, 18

[NJBJec] Nielsen C. B., Jackman S. D., Birol I., Jones S. J. M.: ABySS-Explorer: Visualizing genome sequence assemblies. *IEEE transactions on visualization and computer graphics 15*, 6 (2009 Nov-Dec), 881–888. doi:10.1109/TVCG.2009.116. 7

[NKHC08] New J., Kendall W., Huang J., Chesler E.: Dynamic Visualization of Coexpression in Systems Genetics Data. *IEEE Transactions on Visualization and Computer Graphics 14*, 5 (Sept. 2008), 1081–1095. doi:10.1109/TVCG.2008.61. 18

[NL15] Nobre C., Lex A.: OceanPaths: Visualizing Multivariate Oceanography Data. In *Eurographics Conference on Visualization (EuroVis '15) - Short Papers* (2015), The Eurographics Association. doi:10.2312/eurovisshort.20151124. 17

[NPA*09] Neuweger H., Persicke M., Albaum S. P., Bekel T., Dondrup M., Hüser A. T., Winnebald J., Schneider J., Kalinowski J., Goesmann A.: Visualizing post genomics data-sets on customized pathway maps by ProMeTra – aeration-dependent gene expression and metabolism of Corynebacterium glutamicum as an example. *BMC Systems Biology 3*, 1 (2009), 82. doi:10.1186/1752-0509-3-82. 6

[NSL19] Nobre C., Streit M., Lex A.: Juniper: A Tree+Table Approach to Multivariate Graph Visualization. *Transaction on Visualization and Computer Graphics (InfoVis '18) 25*, 1 (2019). doi:10.1109/TVCG.2018.2865149. 4, 12, 13, 15, 20

[OFH*11] Oeltze S., Freiler W., Hillert R., Doleisch H., Preim B., Schubert W.: Interactive, Graph-based Visual Analysis of High-dimensional, Multi-parameter Fluorescence Microscopy Data in Toponomics. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (Dec. 2011), 1882–1891. doi:10.1109/TVCG.2011.217. 18

[OGK06] Oberheide J., Goff M., Karir M.: Flamingo: Visualizing Internet Traffic. In *2006 IEEE/IFIP Network Operations and Management Symposium NOMS 2006* (Apr. 2006), pp. 150–161. doi:10.1109/NOMS.2006.1687547. 19

[OJK17] Okoe M., Jianu R., Kobourov S.: Revisited Network Representations. *25th Symposium on Graph Drawing* (2017), 16. 9

[OJK18] Okoe M., Jianu R., Kobourov S. G.: Node-link or Adjacency Matrices: Old Question, New Insights. *IEEE Transactions on Visualization and Computer Graphics* (2018), 1–1. doi:10.1109/TVCG.2018.2865940. 9

[OK14] Olson J. S., Kellogg W. A.: *Ways of Knowing in HCI*. Springer Publishing Company, Incorporated, May 2014. 16

[PGS*16] Partl C., Gratzl S., Streit M., Wassermann A. M., Pfister H., Schmalstieg D., Lex A.: Pathfinder: Visual Analysis of Paths in Graphs. *Computer Graphics Forum (EuroVis '16) 35*, 3 (2016), 71–80. doi:10.1111/cgf.12883. 4, 12, 13, 15, 18

[PHE*18] Pienta R., Hohman F., Endert A., Tamersoy A., Roundy K., Gates C., Navathe S., Chau D. H.: VIGOR: Interactive Visual Exploration of Graph Query Results. *IEEE Transactions on Visualization and Computer Graphics 24*, 1 (Jan. 2018), 215–225. doi:10.1109/TVCG.2017.2744898. 12, 15

[PHT15] Peysakhovich V., Hurter C., Telea A.: Attribute-driven edge bundling for general graphs with applications in trail analysis. In *2015 IEEE Pacific Visualization Symposium (PacificVis)* (Apr. 2015), pp. 39–46. doi:10.1109/PACIFICVIS.2015.7156354. 8, 19

[PLS*12] Partl C., Lex A., Streit M., Kalkofen D., Kashofer K., Schmalstieg D.: enRoute: Dynamic Path Extraction from Biological Pathway Maps for In-Depth Experimental Data Analysis. In *Proceedings of the IEEE Symposium on Biological Data Visualization (BioVis '12)* (2012), pp. 107–114. doi:10.1109/BioVis.2012.6378600. 4, 12, 13, 18

[PLS*13] Partl C., Lex A., Streit M., Kalkofen D., Kashofer K., Schmalstieg D.: enRoute: Dynamic Path Extraction from Biological Pathway Maps for Exploring Heterogeneous Experimental Datasets. *BMC Bioinformatics 14*, Suppl 19 (2013), S3. doi:10.1186/1471-2105-14-S19-S3. 12, 13

[PLS*14] Partl C., Lex A., Streit M., Strobelt H., Wassermann A. M., Pfister H., Schmalstieg D.: ConTour: Data-Driven Exploration of Multi-Relational Datasets for Drug Discovery. *IEEE Transactions on Visualization and Computer Graphics (VAST '14) 20*, 12 (2014), 1883–1892. doi:10.1109/TVCG.2014.2346752. 7, 8

[PPS14] Pretorius A. J., Purchase H. C., Stasko J. T.: Tasks for Multivariate Network Analysis. In *Multivariate Network Visualization*, Kerren A., Purchase H. C., Ward M. O., (Eds.), no. 8380 in Lecture Notes in Computer Science. Springer International Publishing, Jan. 2014, pp. 77–95. 3

[PR07] Pearlman J., Rheingans P.: Visualizing Network Security Events Using Compound Glyphs from a Service-Oriented Perspective. In *VizSEC* (2007). doi:10.1007/978-3-540-78243-8_9. 19

[PSM98] Plaisant C., Shneiderman B., Mushlin R.: An information architecture to support the visualization of personal histories. *Information Processing & Management 34*, 5 (Sept. 1998), 581–597. doi:10.1016/S0306-4573(98)00024-7. 12

[PvW08] Pretorius A. J., van Wijk J. J.: Visual Inspection of Multivariate Graphs. *Computer Graphics Forum (EuroVis '08) 27*, 3 (2008), 967–974. doi:10.1111/j.1467-8659.2008.01231.x. 7, 8, 16, 17

[RMF12] Rufiange S., McGuffin M. J., Fuhrman C. P.: TreeMatrix: A Hybrid Visualization of Compound Graphs. *Computer Graphics Forum 31*, 1 (2012), 89–101. doi:10.1111/j.1467-8659.2011.02087.x. 9, 16

[RMS*11] Rodrigues E. M., Milic-Frayling N., Smith M., Shneiderman B., Hansen D.: Group-in-a-Box Layout for Multi-faceted Analysis of Communities. In *2011 IEEE Third Int'l Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third Int'l Conference on Social Computing* (Boston, MA, USA, Oct. 2011), IEEE, pp. 354–361. doi:10.1109/PASSAT/SocialCom.2011.139. 7

[Rob00] Roberts J. C.: Multiple-View and Multiform Visualization. In *Proceedings of the SPIE Conference on Visualization and Data Analysis (VDA '02)* (2000), vol. 3960, SPIE, pp. 176–185. 12

[RVET14] Reniers D., Voinea L., Ersoy O., Telea A.: The Solid* toolset for software visual analytics of program structure and metrics comprehension: From research prototype to product. *Science of Computer Programming 79* (Jan. 2014), 224–240. doi:10.1016/j.scico.2012.05.002. 18

[SA06] Shneiderman B., Aris A.: Network Visualization by Semantic Substrates. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (Sept. 2006), 733–740. doi:10.1109/TVCG.2006.166. 3, 7, 8

[Sch11] Schulz H.-J.: Treevis.net: A Tree Visualization Reference. *IEEE Computer Graphics and Applications 31*, 6 (2011), 11–15. 2

[Sco17] Scott J.: *Social Network Analysis*. SAGE, Feb. 2017. 17

[SDMW09] Schreiber F., Dwyer T., Marriott K., Wybrow M.: A generic algorithm for layout of biological networks. *BMC Bioinformatics 10*, 1 (Nov. 2009), 375. doi:10.1186/1471-2105-10-375. 18

[SDW09] Slingsby A., Dykes J., Wood J.: Configuring Hierarchical Layouts to Address Research Questions. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov. 2009), 977–984. doi:10.1109/TVCG.2009.128. 11

[SGL08] Stasko J., Görg C., Liu Z.: Jigsaw: Supporting Investigative Analysis through Interactive Visualization. *Information Visualization 7*, 2 (June 2008), 118–132. doi:10.1057/palgrave.ivs.9500180. 12

[SHQ08] Shannon R., Holland T., Quigley A.: *Multivariate Graph Drawing Using Parallel Coordinate Visualisations*. Tech. rep., University of St Andrews, 2008. 12

[SHS11] Schulz H.-J., Hadlak S., Schumann H.: The Design Space of Implicit Hierarchy Visualization: A Survey. *IEEE Transactions on Visualization and Computer Graphics 17*, 4 (2011), 393–411. doi:10.1109/TVCG.2010.79. 2, 10, 11

[SIB*11] Sedlmair M., Isenberg P., Baur D., Mauerer M., Pigorsch C., Butz A.: Cardiogram: Visual analytics for automotive engineers. In *Proceedings of the 2011 Annual Conference on Human Factors in Computing Systems - CHI '11* (Vancouver, BC, Canada, 2011), ACM Press, p. 1727. doi:10.1145/1978942.1979194. 17

[SJUS08] Schulz H.-J., John M., Unger A., Schumann H.: Visual Analysis of Bipartite Biological Networks. In *Proceedings of the Eurographics Workshop on Visual Computing for Biomedicine (VCBM '08)* (2008), pp. 135–142. doi:10.2312/VCBM/VCBM08/135-142. 7, 8

[SM07] Shen Z., Maz K.-L.: Path Visualization for Adjacency Matrices. In *Proceedings of the 9th Joint Eurographics / IEEE VGTC Conference on Visualization* (Aire-la-Ville, Switzerland, Switzerland, 2007), EUROVIS'07, Eurographics Association, pp. 83–90. doi:10.2312/VisSym/EuroVis07/083-090. 9

[SNHS13] Schulz H.-J., Nocke T., Heitzler M., Schumann H.: A Design Space of Visualization Tasks. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2366–2375. doi:10.1109/TVCG.2013.120. 3

[SOR*11] Smoot M. E., Ono K., Ruscheinski J., Wang P.-L., Ideker T.: Cytoscape 2.8: New features for data integration and network visualization. *Bioinformatics 27*, 3 (Jan. 2011), 431–432. doi:10.1093/bioinformatics/btq675. 6, 12

[SS02] Seo J., Shneiderman B.: Interactively Exploring Hierarchical Clustering Results. *Computer 35*, 7 (2002), 80–86. doi:10.1109/MC.2002.1016905. 13, 16

[SS06] Schulz H., Schumann H.: Visualizing Graphs - A Generalized View. In *Tenth International Conference on Information Visualisation (IV'06)* (July 2006), pp. 166–173. doi:10.1109/IV.2006.130. 6

[SSE16] Schöffel S., Schwank J., Ebert A.: A User Study on Multivariate Edge Visualizations for Graph-Based Visual Analysis Tasks. In *2016 20th International Conference Information Visualisation (IV)* (July 2016), pp. 165–170. doi:10.1109/IV.2016.41. 7, 19

[SSG12] Shiravi H., Shiravi A., Ghorbani A. A.: A Survey of Visualization Systems for Network Security. *IEEE Transactions on Visualization and Computer Graphics 18*, 8 (Aug. 2012), 1313–1329. doi:10.1109/TVCG.2011.144. 19

[SSK14] Saket B., Simonetto P., Kobourov S.: Group-Level Graph Visualization Taxonomy. *arXiv:1403.7421 [cs]* (Mar. 2014). arXiv:1403.7421. 13

[SSSE16] Schwank J., Schöffel S., Stärz J., Ebert A.: Visualizing Uncertainty of Edge Attributes in Node-Link Diagrams. In *2016 20th International Conference Information Visualisation (IV)* (July 2016), pp. 45–50. doi:10.1109/IV.2016.19. 7

[SW05] Sun D., Wong K.: On evaluating the layout of UML class diagrams for program comprehension. In *13th International Workshop on Program Comprehension (IWPC'05)* (May 2005), pp. 317–326. doi:10.1109/WPC.2005.26. 18

[SZ00] Stasko J., Zhang E.: Focus+Context Display and Navigation Techniques for Enhancing Radial, Space-Filling Hierarchy Visualizations. In *Proceedings of the IEEE Symposium on Information Vizualization (InfoVis '00)* (2000), IEEE Computer Society Press, pp. 57–65. doi:10.1109/INFVIS.2000.885091. 10, 11

[TA08] Telea A., Auber D.: Code Flows: Visualizing Structural Evolution of Source Code. *Computer Graphics Forum (EuroVis '08) 27*, 3 (2008), 831–838. doi:10.1111/j.1467-8659.2008.01214.x. 18

[TS07] Tu Y., Shen H.: Visualizing Changes of Hierarchical Data using Treemaps. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (Nov. 2007), 1286–1293. doi:10.1109/TVCG.2007.70529. 11

[TS13] Tu Y., Shen H. W.: GraphCharter: Combining browsing with query to explore large semantic graphs. In *2013 IEEE Pacific Visualization Symposium (PacificVis)* (Feb. 2013), pp. 49–56. doi:10.1109/PacificVis.2013.6596127. 4, 12, 15, 20

[VBW15] Vehlow C., Beck F., Weiskopf D.: The State of the Art in Visualizing Group Structures in Graphs. *Eurographics Conference on Visualization (EuroVis) - STARs* (2015). doi:10.2312/eurovisstar.20151110. 2, 16

[VD04] Viégas F. B., Donath J.: Social Network Visualization: Can We Go Beyond the Graph? p. 4. 17

[vdEvW13] van den Elzen S., van Wijk J. J.: Small Multiples, Large Singles: A New Approach for Visual Data Exploration. *Computer Graphics Forum (EuroVis '13) 32*, 3pt2 (2013), 191–200. doi:10.1111/cgf.12106. 13

[vdEvW14] van den Elzen S., van Wijk J. J.: Multivariate Network Exploration and Presentation: From Detail to Overview via Selections and Aggregations. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '14) 20*, 12 (2014), 2310–2319. doi:10.1109/TVCG.2014.2346441. 7, 8, 15, 19

[VER18] Verbi Software: MAXQDA Analytics Pro, 2018. 16

[vHP09] van Ham F., Perer A.: "Search, Show Context, Expand on Demand": Supporting Large Graph Exploration with Degree-of-Interest. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (Nov. 2009), 953–960. doi:10.1109/TVCG.2009.108. 4, 15 [vHSD*09] van Ham F., Schulz H.-J., Dimicco J. M., van Ham F., Schulz H.-J., Dimicco J. M.: Honeycomb: Visual Analysis of Large Scale Social Networks. In *Human-Computer Interaction – INTERACT 2009* (Berlin, Heidelberg, 2009), Oestreicher L., Palanque P., Prates R. O., Winckler M., (Eds.), vol. 5727, Springer Berlin Heidelberg, pp. 429–442. doi:10.1007/978-3-642-03658-3_47. 4, 16, 17

[vLKS*11] von Landesberger T., Kuijper A., Schreck T., Kohlhammer J., van Wijk J., Fekete J.-D., Fellner D.: Visual Analysis of Large Graphs: State-of-the-Art and Future Research Challenges. *Computer Graphics Forum 30*, 6 (2011), 1719–1749. doi:10.1111/j.1467-8659.2011.01898.x. 2

[VMCJ10] Viau C., McGuffin M. J., Chiricota Y., Jurisica I.: The FlowVizMenu and Parallel Scatterplot Matrix: Hybrid Multidimensional Visualizations for Network Exploration. *IEEE Transactions on Visualization and Computer Graphics (InfoVis '10) 16*, 6 (2010), 1100–1108. doi:10.1109/TVCG.2010.205. 8, 12, 17

[VPF06] Valiati E. R. A., Pimenta M. S., Freitas C. M. D. S.: A taxonomy of tasks for guiding the evaluation of multidimensional visualizations. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors Novel Evaluation Methods for Information Visualization - BELIV '06* (Venice, Italy, 2006), ACM Press, p. 1. doi:10.1145/1168149.1168169. 3

[vWvdW99] van Wijk J. J., van de Wetering H.: Cushion treemaps: Visualization of hierarchical information. In *In Proceedings of the IEEE Symposium on Information Visualization (InfoVis99)* (1999), pp. 73–78. 10

[Wat06] Wattenberg M.: Visual Exploration of Multivariate Graphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New York, NY, USA, 2006), CHI '06, ACM, pp. 811–819. doi:10.1145/1124772.1124891. 8, 14, 15, 17, 20

[WBS*08] Watson B., Brink D., Stallman M., Devajaran R., Rhyne T.-M., Patel H.: Visualizing very large layered graphs with quilts. p. 8. 10

[WBWK00] Wang Baldonado M. Q., Woodruff A., Kuchinsky A.: Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces - AVI '00* (Palermo, Italy, 2000), ACM Press, pp. 110–119. doi:10.1145/345513.345271. 12

[WC03] Wang X. F., Chen G.: Complex networks: Small-world, scale-free and beyond. *IEEE Circuits and Systems Magazine 3*, 1 (2003), 6–20. doi:10.1109/MCAS.2003.1228503. 2

[WF94] Wasserman S., Faust K.: *Social Network Analysis: Methods and Applications*. Cambridge University Press, Nov. 1994. 17

[WL07] Wettel R., Lanza M.: Visualizing Software Systems as Cities. In *4th Ieee International Workshop on Visualizing Software for Understanding and Analysis* (2007), Society Press, pp. 92–99. 18

[Wol07] Wolff A.: Drawing Subway Maps: A Survey. *Informatik - Forschung und Entwicklung 22* (2007), 23–44. doi:10.1007/s00450-007-0036-y. 19

[WvHKR08] Westenberg M. A., van Hijum S. A. F. T., Kuipers O. P., Roerdink J. B. T. M.: Visualizing Genome Expression and Regulatory Network Dynamics in Genomic and Metabolic Context. *Computer Graphics Forum 27*, 3 (May 2008), 887–894. doi:10.1111/j.1467-8659.2008.01221.x. 7

[XRP*12] Xu K., Rooney C., Passmore P., Ham D., Nguyen P. H.: A User Study on Curved Edges in Graph Visualization. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (Dec. 2012), 2449–2456. doi:10.1109/TVCG.2012.189. 7

[YDGM17] Yang Y., Dwyer T., Goodwin S., Marriott K.: Many-to-Many Geographically-Embedded Flow Visualisation: An Evaluation. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (Jan. 2017), 411–420. doi:10.1109/TVCG.2016.2598885. 9, 19

[YEL10] Yi J. S., Elmqvist N., Lee S.: TimeMatrix: Analyzing Temporal Social Networks Using Interactive Matrix-Based Visualizations. *International Journal of Human-Computer Interaction 26*, 11-12 (Nov. 2010), 1031–1051. doi:10.1080/10447318.2010.516722. 17

[YYT*04] Yin X., Yurcik W., Treaster M., Li Y., Lakkaraju K.: VisFlowConnect: Netflow visualizations of link relationships for security situational awareness. In *Proceedings of the 2004 ACM Workshop on Visualization and Data Mining for Computer Security - VizSEC/DMSEC '04* (Washington DC, USA, 2004), ACM Press, p. 26. doi:10.1145/1029208.1029214. 19

[ZFA*14] Zeng W., Fu C., Arisona S. M., Erath A., Qu H.: Visualizing Mobility of Public Transportation System. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (Dec. 2014), 1833–1842. doi:10.1109/TVCG.2014.2346893. 19

[ZMC05] Zhao S., McGuffin M., Chignell M.: Elastic hierarchies: Combining treemaps and node-link diagrams. In *Proceedings of the IEEE Symposium on Information Visualization (InfoVis '05)* (2005), IEEE Computer Society Press, pp. 57–64. doi:10.1109/INFVIS.2005.1532129. 14