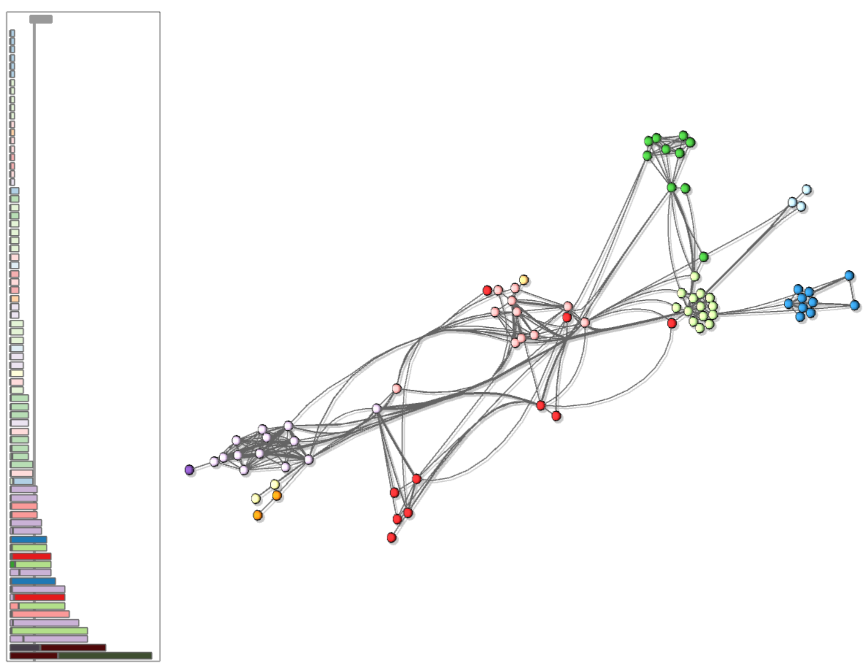
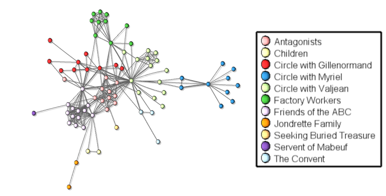
Driving Interactive Graph Exploration Using 0-Dimensional Persistent Homology Features

A. Suh[1], M. Hajij[1], B. Wang[2], C. Scheidegger[3], and P. Rosen[1]

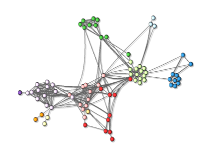
[1]University of South Florida, Tampa FL, USA
[2]University of Utah, Salt Lake City UT, USA
[3]University of Arizona, Tucson AZ, USA



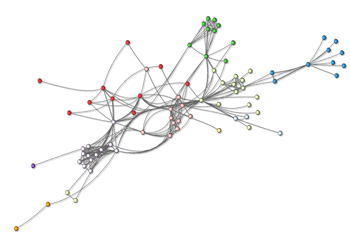
(a) Both Contraction and Repulsion



(b) Standard Force-Directed Layout



(c) Contraction Only



(d) Repulsion Only

Figure 1: *(b) The Lés Miserables graph is drawn using a force-directed layout. Our approach provides two mechanisms for interacting with the force-directed layout using the persistence barcodes to the left. (c) The first mechanism contracts nodes of the graph with low significance or* persistence. *(d) The second mechanism causes nodes separated by topologically significant events to repulse from one another. (a) When combined, this approach allows interactively controlling the graph layout based upon underlying topological structure.*

Abstract

Graphs are commonly used to encode relationships among entities, yet, their abstractness makes them incredibly difficult to analyze. Node-link diagrams are a popular method for drawing graphs. Classical techniques for the node-link diagrams include various layout methods that rely on derived information to position points, which often lack interactive exploration functionalities; and force-directed layouts, which ignore global structures of the graph. This paper addresses the graph drawing challenge by leveraging topological features of a graph as derived information for interactive graph drawing. We first discuss extracting topological features from a graph using persistent homology. We then introduce an interactive persistence barcodes to study the substructures of a force-directed graph layout; in particular, we add contracting and repulsing forces guided by the 0*-dimensional persistent homology features. Finally, we demonstrate the utility of our approach across three datasets.*

(12/2017).Categories and Subject Descriptors (according to ACM CCS): Visualization [Human-centered computing]: Visualization techniques—Graph drawings

1. Introduction

Graphs are ubiquitous for representing complex relationships between individuals or objects and are often the byproduct of modeling social networks, energy grids, computer networks, brain neural connections etc. The abstractness of graphs provides significant flexibility in visualization. However, their interconnectedness often leads to confusing visualizations that appear as "hairballs". A good graph visualization should present *structure* quickly and clearly, and it should guide further *exploration* of the underlying data.

A widely used graph visualization technique is the node-link diagram, which relies on directly drawing nodes and edges. A key element of these techniques are the layout methods that place nodes on the display (semi-)automatically. The problem of automatic graph layout has a rich literature, in which many approaches focus on finding an embedding of the graph by optimizing a readability metric, such as symmetry of the graph, length of the edges, or number of edge crossings [EKLN03]. A significant advancement is the realization that the use of derived information, such as node rank, graph distance, or approximate clustering, could improve the graph layouts [GKNV93, Noa07]. However, many of these produce static imagery and lack interactive exploration capabilities.

When considering interactive graph layouts, likely the most commonly used (though not necessarily the *best*) method are force-directed or mass-spring layouts [FR91a], which rely on converting the graph into a physical system of attractive springs and repulsive forces that iteratively minimize an energy function. These systems rely upon local relationships to reveal the overall shape in the graph. The result is an interactive method that shows structure in certain graphs, particularly sparse ones. However, this approach can, and often does, cause unrelated structures to overlap or cross paths, making them difficult to differentiate. Force directed layouts provide the capacity to address this through interaction. Unfortunately, this is most often in the form of click-and-dragging of individual nodes, which is highly ineffective for larger graphs.

This paper addresses the graph drawing challenge by leveraging topological data analysis (TDA) [Car09] as derived information for interactive graph visualization. Our approach uses topological features—captured by *persistent homology* (PH) [EH08b, Ghr08a] to interactively study substructures within a force-directed graph layout. By topological features, we do not mean the arrangement or configuration of nodes and edges, but instead the 0-dimensional homology groups of a metric space structure induced by a certain distance defined on the graph [EH10a].

Persistent homology has a few key qualities that make it ideal for this application. First, persistent homology can extract features at any scale, without the need to pick any parameters. Second, it is able to quantify and rank topological events according to their significance (*persistence*). The set of all topological events can be represented using a *persistence barcodes* that is used to interactively manipulate the graph indirectly via its *structure* instead of the traditional approach of direct node manipulation.

In brief, our approach works as follows. We extract the topological features of a graph using persistent homology; such features are sorted by their significance (or persistence). We employ a selected topological features in two different ways throughout the visualization pipeline. First, a selected feature is used to cluster/partition the graph into two subsets, which are repulsed from one another. Second, a selected feature could help to create a strong attractive force between the two nodes that create the feature. The user may select multiple features, adding as many additional attractive or repulsive forces as necessary for exploring the graph.

Figure 1 gives an example. The Lés Miserables graph is drawn initially (Figure 1(b)) with a standard force-directed layout. On the left of Figure 1(a), the persistence barcode shows the topological features of the graph as identified by persistent homology. These bars are used to manipulate the graph. In Figure 1(c) a set of bars are selected to contract their associated graph features. In Figure 1(d) a different set of bars are selected that cause sets of nodes to repulse from one another. In Figure 1(a) the contraction and repulsion are combined into the final graph layout.

In summary, our contributions are:

- We use 0-dimensional topological features for interactive graph drawing;
- We provide an interactive interface based upon persistence barcodes, that enables explorations of the graph structure beyond the individual nodes;
- In particular, we introduce a modification of the forces involved in force-directed layouts to be adaptive with respect to the topological structures selected by the user.

2. Prior Work

2.1. Graph Visualization

The prior work in graph visualization is quite vast. Our treatment will focus on approaches for drawing node-link diagrams. For a broader and more detailed overview, see von Landesberger et al.'s survey [VLKS*11].

Most graph visualization systems, including Gephi [BHJ09], NodeXL [HSS10], and Graphviz [EGK*02], use variations on node-link visualizations to display graphs.

The first automated techniques for laying out node-link diagrams is Tutte's barycentric coordinate embedding [Tut63], followed by linear programming techniques [GKNV93], force-directed/mass-spring embeddings [FR91a, Hu05], embeddings of the graph metric [GKN05], and linear-algebraic properties of the connectivity structures [BP07, KHKS12, Kor03, KCH02].

The problem of visual clutter is quite challenging with node-link diagrams. For denser graphs, edge bundling can reduce visual clutter by routing graph edges to the same portion of the screen [HVW09a]. In terms of quality, divided edge bundling [SHH11] produces high-quality results, while hierarchical edge bundling [GHNS11] scales to millions of edges.

Other visual metaphors have been proposed to reduce clutter, ranging from relatively conservative proposals, such as replacing nodes with motifs [DS13] or modules [DRMM13], to more aggressive forms, such as variants of matrix diagrams [DWvW12] and abstract displays of graph statistics [KMSH12].

When displaying a large dataset, it is natural to question the hard visual limits for simplified networks. Popular approaches such as pixel-based visualizations [Kei00, KSS07] encode large amounts of data within small rectangles or display pixels. Space-filling curves have also been used to build pixel-based graph visualizations [MM08]. Furthermore, using visual boosting [OJS*11] specifically tailored towards network data may further reveal hidden information.

Research into interactive exploration in node-link diagrams has been fairly limited. In addition to interacting directly with nodes in force-directed layouts [FR91a, Hu05], interactive approaches have included hierarchical layout construction [HH91], fisheye lenses [SB92], interactive refinement of automatic layouts [FW94], or constraint-based optimization [RMS97].

Unlike much of the prior work, our approach for reducing visual clutter focuses on enabling graph exploration by interactively manipulating the graph structure, as defined by persistent homology. This done by manipulating the forces applied to force-directed layouts in an intuitive way, based upon user selected persistent homology features.

2.2. Topological Data Analysis of Networks

Persistent homology is an emerging tool in studying complex networks [DPS*12, ELY12, HMR09, PSDV13a, PSDV13b] including collaboration [BG14, CH13] and brain networks [CRS15, DMFC12, LCK*11a, LCK*11b, LKC*12a, LKC*12b, PRMM15]. Reeb graphs have been studied in the context of time-varying spatial data [EHMP04].

More recently in the visualization community, persistent homology has been used for graph analysis looking at clique communities [RFLL17] and time-varying graphs [HWSR17]. However, we are not aware of work using persistent homology directly in graph drawing.

A key idea of using topological techniques in the study of graphs is that we can obtain a *compressed description* of the data *invariant under small deformations*, making them less sensitive to noise and other small variations in data (e.g. the loss a of low weight edge does not really change the graph) [LSL*13]. This lends itself to focusing on only the most important structures of the graph.

3. Extracting the Topological Features of a Graph

To extract topological features from a graph, we apply persistent homology to a metric space representation of the graph [HWSR17]. See [EH08a] for an introductory survey on persistent homology and [EH10b] for a formal treatment.

3.1. Persistent Homology of a Graph

In our context, we care about the 0-dimensional topological features of a graph. Such features, formally known as the 0-dimensional homology, describe the connected components of a space. Since it is an exploratory process to understand the scales of interesting features, we use a multi-scale notion of homology, called *persistent homology*, to describe the topological features of the space at different spatial resolutions.

Given a weighted graph $G = (V,E,w)$ with positive edge weights defined by $w : E \to \mathbb{R}$, our first step is to associate the graph G with a metric space representation. Considering the *inverse*[†] of the positive edge weight as the length of an edge, a classical *shortest-path* metric d can be defined on G where the distance between each pair of nodes $x,y \in G$ is the length of the shortest-path between them. Such a metric can be computed using the Dijkstra's algorithm [Dij59]. See Figure 2(a) for an illustration.

With the above metric space embedding, every node in G corresponds to a point in the metric space. In order to compute the 0-dimensional persistent homology of G, we apply a rather simple geometric construction on its metric space representation. Consider the set of balls centered at every point in the metric space with a radius t, we keep track of how the (connected) components of the union of balls evolve as t increases from 0 to ∞. As t increases, the unions of balls form components in a hierarchical fashion. Starting with each point as a component when $r = 0$; as r increases, the number of connected components decreases by one when two components merges into one – formally, this is referred to as a *topological event*.

A topological event corresponds to the birth (appearance) and death (disappearance) of a component. The *birth* time of a component is the time when the component appears, in our case, it is always at time 0. The *death* time is when a component disappears, that is, when it merges with another component that was born earlier. The lifetime of a component (i.e. its death time minus its birth time) is its *persistence*.

The topological events associated with G are placed into a *persistence barcodes* [Ghr08b], which consist of a collection of bars, each corresponding to a topological event, whose starting and ending points correspond to the birth time and the death time of its associated component, with a width equals its persistence. See Figures 2(b) for an illustration. Since we care about 0-dimensional topological features, we only need to compute its corresponding 0-dimensional barcodes.

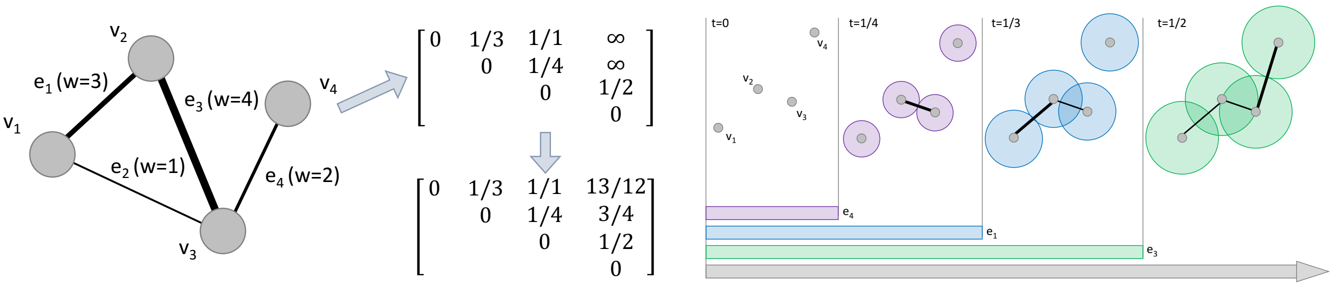
3.2. Fast Computation of the 0-Dimensional Barcodes

The evolution of connected components of union of balls can be completely captured by a combinatorial structure consisting of a collection of edges: at time t an edge appears between a pair of points when their balls of radius $t/2$ intersect. We provide a simple algorithm to compute the 0-dimensional barcodes of the graph G. For simplicity, we suppose that G is connected. In a nutshell, our algorithm consists of computing the minimal spanning tree T of $G = (V,E,w)$ based on its metric space embedding, while keeping track of the times when an edge is added to the tree.

Let $V = \{v_1, \cdots, v_m\}$ be the node set of G and let $E = \{e_1, \cdots, e_n\}$ be its edge set ordered with respect to $1/w$.

The algorithm starts by creating an empty spanning tree. It then creates sets C_i and bars $\mathcal{B}_i$, one per graph node. Each bar (at this step) in the persistence barcodes is represented by a pair of real numbers $(birth, death)$ where $birth = 0$ and $death = \infty$. The second

[†] In our setting, the inverse of the edge weight between two nodes $1/w(x,y)$ captures the similarity between them.



(a) (b)

Figure 2: *Extracting 0-dimensional persistent homology from a graph. (a): Given a weighted graph G with edge weights w, we obtain a metric space representation based on the shortest-path distance metric. (b): We construct a filtration from G by growing balls around the points in the metric space. Edges are added when two balls intersect. A topological event occurs when two connected components merge into one, consequently a bar in the persistence barcodes terminates.*

Data: A weighted graph $G = (V, E, w)$
Result: A minimal spanning tree T and the 0-dimensional barcodes $\mathcal{B}$ of G

Create an empty spanning tree T
foreach *node* v_i **do**
  Create set C_i
  Create a bar $\mathcal{B}_i$ with *birth* $= 0$ and *death* $= \infty$
end
foreach *edge* e_i *in* E **do**
  if e_i *connects two different sets* C_1 *and* C_2 **then**
    Join C_1 and C_2
    Set the *death* of $\mathcal{B}_1$ to $1/w(e_i)$
    Add e_i to the spanning tree T
  end
end

step of the algorithm looks at the edges one at a time, ordered by $1/w_i$. For each e_i we check if vertices of this edge belong to two different sets. If this is the case then we set *death* of one of the nodes of e_i to be the $1/w_i$. This step can be performed efficiently using the *disjoint set* data structure.

The addition of an edge to the minimal spanning tree T coincides with the topological event when two components merge into one. Therefore, there is a one to one correspondence between the edges of T and the 0-dimensional barcodes of G with finite persistence. When the event occurs at time $1/w_i$, and hence the length of its corresponding bar is associated with edge length at $1/w_i$.

3.3. Node Clustering with the Spanning Tree

We will now use the information encoded by the minimal spanning tree to cluster the vertices of the graph G. Denote the spanning tree as $T(V, E(T))$, where $E(T)$ denote the edges in the tree. Deleting an edge e from $E(T)$ splits the tree T into two disjoint clusters. Similarly, deleting k edges from $E(T)$ split V into $k+1$ clusters. We utilize this idea along with the fact that edges $E(T)$ correspond to the 0-dimensional barcodes with finite persistence to interactively split the node set of G into clusters. This is essentially Zahn's clustering algorithm [Zah71].

Each topological feature (i.e. a topological event, a bar in the barcodes) is associated with the following information that we utilize in our visualization. Such information is computed separated for each feature and is also illustrated in Figure 3.

- The *persistence* of the bar is equal to $1/w$. This helps to indicate how "important" a topological event is.
- The *cause of death* are the vertices of the edge that cause the topological event to occur. These will be used to further differentiate interesting topological events.
- The *subsets of vertices* representing removal of the edge from the spanning tree. These sets will also be important when updating graph forces to the reflect topological feature selection.
- The *subset ratio* is a measure of the number of vertices in the two subsets of vertices. This indicates how evenly a feature bifurcates the graph.

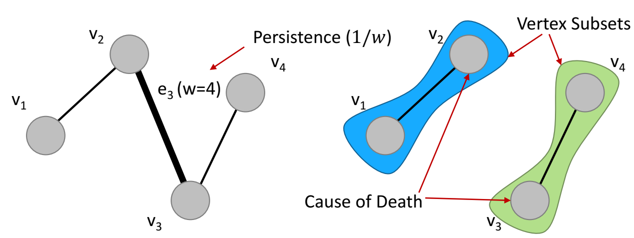
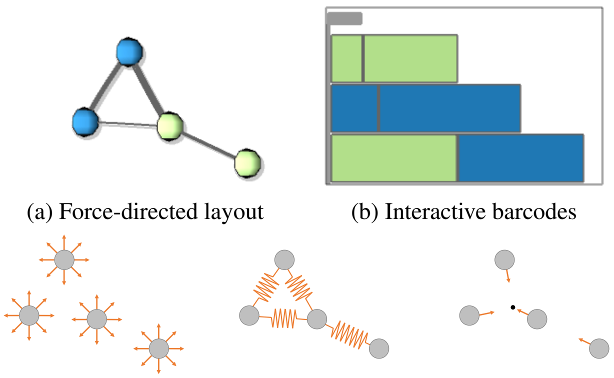


Figure 3: *Example of information extracted from a spanning tree (left) for edge e_3 from Figure 2(b) (the purple bar). The right shows the clusters created when a selected edge is removed from the spanning tree.*

4. Visual Design and Interaction Design

Our design goal is to provide a simple interface that enables interactive exploration of a graph using its topological structure to guide the layout. Our objective is to declutter the display in order to emphasize substructures of interest.



(a) Force-directed layout (b) Interactive barcodes

(c) Repulsive (d) Spring attractive (e) Centering

Figure 4: *The force-directed layout (a) for the graph in Figure 2 is constructed using a combination of (c) repulsive forces, (d) spring attractive forces, and (e) a centering force. The nodes have been arbitrarily labeled for demonstration purposes. The interactive barcodes (b) is used to manipulate the display.*

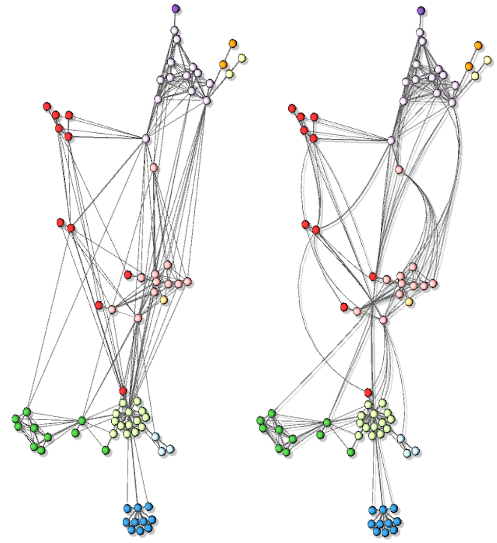


Figure 5: *Example of Lés Miserables graph before (left) and after (right) edge bundling.*

4.1. Graph Drawing

Our design starts with a force-directed layout that uses edge bundling to declutter the display.

Force-Directed Layout. A graph is initially drawn with a standard force-directed layout [FR91b]. Such a layout is chosen because of its interactivity and flexibility in selecting forces. The layout starts with three types of forces. The first force enables all nodes to repulse one another (Figure 4(c)). The second force is a spring attraction for nodes connected by an edge (Figure 4(d)). Finally, a very weak centroidal force draws all nodes towards the center essentially centering the layout (Figure 4(e)). The parameters for these forces, such as mass, force strength, and spring resting length require manual tuning.

Edge Bundling. We use edge bundling to reduce the distractive impact of visual clutter caused by overlapping edges. We require a special edge bundling design in which the display is interactive, and thus the bundling must be temporally coherent. To accomplish this, we use a slightly modified version of the *force-directed edge bundling* [HVW09b]. This technique subdivides edges and uses a variant of a force-directed layout to attract edges with similar proximity and direction. Figure 5 shows an example graph before and after edge bundling is enabled.

Node Coloring. Our experimental graphs have categorical data attached to the nodes. Such categorical information is used for node coloring using a categorical colormap.

4.2. Persistence Barcodes

We use persistence barcodes to represent the persistent homology features, where each feature is shown with a bar starting and ending at its birth and death time respectively, with the bar length indicating its persistence. Since we only employ 0-dimensional features in this paper, each bar is born at time 0, and its persistence is its death time. We augment the barcodes with additional visual encodings (Figure 4(b)) in order to guide interaction with the graph.

Subset Ratio. Each bar is augmented with a vertical line, splitting it into 2 based upon the *Subset Ratio*. For example, in Figure 4(b) the bottom bar, representing e_3, has a 50/50 split. This is because exactly 2 vertices exist on either side of that edge in the spanning tree, see Figure 3.

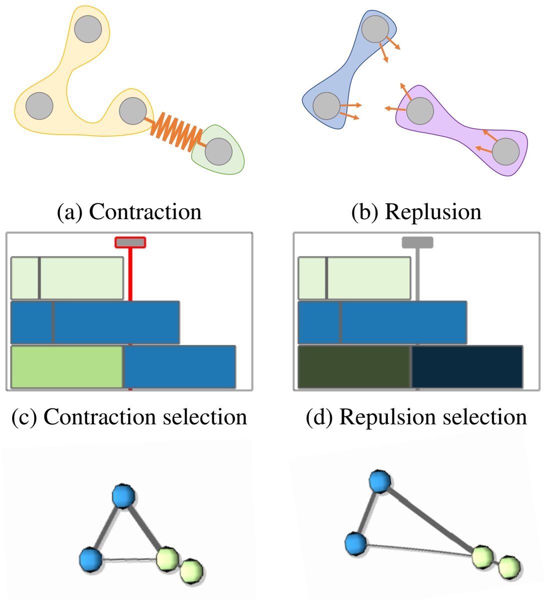
Color. For vertices with categorical data, each side of the split on the bar is colored based on the category of the *Cause of Death* nodes for the topological event.

Bar Sorting. Bars are sorted based upon 2 criteria. The first is *persistence*. This helps to differentiate high weight (i.e. high persistence) edges from low weight. If the persistence of 2 bars is equal, they are then sorted based upon their *Subset Ratio* with bars having 50/50 ratios appearing further down in the barcode, since those closer to 50/50 more evenly split the graph.

4.3. Interaction Using Persistence Barcodes

Topological Event Contraction. We provide a persistence simplification tool that enables contractions of graphs for all topological events whose persistence is below a user-selected threshold. This is done by dragging a scrollbar at the top of the barcode, see scrollbar in red in Figure 6(c). As the threshold is dragged left to right, topological events with persistence below the threshold will have their color washed out, and their graph nodes contracted. This contraction is accomplished by adding a strong spring force between the *cause of death* nodes for the event. Figure 6(a) illustrates this force and Figure 6(e) shows an example of contraction.

Topological Event Repulsion. When a bar is individually selected, the bar color is darkened, and a strong repulsive force is added between the *subsets of vertices* associated with that topological event.



(a) Contraction (b) Replusion

(c) Contraction selection (d) Repulsion selection

(e) Adding contraction (f) Adding repulsion

Figure 6: *Illustration of forces applied to the graph in Figure 4(a). (a), (c) and (e): A contracting force applied to the layout. (b),(d) and (f): A repulsive force is then added to the layout from (e).*

The repulsion of the nodes from these two groups allows the layout to naturally form clusters. For example, Figures 6(b) illustrates the force when the bottom bar in Figure 6(d) is selected for repulsion, causing the subsets to push apart. In other words, in Figure 6(b), all of the blue points have an additional repulsion from all of the purple points and vice-versa. Figure 6(f) shows the result of adding this force to the graph.

Selecting Multiple Bars. Multiple bars may be selected simultaneously for attraction and repulsion, since they do not directly depend on each other. Whenever a new bar is selected, an additional force is simply added to the layout.

Preview Hovering. To help users preview the impact of a bar selection, when the mouse hovers over a bar from the barcodes, bubble sets [CPC09] are employed on the *subsets of vertices* to differentiate which nodes belong to which subset. Figure 7(b) and 7(c) show examples of bubble sets employed on a dataset. The bubble sets demonstrate before and after, respectively, how the graph will be divided into two interesting groups.

Hyperbolic Zoom for Large Barcodes. The number of bars is equal to the number of nodes in the graph minus one. In order to scale the barcodes appropriately for a large graph, a hyperbolic zoom is used. As the mouse moves up and down the barcodes, the focus of the hyperbolic zoom is modified, therefore revealing the local bar data. An example of this can be seen in our accompanying video.

5. Results

Our framework is implemented using Processing. Once the data is loaded, the 0-dimensional topological features can be extracted very efficiently; since the Boruvka's algorithm for minimum spanning tree takes $O(m \log n)$ time with m being the number of edges and n being the number of nodes. Then the software runs interactively. Our accompanying video is produced on a 2017 MacBook Pro with a 3.1 Ghz i5 processor to demonstrate the interactivity of our interface. To evaluate our approach, we examine three datasets.

5.1. Lés Miserables

We first examine an undirected, weighted network for co-occurrences of characters in Victor Hugo's novel "Lés Miserables" [Knu93]. The network has 77 nodes, where each node represents a character in the novel and 254 edges, weighted by how many scenes two characters share during any chapter of the novel.

In Figure 1(b), we show the conventional force-directed layout for this dataset with the nodes colored based on group IDs. When hovering over a bar in the persistence barcode, the user will be shown how the groups are split based on the topological event associated with that feature. Figure 7(b) and 7(c) shows the before and after, respectively, for modifying the layout by selecting the most significant feature from the barcode.

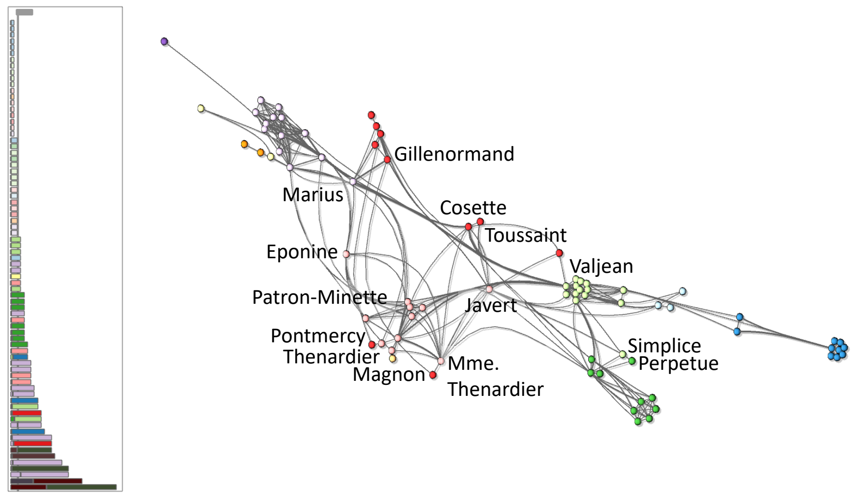
By selecting the five most persistent bars and collapsing a set of the lowest persistent features, we are able to easily and interactively explore the dataset by revealing its underlying structure, seen in Figure 7(a). Labels are provided for further discussion.

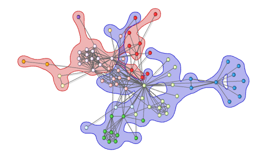
On two opposing sides of the layout are nodes Marius and Valjean, two main characters, who both have a large cluster of nodes surrounding them. Isolated nodes between these two include Éponine–a woman in love with Marius; Cosette–Valjean's daughter and Marius' lover; and Javert–the primary antagonist to Valjean.

There are several notable features here told by the structure of our layout: Éponine and Javert are both labeled as part of the antagonists group; Éponine is Thénardier's daughter (a man determined to exact revenge on Valjean); and Javert, an inspector who is described as obsessed with the capture of Valjean. However, both of these nodes have been strongly repulsed and isolated from their original group when compared to the conventional layout.

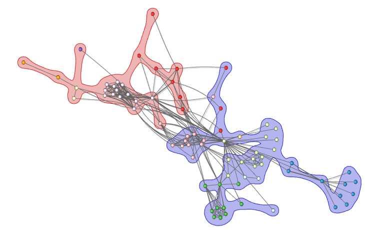
During Volume IV, Éponine is responsible for opposing and turning back a group of four robbers (found in the cluster labeled "Patron-Minette") who were sent by her father Thénardier to protect her love interest Marius. During Volume V, Inspector Javert, shown as the figure most directly opposite of Valjean, decides not to turn Valjean in to authorities after an act of kindness Valjean shows Javert, saving his life. Cosette, structured exactly in the middle of both her father Valjean and her lover Marius, is grouped together with the servant Toussaint. In the conventional layout, it is not identifiable that these two would be paired together, however, in the novel Toussaint is a motherly-figure assisting in raising Cosette through her childhood.

Other notable groups formed that are not shown in the conventional layout include: Sister Simplice and Sister Perpetue–both nuns that aid in taking care of Fantine. Mme. Thénardier





(b) Bubble set for highest persistence bar.



(a) Layout with a set of features selected for attraction and five bars selected for repulsion. Labels are provided to analyze the structures created with Lés Miserables characters.

(c) Results after selecting the highest persistence bar.

Figure 7: *Examples from the Lés Miserables dataset.*

and Magnon–Mme. Thénardier gives her two youngest children to Magnon in exchange for money. Pontmercy and Thénardier–Pontmercy believes Thénardier is responsible for saving his life. Marius and Gillenormand's circle–Gillenormand is Marius' grandfather and has raised him since he was a child.

5.2. Madrid Train Bombing

The Madrid Train dataset contains 70 nodes and 243 edges, where a node represents "individuals involved in the bombing of commuter trains in Madrid on March 11, 2004"[‡] [Rod05]. Each group has been identified and colored based on if the person was involved in previous terrorist acts and whether they were a member of the Field Operations Group. A link is connected if two individuals were related prior to or during the bombing. Weight is calculated on an index between 1-4, where each of the following four parameters are summed per pair:

- Trust–friendship (contact, kinship, links in the telephone center).
- Ties to Al Qaeda and to Osama Bin Laden.
- Co-participation in training camps and/or wars.
- Co-participation in previous terrorist Attacks (Sept. 11, Casablanca, etc.).

We begin by examining the conventional force-directed layout in Figure 8(a). There are two nodes that notably cluster together towards the center of the graph. However, it is difficult to understand what their role is in the network or who the other important players were in the Madrid bombing.

When collapsing the set of low persistence bars and selecting all other bars for repulsion, shown in Figure 8(b), a central node clearly combines two sides of the network. This node is Jamal Zougam, a member of the Field Operations Group (FOG) who was involved in three other terrorist attacks. Jose Rodriguez describes Zougam as the most important person in the network, which was not easily identifiable in the conventional force-directed graph [Rod05]. Our layout shows how he served as a link between the left and right side of the network, where the left side connects those involved in the Sept. 11 attack. On the right side of Zougam, a tight cluster of those unconnected to any prior act of terrorism collapse together. As Rodriguez notes in his paper, these are acquaintances of Zougam's brother Mohhamed Chaoui who were unanimously uninvolved with the FOG prior to the Madrid bombing.

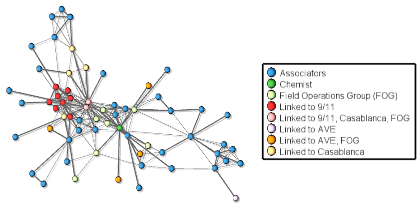
One of the key players Rodriguez notes is "The Chemist", Abderrahim Zbakh. However, when observing the conventional force-directed graph, it is difficult to determine what connections he creates in the network. We select all persistence features that will involve Zbakh, shown in Figure 8(c), resulting in a large divide where he is at the center. We change the group colors to only display members and non-members of the FOG to better understand Rodriguez's statement "The Chemist links each of the unconnected members of FOG with the central structure, through ties based on friendship and personal contact" [Rod05]. With our layout, we see on the left side four FOG members collapsing together tightly around Jamal Zougam, while on the right side eight FOG members are significantly more isolated and only linked through Zbakh.

5.3. 1998 FIFA World Cup

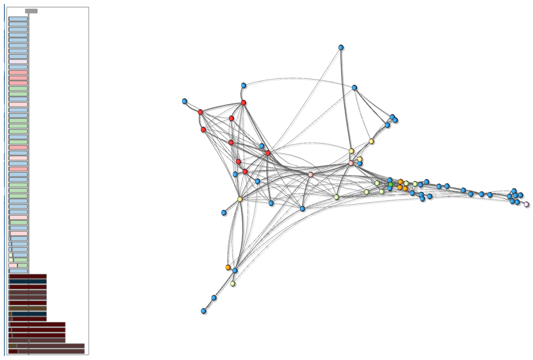
The final dataset we examine is a player network with 35 vertices and 118 edges. This dataset involves 22 soccer teams that participated in the 1998 FIFA World Cup, where a node represents a country that bore a contract with players of another national team. An edge is created if one country imported or exported players from another country. Countries are grouped and colored based on which of the six FIFA confederations they belonged to in 1998.

We modify the conventional force-directed layout, shown in Fig-

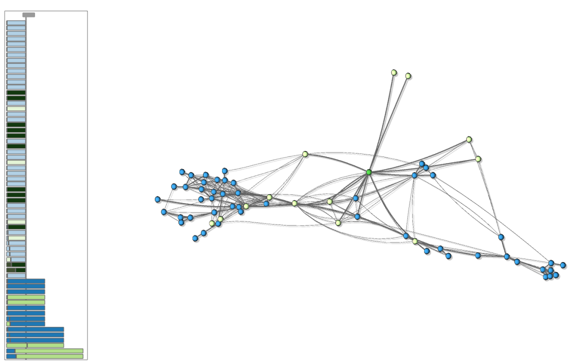
‡ http://moreno.ss.uci.edu/data



(a) Conventional layout.



(b) Final result using our layout when low persistence bars are collapsed and remaining high persistence bars are selected.



(c) A modified version of our layout when all bars involving The Chemist are selected.

Figure 8: *Examples from the Madrid Train Bombing dataset.*

ure 9(a), by selecting to repulse and collapse the five highest and lowest persistence bars, respectively, shown in Figure 9(b). The top five highest persistent bars have only one node being split in a group from the others: this would not seem to tell a very interesting story. However, all nodes that are split from the layout played a significant role in the 1998 FIFA World Cup: these countries were responsible for the highest number of trades or highest number of players exported/imported.

- Norway to England: Norway imported 4x the amount of players (12) to England than to any other country they traded with. In fact, England imported their largest player base from Norway, where the second highest amount of players (7) came from both Jamaica and Scotland.
- Paraguay to Brazil: Paraguay traded 13 of their players away, the majority of them (10) to Brazil, while Brazil imported 12. However, Paraguay did not import any players in this network.
- Yugoslavia to Spain and Italy: Yugoslavia exported 9 players to Spain and 7 players to Italy.
- Argentina to Italy: Argentina exported players to only two countries this year, 9 players to Italy and 4 to Spain.
- Cameroon to France: This high persistence feature highlights the first trade that occurred between two different regions. Interestingly, France imported the majority of their players from Africa (7 from Cameroon, 1 from Morocco, 3 from Nigeria, 3 from Tunisia, and 1 from South Africa), as indicated by the collapse of yellow nodes surrounding France in Figure 9(b).

It is important to note that the conventional layout for the football dataset did not intuitively cluster groups by number of trade deals. For example, with our layout, Italy is shown to be significantly far away from the cluster of European countries due to trading with three of five South American countries in the network. Italy was also the most active country in the network with 18 imports and three exports. Other outliers seen on our layout—Yugoslavia, Paraguay, Argentina, Norway, Scotland, Jamaica—traded the highest number of players (between 7 to 13) to countries that are central to their nearest cluster (Italy, England, France).

6. Discussion

Scalability. The visual scalability of our approach is fairly limited. We have experimented with datasets of a few hundred nodes, though they are not shown. At that size, the display becomes clustered by the shear number of nodes and edges displayed. This is not unlike conventional node-link diagrams.

Relationship to Hierarchical Clustering. Calculating 0-dimensional persistent homology features has a strong relationship to finding hierarchical clusters. However, the main differentiating factor is the treatment of features in persistent homology. For example, persistent homology sees low weight edges as "noise", hence collapsing them makes sense. At the same time, it sees high weight edges as signal, hence separating features makes sense.

7. Conclusion

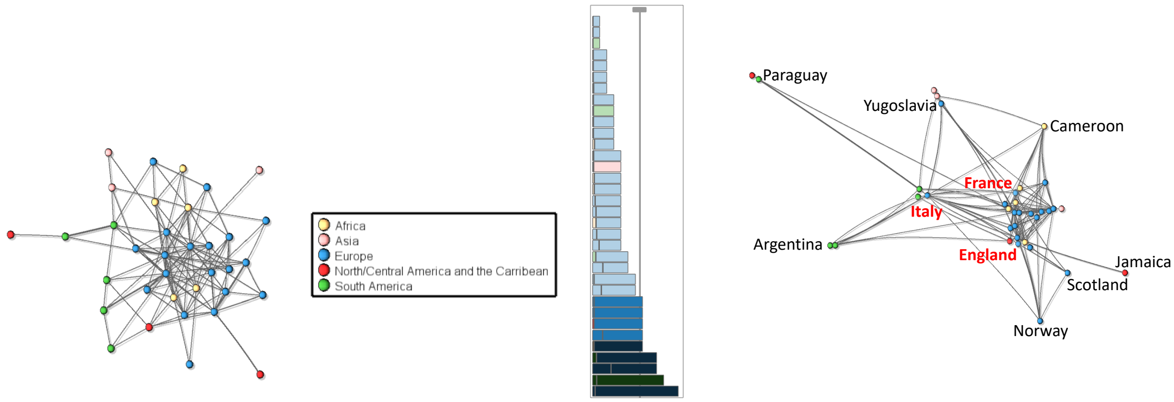
In conclusion, we have presented an approach to graph drawing that uses persistent homology to extract features that are then used to interactive modify a force-directed layout. Finally, in the future, we would like to look at the potential of using higher-dimensional persistent homology features to control the graph drawing.

References

[BG14] Bampasidou M., Gentimis T.: Modeling collaborations with persistent homology. *CoRR abs/1403.5346* (2014).

[BHJ09] Bastian M., Heymann S., Jacomy M.: Gephi: an open source software for exploring and manipulating networks. In *ICWSM* (2009), pp. 361–362.

[BP07] Brandes U., Pich C.: Eigensolver methods for progressive multidimensional scaling of large data. In *Graph Drawing* (2007), Springer, pp. 42–53.



(a) Conventional layout.

(b) Our version of the layout. Red labels denote those who imported a high number of players. Black labels denote those who exported a high number of players to the three countries labeled in red.

Figure 9: *Example of the 1998 FIFA World Cup dataset.*

[Car09] CARLSSON G.: Topology and data. *Bulletin of the American Mathematical Society 46*, 2 (2009), 255–308.

[CH13] CARSTENS C. J., HORADAM K. J.: Persistent homology of collaboration networks. *Mathematical Problems in Engineering 2013* (2013).

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

[CRS15] CASSIDY B., RAE C., SOLO V.: Brain activity: Conditional dissimilarity and persistent homology. *IEEE 12th International Symposium on Biomedical Imaging (ISBI)* (2015), 1356 – 1359.

[Dij59] DIJKSTRA E. W.: A note on two problems in connexion with graphs. *Numerische Mathematik 1* (1959), 269–271.

[DMFC12] DABAGHIAN Y., MÉMOLI F., FRANK L., CARLSSON G.: A topological paradigm for hippocampal spatial map formation using persistent homology. *PLoS Computational Biology 8*, 8 (2012), e1002581.

[DPS*12] DONATO I., PETRI G., SCOLAMIERO M., RONDONI L., VACCARINO F.: Decimation of fast states and weak nodes: topological variation via persistent homology. *Proceedings of the European Conference on Complex Systems* (2012), 295–301.

[DRMM13] DWYER T., RICHE N. H., MARRIOTT K., MEARS C.: Edge compression techniques for visualization of dense directed graphs. *IEEE Transactions on Visualization and Computer Graphics 19*, 12 (2013), 2596–2605.

[DS13] DUNNE C., SHNEIDERMAN B.: Motif simplification. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2013).

[DWvW12] DINKLA K., WESTENBERG M. A., VAN WIJK J. J.: Compressed adjacency matrices: untangling gene regulatory networks. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (2012), 2457–2466.

[EGK*02] ELLSON J., GANSNER E., KOUTSOFIOS L., NORTH S. C., WOODHULL G.:. In *Graph Drawing* (2002), Springer, pp. 483–484.

[EH08a] EDELSBRUNNER H., HARER J.: Persistent homology - a survey. *Contemporary Mathematics 453* (2008), 257–282.

[EH08b] EDELSBRUNNER H., HARER J.: Persistent homology-a survey. *Contemporary mathematics 453* (2008), 257–282.

[EH10a] EDELSBRUNNER H., HARER J.: *Computational topology: an introduction*. American Mathematical Soc., 2010.

[EH10b] EDELSBRUNNER H., HARER J.: *Computational Topology: An Introduction*. American Mathematical Society, Providence, RI, USA, 2010.

[EHMP04] EDELSBRUNNER H., HARER J., MASCARENHAS A., PASCUCCI V.: Time-varying reeb graphs for continuous space-time data. In *Proceedings of the twentieth annual symposium on Computational geometry* (2004), ACM, pp. 366–372.

[EKLN03] ERTEN C., KOBOUROV S. G., LE V., NAVABI A.: Simultaneous graph drawing: Layout algorithms and visualization schemes. In *International Symposium on Graph Drawing* (2003), Springer, pp. 437–449.

[ELY12] E W., LU J., YAO Y.: The landscape of complex networks. *CoRR abs/1204.6376* (2012).

[FR91a] FRUCHTERMAN T. M., REINGOLD E. M.: Graph drawing by force-directed placement. *Software: Practice and experience 21*, 11 (1991), 1129–1164.

[FR91b] FRUCHTERMAN T. M., REINGOLD E. M.: Graph drawing by force-directed placement. *Software: Practice and experience 21*, 11 (1991), 1129–1164.

[FW94] FRÖHLICH M., WERNER M.: Demonstration of the interactive graph visualization system da vinci. In *International Symposium on Graph Drawing* (1994), Springer, pp. 266–269.

[GHNS11] GANSNER E. R., HU Y., NORTH S., SCHEIDEGGER C.: Multilevel agglomerative edge bundling for visualizing large graphs. In *IEEE Pacific Visualization Symposium* (2011), pp. 187–194.

[Ghr08a] GHRIST R.: Barcodes: the persistent topology of data. *Bulletin of the American Mathematical Society 45*, 1 (2008), 61–75.

[Ghr08b] GHRIST R.: Barcodes: The persistent topology of data. *Bullentin of the American Mathematical Society 45* (2008), 61–75.

[GKN05] GANSNER E. R., KOREN Y., NORTH S.: Graph drawing by stress majorization. In *Graph Drawing* (2005), Springer, pp. 239–250.

[GKNV93] GANSNER E. R., KOUTSOFIOS E., NORTH S. C., VO K.-P.: A technique for drawing directed graphs. *IEEE Transactions on Software Engineering 19*, 3 (1993), 214–230.

[HH91] HENRY T. R., HUDSON S. E.: Interactive graph layout. In *Proceedings of the 4th annual ACM symposium on User interface software and technology* (1991), ACM, pp. 55–64.

[HMR09] HORAK D., MALETIĆ S., RAJKOVIĆ M.: Persistent homology of complex networks. *Journal of Statistical Mechanics: Theory and Experiment* (2009), P03034.

[HSS10] HANSEN D., SHNEIDERMAN B., SMITH M. A.: *Analyzing social media networks with NodeXL: Insights from a connected world*. Morgan Kaufmann, 2010.

[Hu05] HU Y.: Efficient, high-quality force-directed graph drawing. *Mathematica Journal 10*, 1 (2005), 37–71.

[HVW09a] HOLTEN D., VAN WIJK J. J.: Force-directed edge bundling for graph visualization. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 983–990.

[HVW09b] HOLTEN D., VAN WIJK J. J.: Force-directed edge bundling for graph visualization. In *Computer graphics forum* (2009), vol. 28, Wiley Online Library, pp. 983–990.

[HWSR17] HAJIJ M., WANG B., SCHEIDEGGER C., ROSEN P.: Visual detection of structural changes in time-varying graphs using persistent homology. *arXiv preprint arXiv:1707.06683* (2017).

[KCH02] KOREN Y., CARMEL L., HAREL D.: Ace: A fast multiscale eigenvectors computation for drawing huge graphs. In *IEEE Symposium on Information Visualization* (2002), pp. 137–144.

[Kei00] KEIM D. A.: Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics 6*, 1 (2000), 59–78.

[KHKS12] KHOURY M., HU Y., KRISHNAN S., SCHEIDEGGER C.: Drawing large graphs by low-rank stress majorization. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 975–984.

[KMSH12] KAIRAM S., MACLEAN D., SAVVA M., HEER J.: Graphprism: Compact visualization of network structure. In *Advanced Visual Interfaces* (2012). URL: `http://vis.stanford.edu/papers/graphprism`.

[Knu93] KNUTH D. E.: *The Stanford GraphBase: a platform for combinatorial computing*, vol. 37. Addison-Wesley Reading, 1993.

[Kor03] KOREN Y.: On spectral graph drawing. In *Computing and Combinatorics*. Springer, 2003, pp. 496–508.

[KSS07] KEIM D. A., SCHNEIDEWIND J., SIPS M.: Scalable pixel based visual data exploration. *Pixelization Paradigm, Lecture Notes in Computer Science 4370* (2007), 12–24.

[LCK*11a] LEE H., CHUNG M. K., KANG H., KIM B.-N., LEE D. S.: Computing the shape of brain networks using graph filtration and gromov-hausdorff metric. *International Conference on Medical Image Computing and Computer Assisted Intervention* (2011), 302–309.

[LCK*11b] LEE H., CHUNG M. K., KANG H., KIM B.-N., LEE D. S.: Discriminative persistent homology of brain networks. *IEEE International Symposium on Biomedical Imaging: From Nano to Macro* (2011), 841–844.

[LKC*12a] LEE H., KANG H., CHUNG M. K., KIM B.-N., LEE D. S.: Persistent brain network homology from the perspective of dendrogram. *IEEE Transactions on Medical Imaging 31*, 12 (2012), 2267–2277.

[LKC*12b] LEE H., KANG H., CHUNG M. K., KIM B.-N., LEE D. S.: Weighted functional brain network modeling via network filtration. *NIPS Workshop on Algebraic Topology and Machine Learning* (2012).

[LSL*13] LUM P. Y., SINGH G., LEHMAN A., ISHKANOV T., VEJDEMO-JOHANSSON M., ALAGAPPAN M., CARLSSON J., CARLSSON G.: Extracting insights from the shape of complex data using topology. *Scientific Reports 3* (2013).

[MM08] MUELDER C., MA K.-L.: Rapid graph layout using space filling curves. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1301–1308.

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

[OJS*11] OELKE D., JANETZKO H., SIMON S., NEUHAUS K., KEIM D. A.: Visual boosting in pixel-based visualizations. *Computer Graphics Forum 30*, 3 (2011), 871–880.

[PRMM15] PIRINO V., RICCOMAGNO E., MARTINOIA S., MASSOBRIO P.: A topological study of repetitive co-activation networks in in vitro cortical assemblies. *Physical Biology 12*, 1 (2015).

[PSDV13a] PETRI G., SCOLAMIERO M., DONATO I., VACCARINO F.: Networks and cycles: A persistent homology approach to complex networks. *Proceedings European Conference on Complex Systems 2012, Springer Proceedings in Complexity* (2013), 93–99.

[PSDV13b] PETRI G., SCOLAMIERO M., DONATO I., VACCARINO F.: Topological strata of weighted complex networks. *PLoS ONE 8*, 6 (2013), e66506.

[RFLL17] RIECK B., FUGACCI U., LUKASCZYK J., LEITTE H.: Clique community persistence: A topological visual analysis approach for complex networks. *IEEE Transactions on Visualization and Computer Graphics* (2017).

[RMS97] RYALL K., MARKS J., SHIEBER S.: An interactive constraint-based system for drawing graphs. In *Proceedings of the 10th annual ACM symposium on User interface software and technology* (1997), ACM, pp. 97–104.

[Rod05] RODRÍGUEZ J. A.: The march 11 th terrorist network: In its weakness lies its strength.

[SB92] SARKAR M., BROWN M. H.: Graphical fisheye views of graphs. In *Proceedings of the SIGCHI conference on Human factors in computing systems* (1992), ACM, pp. 83–91.

[SHH11] SELASSIE D., HELLER B., HEER J.: Divided edge bundling for directional network data. *IEEE Transactions on Visualization and Computer Graphics 17*, 12 (2011), 2354–2363.

[Tut63] TUTTE W. T.: How to draw a graph. *Proceedings of the London Mathematical Society s3-13*, 1 (Jan 1963), 743–767. URL: `http://dx.doi.org/10.1112/plms/s3-13.1.743`, `doi:10.1112/plms/s3-13.1.743`.

[VLKS*11] VON LANDESBERGER T., KUIJPER A., SCHRECK T., KOHLHAMMER J., VAN WIJK J. J., FEKETE J.-D., FELLNER D. W.: Visual analysis of large graphs: State-of-the-art and future research challenges. In *Computer graphics forum* (2011), vol. 30, Wiley Online Library, pp. 1719–1749.

[Zah71] ZAHN C. T.: Graph-theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on computers 100*, 1 (1971), 68–86.