Data Visualization Management Systems

Eugene Wu
Columbia University
ew2493@columbia.edu

1. INTRODUCTION

The holy grail of visualization systems makes exploring different data facets so intuitive, and recommends views that are so relevant, that users rapidly converge onto valuable insights – irrespective of dataset size. Unfortunately, existing systems fall far short of this goal.

Most visualizations are produced by retrieving raw data from a database and using a specialized visualization tool to process and render it. Although the database can sometimes be used to filter the raw data (e.g., return data within a visible bounding box), visualization tools try to avoid roundtrips to the database by managing their own results cache and executing data transformations directly.

This approach is plagued with high communication costs, a loss of visualization-level semantic information, and over-engineering to ensure low latency (clients often re-implement data-processing operations).

I propose instead to blend these two systems into a *Data Visualization Management System* (DVMS) to make all database features available for visualization. The DVMS expresses data processing, visualization specifications, and interaction queries through a consistent language that is compiled and optimized as a whole. This unified approach leads to exciting opportunities and research problems that I detail below.

1.1 Perceptual Optimizations

Visualizations should be both pixel-perfect and fast to compute. By taking advantage of human perceptual errors, as well as the constraints on the output resolution, the DVMS can intelligently optimize the visualization execution to minimize latency.

In addition, execution placement can vary depending on the resources that are available to the system. For instance, if the client browser has support for WebGL, then the data processing and computation may be moved to the client. In contrast, if the network bandwidth is low or the client lacks computational resources, then the DVMS may render the visualization directly on the server.

1.2 Contextual Suggestions

Recommending relevant or surprising data is a key tool as users interactively explore their datasets. Prior work has focused on recommending visualizations and queries based on singular, but semantically different features such as data statistics, image features, or historical queries. A DVMS controls, and can thus use, all of these features to construct more salient recommendations to the user. For example, image features such as mountain ranges may be of interest when rendering maps, whereas the slope of a line chart is important when plotting monthly expense reports.

1.3 Relational All the Way Down

A unified execution framework also means that every part of the data processing, layout, and rendering process is modeled using a consistent data model. Thus users can issue queries to the inputs and outputs of any execution step. For example, the rendered pixel objects on the screen are represented in a relational table that can be queried using a SQL-like language. Correspondingly, the system can express interaction events as triggers over subsets of the rendered elements.

1.4 Full-stack Provenance

A key benefit of a single system approach is that provenance information can be captured from the source data all the way to the pixels that are rendered on the screen. By doing so, users can also execute forward provenance queries to find the rendered elements that depend on a particular subset of the database, or inversely, access the records that computed a rendered element of interest (e.g., a bar in a bar chart). Common interaction techniques such as brushing and linking can be declaratively expressed as a provenance query, thus allowing the DVMS to optimize and scale the interactions to large datasets.

In addition, the system can also track the provenance of how the visualization *changes* and the interaction history. Exploring how this information can be uniformly modeled and queried is an exciting area of research.

1.5 Interaction Based Execution

An interactive visualization means that not only the initial rendering, but subsequent interactions must be executed with low latency. In an interactive environment, queries are unlikely to differ dramatically from previous queries. This insight can be leveraged by intelligently materializing intermediate query results that will likely be used in subsequent queries.

7th Biennial Conference on Innovative Data Systems Research (CIDR '15) January 4-7, 2015, Asilomar, California, USA..