Reducing Snapshots to Points: A Visual Analytics Approach to Dynamic Network Exploration

Stef van den Elzen, Danny Holten, Jorik Blaas, Jarke J. van Wijk

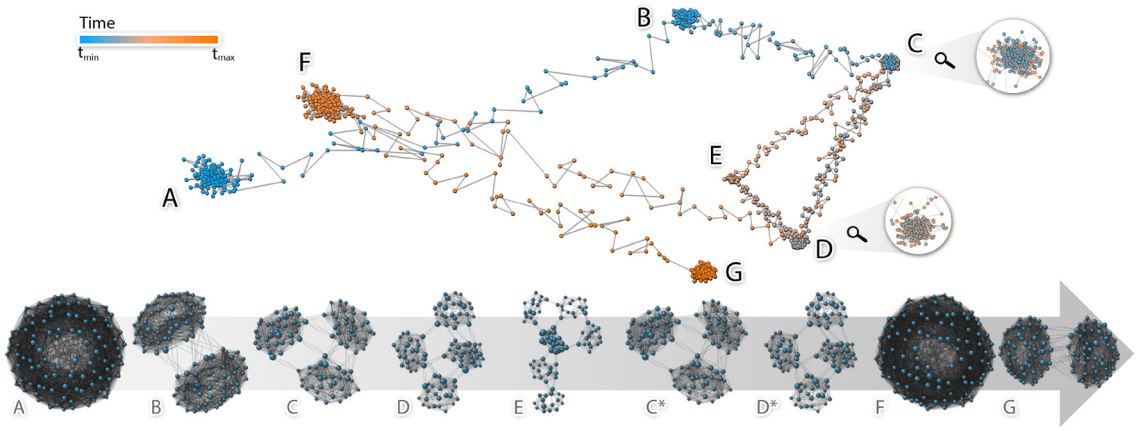


Fig. 1. Reducing snapshots of the dynamic network to points enables the exploration of network evolution. The projection of snapshots (top) reveals that the network contains 7 stable states (A-G), with 2 recurring states (C,D), and shows the transitions between them. A representative network for the snapshots in each stable state is shown (bottom).

Abstract—We propose a visual analytics approach for the exploration and analysis of dynamic networks. We consider snapshots of the network as points in high-dimensional space and project these to two dimensions for visualization and interaction using two juxtaposed views: one for showing a snapshot and one for showing the evolution of the network. With this approach users are enabled to detect stable states, recurring states, outlier topologies, and gain knowledge about the transitions between states and the network evolution in general. The components of our approach are *discretization*, *vectorization and normalization*, *dimensionality reduction*, and *visualization and interaction*, which are discussed in detail. The effectiveness of the approach is shown by applying it to artificial and real-world dynamic networks.

Index Terms—Dynamic Networks, Exploration, Dimensionality Reduction

1 Introduction

Networks are ubiquitous, as they describe relations between objects. Often these networks are large and dynamic; they change over time. Some examples of dynamic networks are (tele-)communication networks, social networks, financial networks, and transportation networks. Understanding the evolution of dynamic networks is a challenge. Typical insights to be gained are the discovery of states in dynamic networks that characterize the network over time. The identification of stable states, recurring states, outlier states, and transitions between these states helps in understanding the network. For example, the network can change gradually from one state to another, it could alternate between multiple states, or it might not be stable at all. An approach for the identification of these states and obtaining insight in the evolution of the network in general is needed.

The challenge is twofold; how to visualize, interact with, and analyze a large static network for one point in time (*snapshot*), and second, how to visualize, explore and analyze many of these snapshots. For the analysis of a static network (one snapshot) many methods are available, such as node-link diagrams and visual adjacency matrices [37]. In this paper we address the second issue, and present an approach to explore and analyze many snapshots of a dynamic network. Currently, there are two major visual approaches to analyze network evolution: mapping *time-to-time* and mapping *time-to-space* [9]. Two implementations of these techniques are *animation* and *small-multiples*. For small multiples one problem is to find an optimal balance between using few images, lacking temporal detail, and many smaller images, which are hard to interpret. Furthermore, it is demanding to focus on many items simultaneously and relate these to each other. The use of animation also has problems, such as the difficulty to track changes over (longer) time periods, and how to visually encode these changes.

Here we present a novel method that enables users to trace the network over time: the reduction of snapshots to points. More specifically, we contribute a visual analytics approach for dynamic network exploration, enabling:

- the visual identification of stable states, recurring states, and outlier states;
- the transitions between states, and;
- the analysis of network evolution in general.

The approach consists of four steps: 1) discretization; 2) vectorization and normalization; 3) dimensionality reduction; and 4) visualiza-

- *Stef van den Elzen is with Eindhoven University of Technology and SynerScope B.V. E-mail: s.j.v.d.elzen@tue.nl.*
- *Danny Holten and Jorik Blaas are with SynerScope B.V. E-mail: {danny.holten, jorik.blaas}@synerscope.com.*
- *Jarke J. van Wijk is with Eindhoven University of Technology. E-mail: j.j.v.wijk@tue.nl.*

Manuscript received 31 Mar. 2015; accepted 1 Aug. 2015; date of publication 20 Aug. 2015; date of current version 25 Oct. 2015. For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org. Digital Object Identifier no. 10.1109/TVCG.2015.2468078

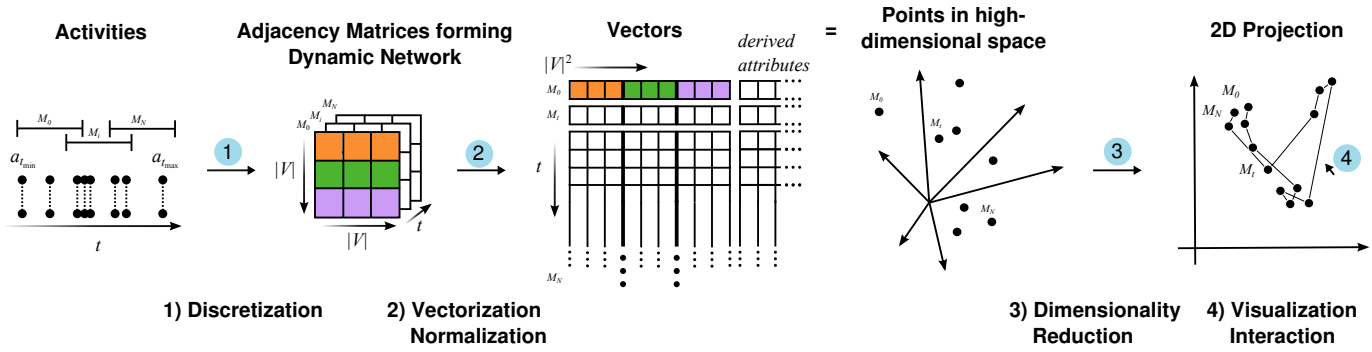


Fig. 2. Visual analytics approach for the exploration of dynamic networks with different steps 1) discretization, 2) vectorization and normalization, 3) dimensionality reduction, and 4) visualization and interaction.

tion and interaction. These steps are discussed in detail, and guidelines and defaults for parameters are provided. Together these steps enable dynamic network exploration.

The paper is organized as follows. First, related work is discussed in Section 2. The visual analytics approach is presented in Section 3. In Section 4 we apply our approach to artificial and real-world dynamic networks. The approach is discussed in Section 5 and finally, conclusions and directions for future work are given in Section 6.

2 Related Work

For static networks many visualization methods are available such as node-link diagrams, visual adjacency matrices, and hierarchical edge bundles. The two dominant methods for dynamic networks, using these techniques, are animation and small-multiples [10, 50].

In animation the different instances of the network at each timestep are shown as a movie, e.g., [24, 36, 41]. This approach leads to a high cognitive load, as users need to focus on many moving or changing items simultaneously. Also, tracking (multiple) changes over time is difficult. To reduce this burden, animation is often combined with a timeline control, for example in [6, 45]. If a node-link diagram is used as network visualization in the animation, then the stability of the network layout for subsequent timesteps influences the viewers mental map. Therefore, mental map preservation is an important area of research in graph drawing [18, 22, 40].

In the small-multiples technique the different timesteps are presented as juxtaposed visualizations using a filmstrip or grid layout [21, 26, 42]. In general, it is difficult to determine the number of multiples to use in the visualization. Furthermore, the multiples might be far apart from each other, making it harder to relate them to each other for the discovery of patterns. Also, visualization space for each multiple becomes smaller as the number of multiples grows. This decreases the readability of the individual visualizations. If small-multiples are used interactively, a solution to this is the use of large singles for detailed inspection [54]. A variation on the small-multiples approach is superposition; stacking the multiple instances of the network and optionally connecting related nodes of sequential timesteps [8, 19, 28]. Next to the generic techniques animation and small-multiples, sophisticated visualizations that provide an overview of the entire timespan of the network have been proposed [12, 13, 14, 15, 16, 17, 27, 44, 53]. However, these specialized visualizations are often difficult to interpret, difficult to reproduce, and generally pose restrictions on the network type.

Visual analysis techniques have been proposed for the exploration of dynamic networks, e.g., Von Landesberger et al. [60] also employ dimensionality reduction techniques with a focus on contagion in networks. Hadlak et al. [29] cluster temporal attributes associated with the nodes and edges. Steiger et al. [49] extend upon this by enabling the analysis of repeating time-series patterns. In contrast to our method, the structural connection between nodes and edges is not considered in both methods. Moreover, for our method we do not need temporal attributes to enable exploration and analysis. Also, visual analytics solutions for the exploration of dynamic networks are proposed, e.g., [1, 7, 20]. Falkowski et al. detect and show communities over time that are connected based on similarity. This method, in contrast to ours, focuses on community detection and not on states in the dynamic network. Outlier states as well as the absence of communities will not be picked up by the community detection. Furthermore, it is difficult to relate combinations of communities to identify recurring or stable states. Other methods, not related to dynamic networks, also propose visual analytics solutions for the identification of recurring states, e.g., [3, 58]. We only briefly discuss related work here, as dynamic network literature is plenty. For further reading we refer to recent surveys providing a broader perspective [5, 9, 33, 61]. In summary, current methods are either based on mapping time-to-time such as animation, or mapping time-to-space with (variations of) small multiples. Both techniques have their own issues and shortcomings. In this paper we explore our simple yet powerful method that deviates from these traditional techniques, by reducing snapshots of the dynamic network to points for exploration and analysis.

3 Reducing Snapshots to Points

First, the global concept is introduced by providing an overview of the approach. Next, dynamic networks are defined, followed by a description of the four steps of the approach: discretization (Section 3.3), vectorization and normalization (Section 3.4), dimensionality reduction (Section 3.5), and finally visualization and interaction (Section 3.6). Figure 2 provides a high-level overview of the approach.

3.1 Visual Analytics Approach

The visual analytics method consists of four steps, starting with dynamic network discretization. In this step the dynamic network data is prepared for analysis. The basic idea is to create a series of snapshots of the dynamic network at subsequent points (or short periods) in time based on event log data, such as transactions or communications. Each snapshot contains an instance G_i of the dynamic network.

The next step is simple but simultaneously crucial to our approach and is to the best of our knowledge not considered in previous literature: the network snapshots are vectorized and considered as points in high-dimensional space. This effectively represents the network edges at a point (or interval) in time as a row feature vector; each such point in this high-dimensional space represents the network at a different time-interval. The position of the snapshots in this space provides insights in the evolution of the network: snapshots where networks are similar will be positioned closer to each other and form clusters. Similarly, for timesteps where the network is different compared to more common network snapshots, the points will be outliers. Clusters of points indicate stable or recurring network states. Points lying in between clusters provide insight in how the network evolves from one state to another.

To enable analysis and exploration of the dynamic network snapshots we apply dimensionality reduction techniques and project the points to two dimensions in the third step. Finally, we introduce two juxtaposed linked views: a projection of the snapshots and a second view that provides a network visualization for a selected snapshot.

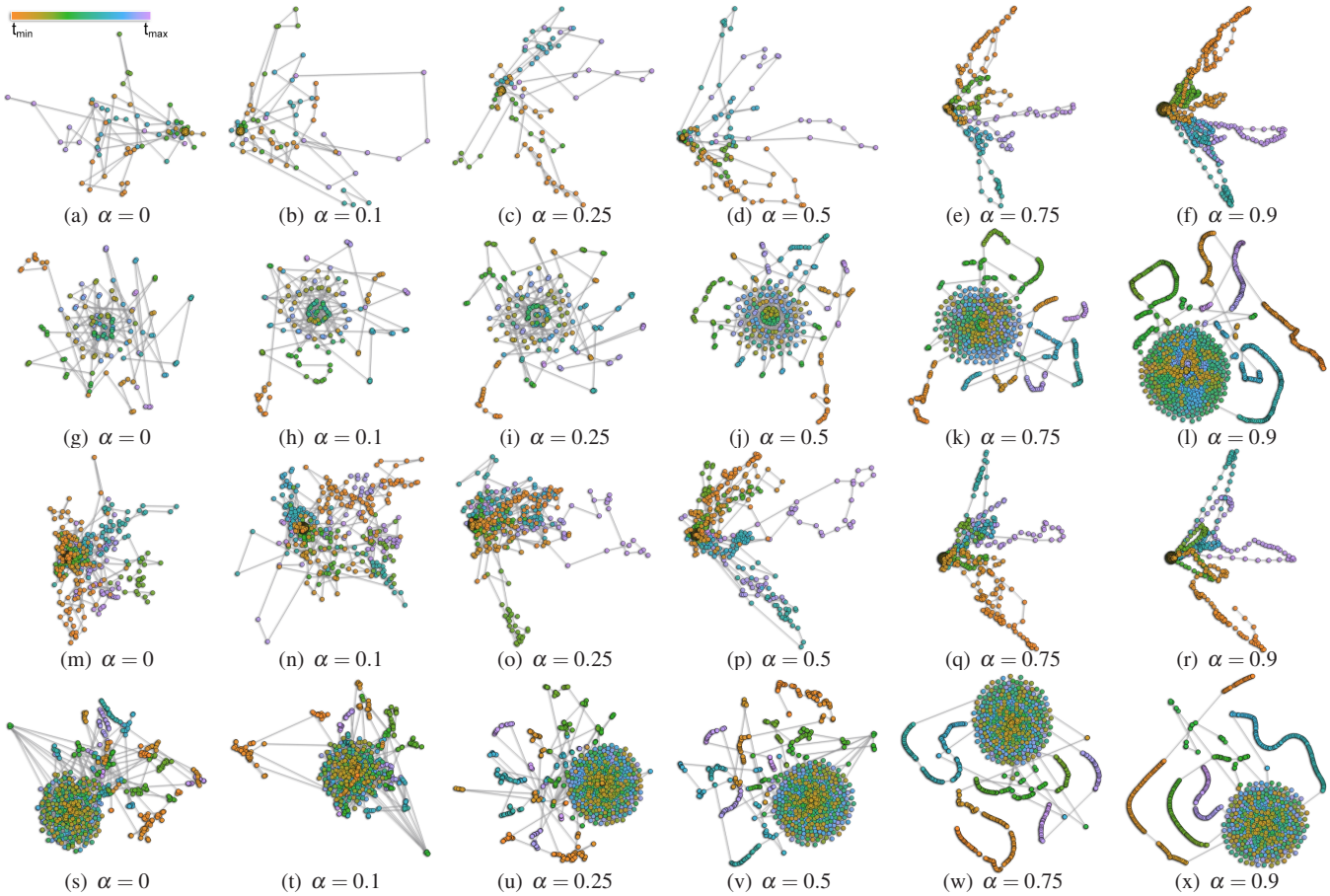


Fig. 3. Effect of different overlap values α for the snapshots in the discretization process. Images show a two dimensional projection for the *thiers-2011* [23] dataset computed using linear dimensionality reduction PCA (first and third row) and non-linear dimensionality reduction t-SNE (second and fourth row). The window width ω of the snapshots is kept constant in the first two rows, Δt (and hence the number of samples) varies. The number of snapshots a-f and g-l are, 151, 167, 201, 301, 602, 1500. In the third and fourth row the number of snapshots is kept constant at 1000 and the window width ω is varied. Generally, the more overlap, the more structure is visible in the resulting projections. Subsequent snapshots are connected with lines to reveal patterns over time. Points are colored according to time with a perceptually linear colormap (top left).

3.2 Dynamic Network Model

We model a dynamic network Γ as a sequence of N snapshots:

$$\Gamma = (G_1, G_2, \ldots, G_N), \tag{1}$$

where a snapshot is a directed graph $G_i = (V, E_i, t_i)$, with node (vertex) set V and edge (link, transaction, event) set $E_i \subseteq V \times V$ with from and to vertex tuples (v_m, v_n), and where t_i denotes the i-th timestep. The set of all edges in the dynamic network is the union of edge sets of all snapshots:

$$E = \bigcup_{i=1}^{N} E_i. \tag{2}$$

Furthermore, we consider weighted graphs. Let w be a function defined on the edge set as $w : i \times E \to \mathbb{R}$ assigning a real-valued weight $w(i,e)$ to edge $e \in E_i$.

3.3 Discretization of Dynamic Networks

In practice, datasets containing timestamped activities (e.g., log-files, email-traffic) are more ubiquitous than predefined sets of snapshots of a dynamic network. If these activities involve two objects each, like for instance sending a message from one address to another or a financial transaction between two accounts, it is natural to view such a dataset as a dynamic network. An activity log is modeled as:

$$A = (A_1, A_2, \ldots, A_M), \tag{3}$$

where $A_j = (a_j, s_j) \in E \times \mathbb{R}$ is an activity with edge $a_j(v_m, v_n)$ and time-stamp s_j. The activity log is transformed to a dynamic network by creating a sequence of snapshots for analysis. We assume $t_{j+1} - t_j = \Delta t$ is equidistant, representing days, hours, etc. Each snapshot G_i has an associated sampling time-window $[t_i - \omega/2, t_i + \omega/2)$ with width ω. Now the edge-set E_i of a snapshot is:

$$E_i = \{a_j \mid s_j \in [t_i - \omega/2, t_i + \omega/2)\}. \tag{4}$$

The weight of an edge e in snapshot i is simply the number of edges in the edge set, i.e., set cardinality:

$$w(i,e) = |\{A_j \mid a_j = e \wedge e \in E_i\}|. \tag{5}$$

To create the snapshots we have different choices for Δt. In the most extreme case each snapshot contains a single activity and at the other end of the spectrum there is only one snapshot containing all activities. Neither of these two options is desirable. In practice it is best to choose Δt based on domain knowledge or such that they result in a logical division of the timespan of the data, e.g., divide a day in hours, half-hours or quarters of an hour. A visualization of the events over time (e.g., a Reordered Massive Sequence View [52, 53]) could help in determining an appropriate window-length.

Here we choose to keep Δt constant. It may be justifiable to have varying lengths, however, this makes interpretation later on more difficult. To prevent missing patterns due to hard boundaries, time-windows are allowed to have an overlap α with neighboring time-intervals using a sliding window, i.e., we use $\omega > \Delta t$. The window

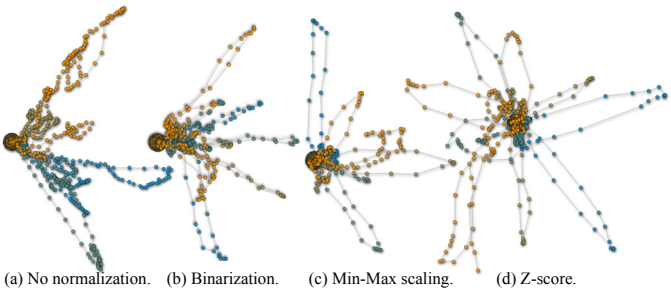


Fig. 4. Linear dimensionality reduction technique PCA is applied on dynamic network snapshots of the *thiers-2011* [23] dataset a) no normalization, b) binarization, c) min-max scaling, and d) z-score normalization. Z-score normalization is the preferred choice for PCA since we are interested in the components that maximize the variance, however, no normalization already reveals much of the structure and evolution of the dynamic network.

acts as a moving average, effectively smoothing the data by adding new edges and dropping old edges as time progresses. The overlap $\alpha \in [0,1]$ of successive windows is defined as:

$$\alpha = (\omega - \Delta t)/\omega. \tag{6}$$

Instead of uniform weighting of instances across time intervals, i.e., using a box kernel, smoother kernels could be used. However, the simple scheme used here gives already satisfying results and is easier to explain to non-experts. Figure 3 shows the influence of different fractions of overlap on the final projection for an example dataset: Figures 3, 4, 5, and 6 use the same dataset, which is described in detail in Section 4.2. We suggest a fractional overlap of 0.5 or more for a smoother image, however this is dataset dependent.

Each snapshot contains a number of edges, that when combined form a (static, flattened) weighted network. A snapshot G_i can be represented as an *adjacency matrix* M_i of size $|V|\times|V|$ with

$$M_{i,jk} = w(i,(v_j,v_k)). \tag{7}$$

As a result, we have a set of N matrices with snapshots that together define the dynamic network. In summary, we introduce discretization to create snapshots for activity-based datasets, and as an added bonus we can control the number of snapshots to better scale and deal with large datasets. Overlap is introduced to smooth snapshots of subsequent time-steps, which improves the interpretation of the final projection. Also, overlap reduces missing temporal patterns with otherwise hard boundaries. The resulting snapshots of the dynamic network are used in the next step of our approach by interpreting each snapshot as a point in high-dimensional space.

3.4 Vectorization and Normalization

Now that we have snapshots of the dynamic network, the goal is to analyze these and gain insight in the evolution. We enable this by simply reducing each snapshot to a point. In the following we describe how this is achieved.

All N network snapshots represented by $|V|\times|V|$ adjacency matrices are rearranged to $1\times|V|^2$ row-vectors which are points in $|V|^2$-dimensional space. The row-vectors are stacked to form a $N\times|V|^2$ matrix (see Figure 2, step 2). The columns of this matrix represent edges of the network and the rows represent different snapshots. A column is a feature vector, which represents the behavior of one edge over time. Instead of the adjacency matrix, also other attributes derived from the matrix can be used. Examples are simply the number of (active) edges and vertices, or the degree distribution of the snapshot.

The next step in the approach is to reduce and project the high-dimensional points to two dimensions. This is done using linear and non-linear dimensionality reduction methods, a heavily studied topic in statistics and data-mining. Before applying the methods the matrix

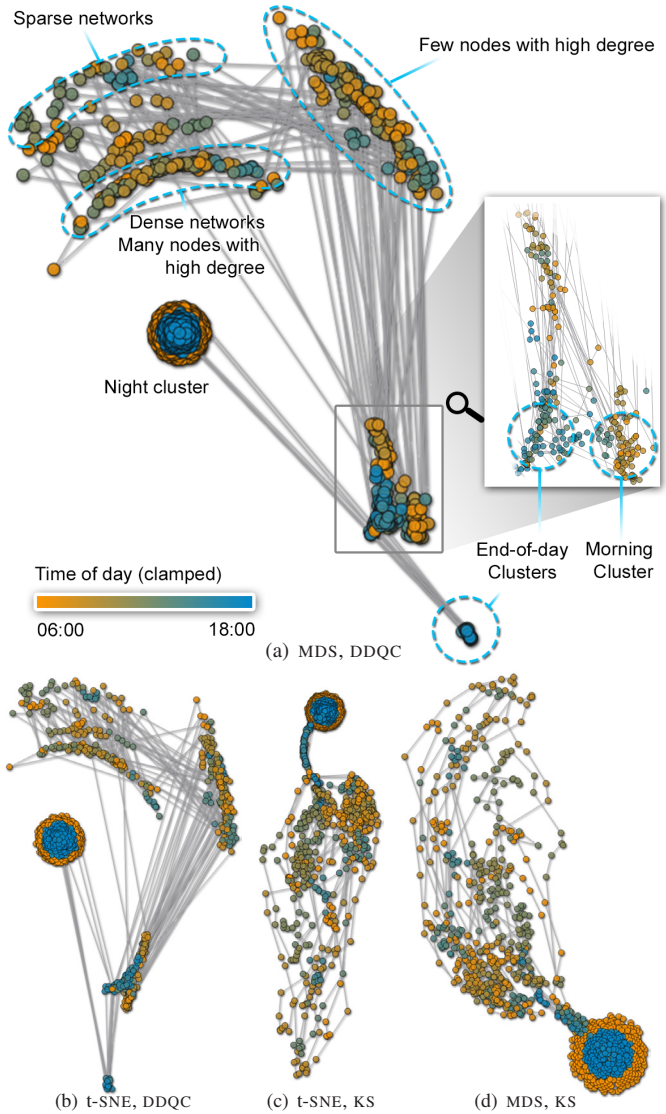


Fig. 5. Non-linear dimensionality reduction using a,d) MDS and b,c) t-SNE for degree distribution distance measures c,d) Kolmogorov-Smirnov and a,b) Degree Distribution Quantification and Comparison [2], applied on the *thiers-2011* dataset. Both non-linear dimensionality reduction methods result in similar images. This projection reveals a large night cluster, morning and end-of-day clusters that are connected to the night state. Three more network states are identified; a cluster with a network state where a few nodes have a high degree, a cluster with a sparse network state, and a cluster with a dense network state.

can be normalized to achieve better projection results; that is, we normalize each dimension of the high-dimensional space. We performed experiments with three different normalization methods: binarization, min-max, and z-score, see Figure 4. With binarization, 1 is assigned to each cell with a value $M_{i,jk} > 0$ and 0 otherwise. Min-max normalization scales the data to a fixed range $[0,1]$. Z-normalization transforms the data to have zero mean and unit variance by subtracting the mean and dividing by standard deviation. For a linear dimensionality reduction method, such as PCA, z-normalization is the preferred choice, since we are interested in the components that maximize the variance. Min-max normalization leads to smaller standard deviations, thus suppressing the effect of outliers.

For the cases we considered, there was no clear best method, and the choice seems to depend on the dataset and patterns to look for. Normalization might emphasize specific patterns in the data for easier identification, for example, z-score normalization suppresses outliers while min-max scaling preserves outliers. We achieved good results

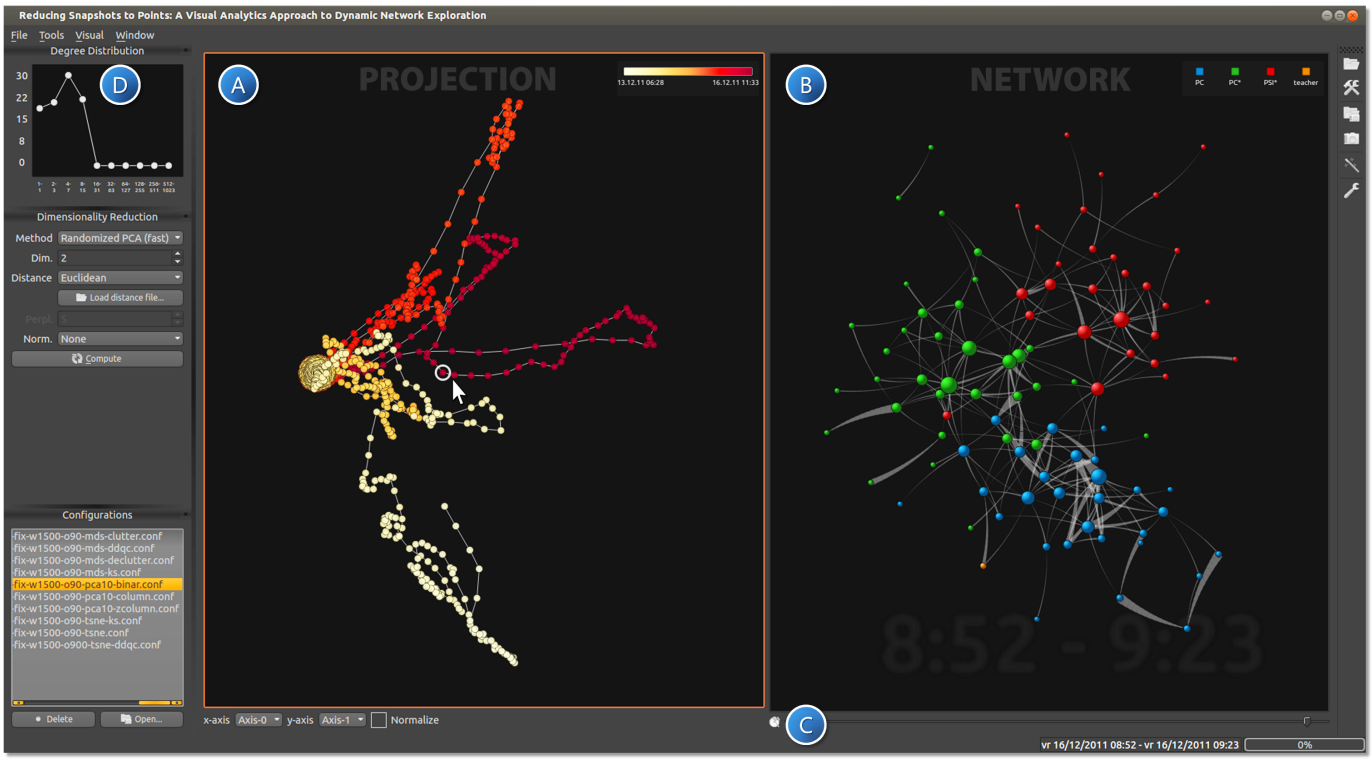


Fig. 6. Graphical user interface of the prototype implementing the different components of the visual analytics approach with a) the projection view with snapshots of the dynamic network reduced to points, b) the linked network view with a node-link diagram of a selected snapshot in the projection view, and c) linked time control, and d) auxiliary degree distribution view for a selected snapshot.

without normalization and leave further exploration for future work. Also, it is not strictly necessary, because every feature is measured on the same scale and unit. However, if derived attributes are added it is advised to apply normalization first.

3.5 Dimensionality Reduction

Our network snapshots are points in high-dimensional space but multiple dimensions are hard to comprehend and difficult to visualize. Therefore we use dimensionality reduction techniques [57], which project points to lower dimensional subspaces such that data characteristics of the features in the lower dimensional subspace approximate geometric characteristics of the data in the original high-dimensional space. Our goal is to reduce the feature space to two dimensions and project the snapshots as points for visualization and interaction.

There are many linear and non-linear dimensionality reduction techniques that can be used, such as *Principal Component Analysis* (PCA) [32, 39], *Multidimensional Scaling* (MDS) [35], or *t-Distributed Stochastic Neighbour Embedding* (t-SNE) [56]. The computation times for both PCA and t-SNE can be strongly reduced by using improved variants such as *Randomized* PCA [30, 43] and *Barnes-Hut*-SNE [55], respectively. This makes them usable for large datasets and enables interactive real-time analysis. PCA is a linear dimensionality reduction technique, i.e., the resulting dimensions are linear combinations of the original dimensions such that the variance of the data is described best. This restriction can be overcome by applying a kernel-trick [47] to achieve non-linearity [46].

A recent comparison reveals that nonlinear dimensionality reduction techniques perform well on selected artificial tasks, but PCA still has a better performance on real-world datasets [57]. Therefore, we choose for (Randomized) PCA as initial dimensionality reduction technique, but also provide MDS and t-SNE in our application, using source code provided by the authors [55, 62]. Both MDS and t-SNE have the advantage that besides the standard Euclidean distance between network snapshots, more sophisticated network distance measures can be directly employed with the use of a pre-defined distance matrix. For example, we can precompute distances based on degree distribution. As a measure for this, the widely used Kolmogorov-Smirnov test [34], or the more recent Degree Distribution Quantification and Comparison (DDQC) measure [2] can be used. This provides us with a more structural comparison of the network over time, see Figure 5.

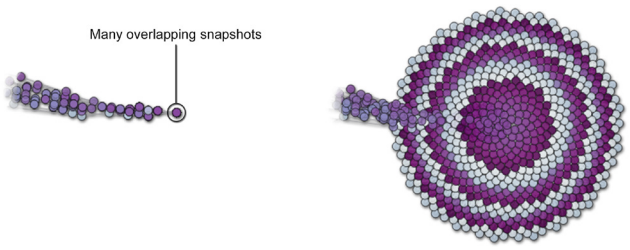
The dimensions of the resulting dimensionality reduction algorithm are often difficult to interpret, especially for the non-linear dimensionality reductions such as MDS and t-SNE. However, this is not an issue here, because we are not interested in the dimensions but rather in the clusters and outliers of the resulting projection of the dynamic network snapshots. For analysis and exploration, the snapshots, now reduced to points, are visualized in the resulting two-dimensional space.

3.6 Visualization and Interaction

To enable the exploration of the dynamic network we use two linked juxtaposed views implemented in a prototype application. Figure 6 shows the graphical user interface of the prototype with the *projection* view and the *network* view.

Projection View

The projection view visualizes each snapshot of the dynamic network as a dot. Dots that are close to each other indicate that the snapshots have a similar network state. Sequential dots in time can be connected with a line to emphasize transitional patterns between clusters. This also enables the identification of snapshots close in time, but far apart in the projection view and vice versa. Dots are assigned a user defined colormap that is initially set to index in time. By coloring dots according to time more structure in the clusters can be identified. If a cluster has a (nearly) uniform color, this indicates a stable state in the network, i.e., the network stays the same for a longer period. If a cluster consists of multiple colors, it reveals a recurring state. Next to color-mapping by global time index as described above, other options are available such as coloring dots according to hour of the day to reveal day patterns. Finally, dots could also be colored according to a derived attribute such as network density to reveal structure.



(a) Standard projection. (b) Emphasizing clusters.

Fig. 7. Simplifying the identification of states in the network by emphasizing clusters in the projection view by repositioning nodes with equal coordinates. a) Many snapshots containing similar network states that are given equal coordinates and b) after repositioning for better identification of the network state.

Exploration is supported by zoom-and-pan techniques. Zooming facilitates the close up inspection of clusters of snapshots. To simplify the identification of clusters we de-clutter the projection view by repositioning points with equal positions using Phyllotactic arrangement techniques [59]. This results in the emergence of clusters that emphasize the states in the network, see Figure 7.

We use two dimensional projections to prevent clutter and to keep navigation techniques simple. However, different axes resulting from the dimensionality reduction can be selected to project the snapshots. For example, it may be fruitful to check dimensions beyond the first two dimensions if PCA is used. Additionally, time can be selected as one of the axes of the projection. Upon repositioning of the points due to switching between different projections the users mental map is preserved by using animation to support the impression of one unified space and also simplifying the tracing of points. Dots can be selected and highlighted to show the associated network snapshot in the network view.

Network view

The network view visualizes the network of the selected snapshot using a node-link diagram where each dot represents a node of the network and each line an edge. The network configuration is computed using a user selectable force-directed graph layout [25].

The nodes in the network view can be set to be visible and static for each snapshot or be repositioned on each newly selected snapshot. The advantage of recomputing the layout is that it often results in a clearer visualization of the structure of the network, however, this might break the preservation of the mental map. Stability of the nodes is supported by initializing each node position to the current position when computing a new force-directed layout. If nodes are repositioned, animation is used to make tracking of changes easier. Also, a static layout based on the union of all snapshots can be used.

Nodes in the network can be colored according to an associated multivariate attribute value. Similar to the projection view, exploration is enabled by zoom-and-pan techniques. Furthermore, the start- and end-time of the highlighted snapshot is rendered in the network view to provide context. Simple linking and brushing techniques are implemented to enable visual querying. Brushing nodes in the network view highlights snapshots in the projection view according to user selectable rules: a snapshot is highlighted if a) one or more nodes have a link, b) all nodes have a link, c) one or more nodes have a link with each other, d) all nodes are linked to each other. This enables users to visually perform queries on the behavior of particular nodes over time.

In addition to the projection view and the network view we implemented two auxiliary views, a linked *timeline* control and *degree distribution* view. The timeline view provides a context and enables to scan through the dynamic network. The degree distribution view shows the degree distribution of the network that is highlighted in the projection view and shown in the network view. This enables the comparison of snapshots on a more structural level and proves useful if alternative distance measures are used based on degree distribution, such as in Figure 5.

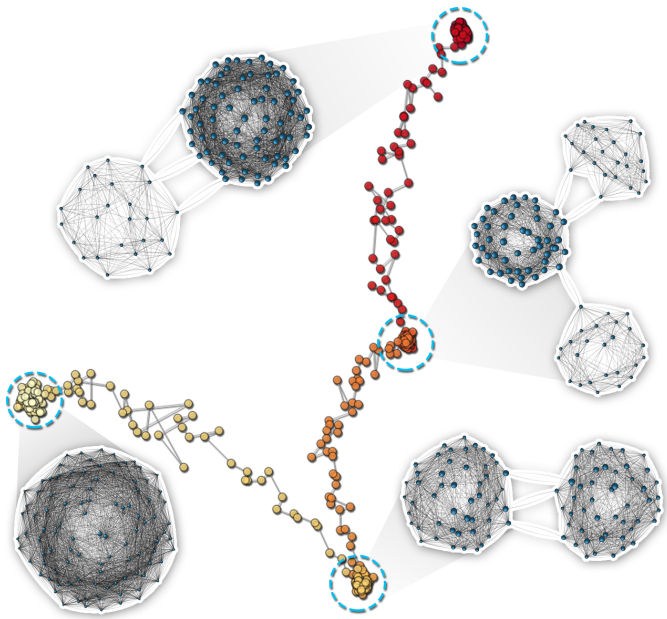
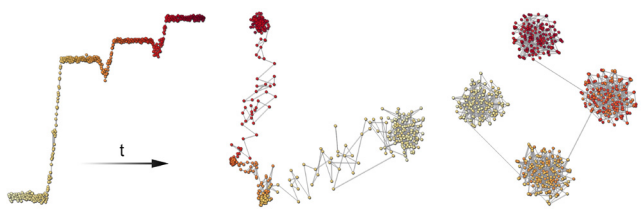


Fig. 8. Linear reduction with PCA reveals four stable network states and transitions between them. Gray insets show representative snapshots.



(a) Time. (b) Normalization. (c) Non-linear.

Fig. 9. Alternative projections of the dynamic network snapshots: a) time vs. 1^{st} principal component, b) PCA with z-normalization, and c) non-linear dimensionality reduction with t-SNE.

4 Use Cases

We have applied our approach to artificial and real-world datasets. We present results and discuss choices for parameters. The visual analytics approach helps in understanding the network dynamics by revealing stable states, recurring states, outlier states and provides insight on the evolution of the networks in general.

4.1 Artificial Dynamic Networks

We created different artificial dynamic network datasets to test and evaluate the visual analytics approach. We show the results of our approach on two of such datasets here, a third example is shown in Figure 1. In our network creation model we define the number of nodes and number of stable states in the dynamic network. A stable network state is created by using the small world model of Newman, Watts, and Strogatz [38, 63]. Next, small variants of the resulting network are created for a number of user defined timesteps. After creation of a number of these stable states, transitional network states are created

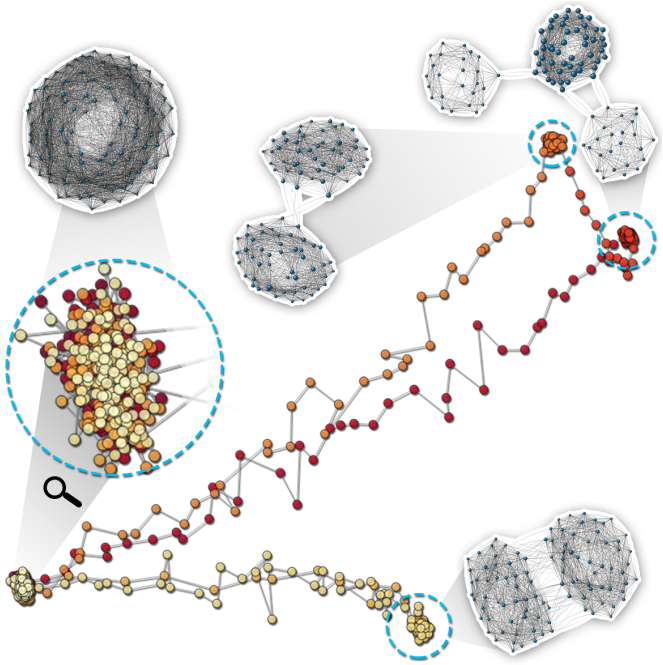
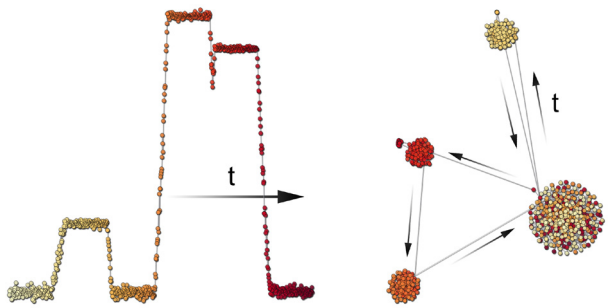


Fig. 10. Linear reduction PCA reveals four network states, three stable and one recurring state. The largest blue circle shows a zoomed-in version of the recurring state and the gray insets show representative snapshots.



(a) Time vs. 1^{st} principal component. (b) Non-linear reduction.

Fig. 11. Alternative projections providing more insight in the same dataset as in Figure 10.

by interpolating between the stable states over a predefined number of timesteps.

First, we create a dynamic network with 100 nodes, 9900 edges, and 650 timesteps. The dynamic network contains four stable states and three transitions; at the start the network consists of one large small-world community. Next, the community falls apart and two separate communities emerge. Then one of the smaller communities is divided into two even smaller communities. Finally, one of these joins the largest community again. All states are stable for 125 timesteps and transitions take 50 timesteps.

If PCA is applied to the network snapshots we clearly see four clusters of snapshots and the transitions between them (see Figure 8). By highlighting the snapshots in the clusters we discover the structure of the network in the network view. If we select time on the x-axis we see that the last three stable states are more similar than the first stable state as their distance to each other on the y-axis (first principal component) is smaller, see Figure 9(a). If we apply z-normalization, the structure within clusters becomes clearer, and the clusters themselves are emphasized. However, the transitions between the clusters become less clear, see Figure 9(b). Next we switch to t-SNE, Figure 9(c), and the clusters of stable states become even more clear, however, we do not see the transitions between the stable states anymore. This might be resolved after fine-tuning of the algorithm parameters.

The second dynamic network, again 100 nodes and 9900 edges, contains four states, five transitions and has a recurring state in the network that occurs at different points in time. The network starts with one big community (the recurring state) and then falls apart in two smaller communities. Next one big community is formed again after which the network again falls apart into two different smaller communities. Then the network splits into three communities and finally merges back to one big community again. The sequence consists of 900 timesteps.

First we again apply PCA on the snapshots enabling us to identify the four different stable states. We see the transitions between the states and identify one state as recurring because the cluster of snapshots consists of different colors, see Figure 10. If we switch to time on the x-axis we more clearly see the stable states and their duration in Figure 11(a). Furthermore, we now see when in time the recurring state occurs. By switching to t-SNE we see the stable states as clusters (see Figure 11(b)). The transitions between the stable states are not visible here, but we are still able to identify the sequence of states due to the connected subsequent snapshots. Also, the recurring state is more clear here due to a better spread of the points in the cluster.

4.2 High-School Contact Patterns

Two different datasets were collected from the SocioPatterns initiative [48]. Both datasets contain timestamped events of face-to-face contact between persons based on wearable sensors in context of determining how infectious diseases spread within a population. The first dataset contains face-to-face contacts of high-school students between three different classes during 4 school days in 2011 (see also Figures 3, 4, 5, and 6). The second dataset contains 7 school days of face-to-face contacts (from a Monday to the Tuesday of the following week in 2012) between students of 5 different classes, again logged at the French high-school Lycée Thiers in Marseilles [23]. The results we obtained with the two datasets are similar in nature with identical findings, therefore we only discuss the richer 2012 dataset below.

The dynamic network consists of 180 nodes (students), 45,047 contacts, and 10,104 unique edges. We create snapshots by choosing a window width ω of 60 minutes with an overlap α of 0.9. This results in 6 minutes (Δt) of new edges and dropping old edges for each subsequent snapshot. The total number of produced snapshots is 2015.

First we apply PCA to the snapshots (see Figure 12). Here we use a linear color-scale and the first thing we notice is that each day can be identified by a separate color. Furthermore, we see a big cluster of snapshots in the middle. This cluster consists of different colors, so this indicates a recurring state in the network. After inspection by highlighting, we see that this cluster represents the network during night, i.e., no activity. Each day starts at the night cluster and returns to it with a loop.

Next, we use z-normalization and apply PCA again (see Figure 12, right). Similar to before, the recurring night cluster is present and the days are visualized as loops. There is a more clear separation now between clusters far away and close to the night cluster. The snapshots in the clusters further away from the center contain more dense networks whereas the clusters closer to the center (night) contain sparse networks. If snapshots are colored according to hour of day, we see that the dense network states often occur at the start of the day and in the afternoon the network is predominantly sparse.

If we switch the x-axis to time and the y-axis to first principal component, the nights and weekend are clearly visible in Figure 13(a). It also indicates that all days have a similar variance as the day patterns have similar shape. We continue the exploration by switching to dimensionality reduction using t-SNE. Again, the night cluster appears and the days are shown as separate trails. Here we conclude that likely each day is different from one another, because no trails are overlapping or forming clusters, see Figure 14, left. We also see more struc-

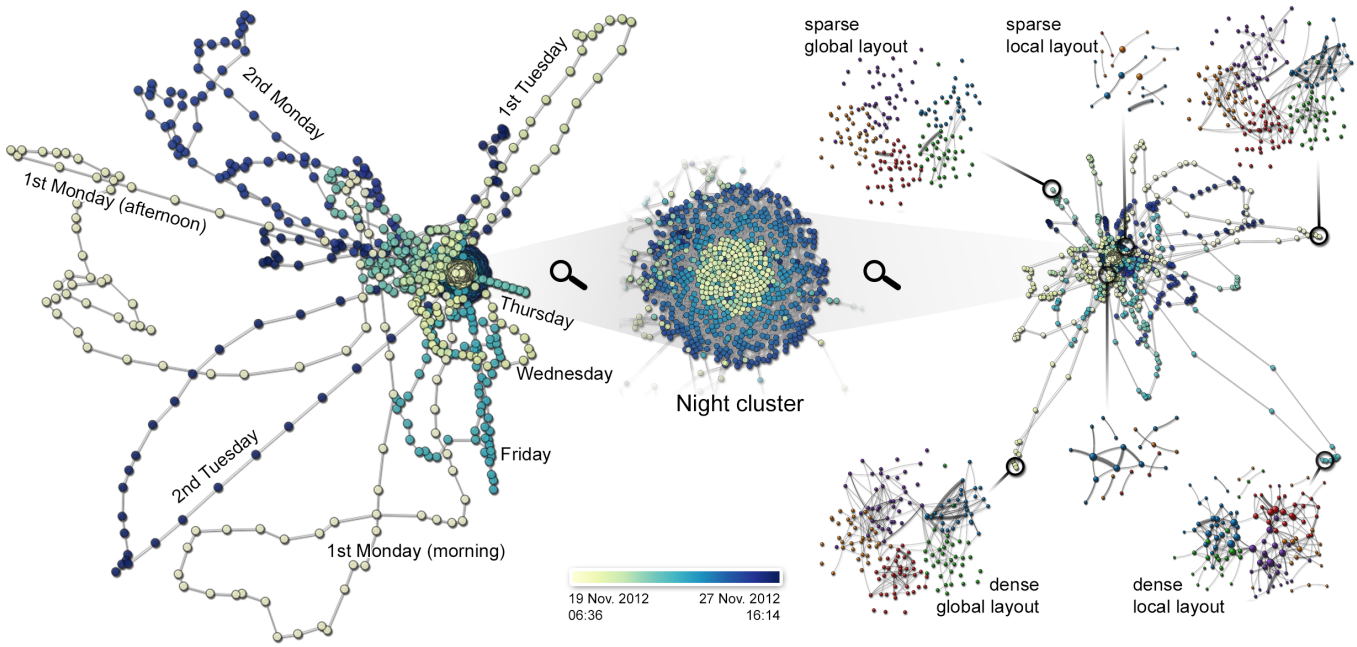
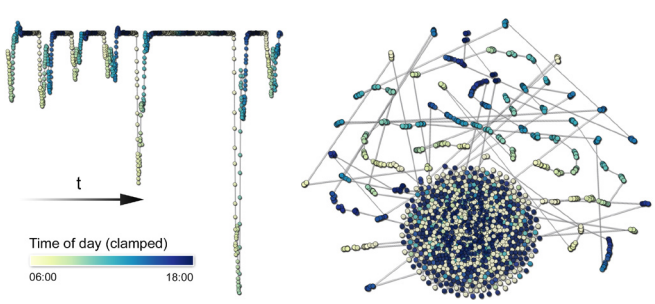


Fig. 12. Linear reduction PCA without (left) and with (right) z-normalization for the *thiers-2012* dataset. The middle cluster shows a zoomed-in version of the recurring night state. Days are identified by color and form a loop that starts and ends at the night cluster. The annotated networks show the network structure for the black-circled snapshots in the projection.

ture in the days by means of gaps in the trails. This indicates a daily rhythm, presumably caused by breaks between classes and lunches. If we color dots by hour of day we see that the gaps occur at similar time-periods in Figure 14, right. Upon closer inspection we identify a cluster of snapshots that is present in each day between 8 and 9. At this cluster of snapshots the network is very dense and the clusters before and after are both sparse. Now that snapshots are colored by hour of day we see that one trail of snapshots is different compared to the rest. The Wednesday trail is shorter and afternoon snapshots are not attached. After inspection we see that the network state at Wednesday afternoon is much sparser compared to the other days. This is explained by the fact that students take exams on Wednesday which typically last the whole afternoon without breaks [23]. Also, a cluster of snapshots is identified representing the network state at Thursday afternoon. Here, this network state represents the communication of one class only.

Finally, we apply z-normalization before applying the t-SNE dimensionality reduction and this results in many small groups of snapshots that generally span a time period of an hour (see Figure 13(b)). This indicates that classes probably last an hour. Furthermore, we see that mornings are more consistent and afternoons differ, indicated by longer trails in the morning and smaller clusters in the afternoon.



(a) Time vs. 1^{st} principal component. (b) Non-linear reduction.

Fig. 13. Alternative projections with snapshots colored by hour of day.

5 Discussion

The basic steps of the visual analytics approach presented in this paper are all existing techniques: 1) discretization, 2) vectorization and normalization, 3) dimensionality reduction, and 4) visualization and interaction. However, their combined use for the analysis of dynamic networks is novel and provides new views and insights. The cornerstone of our approach, both simple and effective, is to reduce snapshots of a dynamic network to points.

There are many different choices for the exact implementation and parameter settings of our approach, for example, how to choose the time step and window overlap in discretization, the dimensionality reduction technique, and what (derived) features to use. Many options depend on the characteristics of the dynamic network to analyze and might be best determined in a joint effort between domain expert and a visualization professional or data scientist. Still, we consider this flexibility a strong point as it enables to use the approach for a broad range of datasets and perspectives on these, and opens possibilities for future work.

We found that in general non-linear dimensionality reduction techniques work best for the reduction of snapshots to points. This has the downside that it is harder to interpret the resulting dimensions. Though, this is not that important since we do not need a precise explanation but rather the ability to visually identify clusters and outliers.

There are two types of visual scalability concerns in our method; scalability with respect to network size and number of timesteps involved. In our case we found that our approach works well for a network with 180 nodes, 10,104 edges, and 2015 timesteps. We believe our method is able to deal with both aspects of scalability, up to a certain limit depending on memory– and time–availability. If the network at each snapshot is large, then for the visualization in the network view a (sorted) visual adjacency matrix [37] or hierarchical edge bundles [31] have to be used rather than a node-link diagram. Also, the choice for window-length of the snapshots commonly influences the network size at each snapshot; in general, if window-length is chosen smaller, the resulting networks at each snapshot will also be smaller compared to larger window-lengths. Also, scalability for the number of timestamps can be controlled in the discretization step. If only a few timesteps are available the method will be less effective, since we

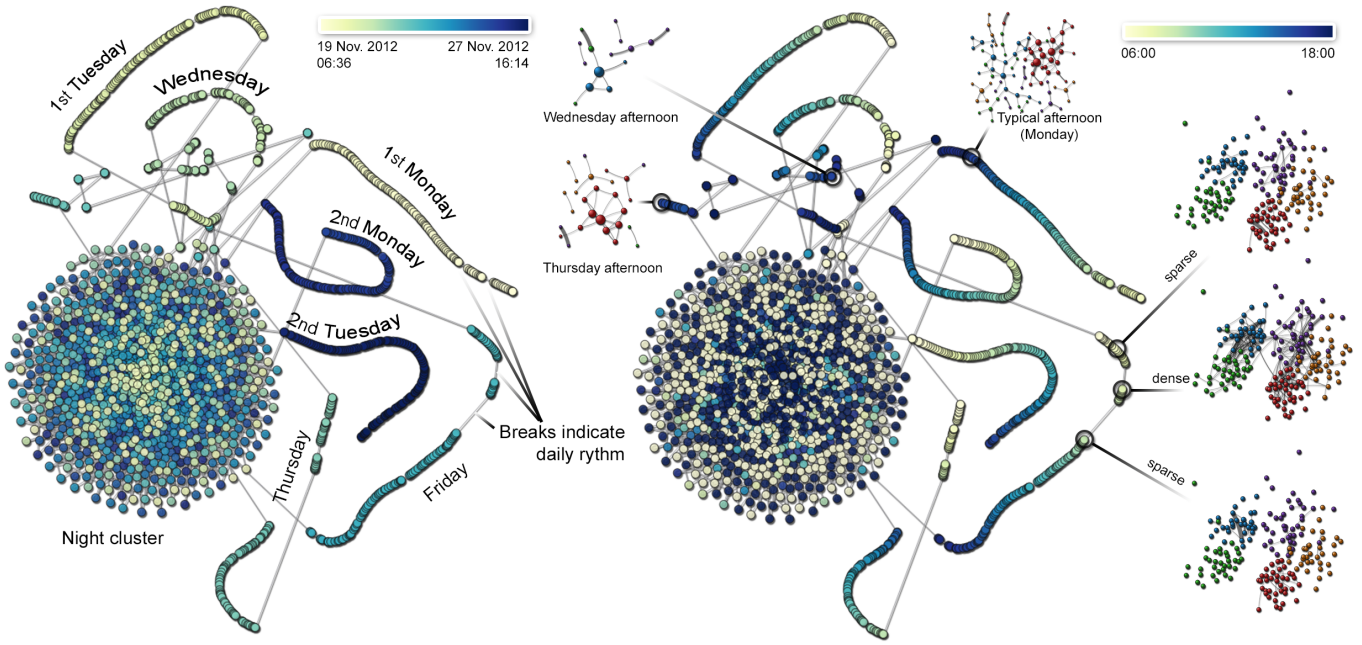


Fig. 14. Non-linear dimensionality reduction of snapshots colored globally (left) and by hour of the day (right) revealing patterns and providing insight in stable states and the dynamic network evolution. The annotated networks show the network structure for the black-circled snapshots in the projection.

rely on the visual identification of clusters. Therefore, one should aim for at least 20 timesteps.

Computational scalability of the dimensionality is addressed by using scalable variants of PCA and t-SNE, as described in Section 3.5. This enables near real-time interaction and being able to deal with large datasets.

6 Conclusions

In this paper we presented a novel visual analytics approach for dynamic network exploration. The approach consists of four different steps: *discretization*, *vectorization and normalization*, *dimensionality reduction*, and *visualization and interaction*. The crucial step in the approach that enables the exploration and analysis of dynamic networks is to reduce snapshots to points. The individual components are not new, however, the combination and their application to dynamic networks is novel. This enables users to visually identify stable and recurring states in the network and provides insight in the transitions between them. As a proof of concept the approach is implemented in a prototype. The approach is highly flexible and adaptable to users needs, e.g., how to create the snapshots and what dimensionality reduction technique to use. For the creation of snapshots we suggest to have at least 50 percent overlap of subsequent timesteps. However, this is merely a guideline and depends on the dataset at hand. We experimented with different dimensionality reduction techniques and found that standard PCA without normalization gives good results but non-linear reduction methods such as t-SNE work best. However, the general approach is independent from implementation details and with this work, deviating from standard approaches such as animation, timeline and small-multiples, we hope to have inspired new work in the area of dynamic network exploration. We have shown the effectiveness of our approach by applying it to artificial and real-world dynamic networks and were able to get insights and understanding on the evolution of the networks.

Future Work For future work there are several directions. First, the parameters involved in the different steps of the approach might be improved. Finding the best distance measure and dimensionality reduction technique is a challenge and typically depends on the dataset.

Second, the approach now relies on visual identification of clusters and outliers, which might be improved with automatic clustering in high-dimensional space, outlier detection [4], or graph similarity [11]. The visualization and interaction step might also benefit from high-dimensional visual analysis methods, e.g., a dual setup between item space and dimensions space [51]. Finally, the visualization setup might profit from other linked views showing network complexity properties.

References

[1] J. Abello, S. Hadlak, H. Schumann, and H.-J. Schulz. A Modular Degree-of-Interest Specification for the Visual Analysis of Large Dynamic Networks. *IEEE Trans. Vis. Comput. Graphics*, 20(3):337–350, March 2014.

[2] S. Aliakbary, J. Habibi, and A. Movaghar. Feature Extraction from Degree Distribution for Comparison and Analysis of Complex Networks. *The Computer Journal*, 2015.

[3] G. Andrienko, N. Andrienko, S. Bremm, T. Schreck, T. von Landesberger, P. Bak, and D. Keim. Space-in-time and Time-in-space Self-organizing Maps for Exploring Spatiotemporal Patterns. In *IEEE Eurographics Proc. Conf. Visualization*, pages 913–922, 2010.

[4] F. Angiulli and C. Pizzuti. Fast Outlier Detection in High Dimensional Spaces. In T. Elomaa, H. Mannila, and H. Toivonen, editors, *Principles of Data Mining and Knowledge Discovery*, volume 2431 of *Lecture Notes in Computer Science*, pages 15–27. Springer Berlin Heidelberg, 2002.

[5] D. Archambault, J. Abello, J. Kennedy, S. Kobourov, K.-L. Ma, S. Miksch, C. Muelder, and A. Telea. Temporal Multivariate Networks. In A. Kerren, H. C. Purchase, and M. O. Ward, editors, *Multivariate Network Visualization*, volume 8380 of *Lecture Notes in Computer Science*, pages 151–174. Springer International Publishing, 2014.

[6] B. Bach, J.-D. Fekete, and E. Pietriga. GraphDiaries: Animated Transitions andTemporal Navigation for Dynamic Networks. *IEEE Trans. Vis. Comput. Graphics*, 20(5):740–754, 2014.

[7] B. Bach, N. Henry-Riche, T. Dwyer, T. Madhyastha, J.-D. Fekete, and T. Grabowski. Small MultiPiles: Piling Time to Explore Temporal Patterns in Dynamic Networks. *Comput. Graph. Forum*, 2015.

[8] B. Bach, E. Pietriga, and J.-D. Fekete. Visualizing Dynamic Networks with Matrix Cubes. In *Proc. SIGCHI Conf. Human Factors in Computing Systems*, CHI, pages 877–886, New York, NY, USA, 2014. ACM.

[9] F. Beck, M. Burch, S. Diehl, and D. Weiskopf. The State of the Art in Visualizing Dynamic Graphs. In *EuroVis - STARs*, pages 83–103, 2014.

[10] J. Bertin. *Semiology of graphics: diagrams, networks, maps*. University of Wisconsin Press, 1983.

[11] C. C. Bilgin and B. Yener. Dynamic network evolution: Models, clustering, anomaly detection. *IEEE Networks*, 2006.

[12] M. Burch, F. Beck, and S. Diehl. Timeline Trees: Visualizing Sequences of Transactions in Information Hierarchies. In *Proc. Working Conf. Advanced Visual Interfaces*, pages 75–82, New York, USA, 2008. ACM.

[13] M. Burch and S. Diehl. TimeRadarTrees: Visualizing Dynamic Compound Digraphs. *Comput. Graph. Forum*, 27(3):823–830, 2008.

[14] M. Burch, M. Fritz, F. Beck, and S. Diehl. TimeSpiderTrees: A Novel Visual Metaphor for Dynamic Compound Graphs. In *IEEE Symp. Visual Languages and Human-Centric Computing*, pages 168–175, 2010.

[15] M. Burch, M. Hoferlin, and D. Weiskopf. Layered TimeRadarTrees. In *Proc. 15th Int. Conf. Information Visualisation*, pages 18–25, 2011.

[16] M. Burch, C. Vehlow, F. Beck, S. Diehl, and D. Weiskopf. Parallel Edge Splatting for Scalable Dynamic Graph Visualization. *IEEE Trans. Vis. Comput. Graphics*, 17(12):2344–2353, 2011.

[17] M. Burch and D. Weiskopf. Visualizing Dynamic Quantitative Data in Hierarchies - TimeEdgeTrees: Attaching Dynamic Weights to Tree Edges. In *Proc. Int. Conf. Information Vis. Theory App.*, pages 177–186, 2011.

[18] S. Diehl and C. Görg. *Graphs, They Are Changing*, volume 2528, pages 23–30. Springer-Verlag, 2002.

[19] T. Dwyer and P. Eades. Visualising a Fund Manager Flow Graph with Columns and Worms. In *Proc. 6th Int. Conf. Information Vis.*, pages 147–152, 2002.

[20] T. Falkowski, B. Bartelheimer, and M. Spiliopoulou. Mining and Visualizing the Evolution of Subgroups in Social Networks. In *IEEE/WIC/ACM Int. Conf. Web Intelligence*, pages 52–58, Dec. 2006.

[21] M. Farrugia, N. Hurley, and A. Quigley. Exploring Temporal Ego Networks Using Small Multiples and Tree-ring Layouts. In *Proc. 4th Int. Conf. Advances in Human Computer Interfaces*, 2011.

[22] P. Federico, W. Aigner, S. Miksch, F. Windhager, and L. Zenk. A Visual Analytics Approach to Dynamic Social Networks. In *Proc. 11th Int. Conf. Knowledge Management and Knowledge Technologies*, pages 1–8, New York, NY, USA, 2011. ACM.

[23] J. Fournet and A. Barrat. Contact Patterns among High School Students. *PLoS ONE*, 9(9):e107878, 09 2014.

[24] Y. Frishman and A. Tal. Dynamic Drawing of Clustered Graphs. In *Proc. IEEE Symp. Information Visualization*, pages 191–198, Washington, DC, USA, 2004. IEEE Computer Society.

[25] E. R. Gansner and S. C. North. An Open Graph Visualization System and its Applications to Software Engineering. *Software - Practice and Experience*, 30(11):1203–1233, 2000.

[26] R. Gove, N. Gramsky, R. Kirby, E. Sefer, A. Sopan, C. Dunne, B. Shneiderman, and M. Taieb-Maimon. NetVisia: Heat Map & Matrix Visualization of Dynamic Social Network Statistics & Content. In *Proc. 3th Int. Conf. Social Computing*, pages 19–26, 2011.

[27] M. Greilich, M. Burch, and S. Diehl. Visualizing the Evolution of Compound Digraphs with TimeArcTrees. *Comput. Graph. Forum*, 28(3):975–982, 2009.

[28] G. Groh, H. Hanstein, and W. Wörndl. Interactively Visualizing Dynamic Social Networks with DySoN. In *Proc. Work. Visual Interfaces to the Social and the Semantic Web*, VISSW, 2009.

[29] S. Hadlak, H. Schumann, C. Cap, and T. Wollenberg. Supporting the Visual Analysis of Dynamic Networks by Clustering associated Temporal Attributes. *IEEE Trans. Vis. Comput. Graphics*, 19(12):2267–2276, 2013.

[30] N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding Structure with Randomness: Probabilistic Algorithms for Constructing Approximate Matrix Decompositions. *ArXiv e-prints*, Sep. 2009.

[31] D. Holten. Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data. *IEEE Trans. Vis. Comput. Graphics*, 12(5):741–748, 2006.

[32] H. Hotelling. Analysis of a Complex of Statistical Variables into Principal Components. *J. Educ. Psych.*, 24, 1933.

[33] N. Kerracher, J. Kennedy, and K. Chalmers. The Design Space of Temporal Graph Visualisation. In N. Elmqvist, M. Hlawitschka, and J. Kennedy, editors, *EuroVis - Short Papers*. The Eurographics Association, 2014.

[34] G. Kossinets and D. J. Watts. Empirical Analysis of an Evolving Social Network. *Science*, 311(5757):88–90, 2006.

[35] J. Kruskal. Multidimensional Scaling by Optimizing Goodness of Fit to a Nonmetric Hypothesis. *Psychometrika*, 29(1):1–27, 1964.

[36] G. Kumar and M. Garland. Visual Exploration of Complex Time-Varying Graphs. *IEEE Trans. Vis. Comput. Graphics*, 12(5):805–812, 2006.

[37] I. Liiv. Seriation and Matrix Reordering Methods: An Historical Overview. *Statistical Analysis and Data Mining*, 3(2):70–91, 2010.

[38] M. Newman and D. Watts. Renormalization Group Analysis of the Small-World Network Model. *Physics Letters A*, 263(46):341–346, 1999.

[39] K. Pearson. LIII. On Lines and Planes of Closest Fit to Systems of Points in Space. *Philosophical Magazine Series 6*, 2(11):559–572, 1901.

[40] M. Pohl, F. Reitz, and P. Birke. As Time Goes By: Integrated Visualization and Analysis of Dynamic Networks. In *Proc. Working Conf. Advanced Vis. Interfaces*, pages 372–375. ACM, 2008.

[41] F. Reitz, M. Pohl, and S. Diehl. Focused Animation of Dynamic Compound Graphs. In *Proc. 13th Int. Conf. Information Visualisation*, pages 679–684, 2009.

[42] G. Robertson, R. Fernandez, D. Fisher, B. Lee, and J. Stasko. Effectiveness of Animation in Trend Visualization. *IEEE Trans. Vis. Comput. Graphics*, 14(6):1325–1332, 2008.

[43] V. Rokhlin, A. Szlam, and M. Tygert. A Randomized Algorithm for Principal Component Analysis. *SIAM J. Matrix Anal. Appl.*, 31(3):1100–1124, Aug. 2009.

[44] M. Rosvall and C. T. Bergstrom. Mapping Change in Large Networks. *PLoS ONE*, 5(1):e8694, 2010.

[45] S. Rufiange and G. Melançon. AniMatrix: A Matrix-Based Visualization of Software Evolution. In *Proc. 2nd IEEE Work. Conf. Software Vis.*, pages 137–146, 2014.

[46] B. Schölkopf, A. J. Smola, and K.-R. Müller. Advances in Kernel Methods. chapter Kernel Principal Component Analysis, pages 327–352. MIT Press, Cambridge, MA, USA, 1999.

[47] J. Shawe-Taylor and N. Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA, 2004.

[48] SocioPatterns. website. `http://www.sociopatterns.org/datasets/`, 2015. [Online; accessed 2015-03-18].

[49] M. Steiger, J. Bernard, S. Mittelstdt, H. Lcke-Tieke, D. Keim, T. May, and J. Kohlhammer. Visual Analysis of Time-Series Similarities for Anomaly Detection in Sensor Networks. *Comput. Graph. Forum*, 33(3):401–410, 2014.

[50] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press, second edition, 2001.

[51] C. Turkay, P. Filzmoser, and H. Hauser. Brushing Dimensions – A Dual Visual Analysis Model for High-Dimensional Data. *IEEE Trans. Vis. Comput. Graphics*, 17(12):2591–2599, Dec. 2011.

[52] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk. Reordering Massive Sequence Views: Enabling temporal and structural analysis of dynamic networks. In *IEEE Pacific Vis. Symp.*, pages 33–40, Feb 2013.

[53] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk. Dynamic Network Visualization with Extended Massive Sequence Views. *IEEE Trans. Vis. Comput. Graphics*, 20(8):1087–1099, Aug 2014.

[54] S. van den Elzen and J. J. van Wijk. Small Multiples, Large Singles: A New Approach for Visual Data Exploration. *Comput. Graph. Forum*, 32(3pt2):191–200, 2013.

[55] L. van der Maaten. Barnes-Hut-SNE. In *Int. Conf. Learning Representations*, 2013.

[56] L. van der Maaten and G. E. Hinton. Visualizing High-Dimensional Data Using t-SNE. *J. Machine Learning Research*, 9:2579–2605, 2008.

[57] L. Van der Maaten, E. Postma, and H. Van den Herik. Dimensionality Reduction: A Comparative Review. *Technical Report TiCC TR 2009-005*, 2009.

[58] J. J. van Wijk and E. Van Selow. Cluster and calendar based visualization of time series data. In *IEEE Proc. Symp. Information Visualization*, pages 4–9, 140, 1999.

[59] H. Vogel. A Better Way to Construct the Sunflower Head. *Mathematical Biosciences*, 44(3–4):179–189, 1979.

[60] T. von Landesberger, S. Diel, S. Bremm, and D. W. Fellner. Visual analysis of contagion in networks. *Information Visualization*, 14(2):93–110, 2015.

[61] T. von Landesberger, A. Kuijper, T. Schreck, J. Kohlhammer, J. J. van Wijk, J.-D. Fekete, and D. Fellner. Visual Analysis of Large Graphs: State-of-the-Art and Future Research Challenges. 2011.

[62] Q. Wang and K. Boyer. Feature Learning by Multidimensional Scaling and Its Applications in Object Recognition. In *Proc. 26th Conf. Graph., Patterns and Images*, pages 8–15, 2013.

[63] D. Watts and S. Strogatz. Collective Dynamics of 'Small-World' Networks. *Nature*, 393(6684):440–442, 1998.