GridSet: Visualizing Individual Elements and Attributes for Analysis of Set-Typed Data

Haeyong Chung, Santhosh Nandhakumar, Seungwon Yang

Abstract— We present GridSet, a novel set visualization for exploring elements, their attributes, intersections, as well as entire sets. In this set visualization, each set representation is composed of glyphs, which represent individual elements and their attributes utilizing different visual encodings. In each set, elements are organized within a grid treemap layout that can provide space-efficient overviews of the elements structured by set intersections across multiple sets. These intersecting elements can be connected among sets through visual links. These visual representations for the individual set, elements, and intersection in GridSet facilitate novel interaction approaches for undertaking analysis tasks by utilizing both macroscopic views of sets, as well as microscopic views of elements and attribute details. In order to perform multiple set operations, GridSet supports a simple and straightforward process for set operations through dragging and dropping set objects. Our use cases involving two large set-typed datasets demonstrate that GridSet facilitates the exploration and identification of meaningful patterns and distributions of elements with respect to attributes and set intersections for solving complex analysis problems in set-typed data.

1 Introduction

A wide variety of data-analysis problems can be addressed through the use of sets, which can be defined as a collection of data entities (i.e., elements) with different types of attributes. For instance, for a movie dataset, each movie title becomes an element; while a movie genre, which is one of the attributes, refers to a set. Accordingly, the genre of movies becomes a *set-typed attribute* that decides sets, and each movie title belongs to different genre sets. Each movie title can also feature multiple other attributes, such as ratings, budget, release date, language, etc. Additionally, the co-occurrence of movie elements within different genre sets can indicate one or more set intersections.

In general, analysis of set-typed data typically involves exploring all of these sets, elements, and attributes [1]. Particularly, analysts should be able to explore patterns and distributions of elements in terms of attribute values, while at the same time being able to identify their different set memberships and intersections. There are, however, three main challenges associated with analyzing set-typed data with existing visualization tools.

First, it is difficult for a user to analyze and explore details of individual elements and attributes associated with each element, while still considering their overall set memberships and intersections.

Second, few existing set visualization techniques have been designed to support analysis of multiple attributes that describe common/individual properties of a set. Specifically, it is important for the user to see and understand two main factors: (a) how attribute values of elements are distributed in specific sets or intersections, and (b) how such attribute distributions of elements are related to their set membership.

Third, when conducting set-typed data analysis with visualizations, a user may undertake multiple procedures in performing combined set operations (e.g., a combination of multiple intersections and unions) through the use of a graphical user interface (GUI); this is typically followed by additional steps for saving the results of set operations for later comparison. As a result, users may be challenged by a gap between their set operation goals and the actions or procedures needed to attain those goals with current visualization approaches [2].

To address these challenges, we propose a novel set visualization, *GridSet*, which supports comprehensive analyses of both elements and attributes, as well as their set-memberships, in set-typed data. Particularly, GridSet is capable of visualizing a large number of individual elements and attributes by combining unit visualization [3, 4] and pixel-oriented visualization techniques [5] (Fig. 1). As such, GridSet will help users view and understand how elements' set memberships and attributes are interrelated and what elements (with specific attribute values) belong to different sets of interest.

GridSet can effectively display different levels of detail in sets. A set grid, which is a visual representation of a set, is composed of grid cells that show elements and their attributes mapped to different visual encodings (color, size, shape, icon, or even nested visual representations). These elements are organized in the Grid Treemap layout [6], which can provide space-efficient overviews of the elements structured by their set memberships and intersections. A group of intersecting elements can be linked with contours among multiple sets to show n-set intersections. These visual representations also allow users to highlight and filter particular elements and intersections based on query conditions for attributes and associated sets.

- *Haeyong Chung is with the Department of Computer Science, University of Alabama in Huntsville, Huntsville, AL 35899. E-mail: hc0021@uah.edu.*
- *Santhosh Nandhakumar is with the Department of Computer Science, University of Alabama in Huntsville. E-mail: sn0026@uah.edu.*
- *Seungwon Yang is with the Department of Information Science, Louisiana State University. E-mail: seungwonyang@lsu.edu.*

xxxx-xxxx/0x/$xx.00 © 200x IEEE Published by the IEEE Computer Society

Additionally, GridSet provides new ways to compare element attributes across sets and to find commonalities among them based on attributes or set intersections. For example, it allows for ordering element glyphs based on attribute values, while multiple set representations can be spatially organized by the user to represent possible latent relationships among sets [7]. Separate and draggable set representations also support a straightforward process for performing combined set operations.

2 Related Work

GridSet builds upon prior work on set visualization techniques, focusing on how visualizations can be empowered to better represent elements/attributes, sets, and intersections. Notably, a number of set visualizations have presented effective techniques to support set-typed data analysis through diverse visual representations: region-based overlay [8, 9], line-based overlay [10], glyph-based overlay [11, 12], node-link [13, 14], matrix-based techniques [15, 16], etc. In this section, we discuss and compare GridSet with existing set-visualization techniques.

2.1 Visualizing Elements Individually

Three types of set visualization techniques, which enable users to visualize each element individually, are particularly relevant to GridSet. First, Euler diagram-based approaches employ overlapped contour regions to represent intersections between sets, thereby aiding comprehension. For example, Simonetto et al. [17] present an algorithm that can automatically generate Euler diagrams for set-typed data. The researchers use Bezier curves with translucent-shaded contours to represent sets and increase the clarity of set boundaries. Each contour includes small glyphs or icons with labels to represent set elements that belong to the sets. Riche and Dwyer propose Untangling Euler Diagrams [18], whereby a hierarchical structure of intersecting elements helps to reduce visual complexity. Their approach is able to visualize each intersecting element and place it within the overlapping set representations. In contrast, GridSet duplicates elements within a separate grid and links them across multiple sets to represent intersections visually.

Second, some of the more recent set visualizations focus on depicting set relations over some predefined points or nodes as elements on top of an existing visualization. Using this approach, each element can have a spatial reference based on the location of the visualization component. Meanwhile, a contour or visual link as a set is drawn among these elements to represent set memberships and relations, while overlapping contours indicate intersections. BubbleSet [8] and LineSet [19] enable the analyst to draw set boundaries over existing visualizations based on locations and other attribute values of elements. These visualization techniques use colored contours to visually connect the elements placed over another visualization, such as map or node-link graphs. Similarly, the Kelp diagram [10] depicts set memberships over elements by connecting them with thick, colored curves. In contrast to the link-based set visualizations in which the link visualizes set memberships, GridSet uses visual links to represent intersections of the same group of elements among sets.

Lastly, GridSet's visual representation for a set shares similarities with other visualization approaches based on matrix representations. Notably, these prior visualizations represent the elements as cells in a rectangular grid layout, and each cell includes a glyph that represents the element's set membership. There are several visualizations based on matrix representations. NodeTrix [20] integrates a node-link diagram and adjacency matrix. Each node is overlaid with a matrix representation that represents a graph; the nodes are then connected through links showing how each matrix element interacts with each of the others. ConSet [15] also visualizes each element individually by using a permutation matrix, where each column represents the elements and each row represents a set. In OnSet [16], elements are duplicated, but each element within a different set matrix is placed at the universal cell position across different sets. Frequency Grids [21] also represent the elements as cells in a matrix. Each cell represents a small circle, but it encodes different sets with different colors. However, in contrast to GridSet, these set visualizations do not visualize the attribute values of the elements directly.

2.2 Visualizing Elements and Attributes with Aggregation

Several set visualization techniques focus on achieving better scalability by aggregating a large number of elements and attributes into summary statistics, such as minimum, maximum, sum, average, and standard deviation. Thus, instead of visualizing elements and sets individually, these techniques focus on employing charts (e.g., bar graphs, line charts, pie charts, and scatter plots), in which information related to attributes, elements, and sets is aggregated or summarized [22, 23]. Using these tools, specific information about elements can be aggregated in the visualizations, while more detailed information about each element in a set has to be accessed through a separate view.

Several existing set visualization approaches aggregate information about individual set elements and their multiple associated attributes. Set'o'gram [24] employs bar graphs to visualize elements by aggregating them based on their degrees of membership in sets. Radial Set [25] is similar to Set'o'gram in that it aggregates elements based on the degrees as well, but it uses a node-link diagram in a radial layout to represent the intersections of sets (as nodes). The size of the circular nodes that represent sets is encoded based on the cardinality of the set, while any intersections among these sets are visualized by hyperedges. This technique also maps the attributes to the color of the set interaction links and glyphs. UpSet [22] aggregates elements as a combination matrix, in which the columns represent sets, and rows indicate all possible exclusive intersections of the sets. UpSet provides a separate element view for statistics that are related to element attributes. Similar to GridSet, UpSet utilizes queries to perform set operations. PowerSet [26] uses a treemap layout to provide a compact overview of all intersections in a set system, along with their element attributes, wherein the colors of the treemap tiles are mapped to an aggregated value of an

Fig. 1. The main interface of GridSet: (a) the Main view, (b) the Visual Property menu, (c) the Query view, (d) the Set view (orange-highlighted views represent added sets on the Main view), and (e) the Detail view that provides detailed information of the elements. This figure depicts an analysis of an Academy Awards (AA) dataset defining the award categories as sets, and the individual nominees as elements (blue or orange glyphs in the grids) who have been nominated in each category since the first AA ceremony in 1929. The sets are spatially arranged based on similar award categories. The nominees for the 2017 AA are highlighted in orange. The size of the element glyph is defined by the total number of nominations since 1929. The grids are divided into subdivisions based on common nominees across different award categories; note that they are connected by colored intersection links across different sets.

additional attribute. AggreSet [23] uses a matrix layout to show all the pair-wise intersections between sets; its matrix layout has a relative mode that reveals inclusion and exclusion relationships of sets.

In contrast to these approaches, GridSet individually visualizes different components (attribute, element, set, and intersection) of set-typed data without data aggregations or summary statistics.

3 Design Considerations

Our design considerations (C1-C5) for GridSet are informed primarily by existing techniques from Unit Visualization Framework [27], pixel-oriented visualization [5, 28-30], visual links [31-33], direct manipulation [34], and space-to-think [7].

C1. Visualize individual sets, elements, and attributes: GridSet is capable of visualizing not only the set membership of each element, but also visual patterns and distributions of attributes of elements without converting data into summary statistics or charts. GridSet visualizes each element and its attributes individually and groups them in the Grid Treemap layout. Particularly, each set element can be represented by mapping attributes to the visual properties of the glyphs, such as the size, color, shape, and icon. This approach allows one to access both the set-level patterns and element-level details that might facilitate important insights in the analysis process.

C2. Duplicate elements based on their co-occurrence in the set representations: As the number of sets increases, the potential intersections of elements among those sets can increase as well. In GridSet, we chose to duplicate visual representations (glyphs) of elements that belong to different sets, and connect them through visual links to show their intersections instead of overlapping set regions [35-37]. Duplicating elements in sets enables the user to maintain separate set objects on the visualization view. Accordingly, it enables several new visualization and interaction approaches: creating an ordering of elements within each set; comparing elements and attributes according to their set memberships and intersections; and finding patterns of attributes among sets or n-set intersections. Additionally, this visual approach will assist users in exploring set-typed datasets with queries for searching, highlighting, or filtering individual and multiple elements and attributes in different sets. According to a study by Riche et al. [18], study participants prefer being able to duplicate elements within set visualizations to enhance set-related analysis tasks.

C3. Manage visual complexity resulting from many individual elements and intersections: Since GridSet's visual representation uses a single glyph per element, it is useful for generating insights into relationships among individual elements, attributes, intersections, and sets [38]; however, GridSet's visual representation cannot entirely avoid visual complexity issues resulting from a large number of elements and interactions displayed on the screen. A high level of visual complexity in a visualization view is likely to be exacerbated by an increase in the number of elements and their intersections among sets. To mitigate visual complexity, we adapt a dynamic query for reducing the number of presented elements and intersections, whereby filtering is applied to elements based on a range of their attribute values or keywords. The user can also reduce the number of intersections shown according to two

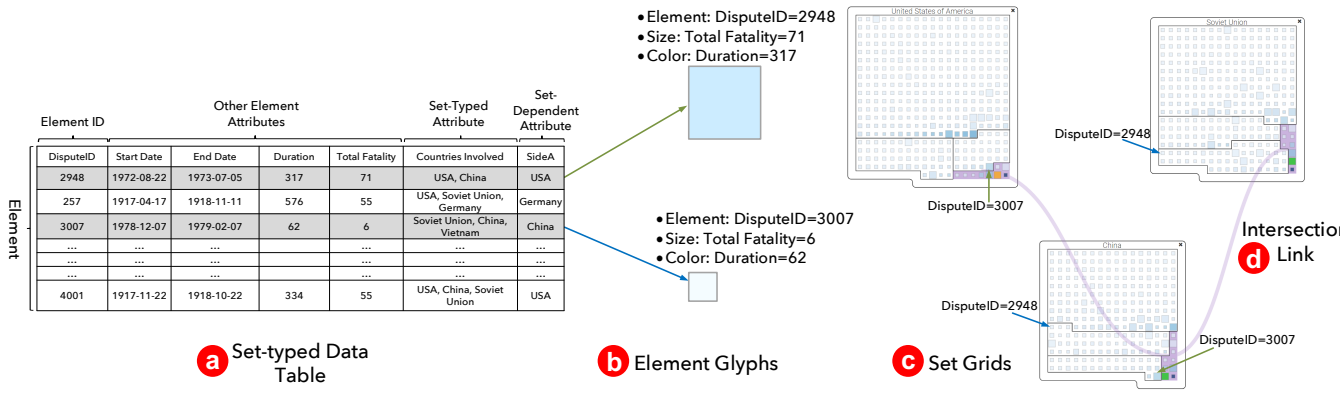


Fig. 2. GridSet's visual mappings from set-typed data table into visual representations that combine small element glyphs, visual properties for attributes, set grids. Each set represents a country and elements in the sets represent different dispute events (e.g., a military conflict or war). The intersection links connected among three sets represent a group of disputes involved in three countries.

query conditions: (a) intersections associated with particular sets, and (b) the number of sets to which the intersecting elements belong. Since a dynamic query will facilitate the ability to omit less important elements and intersections, this strategy will therefore increase the likelihood that information more important to the analyst can be more readily accessed.

C4. Facilitate combined set operations: Instead of depending solely on somewhat indirect or complex interactions associated with the use of GUI widgets, we chose to employ direct manipulation with associated set objects to perform set operations. Using this more interactive approach, combined set operations in series can be performed by simply dragging and dropping set objects. For example, to create a larger union between two sets, one set object is dropped over the other set object. This simple interaction with set objects can reduce a series of intermediary steps involved with a chain of set operations on multiple sets, thereby minimizing the user's effort.

C5. Spatialize semantic relationships among sets: GridSet is designed to represent complex and latent information among sets. The information for semantic relationships among sets may not be specified in given data, but rather emerge from an analyst's external knowledge and/or understanding of the sets. In this regard, it is important for set visualizations to provide users with an additional mode of representing latent relationships among sets or incorporating personal knowledge about sets with information described in data. Flexible spatial layouts of set objects can be applied to externalize and further represent set relationships on screen. Specifically, the analyst can encode different understandings of set relationships according to the set objects' spatial relations and positions on the screen (e.g., proximity, ordering, and alignment of the set objects in terms of specific attributes). For instance, when set objects are placed closer together, they may be perceived to be semantically similar. This type of spatial organization of set objects allows users to transform a layout of set objects into a useful semantic structure (e.g., categories, geographical regions, people, timelines, events, etc.) [7]. (See Section 5 for the actual usage of this approach.)

4 GridSet Techniques

This section describes the design of GridSet's visual representations, user interface, and workspace, which consists of views and menus to support set-typed data analysis tasks. The Main view visualizes the individual sets, elements, and attributes through which users can explore various visual encodings (Fig. 1a). The Visual Property menus allow users to assign visual mappings of attributes for element glyphs and visual encodings (Fig. 1b). The two views on the right (the Set and Detail views) enable the user to add sets and also view detailed information on the sets, elements, and attribute values (Fig. 1d,e). Situated on the left side (Fig. 1c) of the Main view, the Query view enables users to query and filter elements and intersections for exploration of the set-typed data, as well as create new sets based on certain conditions of set relationships and attributes.

4.1 Visual Representations

In GridSet, a data table in set-typed data is mapped into different components of a set representation that consists of a set grid, small glyphs for elements, and visual encodings of attributes (Fig. 2a,b).

4.1.1 Elements and Attributes

GridSet maintains a one-to-one mapping approach between each set element and the respective glyph (Fig. 2b). Each element is represented as a square or circle glyph, which is placed in a set representation to encode both its respective set memberships and attributes. We extend the concepts associated with unit visualizations [4, 29, 39, 40] and pixel-oriented visualizations [5] to visualize each element and its attribute as a pixel or unit.

Specifically, multiple attributes of a set element may have different data types, including numerical, ordered, categorical, nominal, a series of data points, etc. Each element glyph varies with assorted visual encodings by mapping the different types of attributes into different sizes, shapes, colors, icons/images (Fig. 3 right), and nested visualization representations. For categorical attributes, each glyph can be filled in with a unique color to represent different categories. Numerical attributes can be represented

by differently sized glyphs or a different level of color saturation.

Additionally, GridSet's element glyphs can visualize multivariate or temporal attributes of elements using a pre-rendered texture of each glyph. For example, multiple time-varying attribute values associated with the same element can be visualized as a single sparkline, bar graph, etc., on each element glyph (Fig. 3 left). Thus, each glyph also serves as a small multiple display. Importantly, these small multiple or icon glyphs enable users to view and compare localized element patterns in each set, as well as overall patterns across multiple sets. By comparing any differences in the visual patterns of glyphs belonging to intersections, users can also see how set memberships and intersections affect specific attribute patterns.

There are existing set visualizations [16, 21] based on small glyphs/marks in a matrix layout to represent elements and sets. However, they focus on depicting only the occurrence of elements in sets. In contrast, the element glyphs in GridSet can vary with respect to size, shape (square or circle), color, and icon, thereby representing multiple attributes simultaneously. Importantly, these individual representations of elements also allow for a spatial ordering of the elements within each set grid based on both attribute values. The ordering of elements facilitates locating elements with specific attribute values.

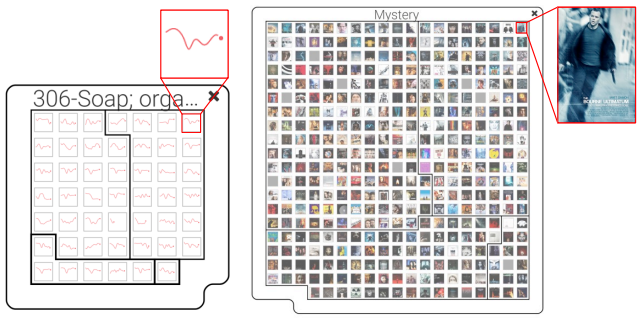


Fig. 3. Additional visual properties of element glyphs. Left: different countries' GDP growth from 2006 to 2016 is visualized as spark lines inside each element. Right: Each movie glyph (element) in Mystery genre (set) shows its movie poster.

4.1.2 Set-Dependent Attributes

In general, the values of one attribute for the same element should be identical across different sets, even though they belong to different sets. However, there is a special type of attribute known as a *set-dependent attribute*, for which the attribute value of the same element may change according to the set to which it belongs. In GridSet, visualizing individual elements and their attributes enables the user to support such set-dependent attributes, which heretofore have not been supported by existing set visualization approaches [1]. For instance, each war (element) would involve multiple countries (sets). However, such war elements may have set-dependent attributes, such as any military alliances or the hostility level of each country. Particularly, set grids for countries can be shown to form different military alliances for different wars. Since GridSet duplicates elements in multiple sets, it can visualize the set-dependent attributes in a direct way. An attribute depicting allies can be visualized easily with simple color-coding of each war element in the set grids for different countries. Even though the same war elements belong to multiple countries, each element glyph can be represented with different colors or icons based on their ally status. (See the supplementary document for an example.)

4.1.3 Sets

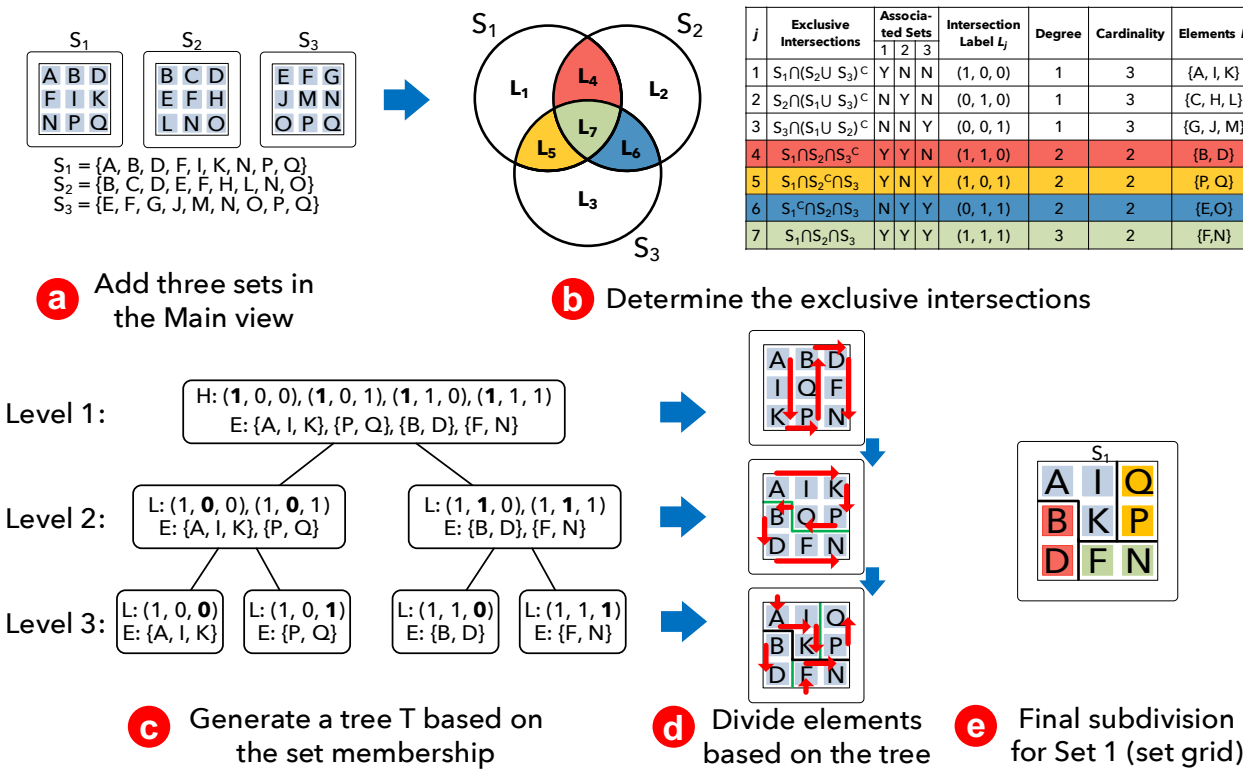
GridSet represents each set as a grid of element glyphs. We will refer to this set representation as a *set grid* (Fig. 2c & 3). In a set grid, elements are divided visually based on their associated exclusive intersections in the treemap layout (see Section 4.1.4). Since GridSet computes a grid for each set that contains only its elements in the set representation, the size of each set grid is determined proportionally by the number of elements contained in the grid [24]. Thus, the width and the height of the grids are different for each set, enabling the user to compare the cardinality (the size) of sets. However, since the number of elements cannot always maintain a square aspect ratio of 1:1, we apply a rule [16] to determine the size of row (m) and column (n) of the grid and to generate more square-like shapes; when the total number of elements in a set cannot form an exact square ($m \neq n$) with an aspect ratio of 1:1, empty cells are added in the last row to maintain the square aspect ratio of the set grid. In addition, each set grid includes a border area that is used to display the set label and additional icons (remove and undo) and grab the set grid for drag-and-drop operations.

4.1.4 Set-Grid Layout

When a new set grid is added onto the Main view, the subdivisions of intersecting elements in set grids already presented in the view are renewed according to the updated intersections between the new and existing sets.

Let n be the number of sets in the Main view and i be the index of a set $S_i \subseteq U$, $1 \leq i \leq n$, if U is the universal set. We index the sets according to the order in which they are added by a user as a set grid onto the Main view (*setIndex* in Algorithm 1 of the supplementary document). Assume that each set is not empty. Detailed procedural steps (P1 to P4) for placement of the elements and subdivisions in a set grid are described as follows. We will explain these steps using examples of three sets ($n = 3$)—S_1, S_2, and S_3—and their exclusive intersections (Fig. 4).

P1. Group elements based on exclusive intersections: All of the elements in each set grid are assigned into different subsets based on their *exclusive intersections* among the sets on the screen (Fig. 4b). For three sets, S_1, S_2, and S_3 (Fig. 4a), the exclusive intersection between S_1 and S_2 refers to a set of elements that belong only to S_1 and S_2, but not to S_3 (i.e., the L_4 region of the Venn diagram shown in Fig. 4b). Thus, exclusive intersections allow all of the elements to be grouped into mutually exclusive subsets, which belong only to specific sets associated with each intersection. Based on this step, we are able to maintain a data structure for exclusive intersections and their associated elements (Fig. 4b Table).



j	Exclusive Intersections	Associated Sets 1	2	3	Intersection Label L_j	Degree	Cardinality	Elements E
1	$S_1 \cap (S_2 \cup S_3)^C$	Y	N	N	(1, 0, 0)	1	3	{A, I, K}
2	$S_2 \cap (S_1 \cup S_3)^C$	N	Y	N	(0, 1, 0)	1	3	{C, H, L}
3	$S_3 \cap (S_1 \cup S_2)^C$	N	N	Y	(0, 0, 1)	1	3	{G, J, M}
4	$S_1 \cap S_2 \cap S_3^C$	Y	Y	N	(1, 1, 0)	2	2	{B, D}
5	$S_1 \cap S_2^C \cap S_3$	Y	N	Y	(1, 0, 1)	2	2	{P, Q}
6	$S_1^C \cap S_2 \cap S_3$	N	Y	Y	(0, 1, 1)	2	2	{E,O}
7	$S_1 \cap S_2 \cap S_3$	Y	Y	Y	(1, 1, 1)	3	2	{F,N}

Fig. 4. An example of the set-grid layout algorithm. For each set grid added in the Main view, the algorithm forms a tree for the exclusive intersections with other sets on screen and then determines the grid treemap layout of elements based on the tree structure.

P2. Construct a tree structure for set intersections: Once all of the exclusive intersections have been identified in order to group elements in each set, these element subsets could be arranged into three different types of tree structures based on: (1) set memberships of the intersecting elements (common sets to which the intersecting elements belong), (2) degrees of the intersecting elements (the number of sets to which the intersecting elements belong), and (3) cardinalities of the intersecting elements (the number of elements in the intersections). These three tree structures generate different layouts of subdivisions within the set grid.

To generate a tree structure based on exclusive intersections among sets, we extended the Untangling Euler Diagram algorithm [18] and UpSet [22] to create a strict hierarchy of exclusive intersections with respect to their associated sets. Algorithm 1 (see the supplementary material for its pseudocode) shows how the tree structure is generated based on the set memberships of the elements in a recursive way. The following four steps demonstrate how the algorithm is able to construct a binary tree based on exclusive intersections (Fig. 4c).

1. Suppose that there are n sets and m available exclusive intersections among these sets in the Main view, represented by the intersection labels $L = (L_1, \ldots, L_m)$. Specifically, let j be an index of a specific exclusive intersection and let L_j be an associated intersection label for the exclusive intersection j, defined by a binary vector $L_j=(p_1, p_2, p_3, \ldots, p_n)$, where each $p_i \in \{0, 1\}^n$ and $1 \le i \le n$, and p_i indicates either the presence (1) or absence (0) of the set S_i in an exclusive intersection j. For example, as shown in Fig. 4b, if the two sets S_1 and S_3 have elements in an exclusive intersection with each other, but not with the set S_2, the corresponding intersection label becomes $L_5 = (p_1, p_2, p_3) = (1, 0, 1)$. Since the intersection labels can also represent the absence of sets in an exclusive intersection, a label can represent a set of elements that belongs only to a single set without any associated intersection; for instance, an intersection label L_1=(1, 0, 0) indicates that elements associated with L_1 do not belong to any other set intersection(s) except S_1 (Fig. 4b).
2. Initiate a binary tree T for S_1, which is the first set to be added to the Main view (Fig. 4c). For all possible exclusive intersections L, let H be the union of the exclusive intersection labels in which S_1 is always present (e.g., (1, *, *)); place H and its associated elements in the root node of T at Level 1, as shown in Fig. 4c. The intersection label L_j can be considered as an n-bit binary number, and the intersection labels within H in the root node should be sorted in ascending order of the binary number (e.g., Level 1 in Fig. 4c).
3. Divide the root node and H into left and right child nodes according to the presence of the next set, S_2, at Level 2 (Fig. 4c Level 2). The right child contains a set of the exclusive intersection label vectors in which S_2 is present (i.e., (*, 1, *)) while the left child node will consist of the remainder of the exclusive intersection label arrays in which S_2 is absent (i.e., (*, 0, *)).
4. Further divide all of the other child nodes at the deeper levels of T recursively in the same manner as detailed in the previous step, until there exists only one exclusive intersection label vector in a child node.

As a result, all the leaf nodes of T have only one exclusive intersection label vector.

Basically, a set grid layout is determined based primarily on the tree structure of the set membership. However, the user can select two alternative subdivision layouts from the Visual Property menus. We use the same algorithms for both degrees and cardinalities of intersections. Specifically, we generate a sorted tree based on either the degree or cardinality of intersections. The exclusive intersections are sorted by the intersection degree or cardinality, after which the median of the degree (or cardinality) is then computed and utilized. Subsets with degrees (or cardinalities) that are less than the median are then added to the left child, while those that are greater than the median are added to the right child. Algorithm 2 in the supplementary document provides detailed steps for generating the tree structure based on either degree or cardinality of sets.

P3. Lay out element glyphs in set grids based on the tree structure: After the tree structure for the sets are formed, they can then be visualized and converted to the treemap layout in each set grid. We use the Grid TreeMap algorithm (GTM) [7] (Fig. 4d), which recursively partitions and allocates a sequence of element glyphs into multiple subdivisions in a set grid; this process is based on the tree T built from P2. GTM produces more space-efficient treemap layouts of element glyphs in which the set grid can be quantized to grid dimensions corresponding to the exact number of element glyphs without any unoccupied grid slots. GridSet employs the following four steps to lay out element glyphs in the set grid:

1. Calculate the size of a set grid based on the total number of elements for a set and default glyph size in the root node of the tree T (i.e., all the elements of the set) to contain every element glyph in the corresponding set. All the elements contained in the root node are first filled in the set grid vertically—either from top to bottom or from bottom to top. (In Fig. 4d, the red arrows indicate the scanning direction in each subdivision.)
2. Divide elements in the initial set grid into subdivisions based on the tree hierarchy of T. The set grid can be subdivided either horizontally or vertically, and the elements in the nodes of T are assigned into grid slots in the subdivisions as we move down toward deeper levels of T. For example, at Level 1 of T, the algorithm allocates elements in the left child to the upper subdivision, and elements in the right child to the bottom subdivision, as shown in Fig. 4d middle. As a result, this allocation process leads to splitting the entire element grid into two vertical subdivisions (split with a green stair-step line in Fig. 4d middle). At the next level, it then divides each vertical subdivision into two horizontal subdivisions and arranges associated elements in these subdivisions by scanning the columns.
3. For lower levels of T, GTM divides and lays out subdivisions recursively, after which it fills the grid slots in the subdivisions with associated elements, either vertically or horizontally (Fig. 4d bottom). This division/splitting step is repeated until all nodes in T have been processed.
4. Finally, these computed subdivisions of elements are divided visually by split lines that show the boundaries of subdivisions enclosing groups of element glyphs. Hierarchical separators of subdivisions are more likely to be represented by stair-step lines to partition them (Fig. 4e).

P4. Draw visual links for intersecting subdivisions in the set grids: After all elements in the set grids onscreen are divided into subdivisions based on the tree hierarchies for the exclusive intersections, we can further visualize the exclusive intersections among sets using semi-transparent visual links. Since the elements are duplicated across all the set grids to which they belong, each intersecting subdivision of elements in the set grids is linked visually with its corresponding subdivisions in other set grids via an intersection link (Fig. 5). Specifically, all the same intersection subdivisions in different set grids are linked using Bézier curve contours, with a different thickness being proportional to the cardinality (the number of elements) of the exclusive intersection.

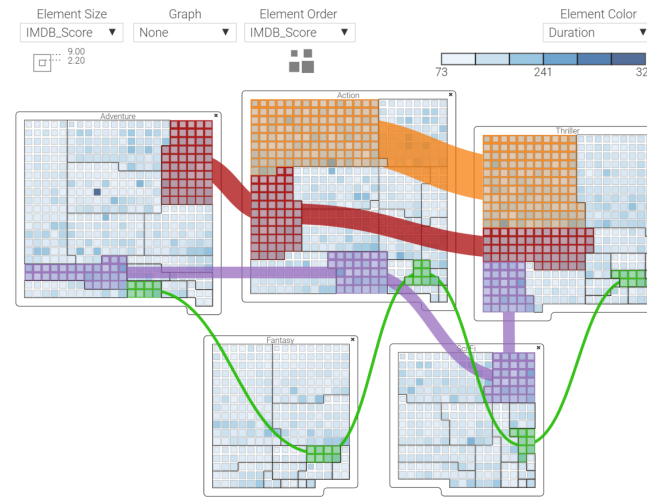


Fig. 5. Intersection links with distinct colors are rendered among the subdivisions in five sets, representing the exclusive intersections of sets. These colored links visualize that different movie elements are categorized into multiple genre sets. Eight movie elements are connected through the green links belonging to five different genres.

4.2 Interaction with Sets and Elements

GridSet supports a rich toolbox of interactive techniques to perform set-typed data analysis. Particularly, draggable set representations support powerful, yet more intuitive, interactions with sets. It also supports interactive query widgets for attributes, which can be used to dynamically highlight or filter elements and intersections.

4.2.1 Adding and Removing a Set

The user can initiate set-typed data analysis by adding sets to the Main view in order to investigate specific sets and elements. All of the sets from the dataset are listed in the Set view (Fig. 1d). The user can simply drag and drop sets from the Set view onto the Main view. The set grids added to the Main view can be dragged freely to a new position to further represent relationships of sets or improve the layout of sets and intersection links when the view becomes too crowded. Additionally, the user can remove the desired set grid in the Main view by clicking on the 'x' icon

at the top-right corner of each set grid.

4.2.2 Selecting and Linking Elements

Individual elements and intersection subdivisions in a set grid can be selected and highlighted by simply clicking on them. All the elements and intersecting subdivisions in the sets are connected through brushing and linking. Selections of elements in one set grid can be propagated to other set grids, which are then highlighted. This allows users to recognize the co-occurrence of same-element relationships in other sets. Detailed information of the selected elements is also shown on the Detail view (Fig. 1e). Additionally, it is possible for the user to highlight multiple elements by clicking on them.

Inevitably, however, many intersection links may occlude elements and sets in a crowded display as the number of sets, elements, and intersection links increases. To mitigate such visual clutter, an intersection link can be shown on demand when the analyst places their cursor over a subdivision. A double-click on the subdivision region allows the intersection link to remain on the view or be removed from the view.

4.2.3 Performing Combined Set Operations

In GridSet, users can initiate and perform common set operations, such as union, intersection, and difference, using a simple drag-and-drop gesture followed by the menu selection (Fig. 6c). For example, to perform a union operation of set Mystery and set Horror in the movie dataset, the user can drag a Mystery set grid and drop it over a Horror set grid (Fig. 6b). If the user performs a set operation by overlapping two sets and then selects specific set operations in a context menu (Fig. 6c), a resulting set grid is created from that set operation (e.g., one union set created from two overlapped sets). The resulting new set grid then replaces those two original set grids (Fig. 6d). The borders of these resulting set grids are highlighted with blue to differentiate them from the original/regular sets for the analyst.

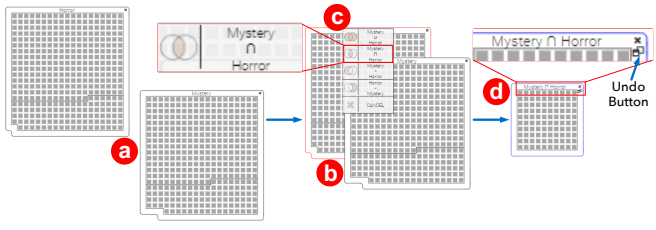


Fig. 6. Performing a set operation through a drag-and-drop gesture. (a) Drag and drop set grids on the Main view. (b) Overlap set grids and (c) Select a set operation (Intersection) from the set operation menu. (d) A resulting set grid from the intersection of two sets (Mystery ∩ Horror).

Importantly, this simple interface for set operations enables the analyst to perform chained set operations involving several sets and set operations by overlapping resulting set grids created by set operations. For instance, a resulting set grid from an intersection of Mystery and Horror movie sets (Mystery ∩ Horror) can be overlapped again with another movie set (Adventure ∪ Crime) for finding a set difference, thus producing a new set grid for chained set operations: (Mystery ∩ Horror) – (Adventure ∪ Crime). The label on the resulting set grid is automatically chosen based on the set operations. Specifically, the label of the resulting set grid includes the names of the associated sets with set theory symbols. The resulting set grids feature all the capabilities of a normal set grid, but includes an "*undo*" button at the top-right corner of the set grids (Fig. 6d). This option enables the user to reverse the last set operation conducted on a resulting set grid and revert to the original set grids.

4.2.4 Search and Query Elements and Intersections

While the ideal scenario would enable an analyst to effectively map all sets, elements, attributes, and set intersections onto a single visualization view, it is generally very difficult to support the higher scalability and complexity of a set-typed dataset using visualization techniques. To explore attributes and intersections of elements among sets, GridSet provides interactive query widgets [41]. Importantly, these query widgets can be used to dynamically highlight or filter the element glyphs based on their attribute values in the Main view. Visual query widgets, such as the range slider (Fig. 7a & 1c), enable users to rapidly adjust query parameters and immediately update filtered results on the Main view. The elements within the set grids can be visually highlighted or un-highlighted based on query results (Fig. 7).

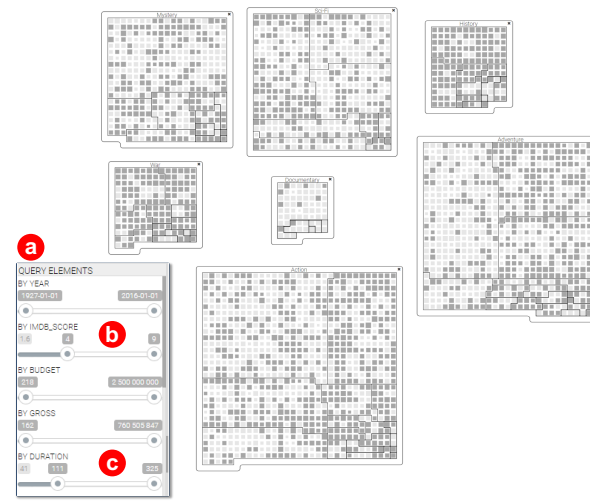


Fig. 7. Query results for the elements in a movie dataset. (a) Using the Query view, the Movies are queried based on two conditions: (b) The IMDb rating is greater than 4; and (c) Running time is longer than 111 minutes. The movie elements that meet the conditions are highlighted in dark gray color.

GridSet provides querying widgets for different data types of attributes. On one hand, for numerical and continuous attributes, elements with specific attributes can be queried based on a desired value range by defining the maximum and minimum values using query sliders (Fig. 7b,c). On the other hand, a range of categorical and nominal attributes can also be queried using multiple check and search boxes in which users can enter specific attribute values (e.g., keywords) directly. With these widgets, multiple query conditions from attributes of the different data types can be combined and used simultaneously. It is possible to query and locate specific elements that satisfy user-defined

conditions for multiple attributes with numerical and categorical data types at the same time.

In addition, the user can query and filter particular intersections by setting the rules for the intersection degrees, as well as by specifying logical rules of sets with the slider in the Query view. Using the degree slider, the user can highlight intersection subdivisions and associated links that satisfy a particular query. For example, the user can highlight any set intersection associated with three to five sets. The user can also define a logical rule based on "AND," "OR," or "NOT" conditions to highlight intersections (i.e., the corresponding intersection links and subdivisions) in which specific sets are included (or not).

4.2.5 Creating New User-Defined Sets

GridSet provides GUI widgets (Fig. 1c) in the Query view to allow users to create a separate new set by choosing and combining the following two conditions:

A set that contains specific elements: The selected elements are listed in the "selected" tab of the Detail view. The user can create a new set in which the selected elements are included.

A set that contains elements that have specific attribute values: GridSet also allows the user to create a new set that consists of elements with a specific attribute value or range of values.

Once the new set grid featuring elements based on these user-defined conditions has been created, it can then be used as a normal set grid on the Main view; users can also perform additional set operations between the user-created sets and other existing sets.

4.2.6 Implementation

A prototype of GridSet was written in HTML5, CSS3, SVG, and JavaScript. Additionally, we employ the web-based data-visualization library, D3.js [42]. In the Query view, the sliders are implemented using the ionRanger.slider.js library, while the search boxes (for searching and querying multiple values) are implemented using Multiple-Select.js. A dataset in CSV format can be retrieved, fully loaded, processed, and visualized on the client-side. A JSON file is used to define the metadata of the dataset, such as datasource and types of the elements and attributes. The source code and datasets of GridSet are available at `http://www.cs.uah.edu/~hchung/projects/GridSet/`.

5 Use Cases

In this section, we demonstrate the utility of GridSet utilizing three different set-typed datasets. We recruited three experts (E1 to E3) with prior visual-analysis experience from two large public universities. Two domain experts—one in information science (E1) and another in political science (E2)—were asked to perform an analysis of the Militarized Interstate Dispute (MID) dataset [43] collaboratively. Separately, one visualization researcher (E3), who has conducted research on visualizations for more than four years, analyzed a combined dataset of two movie datasets. All experts conducted their analysis sessions in their individual offices. One of the experts used a MacBook display set at 2,880 by 1,980 pixels, and the others employed their desktop PCs with 4K displays.

Prior to engaging in the actual analysis, each individual took part in a short tutorial session (15 to 20 minutes) using a sample dataset; additionally, they were encouraged to ask questions about the system via phone or email while conducting the analysis.

The experts conducted open-ended analysis tasks. All of the visual properties for attributes shown/discussed in the following sections were determined solely by the experts**.** As part of the analysis, they were required to formulate and then write down any interesting questions/answers, points of interest, and analysis processes they employed during their analysis sessions. Additionally, they were asked to take screenshots supporting their answers or showing any interesting findings. After completing their analysis sessions, the experts discussed their experiences of performing the analysis using GridSet with the authors.

5.1 Militarized Interstate Dispute Data

The Militarized Interstate Dispute (MID) dataset includes data about various types of disputes or hostile behaviors between one or more countries between the years 1816 and 2010. According to the MID, a "dispute" is defined as *a threat*, *an act of aggression, the use of force toward other countries,* and *an actual war* [43]. Experts E1 and E2 were charged with understanding relationships among countries, finding interesting patterns of disputes between multiple countries, and identifying outlier events.

In the MID dataset, the authors defined 2,586 dispute events as elements, while the 196 countries that took part in one or more disputes represented the total number of sets. Each dispute element also contained a range of specific attributes, such as *start date, end date, duration, total fatalities, maximum fatalities level,* and *maximum hostility level*. This information was documented over the course of the nearly 200-year period noted above.

The two experts started their analysis sessions by organizing all countries based on certain geographical relationships and examining disputes among them (Fig. 8). Initially, they were more interested in investigating disputes among the countries in the Middle East. Throughout history, there have been numerous disputes in the Middle East—notably, conflicts between Arab countries and Israel. To externalize dispute relationships among the countries, they placed the Israel set in the center, and started to add and arrange Arab countries' sets around the Israel set (Fig. 9).

Particularly, to identify any dispute in which several countries were involved, they set the degree slider to two on the Query view and selected the "AND" condition for Israel in the set query; this highlighted all the subdivisions in set grids that met these conditions (i.e., the degree of intersection is "two" and a set intersection should include dispute elements belonging to the set Israel), which then showed the corresponding intersection links. Then, the experts spatially organized the Arab country sets in a clockwise pattern based on the country's frequency of disputes against Israel (Fig. 9).

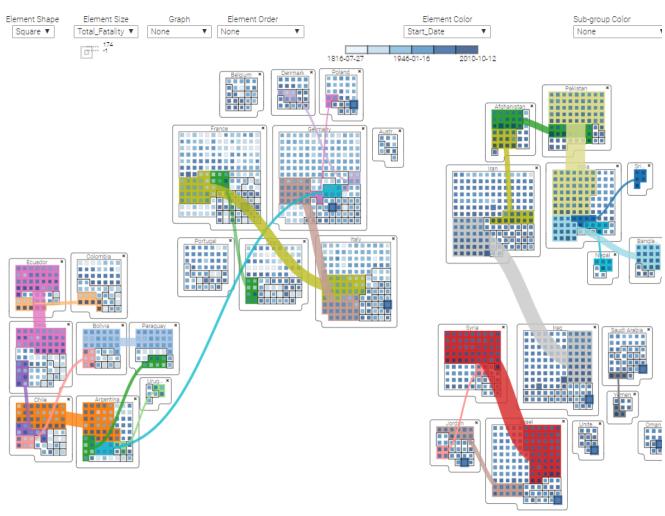


Fig. 8. E1 and E2 started their analyses by spatially arranging the country sets with their geographical neighbors to study disputes between those countries. Each element in the sets represents a dispute event. The leftmost cluster of sets shows South American neighbors; the center cluster shows the western European countries; the top-right cluster shows the countries in South Asia; and the bottom-right cluster shows the countries in the Arabian Peninsula. The links connect the intersections of three or more disputes between the countries sharing borders.

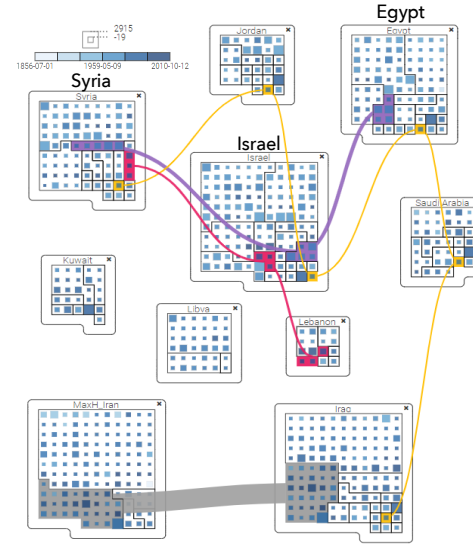


Fig. 9. E2 decided to focus on disputes in the Middle East and externalized the relationship between Israel and other Arab country sets by spatially organizing their set grids. Specifically, she situated Israel in the center and arranged other Arab countries around Israel's set grid. The glyph size was based on the duration of the dispute, and the order of elements in each grid was based on fatality level. The glyph color was based on the start date of the dispute.

This type of spatial organization for sets also enabled them to create an overview of dispute frequencies between Israel and other Arab countries. In terms of results, Syria and Egypt had a larger number of disputes with Israel in comparison to the other countries (Fig. 9 purple intersection link of three countries). E1 and E2 were also interested in further examining specific dispute information for these three countries (namely, conflicts of shorter duration, but with a higher level of fatalities) by locating specific dispute elements with the required duration range and fatality levels. For this operation, they performed a query for these three country sets (Israel, Syria, and Egypt) by selecting the "AND" condition. This "AND" condition resulted in five disputes that highlighted a subdivision and intersection link of dispute events involving the three countries (Fig. 9 purple subdivisions and links).

For these five disputes, they sought to identify a dispute with the shortest duration, but with the highest level of fatality, by sorting the elements using the following two approaches in set grids. First, they selected the element glyph size as the duration of a dispute. Second, the experts sorted the elements in the subdivision in ascending order of the level of fatality. Thus, both experts looked for the element with the smallest-sized glyph placed in the bottom far-right corner in the subdivision. As a result, they further examined two events from the bottom right corner (toward a higher level of fatality) by selecting two small-sized elements (toward a shorter duration). While the first dispute indicated that a war lasted for 23 days with a fatality level of six, the second one indicated that the duration of the war was 170 days with the same fatality level of six. Accordingly, the experts confirmed that the first dispute element, which had the shortest duration among disputes involving these three countries, had the highest level of fatality. Interestingly, they found that the first dispute was the *Third Arab-Israeli War*, which included multiple hostile interactions among Arab countries and Israel. The Detail view provided the names of all countries (e.g., Israel, Syria, Egypt, Saudi Arabia, Jordan, Kuwait, and Iraq) involved in this specific dispute.

In addition, E2 was interested in examining other conflicts in South Asia and wanted to know which countries tended to experience more disputes with higher causalities. Like the previous analysis, she first examined South Asian countries by spatially grouping them on the Main view. More specifically, she added India and Pakistan as sets onto the Main view, and then wanted to determine how neighboring countries interacted with these two countries. Accordingly, based on her personal geographical knowledge, she added Bangladesh, Sri Lanka, Afghanistan, Nepal, Bhutan, and Iran sets at their relative geographical positions on the Main view. Interestingly, for all the disputes between Pakistan (East Pakistan) and Bangladesh (West Pakistan), India was also involved by becoming an ally of Bangladesh. Their military alliance was visible by a Boolean set-dependent attribute "isSideA," which refers to "ally status."

Another interesting insight the expert identified was that in all the disputes between India and Sri Lanka, there were no other countries involved. This finding is certainly tied to the fact that Sri Lanka's nearest national neighbor is India, less than 35 miles away. Arranging the sets relative to geographical position allowed the analyst to understand this information on the Main view.

After their analysis sessions, both E1 and E2 highlighted the fact that GridSet enabled them to view overall relationships among many countries (sets) and their involved dispute events (elements). In addition, GridSet enabled them to directly examine how the attributes of the dispute elements (e.g., the duration, ally status and fatality) are co-related to the broader set relationships.

5.2 Movie Datasets

E3 analyzed a combined dataset using the IMDb Movies

dataset and the Academy Awards (AA) database [44]. The IMDb dataset includes nearly 5,050 movies across 26 different genres. The AA database contains all records of prior winners, as well as the 6,528 nominees in 42 different categories from the first Academy Awards in 1927 to the 88th in 2015. This combined dataset enabled E3 to add new attributes to each movie element, such as whether each movie had been nominated in one or more particular award categories. Additionally, each genre was defined as a set, and each movie was defined as an element that could belong to one or multiple genres.

For his first analysis, E3 wanted to understand and compare cross-genre movies according to other attributes such as *total production budget, running time, gross profits, movie rating score, release date*, etc. He started the set-typed data analysis by formulating one question (Q1): "*What is the distribution of various attributes of movies belonging to three or more genres?*"

To investigate Q1, the expert began by adding four personally favorite movie genre sets (Action, Adventure, Comedy, and Thriller) from the Set view to the Main view as shown in Fig. 10. With the query interfaces, he set the range of intersection degrees for elements from a minimum of three (*min*=3) to a maximum of four (*max*=4) using the degree-slider widget. This process revealed the overall distribution of movie elements in different intersection subdivisions. The largest number of elements (147 movies) in all intersections for degree three belonged to three genres: Action, Adventure, and Thriller (orange subdivision and link in Fig. 10). In addition, there were a total of 14 movies among the four sets that intersected simultaneously (red subdivision and link in Fig. 10). The expert then explored the detailed attributes for these cross-genre movies using the Detail view. According to his investigation, E3 concluded that a total of 230 movies, categorized in either three or four cross-genres, had been produced in the 30-year period between 1987 and 2016, adding that 80 percent of these movies featured a PG-13 rating. He also determined that the IMDb ratings for these cross-genre movies ranged between 3.80 and 6.90 (with 10 being the max score).

For the second analysis, E3 chose to conduct his analysis using the combined dataset of IMDb and AA as if he were a movie producer who was seeking to produce a movie with a greater likelihood of being nominated for a Best Picture (BP) Academy Award. Thus, certain attributes of movie production could potentially be more significant, such as genre, time of release, directing, writing, and budget. In conducting his analysis, E3 first wanted to discover cross-genres containing the highest number of nominated movies in the BP category. Using GridSet, he started his analysis by raising a new analysis question (Q2): "*Which cross-genre has been nominated most for a Best Picture Academy Award?*"

To identify the specific cross-genre, E3 first created a new set of previously Nominated Movies in the BP category (Fig. 11). Specifically, he generated a union of all major genres (from the top-ten largest genre sets on the Set view) and then isolated movie elements that were nominated for a BP award using the Visual Property menus. He first checked

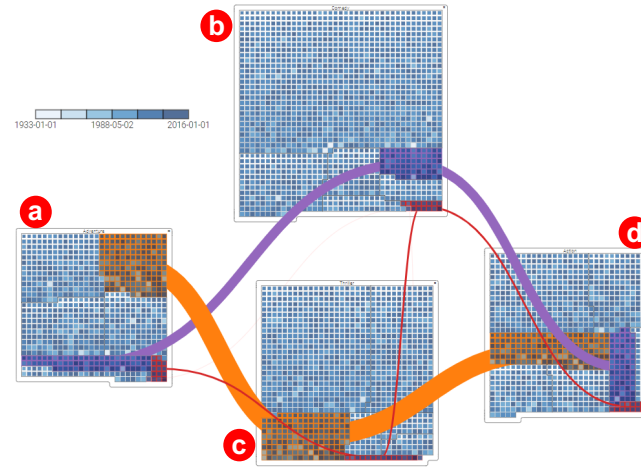


Fig. 10. Intersection links that indicate cross-genre movies among four genre sets using the query interface. E3 selected four movie genre sets: (a) Adventure, (b) Comedy, (c) Thriller, and (d) Action and intersections among the genre sets are shown based on the range of intersection degrees for elements from min=3 to max=4 using the query slider. The intersection links are represented and highlighted with links in different colors. Both the glyph size and order are based on IMDb ratings and the glyph color is based on release year. The red links indicate 14 intersecting movies across all four genres.

the cardinality of intersections (the number of elements in intersections) among more than two genre sets simply by examining the width of intersection links across sets. As shown in Fig. 11a (dark brown links across three sets), a set intersection of drama and romance (Romantic ∩ Drama) movies that had competed for a BP award represented the largest set grid with 36 movie elements. He then wanted to determine the time period for these movies by sorting elements in the subdivision of the set grids by release year. The oldest movie in the romantic-drama movie set was "*The Best Years of Our Lives*" (1946), while the most recent was "*Anomalisa*" (2015). E3 was able to confirm that romantic and dramatic movies tend to be favored by Oscar judges over other cross-genre movies during this period. Accordingly, he chose a romantic-drama as the theme of his new movie project.

E3 also wanted to determine the ideal content rating of the romantic-drama movies he was intending to produce, asking a new question (Q3): "*Which content rating of the movie would increase the possibility of a BP Oscar nomination?*" He decided to set the categorical attribute "content rating" to different colors of the element glyph and sort them by the same rating categories. Among the nominated romantic-drama movie sets (i.e., the intersection subdivision and brown link between Nominated Movies and Romantic ∩ Drama), "R-rated" movies (red glyphs; 12 movies) and "PG-13" movies (blue colored glyphs; 11 movies) had been nominated a greater number of times (Fig. 11b).

Since multiple attributes could be mapped simultaneously to visual properties of an element glyph, E3 could correlate budget with the IMDb ratings of the BP Oscar-nominated movies. However, he was unable to determine significant interrelated patterns between budget (glyph size) and the IMDb ratings (glyph color) for the romantic-drama movies he identified.

To further examine relationships between other attributes and IMDb rating, E3 created several new cross-genre

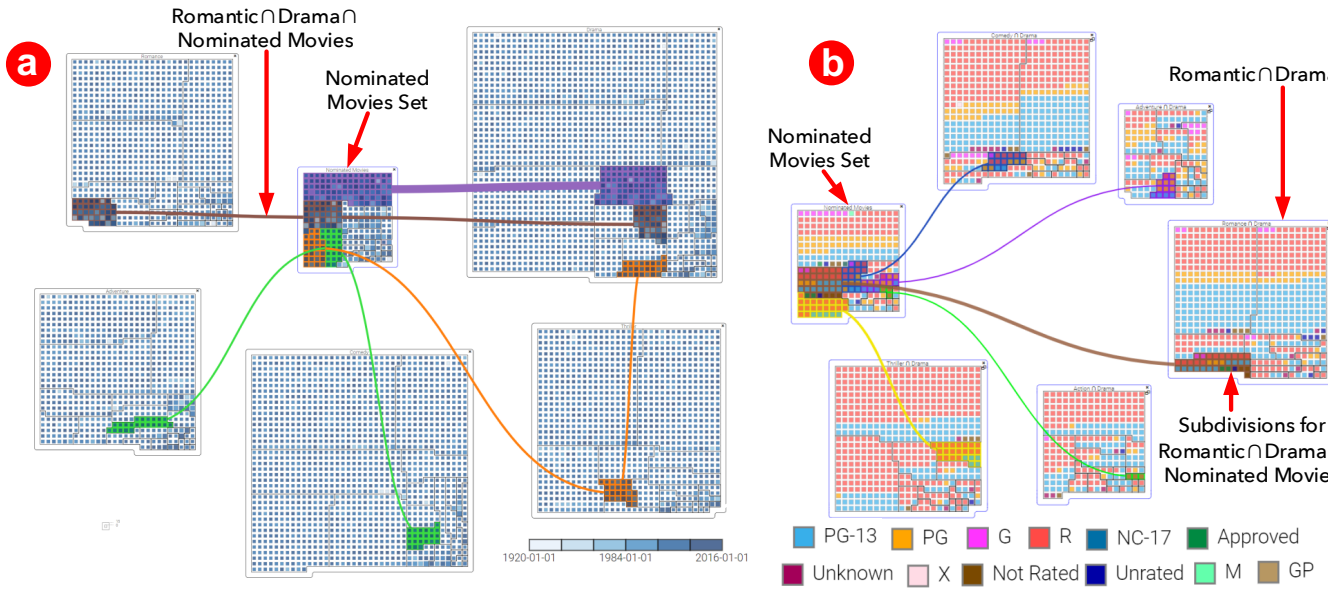


Fig.11. Analyzing a combined movie dataset: (a) The "Best-Picture-nominated movies" set in the center was compared with the five largest genre sets to determine the largest cross-genre with the greatest number of nominated movies (glyph size: total number of nominations; color: release year). (b) The categorical attribute "content rating" is represented with glyph colors and is compared with the other cross-genre movies.

movie set grids. He set the number of BP-nominated movies with the size of the element glyphs and set the color of glyphs according to IMDb rating (1 to 10 points). He also arranged the element glyphs in each subdivision in ascending order of the movies' runtimes. As shown in Fig. 12, he first filtered out movies that had not been nominated for BP using the Query slider.

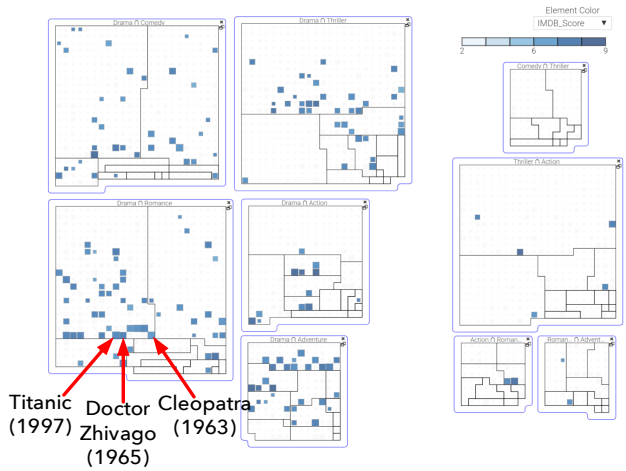


Fig. 12. Discovering cross-genre movies with multiple user-created sets. E3 created multiple cross-genre movie sets based on set intersections. He filtered and highlighted nominated movies with higher IMDb ratings in these sets and sorted them based on runtime in ascending order. He further analyzed with glyphs' visual encodings (glyph order: movie runtime, glyph size: the number of nominations, and glyph color: IMDb ratings). From this investigation, he identified three specific movies with longer running times and higher IMDb ratings.

Fig. 12 also shows that only nominated movie elements remain within each set grid after filtering. It appears that these larger, darker-colored glyphs are more populated in the lower part of each subdivision of the Romantic ∩ Drama set. This outcome confirms that most of the nominated movies had very high IMDb ratings (dark blue color) with multiple nominations (large glyph size). Additionally, the movies were sorted based on runtime in ascending order. Thus, based on the glyphs' visual properties (larger dark blue glyphs in Fig. 12) and their location (bottom right) in the Romantic ∩ Drama set, the expert determined that movies with longer running times appeared to have a better chance of being nominating for a BP Oscar and generating higher IMDb ratings.

Lastly, the visual patterns afforded by GridSet indicated that a number of highly rated movies tended to have longer running times. As shown in Fig. 12, the nominated movies "*Cleopatra*" (1963), "*Doctor Zhivago*" (1965), and "*Titanic*" (1997) that appeared in the set grid of Romantic ∩ Drama had longer running times (more than three hours), while also earning high IMDb ratings (more than seven points). Importantly, GridSet also allowed E3 to identify a small number of outlier movies against these dominant BP Oscar patterns.

Based on his findings from the analysis, E3 concluded that if a film producer is seeking to produce a cross-genre movie that is more likely to receive an Academy Award for Best Picture, the producer should consider (a) making a romantic-drama, (b) producing a movie that is on the long side (longer than three hours), and (c) ensuring that the movie is either "R-rated" or "PG-13."

6 Discussions and Future Work

In this section, we discuss our observations of the salient usage of GridSet during our use cases, review limitations, and suggest future work.

6.1 Spatial Organizations of Sets

In reviewing findings from our use cases, we observe that all three experts spatially organized different set grids to better understand certain set relationships. This straightforward interaction with visual objects helped them externalize and augment their understanding of set relationships. As the analysis progressed, however, the layout of sets began to take on more semantically meaningful forms.

For example, we observed that the experts used this feature to better understand frequencies of disputes, geographical regions, and interrelated movie genres.

Particularly, both E1 and E2 relied on the spatial organization of the set grids to facilitate their understanding of different relationships between sets (countries) and their elements (disputes). They spatially arranged the Middle East country sets in a clockwise layout based on a country's frequency of disputes against another country (Fig. 9). They verbalized that this organized set grid capability helped them assess the occurrence and severity of disputes between one country and its neighbors. Additionally, E2 placed set grids of South Asian countries at the relative positions of the countries on a screen based on her geographical knowledge. She employed this strategy to determine if geographical location had any influence on the different types of disputes among countries. Similarly, E3 arranged and clustered more relevant sets based on their intersections (see supplementary material for an example). Most interestingly, E3 continuously rearranged set grids as needed to understand other information and to answer different questions throughout his analysis, as shown in Figs. 11 and 12. He stated that this process facilitated his understanding of relationships among sets, while at the same time reducing visual clutter.

6.2 Using User-Defined Sets

In our use cases, we also observed that the experts created new sets for their analyses, and that these newly created sets assisted the experts in performing more complex set operations that generated useful information. For instance, E3, as a would-be movie producer, began his analysis by creating a new set for "nominated movies for Best Picture." Instead of depending solely on intersection links among the existing sets, he wanted to maintain a separate set for the Best-Picture-nominated movies to facilitate further investigation of this set. For the BP-nominated movies, he first created a single union set that encompassed all movie elements in all major genres, and then filtered out the movies that were not nominated for BP by using the slider in the Query view. It should be noted that such user-created sets remain on the Main view and can be referred to by the analyst as needed.

Considering that a set grid may be involved in multiple set operations and conditions, we were concerned that it might be unnecessarily confusing for individuals to readily recall and compare those set operations and conditions for their analysis needs. In point of fact, when we discussed this issue with E1, he mentioned that the current interface does not allow viewing the entire sequence of set operations applied to the set grid very easily.

To facilitate revisiting an earlier result for a given chained set operation, we plan to incorporate a provenance view and interface with the Detail view. This view can be accessed simply by clicking an additional tab in the Detail view, which will show the entire sequence of the combined set operations applied to the selected sets. This view will also provide features for saving, reviewing, and reproducing a user-defined set grid resulting from a particular set operation over its entire history.

6.3 Scalability Issues

In this section we discuss three interrelated scalability issues that need to be addressed in GridSet. We also present potential remedies for scaling the three factors and suggest future research avenues to further advance the potential of GridSet.

6.3.1 Number of Attributes

In GridSet, it still remains difficult to visualize more than three types of attributes simultaneously, since the visual properties possible for each glyph are limited to color, shape, and size. Many of the prior visualization approaches [45, 46, 47, 29, 48] for multivariate/multi-dimensional data have emphasized glyph-based visual representations. One potential approach for addressing the scalability of multiple attributes is to further utilize element glyphs by generating nested visual representations including the pixel-oriented visualizations and small-multiple displays.

A pixel-oriented visual representation can map multiple attributes to different sets of colored pixels within element glyphs. Specifically, available pixels of a glyph are partitioned into n sub-windows in different layouts (e.g., a matrix layout, a line-by-line layout, and a column-by-column layout), with each of the n attributes capable of being mapped to each sub-window [49]. Based on different pixel patterns, interesting correlations between attributes and intersections/sets can be revealed. However, these pixel-based approaches rely on categorical colors that are typically based on a different hue or saturation level. Thus, it may be difficult for a user to distinguish multiple attributes mapped to many colors [50].

In addition, if there are available pixels in a glyph, we can encode a series of attribute values associated with an element into a small-multiple display or visual representation. For instance, a small-multiple display for a glyph visualizes more than three different attributes by binning the attribute values in a small bar graph, line chart, and radar chart. However, it would be difficult to map multiple attributes onto visual representations since one is allocated only a small portion of the glyph space. If a visualization representation consists of multiple shapes, the shapes might even obstruct one another and create artifacts due to the small glyph space.

In this regard, instead of relying solely on colored pixels and visualization representations nested in each glyph, we will incorporate additional multi-dimensional visualization with the Main view. Particularly, we will be able to use a parallel coordinate to visualize many attributes associated with each element. Each of the n attributes corresponds to one of n vertical axes, and each element is represented as a polyline linked across these axes. If a single element or a group of elements in a set grid or intersection subdivision/link are selected, corresponding polylines will be highlighted or clustered on the parallel coordinate.

6.3.2 The Number of Elements

As documented in the supplementary materials, we were able to visualize two large datasets related to our use cases on the 4K display without difficulty: (a) 114 sets with 6,315

unique elements (15 × 15 pixels for each), and (b) 2,561 disputed events (elements) among 392 countries (sets). Notably, glyphs of a smaller pixel size (less than eight pixels) could still be distinguished on both display resolutions. However, if the size of a glyph is too small, we would be unable to represent variable shape sizes or visual encodings for mapping multiple attribute values. Thus, we determined empirically that glyphs need to be approximately 15 × 15 pixels wide in order to represent reasonable visual discretion of quantitative data without zooming.

To improve the scalability of elements shown in the Main view, we will explore new visualization techniques that combine two principal features: an aggregate visual representation and navigation strategies extending prior work [51, 52]. Specifically, to reduce the number of elements displayed in each set grid, more than four adjacent elements can be aggregated into a single glyph. For instance, these elements can be replaced with a single glyph in a set grid. The size of the glyph can be determined by the number of the aggregated elements proportionally in each set grid.

Additionally, attribute values associated with the nearby elements can be also aggregated in a visual representation with statistical summaries (e.g., mean, minimum, maximum, standard deviations, etc.). For example, numerical attributes from the four elements can be aggregated in a glyph of a small histogram (binning the attribute values from the four elements in intervals), a min/max range, or Tukey box [48].

In addition, we will explore a new focus+context technique to permit the user to better explore these small aggregated visual representations. By hovering a mouse cursor over an aggregated glyph, the glyph will expand a region directly within the set grid. Also, a visual representation in the glyph can be enlarged in this way. An aggregate visual representation nested within this type of hovered glyph, which plays the role of focus, is thereby enlarged and magnified, revealing desired details. To spatially accommodate the expanded focal region for the aggregated glyph, the surrounding glyphs must be partially pushed back by distorting or warping the layout of a set grid.

6.3.3 The Number of Intersections

In GridSet, the existence of many intersections among sets may potentially introduce visual complexity and edge-tracing problems for the intersection links due to a large number of overlapping links and subdivisions of set grids.

Because the intersection links are rendered on top of set grids, small subdivisions in set grids, which become the source and target nodes of the link, can become cluttered with many links (see the supplementary materials for examples). As evidenced in our use cases, the experts commented that when there were too many single and two-element subdivisions in set grids, they prevented the experts from keeping track of intersections across multiple set grids. As a result, they often needed to hover over the links to highlight and trace the links and subdivisions. The principal reason for this problem was because many such links tended to obscure other links and small subdivisions/element glyphs.

Thus, our future work will focus on ensuring the enhanced edge-node readability and legibility of a large number of links (edges) and elements/subdivisions (nodes) through three strategies: (a) reducing overlaps between the subdivisions and links, (b) making links visually more distinct, and (c) improving spatial layouts.

Identifying effective intersection links can be formalized as optimization problems, whereby criteria for the effective link will need to be satisfied. Extending Steinberger et al.'s context-preserving visual link approach [33], we will determine the intersection links according to the pertinent criteria, with the goal of minimizing the occlusion of set grids and subdivisions overlapped by intersection links.

Additionally, instead of assigning a link color from a predefined color palette, more distinguishable and perceivable link colors can be selected adaptively based on color information from the intersection link's neighbor pixels in the Main view.

In our use cases, our experts frequently reorganized and clustered more related sets in closer proximity to improve the layout of the intersection links and sets. It appears that this approach can mitigate the occlusion and unwanted cluttering associated with an excessive number of intersection links, as shown in Untangling Euler Diagrams [18]. In our movie-related use case, award categories, which are more highly (semantically) related, shared the same nominees and their intersections, and organizing these sets could reduce visual complexity caused by the intersection links. (See supplementary materials.)

6.4 Comparison with Two State-of-the-Art Set Visualizations

We further compare in detail two recent set-typed visualizations with GridSet. GridSet's visual representation is closely related to a combination of ideas supported by Untangling Euler Diagrams [18] and OnSet [16].

Untangling Euler Diagrams: Untangling Euler Diagrams (UED) uses rectangular shapes to represent individual sets, while set intersections are visualized by connecting between duplicated (dupED model) or non-duplicated (ComED) elements using contours. Both UED and GridSet can duplicate elements within different set boundaries to avoid overlapping set regions. GridSet uses semi-transparent contours among duplicated elements to visualize the intersection of elements in different sets. One potential problem of connecting duplicated elements with links is that it may be difficult for the user to discern individual intersection links when a large number of links are shown on the screen. UED deals with this type of screen clutter by using nested and bundled contours. As such, UED's approach is useful for making many intersecting elements more readable. However, the number of elements in each set may not be scalable for the nested approach, since each set representation needs to reserve certain screen space to show a number of elements. In contrast, Gridset allows users to exploit screen space by placing set grids at any location around the screen, while still enabling users to see intersections among sets. Within each set grid, space for elements can be organized efficiently by the Grid Treemap

layout.

OnSet: OnSet shares visual similarities with GridSet in that both visualizations represent elements individually with a square glyph in each set object. However, there are key differences between the two set visualization techniques that must be noted. Specifically, OnSet allocates cells for all the elements in the universal set within each set object; accordingly, each element glyph represents a binary state of either on or off in each of the sets in the side-by-side layout. This approach is very useful for comparing the occurrence of specific elements in each set, based on the universal set. However, each set object may require much space if the size of the universal set is very large.

Conversely, in GridSet the element glyphs can vary in terms of their size, shape, and color based on different attribute types and values. GridSet can also create icons or element attributes that can be placed on top of these element glyphs to represent a variety of attributes. Most notably, in GridSet each set grid contains only the elements that belong to the respective set without securing space for the other elements in the universal set. Abandoning the universal element positions in each set grid enables the user to compare the size of each set. This approach also facilitates ordering the elements in each set grid based on attribute values.

For viewing set intersections with OnSet, multiple set objects can be overlaid and element cells are, in turn, encoded in saturations of a color in a heatmap, thereby showing the frequency of occurrence in the overlaid sets. Conversely, GridSet's glyphs within different set grids are linked using semi-transparent contours. This visual link approach also allows users to organize set grids around the screen space while still revealing relationships among the sets.

7 Conclusion

In this article, we presented GridSet—a novel visualization technique for visualizing individual sets, a large number of their elements, and associated attributes. Visualizing individual components of set-typed data can help users explore a dataset at both the set and element levels simultaneously. GridSet facilitates intuitive interactions for performing combined set operations and creating new sets; it also supports the visual spatialization of sets around the screen space to externalize certain semantic relationships of sets.

Our use cases demonstrated that GridSet's techniques were deployed successfully by the experts who conducted complex set-typed data analyses. Indeed, the experts were able to both view overall relationships among sets, as well as directly access detailed information about individual elements and their associated attributes.

Overall, we believe that the suite of techniques supported by GridSet will open up a new space for set visualization techniques that can empower researchers to explore and analyze various types of set-typed data.

References

[1] B. Alsallakh, L. Micallef, W. Aigner, H. Hauser, S. Miksch, and P. Rodgers, "The State-of-the-Art of Set Visualization," *Computer Graphics Forum*, vol. 35, no. 1, pp. 234-260, 2016.

[2] E. L. Hutchins, J. D. Hollan, and D. A. Norman, "Direct manipulation interfaces," *Human-computer interaction*, vol. 1, no. 4, pp. 311-338, 1985.

[3] J.-D. Fekete and C. Plaisant, "Interactive information visualization of a million items," *The Craft of Information Visualization:* Elsevier, pp. 279-286, 2003.

[4] D. Park, S. M. Drucker, R. Fernandez, and N. Elmqvist, "ATOM: A Grammar for Unit Visualizations," *IEEE Trans. Visualization and Computer Graphics*, vol. 24, no. 12, pp. 3032-3043, 2018.

[5] D. A. Keim, "Designing pixel-oriented visualization techniques: Theory and applications," *IEEE Trans. Visualization and Computer Graphics*, vol. 6, no. 1, pp. 59-78, 2000.

[6] T. . Schreck, D. Keim, and F. Mansmann, "Regular treemap layouts for visual analysis of hierarchical data," *Proc. ACM Conf. on Computer Graphics*, pp. 183-190, 2006.

[7] C. Andrews, A. Endert, and C. North, "Space to think: large high-resolution displays for sensemaking," *Proc. ACM CHI 2010*, pp. 55-64, 2010.

[8] C. Collins, G. Penn, and S. Carpendale, "Bubble sets: Revealing set relations with isocontours over existing visualizations," *IEEE Trans. Visualization and Computer Graphics*, vol. 15, no. 6, pp. 1009-1016, 2009.

[9] J. Vihrovs, K. Prūsis, K. Freivalds, P. Ručevskis, and V. Krebs, "An inverse distance-based potential field function for overlapping point set visualization," *Proc. IEEE IVAPP'14*, pp. 29-38, 2014.

[10] K. Dinkla, M. J. van Kreveld, B. Speckmann, and M. A. Westenberg, "Kelp diagrams: Point set membership visualization," *Computer Graphics Forum*, vol. 31, no. 3, pp. 875-884, 2012.

[11] D. Oelke, H. Strobelt, C. Rohrdantz, I. Gurevych, and O. Deussen, "Comparative exploration of document collections: a visual analytics approach," *Computer Graphics Forum*, vol. 33, no. 3, pp. 201-210, 2014.

[12] Z. S. Michael, "BiblioViz: A System for Visualizing Bibliography Information," *Proc. the Asia-Pacific Symp. on Information Visualisation*, pp. 93-102, 2006.

[13] J. Stasko, C. Gorg, and Z. Liu, "Jigsaw: supporting investigative analysis trhough interactive visualization," *Information visualization*, vol. 7, pp. 118-132, 2008.

[14] K. Misue, "Drawing bipartite graphs as anchored maps," *Proc. the Asia-Pacific Symp. on Information Visualisation*, pp. 169-177, 2006.

[15] B. Kim, B. Lee, and J. Seo, "Visualizing set concordance with permutation matrices and fan diagrams," *Interacting with computers*, vol. 19, no. 5-6, pp. 630-643, 2007.

[16] R. Sadana, T. Major, A. Dove, and J. Stasko, "Onset: A visualization technique for large-scale binary set data," IEEE transactions on visualization and computer graphics, vol. 20, no. 12, pp. 1993-2002, 2014.

[17] P. Simonetto, D. Auber, and D. Archambault, "Fully automatic visualisation of overlapping sets," *Computer Graphics Forum*, vol. 28, no. 3, pp. 967-974, 2009.

[18] N. H. Riche and T. Dwyer, "Untangling euler diagrams," *I IEEE Trans. Visualization and Computer Graphics*, vol. 16, no. 6, pp. 1090-1099, 2010.

[19] B. Alper, N. Riche, G. Ramos, and M. Czerwinski, "Design study of linesets, a novel set visualization technique," *IEEE Trans. Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2259-2267, 2011.

[20] N. Henry, J.-D. Fekete, and M. J. McGuffin, "NodeTrix: a hybrid visualization of social networks," *IEEE Trans. Visualization and Computer Graphics*, vol. 13, no. 6, pp. 1302-1309, 2007.

[21] L. Micallef, P. Dragicevic, and J.-D. Fekete, "Assessing the effect of visualizations on bayesian reasoning through crowdsourcing," *IEEE Trans. Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2536-2545, 2012.

[22] A. Lex, N. Gehlenborg, H. Strobelt, R. Vuillemot, and H. Pfister, "UpSet: visualization of intersecting sets," *IEEE Trans. Visualization and Computer*

Graphics, vol. 20, no. 12, pp. 1983-1992, 2014.
[23] M. A. Yalcin, N. Elmqvist, and B. B. Bederson, "AggreSet: Rich and scalable set exploration using visualizations of element aggregations," *IEEE Trans. Visualization and Computer Graphics*, vol. 22, no. 1, pp. 688-697, 2016.
[24] W. Freiler, K. Matkovic, and H. Hauser, "Interactive visual analysis of set-typed data," *IEEE Trans. Visualization and Computer Graphics*, vol. 14, no. 6, 2008.
[25] B. Alsallakh, W. Aigner, S. Miksch, and H. Hauser, "Radial sets: Interactive visual analysis of large overlapping sets," *IEEE Trans. Visualization and Computer Graphics*, vol. 19, no. 12, pp. 2496-2505, 2013.
[26] B. Alsallakh and L. Ren, "Powerset: A comprehensive visualization of set intersections," *IEEE Trans. Visualization and Computer Graphics*, vol. 23, no. 1, pp. 361-370, 2017.
[27] S. Drucker and R. Fernandez, "A unifying framework for animated and interactive unit visualizations," Technical Report MSR-TR-2015-65, Microsoft Research, 2015.
[28] D. Oelke, H. Janetzko, S. Simon, K. Neuhaus, and D. A. Keim, "Visual Boosting in Pixel-based Visualizations," *Computer Graphics Forum*, vol. 30, no. 3, pp. 871-880, 2011.
[29] D. A. Keim, M. C. Hao, U. Dayal, and M. Hsu, "Pixel bar charts: a visualization technique for very large multi-attribute data sets," *Information Visualization*, vol. 1, no. 1, pp. 20-34, 2002.
[30] H. Chung, Y. J. Cho, J. Self, and C. North, "Pixel-oriented Treemap for multiple displays," *Proc. IEEE VAST 2012*, pp. 289-290, 2012.
[31] H. Chung and C. North, "SAViL: Cross-display visual links for sensemaking in display ecologies," *Personal and Ubiquitous Computing*, Vol. 22, Issue 2, pp 409–431, 2017.
[32] M. Waldner, W. Puff, A. Lex, M. Streit, and D. Schmalstieg, "Visual links across applications," *Proc. Graphics Interface 2010*, pp. 129-136, 2010.
[33] M. Steinberger, M. Waldner, M. Streit, A. Lex, and D. Schmalstieg, "Context-preserving visual links," *IEEE Trans. Visualization and Computer Graphics*, vol. 17, no. 12, pp. 2249-2258, 2011.
[34] B. Shneiderman, "Direct manipulation for comprehensible, predictable and controllable user interfaces," *Proc. ACM Intelligent user interfaces* 1997, pp. 33-39, 1997.
[35] L. Wilkinson, "Exact and approximate area-proportional circular Venn and Euler diagrams," *IEEE Trans. Visualization and Computer Graphics*, vol. 18, no. 2, pp. 321-331, 2012.
[36] D. B. Neale et al., "Decoding the massive genome of loblolly pine using haploid DNA and novel assembly strategies," Genome biology, vol. 15, no. 3, p. R59, 2014.
[37] A. D'hont et al., "The banana (Musa acuminata) genome and the evolution of monocotyledonous plants," *Nature*, vol. 488, no. 7410, pp. 213, 2012.
[38] C. D. Wickens, J. G. Hollands, S. Banbury, and R. Parasuraman, *Engineering psychology and human performance*. Psychology Press, 2015.
[39] D. Fisher, I. Spiridonova, R. Fernandez, and S. Drucker. "SandDance." https://www.microsoft.com/en-us/research/project/sanddance/. 2018.
[40] L. Wilkinson, "Statistical methods in psychology journals: Guidelines and explanations," *American psychologist*, vol. 54, no. 8, p. 594, 1999.
[41] C. Ahlberg, C. Williamson, and B. Shneiderman, "Dynamic queries for information exploration: An implementation and evaluation," *Proc. ACM CHI 1992*, pp. 619-626, 1992.
[42] M. Bostock, V. Ogievetsky, and J. Heer, D³ data-driven documents. *IEEE Trans. visualization and computer graphics*, *17*(12), pp. 2301-2309, 2011.
[43] D. M. Jones, S. A. Bremer, and J. D. Singer, "Militarized interstate disputes, 1816–1992: Rationale, coding rules, and empirical patterns," *Conflict Management and Peace Science*, vol. 15, no. 2, pp. 163-213, 1996.
[44] Academy of Motion Picture Arts and Sciences, "THE OFFICIAL ACADEMY AWARDS DATABASE." https://www.oscars.org/oscars/awards-databases-0. 2019.
[45] J. M. Chambers, *Graphical methods for data analysis*. CRC Press, 2018.
[46] H. Wickham, H. Hofmann, C. Wickham, and D. Cook, "Glyph-maps for visually exploring temporal patterns in climate data and models," *Environmetrics*, vol. 23, no. 5, pp. 382-393, 2012.
[47] M. O. Ward, "A taxonomy of glyph placement strategies for multidimensional data visualization," *Information Visualization*, vol. 1, no. 3-4, pp. 194-210, 2002.
[48] N. Elmqvist, T.-N. Do, H. Goodell, N. Henry, and J.-D. Fekete, "ZAME: Interactive large-scale graph visualization," *Proc. IEEE Pacific Visualization Symposium*, pp. 215-222, 2008.
[49] D. A. Keim, "Pixel-oriented visualization techniques for exploring very large data bases," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 58-77, 1996.
[50] S. Haroz and D. Whitney, "How capacity limits of attention influence information visualization effectiveness," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 12, pp. 2402-2410, 2012.
[51] S. Eick, J. L. Steffen, and E. E. Sumner Jr, "Seesoft-a tool for visualizing line oriented software statistics," *IEEE Transactions on Software Engineering*, no. 11, pp. 957-968, 1992.
[52] R. Rao and S. K. Card, "The table lens: merging graphical and symbolic representations in an interactive focus+ context visualization for tabular information," in *Proc. of ACM CHI 1994*, pp. 318-322, 1994.

Haeyong Chung is an assistant professor of Computer Science at University of Alabama in Huntsville. His research interests include visual analytics, information visualization, and supporting sensemaking with display ecologies and large high-resolution displays. He received his Ph.D. in Computer Science from Virginia Tech.

Santhosh Nandhakumar is a Ph.D. student at University of Alabama in Huntsville. His research interests are in data visualization and visual analytics, focusing on complex data, including text documents and set-typed data. Nandhakumar earned his Bachelor's in Information Technology from Anna University, Chennai, India. Before undertaking his Ph.D. studies, he worked as a project engineer at WIPRO Technologies.

Seungwon Yang received his BS, MS, and Ph.D. degrees in Computer Science from Virginia Tech. He is an assistant professor with a joint position in both the School of Information Science and in the Center for Computation and Technology at Louisiana State University (LSU). His research interests include social media analysis, crisis informatics, and information visualization.