SightBi: Exploring Cross-View Data Relationships with Biclusters

Maoyuan Sun, Abdul Rahman Shaikh, Hamed Alhoori, Jian Zhao

Fig. 1. SightBi shows computed cross-view data relationships in stand-alone relationship-views and supports users in flexibly organizing the layout of multiple views. As a user drags a relationship-view, its connected views automatically move with it. This enables a user to move a set of views out of a cluttered layout and helps preserve a meaningful spatialization previously created by the user.

Abstract—Multiple-view visualization (MV) has been heavily used in visual analysis tools for sensemaking of data in various domains (e.g., bioinformatics, cybersecurity and text analytics). One common task of visual analysis with multiple views is to relate data across different views. For example, to identify threats, an intelligence analyst needs to link people from a social network graph with locations on a crime-map, and then search for and read relevant documents. Currently, exploring cross-view data relationships heavily relies on view-coordination techniques (e.g., brushing and linking), which may require significant user effort on many trial-and-error attempts, such as repetitiously selecting elements in one view, and then observing and following elements highlighted in other views. To address this, we present SightBi, a visual analytics approach for supporting cross-view data relationship explorations. We discuss the design rationale of SightBi in detail, with identified user tasks regarding the use of cross-view data relationships. SightBi formalizes cross-view data relationships as biclusters, computes them from a dataset, and uses a bi-context design that highlights creating stand-alone relationship-views. This helps preserve existing views and offers an overview of cross-view data relationships to guide user exploration. Moreover, SightBi allows users to interactively manage the layout of multiple views by using newly created relationship-views. With a usage scenario, we demonstrate the usefulness of SightBi for sensemaking of cross-view data relationships.

Index Terms—Cross-view data relationship, multi-view visualization, bicluster, visual analytics

1 Introduction

Multiple-view visualization (MV) has been heavily used for sensemaking of data. Similar to a multi-focus approach [70], MV highlights that each view supports certain analysis tasks by showing data in a specific type of visualization. Different views either show different parts of data or display the same data from different perspectives as different visualizations. MV has been incorporated in visual analysis tools in various fields, such as Caleydo [34] for biomolecular data analysis, Canopy [14] for multimedia analysis, IN-SPIRE [2] and Jigsaw [49] for text analytics, and Tableau [4] and Spotfire [5] for business intelligence.

To gain a comprehensive understanding of data, analysts often need to relate data from multiple views. For example, as is shown in Figure 2, in a text analytics scenario, an analyst, Sarah, explores intelligence reports to identify threats by using Jigsaw [49]. After loading the dataset, Sarah works on three views offered by Jigsaw: a list view showing connections between entities, a document view displaying text, and a graph view presenting links between entities and documents. Sarah tries to relate their information by exploring entity relations, following connections between entities and documents, and reading relevant text. With Jigsaw's view coordination functions, when Sarah clicks on an entity in the list view, corresponding entities and documents are highlighted in other views. Sarah needs to repetitiously click entities in one view and follow highlighted ones in other views to explore various possible combinations of related data across the three views. As the number of clicked entities grows, Sarah soon gets confused about which selected entities in the list view connect to which pieces of highlighted text in the document view (e.g., are they all related or are there only partial connections between some of them).

Exploring *cross-view data relationships* is not as simple as it looks like. Currently, MV-based visual analysis tools heavily rely on view-coordination techniques [11, 39, 41, 43, 63] for users to relate data from different views. Since these techniques offer limited visual guidance, users have to put significant amount of effort into many trial-and-error attempts (e.g., trying to select different entities in a view, following and checking highlighted parts in other views). This potentially forces users to manually solve a combinatorial problem: discovering sets of entities in each view and requiring that entities in such sets from different views are related (e.g., finding sets of nodes in a social network graph that are related to sets of locations on a map and sets of organizations in a list).

Computation can help save human effort in finding connected sets of entities between different domains (e.g., person and location). Specifically, *biclustering* [36] has been developed to simultaneously cluster

- *Maoyuan Sun (corresponding author), Abdul Rahman Shaikh, and Hamed Alhoori are with Northern Illinois University. E-mail: {smaoyuan, ashaikh2, alhoori}@niu.edu.*
- *Jian Zhao is with University of Waterloo. E-mail: jianzhao@uwaterloo.ca.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

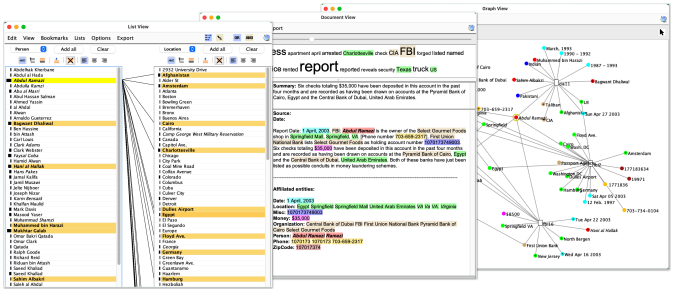

Fig. 2. An example of relating data from three views in Jigsaw: a list view (left), a document view (middle), and a graph view (right).

two related sets of entities into related subsets. Such a related pair of subsets is a *bicluster*, where entities in one set are related to those in another. Based on shared entities, biclusters can be linked together to form *bicluster-chains*. Figure 3 shows examples of biclusters and bicluster-chains. With them, we can compute data relationships between a pair of views or among several views. While using biclusters helps free users from a time-consuming trial-and-error process, it brings challenging problems. How can we apply biclustering to MV, especially when the number of views goes beyond two? How can we visualize biclusters in MV to support users in exploring cross-view data relationships, but not affecting much of existing views?

To address such challenges, we propose SightBi, a novel visual analytics technique for exploring cross-view data relationships. With a relational data model, SightBi computes data relationships between pairs of views as biclusters, and shows them in newly created relationship-views. Moreover, SightBi allows users to steer cross-view data relationships computing by interactively including or excluding views, which drives the process of chaining multiple biclusters together. In summary, this work highlights the following three contributions.

1) We formalize cross-view data relationships as biclusters and present a data model to support this. It enables computing data relationships between pairs of views and supports flexibly including or excluding views for cross-view data relationship exploration.

2) We propose a *bi-context* design concept to show computed cross-view data relationships. It highlights separating cross-view data relationships from existing views by creating stand-alone views for computed relationships. The newly added relationship-views help preserve existing views and can serve as overviews to guide user exploration.

3) We develop a visualization prototype, SightBi. It implements the proposed concept and supports users in interactively exploring data relationships across multiple views. We demonstrate the usefulness of SightBi with an investigative analytics scenario.

2 Related Work

2.1 Relating Information from Multiple Views

Three major designs support relating data across multiple views: 1) *linkage*, 2) *coordination*, and 3) *proximity*.

Linkage is a straightforward approach that maps data relations across multiple views to *visible links*. It offers the most explicit way of showing cross-view data relationships. By following a visible link, users can see a cross-view relationship and investigate its involved components. This design concept has been applied to show connections between visual elements in different views, such as Bixplorer [52], ConnectedCharts [60], GPLOM [31], Flowstrates [13], ImAxes [19], MyBrush [33], PivotSlice [71], Semantic Substrates [47], VisLink [16], and variant versions of VisLink [24, 61, 62]. As each visible link associates two components, it can only reveal a one-to-one relationship. When the number of relationships is large, this design may cause visual clutter (e.g., many links crossing each other). For complex relationships (e.g., many-to-many relationships), such visible links cannot directly reveal them, so users have to manually trace linked elements, identify shared ones, and then group some links together.

Coordination is a relatively implicit strategy to show cross-view data relationships. It highlights a *dynamic updating* approach supported by necessary user interactions (e.g., brushing). Cross-view relationships will not be revealed until users interact with some visual elements. User interactions trigger some changes in visual encodings in multiple views. Users need to refer to such changes to understand relationships. Coordination has been heavily explored and used in visual analysis tools (e.g., Cross-filtered Views [65], CViews [12], Improvise [64], SnapTogether [40], and Spotfire [5]). However, coordination suffers from two problems. First, it requires users to pay enough attention to changes in multiple views corresponding to their interactions. Users may not realize cross-view relationships if they fail to recognize changes after their interactions. Second, detailed connections can hardly be revealed by coordination. For example, when users brush five nodes in a scatterplot, seven bars get highlighted in a histogram. Users can learn that these five nodes and the seven bars are related, but whether or not each node is related to all the bars cannot be clearly answered.

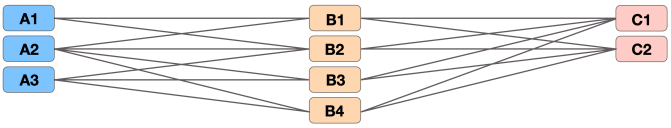

Fig. 3. Three biclusters: 1) $\{A1, A2\} - \{B1, B2\}$, 2) $\{A2, A3\} - \{B2, B3, B4\}$, and 3) $\{B1, B2, B3, B4\} - \{C1, C2\}$. 1) and 2) share $A2$ and $B2$. 1) and 3) share $B1$ and $B2$. 2) and 3) share $B2, B3$ and $B4$. With shared entities, there are two bicluster-chains, composed of 1) and 3), and 2) and 3).

Proximity focuses on an *arrangement oriented* approach. It highlights spatially organizing relationship components. Proximity is less explicit than the other two. Users cannot directly see related visual elements across multiple views. Instead, they have to understand organizations of relationship components to explore cross-view relationships. Proximity can be implemented in a spatial organization that uses relative distances in a 2D space to indicate potential relationships between views. For example, users place documents A and B near each other to indicate that they have similar topics. It has been used to support sensemaking [42] (e.g., Analyst's Workstation [10], Bixplorer [23], Co-Cluster Analysis [67], ForceSPIRE [21] and NodeTrix [28]). Proximity neither suffers from visual clutter issues caused by visible links, nor relies on users perceiving changes to recognize relationships. Yet, to understand cross-view relationships, proximity requires users to create, investigate or reason about spatialization, which needs much cognitive effort. Moreover, it works for small sets of views in a 2D space. For a large number of views, proximity may not work due to overlap.

2.2 Bicluster and Bicluster-Chain

Biclusters are results from biclustering algorithms [36], which simultaneously compute subsets of entities and subsets of conditions where the set of entities behave similarly under the set of conditions. It has been applied to solve real-world problems, such as discovering co-behaved genes in bioinformatics [36], identifying bundled shopping items in marketing analysis [59], finding colluding threats in intelligence analysis [66], and detecting useful missing edges for network analysis [73,74]. From a graph perspective, biclusters are bicliques (i.e., complete bipartite graphs), where each vertex in one set is linked to all vertices in another set (see Figure 3). In data mining, algorithms (e.g., LCM [59] and CHARM [68]) are typically designed for computing *closed biclusters*, which are maximal bicliques.

Biclusters can overlap by sharing entities [55]. Based on shared entities, biclusters, consisting of entities from multiple domains, can be linked to form *bicluster-chains* [56]. Figure 3 shows two bicluster-chains: $\{A1, A2\} - \{B1, B2\} - \{C1, C2\}$ and $\{A2, A3\} - \{B2, B3, B4\} - \{C1, C2\}$. The former is based on shared entities, $B1$ and $B2$. The latter is based on shared entities, $B2, B3$ and $B4$. Different biclusters can share different entities, so it is possible to use the number of shared entities as the level of overlap for finding bicluster-chains [72].

2.3 Bicluster Visualizations

To make computed biclusters usable, bicluster visualizations have been studied. Sun et al. [56] proposed a design framework for bicluster visualizations. It considers designs for five relationship-levels: *entity*-level,

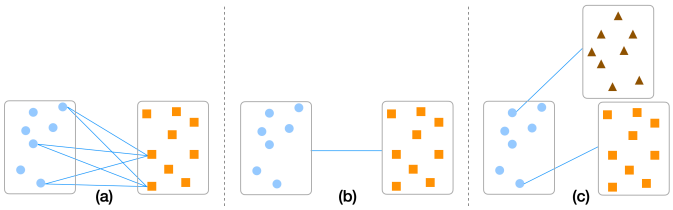

Fig. 4. Three types of cross-view relationships: (a) between visual elements, (b) between views and (c) between visual elements and views.

group-level, *bicluster*-level, *chain*-level, and *schema*-level. Entity-level refers to a one-to-one relationship, and group-level is a one-to-many relationship. Bicluster-level and chain-level are many-to-many relationships that involve two or more sets of entities from different domains (e.g., person, location and date). Schema-level regards the relational schema of a database.

A key design trade-off of visualizing biclusters has been identified: *preserving the entity uniqueness* vs. *maintaining the cluster completeness* [51]. The former requires that visualizing biclusters without duplicating entities. The latter highlights that placing entities of the same biclusters neighboring each other. Due to overlap, it is impossible to achieve both for certain visual layouts (e.g., lists and matrices). Fundamentally, this is an Euler diagram problem for visualizing multiple sets [8]. Based on the trade-off, there are three designs for visualizing biclusters: *entity*-centric, *relationship*-centric, and *cluster*-centric.

An entity-centric design focuses on preserving the uniqueness of entities, with a hypothesis that entity duplication may confuse users [55]. It has been applied to a node-link diagram based layout (e.g., multiple lists or node-link graphs). In a multi-list based layout, biclusters are perceived by tracing edges (e.g., finding bicliques in Jigsaw's List view [49]). In a node-link graph, biclusters can be revealed with set visualization techniques (e.g., Bubble Sets [17], LineSets [7] and Kelpfusion [37]) by creating an additional layer on top of a graph. Users need to trace edges or follow added set-layers to identify a bicluster, because entities spread out over node-link diagrams.

A relationship-centric design emphasizes the relationship between entities. It aims to place entities of the same biclusters near each other. It often picks a matrix as the basic layout to visualize biclusters (e.g., Bicluster viewer [27], BiVoc [26], and BicOverlapper [44]), where rows are entities in one set and columns are another set (e.g., a *person-location* relationship matrix). To show biclusters, it requires users to order rows and columns, so that entities of the same biclusters can be near each other. This generates sub-matrices inside a relationship matrix to show biclusters. Due to overlaps, one order can not show all biclusters. To see all biclusters, users need to change the matrix order, or some rows or columns in the matrix have to be duplicated [50].

A cluster-centric design encodes each bicluster with its own visual mark. To achieve this, entities shared by multiple biclusters are duplicated. It has been used in hybrid visualizations that replace nodes in a node-link diagram with some charts. For example, in Bixplorer [23] and Furby [50], each bicluster is shown as a matrix. If two biclusters share entities, two matrices are linked with edges. BiDot [72] uses bipartite graphs as its basic layout and replaces nodes in the graph with two sets of aligned dots to show biclusters. For each bicluster, the identified trade-off is balanced, as entities are gathered without duplication. However, when the number is above one, the more biclusters overlap, the more entities are duplicated. To address this, BiSet [55] encodes each bicluster as an edge bundle in a multi-list layout and supports users in interactively ordering entities [57] and merging biclusters [54] in lists. It uses marks for biclusters out of entity-lists, which preserves existing entity-lists. This separation inspired the design of the relationship-view in SightBi, because we aim to maintain existing views.

3 Definition and Specification

3.1 Visual Element

By following the definition of *mark* and *channel* discussed by Munzer [38], we denote a visual element as *a graphical representation unit* that encodes data. The appearance of this unit can be controlled based on certain data attributes. We consider that visual elements serve a key role in transforming data from their raw format into a human understandable format. This emphasizes a mapping from data to graphical representation units. For example, nodes in a scatterplot correspond to data entities and the positions of nodes reveal values of two attributes. Based on this data-mapping oriented notion, we do not consider user interface widgets (e.g., button, slider, and checkbox) as visual elements, although they are commonly used in visualizations.

Table 1. Four levels of cross-view data relationships.

Level of Relationships	Number of Views	Number of Visual Elements
Individual Level $(1:1)$	2	2
Group Level $(1:i)$	2	$1+i$ $(i \geq 2)$
Bi-group Level $(i:j)$	2	$i+j$ $(i,j \geq 2)$
Multi-group Level $(i:j:\ldots:z)$	at least 3	$i+j+\ldots+z$ $(i,j,\ldots,z \geq 2)$

3.2 View and Multiple Views

We define a view as a set of visual elements that are spatially organized in a perceivable visual-boundary to support specific analysis tasks. A view can be a window of a desktop application (e.g., Jigsaw's List View [49]), a card in a dashboard visualization, a chart of a bounded area in a web application (e.g., a chart with visible borders in MyBrush [33]), a component of a meta-visualization (e.g., a block in Domino [25]), or a chart of small multiples [58]. For some complex cases (e.g., a composite visualization [32] in which one visualization is overlaid on top of another or embedded in another, like Interver [53]), we consider them one view, because they are intertwined with each other. We treat multiple views as a technique that uses more than one view for data analysis, and these views are organized in a display space with certain layout strategies [15].

3.3 Cross-View Data Relationship

In terms of relationship components, there are three types of cross-view data relationships (Figure 4). The first type highlights *relationships between visual elements* from different views (e.g., three persons in a social network are related to four locations on a map). The second type emphasizes *relationships between views*. It may indicate high-level insights (e.g., placing documents near each other indicates they are relevant in ForceSPIRE [21]). The third type refers to *relationships between visual elements and views*. It can be considered following Shneiderman's visual information seeking mantra [46]. For example, a line chart shows an overview of car sales and a bar chart displays detailed data (e.g., sales at different dealers) for a line in the overview. In this work, we focus on the first type. Compared to others, it requires users exploring detailed connections between visual elements. Such low-level connections lay a necessary foundation for gaining high-level insights (e.g., matching between keywords leads to an understanding of relevant documents). Based on the five levels of relationships discussed in [56], there are four types of visual-element oriented cross-view data relationships. Table 1 gives a summary of them, where $i, j, \ldots, z$ denotes the number of visual elements in each view.

4 Designing SightBi

4.1 High-Level User Tasks

We have analyzed user tasks when using MV, particularly focusing on the role of cross-view data relationships. We performed the analysis based on the dataset used in a prior study on composition and configuration patterns in MV [15]. It includes papers on MV in IEEE VIS, EuroVis, and PacificVis from 2011 to 2019. For each paper in this dataset, we analyzed how multiple views were used and how cross-view data relationships supported such usage. We have identified three types of user tasks:1) *filtering*-oriented tasks, 2) *refocusing*-oriented tasks, and 3) *connecting*-oriented tasks. They are supported by three different roles of cross-view data relationships, as summarized in Table 2.

Filtering-oriented tasks refer to users selecting elements in one view, which filters elements in other views. A typical example includes Cross-filtered views [65] and cross-filtering based dashboard visualizations [45], in which each view can be used to filter data on others.

Table 2. Identified user tasks when using cross-view data relationships.

High-Level User Tasks	The Role of Cross-view Data Relationships
Filtering-oriented	Enabling cross-view data filtering
Refocusing-oriented	Offering one-to-one data mapping
Connecting-oriented	Serving analytical solutions

For such tasks, the view that users interact with serves a role similar to a control panel that enables users to perform dynamic queries [6] on other views. It implies a hierarchy between views (e.g., one controls others). Filtering-oriented tasks are not a focus of our design, as the key information that users care about is filtered data in other views. Thus, cross-view data relationships only provides a way that enables users to filter data, but the relationships are not a key analytical focus.

Refocusing-oriented tasks are users selecting data in one view and based on the selection, users trying to explore the same data in other views. As a simple example, a scatter plot matrix supports this type of task. In a scatterplot matrix, each scatter plot presents the same set of data points with different attributes. When users select some data points in one scatter plot, the same data points in other scatter plots are highlighted. For refocusing-oriented tasks, cross-view data relationships offer a one-to-one data mapping that supports users in shifting their focus from one view to others. Thus, key information that matters to users is how selected data in one view is presented in other views, rather than the relationship across views.

Connecting-oriented tasks are users exploring and identifying connections between data displayed in multiple views. Supporting this type of task is the key focus of our design for two reasons. First, cross-view data relationships are critical for this type of task. The other two tasks care about views to be investigated next by using cross-view data relationships for making transitions from working on one view to others. However, connecting-oriented tasks emphasize data relationships between views. Second, as this type of tasks handles connections between data from different views and such connections may involve various combinations, it is more exploratory in nature than the other two tasks. Solving a combinatorial problem manually is challenging and needs computational support (e.g., finding possible combinations, and showing them in a way that humans can understand).

4.2 Design Trade-off

There is a key design trade-off in showing cross-view data relationships: *view-embedding* (*intertwined with* existing views) versus *view-separating* (*separated from* existing views), particularly when using new marks in addition to existing visual element marks for the relationship. A comparison between them is summarized in Table 3. Figure 5 shows an example in which there are two bi-group level cross-view data relationships consisting of visual elements from two views: a scatterplot and a line chart. Specifically, one is among five nodes and two line segments, and the other is among three nodes and three line segments. They are revealed with marks indicating relationships either on top of the two views (Figure 5 (a)) or outside of the two views (Figure 5 (b)).

View-embedding designs highlight encoding relationships inside existing views, such as using the BubbleSets [17] technique to enclose visual elements of the same relationship. A benefit of this is that a relationship is shown in the context of its involved visual elements, and it does not take extra space outside of existing views. However, such embeddings force displayed relationships intertwined with existing visual elements, which can impact human perception of them. For example, due to many relationship-marks overlaying each other, user attention to existing views may be directed towards the highly overlapped ones (as visually they look salient), whereas users may focus on other visual elements when no such view-embedding is present.

View-separating designs call for displaying relationships outside of existing views by using some empty display space, which can be considered creating a view for relationships. A key benefit of this is that user perception of existing views is not disturbed. It allows users to find relationships in some specific areas in the display, which can facilitate the search for relationships. However, to achieve such a separation, extra display space is necessary to hold the relationship-marks. This requires that all existing views do not take up all the available display space. Moreover, with this separation, users need to switch their focus between existing views and the area where relationship-marks reside to check the detail of relationships. Such a context-switching between visual elements and relationships entails extra cognitive effort [18].

Table 3. A summary of key trade-offs between two designs.

	View-Embedding	View-Separating
Relationship-marks placed	Inside existing views	Outside existing views
Empty space among views	Unnecessary	Necessary
Perception of existing views	Disturbed	Preserved
Get a relationship-overview	Hard	Easy
Check relationship detail	Less effort for context switching	More effort for context switching

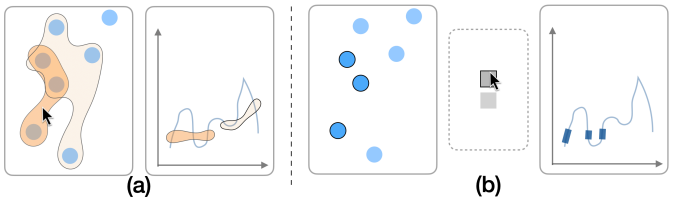

Fig. 5. A trade-off in visualizing cross-view data relationships: (a) *intertwined* with existing views, and (b) *separated* from existing views.

4.3 Design Considerations

Based on the analysis discussed in previous sections and the knowledge tasks presented by Amar and Stasko in [9], we have identified four major considerations that drive the design of SightBi.

C1: Designing relationship encodings. Effective encodings for relationships are critical, which should fulfill four goals, discussed below. They correspond to the knowledge task, concretizing relationships [9], which calls for clearly showing detailed components of relationships.

(a) Relationship encodings should be perceptually discriminated from the encodings of visual marks in existing views, so that users can easily recognize relationships. (b) Relationship encodings should be designed to avoid disturbing user perception of existing views. This helps preserve existing views. (c) Relationship encodings should reveal visual hints about visual elements that belong to the relationship to guide user exploration. This helps users check detailed information of a relationship. (d) Visual encodings for relationships should reveal changes when the state of a relationship is updated (e.g., selected vs. unselected). (a)-(c) calls for a balanced-solution regarding the design trade-off discussed in Section 4.2.

C2: Supporting six ways of explorations. For connecting-oriented tasks, there are six possible ways of user explorations, as summarized in Table 4. We aim to support all to enable flexible user explorations. User explorations can be considered in two dimensions. One cares about user explorations driven by *entity* (e.g., visual elements in existing views) or *relationship*. The other highlights whether user explorations remain in the *same view* or *switch views*. Cross-view data relationships involves visual elements (as entities) and connections between entities (as relationships). Users can start explorations with either of them. This leads to four different explorations: from entity to entity, from entity to relationship, from relationship to entity, and from relationship to relationship. As entities are located in existing views and relationships can be shown outside of existing views (in some empty display area, which may form views for relationships), as discussed in Section 4.2, these four explorations can be further categorized into two major groups: in the same view (T1 and T4) or switching views. Regarding switching views, there are two possible cases: *switching to existing views* (T2 and T5) and *switching to relationship-views* (T3 and T6).

C3: Organizing relationships and views. Relationship should be visually organized. This can ease the search process for finding useful relationships. Moreover, an organized layout may serve as an overview of relationships, which can better guide user explorations than a layout with relationships randomly placed. Furthermore, users may want to flexibly organize multiple views in a personalized, meaningful way for sensemaking (e.g., using spatializations) [10].

C4: Referring to original data on demand. To help users check and understand computed cross-view data relationships, showing orig-

Table 4. A summary of six ways of user explorations with specific tasks.

	In the Same View	Switching Views	
		Switching to Existing Views	Switching to Relationship-Views
Entity-Driven Exploration	T1: Given a set of visual elements in an existing view, find related visual elements in the same view. (*from entity to entity*)	T2: Given a set of visual elements in an existing view, find related visual elements in other existing views. (*from entity to entity*)	T3: Given a set of visual elements in an existing view, find related relationships in relationship-views. (*from entity to relationship*)
Relationship-Driven Exploration	T4: Given a set of relationships in a relationship-view, find related relationships in the same view. (*from relationship to relationship*)	T5: Given a set of relationships in a relationship-view, find related entities in existing views. (*from relationship to entity*)	T6: Given a set of relationships in a relationship-view, find related relationships in other relationship-views. (*from relationship to relationship*)

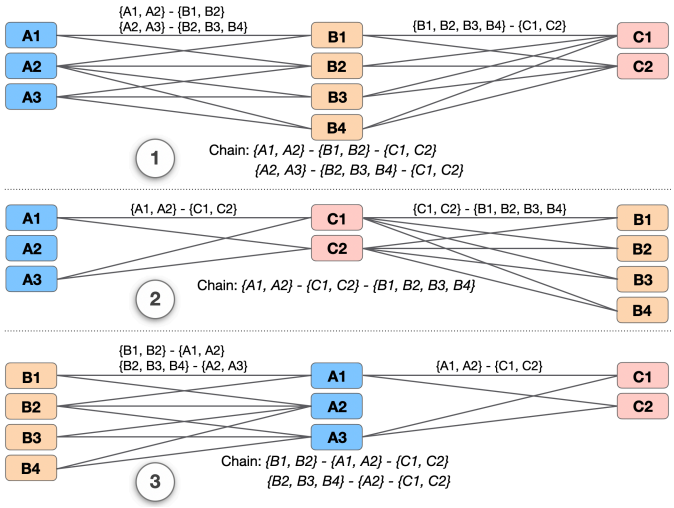

Fig. 6. For the same sets of entities, different combinations of biclusters by permuting the order of the three sets lead to different bicluster-chains.

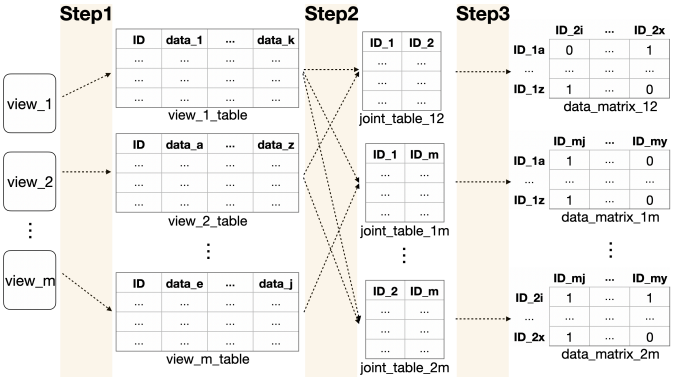

Fig. 7. Step1: reverse mapping from visual elements to a table for each view. Step2: creating a joint table for each pair of tables from previous step. Step3: transforming each joint table into a data matrix.

inal data is necessary. Users need to refer to original data (e.g., documents) to learn detailed context, which helps them verify computed relationships that may imply certain hypotheses. This corresponds to the knowledge task, confirming hypothesis, discussed in [9]. Moreover, original data should be retrieved on demand (e.g., requested based on some relationships, instead of simply showing all data).

5 SightBi Technique

The SightBi technique comprises three parts: a) relationship computation, b) relationship visualization based on a *bi-context* design, and c) interactive relationship management.

5.1 Cross-View Data Relationship Computation

We focus on supporting connecting-oriented tasks, as discussed in Section 4.1, which require users to explore sets of related visual elements from different views. As biclustering is designed for mining related subsets of entities from two entity-sets, we can formalize cross-view data relationships as biclusters. However, how to compute such biclusters remains a problem.

5.1.1 Challenges for Computation

There are two challenges in computing cross-view data relationships. First, since we use biclusters to formalize cross-view data relationships, to enable computing biclusters consisting of visual elements from different views, we need support from a certain *data model*. Second, for multi-group level of relationships, while they can be further formalized as *bicluster-chains* (as more views are involved), computing them is not simple. Given a set of views, there are multiple ways of chaining biclusters, as the sequence of views can vary (see Figure 6). For the same three sets of entities, using different sequences to connect them leads to different bicluster-chains, and entities in one bicluster-chain can be subsets of those in another (Figure 6 ① and ②).

5.1.2 Data Model for Computing Biclusters

To address the challenges, SightBi uses a relational data model to compute biclusters between visual elements from a pair of views. With this model, we can get biclusters that reveal cross-view data relationships. It lays the foundation to fulfill the four design considerations. Specifically, this model can be created with three key steps, shown in Figure 7.

S1: We tabulate visual elements by generating a table for each existing view. In this table, each visual element has a unique ID, and the data that this visual element encodes are associated with the ID. This step highlights a reverse mapping between data and marks in each view, and further organizes them in a table.

S2: For all pairs of views, based on related data encoded by visual elements, we create a joint table that associates the IDs of visual elements from one view with those from another. The way that determines how data is related can be different for different scenarios. For example, in text analytics, two people and three locations are related based on word co-occurrence; whereas in cyber security, five MAC addresses are related to four URLs determined by network traffic patterns [69].

S3: We transform each joint table into a data matrix, where rows and columns are visual element IDs from two views. For a cell with a corresponding row and column ID pair in the joint table, we assign a value of 1, otherwise 0. For each data matrix generated in this step, we can apply biclustering algorithms (e.g., LCM [59] and CHARM [68]) to it. Such computed biclusters can reveal cross-view data relationships, particularly between a pair of views.

5.1.3 Bicluster Chain Computation

As discussed in Section 2.2, we can form bicluster-chains based on shared entities. This allows us to expand cross-view data relationships to cover visual elements from more than two views (e.g., a set of nodes from a network graph, a set of line segments from a line chart, and a set of locations on a map). A key benefit of this approach is that it starts with a pair of views and supports users in flexibly including more views to expand explorations of relationships as their analysis proceeds. This dynamic expansion further enables users to interactively organize computed relationships. To address the second challenge of various ways of chaining biclusters, SightBi uses a four-step approach to get connected sets of visual elements from each involved view:

S1: *Computing biclusters for all pairs of views.* For a given set of views (e.g., $\{A, B, C\}$), we get all pairs of them (e.g., $\{AB, BC, AC\}$), and for each pair, we compute biclusters composed of visual elements from this pair of views.

S2: *Getting sequences for a given set of views.* For the same given set of views in the previous step, we compute their permutation without reverse duplicates (e.g., $\{ABC, ACB, BAC\}$). We do not include reverse

Algorithm 1: Cleaning obtained chains to get selected ones

Input : *allBicChains*, a set of all chains $\{c_i, i = 1 \dots n\}$
entChainDict, a dictionary with elements for each chain
Output : *selBicChains*, a set of selected chains
for $i \leftarrow 1 \dots n$ **do**
  $entSet_i \leftarrow entChainDict(c_i)$;
  for $j \leftarrow i+1 \dots n$ **do**
    $entSet_j \leftarrow entChainDict(c_j)$;
    if $entSet_i \not\subseteq entSet_j$ and $entSet_j \not\subseteq entSet_i$ **then**
      selBicChains.add(c_i);
return *selBicChains*;

duplicates, because bicluster-chains for a sequence (e.g., *ABC*) and its reverse duplicate (e.g., *CBA*) are formed by the same two sets of biclusters (e.g., biclusters computed based on *AB* and *BA* are the same).

S3: *Building bicluster-chains.* Based on the sequences from S2, we separate each sequence of the views (e.g., *ABC*) into consecutive, neighboring pairs (e.g., *AB* and *BC*). For each neighboring pair, we find computed biclusters from S1. Following the sequence of neighboring pairs, we chain corresponding biclusters together based on their shared entities in the same view (e.g., view *B*). Specifically, we use equation (1) to compute the matching between two biclusters that share visual elements in one view.

Given two biclusters consisting of visual elements from view pairs *AB* and *BC*, $bic_{AB} = \{\{a_i\},\{b_j\}\}$ and $bic_{BC} = \{\{b_m\},\{c_k\}\}$, where $a_i \in A$, $b_j, b_m \in B$, and $c_k \in C$, and $i, j, m, k \geq 1$, the matching between them is computed as follows ($|\cdot|$ denotes the cardinality of a set):

$$matching(bic_{AB},\ bic_{BC}) = \frac{|b_j \cap b_m|}{|b_j \cup b_m|} \quad (1)$$

Based on the computed matching score, we determine whether two biclusters are chained or not (e.g., chaining them if the matching score between them is above a threshold). With this step, we get all possible bicluster-chains for the given set of views.

S4: *Getting selected bicluster-chains.* Based on the set of bicluster-chains from S3, for each chain, we check two cases to decide whether to keep it as a selected one. One case is whether this chain is a subset of any other chain(s) in the set. The other is whether any other chain(s) in the set is a subset of this chain. If neither is true, we consider this chain a selected one. Algorithm 1 shows how to check whether a chain is a subset of another. We remove the chain that is a subset of another, since it does not fully cover related visual elements from the given views. This step ensures that there is no inclusion between bicluster-chains.

5.2 Cross-View Relationship Visualization

SightBi uses a *bi-context* design. It separates visual marks that encode computed cross-view data relationships from existing views (**C3**).

5.2.1 Bi-Context Design Concept

We propose a bi-context design concept to visualize cross-view data relationships in SightBi. It highlights that enriching MV by creating extra views to show computed data relationships across different views. We refer to this newly created view as a *relationship-view* to differentiate it from existing views. SightBi allows users to create relationship-views for two-level relationships: 1) *bi-group level*, and 2) *multi-group level*. The former is based on biclusters computed between visual elements from pairs of existing views. The latter is based on bicluster-chains that involve visual elements from more than two existing views.

Relationship-views enrich MV since they transform empty display space into views with useful data. Augmented with relationship-views, MV can offer two types of context. One consists of existing views. The other comes from newly added relationship-views. As they are separated, it helps users ditinguish computed relationships from their involved entities (**C1-a**). Also, this separation helps preserve existing views, so user perception of them is not significantly affected by newly added marks for relationships (**C1-b**). This bi-context design supports all six ways of user explorations (**C2**). With newly created relationship-views, users can choose where to start explorations (i.e., an existing view or a relationship-view) and switch the context usage.

5.2.2 Visual Encodings for Cross-View Data Relationships

Based on computed biclusters and bicluster-chains, discussed in Section 5.1.2, SightBi supports creating relationship-views for bi-group level and multi-group level of cross-view data relationships. SightBi uses the same visual encodings for them. Figure 8 shows an example of detailed visual encodings of a bi-group level relationship-view (a), which relates locations in a map (b1) with organizations in a list (b2).

A relationship-view shares the same UI components with existing views, such as a panel header, a panel body, a pin button (), and a close button (✕). In the body of a relationship-view, computed biclusters or bicluster-chains are displayed. Each bicluster or bicluster-chain is displayed as a circle (). Its radius reveals the total number of involved elements by using a linear mapping function. The more elements are in a bicluster or bicluster-chain, the larger radius a circle has. Moreover, when users click or hover a mouse pointer on a circle, it gets darker in color (update). Also, the color of a circle changes to red (update), after a user chooses to mark it by using a right-click menu on the circle. Such color updates reveal that the state of a bicluster or bicluster-chain has changed from normal to selected, focused or marked (**C1-d**).

The positions of circles are computed with multidimensional scaling (MDS) [20], so the relative distance between two circles reveals the similarity between two biclusters or bicluster-chains (Table 4, **T4**). The closer two circles locate, the more similar two biclusters or bicluster-chains are. To apply MDS, we transform each bicluster or bicluster-chain into a vector. Each dimension of the vector corresponds to a visual element from a view that belongs to this bicluster or bicluster-chain. With this transformation, we generate a $m \times n$ matrix, where m is the number of biclusters or bicluster-chains, and n corresponds to all visual elements in them. The value of cells in this matrix is assigned as 1, if a bicluster or bicluster-chain has a visual element; otherwise, it is 0. With this matrix, we compute the pairwise distance between biclusters or bicluster-chains, and then form a distance matrix in which both rows and columns are biclusters or bicluster-chains. Such a distance matrix is then used as the input for MDS, which consequently generates the coordinates for each bicluster or bicluster-chain.

For each bicluster or bicluster-chain, SightBi allows users to check it in detail (**C1-c**). SightBi offers two levels of detail for a bicluster or bicluster-chain: a *concrete* level and a *summary* level. A concrete level of detail allows users to see exact visual elements that belong to a bicluster or bicluster-chain, as visually linking a visual element with a bicluster or bicluster-chain via a curved line. (or). A blue curved line reveals an automatic visual linking. A red one is a user-created visual linking. They are discussed in detail in Section 5.3. By following the lines, a user can check the detail of a bicluster or bicluster-chain. SightBi uses this by default, so such visual connections appear when a user selects or hovers on a circle in a relationship-view.

A summary level of detail offers users a quick overview of a bicluster or bicluster-chain. It changes a circle to a mini bar chart (), where all bars are horizontally aligned. The height of a bar reveals the number of visual elements in a view that belong to a bicluster or bicluster-chain. The bars are ordered by the sequence of existing views shown in the display space. For example, in Figure 8, the left bar in a mini bar chart shown in (a) corresponds to the visual elements on the map (b1), as the map was added to the display space earlier than the list (b2). Moreover, SightBi applies a bounding box, with the same size, in each bar. They offer a visual reference to help users compare the number of involved visual elements from different views to gain a quick overview of how different views contribute to a bicluster or bicluster-chain. Using a right-click menu on a circle, users can see its summary or choose to view the summary for all relationships. After a user chooses this, the circle changes into a mini bar chart (change), and curved lines connected to the center of the circle, if displayed (), switch to being linked to the center of different bars based on the view in which the connected visual elements are located ().

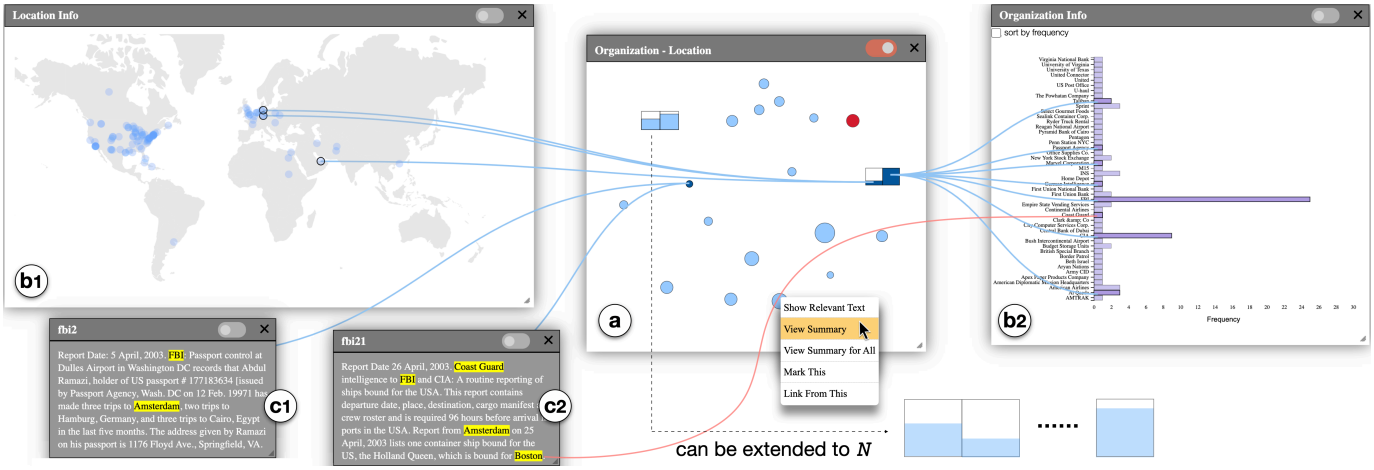

Fig. 8. Visual encodings in SightBi. (a) is a relationship-view showing computed relationships between locations on a map (b1) and organizations in a list (b2). (c1) and (c2) display text, from which locations and organizations have been abstracted. In (a), each relationship is shown as a circle and its involved visual elements in (b1) and (b2) can be linked via curved lines. The radius of a circle reveals the number of elements in this relationship and the relative position between circles indicates the similarity between two relationships. Using a right-click menu on each circle, users can create visual links between visual elements across different views and view the relationship in detail, which visually transforms a circle into a mini bar chart.

5.3 Interactive Cross-View Relationship Management

SightBi offers a set of user interactions for using computed relationships. They serve as complements of the visual encodings, which are designed to help users view and retrieve related information (**C2** and **C4**) and organize information (**C3**).

Resizing. In SightBi, a user can change the size of a view by dragging a handler ( ) on the bottom right corner of each view. Enabling view resizing helps users get empty space, which leaves the room for creating relationship-views (**C1-a, b**).

Pinning and Dragging. SightBi allows flexible layout management at two levels (**C3**): *view*-level (for organizing multiple views) and *relationship*-level (for organizing relationship-marks in a relationship-view). They are supported by enabling users to pin and drag a view, or drag relationship-marks in a relationship-view. When pinning is disabled ( ) for a view, users can move it in the display space by dragging its header. This allows users to create meaningful spatialization that may benefit from using space to think [10]. When pinning is enabled ( ) for a view, it is undraggable. Moreover, SightBi applies the dust and magnet visual metaphor [48] to support users in moving several related views together. Specifically, as a user drags a relationship-view, other views in which some visual elements are visually connected to those in the dragged one via curved lines, can automatically move with the dragged one, if their pinning is disabled (see Figure 1). This helps users quickly pull multiple related views out of a cluttered spatialization (**C3**), such as some related views are covered by a few unrelated ones. In addition to the view-level arrangement, for a relationship-view, if it is pinned, SightBi allows users to adjust the layout of displayed relationships by dragging them (**C3**). This enables users to revise the layout generated based on MDS, which can reveal user understanding of the similarity of computed cross-view data relationships.

Visual linking. SightBi supports two types of visual linking: *automatic* linking and *manual* linking. They are revealed as blue/red curved lines, respectively, connecting visual elements from different views and highlighting the connected visual elements.

Automatic linking shows such visual effects when a user selects or hovers on a visual element in a view or a relationship-mark in a relationship-view. The connections between visual elements are found automatically based on the generated data matrix, computed biclusters or bicluster-chains, discussed in Section 5.1.2. The automatic linking aims to support six ways of user explorations (**C2**), by using a four-way search to create visual links. The search starts with either visual elements in a view, or relationships in a relationship-view, which users interact with, and then checks visual elements in the same view (Table 4, **T1**) or different views (**T2**, **T5**), or relationships in the same relationship-view (**T4**) or different relationship-views (**T3**, **T6**).

For a manual linking, SightBi allows users to specify visual elements to be linked. Specifically, users can flexibly create a visual link between visual marks in any views by using a right-click menu. It works as a complement to the automatic linking, especially for connections based on domain knowledge but not explicitly mentioned in data.

Retrieving original data. SightBi provides a right-click menu on visual elements in a view, or relationship-marks in a relationship-view. From the menu, users can choose to show related parts of original data (e.g., relevant text), which supports on-demand data retrieval (**C4**). Retrieved data is shown in a stand-alone view (Figure 8(c1) and (c2)).

6 Usage Scenario

We present a text analytics scenario that illustrates using SightBi to explore data relationships across multiple views to solve an investigative analysis problem. The scenario explores the Sign of the Crescent dataset [30], which has 41 fictional intelligence reports of suspicious activity. It has been studied to understand human sensemaking process with real-world users in both traditional laboratory settings [52], and crowdsourcing settings [35] using Amazon Mechanical Turk [1].

We extracted named entities (e.g., person, location, organization, etc.) from the text with NLTK [3]. For pairs of sets of named entities (e.g., person and location, person and organization, location and organization, etc.), based on word co-occurrence, we generated a relationship matrix in which each row is an entity from one set, each column is an entity from the other, and each cell is set to 1 or 0 depending on whether the two corresponding entities are related or not. We applied LCM [59] to each relationship matrix to compute biclusters. In total, we had 296 named entities, 2404 entity-connections, and 158 biclusters.

Suppose that Ella is an investigator, who is given a task of identifying potential threats and key players from a collection of intelligence reports. She loads the data into SightBi and starts her investigation. Figure 9 illustrates key steps in Ella's analysis process.

Overviewing data relationships between a pair of views. Ella starts analysis with two views, a graph view and a map view, in the SightBi workspace. The former shows connections among persons in a graph. The latter displays locations mentioned in the reports on a map. Glancing at them, Ella feels that the graph has too many edges and lots of locations have been reported. She thinks that checking each individually will not help solve the mystery. Ella decides to explore relationships between them to see if any collusions exist among persons based on their involved events reported at the same locations. As requested, SightBi shows a relationship-view between the two views in Ella's workspace. By looking at it, Ella quickly gets an overview of the relationships and finds that five circles are near each other, which seems similar (T4). To verify this, she chooses to see the summary of all relationships in this relationship-view (Figure 9 ①), which changes

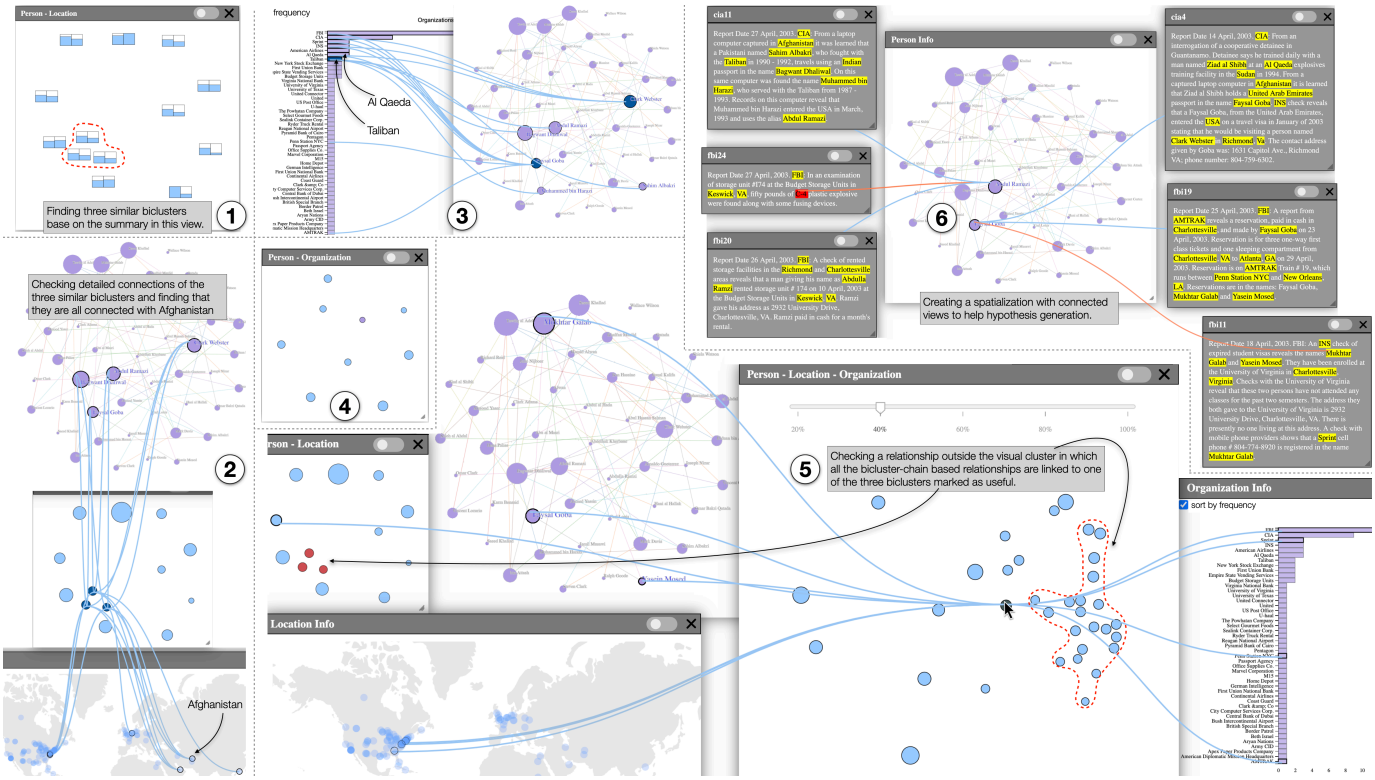

Fig. 9. Key analytical steps of using SightBi to identify an important, colluded threat from the Sign of the Crescent dataset [30].

circles to mini bar charts. By checking the summary, Ella finds that three of the five neighboring clusters are similar (T4), as the number of involved elements from the two views are almost the same.

Switching the usage of bi-context. Following this finding, Ella changes the relationship-view to the simple mode that uses circles and checks the three similar ones in detail (Figure 9 ②), by switching her focus back to the graph view and the map view. Ella hovers her mouse pointer on each circle and SightBi shows its detailed connections with elements in the graph and the map (T5). By following the lines, Ella finds that they each connect with three of four frequently mentioned persons in the graph (circles of a relatively bigger size), and all are linked to `Afghanistan`, a sensitive location for Ella. Ella goes back to the relationship-view and marks the three investigated relationships as useful. Then, she decides to follow this lead to check more information, so Ella adds a list view of organization information to her workspace and sorts it by frequency. In this list, two top-ranked and terrorist-related organizations, `Taliban` and `Al Qaeda` quickly catch her attention, so Ella decides to check connections between persons and them. As she hovers the mouse pointer on them, SightBi shows the linked persons (T2). Ella finds that of the four previously noticed persons, two (`Abdul Ramazi` and `Bagwant Dhaliwal`) are linked with `Taliban`, and others (`Faysal Goba` and `Clark Webster`) are connected to `Al Qaeda` (Figure 9 ③). After learning this, Ella wants to check if there are any similar clusters that group persons based on the two organizations (T3), so she adds another relationship-view, between the graph and the list, to her workspace and switches the focus of her analysis. However, after seeing this relationship view (Figure 9 ④), Ella realizes that there are no obviously similar clusters because the circles are not near each other.

Involving several views to gain deeper insights. As there are three types of information (persons, locations and organizations) displayed in three different views, Ella thinks that fusing the information together may lead to deeper insights. Following Ella's request, SightBi includes all three views to compute cross-view data relationships, and adds a chain-level relationship-view to her workspace (Figure 9 ⑤). Ella adjusts the threshold for computing bicluster-chains to 40% and starts her examination. By checking detailed connections between this chain-level relationship-view and the relationship-view between the graph (person) and the map (location) (T6), Ella finds that the majority of neighboring circles (over 60% of displayed circles) in the chain-level relationship-view are related to the three relationships previously marked as useful. These circles form a visual cluster. Ella checks another circle that is slightly closer to this cluster (T4) than other circles outside the cluster. SightBi shows detailed connections with both the three different views and the two relationship-views (between person and location, and between person and organization). Ella finds that it links to a different circle in the person-location relationship-view (T6) and brings more locations and organizations (T5). Then Ella checks a few more outside circles but none of them are linked to the three relationships marked as useful. This causes Ella to think that the three marked relationships may involve coherent or colluded activities.

On-demand data retrieval and hypothesis generation. To verify this, Ella uses the right-click menu on the three circles to request related documents. Based on information in the three marked relationships, SightBi finds and adds relevant documents to Ella's workspace. For each document, SightBi highlights named entities belonging to the same biclusters (T1). This helps direct Ella's attention to useful information in the document. In reading the text, Ella starts forming a spatialization (Figure 9 ⑥) by placing closely related documents near the persons in the graph and manually creating links between the person and some key words in the text (e.g., `C-4`). After identifying six key documents and linking them to the graph, Ella thinks that she has enough information. Using the dragging and automatically moving feature of SightBi, Ella pulls her created spatialization, including the graph and linked documents, out of the other views. Referring to this spatialization, Ella hypothesizes that `Abdul Ramazi` provided the explosive material, `C-4`, to `Faysal Goba`, who worked with `Mukhtar Galab` and `Yasein Mosed` to plan an attack on `AMTRAK` train 19.

7 Initial Expert Feedback

To better understand the usage of SightBi, we conducted interviews with two experts: a data scientist in an IT company (E1) and a computational scientist of a research institution (E2). Both have worked in their fields for over ten years and are experienced in using multiple views for data analysis. They were asked to use SightBi to identify potential threats from the Atlantic Storm dataset [29]. It is similar to the usage scenario (Section 6), but has more documents. After a 30-minute exploration,

each gave us feedback on SightBi.

Overall, they appreciated the capability of seeing computed relationships in stand-alone views. E1 commented, "*It [a relationship-view] saves me much effort in the tedious trial-and-error attempts for finding groups of suspicious guys.*" E2 mentioned, "*It [a relationship-view] seems working as a summary of related views, so I don't need to check that many nodes in the graph.*" They benefited from the interactive features offered by SightBi. The interactive visual linking was considered useful, with E1 commenting that "*Following [curved] lines, I can keep browsing related information from one piece to another*", and E2 saying that "*These [curved] lines help me to track everything.*" Moreover, they appraised the dragging and automatically moving feature. E1 said, "*It's so convenient that I can move several views at once.*" E2 mentioned, "*This well preserves my effort on the layout work.*"

The two experts made three suggestions for improving SightBi. First, for visual linking, E2 indicated the need for a recommendation mechanism, by saying "*While it seems that everything can be linked, it would be better that the system can tell me which paths [several connected elements from multiple views] to check.*" Second, they both asked for automatic methods to organize multiple views; E1 asked "*Can it [SightBi] organize these views?*" and E2 commented, "*It would be better if the views can be [automatically] ordered.*" Third, considering the trade-off of using relationship-views, E1 said, "*Sometimes using them [relationship-views] causes problems, as I had to move a number of views to leave room for them; otherwise they covered others*."

The feedback suggests the usefulness of SightBi, especially two of its features: showing computed relationships and interactive visual linking. The former saves effort in finding group-level relationships across different views. The latter enables users to check relationships in detail. However, considering that the number of visual links can be large, enabling prioritization of them is helpful to direct user attention to important ones. Regarding the capability of managing the layout of multiple views, the experts appreciated the function of simultaneously moving multiple views, but their feedback indicates that such a manual way of organizing multiple views is not sufficient. Although it helps maintain a spatialization of multiple views, manually moving views to create a spatialization is not effective. This calls for the support of multi-view layout computation. Moreover, the cost of adding relationship-views was raised. Thus, whether the benefit of using relationship-views outweighs the cost remains unanswered.

8 Discussion and Conclusion

We present SightBi, a visual analytics technique for exploring cross-view data relationships. SightBi formalizes cross-view data relationships as biclusters and bicluster-chains (for bi-group and multi-group level of relationships) and computes them from data. SightBi applies a bi-context design that creates stand-alone relationship-views. Moreover, SightBi allows users to interactively check relationships and visual elements and flexibly organize multiple views.

8.1 Comparison to Existing Techniques

The design of SightBi has three advantages. First, compared to view-coordination techniques (e.g., brushing and linking), SightBi saves user effort in finding group-level relationships, as they are computed from data and shown in stand-alone relationship-views. Second, relationship-views separate cross-view data relationships from visual elements in existing views. This helps reduce the interference with user perception of visual elements in existing views. Relationship-views also offer an overview that enables users to explore computed relationships directly. For coordination based techniques, user exploration is limited to visual elements only, as there is no overview of relationships. Third, compared to simple link techniques (e.g., VisLink [16]), SightBi can show more complex relationships (e.g., group-level relationships). Moreover, visual elements in relationship-views potentially serve as edge bundles, which reduces the number of edges shown with simple links.

We walk through an investigative analysis scenario to demonstrate the usefulness of SightBi. While other bicluster-based visual analysis tools (e.g., Bixplorer [52], BiSet [55] and BiDots [72]) have also been reported as helpful for such analyses, the key advantages and differences between them and SightBi lie in two aspects. First, SightBi allows users to browse a variety of views (e.g., map, list, and graph) that enrich visual context for sensemaking activities. In Bixplorer, only matrices being offered, and in BiSet and BiDots, users can only rely on lists of information (e.g., searching for Boston from a list of locations in BiSet versus seeing Boston on a map in SightBi). Second, SightBi offers more advanced layout management capabilities than the other three (e.g., dragging and moving a created spatialization that includes several views), which better supports using space to think [10]. BiSet and BiDots support list ordering but do not allow users to shuffle lists, so users can only see linked information in a fixed sequence. Bixplorer allows users to move matrices, but is limited to one at a time, so users cannot move a previously created spatialization without breaking it.

In summary, SightBi can effectively support users in exploring group-level cross-view relationships, as they are computed as biclusters or bicluster-chains from data and displayed in relationship-views. Such computations do not rely on any application domains, so it is possible to apply SightBi to a variety of analysis scenarios. With relationship-views, SightBi enables six ways of user exploration (Table 4), which cannot be fully supported by existing techniques.

8.2 Limitations and Future Work

SightBi computes cross-view data relationships as biclusters. Such a calculation may lead to information loss (e.g., not all individual-level relations are included in computed biclusters). As relationship-views rely on bicluster computation, individual-level relationships that do not belong to any biclusters cannot be shown. When no biclusters can be found from data, SightBi behaves the same as the brushing and linking technique, so it performs better for group-level relationships.

Regarding relationship-views in SightBi, MDS is used to plot computed biclusters. While it aims to reveal the similarity among biclusters, how effective it is for users to read biclusters and their links to visual elements in the other views is not clear. This requires further studies.

SightBi facilitates exploring and managing multiple views, but with the sacrifice of more visual widgets. Due to adding relationship-views, it takes more display space and may add more cognitive load to users. Whether the expected benefits outweigh these risks, or if it is possible to design certain minimal additions, instead of creating additional views, to the existing UI of multiple views, need further exploration.

Using visual links can cause visual clutter, as discussed in Section 2.1, which is another limitation of SightBi. As SightBi separates relationships from visual elements, users need visual guidance to check relationships in detail. To achieve this, SightBi uses curved lines. However, if there are too many curved lines, especially when users select multiple relationships that involve visual elements from a large number of views, visual clutter occurs. Thus, similar to using proximity to reveal cross-view data relationships, the curved lines used in SightBi work for small sets of views. To improve this, we will need to study other visual encodings to reveal visual elements in relationships.

It is possible to extend SightBi to support users in exploring relations between elements of temporal and spatial proximity, which exist in various scenarios. The two types of elements are often shown in different views and sometimes they do not exactly match each other. For such cases, we can choose to compute partial biclusters [22], which can be easily incorporated into SightBi. Moreover, while it is possible to apply SightBi to a variety of data, besides the Sign of the Crescent dataset and the Atlantic Storm dataset, SightBi currently does not support users in trying their own datasets. We plan to enhance SightBi with functions of computing partial biclusters and loading different datasets.

Acknowledgments

This research is supported in part by NSF Grants IIS-2002082, SMA-2022443, the Research and Artistry Opportunity Grant from Northern Illinois University, and the NSERC Discovery Grant. Maoyuan Sun acknowledges the Argonne Leadership Computing Facility for support during the summer of 2020 and 2021 as part of the Argonne National Laboratory faculty research program.

REFERENCES

[1] Amazon mechanical turk. https://www.mturk.com/.

[2] In-spire. https://in-spire.pnnl.gov/.

[3] Natural language toolkit. https://www.nltk.org/.

[4] Tableau software. https://www.tableau.com/.

[5] C. Ahlberg. Spotfire: an information exploration environment. *ACM SIGMOD Record*, 25(4):25–29, 1996. doi: 10.1145/245882.245893

[6] C. Ahlberg and B. Shneiderman. Visual information seeking: Tight coupling of dynamic query filters with starfield displays. In *The craft of information visualization*, pp. 7–13. Elsevier, 2003. doi: 10.1016/B978-155860915-0/50004-4

[7] B. Alper, N. Riche, G. Ramos, and M. Czerwinski. Design study of linesets, a novel set visualization technique. *IEEE Transactions on Visualization & Computer Graphics*, (12):2259–2267, 2011. doi: 10.1109/TVCG.2011.186

[8] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers. The state-of-the-art of set visualization. In *Computer Graphics Forum*, vol. 35, pp. 234–260. Wiley Online Library, 2016. doi: 10.1111/cgf.12722

[9] R. Amar and J. Stasko. Best paper: A knowledge task-based framework for design and evaluation of information visualizations. In *IEEE Symposium on Information Visualization*, pp. 143–150. IEEE, 2004. doi: 10.1109/INFVIS.2004.10

[10] C. Andrews, A. Endert, and C. North. Space to think: large high-resolution displays for sensemaking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 55–64. ACM, 2010. doi: 10.1145/1753326.1753336

[11] N. Boukhelifa, J. C. Roberts, and P. J. Rodgers. A coordination model for exploratory multiview visualization. In *Proceedings of International Conference on Coordinated and Multiple Views in Exploratory Visualization*, pp. 76–85. IEEE, 2003. doi: 10.1109/CMV.2003.1215005

[12] N. Boukhelifa and P. J. Rodgers. A model and software system for coordinated and multiple views in exploratory visualization. *Information Visualization*, 2(4):258–269, 2003. doi: 10.1057/palgrave.ivs.9500057

[13] I. Boyandin, E. Bertini, P. Bak, and D. Lalanne. Flowstrates: An approach for visual exploration of temporal origin-destination data. In *Computer Graphics Forum*, vol. 30, pp. 971–980. Wiley Online Library, 2011. doi: 10.1111/j.1467-8659.2011.01946.x

[14] R. Burtner, S. Bohn, and D. Payne. Interactive visual comparison of multimedia data through type-specific views. In *Visualization and Data Analysis 2013*, vol. 8654, p. 86540M. International Society for Optics and Photonics, 2013. doi: 10.1117/12.2004735

[15] X. Chen, W. Zeng, Y. Lin, H. M. Al-Maneea, J. Roberts, and R. Chang. Composition and configuration patterns in multiple-view visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 2020. doi: 10.1109/TVCG.2020.3030338

[16] C. Collins and S. Carpendale. Vislink: Revealing relationships amongst visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1192–1199, 2007. doi: 10.1109/TVCG.2007.70521

[17] C. Collins, G. Penn, and S. Carpendale. Bubble sets: Revealing set relations with isocontours over existing visualizations. *IEEE Transactions on Visualization & Computer Graphics*, (6):1009–1016, 2009. doi: 10.1109/TVCG.2009.122

[18] G. Convertino, J. Chen, B. Yost, Y.-S. Ryu, and C. North. Exploring context switching and cognition in dual-view coordinated visualizations. In *Proceedings International Conference on Coordinated and Multiple Views in Exploratory Visualization-CMV 2003-*, pp. 55–62. IEEE, 2003. doi: 10.1109/CMV.2003.1215003

[19] M. Cordeil, A. Cunningham, T. Dwyer, B. H. Thomas, and K. Marriott. Imaxes: Immersive axes as embodied affordances for interactive multivariate data visualisation. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pp. 71–83. ACM, 2017. doi: 10.1145/3126594.3126613

[20] M. A. Cox and T. F. Cox. Multidimensional scaling. In *Handbook of Data Visualization*, pp. 315–347. Springer, 2008. doi: 10.1007/978-3-540-33037-0_14

[21] A. Endert, P. Fiaux, and C. North. Semantic interaction for visual text analytics. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 473–482. ACM, 2012. doi: 10.1145/2207676.2207741

[22] K. Eren, M. Deveci, O. Küçüktunç, and Ü. V. Çatalyürek. A comparative analysis of biclustering algorithms for gene expression data. *Briefings in Bioinformatics*, 14(3):279–292, 2013. doi: 10.1093/bib/bbs032

[23] P. Fiaux, M. Sun, L. Bradel, C. North, N. Ramakrishnan, and A. Endert. Bixplorer: Visual analytics with biclusters. *Computer*, 46(8):90–94, 2013. doi: 10.1109/MC.2013.269

[24] T. Geymayer, M. Steinberger, A. Lex, M. Streit, and D. Schmalstieg. Show me the invisible: visualizing hidden content. In *Proceedings of the 32nd annual ACM Conference on Human Factors in Computing Systems*, pp. 3705–3714. ACM, 2014. doi: 10.1145/2556288.2557032

[25] S. Gratzl, N. Gehlenborg, A. Lex, H. Pfister, and M. Streit. Domino: Extracting, comparing, and manipulating subsets across multiple tabular datasets. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2023–2032, 2014. doi: 10.1109/TVCG.2014.2346260

[26] G. A. Grothaus, A. Mufti, and T. Murali. Automatic layout and visualization of biclusters. *Algorithms for Molecular Biology*, 1(1):1–11, 2006. doi: 10.1186/1748-7188-1-15

[27] J. Heinrich, R. Seifert, M. Burch, and D. Weiskopf. Bicluster viewer: a visualization tool for analyzing gene expression data. In *International Symposium on Visual Computing*, pp. 641–652. Springer, 2011. doi: 10.1007/978-3-642-24028-7_59

[28] N. Henry, J.-D. Fekete, and M. J. McGuffin. Nodetrix: a hybrid visualization of social networks. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1302–1309, 2007. doi: 10.1109/TVCG.2007.70582

[29] F. Hughes. Discovery, proof, choice: The art and science of the process of intelligence analysis, case study 6,"all fall down". 2005.

[30] F. Hughes and D. Schum. Discovery-proof-choice, the art and science of the process of intelligence analysis-preparing for the future of intelligence analysis. *Washington, DC: Joint Military Intelligence College*, 2003.

[31] J.-F. Im, M. J. McGuffin, and R. Leung. Gplom: the generalized plot matrix for visualizing multidimensional multivariate data. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2606–2614, 2013. doi: 10.1109/TVCG.2013.160

[32] W. Javed and N. Elmqvist. Exploring the design space of composite visualization. In *IEEE Pacific Visualization Symposium*, pp. 1–8. IEEE, 2012. doi: 10.1109/PacificVis.2012.6183556

[33] P. Koytek, C. Perin, J. Vermeulen, E. André, and S. Carpendale. Mybrush: Brushing and linking with personal agency. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):605–615, 2018. doi: 10.1109/TVCG.2017.2743859

[34] A. Lex, M. Streit, E. Kruijff, and D. Schmalstieg. Caleydo: Design and evaluation of a visual analysis framework for gene expression data in its biological context. In *IEEE Pacific Visualization Symposium*, pp. 57–64. IEEE, 2010. doi: 10.1109/PACIFICVIS.2010.5429609

[35] T. Li, K. Luther, and C. North. Crowdia: Solving mysteries with crowdsourced sensemaking. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):1–29, 2018. doi: 10.1145/3274374

[36] S. C. Madeira and A. L. Oliveira. Biclustering algorithms for biological data analysis: a survey. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 1(1):24–45, 2004. doi: 10.1109/TCBB.2004.2

[37] W. Meulemans, N. H. Riche, B. Speckmann, B. Alper, and T. Dwyer. Kelpfusion: A hybrid set visualization technique. *IEEE Transactions on Visualization and Computer Graphics*, 19(11):1846–1858, 2013. doi: 10.1109/TVCG.2013.76

[38] T. Munzner. *Visualization analysis and design*. CRC press, 2014.

[39] C. North and B. Shneiderman. A taxonomy of multiple window coordination. Technical report, 1997.

[40] C. North and B. Shneiderman. Snap-together visualization: A user interface for coordinating visualizations via relational schemata. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, pp. 128–135. ACM, 2000. doi: 10.1145/345513.345282

[41] T. Pattison and M. Phillips. View coordination architecture for information visualisation. In *Proceedings of the Asia-Pacific Symposium on Information Visualisation*, vol. 9, pp. 165–169. ACM, 2001. doi: 10.5555/564040.564061

[42] P. Pirolli and S. Card. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of International Conference on Intelligence Analysis*, vol. 5, pp. 2–4. McLean, VA, USA, 2005.

[43] J. Roberts. State of the art: Coordinated & multiple views in exploratory visualization. In *5th International Conference on Coordinated and Multiple Views in Exploratory Visualization*, pp. 61–71. IEEE, 2007. doi: 10.1109/CMV.2007.20

[44] R. Santamaría, R. Therón, and L. Quintales. Bicoverlapper 2.0: visual analysis for gene expression. *Bioinformatics*, 30(12):1785–1786, 2014.

doi: 10.1093/bioinformatics/btu120

[45] A. Sarikaya, M. Correll, L. Bartram, M. Tory, and D. Fisher. What do we talk about when we talk about dashboards? *IEEE Transactions on Visualization and Computer Graphics*, 25(1):682–692, 2018. doi: 10.1109/TVCG.2018.2864903

[46] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *The craft of information visualization*, pp. 364–371. Elsevier, 2003. doi: 10.1016/B978-155860915-0/50046-9

[47] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, 2006. doi: 10.1109/TVCG.2006.166

[48] J. Soo Yi, R. Melton, J. Stasko, and J. A. Jacko. Dust & magnet: multivariate information visualization using a magnet metaphor. *Information visualization*, 4(4):239–256, 2005. doi: 10.1057/palgrave.ivs.9500099

[49] J. Stasko, C. Görg, and Z. Liu. Jigsaw: supporting investigative analysis through interactive visualization. *Information visualization*, 7(2):118–132, 2008. doi: 10.1057/palgrave.ivs.9500180

[50] M. Streit, S. Gratzl, M. Gillhofer, A. Mayr, A. Mitterecker, and S. Hochreiter. Furby: fuzzy force-directed bicluster visualization. *BMC bioinformatics*, 15(6):1–13, 2014. doi: 10.1186/1471-2105-15-S6-S4

[51] M. Sun. *Visual Analytics with Biclusters: Exploring Coordinated Relationships in Context*. PhD thesis, Virginia Tech, 2016.

[52] M. Sun, L. Bradel, C. L. North, and N. Ramakrishnan. The role of interactive biclusters in sensemaking. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1559–1562, 2014. doi: 10.1145/2556288.2557337

[53] M. Sun and G. Convertino. Interver: Drilling into categorical-numerical relationships. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pp. 44–51, 2016. doi: 10.1145/2909132.2909253

[54] M. Sun, D. Koop, J. Zhao, C. North, and N. Ramakrishnan. Interactive bicluster aggregation in bipartite graphs. In *IEEE Visualization Conference (VIS)*, pp. 246–250. IEEE, 2019. doi: 10.1109/VISUAL.2019.8933546

[55] M. Sun, P. Mi, C. North, and N. Ramakrishnan. Biset: Semantic edge bundling with biclusters for sensemaking. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):310–319, 2016. doi: 10.1109/TVCG.2015.2467813

[56] 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. doi: 10.1109/TVCG.2014.2346665

[57] M. Sun, J. Zhao, H. Wu, K. Luther, C. North, and N. Ramakrishnan. The effect of edge bundling and seriation on sensemaking of biclusters in bipartite graphs. *IEEE Transactions on Visualization and Computer Graphics*, 25(10):2983–2998, 2019. doi: 10.1109/TVCG.2018.2861397

[58] E. R. Tufte. *The visual display of quantitative information*. Graphics Press, 2001.

[59] T. Uno, T. Asai, Y. Uchida, and H. Arimura. An efficient algorithm for enumerating closed patterns in transaction databases. In *International Conference on Discovery Science*, pp. 16–31. Springer, 2004. doi: 10.1007/978-3-540-30214-8_2

[60] C. Viau and M. J. McGuffin. Connectedcharts: explicit visualization of relationships between data graphics. In *Computer Graphics Forum*, vol. 31, pp. 1285–1294. Wiley Online Library, 2012. doi: 10.1111/j.1467-8659.2012.03121.x

[61] M. Waldner, W. Puff, A. Lex, M. Streit, and D. Schmalstieg. Visual links across applications. In *Proceedings of Graphics Interface*, pp. 129–136. Canadian Information Processing Society, 2010.

[62] M. Waldner and D. Schmalstieg. Collaborative information linking: Bridging knowledge gaps between users by linking across applications. In *IEEE Pacific Visualization Symposium*, pp. 115–122. IEEE, 2011. doi: 10.1109/PACIFICVIS.2011.5742380

[63] M. Q. Wang Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*, pp. 110–119. ACM, 2000. doi: 10.1145/345513.345271

[64] C. Weaver. Building highly-coordinated visualizations in improvise. In *IEEE Symposium on Information Visualization*, pp. 159–166. IEEE, 2004. doi: 10.1109/INFVIS.2004.12

[65] C. Weaver. Cross-filtered views for multidimensional visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 16(2):192–204, 2010. doi: 10.1109/TVCG.2009.94

[66] H. Wu, M. Sun, P. Mi, N. Tatti, C. North, and N. Ramakrishnan. Interactive discovery of coordinated relationship chains with maximum entropy models. *ACM Transactions on Knowledge Discovery from Data*, 12(1):1–34, 2018. doi: 10.1145/3047017

[67] P. Xu, N. Cao, H. Qu, and J. Stasko. Interactive visual co-cluster analysis of bipartite graphs. In *IEEE Pacific Visualization Symposium*, pp. 32–39. IEEE, 2016. doi: 10.1109/PACIFICVIS.2016.7465248

[68] M. J. Zaki and C.-J. Hsiao. Efficient algorithms for mining closed itemsets and their lattice structure. *IEEE Transactions on Knowledge and Data Engineering*, 17(4):462–478, 2005. doi: 10.1109/TKDE.2005.60

[69] H. Zhang, M. Sun, D. Yao, and C. North. Visualizing traffic causality for analyzing network anomalies. In *Proceedings of the ACM International Workshop on International Workshop on Security and Privacy Analytics*, pp. 37–42, 2015. doi: 10.1145/2713579.2713583

[70] J. Zhao. *Interactive Visual Data Exploration: A Multi-Focus Approach*. PhD thesis, University of Toronto, 2015.

[71] J. Zhao, C. Collins, F. Chevalier, and R. Balakrishnan. Interactive exploration of implicit and explicit relations in faceted datasets. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2080–2089, 2013. doi: 10.1109/TVCG.2013.167

[72] J. Zhao, M. Sun, F. Chen, and P. Chiu. Bidots: Visual exploration of weighted biclusters. *IEEE Transactions on Visualization and Computer Graphics*, 24(1):195–204, 2017. doi: 10.1109/TVCG.2017.2744458

[73] J. Zhao, M. Sun, F. Chen, and P. Chiu. Missbin: Visual analysis of missing links in bipartite networks. In *IEEE Visualization Conference (VIS)*, pp. 71–75. IEEE, 2019. doi: 10.1109/VISUAL.2019.8933639

[74] J. Zhao, M. Sun, F. Chen, and P. Chui. Understanding missing links in bipartite networks with missbin. *IEEE Transactions on Visualization and Computer Graphics*, 2020. doi: 10.1109/TVCG.2020.3032984