Towards Exploratory Visualization of Multivariate Streaming Data *

Zaixian Xie

Computer Science Department
Worcester Polytechnic Institute
xiezx@cs.wpi.edu

ABSTRACT

More and more researchers are focusing on the management, querying and pattern mining of streaming data. The visualization of streaming data, however, is still a very new topic. In this proposal, we discuss our plan to construct a multivariate streaming data visualization system. Three subtasks are identified, including streaming data abstraction, visualization and interaction techniques for streaming data, and visualizing change in data streams. An overview of proposed solutions is provided.

Keywords: Multivariate visualization, data streams.

1 INTRODUCTION

A big challenge the visualization community is facing is that the volume of some datasets is becoming bigger and bigger. Traditional visualization methods cannot convey useful information to people because of limited canvas space and people's perception capabilities. Researchers have proposed a variety of preprocessing algorithms and novel visualization methods to diminish visual clutter to make visualization techniques usable for large datasets [2, 3, 10].

However, streaming datasets, a special type of large datasets, cannot be processed effectively using normal methods for static large datasets. In streaming datasets, new datapoints are generated every day, every hour, or even every second, for example, in application areas such as meteorology, network monitoring, and market analysis. It is true that, in each specified time horizon, we can regard the existing datapoints in a streaming dataset as a regular large dataset and apply algorithms and visualization methods designed for large datasets, but we can see some obvious limitations of this natural solution:

1. Although we can successfully visualize the datasets according to all existing datapoints, probably we have to restart the entire processing algorithm when new datapoints arrive, especially when we use data abstraction methods to process large datasets, such as sampling, clustering, and dimension reduction. The system becomes less efficient or even unusable since sometimes the dataset refresh rate is very fast.

2. For streaming data, analysts sometimes are more interested in dynamic patterns instead of static ones. They probably want to know how data patterns evolved over time or predict future change trends, for example, in the application area of financial analysis. These tasks are difficult to accomplish using existing visual analysis methods for large datasets.

Based on the above analysis, a framework for streaming data visualization should not only provide an effective and efficient solution to stream data abstraction, but also visually reveal the data trends, as well as data patterns via dynamic and static figures. The derived subtasks are as follows:

1. *Streaming data abstraction algorithms* : Clustering is an effective method to find data patterns in large datasets. We plan to develop online clustering algorithms to preprocess streaming data to obtain information about data patterns in an arbitrary time horizon and the change of patterns over time.

2. *Visualization for streaming data* : A natural solution is adding new datapoints to visualizations once they arrive. We must develop some mechanism to remove old or unimportant datapoints to reduce visual clutter when there are too many datapoints in the figure. In addition, some interaction techniques should be integrated into our framework to support analysts in retrieving data patterns, predicting data change trends, and other tasks.

3. *Visualization for change of data patterns:* This subtask aims to directly visualize the change of data patterns over time. Moreover, we plan to design interactions to facilitate the observation of data trends, as well as the relationship between data trends and streaming data.

2 STREAM ABSTRACTION

Although the data mining community has developed some clustering algorithms for data streams [1, 4], improvement is needed to facilitate visual analytics on data streams. We plan to construct an online clustering algorithm based on the BIRCH algorithm [11]. The reason to choose BIRCH is that it constructs a hierarchical structure of clusters, which would help analysts understand the dataset structure and data patterns. Moreover, BIRCH is a one-pass clustering algorithm which we believe can be easily extended to stream mining. In order to perform tasks related to visual mining of streaming data, our algorithm will have the following features:

1. It will detect the change of sizes (number of points), volumes (the occupied space) and positions for clusters, and the appearing, vanishing, merging and splitting of clusters.

2. This algorithm will accept users' input related to timestamps. For example, users can set a warming up time period, in which an initial cluster tree is generated. Then the change of clusters will be detected based on thresholds obtained during warming up.

3 ACCUMULATING VISUALIZATION AND LOAD SHEDDING

In this section, we discuss our solution to visualize streaming data. A natural idea is to add new datapoints one by one into visualizations when they arrive. We name this technique *accumulating visualization*. Obviously, clutter will be more and more serious for most of multivariate visualizations. It is necessary to discard some datapoints or remove them from visualizations. Here we borrow the term *load shedding* [9] used in data stream management systems to represent our actions to discard or remove datapoints. Thus, we have to answer two questions: (1) When do we need to shed datapoints? (2) Which datapoints do we need to shed?

For the first question, our strategy is to apply a metric to measure visual clutter for different visualizations [8] and set a reasonable threshold. When clutter is bigger than the threshold, we start the shedding process until the metric decreases to below the threshold.

*This work is supported under NSF grant IIS-0414380.

Before answering the second question, we clarify that our goal for visualizing streaming data is to highlight the dominant data patterns. Thus outliers should be processed separately. We choose datapoints to discard or remove according to the following policies:

1. If a new datapoint belongs to an existing cluster, we change attributes of this cluster, e.g. color, instead of visualizing this datapoint. So visualizations are not cluttered up but still can convey enough information to users.

2. If an existing datapoint in a visualization is detected to be an outlier, we tag it with visual attributes or move it into a visualization specially for outliers. It can go back to a normal cluster or be discarded according to users' selection or changes in other clusters. For example, a cluster might grow to contain this outlier.

3. If a cluster is old or has not changed for a long time and users are interested in only new data patterns, we remove this cluster.

In order to help users observe data patterns in streams, we will implement the following interactions:

1. *Brushing* : We plan to support time-based brush whose constraints contain timestamps. This idea is inspired by the Timebox of Hochheiser and Shneiderman [6]. We will focus on integrating it with value-based [7] and structure-based brushes [3]. Thus analysts could explore data not only in different levels of detail and data ranges of interest, but also from different time horizons.

2. *Separation* : We can split the overall visualization into several views, with each view only containing datapoints arriving at a specified time period. Thus analysts can easily compare data patterns in different time periods.

4 Visualization for Change in Streams

In Section 3, we focus on data patterns. Now we consider how data changes over time. In other words, we want to visualize the change of data patterns. We focus on change of clusters in streams because clusters denote the dominant data patterns.

In order to formalize our description for rendering and interaction, we introduce a new term, *pattern space*, to describe the change of clusters. As we know, the BIRCH algorithm can generate a cluster tree, where each node is a cluster and a parent cluster always contains its children clusters. In stream processing, cluster trees with different timestamps form a forest. Note that nodes in adjacent trees probably are the same cluster, or one is from the other after merging or splitting. In order to represent this relationship, we add an arrow from node A to node B if (1) they are in the adjacent trees; (2) and A is older than B; (3) and they correspond to the same cluster, or the merging of A and some other clusters forms B, or A splits into B and some other clusters.

Our basic idea to visualize pattern space is using streamlines in flow visualization, along with inspiration from ThemeRiver [5]. In ThemeRiver, Havre et. al. use a river metaphor to convey key notions in the document collection. A river's directed flow, composition, and changing width denote documents' time evolution, selected thematic content and thematic strength, respectively. The overview of our approach is as follows: we regard each cluster as a particle, so its change of position corresponds to the particle's movement, which can be represented by a streamline. Moreover, for each streamline, we can set different sectional areas and colors for different positions denoting timestamps. On a specified position, the sectional area is directly proportional to this cluster's volume and the color represents its timestamp. Such a figure can be static or dynamic. The former would convey the overview of changes in data patterns, and the latter enables users to track the change.

We plan to introduce the following interactions to help analysts extract useful information from streaming data.

1. *Focus+Context* : When users select an arbitrary position on a streamline, we will use a popup window to visualize a subset of the streaming data, which is composed of datapoints in the cluster corresponding to this position.

2. *Brushing* : We allow users to define two kinds of brushes, structure-based and temporal. The former allows users to select clusters in different levels of detail. The latter supports the selection of clusters in a specified time period.

3. *Separation* : Users can split the whole time axis into subsections, and then use multiple views to show these streamlines, with each view corresponding to one subsection. This technique aims to facilitate the observation of details, removal of clutter and comparison of data trends at different times.

Please note that *brushing* and *separation* here are similar to those in accumulating visualizations. Thus we plan to create a new operation, *linking*, to link accumulating visualizations and pattern space visualizations. When users apply *brushing* or *separation* on either of the visualizations, the other one will automatically create brushes or do separation. Thus users could easily find the relationship between data patterns and change of data patterns.

References

[1] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. *VLDB*, pages 81–92, 2003.

[2] A. Dix and G. Ellis. By chance - enhancing interaction with large data sets through statistical sampling. *Proc. Advanced Visual Interfaces*, pages 167–176, 2002.

[3] Y. Fua, M. Ward, and E. Rundensteiner. Structure-based brushes: A mechanism for navigating hierarchically organized data and information spaces. *IEEE Trans. Visualization and Computer Graphics*, 6(2):150–159, 2000.

[4] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan. Clustering data streams: Theory and practice. *IEEE Trans. Knowl. Data Eng.*, 15(3):515–528, 2003.

[5] S. Havre, E. Hetzler, P. Whitney, and L. Nowell. ThemeRiver: Visualizing thematic changes in large document collections. *IEEE Trans. Visualization and Computer Graphics*, 8(1):9–20, 2002.

[6] H. Hochheiser and B. Shneiderman. Dynamic query tools for time series data sets: timebox widgets for interactive exploration. *Information Visualization*, 3(1):1–18, 2004.

[7] A. Martin and M. Ward. High dimensional brushing for interactive exploration of multivariate data. *Proc. IEEE Visualization*, pages 271–278, 1995.

[8] W. Peng, M. Ward, and E. Rundensteiner. Clutter reduction in multidimensional data visualization using dimension reordering. *Proc. IEEE Symposium on Information Visualization*, pages 89–96, 2004.

[9] N. Tatbul, U. Çetintemel, S. B. Zdonik, M. Cherniack, and M. Stonebraker. Load shedding in a data stream manager. *VLDB*, pages 309–320, 2003.

[10] J. Yang, D. Hubball, M. Ward, E. Rundensteiner, and W. Ribarsky. Value and relation display: Interactive visual exploration of large data sets with hundreds of dimensions. *IEEE Trans. Visualization and Computer Graphics*, 13(3):494–507, 2007.

[11] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: an efficient data clustering method for very large databases. *SIGMOD Record*, 25(2):103–114, June 1996.