TableView: portable genomic data visualization

James E. Johnson\(^1\), Martina V. Stromvik\(^1\),
Kevin A.T. Silverstein\(^1, 2\), John A. Crow\(^1\), Elizabeth Shoop\(^1, 5\) and
Ernest F. Retzel\(^1, \ast\)

\(^1\)Center for Computational Genomics and Bioinformatics, University of Minnesota,
Minneapolis, MN 55455, USA and \(^2\)Department of Plant Biology, University of
Minnesota, St Paul, MN 55108, USA

Received on August 2, 2002; revised on October 21, 2002; accepted on December 16, 2002

ABSTRACT

Summary: TableView is a generalized scientific visualization program for exploration of various biological data, including EST, SAGE, microarray and annotation data. Written in Java, TableView is portable, is easily used together with other software including DBMSs and is versatile enough to be applied to any tabular data.

Availability: TableView is freely available at: http://ccgb.umn.edu/software/java/apps/TableView/

Contact: tableview@ccgb.umn.edu

INTRODUCTION

The need to integrate, visualize and mine different genomics data from growing sources is accelerating. The number of data mining tools, especially those for microarray data is also rapidly increasing. A few examples of available tools are the free Genesis (Sturn \textit{et al.}, 2002) and J-Express (Dysvik and Jonassen, 2001) for annotated microarray data or commercial tools such as SpotFire (http://www.spotfire.com/). Most data mining applications are costly, restricted to one or a few computer platforms, or they are specialized for handling data from one type of experiment. Also, using commercial software in development of new suites of software may lead to licensing and distribution issues. In order to accommodate the changing genomic data formats and database schemas without having to change the data mining and visualization tool, we developed TableView. Whether biologists are looking for patterns or genes of interest in SAGE, EST, microarray data or mining other genomics data, even in different databases, TableView can be used to do the work, easing the integration of different data types for visual exploration.

DESCRIPTION

TableView imports tabular data directly from character-delimited files, other applications or from database tables (e.g. MySQL or Oracle). The visual workspace contains several global views (windows) including a spreadsheet-like table, parallel coordinates, histograms, line graphs, hierarchical clusters, 2D and 3D scatter plots (Fig. 1). Since all the views on a data set share the same selection mechanism, selecting data points in one view is automatically reflected in the other views, showing how the selected points relate to the entire data set from each view’s perspective. Selected data can be saved in a tab delimited text file for further analysis in TableView or in other software.

APPLICATION

In Figure 1, we describe using TableView to visualize a set of soybean annotated SAGE tag data (manuscript in preparation). This customized table integrates tags and tag counts from SAGE libraries, matching EST contig identifiers, EST contig counts, scores for predicted contig direction and MetaFam (http://metafam.ahc.umn.edu; Shoop \textit{et al.}, 2001; Silverstein \textit{et al.}, 2001) protein family assignments. TableView is used to mine this aggregated table, visually selecting points of interest. As an example of TableView’s versatility we have used it as a component within other applications. An application written in JavaScript (http://www.mozilla.org/rhino/), uses TableView to select protein family assignments http://metafam.ahc.umn.edu from a custom table. The selections are subsequently examined in the MetaFam protein family database. The selected sequence, along with its MetaFam-assigned protein family is passed to a Java Server Page (JSP; http://java.sun.com/) running ClustalW (Thompson \textit{et al.}, 1994) to perform a multiple alignment displayed in Jalview (http://www2.ebi.ac.uk/embnet.news/vol5_4/embnet/jalview.html).

ARCHITECTURE

TableView uses the model-view-controller architecture of the Java Swing package (http://java.sun.com/). Table data, read from a text file or queried from an SQL-based DBMS, is accessed through a swing TableModel. Selection of data items in the TableModel is maintained in
a swing ListSelectionModel. The user may open various views of a TableModel, e.g. scatterplots, histograms, or a swing JTable. When the user selects data points in a particular view, the ListSelectionModel is altered and all the other views are notified to update their presentation of the data, reflecting the current selections. Database queries from TableView are performed using JDBC (http://java.sun.com/). The 3D scatter plots uses hardware-accelerated Java3D (http://java.sun.com/) if available. The architecture is extensible, allowing new view types to be readily plugged into the framework.

IMPLEMENTATION
TableView is distributed via Java Webstart (http://java.sun.com/), which enables easy installation and updates from a web link. TableView has been tested on the following platforms: Sun, SGI, Linux, MacOS X and Windows.

TableView is limited only by memory and CPU performance and we frequently view tables of 20000 cells. The view most limited by memory is the cluster view where a table of ca 1000 rows is a typical size. A future version of TableView will have a more efficient cluster mechanism and a 3D histogram view. Please refer to the TableView website (http://ccgb.umn.edu/software/java/apps/TableView/) for more on software architecture and a user’s guide.

ACKNOWLEDGEMENTS
The authors would like to thank Shulan Tian for helpful comments and Christina Schmidt, James Shupp and Paul Keim for the SAGE data. This work was supported in part by the National Science Foundation awards DBI-0196197, DBI-0110206, DBI-9975806 and DBI-9872565; by the USDA SCA 58-3625-8-117 funded by the North Central Soybean Research Board and the United Soybean Board, and by the USDA SCA 58-1907-0-030.

REFERENCES