DOI: 10.1111/cgf.12872

COMPUTER GRAPHICS *forum*
Volume 36 (2017), number 6 pp. 201–225

Visualizing Group Structures in Graphs: A Survey

Corinna Vehlow, Fabian Beck and Daniel Weiskopf

VISUS, University of Stuttgart, Germany
{corinna.vehlow, fabian.beck, daniel.weiskopf}@visus.uni-stuttgart.de

Abstract
Graph visualizations encode relationships between objects. Abstracting the objects into group structures provides an overview of the data. Groups can be disjoint or overlapping, and might be organized hierarchically. However, the underlying graph still needs to be represented for analyzing the data in more depth. This work surveys research in visualizing group structures as part of graph diagrams. A particular focus is the explicit visual encoding of groups, rather than only using graph layout to indicate groups implicitly. We introduce a taxonomy of visualization techniques structuring the field into four main categories: visual node attributes vary properties of the node representation to encode the grouping, juxtaposed approaches use two separate visualizations, superimposed techniques work with two aligned visual layers, and embedded visualizations tightly integrate group and graph representation. The derived taxonomies for group structure and visualization types are also applied to group visualizations of edges. We survey group-only, group–node, group–edge and group–network tasks that are described in the literature as use cases of group visualizations. We discuss results from evaluations of existing visualization techniques as well as main areas of application. Finally, we report future challenges based on interviews we conducted with leading researchers of the field.

Keywords: graph visualization, group visualization, literature survey

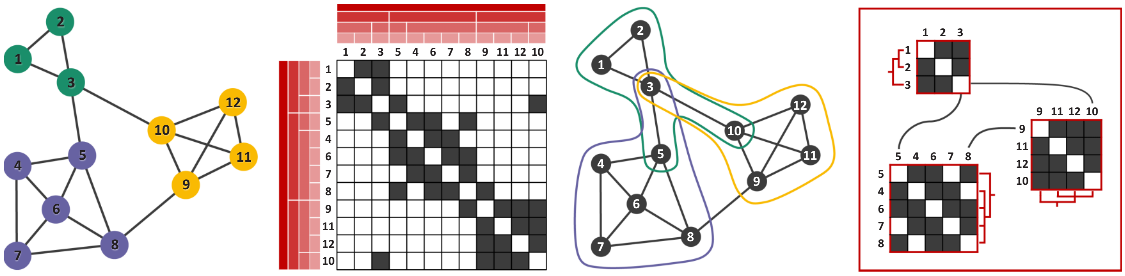
ACM CCS: Categories and Subject Descriptors (according to ACM CCS): H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical user interfaces (GUI)

1. Introduction

Graphs or networks are used to model relationships between objects of any kind. When analyzing graphs exceeding a certain size, however, we do not want to or cannot study each object and each relationship connecting two objects individually. We use visualization to give us a meaningful overview of the graph structure, to highlight central objects, to show similar objects and to reveal outliers. The ability of a visualization to provide these features largely depends on its efficiency to abstract from individual objects into groups or clusters of objects. For instance, applying a random arrangement of visual representatives of objects does not show any of these groups and largely affects the readability of the visualization, for node-link representations [Pur02] as well as for adjacency matrix diagrams [MML07]. In addition to such groups of objects, also the relations among objects are often classified into different types; these types need to be visualized as well to fully understand the depicted graph information.

Indicating groups in the graph by placing similar objects close to each other implicitly shows some group structures. However, it reduces the potentially multi-dimensional concept of object similarity to a two-dimensional (node-link) or one-dimensional (matrix) layout problem: while similarity implies closeness, closeness does not necessarily imply similarity; or in other words, close objects are perceived as similar although their close placement might only be an artifact of the layout algorithm, edge bundling technique, or dimensionality reduction. Moreover, groups could not just be interpreted as disjoint sets of objects, but might be structured hierarchically, might overlap, or might be fuzzy. Implicit encodings of group structures lack the ability to unambiguously define group structures and to encode more complex concepts of groups.

A growing number of visualization approaches have been developed to overcome these limitations of implicit group encodings. These indicate explicitly which group structures are contained in the graph. These group structures can be either automatically identified by clustering or categorization algorithms, or imported from an external source of information. The means to visualize the structures explicitly are versatile (Figure 1): for instance, the group memberships can be encoded in visual node attributes (Figure 1a), they can be shown in a separate view that is dynamically linked to the



(a) Visual node attributes (b) Juxtaposed (c) Superimposed (d) Embedded

Figure 1: *Illustrating examples of the four main categories of visualization techniques to explicitly encode different types of group structures within graph visualizations. (a) Visual node attributes—here color. (b) Juxtaposed—here using an attached approach. (c) Superimposed—here using a contour approach. (d) Embedded—here using a hybrid approach.*

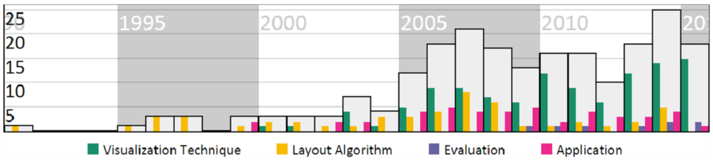


Figure 2: *Number of publications and distribution of paper types from 1991 to 2015 in our literature collection.*

graph view (Figure 1b), the group encoding can be overlaid onto the graph structure (Figure 1c), or both graph and group structure can be merged into an embedded representation (Figure 1d). Describing the large design space of explicit visual encodings of group structures in graphs and classifying existing visualization technique are the main scope of this survey paper.

The literature we collected reveals that already 110 visualization techniques showing vertex (97) or edge (19) group structures in graphs were published, most of them in the past decade (Figure 2, green bars). These are accompanied by various papers on graph layout algorithms that highlight group structures, evaluation papers that study the visualization techniques and application papers that use variants of the techniques in practice (Figure 2, yellow, purple and pink bars).

Although the body of literature is constantly growing, the design space for explicitly encoding group structures in graphs has not yet been surveyed in detail. Existing reports of state of the art focus on other aspects of graph visualization or subproblems: Herman *et al.* [HMM00] describe several approaches that use the hierarchical group structure for navigation and abstraction with a focus on the application to graphs. The survey by Brockenauer and Cornelsen [BC01] contains mainly graph layout algorithms to visualize flat or hierarchical disjoint groups in graphs. Von Landesberger *et al.* [vLKS*11] survey the area of graph and tree visualization in general but only occasionally describe techniques to represent groups in graphs visually. Saket *et al.* [SSK14] introduce a taxonomy of tasks for group-level graph visualization for disjoint groups. General techniques to visualize sets and group structures are reviewed by Alsallakh *et al.* [AMA*14], however, without discussing the integration of these techniques into graph visualizations. Beck *et al.* [BBDW16] survey visualization techniques of graphs that change over time, where in some of them the group structure of the graph is considered. Furthermore, there exist several surveys of general layout algorithms for node-link diagrams [BETT98, DPS02, GFV13]. Elmqvist and Fekete [EF10] provide an overview of how to use a hierarchical group structure of objects for navigation and aggregation in information visualization techniques, such as scatter plots, parallel coordinates, and node-link diagrams.

In this paper, we review the state of the art in visualizing vertex and edge group structures in graphs. It extends our previous publication [VBW15], which focused on vertex group structures only and did not discuss task types. We first introduce the area by discussing the background of the visualized data and, in particular, formulate a consistent terminology (Section 2). We define the scope of the survey and describe the applied methodology to collect and analyze the literature (Section 3). As a basis for the techniques that explicitly visualize graphs and groups, we give an overview of implicit layout methods (Section 4). Our main contribution is the classification of explicit visualization techniques of vertex groups into a two-layered taxonomy that we derived from the collected literature of vertex groups. In contrast our previous publication [VBW15], we apply a similar scheme to further classify and discuss techniques for edge groups (Section 6). As a second addition, we collected tasks that are described in the literature, abstracted them, and classified them with respect to the group structure and task type they refer to (Section 7). We discuss evaluations and applications of the presented techniques (Sections 8 and 9). Based on interviews we conducted with experts of the field, we identify major research challenges for vertex groups that could guide future research (Section 10).

The collected, tagged bibliography is available online[1] in an interactive literature browser. Throughout the paper, we use small icons

[1] http://go.visus.uni-stuttgart.de/groups-in-graphs

Table 1: *Taxonomy of vertex group structures including the respective numbers of technique papers in our literature collection.*

Group structure taxonomy		Overlap	
		Disjoint ○○	Overlapping ⚭
Structure	Flat ‖‖‖	25	23
	Hierarchical ⊓	48	1

as visual cues within the text summarizing and augmenting terms, figures and references. For good comparability, all main figures illustrating the discussed visualization techniques show the same data set (i.e. the same graph and the same groups for each type of group structure).

2. Vertex Group Structures in Graphs

Group structures occur in different applications of graphs structuring the graph vertices in the form of sets, categories or hierarchies. In the following, we define vertex group structures in graphs including a taxonomy for the types of vertex group structures. We further discuss origins that the groups can arise from.

2.1. Definitions

We first introduce a static graph $G = (V, E)$, which consists of a set of vertices V and a set of edges $E \subseteq V \times V$. Vertex groups within graphs, in general, can be defined as a family of sets of vertices $\mathcal{S} = \{S_1, \ldots, S_K\}$, where each $S_k \subseteq V$ and K denotes the number of groups. Groups can be differentiated in several ways (Table 1): they can be disjoint or overlapping, unstructured (flat) or structured (usually, hierarchically).

Overlap: In disjoint group structures ○○, for all pairs (S_{k_1}, S_{k_2}), with $k_1 \neq k_2$: $S_{k_1} \cap S_{k_2} = \emptyset$. Overlapping group structures ⚭, in contrast, contain at least two sets S_{k_1} and S_{k_2} with $S_{k_1} \cap S_{k_2} \neq \emptyset$. Overlapping groups can be further differentiated into crisp ●● and fuzzy ⁂. In crisp overlapping groups, each vertex v_i fully belongs to one or more sets S_k. This belonging can be described, in alternative to the set notation, by a $|V| \times K$ matrix $\mathcal{F}$, where each matrix coefficient $f_{ik} \in \{0, 1\}$ describes if v_i belongs to the k-th set S_k ($f_{ik} = 1$) or not ($f_{ik} = 0$). In contrast, in fuzzy overlapping groups, vertices v_i may belong to different sets S_k to different extent. Here, $f_{ik} \in [0, 1]$ describes to what fraction the vertex v_i belongs to set S_k.

Structure: The groups within the graph might be unstructured, referred to as flat group structures ‖‖‖, or structured. While arbitrarily complex group structures are possible, we only focus on hierarchical group structures ⊓ because other forms are only rarely used in visualizations showing group structure in graphs. We define a hierarchical group structure as a family of sets $\mathcal{H} = \{H_0, H_1, \ldots, H_L\}$, where each $H_l \in \mathcal{H}$ is a set of other group elements from $\mathcal{H}$ or graph vertices $v_i \in V$. These groups represent the inner elements of a hierarchy where H_0 forms the root element. Hence, for all $H_l \in \mathcal{H}$ where $l = 1, \ldots, L$ (i.e. all groups but the root element), there exists exactly one parent group $H_{l'} \in \mathcal{H}$ ($l' \in \{0, \ldots, L\}$) with $H_l \in H_{l'}$; since also each graph vertex is contained in exactly one group, the same applies to all $v_i \in V$ ($\forall v_i \in V\ \exists!\ l' \in \{0, \ldots, L\} : v_i \in H_{l'}$).

To build a taxonomy of group structures, we consider *overlap* and *structure* as orthogonal concepts. Hence, as listed in Table 1, both can be combined into four categories: disjoint flat ○○ ‖‖‖, overlapping flat ⚭ ‖‖‖, disjoint hierarchical ○○ ⊓, and overlapping hierarchical ⚭ ⊓. For the flat approaches ‖‖‖, the group structure is modeled by the family of sets $\mathcal{S}$, whereas the hierarchical taxonomy categories ⊓ require a hierarchical group structure $\mathcal{H}$. In case of disjoint hierarchical groups ○○ ⊓, the hierarchical structure $\mathcal{H}$ replaces $\mathcal{S}$ because the group elements of the hierarchy also provide an overlap-free grouping on every level of the hierarchy. For overlapping hierarchical groups ⚭ ⊓, in contrast, both $\mathcal{S}$ and $\mathcal{H}$ are required to encode both the overlap of groups and the hierarchy. The numbers in Table 1 show that all categories, except for overlapping hierarchical groups ⚭ ⊓, are covered by various visualization techniques, as further discussed below in Section 5.

Graphs can be extended in different directions, for instance, to encode directed or weighted edges, to allow multiple edges between a pair of vertices (multi-graph), or to embed additional multivariate attributes for vertices and edges. Graphs may also change over time regarding their topology and attributes. For a dynamic graph ⋀, the group structure can be defined globally over all points in time, i.e. as a static group structure. Alternatively, a group structure can be derived for each point in time, i.e. the groups are dynamic as well.In the following, techniques that represent a dynamic graph are additionally marked with ⋀; only when the group structure changes over time together with its underlying graph, the technique is marked with ⋀. Since there is a multitude of possible extensions like dynamic graphs that are often orthogonal to the encoding of group structures, we do not explicitly reflect them in our definitions. Hadlak *et al.* [HSS15] formalize these as multi-faceted graphs and give a general overview of visualization techniques.

2.2. Origin of group structures

Graph and group structures only need to be visualized together when there is a relationship between them, which is either known beforehand or should be retrieved through the visual analysis. Group structures can be based on the graph topology or additional vertex attributes. Without further attributes required, topology-based group structures are commonly extracted using graph clustering methods [For10]. Such methods try to detect groups of vertices, the so-called community structure or clustering, with a high density of edges within the groups but low density of edges between groups. These methods usually result in disjoint flat or hierarchical group structures ○○ ‖‖‖ / ⊓, and for some specialized algorithms, crisp or fuzzy overlapping groups ●● / ⁂.

When there are other attributes available to describe the graph vertices in a specific application, these can be used as well to derive a group structure. A categorical attribute directly translates into disjoint flat groups ○○ ‖‖‖; but also overlapping ⚭ and hierarchical structures ⊓ might already be encoded explicitly in a set of attributes. If multiple attributes—in particular, numeric ones—should be aggregated, vertices can be grouped based on these multivariate attributes using standard feature-based clustering and classification algorithms [XW05].

3. Scope and Methodology

To derive a taxonomy of group structure visualizations, we first defined the scope of the survey, collected relevant publications, and tagged all of them with respect to certain categories to structure them. This section describes the methodology we applied and gives an overview of the collected literature data set.

3.1. Scope

The scope of our survey is the visualization of group structures within graphs following the definitions in Section 2.1. Compared to the previous version of our survey [VBW15], in this survey we consider not only groups of vertices but groups of edges as well. Edge groups will be formally introduced in Section 6. We thereby consider only techniques that support the visualization of both the group structure and the graph topology. Techniques that visualize only the groups but not the graph, or vice versa, only the graph were considered out of scope. We further differentiate between implicit and explicit visualization of group structures. There are many layout techniques for node-link representations and vertex sorting algorithms for matrices that can be used to implicitly encode the group structure in the node positions. Such implicit encoding techniques are briefly summarized in Section 4 for vertex groups and Section 6 for edge groups but are not part of our taxonomy unless the implicit encoding was combined with an explicit encoding. Our taxonomy, therefore, comprises only publications that use an explicit encoding of the group structure.

3.2. Data collection and analysis

To collect relevant publications for this survey, we first started with a selection of publications that we knew from previous research and manually inspected the title of all publications of various information visualization journals and proceedings:

- **Journals**
  - *Computer Graphics Forum*
  - *IEEE Transactions on Visualization and Computer Graphics*
  - *Information Visualization*
  - *Journal of Graph Algorithms and Applications*
- **Conferences**
  - *IEEE Pacific Visualization Symposium (PacificVis)* [2001–2004: *InVis.au*; 2005–2007: *APVIS*]
  - *IEEE Symposium on Information Visualization (InfoVis)* [since 2006 a special issue of *IEEE Transactions on Visualization and Computer Graphics*]
  - *International Conference on Information Visualization (IV)*
  - *Joint Eurographics–IEEE VGTC Symposium on Visualization (EuroVis)* [1999–2004: *VisSym*; since 2008 a special issue of *Computer Graphics Forum*]
  - *Symposium on Graph Drawing (GD)*

We also looked at the publications cited by relevant papers and work that cited these relevant publications. This way, we could extend our database step by step to retrieve a comprehensive list of publications relevant to our scope, not just limited to the above journals and conferences.

This literature was structured using tagging as a main instrument, starting with a list of freely assigned reasonable tags that are iteratively merged, extended, and grouped to categories while working through the literature. For further details, we refer to the survey by Beck *et al.* [BBDW16], whose tagging process we followed.

3.3. Literature data set

To analyze the data, we tagged all publications on vertex groups with respect to several categories starting with the paper type. First, we differentiate papers that use only implicit encoding (tag: *layout_technique*; 51 papers) from papers that use explicit encoding. For the latter, we distinguish *application* (41), *evaluation* (7), and *technique* (97) papers. Moreover, each of the publications is assigned at least one tag for each of the following categories: *graph visualization*, *group overlap*, *group structure*, *graph type*, *evaluation* and *application*. Table 2 gives an overview of these tags and the number of technique (#T), evaluation (#E) and application (#A) papers for each tag. In addition, it also acts as a legend for icons used throughout the paper. The main tags for our visualization taxonomy are the tags for the category *group visualization* and further tags to define the subcategories (not part of Table 2). Each explicit visualization paper is assigned to exactly one of the four main *group visualizations*. Only papers that present or evaluate more than one technique are assigned more than one *group visualization* tag if the presented techniques are of different type.

4. Implicit Encodings of Vertex Groups

The most common visual representations of graphs are node-link diagrams (i.e. visual nodes connected by graphical links represent vertices and edges) and adjacency matrices (i.e. rows and columns represent vertices; cells are marked if the two respective vertices are connected by an edge). For both techniques, the visual representatives of vertices need to be positioned on the canvas, i.e. laid out or ordered. By placing related or similar vertices next to each other, group structures can be already indicated implicitly. Please note that our taxonomy and the scope of the paper does not cover these implicit encodings based on vertex positioning. We only give a brief overview of implicit approaches in this section because they are often combined with explicit encodings of groups and part of some of the discussed visualization techniques. We differentiate between one-dimensional and multi-dimensional layout strategies.

4.1. One-dimensional layout

One-dimensional layouts are mainly used for adjacency matrix representations to position vertices along one axis. Groups of vertices that are well connected appear as visual block structures, given that the vertices are ordered appropriately at the matrix axes [Lii10, MML07]. Often, a hierarchical group structure is used to arrange the vertices [EDG*08]; even when using the

Table 2: *Categories and contained tags with descriptions as well as the number of technique, evaluation and application papers using an explicit visualization of vertex group structures. All icons used in this survey are added to the respective tags, except for the icon representing coloring approaches , not listed in the table.*

	#T	#E	#A	
tag (category)	97	7	41	description
graph visualization				graph visualization paradigm
node-link	86	7	39	node-link representation of the graph
matrix	10	1	1	matrix representation of the graph
generic	4		1	being applicable to all graph representations
group overlap				overlap of group
disjoint	73	6	28	no overlap
crisp overlapping	23	1	15	vertices may belong to different groups
fuzzy overlapping	1			vertices may belong to different groups with different extent
group structure				structure type of group
flat	48	5	26	unstructured
hierarchical	49	2	15	groups are hierarchically structured
group visualization				visual representation of groups
visual node attribute	11	3	7	properties of node representation vary
juxtaposed	30	1	3	groups and graph visualized separately
superimposed	35	6	23	use of two aligned visual layers
embedded	22		1	integrate group and graph representation
graph				graph properties
bipartite	2		1	bipartite or semi-bipartite graph
directed	21	1	5	relations are directed
dynamic	17		6	graph changes over time
dynamic_groups	7			group structures changes over time
generic	45	5	20	none of the other graph attributes applies
multi	1		2	multi-graph
multivariate	16	2	8	graph with multivariate attributes
weighted	11	1	2	edges are weighted
evaluation				type of evaluation
algorithmic	8		3	algorithmically using metrics
case study	52		18	application within application domain
comparison	4	1	1	comparison with other visualization technique
user feedback	7	1	1	collection of user feedback
user study	12	7	2	conducting a study involving users
application				application domain
biology	16		20	visualizing biological data
computer	6	1	1	visualizing computer networks
document	5		3	visualizing documents and text
economy	9		2	visualizing business/financial/transport data
media	3	1		visualizing media data
social network	36		12	visualizing social networks (e.g. co-author)
software engineering	22	3	5	visualizing software artifact
sports	3			visualizing sports-related data

hierarchical structure, we can still create different sortings by switching the order of children of a hierarchy element. Some approaches let users interactively build a subjectively satisfying order of rows and columns [Ber11, PDF14] while others solve the sorting problem algorithmically [HF06, MML07]. But also for node-link diagrams , one-dimensional layouts are used, for instance, arranging the nodes on a circle [Hol06] or linear axes [BVB*11].

4.2. Multi-dimensional layout

In contrast, multi-dimensional layouts are only applicable to node-link diagrams because only node-link diagrams allow the free positioning of nodes in a two- or three-dimensional space. Force-directed layout algorithms, such as the Fruchterman–Reingold method [FR91] or the Kamada–Kawai method [KK89], can reveal groups because connected nodes are positioned close to each other. Force-based approaches have been extended in various ways to further enforce the implicit grouping of nodes for disjoint flat groups [BC01, DKM06, DM14b, Noa07]. In general graph layout algorithms, a generic approach to consider disjoint or overlapping groups / is to use pseudo (dummy) vertices that represent sets of vertices and are connected to all contained vertices [EFN99, EH00, GF11]. For disjoint groups , another method is based on a divide-and-conquer strategy [ACJM03, AMA07b, EF97, FT04]:

first, a meta-layout is derived for an aggregated graph with collapsed groups; then, the vertices of each group are laid out independently. For overlapping groups, some approaches apply a sequence of different layout algorithms to first generate a rough layout that is refined in later steps by other algorithms [BALJ06, BCL*07, LDB11, VRW13].

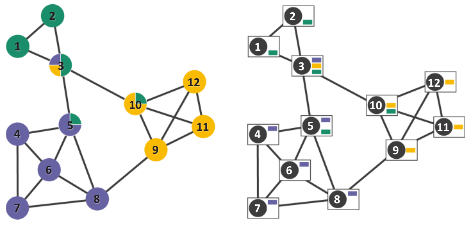
5. Taxonomy of Vertex Group Structure Visualizations

There are various visualization techniques that explicitly encode the vertex group structure within the graph visualization. Some of the explicit encodings are based on layouts already implicitly showing group structures. In total, we collected 97 explicit visualization technique papers, which we categorized according to a hierarchical taxonomy that consists of two layers (Table 3). In the first layer, the four main categories of our taxonomy (illustrated in Figure 1) are visual node attributes, juxtaposed visualization, superimposed visualization and embedded visualization. They are largely disjoint; only some superimposed and embedded visualization approaches use visual node attributes as additional explicit encoding. The second layer further subdivides the categories according to main distinguishing visual features. This section describes all categorized techniques following the hierarchical taxonomy and illustrates them using conceptual sketches.

All techniques were additionally tagged with respect to the type of group structure (see taxonomy of group structures in Section 2.1) that they visualize. The references are therefore marked with the respective icons: flat or hierarchical, disjoint or overlapping. With respect to the type of overlap, by default crisp overlap can be assumed if not indicated otherwise; therefore, only the few fuzzy overlapping groups are marked. Table 3 contrasts both taxonomies by listing all technique papers classified into the respective combination of categories. Few techniques combine two explicit visualization approaches or can be used for different types of group structures; each of these occurs in several cells of the table. Techniques are thereby marked with 1st (2nd) if the approach represents the primary (secondary) visualization approach of this technique. In particular, color is often used as secondary explicit visual mapping of the group structure. As mentioned before, techniques (10) visualizing a dynamic graph with a static group structure will be marked with the symbol. Exclusively those techniques that visualize dynamic groups in dynamic graphs (7) will be marked with the symbol. Finally, depending on the underlying graph visualization, each technique is classified as node-link representation, matrix representation or hybrid.

5.1. Visual node attributes

The association of a vertex with one or more groups can be encoded visually by changing the node representation. Although we can easily distinguish no more than about 7 colors [Hea96], color is widely used to convey group information. Each group $S_k \in \mathcal{S}$ is assigned a color and the nodes of the graph (Figure 1 a) or group representatives (e.g. Figures 5a, b.1 and b.2) are colored, respectively. In total, 40 techniques use visual node attributes, i.e. Color (39) and/or glyphs (7), as primary or secondary explicit encoding of the group memberships; all of them are based on node-link diagrams to represent the graph. Most of these approaches combine the group encoding with one of the other explicit visualization approaches—juxtaposed, superimposed or embedded visualization. Since these other encodings usually dominate the visual appearance, we discuss them in later subsections in detail but indicate the additional encoding via visual attributes by an icon.



(a) Pie-chart (b) Icon

Figure 3: *Visualization of overlapping vertex groups using glyphs and color. (a) Pie-charts encode the fuzzy membership degrees. (b) Icons within the node representatives encode crisp overlapping group memberships.*

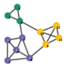
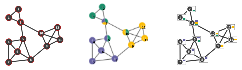
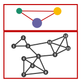
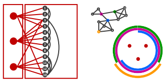
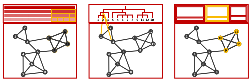
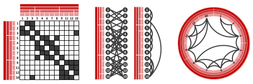
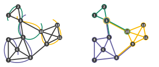
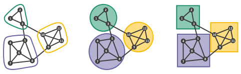
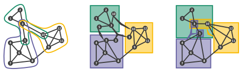
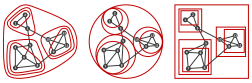
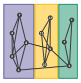
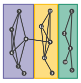
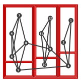
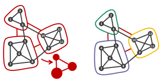
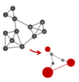
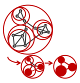
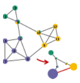
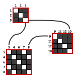
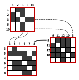
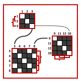
We identified 11 techniques that use only color to visualize group membership explicitly [Dek01, DS13, DYL*15, DYLL15, IMMS09, LWC*14, NIST12, SGKS15, SKL*14, TLTC05, vHW08]. Nodes that belong to only one group are simply colored with respect to that group [DS13, DYL*15, DYLL15, SGKS15, SKL*14, vHW08] (Figure 1a). For flat overlapping groups, nodes are represented using glyphs—'graphical objects designed to convey multiple data values' [War04]. One approach is to represent vertices as pie charts [IMMS09, LWC*14, NIST12, ST08] with sections colored with respect to the groups the vertex belongs to [IMMS09, LWC*14]. For crisp overlapping groups, the sections of the pie charts have equal size [LWC*14, ST08]. In contrast, for fuzzy overlapping groups, they can have different size to encode the fuzzy membership degrees f_{ik} [VRW13] (Figure 3a). Paduano and Forbes [PF15] indicate crisp group overlaps by adding a colored border to a node for each group it belongs to. Another approach for crisp overlapping groups is to represent vertices using boxes that contain icons (Figure 3b), such as cross or check marks, in the particular color for all groups they belong to [TLTC05]. Xu *et al.* [XDC*13] use glyphs to encode the group overlap as well as other metrics by combining different visual channels including intensity of color, hue, size and shape.

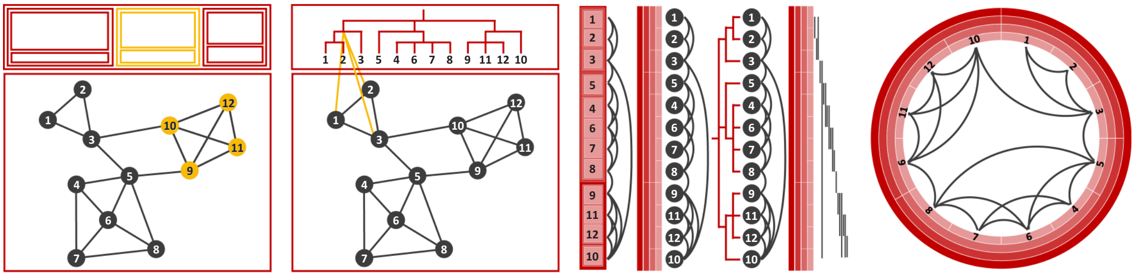
Some techniques optimize the color assignment to maximize either the color differences between neighbouring groups [GHK10, HGK10, LQB12] or the color stability between similar groups [HKV14]. Sansen *et al.* [SLAB15] assign similar colors to nested groups. Vehlow *et al.* [VBAW15] developed a technique for dynamic graphs with dynamic groups. Here, each dynamic group—rather than each individual group—is assigned a color to highlight the evolution of groups, where the optimization approach assigns similar hues to similar dynamic groups.

5.2. Juxtaposed visualization

In juxtaposed visualization approaches, the graph G and the group structure $\mathcal{S}$ are visualized next to each other (Figure 4). We distinguish between separate juxtaposition, where both visualization

Table 3: *Visualization techniques classified by our taxonomy of group visualizations and vertex group structures. References are marked with 1st (2nd) if the visualization approach is used as primary (secondary) visual mapping for the type of group structure. Illustrating images are included only for primary visual mappings.*

		Vertex Group Structure Taxonomy			
		Disjoint flat	Overlapping flat	Disjoint hierarchical	Overl. hier.
Vertex Group Visualization Taxonomy Visual node attributes	Color Section 5.1 Figure 1(a)	 1st [Dek01, DS13, DYL*15, DYLL15, SGKS15, SKL*14, vHW08] 2nd [BPF14, CDA*14, EHKP14, ET07, GHK10, HGK10, HKV14, MH15, SMM13, VBAW15, vdEvW14, WWY*15]	1st [–] 2nd [AHRRC11, BT06, BBT06, DvKSW12, DEKB*14, HRD10, IMMS09, LQB12, LWC*14, NIST12, PF15, TLTC05, VPF*14, XDC*13]	1st [–] 2nd [BD05, BD07, KG06, SBG00, SLAB15]	1st [–] 2nd [VRW13]
	Glyph Section 5.1 Figure 3		 1st [IMMS09, LWC*14, NIST12, TLTC05] 2nd [PF15, ST08, XDC*13]		1st [–] 2nd [VRW13]
Juxtaposed	Separate Section 5.2.1 Figures 4(a)-(b)	 1st [MH15, SMM13, vdEvW14]	 1st [SJUS08, ZXQ15]	 1st [AKY05, AvHK06, CLLT15, CC07]	
	Attached Section 5.2.2 Figures 4(c)-(d)			 1st [AZ13, BBV*12, BD08, BD13, BFBD10, BHW11, BPD11, BMW15, BSW13, BVB*11, GF03, GZ11, GBD09, Hol06, HCvW07, NSC05, PvW06, vH03, vHSD09, VBSW13] 2nd [RMF12]	
Superimposed	Line overlay Section 5.3.1 Figure 5(a)		 1st [AHRRC11, PF15, XDC*13]		
	Contour overlay Section 5.3.2 Figure 5(b)	 1st [BPF14, EHKP14, ET07, GHK10, HGK10, HKV14, NIS15, WWY*15] 2nd [VBAW15]	 1st [BT06, BBT06, BT09b, DvKSW12, DEKB*14, LQB12, HRD10, ST08, VPF*14]	 1st [BD05, BD07, DGC*05, DHRMM13, Hol06, KG06, SBG00, YDG*15] 2nd [NSC05]	
	Partitioning Section 5.3.3 Figure 6	 1st [SKB*14, SA06, ZCCB13]	 1st [LSKS10]	 1st [AFH*10, DWS*14, FWD*03, Hol06]	
Embedded	Node-link Section 5.4.1 Figure 7(a)	 1st [CDA*14, SMER06, VBAW15]	 1st [RHR*10, SZPM10]	 1st [ASH14, AMA07a, AMA08, AMA09, AMA11, DM12, DM14a, HN07b, HN07a, RPD09, SLAB15, vHvW04]	 1st [VRW13]
	Hybrid Section 5.4.2 Figure 7(b)	 1st [HFM07]	 1st [HBF08, MZ11]	 1st [RMF12]	



(a) Separate—brushing&linking (b) Separate—visual links (c) Attached—linear (d) Attached—radial

Figure 4: *Juxtaposed visualization of disjoint hierarchical vertex groups* ○○ ⊤⊤. *(a) Brushing and linking and (b) visual links are used to highlight associated elements of a subhierarchy. In (c)–(d), the hierarchical group structure visualization is aligned with the graph visualization to connect leaves of the hierarchy to the respective nodes of the graph.*

layouts are independent from each other, and attached juxtaposition, where the layouts are aligned, e.g. using the same vertex order. We found 30 technique papers for that category (as primary approach), of which all but 5 visualize disjoint hierarchically structured groups (○○ ⊤⊤; compare to Table 3).

5.2.1. *Separate*

In separate juxtaposed visualizations, the group structure $\mathcal{S}$ or $\mathcal{H}$ is visualized independently of the graph in different views. Although drawn separately, the juxtaposed visualizations are usually linked by interactions (Figure 4a) or visual indicators (Figure 4b). In total, we identified 9 separate juxtaposed visualizations—all but one [SJUS08] for disjoint group structures ○○; 5 for flat ⊓⊓⊓ and 4 for hierarchical ⊤⊤ group structures. All identified separate juxtaposed visualizations show the graph as node-link diagram ⋖.

Disjoint flat group structures ○○ ⊓⊓⊓ can be visualized using node-link diagrams [vdEvW14] ●. Nodes represent groups defined interactively based on multivariate attributes; the number of aggregated edges between groups is mapped to the width of the link connecting the group nodes. The techniques by Sallaberry *et al.* [SMM13] ● ⅄ and Meidiana and Hong [MH15] ● ⅄ visualize the evolution of groups using a time-line approach and the graph of a selected time step separate from the group visualization.

Not only the graph but also disjoint hierarchical group structures ○○ ⊤⊤ are visualized using tree visualization methods such as axis-parallel [AvHK06] (Figure 4b) or radial [CC07] node-link diagrams, layered icicle plots [CLLT15], or a treemap [AKY05] (Figure 4a). Abello *et al.* [AKY05, AvHK06] and Cao *et al.* [CLLT15] link the group structure view with the graph view via brushing and linking. By selecting a subtree in the hierarchical structure H_l, the user can navigate through the graph as only the respective subgraph will be visualized. ASK-GraphView [AvHK06] additionally supports an overview of the complete graph using a matrix representation ⅄ in a third view. VisLink [CC07] arranges two planes showing the group structure and the graph in 3D space. Visual links connect internal nodes of the hierarchy, i.e. group nodes H_l, with all its vertices $v_i \in H_l$, respectively. Again, the highlighting—here using visual links—is done only on demand via selection, and hence, only for a selected subtree of the hierarchy (Figure 4b). Schulz *et al.* [SJUS08] ⚭ ⊓⊓⊓ also make use of visual links between the groups and the graph vertices. They visualize semi-bipartite graphs, i.e. bipartite graphs with possible edges within the bipartite sets of vertices. In their visualization of semi-bipartite graphs, both vertices and groups are arranged separately on two vertical axes and linked visually by straight links, where arcs are used to visualize relations between the vertices. The sorting of either one of the two axes can be adapted to reduce edge crossings. Zhou *et al.* [ZXQ15] ⚭ ⊓⊓⊓ show the graph and the group structure using separate linked views. Within the radial group structure view, vertices are represented by arcs. Groups are encoded by circles inside the ring region.

5.2.2. *Attached*

In contrast to separate juxtaposed group visualizations, attached juxtaposed visualizations align the group structure visualization with the graph visualization. In total, we identified 21 attached juxtaposed visualizations (as primary approach), all for disjoint hierarchical group structures ○○ ⊤⊤. For the alignment, these approaches use the same linear order and place vertices along one axis [AZ13, BPD11, BBV*12, BD13, BVB*11, BHW11, BMW15, BSW13, GF03, GBD09, NSC05, PvW06, PvW08, vH03, vHSD09] (Figures 1b and 4c) or a circle [BD08, BFBD10, GZ11, Hol06, HCvW07, VBSW13] (Figure 4d).

One approach to visualizing disjoint hierarchical group structures is to use a layered icicle plot that is attached to a matrix representing the graph [AZ13, BD13, BSW13, GF03, vH03, vHSD09] ▦ (Figure 1b). The leaves within the icicle plot have to be aligned with the rows and columns of the matrix, i.e. the hierarchical structure is used to generate a linear ordering of the vertices represented in rows and columns. Most of these techniques support an abstraction of the graph based on the hierarchical structure by collapsing and expanding groups to aggregate rows and columns [AZ13, GF03, vH03, vHSD09].

Instead of using a matrix as graph representation, the disjoint hierarchical group structure can also be aligned with a node-link representation of the graph ⋖ (Figure 4c). To be aligned with the hierarchy, the nodes need to be arranged linearly; arcs are usually used instead of straight links to avoid overplotting of nodes and links. The ArcTrees approach [NSC05] combines the linear node-link diagram with a one-dimensional treemap: the arcs are attached to the leaves of the tree visualization (first of Figure 4c). The disjoint hierarchical structure can also be visualized by a node-link diagram [GBD09] (third of Figure 4c) or other tree visualizations with a linear leaf order [PvW06, PvW08] (second of Figure 4c). TimeArcTrees [GBD09] extends these approaches to dynamic graphs. For each time step, the vertices are aligned vertically and directed links are drawn as arcs right (direction is downward) and left (direction is upward) of the vertices. An aligned tree node-link diagram attached at the left visualizes the hierarchy. Increasing the scalability of the graph representation, other approaches place the vertices of the graph on two parallel vertical axes and, instead of arcs, straight links between the two axes visually encode directed graph edges [BBV*12, BPD11, BVB*11]. This technique can be used not only for dynamic graphs [BBV*12, BVB*11] but graph comparison as well [BPD11]. To overcome the problem of visual clutter for dense graphs, edge bundling [BPD11] or edge splatting [BVB*11, BBV*12] (i.e. plotting edge density fields) is applied. Another approach that uses straight links instead of arcs stacksthe links horizontally either above or below the hierarchical group structure visualization to indicate their direction [BMW15] (fourth of Figure 4c, here the layout is rotated by 90 degrees). A timeline attached to the left of each stacked link shows the evolution of the edge weight over time.

The group structure can also be aligned with the graph visualization radially, for instance, by positioning the vertices along a circle circumference and by surrounding the graph visualization with a radial layered icicle plot [GZ11, Hol06, HCvW07] (Figure 4d), by drawing the hierarchical structure as an indented hierarchy in the center of a radial graph representation [BHW11], or by drawing the hierarchical structure on top of the graph visualization [BD08]. The edges of the graph are visualized using arcs [GZ11] or bundled edges [Hol06, HCvW07] (Holten [Hol06] presents three explicit visualization techniques and, therefore, is referenced in three subsections, respectively). Ghou and Zhang [GZ11] furthermore allow an abstraction of the graph by collapsing inner nodes of the tree. For representing dynamic graphs, the graph within the circle needs to be replaced by a sequence of graphs $\mathcal{G} := (G_1, \dots, G_T)$, for instance, arranged in colored pieces of circle rings in TimeRadarTrees [BD08]. Using TimeSpiderTrees [BFBD10], relations are visually indicated by the orientation of shortened links instead of connectedness. In contrast, within the radial layered matrix visualization [VBSW13], edges are represented as color-coded markers in a polar coordinate system.

5.3. Superimposed visualization

Another method to show the graph and its group structure together is to overlay their representations (Figures 5 and 6). In this case, the visualizations of the two layers cannot be rendered independently but have to be fully aligned to create a meaningful superimposition. We identified 35 technique papers that superimpose the group structure onto the graph visualization, where 21 of them use color coding as an additional explicit visual mapping (see also Section 5.1 and Table 3). All of the superimposition techniques are based on two- or three-dimensional node-link diagrams to visualize the graph. We differentiate three main categories of overlays: line overlays (3), contour overlays (25) and partitioning approaches (8).

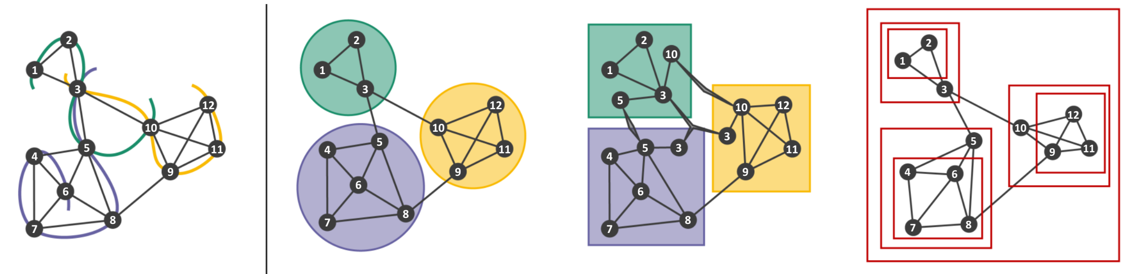
5.3.1. *Line overlay*

When using lines as an overlay, for each group $S_k \in \mathcal{S}$, a line of a particular color connects all nodes of that group without interruption [AHRRC11, XDC*13] (Figure 5a). The LineSets approach [AHRRC11] draws a smoothly curved line for each group, where the shortest path is computed by an adopted Lin–Kernighan's traveling salesman heuristic. In contrast, in the approach by Xu *et al.* [XDC*13], for each group S_k, all nodes $v_i \in S_k$ are connected using a spanning-tree-like shape, which is a generalization of the LineSets approach. While LineSets can be applied to any graph layout, the other approach [XDC*13] uses multidimensional scaling (MDS) to arrange similar items close to each other, i.e. it combines line overlays with an implicit encoding of groups. Within the extended LineSets by Paduano and Forbes [PF15], a line connecting the members of a group replaces the directed links rather than being drawn as curve on top.

5.3.2. *Contour overlay*

Groups can also be visualized within node-link diagrams using closed contours (we identified 25 technique papers): all nodes v_i within the contour are interpreted as belonging to the enclosed group $S_k \in \mathcal{S}$ or $H_l \in \mathcal{H}$ (Figure 5b). Such contours share the characteristics of set diagrams such as Euler diagrams. Contour shapes are versatile, for instance, rectangles [DGC*05, DHRMM13, HRD10, YDG*15] (Figures 5b.2 and 5b.3), circle sections [ET07] or circles [Hol06, KG06, NIS15] (Figure 5b.1), convex hulls [BPF14, ST08, WWY*15], arbitrary two-dimensional curves or splines [BBT06, BD05, BT06, BT09b, DEKB*14, DvKSW12, EHKP14, GHK10, HGK10, HKV14, LQB12, VPF*14] (Figure 1c), or three-dimensional bubbles [BD07, SBG00]. The GMap approach [GHK10, HGK10, HKV14] creates a map of contours that are adjacent to each other using a Voronoi tessellation. In contrast, the contours within MapSets [EHKP14] are generated based on non-crossing spanning trees of points belonging to the same cluster. The trees can be grown to contiguous non-overlapping regions that are optimized with respect to their convexity. Other approaches use such spanning trees as well but draw the filled contours using texture splatting; a splat is defined as radial function for which the transparency increases with the radius. The eXamine technique [DEKB*14] uses an extended self-organizing map neuron grid approach to lay out nodes and links but also to draw the contours. The contours within KelpDiagrams [DvKSW12] are generated using a routing algorithm that links elements of the same group by constructing minimum cost paths over a tangent visibility graph (i.e. a graph including edges that are tangent to the area of linked nodes). Vihrovs *et al.* [VPF*14] create contours using a potential field function. Wu *et al.* [WWY*15] generate them based on the Voronoi treemap using shrinking and smoothing of the Voronoi cells. All nodes connected to a different group are drawn in the gaps between the contours.

Contours may be used alone [DGC*05, ST08], in combination with texture [BT09b] (i.e. each group S_k is assigned a different texture and the contour is filled, respectively), or in combination with



(a) Line overlay in combination with color ⚭ ⊓⊓⊓ ●

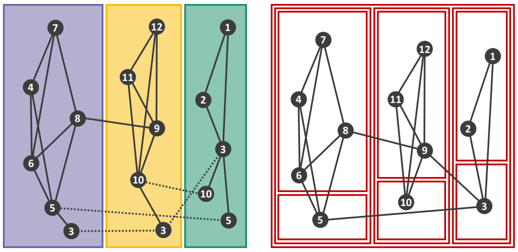
(b.1) Disjoint flat ○○ ⊓⊓⊓ ●

(b.2) Overlapping flat ⚭ ⊓⊓⊓ ●

(b.3) Disjoint hierarchical ○○ 品

(b) Contour overlay in combination with or without color

Figure 5: *Superimposed visualization of the vertex group structure using (a) line or (b) contour overlays, often in combination with* ●. *For overlapping groups, the contours either overlap or, as in (b.2), nodes are duplicated and connected by visual links. In contrast, for hierarchical group structures, the contours are nested (b.3).*



(a) Overlapping flat ⚭ ⊓⊓⊓ ● (b) Disjoint hierarchical ○○ 品

Figure 6: *Superimposed visualization using partitioning of screen space into (a) vertically aligned or (b) nested regions.*

color coding ● (all other approaches). When used in combination with color coding, the contour itself can be colored with respect to the group it surrounds [DEKB*14] (e.g. Figure 1c) or the contour is filled with that color [BBT06, BD05, BD07, BPF14, BT06, DvKSW12, EHKP14, ET07, GHK10, HGK10, HKV14, HRD10, KG06, LQB12, SBG00, VPF*14, WWY*15] (e.g. Figures 5b.1 and 5b.2).

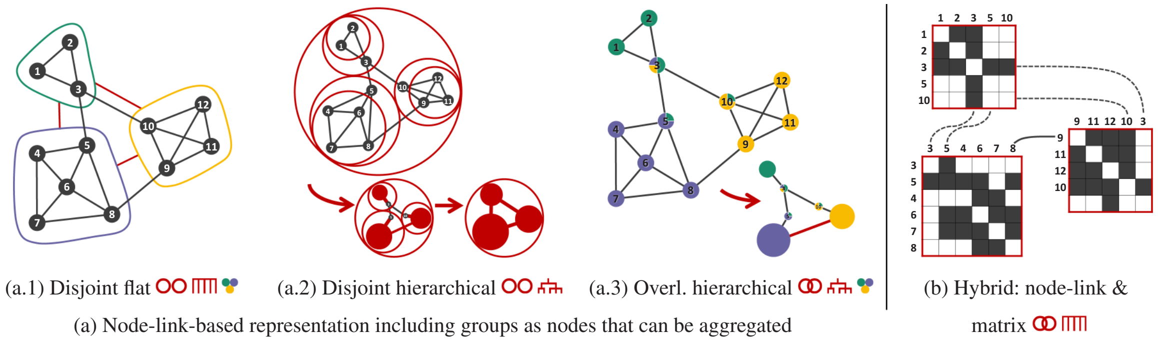
Contours are so far used to visualize disjoint flat ○○ ⊓⊓⊓, overlapping flat ⚭ ⊓⊓⊓ and disjoint hierarchical ○○ 品 group structures (Table 3). For disjoint flat group structures ○○ ⊓⊓⊓ [BPF14, EHKP14, ET07, GHK10, HGK10, HKV14, NIS15, WWY*15], also the contours are disjoint, while the contours representing overlapping group structures ⚭ ⊓⊓⊓ [BT06, BBT06, BT09b, DvKSW12, DEKB*14, HRD10, LQB12, ST08, VPF*14] intersect. To untangle overlapping contours, Henry Riche and Dwyer [HRD10] introduced two techniques for rectangular contour overlays: a splitting approach (groups with intersections are split up, drawn as non-overlapping rectangular shapes, and linked by lines) and a duplication approach (Figure 5b.2) (groups are represented by overlaid rectangles and nodes contained in several groups are duplicated and linked visually). For disjoint hierarchical group structures ○○ 品, the contours or surfaces are nested to encode the hierarchical structure visually [BD05, BD07, DGC*05, DHRMM13, Hol06, KG06, SBG00, YDG*15] (e.g. Figure 5b.3). The circle contour approach by Holten [Hol06] visualizes edges between groups by links that are bundled based on the hierarchical structure (we consider edge bundling an implicit edge grouping technique as discussed in Section 6). ArcTrees [NSC05], although classified as juxtaposed attached visualization, could be considered a contour approach because it uses a contour overlay, in particular, rectangles nested in one dimension.

5.3.3. *Partitioning*

Similar to contour overlays, partitioning indicates group membership by visual enclosing. In contrast to the contour approaches, partitioning is space-filling: the screen space is divided into areas that represent the groups. We identified 8 partitioning approaches, all of them are based on node-link diagrams ∴ to represent the graph. For disjoint flat groups ○○ ⊓⊓⊓, the area of the node-link diagram is partitioned vertically or horizontally into the respective number of areas K—one for each group $S_k \in \mathcal{S}$ [SA06, SKB*14, ZCCB13]; nodes are laid out within the area they belong to (Figure 6). Each area is either surrounded by a rectangular contour [SKB*14, ZCCB13] or colored with respect to the group it presents [SA06] ●. If groups overlap, the same approach can be used, but nodes that belong to different groups are duplicated [LSKS10] ⚭ ⊓⊓⊓ (Figure 6a). Beyond what is shown in the figure, the approach by Lex *et al.* [LSKS10] arranges a two-dimensional area for each group in 3D, like walls of a room, and adds visual links between shared nodes to visualize the overlap. For disjoint hierarchical structures ○○ 品, the screen is partitioned in a space-filling way using a circular icicle plot [AFH*10] ⅄ or a treemap approach [DWS*14, FWD*03, Hol06] (see Figure 6b), where each subsection representing $H_l \in \mathcal{H}$ is surrounded by a contour.

5.4. Embedded visualization

The fourth main category of our taxonomy is the embedded visualization of group structures (Figure 7). At a first glance, this category looks similar to the superimposition approach using contours (Section 5.3.2). But in contrast to overlays, groups are modeled as nodes themselves and are integrated into the graph. In total, we identified 22 technique papers, where 18 approaches are based on node-link representations alone ∴ (Section 5.4.1) and 4 approaches are hybrids of node-link and matrix diagrams ∴ ▦ (Section 5.4.2).



(a.1) Disjoint flat (a.2) Disjoint hierarchical (a.3) Overl. hierarchical

(a) Node-link-based representation including groups as nodes that can be aggregated

(b) Hybrid: node-link & matrix

Figure 7: *Embedded visualization of vertex groups: (a) Node-link-based integrated representations, where groups are included as nodes of the graph and can be aggregated. (b) Using a hybrid of a node-link and matrix representation of the graph and groups within the graph.*

5.4.1. *Node-link*

Using embedded approaches, the groups $S_k \in \mathcal{S}$ or $H_l \in \mathcal{H}$ are drawn as nodes, e.g. using concave shapes (Figure 7a.1). Group nodes are connected by visual links if any of their members are related [CDA*14], [HN07b, HN07a]. Besides these aggregated edges, only edges $e \in E$ within each group are visualized in a non-aggregated way.

While these techniques are static with respect to the group structure visualization, the following approaches support interactive aggregation methods: groups or subtrees can be collapsed to visualize only the group node but not the underlying subset of vertices $v_i \in S_k$ and their within-group edges (Figures 7a.2 and 7a.3). In OntoVis [SMER06], each node representing a group S_k is connected to all its members v_i using visual links in addition to links encoding the edges of the graph. An approach to visualize the evolution of groups for dynamic graphs is to draw groups as rectangles on top of a flow-like group evolution visualization; between-group edges are aggregated and the subgraphs of individual groups are drawn within the group representations [VBAW15]. Disjoint hierarchical groups can be visualized using nested rectangular [ASH14, DM12, DM14a, RPD09] or circular [AMA07a, AMA08, AMA09, AMA11] (3D: spherical [vHvW04]) group structure representations (Figure 7a.2). Reitz *et al.* [RPD09] use the dynamic hierarchical group structure to control the animation of the dynamic graph visualization and to automatically aggregate subhierarchies that do not change. In contrast to these techniques, the Adjasankey diagrams [SLAB15] use a one-dimensional layout of the nodes aligned twice: all nodes that have outgoing links vertically and all nodes with incoming links horizontally. Edges are drawn as flow-like rectangular links connecting two nodes. Based on the hierarchical structure, nodes and links can be aggregated to group nodes and meta edges.

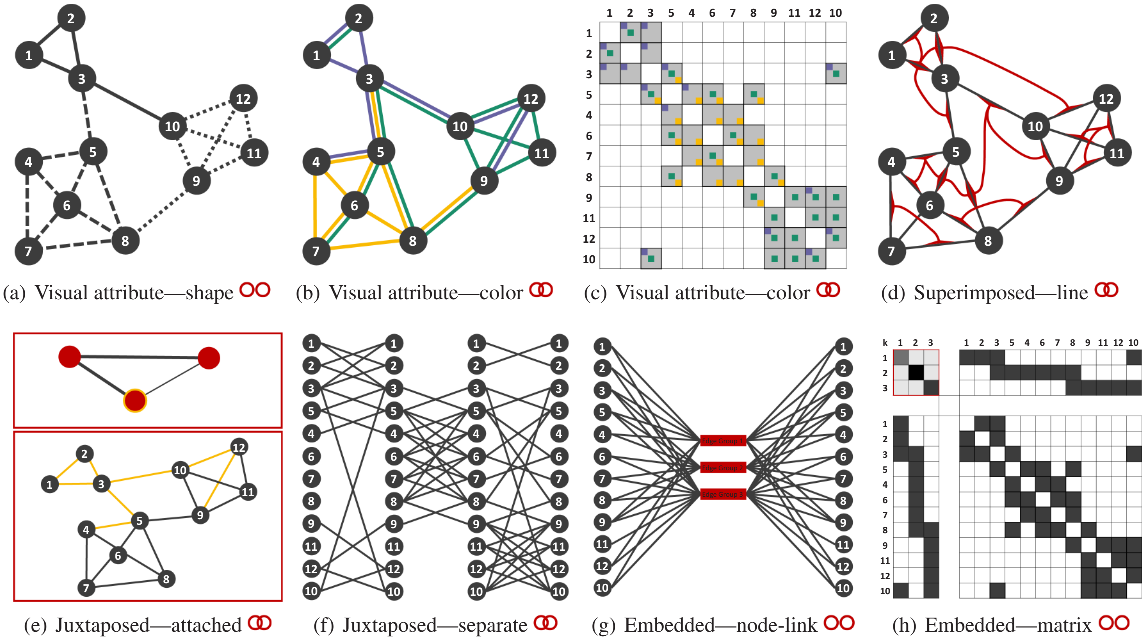
The grid-based visualization approach by Rohrschneider *et al.* [RHR*10] arranges the graph nodes on a regular orthogonal grid, where edges are routed on this grid using a cost minimization technique. Nodes v_i contained in different groups S_k are duplicated. In contrast, Sallaberry *et al.* [SZPM10] place nodes belonging to at least two groups between the respective group nodes, while vertices v_i that belong to only one group S_k can be aggregated and collapsed into group nodes. The approach by Vehlow *et al.* [VRW13] for fuzzy overlapping groups is similar: it aggregates vertices v_i hierarchically based on their membership degrees f_{ik} (Figure 7a.3). Van Ham and Van Wijk [vHvW04] collapse groups by default and show only the area underneath a lens in more detail. For all other approaches, aggregation is performed by individually collapsing or expanding group nodes interactively by clicking on group nodes within the node-link diagram [ASH14, DM12, DM14a, RHR*10, SZPM10] or in a separate tree view [AMA07a, AMA08, AMA09, AMA11].

5.4.2. *Hybrid: Node-link and matrix*

We identified four approaches that use matrix representations to visualize edges within groups and links for relations between groups [HFM07, HBF08, MZ11, RMF12]. In NodeTrix [HFM07], the adjacency matrices are connected to other matrices using edge bundles that visualize the between-group relations. This approach was extended to visualize overlapping groups by duplicating vertices v_i for each group S_k they belong to [HBF08] (Figure 7b). Also the approach by Misue and Zhou [MZ11] allows one to visualize overlaps using node duplication. Here, in addition to the matrices representing groups, a node is drawn for each group and linked to all its members v_i, i.e. to the respective rows or columns of the matrices or to single nodes v_i not contained in any group and hence matrix. TreeMatrix [RMF12] encodes hierarchical structures, where subgraphs H_l are shown as adjacency matrices with an attached hierarchy that is visualized as a node-link diagram or using an icicle plot (see also Section 5.2.2) and can be collapsed interactively (Figure 1d).

6. Edge Group Structure Visualizations

The previous part of this survey focused on groups of vertices. Although considering groups based on graph vertices is more common, also relations can be grouped. This section gives an overview of edge group structures in graphs and how these groups and their visualizations fit into the previously defined taxonomies of group structures and group visualizations. The scope of the survey on visualizations of edge groups is analogous to that for vertex groups (Section 3.1), i.e. we only consider techniques that support the visualization of



(a) Visual attribute—shape ○○ (b) Visual attribute—color ⚭ (c) Visual attribute—color ⚭ (d) Superimposed—line ⚭

(e) Juxtaposed—attached ⚭ (f) Juxtaposed—separate ⚭ (g) Embedded—node-link ○○ (h) Embedded—matrix ○○

Figure 8: *Visualization techniques for edge groups of disjoint ○○ or overlapping ⚭ flat ⫼ groups.*

both the edge group structure and the graph topology. We organized the publications using the same data collection, analysis and tagging process as described in Sections 3.2 and 3.3.

In analogy to vertex group structures, edge groups can be defined as a family of sets of edges $\mathcal{E} = \{E_1, \ldots, E_K\}$, where each $E_k \subseteq E$ and K denotes the number of groups. We identified visualizations for disjoint ○○ as well as for crisp overlapping ⚭ edge groups that were all unstructured, i.e. flat ⫼. In disjoint flat edge group structures ○○, all pairs (E_{k_1}, E_{k_2}), with $k_1 \neq k_2 : E_{k_1} \cap E_{k_2} = \emptyset$. Overlapping group structures ⚭, in contrast, contain at least two sets E_{k_1} and E_{k_2} with $E_{k_1} \cap E_{k_2} \neq \emptyset$. In the following, implicit and explicit visualization techniques for disjoint ○○ or overlapping ⚭ flat ⫼ edge groups will be presented.

Similar to vertex groups, edge groups can be visualized implicitly. Within adjacency matrix representations ▚, edge groups emerge visually as clusters of cells given that the vertices are ordered appropriately at the matrix axes [Lii10, MML07]. Using force-directed layout algorithms for node-link diagrams ⋖ implicitly shows groups of edges as well. In addition, within node-link diagrams ⋖, edge bundling methods can be used to deform and visually group similar edges into bundles. There are three general types of edge bundling methods for graphs: cost-based, geometry-based, and image-based edge bundling methods surveyed by Zhou *et al.* [ZXYQ13]. Edges are thereby commonly bundled hierarchically.

Often, the implicit encoding is combined with an explicit encoding. Edge bundling is sometimes integrated with the visual edge attribute color to explicitly visualize groups of edges. Our taxonomy for edge group visualizations covers only those edge bundling techniques that use an explicit encoding in addition to bundling.

There are some visualization techniques that explicitly encode the edge group structure within the graph visualization. We identified 19 explicit visualization technique papers, which we categorized according to the same hierarchical taxonomy as for vertex groups, including the four main categories: visual attributes, juxtapositioning, superpositioning and embedding. As mentioned before, only flat ⫼ edge groups could be identified. Therefore, all techniques presented in this section will be tagged with respect to the type of overlap only: disjoint ○○ (11 techniques) or overlapping ⚭ (8 techniques).

The association of an edge with one or more groups can be encoded visually by changing the **visual edge attributes**, i.e. the link in a node-link diagram ⋖ or cell within a matrix ▚. The use of visual edge attributes is the most common approach for encoding edge groups, with 13 out of 19 techniques. Among these techniques, 10 make use of color ● to encode disjoint ○○ [EHP*11, ETB11, GK07, HF06, TE10, YS15] or overlapping ⚭ [AZ13, DKL13, YDG*15, Zec10] groups. In node-link diagrams ⋖, links [ETB11] or edge bundles [EHP*11, GK07, TE10, YS15] are colored with respect to the group they belong to. If groups overlap ⚭, links that belong to several groups are duplicated and drawn next to each other in the respective colors [DKL13, YDG*15] (Figure 8b). Color can also be used for adjacency matrix representations ▚ by coloring the cell with respect to the group the edge belongs to [HF06]. For overlapping groups ⚭, each cell is subdivided into regions—one for each group—and the region is filled with the respective group color if the edge belongs to that group [AZ13, Zec10] (Figure 8c). As an alternative to color, the style of the links within node-link diagrams ⋖ can be varied to visualize which group an edge belongs to [JJ10, Kit03, SGKS15] ○○, e.g. using solid and different types of dashed links (Figure 8a).

We identified three **juxtaposed visualization** techniques [BPD11, DLR10, RMM15], which can all be used for overlapping groups. All three techniques show the graph topology using a node-link diagram. Detangler [RMM15] visualizes the representation of groups separately from the graph; both views are connected by brushing and linking (Figure 8e). In the node-link-based group representation, each node represents a group of edges E_k, with relations between groups if they share any edges. Other approaches show the graph multiple times—once for each group of edges [BPD11, DLR10]. Both techniques make use of a fixed one-dimensional [BPD11] (Figure 8f) or two-dimensional [DLR10] node-link layout that is reused for each group, and hence, graph representation. Didimo *et al.*[DLR10] use color in addition to the juxtaposition.

We are aware of only one **superimposed visualization** technique, in particular, a line overlay approach. Vehlow *et al.* [VHTW13] visualize pairs of edges—i.e. $|E_k| = 2$—within a node-link diagram by connecting the respective links with curves (Figure 8d).

We identified two **embedded visualization techniques**, one that shows the graph as a node-link diagram [PvW08] and one that shows the graph as a matrix representation [DHRW15]. Pretorius and van Wijk [PvW08] visualize edge groups as nodes positioned in the center of the visualization. The edges of the graph are partitioned by letting every link pass through the node that represents its group (Figure 8g). The Dual Adjacency Matrix [DHRW15] visualizes the edge groups integrated into the graph visualization (Figure 8h). The rows (columns) of the top left quadrant represent edge groups. The graph is visualized in the bottom right quadrant of the matrix; rows (columns) can be aggregated to vertex groups. The two remaining quadrants show which vertices (vertex groups) are covered by which edge groups.

7. Tasks

Depending on the application and the type of group structure, different tasks are relevant for conducting a visual analysis of group structures in a graph. Lee *et al.* [LPP*06] present a list of tasks for visualization that are commonly encountered while analyzing graph data. Among these tasks, they list the identification of clusters (groups) as an important task. Saket *et al.* [SSK14] introduce a task taxonomy for group-related graph tasks for disjoint flat vertex groups. Their tasks concentrate on vertex groups visualized within node-link diagrams using a map metaphor—a superimposed contour approach. Following their task taxonomy, we group tasks into four categories, including *group-only*, *group–vertex*, *group–edge* and *group–network* tasks. However, we define the task categories slightly different, as described in the following. *Group-only tasks* can be performed by only considering the groups, i.e. the number K or L of groups contained in $\mathcal{S}$, $\mathcal{E}$ or $\mathcal{H}$ or the nesting structure of subgroups $H_k \in \mathcal{H}$; no vertex or edge information is required. For *group–vertex tasks*, both group and vertex information has to be considered, i.e. everything described by $\mathcal{S}$ or $\mathcal{H}$. *Group–edge tasks* can be performed by only taking group and edge information into account, i.e. everything described by the edge group structure $\mathcal{E}$. For *group–network tasks*, all information—group, vertex and edge information—has to be considered. This includes the graph G and the group structure $\mathcal{S}$, $\mathcal{E}$ or $\mathcal{H}$, respectively. Therefore, we categorize tasks such as 'count the number of edges in a given vertex group' as *group–network task*, rather than *group–edge task* as done by Saket *et al.* [SSK14], because these tasks indirectly require the information of which vertices are contained in the groups.

The task taxonomy by Saket *et al.* [SSK14] is limited to disjoint flat vertex group structures. We extend their task taxonomy to cover tasks for disjoint hierarchical groups and overlapping flat groups. We also searched for tasks for overlapping hierarchical groups, but—due to the yet limited coverage of such visualizations in literature—we did not find any specific tasks that were not yet covered by the previously mentioned task categories. To collect tasks for different types of groups structures, we went through all technique, evaluation, and application papers for vertex or edge groups in graphs and searched for particular keywords, including 'task', 'question', '?', 'identify', 'analyze', 'determine' and 'find'. All tasks were abstracted to phrasings using the words vertex, edge and group, e.g. the task 'identify people belonging to a particular society' was rephrased to 'which vertices are associated with a given group?'.

Table 4 shows an overview of all collected and abstracted tasks for each of the four group structure categories. The tasks are subgrouped with respect to their type: *group-only task* (GOT), *group–vertex task* (GVT), *group–edge task* (GET) or *group–network task* (GNT). Most of the tasks relate to vertex groups (marked with VG) and only few tasks refer to edge groups (marked with EG). In addition, the references from which the tasks were collected are integrated into the table. First, all tasks for disjoint flat groups are listed. These also apply to disjoint hierarchical groups, considering subgroups H_l rather than groups S_k. For overlapping flat and overlapping hierarchical groups, these tasks apply as well, where some tasks need to be rephrased slightly to consider that a vertex may belong to different groups. Although not listed in the table, in analogy, tasks for overlapping hierarchical groups are a superset of all tasks listed in the table; so far, no tasks were proposed that require overlapping and hierarchical groups for a single tasks.

8. Evaluation

Our collection of publications contains only few evaluation papers that describe extensive user studies (8 in total, 6 for vertex groups and 2 for edge groups), but most of the technique papers include some kind of evaluation (see evaluation tags in Table 2). In this section, we summarize the results presented in the 8 evaluation papers as well as insights gained from user studies contained in technique papers that thoroughly evaluate group-related tasks (4 in total).

Contrasting visual node attributes (Section 5.1) and superimposed techniques (Section 5.3), a series of four recent user studies, by now, provides the most systematic evaluation of visualization techniques in the field: Saket *et al.* [SSKB14] compared a superimposed contour approach (GMap [GHK10]) against the use of color as a visual node attribute (Section 5.1). They investigated several network-based as well as group-based tasks for disjoint flat group structures including *group–vertex tasks* and one *group–edge task*. The results of their user study with 36 participants suggest

Table 4: *Group-related tasks for different types of group structures: disjoint flat, disjoint hierarchical, overlapping flat and dynamic group structures. The tasks are grouped with respect to their type: group-only task (GOT), group–vertex task (GVT), group–edge task (GET) or group–network task (GNT). Tasks that relate to vertex groups or edge groups are marked with VG or EG, respectively.*

Disjoint flat group structures ○○ ⊓⊓⊓		
GOT	VG	What is the number of groups in $\mathcal{S}$? [HRD10, JRHT14, SSK14, SSKB15]
	EG	What is the number of groups in $\mathcal{E}$? [EHP*11, HF06]
GVT	VG	Which vertex group S_k is the largest (smallest)? [APP10, DS13, HD12, SSK14] – How large is the size difference between two vertex groups? [HRD10, JRHT14] – Which vertex group S_k does a given vertex v_i belong to? [SSK14, SSKB14, SSKB15] – Which (how many) vertices are associated with a given group? [AHRRC11, DEKB*14, DvKSW12, PLS*13, SSK14, VKB*15, VPF*14, XDC*13] ([DS13, SSK14, SSKB14]) – Does a set of vertices belong to the same group? [JRHT14, SSK14, SSKB14]
GET	EG	What is the size of a given edge group E_k? [EHP*11, HF06] – Which group E_k does a given edge e_j belong to? [ETB11] – Which edges e_j are outliers, i.e. isolated edges or missing edges in a group? [HF06]
GNT	VG	Which one is the vertex with the highest degree in a particular group? [CSL*10, GMT09, HD12, JRHT14, PSK11, SSK14] – Which vertices of a group are related to vertices of other groups? [GMT09, WWY*15] – Which group is most connected to a particular vertex or group? [APP10] – Given two vertices A and B, how many groups have to be passed on the path from A to B? [JRHT14, SSK14] – How many vertices (edges) need to be removed to disconnect two given groups? [SSK14] – Given two vertex groups, how strongly are they coupled? [CDA*14, SSKB14, WWY*15, YLZ*13] – Given a vertex group, how strongly is this group coupled to other groups? [CDA*14] – Which vertex group has the maximum number of adjacent groups? [CDA*14, CSL*10, HBF08, LWC*14, SSK14] – What is the number of edges within a given vertex group? [SSK14] – Which group is the most sparsely (most densely) connected vertex group? [CSL*10, GMT09, PSK11, SSK14]
	EG	Which pairs of vertices have a relation from a particular edge group E_k? [ETB11] – Which vertices are covered by a given edge group E_k? [DHRW15] – Which edge groups E_k share many vertices? [DHRW15] – Which edge group E_k is sole vertex connector? [DHRW15]
Disjoint hierarchical group structures ○○ ⊤		
GOT	VG	What is the hierarchical structure of the graph? [HN07b] – What are the top-level or bottom-level parts of the hierarchy? [DWS*14] – Given a group, which subgroups are direct children of this group? [GZ11]
GVT	VG	Which subgroups H_l is a vertex directly or indirectly allocated to? [GZ11] – Which subgroup is the earliest common parent of a set vertices? [GZ11] – What are the top-level or bottom-level parts of the hierarchy? [DWS*14]
GNT	VG	Which subgroups H_l of the hierarchy have a high degree of within-group edges, i.e. which groups are cohesive? [ABZD13, BPD11, RMF12] – Which subgroups H_l of the hierarchy have a high degree of between-group edges, i.e. which subgroups are coupled? [ABZD13, BPD11, DWS*14] – How are edges of the graph linked to the hierarchical group structure, i.e. does the hierarchical group structure reflect the graph topology? [BD13, NSC05] – Are there edges between different layers of the hierarchical group structure? [RMF12] – Which vertices are involved in such cross-layer edges? [RMF12]
Overlapping flat group structures ⚭ ⊓⊓⊓		
GVT	VG	Which groups overlap? [AHRRC11, DEKB*14, LWC*14, VPF*14, XDC*13] – To what extent do groups overlap? [DvKSW12, HRD10, VPF*14, XDC*13] – Which vertices are associated with only one group? [HBF08, ST08, VRW13] – Which vertices are associated with at least two groups? [GMT09, LWC*14, PF15, ST08, SZPM10, VKB*15, VRW13] – Which vertices build bridges between groups, i.e. vertices whose removal disconnects the groups and makes them disjoint? [BPF14, GMT09, HBF08, LPP*06, VRW13] – Which groups does a given vertex belong to? [AHRRC11, DEKB*14, DvKSW12, HRD10, VPF*14, VRW13, ZXQ15, XDC*13] – For fuzzy overlapping ●● groups, this question can be extended to: To what extent does a given vertex contribute to its groups? [VRW13] – Does a set of particular vertices belong to the same groups? [AHRRC11] – Which two groups share the largest number of vertices? [HBF08] – Which vertices are in group A and/or B? [DvKSW12, HRD10] – Which vertices are in group A but not in B? [DvKSW12, HRD10]
GET	EG	Given multiple groups, what are their common edges? [MGK11] – Given an edge, is it part of multiple groups? [MGK11]
GNT	VG	How does the overlapping group structure map to the graph topology? [BT06] – Which group is the most central one, i.e. the group that shares vertices with high degree with the largest number of other groups? [HBF08] – Which groups share the largest number of vertices with high degree, and hence, have the strongest cohesion? [HBF08]

that adding contours does not negatively impact the performance of network-based tasks and the GMap approach outperforms colored nodes with respect to group-based tasks. In a second study, Saket *et al.* [SSKB15] ○○ ⊓⊓⊓ ● report on experiments measuring the extent to which people remember the data depicted in these two types of group structure visualizations. The 40 participants of the study had to do tasks—including one *group-only task* and one *group–vertex task*—four days after being exposed to the visual stimuli. The results suggest that participants recall data shown with colored contour-based approach more accurately than using color only with differences in the accuracy of the tasks performed. Jianu *et al.* [JRHT14] ○○ ⊓⊓⊓ ● replicated the first study by Saket *et al.* [SSKB14] and included two more approaches in their online study comprising 800 participants. They evaluated colored nodes, line overlay and two types of contour overlays—GMap [GHK10] and BubbleSets [CPC09]—based on 5 group-based and 5 network-based tasks. The group-based tasks included two *group-only tasks*, one *group–vertex task* and two *group–network tasks*. With respect to group-based tasks, BubbleSets performs best, followed by lines and the GMap approach, while color appears to be least effective.

Line overlays (LineSets) were also compared to contour overlays—again the BubbleSets technique—by Alper *et al.* [AHRRC11]. They conducted a user study (12 participants) to evaluate the performance on four tasks—two *group-only tasks*, and two *group–vertex tasks* for overlapping flat groups. Compared to the study by Jianu *et al.* [JRHT14], they found that LineSets improve the readability of set membership and set intersection tasks– - with higher accuracy rates and shorter completion times—compared to the BubbleSets technique. They also did an informal, small-scale eye-tracking study that aimed at understanding some of the effects seen in the quantitative results.

Other evaluations focus solely on superimposed contour approaches (Section 5.3.2): Henry Riche *et al.* [HRD10] evaluated their Euler diagram technique with respect to its readability considering five tasks—three *group-only tasks* and two *group–vertex tasks* for overlapping flat groups. In their study (18 participants), they compared their two rectangular contour overlay techniques—the splitting approach and the node duplication approach (Section 5.3.2)—to a third (non-convex) contour overlay. They found that the duplication approach outperforms the other techniques for two of the group-related tasks, but the splitting approach is preferred by many participants. Using a qualitative evaluation, Byelas and Telea [BT09a] compared algorithmically generated contour overlays to hand-drawn contours to improve the rendering algorithm. The GraphDiaries technique [BPF14] was evaluated based on a user study comparing it to two other approaches for dynamic graphs. The focus of the study lies on tasks related to the dynamic behavior analyzing groups of added or removed elements.

Some evaluations also take embedded approaches into account (Section 5.4): Archambault *et al.*[APP10] compared the use of color as a visual node attribute with an embedded approach, where groups of nodes are replaced by colored group nodes. They evaluated how this affects the readability, but with respect to tasks focusing on attributes and graph topology rather than group structures. In contrast, Henry *et al.* [HBF08] evaluated their embedded hybrid approach with respect to six tasks—three *group–vertex tasks*, one *group–edge task*, and two *group–network tasks* for overlapping flat groups. Their user study (12 participants) applied different alternatives of vertex duplications in overlapping groups and compared these to an embedded approach without duplication. As a result, they found that duplications improve group-related tasks but sometimes interfere with other graph readability tasks.

Hierarchical group structures in graphs have been rarely evaluated, so far; the same applies to juxtaposed approaches (Section 5.2). There is only one user evaluation on superimposed contour visualizations in the context of hierarchies [FKH15] comparing a hierarchical against a flat contour-based visualization. The 29 participants of this study were asked typical software system comprehension tasks that do not affiliate in our derived set of group-related tasks. With respect to these tasks, they found a statistically significant increase in task correctness of their hierarchical visualization.

All these evaluations focused on visualizations for vertex groups. Abuthawabeh *et al.* [ABZD13] present a study evaluating two visualization techniques for overlapping edge groups: the use of color within the matrix [AZ13, Zec10] (Figure 8c) and the juxtaposed attached approach as illustrated in Figure 8f [BPD11]. Both techniques visualize the disjoint hierarchical vertex group structure in addition to the edge groups using a juxtaposed attached visualization of the hierarchy. The tasks of their study aim at comparing different edge groups considering the hierarchical vertex group structure at the same time. They found that all 8 participants were able to identify equivalent edge-groups forming hierarchical groups in the presented graphs. Also Melville *et al.*[MGK11] compared two color-based matrix approaches, one that shows all edge groups in one matrix (Figure 8c) and one that shows the edge groups juxtaposed using small multiples of the matrix. Based on their study including 18 participants and two *group–edge tasks*, they found that the comparison of edge groups using one matrix was better than using juxtaposed matrices by nearly 50%.

9. Application

Group structures occur in various application domains of graphs. In total, comprising application, evaluation and technique papers, the most common application domains for the visualization of vertex group structures are social networks (48 papers), biology (36 papers) and software engineering (29 papers). For edge groups, biology (9 papers) and software engineering (7 papers) can be considered the two main application domains. In this section, we summarize mainly the application papers but occasionally also technique papers with a focus on these areas. Further application domains of group visualizations are economy networks representing business, transport, or financial data, computer networks, relations between documents or texts, or relations within media data or sports-related data.

In **biological applications**, graphs are almost exclusively represented as node-link diagrams. In particular in protein–protein interaction networks and gene correlation networks, disjoint and overlapping flat group structures occur. These mainly result from categorical attributes of the genes or proteins, e.g. from cell compartment and pathway associations or from gene ontology annotations; also clustering is applied to extract motifs, i.e. functional groups of proteins. Commonly, group structures are visualized by visual node attributes [BST03, DC11, FGB*07, TvDEF09] and superimposed techniques—including overlaid contours [PLS*13, RRAS08, SXS*12, VHK*13, VKB*15] and partitioning approaches [BMGK08, GFK*14, PK06, SLK*09]. Also attached juxtaposed [SJUS08] and embedded [RHR*10, VRW13] approaches have been applied to biological networks. Edge groups in biological networks usually represent different types of biological reactions or different contexts where these reactions occur. These disjoint groups are commonly visualized using the visual edge attributes color [DC11, LYKB08] or style [GHM*02, JJ10, JKS06, Kit03].

In **social networks** such as friendship, communication, collaboration or co-authorship networks, vertices represent people, whereas edges encode relationships between them. Groups of vertices, therefore, identify circles of friends, groups that cooperate, or the like. Social groups, also called communities, may be disjoint but are often modeled more realistically by overlapping groups because people often participate in a multitude of diverse, yet overlapping social communities. So far, social communities have been visualized mainly within node-link diagrams. Similar to the biological domain, the group structure is commonly visualized by visual node attributes (color) [CMF*14, PSK11], superimposed visualization, in particular using line [AHRRC11, XDC*13] or contour overlays [CCC02, DLM14, PS06, SCL*09], or using color and contour overlays in combination [GMT09, HD12, HB05]. Hierarchical group structures ⏉ in social networks, in contrast, are often visualized in attached juxtaposed views together with static [GZ11] or dynamic graphs [BBV*12, GSZ*11]. Also embedded approaches have been applied to social networks [AMA08, DM14a, HBF08].

In **software engineering**, network visualization is used to analyze program structures and their hierarchical organization ⏉, which is usually given by the modularization of the software system. Within software architecture diagrams, also software metrics can be used to define disjoint ○○ or overlapping ⚭ flat group structures [BT09b, TLTC05]. The hierarchical structure of call graphs or other dependency networks is commonly visualized using attached juxtaposed visualizations [BPD11, BD13, PvW07, PvW08, SJSJ05, vH03], superimposed contours [RFG05], or embedded approaches [PGKG08, RMF12]. Overlapping group structures can be visualized using glyphs [TLTC05] and overlaid colored [BT09a] or textured [BT09b] contours. Edge groups in software engineering might represent different types of code couplings, e.g. inheritance, aggregation, and usage couplings. These overlapping edge groups are commonly visualized using the visual edge attribute color [AZ13, ETB11, TE10, Zec10], juxtaposition [BPD11], or embedding [PvW08].

10. Research Challenges

The taxonomy of techniques shows what has been achieved in the field and reveals possible gaps in the research literature. However, not necessarily, every gap is a good research opportunity and there might be other interesting challenges that are not indicated by gaps in the taxonomy. To provide ideas of worthwhile future research, we discussed research challenges with respect to vertex group structures with other researchers who have substantially contributed to the field. We interviewed 7 experts in graph or group visualization face-to-face—on average for about 40 min per person. We first showed a preliminary version of our taxonomy of vertex group structure visualizations containing illustrations of existing techniques (similar to Table 3), explained our interpretation of vertex groups and group structures in graphs and asked them for feedback on terms and definitions—this feedback is already reflected in the terms used in the definitions (Section 2.1) and taxonomy of visualization techniques (Section 5). The main purpose of these interviews, however, was to ask for the experts' opinion on open problems and challenges on visualizing vertex group structures in graphs. Besides challenges they named, we also discussed the challenges that we identified beforehand, in case they did not mention them already. Based on the feedback we received within the interviews and some of our ideas, we identified five main challenges for vertex groups—each regarded as relevant by 2 to 5 experts. For edge groups, we have not yet conducted any interviews. In general, there is a remarkably great difference in the number and variety of techniques for edge groups compared to vertex groups. The fact that edge groups have been considered less than vertex groups suggests that visualizing edge groups is a challenge itself.

10.1. Time-varying groups and comparison

In many application domains, graphs are not static but change over time, i.e. their topology or their attributes change over time. It follows that the topology-based or node-attribute-based group structures also change over time. If changes in the graph are significant, the group structure should be determined for each point in time individually. In contrast, for minor changes in the graph topology or attributes of the graph, it is often sufficient to visualize the static group structure. Most techniques that have been developed to visualize the evolution of groups, do not visualize the graph topology [FBS06, OMB*07, RTJ*11, RB10]—for this reason, these are not part of our taxonomy. Other approaches focus on the visualization of dynamic graphs but not the temporal evolution of groups; they visualize the group structure aggregated over time.

First attempts have been made to visualize both the evolution of groups and the dynamic graph together either using animation [HKV14, KG06, RPD09] or using a timeline-based approach [AFH*10, MH15, SMM13, VBAW15], but these only cover evolving disjoint flat or hierarchical group structures ○○ ⅄. Related to dynamic graphs is the problem of graph comparison: instead of several graphs in a sequence, an unordered set of graphs is compared. Similarly, the comparison of groups structuring these graphs has not yet been discussed in this context.

10.2. Data complexity

Instead of having multiple versions of the data, the data itself can get more complex by adding or refining data dimensions. For overlapping groups ⚭, for instance, the visualization of fuzzy memberships ●● is challenging, where vertices may belong to different groups with different extent (see Section 2.1). Although the detection of fuzzy overlapping groups has become quite popular in the domain of graph clustering [For10], their visualization was only addressed in one work so far [VRW13]. In many applications, groups need to express a degree of uncertainty that can be modeled as fuzzy groups. Another degree of complexity could be introduced by the topology of the group structure: so far, most of the visualization approaches that were developed for overlapping groups ⚭ can handle only flat group structures ⊓⊓⊓ (Table 1). However, also overlapping groups can be organized hierarchically ⏉, for example, derived from an ontology, through clustering, or other sources. The complexity of the group structure visualization also increases when multivariate attributes of vertices and edges need to be visualized together with the graph. These attributes could, for instance, explain why certain elements are grouped together or why a pair of groups overlaps.

10.3. Scalability

The data do not need to get more complex, but already visualizing more data elements can be challenging. In graph visualizations, questions of scalability usually relate to the number of vertices and density of edges. Visualizing additional group structures, however, introduces further challenges. For an increasing number of groups, for instance, encoding the groups by colors ● becomes difficult for more than about 7 groups [Hea96]. There are already some approaches that optimize the color assignment (Section 5.1), but there is still potential to improve and extend these approaches. Also, for larger numbers of groups, coloring approaches probably need to be replaced by other group representations.

But even having a constant number of groups, scalability issues could arise from increasing the overlap of groups ꝏ. For instance, superimposed approaches (Section 5.3) become more and more cluttered through a denser overlay of group structures. Maybe, even new representations need to be found to handle datasets with large overlap of many groups. In hierarchical groups ⊤, also the depth of the hierarchy could become a problem of scalability for some visualizations.

10.4. Interaction technique

One way to address certain issues of scalability is the use of interaction methods such as aggregation, which already is a widely used method—30 of the 111 collected technique papers support aggregation. But some data are lost through this abstraction. The question, therefore, remains how to aggregate while, at the same time, visually encoding the uncertainty of aggregated groups and the density of edges within these groups. For overlapping groups ꝏ, aggregation is even more difficult because overlaps either need to be represented explicitly or the overlap is not retrievable for the users. Also, there is a need for advanced interactive (semi-) automatic aggregation methods that guide the user through large datasets or define a good default aggregation.

Beyond aggregation, there is also potential in visual analytics approaches that combine data mining methods with the visualization of group structures in graphs into an interactive approach. Clustering and classification algorithms could provide alternative group structures on demand. To update the data in a comprehensible way, the visualization needs to adapt on the fly, which introduces new visualization challenges. Similar updates are required when the users edit the group structures interactively, for instance, by applying set operations to the groups.

10.5. Tasks and evaluation

To choose the right type of group structure visualization for a particular application, we need to be aware of the tasks users want to solve with the help of the visualization. Application-specific tasks can be generalized to abstract data tasks, generalizable to different applications. There was already work done for disjoint flat groups [SSK14], which we extended to overlapping groups ꝏ, hierarchical structures ⊤, and dynamic groups ⤴ as well as to edge group structures (Section 7 summarizes tasks from papers of our bibliography). Also, it is important to study how basic data-related tasks are composed to complex task and which complex tasks are most relevant in specific areas of application. Then, it can be investigated which visualization technique is suitable for which application. Some evaluations have already been conducted (Section 8) but cover the techniques discussed as part of our taxonomy only partially. Advanced evaluation methods to better understand perceptive and cognitive processes such as such eye tracking [KFBW14] have rarely been applied in the field [JRHT14].

11. Conclusions

We presented the state of the art in explicitly visualizing vertex or edge group structures in graphs. Groups are disjoint or overlapping, and might be flat or structured hierarchically. In this survey, we brought together various group visualization techniques for graphs that have been discussed separately, so far. Based on the collected set of publications comprising all these techniques, we derived a taxonomy of visualization techniques consisting of four main categories: *visual node attributes* encode group information in the appearance of a node, *juxtaposed* approaches visualize graph and group structure in separate views, *superimposed* techniques use visual overlays and *embedded* representations combine the graphs and groups into an integrated visualization. In addition, we collected and abstracted group-related tasks described in the papers of our bibliography and categorized them with respect to the type of group structure and type of task including *group-only*, *group–vertex*, *group–edge*, and *group–network tasks*. We intersected the visualization taxonomy with a taxonomy of group structures to a lineup showing which visualization has been already used for what type of structure. The comparison hints at opportunities to fill gaps in existing research literature. Based on interviews with experts in the field, we identified important challenges that could guide future research.

Acknowledgements

We would like to thank all experts of our interviews for the feedback on our taxonomy and the discussions of future challenges: Bilal Alsallakh, Daniel Archambault, Michael Burch, Carsten Görg, Stephen G. Kobourov, Frank van Ham and Jarke J. van Wijk. This work was supported by DFG within SFB 716/D.5.

References

[ABZD13] Abuthawabeh A., Beck F., Zeckzer D., Diehl S.: Finding structures in multi-type code couplings with node-link and matrix visualizations. In *Proceedings of the 1st IEEE Working Conference on Software Visualization* (2013), VISSOFT '13, IEEE, pp. 1–10.

[ACJM03] Auber D., Chiricota Y., Jourdan F., Melançon G.: Multiscale visualization of small world networks. In *Proceedings of the 9th Symposium on Information Visualization* (2003), IV '03, IEEE, pp. 75–81.

[AFH*10] Ahmed A., Fu X., Hong S.-H., Nguyen Q., Xu K.: Visual analysis of history of world cup: A dynamic network with dynamic hierarchy and geographic clustering. In *Visual Information Communication*. Springer, 2010, pp. 25–39.

[AHRRC11] Alper B., Henry Riche N., Ramos G., Czerwinski M.: Design study of linesets, a novel set visualization technique. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2259–2267.

[AKY05] Abello J., Kobourov S. G., Yusufov R.: Visualizing large graphs with compound-fisheye views and treemaps. In *Proceedings of the 12th International Symposium on Graph Drawing*, vol. 3383 of *GD '04*. Springer, 2005, pp. 431–441.

[AMA07a] ARCHAMBAULT D., MUNZNER T., AUBER D.: Grouse: Feature-based, steerable graph hierarchy exploration. In *Proceedings of the 9th Joint Eurographics/IEEE VGTC Conference on Visualization* (2007), EuroVis '07, Eurographics Association, pp. 67–74.

[AMA07b] ARCHAMBAULT D., MUNZNER T., AUBER D.: TopoLayout: Multilevel graph layout by topological features. *IEEE Transactions on Visualization and Computer Graphics 13*, 2 (2007), 305–317.

[AMA08] ARCHAMBAULT D., MUNZNER T., AUBER D.: GrouseFlocks: Steerable exploration of graph hierarchy space. *IEEE Transactions on Visualization and Computer Graphics 14*, 4 (2008), 900–913.

[AMA09] ARCHAMBAULT D., MUNZNER T., AUBER D.: TugGraph: Path-preserving hierarchies for browsing proximity and paths in graphs. In *Proceedings of the IEEE Pacific Visualization Symposium* (2009), PacificVis '09, IEEE, pp. 113–120.

[AMA11] ARCHAMBAULT D., MUNZNER T., AUBER D.: Tugging graphs faster: Efficiently modifying path-preserving hierarchies for browsing paths. *IEEE Transactions on Visualization and Computer Graphics 17*, 3 (2011), 276–289.

[AMA*14] ALSALLAKH B., MICALLEF L., AIGNER W., HAUSER H., MIKSCH S., RODGERS P.: Visualizing sets and set-typed data: State-of-the-art and future challenges. In *EuroVis - STARs* (2014), EuroVis '14, Eurographics Association, pp. 1–21.

[APP10] ARCHAMBAULT D., PURCHASE H. C., PINAUD B.: The readability of path-preserving clusterings of graphs. *Computer Graphics Forum 29*, 3 (2010), 1173–1182.

[ASH14] ALTARAWNEH R., SCHULTZ J., HUMAYOUN S. R.: CluE: An algorithm for expanding clustered graphs. In *Proceedings of the IEEE Pacific Visualization Symposium* (2014), PacificVis '14, IEEE, pp. 233–237.

[AvHK06] ABELLO J., VAN HAM F., KRISHNAN N.: ASK-GraphView: A large scale graph visualization system. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 669–676.

[AZ13] ABUTHAWABEH A., ZECKZER D.: IMMV: An interactive multi-matrix visualization for program comprehension. In *Proceedings of the 1th IEEE Working Conference on Software Visualization* (2013), VISSOFT '13, IEEE, pp. 1–4.

[BALJ06] BOURQUI R., AUBER D., LACROIX V., JOURDAN F.: Metabolic network visualization using constraint planar graph drawing algorithm. In *Proceedings of the 10th International Conference on Information Visualization* (2006), IV '06, IEEE, pp. 489–496.

[BBDW16] BECK F., BURCH M., DIEHL S., WEISKOPF D.: A taxonomy and survey of dynamic graph visualization. *Computer Graphics Forum* (2016), http://dx.doi.org/10.1111/cgf.12791.

[BBT06] BYELAS H., BONDAREV E., TELEA A.: Visualization of areas of interest in component-based system architectures. In *Proceedings of the 32nd EUROMICRO Conference on Software Engineering and Advanced Applications* (2006), SEAA '06, IEEE, pp. 160–169.

[BBV*12] BECK F., BURCH M., VEHLOW C., DIEHL S., WEISKOPF D.: Rapid serial visual presentation in dynamic graph visualization. In *Proceedings of the Symposium on Visual Languages and Human-Centric Computing* (2012), VL/HCC '12, IEEE, pp. 185–192.

[BC01] BROCKENAUER R., CORNELSEN S.: Drawing clusters and hierarchies. In *Drawing Graphs, Methods and Models*. Springer, 2001, pp. 193–227.

[BCL*07] BOURQUI R., COTTRET L., LACROIX V., AUBER D., MARY P., SAGOT M.-F., JOURDAN F.: Metabolic network visualization eliminating node redundance and preserving metabolic pathways. *BMC Systems Biology 1*, 1 (2007), 1–19.

[BD05] BALZER M., DEUSSEN O.: Exploring relations within software systems using treemap enhanced hierarchical graphs. In *Proceedings of the 3rd IEEE International Workshop on Visualizing Software for Understanding and Analysis* (2005), VISSOFT '05, IEEE, pp. 1–6.

[BD07] BALZER M., DEUSSEN O.: Level-of-detail visualization of clustered graph layouts. In *Proceedings of the 6th International Asia-Pacific Symposium on Visualization* (2007), APVIS '07, IEEE, pp. 133–140.

[BD08] BURCH M., DIEHL S.: TimeRadarTrees: Visualizing dynamic compound digraphs. *Computer Graphics Forum 27*, 3 (2008), 823–830.

[BD13] BECK F., DIEHL S.: Visual comparison of software architectures. *Information Visualization 12*, 2 (2013), 178–199.

[Ber11] BERTIN J.: *Semiology of Graphics: Diagrams, Networks, Maps*. ESRI Press, 2011.

[BETT98] BATTISTA G. D., EADES P., TAMASSIA R., TOLLIS I. G.: *Graph Drawing: Algorithms for the Visualization of Graphs*, 1st ed. Prentice Hall PTR, 1998.

[BFBD10] BURCH M., FRITZ M., BECK F., DIEHL S.: TimeSpiderTrees: A novel visual metaphor for dynamic compound graphs. In *Proceedings of the Symposium on Visual Languages and Human-Centric Computing* (2010), VL/HCC '10, IEEE, pp. 168–175.

[BHW11] BURCH M., HÖFERLIN M., WEISKOPF D.: Layered TimeRadarTrees. In *Proceedings of the 15th International Conference on Information Visualisation* (2011), IV '11, IEEE, pp. 18–25.

[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 14* (2008), 1253–1260.

[BMW15] BURCH M., MUNZ T., WEISKOPF D.: Edge-stacked timelines for visualizing dynamic weighted digraphs. In *Proceedings of the 6th International Conference on Information*

Visualization Theory and Applications (2015), IVAPP '15, SciTePress, pp. 93–100.

[BPD11] BECK F., PETKOV R., DIEHL S.: Visually exploring multidimensional code couplings. In *Proceedings of the 6th IEEE International Workshop on Visualizing Software for Understanding and Analysis* (2011), VISSOFT '11, IEEE, pp. 1–8.

[BPF14] BACH B., PIETRIGA E., FEKETE J.-D.: GraphDiaries: Animated transitions and temporal navigation for dynamic networks. *IEEE Transactions on Visualization and Computer Graphics 20*, 5 (2014), 740–754.

[BST03] BREITKREUTZ B.-J. J., STARK C., TYERS M.: Osprey: A network visualization system. *Genome Biology 4*, 3 (2003), 1–6.

[BSW13] BURCH M., SCHMIDT B., WEISKOPF D.: A matrix-based visualization for exploring dynamic compound digraphs. In *Proceedings of the 17th International Conference on Information Visualisation* (2013), IV '13, IEEE, pp. 66–73.

[BT06] BYELAS H., TELEA A.: Visualization of areas of interest in software architecture diagrams. In *Proceedings of the ACM Symposium on Software Visualization* (2006), SoftVis '06, ACM, pp. 105–114.

[BT09a] BYELAS H., TELEA A.: Towards realism in drawing areas of interest on architecture diagrams. *Journal of Visual Languages & Computing 20*, 2 (2009), 110–128.

[BT09b] BYELAS H., TELEA A.: Visualizing metrics on areas of interest in software architecture diagrams. In *Proceedings of the IEEE Pacific Visualization Symposium* (2009), PacificVis '09, IEEE, pp. 33–40.

[BVB*11] BURCH M., VEHLOW C., BECK F., DIEHL S., WEISKOPF D.: Parallel edge splatting for scalable dynamic graph visualization. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2344–2353.

[CC07] COLLINS C., CARPENDALE S.: VisLink: Revealing relationships amongst visualizations. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1192–1199.

[CCC02] CHANG C.-L., CHEN D.-Y., CHUANG T.-R.: Browsing newsgroups with a social network analyzer. In *Proceedings of the 6th International Conference on Information Visualisation* (2002), IV '02, IEEE, pp. 750–755.

[CDA*14] CHATURVEDI S., DUNNE C., ASHKTORAB Z., ZACHARIAH R., SHNEIDERMAN B.: Group-in-a-box meta-layouts for topological clusters and attribute-based groups: Space-efficient visualizations of network communities and their ties. *Computer Graphics Forum 33*, 8 (2014), 52–68.

[CLLT15] CAO N., LIN Y.-R., LI L., TONG H.: g-Miner: Interactive visual group mining on multivariate graphs. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (2015), CHI '15, ACM, pp. 279–288.

[CMF*14] CRNOVRSANIN T., MUELDER C. W., FARIS R., FELMLEE D., MA K.-L.: Visualization techniques for categorical analysis of social networks with multiple edge sets. *Social Networks 37* (2014), 56–64.

[CPC09] COLLINS C., PENN G., CARPENDALE M. S. T.: Bubble Sets: Revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1009–1016.

[CSL*10] CAO N., SUN J., LIN Y.-R., GOTZ D., LIU S., QU H.: FacetAtlas: Multifaceted visualization for rich text corpora. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (2010), 1172–1181.

[DC11] DAVIS D. A., CHAWLA N. V.: Exploring and exploiting disease interactions from multi-relational gene and phenotype networks. *PLoS ONE 6*, 7 (07 2011), e22670.

[Dek01] DEKKER A. H.: Visualisation of social networks using CAVALIER. In *Proceedings of the Australasian Symposium on Information Visualisation* (2001), APVIS '01, IEEE, pp. 49–55.

[DEKB*14] DINKLA K., EL-KEBIR M., BUCUR C.-I., SIDERIUS M., SMIT M., WESTENBERG M. A., KLAU G.: eXamine: Exploring annotated modules in networks. *BMC Bioinformatics 15*, 201 (2014), 1–14.

[DGC*05] DOGRUSOZ U., GIRAL E., CETINTAS A., CIVRIL A., DEMIR E.: A compound graph layout algorithm for biological pathways. In *Proceedings of the 12th International Symposium on Graph Drawing*, vol. 3383 of *GD '04*. Springer, 2005, pp. 442–447.

[DHRMM13] DWYER T., HENRY Riche N., MARRIOTT K., MEARS C.: Edge compression techniques for visualization of dense directed graphs. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2596–2605.

[DHRW15] DINKLA K., HENRY Riche N., WESTENBERG M.: Dual Adjacency Matrix: Exploring link groups in dense networks. *Computer Graphics Forum 34*, 3 (2015), 311–320.

[DKL13] DAI B. T., KWEE A., LIM E.-P.: ViStruclizer: A structural visualizer for multi-dimensional social networks. In *Advances in Knowledge Discovery and Data Mining*. J. Pei, V. S. Tseng, L. Cao, H. Motoda and G. Xu (Eds.), vol. 7819 of *Lecture Notes in Computer Science*. Springer, 2013, pp. 49–60.

[DKM06] DWYER T., KOREN Y., MARRIOTT K.: IPSep-CoLa: An incremental procedure for separation constraint layout of graphs. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 821–828.

[DLM14] DIDIMO W., LIOTTA G., MONTECCHIANI F.: Network visualization for financial crime detection. *Journal of Visual Languages & Computing 25*, 4 (2014), 433–451.

[DLR10] DIDIMO W., LIOTTA G., ROMEO S. A.: Graph visualization techniques for conceptual web site traffic analysis. In *Proceedings of the IEEE Pacific Visualization Symposium* (March 2010), PacificVis '10, pp. 193–200.

[DM12] Didimo W., Montecchiani F.: Fast layout computation of hierarchically clustered networks: Algorithmic advances and experimental analysis. In *Proceedings of the 16th International Conference on Information Visualisation* (2012), IV '12, IEEE, pp. 18–23.

[DM14a] Didimo W., Montecchiani F.: Fast layout computation of clustered networks: Algorithmic advances and experimental analysis. *Information Sciences 260* (2014), 185–199.

[DM14b] Dkaki T., Mothe J.: An energy-based model to optimize cluster visualization. In *Proceedings of the 8th International Conference on Research Challenges in Information Science* (2014), RCIS '14, IEEE, pp. 1–11.

[DPS02] Díaz J., Petit J., Serna M.: A survey of graph layout problems. *ACM Computing Surveys 34*, 3 (2002), 313–356.

[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* (2013), CHI '13, ACM, pp. 3247–3256.

[DvKSW12] Dinkla K., van Kreveld M. J., Speckmann B., Westenberg M. A.: Kelp Diagrams: Point set membership visualization. *Computer Graphics Forum 31*, 3 (2012), 875–884.

[DWS*14] Daniel D. T., Wuchner E., Sokolov K., Stal M., Liggesmeyer P.: Polyptychon: A hierarchically-constrained classified dependencies visualization. In *Proceedings of the 2nd IEEE Working Conference on Software Visualization* (2014), VISSOFT '14, IEEE, pp. 83–86.

[DYL*15] Du X., Ye Y., Lau R. Y., Li Y., Huang X.: Multi-opinion ring: Visualizing and predicting multiple opinion orientations in online social media. *Multimedia Tools and Applications* (2015), 1–28.

[DYLL15] Du X., Ye Y., Lau R. Y., Li Y.: OpinionRings: Inferring and visualizing the opinion tendency of socially connected users. *Decision Support Systems 75* (2015), 11–24.

[EDG*08] Elmqvist N., Do T.-N., Goodell H., Henry N., Fekete J.-D.: ZAME: Interactive large-scale graph visualization. In *Proceedings of the IEEE Pacific Visualization Symposium* (2008), PacificVis '08, IEEE, pp. 215–222.

[EF97] Eades P., Feng Q.-W.: Multilevel visualization of clustered graphs. In *Proceedings of the 4th International Symposium on Graph Drawing*, vol. 1190 of *GD '96*. Springer, 1997, pp. 101–112.

[EF10] Elmqvist N., Fekete J.-D.: Hierarchical aggregation for information visualization: Overview, techniques, and design guidelines. *IEEE Transactions on Visualization and Computer Graphics 16*, 3 (2010), 439–454.

[EFN99] Eades P., Feng Q.-W., Nagamochi H.: Drawing clustered graphs on an orthogonal grid. *Journal of Graph Algorithms and Applications 3*, 4 (1999), 3–29.

[EH00] Eades P., Huang M. L.: Navigating clustered graphs using force-directed methods. *Journal of Graph Algorithms and Applications 4*, 3 (2000), 157–181.

[EHKP14] Efrat A., Hu Y., Kobourov S. G., Pupyrev S.: MapSets: Visualizing embedded and clustered graphs. In *Proceedings of the 22nd International Symposium on Graph Drawing*, vol. 8871 of *GD '14*. Springer, 2014, pp. 452–463.

[EHP*11] Ersoy O., Hurter C., Paulovich F. V., Cantareiro G., Telea A.: Skeleton-based edge bundling for graph visualization. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2364–2373.

[ET07] Elmqvist N., Tsigas P.: CiteWiz: A tool for the visualization of scientific citation networks. *Information Visualization 6*, 3 (2007), 215–232.

[ETB11] Erdemir U., Tekin U., Buzluca F.: E-Quality: A graph based object oriented software quality visualization tool. In *Proceedings of the 6th IEEE International Workshop on Visualizing Software for Understanding and Analysis* (2011), VISSOFT '11, pp. 1–8.

[FBS06] Falkowski T., Bartelheimer J., Spiliopoulou M.: Mining and visualizing the evolution of subgroups in social networks. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence* (2006), WI '06, IEEE, pp. 52–58.

[FGB*07] Freeman T. C., Goldovsky L., Brosch M., van Dongen S., Maziére P., Grocock R. J., Freilich S., Thornton J., Enright A. J.: Construction, visualisation, and clustering of transcription networks from microarray expression data. *PLoS Computational Biology 3*, 10 (2007).

[FKH15] Fittkau F., Krause A., Hasselbring W.: Hierarchical software landscape visualization for system comprehension: A controlled experiment. In *Proceedings of the 3rd IEEE Working Conference on Software Visualization* (2015), VISSOFT '15, IEEE, pp. 36–45.

[For10] Fortunato S.: Community detection in graphs. *Physics Reports 486*, 3-5 (2010), 75–174.

[FR91] Fruchterman T. M. J., Reingold E. M.: Graph drawing by force-directed placement. *Software: Practice and Experience 21*, 11 (1991), 1129–1164.

[FT04] Frishman Y., Tal A.: Dynamic drawing of clustered graphs. In *Proceedings of the 10th IEEE Symposium on Information Visualization* (2004), InfoVis '04, IEEE, pp. 191–198.

[FWD*03] Fekete J.-D., Wang T. D., Dang N., Aris A., Plaisant C.: Overlaying graph links on treemaps. In *Proceedings of the 9th IEEE Symposium on Information Visualization, Poster Compendium* (2003), InfoVis '03, IEEE.

[GBD09] Greilich M., Burch M., Diehl S.: Visualizing the evolution of compound digraphs with TimeArcTrees. *Computer Graphics Forum 28*, 3 (2009), 975–982.

[GF03] Ghoniem M., Fekete J.-D.: Matrix view of graphs and direct manipulation of cluster hierarchies. In *Proceedings of the 15th French-speaking Conference on Human-computer Interaction* (2003), IHM '03, ACM, pp. 206–207.

[GF11] Gibson H., Faith J.: Node-attribute graph layout for small-world networks. In *Proceedings of the 15th International Conference on Information Visualisation* (2011), IV '11, IEEE, pp. 482–487.

[GFK*14] Gerasch A., Faber D., Küntzer J., Niermann P., Kohlbacher O., Lenhof H.-P., Kaufmann M.: BiNA: a visual analytics tool for biological network data. *PLoS ONE 9*, 2 (2014), e87397.

[GFV13] Gibson H., Faith J., Vickers P.: A survey of two-dimensional graph layout techniques for information visualisation. *Information Visualization 12*, 3-4 (2013), 324–357.

[GHK10] Gansner E. R., Hu Y., Kobourov S. G.: GMap: Visualizing graphs and clusters as maps. In *Proceedings of the IEEE Pacific Visualization Symposium* (2010), PacificVis '10, IEEE, pp. 201–208.

[GHM*02] Goesmann A., Haubrock M., Meyer F., Kalinowski J., Giegerich R.: PathFinder: Reconstruction and dynamic visualization of metabolic pathways. *Bioinformatics 18*, 1 (2002), 124–129.

[GK07] Gansner E. R., Koren Y.: Improved circular layouts. In *Proceedings of the 15th International Symposium on Graph Drawing*, vol. 4372 of *GD '07*. Springer, 2007, pp. 386–398.

[GMT09] Gao J., Misue K., Tanaka J.: A multiple-aspects visualization tool for exploring social networks. In *Proceedings of the Symposium on Human Interface and the Management of Information and Interaction*, vol. 5618 of *HCI '09*. Springer, 2009, pp. 277–286.

[GSZ*11] Gilbert F., Simonetto P., Zaidi F., Jourdan F., Bourqui R.: Communities and hierarchical structures in dynamic social networks: Analysis and visualization. *Social Network Analysis and Mining 1*, 2 (2011), 83–95.

[GZ11] Gou L., Zhang X.: TreeNetViz: Revealing patterns of networks over tree structures. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2449–2458.

[HB05] Heer J., Boyd D.: Vizster: Visualizing online social networks. In *Proceedings of the 11th IEEE Symposium on Information Visualization* (2005), InfoVis '05, IEEE, pp. 32–39.

[HBF08] Henry N., Bezerianos A., Fekete J.-D.: Improving the readability of clustered social networks using node duplication. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1317–1324.

[HCvW07] Holten D., Cornelissen B., van Wijk J. J.: Trace visualization using hierarchical edge bundles and massive sequence views. In *Proceedings of the 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis* (2007), VISSOFT '07, IEEE, pp. 47–54.

[HD12] Hascoët M., Dragicevic P.: Interactive graph matching and visual comparison of graphs and clustered graphs. In *Proceedings of the International Working Conference on Advanced Visual Interfaces* (2012), AVI '12, ACM, pp. 522–529.

[Hea96] Healey C. G.: Choosing effective colours for data visualization. In *Proceedings of the 7th Conference on Visualization* (1996), VIS '96, IEEE Computer Society Press, pp. 263–270.

[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.

[HFM07] Henry N., Fekete J.-D., McGuffin M. J.: NodeTrix: A hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics 13*, 6 (2007), 1302–1309.

[HGK10] Hu Y., Gansner E. R., Kobourov S. G.: Visualizing graphs and clusters as maps. *IEEE Computer Graphics and Applications 30*, 6 (2010), 54–66.

[HKV14] Hu Y., Kobourov S. G., Veeramoni S.: Embedding, clustering and coloring for dynamic maps. *Journal of Graph Algorithms and Applications 18*, 1 (2014), 77–109.

[HMM00] Herman I., Melancon G., Marshall M.: Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics 6*, 1 (2000), 24–43.

[HN07a] Huang M. L., Nguyen Q. V.: Navigating large clustered graphs with triple-layer display. In *Proceedings of the 11th International Conference on Information Visualization* (2007), IV '07, IEEE, pp. 684–692.

[HN07b] Huang M. L., Nguyen Q. V.: A space efficient clustered visualization of large graphs. In *Proceedings of the 4th International Conference on Image and Graphics* (2007), ICIG '07, IEEE, pp. 920–927.

[Hol06] Holten D.: Hierarchical Edge Bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 741–748.

[HRD10] Henry Riche N., Dwyer T.: Untangling Euler diagrams. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (2010), 1090–1099.

[HSS15] Hadlak S., Schumann H., Schulz H.-J.: A survey of multi-faceted graph visualization. In *Eurographics Conference on Visualization (EuroVis) - STARs* (2015), EuroVis '15, Eurographics Association, pp. 1–20.

[IMMS09] Itoh T., Muelder C. W., Ma K.-L., Sese J.: A hybrid space-filling and force-directed layout method for visualizing multiple-category graphs. In *Proceedings of the IEEE Pacific*

Visualization Symposium (2009), PacificVis '09, IEEE, pp. 121–128.

[JJ10] Jansson A., Jirstrand M.: Biochemical modeling with systems biology graphical notation. *Drug Discovery Today 15*, 9-10 (2010), 365–370.

[JKS06] Junker B., Klukas C., Schreiber F.: VANTED: A system for advanced data analysis and visualization in the context of biological networks. *BMC Bioinformatics 7* (2006), 109–121.

[JRHT14] Jianu R., Rusu A., Hu Y., Taggart D.: How to display group information on node-link diagrams: An evaluation. *IEEE Transactions on Visualization and Computer Graphics 20*, 11 (2014), 1530–2626.

[KFBW14] Kurzhals K., Fisher B., Burch M., Weiskopf D.: Evaluating visual analytics with eye tracking. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization* (2014), BELIV '14, ACM, pp. 61–69.

[KG06] Kumar G., Garland M.: Visual exploration of complex time-varying graphs. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 805–812.

[Kit03] Kitano H.: A graphical notation for biochemical networks. *BIOSILICO 1*, 5 (2003), 169–176.

[KK89] Kamada T., Kawai S.: An algorithm for drawing general undirected graphs. *Information Processing Letters 31*, 1 (1989), 7–15.

[LDB11] Lambert A., Dubois J., Bourqui R.: Pathway preserving representation of metabolic networks. *Computer Graphics Forum 30*, 3 (2011), 1021–1030.

[Lii10] Liiv I.: Seriation and matrix reordering methods: An historical overview. *Statistical Analysis and Data Mining 3*, 2 (2010), 70–91.

[LPP*06] Lee B., Plaisant C., Parr C. S., Fekete J.-D., Henry N.: Task taxonomy for graph visualization. In *Proceedings of 2006 AVI workshop on BEyond time and errors: Novel evaluation methods for information visualization* (2006), BELIV '06, ACM, pp. 1–5.

[LQB12] Lambert A., Queyroi F., Bourqui R.: Visualizing patterns in node-link diagrams. In *Proceedings of the 16th International Conference on Information Visualisation* (2012), IV '12, IEEE, pp. 48–53.

[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 Pacific Visualization Symposium* (2010), PacificVis '10, IEEE, pp. 57–64.

[LWC*14] Liu S., Wang X., Chen J., Zhu J., Guo B.: TopicPanorama: a full picture of relevant topics. In *Proceedings of the Conference on Visual Analytics Science and Technology* (2014), vol. 20 of *VAST '14*, IEEE, pp. 183–192.

[LYKB08] Letunic I., Yamada T., Kanehisa M., Bork P.: iPath: Interactive exploration of biochemical pathways and networks. *Trends in Biochemical Sciences 33*, 3 (2008), 101–103.

[MGK11] Melville A., Graham M., Kennedy J.: Combined vs. separate views in matrix-based graph analysis and comparison. In *Proceedings of the 15th International Conference on Information Visualisation* (2011), IV '11, IEEE, pp. 53–58.

[MH15] Meidiana A., Hong S.-H.: MultiStory: Visual analytics of dynamic multi-relational networks. In *Proceedings of the IEEE Pacific Visualization Symposium* (2015), PacificVis '15, IEEE, pp. 75–79.

[MML07] Mueller C., Martin B., Lumsdaine A.: A comparison of vertex ordering algorithms for large graph visualization. In *Proceedings of the 6th International Asia-Pacific Symposium on Visualization* (2007), APVIS '07, IEEE, pp. 141–148.

[MZ11] Misue K., Zhou Q.: Drawing semi-bipartite graphs in anchor+matrix style. In *Proceedings of the 15th International Conference on Information Visualisation* (2011), IV '11, IEEE, pp. 26–31.

[NIS15] Nakazawa R., Itoh T., Saito T.: A visualization of research papers based on the topics and citation network. In *Proceedings of the 19th International Conference on Information Visualisation* (2015), IV '15, IEEE, pp. 283–289.

[NIST12] Nakazawa R., Itoh T., Sese J., Terada A.: Integrated visualization of gene network and ontology applying a hierarchical graph visualization technique. In *Proceedings of the 16th International Conference on Information Visualisation* (2012), IV '12, IEEE, pp. 81–86.

[Noa07] Noack A.: Energy models for graph clustering. *Journal of Graph Algorithms and Applications 11*, 2 (2007), 453–480.

[NSC05] Neumann P., Schlechtweg S., Carpendale S.: ArcTrees: Visualizing relations in hierarchical data. In *Proceedings of the 7th Joint Eurographics/IEEE VGTC Conference on Visualization* (2005), EuroVis '05, Eurographics Association, pp. 53–60.

[OMB*07] Ogawa M., Ma K., Bird C., Devanbu P., Gourley A.: Visualizing social interaction in open source software projects. In *Prooceedings of 6th International Asia-Pacific Symposium on Visualization* (2007), APVIS '07, IEEE, pp. 25–32.

[PDF14] Perin C., Dragicevic P., Fekete J.-D.: Revisiting Bertin matrices: New interactions for crafting tabular visualizations. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 2082–2091.

[PF15] Paduano F., Forbes A. G.: Extended LineSets: A visualization technique for the interactive inspection of biological pathways. In *Proceedings of the 5th Symposium on Biological Data Visualization* (2015), vol. 9 of *BioVis '15*, BMC, p. S4.

[PGKG08] Pinzger M., Grafenhain K., Knab P., Gall H. C.: A tool for visual understanding of source code dependencies. In

Proceedings of the 16th International Conference on Program Comprehension (2008), ICPC '08, IEEE, pp. 254–259.

[PK06] Paley S. M., Karp P. D.: The pathway tools cellular overview diagram and omics viewer. *Nucleic Acids Research 34*, 13 (2006), 3771–3778.

[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.

[PS06] Perer A., Shneiderman B.: Balancing systematic and flexible exploration of social networks. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 693–700.

[PSK11] Peng W., Si-Kun L.: Social network analysis layout algorithm under ontology model. *Journal of Software 6*, 7 (2011), 1321–1328.

[Pur02] Purchase H. C.: Metrics for graph drawing aesthetics. *Journal of Visual Languages & Computing 13*, 5 (2002), 501–516.

[PvW06] Pretorius A. J., van Wijk J. J.: Visual analysis of multivariate state transition graphs. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 685–692.

[PvW07] Pretorius A. J., van Wijk J. J.: Bridging the semantic gap: Visualizing transition graphs with user-defined diagrams. *IEEE Computer Graphics and Applications 27*, 5 (2007), 58–66.

[PvW08] Pretorius A. J., van Wijk J. J.: Visual inspection of multivariate graphs. *Computer Graphics Forum 27*, 3 (2008), 967–974.

[RB10] Rosvall M., Bergstrom C.: Mapping change in large networks. *PloS one 5*, 1 (2010), e8694.

[RFG05] Ratzinger J., Fischer M., Gall H. C.: EvoLens: lens-view visualizations of evolution data. In *Proceedings of the 8th International Workshop on Principles of Software Evolution* (2005), IWPSE '05, IEEE, pp. 103–112.

[RHR*10] Rohrschneider M., Heine C., Reichenbach A., Kerren A., Scheuermann G.: A novel grid-based visualization approach for metabolic networks with advanced focus&context view. In *Proceedings of the 17th International Symposium on Graph Drawing*, vol. 5849 of *GD '09*. Springer, 2010, pp. 268–279.

[RMF12] Rufiange S., McGuffin M. J., Fuhrman C. P.: TreeMatrix: A hybrid visualization of compound graphs. *Computer Graphics Forum 31*, 1 (2012), 89–101.

[RMM15] Renoust B., Melançon G., Munzner T.: Detangler: Visual analytics for multiplex networks. *Computer Graphics Forum 34*, 3 (2015), 321–330.

[RPD09] Reitz F., Pohl M., Diehl S.: Focused animation of dynamic compound graphs. In *Proceedings of the 13th International Conference on Information Visualisation* (2009), IV '09, IEEE, pp. 679–684.

[RRAS08] Royer L., Reimann M., Andreopoulos B., Schroeder M.: Unraveling protein networks with power graph analysis. *PLoS Computational Biology 4*, 7 (2008), e1000108.

[RTJ*11] Reda K., Tantipathananandh C., Johnson A., Leigh J., Berger-Wolf T.: Visualizing the evolution of community structures in dynamic social networks. *Computer Graphics Forum 30*, 3 (2011), 1061–1070.

[SA06] Shneiderman B., Aris A.: Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics 12*, 5 (2006), 733–740.

[SBG00] Sprenger T. C., Brunella R., Gross M. H.: H-BLOB: A hierarchical visual clustering method using implicit surfaces. In *Proceedings of the Conference on Visualization* (2000), VIS '00, IEEE, pp. 61–68.

[SCL*09] Shi L., Cao N., Liu S., Qian W., Tan L., Wang G., Sun J., Lin C.-Y.: HiMap: Adaptive visualization of large-scale online social networks. In *Proceedings of the IEEE Pacific Visualization Symposium* (2009), PacificVis '09, IEEE, pp. 41–48.

[SGKS15] Stitz H., Gratzl S., Krieger M., Streit M.: CloudGazer: A divide-and-conquer approach to monitoring and optimizing cloud-based networks. In *Proceedings of the IEEE Pacific Visualization Symposium* (2015), PacificVis '15, IEEE, pp. 175–182.

[SJSJ05] Sangal N., Jordan E., Sinha V., Jackson D.: Using dependency models to manage complex software architecture. In *Proceedings of the 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications* (2005), OOPSLA '05, ACM, pp. 167–176.

[SJUS08] Schulz H.-J., John M., Unger A., Schumann H.: Visual analysis of bipartite biological networks. In *Proceedings of the 1st Eurographics Conference on Visual Computing for Biomedicine* (2008), EG VCBM '08, Eurographics Association, pp. 135–142.

[SKB*14] Schreiber F., Kerren A., Börner K., Hagen H., Zeckzer D.: Heterogeneous networks on multiple levels. In *Multivariate Network Visualization*, vol. 8380. Springer, 2014, pp. 175–206.

[SKL*14] Stolper C. D., Kahng M., Lin Z., Foerster F., Goel A., Stasko J., Chau D. H.: GLO-STIX: Graph-level operations for specifying techniques and interactive exploration. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 2320–2328.

[SLAB15] Sansen J., Lalanne F., Auber D., Bourqui R.: Adjasankey: Visualization of huge hierarchical weighted and directed graphs. In *Proceedings of the 19th International Conference on Information Visualisation* (2015), IV '15, IEEE, pp. 211–216.

[SLK*09] Streit M., Lex A., Kalkusch M., Zatloukal K., Schmalstieg D.: Caleydo: Connecting pathways and gene expression. *Bioinformatics 25*, 20 (2009), 2760–2761.

[SMER06] Shen Z., Ma K.-L., Eliassi-Rad T.: Visual analysis of large heterogeneous social networks by semantic and structural abstraction. *IEEE Transactions on Visualization and Computer Graphics 12*, 6 (2006), 1427–1439.

[SMM13] SALLABERRY A., MUELDER C. W., MA K.-L.: Clustering, visualizing, and navigating for large dynamic graphs. In *Proceedings of the 20th International Symposium on Graph Drawing*, vol. 7704 of *GD '12*. Springer, 2013, pp. 487–498.

[SSK14] SAKET B., SIMONETTO P., KOBOUROV S.: Group-level graph visualization taxonomy. In *EuroVis - Short Papers* (2014), EuroVis '14, The Eurographics Association.

[SSKB14] SAKET B., SIMONETTO P., KOBOUROV S. G., BÖRNER K.: Node, node-link, and node-link-group diagrams: An evaluation. *IEEE Transactions on Visualization and Computer Graphics 20*, 12 (2014), 2231–2240.

[SSKB15] SAKET B., SCHEIDEGGER C., KOBOUROV S. G., BÖRNER K.: Map-based visualizations increase recall accuracy of data. *Computer Graphics Forum 34*, 3 (2015), 441–450.

[ST08] SANTAMARÍA R., THERÓN R.: Overlapping clustered graphs: Co-authorship networks visualization. In *Proceedings of the 8th International Symposium on Smart Graphics*, vol. 5166 of *SG '08*. Springer, 2008, pp. 190–199.

[SXS*12] SMITH A., XU W., SUN Y., FAEDER J., MARAI G. E.: RuleBender: Integrated modeling, simulation and visualization for rule-based intracellular biochemistry. *BMC Bioinformatics 13*, Suppl 8 (2012), S3.

[SZPM10] SALLABERRY A., ZAIDI F., PICH C., MELANÇON G.: Interactive visualization and navigation of web search results revealing community structures and bridges. In *Proceedings of the Graphics Interface Conference* (2010), GI '10, ACM, pp. 105–112.

[TE10] TELEA A., ERSOY O.: Image-based edge bundles: Simplified visualization of large graphs. In *Proceedings of the 12th Joint Eurographics/IEEE VGTC Conference on Visualization* (2010), EuroVis '10, Eurographics Association, pp. 843–852.

[TLTC05] TERMEER M., LANGE C. F. J., TELEA A., CHAUDRON M. R. V.: Visual exploration of combined architectural and metric information. In *Proceedings of the 3rd IEEE International Workshop on Visualizing Software for Understanding and Analysis* (2005), VISSOFT '05, IEEE, pp. 1–6.

[TvDEF09] THEOCHARIDIS A., VAN DONGEN S., ENRIGHT A. J., FREEMAN T. C.: Network visualization and analysis of gene expression data using BioLayout Express(3D). *Nature Protocols 4* (2009), 1535–1550.

[VBAW15] VEHLOW C., BECK F., AUWÄRTER P., WEISKOPF D.: Visualizing the evolution of communities in dynamic graphs. *Computer Graphics Forum 34*, 1 (2015), 277–288.

[VBSW13] VEHLOW C., BURCH M., SCHMAUDER H., WEISKOPF D.: Radial layered matrix visualization of dynamic graphs. In *Proceedings of the 17th International Conference on Information Visualisation* (2013), IV '13, IEEE, pp. 51–58.

[VBW15] VEHLOW C., BECK F., WEISKOPF D.: The state of the art in visualizing group structures in graphs. In *Eurographics Conference on Visualization - STARs* (2015), EuroVis '15, Eurographics Association, pp. 21–40.

[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 20*, 12 (2014), 2310–2319.

[vH03] VAN HAM F.: Using multilevel call matrices in large software projects. In *Proceedings of the 9th IEEE Symposium on Information Visualization* (2003), InfoVis '03, IEEE, pp. 227–232.

[VHK*13] VEHLOW C., HASENAUER J., KRAMER A., RAUE A., HUG S., TIMMER J., RADDE N., THEIS F. J., WEISKOPF D.: iVUN: Interactive visualization of uncertain biochemical reaction networks. *BMC Bioinformatics 14*, S-19 (2013), S2.

[vHSD09] VAN HAM F., SCHULZ H.-J., DIMICCO J. M.: Honeycomb: Visual analysis of large scale social networks. In *Proceedings of the 12th International Conference on Human-Computer Interaction*, vol. 5727 of *INTERACT '09*. Springer, 2009, pp. 429–442.

[VHTW13] VEHLOW C., HASENAUER J., THEIS F. J., WEISKOPF D.: Visualizing edge-edge relations in graphs. In *Proceedings of the IEEE Pacific Visualization Symposium* (2013), PacificVis '09, IEEE, pp. 201–208.

[vHvW04] VAN HAM F., VAN WIJK J. J.: Interactive visualization of small world graphs. In *Proceedings of the 10th IEEE Symposium on Information Visualization* (2004), InfoVis '04, IEEE, pp. 199–206.

[vHW08] VAN HAM F., WATTENBERG M.: Centrality based visualization of small world graphs. *Computer Graphics Forum 27*, 3 (2008), 975–982.

[VKB*15] VEHLOW C., KAO D. P., BRISTOW M. R., HUNTER L. E., WEISKOPF D., GÖRG C.: Visual analysis of biological data-knowledge networks. *BMC Bioinformatics 16*, 135 (2015).

[vLKS*11] VON LANDESBERGER T., KUIJPER A., SCHRECK T., KOHLHAMMER J., VAN WIJK J. J., FEKETE J.-D., FELLNER D. W.: Visual analysis of large graphs: State-of-the-art and future research challenges. *Computer Graphics Forum 30*, 6 (2011), 1719–1749.

[VPF*14] VIHROVS J., PRUSIS K., FREIVALDS K., RUCEVSKIS P., KREBS V.: An inverse distance-based potential field function for overlapping point set visualization. In *Proceedings of the 5th International Conference on Information Visualization Theory and Applications* (2014), IVAPP '14, SciTePress, pp. 29–38.

[VRW13] VEHLOW C., REINHARDT T., WEISKOPF D.: Visualizing fuzzy overlapping communities in networks. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2486–2495.

[War04] WARE C.: *Information Visualization: Perception for Design*. Morgan Kaufmann Publishers Inc., 2004.

[WWY*15] Wu Y., Wu W., Yang S., Yan Y., Qu H.: Interactive visual summary of major communities in a large network. In *Proceedings of the IEEE Pacific Visualization Symposium* (2015), PacificVis '15, IEEE, pp. 47–54.

[XDC*13] Xu P., Du F., Cao N., Shi C., Zhou H., Qu H.: Visual analysis of set relations in a graph. *Computer Graphics Forum 32*, 3 (2013), 61–70.

[XW05] Xu R., Wunsch D.: Survey of clustering algorithms. *IEEE Transactions on Neural Networks 16*, 3 (2005), 645–678.

[YDG*15] Yoghourdjian V., Dwyer T., Gange G., Kieffer S., Klein K., Marriott K.: High-quality ultra-compact grid layout of grouped networks. *IEEE Transactions on Visualization and Computer Graphics PP*, 99 (2015), 1–1.

[YLZ*13] Yang J., Liu Y., Zhang X., Yuan X., Zhao Y., Barlowe S., Liu S.: PIWI: Visually exploring graphs based on their community structure. *IEEE Transactions on Visualization and Computer Graphics 19*, 6 (2013), 1034–1047.

[YS15] Yamashita T., Saga R.: Edge bundling in multi-attributed graphs. In *Human Interface and the Management of Information. Information and Knowledge Design*, vol. 9172 of *Lecture Notes in Computer Science*. Springer, 2015, pp. 138–147.

[ZCCB13] Zhao J., Collins C., Chevalier F., Balakrishnan R.: Interactive exploration of implicit and explicit relations in faceted datasets. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2080–2089.

[Zec10] Zeckzer D.: Visualizing software entities using a matrix layout. In *Proceedings of the ACM Symposium on Software Visualization* (2010), SoftVis '10, ACM, pp. 207–208.

[ZXQ15] Zhou H., Xu P., Qu H.: Visualization of bipartite relations between graphs and sets. *Journal of Visualization 18*, 2 (2015), 159–172.

[ZXYQ13] Zhou H., Xu P., Yuan X., Qu H.: Edge bundling in information visualization. *Tsinghua Science and Technology 18*, 2 (2013), 145–156.