G-language Genome Analysis Environment: a workbench for nucleotide sequence data mining

K. Arakawa, K. Mori, K. Ikeda, T. Matsuzaki, Y. Kobayashi and M. Tomita*

Institute for Advanced Biosciences, Keio University, Fujisawa, 252-8520, Japan

Received on July 11, 2002; revised on August 25, 2002; accepted on September 3, 2002

ABSTRACT

Summary: G-language Genome Analysis Environment (G-language GAE) is an open source generic software package aimed for higher efficiency in bioinformatics analysis. G-language GAE has an interface as a set of Perl libraries for software development, and a graphical user interface for easy manipulation. Both Windows and Linux versions are available.

Availability: From http://www.g-language.org/ under GNU General Public License. CD-ROMs are distributed freely in major conferences.

Contact: info@g-language.org

The short but grand history of bioinformatics has clarified the fact that it must gain higher efficiency in order to process the huge masses of information that it faces. Analyses in bioinformatics often require software creation or computation with programming, which undergo redundancy of efforts from time to time (Stein, 2002). We aim to solve this task by:

1. Constructing an integrated environment for the development of analysis software.
2. Systematically accumulating existing analysis software, methodologies for analysis and their results.
3. Constructing generic analysis packages that allow users to avoid redundancy in the process of analysis.

Accessible from an intuitive graphical user interface or as a single Perl module (Stein, 2001), G-language Genome Analysis Environment (G-language GAE) provides more than 40 bioinformatics analysis programs (for the list and details of each programs, refer to http://www.g-language.org/documentation.html). To name a few, the package includes programs for genome mapping, sequence alignment, gene expression prediction, codon usage analysis, calculation of consensus and free energy of nucleotide sequences, markov analysis, overlapping gene analysis, search for tandem repeats and palindrome sequence structures, graphical display of the genome, and analysis of strand bias of bacteria. The software system can load, base the analyses, and output the results using most common genome database formats such as GenBank, Fasta, EMBL, Swiss, SCF, PIR. Access to the variety of genome database formats was made possible by embedding bioperl (http://www.bioperl.org/) modules for parsers and retrieval of databases from the Internet. G-language GAE contains a specialized parser for the GenBank database format (Benson et al., 2002) enabling high performance to load the genome data. The core module also provides a native access to the R statistics language (http://www.r-project.org), enabling complex statistical analyses. Instance of bioperl can also be used directly instead of the above flat file databases, thus the analysis programs can be directly called from a script based on bioperl or other ‘Bio*’ projects. Users of bioperl can easily gain the extension of the development of graphical user interfaces and the variety of analysis programs with our system.

In the graphical user interface, the analysis programs can be selected, configured with detailed options, and connected to perform a cascade of analyses by simple mouse operations. Output of the analyses is directly displayed visually in graphs and images for instant understanding and observation of the results, but by setting the options, it can be changed to output the data to a file in a format fit for spreadsheet applications. The graphical user interface is specific for the cascade of analyses using the analysis programs in G-language GAE, and the protocol is written in an original format named ‘GCF’. Current package includes GCF files for the Bacteria Analysis System that performs a cascade of analyses fit for bacterial genomes, and the cDNA Analysis System that performs a cascade of analyses on multiple cDNA sequence sets. Using GCF, a cascade of analyses can be directly configured and performed just by switching the subject genome databases. Users can also write their own short GCF scripts to extend the protocols. Moreover, Perl scripts of the cascade of analysis that is configured from the graphical user interface can

*To whom correspondence should be addressed.

Bioinformatics 19(2) © Oxford University Press 2003; all rights reserved.
be generated with a single mouse click. Editing this Perl script enables more flexible analysis for experienced programmers.

Used as a Perl module, G-language GAE provides all analysis programs and manipulation methods for the genome databases as native functions of Perl. For example, a Perl statement of ‘$gb = new G('E.coli.gbk');’ will load the database and output the basic nucleotide content information of the input genome, in this case, *Escherichia coli* (Blattner et al., 1997). Additional statement of ‘`gcskew($gb);’ will output a graph showing the GC skew (Lobry, 1996) of the genome. As this example shows, G-language GAE is simple enough for a novice in programming, and it is effective and powerful enough for experienced programmers.

Using G-language GAE, the time and cost spent in bioinformatics analyses are greatly reduced, because an analysis can be quickly finished with the graphical user interface, or it can be built upon a Perl script automatically generated. Plenty of application programming interface (API) for graphing, for input and output formatting, and for graphical user interface enable easy creation of graphical applications. Analyses on G-language GAE are also format independent. All database format taken in as an instance of G-language GAE is stored as a uniform format in memory, and all results of the analysis functions can be used as Perl objects and variables as well as file output. This realizes a cascade of analyses even with standalone software such as BLAST (Altschul et al., 1997), ClustalW (Thompson et al., 1994), and E-CELL (Tomita et al., 1999) by creating analysis functions using the above API. Therefore, G-language GAE is an integrated environment for software development that can provide uniform interfaces for existing analysis software and methodologies.

The whole system is developed using Perl programming language, and because of the object oriented structure of G-language GAE, an extension of the Perl programming language suited as a workbench for bioinformaticians is realized. G-language GAE is independent of operation systems as a Perl module and as a graphical application developed with the cross-platform API, wxWindows (http://www.wxwindows.org/). Windows and Linux packages are available as well as the source code under the GNU General Public License. The software package has undergone more than nine months of intensive testing as public beta release since October 2001, and we have been distributing the stable version since July 2002. All reported bugs are fixed, and several suggested analysis programs are added during the testing period. Examples of researches analyzed using G-language GAE include genome mapping, comparative study of overlapping genes, prediction of genetic disorders, analyses on homologous recombination, gene expression, horizontal gene transfer, translation initiation, comprehensive analysis of the cDNAs of *Oryza sativa*. In our future work, we hope to speed up the system and gain more flexibility by incorporating XML and relational database access, and to implement analysis methods that are useful for experimental data and proteome as well as further extension of existing genome analysis programs.

ACKNOWLEDGEMENTS

Many of the ideas presented in this paper were inspired by discussions with Hirosada Mori, Yoshihide Hayashizaki, Akio Kanai, Tomoya Baba, Yasuhiro Naito, Yoichi Nakayama, Takanori Washio, Rintaro Saito, Takeshi Ara, Kouichi Takahashi, and Fumihiko Miyoshi. We would like to thank the members of G-Language Project at the Institute for Advanced Biosciences, Keio University, including Haruo Suzuki, Ryo Hattori, Seira Nakamura, Daisuke Kyuma, Koyuki Munakata, Yohei Yamada, Hiromi Komai, Kenji Higashi, and Misa Nakanishi for their support during this work, and The Open Lab (http://bioinformatics.org/) for hosting our project.

This work is supported in part by a grant from the Ministry of Agriculture, Forestry and Fisheries of Japan (Rice Genome Project SY-1104), and a grant from Japan Science and Technology Agency (JST).

REFERENCES


