BIOINFORMATICS APPLICATIONS NOTE

Vol. 21 no. 3 2005, pages 396–398
doi:10.1093/bioinformatics/bth474

Microarray data mining with visual programming

Tomaz Curk[1], Janez Demsar[1], Qikai Xu[3,4], Gregor Leban[1], Uros Petrovic[2], Ivan Bratko[1,2], Gad Shaulsky[3,4] and Blaz Zupan[1,3,*]

[1] Faculty of Computer and Information Science, University of Ljubljana and [2] Jozef Stefan Institute, Ljubljana, Slovenia, [3] Department of Molecular and Human Genetics and [4] Graduate Program in Structural and Computational Biology and Molecular Biophysics, Baylor College of Medicine, Houston, TX 77030, USA

Received on March 31, 2004; revised on June 16, 2004; accepted on August 7, 2004
Advance Access publication August 12, 2004

ABSTRACT

Summary: Visual programming offers an intuitive means of combining known analysis and visualization methods into powerful applications. The system presented here enables users who are not programmers to manage microarray and genomic data flow and to customize their analyses by combining common data analysis tools to fit their needs.

Availability: http://www.ailab.si/supp/bi-visprog

Contact: blaz.zupan@fri.uni-lj.si

Supplementary information: http://www.ailab.si/supp/bi-visprog

INTRODUCTION

Functional genomics often strives to discover relations between gene expression, structure and function, using tools from statistics, visualization and machine learning (Leung and Cavalieri, 2003). Analysis of microarray data is greatly enhanced by including additional information, such as gene annotation, and may provide new insights into the function of biological systems and processes (Troyanskaya *et al*., 2003). Many programs are available to the microarray data analyst. User-friendly programs that do not require programming skills allow users to select and inspect data using a set of predefined tools (see, for instance, http://geneontology.org for a collection of gene ontology tools, and http://ep.ebi.ac.uk/EP for gene expression tools). More powerful programs provide control over data flow and visualization, but they require substantial expertise in programming (e.g. scripting tools in R—http://bioconductor.org, or in Python—http://biopython.org). This situation limits the ability of most biologists to analyze their own data. We explored the application of visual programming to solve this problem.

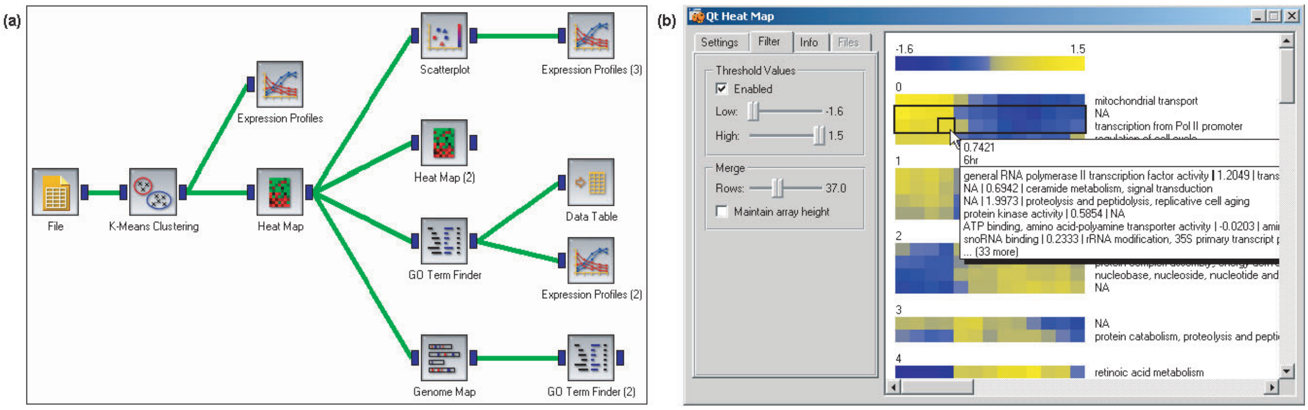
ORANGE GENOMICS WIDGETS

We have developed a visual programming environment for functional genomics data analysis. The environment uses the Orange data analysis framework, which allows users to control data flow without knowing how to program (Demsar and Zupan, 2004). The system is publicly available, modular and user friendly. Its basic data processing units are called widgets. Each widget implements a task of data manipulation, analysis, model building or visualization. The advantage of widgets is in their modularity. Widgets can be connected through channels and communicate with each other by sending and receiving data. The output of one widget is used as an input for one or several other subsequent widgets. Communication channels are typed (i.e. the data type is determined to be integer, text, table, etc.) and the system establishes the proper type of data connections automatically. This property relieves the user from the need to design data structure, which is one of the greatest obstacles for lay users. A collection of widgets and their communication channels is called a schema, which is essentially a program designed by the user for a specific data analysis task. The programming process—creating a schema with widgets and their connections—is done visually through an easy-to-use graphic interface. Schemas can be saved and compiled into executable scripts for later reuse.

We developed a set of functional genomics widgets that address microarray data analysis, gene mapping and annotation with Gene Ontology (GO). They focus on visualization and can be used in combination with other data mining widgets that are already available in Orange. For a detailed description of widgets and their data interfaces see Supplementary information.

To demonstrate the utility of the system, we used microarray data from *Dictyostelium discoideum* development (Van Driessche *et al*., 2002) and *Saccharomyces cerevisiae* cell cycle microarray data (Spellman *et al*., 1998). See Supplementary information for details on datasets and description of the two example analyses.

[*] To whom correspondence should be addressed.



Threshold Values

☑ Enabled

Low: -1.6

High: 1.5

Merge

Rows: 37.0

☐ Maintain array height

mitochondrial transport
NA
transcription from Pol II promoter

0.7421
6hr

general RNA polymerase II transcription factor activity | 1.2049 | trans
NA | 0.6942 | ceramide metabolism, signal transduction
NA | 1.9973 | proteolysis and peptidolysis, replicative cell aging
protein kinase activity | 0.5854 | NA
ATP binding, amino acid-polyamine transporter activity | -0.0203 | ami
snoRNA binding | 0.2333 | rRNA modification, 35S primary transcript p
... (33 more)

nucleobase, nucleoside, nucleotide and
NA

NA
protein catabolism, proteolysis and pepti

retinoic acid metabolism

Fig. 1. (**a**) An example of microarray data analysis schema. (**b**) ‘Heat Map’ widget. Genes from the second and third rows were selected for further analysis; gene annotation is displayed in a tooltip. Sliders are used to set the image granularity.

The schema shown in Figure 1a illustrates the utility of the new functional genomics widgets. To use a widget, the user selects it from a toolbar at the top of the screen (data not shown) and places it in a schema. The widget icon illustrates the operation or the output and its name is shown below each icon in the schema. Connections (green lines, Fig. 1a) are made by a click-and-draw mouse operation. Opening (double-clicking) a widget icon invokes a window that allows the user to vary the widget’s parameters of operation. The first widget in our schema loads the expression data (‘File’ widget) and allows the user to navigate and select data from local resources. Clusters of gene expressions (‘K-Means Clustering’ widget) are sent to the ‘Expression Profiles’ widget for viewing in a line graph form and to the ‘Heat Map’ widget for color-coded viewing of clustered gene expression patterns (Fig. 1b). The number of genes in a window often exceeds the number of pixels on the screen. The ‘Heat Map’ window allows the user to determine how many genes should be merged into a single row for a compact view (Fig. 1b).

The combination of widgets in a schema is quite flexible so users may generate any desired data flow simply by connecting widgets in the desired order. An interesting example in Figure 1a is the combination of ‘Heat Map’ and ‘Heat Map (2)’ widgets, which provides a magnifying glass effect. The selected subset of genes from the first map can be visualized in the second map at a finer granularity, resulting in an enlarged image. The magnifying glass was not pre-programmed in the schema; it is a result of innovative combination of widgets by the user.

Widgets allow users to focus their attention on a selected subset of data and to switch rapidly between data subsets. In the ‘Heat Map’ window, the user selected two rows of genes (Fig. 1b). All subsequent analyses, such as the ‘GO Term Finder’ and ‘Genome Mapping’, are done on the selected subset (Fig. 1a) and the user can select which ones to view in separate windows by double-clicking the desired widgets icons (data not shown). Selecting other rows in Figure 1b would replace the information content of all subsequent widgets.

The ‘GO Term Finder’ widget discovers significant GO terms associated with the input genes and displays gene annotation. The user can select genes based on their annotation and further process the data. The ‘Genome Map’ widget is used to display chromosomal locations of selected genes. That widget also allows selection of genes according to chromosomal location and, in the example, the data are sent to the ‘GO Term Finder (2)’ to discover significantly common GO terms of proximal genes.

Visual programming is a well-developed concept in computer science. Our system uses this powerful approach for microarray data analysis and visualization, allowing biologists to explore their microarray data without any knowledge of programming. The software runs on MS Windows, and the versions for Linux and Mac OS X are under development. We are also developing widgets to handle statistical testing.

ACKNOWLEDGEMENTS

This work was supported in part by a grant from the Slovene Ministry of Education, Science and Sports and by a grant from the National Institute of Child Health and Human Development, P01 HD39691.

REFERENCES

Demsar,J. and Zupan,B. (2004) Orange: from experimental machine learning to interactive data mining. White Paper (www.ailab.si/orange), Faculty of Computer and Information Science, University of Ljubljana, Slovenia.

Leung,Y.F. and Cavalieri,D. (2003) Fundamentals of cDNA microarray data analysis. *Trends Genet.*, **19**, 649–659.

Spellman,P.T., Sherlock,G., Zhang,M.Q., Iyer,V.R., Anders,K., Eisen,M.B., Brown,P.O., Botstein,D. and Futcher,B. (1998)

Comprehensive identification of cell cycle-regulated genes of the yeast *Saccharomyces cerevisiae* by microarray hybridization. *Mol. Biol. Cell*, **9**, 3273–3297.

Troyanskaya,O.G., Dolinski,K., Owen,A.B., Altman,R.B. and Botstein,D. (2003) A Bayesian framework for combining heterogeneous data sources for gene function prediction (in *Saccharomyces cerevisiae*). *Proc. Natl Acad. Sci., USA*, **100**, 8348–8353.

Van Driessche,N., Shaw,C., Katoh,M., Morio,T., Sucgang,R., Ibarra,M., Kuwayama,H., Saito,T., Urushihara,H., Maeda,M. *et al.* (2002) A transcriptional profile of multicellular development in *Dictyostelium discoideum. Development*, **129**, 1543–1552.