GeneQuiz: a workbench for sequence analysis

Michael Scharf, Reinhard Schneider, Georg Casari, Peer Bork,
Alfonso Valencia, Christos Ouzounis, Chris Sander
Protein Design Group
European Molecular Biology Laboratory
Meyerhofstrasse 1
Postfach 10.2209
D-69012 Heidelberg
Germany
Europe
surname@embl-heidelberg.de

Abstract

We present the prototype of a software system, called GeneQuiz, for large-scale biological sequence analysis. The system was designed to meet the needs that arise in computational sequence analysis and our past experience with the analysis of 171 protein sequences of yeast chromosome III. We explain the cognitive challenges associated with this particular research activity and present our model of the sequence analysis process. The prototype system consists of two parts: (i) the database update and search system (driven by *perl* programs and *rdb*, a simple relational database engine also written in perl) and (ii) the visualization and browsing system (developed under *C++/ET++*). The principal design requirement for the first part was the complete automation of all repetitive actions: database updates, efficient sequence similarity searches and sampling of results in a uniform fashion. The user is then presented with "hit-lists" that summarize the results from heterogeneous database searches. The expert's primary task now simply becomes the further analysis of the candidate entries, where the problem is to extract adequate information about functional characteristics of the query protein rapidly. This second task is tremendously accelerated by a simple combination of the heterogeneous output into uniform relational tables and the provision of browsing mechanisms that give access to database records, sequence entries and alignment views. Indexing of molecular sequence databases provides fast retrieval of individual entries with the use of unique identifiers as well as browsing through databases using pre-existing cross-references. The presentation here covers an overview of the architecture of the system prototype and our experiences on its applicability in sequence analysis. The utility of Genequiz has been already proven during the analysis of 331 protein sequences from yeast chromosome XI and a quarter of the *Mycoplasma capricolum* genome, containing 314 proteins. Further developments will allow active guidance of the user by a rule-based system. Also, dependencies will be minimized so that the system can be made publicly available.

Introduction

The goals of large-scale sequence analysis

With genome projects producing vast amounts of sequence data, there is a growing need for faster and more reliable methods of sequence analysis. The technical challenges are two-fold: first, how to identify sequence similarities in molecular databases efficiently without sacrificing sensitivity and second, how to integrate existing software and databases and document the findings of experts in a multi-user interactive environment (Bork et al., 1992).

Need for automatic tools

Large-scale sequence analysis differs from traditional practices in two basic respects: first, computational efficiency using fast algorithms and certain heuristics is essential (Altschul et al., 1990); second, knowledge support for expert users is becoming crucial, as the emerging gene and protein families from genome projects extend beyond the areas of expertise of a single individual (Bork et al., 1992). Therefore, the development of systems performing the necessary analytical steps for a large number of sequences as well as providing access to molecular and bibliographic databases is required.

Prime tasks: speed and reliability

With a few hundred – or even thousand protein sequences [represented by coding DNA sequences called open reading frames (ORFs)] to be analyzed as efficiently as possible, large-scale sequence analysis calls for two fundamental requirements: first, rapid database searches and a preliminary abstraction of the output (which do not constitute the limiting factor - see Results section) and second, further annotation and evaluation of results (where participation of human experts is vital, yet error-prone).

Prediction of protein function by sequence homology

The questions in computational genome analysis differ significantly from a traditional sequence analysis project. Here, the most compelling question is the

identification of homologies in search of a function (Bork et al., 1994a). However, the issue of function prediction for proteins is partly a problem of definition. We can define function prediction as any evidence towards the identification of various protein sequence characteristics indicative of substrate recognition and catalysis, interactions, localization and evolutionary relationships. Therefore, the characterization of a protein sequence (or an ORF coding for a protein) usually takes place at various levels of accuracy, from a simple calculation of coiled-coil forming potential to the derivation of a three-dimensional model, on the basis of homology to a well-characterized protein family (Bork et al., 1994a). With large amounts of data and limited time, the genome informatics expert seeks solutions to the problem of function prediction.

Methods

Phase I: Batch mode for updates and indexing (infrequent)

Use of various sequence databases. Parallel to the explosion of data production from genome projects, various databases have been created to accommodate the needs of specialized scientific communities. The generation of these databases is done locally, and computer networks with appropriate information retrieval systems may provide access (Bork et al., 1994a). The exponential growth of these databases mandates frequent local updates, sometimes even during the analysis process. It has been repeatedly proven that most recent database releases often contain newly deposited sequences that clarify evolutionary relationships and facilitate function prediction by homology (Koonin et al., 1994). These changes complicate the analysis, since an incremental search should be performed often.

Automated database updates. The system uses InterNet only for database updates, using ftp access to remote sites. A module has been developed in *perl* (Wall & Schwartz, 1990) which connects at remote sites and transfers the appropriate database if a most recent creation date appears in the remote disk system. The sequence databases currently used are: PDB, Swiss-Prot, PirOnly, GenPept (for protein) and EMBL, Genbank, dbEST (for DNA). These databases have different formats and are not cross-referenced adequately. We will see below how we make use of available cross-references only, without generating new inter-database references. Our solution has been found satisfactory by expert users. Finally, although there is an overlap between these collections, we have not addressed the issue of a non-redundant database yet.

Database indexing for fast retrieval. We have implemented the indexing of databases to provide fast retrieval of sequence database entries for two purposes: first, browsing of particular entries and instant access to database records, allowing the user to further examine the literature and the available documentation online (Fig. 1); and second, extraction of sequence data for further analyses (e.g., multiple sequence alignment and iterative profile searches). The idea of cross-referencing databases has been explored before in systems such as SRS (Etzold & Argos, 1993). However, our implementation is unique in detail: cross-referencing is exploited only when present in database records. We underline the fact that we do not generate new cross-references.

Phase II: Batch mode for searches and summary (frequent)

A simple model for sequence database searches. To accelerate a first scanning of all databases in the most efficient way, a hierarchical model for database searches was implemented. For each query sequence, a new directory is created, and all search and analysis program output files are stored there. First, searching with the fastest available tool (*blast* suite of programs (Altschul et al., 1990)) allows the verification and exclusion of those cases where a clear homology and a possible function can be readily documented. Next, searches with *fasta* (Pearson & Lipman, 1988), a slower but still widely used, and at times more sensitive, program, are performed. Sequence databases are searched in a serial mode: first the protein databases with the best documentation (such as Swiss-Prot), next other protein databases (such as GenPept) and finally DNA databases if no homologies are found in the former. Searching DNA databases offers three additional assets: detection of previously unidentified ORFs, detection of sequencing errors and detection of most recently deposited entries not yet present in protein databases. The price is significantly increased computing time (a three-fold increase). The database search control program also allows the distribution of jobs in a cluster of workstations for a speed-up of the searching process.

Sequence analysis programs. Except for obvious and well-known cases, other characteristics of newly sequenced ORFs are of interest, especially when function by homology cannot be predicted. For example, coiled-coil regions, transmembrane segments, or previously described sequence patterns can be of extreme importance for further understanding of protein function. Such analyses are always performed, irrespectively of whether an homology is found. However, especially for cases where no relatives are found, hints for function may come from this end (Bork et al., 1994a). The computing time for these analyses is negligible. In addition to the standard analyses, we use filters for shorter and more meaningful output lists, multiple alignments, cluster analysis, secondary structure prediction and views of multidimensional sequence space.

Parsing of search results. Various programs provide a wide range of output formats, usually a compromise between machine and human readability. The

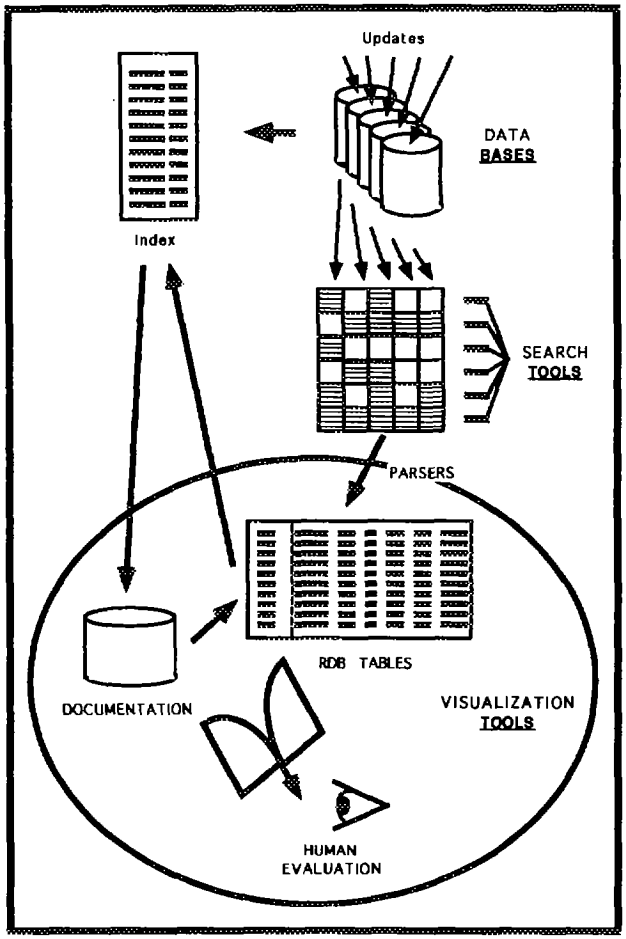

Figure 1 An abstract model of the GeneQuiz system – Various sequence databases are searched by a number of search tools. The output is a large number of files with very different format, content and information. After parsing the output, relational tables are generated, where the lists of hits are merged. Inspection of the lists by human experts, supported by the documentation in sequence and bibliographic databases (via indexing) results in the final judgment and evaluation of homologyVarious sequence databases are searched by a number of search tools. The output is a large number of files with very different format, content and information. After parsing the output, relational tables are generated, where the lists of hits are merged. Inspection of the lists by the human experts, supported by the documentation in sequence and bibliographic databases (via indexing) results in the final judgment and evaluation of homology.

lack of syntax and a standard has necessitated the implementation of parsers for the database search output, under the view that these searches are rarely the final result of an analysis process and that in the future other programs may be used. In that respect, the system is independent of the search software (Fig. 1). Parsers for *blast* and *fasta* have been implemented. For the visualization of results, including sequence alignments, search scores, database documentation and comments, the simple *rdb* format was adopted (Hobbs, 1993). *Rdb* is a simple yet powerful database system written in *perl*, it is highly portable, and can work on intermediate data for later transfer into commercial RDBMS such as INGRES (Hobbs, 1993).

A relational database schema. Organizing principles for the storage and the manipulation of results are necessary elements in this effort, since sequence database searches and other analyses provide us with a large amount of essentially unprocessed data. We decided to use a well-developed formalism in database design, the relational database model. The result parsers produce entity tables that are directly readable by a simple relational database (*rdb*). The visualization component reads the relational database tables and provides various views (Fig. 1, Fig. 2). Merging, sorting, counting and other familiar operations can also be performed.

Merging of results. The two major search methods employed here produce results evaluated by different criteria (both reflecting sequence similarity). In addition, various database searches with a single method also produce lists of database entries that have to be merged to produce a single list of entries (Fig. 1). The latter problem is easier to solve since merging of tables on a unique attribute is readily performed by the database system. The conversion of the search criteria to a common length-dependent criterion has been used (Sander & Schneider, 1991). Using *rdb* queries from command line or within the system, the user can perform complex queries, e.g. "obtain all entries that share more than 50% sequence identity to my query sequence, come from *E. coli* and are longer than 100 amino acids, and then sort them by score and print their names only".

Phase III: Interactive mode for browsing and evaluation

Browsing through structured output. The evaluation of ambiguous sequence similarities and the prediction of function is a complex and iterative process. Instead of coding a vast amount of biological and technical knowledge into the system, we decided to simply transfer the responsibility to experts at present, through the use of a flexible and programmable software component (developed in *C++* (Stroustrup, 1991)/*ET++* (Weinand et al., 1988)). This browser receives a list of database entries in the

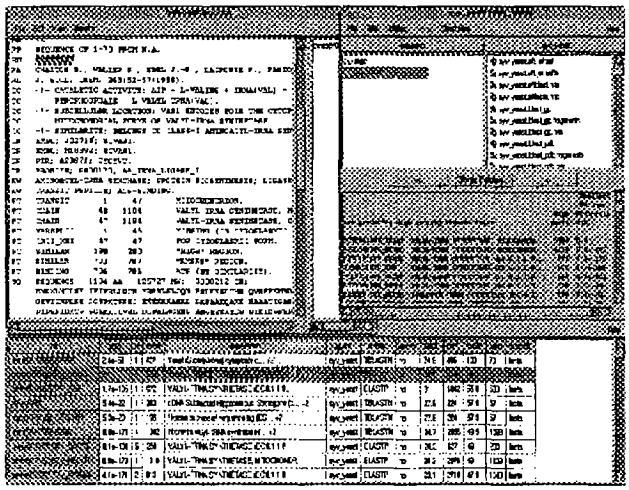

Figure 2 The visualization component of GeneQuiz – Top right, the directory manager: list of files, text files can be viewed in the lower part of the window. Bottom, the merged list: entries from a variety of databases are sorted on a distance measure (Sander & Schneider, 1991); the blackened row can now be browsed (both as database documentation and alignment). Top left, the browser: a database entry is shown, with documentation about this particular protein; cross-references can be used to check other documentation.

form of *rdb* tables that contains the most likely candidate homologues (which can indeed be a very heterogeneous list of genes and proteins) (Fig. 2). The visualization component allows browsing through alignments, scores and documentation available in sequence and bibliographic databases based on the previously mentioned indexing of databases (Fig. 2). The system allows interactive selection of the most interesting cases and produces entry names for other purposes (such as multiple sequence alignment and profile construction and searches).

Evaluation based on additional information. In some cases, currently estimated in less than 50%, further study beyond single-sequence analysis is pursued, in order to demonstrate putative sequence homologies and remote evolutionary relationships (Bork et al., 1992, Bork et al., 1994a). The current implementation allows the automated construction of multiple alignments and the corresponding sequence profile, for facilitation of further sensitive database searches, when necessary. An iterative mode of searching and the identification of more homologues with such methods is often the final stage for automated sequence analysis.

Overview of system components

The automated database updates have been implemented in *perl*. The database indexing has also been implemented using *perl* and shells. The homology searches are driven by a *perl* program, which runs *blast* and *fasta* against the sequence databases as well as activates a number of other programs. The program allows a large number of options that specify use of certain programs and databases. The parsers have also been written in *perl* and output the data in *rdb* database format (Fig. 1). *Perl* is a language that supports data processing, system administration and interprocess communications (Wall & Schwartz, 1990). Thus, it is ideally suited for the above tasks. The visualization component has been implemented in *C++* using *ET++* applications software. The system can be started at the parent directory of the runs, providing an overall view of the analysis (Fig. 2). Provisions were made such that *perl* programs will be incorporated in the browser, therefore converting it to a more 'active', programmable tool for sequence analysis.

Results

Acceleration of sequence analysis

We have clearly accelerated the sequence analysis process by importing and installing public domain sequence databases of highest relevance to genome informatics, a necessary basis for reliable and up-to-date results. In addition, the evaluation of sequence similarities is done in this semi-automatic fashion, where results are parsed and partially processed. At the final stage, the human experts are presented with lists of candidate homologues and the most difficult cases are treated interactively, without the need to code and store biochemical knowledge in our system. All results are organized under a relational database system, which allows error checking and instant flexible retrieval of single entries and database statistics. In cases where profile and other sensitive database searches provide further hints for function prediction, an iterative procedure is usually followed.

Importance of latest database releases

As shown previously, the mere increase of sequence databases allows the identification of a significant proportion of previously unknown cases, without dramatic technological developments (Koonin et al., 1994). We think that this is an important aspect of the whole process, where synchronization of updates and searches provide the optimum possible efficiency in function prediction by homology. The unfortunate lag of derived protein sequence databases with respect to the primary DNA sequence databases necessitates the additional search of the latter, with significantly higher computational cost.

Flexibility of design

The design principles behind the current implementation allow ease in re-design and re-organization. In the discussion, we explain how processing steps will be merged with the visualization and browsing tools,

creating a programmable framework for sequence analysis, using *perl* as a script language. Results will be imported transparently to the users for evaluation.

Performance gain

Justifying the design and implementation of such a complex collection of databases, programs and their interactions, would in principle require the quantitative demonstration of this improvement, both in terms of speed of execution and reliability of results. Although such a demonstration is difficult without a modeled control experiment, we feel that significant progress has been made in terms of accelerating large-scale sequence analysis by a factor of two to five. We discuss how this statement is warranted, based on real-world examples of sequence analysis from two genome projects.

Analysis of yeast chromosome XI

Our whole effort was first geared towards an efficient analysis of the ORFs from the second fully sequenced eukaryotic chromosome and the largest continuous piece of DNA sequence known today, the chromosome XI from *Saccharomyces cerevisiae* (baker's yeast) (666 Kbases). The number of ORFs is approx. 330, that required around two weeks CPU time in a cluster of workstations, including a preliminary evaluation of results. Most effort was spent in the detailed analysis of subtler cases and the verification of these results, with additional database searches and multiple alignments (Dujon et al., 1994).

Timing considerations

The specifications, design and implementation of software for database updates, sequence searches and visualization took about three months, for 3-4 persons working full-time. We estimate that the total amount of work for a functional prototype amounted to 9 person-months. Updates (incl. indexing) usually take one to two days. For chromosome XI, the runs took around three days on four workstations. The analysis of the data and the preparation of reports took another four weeks for three persons (about ten person-weeks). The amount of time for the non-interactive part is negligible and there is probably very little space for significant technological improvements. On the other hand, the bottleneck appears to be the analysis of complex cases (Dujon et al., 1994).

Analysis of the *Mycoplasma capricolum* genome

In addition, during another collaborative project, we acquired a large part of the *Mycoplasma capricolum* genome (234 Kbases) by Pat Gillevet (George Mason University). We estimated the presence of 314 ORFs, longer than 100 amino acids or identifiable by similarity searches (Bork et al., 1994b). Again, in this case, the database searches using our system took a negligible amount of time, in comparison to the time needed for full documentation. The use of GeneQuiz facilitated our tasks and provided summaries through flexible database queries (e.g., "how many ORFs have sequence identity higher than 60% to any protein and are longer than 300 residues?"). Browsing of database records also proved extremely useful for a rapid establishment of homology.

Formalization of human expertise

In addition to the (at least) two-fold acceleration and the increased reliability of the task using GeneQuiz, we believe that the design and implementation of such systems, aided by sequence analysis experts, also helps in the formalization and understanding of this activity. In the foreseeable future, this and similar developing frameworks will eventually result in systems capable of reproducing human action in this particular domain of scientific knowledge. Below, we discuss the possibility of extending our prototype system using rule-based control and coding of domain knowledge in an intelligent database rather than a pseudo-algorithmic control program.

Discussion

Future enhancements of the prototype

Enhancements of the current implementation are clearly needed. First, better merging procedures can be adopted, better views on the summary lists, and a more interactive mode for selection and further analysis. Second, some of the *perl* programs must be incorporated into the browser, making it a more uniform, single-component system, from the user's point of view. Third, care should be taken about a multi-user environment, with file-locking and security checking capabilities (an issue not addressed in a small group). Finally, the system should become more flexible by being programmable, and the user should be able to incorporate personally developed scripts into a database of 'task modules'.

Towards an expert system in sequence analysis

With sequence analysis becoming an important activity of international research efforts for the delineation and understanding of structural, functional and evolutionary relationships in model organisms, the goal will be to design and implement complex computing systems that replace human subjects in the most repetitive and error-prone aspects of this task (e.g., interpretation of high sequence similarity). However, the real challenge is to encode the knowledge of experts so that all their tasks are performed automatically. Such expert systems should have explanatory capabilities for the justification of actions they perform. The problem mainly consists of coding the biological knowledge possessed by human experts that aids them

significantly in the analysis (e.g., prior biochemical information etc). Although the protein universe seems to be finite, it is a vast area of human knowledge, currently beyond the cognitive capabilities of any single human subject. Molecular databases should be able to provide us with such a knowledge, by answering complex queries on structure, function and evolution of known protein families. However, this is not possible at present, partly because of the inadequate or totally absent mechanisms for biological knowledge representation (Karp & Riley, 1993). Once biological knowledge is held in the database of such anticipated expert systems, the problem would be to provide lists of rules that a human expert actually follows. We are far from such objectives. However, the basis has been laid out, with the design and construction of simple automated systems that link database and software for large-scale sequence analysis in genome projects.

Sequence analysis: from art to science

Our whole effort has two goals: the proximate goal is the facilitation of large-scale sequence analysis, with maximal time and resource efficiency, increased reliability and relief of experts from repetitive aspects of the whole process. The ultimate goal is the transformation of the field of sequence analysis from a skill acquired through continual use of current technologies to a set of standardized and generally accepted procedures based on scientific results obtained through basic research.

Acknowledgments. Help by Miguel Andrade and Rob Hooft (EMBL) during the preparation of the manuscript is greatly acknowledged.

Appendix. Specifications: C. Ouzounis, A. Valencia, P. Bork, C. Sander Design: M. Scharf, G. Casari, R. Schneider, C. Sander Implementation: M. Scharf (indexing, visualization and browsing) G. Casari (*perl* scripts and refinement cycles) C. Ouzounis (databases and *rdb*) R. Schneider (updates and database searches) Users: P. Bork, C. Ouzounis, C. Sander, A. Valencia Management: C. Ouzounis Coordination: C. Sander

References

Altschul, S. F., Gish, W., Miller, W., Myers, E. W. & Lipman, D. J. 1990. Basic local alignment search tool. *J. Mol. Biol.* 215: 403–410.

Bork, P., Ouzounis, C. & Sander, C. 1994a. From genome sequences to protein function. *Curr. Opin. Struct. Biol.* Forthcoming.

Bork, P., Ouzounis, C., Casari, G., Schneider, R., Sander, C. & Gillevet, P.M. 1994b. Exploring the *Mycoplasma capricolum* genome: a small bacterium reveals its physiology. *EMBO J.* submitted.

Bork, P., Ouzounis, C., Sander, C., Scharf, M., Schneider, R. & Sonnhammer, E. 1992. Comprehensive sequence analysis of the 182 predicted open reading frames of yeast chromosome III. *Protein Sci.* 1: 1677 1690.

Dujon, B. et al. 1994. Analysis of the 331 ORFs of yeast chromosome XI. Manuscript in preparation.

Etzold, T. & Argos, P. 1993. SRS – an indexing and retrieval tool for flat file data libraries. *Comput. Appl. Biosci.* 9: 49 57.

Hobbs, W. V. 1993. RDB: a relational database management system. RAND Corporation.

Karp, P. D. & Riley, M. 1993. Representations of metabolic knowledge. AI Center, SRI International.

Koonin, E. V., Bork, P. & Sander, C. 1994. Yeast chromosome III: new gene functions. *EMBO J.* 13: 493–503.

Pearson, W. R. & Lipman, D. J. 1988. Improved tools for biological sequence comparison. *Proc. Natl. Acad. Sci. USA* 85: 2444 2448.

Sander, C. & Schneider, R. 1991. Database of Homology-Derived Structures and the Structural Meaning of Sequence Alignment. *Proteins* 9: 56–68.

Stroustrup, B. 1991. The *C++* programming language. Addison–Wesley, Reading, MA.

Wall, L. & Schwartz, R. L. 1990. Programming *perl*. O' Reilly & Associates, Inc., Sebastopol, CA.

Weinand, A., Gamma, E. & Marty, R. 1988. In Object Oriented Programming Systems, Languages, and Applications, pp. 46 57. San Diego, CA.