Too Much Time, Not Enough Space: Multi-Scale Event Sequence Visual Analytics with m-SVEN

Dustin Arendt, Meg Pirrung

Abstract—Many techniques exist to visually represent temporal event sequences. However, few techniques are purposely designed for direct visual comparison of many (e.g., tens or hundreds) event sequences. This task is important for understanding recurring patterns, trends, and detecting anomalies, but visual comparison across event sequences is challenging when sequences are not trivially short. We propose a general framework for visualization of event sequence collections with storylines designed specifically for this comparison task. Storylines, just like parallel coordinate plots, suffer from usability issues when the number of time windows is large, so our framework includes a multi-scale technique to transform high temporal resolution event sequence data into a more appropriate format for storylines. We also built a user interface that supports zooming on the storyline visualization to leverage the multi-scale representation of the temporal event sequences. We evaluated our tool with a case study on the IEEE VAST Challenge 2014 data which included eliciting feedback from an expert user on our interface.

Index Terms—Storyline visualization, Temporal event sequence, Multi-scale modeling, Layout algorithm, Geo-temporal analysis.

1 Introduction

Extracting insight about when, how, and why a system is changing over time is a primary purpose of visual analytics. But what exactly is change and how should it be communicated to the user? Many temporal visualizations, including visualizations for temporal event sequences, rely on time series plots that directly encode the state of the entities over time. However, storylines show how entities change relative to each other [11, 15, 16, 18, 21, 22, 24]. Understanding relative change in temporal event sequences can be helpful for tasks including identifying cohorts that behave similarly over time, detecting branching or merging of groups of entities, or spotting patterns of recurrence. We believe storyline visualization can be an effective technique for making sense of temporal event sequences.

Storyline visualizations represent entities as lines, with time being encoded on the horizontal axis; the vertical axis does not directly encode any variable. Instead, the vertical axis is used to convey similarity between entities. Storylines encode interaction (e.g., have the same state at the same time) by drawing entities' lines close together during interaction, and farther apart otherwise–storylines convey time-dependent relationships across temporal event sequences.

While storyline layout algorithms are growing increasingly sophisticated, a the challenges that has not been adequately addressed is how to effectively draw storylines having many time windows (i.e., high time resolution). This paper presents two main contributions: 1) a general approach for transforming high temporal resolution event sequences into time-windowed data suitable for storyline visualization; and 2) a novel multi-stage, multi-objective layout algorithm to determine the y-coordinate of the storylines.

Our approach, m-SVEN, is designed to address the scalability issues arising when event sequences exhibit high temporal resolution. We propose the following framework for visualizing collections of high time resolution temporal event sequences with storylines.

1. *Find events:* if starting with real-valued feature vectors, quantize the time series into a temporal event sequence.
2. *Find windows:* resample temporal event sequences into time windows.
3. *Find Behaviors:* count the events of each type occurring within each time window.
4. *Find Interactions:* group together storylines within time windows that have similar behaviors.

The remainder of the paper is structured as follows. After discussing work related to temporal event sequences and storyline visualization, we proceed directly to outlining our storyline layout algorithm. Following this, we demonstrate the framework with a case study on the VAST Challenge 2014. In this case study we discuss how we adapted these data to our framework, we discuss how we designed the user interface to support the case study, and we present a brief evaluation of our system attained by soliciting expert feedback.

2 Related Work

Some existing visualization techniques are appropriate for understanding collections of event sequences. Scarf plots [4, 19] and other related visualizations represent event state with colors, and event sequences as horizontal rectangular sequence of colors. Multiple scarf plots can be stacked vertically to support comparison. However, this is difficult when there are many sequences, or when two sequences are far apart in the visualization. Dimension reduction techniques such as principal components analysis, multidimensional scaling, or manifold learning [23] can be helpful for representing event sequences when events are modeled by a n-dimensional feature vectors. For example, Bach et. al. used a two dimensional projection to represent temporal evolution for time series one at a time [3]. If projected into a 1-dimensional space, the time-windowed behavior of the event sequence can then be plotted against time to create a storyline-like visualization [6, 7, 17]. However, visualizations crafted from projected data can be misleading because points that are nearby in the projection are not necessarily nearby in the original space [5].

Storylines provide a more abstract but more explicit representation of relationships over time. This requires partitioning the entities within each time window into groups referred to as "interaction sessions" [21, 22]. From this, the visualization is arranged such that interacting entities form tight groups, and non-interacting entities are placed further apart. Meaningful distance between lines is explicitly enforced by the layout algorithm for the entire visualization, rather than occurring on average, as it does with projected data. This requires a layout algorithm to optimized assumed aesthetic criteria such as crossings, wiggles, and whitespace [21].

Layouts for storyline visualizations of genealogical data were produce by restricting the storyline data to tree-like structures [15]. At the

- *Dustin Arendt is with Pacific Northwest National Laboratory. E-mail: dustin.arendt@pnnl.gov.*
- *Meg Pirrung is with Pacific Northwest National Laboratory. E-mail: meg.pirrung@pnnl.gov.*

Proceedings of the IEEE VIS 2016 Workshop on Temporal & Sequential Event Analysis. Available online at: http://eventevent.github.io

same time, a general purpose algorithm for visualizing software evolution was developed [18]. However, this algorithm relied on a genetic program to search for candidate layouts, and as a result converged very slowly. Following this, the most recent general purpose storyline layout algorithms were developed [16, 21, 22]. These algorithms converge more quickly but are very complex and rely heavily on constrained quadratic programming. An additional contribution from [16] is the ability to specify hierarchal relationships between entities and have these enforced in the layout. A fast and incremental storyline layout algorithm was recently developed for streaming data [21]. Much other work exists leveraging storylines without contributing new layout algorithms–we consider this outside the scope of this review.

3 m-SVEN Storyline Layout Algorithm

We present a layout algorithm that explicitly optimizes the positions of the storylines guided by a set of aesthetic criteria. This algorithm is a refinement of our previous storyline research named SVEN [1, 2]. For storyline visualization, established aesthetic criteria are to reduce crossings, wiggles, and whitespace [21], and algorithms for optimizing one or more these criteria have been the subject recent research [11, 15, 16, 18, 21, 22, 24]. Because existing algorithms for storyline visualization were ad hoc, too slow, did not optimize all established aesthetic criteria, or required proprietary convex solvers, we are in the process of refining our own storyline layout algorithm.

Our purpose in designing a new layout algorithm was to enable rendering a storyline visualization of reasonable aesthetic quality at interactive speeds to support exploratory visual analytics. Our layout algorithm is a multi-stage multi-objective optimization algorithm; in each stage a single aesthetic objective is optimized, and the result from the current stage is respected by the subsequent stages. Wherever possible, we have attempted to match the subproblems in each stage with algorithms already existing in the literature. Our algorithm, summarized in Fig. 1 finds the storyline layout by optimizing the aesthetic objectives in the following order:

1. *Crossings:* the number of pairs of storylines whose relative order changes between adjacent time windows,

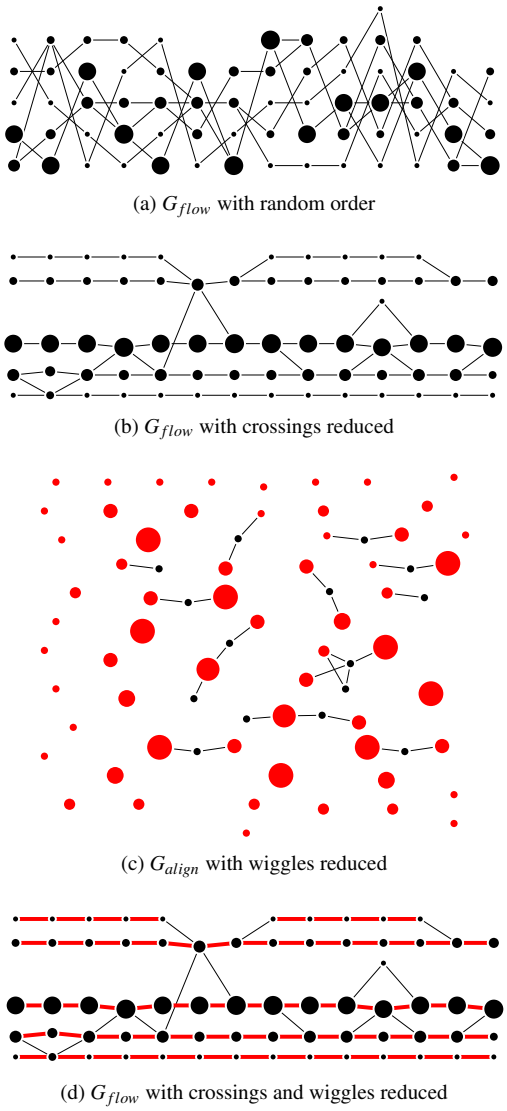
2. *Wiggles:* the number of times a storylines' heights are different between adjacent time windows, and

3. *Whitespace:* the vertical distance between adjacent storylines.

A directed acyclic graph G_{flow} is constructed (see Fig. 1a) to model the defined interaction sessions. Each vertex uniquely represents the entities within the same interaction session. Edges between group vertices model flow from one interaction session to another. The weight of this edge is the number of storylines that flow between the sessions. Each vertex is assigned to a unique layer corresponding to its time window, and the order of the vertices within each layer determines the number of crossings in the layout.

There are several techniques available that minimize the crossings between vertices within the same layer in a directed acyclic graph [13]. We first sweep back and forth along the levels in G_{flow} to reduce crossings by sorting vertices within layers using the "median heuristic" [8]. Following this, we find the pair of adjacent vertices within the same time window that would decrease edge crossings the most when swapped. This is repeated until no swaps exist that decrease the number of crossings, which is illustrated in Fig. 1b. After this, the order of storylines within groups is found in a similar manner.

After the order of the interaction sessions and storylines is determined, we reduce unnecessary line wiggles by finding a maximal set of edges in G_{flow} to align. An edge (u, v) in G_{flow} contains one or more storylines moving from group u to group v in adjacent time windows. Given that the order of the storylines was determined in the previous stage of the algorithm, we can directly determine the ideal offset between u and v, ignoring all other groups. This is the offset that has the fewest wiggles, or, equivalently, straightens the most storylines between u and v. We construct the alignment graph G_{align}



(a) G_{flow} with random order

(b) G_{flow} with crossings reduced

(c) G_{align} with wiggles reduced

(d) G_{flow} with crossings and wiggles reduced

Fig. 1: **Crossing and wiggle reduction:** Vertex size corresponds to the number of storylines within that interaction session. Between (a) and (b) the order of vertices in G_{flow} is changed to reduce crossings. Wiggles are reduced by selecting a maximal amount of edges in G_{flow} to align. Red vertices in G_{align} (c) are equivalent to red edges in G_{flow} (d) that have been aligned.

where each vertices in the G_{align} was an edge in G_{flow}. Each vertex (u, v) in G_{align} is weighted based on the number of storylines that would be straightened if groups u and v were aligned.

We add edges to G_{align} to represent conflicts, illustrated in Fig. 1c. When an edge is aligned, its endpoints are placed at the same height, but we must ensure that aligning an edge does not change the order of the vertices found previously. Only one in- and out-edge can be aligned per group vertex. If two edges in G_{flow} cross, only one of those edges can be aligned. Finding the maximum weighted independent set of G_{align} is equivalent to finding the largest number of edges in G_{flow} to align (see Fig. 1d). Though exact solutions are not found in polynomial time in the general case, we use a simple greedy algorithm that produces high quality approximations [20].

The final stage in our layout algorithm is to solve for the y-coordinates of the groups in a manner that respects the order and alignment found in the previous two stages. We construct another directed acyclic graph $G_{whitespace}$ whose vertices are the connected components of the subgraph induced from only the aligned edges found previously

and whose edges enforce the ordering found previously. Layering $G_{whitespace}$ will solve for the y-coordinate of the groups, and subsequently the storylines; there are many algorithms to compute layerings for directed acyclic graphs [13]. We solve this layering using the Network Simplex algorithm [12]. Due to the special structure of the problem, we found this algorithm to be much faster than off-the-shelf numerical solvers. We had also experienced unpredictable issues with linear solvers unable to find feasible starting points–this was not an issue with the Network Simplex algorithm.

4 Case Study: VAST Challenge 2014

We explore the IEEE VAST Challenge 2014 [14] in this case study. This is a fictional scenario where several executives from a petroleum company, "GasTech," go missing under suspicious circumstances. Several datasets are provided that could help an analyst determine why the employees disappeared, where they might have gone, and who might be responsible. Using our proposed framework, the basic patterns of life and some interesting anomalies are apparent at different timescales in the resulting visualization.

4.1 Application of Approach to VAST Challenge Data

Our exploration found that the GPS data in the IEEE VAST Challenge 2014 exhibited vehicle patterns typical of everyday life, for the most part. Vehicles follow roads, moving quickly between locations, and when a destination is reached the vehicle is switched off and the GPS data stops recording. We fill in these missing data by first downsampling the GPS data to 10 minute intervals and filling forward to impute the missing data lost by the vehicle being switched off. The data is down sampled by taking the last measurement in each time window.

From this modified dataset, we were able to find regions where entities spent most of their time (instead of the roads where most of the cars were driven). This was done by applying the DBSCAN clustering algorithm [9] to the imputed dataset. We used this algorithm because the clusters it produces on 2-D data are easily explainable in terms of its primary parameter ε, it detects clusters of uneven shape and size, and does not require specifying the number of clusters a priori.

Event sequences are found by substituting the real-valued (x,y) coordinates with the corresponding cluster label found previously. Behaviors are found by coarsening the trajectories into time windows, where each window counts the amount of time spent in each cluster. Information is lost during this step, because the histogram does not preserve the order that event states were visited within the time window. Behaviors are found at multiple different time granularities, by resampling using time windows of different sizes.

Interaction sessions are found by partitioning similar behaviors into non-overlapping groups using the Affinity Propagation clustering algorithm [10]. We chose this algorithm because it does not require the number of clusters to be known ahead of time and does not have a bias towards creating evenly sized clusters. Before clustering, we normalized the data by applying TF-IDF normalization. This has the effect of making very commonly visited places (such as GasTech) less important, which can help to identify anomalous behaviors that only occur for short periods of time or very infrequently in the dataset. Following this we performed L2 normalization to give each sample vector the same magnitude, allowing for a more meaningful comparison using Euclidean distance. We ran clustering separately for the following time granularities: 1-day, 2-hour, and 10-minute windows to compute interaction sessions at different time scales.

4.2 User Interface

We implemented a web-based visualization for exploring the IEEE VAST Challenge 2014 data. The tool contained two linked views: storylines that show how behaviors are related over time, and a map view which showed the event sequences in their original geospatial context. When a user selects a storyline, or bundle of storylines, the temporal event sequence for these entities is shown in the map view. The user could also zoom in on a time window to see the storyline rendered at a more fine grained level of detail. For example, the initial view shows two weeks of data with one-day time windows. The user can select "Tuesday" and then see interactions occurring on Tuesday at a 2-hour time resolution. The user can then zoom in further on any 2-hour time window on Tuesday to see that window at 10-minute increments.

4.3 Relevant Patterns Revealed

The storyline visualization readily produces visual artifacts that reveal common patterns of life. Figure 2a shows the storylines at 2 hour time granularity for a single day for Information Technology and Engineering employees at GasTech. We can see the storylines converge in the morning and afternoon, and diverge in between, presumably for the lunch break. Figure 2b shows the end of the day at a 10 minute granularity, and we can see the pattern corresponding to individuals leaving work at slightly different times. After leaving work the visualization shows that some of the individuals rejoin elsewhere.

Understanding interactions with executives is relevant to the scenario, so we filter out any employees who don't have any similar and contemporaneous behavior with executives. The filtered storyline visualization, shown in Fig. 2c, shows two interactions between executives (green) and security (red). These appear to be out of the ordinary, because most executives behavior is either always dissimilar, or always similar to others. This is manifested by either isolated or parallel lines–the two interactions with security violate this pattern, and stand out visually. Zooming in on these anomalous interactions reveal (see Fig.'s 2d and 2e) that the security employees were co-located with the executives during the late night and early morning hours. The security members performing this activity were Hennie Osvaldo and Isia Vann, who we know, from the ground truth, to be involved in the disappearance of the executives. Our multi-scale temporal event sequence visualization of GPS trajectories immediatley revealed patterns of surveillance relevant to the IEEE VAST Challenge 2014.

4.4 Expert Feedback

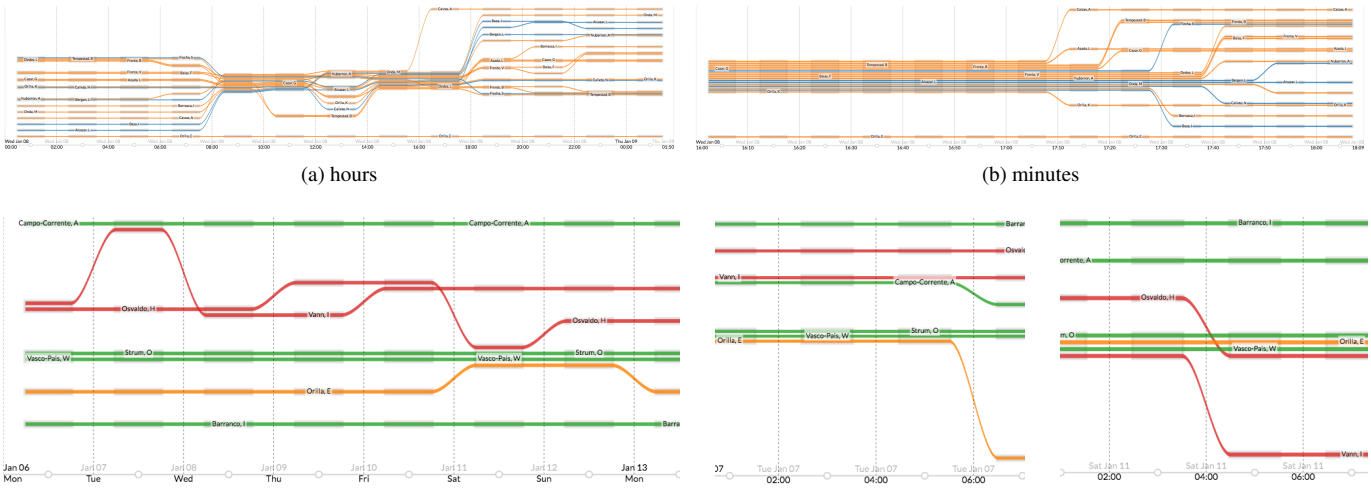
We solicited feedback from a technical analyst who investigates the collective movement patterns of individuals. The analyst commented that the IEEE VAST Challenge 2014 scenario was very similar to the types investigations she performs. She found it initially strange to think about behaviors instead of places, but she stated that after exploring with the tool, this quickly became easy to understand. She commented that the tool is valuable because it makes it easy to see who goes to the same places together, and that abnormal behaviors are visible. She also commented that our color coding scheme was helpful. She employs a similar technique where she manually assigns colors to entities based on their known attributes, and suggested that we consider color-coding to the geospatial view.

The analyst had a few comments about how the general usability of our visualization might be improved. She suggested that while it is helpful to know that two or more entities have similar behavior, analysts may want to know what that behavior is, or when an entity has assumed a particular behavior. She pointed out that it was difficult to understand time from the geospatial map alone, which has no explicit time encoding, and she suggested adding a tooltip for the map to show the time interval that a particular entity visited a particular location.

The technical analyst had several suggestions to make our visualization more effective for her applications. She said that usually she is interested in seeing what behaviors are abnormal, and asked for the capability to train a predictive model on historical data to support this. Furthermore, if this model was able to identify anomalies, she would want these to be overlayed in the visualization. She said it would be important to notify the user when an anomaly was visible at a finer time scale so that she knew when to look deeper.

5 Conclusions & Future Work

Storyline visualization is a powerful technique for communicating how temporal event sequences co-evolve. However, storylines suffer from usability issues when event sequences are long. We focused on solving this problem with a framework for rendering storyline visualizations at multiple time scales at interactive speeds. We conducted a case study using the IEEE VAST Challenge 2014 and found that



(a) hours (b) minutes

(c) days (d) hours (e) hours

Fig. 2: **Relevant storyline patterns:** (a) Tight bundling occurs during the morning and afternoon hours indicating employees working together at the office before and after lunch. (b) Zooming in on the afternoon hours shows a trickle of employees leaving the office at the end of the day. (c) The weekly patterns of the missing executives and those they interacted with reveal two days with anomalous interactions with security employees. (d, e) The anomalous interactions occur late at night or in the early morning hours with the missing executives.

our tool quickly reveals anomalous behavior relevant to the scenario. Feedback from a technical analyst who used the tool was that the tool would be helpful for her types of investigations. Additionally, the analyst highlighted some areas for future work that would be of further benefit, including providing the analyst the ability to create custom groups (through user-trainable machine learning) and anomaly detection with visual a visual representation on the storyline.

Acknowledgments

The research described in this paper is part of the Analysis in Motion Initiative at Pacific Northwest National Laboratory. It was conducted under the Laboratory Directed Research and Development Program at PNNL, a multi-program national laboratory operated by Battelle for the U.S. Department of Energy.

References

[1] D. L. Arendt. SVEN: An Alternative Storyline Framework for Dynamic Graph Visualization. In *International Symposium on Graph Drawing and Network Visualization*, pages 554—-555, 2015.

[2] D. L. Arendt and L. M. Blaha. SVEN: Informative Visual Representation of Complex Dynamic Structure. *arXiv preprint arXiv:1412.6706*, 2014.

[3] B. Bach, C. Shi, N. Heulot, T. Madhyastha, T. Grabowski, and P. Dragicevic. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):559–568, 2016.

[4] T. Blascheck, T and Kurzhals, K and Raschke, M and Burch, M and Weiskopf, D and Ertl. State-of-the-Art of Visualization for Eye Tracking Data. In *Proceedings of EuroVis*, 2014.

[5] J. Chuang, D. Ramage, C. D. Manning, and J. Heer. Interpretation and Trust : Designing Model-Driven Visualizations for Text Analysis. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 443–452, 2005.

[6] T. Crnovrsanin, C. Muelder, C. Correa, and K. L. Ma. Proximity-based visualization of movement trace data. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*, pages 11–18, 2009.

[7] J. Dominik, F. Fischer, T. Schreck, and D. A. Keim. Temporal MDS Plots for Analysis of Multivariate Data. In *Visualization and Computer Graphics, IEEE Transactions on*, volume 22, pages 141–150, 2016.

[8] P. Eades and N. C. Wormald. Edge crossings in drawings of bipartite graphs. *Algorithmica*, 11(4):379–403, 1994.

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD-96*, volume 96, pages 226–231, 1996.

[10] B. J. Frey and D. Dueck. Response to Comment on "Clustering by Passing Messages Between Data Points". *Science*, 319(5864):726, 2008.

[11] S. Gad, W. Javed, S. Ghani, and N. Ramakrishnan. ThemeDelta : Dynamic Segmentations over Temporal Topic Models. *IEEE Transactions on Visualization and Computer Graphics*, 21(5):672–685, 2015.

[12] E. R. Gansner, E. Koutsofios, S. C. North, and K.-P. Vo. A technique for drawing directed graphs. *IEEE Transactions on Software Engineering*, 19(3):214–230, 1993.

[13] P. Healy and N. S. Nikolov. *Hierarchical Drawing Algorithms*. CRC Press, 2013.

[14] IEEE VAST Challenge 2014, Mini-Challenge 2. `http://www.vacommunity.org/VAST+Challenge+2014%3A+Mini-Challenge+2`, 2014.

[15] N. W. Kim, S. K. Card, and J. Heer. Tracing genealogical data with TimeNets. In *Proceedings of the International Conference on Advanced Visual Interfaces - AVI '10*, pages 241–248, 2010.

[16] S. Liu, Y. Wu, E. Wei, M. Liu, and Y. Liu. StoryFlow: Tracking the evolution of stories. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2436–2445, 2013.

[17] T. Löwe, M. Stengel, F. Emmy-Charlotte, S. Grogorick, and M. Magnor. Visualization and Analysis of Head Movement and Gaze Data for Immersive Video in Head-mounted Displays. In *Proc. Workshop on Eye Tracking and Visualization (ETVIS) 2015*, volume 1, 2015.

[18] M. Ogawa and K.-L. Ma. Software evolution storylines. In *Proceedings of the 5th international symposium on Software visualization - SOFTVIS '10*, pages 35–42, 2010.

[19] D. C. Richardson and R. Dale. Looking to understand: the coupling between speakers' and listeners' eye movements and its relationship to discourse comprehension. *Cognitive science*, 29(6):1045–1060, 2005.

[20] S. Sakai, M. Togasaki, and K. Yamazaki. A note on greedy algorithms for the maximum weighted independent set problem. *Discrete Applied Mathematics*, 126:313–322, 2003.

[21] Y. Tanahashi, C.-H. Hsueh, and K.-L. Ma. An Efficient Framework for Generating Storyline Visualizations from Streaming Data. *IEEE Transactions on Visualization and Computer Graphics*, 6(1):1–1, 2015.

[22] Y. Tanahashi and K. L. Ma. Design considerations for optimizing storyline visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2679–2688, 2012.

[23] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science (New York, N.Y.)*, 290(5500):2319–23, 2000.

[24] C. Vehlow, F. Beck, P. Auwärter, and D. Weiskopf. Visualizing the evolution of communities in dynamic graphs. In *Computer Graphics Forum*, volume 34, pages 277–288, 2015.