Visualizing Dynamic Networks of Long Sequences with Pixel Matrix Array

Lijing Lin[*] Peking University
Liwenhan Xie[†] Peking University
Zhuo Zhang[‡] Qihoo 360 Technology Co. Ltd.
Xiaoru Yuan[§] Peking University

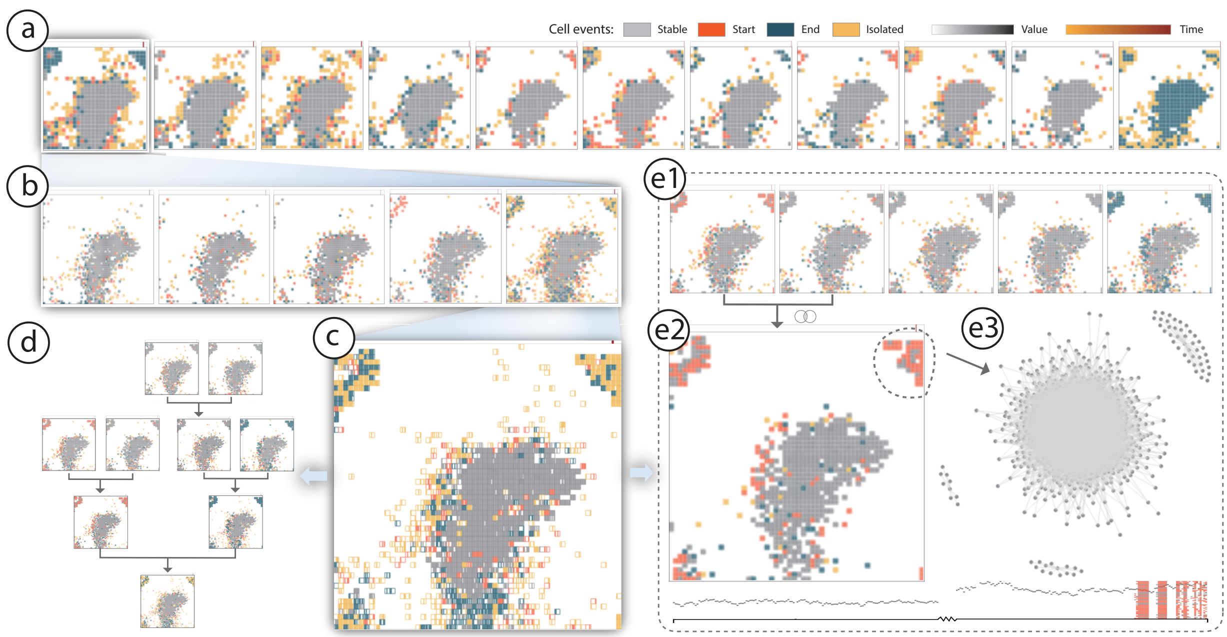

Figure 1: Exploration on a dynamic network with Pixel Matrix Array in a top-down manner. The view illustrates a process to explore a merged matrix in detail. Each matrix cell represents a community in the network. (a) An evolution overview with a series of matrices calculated by high merging and packing strength. (b) Select an merged matrix and adjust the strength lower. (c) Zoom in the matrix and explore the feature status during the merged time steps. (d) Revelation for the hierarchical merging structures of the merged matrix. (e) Split the matrix into individual snapshots, use logic operations for comparison, lasso groups with particular features and inspect their topology structures.

Abstract

We propose Pixel Matrix Array, a novel visual analytic approach supporting interactive exploration of large dynamic networks with long sequences. In our work, group features of each time step of the dynamic network are automatically extracted according to the network community information, or predefined by users. The feature information of each time step is then projected onto a pixel matrix while maintains the adjacent relation globally between features. The matrices are then folded and compressed to provide a compact representation of the whole dynamic networks in limited display space while still keep key information visible. Users are able to interact with the information in the network up to a few thousand time steps by unfolding or zooming the matrices.

Index Terms: Human-centered computing—Visualization—Visualization application domains—Visual analytics;

[*]e-mail: lijinglin29@gmail.com
[†]e-mail: xieliwenhan@pku.edu.cn
[‡]e-mail: zhuowater@gmail.com
[§]e-mail: xiaoru.yuan@gmail.com

1 Introduction

Dynamic network is a series of snapshots describing how entities and their relationships change over time. When the data involves thousands of time steps, which is common in computer network communication logs, etc., a summary of the evolution process of the dynamic network is in demand for analysts to understand the past, analyze trends and make decisions. Two general approaches for dynamic network visualization are animation and small multiples. However, animation suffers from the heavy burden of human perception, especially for long sequence, and current visual designs based on small multiples are usually too abstract to gain much insights from an overview. For example, van den Elzen et al. reduce snapshots to 2D points according to integrated features [1]. Although this effectively reveals various evolution states, users could only look for details in the node-link diagram that is complicated to interpret, particularly for large networks. Another work from Sallaberry et al. develops a new clustering algorithm and visualize the dynamic network with the clusters [2]. Using cluster as visual unites, they success in creating clearer cluster-link diagrams. Nevertheless, adapting edge splatting from Burch et al. [3], their timeline overview still suffers from edge crossings, hence providing limited insights.

Our intention is to narrow the gap between an overview and detailed information for dynamic network analysis. As inner-connected groups of vertices suffice to unveil the existence of a non-trivial internal network organization at a coarse level, we regard a group of vertices to be the smallest unit of the network. In this way, we scale down the snapshots, assign matrix cells to the groups, and encode group features with colors and glyphs by adopting pixel-oriented techniques. To address the problem of overwhelmingly long matrix series, we further merge similar adjacent matrices and pack neighboring matrix cells. Our prototype system includes both overview and detailed view for data navigation in a top-down manner.

2 Abstract Network To Matrix

Converting the dynamic network into a pixel matrix array like the middle part of Figure 2 takes five steps in total.

Detection: We regard certain group of vertices as the smallest unit in the network. For non-tag data, we adopt community detection algorithm for tags. Or we require attribute distinguishing groups.

Mapping: Binding entities into tight-linked groups globally, we project the groups into different matrix cells by a slightly modified MDS algorithm, maintaining group distance under a measurement which ensures that close groups are similar in components and active periods are tightly linked. The trick is to discrete MDS results by re-aligning groups that collide in shape "S" with BFS algorithm.

Merging: Laying out matrix array directly may bring difficulties to discover evolution patterns due to the limited space. Inspired by [3], we merge neighboring matrices whose distance is below the configured threshold in a greedy way from the start. Alternatively, neighboring matrices could be merged manually. In Figure 1, (c) is merged from (e1) in the hierarchical order as (d) illustrates.

Packing: Now that neighboring matrix cells are similar in a sense, we bind them into a larger unit. Analogous to image blurring, packing means to divide the matrix into blocks and represent the block value as the mean value of cells inside the block. It's supported for users to adjust the packing strength according to their need.

Feature Encoding: Matrix cells leave grand design space for feature encoding. We use colors for group events, grey level for value (group size) as the legend of Figure 1 shows. Besides, some glyphs are adopted to reveal the stability status of the cell value during the merged period, i.e. increase, reduction, fluctuation, etc..

Figure 2: User's interface with the synthetic data. The interface includes timeline view, compact matrix array view and detailed view. Tool bars support flexible visual encoding and interactions.

3 Data Exploration Flow

We import a synthetic data into our system to demonstrate the top-down exploration flow and validate its utility. The artificial dynamic network is constructed from a "small world" model. It involves 1,025 time steps, along with 8,400 vertices and 42,000 edges. By purpose we add in four major events: sudden entrance and disappearance of some communities, separation and combination of communities. Figure 2 is a screen shot of the graphical interface after data import, which includes a time line view, a compact matrix array view and a detailed view from top to bottom.

3.1 Evolution Summary

The timeline view serves as a supplementary to the other views. Its x-axis and y-axis (logarithmic scale) stands for time and vertex amount, respectively. And the merged periods are marked by separate lines. An exponential growth and an abrupt drop and rise of entities is easily perceived in the start place and the quartile of the time line severally. In general, the scale of the dynamic network stays steady.

At first, we set the merging threshold small to see how the dynamic network grows. Since the community scale is moderate, there's no need to pack matrix cells. However, packing is practical for easing cognition burden when data are large (see comparison between Figure 1 (a) and (b) where packing strength varies).

Then, a clear division of four major phases emerges as the individual separated line in the time line view suggests (see Figure 2). As a hint, the four highly merged matrix are highlighted in the array automatically. From the current array, basic evolution patterns are derived. In the beginning, communities come into being fast. During the first long-span merged period, though the amount of vertices varies, the community structures maintain relatively stable. When it comes to the second transitional stage, the ancestral groups die out and new groups form, making the scale back to normal.

3.2 Detail Exploration

As we use colors to encode group states in the merged time (see Figure 1 legends), comparing the last three large-merged matrices, we speculate that the middle one is likely to be the slow transition of the others and verify it in the detailed view. For more intuitive comparison, we convert the three matrix into 1D view. Then we do logical operations among the matrices spitted from the middle one and find they transit smoothly, which conforms to the model design.

In addition to the mentioned above, our system offers other convenient interactions for different analytic tasks in the detailed view. For instance, topological structure of the snapshot or certain lassoed groups could be shown in a node-link diagram. And the time distribution and relevant raw data of each group in the matrix could be checked in a list. Moreover, When groups are lassoed, they would be highlighted in the time line view to show the time distribution.

4 Conclusion and Future Works

We present Pixel Matrix Array for analysis on large dynamic network with long sequences. First, we treat the complex dynamic network as flexible-scale matrix series, which is crucial for understanding its evolution process. Then, for insight gaining, various of visualization and interaction methods such as pixel-based techniques are introduced to our system for data exploration from overview to details, which accords with human cognition. Its utility is validated by the data exploration flow on an artificial data.

In the future, a comprehensive expert interview is necessary to accommodate this novel system into practical needs. Besides, our work has a natural advantage for large screen display, because large screen has more space for the matrix array and promotes detail inspection. However, special interaction techniques and customized visualization expressions on large screen need further consideration.

References

[1] Stef van den Elzen, Danny Holten, Jorik Blaas, and Jarke J van Wijk. Reducing snapshots to points: A visual analytics approach to dynamic network exploration. *IEEE transactions on visualization and computer graphics*, 22(1):1–10, 2016.

[2] Arnaud Sallaberry, Chris Muelder, and Kwan-Liu Ma. Clustering, visualizing, and navigating for large dynamic graphs. In *International Symposium on Graph Drawing*, pages 487–498. Springer, 2012.

[3] Michael Burch, Corinna Vehlow, Fabian Beck, Stephan Diehl, and Daniel Weiskopf. Parallel edge splatting for scalable dynamic graph visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2344–2353, 2011.