SIRW: a web server for the Simple Indexing and Retrieval System that combines sequence motif searches with keyword searches

Chenna Ramu*

European Molecular Biology Laboratory, Postfach 10.2209, 69012 Heidelberg, Germany

Received February 14, 2003; Revised and Accepted March 26, 2003

ABSTRACT

SIRW (http://sirw.embl.de/) is a World Wide Web interface to the Simple Indexing and Retrieval System (SIR) that is capable of parsing and indexing various flat file databases. In addition it provides a framework for doing sequence analysis (e.g. motif pattern searches) for selected biological sequences through keyword search. SIRW is an ideal tool for the bioinformatics community for searching as well as analyzing biological sequences of interest.

INTRODUCTION

More and more specialized biological databases are being created newly or secondarily derived from existing large databases with additional information added manually or automatically. This is partly because of the hundreds of databases containing thousands of entries serving only a general need. Although most of the sequence databases are widely available, the usage is often general and limited to keyword searching and entry retrieval, through systems like Sequence Retrieval System (1) and Entrez (2). Studying the subset of the database that represents a particular taxonomy, organs or tissue is time consuming and involves creation of such datasets before applying analysis tools on the dataset (e.g. constructing evolutionary history, inferring protein function).

Most often sequence retrieval tools and sequence analysis tools are separated. Although sequence analysis systems exist such as GCG (3) that provide integration of sequence analysis with keyword search, in practice the keyword search is often restricted to one or few data fields. Since databases such as SWISS-PROT (4) are well annotated, retrieving entries with specific keywords through various fields is enough to pull out all the interested protein entries: for example ‘zinc finger’ containing proteins for a specific species or taxonomic range. In addition it would be interesting to analyze for additional sequence signals for those selected sets of protein sequences.

SIR (5) is now capable of indexing any of the database fields of interest, including the biological sequence itself, a feature lacking in most retrieval tools, indexing the biological sequence such as protein sequences found to be very useful. For example, searching for N- or C-terminal sequence becomes trivial. In addition more complex regular expression queries in search of protein motifs become possible.

Apart from providing such flexible keyword retrieval from various database entry fields, SIRW extends its capability by a flexible framework to analyze them. We show this here with GCGs (3) FINDPATTERN equivalent pattern matching module.

THE SIR SYSTEM

SIR is entirely written in Python (http://www.python.org/), the object-oriented scripting language. The SIR system consists of a database definition file where you define the various files that make up the database, the various fields of the database entry, the field parsing instruction and the token parsing instruction for each field of interest for indexing. The index tokens can be of string type or number type. The number type is useful for searching for the range, for example retrieving proteins of certain residue length or molecular weight.

Database entry and fields

Flat file databases are provided with different structures. A flexible parsing system for flat files has been developed (6) that can deal with different formats along with the modules to manipulate sequence objects.

The database and entry iterator

The generic database module can be used to sequentially iterate over the database and database entries, providing entries for the field and token parsers to build indices, as well as for indexed retrieval of one or more entries. The design of the dual-purpose database module reduces the complexity of handling the database during the indexing phase and the retrieval phase by abstracting the parsing and indexing instructions. We also found sequential iteration is useful in building custom flat file databases for further use, for example creating a local database to use with BLAST (7) with desired field tokens such as specific, organism or description.

*Tel: +49 6221387530; Fax: +49 6221387517; Email: chenna@embl.de


© Oxford University Press 2003; all rights reserved
Entry, field and token parsing

Entry delimiters are used for the entry parsing, for example SWISS-PROT (4) and EMBL (8) entries are always delimited with '/n'. An entry consists of different data fields (e.g. 'Definition', 'Keywords', 'Organism'). The field parsing is done with a regular expression that describes a particular field. For example the definition field of a SWISS-PROT entry is

\[ \text{des}: r'(\text{DE } [^/][\text{n}] / [^+/][\text{n}]+)' \]

The parsing instruction consists of two regular expressions enclosed in a tuple, the first instruction extracts the description field, whereas the second instruction is used to extract the tokens from the description field. Commands can be used within the regular expression description for the tokens that should be skipped (command skiptoken) and the tokens to be printed while parsing (command print), which is useful for debugging, for example, while designing a token parser. The parsing instructions are handled by a generic scanner module which takes 'action' based on the commands.

Indexing the database fields

The purpose of indexing is to make the retrieval faster. The indexing system we use here makes use of the bplustree module (http://starship.python.net/crew/watters/bplustree). Indexing the primary key is already explained in SIR (5). Indexing various other fields of interest is now possible with SIRW. The parsing instructions are taken from the database definition file (Fig. 1). The primary index key should be unique. For some databases this is different from the identifier name, for example for the RefSeq (9) database, the accession number is unique but the LOCUS is not. Within the token parser the command 'key' is used by the indexing system to identify the primary token. Separate index files are created for each data field.

Query and retrieval

The syntax of a typical query would be

\[ \text{find('swissprot', 'tax', 'eukaryota') & find('swissprot', 'ft_key', 'zn_fing')} \]

Each query results in a set of entries. A set of entries can be further refined through the Boolean operators 'and', 'not', 'or' (&, , |). Here there are two queries, one searches the SWISS-PROT database taxonomic field for eukaryota and the second query searches for zinc finger in the feature table entry and the result is combined with & operator that filters the result.

THE WEB INTERFACE SIRW

The web interface is designed with flexibility and simplicity for both the user and the administrator. We used cgimodel (10), a Common Gateway Interface (CGI) framework, for designing web interfaces. The top page lists all the available databases under SIRW with hypertext link to each database. The hypertext links will take the user to the database information page where a table is generated with all the fields listed along

Figure 1. Shows the parsing instructions and database definition for PDBFINDER database. For a different database of similar structure, this module can be imported instead of rewriting a new one.
system can be told to build a selection box or checkbox for that particular field, thus increasing the chances of finding a keyword in the subsequent search. The index browsing will be useful to decide which fields should be filled for the selection.

Query result page

The query result page shows the number of hits found in the header and the query syntax that was constructed based on the keyword selection. Subsequently the primary identifiers are shown with hypertext links. Clicking the hypertext links results in retrieving the full entry. Apart from the primary identifiers, the fields that are selected in the query page are also shown. If the queried field is also selected for display then the keywords are highlighted wherever they occur in that field (Fig. 3).

Index browsing

Index browsing is an important feature of SIRW. Very often the user wants to check the number of proteins that have a particular keyword (e.g. transmembrane). When the sequence is also indexed, it becomes an ideal tool for examining signature peptides or to calculate the peptide probabilities by taking the number of peptide occurrences in the whole database as expected frequency.

SEQUENCE ANALYSIS WITH SIRW

Though data collection is made easy through systems such as SRS (1) and Entrez (http://www.ncbi.nlm.nih.gov/Entrez/), there is a need for further analysis such as searching for a motif pattern in the selected entries. We demonstrate here searching for motifs in protein sequences that is done after querying the database of interest.

Searching for motifs

Conserved motifs in amino acid sequences may reveal regions that play important roles in the function of a protein. There are specialized databases such as PROSITE (14) that record biologically significant sites, patterns and profiles, and our recently developed ELM database (15) which specializes in short functional motifs in eukaryotic proteins. SWISS-PROT (4) entries have extensive feature table annotation such as zinc finger, DNA binding, calcium binding and signal peptide. Typically one may wish to find out additional signals for, say in DNA binding proteins in different organisms. An interesting example is the sequence ‘AHVDCPGHA’ in Haemophilus influenzae, which is part of the elongation factor family of proteins found to occur upstream of the casein kinase II phosphorylation signature motif (16). The pattern-matching module is plugged into the SIRW system and in the database definition file the method is specified. This specification enables the SIRW system to create an additional field called ‘pattern’ in the query page. Thus a pattern field becomes a virtual field, a data-field that is not there with the original database entry fields. A subsequence/motif can be entered along with other keywords. The system retrieves the entries and their protein sequence and a pattern search is performed on the selected protein sequences. The result is shown in the query result page. The matching pattern is shown with defined
flanking region, and with position numbers where they start and end. A compact view of the patterns can be used to align those regions to create consensus patterns. This would allow creation of an alignment that can be used to create a consensus pattern. The pattern matching is an example, we could potentially plug-in a number of other modules like dipeptide calculation or protein property search like transmembrane segments, antigenic regions. The motif patterns are expressed as POSIX style regular expressions.

LIMITATIONS OF SIRW

Since the SIRW package is written in a scripting language, it does not scale up very well for very large databases such as EMBL and GenBank, though the subsections of them (human, vertebrate) can be used with SIRW.

CONCLUSION

SIRW addresses and fulfils a need in the bioinformatics community namely doing analysis on the selected set of entries based on keywords. As far as we know there is no other system that exists with this capability. SIRW is very useful for motif hunters and database curators. SIRW is ideal for the databases as big as SWISS-PROT (4) or TrEMBL (17). The indexing of protein sequences with fixed, overlapping subsequence is possible and it is currently done for the RefSeq (9) protein database. This would enable the study and analysis of proteome databases for the distribution of signature peptides. For this case index browsing would reveal the abundance of a particular peptide.

ACKNOWLEDGEMENTS

I would like to thank Toby Gibson for his support and Francesca Diella for her comments and careful reading of the manuscript. I also thank the unknown referees for their critical comments.

REFERENCES