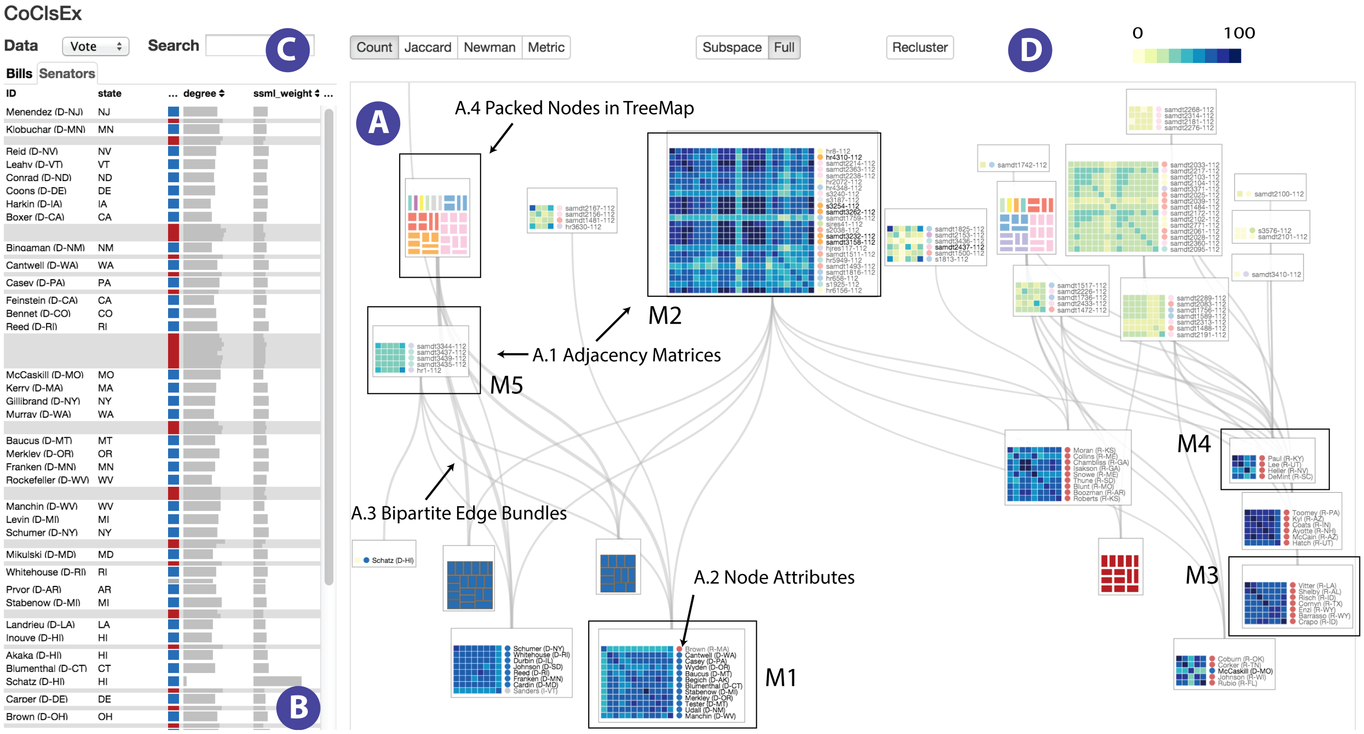
Interactive Visual Co-Cluster Analysis of Bipartite Graphs

Panpan Xu[*] Bosch Research North America

Nan Cao[†] NYU Shanghai

Huamin Qu[‡] HKUST

John Stasko[§] Georgia Tech



ID	state	...	degree	ssml_weight	...
Menendez (D-NJ)	NJ				
Klobuchar (D-MN)	MN				
Reid (D-NV)	NV				
Leahv (D-VT)	VT				
Conrad (D-ND)	ND				
Coons (D-DE)	DE				
Harkin (D-IA)	IA				
Boxer (D-CA)	CA				
Bingaman (D-NM)	NM				
Cantwell (D-WA)	WA				
Casev (D-PA)	PA				
Feinstein (D-CA)	CA				
Bennet (D-CO)	CO				
Reed (D-RI)	RI				
McCaskill (D-MO)	MO				
Kerrv (D-MA)	MA				
Gillibrand (D-NY)	NY				
Murrav (D-WA)	WA				
Baucus (D-MT)	MT				
Merklev (D-OR)	OR				
Franken (D-MN)	MN				
Rockefeller (D-WV)	WV				
Manchin (D-WV)	WV				
Levin (D-MI)	MI				
Schumer (D-NY)	NY				
Mikulski (D-MD)	MD				
Whitehouse (D-RI)	RI				
Prvor (D-AR)	AR				
Stabenow (D-MI)	MI				
Landrieu (D-LA)	LA				
Inouve (D-HI)	HI				
Akaka (D-HI)	HI				
Blumenthal (D-CT)	CT				
Schatz (D-HI)	HI				
Carper (D-DE)	DE				
Brown (D-OH)	OH				

Fig. 1. The interface of our system showing the bipartite relation of U.S. senators' support of bills and amendments based on roll call vote records. The main view (A) displays clusters of senators in the bottom half and bills in the top half, based on whether the senators support common sets of bills and whether the bills are supported by the same group of senators. The clusters are determined automatically with co-clustering algorithms and displayed via adjacency matrices showing the cohesiveness of the clusters, or a treemap-like space filling layout of the nodes. Using color coded party affiliations, an immediate observation is that the senators mostly vote in accordance with their parties.

Abstract—A bipartite graph models the relation between two different types of entities. It is applicable, for example, to describe persons' affiliations to different social groups or their association with subjects such as topics of interest. In these applications, it is important to understand the connectivity patterns among the entities in the bipartite graph. For the example of a bipartite relation between persons and their topics of interest, people may form groups based on their common interests, and the topics also can be grouped or categorized based on the interested audiences. Co-clustering methods can identify such connectivity patterns and find clusters within the two types of entities simultaneously. In this paper, we propose an interactive visualization design that incorporates co-clustering methods to facilitate the identification of node clusters formed by their common connections in a bipartite graph. Besides highlighting the automatically detected node clusters and the connections among them, the visual interface also provides visual cues for evaluating the homogeneity of the bipartite connections in a cluster, identifying potential outliers, and analyzing the correlation of node attributes with the cluster structure. The interactive visual interface allows users to flexibly adjust the node grouping to incorporate their prior knowledge of the domain, either by direct manipulation (i.e., splitting and merging the clusters), or by providing explicit feedback on the cluster quality, based on which the system will learn a parametrization of the co-clustering algorithm to better align with the users' notion of node similarity. To demonstrate the utility of the system, we present two example usage scenarios on real world datasets.

1 Introduction

Bipartite relations, the connections between two different types of entities, play a key role in gaining insight from data in many application domains. Examples of bipartite relations include the votes from legislators for the passage of bills and amendments, the involvement of researchers in various topics, and the affiliation of individuals with different social groups [46].

In bipartite relation data, frequently a person not only wants to know the neighbors and links of individual nodes, but also the commonality in their connections. For example, to what extent do two or more

[*] e-mail: panpan.xu@us.bosch.com
[†] e-mail: nan.cao@nyu.edu
[‡] e-mail:huamin@cse.ust.hk
[§] e-mail:stasko@cc.gatech.edu

researchers investigate similar topics, or how much do senators agree in their votes? Analyzing the commonality of the connections results in clusters of researchers working on the same topics, or groups of legislators supporting similar sets of bills. Moreover, the analysis of similarity in connections can be applied to both types of nodes in the same manner, which is the notion of duality in bipartite relation analysis [3]. For instance, it is possible to identify not only the senators who have voted for a similar set of bills, but also the group of bills that are supported (or not) by similar groups of senators.

Computational approaches including various clustering and co-clustering algorithms [7, 8, 9, 13, 20] can be applied to identify cohesive node groups with similar bipartite connections. Visually denoting the clusters found by the algorithms in a bipartite graph visualization facilitates the process of understanding and cluster identification by relieving the analysts of the burden of identifying and comparing the links individually for each node. By aggregating the nodes, the visual complexity is also reduced, and an overview of the connectivity is possible.

However, we argue that the clusters obtained by running such algorithms should serve as a starting point, rather than the end of analysis, for several reasons. First, the clustering results generated by these algorithms may not be very helpful due to suboptimal parameter settings (e.g., number of clusters [7, 8, 9]). Some clusters might not be very cohesive because nodes that differ significantly in their bipartite connections were grouped together. Second, the ability to explore subspace clusters is desirable in many application scenarios. For example, in clustering the bills, an analyst may consider grouping them based on the support from a particular political party, instead of all the legislators. In this case, the bill clusters are identified within a subspace, if we consider each bill as a data item and each legislator as a dimension [20]. Third, in bipartite relation data, the nodes are often associated with domain specific features. These features are important for integrating analysts' prior knowledge, and for generating insight about the correlation of node attributes with node groups.

To address these shortcomings, we advocate for a visual analytics approach. It fits well here because of the need for both automated cluster analysis and the active engagement of analysts to evaluate and refine the clusters, drill down to a subspace, and explore the correlation between node attributes and their bipartite connectivities [29, 44].Often, these visual analysis tasks (e.g., refining the cluster) require taking into account analysts' notions of node similarities, which might differ from a straightforward comparison of all the bipartite connections for a pair of nodes. Thus, we introduce a prototype visual analytics system that incorporates a semi-supervised clustering approach, which learns similarity metrics for nodes based on user specified constraints. In this way, analysts' judgments can be included in the automatic clustering procedure. The major contributions of this work are:

- The task analysis and design of a visual analytics system for the explorative analysis of connectivity patterns in bipartite graphs through flexible node grouping. Our system employs automatic analysis algorithms, novel visual representations, and advanced interactions to help analysts identify, interpret, compare, and refine co-cluster patterns in bipartite graphs.
- A novel algorithm that applies Laplacian Regularized Metric Learning (LRML) [26], an efficient metric learning method, for semi-supervised co-clustering analysis of bipartite graphs.
- A flexible visualization designed for representing clusters in bipartite graphs that illustrates both relational and feature patterns of nodes in clusters via an adjacency matrix and treemap respectively.
- The demonstration of the utility of the system through example use scenarios on two datasets: the roll call vote records of US Senators on the passage of bills and amendments, and the topic interest of researchers.

2 Background

Given a bipartite graph as described in the Introduction, various clustering algorithms can be employed to detect groups of nodes that are in some sense similar in their bipartite connections. For example, given the bipartite graph describing how scientists jointly publish papers, we can derive a co-authorship graph and apply community detection algorithms to identify groups of researchers working closely together. For clustering algorithms in general, a good result has high inner-cluster similarity (i.e., the data items within the same cluster are similar) and low inter-cluster similarity (i.e., the data items in different clusters are dissimilar).

Co-clustering is another category of clustering algorithms which is able to detect node groupings on the two modes in a bipartite graph simultaneously. Examples of co-clustering algorithms include [8, 9, 7, 13]. The co-clustering algorithms have been applied in different contexts such as analyzing gene expression data to find the relations between genes and conditions, grouping documents and words to identify topic groups, and so forth. Most of the co-clustering algorithms can generate row and column clusters (hence clusters in the two modes) given the biadjacency matrix, create a "checkerboard" structure which reflects the common bipartite connections from the nodes within the same row / column clusters. In the prototype, we apply a spectral co-clustering algorithm [8], although it should be noted that other algorithms can also be integrated in the framework.

3 Related Work

3.1 Graph Visualization

Node-link diagrams and adjacency matrices are the two major visualization techniques for graphs [25, 45]. Adjacency matrices generally introduce less visual clutter for denser graphs, but node-link diagrams are more familiar for general users and are arguably more intuitive for understanding a graph's structure [17, 24]. Hybrid designs combine a node-link diagram and an adjacency matrix by showing the two views simutaneously and synchronizing the interactions on the views [23], or showing the graph structure at global and community level with varying visual representations [24, 36]. We also adopt a hybrid visual design in the system introduced in this paper, to represent both the bipartite connections and the projected one-mode graph.

3.2 Bipartite and Multimodal Graph Visualization

A common approach for visualizing bipartite graphs, or more generally, multimodal graphs, is to allocate separate panes/lists for the different types of nodes. Graphical links connect nodes on different panes, and are displayed selectively based on the user's current focus. Variations of this approach can be found in semantic substrates [40], in the List View in Jigsaw [41], in linked tabular views [38], and more recently, in PivotPath [10] and MMGraph [15]. Using color, shape or other visual channels to denote the node type instead of spatially separating the different types of nodes, and drawing the graph in a unimodal style is also a possible choice. OntoVis [39] and FacetAtlas[6] are examples.

Ploceus [30] and Orion [21] focus on interactive graph modeling and transformation. One important transformative operation supported by the two systems is *projection*, which derives, for example, the co-vote relations among legislators given their voting records. For attributed multimodal graphs, GraphTrail [11] and NetLens [28] aggregate along the nodes' attributes and display views of summary statistics.

3.3 Co-clustering Visualization

Co-clustering (or biclustering, subspace clustering) has been widely adopted for analyzing gene expression data in bioinformatics [31], where the genes and the conditions (which correspond to the two types of entities in bipartite graphs) are clustered simultaneously based on the expression levels. Various visualization systems have been developed to help review and analyze the resulting clusters [1, 18, 22, 37]. Recently, the approach is applied to intelligence analysis, where coordinated relations between sets of entities of different types (people, locations, etc.) are identified and visualized [14, 43]. The entities analyzed can be extracted from textual datasets, connected by their co-occurence in documents. Other work visualizing multimode graphs such as FacetAtlas [6] and SolarMap [4] focuses on one-mode clusters and their interconnections through indirect links.

In this article, we propose an interactive visualization for analyzing bipartite graph data. Our method combines clustering algorithms and visualization, which facilitates the identification of connectivity patterns in bipartite graph data while providing users with enough flexibility to explore based on their domain knowledge and other criteria, thus differentiating it from the previous work.

4 System Design and Implementation

In this section, we explain the design and implementation of our system for co-cluster analysis of bipartite graphs. It is based on a set of design requirements that we formulated by reviewing representative analysis tasks on bipartite graphs from a number of different domains. We begin below by listing the requirements. We focus primarily on tasks related to the understanding of connectivity patterns of a bipartite graph.

R1 *Determine node connections across the two groups.* Connections between nodes across the two groups should be evident. For example, which bills did a senator vote in favor of?

R2 *Determine specific attribute values of each type of node.* The different attributes of multivariate nodes should be evident. For example, from which political party and state is a particular senator?

R3 *Identify similar nodes of each of the two types.* The system should determine and present clusters of similar nodes based on the bipartite connection. For example, which senators voted similarly toward all the bills?

R4 *Analyze correlation of the connectivity pattern with domain specific node attributes.* Mechanisms for selecting and visualizing relevant node attributes should be provided in the system, so that the analysts can inspect their correlation with the bipartite connections to draw insights. For example, which bills did Democratic senators vote for consistently?

R5 *Interpret the node clusters through bipartite relations.* Given a node cluster, it is important to be able to understand which common neighbors made the nodes similar to each other. Thus, besides the clusters of nodes, the bipartite relation also should be viewable. For example, for a cluster of senators, which votes made them be grouped together?

R6 *Explore subspace clusters.* The system should provide means for selecting subsets of nodes to manually identify subspace clusters. The visualization should be updated upon selection to reflect the cohesiveness of node clusters within the subspace. For example, the user should be able to select a subset of senators and observe how similar they are in different ways.

R7 *Evaluate and refine node clusters.* To assist the search for high quality node groupings, the system needs to provide effective visual cues to communicate the cohesiveness of the clusters, and the degree of separation among them. Moreover, if nodes in a cluster are dissimilar, the visual design should enable the identification of subsets that are more coherent than the others in their bipartite connections to suggest potential ways to regroup the nodes. Analysts should be able to merge and split the existing node clusters by directly interacting with the visual representations. The system should give immediate feedback on the cohesiveness of updated clusters to effectively guide the exploration process. Besides providing the visual cues and the interactive tools for the user to regroup the nodes, the system also should ingest the user's notion of similarity and learn a proper metric to compare the connectivity of two nodes. For example, if the user feels a system-determined cluster of senators is not appropriate, it should be possible to change the clustering and have the system interpret and learn from that.

4.1 System Overview

With the above tasks and requirements as a basis, we designed and implemented a system for flexible exploration of bipartitie graphs. Our visual analysis system follows the architectural design illustrated in Fig. 2. Once a dataset is loaded into the system, the analysis module will detect the co-clusters and send the results to be visualized. Analysts navigate through the visual representations of co-clustering results, and refine the clusters based on their prior knowledge, preferences, and judgments by directly interacting with the visualizations. The analysts' beliefs will be fed back into a metric learning model, and captured in a distance metric that will be later used for the next round of co-clustering analysis.

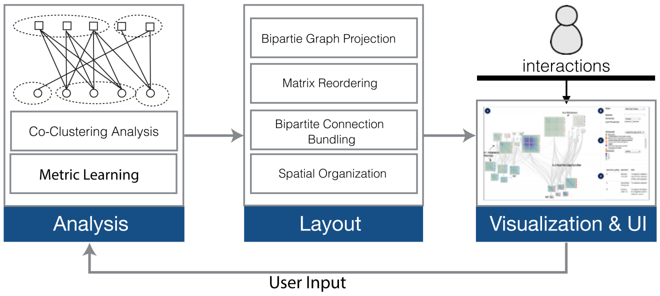


Fig. 2. System overview.

The visualization module consists of multiple visualization regions which are coordinated in the user interface shown in Fig. 1. The UI consists of four major components. The first region (A) contains the bipartite graph representation that shows the co-clustering results. It allows users to directly interact with and refine the node clusters through splitting and merging, or specify the groups of nodes they consider as similar to each other. This region provides two alternative visual displays of each node cluster: an adjacency matrix of the projected graph and a treemap-like compact packing of the nodes. The second region (B) contains a Table Lens display [34] for visualizing the node attributes. The categorical and numerical attribute values are encoded graphically via the colors and lengths of the rectangles within the table cells. The third region (C) provides a data manipulation panel including tools for users to load different datasets and query the nodes by labels. The final region (D) contains a control panel for selecting alternative metrics for measuring the similarity of bipartite connections for a pair of nodes, considering either a subspace or the full set of dimensions.

4.2 Analytic Support

Before describing details of the system interface and interactions, we explain the metric learning approach applied for co-cluster refinement. Here, we formally denote a bipartite graph as $G = (U,V,E)$, where U and V are non-overlapping sets corresponding to the two types of entities. $E \subseteq U \times V$ is the bipartite relation between the two sets of entities. The edges in E are undirected and can be weighted.

4.2.1 Semi-Supervised Co-Cluster Analysis

We introduce a semi-supervised co-clustering method for analyzing bipartite graphs as summarized in Alg. 1, which learns distance metrics between the nodes based on analysts' input and uses it for co-clustering analysis. It is a combination of two techniques: (1) the co-cluster analysis of bipartite graphs with the learned distance metrics in the form of weightings on the nodes U and V in the bipartite graph, and (2) the semi-supervised metric learning for finding the distance metrics based on the similarity constraints specified by the users.

4.2.2 Metric Learning

In cluster analysis, determining a proper distance metric to estimate the similarities among data items is the most important task that directly determines the quality of the cluster results. In most cases, distance metrics are predefined and remain unchanged during the clustering procedure, and the most familiar one is Euclidean distance. However, the metric learning technique [2] suggests an adaptive procedure in which the distance metric is learned based on several examples of the similarities between data items that are manually labeled by the analyzer, thus capturing their prior knowledge and intuition, and making it a good match for interactive visual analysis. Particularly, with information on pairs of nodes that are labeled as 'similar' to each other, the metric learning algorithms can find a parametrization of a family of distance functions (e.g., Mahalanobis distances) such that the computed distances between 'similar' pairs are small.

A bipartite graph $G = (U,V,E)$, can be represented as a biadjacency matrix $B_{\|U\| \times \|V\|}$. The row vectors $B_{i\cdot}$ and the column vectors $B_{\cdot j}$ correspond to the bipartite connections of node $u_i \in U$ and node $v_j \in V$. We employ a semi-supervised metric learning method, Laplacian Regularized Metric Learning (LRML) [26], to find the distance metrics between two row vectors or two column vectors given a set of similarity constraints on the nodes in either partition specified by the analysts.

In particular, LRML finds a parameterization of the Mahalanobis distance for measuring the distance between a pair of nodes. For example, for two nodes u_{i_0} and u_{i_1} in U, the distance function is:

$$d(u_{i_0}, u_{i_1})_A = \sqrt{(B_{i_0\cdot} - B_{i_1\cdot})^T A (B_{i_0\cdot} - B_{i_1\cdot})} \qquad (1)$$

where A is a semi-positive definite matrix of size $|V| \times |V|$. When A is the identity matrix, the metric reduces to the Euclidean distance. When A is a diagonal matrix, the value of the diagonal entry A_{jj} would reflect the relative importance of the node v_j in V for measuring the similarity of the nodes u_{i_0} and u_{i_1} in U. When applying LRML to semi-supervised co-clustering analysis, instead of a full matrix A, we seek for a diagonal matrix as a solution to the optimization problem. This is due to considerations on both the time complexity for solving the optimization problem and the interpretability of the results. As discussed above, the entries on the diagonal of A correspond to nodes in V and the value of the entries is the importance of the node when comparing the bipartite connection of a pair of nodes in U. In the following discussion, we will simply denote the diagonal of A as W_v.

In the LRML algorithm, given the similarity constraint S for pairs of nodes in U, LRML seeks the matrix A that minimizes the sum of distances for the pairs of nodes in S. A Laplacian regularization term is also included in the optimization goal.

$$\begin{aligned} \min_{A \succeq 0} & \sum_{(u_{i_0}, u_{i_1}) \in S} d(u_{i_0}, u_{i_1})_A \\ & + \lambda \sum_{u_{i_0} \in U} \sum_{u_{i_1} \in U} w(u_{i_0}, u_{i_1}) d(u_{i_0}, u_{i_1})_A \\ & - \varepsilon log(det(A)) \end{aligned}$$

$$w(u_{i_0}, u_{i_1}) = \begin{cases} 1, \text{if } u_{i_0} \in N(u_{i_1}) \\ 0, \text{otherwise} \end{cases}$$

$N(u_{i_1})$ is the nearest neighbor list of u_{i_1}, which includes the nodes in U with the most similar bipartite connections to u_{i_1} in terms of Euclidean distance. The term containing $log(det(A))$ is introduced to avoid trivial solutions where all the entries of A are zero.

We respectively apply the above metric learning algorithms for nodes in U and V based on the specified similarity constraints S_u for nodes in U and S_v for nodes in V (line 2,3 in Alg. 1), obtaining weights W_u and W_v. These weights indicate how important a node and its corresponding relationships should be considered during the co-clustering procedure in terms of the given set of similarity constraints.

```
Input :
        Bipartite Graph $G = (U,V,E)$,
        Similarity constraints $S_u$ and $S_v$ for nodes in $U$ and $V$,
        a k-nearest neighbor graph $G_u$ and $G_v$ for nodes in $U$ and $V$
Output:
        Grouping of nodes in $U$ and $V$
1 begin
2   $W_u$ = LRML($S_u$, $G$, $G_u$);
3   $W_v$ = LRML($S_v$, $G$, $G_v$);
4   $B$ = bijaceny_matrix($G$);
5   $B' = B + \alpha W_u^T \times B \times W_v$;
6   co-clustering($B'$);
7 end
```

Algorithm 1: Metric learning for co-cluster refinement

We "modulate" the biadjacency matrix B by these weight vectors (line 4,5 in Alg. 1), making the entry $B'_{i,j}$ (i.e., the bipartite link connecting nodes u_i and v_j) weighted, for example, comparatively less if both u_i and v_j's weights found by the metric learning algorithm are small. When compared to links connecting nodes with higher weights, those links will have less effect on the co-clustering results. The parameter α (line 5 in Alg. 1) controls the extent to which the user specified constraints should affect the clustering results. The co-clustering algorithm used in the prototype is the spectral co-clustering algorithm implemented in python scikit-learn package [8]. It requires the input of a pre-specified number of clusters. After the user input similarity constraints, the co-clustering algorithm is carried out again, with the same specified number of clusters prior to taking any user constraints into account. In this process, the weightings on the nodes learned by the LRML algorithm are used to modulate the weights of the bipartite connections, and this is how the user input is incorporated in the co-clustering algorithm. Finally, the nodes are clustered by running the co-clustering algorithms on B' (line 6 in Alg. 1).

There are a number of reasons of choosing LRML instead of other metric learning methods: (1) it is a semi-supervised algorithm which requires only partial labeling of the similar node pairs, thus it imposes lower demand on the users; (2) updating the parametrization of the distance function can be done efficiently without iterative solvers, and the efficiency is desirable for user-facing systems; (3) when the matrix A that parametrizes the Mahalanobis distance function is required to be diagonal, the value of corresponding entries on the diagonal can be interpreted as the weight of the features in the modulated distance metric. By graphically showing the computed weights in the data table (Fig. 1), analysts can reason about which features are more relevant for the specified pairs of nodes to be considered as similar to each other.

4.3 Visual Representation

The tasks and requirements described earlier guided our visualization design. We employ two iconic visual representations of clusters, adjacency matrices and treemaps, inspired by NodeTrix [24], and Dicon [5] respectively. The representations are switchable for revealing relational (**R3**, **R5**, **R6**, **R7**) and attribute (**R2**, **R4**) patterns of the nodes within the clusters. The main view is divided horizontally with the nodes and clusters of one set on top and the other on the bottom. They are connected by aggregated bipartite links (**R5**). We also provide a set of interactions for exploring and refining the co-clusters (**R6**, **R7**). The details of these designs are explained in this section.

4.3.1 Visualizing Node Clusters as Adjacency Matrices

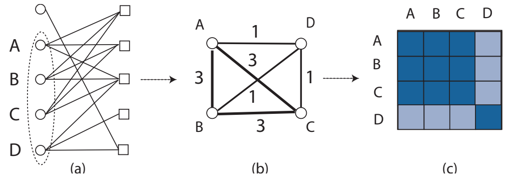


Fig. 3. Projection of the bipartite relation onto a group of nodes: (a) the bipartite graph with a cluster consists of node A, B, C and D; (b) a one-mode graph formed by projecting the bipartite relation on the cluster, the weight of the edges is the number of common neighbors of two nodes (i.e., *concordance*); (c) the adjacency matrix displays the weighted one-mode graph. Nodes A, B, and C have similar bipartite connections.

The process of visualizing node clusters as adjacency matrices consists of two major steps: (1) transforming the bipartite relations into weighted one-mode graphs through projection; (2) visualizing the projected graphs in the form of adjacency matrices.

Bipartite graph projection. For the nodes within a cluster in U, key questions are how to visually encode the consistency of their bipartite connections, and how to help analysts identify subsets of nodes with comparatively similar bipartite affiliations (**R7**). Here, we transform the bipartite relations into a weighted one-mode graph through *projection*, a common operation for bipartite graph analysis. As illustrated in Fig. 3(a) and (b), every two nodes that are affiliated with common neighbors are connected in the projection, and the edge weights reflect the resemblance of the bipartite relations for the pairs.

Either *concordance*, which is the number of common neighbors of two nodes (Fig. 3(b)), or Jaccard Index, or the Mahalanobis distance found by the aforementioned metric learning algorithm can be used to measure the similarity of the neighborhoods of two nodes. The metric learning procedure obtains weightings on U and V for computing the Mahalanobis distance between any two nodes. The distance obtained reflects the user's notion of similarities.

Using *concordance* shows the number of common connections, the viewer can easily read, for example, the number of bills that two senators both voted for, or the number of papers two researchers have co-authored. Using *Mahalanobis distance* shows the similarity of nodes pairs which are adjusted based on user's input.

Visualizing the projected one-mode graphs. The projected one-mode graphs are visually represented as adjacency matrices (Fig. 3(c)). In these matrices, the color intensities of the matrix cells are mapped to the edge weights in the projected graph. With a proper permutation of the rows and columns, blocks in the matrix with higher and more consistent intensity, which signify subsets with more coherent bipartite connections, can be identified. For example, Fig. 3(b) may show a cluster of four senators where the first three are more similar to each other than the fourth. In the prototype, a spectral node sequencing algorithm [27] is utilized. It obtains an ordering of the nodes based on the Fielder vector of the Laplacian matrix of the weighted one-mode graph.

Here, we leverage a matrix visualization design for two reasons. First, with the encoding scheme described above, the matrix view will facilitate the evaluation of cluster qualities (**R7**). A homogeneous color distribution among matrix cells indicates a cohesive cluster with high intra-group similarity, while a heterogeneous color distribution indicates low cluster quality. Second, the projected one-mode graphs are usually dense, and can be visualized with much less visual clutter with adjacency matrices compared to node-link diagrams [16].

In the matrix representation, a categorical or numerical attribute of the nodes can be encoded with a colored circle beside each row of the corresponding matrix, as shown in Fig. 1 (A.2). In this example, the party affiliation of the senators is displayed, illustrating its correlation with the votes the senators cast for bills (**R4**). Other node attributes are displayed in a data table (Fig. 1 (B)), showing additional details (**R2**).

4.3.2 Visualizing Node Clusters as Treemaps

The treemaps consume less space than adjacency matrices ($O(N)$ and $O(N^2)$), and display the distribution of attribute values more effectively (Fig. 1 A.4). In the treemaps, each rectangle corresponds to a node within the cluster. The color intensity or hue of the rectangles encode a selected categorical or numerical attribute of the nodes. The nodes(/rectangles) are organized with two levels of nesting: the children of the root correspond to all the possible categorical attribute values or intervals of numerical values for the nodes in the cluster, and the nodes are attached to the children by the respective attribute values. Such nesting structure results in a layout where the distribution of the attribute values can be easily identified (**R2**). The treemap display also helps analyze the correlation between the bipartite relation to the nodes' attributes – We can compare if nodes in different clusters have different attribute values more easily with treemaps (**R4**).

4.3.3 Visualizing Node Attributes

A TableLens [34] style visualization displays the details of each data record (node), illustrating their textual, categorical, or numerical attributes (**R2**). The attribute values are graphically displayed as color coded blocks or bars with varying lengths based on the data attribute types. The Table Lens supports a generalized table row selection scheme: when the user hovers over an entry in the table, the other rows with similar attribute values will be highlighted and they can be selected by simply clicking on the entry. All selections and highlights on the entries in the table lens are linked with the bipartite graph display.

4.3.4 Cluster Layout

In the main view of our system, as shown in Fig. 1 (A), we place the two types of node clusters produced by co-clustering algorithms in two horizontally separated regions. The layout of the node clusters works in two steps. First, an initial horizontal ordering of the node clusters on the two display regions is obtained with the barycentric heuristic [42]. Second, we adjust the layout to remove overlap of the clusters by applying a force-directed layout algorithm. In the force directed layout phase, the relative horizontal positions of the nodes are contrained to be the same as from the barycentric algorithm, therefore no more edge crossing should be introduced.

4.3.5 Visualizing Bipartite Connections

It is important to show the bipartite connections (**R1, R5**), but to do so on a one-to-one basis will introduce severe edge clutter. Hence, we decided to aggregate the edges for overview and show individual bipartite links through detail-on-demand interaction. The aggregations (bundles) of the bipartite connections are derived based on the node clusters.

The total number of individual edges between two clusters c_u and c_v is normalized by the maximum number ($|c_u| \times |c_v|$) of possible edges between the pair of node clusters. The weight of each bundle are then mapped to the width of the links connecting node clusters. Displaying aggregated edges reduces the large amount of crossings and the visual clutter that may result from drawing individual ones.

As the node clusters are vertically spreaded, the node clusters may overlap with the bundled edges and introduce ambiguities. To solve this issue, edge routing techniques can be applied here. Examples of edge routing techniques include [12] and [33]. These techniques can be incorporated into the current system to improve the readability of the edges.

4.4 User Interaction

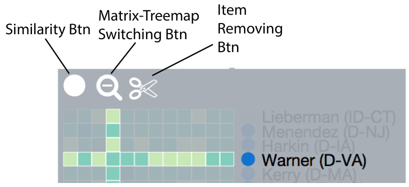


Fig. 4. The buttons affiliated with each cluster visualization that support cluster representation and refinement.

Based on the design requirements, we implemented the following interactions in the system:

Details-on-demand (R1). Besides analyzing the bipartite relations at the level of node clusters, we also design interactions for viewing the details of the interested nodes. For example, as the user hovers on a node in the matrix or treemap, its neighbors will be highlighted and the highlighted state persists after click.

Subspace selection (R6). The analyst can brush on a subset of nodes in one mode, which will be regarded as a subspace in which the similarities among the nodes and the corresponding subspace clusters in the other mode are derived. Here we consider two approaches for specifying a subset of nodes (i.e., the dimensions in the subspace) for analysis. In the first approach, the user directly selects related nodes from the adjacency matrices. In the second approach, the user specifies the value of an attribute and all the nodes with attribute value equal to that will be selected as a subspace. This can be done easily with the generalized selection scheme supported in the Table Lens display.

Cluster refinement (R7). The user can refine the clusters by directly manipulating the visual data items, and explicitly specify the grouping, or mark their confidence in the cluster quality to specify similarity constraints for the metric learning algorithm. Specifically, users can directly interact with the matrices or treemaps to split selected nodes from an existing cluster, or merge two node clusters into one. Two node clusters are merged when the user drags one matrix or treemap and drops it onto another. Users can split a subset of nodes from node clusters by first selecting the nodes to be separated and then click the scissor button shown in Fig. 4. Analysts can specify similarity constraints by identifying a group of nodes which are similar to each other via clicking on the similarity button shown on top of the corresponding node cluster visualization (Fig. 4).

4.5 Alternative Designs

Besides the designs presented above, alternative approaches exist for visually representing a bipartite graph and embedding node group information. One example is a biadjacency matrix with the rows and columns aggregated based on the node cluster they belong to or serialized with matrix reordering algorithms [32]. However, it is relatively difficult with this visualization to compare and access the similarity of the connections for a group of nodes, as the analyst needs to scan over all the rows/column and compare the entries, which can be mentally demanding. It is also possible to use a node-link diagram to represent the bipartite graph and place nodes with similar connections in close proximity, however, it is also hard to compare the connectivities of the nodes directly with it. In our system, we made the decision to visually represent the projected one-mode graph for a node cluster as an adjacency matrix to make these tasks easier. Another possibility is to visualize the entire one-mode projection with node-link diagrams or adjacency matrices, e.g., the co-author graph derived from a bipartite relation between authors and papers. However, the bipartite relation can be important in terms of interpreting how the links and clusters in the projected graphs are formed.

5 Sample Usage Scenarios

5.1 Scenario 1: US Senate Votes

We applied the proposed system to analyze the voting behaviours of legislators. We collected the roll-call voting records on the passage of bills and amendments in the U.S. Senate in 2012 from Govtrack.us [19]. These data include information about whether the senators voted ‘yea’ or ‘nay’ on the bills and amendments. There were 22 bills and 117 amendments voted on by 100 senators, among which 43 bills or amendments were passed or enacted. The bills are further categorized by their subject matter, including *economics and public finance*, *education*, and *armed forces and national security*. A bipartite connection is established if a senator votes for the passage of a bill, in total there are 6962 edges.

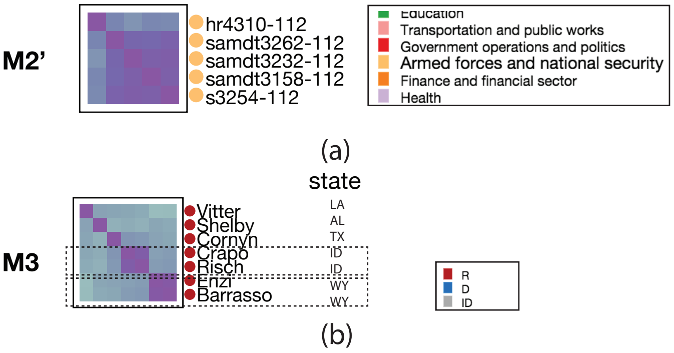
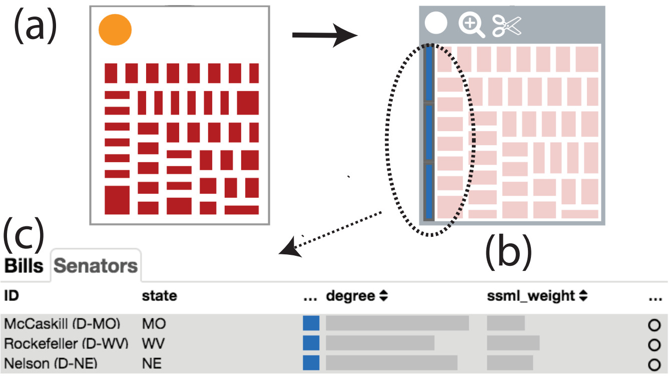


Fig. 5. Investigating the correlation between similarity of bipartite connections and node attributes: (a) a group of bills related to the subject *armed forces and national security* are supported by a similar set of senators, (b) republicans from the same state vote similarly.

Interpretation of co-clustering results. After running the co-clustering algorithm on the data and initializing the visualization (Fig. 1), we immediately observed that the clusters of senators (bottom) and bills (top) are reasonably cohesive. In this figure, the color intensity of the matrix cells encodes the measure of *concordance*, i.e., the number of common voters of two bills/amendments, and the number of bills/amendments supported by both senators. It can be observed that the color is relatively even over most of the matrices, indicating a consensus of opinion for each cluster of senators, and conversely, the bills within each cluster have a common set of supporters. A few senators and bills do not seem to align well with the computed clusters, which can be split out via the supported interactions.

The overall layout of the aggregated bipartite graph shows two distinctive groups of senator clusters (bottom) as well as bill clusters (top). Not surprisingly, the senators are generally grouped by their party affiliations, as indicated by the colored circles on the sides of the matrix showing the party affiliation of each senator, or the colors of the rectangles within the treemaps. In the layout, clusters of bills that lie off the center are those that have received party-line votes, i.e., a majority of a party vote for the bills in the same way in opposition to the other party. We can also identify some clusters of bills that lie in the middle, receiving votes from both parties. Amongst the clusters of senators there were some outliers with senators from both parties (Fig. 1). For example, the cluster **M1** in Fig. 1 was composed of one Republican (Scott Brown) and many Democrats. Hovering over the node highlights the bills he supported. We found that his voting preference aligns with the Democrats to some extent.



ID	state	...	degree ⬍	ssml_weight ⬍	...
McCaskill (D-MO)	MO	🟦			○
Rockefeller (D-WV)	WV	🟦			○
Nelson (D-NE)	NE	🟦			○

Fig. 6. A user marked all the Republicans as similar to each other (a), and the cluster of senators (b) found by the semi-supervised co-clustering algorithm, the clustering algorithm found Democrats with relatively neutral political standings (c).

In general, there was no clear correlation between the subject matter of the bills and the degree of overlap of their supporters. Though we can still identify in cluster **M2**, many of the bills were related to *armed forces and national security* (Fig. 1). The bills can be separated and form a more coherent cluster (**M2’** in Fig. 5(a)). The bipartite links showed that the cluster received support from most senators in both parties. In another cluster (**M5**) (Fig. 1) supported mostly by Democrats three bills were related to *crime and law enforcement*. Two of them were about the Foreign Intelligence Surveillance Act.

In cluster **M3** (Fig. 5 (b)), we found several pairs of Republicans align better with each other in their voting preferences than others in the cluster. In a further investigation of the reasons by brushing the nodes, and highlight them in the data tables, we found the Republicans were actually paired by their home states.

Semi-supervised co-cluster analysis. When we applied the semi-supervised co-clustering analysis based on metric learning in the above data, there were more interesting findings. Specifically, we grouped all the Republicans in the dataset into the same cluster (Fig. 6(a)) as a set of similarity constraint fed back into the proposed semi-supervised co-clustering algorithm and ran the algorithm again. The resulting cluster consisted of all the Republicans and, interestingly, three Democrats, *Claire McCaskill*, *Jay Rockefeller*, and *Ben Nelson*, as shown in Fig. 6(b). After searching on the related news reports and profiles of these senators, we found *Jay Rockefeller* is the only Democrat in the Rockefeller family dynasty (which was supported by Republicans); *Ben Nelson* is regarded as ’the least likely Democrat to stand with his party on legislative initiative’ [47]; *Claire McCaskill* also had a moderate political standing [35]. These findings demonstrate the power of the proposed algorithm.

Table 1. Guiding questions for domain expert interviews.

#	Aim	Question
Q1	Visual Design	Are the matrix/treemap representations of clusters informative to you? and Why?
Q2	Visual Design	Does the current layout of bipartite graphs provide a clear view of different types of nodes? and Why?
Q3	Interaction Design	Are the current interactions useful and easy to use? and Why?
Q4	Visual Analysis	Can you identify different patterns in the matrix/treemap based on different node similarity metrics?
Q5	Visual Analysis	Can you interpret the co-clustering results in contexts of different data?
Q6	Visual Analysis	Do you think the semi-supervised co-clustering is useful and can help better analysis the bipartite graph?
Q7	General	How do you like the system overall in exploring bipartite graphs and identifying insights?

5.2 Scenario 2: Publication Data

Our system has also been used for analyzing the publication records in DBLP, a bibliographic database for Computer Science. We collected the bipartite relation of the authors' affiliation with different conferences. In the analysis, we analyze the authors who have published in more than 12 conferences/journals. The filtered bipartite graph extracted from this dataset consists of 146 authors and 77 conferences/journals. In total there are 2010 edges.

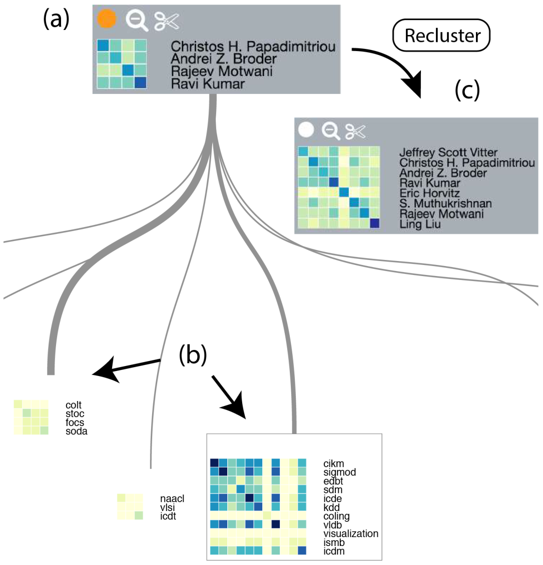


Fig. 7. A group of researchers (a) working on algorithms for massive data processing. They publish mostly in the theoretical computer science and data mining conferences (b) as shown by the thick bundles connecting the author cluster and the conference clusters (d). The user marked (a) as a meaningful cluster. The group expanded and new authors who work in similar area are included (c) after running the co-clustering again.

We applied the semi-supervised co-cluster analysis on this dataset. From the initial clustering result, we identified a group of researchers (Fig. 7(a)) working on algorithms for massive data processing and have publications mostly in the areas of theoretical computer science and data mining (Fig. 7(b)), as indicated by the thick bipartite link bundles connecting the cluster of authors to the two clusters of conferences (Fig. 7(d)). We mark the cluster of researchers as a meaningful one and after using this cluster as the similarity constraint in the semi-supervised co-clustering algorithm, we successfully found more researchers in these research areas who published in similar conferences, and the original group of authors is expanded (Fig. 7(c)).

6 Expert Review

To evaluate the system's usability and potential utility, we conducted interviews with expert users (denoted as E1 and E2) who were senior researchers in the area of machine learning and specialized in active learning techniques, and nonexpert users (denoted as E3 and E4) who are graduate students with domain knowledge in the area of social and political science. In each interview, we first demonstrated all the key features of the system to the expert in a tutorial and encouraged the person to use the system on their own based on a preloaded dataset. They were also encouraged to ask questions whenever they met problems. After they fully explored the tool's functionality, we conducted a semi-structured interview guided by a set of questions summarized in Table 1. The interviews took about one hour each. We deliberately asked the experts not to be constrained by the questions listed, but to use them as a guide and elaborate their thoughts when using the tool based on their domain knowledge. We recorded the interviews and took notes of the experts' comments during the interview.

Results. The experts were impressed by the system regardless of their domain expertise area. They believed the system was "efficient for exploring clusters [in bipartite graphs]". Experts with machine learning background also think the system is a "very useful tool for co-cluster analysis". We summarize their comments as follows.

Visual and interaction designs. The experts believed the visualization generates "informative" and "aesthetic" results and were impressed by the rich interactions. E2 particularly liked the idea of visualizing clusters in both treemaps and matrices and said: "this [design] is very intuitive ... and now I can find both relational patterns of cluster member and patterns of their feature [inside each cluster]". E3 liked the matrix design and said: "with this I can immediately see and compare the senators", although E3 felt the Treemap is less intuitive. E3 and E4 also commented that the connections between bills and senators are depicted in a clear way with the bundled links.

Visual analysis. Both E1 and E2 confirmed the novelty of the system in terms of its visual analysis functions. In particular, E1 said: "this is a very interesting idea and little work has been done for analyzing bipartite graphs in a semi-supervised approach". E2 also commented that "this interactive visualization is much better than those active learning systems [developed based on traditional UIs] that I know" and "Interactively editing on those clusters is very interesting [when compared with traditional data labeling procedures.]". E1 and E2 also believed that the semi-supervised co-clustering algorithm we proposed, although still with room to improve, was a promising idea and generated correct and meaningful results.

Discussion. In addition to the above positive feedback, the experts also identified some limitations of the current system. First, the system may not be scalable enough. However, they mentioned that the querying and filtering functions in the system were helpful for navigating through larger datasets. E1 and E2 also commented that all active learning techniques had a common goal of training the underlying analysis module based only on a small set of high quality samples that are precisely labeled by the analyst, instead of based on a large set of imprecise samples. Therefore, the scalability may not be a problem for the system in this case. The second issue they mentioned was that the visualization designs, although very informative, encoded large amounts of information which took effort to learn. But they both agreed that this issue was primarily due to the complexity of the analysis problem itself and indeed many data properties need to be represented in the visualization.

7 Conclusion

We introduce an interactive visual analysis system for analyzing clusters in a bipartite graph. We employ adjacency matrices and treemaps to visually encode the clusters. We illustrate the proposed techniques via two case studies on real world datasets. Our investigations identify many interesting findings that help to illustrate the usefulness of the system. In the future, we plan to apply the system to analyze bipartite graphs in other application domains, conduct user studies, and extend the functionality of the system for analyzing not only bipartite graphs, but also multimode graphs.

Acknowledgments

This work is partially supported by RGC GRF 618313, HKUST Oversea Research Grant, and HKUST Postdoctoral Matching Fund.

References

[1] S. Barkow, S. Bleuler, A. Preli, P. Zimmermann, and E. Zitzler. Bicat: a biclustering analysis toolbox. *Bioinformatics*, 22(10):1282–1283, 2006.

[2] A. Bellet, A. Habrard, and M. Sebban. A survey on metric learning for feature vectors and structured data. *arXiv preprint arXiv:1306.6709*, 2013.

[3] R. L. Breiger. The duality of persons and groups. *Social Forces*, 53(2):181–190, 1974.

[4] N. Cao, D. Gotz, J. Sun, Y.-R. Lin, and H. Qu. Solarmap: Multifaceted visual analytics for topic exploration. In *IEEE International Conference on Data Mining*, pages 101–110, 2011.

[5] N. Cao, D. Gotz, J. Sun, and H. Qu. Dicon: Interactive visual analysis of multidimensional clusters. *IEEE Transaction on Visualization and Computer Graphics*, 17(12):2581–2590, 2011.

[6] N. Cao, J. Sun, Y.-R. Lin, D. Gotz, S. Liu, and H. Qu. Facetatlas: Multifaceted visualization for rich text corpora. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1172 –1181, 2010.

[7] D. Chakrabarti, S. Papadimitriou, D. S. Modha, and C. Faloutsos. Fully automatic cross-associations. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 79–88, 2004.

[8] I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 269–274, 2001.

[9] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 89–98, 2003.

[10] M. Dörk, N. H. Riche, G. Ramos, and S. T. Dumais. Pivotpaths: Strolling through faceted information spaces. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2709–2718, 2012.

[11] C. Dunne, N. Henry Riche, B. Lee, R. Metoyer, and G. Robertson. Graphtrail: analyzing large multivariate, heterogeneous networks while supporting exploration history. In *Proceedings of the International Conference on Human Factors in Computing Systems*, pages 1663–1672, 2012.

[12] T. Dwyer and L. Nachmanson. Fast edge-routing for large graphs. In *Graph Drawing*, pages 147–158. Springer, 2010.

[13] J. Feng, X. He, B. Konte, C. Böhm, and C. Plant. Summarization-based mining bipartite graphs. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, pages 1249–1257, 2012.

[14] P. Fiaux, M. Sun, L. Bradel, C. North, N. Ramakrishnan, and A. Endert. Bixplorer: Visual analytics with biclusters. *Computer*, 46(8):90–94, 2013.

[15] S. Ghani, B. C. Kwon, S. Lee, J. S. Yi, and N. Elmqvist. Visual analytics for multimodal social network analysis: A design study with social scientists. *IEEE Transaction on Visualization and Computer Graphics*, 19(12):2032–2041, 2013.

[16] M. Ghoniem, J. Fekete, and P. Castagliola. A comparison of the readability of graphs using node-link and matrix-based representations. In *IEEE Symposium on Information Visualization*, pages 17–24, 2004.

[17] M. Ghoniem, J.-D. Fekete, and P. Castagliola. On the readability of graphs using node-link and matrix-based representations: A controlled experiment and statistical analysis. *Information Visualization*, 4(2):114–135, 2005.

[18] J. Goncalves, S. Madeira, and A. Oliveira. *BMC Research Notes*, 2(1):124, 2009.

[19] GovTrack.us. https://www.govtrack.us/, 2014.

[20] J. A. Hartigan. Direct Clustering of a Data Matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.

[21] J. Heer and A. Perer. Orion: A system for modeling, transformation and visualization of multidimensional heterogeneous networks. In *IEEE Conference on Visual Analytics Science and Technology*, pages 51–60, 2011.

[22] J. Heinrich, R. Seifert, M. Burch, and D. Weiskopf. Bicluster viewer: A visualization tool for analyzing gene expression data. In *Advances in Visual Computing*, volume 6938, pages 641–652. 2011.

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

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

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

[26] S. C. Hoi, W. Liu, and S.-F. Chang. Semi-supervised distance metric learning for collaborative image retrieval and clustering. *ACM Trans. Multimedia Comput. Commun. Appl.*, 6(3):18:1–18:26, Aug. 2010.

[27] M. Juvan and B. Mohar. Optimal linear labelings and eigenvalues of graphs. *Discrete Applied Mathematics*, 36(2):153 – 168, 1992.

[28] H. Kang, C. Plaisant, B. Lee, and B. B. Bederson. Netlens: iterative exploration of content-actor network data. *Information Visualization*, 6:18–31, 2007.

[29] D. A. Keim, J. Kohlhammer, G. Ellis, and F. Mansmann, editors. *Mastering The Information Age - Solving Problems with Visual Analytics*. Eurographics, 2010.

[30] Z. Liu, S. Navathe, and J. Stasko. Network-based visual analysis of tabular data. In *IEEE Conference on Visual Analytics Science and Technology*, pages 41–50, 2011.

[31] S. Madeira and A. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004.

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

[33] S. Pupyrev, L. Nachmanson, S. Bereg, and A. E. Holroyd. Edge routing with ordered bundles. In *Graph Drawing*, pages 136–147. Springer, 2012.

[34] R. Rao and S. K. Card. The table lens: merging graphical and symbolic representations in an interactive focus+ context visualization for tabular information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 318–322. ACM, 1994.

[35] D. Reese. Is sen. claire mccaskill a moderate? *The Washington Post*.

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

[37] R. Santamaría, R. Therón, and L. Quintales. Bicoverlapper: a tool for bicluster visualization. *Bioinformatics*, 24(9):1212–1213, 2008.

[38] H.-J. Schulz, M. John, A. Unger, and H. Schumann. Visual analysis of bipartite biological networks. In *Proceedings of the Eurographics Conference on Visual Computing for Biomedicine*, pages 135–142, 2008.

[39] Z. Shen, K.-L. Ma, and T. Eliassi-Rad. Visual Analysis of Large Heterogeneous Social Networks by Semantic and Structural Abstraction. *IEEE Transactions on Visualization and Computer Graphics*, 12(6):1427–39, 2006.

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

[41] J. Stasko, C. Görg, and Z. Liu. Jigsaw: Supporting investigative analysis through interactive visualization. *Information Visualization*, 7(2):118–132, 2008.

[42] K. Sugiyama, S. Tagawa, and M. Toda. Methods for visual understanding of hierarchical system structures. *IEEE Transactions on Systems, Man and Cybernetics*, 11(2):109–125, 1981.

[43] M. Sun, C. North, and N. Ramakrishnan. A five-level design framework for bicluster visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):1713–1722, 2014.

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

[45] T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. van Wijk, J.-D. Fekete, and D. Fellner. Visual analysis of large graphs: State-of-the-art and future research challenges. *Computer Graphics Forum*, 30(6):1719–1749, 2011.

[46] S. Wasserman and K. Faust. *Social Network Analysis: Methods and Applications*. Structural Analysis in the Social Sciences. Cambridge University Press, 1994.

[47] N. Wing. Ben nelson is senate democrat most likely to vote against his party: Analysis. *The Huffington Post*.