Articulate: a Conversational Interface for Visual Analytics

Yiwen Sun* Jason Leigh† Andrew Johnson ‡ Dennis Chau §

Electronic Visualization Laboratory, University of Illinois at Chicago

Abstract

While many visualization tools exist that offer sophisticated functions for charting complex data, they still expect users to possess a high degree of expertise in wielding the tools to create an effective visualization. This poster presents *Articulate*, an attempt at a semi-automated visual analytic model that is guided by a conversational user interface. The goal is to relieve the user of the physical burden of having to directly craft a visualization through the manipulation of a complex user-interface, by instead being able to verbally articulate what the user wants to see, and then using natural language processing and heuristics to semi-automatically create a suitable visualization.

Index Terms: H.5.2 [Information Interfaces and Presentation]: User Interfaces—Graphical user interfaces

1 Introduction

Visual analytic tools have been used in a variety of disciplines to synthesize information and derive insight from abstract data-sets. The combined complexity of certain tools and visual design itself can be daunting, so much so that the modification of visual parameters isn't predictable, even for visualization experts. Unfortunately the users of such tools are usually domain experts with marginal knowledge of visualization techniques. These users often experience difficulties when trying to manage simple visual parameters, such as specifying a desired graph type for a given data set, or assigning proper data fields to certain graph dimensions. To facilitate the use of these tools by domain experts, we propose a semi-automated visual analytic model: *Articulate*. The goal is to provide a streamlined experience to non-expert users, allowing them to focus on using the visualizations effectively to generate new findings.

Mackinlay et al. described Show Me[7], an integrated set of user interface commands and defaults that automatically generate visual presentations based on VisQL specification language. Users place data into columns and rows to specify VisQL commands. In order to generate meaningful visualizations, an understanding of the relationships between columns and rows is needed. Rezk-Salama et al. demonstrated a semantic model for automatic transfer function assignment in volume rendering applications [8]. However the design of the semantic model is such that the presence of computer scientists and domain experts are needed when generating a visualization. Lastly, a 2007 National Science Foundation workshop report on "Enabling Science Discoveries Through Visual Exploration" [6] indicated that "there is a strong desire for conversational interfaces that facilitate a more natural means of interacting with science." This inspired us to adopt a conversantional interface in the semi-automated visual analytic model.

Our Conversational Visualization Language (ConVL) is in many ways an homage to the simplicity of languages such as Apple's HyperTalk [5]. It differs from other languages, such as Gnuplot script or Graphics Production Language [9], in that each command is mapped to a query rather than requiring specific assignment. It allows the user to explain what they want to see in a brief manner. This type of conversational language is ideal for integration with a speech recognition module - one of the ultimate goals of *Articulate*.

*e-mail: ysun25@uic.edu

†e-mail: spiff@uic.edu

‡e-mail: ajohnson@uic.edu

§e-mail: koracas@gmail.com

2 Visual Analytic Model

The system model for *Articulate* is shown in Figure 1. The main idea is to integrate the user's requirements and data properties to automatically generate an appropriate visual result.

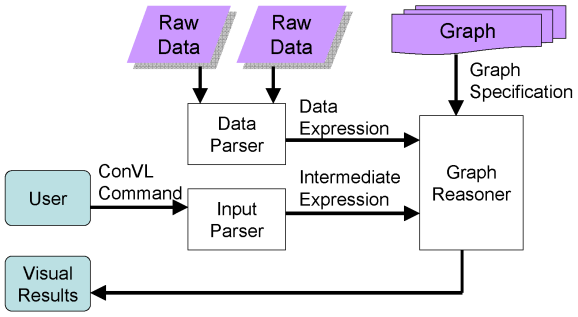

Figure 1: System Model for *Articulate*

There are three main components in the model: *ConVL input parser, data parser*, and *graph reasoner*.

2.1 ConVL Input Parser

Visualization requirements are expressed in ConVL. The grammar of ConVL will be given in Section 3. The input parser translates the ConVL command to a sequence of intermediate expressions which are presented as a pipeline of lower-level instructions. Similar to other low-level visualization languages, it specifies topology assignment, visual parameters, and general graph type. It provides the graph reasoner with a basis for the optimization and generation of the final visual product.

2.2 Data Parser

The data parser reads the original data file searching for data fields such as attribute names, data types (numerical values, text or time) and data structures (tables or trees). Furthermore, data properties are determined such as statistics features (the minimum and maximum values, number of categories). This information is sent to the graph reasoner for visual parameter assignment.

2.3 Graph Reasoner

In our model, the reasoner is the key component responsible for the semi-automated generation of visualizations. Information about the data from the data parser and the intermediate expressions generated from parsing the user's input are funneled into the Graph Reasoner. Properties of various graph types are also funneled in via the graph specification, which include dimensions, layouts and data restrictions. The reasoner uses this information to determine the most appropriate visualization to generate.

For example, a command such as "compare GDP among country" will result in the creation of a bar chart or a line chart. These graphs, according to Abela's chart chooser [4], are often used for

comparison tasks. To make a decision between these candidates, the reasoner checks the data property. It finds that the attribute "country" is a text type with categorized values. Hence a bar chart will be most suitable. Then the reasoner follows the graph specification of a bar chart to the assign attribute "GDP" to the range axis, and assign each instance of the attribute "country" to a seperate bar on the domain axis. Finally the graph generator is invoked.

3 ConVL

ConVL is a high-level conversational language based on first-order predicate logic, specified in a formal grammar. Additional considerations were introduced to attain the simplicity and flexibility needed for a conversational language. To accommodate the different purposes of visual analytics, ConVL is divided into two major categories: visualization commands, and manipulation commands.

Visualization commands are ones that describe the semantics of the visualization task. Each plot type has its own advantages and disadvantages for certain types of data. For example: bar charts are often used for comparing values by category, whereas pie charts are good at illustrating relative magnitudes or frequencies [1]. To determine the proper visualization, further classifying was needed. The visualization commands were divided into four subcategories: 1) find the relationship between two or more attributes using scatter plot, bubble plot or radar plot; 2) compare values over time or categories using time-series plot, bar chart or area chart; 3) use network graphs to show connection and communication; and 4) explore further visual analytic results using histogram or dendrogram. With each subcatergory, there is a list of possible commands. The following are those for "Comparison" under subcategory 2:

$$
\begin{aligned}
<comparison> \quad ::= \quad & COMPARE < attrib\_list > \\
| \quad & COMPARE < attrib\_list > OVERTIME \\
| \quad & OVERTIME\ COMPARE < attrib\_list > \\
| \quad & COMPARE < attrib\_list > AMONG/BY \\
& < attribute > \\
| \quad & AMONG/BY < attribute > COMPARE \\
& < attrib\_list >
\end{aligned}
$$

As shown in the grammar above, a statement is composed of an action with one or two parameters. The action verb is chosen to identify the semantics of this visualization task. "Compare" was chosen because it was the word users would most likely choose when trying to examine similarities or differences. Additionally all combinations of a command are present in the grammar. This allows the user to achieve the same semantics under different syntax. This is exemplified by $COMPARE < attrib\_list > OVERTIME$ and $OVERTIME\ COMPARE < attrib\_list >$. In the future however we intend to leverage natural language processing techniques to minimize the need to encode all the anticipated ways in which a command may be expressed.

Manipulation commands are used to alter the visual metaphors. Similar to the visualization commands, they are also defined as a list of statements. These commands however, focus on the mapping or assignment of visual metaphors. For example:

$$
\begin{aligned}
<mapping> \quad & ::= \quad MAP < attribute > TO\ COLOR \\
<assignment> \quad & ::= \quad SET\ SIZE\ OF\ POINT\ TO < value >
\end{aligned}
$$

4 Results

Our framework is being developed in Java and the ConVL parser is implemented using JFlex [3] and CUP [2]. Figure 2 shows an example of the graphical user interface. The left side of the window lists the data attributes. The bottom contains an input window that accepts ConVL commands and displays all the previous input; the main body is the visualization panel, which shows each visual representation in a seperate view.

Figure 3 illustrates the ConVL command "Relate X,Y,Depth and Conductivity,Temperature,pH" applied on a hydrologic dataset with

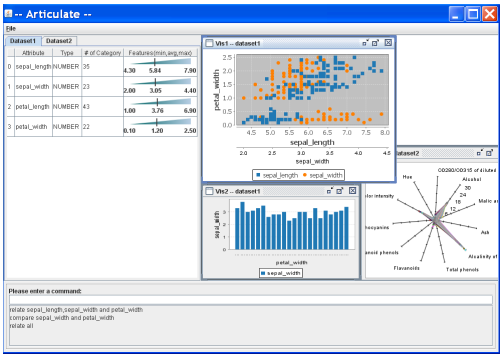

Figure 2: GUI of the framework

10 attributes and 5000 instances. The result shows pairwise relationships among specified attributes using a scatter plot matrix.

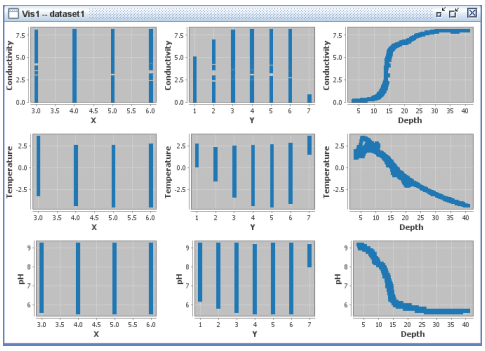

Figure 3: Relationship command represented by scatter plot matrix

5 Conclusions and Future Work

In this poster, we presented our current work on *Articulate*, a semi-automated visual analytic model that incorporates a novel user-interface based on ConVL, a high-level conversational language. We believe this design will help visualization novices produce quick and effective visual representation without extensive knowledge of visualization techniques. It also highlights the trend toward increasing use of speech driven user-interfaces for controlling complex computing systems. Our step is to create a suite of *Analysis* commands to complement the visualization and manipulation commands, and evaluate the system using populations of visualization novices. Future directions in this research will include the integration of natural language processing and artificial intelligence techniques to accomodate a wider range of semantically equivalent articulations of commands, and to enable the system to converse with the user to produce personalized results.

References

[1] A Periodic Table of Visualization Methods. http://www.visual-literacy.org/periodic_table/periodic_table.html.

[2] CUP-LALR parser generator. http://www2.cs.tum.edu/projects/cup/.

[3] JFlex-The Fast Scanner Generator for Java. http://jflex.de/.

[4] A. Abela. *Advanced Presentations by Design: Creating Communication that Drives Action*. Pfeiffer, 2008.

[5] Apple Computer Inc. *Hypercard Script Language Guide: The Hypertalk Language*. Addison Wesley Publishing Company, 1988.

[6] D. Ebert, K. Gaither, and C. Gilpin. Enabling science discoveries through visual exploration. NSF Workshop report, September 2007.

[7] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *Visualization and Computer Graphics, IEEE Transactions on*, 13(6):1137–1144, Nov.-Dec. 2007.

[8] C. R. Salama, M. Keller, and P. Kohlmann. High-level user interfaces for transfer function design with semantics. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1021–1028, 2006.

[9] L. Wilkinson. *The Grammar of Graphics*. Springer, 2005.