A Classification of Tasks in Bioinformatics

Robert Stevens[1,2], Carole Goble[1], Patricia Baker[3] and Andy Brass[2]
Department of Computer Science[1] & School of biological Sciences[2]
the University of Manchester
Oxford Road
Manchester
UK
M13 9PL
Sagitus Solutions Limited[3]
Manchester Incubator Building,
Grafton Street,
Manchester M13 9XX
Email: p.baker@sagitussolutions.com
Web: www.sagitussolutions.com

July 26, 2000

Abstract

Motivation: This paper reports on a survey of bioinformatics tasks currently undertaken by working biologists. The aim was to find the range of tasks that need to be supported and the components needed to do this in a general query system. This enabled a set of evaluation criteria to be used to assess both the biology and mechanical nature of general query systems. **Results:** A classification of the biological content of the tasks gathered offers a check-list for those tasks (and their specialisations) that should be offered in a general bioinformatics query system. This semantic analysis was contrasted with a syntactic analysis that revealed the small number of components required to describe all bioinformatics questions. Both the range of biological tasks and syntactic task components can be seen to provide a set of bioinformatics requirements for general query systems. These requirements were used to evaluate two bioinformatics query systems. **Contact:** robert.stevens@cs.man.ac.uk. **Supplementary information:** the questionnaire , responses and their summaries may be found at http://img.cs.man.ac.uk/tambis/questionnaire/bio-queries.html.

Keywords: Bioinformatics tasks; query survey; user requirements; evaluation; semantics and syntax.

1 Introduction

A knowledge of user requirements is an essential part of the software design process [Ould, 1990]. The discipline of human computer interaction (HCI) exists to research and develop methodologies to gather user requirements, incorporate them into the software design process and then check that the product matches the requirements and is both useful and
usable [Beyer and Holtzblatt, 1998, Dix et al., 1998]. In this paper we investigate the biological nature and syntactic structure of questions asked and the tasks commonly performed in the field of bioinformatics. From this, a set of evaluation principles for bioinformatics query systems were derived. Such principles will be useful to both designers and evaluators of general bioinformatics querying tools – when designing query based systems it is essential to know what range of queries to offer and the mechanisms needed for their support. These evaluation principles allow both of these aspects to be assessed in bioinformatics query applications.

A questionnaire survey of biologists, in academia and industry, was undertaken to gain a representative set of queries and tasks. A classification of the biological content of these tasks gave the scope of biological topics or semantics that should be covered by a general query system (Section 2). The syntactic or mechanical nature of the biological tasks can be derived from these data. This reveals the common processes or components in bioinformatics tasks, showing what happens in these tasks but not how they are performed. These common components describe, at a high-level, all bioinformatics tasks in terms of filters of, and transformations on, data (Section 3).

The two aspects of this analysis afford a means of assessing bioinformatics tools that offer a general query service over the many sources and tools. In particular, the evaluation principles were demonstrated by their application to two popular bioinformatics query tools (Section 4). The findings of this work are discussed in Section 5.

2 Query Classification

A general questionnaire was developed to assess the bioinformatics knowledge and usage of the community. In this paper, only the data pertinent to questions asked by the biology community are reported. A complete set of answers collected may be found at the questionnaire web site. The relevant questions were:

1. What tasks do you most commonly perform?
2. What tasks do you commonly perform, that should be easy, but you feel are too difficult?
3. What questions do you commonly ask of information sources and analysis tools?

4. What questions would you like to be able to ask, given that appropriate sources and tools existed, that may not currently exist?

Questions one and three were placed at distant parts of the questionnaire, within different contexts, in order to obtain as wide as possible a collection of tasks. Question two was used to find which tasks could be either improved or needed to be addressed. Question four was asked to extend the data collected from question one. The questionnaire was constructed according to standard guidelines [Fowler, 1984, Hoinville et al., 1978] and distributed to both academic and industry based biologists, in equal proportions. All were working biologists, rather than bioinformatics specialists, that is, bioinformatics was not their only pursuit. The responses from both the questionnaire and interviews were collected and duplicates due to the repeated questions removed. This gave a total of 315 biological tasks. In order to counter the lack of detail from responses, a small number of person to person interviews were undertaken. Of the 35 respondents, five were selected, simply by availability, for full interview. The questions gathered by this means are marked with a 'i' next to the question number in the summary results.

The responses were placed into broad categories and a set of task summaries were derived, that represented the range of tasks presented in the responses. Table 1 shows the broad classification of responses and the number of occurrences in each category. The classification was based upon a mix of biological task and the bioinformatics used to perform the task. So, multiple sequence alignment is seen as a task, irrespective of the biological purpose to which it is put (pattern finding, for instance). Similarly, searching for non-coding regions of DNA was set as a biological task, separate from similarity searching etc., purely because it had a large number of responses and thus merited a classification of its own.

Three tasks accounted for 54% of reported tasks. These were similarity search, multiple pattern and functional motif search and sequence retrieval. These cover the basic tasks of obtaining a sequence, finding what exists that is similar and what patterns are present that might indicate sequence function. These tasks are subsumed into 'what is the function of my sequence'. Multiple sequence alignment and DNA analysis (Gene finding, restriction mapping, etc.) forms the next largest grouping. These tasks, particularly multiple sequence alignment, are basic techniques for analysing and manipulating sequences.

The responses from the questionnaire primarily reported what appear to be single task queries. In addition, little detail was reported on how sophisticated the users were in their use of the sources. For example, the overwhelming majority of the responses in the similarity search simply stated 'similarity search' or described which type of sequence was searched. There were reports of restrictions placed on the search, such as from a certain species or other sequence collection, or at a certain level of identity. Other tasks were similarly undetailed, but the task summaries show there were a variety of specialisations requested by users.

Other observations from respondents included requests for more sophisticated means to view results and frustration in interoperating between databases. This seamless moving of data between information sources and analysis tools is of great importance when building more complex queries. Though often stated as single tasks, many of the tasks described do, in fact, involve more than one step (collecting sequences for an alignment, before proceeding with primer design or phylogenetic analysis). The

request to view results, especially those intermediate results of multi-source tasks, and have flexible, meaningful results display is obviously important.

The only major gap observed in the biology covered by the responses was in the usage of genomic data. This specific area was covered in questions 18 – 20 of the questionnaire. Of those that answered the questions about use and expected use of genome information, (10%) felt genome data had not had a great impact, but expected it to do so in the future. The most frequent expected use was for searching for sequence homologues and for identification of cloned fragments. Only a few, industry based, biologists expected to use genomic data for its own sake: For inter-genome comparison; gene cluster analysis and genome evolution.

Within these tasks are those that respondents wanted to be able to ask, given that the relevant technology and data were available. An interesting aspect of some of the replies was that the tasks requested are already possible. For example, predicting transmembrane regions. Other information, such as finding if a gene is essential, or exploring gene expression data is a case of having publically available, computationally accessible data. This indicates a lack of knowledge about what can currently be achieved by bioinformatics.

Many of the requests for future tasks were to group together already possible tasks to run in parallel. A typical example was: for a given DNA sample perform the following analyses: Perform similarity search; find presence and location of exons; identify repeats; identify GpC islands, and translation of sequence at specific stop/start conditions. Others asked for multiple current tasks to be performed in series. For example:

> "Identify homologues of a sequence; of these pull out either *n* closest or sequences specified; align them, giving various output options; put into various phylogenetic/dodistic packages. The results from these analyses could go forward into other analyses, depending on results."

This sort of task emphasises the need to be able to interoperate between collecting sequences, analysing those sequences and viewing intermediate results to determine subsequent routes through an analysis.

3 Structure of Queries

The biology or semantics of tasks tells us 'what is wanted' and the corresponding syntax tells us 'how to do it'. This syntax should be able to describe the structure or mechanism of any query. Such a description has two uses: first, it describes what components a query system should have to fully satisfy biologists requirements and second, it can be used to describe a benchmark set of query templates or patterns that occur in biological queries. Information from the questionnaire and interviews yields information on the requirements for such a syntax. The syntax must:

- Have the components appropriate to answer all the queries, together with specialisations, described in Section 2;
- allow automatic interoperation between parts of a query without necessitating user intervention;

- allow, but not demand, user intervention for reviewing intermediate results;
- perform intermediate format transformation etc. that are a necessary part of interoperation.

Such task components will only describe tasks at a high-level. They do not describe how a particular component works, for example a functional motif search, but simply state that there is a generic component that performs this sort of bioinformatics task.

All task components can be regarded as processes that take a collection of objects and return a collection of objects. A collection is either a set, list or bag of objects.[1] The behaviour of the process may be modified by the setting of parameters. These processes can interoperate, i.e., they can be joined together to perform larger, more complex tasks. So, a collection produced by one process can, given the appropriate semantics, act as an input collection to another process. These collections of data objects are either transformed or filtered into collections of either newly transformed collections of data objects or restricted collections of the same data objects respectively. These components may also contain mechanisms for describing criteria such as 'at least *n* results' and boolean operations. The basic components of the syntax are:

Collections – collections of data objects (Figure 1(a)). Data collections have a number of properties: they may be empty, their contents can be viewed and items removed by the user and their contents can act as input to another collection-handling component.

filters – (Figure 1(b)) take three inputs: a restriction collection (e.g., key-words, accession number, species, author, ...), a target source to filter (e.g. the database to be searched and a projection that describes the contents of each output object (e.g. which output fields do you want to see returned).

transformers – (Figure 1(c)) take one input collection, transform the objects in those collections according to the process described and produce one output collection of transformed objects. An optional collection of parameters can also be an input, that influence the operation of the transformation. Examples of the use of transformers would include format conversion, multiple sequence alignment, phylogenetic analysis, primer design, ORF analysis, DNA translation and sequence assembly.

transformer-filter – (Figure 1(d)) some of the queries are, at our level of analysis, a composition of a filter and a transformer, used in either order. This component takes three inputs, as does a filter and produces one output collection. The output collection is a filtered sub-collection of the target source, but also transformed in some way by the process described. This component was raised to the same level as the filter and the transformer, rather than using it compositionally. This was so that tasks, such as similarity searching, could be represented as one component. It is still possible, however, to represent such tasks compositionally;

[1] A set is unordered, with no redundancy; a bag is unordered, but may be redundant; and a list is ordered and may be redundant.

Forks – allow concurrency to take place. There are *conditional* (Figure 1(e)) and *unconditional* (Figure 1(f)) forks. In conditional forks, the process arising from a tine only takes place if some condition attached to the prior tine fails. In an unconditional fork, all processes attached to tines are initiated simultaneously. The results from separate tines can be either gathered together into super-collections, using a reversed unconditional fork, or proceed independently to other processes.

All the tasks presented in Table 1 can be described succinctly in this syntax. Notice that more than one task can be represented with the same structure - many bioinformatics tasks have the same syntactic structure, whilst having very different semantics. Many of the tasks can be adequately described with one filter, transformer or transformer-filter, as the presence of restriction and projection capabilities can fully describe tasks. An example of a common pattern is phylogenetic analysis (Figure 2), which can be represented using a filter followed by two transformations.

A task such as gathering together several DNA analysis tools can be represented as an unconditional-fork (see Figure 3). None of the tasks depend on the results of another, so all can be run simultaneously. The differing performance of the tools would be accommodated in the implementation of the components. Should the results be gathered by a reversed unconditional fork, the overall process would be limited by the slowest component. This concurrency of either transforming or transform-filtering is a commonly requested pattern by users.

The transformer-filter representation of a similarity search can become more detailed by indicating how the input and output collections are generated and manipulated (Figure 4).

4 Evaluation Principles and Survey of Tools

The observations of both the semantic and syntactic nature of bioinformatics queries can be used to give a set of design principles for a general query system:

1. It should cover the range of biological tasks shown in Table 1;
2. It should allow the full range of options for input, target and constraints indicated by users and addressed in Section 2;
3. User defined collections and results of previous queries should be allowed as input to subsequent tasks;
4. Components within the system should be able to be represented as interoperating collections, filters, transformers and transformer-filter;
5. Components should be included that allow forking of processes, both in a conditional and automatically concurrent manner.

These principles do not, however, evaluate whether individual tools or components perform their jobs, but simply whether the necessary components are present within a system.

Table 2 scores two general bioinformatics query tools for their compliance with these principles. Like many evaluation principles it is not always clear cut as to whether there is compliance with the principle. In this high-level evaluation a 'weight of evidence' approach was used. The aim of the evaluation is to ascertain whether the tools offer the semantic and syntactic flexibility needed in a general bioinformatics query tool. The sequence retrieval service[2] (SRS) [Etzold et al., 1996] is a general query system for flat-file databanks and analysis tools. Entrez [Schuler et al., 1996] offers query facilities over a set of biological data repositories. Hence, both are reasonable targets for assessment using these evaluation principles. Other systems, such as Imagene [Medigue et al., 1999] and GCG [Devereux et al., 1984] would also make suitable targets for such evaluations. Individual tools, such as BLAST [Altschul et al., 1997], could also be evaluated by these criteria, as long as the semantic component were relaxed.

SRS is usually presented through an HTML form-based interface (for example, see `http://srs.ebi.ac.uk`). This interface hides the construction of tasks in the SRS query language. The indices existing over the flat-files in SRS allow tasks to be phrased over most attributes within a particular database. In addition, queries can be phrased against groups of databanks and an extensive system of cross-links allows the results of one sub-task to be used to get a collection's counterparts in another database (e.g., collecting a set of proteins by function and automatically finding those with known structure from PDB).

Results may also be stored in variables, to be inspected either before immediate re-use or short-term storage for use in a future task. During the inspection of intermediate results, individual results within may be removed. On the publically available SRS servers, it is not possible to use results as long-term storage for later input, nor to create personal databanks as a part of the general query system. It is, however, possible to create such databanks on the fly. These data collections perform simple format conversions, but do not map terms appropriately between databases. The issue of semantic heterogeneity within biology databases is large and difficult to resolve [Davidson et al., 1995]. Data collections can only be passed on to subsequent tasks serially. The SRS system does not allow concurrent tasks to be performed, either automatically or conditionally. To do this, it is probable that a scripting language will be needed. Some commercial systems offer such a device, but these have not been evaluated.

Irrespective of any issues in the usability of the SRS interface, SRS comes closest to being a general query system that fulfils all the principles laid out in this paper. The WWW service includes access to the SRS query language and many of the transformation and filter-transformer tools (for example, BLAST and Prosite pattern searches), that make the SRS system a general bioinformatics query tool. The large number of biological information sources available in most installations of SRS, with the query facilities and a user interface that allows common analysis tools to be used, ensure that the majority of the biological tasks described may be carried out. Inspection, intervention, query and submission database description are also supported. Individual installations of SRS can be tailored to include resources needed at that site, thus extending the biological scope.

[2] A commercial product of Lion Biosciences AG.

Entrez is a system that links sequence data and keyword searches into the sequence, genomic and protein structure databanks, population studies and the MEDLINE bibliographic databank. It is available through the NCBI web portal (`http://www.ncbi.nlm.nih.gov`), that gathers many resources, such as BLAST OMIM and Pubmed, together with Entrez. Taken as a whole, this portal affords a wide biological scope, but Entrez itself is limited to what might be called the 'core' of biological resources.

In Entrez, query expressions can be built by hand or using a form based interface. A system of limits, indexes and display facilities allows attributes within component databases to be filtered and projected using complex boolean expressions. Having retrieved entries, Entrez supplies links to related entries, called neighbours, by both sequence and bibliographic similarity. These links are pre-computed via similarity searches in the case of sequence data and computed through information retrieval techniques for bibliographical data [Wilbur and Yang, 1996]. The use of these information retrieval techniques allows Entrez to perform conceptual transformations in computing neighbours of entries in results, a facility not available in SRS. Entries have many other links to resources such as mutations, structure and disease. Limited numbers of results can be stored, but not for re-use in queries. the Entrez history facility, however, does allow re-use of previous query results in new queries.

By giving wide ranging access to sequence and bibliographical data Entrez satisfies some, but by no means all, of the basic biological tasks described above. Entrez can be thought of as offering a data warehouse of the core bioinformatics data resources, with full filtering and projection facilities. Unlike SRS, it is not possible to add new databanks. In addition, the SRS WWW interface offers access to some of the transformation tools used in analysis, whereas Entrez has some of these features built into its data. Entrez essentially only provides the filtering components of the syntax described above and, but does include the notions of collections of data acting as input and output to such filters.

5 Discussion

This survey of biological tasks asked by users and the structure of those tasks has sought to provide a basic set of user requirements for developers of general bioinformatics applications. This work was undertaken to provide user requirements for the TAMBIS (Transparent Access to Multiple bioinformatics Information Sources) system [Baker et al., 1998, Stevens et al., 2000]. The requirements described informed what topics should be covered by the system [Baker et al., 1999] and what functionality to offer in the current, prototype and future versions. As TAMBIS is only a prototype system, and we know that many of the requirements are not yet met, it was not reviewed in the prior section. These principles have and will, however, guide the development of the TAMBIS system.

The range of biological tasks emphasised the overwhelming reliance on a small set of tools to perform most tasks, but also indicated the wide range of lesser tasks and specialisations that need to be supported.

A syntactic view of the same tasks revealed that a query system can be described in terms of filters, transformers and transformer-filters, forks and collections of data.

These components can be composed to describe all of the biological tasks, many of them with equivalent structures.

A related technology of importance, and widely used in industry, is workflow [Lawrence, 1997]. Originally developed for co-ordinating documents in business, workflow management has been extended and adopted by the scientific sector, for example the LabBase System at MIT [Stein et al., 1995]. A number of commercial tools exist (for example, InTempo, CSE Workflow, MQSeries Workflow) as does an extensive research literature. See [Fischer, 2000] for a recent series of commercial case studies, including one drawn from the pharmaceutical industry.

Workflow is a set of methods and technologies, which support a business process through the analysis, redesign and automation of information-based distributed activities, usually in the context of distributed information. Workflow is about capturing an entire process, including its rules - for example: individual roles, routing paths, priorities, schedules, and access levels. Using workflow systems, an organisation is able to automatically co-ordinate the sharing, management and routing of 'process knowledge' between applications and people. Typically workflow management systems have specification languages, dynamic resource management schemes, distributed transaction processing, support a range of interfaces to databases etc, and include updating information resources as well as retrieval.

There are four key concepts in workflow – the process, matching human resources to tasks, matching information resources to tasks, and process management. A process is a co-ordinated set of tasks (manual or automated) that are connected in order to achieve a common goal. Each task typically uses a particular application resource. These concepts are coupled with three philosophies that must be captured: what flows, who (process or person) does it flow to, and how does it flow.

Our work has identified the retrieval and analysis tasks commonly performed by biologists, and how these are combined to form higher level processes such as phylogenetic analysis. The tasks have been identified at both a parameterisable common "syntactic" level (filters, transformers etc) and a biological 'semantic' level. The issues of what flows and how it flows have been explored, as have the interactions of the biologists and applications. Thus the work here is a high-level specification of a workflow that could be encoded in a workflow specification language and enacted by a workflow management system.

there was a strong indication from users that the inability to interoperate between tools was a barrier to asking more complex questions. Such a view is supported by others [Davidson et al., 1995, Department of Energy, 1993]. The structural principles set forth in this paper seek to address the basic requirements of interoperating systems, but without describing how it should be implemented. These principles are based on the requirements that users have of such systems. the application to two commonly used bioinformatics tools demonstrates how weakness, as perceived by users, can be exposed in such systems. The biological and structural principles together form a basic set of user requirements for bioinformatics applications.

Acknowledgements: This work is funded by AstraZeneca Pharmaceuticals and the BBSRC/EPSRC Bioinformatics programme, whose support we are pleased to acknowledge.

References

[Altschul et al., 1997] Altschul, S. F., Thomas, L., Madden, A., Schaffer, A., Zhang, J., Zhang, Z., Miller, W., and Lipman, David, J. (1997). Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic Acids Res.*, 25:3389–3402.

[Baker et al., 1998] Baker, P., Brass, A., Bechhofer, S., Goble, C., Paton, N., and Stevens, R. (1998). TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources. An Overview. In *Proceedings of the Sixth International Conference on Intelligent Systems for Molecular Biology*, pages 25–34. AAAI Press.

[Baker et al., 1999] Baker, P., Goble, C., Bechhofer, S., Paton, N., Stevens, R., and Brass, A. (1999). An Ontology for Bioinformatics Applications. *Bioinformatics*, 15(6):510–520.

[Beyer and Holtzblatt, 1998] Beyer, H. and Holtzblatt, k. (1998). *Contextual Design: Defining Customer Centred Systems*. Morgan Kaufmann.

[Davidson et al., 1995] Davidson, S., Overton, C., and Buneman, P. (1995). Challenges in Integrating Biological Data Sources. *Journal of Computational Biology*, 2(4):557–572.

[Department of Energy, 1993] Department of Energy (1993). DOE informatics summit meeting report. Available via `http://www.gdb.org`.

[Devereux et al., 1984] Devereux, J., Haeberli, P., and Smithies, O. (1984). A Comprehensive Set of Sequence Analysis Programs for the VAX. *Nucleic Acids Research*, 12:387–95.

[Dix et al., 1998] Dix, A., Finlay, J., Abowd, G., and Beale, R. (1998). *Human-Computer Interaction*. Prentice Hall Europe, London, second edition edition.

[Etzold et al., 1996] Etzold, T., Ulyanov, A., and Argos, P. (1996). SRS: Information Retrieval System for Molecular Biology Data Banks. *Methods in Enzymology*, 266:114–28.

[Fischer, 2000] Fischer, L. (2000). *Excellence in Practice, Volume III Innovation and Excellence in Workflow Process and Knowledge Management*.

[Fowler, 1984] Fowler, F. (1984). *Survey Research Methods*. London:Sage.

[Hoinville et al., 1978] Hoinville, G., Jowell, R., and Associates (1978). *Survey Research Practice*. London:Heinemann.

[Lawrence, 1997] Lawrence, P. (1997). *Workflow Handbook* . John Wiley and Sons. published in association with the Workflow Management Coalition.

[Medigue et al., 1999] Medigue, C., Rechenmann, F., Danchin, A., and Viari, A. (1999). Imagene: An Integrated Computer Environment for Sequence Annotation and Analysis. *Bioinformatics*, 15:2–15.

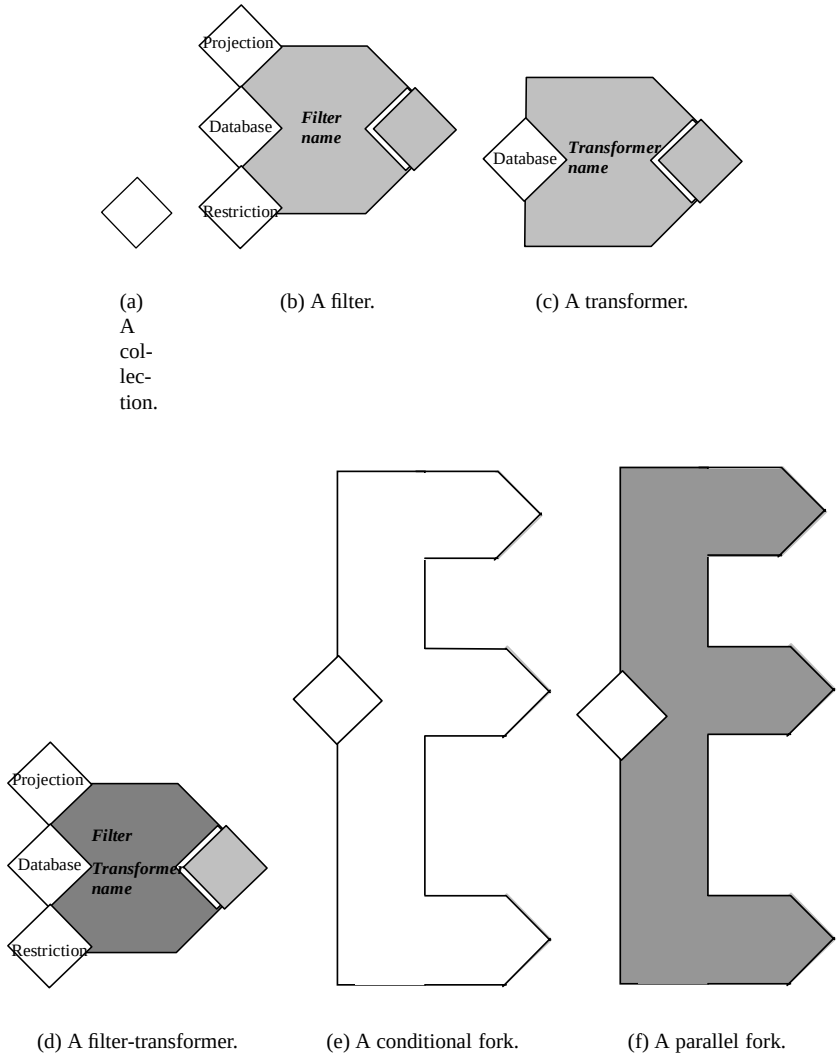
[Ould, 1990] Ould, M. (1990). *Strategies for Software Engineering : The Management of Risk and Quality*. Chichester : Wiley. (Wiley series in software engineering practice.

[Schuler et al., 1996] Schuler, G., Epstein, J., Ohkawa, H., and Kans, J. (1996). Entrez: Molecular Biology Database and Retrieval System. *Methods in Enzymology*, 266:141–162.

[Stein et al., 1995] Stein, L., Rozen, S., and Goodman, N. (1995). Managing Laboratory Workflow with LabBase. In *Proceedings of the 1994 Conference on Computers in Medicine (CompMed94)*. World Scientific Publishing Company.

[Stevens et al., 2000] Stevens, R., Baker, P., Bechhofer, S., Ng, G., Jacoby, A., Paton, N., Goble, C., and Brass, A. (2000). TAMBIS: Transparent Access to Multiple Bioinformatics Information Sources. *Bioinformatics*, 16(2):184–186.

[Wilbur and Yang, 1996] Wilbur, W. and Yang, Y. (1996). An Analysis of Statistical Term Strength and its Use in the Indexing and Retrieval of Molecular Biology Texts. *Comput Biol Med*, 26(3):209–222.



(a) A collection.

(b) A filter.

(c) A transformer.

(d) A filter-transformer.

(e) A conditional fork.

(f) A parallel fork.

Figure 1: the components of the syntax for describing bioinformatics tasks. The **collections**, **filters**, **transformers**, **filtering-transformers** are described in the text.

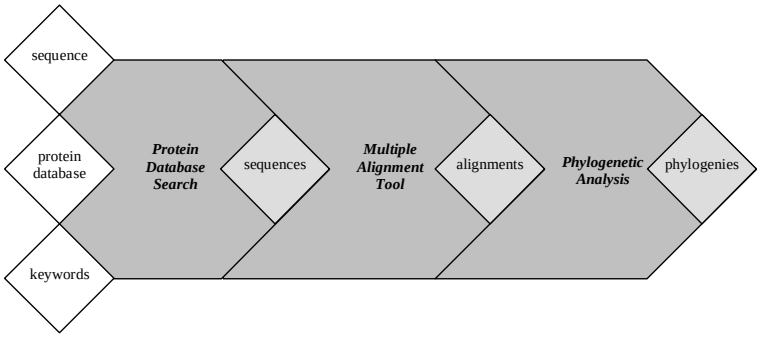


Figure 2: The common pattern of a filter, followed by two transformations. In this case, it represents a phylogenetic analysis. A filter, representing a database search, takes three inputs: A projection (top) indicating that sequences should be the output; a database over which to search (middle); and a restriction (bottom) by keywords. the collection of sequences acts as input to the multiple alignment, performing any transformations of format that are needed. The collection of alignments is then passed to the phylogenetic tool. Like all these patterns, this is a minimum representation, each step could, for instance, be followed by further filters for quality etc (n.b. collections are viewable and editable).

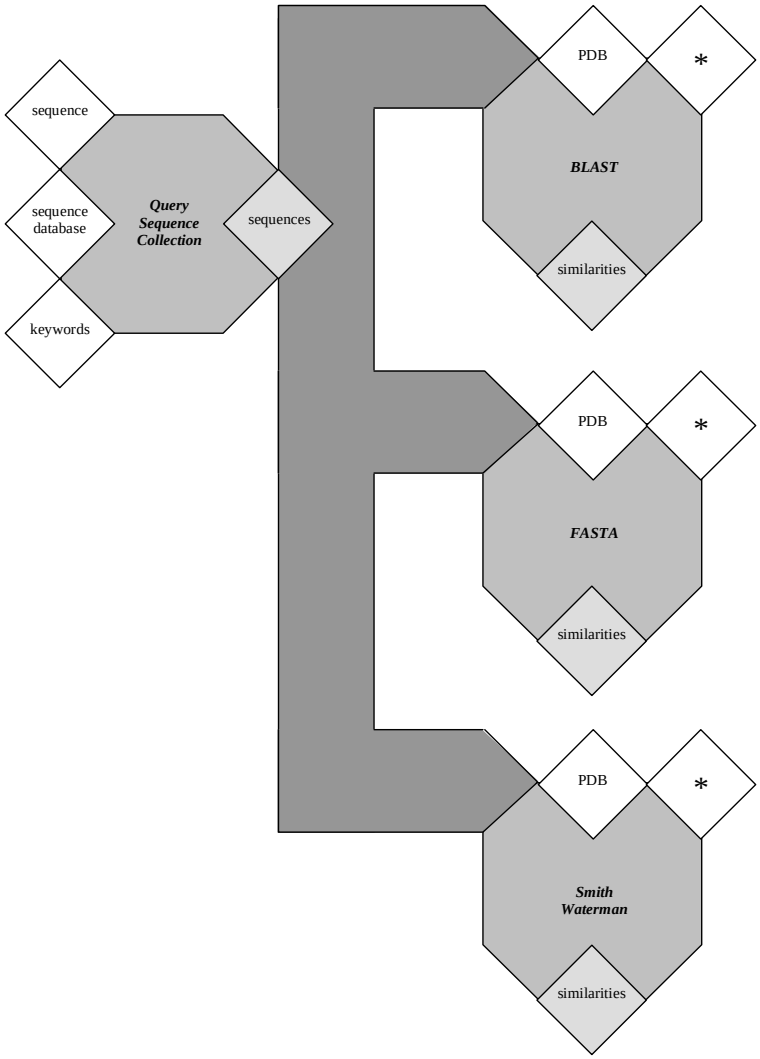


Figure 3: A syntax diagram showing the use of an unconditional fork to gather many DNA analysis tools together. All processes run simultaneously, taking copies of the sequence as inputs. A database search filter uses ‘sequence’ as a projection (top), a database to search over (middle) and keywords to restrict the filter (bottom). the fork distributes the resulting sequences between the three similarity search tools, which take the same inputs: A projection for all attributes; the PDB database and the sequences from the initial search as restriction. As this initial search may yield many sequences, the similarity searches may give collections of collections.

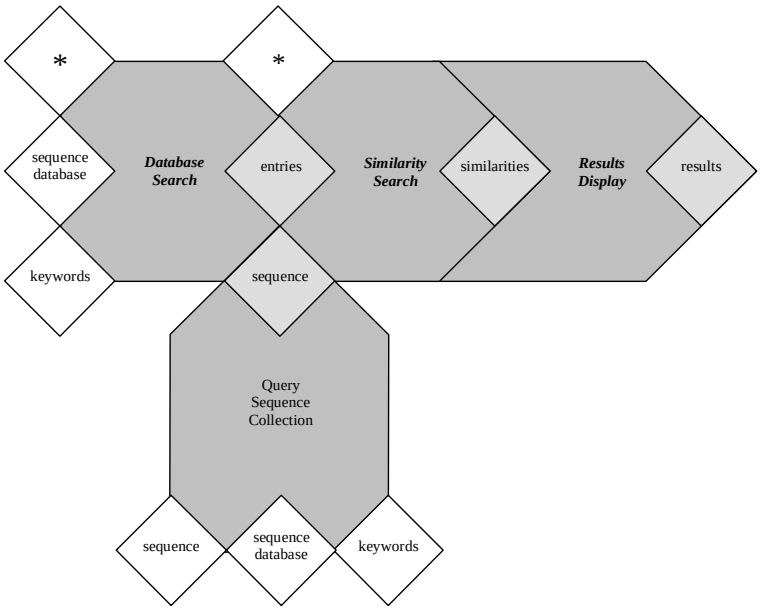


Figure 4: An enhancement of a Blast similarity search. The search itself is a transformer-filter, but the two inputs for the data-collection and the restriction collection are both collections from filters instead of stand alone collections. The output from the search is then fed into a transformer that processes it, for example, to display the results. The similarity search itself remains as a transformer-filter. The source database can be described as a sequence retrieval filter, returning either a complete database (for a standard search) or some user defined collection of sequences (species or kind of sequence). The query sequence itself could be the result of a search or some other user defined collection of query sequences (giving a series of similarity searches and thus a collection of collections of results). The output results can be transformed, either by alignment tools, dot-plot or special similarity viewer.

Question Class	**Frequency**
Sequence similarity searching	
nucleic acid vs nucleic acid	28
protein vs protein	39
Translated nucleic acid vs protein	6
Unspecified sequence Type	29
Search for Non-Coding DNA	9
Functional motif searching	35
Sequence retrieval	27
Multiple sequence alignment	21
Restriction mapping	19
Secondary and Tertiary structure Prediction	14
Other DNA Analysis including Translation	14
Primer design	12
ORF Analysis	11
Literature searching	10
Phylogenetic analysis	9
Protein analysis	10
Sequence assembly	8
Location of expression	7
Miscellaneous	7
Total	315

Table 1: The classes into which the common questions posed by biologists fall, together with their frequency.

Principle	**SRS**	**Entrez**
Biological coverage	high	core
collections	✓	✓
Act as input & output	✓	✓
Viewable	✓	✓
Editable	✓	×
Storable	✓	✓
Format transformation	✓	✓
Conceptual transformation	×	✓
Filters	✓	✓
Result collections as targets	✓	✓
Restrictions	✓	✓
Projections	✓	✓
Transformers	✓	×
Transformer-filters	✓	×
Results collection as restriction	×	×
Unconditional forks	×	×
Conditional forks	×	×
Interoperation	✓	✓

Table 2: The SRS and Entrez bioinformatics query tools evaluated by the principles set forth in Section 4. The '✓' symbol indicates the principle is satisfied in the tool and the '×' symbol indicates the principle is not satisfied. 'Results collection as target' refers to the ability to use the results of one query as target for a subsequent query. 'conceptual transformation' refers to changing the word or label used to denote a concept according to usage by a particular databank or publication. 'Restriction' indicates the ability to use filters, such as keywords, upon a source. 'Projection' is the ability to specify which attributes of a record to display. Version 6 of SRS and the March 2000 revision of Entrez were used in this evaluation.