Journal of Computer Languages 57 (2020) 100959

Contents lists available at ScienceDirect

Journal of Computer Languages

journal homepage: www.editorialmanager.com/cola/default.aspx

Interactive exploration of large time-dependent bipartite graphs

Manuela Waldner*, Daniel Steinböck, Eduard Gröller

TU Wien, Institute of Visual Computing & Human-Centered Technology - E193-02, Favoritenstr. 9 / 5. Stock (5th floor), Vienna A-1040, Austria

ARTICLE INFO

Keywords:
Information visualization
Bipartite graphs
Clustering
Time series data
Insight-based evaluation

MSC:
00A66

ABSTRACT

Bipartite graphs are typically visualized using linked lists or matrices, but these visualizations neither scale well nor do they convey temporal development. We present a new interactive exploration interface for large, time-dependent bipartite graphs. We use two clustering techniques to build a hierarchical aggregation supporting different exploration strategies. Aggregated nodes and edges are visualized as linked lists with nested time series. We demonstrate two use cases: finding advertising expenses of public authorities following similar temporal patterns and comparing author-keyword co-occurrences across time. Through a user study, we show that linked lists with hierarchical aggregation lead to more insights than without.

1. Introduction

A bipartite graph is a special class of graphs, where the vertex (or node) set V of the graph $G = (V, E)$ can be partitioned into two disjoint nonempty sets V_1 and V_2, both of which are independent [1]. In a weighted bipartite graph, every edge connecting a node of V_1 with a node of V_2 has a weight of $\omega \geq 0$. Data sets representing bipartite graphs can be found in many disciplines, ranging from biology, where nodes represent genes and conditions [2–4], over document analysis, where nodes can represent different categories of named entities [5,6], to social network analysis, where nodes can be institutions and projects [7].

Visualizations of bipartite graphs can effectively reveal connections between the two sets of nodes. Classic visualization techniques, like linked lists or matrices, can display up to a few hundred nodes and edges. However, many data sets, such as the IEEE Visualization Publication collection [8], rather have thousands or tens of thousands of elements. Often, these data sets are of interest to a general lay audience, such as the *Media Transparency Database*, containing all media advertising expenses of public authorities in Austria since 2012 [9]. These data sets are collected over a longer period of time and therefore also contain an interesting temporal component. For instance, users might be interested to compare author-keyword relationships in a publication database across different time periods or to detect public authorities following similar advertisement trends over time. The goal of this work is therefore to find an easily understandable interactive visualization, which allows lay users to casually [10] explore connections in large bipartite graphs over time.

In this paper, we propose an interactive visualization technique *Dynamic Bipartite Cluster Flows* (*Dynamic BiCFlows*) combining hierarchical aggregation (i.e., hierarchical grouping of nodes) and filtering (i.e., removing nodes) to visualize large time-dependent bipartite graphs. The user can gradually drill down from an overview to the most detailed level. Depending on the exploration level and group size, we filter the groups to show only the most relevant items. In contrast to *BiCFlows* [11], *Dynamic BiCFlows* not only allows exploration of the connections between the two sets of nodes, but also the temporal development of the edges. To support such a temporal exploration, we introduce two clustering methods with interactive drill-down strategies and demonstrate the usefulness of these approaches with two application cases. In summary, our paper has three main contributions:

1. a new visualization and interaction design for interactive visual exploration of large, time-dependent bipartite graphs,
2. two different hierarchical aggregation and filtering approaches for linked list visualizations, supporting the exploration of the temporal development of bipartite graph data,
3. the results of an insight-based user study, where lay users explored a large bipartite graph containing advertising expenses of public organizations, showing that hierarchical aggregation encourages users to perform a longer exploration of the data, leading to more (unexpected) insights.

2. Related work

The most common visual encodings of graphs are node-link

* Corresponding author.
E-mail addresses: waldner@cg.tuwien.ac.at (M. Waldner), e0826088@student.tuwien.ac.at (D. Steinböck), groeller@cg.tuwien.ac.at (E. Gröller).

https://doi.org/10.1016/j.cola.2020.100959
Received 27 February 2019; Received in revised form 6 November 2019; Accepted 23 February 2020
Available online 13 March 2020

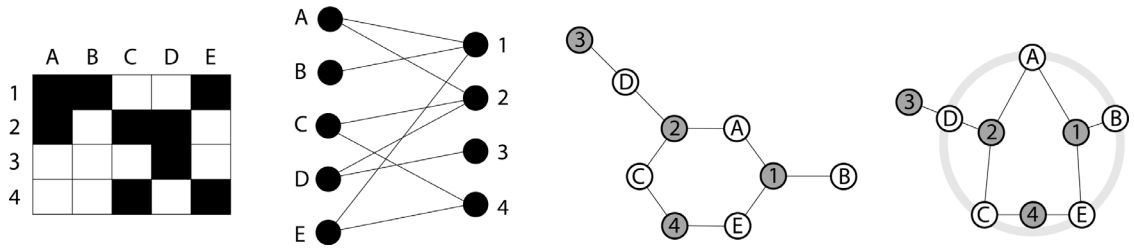

Fig. 1. Common visualizations of bipartite graphs: biadjacency matrix (left), linked lists (middle left), node-link diagram with spring layout and color-coded nodes (middle right), and anchored maps with one set of nodes fixed around a circle (right).

diagrams and matrix-based representations [12,13]. For bipartite or *k*-partite graphs, nodes of the *k* different sets can be differentiated by color in node-link diagrams (see Fig. 1 middle right or the graph view in *Jigsaw* [5]), or nodes of one set can be attracted to nodes of the other set, anchored at fixed locations [14,15] (Fig. 1 right). Bipartite graphs shown as biadjacency matrices have rows and column keys corresponding to the nodes of the two independent sets, with cells representing their connections (e.g., Fig. 1 left or Dormann et al. [16]). Another common way to visualize *k*-partite graphs are linked lists, where the nodes of the independent sets represent the list entries and edges between these lists show their connections [5,7,16–19] (illustrated in Fig. 1 middle left). While this visual encoding is easily understandable and allows for efficient scanning of the node labels, it does not scale well with the number of nodes. To deal with a number of nodes larger than can fit in the screen, these examples use scrolling [5], filtering according to node attributes [18], or focus+context representations [17]. However, when having thousands of nodes in one set of linked lists, these approaches lead to extensive interaction efforts, information loss, or visual clutter.

Common strategies to visualize large graphs – and large data sets in general – are *filtering* (i.e., removing items) and *aggregation* (i.e., grouping items) [13,20]. For instance, *GrouseFlocks* iteratively constructs a graph hierarchy through attributes of the underlying graph data [21]. The user can then interactively create cuts through the graph hierarchy and visualize the cut graph in a node-link diagram with aggregated meta-nodes. Similar nested meta-node circles were recently also used to create representative *Graph Thumbnails* of large graphs consisting of thousands of nodes and tens of thousands of edges [22]. Alternatively, aggregated meta-nodes can be visualized as matrices embedded within a node-link diagram [23,24] or as zoomable adjacency matrices [25]. These examples aggregate nodes either based on node attributes or based on topological properties, such as graph cliques. To the best of our knowledge, hierarchical aggregation of nodes has not been investigated for linked list visualizations or other visualizations of large bipartite graphs.

In bipartite graphs, *biclustering* [26] (or *co-clustering* [27]) finds groups of coherent items. Biclustering is mostly used in bioinformatics for studying gene expression data [2–4] and in document classification [27,28]. Essentially, biclustering simultaneously rearranges rows and columns of the biadjacency matrix to form clusters of certain similarity. Visualization of biclustered graphs use color-coded matrices [29–31], node-link diagrams with cluster enclosings [32], or matrices embedded into node-link diagrams [33–35]. Biclustering has also been used to bundle edges of bipartite graphs shown in linked lists [6,36]. Edge bundling of linked lists can improve the perception of the visualization and the quality of the analysis [37]. Since only edges are bundled, these lists still do not scale well with the number of nodes. In contrast, we present different strategies for aggregating the *nodes* in linked lists.

Recently, new visualization techniques for biclustered graphs with thousands of nodes have been introduced, such as *BiDots* [38] and VIBR [39]. However, these techniques use more complex visual encodings and require user interactions or legends to reveal any node labels. In contrast, our goal is to use a simple visualization that supports labeling of aggregated nodes so that it reveals the most essential information on the first glance.

Another recent approach to visualize very large unweighted bipartite graphs was presented by Pezzotti et al. [40]. They hierarchically cluster nodes of both sets independently based on their connectivity with the adjacent set, and place landmark vertices of HSNE clusters in two parallel axes connected by edges. These landmarks can be brushed to reveal lower hierarchy levels. With their C++ implementation, Pezzotti et al. visualize bipartite graphs with millions of nodes. In our work, we use different clustering strategies, taking into account the connectivity, but also the temporal development of edges, to create hierarchical aggregations.

There is little evidence so far as to which approaches facilitate an effective exploration of large bipartite graphs in practice. While BiDots [38] and ViBr [39] were evaluated through informal interviews with expert users, we contribute first results from a formal insight-based user study of *non-expert users* exploring large bipartite graphs with hundreds to thousands of nodes and edges.

All the visualization examples above are showing static graphs. In dynamic graphs, the structure of nodes and links change over time [41]. These changes can be shown through animating a node-link diagram or by mapping the temporal information to a spatial representation [41], which can be juxtaposed, superimposed, or nested within the graph [42]. For instance, Burch et al. [43] nest time-varying edge weights as time series display in cells of an adjacency matrix. Yi et al. [44] combine this approach with hierarchical aggregation by attributes (e.g., geographic regions) or connectivity, where cells show the temporal distribution of a selected attribute. To the best of our knowledge, no visualization technique for time-dependent bipartite graphs has been proposed so far.

Time-varying edge weights can also be used for clustering a large data set. There are numerous methods how to cluster time series data [45,46]. In the field of visualization, time series clustering has been used to reveal patterns of employee presence over a year and the time of the day [47], DNA copy numbers [48], or sensor networks [49]. However, these clusters are not embedded in a graph structure and are visualized as superimposed line charts [47], stacked color gradients [48], or a 2D projection of items based on their time series similarities [49]. In our system, we cluster nodes of both sets based on their time series similarities, but we maintain the graph structure by visualizing the connections between these clusters.

3. Visualization and interaction design

Our goal is to provide a broad audience access to a large, dynamic graph data set through interactive visualization. The main requirements for the visualization and interaction design of Dynamic BiCFlows therefore are:

1. the visualization should scale up to thousands of nodes and edges,
2. the visualization and interaction design should be easily understandable for a lay audience,
3. it should provide some initial information on the first glance, and

4. it should support in-depth exploration of the data, such as identifying clusters of similar elements of varying size and retrieving connections of a selected element at a particular time.

To achieve Requirement 1, we use a combination of hierarchical aggregation and filtering. We present two hierarchical clustering approaches for time-dependent bipartite graphs (Section 3.1). Per cluster, node filtering is performed based on ranking of accumulated edge weights (Section 3.2).

To fulfill Requirement 2, we opted for a bipartite graph visualization using linked lists (Section 3.2). Lists are ubiquitous and therefore presumably easy to understand for visualization novices. In addition, they can feature sufficiently large text labels so that users can gain some initial understanding on the first glance. In contrast, the similarly popular matrix view of bipartite graphs requires very short or 90° rotated column labels.

Linked lists can easily get visually cluttered when showing a large number of nodes and edges. To provide an overview of a large bipartite graph (Requirement 3), we therefore present a new variation of linked lists visualizations: we aggregate nodes based on the hierarchical clusters and visualize the cluster-wise temporal evolution as nested time series (see Fig. 2 and Section 3.2). Through the hierarchical node aggregation, the visualization conceptually scales to an infinite number of nodes.

To reveal full details (Requirement 4), we present exploration techniques, such as drill-down, filtering, and details-on-demand, in Section 3.3. As successively drilling down into the data may lead to a loss of overview, we extend the list view by interactive context bars to keep the user oriented.

3.1. Hierarchical aggregation

Hierarchical aggregation iteratively partitions the data into groups of similar items so that, initially, a small set of group items can be visualized instead of thousands of individual items [20]. By subsequently selecting grouped items, the users can interactively drill down from the initial overview to a small subset of the data with similar characteristics.

In general, groups can be derived from existing node hierarchies, from datacube aggregations if nodes are associated with multiple attributes [50], or clustering. We are interested in time-varying bipartite graphs, where nodes may appear and disappear over time, and where edge weights between pairs of nodes may increase or decrease. We assume that there are no further node or edge attributes or node hierarchies. We therefore use clustering to create a hierarchical grouping of the nodes. We use two clustering approaches that group items based on different characteristics:

1. *Biclustering*, which groups items to maximize the graph modularity so that clusters contain densely connected sets of nodes (Section 3.1.1). Using this clustering approach, users can explore tightly connected groups, such as groups of authors sharing a lot of common key words in a publication database.
2. *Time series clustering*, which groups nodes based on their time series correlation so that nodes with a similar temporal characteristic remain in the same cluster (Section 3.1.2). This clustering approach facilitates exploration based on common temporal trends, such as groups of media organizations showing similar seasonal patterns for receiving advertisements.

As input data, we consider tabular data, where each row i contains two entities – one from set V_1 and one from set V_2 – and an associated time. In addition, each row may be associated with a quantitative weight ω_i (Fig. 3 left). For instance, in our Media Transparency Database use case (Section 4.1), each row consists of the name of a public authority (V_1), the name of a media corporation (V_2), the year and quarter for which the expenses were reported (t), and the advertisement expenses in Euro (ω). The output of our clustering methods is a $k \times k$ biadjacency matrix C (Fig. 3 right), where the rows correspond to the k clusters of node set V_1, the columns to the k clusters of node set V_2, and the cells to the number of aggregated edges (for unweighted bipartite graphs) or the sum of aggregated edge weights (for weighted bipartite graphs) between the respective clusters. We recommend to pick $k < 10$, depending on the available vertical screen space, to minimize visual clutter due to many edge crossings in the linked lists.

By subsequently selecting sub-clusters, the users can interactively drill down from the initial overview to a subset of the data. The system then clusters the nodes in the selected cluster and subsequently visualizes only those items. Drill-down is possible until the cluster cannot be further subdivided into smaller clusters.

3.1.1. Biclustering

A bipartite graph can be viewed as a weighted biadjacency matrix, where rows represent nodes of set V_1, and columns represent nodes of the other set V_2. Each matrix cell contains the corresponding edge weight between two nodes from V_1 and V_2 (Fig. 4). Using such a data representation, the temporal component gets lost in the resulting visualization, as all weights between two nodes across all time steps are aggregated into a single cell (see Fig. 4, cell B2). Temporal exploration, however, can be achieved by filtering the displayed time steps (see Section 3.3.2).

Biclustering rearranges the rows and columns of the matrix to create coherent blocks. Biclustering is an NP-hard problem [51], but many algorithms that optimize search heuristics have been developed. In our system, we use an algorithm that tries to maximize the modularity of the bipartite graph for a predefined number of clusters [52]. Modularity is a common graph clustering quality measure that quantifies the trade-off between the edge density *within* clusters and *between* clusters [53] to obtain clusters of dense sub-graphs with minimal edges between these clusters. In contrast to other biclustering algorithms, this approach can also handle weighted biadjacency matrices. By choosing a clustering algorithm that maximizes the modularity, edge densities of diagonal edges in the linked lists are minimized, such as depicted in Fig. 5b as in contrast to Fig. 5a.

Biclustering algorithms assume a specific structure of the underlying matrix. Commonly used structures are the *block diagonal structure*, where each row and column is assigned to exactly one cluster, and the *checkerboard structure*, where each row and column is assigned to multiple clusters, so that each cell is assigned to exactly one cluster. For our system, we use a block diagonal structure so that each node is associated with only a single cluster. Fig. 5b illustrates block diagonal biclusters as dense sub-matrices along the biadjacency matrix diagonal and large horizontal edges in the linked lists.

When the user drills down from the initial overview to a subset of the data, the system clusters the sub-matrix of the selected cluster (Fig. 6). In the linked lists, we visualize the nodes and edges of this sub-matrix, as well as nodes from other clusters connected to at least one node from the selected cluster. The user can drill down until the matrix to be clustered has only a single row or column, or if the bipartite graph is too dense to be clustered further.

3.1.2. Time series clustering

To group nodes with similar temporal developments, we first accumulate the edge weights into uniform, discrete time bins (e.g., years or quarters) so that each node is associated with a time series (Fig. 7). For each set, we then compute the pairwise correlation distances between all nodes and apply agglomerative clustering to the resulting matrices to retrieve up to $2 \times k$ clusters for both node sets in total. As a result, each cluster contains nodes of one set with similar time series (i.e., time series with minimal correlation distances). This means, the variance of the aggregated time series of all nodes in all clusters is as small as possible. This allows us to visualize a representative aggregated

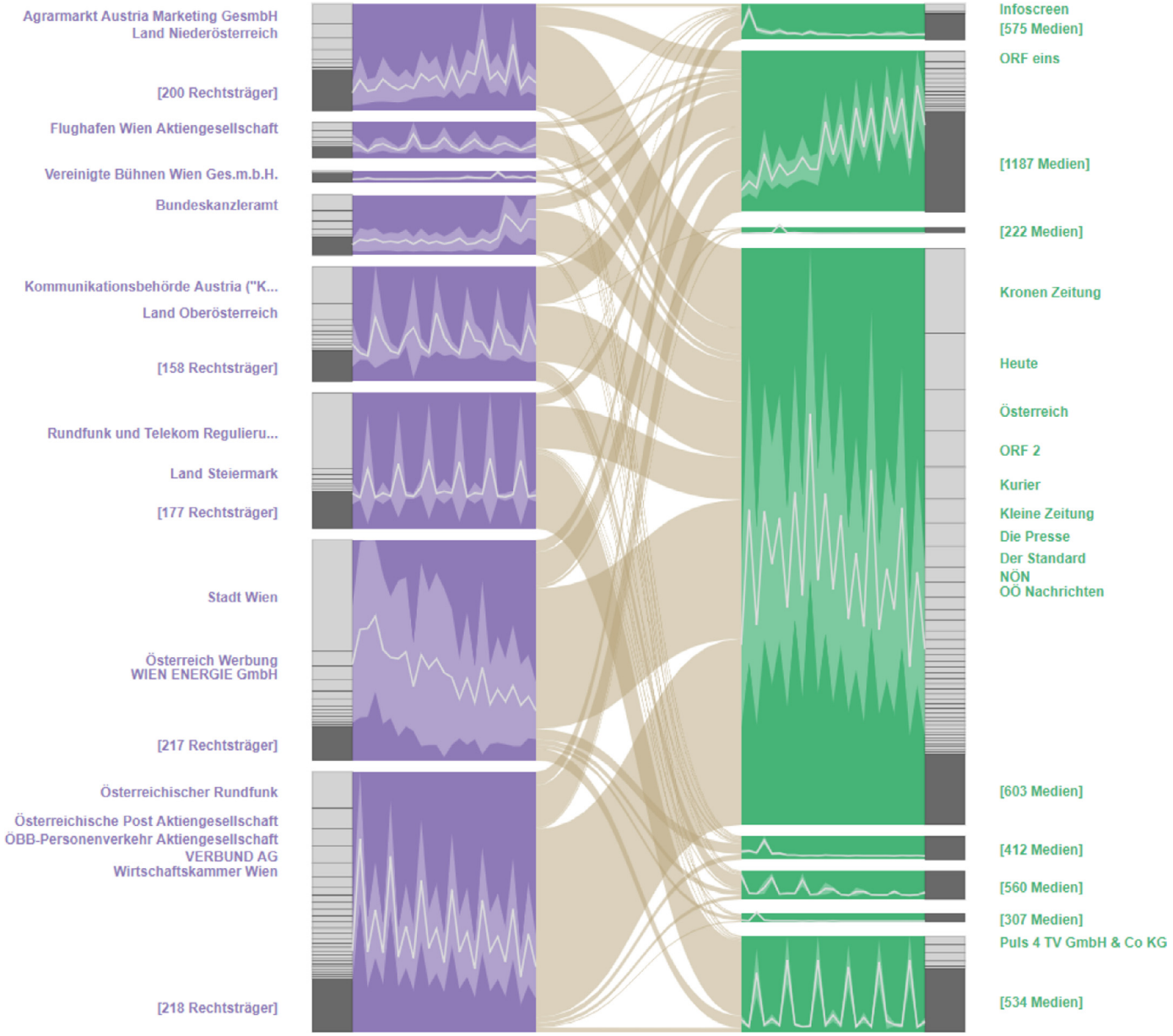

Fig. 2. Dynamic BicFlows with nested time series visualization per cluster per set. Each cluster contains legal entities (left, purple) or media companies (right, green) following similar temporal developments of advertisement expenses or incomes, respectively. The gray stacked bars show individual cluster items sorted by accumulated edge weights, where the most important items receive permanent labels. The brown edges indicate the money flow from clusters of legal entities to clusters of media companies (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

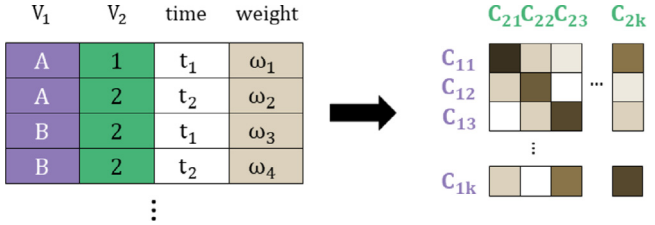

Fig. 3. Input data (left) in tabular format and output data (right) with k clusters for both sets in a biadjacency matrix (luminance indicates the accumulative edge weight).

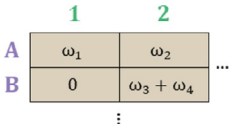

Fig. 4. The biadjacency matrix corresponding to the tabular data from Fig. 3 left.

time series for each cluster (see Fig. 2).

Since the two sets of nodes are clustered independently, the modularity of the resulting clustered bipartite graph can be rather low. This can lead to high edge densities between clusters, as illustrated in Figs. 5a and 8 a. However, the visual clutter can be reduced by maximizing the edge density of opposite clusters in the linked lists so that high density edges are mostly horizontal. This can be achieved by a permutation of the columns of the biadjacency matrix C to maximize the trace of the matrix [54]. The selected permutation defines the order of the clusters of V_2 in the second column of the linked list visualization (Fig. 8b).

However, even with reordered cluster columns, graph modularity

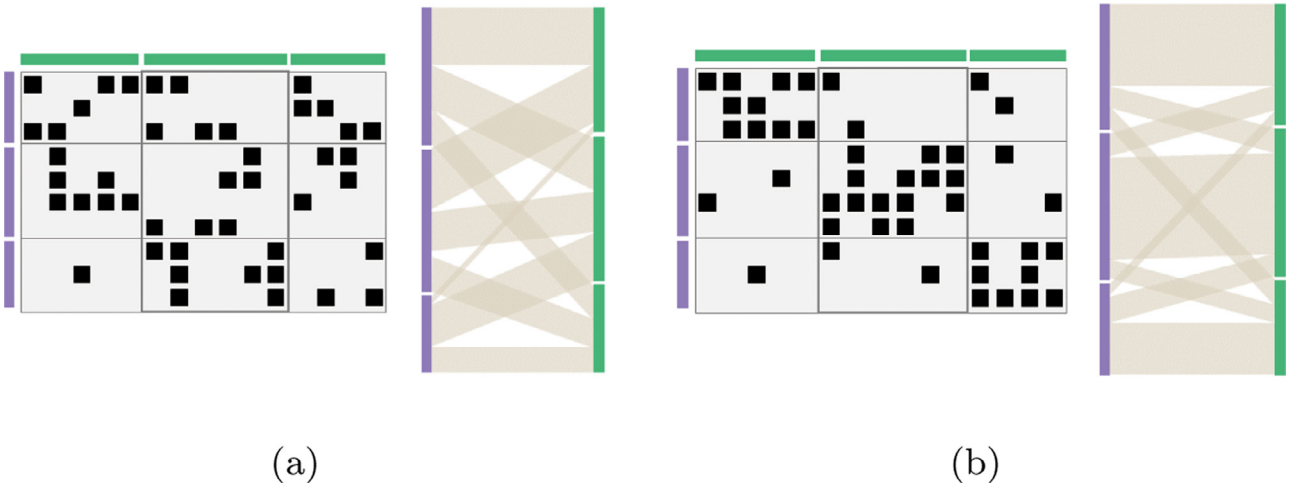

Fig. 5. Sketch of a biadjacency matrix and its associated linked lists representation for an arbitrary grouping (left) and with a biclustering based on a block diagonal structure (right).

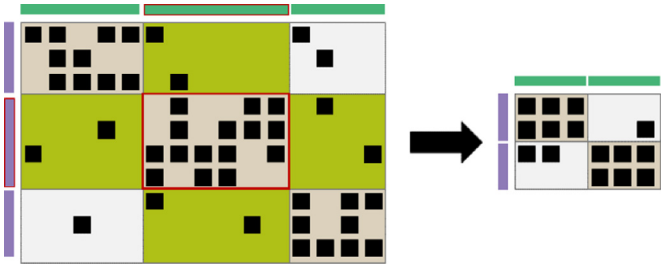

Fig. 6. A biadjacency matrix with $k = 3$ biclusters shown in brown (left) and one bicluster selected (red). The selected sub-matrix is further biclustered (right). The four lime-green sub-matrices (left) contain edges between nodes of the selected bicluster and nodes of other biclusters (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

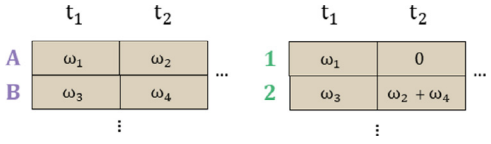

Fig. 7. The two time series matrices (rows are nodes, columns are time bins) for the two node sets corresponding to the tabular data from Fig. 3 left.

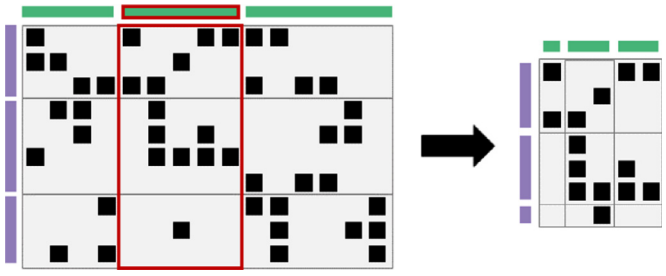

Fig. 9. A biadjacency matrix with $k = 3$ clusters for both sets (left) and one cluster of V_2 selected (red). Unconnected nodes of V_1 are removed, and the selected columns of V_2 are further clustered (right) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

can be quite low compared to biclustering (cf., Figs. 5b and 8 b). To avoid information loss when drilling down, we therefore perform an asymmetric sub-clustering: we distinguish whether a user selected a cluster of V_1 or of V_2 for further sub-clustering. We then select the corresponding rows (if selecting a cluster node of V_1) or columns (if selecting a cluster node of V_2) from the reordered biadjacency matrix. For the other set, we maintain the previous clustering and only filter the nodes that do not link to nodes of the selected node cluster (Fig. 9). For the selected set, we perform a time series clustering of the nodes in the selected cluster.

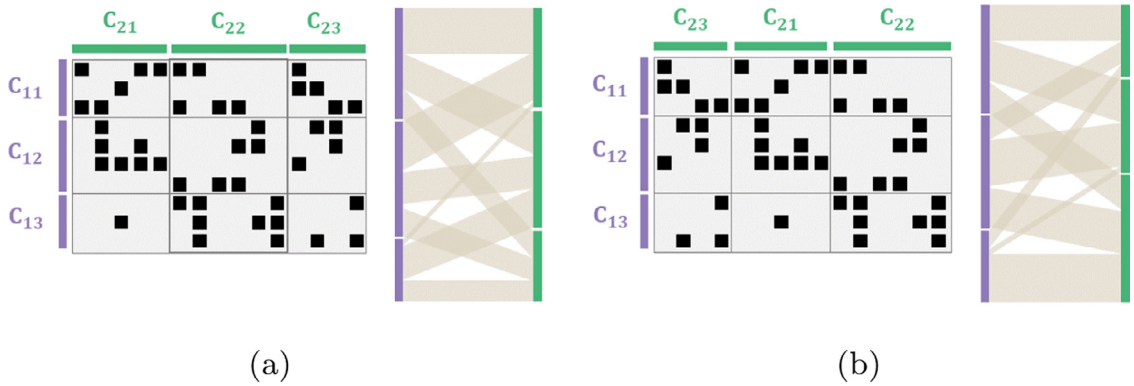

Fig. 8. Sketch of a biadjacency matrix and its associated linked lists of three clusters per set: unordered clusters from Fig. 5a (left) and reordered cluster columns to minimize high density diagonal edges in the linked lists (right).

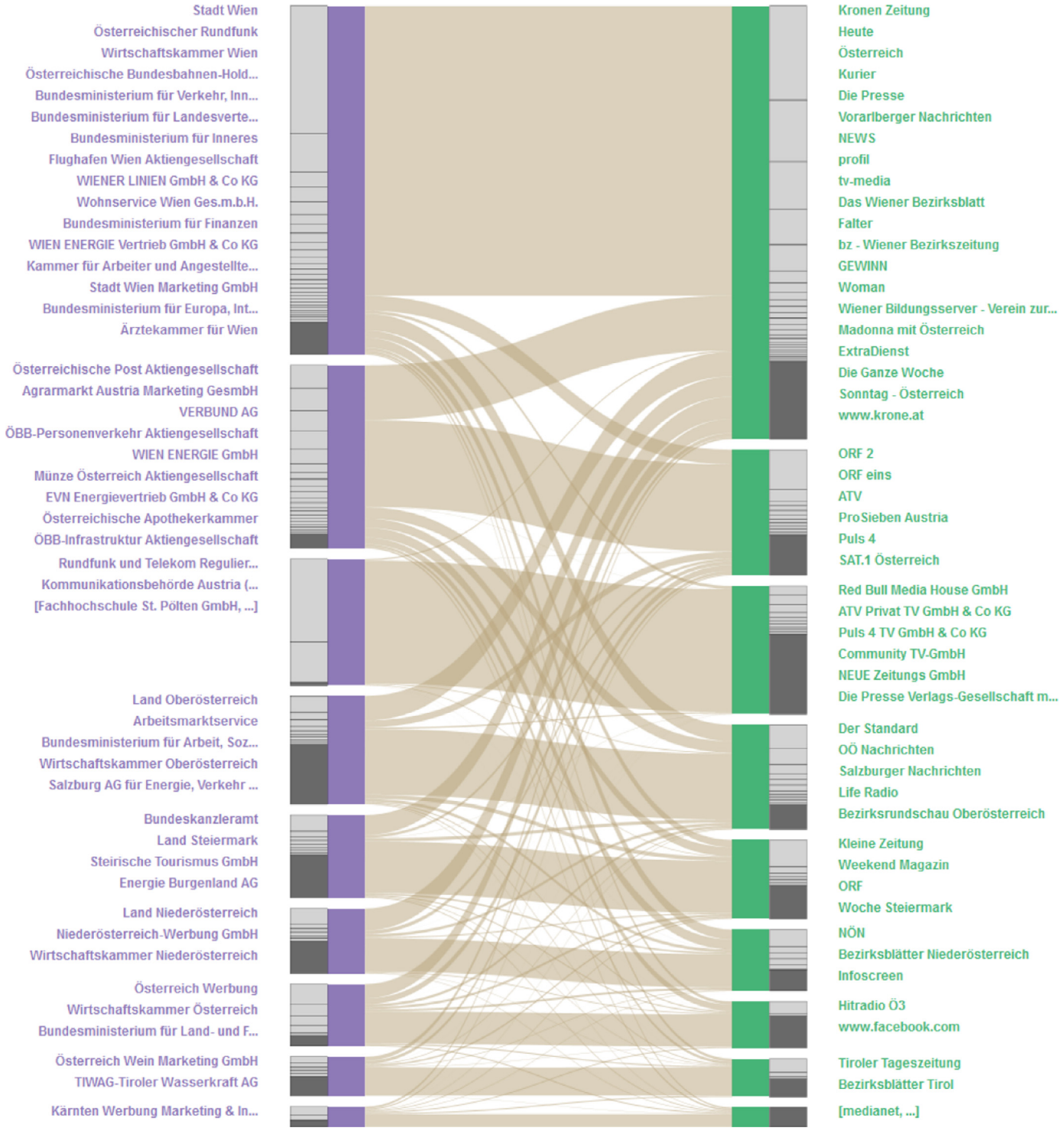

Fig. 10. BiCFlows showing individual nodes (gray), aggregated filtered nodes (dark gray), cluster bars of nodes (purple and green), as well as their connections (brown) (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

3.2. Visual encoding

To visualize the hierarchically aggregated bipartite graph, we use two parallel vertical lists of nodes, where – similarly as for Sankey diagrams [55] and parallel sets [56] – the thickness of an edge connecting two nodes is defined by its edge weight, and the rank of each node is defined by its accumulated edge weights $\Sigma\omega_i$ (Fig. 10). We apply the concept of hierarchical aggregation to such linked lists by grouping nodes and edges into their respective clusters. Clusters are shown as aggregate *cluster bars*, where V_1 is shown in purple and V_2 in green. For each cluster bar, we visualize the contained nodes as stacked bars. Within each cluster, nodes are ranked according to their accumulated edge weights so that the most important nodes are shown at the top.

Since we initially display a large number of nodes per cluster, we filter nodes with small edge weights. Given the sum of all edge weights in the entire graph $\Sigma\omega$, the smallest displayable unit for a node h, and the total height H of the visualization, we only display nodes, which fulfill the following criterion:

$$\sum \omega_i \geq \frac{h \sum \omega}{H}. \quad (1)$$

We aggregate all nodes that would be encoded smaller than the smallest acceptable height h in the list into a dark gray bar on the bottom of the stacked bars. For our use cases, we set h to two pixels and adapt the height of the visualization H dynamically to the display size.

We implemented two labeling methods: The first method places node labels whenever there is enough space next to the stacked bars. We use a 12 pixel font-size, so we place labels whenever a node fulfills the criterion in Eq. (1) for $h = 12$ (Fig. 2). Using this method, the number of node labels can be quite small, however. The other method tries to place as many labels as possible for each cluster by stacking node labels next to a cluster bar (Fig. 10).

To reveal the temporal development, we show an aggregated time series visualization for each cluster as nested visualization. There are several options how to visualize aggregated time series, such as using stacked color-coded graphs [57], "temporal box plots" [58], or confidence bands, which we chose due to their simplicity and scalability. For each discrete time bin, we compute the mean edge weight of all nodes, as well as the 95% confidence interval, where the mean development is shown as a line, and the confidence band as half-transparent overlay (Fig. 2). Like *sparklines* [59], these nested time series visualizations are intended to convey a general trend and therefore do not have axes labels. As they are nested within the cluster bars, the time series are also scaled relatively to the overall weight of their respective cluster. We only visualize the nested time series when aggregating nodes through time series clustering, where the minimized correlation distances between time series within a cluster ensure that an aggregated time series visualization is representative for all nodes in the cluster.

3.3. Interactive exploration

Our system supports three interactive exploration mechanisms: *drill-down, filter*, and *details-on-demand*. With these exploration techniques, we follow the well-known visual information seeking mantra: *"overview first, zoom and filter, then details-on-demand"* [60].

3.3.1. Drill-down

Drill-down is the essential interactive mechanism to reveal filtered nodes from selected clusters by iteratively sub-clustering the selected group. Users perform a drill-down by double-clicking a cluster bar. Depending on the clustering method (see Section 3.1), we employ two different drill-down methods: As biclustering creates coherent groups of both node sets (see Section 3.1.1), drill-down operates symmetrically in this case. When selecting a bicluster, all nodes that do not have connections to the nodes in the selected cluster are discarded (gray sub-matrices in Fig. 6 left). Nodes of other clusters connected to nodes in the selected cluster (lime-green sub-matrices in Fig. 6) are aggregated into one group (lime-green group at the bottom in Fig. 11).

Time series clustering leads to less coherent groups so that the lime-green bars at the bottom would grow considerably as the user drills down. We therefore provide asymmetrical drill-down in this case: depending on which list the user selects a cluster, we keep the entities of the selected set within the selected cluster, as well as all connected entities from the other set (Fig. 9).

To help users maintain orientation and keep an overview, we provide context bars on the side, where each bar shows the selected cluster among the grayed out non-selected clusters. These bars are also used to navigate back to a higher hierarchy level. In Fig. 11, the user selected a large central cluster in the initial overview (indicated by the purple and green bars on the outermost context bars).

3.3.2. Filter

Using multiple coordinated views, we also allow users to filter according to time intervals or other categorical attributes, if available in the data set, using linked bar charts (Fig. 11 left). To support interactive exploration with a fast response and to keep the user oriented, we initially perform clustering on the complete, unfiltered data set and keep this clustering as the user filters nodes by interacting with the linked bar charts. After applying the filter, we only recompute the edge weights and reorder the nodes within the clusters, if necessary.

3.3.3. Details-on-demand

Users can request details-on-demand by either hovering nodes and node labels, respectively, or cluster bars. In the first case, all connections of one individual node are highlighted in red. In the second case, only connections of nodes in the hovered cluster are visualized (Fig. 12). In both cases, a tooltip reveals detail about the selected node and cluster, respectively, such as the aggregated edge weight, its temporal development, and the number of nodes in the selected cluster.

Node highlights can also be triggered from linked text lists (see Fig. 11 on the bottom left), where all entities of both sets are listed and ranked according to their accumulative edge weight. These lists can also be searched for specific entities.

3.4. Implementation

Dynamic BiCFlows is implemented using a client-server infrastructure to separate the computationally expensive clustering from the user interface on the client side. The server is implemented with *Python* and *Numpy* for efficient processing of large data structures. For biclustering, we use the Python implementation of *CoClust* [52]. For time series clustering, we use *SciPy* for the computation of the correlation-based distance matrix, for the agglomerative clustering of the distance matrix, as well as for the linear sum assignment optimization of the inverted biadjacency matrix to minimize the weight of diagonal edges.

As our system is intended for visual exploration by lay users, we host individual data sets as separate web services. For each data set, we use CoClust to determine the optimal number of biclusters k in a pre-computation step. In this step, CoClust computes multiple clusters and finds the resulting biadjacency matrix with the maximum modularity. For small browser windows, we decrease the number of clusters to avoid visual clutter. The actual clustering is performed online, and sub-clustering is invoked whenever a user drills down.

We performed performance tests using the Media Transparency Database (see Section 4.1) on a consumer hardware (Intel i7-8750H CPU with 2.20GHz and 16 GB RAM). We measured the time for constructing the biadjacency matrix from the original CSV file, as well as the time to initially bicluster the entire dataset (Fig. 13). In total, the server requires around 700 ms to read, convert, and cluster the latest version of the dataset (2018, quarter 4: 71,780 entries, consisting of 1323 legal entities and 4538 media companies), and these computations scale linearly with the number of nodes in both sets ($R^2 = .96$). Time series clustering of both sets takes approximately the same amount of time. The initial clustering of the entire data set is therefore only performed once when loading the page. Whenever the user selects a cluster, the clustering results of the higher hierarchy levels are locally stored, so that the user can quickly navigate back to previous views.

The client was implemented using *D3.js* [61] based on an existing bipartite layout [62]. *Crossfilter* [63] was employed to quickly filter the data set on the client side for the linked time slider, category bar chart, and the lists of nodes.

4. Use cases

We will showcase the usefulness and discuss potential limitations of Dynamic BiCFlows using two data sets:

1. the so-called *Media Transparency Database* [9], where all public authorities of Austria have to report their advertisement expenses to media companies above 5,000 Euros beginning from 2012

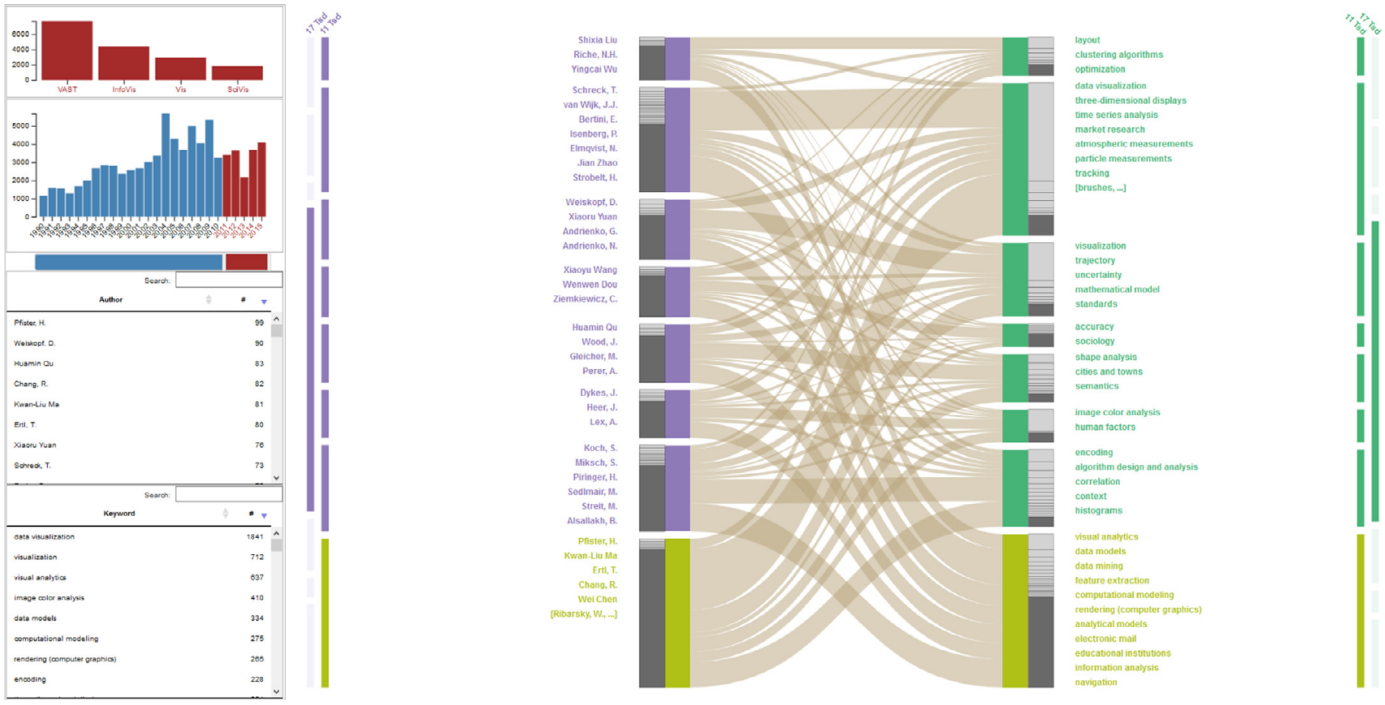

Fig. 11. Selected bicluster divided into seven sub-clusters, with connections to other biclusters (lime-green) and the context bars on the side providing an overview of the entire data set for a selected time period (2011-2015). Linked bar charts (top left) allow for filtering according to categorical attributes (here: conferences) and time. Ranked text lists of all entities (bottom left) can be used to search for hidden nodes (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

(Section 4.1), and

2. the *IEEE Visualization Publication* collection [8], where meta-data of all major IEEE visualization papers since 1990 are collected. Using Dynamic BiCFlows, we visualize the author-keyword relations for selected time intervals (Section 4.2).

Both data sets have thousands of nodes and fulfill the properties of a time-dependent bipartite graph. The visualizations can be accessed online [64].

4.1. Media transparency database

The Austria Media Transparency Database [9] is of great interest to journalists to reveal relations between public and media organizations, by media organizations themselves to investigate their competitors, and to the general public to find out how their tax money is spent. The database is updated quarterly, and journalists regularly parse the database for new interesting money flows. In particular, they are interested to find out if certain ministries advertise in similar media and which ministries spend a high amount of money for advertisement. However, finding this information is tedious, since names of ministries change across legislation periods, and some big media organizations comprise dozens of sub-companies, which all show up as separate entities in the database.

By 2018, the Media Transparency Database contained 1300 legal entities, reporting advertising expenses to over 4500 media organizations. The reported expenses are not evenly distributed, with very few very high values (e.g., around 22 million Euros aggregated advertisement expenses issued from the government of the city of Vienna to the daily newspaper *Kronen Zeitung*), and most of the expenses being around 5,000 Euros. The highest modularity (0.5) was found for nine biclusters (see Fig. 10).

Due to the large public interest, there are already a few online visualizations of the Media Transparency Database available, such as a dashboard visualization by Rind et al. [65], a web service by Salhofer et al. [66], and an interactive exploration interface for data journalists supporting filtering, sorting, and tagging [19]. These existing visualizations rely solely on filtering of the data and therefore only visualize a very small fraction of the existing entities. With these visualizations, users can get information about the most relevant legal entities and media organizations. However, smaller transactions, for instance because advertising expenses are spread across multiple smaller media organizations, cannot be revealed without specifically searching for them.

Like these previous approaches, Dynamic BiCFlows reveals important legal entities and media organizations on the first glance. The two top-most labels in Fig. 10 show the legal entity (*Stadt Wien*, i.e., the government of the city of Vienna) and media organization (*Kronen Zeitung*, the most popular newspaper in Austria) spending and receiving the highest accumulated sums, respectively. These two nodes are grouped into the same bicluster with other popular Austrian newspapers, such as *Heute* or *Kurier*, and other legal entities spending high amounts for advertising in these daily newspapers. The second-ranked legal entity (*Rundfunk und Telekom Regulierungs-GmbH*, the Austrian Regulatory Authority for Broadcasting and Telecommunications) is contained in a different bicluster, which is ranked third in Fig. 10. Selecting the cluster reveals that this legal entity mainly sponsors small radio and TV stations, where most of them do not receive any advertisement money from other legal entities. If only the ten top-ranked media organizations were shown, not a single media organization receiving money from this authority would be visualized.

In contrast to the previous visualizations of the Media Transparency Database, BiCFlows supports untargeted, casual exploration of the data. When drilling into the data using biclustering, a frequently occurring grouping reveals geographic proximity. Often, the groupings contain smaller legal entities and media organizations located in the same regions by just moving one hierarchy level down. This is not surprising, since smaller entities tend to advertise in smaller and more local media. Other clusters are related topic-wise. For instance, drilling down three hierarchy levels reveals a cluster of many media organizations related

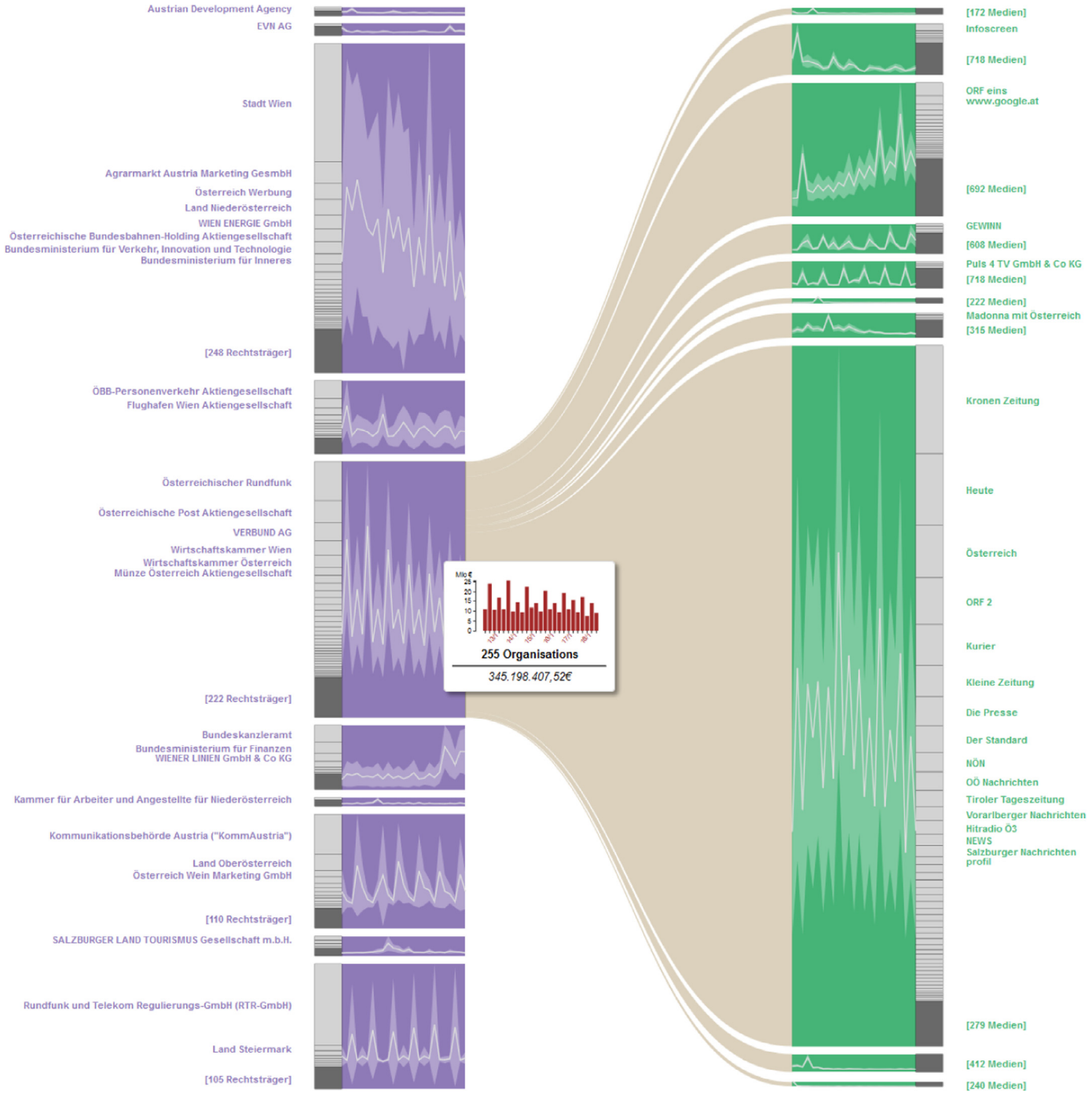

Fig. 12. Hovering a cluster reveals only those nodes in the adjacent list, which have connections to the hovered cluster.

to air travel, such as *Airline Business* or *Air Transport World* associated with a single legal entity – the *Vienna International Airport*.

As one particular interest of journalists is the development of advertisement expenses and press subsidies over legislation periods, time series clustering can help to discover systematic changes in advertisement behavior. By drilling into the second media cluster in Fig. 2 (right), the sub-clustering reveals those media corporations with an apparent upwards trend in the last few years. This cluster contains primarily online media, such as Google, Facebook, web pages of large news, but also regional radio stations. Selecting the forth cluster in the left list of Fig. 2 yields legal entities with increasing expenses starting from the end of 2017, when a new government was installed in Austria. The largest sub-cluster contains exclusively ministries. However, most of these ministries were renamed, and therefore also show up with slightly different names in sub-clusters with a decreasing temporal trend (see Fig. 14).

Other distinct temporal characteristics are reoccurring seasonal peaks, such as shown in the bottom right cluster in Fig. 2. These seasonal peaks are mostly associated with press subsidies to smaller TV or radio stations, which are typically paid at the beginning of the year.

4.2. IEEE visualization publications

Co-authorship networks are a common use case for graph visualization, such as by Henry et al. [23]. Using the IEEE visualization publication collection by Isenberg et al. [8], we pursue a different

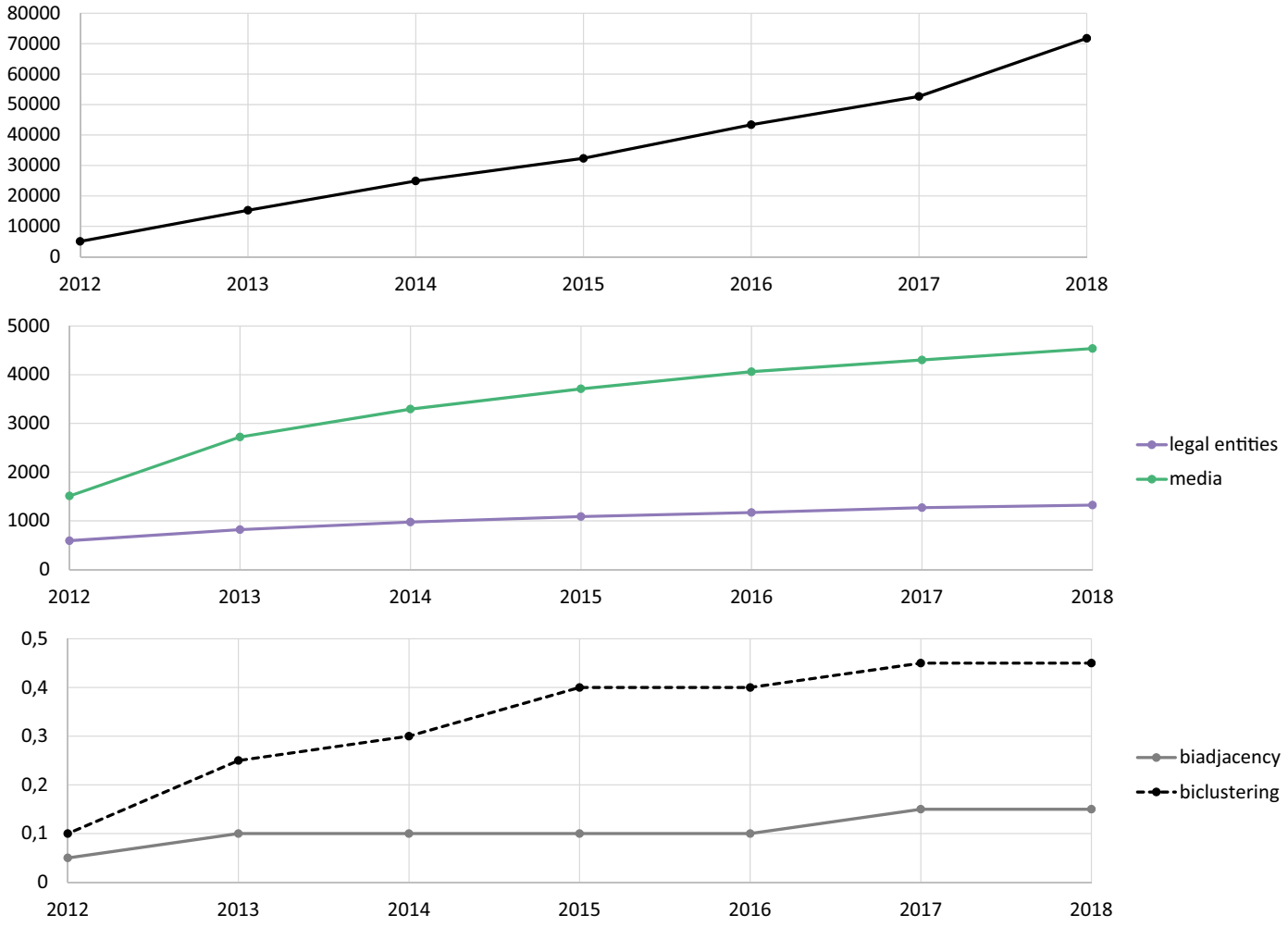

Fig. 13. Performance of hierarchical aggregation of the Media Transparency Dataset from the third quarter of 2012 to the fourth quarter of the given year: The top chart shows the number of rows in the CSV files, the second chart shows the number of resulting nodes in the two sets, and the bottom chart shows the computation time in seconds for contsructing the biadjacency matrix (gray) and performing the initial biclustering (dashed).

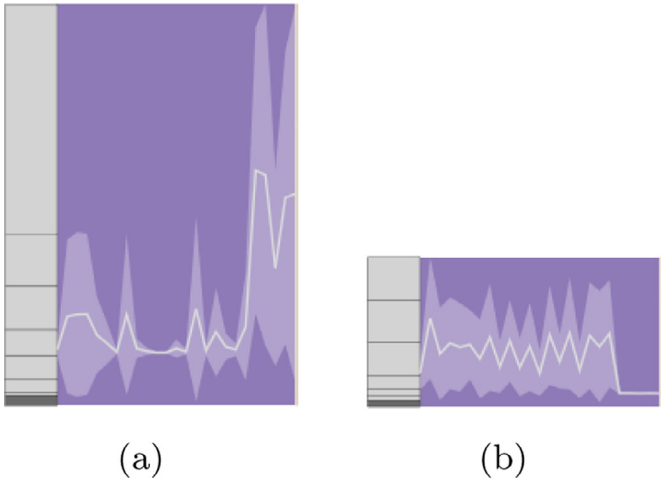

Fig. 14. The sub-cluster of legal entities showing the most distinct increase in the new legislation period in Austria includes mostly ministries, such as the finance, social, defense, and education ministry (a). Some ministries with slightly different names ceased to exist (and advertise) at the same time, such as the ministry for agriculture or the ministry for defense and sports (b).

approach to investigate commonalities between authors. We retrieved 4976 authors from the data set, as well as 2120 IEEE key terms these authors used to classify their papers. Biclustering is employed to reveal groups of authors that tend to use similar key terms – or, conversely, groups of key terms that tend to be used by the same authors. Temporal filtering is used to explore the developmment of those groups over time. Over the entire time period, the modularity of this data set is rather low (0.31 for seven biclusters). This means that the biclustered list will lead to more edge crossings than the Media Transparency Database data set.

Fig. 15 shows the visualization of the seven biclusters in ten years intervals. Notice how the fourth cluster with the top keyword “data visualization” increased in the last decade compared to the previous one. In contrast, the bottom cluster with the top keywords “computer graphics” and “rendering” and the second cluster with top key words “computational modeling” and “data mining” decreased considerably.

Selecting these clusters can yield interesting sub-topics. For instance, selecting the fifth bicluster in Fig. 15 b reveals a sub-cluster with application-specific key terms (Fig. 16 top). Sub-clustering this cluster again reveals a cluster of key terms from the automotive industry with its associated main authors (Fig. 16 bottom).

This example also explains why the clusters have a rather low modularity: While K. Matkovic is the most common co-author of H. Hauser according to DBLP [67], his publication keywords are much broader than suggested by this clustering. Highlighting all key terms used by H. Hauser by hovering his name, we discover that, in fact, his

Fig. 15. Seven biclusters of authors and IEEE key terms of IEEE visualization publications in 10-year blocks (data from Isenberg et al. [8]).

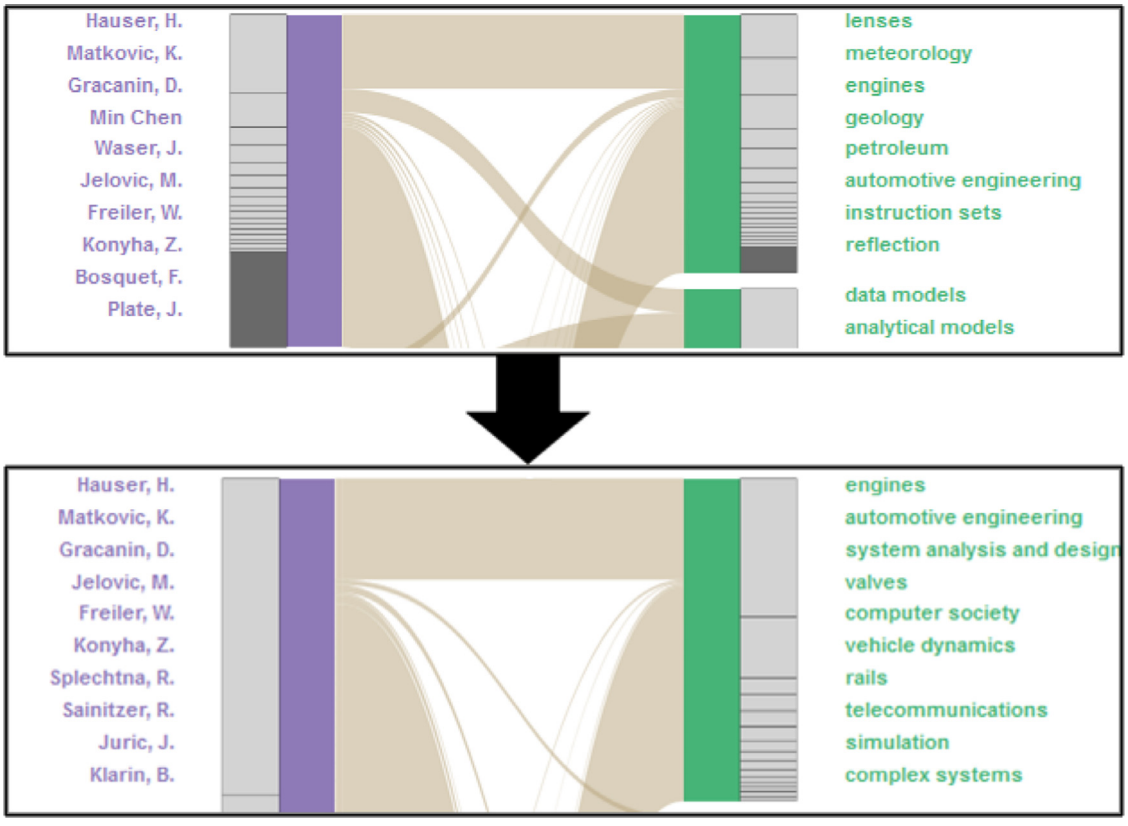

Fig. 16. Two steps of sub-clustering on the fifth bicluster (H. Hauser – data models, analytical models) in Fig. 15 b.

most commonly used key term in the IEEE Visualization Publication data set is *data visualization* (used 19 times), followed by *computational modeling* (used 12 times). The most commonly used key term in the bottom cluster of Fig. 16 (*"engines"*) was used only four times by H. Hauser. This means that biclustering is able to reveal meaningful clusters of key terms in this example. However, the biclusters are not necessarily representative for individual authors.

5. User study

The focus of our user study was to investigate the positive and negative effects of hierarchical aggregation on interactive, casual exploration of large bipartite graphs. We used the Media Transparency Database introduced in Section 4.1 for our evaluation and used biclustering for aggregating the data. We recruited 12 users (four females, eight males, aged 25 to 56) with different backgrounds, including one computer scientist, and all experienced computer and internet users. One user had prior knowledge of the Media Transparency Database, two had heard of it before, and nine did not know it at all. However, all users were roughly familiar with the political and media landscape in Austria.

As there is no measurable "ground truth" in this data set, we performed an insight-based evaluation [68]. Insight has been defined as *"individual observation about the data by the participant"* [69]. To reveal whether users made observations, insight-based evaluations use an open-ended think-aloud protocol, which afterwards is coded and quantified for formal evaluation [68]. Users are encouraged to explore the data as long as they think they can find something new.

As a baseline condition, we used a simplified version of BiCFlows, which reduces the number of displayed items solely by filtering, but does not perform any aggregation (see Fig. 17 b). Here, we refer to this baseline as *Cut-Off*. All nodes that are too small to be labeled are aggregated into "others" nodes (the bottom nodes in Fig. 17 b). The Cut-Off visualization allows for highlighting of selected nodes like with BiCFlows. Legal entities and media organizations that are filtered can be selected from the linked text list to visualize all associated advertisement expenses.

Both visualizations provided the ranked text list of nodes as additional browsing interface, but filtering through the bar charts was not supported. We also did not show the nested time series to keep the two interfaces as similar as possible. This means that the temporal aspect was not considered in this evaluation, and we rather showed a single aggregate of all entries in the Media Transparency Database up to 2017.

5.1. Hypotheses

Our main goal has been to investigate the benefits and limitations of the hierarchical aggregation approach of BiCFlows compared to a simple filtering approach, which is the common method to visualize the Media Transparency Database [19,65,66]. Our assumption has been that iteratively drilling down into the aggregated data would encourage lay users to casually explore the visualized data in more detail and, as a consequence, gain more knowledge. On the other hand, we also assumed that BiCFlows would be perceived as more complex and harder to use than the Cut-Off baseline visualization. We therefore formulated two main hypotheses:

H1: *With BiCFlows, users will gain more insights than with Cut-Off.*

In particular, we expected that users would discover more legal entities and media organizations, as well as transactions between them (**H1.1**), that they would mention more entities with small accumulated advertisement sums (**H1.2**), establish more connections between entities or reason about commonalities (**H1.3**), discover more unknown entities or unexpected information (**H1.4**), and spend more time exploring the data (**H1.5**).

H2: *BiCFlows will be perceived as more complex than Cut-Off.*

As BiCFlows conveys more information on the first glance and requires more interactivity for in-depth exploration, we expect that untrained users find BiCFlows more demanding to use.

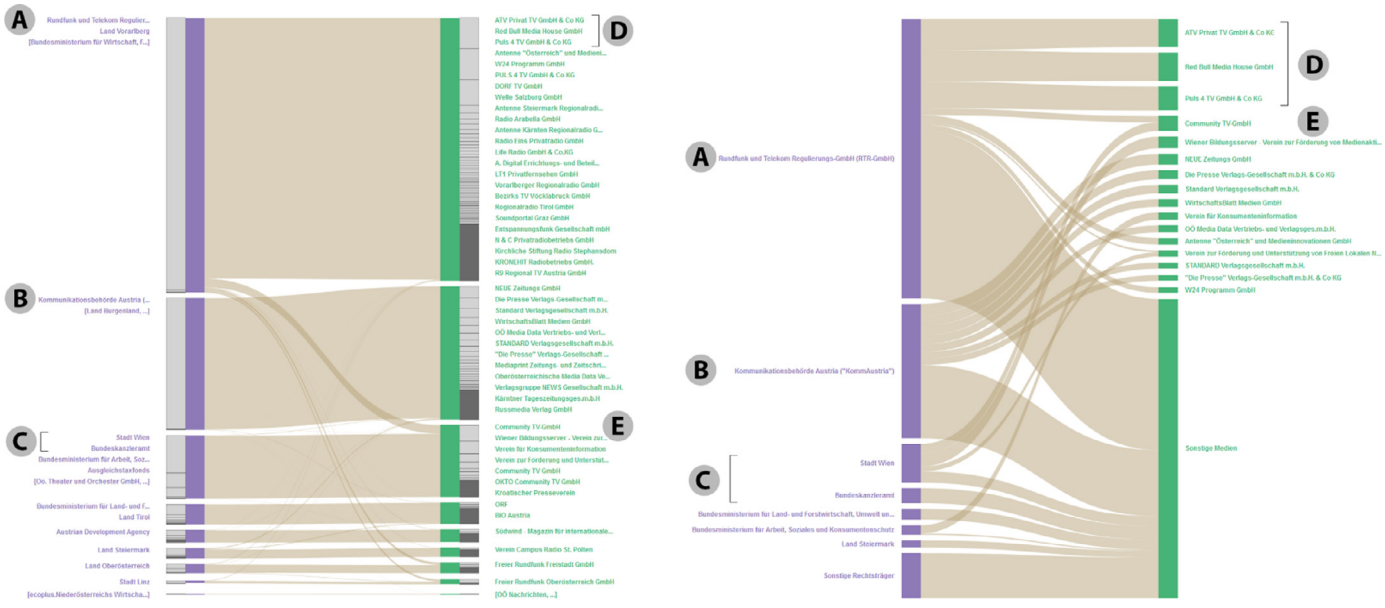

Fig. 17. The two study conditions showing press subsidies reported in the Media Transparency Database: BiCFlows (left) and the baseline condition (Cut-Off) using only filtering, but no aggregation (right). Annotations illustrate how the first three clusters on the left correlate with the four highest ranked legal entities listed in the Cut-Off visualization (A: *Rundfunk und Telekom Regulierungs-GmbH*, B: *Kommunikationsbehörde Austria*, C: *Stadt Wien* and *Bundeskanzleramt*), and the four highest ranked media companies contribute to two clusters (D: *ATV Privat TV GmbH & Co KG, Red Bull Media House GmbH*, and *Puls 4 TV GmbH & Co KG*, E: *Community TV-GmbH*). The two bottom entities in the Cut-Off visualization comprise all other (*"sonstige"*) entities of both sets.

5.2. Design

We employed a within-subjects design with visualization as independent variable, with the two levels BiCFlows (BiC) and Cut-Off (CO). The presentation order of the two visualizations was counterbalanced.

We used two subsets of the Media Transparency Database for the evaluation. The first data set, comprising only advertising objectives, contained 1226 legal entities and 3544 media organizations. We used nine clusters with a modularity of 0.39. The second data set, containing only press subsidies, had 68 legal entities and 885 media organizations (Fig. 17). We also used nine clusters, yielding a modularity of 0.62. This means that the second data set was smaller with more coherent groups. The assignment of the two data sets to the two visualizations was also counter-balanced.

The study was conducted using the Mozilla Firefox web browser on a 27" monitor. Users had to fill out a consent form, a demographic questionnaire, and then read a printed task description. Every condition was preceded by a training period using a test data set. At the end of the evaluation, users had to fill out a post-experiment questionnaire.

5.3. Analysis

We recorded all user sessions using screen capturing and audio recording, and encouraged the participants to comment on everything they see or experience during the data exploration. After the experiment, we transcribed the audio recordings and performed open coding, yielding nine insight categories listed in Table 1. For each user, we aggregated the number of codes per condition and used these numbers for comparing insights to verify hypotheses H1.1-H1.4.

In addition, we also recorded the exploration time (H1.5) and the users' subjective usability ratings through the *System Usability Scale* (SUS) questionnaire [70] (H2). All obtained measures were statistically analyzed using Wilcoxon Signed-Rank tests instead of t-tests, as not all measures were normally distributed.

5.4. Results

To test hypothesis H1.1, we compared the number of *unique* entities mentioned by the users. Using BiC, users mentioned significantly more different entities compared to CO ($Z = 10.5$, $p = .045$, Fig. 18a). They also mentioned significantly more transaction sums ($Z = 1.5$, $p = .005$, Fig. 18b). *We can thereby confirm our hypothesis H1.1: Users mention more entities and transaction sums using BiCFlows.*

For hypothesis H1.2, we calculated the quartiles of all cumulated entity sums and compared the number of mentioned entities separately for the lower three quartiles. However, the number of mentions is almost equivalent for *Q1-Q3* (see Fig. 18c). *This disproves our hypothesis H1.2: Users do not mention more entities with smaller transaction sums using BiCFlows.*

To verify H1.3, we looked at utterances coded as duplicates, time, geography, comparisons, and reasonings. Detection of duplicates was generally low, and the difference between the two conditions is not significant ($Z = 2$, $p = .257$, Fig. 18d). Mentions of temporal relations were a little bit more common, but also comparable between the conditions ($Z = 14$, $p = .310$, Fig. 18d). While no user made any remark on geographic connections using CO, there were a few mentions of geographic relations using BiC (see Fig. 18f). Finally, users did not make significantly more comparisons in either interface ($Z = 16$, $p = .774$, Fig. 18g) and did not significantly reason more about the data using BiC ($Z = 20$, $p = .234$, Fig. 18h). *We can therefore partially confirm H1.3: users discovered some geographic connections between entities using BiC and none with CO, but they did not find significantly more duplicates, temporal relations, or other commonalities or differences between entities.*

For hypothesis H1.4, we compared the number of unknown entities and unexpected findings. There is no significant difference between the number of unknown entities discovered in the data set ($Z = 18.5$, $p = .633$, Fig. 18i). However, users discovered more unexpected information, which was indicated by astonished or disbelieving reactions, using BiC than using CO ($Z = 8$, $p = .045$, Fig. 18j). *Thus, we can partially confirm our hypothesis H1.4: Users discovered more unexpected information using BiCFlows, but did not find more unknown entities.*

To test H1.5, we compared the time each user spent exploring the two different interfaces. Users spent more time exploring data using BiC (23 min on average) than CO (17.5 min), which is a significant difference ($Z = 6.5$, $p = .032$, Fig. 18k). The average number of unique entities mentioned per minute, however, is very similar ($Z = 35$, $p = .754$, Fig. 18l). *This confirms hypothesis H1.5: Users did not discover entities at a faster rate using BiC, but rather spent a longer time exploring the data.*

Finally, we compared the users' ratings of the SUS questionnaire to test hypothesis H2. With an average SUS score of 82, CO was rated significantly higher than BiC with 72 ($Z = 4$, $p = .028$). *This confirms our hypothesis H2: Users perceived BiCFlows as more complex than the Cut-Off approach.*

5.5. Discussion

In summary, our study showed that users explored the visualization for a longer time using BiCFlows than the Cut-Off visualization, which does not use any hierarchical aggregation. The rate of insights per minute was comparable. This means that users discovered more entities (i.e., nodes) and more transaction sums (i.e., edges) when exploring the Media Transparency Database using BiCFlows because they were encouraged to perform longer explorations. In particular, they made more unexpected findings.

This higher number of insights, however, comes with a lower perceived usability. While both interfaces were rated as excellent according to SUS [70], their average scores are on the upper and lower bounds of the excellent rating, respectively. Informal feedback indicates that some users found *"this arrangement into groups"* irritating at the beginning compared to direct selection of entities from a sorted list, but gained sufficient understanding after exploring for a while. In particular, one user appreciated the possibility *"to go into more detail"* using the hierarchically aggregated drill-down interface.

It was therefore surprising to us that users did not find more entities with smaller expenses using BiCFlows than the baseline. We initially reasoned that a major strength of hierarchical aggregation would be that – after drilling down – the visualization would reveal those lower ranked nodes and edges, which never show up in the Cut-Off visualization. From the video recordings, one observation was that participants usually only went down one or two hierarchy levels. Entities with low accumulated edge weights are potentially not yet revealed. Using BiCFlows, most users did not interact with the text lists. In contrast, the major exploration interface of the Cut-Off approach was not the visualization itself, but the text list of ranked legal entities and media organizations. When using the Cut-Off approach, most users scrolled these lists far down and mentioned entities from these lists while scrolling.

The most common unexpected findings across both conditions were the advertising expenditures of the daily newspapers *Kronen Zeitung, Heute*, and *Österreich*, as well as irritation about the fact that the more popular TV station *ORF1* receives less money than the smaller TV station *ORF2*. However, users of BiCFlows mentioned more often that the government of the city of Vienna (*Stadt Wien*) advertises in a large number of media. We assume it is due to the aggregation into the large "others" group in the Cut-Off visualization that users cannot easily grasp the true number of edges of a selected node. Indeed, one user mentioned during the study that being able to reveal all nodes in this "others" group *"would be a dream"*.

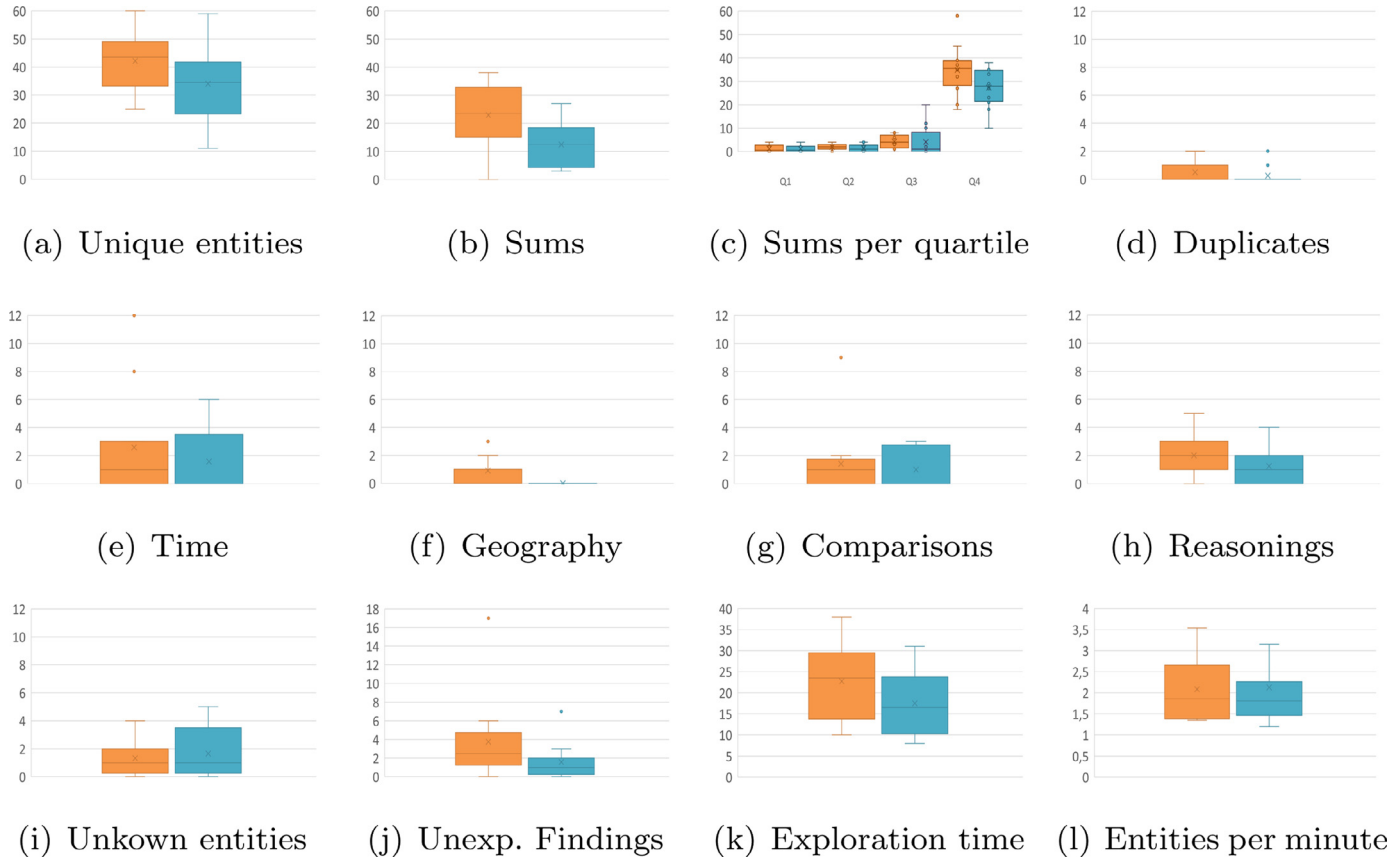

Fig. 18. Box plots of the number of coded insights per category (Table 1), as well as exploration times in minutes (k) and mentioned unique entities per minute (l). The left orange box plot shows the results of BicFlows, the right blue one of the Cut-Off visualization (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.).

Table 1
Coding categories of think-aloud protocols.

Code	Description
Entities	A mentioned legal entity or media organization.
Sums	Mentioned transaction sums between one legal entity and one media organization, or a total sum spent by a legal entity or received by a media organization.
Duplicates	Discovered entities with same or similar name, e.g., *google.at* and *google.de*, where users explicitly mentioned that these are the same.
Time	Quarters, years, or periods mentioned.
Geography	Geographical connections made for certain entities, e.g., "*DORF TV* is probably from Upper Austria too, because it's in the same group as other media organizations from Upper Austria."
Comparisons	Comparisons between entities or time periods, e.g., "*ÖBB* spent 19 million Euros, but compared to *Stadt Wien* that's nothing."
Reasoning	Generating hypotheses to explain an observation, e.g., "*Heute, Krone*, and *Österreich* receive most money, that's probably because they have most readers."
Unknown Entities	Entities that were unknown to the user, e.g., "*a3ECO*? - Never heard of it before."
Unexpected Findings	Unexpected findings or astonishments, e.g., "I can't believe *Stadt Wien* spends that much money."

5.6. Limitations

When formally comparing two user interfaces in a study, a potential validity threat is always not to be able to fully control for confounding factors. One potentially confounding factor in our study is the text label strategy, which differs between the two conditions. While we tried to maximize the number of node labels per group in BiCFlows to maximize the expressiveness of the cluster nodes, there is only a single label for each node in the Cut-Off visualization (see Fig. 17 b). This resulted in up to three times as many node labels in BiCFlows compared to the Cut-Off visualization. This can be an alternative explanation for the higher number of mentioned entities using BiCFlows. This can also partially explain why the users found the interface more complex initially. We therefore created the sparser labeling method shown in Fig. 2.

Since we did not systematically vary the data characteristics, our study also does not reveal how the size of the data set and the modularity of the clusters influence the effectiveness and understandability of the visualization. With more data, the system response will be slower and users will have to perform more interaction steps to reveal weaker nodes. With lower modularity, the meaningfulness of the visualized biclusters will decrease and may lead to misinterpretations of the data.

Generally, we could not thoroughly evaluate the quality of the coded comparisons, reasonings, and temporal or geographical insights, because such a quality analysis would require ground truth that does not exist for this dataset. For a deeper understanding of hierarchical aggregation based on graph topologies, future studies could investigate how users characterize commonalities of cluster elements to assess whether they correctly interpret the grouping. An example for the present dataset would be whether users believe that clusters were derived based on geographical locations of entities and incorrectly conclude that all legal entities and media organizations of a certain region are present in a selected cluster.

Finally, we did not thoroughly investigate the time-varying aspect of the dataset in this study. As a consequence, the number of mentioned temporal relations was quite low in both conditions (see Fig. 18d). In

the future, it will be necessary to formally evaluate the usability of the interface and the expressiveness of the visual encoding for the exploration of a large, time-dependent bipartite graph, such as shown in Fig. 2.

6. Conclusions

Dynamic BiCFlows is a new interactive visualization method to display large time-dependent bipartite graphs by combining hierarchical aggregation and filtering in linked lists. We explored two data sets with thousands of nodes and edges using two different exploration strategies: (1) aggregation through biclustering in combination with temporal filtering and (2) aggregation through time series clustering. We showed how the first method can be used to track the development of coherent groups over time and how the second method reveals groups of entities with similar temporal trends.

From our evaluation, we conclude that the major strength of hierarchical aggregation for large bipartite graphs is that users are encouraged to perform a deeper exploration of the data. As a consequence, they have more insights and discover more unexpected information. The limitation of such a hierarchical aggregation is a higher cognitive demand – at least initially – and a lower perceived usability for a lay audience. Based on these observations, we conclude that hierarchical aggregation is beneficial if the goal is to encourage users to perform a deep exploration of a large bipartite graph to discover unexpected information. However, if the goal is to provide a simple interface to primarily look for specific entities in a static bipartite graph, a visualization based on simple filtering combined with a search tool seems to be the more promising option.

For our use cases, we employed data sets with tens of thousands of entries, leading to thousands of nodes and edges, and dozens of time steps. In the future, we plan to extend our approach to much larger data sets. This will require significantly faster or incremental clustering methods. In addition, users will have to drill down more hierarchy levels to reveal all nodes. We also plan to investigate alternative visual encodings and interaction techniques to lower the initial cognitive demand and keep non-expert users engaged.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Acknowledgments

This work was financed by the Austrian Science Fund (FWF): T 752-N30.This paper was partly written in collaboration with the VRVis CompetenceCenter. VRVis is funded by BMVIT, BMWFW, Styria, SFG and ViennaBusiness Agency in the scope of COMET - Competence Centers for ExcellentTechnologies (854174) which is managed by FFG.

References

[1] M.S. Rahman, Basic Graph Theory, Undergraduate Topics in Computer Science, Springer International Publishing, Cham, 2017.
[2] S.C. Madeira, A.L. Oliveira, Biclustering algorithms for biological data analysis: a survey, IEEE/ACM Trans. Comput. Biol. Bioinf. 1 (1) (2004) 24–45, https://doi.org/10.1109/TCBB.2004.2.
[3] D. Jiang, C. Tang, A. Zhang, Cluster analysis for gene expression data: a survey, IEEE Trans. Knowl. Data Eng. 16 (11) (2004) 1370–1386, https://doi.org/10.1109/TKDE.2004.68.
[4] B. Pontes, R. Girldez, J.S. Aguilar-Ruiz, Biclustering on expression data: A review, J. Biomed. Inf. 57 (2015) 163–180, https://doi.org/10.1016/j.jbi.2015.06.028.
[5] J. Stasko, C. Görg, Z. Liu, Jigsaw: Supporting Investigative Analysis through Interactive Visualization, Proceedings of the IEEE Symposium on Visual Analytics Science and Technology, (2007), pp. 131–138.
[6] M. Sun, P. Mi, C. North, N. Ramakrishnan, BiSet: Semantic Edge Bundling with Biclusters for Sensemaking, IEEE Trans. Visual. Comput. Graph. 22 (1) (2016) 310–319, https://doi.org/10.1109/TVCG.2015.2467813.
[7] S. Ghani, B.C. Kwon, S. Lee, J.S. Yi, N. Elmqvist, Visual analytics for multimodal social network analysis: a design study with social scientists, IEEE Trans. Visual. Comput. Gr. 19 (12) (2013) 2032–2041.
[8] P. Isenberg, F. Heimerl, S. Koch, T. Isenberg, P. Xu, C.D. Stolper, M.M. Sedlmair, J. Chen, T. Möller, J. Stasko, vispubdata.org: a metadata collection about IEEE visualization (VIS) publications, IEEE Trans. Visual. Comput. Gr. 23 (9) (2017) 2199–2206.
[9] Bundeskanzleramt, BGBl. I Nr. 125/2011, 2011, (https://www.ris.bka.gv.at/eli/bgbl/I/2011/125/20111227). [Online; accessed Feb-2019].
[10] Z. Pousman, J. Stasko, M. Mateas, Casual information visualization: Depictions of data in everyday life, IEEE Trans. Visual. Comput. Gr. 13 (6) (2007) 1145–1152.
[11] D. Steinböck, E. Gröller, M. Waldner, Casual visual exploration of large bipartite graphs using hierarchical aggregation and filtering, Proceedings of teh International Symposium on Big Data Visual and Immersive Analytics (BDVA), IEEE, 2018, pp. 1–10.
[12] I. Herman, G. Melançon, M.S. Marshall, Graph visualization and navigation in information visualization: A survey, IEEE Trans. Visual. Comput. Graph. 6 (1) (2000) 24–43.
[13] T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J.J. van Wijk, J.-D. Fekete, D.W. Fellner, Visual Analysis of Large Graphs: State-of-the-Art and Future Research Challenges, Computer Graphics Forum 30 (6) (2011) 1719–1749.
[14] K. Misue, Drawing Bipartite Graphs As Anchored Maps, Proceedings of the Asia-Pacific Symposium on Information Visualisation - Volume 60, Australian Computer Society, Inc., 2006, pp. 169–177.
[15] M. Dumas, M.J. McGuffin, J.-M. Robert, M.-C. Willig, Optimizing a radial layout of bipartite graphs for a tool visualizing security alerts, Proceedings of the International Symposium on Graph Drawing, Springer, Berlin, Heidelberg, 2011, pp. 203–214, https://doi.org/10.1007/978-3-642-25878-7_20.
[16] C.F. Dormann, J. Fründ, N. Blüthgen, B. Gruber, Indices, graphs and null models: Analyzing bipartite ecological networks, Open Ecol. J. 2 (1) (2009) 7–24.
[17] H.-J. Schulz, M. John, A. Unger, H. Schumann, Visual Analysis of Bipartite Biological Networks, Proceedings of the First Eurographics Conference on Visual Computing for Biomedicine, Eurographics Association, 2008, pp. 135–142.
[18] M. Dörk, N.H. Riche, G. Ramos, S. Dumais, PivotPaths: Strolling through Faceted Information Spaces, IEEE Trans. Visual. Comput. Gr. 18 (12) (2012) 2709–2718.
[19] C. Stoiber, A. Rind, F. Grassinger, R. Gutounig, E. Goldgruber, M. Sedlmair, Š. Emrich, W. Aigner, Netflower: Dynamic network visualization for data journalists, Comput. Graph. Forum 38 (3) (2019) 699–711, https://doi.org/10.1111/cgf.13721.
[20] N. Elmqvist, J.-D. Fekete, Hierarchical Aggregation for Information Visualization: Overview, Techniques, and Design Guidelines, IEEE Trans. Visual. Comput. Gr. 16 (3) (2010) 439–454.
[21] D. Archambault, T. Munzner, D. Auber, GrouseFlocks: Steerable Exploration of Graph Hierarchy Space, IEEE Trans. Visual. Comput. Graph. 14 (4) (2008) 900–913.
[22] V. Yoghourdjian, T. Dwyer, K. Klein, K. Marriott, M. Wybrow, Graph thumbnails: Identifying and comparing multiple graphs at a glance, IEEE Trans. Visual. Comput. Gr. 24 (12) (2018) 3081–3095.
[23] N. Henry, J.-D. Fekete, M.J. McGuffin, NodeTrix: a Hybrid Visualization of Social Networks, IEEE Trans. Visual. Comput. Gr. 13 (6) (2007) 1302–1309.
[24] S. Rufiange, M.J. McGuffin, C.P. Fuhrman, TreeMatrix: A Hybrid Visualization of Compound Graphs, Comput. Gr. Forum 31 (1) (2012) 89–101.
[25] N. Elmqvist, T.-N. Do, H. Goodell, N. Henry, J.-D. Fekete, Zame: Interactive large-scale graph visualization, Proceedings of the IEEE Pacific Visualization Symposium, 2008, (2008), pp. 215–222.
[26] B. Mirkin, Mathematical classification and clustering: From how to what and why, Classification, Data Analysis, and Data Highways, Springer, 1998, pp. 172–181.
[27] I.S. Dhillon, Co-clustering documents and words using bipartite spectral graph partitioning, Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, (2001), pp. 269–274, https://doi.org/10.1145/502512.502550.
[28] M. Rege, M. Dong, F. Fotouhi, Co-clustering Documents and Words Using Bipartite Isoperimetric Graph Partitioning, Proceedings of the Sixth International Conference on Data Mining, IEEE, 2006, pp. 532–541, https://doi.org/10.1109/ICDM.2006.36.
[29] S. Barkow, S. Bleuler, A. Preli, P. Zimmermann, E. Zitzler, BicAT: a biclustering analysis toolbox, Bioinformatics 22 (10) (2006) 1282–1283, https://doi.org/10.1093/bioinformatics/btl099.
[30] D. Filippova, A. Gadani, C. Kingsford, Coral: an integrated suite of visualizations for comparing clusterings, BMC Bioinformatics 13 (1) (2012) 276, https://doi.org/10.1186/1471-2105-13-276.
[31] M. Sun, C. North, N. Ramakrishnan, A Five-Level Design Framework for Bicluster Visualizations, IEEE Trans. Visual. Comput. Gr. 20 (12) (2014) 1713–1722, https://doi.org/10.1109/TVCG.2014.2346665.
[32] R. Santamara, R. Thern, L. Quintales, BicOverlapper: A tool for bicluster visualization, Bioinformatics 24 (9) (2008) 1212–1213.
[33] P. Fiaux, M. Sun, L. Bradel, C. North, N. Ramakrishnan, A. Endert, Bixplorer: visual analytics with biclusters, Computer 46 (8) (2013) 90–94.
[34] M. Streit, S. Gratzl, M. Gillhofer, A. Mayr, A. Mitterecker, S. Hochreiter, Furby: Fuzzy force-directed bicluster visualization, BMC Bioinf. 15 (6) (2014) S4.
[35] P. Xu, N. Cao, H. Qu, J. Stasko, Interactive visual co-cluster analysis of bipartite graphs, Proceedings of the IEEE Pacific Visualization Symposium, (2016), pp. 32–39.
[36] Y. Onoue, N. Kukimoto, N. Sakamoto, K. Koyamada, Minimizing the number of

edges via edge concentration in dense layered graphs, IEEE Trans. Visual. Comput. Gr. 22 (6) (2016) 1652–1661, https://doi.org/10.1109/TVCG.2016.2534519.
[37] M. Sun, J. Zhao, H. Wu, K. Luther, C. North, N. Ramakrishnan, The effect of edge bundling and seriation on sensemaking of biclusters in bipartite graphs, IEEE Trans. Visual. Comput. Gr. (2018). 1–1
[38] J. Zhao, M. Sun, F. Chen, P. Chiu, BiDots: Visual Exploration of Weighted Biclusters, IEEE Trans. Visual. Comput. Gr. 24 (1) (2018) 195–204.
[39] G.Y.-Y. Chan, P. Xu, Z. Dai, L. Ren, V i b r: Visualizing bipartite relations at scale with the minimum description length principle, IEEE Trans. Visual. Comput. Gr. 25 (1) (2019) 321–330.
[40] N. Pezzotti, J.-D. Fekete, T. Höllt, B. Lelieveldt, E. Eisemann, A. Vilanova, Multiscale visualization and exploration of large bipartite graphs, Proceedings of the Computer Graphics Forum, 37 Wiley Online Library, 2018, pp. 549–560.
[41] F. Beck, M. Burch, S. Diehl, D. Weiskopf, A taxonomy and survey of dynamic graph visualization, Comput. Gr. Forum 36 (1) (2017) 133–159.
[42] S. Hadlak, H. Schumann, H.-J. Schulz, A survey of multi-faceted graph visualization, Proceedings of the Eurographics Conference on Visualization (EuroVis). The Eurographics Association, (2015), pp. 1–20.
[43] M. Burch, B. Schmidt, D. Weiskopf, A matrix-based visualization for exploring dynamic compound digraphs, Proceedings of the 17th International Conference on Information Visualisation, IEEE, 2013, pp. 66–73.
[44] J.S. Yi, N. Elmqvist, S. Lee, Timematrix: Analyzing temporal social networks using interactive matrix-based visualizations, Int. J. Human Comput. Interact. 26 (11-12) (2010) 1031–1051.
[45] T.W. Liao, Clustering of time series data – a survey, Pattern Recogn. 38 (11) (2005) 1857–1874.
[46] S. Aghabozorgi, A.S. Shirkhorshidi, T.Y. Wah, Time-series clustering – a decade review, Inf. Syst. 53 (2015) 16–38.
[47] J.J. Van Wijk, E.R. Van Selow, Cluster and calendar based visualization of time series data, Proceedings of the IEEE Symposium on Information Visualization, IEEE, 1999, pp. 4–9.
[48] R. Kincaid, H. Lam, Line graph explorer: scalable display of line graphs using focus + context, Proceedings of the Working Conference on Advanced Visual Interfaces, ACM, 2006, pp. 404–411.
[49] M. Steiger, J. Bernard, S. Mittelstädt, H. Lücke-Tieke, D. Keim, T. May, J. Kohlhammer, Visual analysis of time-series similarities for anomaly detection in sensor networks, Proceedings of the Computer Graphics Forum, 33 Wiley Online Library, 2014, pp. 401–410.
[50] C. Stolte, D. Tang, P. Hanrahan, Multiscale visualization using data cubes, IEEE Trans. Visual. Comput. Gr. 9 (2) (2003) 176–187.
[51] A. Tanay, R. Sharan, R. Shamir, Discovering statistically significant biclusters in gene expression data, Bioinformatics 18 (1) (2002) 136–144, https://doi.org/10.1093/bioinformatics/18.suppl_1.S136.
[52] M. Ailem, F. Role, M. Nadif, Co-clustering Document-term Matrices by Direct Maximization of Graph Modularity, Proceedings of the 24th ACM International Conference on Information and Knowledge Management, (2015), pp. 1807–1810, https://doi.org/10.1145/2806416.2806639.
[53] U. Brandes, D. Delling, M. Gaertler, R. Gorke, M. Hoefer, Z. Nikoloski, D. Wagner, On modularity clustering, IEEE Trans. Knowl. Data Eng. 20 (2) (2007) 172–188.
[54] F. Bourgeois, J.-C. Lassalle, An extension of the munkres algorithm for the assignment problem to rectangular matrices, Commun. ACM 14 (12) (1971) 802–804.
[55] P. Riehmann, M. Hanfler, B. Fröhlich, Interactive Sankey diagrams, Proceedings of the IEEE Symposium on Information Visualization, (2005), pp. 233–240.
[56] F. Bendix, R. Kosara, H. Hauser, Parallel Sets: Visual Analysis of Categorical Data, Proceedings of the IEEE Symposium on Information Visualization, IEEE, 2005, pp. 133–140.
[57] L. Byron, M. Wattenberg, Stacked graphs – geometry & aesthetics, IEEE Trans. Visual. Comput. Gr. 14 (6) (2008).
[58] R. Bade, S. Schlechtweg, S. Miksch, Connecting time-oriented data and information to a coherent interactive visualization, Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, ACM, 2004, pp. 105–112.
[59] E.R. Tufte, Beautiful Evidence, 1 Graphics Press Cheshire, CT, 2006.
[60] B. Shneiderman, The eyes have it: A task by data type taxonomy for information visualizations, The Craft of Information Visualization, Elsevier, 2003, pp. 364–371.
[61] M. Bostock, V. Ogievetsky, J. Heer, D^3 - Data-Driven Documents, IEEE Trans. Visual. Comput. Gr. 17 (12) (2011) 2301–2309.
[62] NPashaP, Viz - biPartite - default, 2018, (http://bl.ocks.org/npashap/12cd547b1a3270603a139186b05415ff). [Online; accessed Feb-2019].
[63] Square, Crossfilter, 2012, (https://square.github.io/crossfilter). [Online; accessed Feb-2019].
[64] D. Steinböck, E. Gröller, M. Waldner, BiCFlows, 2018, (https://users.cg.tuwien.ac.at/~waldner/bicflows/). "[Online; accessed Feb-2019]".
[65] A. Rind, D. Pfahler, C. Niederer, W. Aigner, Exploring media transparency with multiple views, Proceedings of the 9th Forum Media Technology, CEUR-WS, 2016, pp. 65–73.
[66] P. Salhofer, MEHR! Medientransparenz, 2017, (https://www.medien-transparenz.at). [Online; accessed Feb-2019].
[67] University of Trier, dblp computer science bibliography, 2018, (https://dblp.uni-trier.de/). [Online; accessed Feb-2019].
[68] C. North, Toward measuring visualization insight, IEEE Comput. Gr. Appl. 26 (3) (2006) 6–9.
[69] P. Saraiya, C. North, K. Duca, An insight-based methodology for evaluating bioinformatics visualizations, IEEE Trans. Visual. Comput. Gr. 11 (4) (2005) 443–456.
[70] J. Brooke, SUS - A quick and dirty usability scale, Usability Evaluat. Ind. 189 (194) (1996) 4–7.