# Interactive Time-Series of Measures for Exploring Dynamic Networks

Liwenhan Xie
liwenhan.xie@pku.edu.cn
Peking University

James O'Donnell
james88od@gmail.com
University of Edinburgh

Benjamin Bach
bbach@inf.ed.ac.uk
University of Edinburgh

Jean-Daniel Fekete
Jean-Daniel.Fekete@inria.fr
Université Paris-Saclay, CNRS, Inria, LRI

## ABSTRACT

We present *MeasureFlow*, an interface to visually and interactively explore dynamic networks through time-series of network measures such as link number, graph density, or node activation. When networks contain many time steps, become large and more dense, or contain high frequencies of change, traditional visualizations that focus on network topology, such as animations or small multiples, fail to provide adequate overviews and thus fail to guide the analyst towards interesting time points and periods. MeasureFlow presents a complementary approach that relies on visualizing time-series of common network measures to provide a detailed yet comprehensive overview of when changes are happening and which network measures they involve. As dynamic networks undergo changes of varying rates and characteristics, network measures provide important hints on the pace and nature of their evolution and can guide an analysts in their exploration; based on a set of interactive and signal-processing methods, MeasureFlow allows an analyst to select and navigate periods of interest in the network. We demonstrate MeasureFlow through case studies with real-world data.

## CCS CONCEPTS

• **Human-centered computing** → *Visual analytics*; *Visualization design and evaluation methods.*

## KEYWORDS

dynamic networks, exploratory data analysis, network measures, interaction, signal processing

**ACM Reference Format:**
Liwenhan Xie, James O'Donnell, Benjamin Bach, and Jean-Daniel Fekete. 2020. Interactive Time-Series of Measures for Exploring Dynamic Networks. In *International Conference on Advanced Visual Interfaces (AVI '20), September 28-October 2, 2020, Salerno, Italy*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3399715.3399922


*AVI '20, September 28-October 2, 2020, Salerno, Italy*

ACM ISBN 978-1-4503-7535-1/20/09…$15.00
https://doi.org/10.1145/3399715.3399922

## 1 INTRODUCTION

Dynamic networks, also called temporal graphs, represent relational data changing over time. The types of changes can be broad and diverse, and many visualization techniques have been developed to cope with the complexity of understanding temporal change in relational data [11]. In this paper, we focus on networks with event sequences, rather than networks that exist as temporal snapshots; while snapshots contain a complete network topology for each time step, including several nodes and links, event sequences represent dynamic networks as sequences of links denoting individual and time-limited interactions between nodes.

Event sequence networks are common in the social sciences, including twitter-retweet networks, networks of letter correspondences or contracts and other human interactions, as well as networks from the sciences where data is taken at non-uniform intervals, such as during the development of embryos that evolve on a faster pace in the early stages after conception [3]. In many of these examples, we found two major and widely unaddressed challenges, **C1:** links are defined by a relatively small temporal granularity with respect to the length of the entire network, *e.g.,* milliseconds, hours, or days, and **C2:** events are not distributed evenly across the duration of the network.

Most visualization techniques focus on network topology, *e.g.,* small multiples [12], animation [7], space-time cubes [4]. However, as networks become large, dense, involve many events, and exhibit diverse temporal trends, such topology-based visualizations fail to provide appropriate interfaces to explore the networks as visual complexity of the topology increases, and visual or interactive exploration of the network becomes hard through the amount of change and long sequences of animations or small multiples. In these examples, choosing a meaningful time-interval size is key but is usually either set by default or selected with little data-backed rationale, rather than based on informed data exploration.

To address these issues, this paper presents an approach to explore and analyze dynamic event sequence networks by guiding user exploration and analysis through visualizing network measures over time. Rather than relying on the network topology, we visualize how network measures—such as density, node-degree, or link activation—change over time. Not only does this provide us with a complementary view on network evolution that can show trends and periods in the network at different time granularities, but it also aims to guide an analyst to time points of interest. However, in trying to visualize network measures over time, we identified two main problems:

- **P1:** While some measures are *scalar* and over time can be displayed as time series (e.g., density), other measures are *vectors* (e.g., nodes degree), requiring more careful design choices.
- **P2:** Moreover, measure time series are only meaningful when time intervals have been well defined, i.e., are equidistant and meaningful. However, in many cases, time intervals are not well defined. In such cases, especially where events are distributed non-uniformly across time, interpreting and visualizing measures is non-trivial and requires a flexible choice of intervals.

In order to develop interactive and visual solutions for P1 and P2, we create a task taxonomy (Section 3) and worked with analysts in twitter networks (A1), historical social networks (A2, A3), and network science (A4) and went through an iterative design phase. The resulting interface, *MeasureFlow*, is contributing

- **Novel visualizations** for showing a diverse range of measures over time addressing P1 and P2;
- **Slicing strategies** (manual and automatic) to explore a dynamic network at different temporal granularities; and
- An **interactive approach** to guide users in finding a good separation for snapshots, where users may discover particular periodicity in the data and slice the network properly in service for further analysis.

We evaluate MeasureFlow through case studies on the analysts' networks (Section 6).

## 2 RELATED WORK

We review existing works that are most related to our method. These include techniques developed for the exploration of dynamic network evolution, graph visualizations with a focus on numerical measures, dynamic network slicing approaches.

### 2.1 Visualizing Dynamic Network Evolution

Static networks are commonly represented as node-link diagrams or matrices [24]. To further encode time, the two most common families in dynamic network visualization are animation [7] and small multiples [5, 21, 26], well summarized by Beck et al. [11]. While the animation plays a sequential walk-through through the network, in small multiples-based approaches, the dynamic network is regarded as a series of graphs composed by connections during a certain period, which is called "snapshots" (interval). According to a qualitative study from Boyandin et al. [12], animation is more effective for explorations on neighboring time steps while small multiples is more capable for suggesting findings on snapshots lasting over longer periods. Other than focusing on topology alone, our approach visualizes measures first, since that provides for a complementary approach that scales with size, length, and general topological complexity of the network.

In addition to topology focused visualizations, some visualization techniques try to scale to large networks, by demonstrating alternative approaches to visualize dynamic network evolution. For example, to scale with size, length, and density, Burch et al. [13] proposed parallel edge splatting to present the dynamic network in a sequence of narrow strips highlighting recurrent patterns in the network topology. Similarly, Massive Sequence Views [30] and Cui et al. [14] propose approaches to avoid clutter and to scale to very large networks based on dense visual summaries of changes [30] and the calculation of clusters [14]. In the most extreme case, each network snapshot (time step) can be collapsed into an individual 'points', laid out in a lower-dimensional space using multi-dimensional scaling [8, 19, 31]. However, abstracting networks comes to the cost of detail and involves decisions (e.g., parameters) opaque the to the user. Instead, our approach aims to provide analysts with transparency over choosing time windows.

### 2.2 Measures in Network Visualization

Summary statistics or metrics are highly condensed values, quantitatively measuring a certain aspect of the network and are common in graph visualization systems. For static networks, measurement visualizations have been well explored. For instance, Network Repository [27], a large open graph dataset, supported interactive examination on elaborate measures. Another example, Social Action [25], integrated statistics with network visualizations for basic operations like filtering to extract communities and super graphs. However, in these existing approaches, measures are visualized for static graphs only, often by coloring nodes. HoneyComb [32] used a probabilistic-based metric into a scalable adjacency matrix visualization for large social networks, to suggest the randomness of communities inside the network.

GraphPrism [20] generalized the idea of B-matrix [9] with multiple graph measures for static networks. A B-matrix is composed of stacked histograms, each of which accords to a certain local (link-specific or node-specific) measure distribution in different sizes of neighboring area. ManyNets [17] allows for the comparison of static networks by visualizing global measures such as, density or node number in a table overview. Dynamic networks could be visualized using ManyNets as each snapshot in a dynamic network can be considered a static network. Our approach is related but addresses issues of creating and interactively exploring time intervals (snapshots), visualizing measures at several levels of aggregation simultaneously, and providing additional interactions and visualizations to cope with problems in dynamic networks, such as removing empty interval.

Most similar to our work and applied to dynamic networks, the number of added and removed links have been visualized as bar charts along a timeline in TempoVis [2]. In addition to links, we visualize a set of additional measures as explained in Section 4, refine the visualization of measures (P1) and support the automatic creation of time slices (P2).

### 2.3 Dynamic Network Slicing

How to "slice" the time range properly is crucial to understand a dynamic network, as the number and size of time slices (snapshots) influences the topology shown and measures taken for each time slice [23]. For example, in Gephi [10], the size of the time window for dynamic network animation is left to the user by specifying an arbitrary time window. To generalize our method to arbitrary dynamic networks and, *e.g.,* to retrieve information about the periodicity of events, we adopt uniform slicing in MeasureFlow. However, rather than giving a fixed interval, we provide more space for users to decide a proper slicing (see Section 5), as suggested by Devineni et al. [15], there is no absolute optimal slicing scheme.

Several works are devoted to slicing dynamic networks non-uniformly. For example, in MultiPiles [5], a dynamic network can be sliced either in a greedy manner, grouping neighboring snapshots that are similar in the adjacent matrices, or through user-defined manual slicing like cutting a video clip. More recently, work from Wang et al. [33] proposed to equalize the number of temporal links and derive cutting points that promote to find snapshots with consistent scales. The process is similar to the histogram equalization technique in digital image processing. Interactively creating non-uniform time intervals will be implemented into our tool in the future but it is not the scope of our current research.

## 3 TASK SUPPORT

Our main goals are to facilitate the exploration of potentially long, large, and dense networks. On a high-level this aims to, *e.g.,* find interesting time points or meaningful time granularity for further analysis. To that end, we aim to interactively support lower-level tasks across three dimensions: *time*, *measures*, and *subgraphs*.

- **T1. Analyse time** involves observing temporal trends in network measures on a global level (including the entire network) and local levels (nodes and subgraphs). Example tasks include:
  - T1.1 Understand overall **trends** in a network, e.g., with respect to size, density and stability.
  - T1.2 Find **outlier** time points, time periods, and subgraphs that do not follow a general trends;
  - T1.3 Assess **regularities** over time and find patterns of periodicity in measures;
  - T1.4 Find **breakpoints** between potential stages in the network's evolution.
- **T2. Analyse measures** involves assessing and comparing measures at individual time points or across time. Example tasks include:
  - T2.1 Describe the evolution of **a measure over time**; understand where it spikes and what its overall trend is.
  - T2.2 Identify the **value of a measure at a given time point or over a given time period** (e.g., a month, a year, a user defined period).
  - T2.3 **Compare two measures** at a given time point or over time to look for temporal behaviors, like correlation and covariance.
- **T3. Analyse subgraphs** involves tracking the evolution of subgraphs over time. Example tasks include:
  - T3.1 Assess the value of **a measure for a given subgraph** at a given time point (or period);
  - T3.2 **Compare the evolution of subgraphs** using individual measures and across measures;

Our list of tasks is not meant to be complete, nor do we think that these tasks can necessarily be categorized into strictly separated categories, classes, and taxonomies. More tasks and combinations of tasks can be generated or described, including related taxonomies for network [22], temporal networks [1, 7] or visualization in general. Still, to our knowledge, no description of tasks exist for the analysis of network measures in visualization. More importantly, our taxonomy is describing a space around the three aspects or $time \times measures \times network$ elements and possible combinations thereof. Our taxonomy provides us with a systematic framework to generate example tasks and to inform the design of new visualizations for dynamic networks (Section 5).

## 4 DYNAMIC NETWORK MEASURES

### 4.1 Dynamic Networks

Generally, we can identify three ways to define a dynamic graph: (1) *event sequences* $E = e_1, ..., e_n$ where each *event* $e_i$ represents a link $l_{jk}$ between two nodes $l$ and $k$ at a specific time point $t$, (2) *interval graphs* where non-overlapping intervals $[t_i, t_{i+w}]$, $[t_{i+w+1}, t_{i+2n}]$, ... are defined on an sequence of events $E$ with the time window size $w$, and (3) *snapshots*, i.e., a sequence of static networks to every time point $t$, without keeping information about lower level temporal granularities. In our approach, we focus on designs for creating intervals by interactively aggregating event sequences through uniform intervals (see Fig. 1) as explaind in Section 5.

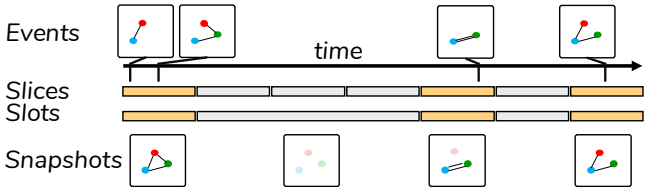


**Figure 1: Illustration of interval generation. Events happen at any time point. We slice the time line into intervals according to a fixed interval length (top). Neighboring slices without events (gray bars) are aggregated into a single empty time interval to facilitate navigation (bottom).**

### 4.2 Graph Measures

Numerous network measures exist in network analysis, each expressing a different graph property, and analysts often formulate and calculate their own measures. In this section, we overview the measures incorporated in MeasureFlow. The major consideration is the popularity and computation complexity of the measures but other measures can be included.

Generally, measures for networks can be divided into global measures and local measures. Global measures are calculated over the a set of nodes or links, suggesting an overall network property (e.g., density), while local measures are based on a small neighboring region of a particular node or edge (e.g., node degree). Some of these measures, such as density, result in a single (scalar) value per interval while other measures, such as node degree, result in a vector with the length of the number of nodes in the network. We leverage ten global measures to create time series in the center view of our interface and offer auxiliary visualizations to the distribution of node degrees over time.

For dynamic networks, we distinguish between *measures for static interval*, and *dynamic measures*, detailed in the following.

**Measures for static intervals** refer to measures that can be calculated per interval, independent from its preceding or succeeding interval.

- **Connected Nodes**: The total number nodes with at least one links during an interval.

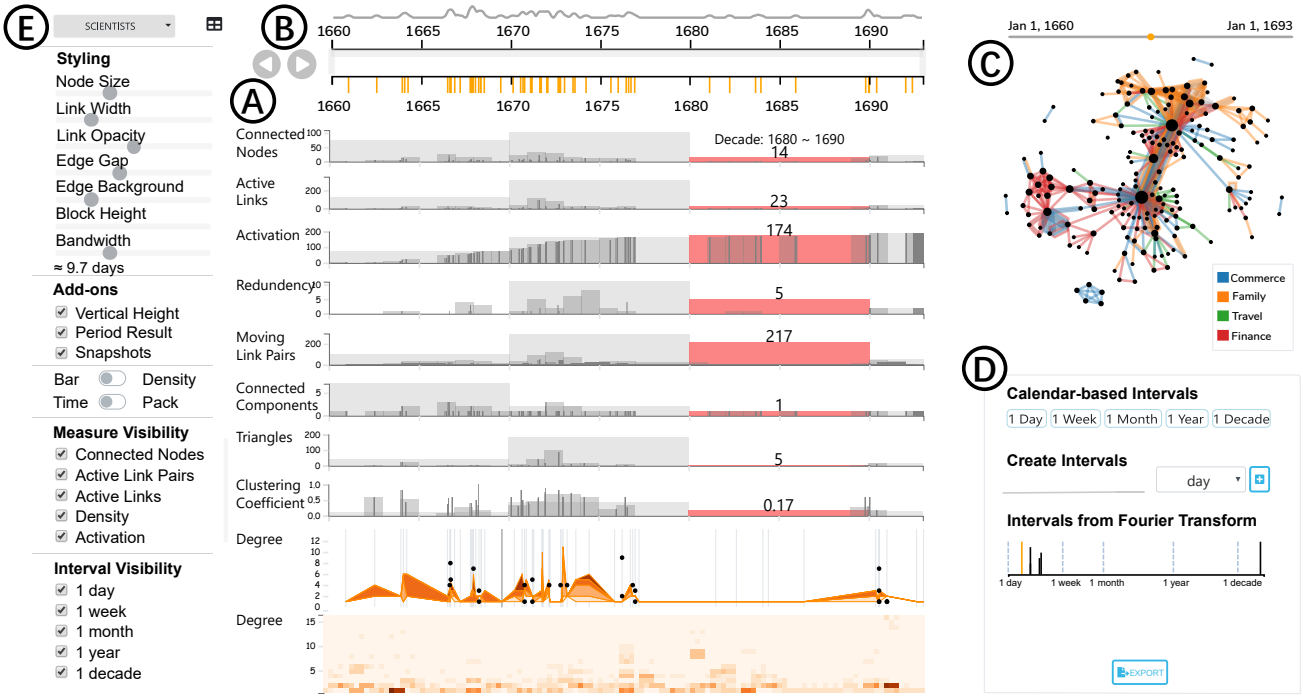


**Figure 2: The interface, MeasureFlow. (A) The measure view displaying an array of measures over time; (B) The time line view with events shown as yellow tick marks; (C) The node-link diagram; (D) The interval experiment panel, with a snippet of the top five periods from a Fourier Transformation. (E) Configuration panel with dataset selection and parameters for styling and adjusting the views. Note that not all measure blocks are displayed in the figure.**

- **Active Links & Node Pairs**: The number of links or connected node pairs during an interval. Links connecting the same pair of nodes are regarded as the same node pair.
- **Number of nodes/links with a specific type**: such statistics reveal the composition of individual interval.
- **Density**: The ratio of the link number and the square of the node number. We use the total number of nodes throughout the time range to enable intuitive comparison between periods.
- **Connected Component Number**: a group of nodes in a graph is considered as a connected component if for any node in this component, there exists a path between any other node in the component.
- **Motif Number**: The number of occurrences of a certain motif in the interval. This measure suggests the strength of a particular pattern of connections. Currently, we limit the motif to triangles.
- **Clustering Coefficient**: The ratio of closed triplet number and the open triplet number in the interval, which weights the tendency to be clustered.
- **Node degree**: The neighbor number of a specific node. In fact, this can be any alternative centrality measure calculated for an individual node.

**Dynamic measures** are calculated for each interval, taking its preceding interval into account.

- **Redundancy**: The number of nodes with at least one connection in a given interval and its predecessor interval—the first interval having redundancy of 0).
- **Activation**: The accumulated node number that has at least one connection from the very beginning till the current time point.
- **New Node Pairs & Leaving Node Pairs**: The number of node pairs connected, or disconnected in a given interval, with respect to its previous interval.
- **Changing Node Pairs** is similar to redundancy, but for connected node pairs.

## 5 INTERFACE DESIGN

Guided by the analytical tasks discussed in Section 3 as well as discussions with our analysts (A1-A4), we design an interactive system for exploratory data analysis of dynamic networks, addressing problems P1 and P2. The interface (Fig. 2) is structured into five major visualization components: (A) the *measure view* showing an array of graph measures over time, (B) the *timeline* being linked to the measure view, showing events (links) over time as small orange tick marks, and allowing for high-level temporal navigation and zoom, (C) the *node-link diagram* providing a topology view of a selected time period, (D) an *interval selection* menu to select different interval length for slicing the network, and (E) a general *menu* to set filter measures and adjust other visual settings. Interactions in MeasureFlow are designed to follow the "Overview first, zoom and filter, then details-on-demand" mantra [29]. Individual strategies are explained in Section 5.4.

- **Overview:** the measure view provides an overview of the network measures' evolution. Users can explore individual time periods through a time slider in the timeline (B), slice the dynamic

network by different units, shown in the interval panel (D) through the manual specification of Fourier Transformation.

- **Zoom & Filter:** once a period of interest is identified, users may brush on the timeline to zoom in for a closer look. They can also select interested subgroups in the graph by lassoing them in the graph view (Fig. 2) and see the measures in the measurement view for this subgraph only.
- **Details-on-Demand.** After finding a suitable interval, an animation can be played, or small multiples of the dynamic node-link (Fig.7) can be shown, supporting the comparison of periods and topological structures.

MeasureFlow is a module of Vistorian [6],[1] which provides a set of complementary visualizations for additional analysis, including an adjacency matrix, time arcs, and geographical node-link diagram. We demonstrate MeasureFlow through a social network of *Marie Boucher*, assembled from letter correspondence between 1660 and 1693, with a total of 190 nodes (persons), 489 individual links (letters=events) [16]. Letters include topics from commerce, to travel, to finance, and families, which we interpret as link type. For this network, our analyst A3 is interested in understanding which different period exists (T1.2, T1.4), the temporal distribution of link types (T2.3), and how specific nodes evolve (T3.2).

## 5.1 Visualizations

**Measure View**—The measure view (Fig. 2A), shows bar charts of global graph measures, computed for the standard set of calandar-based intervals (days, weeks, etc.). After discussion and prototyping, we decided to visualize these time series as bar charts, rather than line charts, as each bar represents a measure at a given interval, the width of the bar. To account for multiple overlapping time granules, each time series features multiple transparent layers of bars superimposed: narrower bars present finer granules, wider bars present values for coarser granules. To avoid the problem of visual occlusions, we make such an arrangement that the shorter the period, the darker the gray-scale of the bars, visually emphasizing smaller time intervals. The initial granularities are based on possible calendar-based time intervals. When hovering over a bar, the specific measure value shows up on top of the bar, together with other measures in the same period highlighted.

While bar charts represent scalar values over time, two additional visualizations (Fig. 3) show vectors for distribution of node degrees: (1) a connected box plot shows quartiles at different event point, connected by a curve; and (2) a heatmap, where the x-axis is showing time while the y-axis shows degree bins and the color for the number of nodes with a degree of the bin.

Observing measures over time in the measure view, we can see that the Merchant network roughly undergoes three phases. In the first five years (1660-1665), the network grows as more and more notes create first connections (activation). However, the connected components are not stable during the period, especially when seen from a short period. This suggests that the network is formed by several tiny groups cut off from each other. Then, the network remains relatively active from 1665 to 1678, with the fast-growing number of activated nodes. At last, the network gradually shrinks, with an almost flat activation and a rather small size. As confirmed

[1]https://vistorian.net

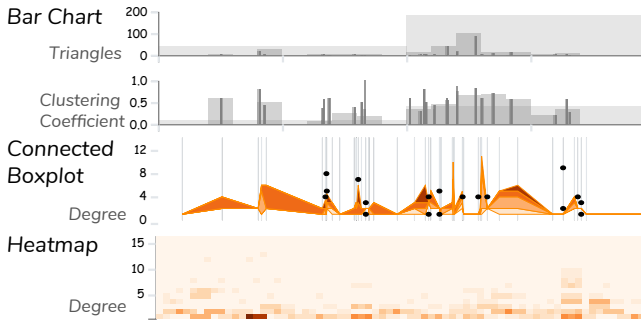


**Figure 3: Detail of measures, shown at different temporal granularities. (top two) Visualizing scalar values through narrow bars (small time granules) and wider bars (larger granules). Visualizing vectors over time using (center) connected boxplot, and (bottom) heatmap.**

by A3, the network indeed undergoes three major phases, related to contextual circumstances in the social network.

**Node-link Diagram**—Having obtained an overview, we can continue analysis with the aid of the node-link diagram and the small timeline just above that functions as time slider. The period from 1670 to 1675 seems extraordinarily active (i.e., many links), and therefore we brush the time slider to zoom in in order to have a closer look of this particular time range. Focusing on the monthly granularity, we can find a shift of communication types.

## 5.2 Visual Event Summaries with KDE Lines

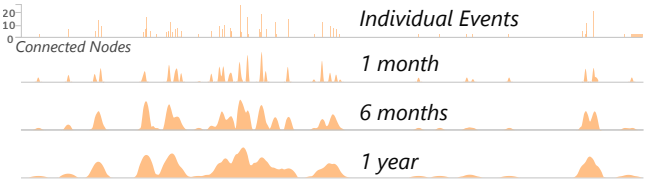


**Figure 4: An example of the kernel density estimate of the connected nodes in the *Merchant* dataset with bandwidths of 1 month, 6 months, and 1 year, from top to bottom.**

Bar charts had been chosen for visualizing measures over time as they better reflect measures for individual intervals. However, this leaves many bars very small and narrow. Instead, we were looking for a way to show temporal "density" to provide for a better impression of where and when events (links) happen. To that end, we introduce a kernel density estimate (KDE) into our system. It is closely related to histograms but can be endowed with properties such as smoothness or continuity leveraging a proper kernel. With a slight adjustment by treating the millisecond-based timestamp as the variable and the measure value as a weight, we adopt the KDE into our system, leveraging the measure values based on the lowest granularity (timestamp). The formula gives as below.

$$\hat{f}_h(t) = \frac{1}{nh\sum_{j=1}^{n} y_j} \sum_{i=1}^{n} y_i Ker(\frac{t-t_i}{h}),$$

where $\hat{f}_h(t)$ is the density estimate of a given time point $t$, $n$ the total number of timestamps within the bandwidth around $t$, $y_i$ the measure value of the timestamp $i$, $Ker$ the selected kernel, and $h$

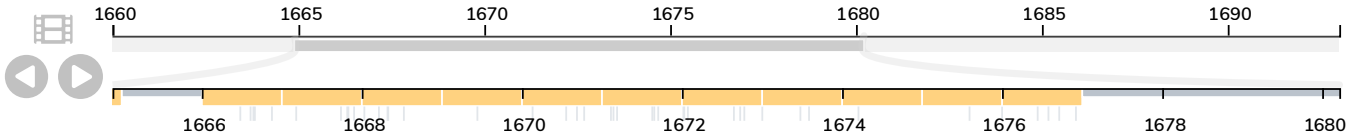


**Figure 5: The timeline under the fixed-interval mode. Orange bricks and gray stripes represent the interval with and without events, respectively, which helps to navigate among periods. Small multiples are shown when clicking on the film button.**

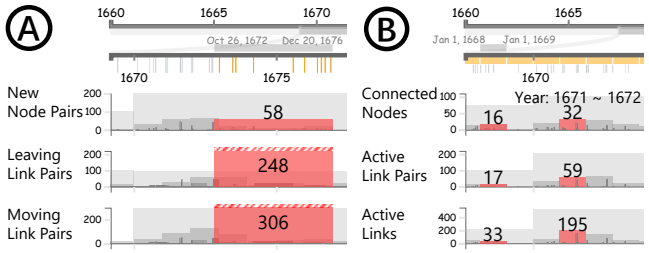


**Figure 6: (A) Manually creating a custom-sized intervals with the time slider (lower timeline) shows measure values; strips on top of bars indicate that the value is higher (e.g., *248*) than it can be shown on the Y-axis (*200*). (B) comparing measures between two intervals by highlighting both intervals on the timeline.**

the parameter concerning bandwidth. The selected kernel is the Epanechinikov kernel, for it is optimal in terms of the square error.

$$Ker(z) = \begin{cases} \frac{3(1-z^2)}{4h} & \text{if } |z|<h \\ 0 & \text{otherwise} \end{cases}$$

Intuitively, any time period without an event will cause the line to drop, while periods with frequent events or large values will bring up the line, which accords with the basic understanding of data. As shown in Fig. 4, the KDE result on a measure brings a more straightforward impression about the evolution.

## 5.3 Exploration Strategies

Finding crucial time points is one of our goals (Section 3). With the following six strategies and respective interface components, we aim to support exploration in MeasureFlow.

**S1: Calendar-based Intervals:** 5 shows the main timeline in MeasureFlow, which allows for fast and easy browsing of time intervals. As different temporal slicing methods provide different levels of detail, a user first specifies a particular calendar-based interval size (e.g., days, months, years) in the interval panel (Fig. 2D); then these intervals are shown as orange 'bricks' on the timeline (Fig. 5). Clicking a brick highlights the measure values for that period in the measure view below (Fig. 2A) and the respective topology in the node-link view (Fig. 2C). Gray bricks show periods without any events (links); to avoid many empty intervals (gray bricks) cluttering the timeline and making navigation tedious, we combine all adjacent empty intervals into a single continuous empty interval, represented by long gray bricks. Three buttons to the left of the timeline allow to move though time intervals step-by-step and to play an animation as well as showing small multiples (Fig. 7).

**S2: Manual Intervals:** From the initial calendar-based overview of the graph measures, analysts can form a rough idea of the development of the dynamic graph. By brushing on the bottom

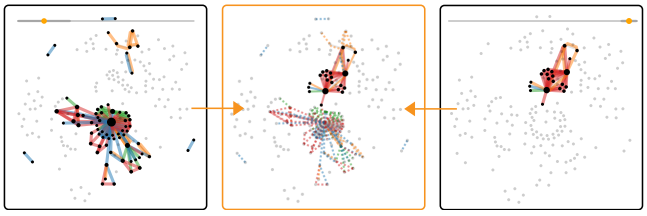

**Figure 7: Comparisons of the topological structure (middle) between snapshots the fist decade (left) and the last decade (right, selected as the basis) in the *Merchant* dataset.**

timeline and zooming through the top timeline appropriately, it is possible to show measures for this manual time interval, overlaid over the existing bar charts in the measure view (Fig. 6A). As such manual intervals can result in significantly higher values than the Y-axis in the bar charts show—the Y-axis shows the max value from standard calendar-based time granules—higher values are indicated by stripes on top of the bar.

**S3: Manual Interval offsets.** After selecting a calendar-based interval, the user can apply an offset of these intervals to make, e.g., a week starting at any day of the week.

**S4: Intervals based on Fourier-Transform:** Each graph measure calculated over time can be regarded as a temporal signal of the network. We apply signal analysis to obtain its frequency distribution and derive prominent periodicity in terms of the trigonometric function. To present the result in an understandable way, we run Fast Fourier Transform on the first measure (connected nodes) and then select the top five frequencies with the highest amplitude. Frequencies are visualized by narrow lines showing in a chart (Fig. 2D) with the frequency on the X-axis and the frequencies' rank on the Y-axis (top=high rank, bottom=low rank). As different measure signals lead to different decomposition, the default measure is set to the link number. We adopt the Fast Fourier Transformation on the measure signal [28]:

$$M_k = \sum_{n=0}^{N-1} m_n e^{-i2\pi kn/N},$$

where $M_k$ is the $k$th component in the frequency domain, and $m_n$ is the value of the signal at the $n$th slices (Fig. 1).

**S5: Period Comparison:** When a user selects one time period in the timeline (Fig. 5) he/she can hover any other time period to show measures for both periods in the measure view below the timeline (Fig. 6B). In the node-link view (Fig. 2C), links of the first period will be highlighted with solid lines, while the links of the compared period will be encoded by dashed lines (Fig. 7).

**S6: Subgraph Comparison:** Users can specify a subgroup of nodes in the node-link view by lasso selection. Measures will consequently be shown for the selected subgroups, differentiated by color. Hovering any bars in the measure view reveals the value.

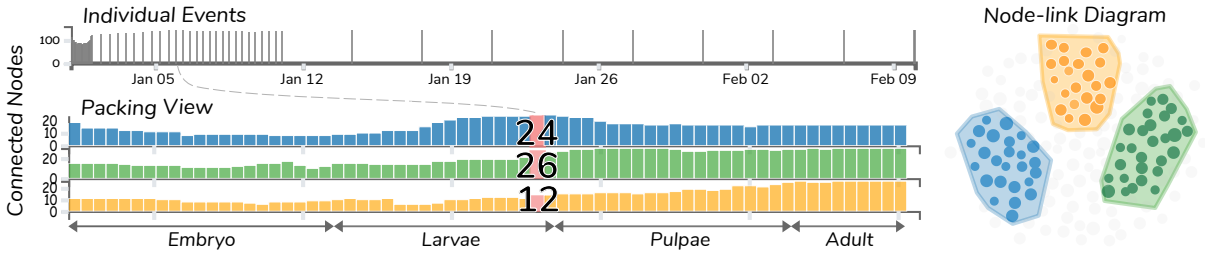


**Figure 8: A packing view for the hourly events for connected nodes in three subgraphs. The upper diagram is the original bar chart without packing.**

## 6 CASE STUDIES

In this section, we use two more real-world examples to demonstrate usefulness and effectiveness of MeasureFlowin support of the three task dimensions from Section 3, i.e., analysis over time (T1), measures (T2), and subgraphs (T3).

### 6.1 Gene Dataset

The *Gene* dataset [18] is about gene expression of drosophila melanogaste, covering the 40 days of development phases. Other than the other examples in this paper, this network comes as a snapshot at different time intervals: including embryos (hourly snapshops for 1 day), larvae (daily snapshots for 4 days), pulpae (daily snapshots for 5 days), and adults (one snapshots every 5 days for 30 days). Each node corresponds to a gene, and edges represent inferred conditional dependencies between the genes. There are a total of 150 nodes, 1,750 node pairs, and 27,899 links, which contribute to a dense network. We are interested in its development situation comparing three clusters (blue, green, yellow, Fig. 8).

**Overview**—From an overview over the daily measures we see that the network is relatively stable (T1.1). Fig. 8 shows the number of connected nodes for each of the non-equidistant snapshots, across three clusters (blue, green, yellow). In this measure view, we removed all empty hours and days and make time intervals the same width for better comparison. From the hourly granularity of the *Embryo* period (first few values), we can see that the embryo stage undergoes some variations (T1.2); The number of connected nodes decreases first and then increases (T2.1, T2.2). Generally, connectivity grows in all clusters towards the end of the process.

**Period Inspection**—In particular, we zoom into the embryo phase for it seems highly unstable. Looking into the starting point, midpoint, and the end point, massive connections among nodes disappear at first. The density of the network drops abruptly from 0.0438 to 0.0196 (T1.4, T2.2). While comparing the midpoint and the end point, we see that though density continues to decrease, a number of connections among nodes come into beings later at the end of the embryo stage (T2.1). Besides, examining the degree view, we discover that genes tend to reduce reliance to others with the past of time (T2.3), and the outliers at different time slots are actually the same group of genes.

**Cluster Comparison**—Generally, all three clusters behave similar over the entire process with rather unstable connectivity in the *Embryo* and early *Larvae* phase. Eventually, the blue cluster reaches peak connectivity around January 7, while the green cluster reaches peak connectivity a few days later and mainly stays on that level. The yellow cluster sees its peak a at the very end of the process, preceded by a monotonous increase.

### 6.2 Highschool Dataset

The *Highschool* dataset[2] contains four days of minute-wise face-to-face contacts in a high school in France in 2011. Nodes include 8 teachers and 118 students from three classes (with 31, 45, 42 nodes respectively). It is a dense network with 1,710 connected node pairs, 28,561 links, and 1940 time points. We analyse the general pattern of contacts and compare classes.

**Analyzing Time**—Since events are based on minutes, we create time intervals of 15 minutes (Fig. 2D), to better understand the general pattern without loss of detail. From the measure view (Fig. 9), we find that daily communication shows regularity in daily outbursts for most measures at around 8am, 12 pm, and again in the afternoon 4pm with strong fluctuations (T1.3). It seems these bursts refer to breaks. Days 1 and 3 also exhibit a burst later in the day, suggesting an evening activity.

**Comparing Measures**—Comparing measures, we find that some measures are highly correlated, e.g,. density, connected nodes, and moving links (T2.3). Over time, we see that density is highest on the first day in our data set but lower on average on the other days—while the number of connected nodes remains around the same. Redundancy tells us that during a day, connections remain stable, e.g., that the same nodes that connect early in the day, also connect later in the day, suggesting mutual friendships and structured interactions. Cluster coefficient is very much higher and regular on day 1, compared with the other days. Generally, from the measures in Fig. 9, we can see day 1 showing slightly more activity and connections than the other other days and that day 2 seems to be slightly shorter, perhaps not involving any evening activity.

## 7 DISCUSSION

**Summary**—MeasureFlow integrates a wide set of features to explore dynamic networks based on analytical measures and to guide a user to interesting time points. In particular, we focus on networks with irregular occurrences of links (events) over time (Section 1: C1, C2). Besides the visualizations for measures and topology, MeasureFlow includes features such as KDE for smoothing measures, removing empty periods (packing), calculating interval length based on Fourier transformation, selecting and comparing subgraphs, explore differences in topology across intervals and a

[2]http://www.sociopatterns.org/datasets/high-school-dynamic-contact-networks

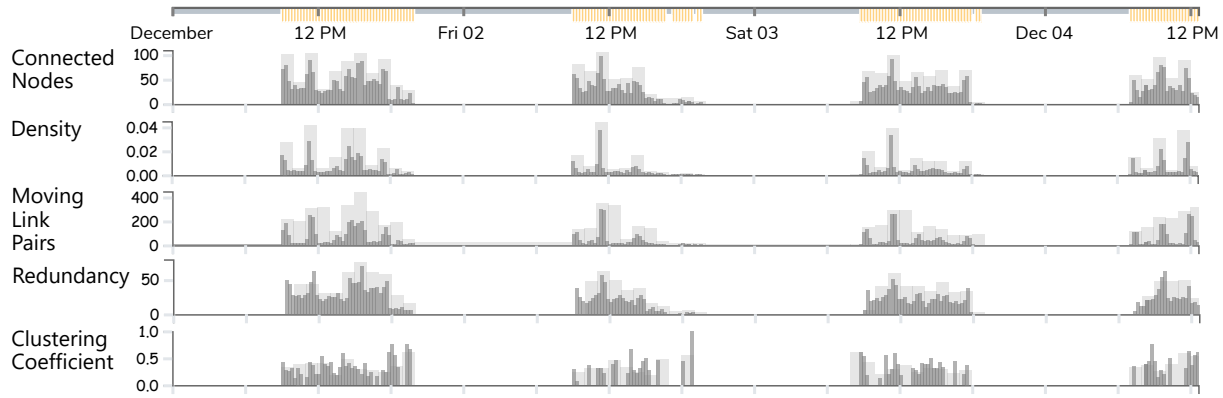


**Figure 9: Multiple measure series for the *Highschool* dataset with periods of 15 minutes and 1 hour, showing a recurrent pattern of outburst at around 12pm.**

novel way to show measures at different time granules simultaneously.

MeasureFlow is complementary to other visualizations that visualize network topology or otherwise require a well-defined notion of time window-size and time intervals. To that end, MeasureFlow is designed with a set of tasks in mind (Section 3), spanning *measures*, *time*, and *topology* and related higher-level tasks such as finding time points of interest, e.g., by analyzing trends, anomalies, or correlations between measures across time, or finding appropriate sizes for time intervals for future analysis, e.g., by applying FFT (Section 5.3) or manually exploring time interval sizes (Section 5.4).

**Visualizations**—The visualizations in MeasureFlow aim to expose measures' evolution over time for both scalar values (e.g., network density) and vectors (e.g., node degrees). To solve the problem of visualizing measures at different temporal granularities (P1), we used superimposed bar charts, one for every granularity. This allows values to be displayed for each granularity at the same time and allow analysis. Kernel Density Estimation allows to show time periods with many events (Fig. 4).

To visualize vector values over time (P2), we present two visualizations (heatmap, and connected box plots, Fig. 2). Alternative visualizations could include multiple superimposed line charts or potentially other visualizations aggregating the vectors. There are open questions here with respect to finding appropriate visualizations both vectors (P2) and improving the visualization of scalar values (P1). One possibility to address P1 might be through applying KDE to measures other than events and thus to provide a better idea of density of events and value of the measure.

**Measures**—We selected some common measures to include in MeasureFlow, but any extension is straightforward. The Menu (Fig. 2E) allows to hide and filter the measures displayed, and thus their list can be extended as desired. Three of the measures have been informed by discussions with the analysts A1-A4 as these measures were important for their work. MeasureFlow can inform future work on assessing which measures allow for which insights and which measures do best support specific tasks. Besides the specific domain and task of the analyst, we believe the choice of 'a good set of measures' will depend on the character of the networks. For example, our measures *redundancy* or number of *triangles* both support very different tasks. Moreover, we hope that a visual approach such as MeasureFlow will inspire the creation and application of additional measures to analyze dynamic networks over time. For example, it could be possible to extend MeasureFlow with a console that allows analysts to create their own measures and derivatives thereof. This could, eventually, include the definition and visualization of 'meta-measures' that quantify, e.g., the rate of change (similar to existing approaches [5, 8]). Another possibility is the visualization of individual node measures (e.g,. node degree, number of new neighbors) over time, posing challenges to visualize that amount of information in an understandable way.

**Defining time-intervals:** As many measures about dynamic networks, rely on the proper definition of time-intervals, MeasureFlow supports a range of techniques to explore and select time intervals. In the future, we aim to investigate techniques to obtain non-uniform time-intervals, i.e., where time intervals change in length to capture identified stages of the graph and to account for periods of changing event density. This can be a powerful approach to aggregate networks with many changing characteristics over time, compared to networks with rather uniform occurrences of links and changes.

**Usability**—Eventually, we need stronger evidence on how MeasureFlow does support analysts in their workflows. We continue working with our analysts and investigate different data sets in the future to better understand which measures are of importance and how MeasureFlow supports exploration.

## 8 CONCLUSION

This paper presented MeasureFlow, an interactive approach to visualize measures for dynamic networks over time to aid visual exploration of trends, periods, and stages and to guide analysts to interesting network measures. MeasureFlow shows how a measure-based approach can complement the visual exploration of dynamic networks and list a set of open questions for future work. We invite the community to extend MeasureFlow and integrate more high-level guidance into their network visualization tools.

# REFERENCES


[1] Jae-wook Ahn, Catherine Plaisant, and Ben Shneiderman. 2013. A task taxonomy for network evolution analysis. *IEEE Trans. Vis. Comput. Graph.* 20, 3 (2013), 365–376. https://doi.org/10.1109/TVCG.2013.238

[2] Jae-Wook Ahn, Meirav Taieb-Maimon, Awalin Sopan, Catherine Plaisant, and Ben Shneiderman. 2011. Temporal Visualization of Social Network Dynamics: Prototypes for Nation of Neighbors. In *Proceedings of the 4th International Conference on Social Computing, Behavioral-Cultural Modeling and Prediction (SBP'11)*. Springer, Berlin, Heidelberg, Germany, 309–316. https://doi.org/10.1007/978-3-642-19656-0_43

[3] Michelle N. Arbeitman, Elleen E. M. Furlong, Farhad Imam, Eric Johnson, Brian H Null, Bruce S. Baker, Mark A. Krasnow, Matthew P. Scott, Ronald W. Davis, and Kevin P. White. 2002. Gene Expression During the Life Cycle of Drosophila melanogaster. *Science* 297, 5590 (2002), 2270–2275. https://doi.org/10.1126/science.1072152

[4] Benjamin Bach. 2016. Unfolding Dynamic Networks for Visual Exploration. *IEEE Comput. Graph. Appl.* 36, 2 (2016), 74–82. https://doi.org/10.1109/MCG.2016.32

[5] Benjamin Bach, Nathalie Henry-Riche, Tim Dwyer, Tara Madhyastha, Jean-Daniel Fekete, and Thomas Grabowski. 2015. Small MultiPiles: Piling time to explore temporal patterns in dynamic networks. *Comput. Graph. Forum* 34, 3 (2015), 31–40. https://doi.org/10.1111/cgf.12615

[6] Benjamin Bach, Nathalie Henry Riche, Roland Fernandez, Emmanoulis Giannisakis, Bongshin Lee, and Jean-Daniel Fekete. 2015. NetworkCube: Bringing Dynamic Network Visualizations to Domain Scientists. Posters of the IEEE Visualization Conference (VIS'15). Poster.

[7] Benjamin Bach, Emmanuel Pietriga, and Jean-Daniel Fekete. 2013. GraphDiaries: Animated Transitions and Temporal Navigation for Dynamic Networks. *IEEE Trans. Vis. Comput. Graph.* 20, 5 (2013), 740–754. https://doi.org/10.1109/TVCG.2013.254

[8] Benjamin Bach, Conglei Shi, Nicolas Heulot, Tara Madhyastha, Tom Grabowski, and Pierre Dragicevic. 2015. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Trans. Vis. Comput. Graph.* 22, 1 (2015), 559–568. https://doi.org/10.1109/TVCG.2015.2467851

[9] James P Bagrow, Erik M Bollt, Joseph D Skufca, and Daniel Ben-Avraham. 2008. Portraits of complex networks. *Europhysics Letters (EPL)* 81, 6 (2008), 68004. https://doi.org/10.1209/0295-5075/81/68004

[10] Mathieu Bastian, Sebastien Heymann, and Mathieu Jacomy. 2009. Gephi: An Open Source Software for Exploring and Manipulating Networks. In *Proceedings of the Third International Conference on Weblogs and Social Media (ICWSM'09)*. AAAI Press, Menlo Park, CA, USA, 2.

[11] Fabian Beck, Michael Burch, Stephan Diehl, and Daniel Weiskopf. 2017. A taxonomy and survey of dynamic graph visualization. *Comput. Graph. Forum* 36, 1 (2017), 133–159. https://doi.org/10.1111/cgf.12791

[12] Ilya Boyandin, Enrico Bertini, and Denis Lalanne. 2012. A Qualitative Study on the Exploration of Temporal Changes in Flow Maps with Animation and Small-Multiples. *Comput. Graph. Forum* 31, 3 (2012), 1005–1014. https://doi.org/10.1111/j.1467-8659.2012.03093.x

[13] Michael Burch, Corinna Vehlow, Fabian Beck, Stephan Diehl, and Daniel Weiskopf. 2011. Parallel Edge Splatting for Scalable Dynamic Graph Visualization. *IEEE Trans. Vis. Comput. Graph.* 17, 12 (2011), 2344–2353. https://doi.org/10.1109/TVCG.2011.226

[14] Weiwei Cui, Xiting Wang, Shixia Liu, Nathalie H Riche, Tara M Madhyastha, Kwan Liu Ma, and Baining Guo. 2014. Let it flow: a static method for exploring dynamic graphs. In *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis'14)*. IEEE, Piscataway, NJ, USA, 121–128. https://doi.org/10.1109/PacificVis.2014.48

[15] Pravallika Devineni, Evangelos E. Papalexakis, Danai Koutra, A. Seza Dogruöz, and Michalis Faloutsos. 2017. One Size Does Not Fit All: Profiling Personalized Time-Evolving User Behaviors. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM'17)*. ACM, New York, NY, USA, 331–340. https://doi.org/10.1145/3110025.3110050

[16] Nicole Dufournaud, Bernard Michon, Benjamin Bach, and Pascal Cristofoli. 2017. L'analyse des réseaux, une aide à penser: réflexions sur les stratégies économique et sociale de Marie Boucher, marchande à Nantes au XVII e siècle. In *Réseaux de femmes, femmes en réseaux (XVIe-XXIe siècles)*. Presses Universitaires de Bordeaux, Bordeaux, France, 109–137.

[17] Manuel Freire, Catherine Plaisant, Ben Shneiderman, and Jennifer Golbeck. 2010. ManyNets: an interface for multiple network analysis and visualization. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'10)*. ACM, New York, NY, USA, 213–222. https://doi.org/10.1145/1753326.1753358

[18] Alexander J. Gibberd and James D. B. Nelson. 2017. Regularized Estimation of Piecewise Constant Gaussian Graphical Models: The Group-Fused Graphical Lasso. *Journal of Computational and Graphical Statistics* 26, 3 (2017), 623–634. https://doi.org/10.1080/10618600.2017.1302340

[19] Mustafa Hajij, Bei Wang, Carlos Scheidegger, and Paul Rosen. 2018. Visual Detection of Structural Changes in Time-Varying Graphs Using Persistent Homology. In *Proceedings of the IEEE Pacific Visualization Symposium (PacificVis'18)*. IEEE, Piscataway, NJ, USA, 125–134. https://doi.org/10.1109/PacificVis.2018.00024

[20] Sanjay Kairam, Diana MacLean, Manolis Savva, and Jeffrey Heer. 2012. GraphPrism: Compact Visualization of Network Structure. In *Proceedings of the International Working Conference on Advanced Visual Interfaces (AVI'12)*. ACM, New York, NY, USA, 498–505. https://doi.org/10.1145/2254556.2254651

[21] Alexandra Lee, Daniel Archambault, and Miguel Nacenta. 2019. Dynamic Network Plaid. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'19)*. ACM, New York, NY, USA, 14. https://doi.org/10.1145/3290605.3300360

[22] Bongshin Lee, Catherine Plaisant, Cynthia Sims Parr, Jean-Daniel Fekete, and Nathalie Henry. 2006. Task Taxonomy for Graph Visualization. In *Proceedings of the 2006 AVI Workshop on BEyond Time and Errors: Novel Evaluation Methods for Information Visualization (BELIV '06)*. ACM, New York, NY, USA, 1–5. https://doi.org/10.1145/1168149.1168168

[23] James Moody, Daniel McFarland, and Skye Bender-deMoll. 2005. Dynamic network visualization. *American journal of sociology* 110, 4 (2005), 1206–1241. https://doi.org/10.1086/421509

[24] Tamara Munzner. 2015. *Visualization analysis and design*. CRC Press, Boca Raton, FL, USA.

[25] Adam Perer and Ben Shneiderman. 2008. Integrating Statistics and Visualization: Case Studies of Gaining Clarity during Exploratory Data Analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 265–274. https://doi.org/10.1145/1357054.1357101

[26] Adam Perer and Jimeng Sun. 2012. Matrixflow: temporal network visual analytics to track symptom evolution during disease progression. In *AMIA annual symposium proceedings*, Vol. 2012. American Medical Informatics Association, 716.

[27] Ryan A. Rossi and Nesreen K. Ahmed. 2015. The Network Data Repository with Interactive Graph Analytics and Visualization. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, Menlo Park, CA, USA, 4292–4293.

[28] Ali Saidi. 1994. Decimation-in-time-frequency FFT algorithm. In *Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP '94),*. IEEE, Piscataway, NJ, USA, 453–456. https://doi.org/10.1109/ICASSP.1994.389992

[29] Ben Shneiderman. 1996. The Eyes Have It: A Task by Data Type Taxonomy for Information Visualizations. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*. IEEE, Piscataway, NJ, USA, 336–343. https://doi.org/10.1109/VL.1996.545307

[30] Stef van den Elzen, Danny Holten, Jorik Blaas, and Jarke J van Wijk. 2013. Dynamic network visualization with extended massive sequence views. *IEEE Trans. Vis. Comput. Graph.* 20, 8 (2013), 1087–1099. https://doi.org/10.1109/TVCG.2013.263

[31] Stef van den Elzen, Danny Holten, Jorik Blaas, and Jarke J. van Wijk. 2016. Reducing Snapshots to Points: A Visual Analytics Approach to Dynamic Network Exploration. *IEEE Trans. Vis. Comput. Graph.* 22, 1 (2016), 1–10. https://doi.org/10.1109/TVCG.2015.2468078

[32] Frank van Ham, Hans-Jörg Schulz, and Joan Morris DiMicco. 2009. Honeycomb: Visual Analysis of Large Scale Social Networks. In *Proceedings of IFIP Conference on Human-Computer Interaction*. Springer, Berlin, Heidelberg, Germany, 429–442. https://doi.org/10.1007/978-3-642-03658-3_47

[33] Yong Wang, Daniel Archambault, Hammad Haleem, Torsten Moeller, Yanhong Wu, and Huamin Qu. 2019. Nonuniform Timeslicing of Dynamic Graphs Based on Visual Complexity. In *Proceedings of IEEE Visualization Conference (VIS'19)*. IEEE, Piscataway, NJ, USA. https://doi.org/10.1109/VISUAL.2019.8933748 Short paper.