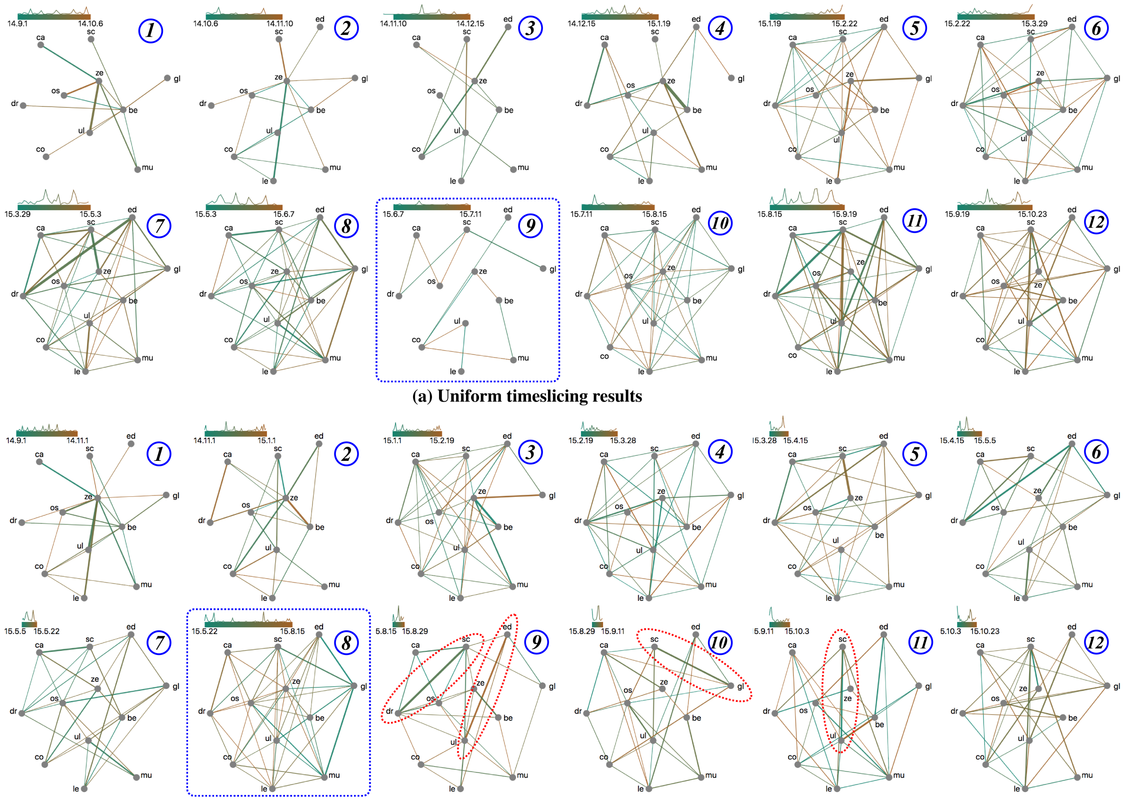
Nonuniform Timeslicing of Dynamic Graphs Based on Visual Complexity

Yong Wang* HKUST
Daniel Archambault † Swansea University
Hammad Haleem ‡ HKUST
Torsten Moeller § University of Vienna
Yanhong Wu ¶ Visa Research
Huamin Qu ‖ HKUST



(a) Uniform timeslicing results

(b) Nonuniform timeslicing results based on histogram equalization

Figure 1: Comparison of uniform timeslicing and non-uniform timeslicing using the Rugby Dataset. (a) Traditional uniform timeslicing, (b) non-uniform timeslicing. The whole stream of temporal edges are divided into 12 intervals with the sequence number marked at the top right corner of each snapshot. The top left bars and smoothed line chart show the time range of each interval (in a format of "*year.month.day*") and the edge frequency distribution, respectively. The dotted blue rectangles highlight two interesting intervals and the red dotted ellipses highlight several thick edges. The teams are: *ca - Cardiff, sc - Scarlets, dr - Dragons, os - Ospreys, le - Leinster, mu - Munster, ul - Ulster, co - Connacht, ed - Edinburgh, gl - Glasgow, ze - Zebre, and be - Benetton.*

Abstract

Uniform timeslicing of dynamic graphs has been used due to its convenience and uniformity across the time dimension. However, uniform timeslicing does not take the data set into account, which can generate cluttered timeslices with edge bursts and empty timeslices with few interactions. The graph mining filed has explored nonuniform timeslicing methods specifically designed to preserve graph features for mining tasks. In this paper, we propose a nonuniform timeslicing approach for dynamic graph visualization. Our goal is to create timeslices of equal visual complexity. To this end, we adapt histogram equalization to create timeslices with a similar number of events, balancing the visual complexity across timeslices and conveying more important details of timeslices with bursting edges. A case study has been conducted, in comparison with uniform timeslicing, to demonstrate the effectiveness of our approach.

Index Terms: Human-centered computing—Visualization—Visualization techniques—Graph Visualization;

1 Introduction

Graphs are widely used to represent the relations between different objects. Many of these graphs dynamically change over time and are ubiquitous across various applications and disciplines, such as social networks, (tele-)communication networks, biological networks, international trade networks and others. Therefore, the visualization of such dynamic graphs is of great importance in revealing their

*e-mail: ywangct@cse.ust.hk
†e-mail: d.w.archambault@swansea.ac.uk
‡e-mail: hammadhaleem@gmail.com
§e-mail: Torsten.moeller@univie.ac.at
¶e-mail: wdan1990@gmail.com
‖e-mail: huamin@cse.ust.hk

temporal evolution process and many dynamic graph visualization techniques have been proposed. According to the survey by Beck et al. [8], *small multiples*, i.e., showing a sequence of static graphs, is one of the most important and basic methods for dynamic graph visualization. Prior studies [2, 15] further demonstrated that small multiples are more effective than *animation*, the other basic method of visualizing dynamic graphs.

Sometimes dynamic graphs are *event-based*. In an event based dynamic graph, edges and nodes appear as individual events across time at the given temporal resolution of the data. An important problem is the effective selection of timeslices from the data. In the graph drawing community, uniform timeslicing is often chosen due to its simplicity. When selecting uniform s timeslices from dynamic graphs of T time units, time is divided into intervals of T/s and all events are projected down onto one plane for visualization. Uniform timeslicing has the advantage that each timeslice spans exactly the same interval of time. However, it does not take into account the underlying structure of the data.

In graph mining, studies have demonstrated that the length of time intervals selected for each timeslice strongly influences the structures that can be automatically measured from the dynamic graph [24, 27, 29, 37] and affects the performance of graph mining algorithms [16]. Prior work in the graph mining community has explored methods for timeslicing dynamic graphs effectively. Researchers have tried to identify appropriate window sizes for uniform timeslicing [34, 37] or have conducted nonuniform timeslicing [33, 35]. There appears to be no single timeslicing method that is optimal for all graph mining tasks [10, 11, 37]. Some studies have shown that different time window sizes are necessary for different analysis tasks [17, 18] and different periods of the whole dynamic graphs [11].

In dynamic graph visualization, timeslice selection received little attention beyond dividing the data into uniform timeslices. More specifically, how to select timeslices that are data dependent for effective visualization of dynamic graphs still remains an open problem. Uniform timeslicing implicitly assumes that all events will be uniformly distributed across time. However, events in a dynamic graph are rarely distributed in this way. For example, social media streams can have a burst of edges when a topic becomes important while other time periods have very few edges. Given the limited screen space, small multiples cannot afford a large number of timeslices and we need to carefully use the limited number of timeslices. However, a uniform timeslicing of such data sets will make the bursting periods suffer from visual clutter and the sparse periods remain relatively empty.

In this work, we propose a nonuniform timeslicing approach for dynamic graph visualization, which balances the visual complexity (number of edges/events per timeslice) across different intervals (Fig. 2). The timeslicing is computed based on the events present in the dynamic graph. Given a temporal resolution of the data set (e.g., second, minute, hour), we use a form of histogram equalization to make those histogram bins uniformly distributed across time. To make viewers aware of the actual length of each interval, a horizontal bar is also shown beside the graph of each snapshot. The major contributions of this paper can be summarized as follows:

- We propose a novel nonuniform timeslicing approach for visualizing dynamic graphs based on balanced visual complexity.
- We investigate the effectiveness of the proposed nonuniform timeslicing approach through a case studies, where the common uniform timeslicing approach is also compared.

2 Related Work

This work is related to prior research on appropriate timeslicing of dynamic graphs and dynamic graph visualization.

Timeslicing of Dynamic Graphs: Prior studies in the graph mining field have been run to find ideal timeslicing methods for dynamic graphs to improve the performance of algorithms for detecting structure in dynamic graphs. These methods can generally be classified into three categories: *change point detection, minimizing the variance of a graph metric and task-oriented approaches*. The methods based on change point detection evaluate the similarity between graphs of consecutive time units and detect *change points* along time to divide the whole time range [35]. The variance-based approaches [33, 34, 37] mainly determine the suitable timeslicing through minimizing the variance of certain graph metrics such as node degree, node positional dynamicity, etc. Other approaches [17, 18] determine optimal timeslicing by using the accuracy of different graph mining algorithms (e.g., anomaly detection and link prediction). In the visualization community, a fixed interval (e.g., one day, one month and one year) is often used to divide the graph into slices [6, 30, 31, 38]. However, we are not aware of methods that perform a nonuniform timeslicing for dynamic graph visualization based on graph structures across different intervals.

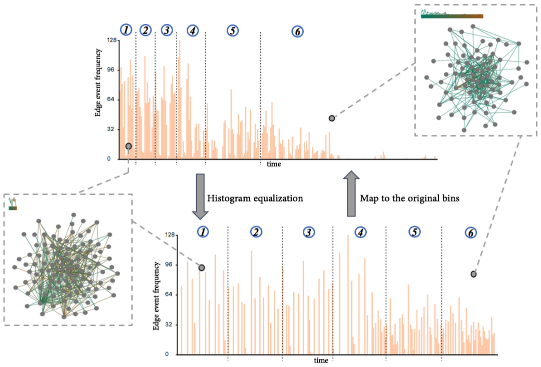


Figure 2: An illustration of the proposed nonuniform timeslicing of dynamic graph based on visual complexity. The top and bottom histograms show the original histogram of a dynamic graph dataset and the histogram equalization result, respectively. By dividing the bins evenly in the histogram equalization result, we balance the visual complexity across intervals, enhancing the detailed exploration for time periods with bursting edges while coarsening the periods with sparse edges, as shown by the two graphs of the first and last intervals.

Dynamic Graph Visualization: Dynamic graph visualization aims has been extensively explored in the past decades [1, 5, 8, 25, 26]. Animation is the most natural way to visualize dynamic graphs, as it directly maps the evolution of the graph to an animation result [8]. Prior work of this type mainly attempted to preserve the mental map (i.e. the stability of the drawing) in dynamic graph visualizations [3, 4], which are conducted through spring algorithms on the aggregated graph [12, 13, 23] or linking strategies across time [7, 9, 14, 20, 32]. However, animation is often less effective for long dynamic graphs [36], as viewers need to memorize the dynamic evolution of a graph and check back and forth to compare different graph snapshots [6]. The small multiples visualization is the other major way to visualize the temporal evolution of dynamic graphs, which shows a sequence of static representations of the graph at different time intervals [8, 25, 26]. Prior work has shown that the small multiples visualization is more effective than animation [2, 15] in terms of a quick exploration of the temporal evolution of dynamic graphs. Its major limitation is the visual scalability due to the limited space [8]. Our approach, belonging to the small multiples visualization, assigns nonuniform time ranges for each snapshot based on the visual complexity, which partially mitigates the visual scalability issue of small multiples.

3 Problem Definition and Notations

We formally define a dynamic graph and nonuniform timeslicing of dynamic graphs according to prior work [32]. For a dynamic graph defined on a node set V and edge set $E \subseteq V \times V$, an edge in this dynamic graph e_{ij} that appears at a time t for a duration d

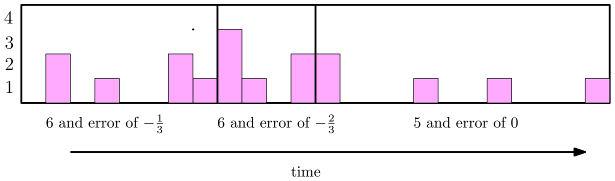


Figure 3: Equal Event Partitioning. The edges are sorted in order of earliest to latest. Given a number of timeslices, in this case 3, a target number of events is selected per timeslice ($\frac{17}{3}$ or 5.66). The algorithm counts off events, independent of timeslices, in order to fill the timeslices. Equal event partitioning distributes the error evenly through time instead of giving all the error to the last timeslice.

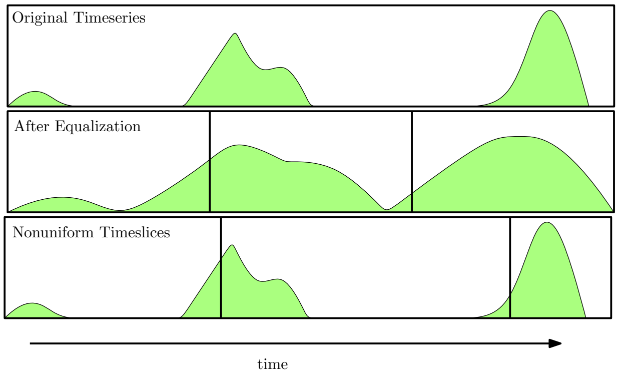


Figure 4: Histogram Equalization of Events. Histogram Equalization is adapted to event streams for dynamic graphs. The event distribution is transformed by histogram equalization to accentuate bursts. Regular timeslices are taken in the transformed space. In the untransformed space, this results in a nonuniform timeslicing that accentuates bursts in the data set and skips over areas of low activity.

is a *temporal edge*, denoted as (e_{ij}, t, d), where $e_{ij} \in E$, $i, j \in V$, $t \in [0, T]$. In this paper, we choose a fixed small duration d for each edge, which can also be referred as (e_{ij}, t). Therefore, a *dynamic graph* is a set of time-stamped edges that are ordered by their time stamp t, as defined as follows:

$$\langle V, E, T \rangle = \{(e_{ij}, t) | E \subseteq V \times V, t \in [0, T]\}. \tag{1}$$

Our definition does not consider timeslices as a basic unit of the dynamic graph. The *temporal resolution* of a dynamic graph is the minimum, positive distance in time between two events based on the accuracy of the time measurements. For example, edges could have an accuracy down to a day or down to a second. This temporal resolution is an important factor in our approach.

A *timesliced dynamic graph* $\Gamma = (G_1, G_2, ..., G_k)$ is a sequence of static graphs computed on $\langle V, E, T \rangle$ by dividing $[0, T]$ into intervals and projecting all temporal edges in a given interval down onto a 2D plane. Therefore, a *timeslicing* S on the time $[0, T]$ is:

$$S = [0, t_1), [t_1, t_2), [t_2, t_3), ..., [t_{k-1}, T], \tag{2}$$

and

$$G_l = (V_l, E_l), \tag{3}$$

$$V_l \subseteq V, E_l = \{(e_{ij}, t) | t_{l-1} \leq t < t_l\}, l = 1, 2, ..., k. \tag{4}$$

where $t_0 = 0$, $t_k = T$, E_l represents all the edge instances within the l-th time interval and k is the total number of time intervals we want to use for showing a dynamic graph. If all time intervals $[t_{l-1}, t_l)$ have uniform duration, it is a *uniform timeslicing*. Otherwise, it is a *nonuniform timeslicing*.

4 The Proposed Method

In this work, we develop timeslicing methods to optimize the visualization of dynamic graphs. Specifically, we aim to have timeslices of uniform visual complexity and two methods will be introduced.

4.1 Nonuniform Timeslicing Methods

Our definition of visual complexity in this paper is based on the number of events (in our case edges) projected into one static graph G_i of the timesliced dynamic graph. If the variance of the number of events between timeslices is small, all static representations of the graph have a similar number of events and are equally complex. Otherwise, a large variance will indicate that some timeslices are more visually cluttered, making it difficult to read the graph during bursts in the event stream. Thus, the goal is to find a nonuniform partition of $[0, T]$ whereby each graph G_i has approximately the same number of events. We accomplish this via selecting nonuniform intervals of time $[t_{l-1}, t_l)$ for which the events projected onto the graph G_l are equally distributed.

The problem of computing a uniform distribution of events within each timeslice has a strong relationship with problems in image processing [21]. All our methods to select a nonuniform timeslicing of a dynamic graph are inspired by image processing approaches originally designed to either enhance contrast or reduce errors in a digital image. In essence, *bursts of edges in the event stream will be given more emphasis through additional timeslices, while areas of the dynamic graph where there are few events will have few timeslices to represent them.*

In both visualization and graph mining community, a question that is often posed is what is the optimal number of timeslices that should be selected for a particular dataset. In a visualization context, we are frequently limited by the screen space available. Our approach is to select timeslices according to our definition of visual complexity given the budget of timeslices.

To ensure that the number of timeslices does not have an effect on the layout and that our techniques are comparable, we use the DynNoSlice algorithm [32] to draw the graph once in the space time cube. All of our techniques are applied to this same drawing in 2D + time, making them comparable.

4.1.1 Equal Event Partitioning

The most basic way to ensure a uniform distribution of events is to place the events in order and count them until a specified number of events is reached. More specifically, given $|E|$ events in the dynamic graph and a budget of k timeslices, we can simply create a new timeslice every $|E|/k$ events. Fig. 3 provides an overview of equal event partitioning. If this method is applied directly, the error can accumulate as fractional events cannot be assigned. Inspired by dithering [19], we propagate the negative or positive error closest to zero based on if we withhold from or assign an event to the next timeslice.

The strength of this method for nonuniform timeslicing is that it is very simple to implement and does ensure a uniform distribution of events across all timeslices. But its main disadvantage is that it does not use any information about the temporal resolution of the dataset and only considers events in sequence. It may also combine edges that are distant in time into one timeslicing. Thus, a histogram equalization based approach is further proposed.

4.1.2 Histogram Equalization on Events

In image processing, histogram equalization can be used to enhance the contrast of images [21]. Histogram equalization considers a histogram of all intensity values of a greyscale image (for example $[0, 255]$) and transforms the histogram by rebinning it, so that the difference between the number of elements in each bin is reduced.

Intuitively, the algorithm reduces or removes bins where the histogram values are low and devote more bins to areas where the histogram values are high, resulting in an image of higher contrast.

We adapt histogram equalization to process streams of edges in dynamic graphs as shown in Fig. 4. The algorithm starts by considering a histogram where the bins are set to a temporal resolution of the dataset greater or equal to the finest temporal resolution. The histogram represents the number of events occurring at given times across $[0,T]$. Given this histogram with $B+1$ bins, where E_i is defined to be the number of events that occur in bin i, we can define the probability distribution as $p(i)$, where:

$$p(i) = |E_i|/|E| \tag{5}$$

The cumulative distribution function $P(i)$ can then be defined as

$$P(i) = \sum_{j=0}^{i} p(j) \tag{6}$$

We can now apply a form of histogram equalization to transform the histogram of events into a new histogram of events $s_0, s_1, s_2, \ldots s_B$:

$$s_i = \lfloor (T-1) \sum_{j=0}^{i} p(j) \rfloor = \lfloor (T-1)P(i) \rfloor \tag{7}$$

This transformed version of $[0,T]$ accentuates bursts in the event stream and diminishes areas of low activity. If one were to watch the graph as a video, areas of bursty activity in the graph would be played in slow motion while areas of inactivity would be played in fast forward. In our approach, we uniformly sample this transformed histogram into k intervals, devoting more timeslices to areas of high activity, as shown in Fig. 2.

According to our experiments, if a fine-grained temporal resolution is used, the timeslicing results of histogram equalization of events and equal event partitioning are quite similar. However, if the data is recorded at coarser resolutions (e.g., month or year), histogram equalization better preserves the data granularity. Therefore, only histogram equalization of events is used in this paper.

4.2 Visualization

The graph drawing of dynamic graphs is not the focus of this paper, so we directly use DynNoSlice [32], which allows us to use the same space-time cube to draw and compare the graph visualization results by uniform and nonuniform timeslicings.

As the intervals of events for nonuniform timeslicing are not of equal duration by definition, we add a small glyph, consisting of a bar and line chart, to explicitly show the time range and edge event frequency of each interval, as shown in Fig. 1. To further reveal the detailed time information of each edge, a color mapping from teal to brown is used to encode the time order using a color time flatting approach [5] in each interval. For edges representing multiple edge events, their color is mapped to the median time of the events and the edge width indicates the number of events.

5 An Case Study on Rugby Dataset

We conduct a case study on Rugby Dataset [32] to demonstrate the effectiveness of our technique. It contains over 3000 tweets between the 12 Rugby teams in the Guinness Pro12 competition. Each tweet includes the involved teams and the accurate time stamp with a precision of one second. Fig. 1 shows the timeslicing results generated by uniform timeslicing and the proposed nonuniform timeslicing approach. Both techniques divide the whole dynamic graph into the same number of intervals (i.e., 12) for a fair comparison.

Uniform timeslicing divides the whole time range (Sept. 1st, 2014 to Oct. 23rd, 2015, 418 days in total) into 12 intervals of around 35 days each, as shown in Fig. 1a. The visual complexity across different intervals varies significantly. For example, Intervals 1, 2, 3 and 9 have sparse edges and Interval 9 even contains two disconnected graph components, revealing the infrequent interactions between the rugby teams in these time period. But some intervals like Intervals 11 and 12 of Fig. 1a have dense interactions between the rugby teams. There are several bursts in Intervals 11 and 12 (indicated by the top left line charts) and it is difficult to tell their order and structure, since uniform timeslicing does not take features of the data into consideration and often generates graphs with highly-aggregated edges for intervals of dense edges.

On the contrary, the proposed nonuniform timeslicing approach generates a sequence of graph snapshots with a more balanced visual complexity in terms of the number of edges in each interval, as shown in Fig. 1b. It is still easy to recognize the overall trend of interactions among rugby teams with the help of the time range bars in the top left corner. For example, the long time range bars in Intervals 1, 2, 3 and 8 of Fig. 1b indicates that the interactions among the rugby teams in those time periods are infrequent. More specifically, Interval 8 (late May to late August) of Fig. 1b has the longest time range bar, which corresponds to the summer break where there are no fixtures. However, at the beginning of this interval, there is a burst (teal colored edges) which corresponds to the date of the Grand Final between Munster (*mu*) and Glasgow (*gl*) at the end of the season in 2015. The final is not easily visible in uniform timeslicing because uniform timeslicing does not accentuate it.

More interesting findings can be revealed by the nonuniform timeslicing approach, when there are a series of bursting edge events. For example, as the season begins, a number of bursts occur indicated by the short time range bars of Intervals 9-12 of Fig. 1b. The nonuniform timeslicing approach is able to better accentuate certain details. For example, "scarlets_rugby" (Node *sc*) communicated the most with the team "dragonsrugby" (Node *dr*) in late August, then interacted the most with the team "glasgowwarriors" (Node *gl*) in early September, and further switched to mainly contact the team "ulsterrugby" (Node *ul*) in late September, as demonstrated by the thickest edges linked to Node *sc* in Intervals 9-11 of Fig. 1b. August (Interval 9) corresponds to just before the beginning of the season. Posting activity around preseason fixtures involving Scarlets (*sc*) and Dragons (*dr*) as well as Edinburgh (*ed*) and Ulster (*ul*) are the two most prominent edges in this interval. Scarlets-Glasgow (Interval 10) and Scarlets-Ulster (Interval 11) correspond to the first two fixtures for Scarlets in the 2015-16 season and therefore are the first two bursts of activity. The order of these bursts is apparent because they are given separate timeslices in nonuniform timeslicing, which, however, are compacted within a single interval (Interval 11 of Fig. 1a) in nonuniform timeslicing.

6 Conclusion and Future work

In this paper, we present a nonuniform timeslicing approach for dynamic graph visualization, which can balance the visual complexity across different time intervals by assigning more intervals to the periods with bursting edges and less intervals to the periods with fewer edges. A case study on a real dynamic graph (i.e., the Rugby Dataset) shows that it can achieve similar visual complexity across different time intervals for a dynamic graph and better visualize the time ranges with edge bursts.

However, several aspects of the proposed nonuniform timeslicing approach still need further work. First, the number of intervals is empirically selected. Prior studies (e.g., [34]) have explored empirical methods to determine the suitable number of intervals for graph mining tasks, but has not yet investigated from the perspective of graph visualization. Also, we define the visual complexity as the number of edges/events per timeslice. Other definitions of visual complexity for dynamic graph visualization can be further explored. Furthermore, our case study shows that our non-uniform timeslicing approach can better visualize time periods with bursting edges. However, it remains unclear which detailed graph analysis tasks can benefit from the non-uniform and uniform timeslicing approach, which is left as future research.

Acknowledgments

This work is partially supported by grant RGC GRF 16241916.

References

[1] D. Archambault, J. Abello, J. Kennedy, S. Kobourov, K.-L. Ma, S. Miksch, C. Muelder, and A. C. Telea. Temporal multivariate networks. In A. Kerren, H. C. Purchase, and M. O. Ward, eds., *Multivariate Network Visualization*, vol. 8380, pp. 151–174. Springer, 2014.

[2] D. Archambault, H. Purchase, and B. Pinaud. Animation, small multiples, and the effect of mental map preservation in dynamic graphs. *IEEE Transactions on Visualization and Computer Graphics*, 17(4):539–552, 2011.

[3] D. Archambault and H. C. Purchase. Mental map preservation helps user orientation in dynamic graphs. In *Proceedings of Graph Drawing*, pp. 475–486, 2012.

[4] D. Archambault and H. C. Purchase. Can animation support the visualization of dynamic graphs? *Information Sciences*, 330:495–509, 2016.

[5] B. Bach, P. Dragicevic, D. Archambault, C. Hurter, and S. Carpendale. A descriptive framework for temporal data visualizations based on generalized space-time cubes. *Computer Graphics Forum*, 36(6):36–61, 2017.

[6] B. Bach, E. Pietriga, and J.-D. Fekete. GraphDiaries: Animated transitions and temporal navigation for dynamic networks. *IEEE Transactions on Visualization and Computer Graphics*, 20(5):740–754, 2014.

[7] M. Baur and T. Schank. *Dynamic graph drawing in visone*. Univ., Fak. für Informatik, 2008.

[8] F. Beck, M. Burch, S. Diehl, and D. Weiskopf. A taxonomy and survey of dynamic graph visualization. *Computer Graphics Forum*, 36(1):133–159, 2017.

[9] U. Brandes and M. Mader. A quantitative comparison of stress-minimization approaches for offline dynamic graph drawing. In *International Symposium on Graph Drawing*, pp. 99–110, 2011.

[10] R. S. Caceres and T. Berger-Wolf. Temporal scale of dynamic networks. In *Temporal Networks*, pp. 65–94. Springer, 2013.

[11] P. Devineni, E. E. Papalexakis, D. Koutra, A. S. Doğruöz, and M. Faloutsos. One size does not fit all: Profiling personalized time-evolving user behaviors. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 331–340, 2017.

[12] S. Diehl and C. Görg. Graphs, they are changing. In *International Symposium on Graph Drawing*, pp. 23–31, 2002.

[13] S. Diehl, C. Görg, and A. Kerren. Preserving the mental map using foresighted layout. In *Eurographics/IEEE VGTC Symposium on Visualization*, pp. 175–184, 2001.

[14] C. Erten, P. J. Harding, S. G. Kobourov, K. Wampler, and G. Yee. Graphael: Graph animations with evolving layouts. In *International Symposium on Graph Drawing*, pp. 98–110. Springer, 2003.

[15] M. Farrugia and A. Quigley. Effective temporal graph layout: A comparative study of animation versus static display methods. *Information Visualization*, 10(1):47–64, 2011.

[16] B. Fish and R. S. Caceres. Handling oversampling in dynamic networks using link prediction. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 671–686. Springer, 2015.

[17] B. Fish and R. S. Caceres. A supervised approach to time scale detection in dynamic networks. *arXiv preprint arXiv:1702.07752*, 2017.

[18] B. Fish and R. S. Caceres. A task-driven approach to time scale detection in dynamic networks. In *Proceedings of the 13th International Workshop on Mining and Learning with Graphs (MLG)*, 2017.

[19] R. W. Floyd and L. Steinberg. An adaptive algorithm for spatial greyscale. *Proceedings of the Society for Information Display*, 17(2):75–77, 1976.

[20] D. Forrester, S. G. Kobourov, A. Navabi, K. Wampler, and G. V. Yee. graphael: A system for generalized force-directed layouts. In *International Symposium on Graph Drawing*, pp. 454–464. Springer, 2004.

[21] R. C. Gonzales and R. E. Woods. *Digital Image Processing*. Addison-Wesley, 2018.

[22] K. Higuchi, R. Yonetani, and Y. Sato. EgoScanning: Quickly scanning first-person videos with egocentric elastic timelines. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*, pp. 6536–6546, 2017.

[23] M. L. Huang, P. Eades, and J. Wang. Online animated visualization of huge graphs using a modified spring algorithm. *Journal of Visual Languages & Computing*, 9(6):623–645, 1998.

[24] M. Karsai, N. Perra, and A. Vespignani. Time varying networks and the weakness of strong ties. *Scientific Reports*, 4:4001, 2014.

[25] N. Kerracher, J. Kennedy, and K. Chalmers. The design space of temporal graph visualisation. In *Proceedings of the Eurographics Conference on Visualization (EuroVis '14)*, pp. 7–11, 2014.

[26] N. Kerracher, J. Kennedy, and K. Chalmers. A task taxonomy for temporal graph visualisation. *IEEE Transactions on Visualization and Computer Graphics*, 21(10):1160–1172, 2015.

[27] G. Krings, M. Karsai, S. Bernhardsson, V. D. Blondel, and J. Saramäki. Effects of time window size and placement on the structure of an aggregated communication network. *EPJ Data Science*, 1(1):4, 2012.

[28] Y.-Y. Lee, C.-C. Lin, and H.-C. Yen. Mental map preserving graph drawing using simulated annealing. In *Proceedings of the 2006 Asia-Pacific Symposium on Information Visualisation*, pp. 179–188. Australian Computer Society, Inc., 2006.

[29] B. Ribeiro, N. Perra, and A. Baronchelli. Quantifying the effect of temporal resolution on time-varying networks. *Scientific Reports*, 3:3006, 2013.

[30] S. Rufiange and M. J. McGuffin. DiffAni: Visualizing dynamic graphs with a hybrid of difference maps and animation. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2556–2565, 2013.

[31] L. Shi, C. Wang, and Z. Wen. Dynamic network visualization in 1.5D. In *Proceedings of the 2011 IEEE Pacific Visualization Symposium*, pp. 179–186, 2011.

[32] P. Simonetto, D. Archambault, and S. Kobourov. Drawing dynamic graphs without timeslices. In *Proceedings of the 25th International Symposium on Graph Drawing and Network Visualization (GD 2017)*, pp. 394–409. Springer, 2017.

[33] S. Soundarajan, A. Tamersoy, E. B. Khalil, T. Eliassi-Rad, D. H. Chau, B. Gallagher, and K. Roundy. Generating graph snapshots from streaming edge data. In *Proceedings of the 25th International Conference Companion on World Wide Web*, pp. 109–110, 2016.

[34] R. Sulo, T. Berger-Wolf, and R. Grossman. Meaningful selection of temporal resolution for dynamic networks. In *Proceedings of the Eighth Workshop on Mining and Learning with Graphs*, pp. 127–136, 2010.

[35] J. Sun, C. Faloutsos, S. Papadimitriou, and P. S. Yu. Graphscope: parameter-free mining of large time-evolving graphs. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 687–696, 2007.

[36] B. Tversky, J. B. Morrison, and M. Betrancourt. Animation: can it facilitate? *International Journal of Human-computer Studies*, 57(4):247–262, 2002.

[37] S. Uddin, N. Choudhury, S. M. Farhad, and M. T. Rahman. The optimal window size for analysing longitudinal networks. *Scientific Reports*, 7(1):13389, 2017.

[38] S. van den Elzen, D. Holten, J. Blaas, and J. J. van Wijk. Reducing snapshots to points: A visual analytics approach to dynamic network exploration. *IEEE Transactions on Visualization and Computer Graphics (VAST 2015)*, 22(1):1–10, 2016.