FACILITIES FOR EXPLORING MOLECULAR BIOLOGY DATABASES ON THE WEB: A COMPARATIVE STUDY

Victor M. Markowitz, I-Min A. Chen, Anthony S. Kosky, and Ernest Szeto
Data Management Research and Development Group
Lawrence Berkeley National Laboratory, Berkeley, CA 94720

We discuss criteria for evaluating and comparing the main facilities provided by molecular biology databases (MBDs) for exploring (that is, retrieving and interpreting data) on the Web. We use these criteria for examining the facilities supported by typical MBDs such as Genbank, AtDB, GSDB, GDB, and MGD (as of September 5, 1996).

1 Introduction

Molecular biology databases (MBDs) are implemented using different data management systems ranging from file management systems to database management systems (DBMSs). The data in an MBD are structured according to a *schema* specified in a *data definition language* and are accessed using a *query language*, where these languages are based on a *data model* that defines the semantics of their constructs and queries. For example, the schema of Genome Sequence Data Base (GSDB) [10] and the Mouse Genome Database (MGD) [16] are defined using the data definition language of the Sybase relational DBMS, the structure of the Arabidopsis thaliana database (AtDB) [1] (as well as numerous other MBDs) is defined using ACeDB [6], and the structure of Genome Database (GDB) [8] and the 3DB version of the Protein Data Bank (PDB) [21] are defined using OPM [3] on top of Sybase. For MBDs maintained as flat-files, the data definition languages used for defining their structure are not based on a data model per se and range from generic notations such as the ASN.1 data exchange format used for Genbank [18] to ad-hoc data definition languages such as that employed for EMBL [9].

Exploring MBDs involves examining the structure (metadata) of MBDs, browsing and querying MBDs, and interpreting the results of queries expressed over MBDs. MBDs must provide facilities that would help users to identify an MBD as containing relevant data, formulate queries against the MBD, and to interpret or make use of the data retrieved. Many of these facilities amount to *documentation* and query and browsing facilities which are supported to some extent by most MBDs.

In this paper, we discuss criteria for evaluating the main facilities required for exploring MBDs on the Web, that is, retrieving and interpreting MBD metadata and data. We apply these criteria in comparing the main facilities provided by typical MBDs, such as Genbank, AtDB, GSDB, GDB, and MGD. These MBDs were chosen to provide a representative selection of MBDs accessible via the Web, and for illustrating the various levels of support provided for accessing MBDs on the Web. Space limitations do not allow us to provide a comprehensive list of MBDs that can be accessed via the Web nor to discuss their facilities in detail.

The remainder of this paper is organized as follows. In section 2 we discuss the facilities required for exploring and accessing metadata. In section 3 we examine to what extent typical MBDs meet these requirements. In section 4 we discuss criteria for evaluating MBD browsing and querying facilities. In section 5 we apply these criteria in examining the browsing and query facilities of typical MBDs. Section 6 contains concluding remarks.

2 Metadata Support

Ideally, an MBD should provide comprehensive metadata as well as facilities for easy access to and interpretation of this information. MBD metadata would preferably involve descriptions in alternative formats, such as text, Html, and notations, such as diagrams, lists. The need for comprehensive MBD metadata capturing domain knowledge about biology, was discussed extensively at the Meetings on Interconnection of Molecular Biology Databases[15] and in papers such as [11,14]. Comprehensive MBD metadata should include general information regarding MBDs, information on the structure (schemas) of MBDs, alternative MBD views, and information on relationships between MBDs.

General information describes the underlying MBD and should include (i) the MBD name, a brief description of the purpose of the MBD, the physical location and history of the MBD; (ii) information on the data definition language and implementation details for the MBD, including either precise definitions and examples for the data definition language or references to where such information can be found, information on the underlying DBMS and implementation strategy for the MBD; and (iii) keywords to allow high level searches for relevant MBDs.

MBD schemas define the MBD structure in the data definition language of the native data management system underlying the MBD, and perhaps also in alternative data definition languages. Detailed schema definition helps expert users understand certain database design issues and/or choices; it can also be reused in defining similar molecular biology applications. Moreover, schema definitions are essential for users who want to use the *ad hoc* query capabilities provided by some MBDs (see section 4 for more details).

MBD views represent alternative interpretations of the MBD. Metadata on each view should include: (i) an *overview* grouping related components together into higher-level components in order to provide a concise and comprehensible high-level description of the MBD view; and (ii) information on each major construct (class, attribute), including: (a) *semantic definitions* describing the real world or abstract concepts represented by the view construct, and possibly constraints on the values of instances of the construct that cannot be expressed in the underlying data definition language; (b) *design motivation* for the use of a particular construct in representing application data; (c) *synonyms* and *keywords* for identifying differences in terminology and for establishing potential correspondences between the components of different MBDs; and (d) *sample data* providing examples of how the view constructs are used and how typical data might appear. In addition general information on each view should be available, including explanations and motivation for the particular interpretation of the MBD provided by the view.

MBD relationships describe any known relationships (links) with other MBDs, including descriptions of external references stored in the MBD, descriptions of the semantics of the links and information on how to follow the links, and references to metadata for other MBDs.

The ease of locating and accessing MBD metadata is also important to consider. Some MBDs provide explicit metadata information on their Web pages, while others provide metadata in ftp files. The later entails a slow and tedious process of locating and downloading the files, and examining the metadata in these files in whatever format it is provided. A very useful facility is on-line *metadata browsing*, based on hyperlinks between related metadata components.

Maintaining comprehensive MBD metadata can be complex and time consuming if it is not supported by appropriate tools. For example, providing MBD metadata (e.g., schemas, views) in different data definition languages requires *schema conversion* tools; similarly, providing MBD metadata in different formats and notations can be automated by *metadata publishing* tools. Additional tools are needed for revising and/or updating metadata, such as tools for keeping track of MBD changes and for synchronizing the various MBD metadata components mentioned above.

3 Metadata Support: Case Studies

Almost all MBDs provide some metadata support though the level of this support varies greatly. We briefly describe the main metadata support provided by typical MBDs, such as GSDB, MGD, Genbank, AtDB, and GDB, and discuss the main problems of providing such support.

3.1 Case Studies

The GSDB and MGD metadata consist of documentation describing the structure of the relational databases underlying GSDB and MGD in terms of tables and columns. Understanding this documentation requires some familiarity with relational database modeling, for example, in order to understand the role of primary and foreign keys in representing cross-table relationships. No information is provided by GSDB and MGD regarding relational data modeling and its employment in designing GSDB and MGD. MGD provides on-line schema information; users can easily find the schema definition on the Web. The table definitions of the relational MGD schema as well as comments and examples for table attributes can be reviewed (one-level browsing), but cannot be browsed recursively, that is, by following links representing inter-table relationships. The GSDB documentation is available as postscript and Portable Document Format documents. GSDB does not provide on-line schema information nor schema browsing facilities.

The structure of Genbank is defined in ASN.1. ASN.1 is a flexible notation for describing nested file structures but does not constitute a data model per se since it lacks support for integrity constraints and data manipulations. The ASN.1 definition for Genbank requires familiarity with ASN.1, and is not available directly on the NCBI Web site, but can be obtained, together with an ASN.1 tutorial, by ftp. Since NCBI provides large amounts of information via ftp, extra effort is required for locating the files containing metadata. On-line schema information is not available and schema browsing is not supported by Genbank.

The structure of AtDB is defined using ACeDB, which is based on a data model with an object-oriented flavor. The AtDB documentation consists only of the ACeDB schema definition and requires familiarity with the ACeDB flavor of object-oriented constructs; ACeDB documentation is available on line. The AtDB schema is available on the Web and can be recursively browsed on-line. Furthermore, for each ACeDB class definition one can find a data sample (example) for an instance of that class.

The structure of GDB 6 is defined using OPM, a data model that combines standard object-oriented constructs (object class, attribute) with constructs for modeling scientific experiments (protocols). The structure of GDB databases is available in alternative formats, including OPM and relational formats, and notations, including Html, diagrams, and Ascii. Understanding the OPM or relational schemas for GDB requires familiarity with object-oriented or relational database concepts. A tutorial of object-oriented concepts is available on-line. Descriptive information for GDB 6 is also available for non-database expert users. GDB 6 metadata includes data samples (e.g., map examples in both database and diagrammatic form), and can be browsed on line.

Most MBDs describe relationships to related MBDs merely through links to related Web sites; the semantics of the relationships between related schema components are not provided.

3.2 Discussion

The main problem with providing metadata support is the large number of different data models and DBMSs that are used to implement MBDs, including the relational model, ASN.1, ACeDB and OPM. Understanding and using MBDs requires familiarity with different data models. OPM and ACeDB provide comprehensive metadata support. The object-oriented or type constructs of these models provide more concise and easier to understand database definitions when compared with relational database constructs. Furthermore, ACeDB and OPM schemas can be annotated with descriptions of their various elements. ASN.1 provides a formal and concise notation for data structures, but is not a data model per se and therefore does not support specifying important schema information such as integrity constraints. The relational model is oriented towards the efficient storage and management of data, but does not provide constructs for capturing the semantics of data well: the representation of a single conceptual object in a relational database may be spread over many records in several distinct tables, thus making a relational schema a poor vehicle for communicating the semantics of a database.

Providing MBD metadata in a uniform and consistent way (common data models, formats, and notations) would greatly simplify the task of gathering and interpreting metadata. Uniform metadata representation can be achieved via tools for defining semantically enhanced views for MBDs in a common data model, together with tools for automatically generating alternative representations of these views. An example of such tools are the OPM retrofitting tools that allow constructing OPM views on top of relational or ASN.1 databases[4], and the OPM publishing tools, which allow representing OPM schemas in various formats (Html, LaTeX) and notations (diagrams, lists). Some of these tools have been used for generating the GDB 6 documentation in various formats and notations as part of GDB's metadata support. However effective use of such tools first requires comprehensive metadata and documentation to be available in some form.

4 Browsing and Query Support

Browsing and query facilities supported by MBDs range from fixed-form queries to free-form or ad-hoc queries expressed using a query language.

4.1 Fixed-Form vs Free-Form Querying

Fixed-form or *canned* queries have a fixed structure involving a predetermined set of tables, classes or other database components, and a predetermined set of attributes for each database component. Fixed-form queries can be parameterized on certain values and support some options, allowing the user to control the conditions used in the query or set the values used by the conditions.

Fixed-form queries define *interfaces* through which the underlying MBDs can be examined. These interfaces may not necessarily reflect the structure of the underlying MBD. For example, the list of attributes or fields retrieved by a fixed-form query may be only a subset of the attributes and fields in the underlying MBD.

Unlike fixed-form queries, *free-form* or *ad-hoc* queries do not have a predetermined structure and are usually specified in some query language. The query language, its syntax and its expressive power, depend on the underlying data model and DBMS. Even for the same data model, different DBMSs are likely to support different query languages. For example each major relational DBMS vendor offers their own variant of SQL. Consequently free-form querying requires detailed knowledge of the data model, schema and the particular query language used by the MBD.

SQL is a powerful query language that serves as the underlying standard for the query language of commercial relational DBMSs. Specifying SQL queries, however, requires non-trivial knowledge of the structure and manipulation of relational databases. For example, one needs to understand the meaning of organizing data in tables and of joining tables in order to retrieve related data from tables.

The ODMG-93 standard[17] is followed by most object-oriented query languages, such as the OPM query language (OPM-QL), while systems such as ACeDB have their own brand of object-based query languages. Specifying object queries is simpler than specifying SQL queries mainly because an object query involves higher-level, and therefore fewer elements. For example, a typical OPM-QL query is substantially more concise (in number of its component elements) than an equivalent SQL query. However, specifying object queries still requires understanding the syntax and semantics of the query language, which may be a non-trivial task for non-expert users.

4.2 Formulating, Processing, and Interpreting Queries

The process of querying an MBD involves: (1) formulating a query, which requires understanding how data are represented in the underlying MBD; (2) processing the query, which involves parsing the query, finding an optimized execution plan and retrieving the results of the query; and (3) interpreting the results of the query.

Formulating queries could involve browsing and keyword-search of MBD metadata for identifying MBDs of interest and for determining whether these MBDs contain relevant data. It is important for an MBD to provide support in formulating queries. For fixed-form queries, for example, such support can involve providing the set of allowed attribute values (if this set is not too large), sample (example) attributes values, and help in formulating condition expressions. Free-form queries require additional support in examining the underlying schema, specifying queries involving multiple, not necessarily directly related, elements, managing the query language syntax, and so on.

Fixed-form query processing is relatively simple and can, in certain cases, be provided via information retrieval systems. Indexes can be provided for certain attributes in order to increase the query performance. Free-form query processing is substantially more complex, where the complexity depends on the power of the query language. Commercial DBMSs provide query processing (and various levels of query optimization) for their own brand of SQL. OPM query processing is provided by an OPM query translator that generates equivalent, possibly semantically optimized, SQL queries for the underlying (Sybase or Oracle) database[5].

MBD metadata together with the semantics of the operations underlying MBD query processing can be used for *interpreting* the meaning (semantics) of query results. For example, information on the semantics of objects in a given class can be used for annotating query results, for describing the searching scope of a query, for explaining null query results, for suggesting alternative queries when no results are returned, and so on.

5 Browsing and Query Support: Case Studies

We briefly discuss in this section the query facilities of MBDs such as GSDB, MGD, EMBL, Genbank, AtDB, and GDB, and examine the main problems of these facilities.

EMBL provides a simple sequence retrieval based on accession numbers as well as fixed-form queries via the SRS system [7]: SRS supports browsing and querying multiple MBDs, including EMBL, where querying is restricted to simple conditions of up to four matching expressions involving a limited number of indexes. Forms are linked to explanatory text for assisting users in constructing index matching expressions.

MGD supports fixed-form queries that represent a number of entities of interest (e.g., Probes, PCR Primers, References) which are based on, but do not necessarily reflect the structure of the underlying database. Forms are linked to information on fields and explanatory text for assisting users in constructing query conditions. Sets of allowed values or format information for certain attributes are provided in the query forms together with menu buttons for selecting comparison operators for the query conditions.

Genbank's interface, Entrez [20] is in fact the query interface for a warehouse consisting of Genbank, Medline and several sequence and macromolecular MBDs. Entrez supports a single fixed-form query for the underlying MBD and provides a mechanism of constructing conditions involving a limited number of indexed fields. Help is provided for constructing these conditions. Entrez provides information on the number of data items that may satisfy a given condition, thus assisting users in determining whether the condition is too restrictive or too general (e.g., more than 1000 data items satisfy the condition). Such information allows calibrating conditions.

GDB's Web interface supports only fixed-form queries. Concise queries involving only few fields (e.g., accession number, genome name) support simple searches. More detailed query forms are provided for each class in the underlying databases, where each form reflects the structure of the class. Lists of valid values are provided for attributes associated with controlled vocabularies so that selection conditions for these attributes can be specified easily. Forms are linked to schema information and explanatory text for assisting users in constructing query conditions.

GSDB provides a limited number of fixed-form queries, including a concise 'Sequence and Feature Retrieval' form for retrieval based on accession numbers or sequence names. Sample values are provided for the fields in the query forms, but no other help is provided for constructing or customizing queries. GSDB also supports the specification of free-form queries expressed in the Sybase variant of SQL. Sample SQL queries are provided for main GSDB objects such as genes and products. As mentioned in the previous section, specifying new or customizing existing SQL queries requires non-trivial knowledge of SQL as well as understanding the structure and semantics of the underlying relational database, and therefore is beyond the ability of most casual users.

AtDB can be queried by using both fixed-form and free-form queries supported by ACeDB. The fixed-form queries reflect the structure of AtDB: each ACeDB class underlies a *query-by-example* interface for that class. AtDB supports a different strategy to help users formulating queries. If a query condition is too general (i.e., more than 100 data items satisfy the condition), then the system automatically suggests "sub-selection" conditions for reducing the number of the returned items. Free-form queries in AtDB are expressed in ACeDB's query language, which is a restricted and rather peculiar language: using it requires knowledge of its syntax and semantics and therefore it is not straightforward. AtDB provides on-line schema browsing and a link to ACeDB query language documentation to help users formulating free-form ACeDB queries. However, anecdotal evidence suggests that free-form ACeDB queries are seldom used by ACeDB users.

5.2 Discussion

Fixed-form queries are limited in that they must conform to some predetermined view of the underlying MBD. This limitation can be offset by mechanisms supporting multiple MBD views: that is multiple schemas for a single MBD whose instances are either materialized or computed at run time from the data in the underlying MBD. For example, OPM's support for defining new derived classes allows rapidly extending GDB's query interface by defining new derived classes.

Query interfaces may need to be changed whenever the underlying MBD changes or new features need to be supported. In such cases, schema driven query interfaces, that is, interfaces that are automatically generated from the underlying MBD metadata are easier to extend than manually maintained interfaces. For example, GDB's Web-based query interfaces are easy to maintain since they are automatically generated from the underlying schemas by the Genera tool[12]. We are not aware of similar facilities for other MBD query interfaces.

Since free-form querying requires detailed knowledge of the query language, data model, and MBD structure, some MBDs supporting free-form queries, such as GSDB, try to address these problems by providing sample queries or templates: rather than writing a query from scratch users can adapt or alter the sample queries to suit their needs. However this is only a partial solution: if the query a user wishes to ask diverges significantly from the sample queries provided then significant knowledge of the query language, schema and data model are still required. Consequently, for free-form query facilities to be of use, the query languages and schemas provided must be concise and intuitive, and comprehensive on-line documentation for both must be provided. We are not aware of any MBD that provides such facilities.

Often MBDs do not provide sufficient information for clarifying the semantics of queries specified by users or to interpret the semantics of the query results. For example the Entrez interface provides a number of query forms and various choices of attributes on which to search, but does not offer any description of the extents over which these queries search, or the semantics of the individual attributes. We do not feel that any known MBD supports the level of query interpretation we described in the previous section. These problems are compounded by the extremely large and complex molecular biology nomenclature and by various differences in interpretations of this nomenclature within the molecular biology community.

6 Concluding Remarks

There is a significant disparity between the levels of support provided for on-line or Web access to various MBDs. Supporting MBD access on the Web requires providing comprehensive metadata and documentation on line. Ideally, such metadata should include schemas in a variety of commonly used data definition languages, semantic descriptions of MBDs and their components, alternative views of MBDs, and sample data. Such metadata should be easy to access and explore, using tools such as Web-based schema browsers.

Many MBDs provide connections to other MBDs via hyperlinks, often embedded in the results of a query. Interpreting and correctly using such links requires documentation describing the semantics of the links, as well as documentation on the linked MBDs. In practice, such documentation is seldom available.

The question of the best way of providing Web-based query facilities for MBDs remains open. A good option is to support fixed-form queries for the most common queries, combined with support for ad hoc queries for flexible data exploration and non-standard queries. Both fixed-form and ad hoc query facilities require comprehensive on-line documentation. In addition, ad hoc query facilities require support for specifying and interpreting queries. It is not clear how best to compromise between expressibility and simplicity/conciseness of a Web-based ad hoc query tool.

Reasons for the lack of comprehensive metadata for some MBDs include the fact that the task of making such metadata available is very labour intensive, and that schemas are revised frequently. Software tools, based on various data models such as OPM and ACeDB, have been shown to make this task substantially easier, and provide a high level of on-line support automatically. A consistent way of documenting MBDs could also facilitate MBD exploration.

The natural extension of exploring individual MBDs is exploring multiple, related, MBDs. Exploring data across multiple MBDs involves coping with the additional problems entailed by the distribution of data among MBDs, the heterogeneity of the systems underlying these MBDs, and the semantic (schema representation) heterogeneity of these MBDs. However, the requirements and facilities for exploring data from individual MBDs apply equally well to exploring data from multiple MBDs.

Acknowledgments

This work was supported by the Office of Health and Environmental Research Program of the Office of Energy Research, U.S. Department of Energy under Contract DE-AC03-76SF00098.

References

1. Arabidopsis thaliana Database, School of Medicine, Department of Generics, Stanford University, August 1996; http://genome-www.stanford.edu/Arabidopsis/; see also http://genome-www.stanford.edu/cgi-bin/model/AtDB.
2. Buneman, P., Davidson, S., Hart, K., Overton, C., and Wong, L. A Data Transformation System for Biological Data Sources. In *Proc. of the 21st Int. Conference on Very Large Data Bases*, pp. 158-169, 1995. See also http://agave.humgen.upenn.edu/cpl/cplhome.html.
3. Chen, I.A., and Markowitz, V.M. An Overview of the Object-Protocol Model (OPM) and OPM Data Management Tools. *Information Systems*, 20(5), pp. 393-418, 1995.
4. Chen, I.A., and Markowitz, V.M., Constructing and Maintaining Scientific Database Views, Technical Report LBL-38359, 1996.
5. Chen, I.A., Kosky, A., Markowitz, V.M., and Szeto, E., The OPM Query Language and Translator, Technical Report LBL-33706, 1996. Available at http://gizmo.lbl.gov/opm.html.
6. Durbin, R., and Thierry-Mieg, J., Syntactic Definitions for the ACEDB Data Base Manager, 1992; available together with additional documentation at http://probe.nalusda.gov:8000/acedocs/.
7. Etzold, T., and Argos, P. SRS, An Indexing and Retrieval Tools for Flat File Data Libraries. *Computer Applications of Biosciences*, **9**, 1, pp.

49-57, 1993. See also http://www.embl-heidelberg.de/srs/srsc.

8. Fasman, K.H., Letovsky, S.I., Cottingham, R.W., and Kingsbury, D. T. Improvements to the GDB Human Genome Data Base. Nucleic Acids Research, Vol. 24, No. 1, pp. 57-63, 1996. See also http://gdbgeneral.gdb.org/gdb/schema.html.
9. The EMBL Nucleotide Sequence Database, European Bioinformatics Institute, http://www.ebi.ac.uk/ebi_docs/embl_db/ebi/topembl.html.
10. Genome Sequence DataBase (GSDB) 1.0. The National Center for Genome Resources, August 1996; http://www.ncgr.org/gsdb/gsdb.html.
11. Karp, P., A Strategy for Database Interoperation, Journal of Computational Biology, Vol 2, No 4, 1995.
12. Letovsky, S., Genera: A Specification-Driven Web/Database Gateway Tool, http://gdbdoc.gdb.org/letovsky/wgen.html.
13. Markowitz, V.M., Chen, I.A., and Kosky, A., Exploring Heterogeneous Molecular Biology Databases in the Context of the Object-Protocol Model. *Theoretical and Computational Genome Research*, (S. Suhai, ed.), Plenum. In press. See also Chen, I.A., Kosky, A., Markowitz, V.M., and Szeto, E., OPM*QS: The Object-Protocol Model Multidatabase Query System, Technical Report LBL-38181, 1995, http://gizmo.lbl.gov/opm.html.
14. Markowitz, V.M., and Ritter, O., Characterizing Heterogeneous Molecular Biology Database Systems, *Journal of Computational Biology*, Vol 2, No 4, 1995.
15. Meeting on Interconnection of Molecular Biology Databases (MIMBD): http://www.sri.ai.com/people/pkarp/mimbd.html.
16. Mouse Genome Database 3.1, The Jackson Laboratory, February 1996; http://www.informatics.jax.org/mgd.html.
17. *The Object Database Standard: ODMG-93*, Cattell, R. G. G. (ed), Morgan Kaufmann, 1994.
18. Programmer's Reference. National Center for Biotechnology Information, Bethesda, Maryland, November 1991.
19. Ritter, O. The Integrated Genomic Database. In *Computational Methods in Genome Research* (S. Suhai, ed.), pp. 57-73, Plenum, 1994. See also http://genome.dkfz-heidelberg.de:80/igd/start_igd_doc.html.
20. Shuler, G.D., Epstein, J.A., Ohkawa, H., Kans, J.A. Entrez. In *Methods in Enzymology*, (R. Doolittle, ed.). Academic Press, Inc. In press. See also http://www3.ncbi.nlm.nih.gov/Entrez/.
21. 3DB, the Protein Data Bank, Brookhaven National Laboratory, 1996; http://terminator.pdb.bnl.gov:4148/