

Open source tools and toolkits for bioinformatics: significance, and where are we?

Jason E. Stajich and Hilmar Lapp

Submitted: 11th October 2005; Received (in revised form): 28th June 2006

Abstract

This review summarizes important work in open-source bioinformatics software that has occurred over the past couple of years. The survey is intended to illustrate how programs and toolkits whose source code has been developed or released under an Open Source license have changed informatics-heavy areas of life science research. Rather than creating a comprehensive list of all tools developed over the last 2–3 years, we use a few selected projects encompassing toolkit libraries, analysis tools, data analysis environments and interoperability standards to show how freely available and modifiable open-source software can serve as the foundation for building important applications, analysis workflows and resources.

Keywords: *bioinformatics; open source; software; genomics*

INTRODUCTION

Compared to several years ago, there are numerous open source projects active in the life science arena, each offering freely available source code that promises to address a specific problem or problem domain of biological sciences in a reusable way. Bioinformatics.org (<http://www.bioinformatics.org>) alone is host to 275 projects, which address a bioinformatics need by definition. In addition, Sourceforge (<http://www.sourceforge.net>) hosts around 750 projects categorized as 'Bioinformatics' including projects such as Generic Model Organism Database (GMOD), Microarray Gene Expression Data society (MGED) and Life Sciences Identifiers (LSID). There are many projects hosted by the developers' home institutions or by other open source-devoted umbrella organizations such as the Open Bioinformatics Foundation (<http://www.open-bio.org>). The latter in fact hosts some of the toolkits most widely used in the life sciences, such as BioPerl and Biojava.

This picture is one of a multitude of projects, which at the surface seem to have commonalities, but upon closer inspection have many differences and lack an overall organization or direction that would tie them together by more than just the shared principle of open source. This scenario of each project offering 'stuff' the utility or application of which is often not immediately clear, is much reminiscent of a market bazaar where goods are sold, except that the 'stuff' on display here is software source code. Indeed, it was the metaphor of a bazaar that Eric S. Raymond coined in an article about hallmarks of successful open-source software development, 'The Cathedral and the Bazaar' [1]. The bazaar metaphor is contrasted with that of a cathedral representing a software development model with high aspirations for scope, strict development rules and designations for each individual piece of a project so that the sum of the parts forms an awe-inspiring coherent whole.

Corresponding author. Jason E. Stajich, Department of Plant and Microbial Biology, University of California, Berkeley, 321 Koshland Hall Berkeley, CA 94720, USA. Tel: (919)684-2720; Fax: (919)681-1035; E-mail: jason.stajich@duke.edu

Jason Stajich, received his PhD from Duke University in 2006 and is currently a Miller research fellow at University of California, Berkeley. He serves as the President of the Open Bioinformatics Foundation, and has authored numerous modules in BioPerl and GMOD.

Hilmar Lapp, joined NESCent as Assistant Director of Informatics in May 2006. He has served on the Board of Directors of the Open Bioinformatics Foundation since 2001, and leads development of several open source projects and modules. National Evolutionary Synthesis Center, Durham, NC 27708, USA. E-mail: hlapp@duke.edu

Inevitably, the question for the end user, developer, or informatics manager in the life sciences searching to find a solution for a particular informatics problem is which role to give to open-source software in the overall longer term strategy of software development. How can one decide between embracing an open source project, purchasing a proprietary solution, or finding a blend of the two for a particular problem? This may involve weighing the potential of an open source product and its likelihood of successfully achieving its goals. Does the increasing amount of ‘stuff’ or ‘software’ being offered for free mean that open source is the winning model in bioinformatics? Does the wide variety of problems addressed in open source bioinformatics projects mean that there will be a freely available solution to almost any problem already?

We feel that there is no universal answer for every research group to these and related questions. The term *open source* has achieved buzzword quality over the past few years, but the principles of freely available software have been tenants of collaborative science for quite a long time including the basic premise of sharing strains, plasmids, and the ‘Bermuda accords’ that outline immediate release of data from publicly funded genome sequencing projects [2]. Open source pledges more than simply inexpensive software. An Open Source-compliant license (<http://www.opensource.org>) states that not only is an application available for use, but anyone can obtain, inspect, modify and build upon the source code in new ways and redistribute the changed source. Freely available software is not meant to be ‘free’ in the sense of having no cost on the side of the end user or developer who adopts it, rather, open-source software projects offer intellectual property in a way that anyone can learn from, improve upon, participate in and extend. In contrast, closed-source software is a black box that can only be used in a manner dictated by the software publisher. Organizations who adopt an open source project with the principles of collaboration and contribute to the extension of the software will likely reap more benefits than those who see it as something with an unbeatable price tag.

In this review we aim to arm the reader not with a comprehensive laundry list of relevant open source projects, but rather with a sense of how far open-source software has come in addressing needs in the bioinformatics arena that range from analysis algorithms and providing interfaces to end users,

to toolkit libraries and to standards for interoperability and data dissemination. We select a few projects, primarily toolkit libraries that have had a significant impact on genomics science, to illustrate what open source projects have been able to achieve over the past few years, and what they could help to accomplish in the future. We also highlight the establishment of several standards for data exchange, and the utilization of these standards in research.

OPEN-SOURCE SOFTWARE LIBRARIES

Probably the most ubiquitous task in bioinformatics is to parse the results of an analysis program. BLAST is certainly the most commonly used program for sequence analysis. Hence, it is no surprise that writing a BLAST report parser has become the proverbial ‘Hello World’ programming exercise for newcomers to the field of bioinformatics. The learning process achieved through crafting one’s own parser is not to be underappreciated, especially as typically one of the lessons is that it is much less trivial than it initially appears. For example, many different formats (or flavors) exist for BLAST reports from NCBI, Washington University, and commercial implementations of the algorithm. In addition, it can become more complicated if one also wants to build reusable and extensible software so that parsing reports from different pairwise alignment programs still present the developer with the same programming interface and data objects. This enumeration of features describes the more salient capabilities of the BioPerl [3**] parsing system called Bio::SearchIO for BLAST and other pairwise search programs. The example also generally describes the target that any toolkit of reusable software components (‘modules’) will aim for when processing data that may come in different file formats such as multiple sequence alignments or sequence data. The development of generic and reusable software libraries for data processing is one of the most common tasks in bioinformatics and helps scientists and programmers alike to focus their efforts on applying the software and addressing one of the many unsolved problems in bioinformatics, rather than on the nuances of parsing different common data formats and their variants.

Software libraries have been developed for the purpose of providing fundamental building blocks for algorithms and applications alike. A vocabulary for sequence analysis includes sequence data, features,

annotation and sequence alignments. Therefore any general-purpose software library for bioinformatics will have parsers and data objects meant to read and represent these data. This infrastructure performs tasks such as parsing sequence data files or BLAST reports, and forms the building blocks for constructing workflows (protocols) for bioinformatics analyses. Because there is no *true* language for bioinformatics toolkits in several different languages have been written to provide a developer with choices so he or she can choose the best language for the job at hand. Libraries of routines and data objects in C, C++, Java, Perl, Python, R, Ruby and even Lisp exist for bioinformatics, implemented in toolkits like EMBOSS's AJAX library (<http://www.emboss.org>; [4*]), NCBI's C++ toolkit (http://ncbi.nih.gov/IEB/ToolBox/Cpp_DOC/) and Bioinformatics Toolkit Library [5], BioJava (<http://biojava.org>; [6]) and caBIO (http://ncicb.nci.nih.gov/NCICB/infrastructure/cacore_overview/caBIO; [7]), BioPerl (<http://bioperl.org>; [3]), BioPython (<http://biopython.org>; [8]), BioConductor (<http://bioconductor.org>; [9]), BioRuby (<http://bioruby.org>), and BioBike (lisp) (<http://biobike.org>; [10]).

The Perl programming language has been widely adopted in bioinformatics in part because of the ease of processing text files. Perl has also been appreciated by developers who do not come from a computer science background for being an interpreted language, which allows for more rapid code, test and deploy cycles than a compiled language. One of the most practical advantages though is the rich set of available extensions, or modules, that provide pre-written functionality. The Perl community has built a large web-accessible and searchable collection of freely available modules (<http://www.cpan.org/>). The BioPerl [3**] project is an effort to create a library of modules and routines in Perl that can be used to connect bioinformatics applications and datasets for rapid development of an application. The purpose of the toolkit is to abstract away the aspects of parsing data so that one can focus on accessing the information from analysis reports or raw data files. Developments over the past few years have provided a range of supported data and applications beyond the core, and a unified API to parsing various database search and alignment programs. It also has data models and operations for ontologies, phylogenetic trees, genetic maps and markers and population genetics [11]. The typical use of BioPerl includes parsing FastA, GenBank, or

EMBL format sequences files, processing a BLAST report and generating GFF (General Feature Format) files (<http://www.sanger.ac.uk/Software/formats/GFF/> and <http://song.sourceforge.net/gff3.shtml>). The Bio::Graphics package, which is part of BioPerl and contains the drawing code for the Generic Genome Browser [12**], can then be used to create sophisticated graphical representation of the features on top of the reference sequence. The core objects in BioPerl, particularly the sequence and feature model served as an initial backbone to the Ensembl [13*] project libraries. While the Ensembl project has stabilized towards using their own internal objects, it uses the parsers for BLAST [14], Exonerate [15] and FASTA [16] sequence analysis tools provided through Bio::SearchIO as necessary wrappers to merge similarity data into the Ensembl databases.

There are several open source libraries in the C and C++ programming languages libraries available for building bioinformatics applications. C and C++ applications tend to be the fastest implementations for bioinformatics solutions because they are compiled to machine-executable code, but also have the greatest overhead for learning to use the language productively and to master debugging challenges including memory allocation and leaks. In addition, the necessity to compile the code before one can test it adds time to the development cycle during prototyping. The Ajax library is a C library that is part of the EMBOSS package [4*] and fills a variety of sequence and alignment needs. Additional toolkits specifically tied to EMBOSS include the JEMBOSS package [17], which is a graphical interface to the applications in EMBOSS, and the Bio-EMBOSS (<http://search.cpan.org/~pernst/Bio-Emboss/>) Perl modules that provide access to the EMBOSS libraries in a Perl language environment. The libsequence [18] library also provides a collection of C language routines for population genetics analyses. For the C++ language, NCBI has developed the general purpose NCBI Toolkit (http://www.ncbi.nlm.nih.gov/IEB/ToolBox/Cpp_DOC/) and TIGR has released a number of algorithm-oriented libraries (<http://cbcb.umd.edu/software/pirate/>), both of which can be used as reusable components for building bioinformatics applications.

There is a wide array of toolkit libraries and standalone applications for bioinformatics written in Java. Java is particularly good at providing graphical user interfaces (GUI) in a completely platform-agnostic fashion. Early versions of the Java

Development Kit (JDK) were inconvenient for parsing text-oriented files, but since the integration of a straightforward regular expression API into JDK 1.4 implementing this type of task is now much more accessible to even less experienced programmers, and so Java has enjoyed increasing popularity in bioinformatics for building applications from reusable component libraries. Due to optimization of the Java Virtual Machine in modern JDKs, applications often enjoy an execution speed that obviates the need to use C or C++ to generate machine-executable code. As a strongly typed object-oriented language with a rigorous exception handling system it is also a popular choice for applications or environments that require a high degree of reliability and robustness. There are several tools for visualizing genomic annotations and comparative genomic data, such as Apollo [19], Sockeye [20] and GATA [21]. JalView provides an interactive visual environment to edit multiple sequence alignments [22]. The Mesquite project [23] has produced a suite of java packages for visually running evolutionary analysis algorithms, some of which build on the Phylogenetic Analysis Library PAL [24]. The BioJava package [6] provides rigorously designed object models for sequences, sequence features, alignments, dynamic programming algorithms, and for building classification algorithms such as support vector machines. More recently, the BioJavax extension to BioJava has added rich structured annotation to sequences and full read-write support for the BioSQL schema. Finally the caBIO system for cancer bioinformatics from the National Cancer Institute (http://ncicb.nci.nih.gov/NCICB/infrastructure/cacore_overview/caBIO; [7]) provides a framework of objects for computing and processing data related to cancer research that are general purpose enough to be employed in other bioinformatics applications. It is worth noting that there are also general-purpose Java libraries with machine learning (e.g. WEKA, <http://www.cs.waikato.ac.nz/ml/weka/>) and numerical linear algebra algorithms (e.g. JAMA, <http://math.nist.gov/javanumerics/jama/>) readily implemented to serve as building blocks for new analysis algorithms in bioinformatics.

The Python programming language has also gained much popularity in bioinformatics as a scriptable language that, in contrast to Perl, is object oriented from its outset. The BioPython project [8*] provides a library of bioinformatics

routines for sequence, structure [25], and alignment parsing in the Python programming language. The BioRuby (<http://www.bioruby.org/>) project is also rapidly developing a bioinformatics library in the relatively new scripting and object-oriented programming language Ruby. The adoption of Ruby as a programming language in bioinformatics is still rare, but may accelerate with the availability of 'Ruby on Rails' (<http://www.rubyonrails.org/>) which addresses a very common task in bioinformatics, namely developing a web-application and database for a collection of primary or secondary data.

The R software environment is an extraordinarily powerful platform for rapidly prototyping and assessing statistical analyses. R is a scripted language with a command-line interpreter that executes commands as they are typed, which together with the sophisticated plotting capabilities allows for quick successions of analysis refinement and result visualization. Implementations of R exist for Mac, Unix and Windows allowing deployment on machines in most laboratories. R can easily load algorithms at runtime that are coded in C for speed efficiency, and it has built-in support for creating packages without requiring much effort from the developer. R packages are organized in the CRAN repository (<http://www.r-project.org/>), which has several mirror sites across the globe and is organized much like the CPAN repository for Perl modules. This has led to a large number of R add-on packages available for virtually any kind of statistical analysis. In the bioinformatics realm this includes most notably the BioConductor package [9**] for microarray analysis and the APE package for Analyses of Phylogenetics and Evolution [26]. BioConductor is not so much a single package as it is a collection that achieves coherence by the shared scientific application domain, by providing a common utility framework used by all packages, and finally by defining certain data types that ensure easy data interoperability between the packages, for example the `exprSet` type for representing microarray profiling data. The transformation of how and how rigorously results from biological experiments are analysed, especially in molecular biology such as RNA profiling, is a typical example of the impact that open source can have on science, in this case by putting state-of-the-art rigorous statistical analysis algorithms within reach of every bench scientist with a desktop computer.

The BioSQL project [27] (<http://www.biosql.org>) presents a generic relational model (schema)

for representing biological sequence objects and features, their annotations and ontologies independent of their actual data source. The BioSQL schema is also the designated interoperable persistence interface between the so-called Bio* projects (BioPerl, BioJava, BioPython and BioRuby) hosted by the Open Bioinformatics umbrella foundation (<http://www.open-bio.org/>). These toolkits each have object-relational bridges to BioSQL that bind objects in the data model of the respective toolkit to corresponding entities (tables) in the relational model. The schema is moderately simple with less than 30 tables in total. The relational model is 'weakly typed,' a design approach that instead of representing each data type by its own table with columns named and typed according to an object's attributes consists of entities that in essence represent base data types with only a minimum set of attributes common to all derived data types (e.g. *feature* as base type and *DNA* feature, *protein* feature, *exon* feature and *gene* feature as derived types). Additional attributes as well as the actual type of a row are attached, and thereby typed, through association with ontology terms. This approach provides not only for almost infinite flexibility in terms of derived data types that the schema can accommodate, but also allows attaching semantics to the types of rows or attributes.

Another important area of bioinformatics data management is the consolidation of several different types of data into data warehouses or streamlined warehouses called marts. These data marts provide an effective means for quickly retrieving data integrated across different datasets including genomic, phenotypic, comparative and expression data. One such solution to this data integration comes from the Ensembl project's BioMart [28] project, an outgrowth of the EnsMart project within Ensembl which was developed to provide easy and fast data mining access to the large datasets of mammalian genomes. For example, a query for a table of the average gene expression, chromosomal location and synonymous substitution rate for all human genes with a mouse ortholog can be easily retrieved in BioMart. The BioMart implementation has been integrated in other non-Ensembl sites as well, including the nematode site Wormbase [29,30] (<http://www.wormbase.org/Multi/martview>) and the model organism database for the grasses, Gramene [31] (<http://www.gramene.org/Multi/martview>). BioMart is also used by the human HapMart tool of the HapMap [32] project

(<http://hapmart.hapmap.org/BioMart/martview>), and has lately been made part of the Generic Model Organism Database (GMOD) project (<http://www.gmod.org>). The GMOD team has developed tools so that any site using GMOD's Chado schema can easily publish a web-based BioMart data-mining interface to its data content. The software for the mart tools was easily adopted across the different projects because it was both open source and developed in a collaborative community of the labs involved. The modular and simple structure of the BioMart database will allow additional experimental data to be stored in the system across a variety of organisms and data types.

STANDARDS

Having an agreed upon standard for file formats and data exchange protocols is a critical aspect of bioinformatics; without a common language it is difficult to integrate data from different sources and write general purpose software to process it. For example, beyond the simple and easily understood FastA format, which is inadequate for any structured sequence annotation beyond a simple description, individual databases have each established their own plain text file format for richly annotated sequence data, such as the GenBank, EMBL and SwissProt formats. Additionally, there are database-specific XML formats, for example those from NCBI and TIGR. One of few recognized database-neutral XML formats is the Biomolecular Sequence Markup Language (BSML). Efforts from the GMOD consortium has defined a database and XML schema called Chado for representing genomic, phenotypic and genetic information. While there are a finite number of file formats, it can be tedious to write support for several formats that convey the same amount of information in different ways. The situation is very similar if not worse for sequence alignments, as the array of sequence alignment formats is typically dictated by the applications themselves and requires a bioinformatics toolkit to define reading and writing tools for each alignment format created by programs such as Clustal, GCG/MSF, BLAST, FastA, BLAT. There are formats, such as CIGAR, for compacting (encoding) alignments for output or for storing in a database in order to save space (described in the manual for Exonerate [15]). BLAST in its recent versions can also create XML-formatted output. Having the data in an XML

format simplifies validation of the input and makes for more robust extraction of the desired elements, but still requires properly mapping the data into an object model that satisfies the needs of the developer.

In order to achieve a common language for basic genome annotation Lincoln Stein proposed an updated standard for the tab delimited columnar Gene Finding Format (GFF, a.k.a. General Feature Format), called GFF3. GFF3 (<http://song.sourceforge.net/gff3.shtml>) extends the GFF1 and GFF2 (also known as GTF) formats with a more explicitly structured ninth column, sometimes called the Group column. The key improvement of the new standard includes a *Parent* definition that allows unambiguous relationships of genes, transcripts, exons and coding sequences. GFF3 requires that the type of a feature as well as the types of its hierarchical relationships be taken from the Sequence Ontology [33]. As the ontology in essence is a controlled vocabulary, and its terms are semantically defined, including which feature types may stand in a direct hierarchical relationship, this requirement prevents ambiguity as to how to represent a particular set of genomic features as a hierarchical 'feature graph'.

The Gene Ontology project [34] has seen a flurry of tools developed and adapted for accessing the databases and mapping gene products to ontology terms. The project itself has produced several tools to make querying the data structures easier. The directed acyclic graph (DAG) nature of the ontology means that, due to the lack of recursive queries in SQL, simple database queries do not suffice for retrieving all the terms related to a particular process. The GO-Dev suite of tools (<http://www.godatabase.org/dev/>; [35]) includes an API for the database, utilities for translation of different file-formats, and an interactive tool OBO-Edit (<http://www.godatabase.org/dev/java/oboedit/docs/>) for editing and adding terms to the ontology.

The Distributed Annotation System (DAS) (<http://www.biodas.org/>; [36]) was created to allow genome annotation exchange and to facilitate development of a client-server architecture to allow general purpose genome browsers (client) to be written without regard to how or where the annotation data is stored. This technology was adopted by several major genome annotation providers, including UCSC, TIGR and Ensembl. NCBI, one of the largest genome annotation data

providers, is a notable exception. Several classes of open-source software implementations have been written and released including a server written in Java called Dazzle (<http://www.biojava.org/wiki/Dazzle>) and a Perl server built into the Bio::DB::GFF system of BioPerl [3] and Gbrowse [12]. More recently the DAS project has undertaken an NIH-funded project to update the specification to a second version called DAS/2. DAS/2 supports write-back, facilitating its use for genome annotation, but also removes ambiguities in query interpretation and response generation that emerged through different implementations of the first version of the standard. In addition, DAS/2 provides the framework for significant performance optimization by allowing the client and server to negotiate other, typically much more efficient, formats for the response than the default text-based format.

Standards also play a role in connecting software together. Standards for specifying input and output of software applications allow them to be deployed on a compute infrastructure like a Grid or a local compute cluster and to shield the user from the details of where or how the programs are actually run. These standards also allow workflows or pipelines to be built that are effectively a bioinformatics protocol. Some existing solutions to this include BioPipe [37] which is an XML based pipeline system relying on BioPerl and the open source database engine MySQL to run a series of tasks that may build on the results of previous steps. The Pegasys application [38*] is a more advanced system with a friendly GUI for building workflows to run on local clusters with drag and drop ease. The Taverna [39*] project is another robust workflow construction tool geared towards running jobs on a compute Grid such as myGrid. The Kepler project (<http://kepler-project.org/>) aims to provide an extensible workflow construction and execution system with a user-friendly GUI. Steps in the workflow are called 'actors'. So far the collection of actors packaged with the distribution is focused on phylogenetic analysis. These pipeline and grid technologies are generally most useful to advanced bioinformatics users.

For the bench scientists who are interested in a smaller and less complicated protocols to run a series of automated analyses, the web is an excellent platform for biocomputing. The PISE project [40] was one of the first to build a standardized web

interface by generating an XML description of input, parameters, and output of over 150 bioinformatics and phylogenetic tools. Extending this technology from a single site providing web interfaces to many different sites that provide tools is the goal of the MOBY project (<http://www.biomoby.org/>). MOBY and S-MOBY [41,42] are web services which use ontologies and standards to connect input and output from different programs into a semantic web framework. The project will standardize bioinformatics web technologies so they can be automated and interconnected between different sites. This puts the power of automation with the ease of web-based interaction in the hands of the bench scientist who is interested in answering a few questions about their favorite genes.

OPEN SOURCE AND RESEARCH

The software libraries and basic infrastructure described have been instrumental in building new bioinformatics tools and in published research projects. The tools are reused not only because they are free but also because the open source development process has garnered input from a community so that the tools can be continuously improved in their usefulness.

The BioPerl project serves as a model for how parts of a reusable toolkit can be applied to research questions and development of new tools. The collaborative nature of the project has fostered a community of users providing feedback, which in turn has made a more useful product, illustrated by the breadth of research projects and software tools that have been built using it. The core functionality provided by the toolkit has been used in creating and validating annotation of genomes [29,43–47], distinguishing orthologous and paralogous genes [48,49], building synteny comparison tools [50], predicting miRNA genes [51], transcription factor binding site identification [52,53], and automating the generation of PCR [54] primers, morpholino oligos [55], or RNAi constructs [56,57]. Recently added capabilities include modules to calculate population genetics summary statistics, generate polymorphism information from multiple sequence alignments, parse and manipulate phylogenetic trees, calculate pairwise sequence distances, construct neighbor-joining trees and render trees in postscript or SVG graphical formats. Some of

these evolutionary tools have been used for whole genome comparisons of gene loss and gain rates [58] and human population genetic analyses [11,59].

The GMOD project is creating a set of generic components necessary for genome databases. In this example the tools created for GMOD are intended to be reusable components for deployment in model organism and genome databases. Projects under the umbrella of GMOD include the aforementioned Chado project, which provides a generic database schema for storing genome and sequence annotation, as well as expression, mapping and genetic data. Textpresso [60*], another project under the GMOD umbrella, is a literature search engine which indexes full scientific articles split at the sentence level with terms mapped into semantic categories. Gbrowse, the Generic Genome Browser [12**] and one of the first tools submitted to GMOD, has continued to mature and become a very stable browser supporting several different database schemas such as Chado, aside from its own schema defined in the Bioperl module Bio::DB::GFF. A GBrowse extension called SynBrowser [61] was recently built for visualizing synteny between genomes.

As an example, research in gene finding that requires secondary structure prediction algorithms has been advanced and rapidly disseminated through open source release of algorithms, RNA gene prediction has recently made significant progress. The stemloc application [62], part of the DART (<http://dart.sourceforge.net/>) package, provides a tool to identify conserved secondary structure by simultaneously aligning and folding a sequence using pairwise stochastic context-free grammars (SCFG). Another RNA finding tool, RNAz [63], evaluates multiple sequence alignments of putative genomic loci to identify conserved secondary structure and compensatory mutations to predict if the sequence encodes an RNA. RNAz builds on the open-source ViennaRNA package [64*] which provides C and Perl APIs for sequence folding and secondary structure evaluation. This library is also a critical component in the MiRscan tool for microRNA prediction in *Caenorhabditis elegans* [65]. Other open source RNA finding and evaluation tools such as Infernal [66] and RSEARCH [67], use SCFGs to search for homologs of genes with *a priori* knowledge of the structure. Infernal, with some heuristics, is an essential part of the Rfam project [68] to identify orthologous RNA genes in different

species based on models of the gene's secondary structure and conservation across several different species. Because the tools are written as modular components they can be interchanged between different research questions.

SUMMARY

There has been a great deal of progress in development of new bioinformatics tools for infrastructure, but also for machine learning and analysis algorithms. In addition to the productive development of new tools, user interfaces and documentation are available putting these tools into the hands of bench scientists in a user-friendly manner. Many of these tools and toolkits are open source, allowing them to be folded into other applications and pipelines as reusable building blocks. While we concentrated primarily on toolkits in the genomics and molecular biology application domain, there are mature open source projects with visionary goals in many other biomedical domains as well, for example OME (<http://openmicroscopy.org/>) in biomedical imaging. The development of open and freely available infrastructure for bioinformatics analyses across the web is bringing the algorithms and tools closer to scientists and enabling new ways of integrating data analyses. Future work will continue to expand this infrastructure and the types of software applications that can be used and provide more powerful and robust analysis tools for bioinformatics.

WEBSITES REFERENCED

BioConductor	http://www.bioconductor.org/
Bio-EMBOSS Perl	http://search.cpan.org/~pernst/Bio-Emboss/
BioJava	http://www.biojava.org/
BioMart	http://www.biomart.org/
BioMoby	http://www.biomoby.org/
BioPerl	http://www.bioperl.org/
BioPython	http://www.biopython.org/
caBIO	http://ncicb.nci.nih.gov/NCICB/infrastructure/cacore.overview/caBIO
DAS	http://www.biodas.org/
DART	http://dart.sourceforge.net/
Dazzle	http://www.biojava.org/wiki/Dazzle
EMBOSS	http://www.emboss.org/
Ensembl	http://www.Ensembl.org/
Gene Ontology	http://www.geneontology.org/
GFF Specification	http://www.sanger.ac.uk/Software/formats/GFF/
GFF3 Specification	http://song.sourceforge.net/gff3.shtml
GO-Dev	http://www.godatabase.org/dev/
Gramene Mart	http://www.gramene.org/Multi/martview

HapMart	http://hapmart.hapmap.org/BioMart/martview
JAMA	http://math.nist.gov/javanumerics/jama/
JEMBOSS	http://emboss.sourceforge.net/Jemboss/
NCBI C++ Toolkit	http://www.ncbi.nlm.nih.gov/IEB/ToolBox/CPDOC/
OBO-Edit	http://www.godatabase.org/dev/java/oboedit/docs/
Open Bioinformatics Foundation	http://www.open-bio.org/
Open Source	http://www.open-source.org/
OME	http://openmicroscopy.org/
Ruby on Rails	http://www.rubyonrails.org/
Sequence Ontology	http://song.sourceforge.net/
TIGR C/C++ libraries	http://cpcb.umd.edu/software/pirate/
WEKA	http://www.cs.waikato.ac.nz/ml/weka/
Wormbase	http://www.wormbase.org/
Wormbase BioMart	http://www.wormbase.org/Multi/martview

References

1. Raymond ES. *Cathedral and the Bazaar: Musings on Linux and Open Source by and Accidental Revolutionary*. Sebastopol, CA: O'Reilly & Associates, 1999.
2. Waterston R, Sulston JE. The Human Genome Project: reaching the finish line. *Science* 1998;**282**:53–4.
3. **Stajich JE, Block D, Boulez K, *et al*. The Bioperl toolkit: Perl modules for the life sciences. *Genome Res* 2002;**12**: 1611–8.
The development and design of the BioPerl project.
4. *Rice P, Longden I, Bleasby A. EMBOSS: the European Molecular Biology Open Software Suite. *Trends Genet* 2000; **16**:276–7.
EMBOSS is a great collection of sequence analysis applications and has library of routines for building additional tools.
5. Pitt WR, Williams MA, Steven M, *et al*. The Bioinformatics Template Library—generic components for biocomputing. *Bioinformatics* 2001;**17**:729–37.
6. Pocock MR, Down T, Hubbard T. BioJava: open source components for bioinformatics. *SIGBIO Newsl.* 2000;**20**: 10–12.
7. Covitz PA, Hartel F, Schaefer C, *et al*. caCORE: a common infrastructure for cancer informatics. *Bioinformatics* 2003;**19**: 2404–12.
8. de Hoon MJL, Chapman B, Friedberg I. Bioinformatics and Computational Biology with Biopython. In: Gribskov M, Kanehisa M, Miyano S, *et al*, (eds). *Genome Informatics*. Pacifico Yokohama, Japan: Universal Academy Press, 2003; 298–9.
9. **Gentleman RC, Carey VJ, Bates DM, *et al*. Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol* 2004;**5**:R80.
The BioConductor project is a remarkable set of integrate statistical and graphical tools for analyzing microarray data in the R environment.
10. Massar JP, Travers M, Elhai J, *et al*. BioLingua: a programmable knowledge environment for biologists. *Bioinformatics* 2005;**21**:199–207.
11. Stajich JE, Hahn MW. Disentangling the effects of demography and selection in human history. *Mol Biol Evol* 2005;**22**:63–73.

12. **Stein LD, Mungall C, Shu S, *et al.* The generic genome browser: a building block for a model organism system database. *Genome Res* 2002;**12**:1599–610.
The generic genome browser is one of the most deployed genome browser because of its straightforward use and installation.
13. *Stabenau A, McVicker G, Melsopp C, *et al.* The Ensembl core software libraries. *Genome Res* 2004;**14**:929–33.
Then EnsEMBL software system provides a robust automated annotation of mammalian genomes and one of the most advanced interactive website to interact with genome data.
14. Altschul SF, Madden TL, Schaffer AA, *et al.* Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Res* 1997;**25**:3389–402.
15. Slater GS, Birney E. Automated generation of heuristics for biological sequence comparison. *BMC Bioinformatics* 2005;**6**:31.
16. Pearson WR, Lipman DJ. Improved tools for biological sequence comparison. *Proc Natl Acad Sci USA* 1988;**85**:2444–8.
17. Carver T, Bleasby A. The design of Jemboss: a graphical user interface to EMBOSS. *Bioinformatics* 2003;**19**:1837–43.
18. Thornton K. Libsequence: a C++ class library for evolutionary genetic analysis. *Bioinformatics* 2003;**19**:2325–7.
19. Lewis SE, Searle SM, Harris N, *et al.* Apollo: a sequence annotation editor. *Genome Biol* 2002;**3**:RESEARCH0082.
20. Montgomery SB, Astakhova T, Bilenky M, *et al.* Sockeye: a 3D environment for comparative genomics. *Genome Res* 2004;**14**:956–62.
21. Nix DA, Eisen MB. GATA: a graphic alignment tool for comparative sequence analysis. *BMC Bioinformatics* 2005;**6**:9.
22. Clamp M, Cuff J, Searle SM, *et al.* The Jalview Java alignment editor. *Bioinformatics* 2004;**20**:426–7.
23. Maddison WP, Maddison DR. Mesquite: a modular system for evolutionary analysis. 2005.
24. Drummond A, Strimmer K. PAL: an object-oriented programming library for molecular evolution and phylogenetics. *Bioinformatics* 2001;**17**:662–3.
25. Hamelryck T, Manderick B. PDB file parser and structure class implemented in Python. *Bioinformatics* 2003;**19**:2308–10.
26. Paradis E, Claude J, Strimmer K. APE: Analyses of Phylogenetics and Evolution in R language. *Bioinformatics* 2004;**20**:289–90.
27. Lapp H, Mackey A, Down T, *et al.* BioSQL — Generic and interoperable persistence for biological sequences, features, and their annotations, in Prep.
28. Kasprzyk A, Keefe D, Smedley D, *et al.* EnsMart: a generic system for fast and flexible access to biological data. *Genome Res* 2004;**14**:160–9.
29. Chen N, Harris TW, Antoshechkin I, *et al.* WormBase: a comprehensive data resource for *Caenorhabditis* biology and genomics. *Nucleic Acids Res* 2005;**33**:D383–9.
30. Harris TW, Chen N, Cunningham F, *et al.* WormBase: a multi-species resource for nematode biology and genomics. *Nucleic Acids Res* 2004;**32**:D411–7.
31. Ware D, Jaiswal P, Ni J, *et al.* Gramene: a resource for comparative grass genomics. *Nucleic Acids Res* 2002;**30**:103–5.
32. Thorisson GA, Smith AV, Krishnan L, *et al.* The International HapMap Project Web site. *Genome Res* 2005;**15**:1592–3.
33. Eilbeck K, Lewis SE, Mungall CJ, *et al.* The Sequence Ontology: a tool for the unification of genome annotations. *Genome Biol* 2005;**6**:R44.
34. Harris MA, Clark J, Ireland A, *et al.* The Gene Ontology (GO) database and informatics resource. *Nucleic Acids Res* 2004;**32**:D258–61.
35. Gene Ontology Consortium. The Gene Ontology (GO) project in 2006. *Nucleic Acids Res* 2006;**34**:D322–6.
36. Dowell RD, Jokerst RM, Day A, *et al.* The distributed annotation system. *BMC Bioinformatics* 2001;**2**:7.
37. Hoon S, Ratnapu KK, Chia JM, *et al.* Biopipe: a flexible framework for protocol-based bioinformatics analysis. *Genome Res* 2003;**13**:1904–15.
38. *Shah SP, He DY, Sawkins JN, *et al.* Pegasys: software for executing and integrating analyses of biological sequences. *BMC Bioinformatics* 2004;**5**:40.
39. *Oinn T, Addis M, Ferris J, *et al.* Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics* 2004;**20**:3045–54.
40. Letondal C. A Web interface generator for molecular biology programs in Unix. *Bioinformatics* 2001;**17**:73–82.
41. Wilkinson M, Schoof H, Ernst R, *et al.* BioMOBY successfully integrates distributed heterogeneous bioinformatics Web Services. The PlaNet exemplar case. *Plant Physiol* 2005;**138**:5–17.
42. Wilkinson MD, Links M. BioMOBY: an open source biological web services proposal. *Brief Bioinform* 2002;**3**:331–41.
43. Jaffe JD, Stange-Thomann N, Smith C, *et al.* The complete genome and proteome of *Mycoplasma mobile*. *Genome Res* 2004;**14**:1447–61.
44. Blakesley RW, Hansen NF, Mullikin JC, *et al.* An intermediate grade of finished genomic sequence suitable for comparative analyses. *Genome Res* 2004;**14**:2235–44.
45. Saunders NF, Thomas T, Curmi PM, *et al.* Mechanisms of thermal adaptation revealed from the genomes of the Antarctic Archaea *Methanogenium frigidum* and *Methanococcoides burtonii*. *Genome Res* 2003;**13**:1580–8.
46. Stein LD, Bao Z, Blasiar D, *et al.* The genome sequence of *Caenorhabditis briggsae*: a platform for comparative genomics. *PLoS Biol* 2003;**1**:E45.
47. Yandell M, Bailey AM, Misra S, *et al.* A computational and experimental approach to validating annotations and gene predictions in the *Drosophila melanogaster* genome. *Proc Natl Acad Sci USA* 2005;**102**:1566–71.
48. Cannon SB, Young ND. OrthoParaMap: distinguishing orthologs from paralogs by integrating comparative genome data and gene phylogenies. *BMC Bioinformatics* 2003;**4**:35.
49. O'Brien KP, Remm M, Sonnhammer EL. Inparanoid: a comprehensive database of eukaryotic orthologs. *Nucleic Acids Res* 2005;**33**:D476–80.
50. Cannon SB, Kozik A, Chan B, *et al.* DiagHunter and GenoPix2D: programs for genomic comparisons, large-scale homology discovery and visualization. *Genome Biol* 2003;**4**:R68.
51. Adai A, Johnson C, Mlotshwa S, *et al.* Computational prediction of miRNAs in *Arabidopsis thaliana*. *Genome Res* 2005;**15**:78–91.

52. Berman BP, Pfeiffer BD, Lavery TR, *et al.* Computational identification of developmental enhancers: conservation and function of transcription factor binding-site clusters in *Drosophila melanogaster* and *Drosophila pseudoobscura*. *Genome Biol* 2004;**5**:R61.
53. Lenhard B, Wasserman WW. TFBS: Computational framework for transcription factor binding site analysis. *Bioinformatics* 2002;**18**:1135–6.
54. Gadberry MD, Malcomber ST, Doust AN, *et al.* Primaclade—a flexible tool to find conserved PCR primers across multiple species. *Bioinformatics* 2005;**21**:1263–4.
55. Klee EW, Shim KJ, Pickart MA, *et al.* AMOD: a morpholino oligonucleotide selection tool. *Nucleic Acids Res* 2005;**33**:W506–11.
56. Arziman Z, Horn T, Boutros M. E-RNAi: a web application to design optimized RNAi constructs. *Nucleic Acids Res* 2005;**33**:W582–8.
57. Dudek P, Picard D. TROD: T7 RNAi Oligo Designer. *Nucleic Acids Res* 2004;**32**:W121–3.
58. Babenko VN, Krylov DM. Comparative analysis of complete genomes reveals gene loss, acquisition and acceleration of evolutionary rates in Metazoa, suggests a prevalence of evolution via gene acquisition and indicates that the evolutionary rates in animals tend to be conserved. *Nucleic Acids Res* 2004;**32**:5029–35.
59. Hahn MW, Rockman MV, Soranzo N, *et al.* Population genetic and phylogenetic evidence for positive selection on regulatory mutations at the factor VII locus in humans. *Genetics* 2004;**167**:867–77.
60. *Muller HM, Kenny EE, Sternberg PW. Textpresso: an ontology-based information retrieval and extraction system for biological literature. *PLoS Biol* 2004;**2**:e309.
61. Pan X, Stein L, Brendel V. SynBrowse: a synteny browser for comparative sequence analysis. *Bioinformatics* 2005;**21**:3461–8.
62. Holmes I. Accelerated probabilistic inference of RNA structure evolution. *BMC Bioinformatics* 2005;**6**:73.
63. Washietl S, Hofacker IL, Stadler PF. Fast and reliable prediction of noncoding RNAs. *Proc Natl Acad Sci USA* 2005;**102**:2454–9.
64. Hofacker IL, Fontana W, Stadler PF, *et al.* Fast folding and comparison of RNA secondary structures. *Monatshfte Fur Chemie* 1994;**125**:167–88.
65. Lim LP, Lau NC, Weinstein EG, *et al.* The microRNAs of *Caenorhabditis elegans*. *Genes Dev* 2003;**17**:991–1008.
66. Eddy SR. A memory-efficient dynamic programming algorithm for optimal alignment of a sequence to an RNA secondary structure. *BMC Bioinformatics* 2002;**3**:18.
67. Klein RJ, Eddy SR. RSEARCH: finding homologs of single structured RNA sequences. *BMC Bioinformatics* 2003;**4**:44.
68. Griffiths-Jones S, Moxon S, Marshall M, *et al.* Rfam: annotating non-coding RNAs in complete genomes. *Nucleic Acids Res* 2005;**33**:D121–4.