Bioinformatics Open Source Conference 2002

Table of Contents

Welcome

Welcome to BOSC 2002! This is the 3rd official Bioinformatics Open Source Meeting and the first since the creation of the Open Bioinformatics Foundation. We're very excited to have full days of talks this year with 3 keynote speakers to talk about the open source principals in the context of their scientific work. We hope the field of invited speakers will provide a broad perspective of ongoing projects that focus on needs in the community from infrastructure and middle layers to user applications to novice user education to broader treatments of open source principals in the academic field. A poster session will occur on the afternoon of August 1 please come by and discuss the poster presentations with the authors. Software demonstrations will occur as well during this period. In the evenings we have scheduled space for Birds of a Feather (BoF) discussions, please take advantage of this time to attend discussion on specialized topics. If you would like to schedule a BoF see the signup chart that will be available in the mornings. If you have any questions or concerns about the conference please let one of the conference committee member know.

We hope you enjoy yourself, learn a lot, and most importantly get to know each other and become part of the community of open source development in the life sciences.

Conference Committee

Jason Stajich (chair)	Duke University
Ewan Birney	European Bioinformatics Institute
Chris Dagdigian	BioTeam.Net
Andrew Dalke	Dalke Scientific Software
Nomi Harris	University of California, Berkeley

Abstracts

Ewan Birney Keynote

Ewan Birney, European Bioinformatics Institute

August 1, 9:15-10:15

Are we growing up? Reports from the open bioinformatics foundation and the open bioinformatics database access project.

Bioperl project report

Jason Stajich, Duke University

August 1, 10:15-10:40

The Bioperl project recently released version 1.0 of our toolkit for life science programming. The components include modules which support sequences, sequence reading and writing, sequence features and annotations and features with simple and complex locations, multiple sequence alignments, phylogenetic trees, BLAST & FASTA parsing, building and accessing local sequence databases, accessing remote sequence databases, retrieving and manipulating bibliographic references, interoperating with BioCORBA and OBDA biological object standards. The toolkit has been used in a wide array of situations from simple laboratory situations and as the building blocks for enterprise solutions in EnsEMBL and the Generic Model Organism Database (GMOD).

The toolkit is built with an easily extensible architecture which can be used for quickly building Perl programs to address specific research questions. Several examples of its use to answer real laboratory questions will be discussed.

License: Perl Artistic.

Bioperl-Pipeline System

Shawn Hoon, Fugu Genome Project, Institute Of Molecular and Cell Biology, Singapore

August 1, 11:00-11:25

The prominence of the in-silico laboratory coupled with the explosion of comparative genomics have made the nature of computational biological analysis increasingly complex. This is exacerbated by the plethora of software that are now available. It is not uncommon for an analysis to involve large amounts of data from disparate sources and formats, different programs with specific requirements and output formats that must be suitable for human interpretation. There thus exists a need for a flexible workflow framework that will hide such complexity, allowing scientists to focus on their analysis, while providing bioinformaticians a coherent methodology for which to extend the system. It was with this in mind that we developed the bioperl-pipeline system. Largely adapted from the Ensembl Pipeline Annotation System, some of the features in the current system include:

a. Handling of various input and output data formats from various databases.

b. A bioperl interface to non-specific loadsharing software (LSF,PBS etc) to ensure that the various analysis programs are run in proper order and are successfully completed while re-running those that fail.

c. A flexible pluggable bioperl interface that allows programs to be 'pipeline-enabled'.

We are currently looking at extending the system in the following way:

a. A 'grid-aware' system that allows jobs to be distributed over a bio-cluster network harnessing collective computing power that will be especially useful for small groups looking to perform compute-intensive analysis.

b. A user-friendly click and drag GUI system to allow easy workflow design and job tracking.

We are now applying this framework to our compara system for high throughput multi-species studies. We will discuss the design and implementation details of the bioperl-pipeline package.

License: Perl Artistic

Cancer Bioinformatics Infrastructure Objects (caBIO): An open-source, object oriented API for biomedical informatics

Peter A. Covitz, Himanso Sahni, Scott Gustafson, and Kenneth Buetow; National Cancer Institute Center for Bioinformatics

August 1, 11:25-11:35

The National Cancer Institute has established a Center for Bioinformatics (NCICB) whose mission is to support the NCI's programs in basic and clinical cancer research. The NCICB is aggressively pursuing a program to develop a core infrastructure and API for biomedical information management and retrieval. The initiative employs industry-standard software engineering methodologies to develop data models, middleware, vocabularies and ontologies for biomedical research.

caBIO is the primary programming interface to the core infrastructure. caBIO objects are implemented using Java and Java Bean technology, and represent biological and laboratory entities such as genes, chromosomes, sequences, libraries, clones, pathways, and ontologies. caBIO provides uniform API access to a variety of genomic, biological, and clinical data sources including GenBank, Unigene, LocusLink, Homologene, Ensembl, Golden Path, DAS servers, CGAP, NCI Enterprise Vocabulary Services, and clinical trials protocols. Any client can retrieve HTML and XML from caBIO via HTTP. Java-based clients can further communicate with caBIO via the domain objects provided by the caBIO JAR, while server components can communicate via Java RMI. Non-Java applications can communicate via SOAP. RDF is currently used to advertise services to crawlers and agents, and a UDDI registry is planned. For its presentation layer, caBIO uses servlets and JSPs under Jakarta Tomcat. All caBIO objects can be transformed into XML, and XSL/XSLT is used to present data in documents, web pages or other interfaces.

NCICB makes the caBIO interfaces available on its public servers, and also makes the underlying software available for use at at local sites. More information is available at http://ncicb.nci.nih.gov/core". The open source license covering caBIO software can be found at http://ncicb.nci.nih.gov/core/caBIO/developer_resources/core_jar/license.

Biopython and the Laboratory Scientist

Brad Chapman, University of Georgia

August 1, 11:35-12:00

Biopython is a collection of open-source tools in the Python programming language. Developed by a collection of programmers from around the world, the Biopython toolkit is designed to provided re-usable code for anyone answering biological questions using Python. Biopython has been around since 1999, and has a number of active contributors and users. In this talk, I will briefly describe the basic components provided in the Biopython toolkit. From there, I will describe how Biopython can be used in a academic laboratory environment, taking examples from my own lab. The emphasis will be on utilization of Biopython code for automating everyday tasks faced by wet lab researchers. I will try to show that Python and Biopython can be used productively by researchers lacking formal training in computer science. Finally, I will describe integrating Biopython into larger bioinformatics projects. Again, this will draw on my own experience using Biopython and will describe how using Biopython can help make your coding life easier when approaching a large project. The aim of the entire talk it to convince you that using open source libraries like Biopython is worth the time invested in learning it.

The Open Source Authors' Contract

Steven Brenner, University of California, Berkeley

August 1, 1:50-2:15

Most universities, national laboratories, companies, and other employers have clauses in their employment contracts that prevent or restrict the creation and use of open source software. Indeed, it seems likely that much of the biological open source software is being produced illegally, in violation of institutions' terms. While benign neglect of enforcement of the institutions' regulations has led to a situation that is generally acceptable, it is not ideal.

Several individuals have sought the ability to produce open source software by seeking exemptions or variations of their institutions' intellectual property agreements. However, this is a painstaking process, and the associated legal fees can be costly. I propose that a general contract be drawn up, which has standard terms for individuals to create open source software without undue constraints.

Since this idea was first broached a year ago, there has been widespread discussion regarding regulations governing production open source software. This talk will provide a background to the motivation for the Authors' contract, as well as recent responses which suggest productive ways forward.

BioJava Toolkit Progress

Matthew Pocock, BioJava Consulting Limited

August 1, 2:15-2:40

BioJava is an open-source software project that aims to provide an industry-quality Java library for common bioinformatics tasks. BioJava is part of the open-bio foundation. BioJava was started in the autumn of 1998, and now has over 25 developers. In the past two years, the core development team has expanded from the original team of two to five. This has brought with it a greater range of views and expertise, as well as a greater stability. In parallel with this, we are in the process of integrating unit testing to maintain the quality of the >130,000 lines of code and documentation in the core library.

BioJava has taken an active role in participating in the open-bio hackathons. Representatives have attended both legs of the hackathon (Tuscon, AZ, USA and Cape Town, SA). During this time, several important interoperable technologies were designed and implemented. These include a registry file format for biological entities, an SQL schema for storage of sequences and their annotations, BioCorba-based CORBA clients and servers, bibliographic web services, web services for publishing sequence data and flat file indexing. All of these have been implemented in BioJava, and interoperate with implementations in the other open-bio language projects, as well as with some external implementations.

Over the next year, we hope to mature the library's functionality in areas related to sequence manipulation, pipeline management, alignments, Sequence GUIs and file parsers. In parallel, we shall be integrating code-generation, more flexible transaction management and ontology representations with the current free-form annotation model and BioJava interfaces to allow the representation of more fluid data types, and more maintainable and robust implementation of standard interfaces.

GOET: the General Ontology Editing Tool

John Richter, Berkeley Drosophila Genome Project

August 1, 2:40-2:50

GOET is a Java application designed to facilitate the creation of ontology schemas and data. GOET allows a user to define DAML+OIL-like schemas and then populate those schemas with data. Data can loaded from and saved to DAML+OIL flat files, as well as numerous other formats.

GOET is highly customizable via pluggable editor kits. Editor kits are Java jar files that define a custom user interface for GOET, tailored to a particular kind of data. Editor kits allow programmers to create the most efficient user interface for any given ontology. GOET comes with a generic editor kit that can edit *any* ontology, making it easy for users to experiment with new schemas.

GOET provides a strong toolkit for ontology editing, with automatic support for history tracking, undo/redo, cycle checking, and other important graph editing tools. This toolkit makes it easy for programmers to develop new, powerful editor kits.

Other information:

GOET is being developed as part of the gmod project at http://sourceforge.net/projects/gmod.

Like all gmod components, GOET is distributed under the terms of the Artistic License.

GHMM & HMMed: A comprehensive HMM toolkit

Alexander Schliep, Max-Planck-Institut for Molecular Genetics

August 1, 2:50-3:00

Hidden Markov Models (HMMs) are one of the most successful tools for analyzing biological sequences.

We have developed a graphical editor for HMMs called HMMEd which allows to create sophisticated models manually using a graphical user interface. Hierarchical models are supported (e.g. a three state model representing a single codon as one 'super state'), as well as a wide range of HMM extensions and user data associated with the states of the HMM. Graphical editors for discrete emission distributions as well as mixtures of continuous pdfs are integrated.

For the exchange of HMMs we propose a XML-based format which is loosely based on GraphML, is hierarchical and also incorporates necessary extensions for proper graphical display.

The GNU (pending permission from the FSF) HMM library (GHMM) is a C-library providing efficient implementations of a comprehensive collection of algorithms for both discrete and continuous emission HMMs. Python bindings allow interactive work with HMMs from the Python command line and, at some later stage, tight integration with HMMEd, which is also written in Python using Tkinter.

HMMEd (pronounced Hammered) and the GHMM are licensed under the LGPL.

Usability

Andrew Dalke, Dalke Scientific Software

August 1, 3:00-3:10

Open source software is often said to be "unusable." On the surface this doesn't make sense because many of the projects are widely used to do real work. But usability isn't a binary value, it refers to ease of use. Two packages can be equally featureful but one be much more usable than the other.

A lot of research has gone into understanding how to make more usable software. This knowledge is starting to make its way into mainstream software projects, but is still relatively unused in bioinformatics. I'll discuss several reasons why this might be so, the major one being that few even know this topic exists.

In my talk I'll cover some of the standard techniques of usability design, including testing, persona development, use cases, and paper prototyping. These are simple, inexpensive techniques that can be applied to almost any project to make them more usable and enjoyable. To keep my presentation grounded, I'll include examples from my experiences in applying them to real projects.

Michael Eisen Keynote

Michael Eisen, University of California, Berkeley

August 1, 4:00-5:00

Creating an Electronic Public Library of Scientific Knowledge.

Winston Hide Keynote

Winston Hide, South African National Bioinformatics Institute

August 2, 9:00-10:00

Dr. Hide will be presenting on the impact of Open Source in the Real World.

OmniGene

Brian Gilman, Whitehead Institute

August 2, 10:00-10:25

The OmniGene project has produced modules to build web services for data analysis, integration, and visualization. OmniGene accomplishes this goal through the utilization of Java Enterprise and web service technologies. The core API consists of modules to:

1. perform queries across disparate databases,
2. Transform queries into commonly used XML formats,
3. Parse the output of these queries into an object graph,
4. Visualize and share knowledge in a client server or true peer to peer network,
5. Dynamically discover another web service,
6. Easily plug in analysis applications.

One major goal of the OmniGene development team is to abstract away XML parsing, Enterprise Java Bean, and transaction code from the bioinformatician so that they may concentrate on their data and data analysis.

The OmniGene system will utilize the output and API from the BioMOBY project to perform dynamic discovery of services. The BioMOBY and OmniGene developers are now working together to integrate their two projects. OmniGene is an open source, open standards initiative and is distributed under the BSD license.

Generic Genome Browser & Apollo

Joint talk by Lincoln Stein & Nomi Harris, University of California, Berkeley.

August 2, 10:45-11:10

The Generic Genome Browser: A Building Block for a Model Organism System

Lincoln D. Stein, Allen Day, Todd Harris, Adrian Arva

Cold Spring Harbor Laboratory

ShengQiang Shu, Suzanna Lewis, Christopher Mungall

Berkeley Drosophila Genome Project, Lawrence Berkeley Laboratory

The Generic Model Organism System Database Project (GMOD) seeks to develop and release reuseable software components for model organism system databases. Here we describe the Generic Genome Browser (GBrowse), a web-based application for displaying genomic annotations. For the end user, features of the browser include the ability to scroll and zoom through arbitrary regions of a genome, to enter a region of the genome by search for a landmark, full text search of all annotations, the ability to enable and disable tracks and change their relative order and appearance, the ability to upload private annotations and view them in the context of the public ones, and the user's ability to publish his own annotations to the community. For the data provider, features of the browser software include reliance on readily-available Open Source components, simple installation, flexible configuration, and easy integration with other components of a model organism system web site.

GBrowse is written in the Perl programming language and makes extensive use of the BioPerl middleware layer. This gives the browser the flexibility to take advantage of a number of underlying databases and data sources, including ones based on the Distributed Annotation System (DAS). For new developers, GBrowse uses a minimal MySQL-based database called Bio::DB::GFF. Developers requiring a richer database back end can use the GadFly database, an outgrowth of the Berkeley Drosophila Genome Sequencing project.

GBrowse is currently used as the genome browser for the WormBase (http://www.wormbase.org/) and FlyBase (http://www.flybase.org/) projects. Its source code, example data and configuration files, and support are all available at the GMOD web site, http://www.gmod.org.

The Apollo Genome Annotation Tool

Nomi L. Harris, Suzanna E. Lewis, Mark Gibson, Colin Wiel, John Richter

Berkeley Drosophila Genome Project

Stephen M.J. Searle, Michele E. Clamp

The Welcome Trust Sanger Institute and the European Bioinformatics Institute

Apollo is an Open Source genome annotation viewer and editor. It was developed as a collaboration between the Berkeley Drosophila Genome Project (part of the FlyBase consortium) and The Sanger Institute in Cambridge, UK. Apollo allows researchers to explore genomic annotations at many levels of detail, and to perform expert annotation curation, all in a graphical environment. It is being used by the FlyBase biologists to make the final annotations on the finished *Drosophila melanogaster* genome, and will also be the primary vehicle for sharing these annotations with the community.

An increasing number of research groups are using Apollo as a starting point for customizing their own annotation visualization tool. Because Apollo was developed from the beginning to serve the needs of two groups working on different organisms (fruitfly and human) with different types of data, it was specifically designed to be flexible and extensible. The Generic Model Organism Database (GMOD) project, which aims to provide a complete ready-to-use toolkit for analyzing whole genomes, has adopted Apollo as its annotation workbench.

Apollo can read annotation data in a variety of formats (including GAME XML, GFF, and GenBank) via CGI, CORBA, and flat files, and the data adapter interface makes it straightforward to add new formats. The "look and feel" of the display is also highly configurable. For example, rows of data (known as "tiers") can be moved with the mouse; the threshold for displaying results can be changed; and the colors of each result can be customized, all from within Apollo.

Apollo is a powerful annotation curation workbench, simplifying the task of biologists who are poring over thousands of computational results and deciding how to summarize all the relevant data into a complete and accurate description of the genome and, ultimately, the proteome. Gene transcripts are easily created by dragging computational results and dropping them in the annotation zone. Other tools within Apollo let the curators edit the annotations in many ways, including adding canned or customized comments. Validation of annotations is simplified by edge matching, start/stop codons, and splice site detection. The new Synteny Viewer allows visualization of cross-genome comparisons.

Apollo is available at SourceForge: http://sourceforge.net/projects/gmod/. Like all gmod components, it is distributed under the terms of the Artistic License.

BioMOBY

Mark Wilkinson, Plant Biotechnology Institute, NRC Canada

August 2, 11:10-11:35

The BioMOBY project will generate a web services registry for biological data. Two main components are required: 1: a defined set of approximately 200-250 lightweight XSD templates describing basic biological data objects, 2: a central registry, "MOBY-Central", which stores the URL and protocol of a web service, as well as the input and output objects that the service accepts and generates. At present it is planned that MOBY-Central, upon client request, will generate a WSDL service specification on-the-fly and return this to the Client. The Client then transacts the service directly from the service provider. Such a system differs in several ways from the UDDI model; it simplifies participation of Servers by assuming the task of WSDL document creation, but most importantly it is "object driven" in that the Client may request that MOBY-Central report all services that can accept the object in-hand as input. This enables the dynamic discovery of information without necessitating prior Client (user) knowledge of the existence or location of this information.

A similar but independent project, OmniGene, stands as proof of concept for the BioMOBY proposals and the OmniGene and BioMOBY developers are now working together to create the BioMOBY system. No license agreement has yet been decided among the BioMOBY project participants, however it is likely that BioMOBY will be released under the same license as Perl itself.

BioSQL

Chris Mungall, Berkeley Drosophila Genome Project

August 2, 11:35-12:00

The BioSQL project defines a general bioinformatics schema intended for use in multiple projects across different database management systems and programming languages. Adapters exist for Perl, java, python and ruby. These adapters provide an API for accessing a BioSQL database via that languages bio-project object model. The BioSQL schema and adapter code works with both MySQL and postgres, and should be easily adaptable to other DBMSs e.g. Oracle.

BioSQL takes a modular approach to schemas. The existing core modules deal primarily with sequence annotation data. There is also a module for controlled vocabularies and ontologies, and modules can be added for dealing with other data classes, e.g. expression data.

The BioSQL schema is designed to be extensible, individual projects can extend the data model via controlled property tables, whilst still conforming to the core schema.

Open Source Bioinformatics and Systems Biology

Dan Kilburn, Beyond Genomics

August 2, Lunch talk

A new phase of the work in the field of bioinformatics has begun: the task of elucidating how living biological systems function. Identifying how genes function and how their products interact with other molecules occurs within the context of processes, location and their assumed roles. This places new demands for representing and handling forms of complex information, including pathways and causal models in biology, and places new demands on bioinformatics scientists to bridge the gape between computation and biology. Annotation and testing of models of biological systems requires the use of knowledge representations e.g., ontologies, which can links to large volumes of diverse experimental data. Beyond Genomics is developing a systems biology approach for integrating and analyzing information from multiple platforms called BioSystematics™ that integrates three parallel data producing platforms: MS/MS proteomics, MS/NMR metabolomics, and microarray transcriptomics. This enables us to build causal and dynamic models for normal and diseased systems, capturing not only sequence/structural information, but also process-oriented information relevant to biochemical systems. An initial challenge we face is to build an infrastructure utilizing scalable and distributed solutions that incorporate a number of novel bioinformatics technologies, and which use open source solutions and standards to simplify challenges of data integration and interchange.

MAGEstk - the MAGE software toolkit

Jason Stewart, OpenInformatics.com and OpenInformatics.org

August 2, 1:50-2:15

Location: http://mged.sf.net/

License: MIT

The MAGE software toolkit (MAGEstk) project has created a software infrastructure we believe will be useful to many biologists. It was originally created to provide a complete software API to the MicroArray Gene Expression Object Model (MAGE-OM) developed by the OMG (omg.org) in collaboration with MGED (mged.org). Since it's inception, MAGEstk has evolved into a powerful generic tool for automatically generating a broad informatics infrastructure from a UML object model.

When provided with a UML data model the MAGEstk tools are able to automatically generate the following:

1. A software API in Java, Perl, C++, and (soon) Python
2. An XML markup language to transmit data, defined by a DTD
3. software for serialization of data objects to XML and de-serialization XML into a relational DB schema in SQL to store the data objects persistently
4. (soon) the code for serializing objects to the DB and de-serializing data from the DB into objects

By providing such a useful set of infrastructure so easily, many informatics projects will be able to get started quickly without getting bogged down re-implementing basic infrastructure. Since it requires a UML data model, this forces biologists to spend time developing a good model for the data, and not on software development. This should help improve the overall quality of the final system. We believe that MAGEstk will prove itself especially useful for enabling informatics projects to communicate data using emerging WWW services protocols such as MOBY.

The FreeBSD-bio porting project

Johann Visagie, Electric Genetics

August 2, Lightning talk

FreeBSD is a free and open source operating system based on 4.4BSD, the final version of Berkeley's academic operating system which formed the basis for the development of much of the technologies underlying the Internet, including the initial incorporation of TCP/IP into a Unix operating system. As such it has a long and widely published academic track record, and over this long history it has developed the sort of benchmark-stretching performance and rock-solid stability required of a Bioinformatics server.

FreeBSD provides a third party application infrastructure known as the "ports collection". Going beyond most of the various binary packaging systems found in commercial and open source Unices, FreeBSD's ports provide an integrated infrastructure for downloading, patching, building and installing applications from source code, as well as for the further maintenance, upgrading and (if necessary) removal of such applications once installed.

The ports system extends open source principles from development to system administration; it allows the experienced user to share his experience in compiling and installing and configuring a particular application on FreeBSD, and it allows the novice user to emulate this experience with a single command.

The FreeBSD-bio project is a (highly) informal group of volunteers - co-operating via a mailing list - who share the goal of committing a large variety of commonly used bioinformatics tools and packages to the FreeBSD ports collection's CVS tree. Since FreeBSD ports are packaged in a pre-compiled binary format on FreeBSD distribution CDs and DVDs, this will eventually allow even the novice end user to set up a fully functional bioinformatics server "out of the box" with minimal effort.

URL (mailing list): http://www.plig.net/mailman/listinfo/freebsd-bio/

License: BSD, of course. :-)

A controlled vocabulary for gene expression

Johann Visagie[1], Janet Kelso[2], Soraya Bardien-Kruger[2], Alan Christoffels[2], Tania Hide[1], Winston Hide[2]

[1] Electric Genetics (Pty) Ltd
[2] South African National Bioinformatics Institute

August 2, Lightning talk

Electric Genetics and SANBI have developed a tool that integrates transcript information, genomic sequence, genetic mapping information and a standardized controlled vocabulary to serve as the basis for a more complex system that will aid in the identification of disease genes candidates.

A controlled vocabulary which consists of a predefined orthogonal set of hierarchical schemas was constructed. These currently include schemas containing terms describing anatomical site, cell type, developmental stage and pathology.

Expression data was mapped to the vocabulary by associating 6937 individual cDNA libraries (including dbEST and SAGE) with one or more terms in as many of these schemas as possible.

The vocabulary was implemented as a relational database schema. A Python API was constructed to provide a number of facilities to mine the vocabulary database, including a parser for a simple boolean query language.

The controlled vocabulary demonstrates that mapping expressed sequences to terms in a number of separate hierarchical schemas allows detailed mining of gene expression state information.

Current ongoing work includes the investigation of the utility of DAS v1.5 to serve as a query front-end for the controlled vocabulary.

LumberJack

Carolyn J. Lawrence [1], R. Kelly Dawe [1&2], Russell L. Malmberg [1]

Departments of [1] Plant Biology and [2] Genetics, University of Georgia

August 2, Lightning talk

LumberJack is a ML heuristic search tool written in JAVA that progressively jackknifes windows from a sequence alignment to generate multiple neighbor joining trees. It compares the trees statistically on the basis of their relative likelihood scores.

This allows the program not only to identify reasonable phylogenetic trees quickly, but also to map phylogenetic signal onto the alignment.

Reviewers' Note: The non-open source licensing restrictions were removed after submission of the abstract.

Analyzing cDNA microarray data using Python and the C clustering library: Why scripts are better than GUIs.

Michiel de Hoon, Seiya Imoto, Satoru Miyano

Laboratory of DNA Information Analysis, Human Genome Center, Institute of Medical Science, University of Tokyo

August 2, Lightning talk

Gene expression data generated in cDNA microarray experiments are commonly analyzed by clustering methods using GUI-based codes such as Cluster/TreeView program (Eisen, 1999) and GeneCluster (Tamayo, 1999). While the former is open source, the latter is not.

Whereas a GUI makes an analysis tools easier to use initially, analysis software based on scripting languages such as Python have several advantages. Script-based codes are generally more flexible and easier to develop. In addition, scripting languages usually offer a range of useful features, such as text processing capabilities, database access, and plotting routines that would have to be programmed from scratch in a GUI-based code. Numerical routines that would be too slow if implemented in a scripting language directly, can be written in a lower-level language such as C and Fortran. These routines can then be called from the script. Finally, we note that GUI-based codes usually rely on a specific (often commercial) compiler, which makes porting code, open-source development and code improvement very difficult.

While script-based code is already commonly being used for sequence analysis (e.g., BioPerl and BioPython), analysis of gene expression data is still dominated by GUI-based codes. We have therefore written a C library of routines for hierarchical clustering and for Self-Organizing Maps, as well as an advanced k-means clustering algorithm. This clustering library can be called from Python, or linked to other codes. Unlike Cluster/TreeView, no licensed code was used to develop this library, making the C clustering library truly open source. The library can be compiled with the GNU C compiler.

The C clustering library is freely available for academic and non-commercial use at

http://bonsai.ims.u-tokyo.ac.jp/~mdehoon/software/software.html

Contact: mdehoon@ims.u-tokyo.ac.jp

Schedule::Depend

Steven Lembark, Workhorse Computing

August 2, Lightning talk

One of the more challenging areas of BioInformatics is simply getting computer work organized. The challenge is complicated by competing requirements for speed, organization and dynamic schedules. Most existing tools are ill-suited for a dynamic parallel scheduling of interactive jobs. Traditional tools such as cron are generally too static, make syntax for handling dynamic job lists is byzantine at best, and the commercial packages are frequently too complicated to handle dynamic schedules easily.

Schedule::Depend is a OO Perl module designed for specifically this kind of task. It uses a simple syntax of colon-separated dependencies (pretty much like make's) with the individual job tags being passed through an "unalias" method. The default method provided can handle object or class methods, subroutine references or Perl blocks along with shell paths. This avoids having to wrap everything in shell code, and simplifies the syntax enormously. The simplified syntax makes generating dynamic schedules far simpler than with other tools. The module provides for job tracking, restarting (i.e., skipping previously completed tasks) and continuing after an aborted job (much like "make -k").

As an example I'd bring in a working schedule used in a BioInfo. lab or schedule from update_wormbase.

The module itself is on CPAN, released under the same terms as Perl5.

Biological Information Integration Toolkit

Jeremy Praissman, Dawei Lin, John Rose, Bi-Cheng Wang

Department of Biochemistry and Molecular Biology, University of Georgia

August 2, Lightning talk

Rapid progress of genome sequencing projects and the NIH protein structure initiative has brought about an increased focus on biological data integration and computationally guided experimentation. As part of the effort to develop high throughput structure determination technologies, it is crucial to integrate and perform large scale analysis of interdisciplinary data. While many tools exist for parsing biological databases and interfacing with existing analysis programs, there is currently a lack of support infrastructure for correlating biological information from disparate sources. Our toolkit is an attempt to address this need using mathematical set and combinatorial graph techniques.

The toolkit, which is implemented in Perl and uses some Bioperl modules, has already been applied in analyzing neighbour gene relatedness for complete bacterial genomes released at NCBI. It will be extended to support further correlation of genes with other genomic, structural and functional information.

Current features:

calculating basic genome statistics
calculating statistics for genes grouped by intergenic distance
parsing and classifying functional annotation
analyzing annotation within sets of genes grouped by intergenic distance
generating user specified sequence fragments

Planned features:

subset generating operations on gene sets using user supplied functions
generic filtering iterators
combinatorial data structures for further correlating biological information

NBLAST: a cluster variant of BLAST for NxN comparisons

Michel Dumontier, Samuel Lunenfeld Research Institute, Mt. Sinai Hospital, Toronto, ON Canada M5G 1X5

August 2, Lightning talk

The BLAST algorithm compares biological sequences to one another in order to determine shared motifs and common ancestry. However, the comparison of all non-redundant (NR) sequences against all other NR sequences is a computationally intensive task. We developed NBLAST as a cluster computer implementation of the BLAST family of sequence comparison programs for the purpose of generating pre-computed BLAST alignments and neighbour lists of NR sequences.

NBLAST performs the heuristic BLAST algorithm and generates an exhaustive database of alignments, but it only computes alignments (i.e. the upper triangle) of a possible N⊃2 alignments, where N is the set of all sequences to be compared. A task-partitioning algorithm allows for cluster computing across all cluster nodes and the NBLAST master process produces a BLAST sequence alignment database and a list of sequence neighbours for each sequence record. The resulting sequence alignment and neighbour databases are used to serve the SeqHound query system through a C/C++ and PERL Application Programming Interface.

NBLAST offers a local alternative to the NCBI's remote Entrez system for pre-computed BLAST alignments and neighbour queries. On our 216-processor 450 MHz PIII cluster, NBLAST requires ~24 hrs to compute neighbours for 850000 proteins currently in the non-redundant protein database. NBLAST source code and binaries are available at http://sourceforge.net/projects/slritools and the NBLAST article is freely available at BioMed Central Bioinformatics http://www.biomedcentral.com/1471-2105/3/13/.

OBF Simple Flat-file indexing

Andrew Dalke, Dalke Scientific Software

August 2, Lightning talk

Most bioinformatics data is available as a flat-file. Many labs only need simple retrieval of the text of a record in the file given some identifier, like the record number, but don't need the extra overhead of setting up database management system.

As part of the Open Bioinformatics Foundation biohackathon, the Biopython, Bioperl, BioJava, and C coders got together to define and implement for those platforms a standard, cross-platform indexing scheme. Given an identifier and optional namespace (eg, to distinguish between the entry id and the accession name) it works like a lookup table to return the list of matching locations (filename and byte range in the file).

The spec defines two types of indexing schemes. The simplest uses a flat files to store the index and is for labs which don't want to install any extra software and are willing to trade off some space and ease of modification. The other fixes those limitations but requires that Sleepycat's BerkeleyDB be installed on the local machine. The choice is up to the user and the indexing type can be automatically determined so is invisible to software using the index.

I will describe the design of the indexing system, how it can be used, and show interoperability between the different language implementations.

Bibliographic Query Service in Bioperl

Martin Senger, European Bioinformatics Institute

August 2, Lightning talk

Bioperl has now modules that can access bibliographic repositories. They allow to query a bibliographic database (such as MEDLINE) and then to retrieve resulting citations from it. The citations are returned in an XML format which is native to the repository but there are also supporting modules for converting them into Perl objects.

Proposed OMG/LSR Effort for Modeling Domain Objects

Scott Markel, LION Bioscience Inc.

August 2, Lightning talk

The domain portion of the OMG's Biomolecular Sequence Analysis specification has been largely superceded by open source efforts, e.g., BioJava and BioCORBA. The OMG's Life Sciences Research (LSR) Domain Task Force is considering an RFP for a domain model based on these open source objects. This approach leverages OMG's Model Driven Architecture (MDA). I'm looking for feedback on the idea and for participants.

myGrid--personalised biological services on the Grid

Philip Lord (p.lord@russet.org.uk)

August 2, Lightning talk

University of Manchester, and the Mygrid consortium (which is University of Manchester, University of Newcastle, University of Nottingham, University of Sheffield, University of Southampton, IT Innovation Centre, European Bioinformatics Institute)

MyGrid is a Grid project aimed at biologists. We aim to facilitate resource discovery and composition, and to support personalisation, and gathering of provenance of data.

http://www.mygrid.org.uk/

Bio::Structure: manipulating structure related (PDB) data in BioPerl

Kris Boulez, AlgoNomics NV

August 2, Lightning talk

Analysis of (protein) structure data will become more and more important in bioinformatics in the future. The Bio::Structure class of modules provides this functionality for BioPerl. The way the code originated will be presented as an aid for future module writers. Example usage will be discussed. Shortcomings of the present code and ideas for future extensions will be shown.

Eponine: machine learning about your favourite sequences

Thomas Down, Wellcome Trust Sanger Institute

August 2, Lightning talk

Eponine is an implementation of the Relavance Vector Machine, a statistical method which can combine an arbitrary set of 'sensor' functions to build the simplest possible classification model. It has been applied successfully to a number of sequence analysis problems, including transcription start site prediction.

The code is LGPL. http://www.sanger.ac.uk/Software/analysis/eponine/

The Generic Feature Database

Shawn Hoon, Fugu Genome Project, Institute Of Molecular and Cell Biology, Singapore

August 2, Lightning talk

The GFD that our group have developed is a flexible scheme for which to store results of analysis. It is made up of Entities and Relationships through which more complex biological objects can be made through an aggregation of these objects.

EMBOSS

Jon Ison, HGMP, MRC

August 2, Lightning talk

EMBOSS is a free, Open Source package of software tools developed for the specific needs of the molecular biology community. EMBOSS includes over 200 applications for sequence analysis and a variety of other computatational tasks, and integrates a range of other packages into a consistent whole. EMBOSS also includes extensive C programming libraries, making it a platform to allow other scientists to develop and release software in the true open source spirit.