14th Annual Bioinformatics Open Source Conference

BOSC 2013

Berlin, Germany
July 19-20, 2013

http://www.open-bio.org/wiki/BOSC_2013

Welcome to BOSC 2013! The Bioinformatics Open Source Conference, established in 2000, is held every year as a Special Interest Group (SIG) meeting in conjunction with the Intelligent Systems for Molecular Biology (ISMB) Conference.

BOSC is sponsored by the Open Bioinformatics Foundation (O|B|F), a non-profit group dedicated to promoting the practice and philosophy of Open Source software development within the biological research community.

This year's keynote speakers are Sean Eddy and Cameron Neylon. Sean Eddy, a group leader at the Howard Hughes Medical Institute's Janelia Farm, is the author of several well-known open source computational tools for sequence analysis including the HMMER and Infernal software suites, as well as a coauthor of the Pfam database of protein domains. Cameron Neylon is Advocacy Director for the Public Library of Science, a research biophysicist and well-known agitator for opening up the process of research. He speaks regularly on issues of Open Science including Open Access publication, Open Data, and Open Source.

Session topics this year include Software Interoperability, Visualization, Cloud and Genome-Scale Computing, and several other topics that previous BOSC attendees will recognize. New this year is a session on Open Science and Reproducible Research, which will end with a short talk entitled “Ten Simple Rules for the Open Development of Scientific Software” that will be followed by time for discussion by the audience. Our panel discussion on Day 2, Strategies for Funding and Maintaining Open Source Software, will include experts on various funding approaches ranging from grant-based to commercial with value-added services.

There are three scheduled poster sessions. We have space for several last-minute posters in addition to those listed in the program.

Thanks in part to generous support from Eagle Genomics, we were able to award Student Fellowships to the authors of the three best student abstracts. Congratulations to the student winners, all of whom received free admission to BOSC and $250 towards their travel expenses: Markus List, Joeri van der Velde, and Yuriy Vaskin.

BOSC is a community effort—we thank all those who made it possible, including the organizing committee, the program committee, the session chairs, and the ISMB SIG chair, Steven Leard. If you are interested in helping to organize BOSC 2014, please email bosc@open-bio.org.

2013 Organizing Committee:

Nomi Harris (Chair), Jan Aerts, Brad Chapman, Peter Cock, Christopher Fields, Jeremy Goecks, Hans-Rudolf Hotz, Hilmar Lapp

2013 Program Committee:

Heikki Lehväslaiho, Hans-Rudolf Hotz, Tiago Antão, Brad Chapman, Thomas Down, Peter Cock, Francesco Strozzi, Hilmar Lapp, Jeremy Goecks, Ben Temperton, Jan Aerts, Chris Fields, Shiran Pasternak, Kam Dahlquist, Kazuharu Arakawa, Scott Markel, Michael Reich, Timothy Booth, Sophia Cheng, Heiko Dietze, Hervé Ménager, Peter Robinson, Olivier Sallou, Raoul Bonnal, Imtiaz Khan, Ronald Taylor, Nomi Harris

BOSC 2013 Schedule

Day 1 (Friday, July 19, 2013)

Time	Title	Speaker or Session Chair
7:30-9:00	**Registration**	
9:00-9:15	**Introduction and Welcome**	Nomi Harris (Chair, BOSC 2013)
9:15-10:15	**Keynote: Network ready research--the role of open source and open thinking**	Cameron Neylon
10:15-10:45	***Coffee Break***	
10:45-12:30	**Session: Open Science**	Chair: Hilmar Lapp
10:45-11:00	Open Science Data Framework: A Cloud enabled system to store, access, and analyze scientific data	Anup Mahurkar
11:00-11:15	myExperiment Research Objects: Beyond Workflows and Packs	Stian Soiland-Reyes
11:15-11:30	Empowering Cancer Research Through Open Development	Juli Klemm
11:30-11:45	DNAdigest - a not-for-profit organisation to promote and enable open-access sharing of genomics data	Fiona Nielsen
11:45-11:50	Jug: Reproducible Research in Python	Luis Pedro Coelho
11:50-11:55	OpenLabFramework: A Next-Generation Open-Source Laboratory Information Management System for Efficient Sample Tracking	Markus List
12:00-12:30	Ten Simple Rules for the Open Development of Scientific Software [discussion]	Andreas Prlic
12:30-1:30	***Lunch***	
1:00-2:00	**Poster Session I**	
2:00-3:30	**Session: Visualization**	Chair: Jan Aerts
2:00-2:25	Refinery Platform - Integrating Visualization and Analysis of Large-Scale Biological Data	Nils Gehlenborg
2:25-2:40	MetaSee: An interactive visualization toolbox for metagenomic sample analysis and comparison	Kang Ning
2:40-2:55	DGE-Vis: Visualisation of RNA-seq data for Differential Gene Expression analysis	David Powell
2:55-3:10	Genomic Visualization Everywhere with Dalliance	Thomas Down
3:10-3:25	Robust quality control of Next Generation Sequencing alignment data	Konstantin Okonechnikov
3:25-3:30	Visualizing bacterial sequencing data with GenomeView	Thomas Abeel
3:30-4:00	***Coffee Break***	

Time	Title	Speaker or Session Chair
4:00-5:30	**Session: Bioinformatics Open Source Project Updates**	Chair: Hans-Rudolf Hotz
4:00-4:15	BioRuby project updates - power of modularity in the community-based open source development model	Toshiaki Katayama
4:15-4:30	Biopython project update	Peter Cock
4:30-4:45	InterMine - Collaborative Data Mining	Alex Kalderimis
4:45-5:00	GenoCAD 2.2 Grammar Editor	Jean Peccoud
5:00-5:15	Improvements and new features in the 7th major release of the Bio-Linux distro	Timothy Booth
5:15-5:20	**Announcements**	Nomi Harris
5:20-6:30	**Poster Session II**	
5:20-6:30	**BOFs**	
7:00	Pay-your-own-way BOSC dinner, Hendrik's (www.hendriks-berlin.de), Straße des 17. Juni 13, 10623 Berlin. RSVP at bit.ly/BOSC2013-dinner	

Day 2 (Saturday, July 20, 2013)

Time	Title	Speaker or Session Chair
8:45-8:50	**Announcements**	Nomi Harris
8:50-9:00	Codefest 2013 Report	Brad Chapman (Codefest 2013 Organizer)
9:00-9:15	Open Bioinformatics Foundation: A Community For, By, and Of You	Hilmar Lapp (President, O|B|F)
9:15-10:15	**Keynote: Biological sequence analysis in the post-data era**	Sean Eddy
10:15-10:45	***Coffee Break***	
10:45-12:30	**Session: Software Interoperability**	Chair: Jeremy Goecks
10:45-11:10	BioBlend - Enabling Pipeline Dreams	Enis Afgan
11:10-11:35	Taverna Components: Semantically annotated and shareable units of functionality	Alan Williams
11:35-11:50	UGENE Workflow Designer – flexible control and extension of pipelines with scripts	Yuriy Vaskin
11:50-12:05	Reproducible Quantitative Transcriptome Analysis with Oqtans	Vipin T. Sreedharan
12:05-12:10	PhyloCommons: community storage, annotation and reuse of phylogenies	Hilmar Lapp
12:10-12:15	GEMBASSY: an EMBOSS associated package for genome analysis using G-language SOAP/REST web services	Hidetoshi Itaya

Time	Title	Speaker or Session Chair
12:15-12:30	Rubra - flexible distributed pipelines for bioinformatics	Clare Sloggett
12:30-1:30	***Lunch***	
12:30-1:30	**Poster Session III**	
1:30-3:30	**Session: Cloud and Genome-Scale Computing**	Chair: Peter Cock
1:30-1:45	Towards Enabling Big Data and Federated Computing in the Cloud	Enis Afgan
1:45-2:00	MyGene.info: Making Elastic and Extensible Gene-centric Web Services	Chunlei Wu
2:00-2:15	An update on the Seal Hadoop-based sequence processing toolbox	Luca Pireddu
2:15-2:30	Open Source Configuration of Bioinformatics Infrastructure	John Chilton
2:30-2:55	An Open Source Framework for Gene Prioritization	Hoan Nguyen
2:55-3:10	RAMPART: an automated de novo assembly pipeline	Daniel Mapleson
3:10-3:30	OmicsConnect: flexible multi-omics data capture and integration tools for high-throughput biology	Joeri van der Velde
3:30-4:00	***Coffee Break***	
4:00-4:40	**Session: Translational Genomics**	Chair: Nomi Harris
4:00-4:25	Community development of human variant calling and validation pipelines	Brad Chapman
4:25-4:40	Understanding Cancer Genomes Using Galaxy	Jeremy Goecks
4:40-5:30	**Panel: Strategies for Funding and Maintaining Open Source Software**	*Moderator*: Brad Chapman *Panelists*: Peter Cock, Sean Eddy, Carole Goble, Scott Markel, Jean Peccoud
5:30-5:40	Presentation of Student Travel Awards	Nomi Harris
5:40-6:40	**BOFs**	

Any last-minute schedule updates will be posted at

http://www.open-bio.org/wiki/BOSC_2013_Schedule

Keynote Speakers

Sean Eddy

Sean Eddy is a group leader at the Howard Hughes Medical Institute's Janelia Farm. He is interested in deciphering the evolutionary history of life by comparison of genomic DNA sequences. His expertise is in the development of computational algorithms and software tools for biological sequence analysis. He is the author of several computational tools for sequence analysis including the HMMER and Infernal software suites, as well as a coauthor of the Pfam database of protein domains. He serves as an advisor to several foundations and US science agencies, including the National Institutes of Health and the National Academy of Sciences, often on matters of large-scale computation and data analysis in biology.

Sean's talk is entitled *Biological sequence analysis in the post-data era*.

> Biological systems are almost unfathomably complex, yet their complexity is reproducibly specified by a small digital genome. We understand many basics of development and evolution but we lack a truly satisfying quantitative understanding of how biological complexity is specified and how it evolves. One important line of attack on the problem is to reconstruct the history of molecular evolution by comparative genome sequence analysis. Biological sequence comparison has a long intellectual history, but only recently, with the advent of inexpensive large scale DNA sequencing, have we gained comprehensive access to genome sequences from essentially all species. Though welcome, this influx of genome sequence data is exposing structural flaws in computational biology research tools. Because the research community values innovative science over infrastructure in any short-term decision, academic researchers have difficulty investing sufficient effort in robust software and datasets that may enable even more innovative science over the long term. Meanwhile, professional commercialization of the software and data infrastructure also continues to prove difficult, in part because open source code and data availability is a fundamental principle of scientific publication of reproducible, reusable results. I'll discuss what I see as some of the key tensions, challenges, and opportunities in these regards, in part in the context of our work at Janelia Farm on the HMMER and Infernal codebases, and our nascent work on the genomic specification of neural circuits in Drosophila.

Cameron Neylon

Cameron Neylon is Advocacy Director for the Public Library of Science, a research biophysicist and a well-known agitator for opening up the process of research. He speaks regularly on issues of Open Science including Open Access publication, Open Data, and Open Source as well as the wider technical and social issues of applying the opportunities the internet brings to the practice of science. He was named as a SPARC Innovator in July 2010 for work on the Panton Principles and is a recipient of the Blue Obelisk for contributions to open data. He writes regularly at his blog, Science in the Open.

Cameron will speak about *Network ready research: The role of open source and open thinking*.

> The highest principle of network architecture design is interoperability. If Metcalfe's Law tells us that a network's value can scale as some exponent of the number of connections then our job in building networks is to ensure that those connections are as numerous, as operational, and as easy to create as possible. Where we make it easy for anyone to wire in new connections we maximise the ability of others to contribute to the value of our shared networks.
>
> Bioinformatics has, from time to time, been derided as "slidedecks full of hairballs", yet those hairballs, and their ubiquity are emblematic of the fact that at its heart bioinformatics is a science of networks. Networks of physical interactions, of genetic control, of degree of similarity, or of ecological interactions amongst many others. Bioinformatics is also amongst the most networked of research communities and amongst the most open in the

sharing of research papers, of research data, tools, and even research in process in online conversations and writing.

Lifting our gaze from the networks we work on to the networks we occupy is a challenge. Our human networks are messy and contingent and our machine networks clogged with things we can't use, even if we could access them. What principles can we apply so as to build our research into networks that make the most of the network infrastructure we have around us. Where are the pitfalls, and where are the opportunities? What will it take to configure our work so as to enable "network ready research"?

O|B|F Membership

Professionals, scientists, students, and others active in the Open Source Software arena in the life sciences are invited to join the Open Bioinformatics Foundation (the O|B|F). The O|B|F grew out of the volunteer projects BioPerl, BioJava and Biopython and was formally incorporated in 2001 in order to handle modest requirements of hardware ownership, domain name management and funding for conferences and workshops. In 2005, we enacted bylaws for the first time, and along with it created a formal membership.

In 2012, we decided to give up our own incorporation to associate ourselves with Software In The Public Interest, Inc., a fiscal sponsorship organization that we felt aligned well with our own values and culture. The bylaws underwent a series of changes, in part to better reflect our current practices, and in part to pave the way for joining SPI. The changes were approved on Sep 11, 2012, our membership overwhelmingly approved of associating with SPI, and as of October 12, 2012, O|B|F is an SPI-associated project.

This program includes a form you can fill out to join the O|B|F. (Yes, we realize that a paper form is kind of retro, but at the moment, this is the way we meet the requirements for documenting our membership.) Also, f you are interested in meeting and talking to some of the O|B|F Directors and members, please join us at the BOSC dinner (see below).

Talk and Poster Abstracts

Talk abstracts are included in this program in the order in which they will be presented at the conference. Some, but not all, of the talks will also be presented as posters. There are also a few spaces available for last-minute posters. If you would like to present one, please email your abstract (which must meet the BOSC criteria of available source and recognized open source license) to bosc@open-bio.org.

Authors should put up their posters in their assigned poster spot before the first poster session (which starts at 12:30 on the first day). After that time, any unused poster slots will be made available for last-minute posters. The ISMB staff specify that posters should not exceed the following dimensions: 0.95 m wide x 1.30 m high.

Optional BOSC Dinner

We invite you to join BOSC organizers and attendees at a pay-your-own-way dinner the first evening of BOSC (Friday, July 19, at 7pm) at Hendrik's (www.hendriks-berlin.de), Straße des 17. Juni 131, 10623 Berlin. Take the S-Bahn S5 or S7 line from the Westkreuz station to Tiergarten and walk back 100m (across Straße des 17. Juni). The restaurant is under the railway arches.
If you want to join us for dinner, RSVP at http://bit.ly/BOSC2013-dinner before Friday at noon. The restaurant has space for 30 BOSC guests; only those who RSVP will be admitted.

Talks and Posters

Title	Author	Poster #
Open Science Data Framework: A Cloud enabled system to store, access, and analyze scientific data	Anup Mahurkar	
myExperiment Research Objects: Beyond Workflows and Packs	Stian Soiland-Reyes	1
Empowering Cancer Research Through Open Development	Juli Klemm	
DNAdigest - a not-for-profit organisation to promote and enable open-access sharing of genomics data	Fiona Nielsen	2
Jug: Reproducible Research in Python	Luis Pedro Coelho	3
OpenLabFramework: A Next-Generation Open-Source Laboratory Information Management System for Efficient Sample Tracking	Markus List	4
Ten Simple Rules for the Open Development of Scientific Software	Andreas Prlic	
Refinery Platform - Integrating Visualization and Analysis of Large-Scale Biological Data	Nils Gehlenborg	5
MetaSee: An interactive visualization toolbox for metagenomic sample analysis and comparison	Kang Ning	6
DGE-Vis: Visualisation of RNA-seq data for Differential Gene Expression analysis	David Powell	
Genomic Visualization Everywhere with Dalliance	Thomas Down	
Robust quality control of Next Generation Sequencing alignment data	Konstantin Okonechnikov	7
Visualizing bacterial sequencing data with GenomeView	Thomas Abeel	8
BioRuby project updates - power of modularity in the community-based open source development model	Toshiaki Katayama	
Biopython project update	Peter Cock	
InterMine - Collaborative Data Mining	Alex Kalderimis	
GenoCAD 2.2 Grammar Editor	Jean Peccoud	9
Improvements and new features in the 7th major release of the Bio-Linux distro	Timothy Booth	10
BioBlend - Enabling Pipeline Dreams	Enis Afgan	
Taverna Components: Semantically annotated and shareable units of functionality	Alan Williams	11
UGENE Workflow Designer – flexible control and extension of pipelines with scripts	Yuriy Vaskin	
Reproducible Quantitative Transcriptome Analysis with Oqtans	Vipin T. Sreedharan	12
PhyloCommons: community storage, annotation and reuse of phylogenies	Hilmar Lapp	
GEMBASSY: an EMBOSS associated package for genome analysis using G-language SOAP/REST web services	Kazuharu Arakawa	
Rubra - flexible distributed pipelines for bioinformatics	Clare Sloggett	13
Towards Enabling Big Data and Federated Computing in the Cloud	Enis Afgan	
MyGene.info: Making Elastic and Extensible Gene-centric Web	Chunlei Wu	14

O|B|F – Open Bioinformatics Foundation

Membership Application

I wish to apply for membership in the Open Bioinformatics Foundation (O|B|F).

First and Last Name: ______________________________

Street Address: ______________________________

City, State, Zip Code: ______________________________

Country of Residence: ______________________________

Email Address: ______________________________

All fields are mandatory. The O|B|F will treat all personal information as strictly confidential and will not share personal information with anyone except members of the O|B|F Board of Directors, or entities or persons appointed by the Board to administer membership communication. This may be subject to change; please see below.

I am an attendee of BOSC 201___: □ Yes □ No

If you answered No, please state why you meet the membership eligibility requirement of being interested in the objectives of the O|B|F:

(Use back of page if you need more space)

I understand that membership rights and duties are laid down in the O|B|F Bylaws which may be downloaded from the O|B|F homepage at http://www.open-bio.org/. I understand that if the O|B|F's privacy statement changes I will be notified at my email address (as known to O|B|F), and if I do not express disagreement with the proposed change(s) by terminating my membership within 10 days of receipt of the notification, I consent to the change(s).

____

Signature

Talk and Poster Abstracts

In the pages that follow, talk abstracts appear in the order in which the talks will be presented. Some authors will also present their work as posters. Those abstracts have a poster number at the bottom of the page. Poster-only abstracts appear after the talk abstracts.

Open Science Data Framework: A Cloud enabled system to store, access, and analyze scientific data

Anup Mahurkar[1], Victor Felix[1], Jonathan Crabtree[1], Michael Schor[1], Owen White[1]

[1]University of Maryland School of Medicine, Institute for Genome Sciences

Project site: osdf.igs.umaryland.edu

Code: http://sourceforge.net/projects/osdf/

License: GNU GPL v2

We have developed the Open Science Data Framework (OSDF), a Cloud-enabled system designed to provide a high-performance, scalable data storage service for scientific data including genomic sequence data and associated results. OSDF consists of a database, a data exchange format, a web UI for browsing, and an associated API that supports data retrieval and submissions from the community.

Although conceived for hosting the NIH Common Fund supported Human Microbiome Project Data Analysis and Coordination Center (HMP-DACC), this system is designed to be flexible and would ensure that members of a research consortium storing common data are able to define domain specific data models that describe each element to be stored and their interrelationships. While designed with consortia and collaborative projects in mind the system is just as applicable for individual labs to manage their scientific data sets.

JavaScript Object Notation (JSON) is used for modeling data objects derived from a genome project, such as subjects, samples, sequence sets, assemblies, alignment to reference genomes, gene predictions, functional annotations or metabolic reconstructions. These JSON objects include a rich set of metadata tags organized by ontologies and stored as key-value pairs. The metadata for each object is modeled using pre-existing standards such as GO, MIGS, MIMS, and MIMARKS. When standards do not exist users can build their own custom CVs and ontologies. All the metadata associated with the scientific data sets are indexed using Apache Lucene to allow rapid search and retrieval. The node.js application is used to serve the JSON objects allowing for a high-performance scalable data server.

We implemented a generic RESTful API to access and place data, and intend to develop domain specific APIs where necessary. The scientific community can use the generic web UIs bundled with OSDF to browse and search for data or use the API for programmatic access.

A major portion of the Healthy Human Subjects Phase I data currently available at the DACC is now powered by OSDF. Currently the HMPDACC OSDF instance includes over 30,000 data elements or nodes that include over 14,000 sequence nodes, 6,000 sample nodes, 5,600 16S nodes, and over 1,000 reference genome nodes representing over 30 TBs of data on the disk.

Another design goal was to allow Cloud-based workflows to perform operations on the data hosted in OSDF. Towards that end we have used the Cloud Virtual Resource (CloVR) Virtual Machine to pull data from the DACC OSDF server and analyze the data in Amazon EC2 and Data Intensive Academic Grid clouds demonstrating the utility of this framework in Cloud-based analysis workflows.

We will present performance tests that demonstrate OSDF allows a single server to deliver large data files to multiple asynchronous requests along with the other features described above

myExperiment Research Objects: Beyond Workflows and Packs

Stian Soiland-Reyes[1], Don Cruickshank[2], Finn Bacall[1], Jun Zhao[2], Khalid Belhajjame[1], David De Roure[3], Carole A. Goble[1]

{soiland-reyes, finn.bacall, khalid.belhajjame, carole.goble}@cs.manchester.ac.uk
{donald.cruickshank, jun.zhao}@zoo.ox.ac.uk david.deroure@oerc.ox.ac.uk

[1] School of Computer Science, University of Manchester, UK
[2] Department of Zoology, University of Oxford, UK
[3] Oxford e-Research Centre, University of Oxford, UK

Web site:	http://www.myexperiment.org/	http://www.wf4ever-project.org/
Source code:	http://myexperiment.rubyforge.org/svn/branches/wf4ever/	https://github.com/wf4ever/
Licence:	BSD 3-Clause License	

myExperiment is a Virtual Research Environment for collaboration and sharing of experiments [1]. It allows users to share and publish scientific workflows from a number of workflow systems. People and third party applications can discover workflows published and shared on myExperiment; the workflows can then be reused and repurposed to satisfy new requirements.

As well as workflows, myExperiment supports the publication and sharing of another kind of digital object known as a myExperiment pack. Packs are collections of items such as workflows, example input data, results, and other files such as PowerPoint slides and PDF files of scientific papers. Packs can also contain links to data sets and services on the web. Since packs can also be the subject of sharing, tagging and discovery, they extend the application of myExperiment beyond workflows to any type of digital object associated with a scientific experiment involving computation.

In collaboration with the Wf4Ever project, myExperiment is now being enhanced to support richer forms of packs, known as Research Objects [2], with the objective of supporting open and reproducible research by sharing investigation methods and outcomes. These include common types of artifacts such as hypothesis, results, workflow, workflow runs, presentation, and documentation. Individual resources and the research objects have provenance records to detail their attribution and origin, in addition to annotations for comments, descriptions and expressing relationships.

The architecture for Research Objects is realized as a Linked Data platform of a RESTful web services that support preservation aspects such as decay monitoring and evolution tracking, and is presented to the user through a regular web interface on myExperiment. The developed Research Object data model has lead to further standardization and collaboration by the newly formed W3C Research Object for Scholarly Communication Community Group [3].

[1] Carole A. Goble, Jiten Bhagat, Sergejs Aleksejevs, Don Cruickshank, et al.: **myExperiment: a repository and social network for the sharing of bioinformatics workflows**. *Nucleic Acids Research (2010) 38 (suppl 2): W677-W682*. doi:10.1093/nar/gkq429
[2] Khalid Belhajjame, Oscar Corcho, Daniel Garijo, Jun Zhao, et al.: **Workflow-centric research objects: First class citizens in scholarly discourse**. *Proc. Workshop on the Semantic Publishing (SePublica)* 2012, pp. 1-12.
[3] **Research Object for Scholarly Communication Community Group** http://www.w3.org/community/rosc/

This work was enabled by the Wf4Ever project funded by the European Commission's 7th FWP (FP7-ICT-2007-6 270192), and the myGrid platform grant by the EPSRC (EP/G026238/1)

Poster 1

Empowering Cancer Research through Open Development

Authors: Juli D. Klemm[1], Robert Shirley[1], Lawrence Brem[2], Luis Ibanez[3], Brad King[3], Anthony R. Kerlavage[1], George A. Komatsoulis[1]
Author affiliations: [1]National Cancer Institute, Center for Biomedical Informatics and Information Technology; [2]SAIC-Frederick, Inc.; [3]Kitware, Inc. Presenting author email address: klemmj@mail.nih.gov.
URL for project website: https://github.com/ncip/
Code URL: https://github.com/ncip/
Open Source License: BSD 3-Clause

Evolving a collaborative development culture comes with many challenges. This is particularly true in government, where regulations and institutional tradition lead to highly structured practices. This talk will provide an overview of the process the National Cancer Informatics Program (NCIP) at the US National Cancer Institute's (NCI) Center for Biomedical Informatics and Information Technology (CBIIT) has undertaken to bring to cancer research the benefits of agility and rapid pace of innovation that open source development has to offer.

Starting in 2003, NCI deployed an informatics initiative to develop an interoperability infrastructure suitable to support both basic and clinical research activities. The initiative was crafted to follow the model of open source projects, and indeed, made all the developed resources openly available to the public under a reciprocal license. True open development, however, has been limited and most software enhancements have been implemented through centralized development teams with limited ability to react in an agile way to the immediate needs of cancer researchers.

To address the critical need, the NCIP has reached out to a larger community for input on how to bring a large catalog of cancer research software projects into a fully functional open source environment. Out of this review, a consensus has emerged on moving all the software projects hosted on government servers at NCI into GitHub, the online hosting service that has become the nucleus for many open source projects. In addition, a decision was made to simplify the existing licensing and to migrate to use of an Open Source Initiative (OSI) -approved license.

Acting on this consensus, the NCIP put together a migration plan to move approximately 50 projects, with about 100 associated repositories, from an internal NCI SVN repository to GitHub under a BSD-3 license. This talk will provide an overview of the NCIP philosophy on open-development, open-science, and reproducibility, as well as the process of this migration, the challenges encountered, the lessons learned, and the current status.

The migration, of course, is not the end of this process. Now that this valuable collection of software is more readily accessible, the next mission is to stimulate the growth of communities around the projects, which can become self-sustained, and thus ensure the long-term vitality of the projects. Each one of these projects is a rich opportunity for developers to work on "Stuff that Matters," and help NCI to pursue its mission of supporting and furthering cancer research.

Title
DNAdigest - a not-for-profit organisation to promote and enable open-access sharing of genomics data

Authors
Fiona G Nielsen

Author affiliations
DNAdigest.org, Cambridge, UK

URL for project web site
http://DNAdigest.org

URL for accessing code
-

The genomics revolution is already here: the techniques for researching and characterising genomics diseases is available to both researchers (next generation DNA sequencing) and the general public (in form of personal testing like 23andme), so we should soon be able to diagnose any genetic disease by sequencing a patients DNA. This is the glorified goal of research into all genetic diseases, including hereditary diseases and cancer.

However, while data output is flooding research centres around the world, and genomics results published in high-esteemed journals, the sharing of the data that enables this research is embarassingly limited. The data ownership, the legal consent of the patients involved, the privacy of the patients involved and the mere volume and complexity of these datasets are a major hindrance to sharing of personal genetics data. As a result, each research unit is currently maintaining their own 'silo' of potentially valuable sequence and patient data. Needless to say, there may be several big genetic discoveries "out there" already sequenced, but not discovered, because noone has had the means to bring together the matching pieces of the puzzle.

The technological means to solve this problem are already existing and available, but no solution has been proposed until now. We are founding DNAdigest as a not-for-profit enterprise to sufficiently engage all stakeholders including researchers and patients and address their concerns while maintaining the goal: the advancement of genomics research.

We will present the underlying idea and strategic plan for how DNAdigest will enable researchers to publish their genomics data in an online open-access fashion without compromising patient consent or data privacy.

Poster 2

Jug: Reproducible Research in Python

Luis Pedro Coelho <luis.pedro.coelho@embl.de>
European Molecular Biology Laboratory (EMBL)

Homepage: http://luispedro.org/software/jug
Repository: http://github.com/luispedro/jug
License: MIT

As computational pipelines become a bigger part of science, it is important to ensure that the results are reproducible in an automated fashion. It should be possible to run the complete analysis pipeline without any user intervention. In addition to the value to the community of being able to reproduce an analysis, reproducible research practices allow for better control over the project while it is being developed. For example, if necessary parameters to run a pipeline are kept separately from the code that implements it (perhaps even in the researcher's mind), this leads to error-prone analysis and opens up the possibility that when the results are to be written up for publication, the researcher will no longer be able to even completely describe the process that led to them.

At the same time, for large projects, the use of multiple processors (either in the same machine or distributed across a cluster) is necessary to obtain results in a useful time frame. Furthermore, it is often the case that, as the project evolves, it becomes convenient to save intermediate results while down-stream analyses are designed (or redesigned) and implemented. Therefore, having a single point of entry (often in the form of a single script or a single main programme) for the computation becomes increasingly difficult. Errors arising from the use of stale intermediate results will also be common. Before he developed Jug, the author was often forced to delete all intermediate files when part of the project had changed as he could no longer be sure which ones were up to date; this resulted in many unneeded recomputations and time lost waiting for those results.

Jug is a Python-based software framework which solves all of these problems in a simple way. Jug supports caching of intermediate results, distribution of computation as tasks using multiple cores or multiple computers in a network. It works well with batch-based systems such as those that are typically used in scientific clusters.

A task in Jug can be built from almost any Python function and take any argument (as long as it can be serialized using standard Python's pickle module). Using Jug adds very little overhead over developing in simple Python while making any library or functionality available in that language available to the user.

Jug is written in pure Python, is completely cross-platform, and available as free software under the MIT license. Having been developed over the last 5 years and has no known bugs. It does have an extensive automated test suite.

OpenLabFramework: A Next-Generation Open-Source Laboratory Information Management System for Efficient Sample Tracking

Markus List[1,2,3], *Steffen Schmidt*[1,2] , *Jakub Trojnar*[1,2,4], *Jochen Thomas*[5], *Mads Thomassen*[1,3], *Torben Kruse*[1,3], *Qihua Tan*[3], *Jan Baumbach*[6] , *Jan Mollenhauer*[1,2]

1. Lundbeckfonden Center of Excellence in Nanomedicine NanoCAN, University of Southern Denmark, Odense, Denmark
2. Molecular Oncology, University of Southern Denmark, Odense, Denmark
3. Clinical Institute, University of Southern Denmark, Odense, Denmark
4. Department of Biochemistry and Molecular Biology, University of Southern Denmark, Odense, Denmark
5. io-consultants GmbH & Co. KG, Heidelberg, Germany
6. Department of Mathematics and Computer Science, University of Southern Denmark, Odense, Denmark

Contact E-Mail:	mlist@health.sdu.dk
Project URL:	https://github.com/NanoCAN/OpenLabFramework/
Demo-Application:	http://openlabframework.cloudfoundry.com (username: admin, password: demo0815)
Software License:	GPL v3

Background:

The advance of new technologies in biomedical research has led to a dramatic growth in experimental throughput. Projects therefore steadily grow in size and involve a larger number of researchers, often across several laboratories and institutions. Spreadsheets traditionally used are thus no longer suitable for keeping track of the vast amounts of samples created and need to be replaced with state-of-the-art laboratory information management systems. Such systems have been developed in large numbers in the past years, but they are often limited to specific research domains and types of data. More flexible systems are required that are preferably open-source and thus can be continuously extended and adapted to fulfill new requirements. Moreover, hurdles in deployment and user acceptance have to be reduced.

Results:

We present OpenLabFramework (OLF), a web-application developed for sample tracking, which is particularly laid out for the management of large libraries of vector constructs and genetically engineered cell lines, but has an open architecture for the addition of modules for other biological materials and functional data. OLF is built using state-of-the-art web techniques, and in a strictly modular fashion. OLF offers basic reporting and document management capabilities and features modern technologies like QR barcodes, direct label printing, and mobile devices. In addition, OLF is flexible through support of various database systems. It can be installed locally, on a server, or in the cloud.

Conclusions:

The specific requirements for the management of vector construct and cell line libraries have, to this extend, not been covered by existing solutions before. With OLF, we present an adaptable, extendable, robust, and flexible open-source alternative. The intuitive and responsive web-interface may facilitate user acceptance and productivity, which is further supported through the systematic use of barcode labels and mobile device functionality integration. OLF has the potential to become a driver in defining the characteristics of next generation laboratory information management systems.

Poster 4

Ten Simple Rules for the Open Development of Scientific Software

Andreas Prlić [1], Jim Procter [2], Hilmar Lapp [3]

1) andreas.prlic@gmail.com San Diego Supercomputer Center, University of California San Diego, La Jolla, California, United States of America
2) J.Procter@dundee.ac.uk School of Life Sciences Research, College of Life Sciences, University of Dundee, Dundee, Scotland, United Kingdom
3) hlapp@nescent.org National Evolutionary Synthesis Center (NESCent), Durham, North Carolina, United States of America

Open-source software development has had significant impact, not only on society, but also on scientific research. Papers describing software published as open source are amongst the most widely cited publications, suggesting many scientific studies may not have been possible without some kind of open software to collect observations, analyze data, or present results. It is surprising, therefore, that so few papers are accompanied by open software, given the benefits that this may bring.

To foster a culture of open exchange and reuse of software, the journal *PLOS Computational Biology* has recently started to publish open-source *Software Articles* in a new section of the journal. A recent editorial carries the same title as this talk and points out ten rules that should make the development of open scientific software more rewarding and the experience of using software more positive.

The goal of this talk is to reflect upon the way we are building our software and the communities around it, using the ten simple rules as a guideline.

The PLOS Computational Biology Software Collection:

http://www.ploscollections.org/article/browse/issue/info%3Adoi%2F10.1371%2Fissue.pcol.v03.i10

License: CC-BY

Refinery Platform - Integrating Visualization and Analysis of Large-Scale Biological Data

Nils Gehlenborg[1]*, Richard Park*[1]*, Ilya Sytchev*[2]*, Psalm Haseley*[1]*, Shannan Ho Sui*[2]*, Winston Hide*[2]*, Peter J Park*[1]

[1] Center for Biomedical Informatics, Harvard Medical School, Boston, MA, USA; [2] Department of Biostatistics, Harvard School of Public Health, Boston, MA, USA | Correspondence: *nils@hms.harvard.edu*

Web: *refinery-platform.org* · Code: *github.com/parklab/refinery-platform* · License: *MIT + additional clause*

Data sets with dozens or hundreds of samples are now common in molecular biology. Keeping track of data files generated during the analysis process is tedious and error prone, and dealing with such large and complex data sets requires significant infrastructure developments. To address these challenges, we are developing the web-based *Refinery Platform* for large-scale biological data. This flexible platform is designed to accommodate diverse data and analyses; our current implementation focuses on epigenomics and cancer genomics. Our ultimate goal for the system is to enable domain experts to efficiently run dozens or hundreds of samples through complex analysis pipelines, to visualize and explore the results, as well as to plan and execute follow-up analyses. We envision three types of users for the *Refinery Platform*: experimental scientists generating large collections of data can use *Refinery* to manage, analyze, and visualize their data using tools provided by computational scientists. Bioinformatics experts can develop new workflows and deploy them to support domain experts. Finally, the data management features and APIs built into the *Refinery Platform* will enable visualization experts to develop tools for large biological data sets, which could not be implemented without the infrastructure provided by the *Refinery Platform*.

The *Refinery Platform* consists of three major components: (1) a data repository with rich metadata capabilities, (2) a workflow engine based on the popular Galaxy system, and (3) visualization tools to support the exploration and interpretation of results at all stages of the analysis process. The data repository is built around a data model that was strongly influenced by the ISA-Tab general purpose file format (www.isacommons.org) to describe biological experiments. All metadata attributes can be freely defined by the user. A faceted-browsing interface and a flexible matrix view are provided to enable users to quickly filter through data sets with thousands of samples. The *Refinery Platform* data model also provides extensive provenance information in the “experiment graph”, which links all files in the data set to the biological specimens that they were derived from. Currently, ISA-Tab and tab-delimited text files can be used to import metadata into the system. Data files can be either stored locally or remotely. Remote files are automatically imported when requested for analysis or visualization.

Workflows available in the *Refinery Platform* are implemented in Galaxy (www.galaxyproject.org). The Galaxy workflow editor is used to create a “workflow template” that is imported by the *Refinery Platform*, instantiated based on the inputs selected by the user, and exported back into Galaxy along with the data files. All workflow setup, execution, and monitoring is handled through the Galaxy API. Workflow results are downloaded into *Refinery* from Galaxy, added to the experiment graph, associated with their corresponding inputs, and made available for visualization or as input for further analyses.

Currently, the *Refinery Platform* integrates the Integrated Genomics Viewer (www.broadinstitute.org/igv, IGV) through Java Web Start for data visualization and exploration. IGV supports dozens of data types and a wide range of applications and has proven a very flexible solution. More advanced visualization tools will be the focus of the second phase of the development of the *Refinery Platform*. We will develop web-based tools for both common data types and specific analyses, such as structural genomic variants or comparison of epigenetic profiles. Since many other popular visualization tools such as Cytoscape (www.cytoscape.org) and Caleydo (www.caleydo.org) can be launched as Java Web Start applications, we will also integrate a generic interface to deploy such applications with data served by the *Refinery Platform*.

The *Refinery Platform* server components are implemented in Python based on the Django web framework (www.djangoproject.org) supported by a Postgresql database and Apache Solr (lucene.apache.org/solr). The web interface is implemented with Bootstrap (twitter.github.io/bootstrap). A REST API is available.

Instances of the *Refinery Platform* are currently deployed as the ENCODE-X data browser (encode-x.med.harvard.edu) and as part of the Stem Cell Commons project (www.stemcellcommons.org).

MetaSee: An interactive visualization toolbox for metagenomic sample analysis and comparison

Authors and Affiliations

Xiaoquan Su, Baoxing Song, Jian Xu, Kang Ning*

Qingdao Institute of Bioenergy and Bioprocess Technology, Chinese Academy of Sciences

suxq@qibebt.ac.cn

URL and License

The URL of MetaSee is http://www.metasee.org.

The source code can be access from http://www.metasee.org/download.jsp.

MetaSee was released under *MIT License.*

Abstract

The next generation sequencing – based metagenomic data analysis is becoming the mainstream for the studies of microbial communities. Faced with a large amount of data in metagenomic research, data and result visualization would thus be important for scientists to effectively explore, interpret and manipulate such rich information. The visualization of the metagenomic data, especially multiple sample data, is one of the most challenging tasks for biological data visualization: the different data sample sources, different sequencing approaches, heterogeneous data format make the robust and seamless visualization system difficult. Moreover, researchers have different focuses for metagenomic studies: taxonomical or functional, sample-centric or genome-centric, single sample or multiple samples, etc. However, current efforts in visualization of metagenomic data cannot fulfill all of these needs, and it is extremely hard to organize all of these visualization effects in a systematic manner. Therefore, an extendable, interactive visualization tool would be the solution of choice to fulfill all of these visualization needs.

In this work, we have proposed MetaSee, an extendable toolbox that facilitates the interactive visualization of metagenomic samples of interests. The main components of MetaSee include: (I) a core visualization engine that is composed of different views for comparison of multiple samples: global view, phylogenetic view, sample view and sub-sample view, as well as taxonomy link-out for more in-depth analysis of certain taxa, (II) front-end user interface with real environment models that connect to the above core visualization engine and (III) open-source portal for development of plug-ins. MetaSee accepts input in various formats such as analysis results by Parallel-META (Su, *et al.*, BMC Systems Biology, 2012), MEGAN (Huson, *et al.* Genome Research, 2007), MG-RAST (Meyer, *et al.* BMC Bioinformatics, 2008) and so on. In addition, we have also applied the MetaSee into project "Global Systems Biology", in which thousands of metagenomic samples were analyzed by Parallel-META (Su, *et al.*, BMC Systems Biology, 2012), visualized by MetaSee and then marked via the Google Map API. Each marker in the map that represents a sample can be clicked to check the visualized analysis results by MetaSee.

The integrative visualization tool would not only provide the needed visualization effects, but also facilitate researcher for in-depth analysis of the metagenomic sample of interests. Moreover, its open-source portal has enabled the redevelopment or design of plug-ins for MetaSee, which would facilitate the development of any additional visualization effects or building it in other applications.

Poster 6

DGE-Vis : Visualisation of RNA-seq data for Differential Gene Expression analysis

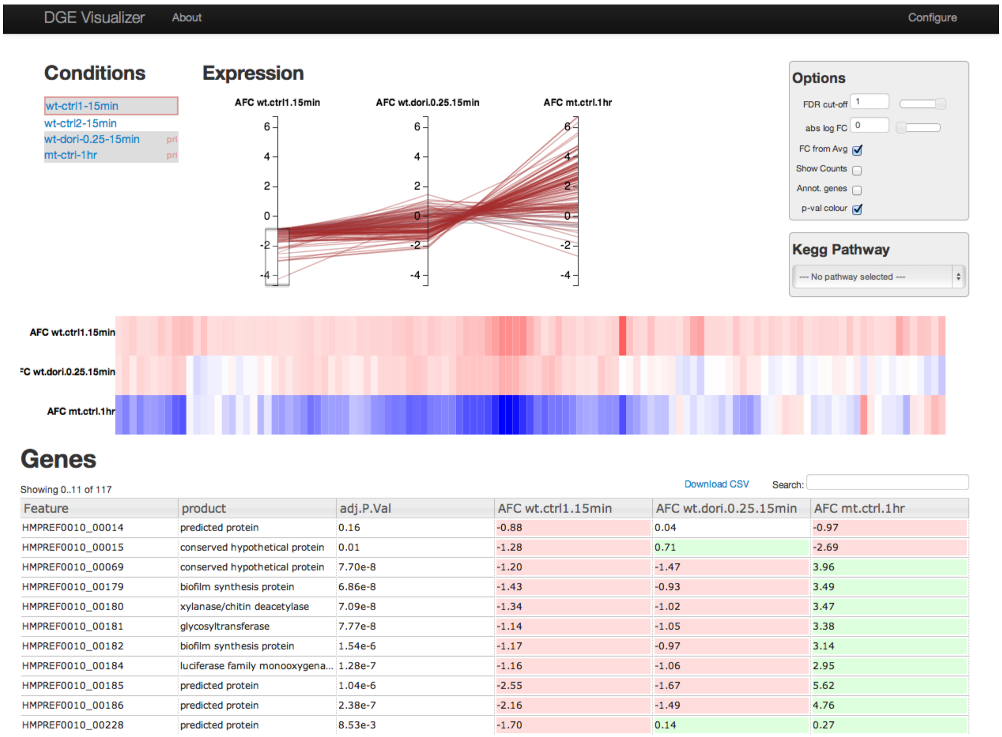
David R. Powell[1]
<David.Powell@monash.edu>
[1]Victorian Bioinformatics Consortium, Monash University, Clayton, Australia and
Life Sciences Computation Centre, Victorian Life Sciences Computation Initiative, Australia

Project website: http://www.vicbioinformatics.com/software.dge-vis.shtml
Source code: https://github.com/Victorian-Bioinformatics-Consortium/dge-vis
License: GPLv3

RNA-seq experiments are popular, but analysing and interpreting the results is difficult. We present our new software, DGE-Vis, that may be used as a web tool to both analyse RNA-seq data using the proven statistical software limma and edgeR, and to visualize the results of this analysis in a novel and interactive manner. The software runs in browser and is written using a combination of Haskell, R and CoffeeScript.

The outcome of an RNA-seq analysis is often just a list of differentially expressed genes for a pair of conditions, which loses much of the useful information in the data. DGE-Vis allows a user to select two, **or more**, conditions to analyse for differential expression. The expression data is presented on a parallel-coordinates plot and on a heatmap, both are dynamic graphs which can be interactively filtered by different parameters such as fold-change, and p-value significance. Further, gene expression can be overlayed on KEGG pathways to highlight which genes are differentially expressed in which pathways. This allows researchers to find results and patterns in their data that may not have been apparent otherwise.

DGE-Vis is simple to use - just upload a CSV file of gene counts; define your replicates in the web interface; then view the resulting differential expression analysis; and share the analysis with your colleagues.



FDR cut-off 1
abs log FC 0
FC from Avg ☑
Show Counts ☐
Annot. genes ☐
p-val colour ☑

--- No pathway selected ---

Genes

Showing 0..11 of 117

Feature	product	adj.P.Val	AFC wt.ctrl1.15min	AFC wt.dori.0.25.15min	AFC mt.ctrl.1hr
HMPREF0010_00014	predicted protein	0.16	-0.88	0.04	-0.97
HMPREF0010_00015	conserved hypothetical protein	0.01	-1.28	0.71	-2.69
HMPREF0010_00069	conserved hypothetical protein	7.70e-8	-1.20	-1.47	3.96
HMPREF0010_00179	biofilm synthesis protein	6.86e-8	-1.43	-0.93	3.49
HMPREF0010_00180	xylanase/chitin deacetylase	7.09e-8	-1.34	-1.02	3.47
HMPREF0010_00181	glycosyltransferase	7.77e-8	-1.14	-1.05	3.38
HMPREF0010_00182	biofilm synthesis protein	1.54e-6	-1.17	-0.97	3.14
HMPREF0010_00184	luciferase family monooxygena...	1.28e-7	-1.16	-1.06	2.95
HMPREF0010_00185	predicted protein	1.04e-6	-2.55	-1.67	5.62
HMPREF0010_00186	predicted protein	2.38e-7	-2.16	-1.49	4.76
HMPREF0010_00228	predicted protein	8.53e-3	-1.70	0.14	0.27

Genomic Visualization Everywhere with Dalliance

Thomas A. Down[1] and Tim J. P. Hubbard[1]
[1] Wellcome Trust Sanger Institute, Cambridge, UK

Project website: http://www.biodalliance.org/
Source download: http://github.com/dasmoth/dalliance
License: BSD

Genome browsers are a vital part of the genomics workflow. Visual inspection of annotations and experimental data are indispensable for spotting unexpected correlations, formulating new hypotheses, or simply sanity-checking new data sets. The sequencing revolution has made this even more important. Today, even small labs are routinely applying techniques like ChIP-seq, RNA-seq, or resequencing, often with limited bioinformatic support. To keep pace with these trends, we need powerful visualization and data integration tools that make loading large datasets – either directly or as an automatic final set in analysis workflows – as simple as possible.

Dalliance [1] is a new genome browser which makes aggressive use of HTML5 technologies to offer a high level of interactivity and powerful navigation and exploration tools while running within a normal web browser. The display can be freely scrolled and zoomed with mouse gestures and keyboard controls. Shortcuts for navigation between features allows sparse datasets to be rapidly explored. We have adopted a fully distributed approach, with no master server hosting the primary dataset. Data can be integrated using either the standard DAS protocol [2], or from data packaged in a standard indexed binary format such as BigWig or BAM on a standard web server. The latter option makes integrating new genomic datasets, particularly the results of high-throughput sequencing experiments, very quick, and accessible to occasional bioinformaticians who often have limited sysadmin experience – and limited enthusiasm for installing extra server software. It also allows instant access to datasets in this format from remote web servers without time-consuming downloading (e.g. ENCODE datasets from UCSC). Unusually for a web-based application, Dalliance also allows viewing of data directly from local disk on your own machine.

Recently, Dalliance has been substantially re-written to use HTML5 canvas facilities as the primary rendering system (while still allowing SVG graphics to be generated when a print-quality static figure is needed). We are working to make it as easy as possible to integrate within other web applications, or as an interactive figure for papers or blog entries.

[1] Down TA, Piipari M, Hubbard TJ. *Dalliance: interactive genome viewing on the web* Bioinformatics (2011) 27:889-890

[2] Jenkinson A et al. *Integrating biological data – the Distributed Annotation System.* BMC Bioinformatics (2008) 9(suppl8):S3

Robust quality control of Next Generation Sequencing alignment data

Konstantin Okonechnikov* (1), Javier Santoyo (2), Joaquín Dopazo (3), Thomas F. Meyer (1), Ana Conesa (3), Fernando García-Alcalde (1)

(1) Max Planck Institute For Infection Biology, Berlin, Germany. { *okonechnikov@mpiib-berlin.mpg.de }
(2) Medical Genome Project, Andalusian Center for Human Genomic Sequencing, Sevilla, Spain.
(3) Institute of Computational Genomics, Centro de Investigación Príncipe Felipe

Web-site: http://qualimap.bioinfo.cipf.es/
Source code: http://qualimap.bioinfo.cipf.es/archive.html
License: GNU GPL v2

Next Generation Sequencing (NGS) has become de facto an important and conventional discovery instrument in modern genomics, which allows to investigate biological processes at an unprecedented level of detail. Unfortunately, NGS is prone to different types of biases, usually specific to the particular technology being used. Moreover, steps involved in the sequencing process such as sample preparation, protocol implementation and data processing may introduce additional errors. Therefore quality control and detection of possible biases are required for reliable data interpretation.

A number of tools for NGS data quality assessment were developed recently. Some are designed to directly analyze the raw reads output of the sequencer machines, e.g. FastQC[1]. Other programs are designed to work only with data generated by a specific sequencing approach and not suitable for general alignment analysis [2,3] . Finally, there exist command-line toolkits [4], which are able to compute various alignment metrics by means of individual programs, however requiring additional programming effort to create a common pipeline that accumulates and interprets results from different sources.

In order to advance in this context we have developed Qualimap, a tool for NGS alignment quality assessment [5]. Qualimap is an open-source Java application which provides an overview of the alignment data that helps to detect biases and eases decision-making for further analysis. It offers three pipelines: BAM QC, RNA-seq QC and Counts QC. Each pipeline performs analysis of input data and produces an easy-to-interpret report with a descriptive summary and a number of plots. BAM QC pipeline can be applied on a whole genome scale or for a set of arbitrary regions, delivering statistics both inside and outside of given regions, thus being applicable for any type of sequencing data. The pipeline calculates general alignment metrics like mapping statistics or homopolymer indel distribution, along with particular aspects of the alignment such as genome fraction coverage or duplication rate. RNA-seq QC pipeline is designed specifically for transcriptome sequencing data. It provides gene coverage plots and 5'-3' bias estimation. Counts QC pipeline is also designed for RNA-seq data, but accepts any general feature counts. It computes global saturation, samples correlation and compares counts in various feature groups. Qualimap provides both graphical user interface and command line interface, which makes it possible to use the tool on a remote server or cluster. The application is being improved and updated on a regular basis and currently used by several labs worldwide. Qualimap is accompanied with a detailed user manual and supported via Google groups discussion forum.

References:
[1] http://www.bioinformatics.babraham.ac.uk/projects/fastqc
[2] Deluca, et al, (2012) Bioinformatics (2012) 28 (11): 1530-1532
[3] Statham, et al. "Repitools: an R package for the analysis of enrichment-based epigenomic data." *Bioinformatics* 26.13 (2010): 1662-1663.
[4] http://picard.sourceforge.net
[5] Garcia-Alcalde et.al, *Bioinformatics* (2012) 28(20): 2678-2679

Visualizing bacterial sequencing data with GenomeView

Thomas Abeel [1,2], Yves Van de Peer [2], Bruce Birren [1], Ashlee Earl [1]

1) Broad Institute of MIT and Harvard, Cambridge, USA
2) VIB Department of Plant Systems Biology, UGent, Belgium

tabeel@broadinstitute.org – thomas@genomeview.org

Short abstract

GenomeView, a next-generation genome browser, enables users to interactively browse high volumes of sequencing data and whole genome alignments of dozens of genomes with dynamic navigation and zooming. It can handle dozens of aligned genomes, thousands of annotation features and millions of short reads.

Full abstract

Advances in DNA sequencing methods have allowed for the creation of billions of nucleotide sequences on a daily basis. As the size of these data sets grow, the ability to visually explore sequencing data becomes increasingly valuable. Visualization is crucial at any stage of data analysis, including in the upfront quality evaluation of data sets and in the assessment of downstream analytical derivatives and final results. The right image can make solutions to challenging analytical problems obvious and analytical results much more easily understood for both the analyst and the scientific community. We present GenomeView, a tool specifically designed to visualize and manipulate a multitude of genomics data in particular from a comparative genomics perspective. GenomeView enables users to browse high volumes of aligned short read data, with dynamic navigation and semantic zooming. At the same time, the tool enables visualization of whole genome alignments of dozens of genomes relative to a reference sequence. GenomeView is also unique in that it allows for sharing and interactive handling of huge data sets consisting of dozens of aligned genomes, thousands of annotation features and millions of mapped short reads.

GenomeView has been an Open Source project hosted on Sourceforge since 2006 with thousands of users.

License: GNU GPL 2 or later

URL: http://genomeview.org

Code hosting: https://sourceforge.net/projects/genomeview/

Reference:

Abeel, T., Van Parys, T., Saeys, Y., Galagan, J., Van de Peer, Y. 2011. GenomeView: a next-generation genome browser. *Nucleic Acids Research*. doi: 10.1093/nar/gkr995

Poster 8

Title: BioRuby project updates - power of modularity in the community-based open source development model

Authors: Toshiaki Katayama[1], Pjotr Prins[2], Raoul J.P. Bonnal[3], Francesco Strozzi[4], Naohisa Goto[5]

Affiliations:
[1] Database Center for Life Science, Research Organization of Information and Systems, Faculty of engineering Bldg. 12, The University of Tokyo 2-11-16, Yayoi, Bunkyo-ku, Tokyo 113-0032, Japan (ktym@dbcls.jp)
[2] Laboratory of Nematology, Wageningen University, Wageningen 6708 PB, The Netherlands
[3] Integrative Biology Program, Istituto Nazionale Genetica Molecolare, Milan 20122, Italy
[4] CeRSA, Parco Tecnologico Padano, Lodi 26900, Italy
[5] Department of Genome Informatics, Genome Information Research Center, Research Institute for Microbial Diseases, Osaka University, Suita, Osaka 565-0871, Japan

Project URL: http://bioruby.org/, http://biogems.info/
Code URL: https://github.com/bioruby/bioruby
Open Source License: The Ruby License

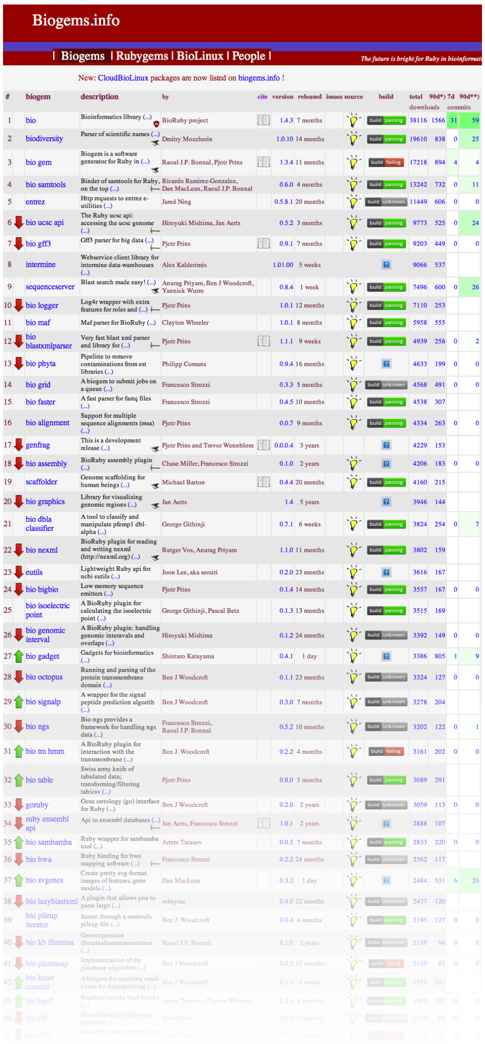
The BioRuby project was originally started in year 2000 and soon affiliated with the Open Bio Foundation. This bioinformatics library for the Ruby programming language is well suited for scripting parsers and converters of the database entries, wrappers for command-line tools and Web services, manipulating biological object models, and developing bioinformatics applications especially with a Web interface thanks to the Ruby on Rails framework. Also, with JRuby, user can use the library and run applications on a Java VM in combination with other Java libraries.

The BioRuby code base has been maintained as a public open source repository, formerly in CVS then SVN and currently hosted at GitHub, however, it had been getting difficult to incorporate new modules in a timely manner while keeping stable compatibility that is essential for the scientific reliability.

Therefore, to accelerate the development process, we introduced a plug-in system, Biogem, in 2011. Biogem provides an integrated environment for creation and distribution of new "gem" packages for bioinformatics, where "gem" is the official package manager for Ruby.

We found this decentralized approach very successful for inviting new developers to the BioRuby community. They brought not only state-of-the-art libraries such as for NGS data analysis but also useful applications like visualization tools or biomedical Web servers. To date, we counted more than 80 Biogem packages covering variety of domains in life sciences (Fig. 1).

Here we report recent updates of the BioRuby core library and active Biogem projects. We believe this collaborative development model can also be applied to broader Open Bio* community.


Biogems.info
| Biogems | Rubygems | BioLinux | People |
The future is bright for Ruby in bioinformati

New: CloudBioLinux packages are now listed on biogems.info !

#		biogem	description	by	cite	version	released	issues	source	build	total downloads	90d*)	7d commits	90d**)
1		bio	Bioinformatics library (...)	BioRuby project	▯	1.4.3	7 months		💡	build passing	38116	1566	31	59
2		biodiversity	Parser of scientific names (...)	Dmitry Mozzherin		1.0.10	14 months		💡	build passing	19610	838	0	25
3		bio gem	Biogem is a software generator for Ruby in (...)	Raoul J.P. Bonnal, Pjotr Prins	▯	1.3.4	11 months		💡	build failing	17218	894	4	4
4		bio samtools	Binder of samtools for Ruby, on the top (...)	Ricardo Ramirez-Gonzalez, Dan MacLean, Raoul J.P. Bonnal		0.6.0	4 months		💡	build passing	13242	732	0	11
5		entrez	Http requests to entrez e-utilities (...)	Jared Ning		0.5.8.1	20 months		💡	build unknown	11449	606	0	0
6	↓	bio ucsc api	The Ruby ucsc api: accessing the ucsc genome (...)	Hiroyuki Mishima, Jan Aerts		0.5.2	3 months		💡	build passing	9773	525	0	24
7	↓	bio gff3	Gff3 parser for big data (...)	Pjotr Prins	▯	0.9.1	7 months		💡	build passing	9203	449	0	0
8		intermine	Webservice client library for intermine data-warehouses (...)	Alex Kalderimis		1.01.00	5 weeks			?	9066	537		
9		sequenceserver	Blast search made easy! (...)	Anurag Priyam, Ben J Woodcroft, Yannick Wurm		0.8.4	1 week		💡	build passing	7496	600	0	26
10	↓	bio logger	Log4r wrapper with extra features for roles and (...)	Pjotr Prins		1.0.1	12 months		💡	build passing	7110	253		
11		bio maf	Maf parser for BioRuby (...)	Clayton Wheeler		1.0.1	8 months		💡	build passing	5958	555		
12	↓	bio blastxmlparser	Very fast blast xml parser and library for (...)	Pjotr Prins	▯	1.1.1	9 weeks		💡	build passing	4939	256	0	2
13	↓	bio phyta	Pipeline to remove contaminations from est libraries (...)	Philipp Comans		0.9.4	16 months		💡	?	4633	199	0	0
14		bio grid	A biogem to submit jobs on a queue (...)	Francesco Strozzi		0.3.3	5 months		💡	build unknown	4568	491	0	0
15		bio faster	A fast parser for fastq files (...)	Francesco Strozzi		0.4.5	10 months		💡	build passing	4538	307		
16		bio alignment	Support for multiple sequence alignments (msa) (...)	Pjotr Prins		0.0.7	9 months		💡	build passing	4334	263	0	0
17	↓	genfrag	This is a development release (...)	Pjotr Prins and Trevor Wennblom	▯	0.0.0.4	3 years			?	4229	153		
18	↓	bio assembly	BioRuby assembly plugin (...)	Chase Miller, Francesco Strozzi		0.1.0	2 years		💡	?	4206	183	0	0
19		scaffolder	Genome scaffolding for human beings (...)	Michael Barton	▯	0.4.4	20 months		💡	build passing	4160	215		
20	↓	bio graphics	Library for visualizing genomic regions (...)	Jan Aerts		1.4	5 years			?	3946	144		
21		bio dbla classifier	A tool to classify and manipulate pfemp1 dbl-alpha (...)	George Githinji		0.7.1	6 weeks		💡	build passing	3824	254	0	7
22	↓	bio nexml	BioRuby plugin for reading and writing nexml (http://nexml.org) (...)	Rutger Vos, Anurag Priyam		1.1.0	11 months		💡	build passing	3802	159		
23	↓	eutils	Lightweight Ruby api for ncbi eutils (...)	Joon Lee, aka seouri		0.2.0	23 months		💡	?	3616	167		
24	↓	bio bigbio	Low memory sequence emitters (...)	Pjotr Prins		0.1.4	14 months		💡	build passing	3557	167	0	0
25		bio isoelectric point	A BioRuby plugin for calculating the isoelectric point (...)	George Githinji, Pascal Betz		0.1.3	13 months		💡	build passing	3515	169		
26	↓	bio genomic interval	A BioRuby plugin: handling genomic interavals and overlaps (...)	Hiroyuki Mishima		0.1.2	24 months		💡	build unknown	3392	149	0	0
27	↑	bio gadget	Gadgets for bioinformatics (...)	Shintaro Katayama		0.4.1	1 day		💡	?	3386	805	1	9
28	↓	bio octopus	Running and parsing of the protein transmembrane domain (...)	Ben J Woodcroft		0.1.1	23 months		💡	build unknown	3324	127	0	0
29	↑	bio signalp	A wrapper for the signal peptide prediction algorith (...)	Ben J Woodcroft		0.3.0	7 months		💡	build unknown	3278	204		
30	↓	bio ngs	Bio-ngs provides a framework for handling ngs data (...)	Francesco Strozzi, Raoul J.P. Bonnal		0.5.2	10 months		💡	build unknown	3202	122	0	1
31	↑	bio tm hmm	A BioRuby plugin for interaction with the transmembrane (...)	Ben J. Woodcroft		0.2.2	4 months		💡	build failing	3161	202	0	0
32	↑	bio table	Swiss army knife of tabulated data; transforming/filtering tab/csv (...)	Pjotr Prins		0.8.0	5 months		💡	build passing	3089	291		
33	↓	goruby	Gene ontology (go) interface for Ruby (...)	Ben J Woodcroft		0.2.0	2 years		💡	build unknown	3059	113	0	0
34	↓	ruby ensembl api	Api to ensembl databases (...)	Jan Aerts, Francesco Strozzi	▯	1.0.1	2 years		💡	?	2888	107		
35	↑	bio sambamba	Ruby wrapper for sambamba tool (...)	Artem Tarasov		0.0.5	7 months		💡	build passing	2833	220	0	0
36	↓	bio bwa	Ruby binding for bwa mapping software (...)	Francesco Strozzi		0.2.2	24 months		💡	build unknown	2562	117		
37	↑	bio svgenes	Create pretty svg-format images of features, gene models (...)	Dan MacLean		0.3.2	1 day		💡	?	2484	531	6	26
38	↓	bio lazyblastxml	A plugin that allows you to parse large (...)	robsyme		0.4.0	22 months		💡	build unknown	2437	120		
39		bio pileup iterator	Iterate through a samtools pileup file (...)	Ben J. Woodcroft		0.0.4	4 months		💡	build passing	2195	127	0	0
40	↓	bio kb illumina	Geneexpression illuminahumanannotation (...)	Raoul J.P. Bonnal		0.1.0	2 years		💡	build unknown	2159	96	0	0
41	↓	bio plasmoap	Implementation of the plasmoap algorithm (...)	Ben J Woodcroft		0.0.2	15 months		💡	build failing	2139	85	0	0
42	↑	bio kmer counter	A biogem for counting small kmers for fingerprinting (...)	Ben J Woodcroft		0.1.1	2 weeks		💡	build passing	1954	253		
43	↑	bio bgzf	Reading/writing bgzf blocks (...)	Artem Tarasov, Clayton Wheeler		0.2.1	8 months		💡	build passing	1902	182	0	0
44	↓	bio rdf	Bioinformatics triple-store support (...)	Pjotr Prins		0.0.2	10 months		💡	build unknown	1793	82	0	0
45	↓	bio affy	Parse affymetrix cel/cdf files (...)	Pjotr Prins		0.5.1	16 months		💡	build failing	1714	87	0	0
			BioRuby core plugins, most											

Fig. 1 The biogems.info site showing a number of plug-in packages (82 as of April 12, 2013) with statistics.

Biopython Project Update

Peter Cock,[*] *et al.*

Bioinformatics Open Source Conference (BOSC) 2013, Berlin, Germany

Website: `http://biopython.org`
Repository: `https://github.com/biopython/biopython`
License: Biopython License Agreement (MIT style, see `http://www.biopython.org/DIST/LICENSE`)

In this talk we present the current status of the Biopython project, a long-running, distributed collaboration producing a freely available Python library for biological computation [1]. Biopython is supported by the Open Bioinformatics Foundation (OBF).

Since BOSC 2012, we will have made two releases, and a manuscript describing the `Bio.Phylo` module for phylogenetics has been published [2]. All our releases have had more unit tests, more documentation, and new contributors. In addition to nightly unit tests run on Linux, Windows and Mac OS X via an OBF hosted buildbot, we now run continuous integration tests via TravisCI. Together this covers the main operating systems (Linux, Mac OS X and Windows) and Python implementations (Jython, PyPy and both versions 2 and 3 of the primary Python implementation in C).

Biopython 1.61 (February 2013) included a number of small enhancements and additions. This release also introduced the idea of including some *beta* modules within the standard install for testing new experimental code, used to ship some of the GSoC work (see below).

Biopython has for some time run under Python 3, and we have been asking users to try this for testing purposes for some time. Biopython 1.62 (expected May 2013) is intended be our first release to officially support Python 3. [*NB: to be revised during abstract review*].

In Summer 2012 we had two Google Summer of Code (GSoC) students. Wibowo 'Bow' Arindrarto wrote a Biopython equivalent to BioPerl's SearchIO covering sequence search results from BLAST, HMMER, FASTA etc able to read and write multiple file formats with a common object model, and index large files for memory efficient random access. Lenna Peterson worked on the representation and manipulation of genomic variants (HGVS, GFF, VCF files). Both students finished the GSoC programme successfully, and have continued to be contribute to the project since. Bow's SearchIO code has already been included in recent releases, marked as experimental for wider testing and feedback. Biopython hopes to mentor more students for GSoC 2013, for example some phylogenetics project ideas have been put forward which would be mentored under the NESCent project. [*NB: to be revised during abstract review*].

References

[1] Cock, P.J.A., Antao, T., Chang, J.T., Chapman, B.A., Cox, C.J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., de Hoon, M.J. (2009) Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics* **25**(11) 1422-3. doi:10.1093/bioinformatics/btp163

[2] Talevich, E., Invergo, B.M., Cock, P.J.A., and Chapman, B.A. (2012) Bio.Phylo: A unified phylogenetics toolkit for Biopython. *BMC Bioinformatics* **13**:209. doi:10.1186/1471-2105-13-209

[*] Information and Computational Sciences, James Hutton Institute (formerly SCRI), Invergowrie, Dundee DD2 5DA, UK

BOSC 2013 Proposal: InterMine

Title	Collaborative Data Mining
Licence	LGPL
Funding	**NIH** and **Wellcome Trust**
Source URL	`https://github.com/intermine`
Project URL	`http://www.intermine.org`
Affiliation	Cambridge Systems Biology Centre & Department of Genetics, University of Cambridge, Cambridge CB2 3EH, UK
Authors	Alex Kalderimis Julie Sullivan Radek Štěpán Fengyan Hu Sergio Contrino Mike Lyne Rachel Lyne Gos Micklem (PI)
	alex@intermine.org julie@intermine.org radek@intermine.org fengyuan@modencode.org sergio@modencode.org mike@intermine.org rachel@flymine.org g.micklem@gen.cam.ac.uk

InterMine is an active open-source project producing integrated data-warehousing and data-mining applications for the life-sciences. Currently in use or development in many major Model Organism Databases (MODs) as part of the NIH InterMOD project. This includes budding yeast (SGD), rat (RGD), zebrafish (ZFIN), mouse (MGI), and nematode (Wormbase), as well as other projects such as FlyMine (D. melanogaster) and modMine (modENCODE data). Across these projects InterMine offers a common platform for integrated data-mining and analysis.

Key features of the InterMine platform include a flexible, domain driven design, the ability to integrate data from a wide range of common biological formats, automatic and transparent query optimization, a powerful RESTful API, embeddable graphical components, and a configurable data-mining web-application. Developments over the past year have lead to more powerful data exploration tools, and a framework for collaboration on sets of data.

2012-2013 saw two major releases of the core InterMine software, as well as several patch releases and a major rewrite of the core InterMine JavaScript library. This work has brought new features to the graphical components of the web-application, such as information rich item previews, interactive and exportable graphical summaries, and custom background populations for enrichment calculations. New query types are now available, with genomic region search now integrated into the main query mechanism. We have continued to expand our collaborative efforts with the introduction of *Genomespace* integration.

A completely new set of features now improves collaboration by allowing the use of shared data sets amongst a set of authorised users. Data and analysis previously available to a single researcher can now be shared with collaborators, with fine-grained control of permissions which are easily extended or revoked.

The GenoCAD 2.2 grammar editor

Mandy L Wilson[1], Laura Adam[1] and Jean Peccoud[1, 2]

[1]Virginia Bioinformatics Institute, Virginia Tech, Washington Street MC 0477, Blacksburg, VA 24061, USA

[2]ICTAS Center for Systems Biology of Engineered Tissues, MC 0193 Virginia Tech, Blacksburg, VA 24061, USA

peccoud@vt.edu

Project URL: www.genocad.org

Source code: www.sourceforge.net/projects/genocad/

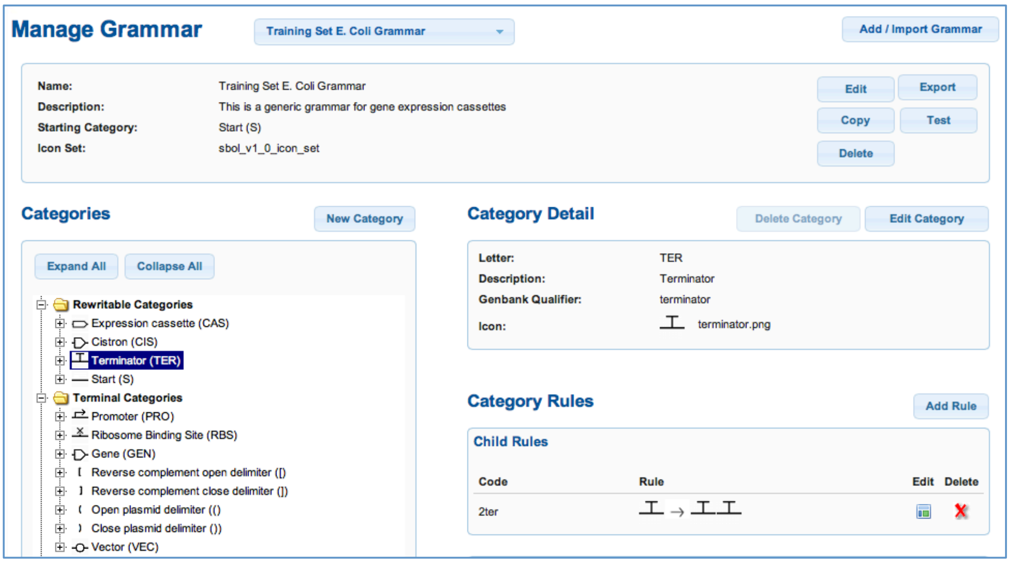
License: Apache 2.0

Funding: GenoCAD is supported by the National Science Foundation under Grant No. EF-0850100.

When it was released in 2007, GenoCAD was one of the first applications developed specifically to simplify the task of designing synthetic DNA sequences derived from libraries of genetic parts. GenoCAD enforces a design workflow constrained by specific design strategies formalized as context-free grammars (CFG). The grammar rewriting rules specify how parts of different categories may be combined. This theoretical framework supports a wizard-like sequence design tool that guides the users through a series of design decisions consistent with the grammar rewriting rules. The tool proved easy to use by biologists who may not have had previous exposure with linguistic models of DNA sequences.

Initially users could only choose from a set of built-in grammars when using the design tool. Our initial vision was to develop a few generic grammars for different types of protein expression hosts (bacteria, yeast, mammalian, plants). However, we soon understood that these generic grammars did not necessarily support the scientists' specific research needs. Without an editor to allow the users to develop their own grammars, users found the design tool cumbersome rather than helpful.

GenoCAD now offers a grammar editor, so users may revise existing grammars to meet their needs, or even develop new grammars from scratch. The development of a new CFG goes through a top-down process that breaks down large DNA sequences into categories corresponding to increasingly smaller DNA sequences. The grammar is therefore built by developing an abstraction hierarchy describing a set of DNA sequences generated by the CFG. Each step requires first defining parts categories, then production rules describing how they can be combined. As categories and rules are added to the grammar, it is convenient to identify three groups of categories. Rewritable categories are the ones that are used at least once on the left side of a production rule. Terminal categories are used only on the right side of rewriting rules. Finally, orphan categories correspond to categories that have not been used in any production rule. A well-designed grammar should not have any orphan categories. Once parts categories and production rules have been defined, the CFG can be completed by adding parts to categories, and, in particular, to all terminal categories.



Manage Grammar

Name: Training Set E. Coli Grammar
Description: This is a generic grammar for gene expression cassettes
Starting Category: Start (S)
Icon Set: sbol_v1_0_icon_set

Categories

- Rewritable Categories
  - Expression cassette (CAS)
  - Cistron (CIS)
  - Terminator (TER)
  - Start (S)
- Terminal Categories
  - Promoter (PRO)
  - Ribosome Binding Site (RBS)
  - Gene (GEN)
  - Reverse complement open delimiter ([)
  - Reverse complement close delimiter (])
  - Open plasmid delimiter (()
  - Close plasmid delimiter ())
  - Vector (VEC)

Category Detail

Letter: TER
Description: Terminator
Genbank Qualifier: terminator
Icon: ⊥ terminator.png

Category Rules

Child Rules

Code	Rule	Edit	Delete
2ter	⊥ → ⊥⊥	▦	✗

Poster 9

Improvements and new features in the 7th major release of the Bio-Linux distro

Timothy Booth, H. Soon Gweon, Mesude Bicak, Dawn Field
NERC Centre for Ecology and Hydrology, Maclean Building, Benson Lane, Wallingford, UK, OX10 8BB
http://nebc.nerc.ac.uk/tools/bio-linux
https://launchpad.net/~nebc/+archive/bio-linux
Incorporates packages under various OSI-approved licenses

The Bio-Linux project releases an Ubuntu-based Linux distro incorporating a large number of OSS bioinformatics tools and suitable for Linux beginners and experts alike. The goal is to provide a turnkey solution for accessing the latest FOSS bioinformatics tools and running standard pipelines on an integrated platform with the minimum of setup hassle. **The Bio-Linux project turns 10 this year**, and our long-term efforts in continuous maintenance mean users can rely on having a consistently high-quality and well documented platform with automated package updates. It has also enabled various new projects to build off the Bio-Linux as a base.

Bio-Linux 7, released in November, is a major update onto the latest Ubuntu LTS 12.04 platform and also adds many new capabilities and packages. The number of bundled tools has continued to increase, with the over 200 scientific packages now in the standard download providing over 1000 distinct application binaries.

Highlights of Bio-Linux 7 are:

- A basic installation of the **Galaxy** server and core tools runs out-of-the-box.
- Full and up-to-date **QIIME and Mothur** toolkits for analysis of microbial marker genes.
- The **PredictProtein** suite from the Rost lab.
- **Bowtie-Bio** tools (ie. Cufflinks, Tophat, Bowtie2) plus **Velvet and MIRA** assemblers
- The **CloudBioLinux** sister project (cloudbiolinux.org) includes all the software updates from Bio-Linux 7, ready to run in a cloud environment (eg. Amazon EC2).
- Bio-Linux is now tied more closely with the development of **Debian-Med** (http://wiki.debian.org/DebianMed), reducing duplication of packaging effort.
- **Courses and tutorial** material have been updated and expanded, with some video tutorials now becoming available

We provide an ISO image suitable for making a bootable USB stick, burning a DVD or loading directly into a VM system like VirtualBox. This can be booted directly into a "Live" desktop or else installed to the hard drive. Users work in a modern Unity desktop customised to allow rapid discovery and access of relevant tools, links and documentation.

Development of Bio-Linux is ongoing and upcoming new features will be discussed. We anticipate that virtualised (VM) deployment both locally and via cloud providers will become more common and we will show how Bio-Linux is being be enhanced to enable this. The Live version of Bio-Linux has already proven to be excellent for training, and we plan that it should be a key platform for the nascent training node of ELIXIR (http://www.elixir-europe.org/researchers/need-training).

We acknowledge the support of NERC, the NERC Biomolecular Analysis Facility (NBAF) and Dr. Casey Bergman at the University of Manchester.

Title: BioBlend - Enabling Pipeline Dreams

Authors: Clare Sloggett[1], Nuwan Goonasekera[1,2,4], Enis Afgan[1,3,4]
Author affiliations:
[1] Victorian Life Sciences Computation Initiative (VLSCI), University of Melbourne
[2] Victorian eResearch Strategic Initiative (VeRSI), University of Melbourne, Melbourne, Australia
[3] Center for Informatics and Computing (CIR), Ruđer Bošković Institute (RBI)
[4] Galaxy Project (http://usegalaxy.org)
{E.A. email: enis.afgan@irb.hr}

Projects' websites: http://bioblend.readthedocs.org
Projects' source code: https://github.com/afgane/bioblend
Open Source License used: MIT

Abstract

Since the advent of high-throughput sequencing technologies, genomics has become a large-data science with immense opportunity for biological insight. However, the analysis of such data is technically challenging, and collaboration between biologists and bioinformaticians is required to interpret the data. Galaxy is a popular and accessible platform for bioinformatics analysis, which provides many advantages such as visualisation, interactive analysis, reproducibility, and data and workflow sharing via a graphical interface. CloudMan is a cloud-based job runtime platform, which allows researchers to easily provision scalable 'virtual clusters' to run Galaxy and other applications in a cloud computing environment.

We created the BioBlend library, a unified API in a high-level language (python) that wraps the functionality of both Galaxy and CloudMan APIs. BioBlend exposes the programmable functionality of the two applications in a format that is more suitable for programming and thus makes it easier for bioinformaticians to automate end-to-end large-data analysis, from scratch. Because the end result of a data analysis is still available in the Galaxy environment, the resulting pipeline is highly accessible to collaborators. In combination with CloudMan, it is possible to both provision the required infrastructure, and automate complex analyses over large data sets on an as needed basis.

The library is easily installable via PyPi and comes with detailed documentation and example scripts in both the project website and the source code. This talk will provide an overview of the library, details of the available functionality, and highlight some of the available scripts.

Taverna Components: Semantically annotated and shareable units of functionality

Alan Williams, Donal Fellows, Finn Bacall,
Stian Soiland-Reyes, Khalid Belhajjame, David Withers, Carole Goble
{alan.r.williams, donal.k.fellows, finn.bacall, soiland-reyes, khalid.belhajjame, carole.goble }@manchester.ac.uk,
david.withers@gmail.com

School of Computer Science, University of Manchester, UK

Project's Web site: http://www.taverna.org.uk/

Source code: https://taverna.googlecode.com/svn/unsorted/component/trunk/component
Licence: GNU Lesser General Public License (LGPL) 2.1

Scientific workflows provide a powerful mechanism to create analysis pipelines using local or distributed tools and data resources. Workflow systems often provide the capability to include sub-workflows. In addition, workflows can be shared and re-used from sites such as myExperiment (http://www.myexperiment.org).

Workflows and tools are often poorly annotated making it difficult to understand their intended usage or functionality. As a result, it is hard for workflow designers to determine how services can be connected, and also how they can be easily replaced when a tool becomes redundant or goes offline. The lack of annotation also makes it difficult to "tweak" workflows to replace a tool to alter the purpose of the workflow or to allow it to cope with different data.

The BioVeL - Biodiversity Virtual e-Laboratory (http://www.biovel.eu), Scalable Preservation Environments (http://www.scape-project.eu) and Workflow 4 Ever (http://www.wf4ever-project.org) projects have collaborated on the specification and implementation of workflow components for the Taverna Workbench (http://www.taverna.org.uk). The components can be richly decorated with annotations that formally state their characteristics. This information can allow workflow applications to solve many of the problems described above.

A Taverna Component conforms to a Component Profile that describes the semantic annotations that can be associated with the component and also what exceptions and provenance will be produced by the component. Different types of components have different component profiles, so the different annotations can be placed on components for astronomy than for ecological niche modelling. Wf4ever have described a base component profile to which components must conform and which can be extended for particular types of component. The base component profile uses terms from ontologies and vocabularies such as Dublin Core and Provenance, Authoring and Versioning (http://purl.org/pav).

A Component can be realized by a workflow. The overall workflow, its input and output ports and the services within it can be annotated with predicates according to the chosen profile. Components can be stored, grouped within component families and shared within component registries. Capability has been added to myExperiment to allow it to be used as a common component registry. Components can be constrained by the license and sharing restrictions. Component definitions can be edited and new versions created.

The semantic annotations on a component are stored as RDF (in Turtle format). This annotation can be used as the basis for semantic searches for components, or workflows that use components, using constrained SPARQL queries. myExperiment is being extended to allow such searching through its REST API.

A component definition can be used within a workflow. When the component is invoked, it can be thought of as the invocation of the realizing workflow. A component within a workflow is tied to a specific version of the component. When other versions of the component exist, a user may update or rollback the version. The component annotations can be used to easily replace services within a workflow. This capability, combined with templates is being used within the BioVeL project to support wizard-enabled workflow creation.

This work was enabled by BioVeL, a project (Grant no. 283359) funded by the European Commission 7th Framework Programme (FP7) as part of its e-Infrastructures activity, and by the SCAPE project (Grant no. 270137) and the wf4ever project (Grant no. 270192) both funded by the European Commission 7th Framework Programme (FP7) as part of its ICT activity

UGENE Workflow Designer – flexible control and extension of pipelines with scripts

Yuriy Vaskin[1,2], Yuliya Algaer[2], Mikhail Fursov[2]
vaskin90@gmail.com, yalgaer@unipro.ru, mfursov@unipro.ru
[1]Novosibirsk State University, Novosibirsk, Russia
[2]Center of Information Technologies "Unipro", Novosibirsk, Russia

Unipro UGENE [Okonechnikov et al. 2012] is an open-source bioinformatics toolkit that integrates popular tools along with original instruments for molecular biologists within unified user interface. It has grown to a platform that can be applied for a large variety of tasks including multiple alignment, phylogeny, functional annotation, *in silico* cloning, protein structure analysis, TFBSs recognition. To perform various types of analysis UGENE gathers together computational algorithms, visualization capabilities and data processing instruments.

The key feature of UGENE platform is Workflow Designer. The Workflow Designer allows creating computational pipelines from built-in algorithmic blocks or by integrating external instruments. Biologists can construct their own pipelines by dragging elements in the scheme and connecting them with each other. It is also possible to choose a pipeline from the library of samples. Unlike Galaxy which is web-based, the Workflow Designer is a desktop pipeline manager which avoids data upload/download to a server, and can take advantage of the user's hardware.

Recent developments were focused on improvements of the workflow process and integration of NGS data processing pipelines like Cistrome [Liu et al. 2011], Tuxedo [Trapnell et al. 2012] and SAMTools [Li et al. 2009] variant calling. Special elements can filter data using conditions defined by users. There is an option to pause a computational process on any step and check intermediate data.

Among other features of the Workflow Designer scripting-related functionality is one of the most advanced. There are two different problems for which UGENE scripting capabilities can be applied. The first one has to do with running UGENE pipelines from other tools. Many bioinformatics systems use well-known tools for particular subtasks. For instance, analyzing BAM files with SAMTools [Li et al. 2009] or searching for similar sequences with BLAST [Stephen et al. 1997]. These two subtasks might be the initial steps of analysis in a large system. To use UGENE pipelines as a part of a bigger system there is an API that provides bindings to high-level programming languages. Now UGENE provides bindings for Node.js. It is planned to develop a similar API for Python. So any UGENE pipeline can be run from Node.js code through the API and scripts can use result of computations for further processing.

The second problem is that researchers often need to customize an existing pipeline for their needs. For instance, add a filtration or a data modification step. Although the changes in the pipeline could be very small, the implementation might require editing the source code of an application. To satisfy various customization requests UGENE Workflow Designer integrates scripting language with JavaScript syntax. Each element in a pipeline can be extended with a script. Objects like sequences, annotations or multiple alignments that flow through an element can be modified or filtered. Scripting language provides a set of operations for that objects. Apart from those operations any JavaScript code can be run, for instance, for statistics computations or extra outputs.

Project Web Site: http://ugene.unipro.ru/
Software and source code: http://ugene.unipro.ru/download.html
License: GNU General Public License v.2

Oqtans: A Multifunctional Workbench for RNA-seq Data Analysis

Vipin T. Sreedharan*,[1], Sebastian J. Schultheiss*,[2], Géraldine Jean*,[2], André Kahles[1], Regina Bohnert[2], Philipp Drewe[1], Pramod Mudrakarta[2], Nico Görnitz[3], Georg Zeller[4], and Gunnar Rätsch[1]

1 Computational Biology Center, Memorial Sloan-Kettering Cancer Center, New York, USA
2 Friedrich Miescher Laboratory, Max Planck Society, Tübingen, Germany
3 Machine Learning/Intelligent Data Analysis Group, Technical University Berlin, Germany
4 Structural and Computational Biology Unit, European Molecular Biology, Germany

Keywords: Deep Sequencing, RNA-Seq, quantitation, assembly, gene expression

Abstract

The current revolution in sequencing technologies allows us to obtain a much more detailed picture of transcriptomes via deep RNA Sequencing (RNA-Seq). In considering the full complement of RNA transcripts that comprise the transcriptome, two important analytical questions emerge: what is the abundance of RNA transcripts and which genes or transcripts are differentially expressed. In parallel with sequencing technology development, data analysis software is also constantly updated to improve accuracy and sensitivity while minimizing run times. The abundance of software programs, however, can be prohibitive and confusing for researchers to determine which to use for their RNA-Seq anlaysis.

We present an open-source workbench, *Oqtans*, that can be integrated into the Galaxy framework that enables researchers to set up a computational pipeline for quantitative transcriptome analysis. Its distinguishing features include a modular pipeline architecture, which facilitates comparative assessment of tool and data quality. Within *Oqtans*, the *Galaxy*'s workflow achitecture enables direct comparison of several tools. Furthermore, it is straightforward to compare the performance of different programs and parameter settings on the same data and choose the best suited for the task. *Oqtans* analysis pipelines are easy to set up, modify, and (re-)use without significant computational skills.

Oqtans integrates more than twenty sophisticated tools that perform very well compared to the state-of-the-art for transcript identification/quantification and differential expression analysis. The toolsuite contains several tools developed in the Rätsch Laboratory, but the majority of the tools were developed by other groups. In particular, we provide tools for alignment (bwa, tophat, PALMapper, ...), transcript prediction (cufflinks, trinity, mTIM, ...) and quantitative analyses (DESeq, rDiff, rQuant, ...). In addition, we provide tools for alignment filtering (RNA-geeq toolbox), GFF file processing (GFF toolbox) and tools for predictive sequence analysis (easySVM, ASP, ARTS, ...). See `http://oqtans.org/tools` for more details on included tools.

Oqtans is integrated into the *Galaxy* server `http://galaxy.raetschlab.org` maintained by the Rätsch Laboratory. It is also available as source code in a public github repository `http://bioweb.me/oqtans/git` and as an Amazon Machine Image for the AWS cloud environment (instructions available at `http://oqtans.org`). Finally, *Oqtans* sets a new standard in terms of reproducibility and builds upon Galaxys features to facilitate persistent storage, exchange, and documentation of intermediate results and analysis workflows.

URL for the overall project web site: `http://oqtans.org`
URL for accessing the code: `http://bioweb.me/oqtans/git`
Public instance: `http://galaxy.raetschlab.org`
Support: `support@oqtans.org`
Contact: `ratschg@mskcc.org`

*

* Authors contributed equally.

PhyloCommons: Sharing, Annotating and Reusing Phylogenies.

Benjamin Morris(1) and Hilmar Lapp(2)
(1) University of North Carolina at Chapel Hill, Chapel Hill, NC, USA. ben@bendmorris.com
(2) National Evolutionary Synthesis Center (NESCent), Durham, NC, USA. hlapp@nescent.org

Homepage: http://www.phylocommons.org
Source: http://github.com/bendmorris/phylocommons *(available under the MIT license)*

Phylogenetic trees have numerous applications across biology, including comparative genomics and microbial ecology, in particular in accounting for shared evolutionary ancestry and thus non-independence of data in comparative studies. Owing to continuing methodological and technological advancements, phylogenies are being generated and published at an accelerating rate, with increasingly comprehensive taxonomic coverage. However, despite the existence of community resources for archiving phylogenies (TreeBASE), most phylogenies remain difficult to reuse. The predominant reasons include that they are often not shared online in a way that allows easy discovery and machine consumption, and important metadata regarding their underlying data and method of construction can be difficult to find.

To address these obstacles, we have created PhyloCommons, a system for ingesting, annotating, discovering, sub-setting, and retrieving phylogenetic trees using interoperable technologies guided by Linked Open Data (LOD) principles. PhyloCommons is designed to be lightweight enough to be deployed for small groups or by individual researchers, while still being sufficiently scalable to serve as a data sharing resource for whole communities. It consists of a tree store implemented as a Resource Description Framework (RDF) triple store and a web-application that serves as an interactive front end. The tree store uses the Comparative Data Analysis Ontology (CDAO) standard to represent the data elements of phylogenetic trees, and can link those elements to any kind of annotation expressed in RDF. Accompanying software utilizes the Biopython toolkit to support ingesting and retrieving trees in commonly used formats, including the emerging NeXML standard. Study-, tree-, and node-level metadata can be used to find trees and to filter query results. Matching trees can be retrieved as subtrees containing only the taxa of interest. The front end web-application is intended to enable collaborative tree storage and annotation, where users can upload, annotate, query, and download trees, allowing researchers to more easily find, augment, and reuse existing data and thereby benefit from each others' efforts and expertise. PhyloCommons can also serve as the central tree store component within the Phylotastic architecture, a specification of distributed interoperable components for sharing, discovering, and reusing phylogenetic data developed at a series of recent hackathons organized by the Hackathons, Interoperability, Phylogenies (HIP) Working Group at the National Evolutionary Synthesis Center (NESCent).

Our goal with PhyloCommons, and the larger Phylotastic architecture, is to increase the availability and utility of phylogenies by removing obstacles to publishing them on the web using standards that enable users to easily find and access trees that match the needs of their research and their analytic tools.

Funding: This work was supported by NESCent (NSF #EF-0905606), the iPlant Collaborative (NSF #DBI-0735191), and the Biodiversity Synthesis Center (BioSynC).

GEMBASSY: an EMBOSS associated package for genome analysis using G-language SOAP/REST web services

Hidetoshi Itaya[1] (celery@g-language.org), Kazuki Oshita[1, 2], Kazuharu Arakawa[1, 2], Masaru Tomita[1, 2]
[1]Institute for Advanced Biosciences, Keio University
[2]System Biology Program, Graduate School of Media and Governance, Keio University

URL (project): http://www.g-language.org/gembassy/
URL (code): https://github.com/celery-kotone/GEMBASSY
License: GNU General Public License

The popular European Molecular Biology Open Software Suite (EMBOSS) currently contains over 400 tools used in various bioinformatics researches, equipped with sophisticated development frameworks for interoperability and tool discoverability as well as rich documentations and various user interfaces. In order to further strengthen EMBOSS in the fields of genomics, we here present a novel EMBOSS associated software (EMBASSY) package named GEMBASSY, which adds more than 50 analysis tools from the G-language Genome Analysis Environment (G-language GAE) [1-3] and its Representational State Transfer (REST) and SOAP web services [4]. G-language GAE is a workbench for genome analysis mainly targeted to bacteria, which is equipped with more than 100 methods for various analyses. GEMBASSY basically contains wrapper programs of G-language REST/SOAP web services to provide intuitive and easy access to various annotations within complete genome flatfiles, as well as tools for analyzing nucleic composition, calculating codon usage, and visualizing genomic information. Web service APIs were used in order to lower the cost of implementation by developing the package based on the Keio Bioinformatics Web Service (KBWS) [5] package which also provides wrappers to SOAP service methods. GEMBASSY is capable of performing, for example, analysis methods such as for calculating distance between sequences by genomic signatures and for predicting gene expression levels from codon usage bias are effective in the interpretation of meta-genomic and meta-transcriptomic data. GEMBASSY tools can be used seamlessly with other EMBOSS tools and UNIX command line tools.

References

1. Arakawa K, Mori K, Ikeda K, Matsuzaki T, Kobayashi Y, Tomita M: G-language Genome Analysis Environment: a workbench for nucleotide sequence data mining. *Bioinformatics* 2003, **19:**305-306.
2. Arakawa K, Tomita M: G-language System as a platform for large-scale analysis of high-throughput omics data. *J Pestic Sci* 2006, **31:**282-288.
3. Arakawa K, Suzuki H, Tomita M: Computational Genome Analysis Using The G-language System. *Genes, Genomes and Genomics* 2008, **2:**1-13.
4. Arakawa K, Kido N, Oshita K, Tomita M: G-language genome analysis environment with REST and SOAP web service interfaces. *Nucleic Acids Res* 2010, **38:**W700-705.
5. Oshita K, Arakawa K, Tomita M: KBWS: an EMBOSS associated package for accessing bioinformatics web services. *Source Code Biol Med* 2011, **6:**8.

Title: Rubra - flexible distributed pipelines for bioinformatics

Authors: Clare Sloggett[1], Gayle Philip[1], Matthew Wakefield[1], Bernard Pope[1,2]
Author affiliations:
[1] Victorian Life Sciences Computation Initiative (VLSCI), The University of Melbourne, Australia.
[2] The Department of Computing and Information Systems, The University of Melbourne, Australia.
Presenting author email: sloc@unimelb.edu.au

Projects' websites: https://github.com/bjpop/rubra
Projects' source code: https://github.com/bjpop/rubra
Open Source License used: MIT

We present Rubra, a command-line tool for running bioinformatics pipelines, built on the Ruffus library (http://www.ruffus.org.uk/)[1]. Rubra provides a simple syntax for executing Ruffus-based pipelines. The Ruffus framework allows users to write declarative python scripts describing a pipeline in terms of execution logic and task dependencies. Rubra makes use of Ruffus' features to provide dependency checking and automated parallelisation of tasks, logging, control of pipeline execution, and to allow visualisation of task dependencies. In addition, Rubra provides several enhancements on top of Ruffus: a convenient command-line syntax with configuration files to describe pipeline parameters, the ability to run jobs either locally or on HPC job schedulers (such as Torque/PBS), and job checkpointing to ensure task completion. VLSCI maintains Rubra-based pipelines which are used to manage analyses on our compute clusters. We will present an overview of Rubra along with examples of implementing end-to-end exome and whole-genome variant calling pipelines.

[1] *Ruffus: a lightweight Python library for computational pipelines,* Goodstadt L., Bioinformatics 2010 Nov 1;26(21):2778-9

Title: Towards Enabling Big Data and Federated Computing in the Cloud

Authors: Yousef Kowsar[1], Enis Afgan[1,2,3]
Author affiliations:
[1] Victorian Life Sciences Computation Initiative (VLSCI), University of Melbourne
[2] Center for Informatics and Computing (CIR), Ruđer Bošković Institute (RBI)
[3] Galaxy Project (http://usegalaxy.org)
{E.A. email: enis.afgan@irb.hr}

Projects' websites: http://usecloudman.org
Projects' source code: https://bitbucket.org/galaxy/cloudman
Open Source License used: MIT

Abstract

As the information age continues, the rate at which data is produced is continuing its exponential growth. Although often primarily described as a challenge and an obstacle, the reality is that the availability of the increasing data volume presents enormous opportunities. The real power of the data will not come just from the sheer volume, but from the ability to analyze it. It is thus vital to provide flexible yet accessible solutions that enable researchers to move beyond the data collection and step into the world of data analytics.

Simultaneously, Infrastructure-as-a-Service (IaaS) compute infrastructure model (i.e., the cloud) has showcased its ability to transform how access to compute resources is realized; it delivered on the notion of Infrastructure-as-Code and enabled a new wave of compute adaptability. Over the past few years, we have been developing CloudMan as a versatile solution for enabling and managing compute clusters in cloud environments via a simple web interface or an API. However, CloudMan only supported batch processing workloads. As the magnitude of the data produced and processed in digital form grows, the need to support more types of applications in the cluster-in-the-cloud model is becoming more evident.

We have thus extended the batch processing capability of CloudMan and added support for different types of analysis workloads to the created cloud environment. This was accomplished by adopting and utilizing a well established big data platform component: Hadoop [9] and adding support for federated computing using HTCondor. With these additions, CloudMan provides support for three types of workloads: batch, Hadoop, and federated. This talk will discuss the new opportunities behind this solution, the developed architecture, and showcase usability of the available implementation.

Title: MyGene.info: Making Elastic and Extensible Gene-centric Web Services
Authors: Chunlei Wu and Andrew I. Su (presenting author underlined)
Email: cwu@scripps.edu asu@scripps.edu
Affiliations: The Scripps Research Institute, 10550 N Torrey Pines Rd, La Jolla, CA 92037
Project web site: http://mygene.info
Source code: https://bitbucket.org/sulab/genedoc-hub/src
Open Source License being used: **Apache License**
Considered for a talk, a poster, or both: **talk**

Web applications are increasingly becoming the most common and convenient way for scientists to publish and exchange research data and results. Many biological web applications provide an interface for users to query for their favorite genes and then display the data relevant to the gene hits. Building an interface like that often requires developers to maintain a dedicated gene annotation database to translate user queries into the desired gene identifiers, and also to obtain the specific gene annotation data based on these identifiers. Setting up a database server and keeping it updated can be a time-consuming and cumbersome tasks. Since the majority of raw gene annotation data are coming from several large data centers like NCBI and Ensembl, developers are also duplicating their efforts to setup gene annotation databases from essentially the same data providers.

MyGene.info (http://mygene.info) is a cloud-based solution to abstract the task of building a gene annotation database into a set of elastic and extensible web services. End users have access to two simple-to-use REST web services for gene query and gene annotation retrieval, without the need to worry about designing, building and maintaining a dedicated database. The gene query service takes the user query string and returns the matching gene hits with desired identifiers; and the gene annotation service returns annotation data for given gene IDs. Both services return JSON (Javascript Object Notation) formatted data, making them easy to integrate into web applications.

MyGene.info web services are elastic, which means they are always on and always up-to-date. They are built on a cluster of instances in AWS EC2 environment. It's trivial to scale up the cluster to handle increased bandwidth. The data updating process happens continuously, so each instance is always synchronized with the sources, without bringing the services down.

MyGene.info web services are extensible. Our database currently includes 12 million genes from all species (~11K species) supported by NCBI and Ensembl and 40 different types of annotations from NCBI, Ensembl, Uniprot, PharmGKB, NetAffy, etc. The data-loading module can be easily extended to add additional types of annotation data.

On the technical side, MyGene.info stores gene annotation data in MongoDB, a NoSQL database, and an ElasticSearch cluster is used to provide high-performance and rich query functions. A Python/Tornado layer provides a user-friendly web services API.

An update on the Seal Hadoop-based sequence processing toolbox

Luca Pireddu, Simone Leo, Gianluigi Zanetti
CRS4, Pula, Italy
Email: luca.pireddu@crs4.it

Web site URL:	`http://biodoop-seal.sourceforge.net/`
Code URL:	`git://git.code.sf.net/p/biodoop-seal/code`
License:	GPLv3

Regular advances in high-throughput DNA and RNA sequencing technologies are continuously pushing the limits of typical bioinformatics data processing techniques. Medium-sized sequencing laboratories can generate Terabytes of data per week; large laboratories can produce even more. Unfortunately, most software tools available for sequence processing are not designed to scale easily to such high data rates, nor are the typical bioinformatics workflow designs. Data scalability issues such as these have already been faced by the "big data revolution" in data-based activities resulting in novel computational paradigms such as MapReduce and computing frameworks such as Hadoop.

Seal is a suite of tools that harnesses the Hadoop framework to process sequencing data. It is currently used in the production pipeline at the CRS4 Sequencing and Genotyping Platform, which houses 3 Illumina HiSeq 2000 sequencers for a total capacity of about 5000 Gbases/month. While in its first release Seal only included tools to perform sequence alignment on Hadoop (with an embedded version of BWA [1]), it has since grown, gradually removing processing bottlenecks with new scalable tools. The current (stable) `master` branch includes the following Hadoop-based distributed processing tools:

Demux: demultiplex reads from a multiplexed sequencing run;

Prq: reformat reads in `qseq` or `fastq` format in the `prq` format for alignment with Seqal;

Seqal: BWA-based distributed read mapping and duplicate identification;

ReadSort: distributed read sorting based on read id or alignment position;

RecabTable: extract empirical base quality statistics (for recalibration).

Shortly Seal will also acquire Hadoop-based tools to convert Illumina BCL files (produced by the sequencer) to `fastq` and to recalibrate base qualities; the former has already been written and will be included in the next release.

The Seal tools can be chained consecutively to implement most of the typical variant calling pipeline. At CRS4 work has been done to integrate Seal with Galaxy in order to manage the workflows through the popular web application, while the toolbox has also been independently integrated into other high-level workflow tools such as Clougene [3]. In addition, Seal can also be used as a library, borrowing its functionality for new custom and complementary applications—e.g., SeqPig (`http://seqpig.sf.net/`).

Seal tools have been shown to scale well in the amount of input data and the amount of computational nodes available [2]; therefore, with Seal one can increase processing throughput by simply adding more computing nodes. Moreover, thanks to the robust platform provided by Hadoop, the effort required by operators to run the analyses on a large cluster is generally reduced, since Hadoop transparently handles most hardware and transient network problems, and provides a friendly web interface to monitor job progress and logs. Finally, the Hadoop Distributed File System (HDFS) provides a scalable storage system that scales its total throughput and volume hand in hand with the number of processing nodes. Thus, it avoids creating a bottleneck at the shared storage volume and the need for an expensive high-performance parallel storage device.

References

[1] Heng Li and Richard Durbin. Fast and accurate short read alignment with Burrows-Wheeler Transform. *Bioinformatics*, 25(14):1754—1760, 2009.

[2] Luca Pireddu, Simone Leo, and Gianluigi Zanetti. Seal: a distributed short read mapping and duplicate removal tool. *Bioinformatics*, 27(15):2159–2160, 2011.

[3] Sebastian Schonherr, Lukas Forer, Hansi WeiSZensteiner, Florian Kronenberg, Gunther Specht, and Anita Kloss-Brandstatter. Cloudgene: A graphical execution platform for mapreduce programs on private and public clouds. *BMC Bioinformatics*, 13(1):200, 2012.

Title: Open Source Configuration of Bioinformatics Infrastructure

Authors: John Chilton[1] (chilton@msi.umn.edu), Pratik Jagtap[1], Benjamin Lynch[1], Brad Chapman[2], Timothy Griffin[3]

[1] University of Minnesota Supercomputing Institute, Minneapolis, Minnesota, USA

[2] Harvard School of Public Health

[3] University of Minnesota, Minneapolis, Minnesota, USA

CloudBioLinux: http://cloudbiolinux.org/ | https://github.com/chapmanb/cloudbiolinux

License: MIT

Mirroring broader information technology trends, bioinformatics software systems are growing more interconnected, complex and being deployed in more diverse environments (lab servers, institutional data centers, public and private clouds). In this light, the appeal of open source projects such as CloudBioLinux to automate the configuration of such systems is apparent.

While Fabric, the remote execution library CloudBioLinux is built on, is an excellent tool for the automating the building of command-line applications or installation of system-level packages, it is not a full featured configuration management tool and is likely not the ideal way to configure complex, interconnected, customized web applications. Puppet and Chef currently dominate the configuration management landscape and go beyond deployment tools such as Fabric by offering high-level built-in templating of configuration files, constructs for managing composition and declaring dependencies, and easy unit testing.

This talk will describe extensions to CloudBioLinux to provide deep integration between CloudBioLinux and both Puppet and Chef. CloudBioLinux idioms such as flavors and packages can be used to describe what classes (Puppet) and recipes (Chef) are installed and the same configuration properties file can be used to customize these configurations.

Time permitting, this integration will be illustrated quickly with some example applications. A puppet module to configure an LWR server (an application which lets the popular Galaxy framework run jobs remotely without a shared file system) will be presented. As will integration with the Globus Online Chef cookbooks which can be used to install and configure Globus client utilities on a CloudBioLinux image.

The usage of git submodules paired with standalone repositories enable these configuration repositories to be readily reused and extended outside the context of CloudBioLinux, for instance in combination with other tools such as Globus Provision or in the large institutional repositories typically maintained by infrastructure teams. Ultimately, the goal of this talk is not to advocate for CloudBioLinux, but for the creation a collection of well tested, extensible, interoperable configuration modules for a variety of bioinformatics applications.

GEPETTO: An Open Source Framework for Gene Prioritization

Vincent Walter, Julie Thompson, Olivier Poch and Hoan Nguyen
Contact: nguyen@igbmc.fr
Laboratoire de Bioinformatique et Génomique Intégratives, IGBMC

In the era of omics 'big data', and in particular next-generation sequencing (NGS), gene prioritization[1] is a crucial task, involving the integration of huge amounts of heterogeneous data and the subsequent selection and analysis of genes predicted to be involved in a specific biological process, such as a pathology. Large sets of genes must be evaluated, in order to score and rank them according to their similarity to known genes and their potential viability as candidates for important applications, such as diagnostic/prognostic markers, drug targets, etc. The biomedical community urgently needs a customizable and extensible framework for gene selection that can handle large-scale biological information from public, as well as private data resources. To our knowledge, no other open source framework for gene prioritization has previously been developed.

GEPETTO (GEne PrioriTization Tool) is an original open-source framework, distributed under the LGPL license, for gene selection and prioritization on a desktop computer that ensures confidentiality of personal data. It takes advantage of the data integration capabilities in the SM2PH-Central knowledgebase[2,3,4], combined with in-house developed gene prioritization methods. It currently incorporates six prioritization modules, based on gene sequence, protein-protein interactions, gene expression, disease-causing probabilities, protein evolution and genomic context).
Gepetto is written in Java/Python and supported by an advanced modular architecture, which means that it can easily be modified and extended by the user, in order to include alternative scoring methods and new public/private data sources. The GEPETTO software is available at sourceforge.net/projects/gepetto/files/ or decrypthon.igbmc.fr/sm2ph/cgi-bin/gepetto.

1) Moreau Y, Tranchevent LC. Computational tools for prioritizing candidate genes: boosting disease gene discovery. Nat Rev Genet. 2012 Jul 3;13(8):523-36.
2) Friedrich A, Garnier N, Gagnière N, Nguyen H, Albou LP, Biancalana V, Bettler E, Deléage G, Lecompte O, Muller J, Moras D, Mandel JL, Toursel T, Moulinier L, Poch O. SM2PH-db: an interactive system for the integrated analysis of phenotypic consequences of missense mutations in proteins involved in human genetic diseases. Hum Mutat. 31:127-35 (2010).
3) Luu TD, Rusu AM, Walter V, Ripp R, Moulinier L, Muller J, Toursel T, Thompson JD, Poch O, Nguyen H. MSV3d: database of human MisSense Variants mapped to 3D protein structure. Database (Oxford). 2012:bas018 (2012).
4) Luu TD, Rusu A, Walter V, Linard B, Poidevin L, Ripp R, Moulinier L, Muller J, Raffelsberger W, Wicker N, Lecompte O, Thompson JD, Poch O, Nguyen H. KD4v: Comprehensible Knowledge Discovery System for Missense Variant. Nucleic Acids Res. 40:W71-5 (2012).

RAMPART (a Robust Automatic Multiple AssembleR Toolkit)

Daniel Mapleson (daniel.mapleson@tgac.ac.uk), Bernardo Clavijo (bernardo.clavijo@tgac.ac.uk), Nizar Drou (nizar.drou@tgac.ac.uk)

The Genome Analysis Centre (TGAC)

URL: https://github.com/TGAC/RAMPART - License: GPL v3

The *de novo* assembly of genomes from modern sequencing devices is a computationally intensive, multi-stage task. It is typical, after sequencing and quality assessment, that a pipeline involving read analysis, quality trimming, contig assembly, scaffolding and gap closing is executed in order to build a first pass assembly. Each step in the pipeline requires careful analysis and decision making before proceeding to the next step.

Here we present RAMPART, a pipeline for automating the production of first pass assemblies. RAMPART supports a variety of assemblers and scaffolding tools, can assist with assembly validation and, if requested, make decisions automatically. Each step in the pipeline produces statistics and plots to help interpret, compare and visualise results. This assists the user to explain and justify decisions that were taken. An assembly validation step helps to assess the quality of the final assembly, which involves read alignment, feature response curves, and other novel validation techniques. Finally RAMPART can produce a final report describing the assembly process and validation results across all stages.

RAMPART is developed at TGAC and is built on top of a flexible multi-layer architecture, which is designed in order to efficiently leverage computational resources across different HPC environments. The bottom layer of the architecture is a customised version of EBI's Conan pipeline, a light-weight workflow management application. The customisations enable conan tasks to run within an execution context, which defines how and where each Conan process is executed. For example, processes can be executed with or without a job scheduling service, such as LSF. The middle layer is a series of java classes which wrap external tools as self contained conan classes, which implement predefined interfaces to enable compatibility between tools. These wrapped processes can then to construct any Conan pipeline, promoting code reuse. The top layer is the RAMPART pipeline itself, which implements the pipeline and domain logic. In the future we will reuse the lower layers of the architecture to rapidly build new pipelines for other bioinformatics tasks.

RAMPART is currently used at TGAC to run production jobs on TGAC's LSF computing cluster. For large memory jobs, it will be possible to execute the pipeline on TGAC's UV systems, which use PBS. However, the software is open source and should work in most unix environments without a scheduling system. Only minor changes to a configuration file are required to accomplish this. RAMPART is currently a command line tool but in the future we aim to include a web-front end to control and track progress.

OmicsConnect: flexible multi-omics data capture and integration tools for high-throughput biology

K. Joeri van der Velde[1,3], Robert K. Hastings[2,3], Charalambos Chrysostomou[2,3], Chao Pang[1,3], Dennis Hendriksen[1,3], Anthony J. Brookes[2,3] and Morris Swertz[1,3]

1: Genomics Coordination Center, University Medical Center Groningen, The Netherlands
2: Department of Genetics, University of Leicester, United Kingdom
3: EU-BioSHaRE, EU-BioMedBridges and EU-GEN2PHEN consortia
Contact: k.j.van.der.velde@umcg.nl
Project website: http://www.molgenis.org
License: GNU Lesser General Public License version 3 (GNU LGPLv3)

Recent advances in molecular characterization generate many high-throughput multi-omics data for biomedical research; including transcriptomics, proteomics and metabolomics but also epigenomics, pharmacogenomics, toxicogenomics and epidemiological phenotypes. Integrating and analyzing these large multi-dimensional data in an effective and reproducible way across experiments and data types is an ever increasing challenge.

Existing tools to enable data integration include the MOLGENIS platform, XGAP 'omics' database, Observ-OM 'pheno' model, xQTL 'workbench', and successful adaptations such as WormQTL.org. However, now biologists want an all-in-one package.

Here we present the next generation in Observ-OMX, a data model that captures all data modalities and context information in a flexible and powerful way. 'Best of' practices from existing domain-specific models are harmonized and re-used via the core model. Observ-OMX captures the essential features, protocols, targets and values of any experimental setup without sacrificing details or strong data types.

Also we present OmicsConnect, a software toolbox build using the Observ-OMX model for labs and consortia to import, catalogue, manage, query, and analytically interface with large multi-omics data and complex phenotypes in biobanks, translational medicine, epidemiology, genetics and model organism studies. A straightforward tabular data format allows researchers to upload and exchange data without the need for technical know-how.

OmicsConnect enables big data import, management and exploration capitalizing the Observ-OMX standard as programming foundation for integrative analysis and visualization tools (in Java, R, REST, Genome browser, web user interface). This enables integrated multi-omics data management, browsing and filtering, across modules for xQTLs, GWAS, NGS, pathogenicity reports, big data compute pipelines, biobank cataloguing and sharing. All is implemented in MOLGENIS and available as open source at http://github.com/molgenis.

Title	Community development of human variant calling and validation pipelines
Author	*Brad Chapman*, Rory Kirchner, Oliver Hofmann, Winston Hide
Affiliation	Harvard School of Public Health
Contact	bchapman@hsph.harvard.edu
URL	https://github.com/chapmanb/bcbio-nextgen
License	MIT

Translational research relies on accurate identification of human genomic variants from populations, families and cancer tumor/normal pairings. Prioritization of informative variants starts with a quality set of single nucleotide changes, insertions and deletions, copy number variations and larger structural rearrangements. However, rapidly changing best practice approaches in alignment and variant calling, coupled with large data sizes, make it a challenge to develop scalable, accurate pipelines. Coordinated community development of these pipelines can help overcome these challenges by sharing testing and updates across the numerous groups relying on the same infrastructure.

We will describe bcbio-nextgen, a distributed multi-architecture pipeline that automates variant calling, validation and organization of results for query and visualization. It creates an easily installable and runnable infrastructure from best-practice open source tools:

- Multiple variant calling approaches using pipelines from Broad's GATK best practices and the Marth lab's gkno software.
- Validation of calls against known reference materials developed by the Genome in a Bottle consortium. The bcbio.variation toolkit automates scoring and assessment of calls to identify regressions in variant identification as calling pipelines evolve.
- Tracking of software versions, command lines and file provenance using the BioLite framework.
- Query and investigation of variants using the gemini framework, providing a scalable SQLite database with biology specific query capability.
- Visual variant prioritization and false positive detection using o8, a web-based tool for exploring variation data.
- Fully automated installation of third party software and required reference genomes using CloudBioLinux.

We use bcbio-nextgen on large population studies handling hundreds of whole genome samples and will detail our approaches to scaling across multiple architectures. The pipeline parallelizes on a wide variety of schedulers (LSF, SGE, Torque, PBSPro) and multicore machines. Work in progress includes porting to cloud environments, including Amazon Web Services with StarCluster and Microsoft Azure. Finally, we'll describe plans to integrate with web-based front ends like STORMseq that democratize the pipelines for use by both researchers and non-scientists.

Understanding Cancer Genomes Using Galaxy

Jeremy Goecks (jeremy.goecks@emory.edu)[1], H. Jean Khoury[2], Bassel El Reyes[2], The Galaxy Team[3], James Taylor[1], and Michael R. Rossi[4],
[1]Departments of Biology and Math & Computer Sciences, Emory University
[2]Department of Radiation Oncology, School of Medicine, Emory University
[3]http://galaxyproject.org
[4]Department of Hematology and Medical Oncology, School of Medicine, Emory University
Website: http://galaxyproject.org, Code: http://bitbucket.org/galaxy/galaxy-central/
License: Academic Free License

The ability to rapidly identify and interpret complex molecular changes in cancer genomes is transforming approaches to treating and managing high risk disease. Comprehensive and integrated assessment of gene mutations, structural rearrangements, and differential gene expression in mixed populations of cells is crucial to understanding cancer biology and the processes leading to malignant transformation. Therefore, development of open-source tools that can assist in the interpretation of cancer genomes is crucial to advancing our knowledge and improving outcomes for all cancer patients.

There are few open-source pipelines for analyzing cancer genomes, and those that are available typically require programming experience that many cancer researchers may not have. We have developed pipelines and visualizations for the popular Web-based genomics platform Galaxy (http://usegalaxy.org) that enable anyone to analyze high-throughput sequencing data from cancer genomes using only a Web browser.

Our Galaxy cancer genomics pipelines enable analysis of both exome/whole genome and transcriptome sequencing data. Using resequencing data, the exome/whole genome workflow identifies small variations (i.e., single-base mutations, insertions, and deletions) based on a reference genome and provides options for annotating and filtering variants. The transcriptome workflow analyzes RNA-seq data to find gene fusions and small variations, calculate gene expression levels, and perform differential expression. Finally, we have developed an integrative pipeline that combines outputs from both workflows to (a) identify high-confidence small variations that appear in both resequencing and RNA-seq data and (b) derive allele-specific expression. All pipelines include only open-source tools.

We have also enhanced Galaxy's visualizations by developing an interactive Circos plot and by implementing support for large-scale VCF files. The Circos visualization can be used to simultaneously explore genome-wide data and genomic rearrangements. Support for viewing VCF files derived from many tumor samples has been added to Trackster, Galaxy's genome browser.

We have evaluated variants produced by these pipelines by analyzing targeted exome sequencing and whole sequencing data from the pancreatic cancer cell line MiaPaCa2. The exome analysis pipelines identified all known variants from the Cancer Cell Line Encyclopedia (CCLE), while the transcriptome pipeline identified a subset of known variants. The integrated pipeline found 11 additional high-confidence variants that are not reported in the CCLE. We applied the transcriptome analysis pipeline to 3 acute myeloid leukemia patients with different outcomes. The allelic expression results from this analysis provides insight into why the patients' outcomes differed, further demonstrating how Galaxy can be used to interpret and integrate data from cell lines and primary cancer specimens.

Emergence: data-driven pipeline discovery interface integrating multiple bioinformatics platforms

Joshua Orvis[1], Anup Mahurkar[1], Owen White[1]
[1] University of Maryland School of Medicine, Institute for Genome Sciences

Project site: https://code.google.com/p/emergence/
Getting the code: svn co http://emergence.googlecode.com/svn/trunk/ Emergence
License: GNU GPL v3

There is a variety of useful bioinformatics analysis pipeline platforms (Galaxy, Taverna, Ergatis, etc.) but also an unfounded notion that users should have to choose between them. Each has a rich set of stored pipeline and analysis templates which should be available regardless of the framework they're stored in, yet no interface exists which provides easy integration and extension of these combined with visualization of their interconnectivity and output.

These utilities are extremely useful, but are also very tool-centric, presenting lists of tools and pipelines available without any consideration of the data users actually have available. We have envisioned a system which takes the opposite, data-centric approach, where users are guided through available analyses based on the data they present. If a user specifies paired-end Illumina genomic reads as input, for example, Emergence shows options such as sequence assembly or digital normalization. When a reference genome is added the interface also presents a SNP pipeline implemented within Galaxy. Available analysis tools and pipelines emerge from the data the user presents.

While Emergence does provide an analysis and visualization framework of its own, there is an emphasis on integration of existing tools, which are added by defining input/output datatype specifications and execution parameters. It initially has support for local and grid execution of jobs via Grid Engine, with planned extensions to support both Platform LSF and the generic DRMAA API.

The project is at an early development stage and is written using Python3 and the Django web framework on the server-side, with a UI implemented in HTML5, JQuery and CSS3. Its development has been guided by the needs of active projects in eukaryotic annotation, RNA-Seq, SNP analysis and metagenomics.

A Targeted Approach To Sequence Generation and Artificial Phylogenies

Khalique Williams

McGill University (khalique.williams@mail.mcgill.ca)

http://khwillia.github.io/naive-needleman-wunsch-sequence-generator/

https://github.com/khwillia/naive-needleman-wunsch-sequence-generator/zipball/master (download)

GNU General Public License v3.0

ABSTRACT

In this paper we provide a novel approach to sequence generation and the maintenance of artificial phylogenies. This new approach follows an evolutionary model where no data of the changing operations are lost between mutations (as they are in real-life). Furthermore any two members chosen at random from the entire pool of generated sequences, would each be representative of a germline at a particular instance in time; some being direct modifications of the original set, at time zero, while others the product of a sort of sexual reproduction involving already existing sequences in the system (recombination). The intent here is to emulate real-life as closely as possible; tracking sequences before and after they've been replicated and combined with others during the course of evolution. This all in a space efficient manner. The result is a highly flexible implementation capable of being leveraged in many applications, not to mention the effective scoring and appraisal of sequence alignment algorithms.

Poster 21

Integrating R/Bioconductor with Microsoft Azure

Hugh P. Shanahan[1*], **Anne M Owen**[2], **Andrew P. Harrison** [2,3]

1. Department of Computer Science, Royal Holloway, University of London, Egham, Surrey, TW20 0EX, U.K.,
2. Department of Mathematical Sciences, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, U.K.
3. Department of Biological Sciences, University of Essex, Wivenhoe Park, Colchester, CO4 3SQ, U.K.

*Contact author: hugh.shanahan@rhul.ac.ukhugh.shanahan@rhul.ac.uk

Site/Source URL: https://github.com/hughshanahan/GWydiR

Licence: GPL

Cloud Computing is increasingly being used by Bioinformatics researchers as well as by the scientific community in general. This has been largely encouraged by the rapid increase in the size of Omic data sets Stein (2010). There are advantages in using a cloud for short usages of powerful computers when scaling up programs which have been tested on a small amount of data. Much of the emphasis has been on the use of Infrastructure as a Service platforms, such as Amazon's EC2 service where the user gets direct access to the console of the Virtual Machines(VM's) and *MapReduce* frameworks, in particular *Hadoop* Taylor (2010). An alternative to this is to use a Platform as a Service (PaaS) infrastructure, where access to the VM's is programmatic. An example of this is the Microsoft Azure platform which we have made use of via the VENUS-C EU network.

A PaaS interface can offer certain advantages over the other approaches. In particular, it is more straightforward to design interfaces to software packages such as *R* and it obviates the need to port codes designed for single processors into a *MapReduce* framework. In the case of Azure, another advantage is that Microsoft Research have provided a set of freely available *C#* libraries called the Generic Worker which has allowed us to easily scale up the number of jobs we are processing.

We have developed software that makes use of these libraries to run *R/Bioconductor* scripts to analyse almost all of a specific microarray data set (HG_U133A - an Affymetrix GeneChip for humans) in the public database ArrayExpress. We have previously demonstrated that a small set of publicly deposited experiments that use this type of microarray are susceptible to a bias due to specific sequences that probes of the microarray hybridise with (runs of 4 or Guanines) Shanahan et al. (2011). We have used Azure to extend our analysis to 576 experiments deposited at ArrayExpress before May, 2012. In particular we have shown that correlations between probe sets can be significantly biased, suggesting that probe sets that have such probes will be more correlated with each other than they should be. This will bias a large number of conclusions that have been drawn on the basis of individual experiments and conclusions based on the inference of gene networks using correlations between probe sets over many experiments.

This analysis provides an exemplar to run multiple *R/Bioconductor* jobs that need to be run in an trivially parallel fashion with each other on the Azure platform and to make use of its mass storage facilities. We will discuss an early generalisation we have dubbed **GWydiR** to run any *R/Bioconductor* script on Azure in this fashion, with a goal on providing as simple a method as possible for a user more experienced using a Windows platform to scale up their *R/Bioconductor* jobs.

References

Shanahan, H. P., Memon, F. N., Upton, G. J. G. and Harrison, A. P. (2011, December). Normalized Affymetrix expression data are biased by G-quadruplex formation. *Nucleic Acids Research 40*(8), 3307-3315.

Stein, L. D. (2010, January). The case for cloud computing in genome informatics. *Genome biology 11*(5), 207.

Taylor, R. C. (2010, January). An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *BMC Bioinformatics 11 Suppl 1*, S1.

A system for semi-automatic matching of biobank variables using ontology terms

Chao Pang[1,2], Dennis Hendriksen[1], K. Joeri van der Velde[1], EU-BioSHaRE Consortium, Hans Hillege[2], Morris Swertz[1]

1.Genomics Coordination Center and Epidemiology, UMCG
2.Epidemiology, UMCG
Contact email: ChaoPang229@gmail.com
Project website: http://www.molgenis.org
License: GNU Lesser General Public License version 3 (GNU LGPLv3); Apache License, Version 2.0

Increasingly large data sets are needed to uncover subtle phenotype/disease associations with sufficient statistical power. Therefore many Biobanks with patient-phenotype information need to be pooled. However, the variables needed are not necessarily collected in the same fashion due to different questionnaire designs and terminologies used. Differences in naming, descriptions and value codes make it very time intensive to find, harmonize and integrate data items. For example, in an example projects of 80 variables harmonization took up to man-months!

In order to dramatically speed up this harmonization process, we have developed a new method based on ontologies to automatically shortlist candidate data items from cohorts that human experts can map to their research variables. We implemented four steps: (1) each variable is annotated with ontology terms manually using BioPortal ontology service; (2) information from the ontology terms, such as synonyms and subclasses, are added to expand the semantics of the variables; (3) expanded terms for variable are used to match against data items using a string matching algorithm which calculates the similarity score between pairwise strings; (4) Candidate mappings are sorted by matching score and correct mappings can be selected by the human expert.

We validated our strategy by comparing the rank of manual produced mappings in the automatically produced list. If the rank is ‘1’ then the automated method has produced the same result as the human curator. Obviously, the higher the ranking the better our method can be proven to be. In a test across three Biobanks and 15 variables the average rank was top 3 in 95 % of the cases.

System was implemented using MOLGENIS, OntaCAT, Lucene and Bioportal and will be made available as open source at http://www.github.com/molgenis.

BioSeq.jl : A package for bioinformatics in Julia

Diego Javier Zea[1] and Kevin Squire[2]
[1] Departamento de Ciencia y Tecnología, Universidad Nacional de Quilmes, Bernal, Argentina
diegozea@gmail.com
[2] Department of Human Genetics, David Geffen School of Medicine, UCLA, California, USA
https://github.com/diegozea/BioSeq.jl
MIT License

Bioinformatics is a broad field with diverse needs in informatic tools. With the explosion of bioinformatics data available, performance is becoming a key ingredient of bioinformatic workflows. While high level dynamic languages languages like R, Perl and Python are commonly used for daily bioinformatics tasks and workflows, bioinformaticians frequently move to compiled languages like C or C++ when performance is crucial [1]. In these situations, we believe Julia [2] to be an excellent candidate language for bioinformatics tasks. Julia is a high level dynamic language focused on performance, which runs close to the speed of C. Like Python, Perl, and R, it contains many high level features, including perl-compatible regular expressions, integration with best-of-breed mathematical libraries, strong support for running external programs, and a growing collection of packages. These features are a strong fit for bioinformatics processing, which commonly includes parsing, scripting of external programs, and various types of mathematical analysis. Additionally, for bioinformatics tasks which do not yet have good support within Julia, the language has the ability to seemlessly interact with Python, R, Matlab, and C. In order to use this Julia potential for bioinformatics, we have started to implement basic types and functionality for working with nucleotide and amino acid sequences. These tools are in the package BioSeq.jl and now cover from classical 8 bit ASCII coding scheme nucleotides and amino acid to 2 bit nucleotide sequences and an alternative 8 bit bit-level coding scheme [3]. Julia support for regex and metaprogramming makes it possible to match with PROSITE patterns or create regular expressions directly using IUPAC ambiguities. In Julia, sequences objects are mutable arrays, but many classic strings methods are defined for them. This allows the exploration of Julia's other capabilities, including parallel computation and memory-mapping. BioSeq.jl design was inspired by BioPython's Bio.Seq module [4] and by Bioconductor's Biostrings [5] functionality, with a great focus on performance and flexibility. We are continuing to add new sequence types and functionality, and we hope this flexibility and performance provides the fundamental basis for a huge amount of Bioinformatic tools written in Julia.

[1] Mathieu, Fourment, and Gillings Michael. "A comparison of common programming languages used in bioinformatics." *BMC Bioinformatics* 9. (2008)

[2] Bezanson, Jeff, Stefan Karpinski, Viral B. Shah, and Alan Edelman. "Julia: A Fast Dynamic Language for Technical Computing." *arXiv preprint arXiv:1209.5145* (2012).

[3] Paradis, Emmanuel. "A Bit-Level Coding Scheme for Nucleotides." (2007). Available at http://ape.mpl.ird.fr/misc/BitLevelCodingScheme_20April2007.pdf

[4] Cock, Peter JA, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg et al. "Biopython: freely available Python tools for computational molecular biology and bioinformatics." *Bioinformatics* 25, no. 11 (2009): 1422-1423.

[5] H. Pages, P. Aboyoun, R. Gentleman and S. DebRoy. Biostrings: String objects representing biological sequences, and matching algorithms. R package version 2.26.2.

Poster 24

Open source solutions to the infrastructure challenges of NGS core bioinformatics

Robert Davey[‡], Ricardo Ramirez-Gonzalez[‡], Richard Leggett[†], Xingdong Bian[†], Anil Thanki[†], Nizar Drou[†], Mario Caccamo[†]

† ***The Genome Analysis Centre, Norwich Research Park, Norwich, NR4 7UH, UK***

‡ ***University of East Anglia, Norwich, UK***

Presenting author email: robert.davey@tgac.ac.uk

Website: http://www.tgac.ac.uk/tools-resources/

Source code: https://github.com/TGAC

Licence: GPL v3

As the field of sequencing changes rapidly, bioinformatics infrastructure needs to follow suit. This is not always an easy process to model in the event of new chemistry, software or hardware whilst attempting to maintain cohesion. Using structured accessible software and providing neatly unified interfaces and pipelines will avoid long lead times in managing the impact of changes to workflow. To reach this goal, at TGAC we have developed a number of interoperable open source tools to facilitate the efficient creation, management and visualisation of data and metadata surrounding next generation sequencing.

MISO, our novel LIMS system, records metadata from sample receipt, through library preparation and pooling, on to run tracking, tightly following the standards set out by the European Nucleotide Archive schemas[1] for sequence data submission. This facilitates simpler submission workflow as all required metadata are entered by technicians as part of their day-to-day work. Simpler access to community-defined workflows to MISO is currently under development, which will allow custom extensions and views to be plugged into the framework. Run information can be automatically imported into MISO whilst a sequencer is running via the notification daemon, removing the need for lab technicians to enter this information manually. Many of the current NGS platforms are already supported. Centre-specific analysis tasks residing on an HPC environment can also be initiated from MISO via a dedicated daemon, allowing simple access to primary analysis over the raw sequencer output to produce quality metrics.

These metrics are stored in StatsDB, a generic schema and API toolkit developed at TGAC for recording quality output from any number of QC assessment applications, such as FastQC[2]. Having a consistent consolidated schema that describes QC output means that more powerful queries can be presented to the API thus generating statistics not just at the lane-scale, but covering ever more granular time-based views over sequencers, runs, and adapter barcodes. Front-end visualisations are also available, making report generation an easier, more configurable and a more powerful tool for QC assessment.

Building knowledge from bases to genomes can be aided by the TGAC Browser, which has been developed in close collaboration with domain expert end users to facilitate efficient data dissemination, and is based on the web-based frameworks that, ever increasingly popular, are driving client-side visualisation of genomic datasets. Comprising a stock Ensembl database and optimised rendering processes, TGAC Browser provides intuitive displays intended to facilitate comparative studies and knowledge transfer.

[1] http://www.ebi.ac.uk/ena/about/sra_submissions

[2] http://www.bioinformatics.babraham.ac.uk/projects/fastqc/

Poster 25

Title: Combing the Hairball With BioFabric
Author: William Longabaugh
Organization: Institute for Systems Biology
Email: wlongabaugh@systemsbiology.org
Web site: http://www.BioFabric.org
Source code: https://github.com/wjrl/BioFabric
License: LGPL

BioFabric [1] is a new open-source software application for visualizing networks that uses a novel presentation method: it represents nodes as horizontal line segments, one per row. Edges are then represented as parallel, vertical line segments in a rational, highly organized, unambiguous, and scalable manner. This is in contrast with the traditional approach used for node-link diagrams, where nodes are basically depicted as points, which inevitably leads to edge crossings and lack of scalability.

Shown in Figure 1 is a BioFabric depiction of a network of yeast protein-protein and protein-DNA interactions [2,3,4], where each node appears as a horizontal line segment. Node rows are laid out using the default layout algorithm, which uses a breadth-first search of the network starting from the highest-degree node, visiting neighbors in order of decreasing node degree. As the detailed insert shows, the contiguous parallel edge sets for each node create distinct, characteristic, large-scale "edge wedges" that allows the user to quickly compare the node neighborhoods of large sets of nodes. Additionally, the ability to organize links into visually distinct sets allows for clear presentations of network clusters, link communities [5], and network comparisons.

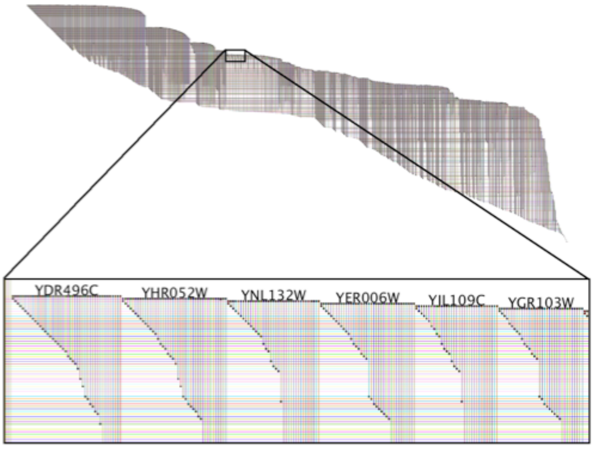


Figure 1: BioFabric depiction of a protein-protein and protein-DNA interaction network in S. cerevisiae containing over 3,000 nodes and 6,800 edges.

BioFabic is written in Java and is licensed under the LGPL.

References:

1. Longabaugh WJR: BMC Bioinformatics 2012, 13:275.
2. http://www.cytoscape.org/download.html: `yeastHighQuality.sif`
3. von Mering C, et al.: Nature 2002, 417:399-403.
4. Lee TI, et al.: Science 2002, 298:799-804.
5. Ahn YY, et al.: Nature 2010, 466(7307):761-764.

BioXSD: An XML Schema for sequence data, features, alignments, and identifiers

Matúš Kalaš[1,2], Edita Karosiene[3], László Kaján[4], Sveinung Gundersen[5], Jon Ison[6], Pål Puntervoll[1], Christophe Blanchet[7], Kristoffer Rapacki[3] and Inge Jonassen[1,2]

Contact: matus.kalas@uib.no, support@bioxsd.org

[1]Computational Biology Unit, Uni Computing and [2]Department of Informatics, University of Bergen, Bergen, Norway. [3]Center for Biological Sequence Analysis, Technical University of Denmark, Kongens Lyngby, Denmark. [4]Bioinformatics and Computational Biology Department, Technische Universität München, Garching, Germany. [5]Institute for Cancer Research, Oslo University Hospital, Norway. [6]European Bioinformatics Institute, EMBL, Hinxton, UK. [7]Institut de Biologie et Chimie des Protéines, CNRS, Université Lyon 1, France.

http://BioXSD.org

http://bioxsd.org/BioXSD-1.1.xsd

Available under the **Creative Commons BY-ND 3.0** licence with additionally allowed: a) usage within XSD import, types, extensions, restrictions, unions and lists, in user's XML namespace; b) redistribution of versions in the original *http://bioxsd.org/...* XML namespace with custom XSD appinfos or SAWSDL attributes. For more details, please see http://bioxsd.org/BioXSD-1.1xsd. Contributions to new canonical versions, in the *http://bioxsd.org/...* XML namespace, are welcome under the supervision of the BioXSD consortium (in order to keep BioXSD a common, canonical data model).

Using a common exchange format is beneficial for smooth compatibility of heterogeneous tools and data resources. As an alternative to tabular formats and RDF, XML formats defined in a machine-understandable XML Schema (XSD) are useful in a number of scenarios, including Web services. A lack of a common XSD-based format for the basic bioinformatics data has motivated the development of BioXSD [1].

BioXSD is a format of the main bioinformatics types of data that are not standardised by specialised XSDs such as SBML, PDBML, MAGE-ML, MIF, GCDML, phyloXML or NeXML [2-8]. BioXSD focusses on sequences, alignments, and annotation with any kinds of features or properties. These main types are accompanied by definitions of formats for data-resource and ontology references including identifiers, provenance metadata, scores, and others. BioXSD development has been initiated by the EMBRACE project [9].

The aim of BioXSD is to become a canonical, "standard" XML format for sequence data and feature records. Tools can produce and consume BioXSD directly, or it can be used as an intermediate format that is rich enough to enable conversions between diverse formats. BioXSD types can be directly used in other XSDs, and they can be further extended or restricted. The XSD can serve as a specification for generating efficient binary representations, such as with EXI [10]. Semantics of the "syntactic" BioXSD types is defined via SAWSDL annotation with concepts from the EDAM ontology, RDFS, and Dublin Core [11-13].

The highlights of BioXSD are: structured metadata of sequence records (as opposed to *ad hoc* FASTA *deflines*), provenance metadata of features and alignments, structured references to data resources and ontology concepts including a *meaning* of the relation, complex relations between features, complex scores with meanings, and more. The BioXSD version 1.1 has optimised syntax of feature records, scores, and references [14]. It allows dense sequence features, applicable to whole-genome annotations exchanged for example in the standard binary EXI format. The recent version 1.1.2 allows optimised representation of alignments, and has improved interoperability with more XML libraries, *e.g.* ones for C++. Optimisations for whole-genome alignments, individual genomics, and sequence profiles and patterns will be added in the next versions.

Involvement of the community is essential both for the uptake and the further evolution of BioXSD. For example supportive tools are currently being developed by the community, such as converters to and from other formats, including tabular ones and RDF.

[1] Kalaš, M. *et al*. (2010) BioXSD: the common data-exchange format for everyday bioinformatics web services. *Bioinformatics*, **26**, i540-i546.
[2] Hucka, M. *et al*. (2003) The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models. *Bioinformatics*, **19**, 524-531.
[3] Westbrook, J. *et al.* (2005) PDBML: the representation of archival macromolecular structure data in XML. *Bioinformatics*, **21**, 988–992.
[4] Spellman, P.T. *et al.* (2002) Design and implementation of microarray gene expression markup language (MAGE-ML). *Genome Biol.*, **3**, research0046.1-0046.9.
[5] Hermjakob, H. *et al.* (2004) The HUPO PSI's Molecular Interaction format—a community standard for the representation of protein interaction data. *Nat. Biotechnol.*, **22**, 177-183.
[6] Kottmann, R. *et al.* (2008) A standard MIGS/MIMS compliant XML schema: toward the development of the Genomic Contextual Data Markup Language (GCDML). *OMICS*, **12**, 115-121.
[7] Han, M.V. and Zmasek, C.M. (2009) phyloXML: XML for evolutionary biology and comparative genomics. *BMC Bioinformatics*, **10**, 356.
[8] Vos, R. *et al*. (2011) BIO::Phylo-phyloinformatic analysis using perl. *BMC Bioinformatics*, **12**, 63.
[9] Pettifer, S. *et al*. (2010) The EMBRACE Web service collection. *Nucleic Acids Res.*, **38**, W683–W688.
[10] Efficient XML Interchange (EXI) Format 1.0. http://www.w3.org/TR/exi/
[11] Ison, J. *et al*. (2013) EDAM: an ontology of bioinformatics operations, types of data and identifiers, topics and formats. *Bioinformatics*, **29**, 1325-1332.
[12] RDFS. http://www.w3.org/TR/rdf-schema
[13] DublinCore. http://dublincore.org
[14] Gundersen, S. *et al*. (2011) Identifying elemental genomic track types and representing them uniformly. *BMC Bioinformatics*, **12**:494.

MOLGENIS compute:

A lightweight toolbox for high-throughput biology pipelines

Martijn Dijkstra*[1], George Byelas*[1,3], Freerk van Dijk[1], Pieter Neerincx[1,3], Laurent Francioli[2], Patrick Deelen[1], Tom Visser[3], Jan Bot[3], BBMRI-NL/eBioGrid,NBIC Genome of the Netherlands Team, Morris Swertz[1]

[1]University Medical Center Groningen, [2]University Medical Center Utrecht, [3]eBioGrid/BigGrid/SARA team, *equal contribution. Correspondence: m.a.swertz@rug.nl

License: LGPLv3

Extensively used for NGS and GWAS

MOLGENIS compute is a toolbox to configure and run parallel bioinformatics pipelines, actively used for DNA re-sequencing, genotype imputations with QTL and GWAS pipelines underway. It was first used in the Genome of the Netherlands (GoNL), a nation-wide BBMRI-NL project of 769 whole genome sequence samples (12x). All those needed alignment and variant calling, a daunting challenge: input was 2250/50 TB *.fq files; analysis required >30 analysis steps; the complete analysis was >50.000 jobs accumulating 200,000 hours. Top candidate systems Taverna and Galaxy are not optimized for these large pipelines, numbers of jobs and distributed analysis. Moreover, bioinformaticians want complete influence on the shell scripts executed to optimally run these on big servers, clusters and grids, which was not readily possible.

Use simple text and command line tools to run big data on clusters and grids

MOLGENIS compute provides a lightweight, easy to change, pipeline system that can be run from command line or web interface. It uses three simple text files: (1) Workflow CSV file where each row is a step (optionally depending on previous steps); (2) Each Step is a 'template' that automatically generates the cluster/grid shell scripts; (3) Parameters CSV that describes parameter values for each step. Steps can configure the parameter to iterate over, e.g., per lane, per sample, per family and/or only once. Also there are automatic mechanisms for file transfer, job submission and progress monitoring, and a web-server for pilot job 'grid' based analyses across clusters.

Download manuals and software

Download manuals from http://www.molgenis.org/wiki/ComputeStart and software and pipelines from http://www.github.com/molgenis/molgenis

Poster 28

ratatosk - a light-weight bioinformatics workflow management system

Per Unneberg
Science for Life Laboratory
DBB, Stockholm University
per.unneberg@scilifelab.se

June 7, 2013

ratatosk [1] is a simple bioinformatics workflow management system written in python. It is built on the **luigi** [2] module, a framework for creating complex pipelines of batch jobs developed and used in production by the digital music provider Spotify.

The design goals of **ratatosk** are simplicity and flexibility; it is completely based on a plain text configuration and scripting environment, and can conceptually be thought of as 'make' in python. It provides wrappers for some commonly used tools for analysis of next-generation sequencing data, although many more could easily be added.

ratatosk jobs are executed via a wrapper script that can be run locally, or submitted to compute clusters using the Distributed Resource Management Application API (drmaa). Jobs are tracked via luigi's builtin scheduler, and task dependencies can be easily visualized via a web interface.

Although **ratatosk** is still pre-mature and under very active development, preliminary tests on production-scale data have indicated the feasibility and scalability of this approach. Future plans include better integration with drmaa and leveraging the power of luigi's builtin support for hadoop.

The source code can be accessed at github [3]. **ratatosk** is licensed under the Apache License, Version 2.0.

References

[1] *ratatosk*. URL: https://ratatosk.readthedocs.org/en/latest/.

[2] *luigi*. URL: https://github.com/spotify/luigi.

[3] *ratatosk source*. URL: https://github.com/percyfal/ratatosk/.

ProtocolNavigator: enhancing the reuse of research data

Imtiaz Khan[1,2], **Adam Fraser**[2], **Mark-Anthony Bray**[2], **Paul Smith**[1], **Anne Carpenter**[2], **Rachel Errington**[1]

[1]School of Medicine, Cardiff University, UK, [2] Imaging Platform, Broad Institute of MIT and Harvard, USA

Email: wpciak@cf.ac.uk

Website:	http://protocolnavigator.org/
Source code:	https://github.com/imtiazKHAN/ProtocolNavigator
License:	GPLv2

Despite the exponential growth of electronic laboratory notebook, laboratory information management systems and databases, consumption or reuse of biological data beyond the proximity of the data originator remains rare. This imbalance is primarily due to the lack of effective communications among researchers. Markup and other structured languages utilized by these approaches enable researchers to share information; but these do not resonate with the day-to-day data curation culture, nor do they ease information interpretation for researchers who wish to reuse or assess the data. This is particularly prominent in an interdisciplinary research context, where the variability of methodologies, curation culture, and terminologies are highly idiosyncratic. Addressing this reality, we have developed ProtocolNavigator – a virtual laboratory environment that allows emulation of real life laboratory actions as the basis for curation. The emulation leads to the automatic representation of a time-integrated, interactive map of an experiment that includes action patterns, manipulations, and data acquisition signified by symbols. Association and sequential analysis of these symbols divulge patterns, which in turn provide a language-independent visual perception of experimental design. Navigation through this design reveals provenance trails for data and metadata; importantly, this interaction delivers contextualization, which facilitates knowledge abstraction and assessment. The fully navigable map can be shared with colleagues for design modification and optimization; also it can be converted and printed into a text format for publication. Our initial survey indicates that this mapping format facilitates intelligible data reuse and consumption, even within interdisciplinary and inter-institutional research scenario. ProtocolNavigator has enabled us to curate and compare practice variation – a quantitative approach for establishing best practice. This we believe a crucial social factor for wider community engagement and uptake.

Poster 30

SeqPig: Scripting for large-scale sequencing based on Hadoop

André Schumacher[1,2,3], Luca Pireddu[4], Aleksi Kallio[5], Matti Niemenmaa[2,3], Eija Korpelainen[5], Gianluigi Zanetti[4], Keijo Heljanko[2,3]
[1] ICSI, Berkeley, USA, [2] Helsinki Institute for Information Technology HIIT, Helsinki, Finland, [3] Aalto University, Espoo, Finland, [4] CRS4, Pula, Italy, [5] CSC-IT Center for Science, Helsinki

Contact: aleksi.kallio@csc.fi
Web: http://seqpig.sourceforge.net, http://sourceforge.net/projects/seqpig/files/
Licensed under the MIT license.

SeqPig is a tool that facilitates the use of the Pig Latin scripting language to manipulate, analyze and query sequencing data. SeqPig provides access to popular data formats and implements a number of high level functions. It operates on top of Hadoop and Pig and augments them to facilitate their use to process sequencing data.

SeqPig extends Pig with a number of features and functionalities conceived for processing sequencing data. Specifically, it provides: 1) data input and output components, 2) specialized functions to extract fields and to transform data and 3) a collection of scripts for frequent tasks (e.g., pileup, QC statistics).

SeqPig provides import and export functions for file formats commonly used for sequencing data: Fastq, Qseq, SAM and BAM. SeqPig supports ad hoc – scripted or even interactive – distributed manipulation and analysis of large sequencing datasets. Unlike traditional methods, the scalable nature of Pig allows the speed of its operations to scale with the computing resources available. SeqPig includes functions to access SAM flags, split reads by base (for computing base-level statistics), reverse-complement reads, calculate read reference positions in a mapping (for pile-ups, extracting SNP positions), and more. The authors are currently working on expanding the library of functions, and SeqPig is an open source project that welcomes and encourages contributions from the community.

SeqPig has been tested on Amazon's Elastic MapReduce service. Users may rent computing time on the cloud to run their SeqPig scripts, and even share their S3 storage buckets with other cloud-enabled software.

This work was supported by the Cloud Software and D2I Programs of the Finnish Strategic Centre for Science, Technology and Innovation TIVIT, the Finnish ELIXIR node, and by the Sardinian (Italy) Regional Grant L7-2010/ COBIK.

Mobyle Web Framework v1.5

Hervé Ménager[1], Bertrand Néron[1], Vivek Gopalan[2], Jennifer Dommer[2], Ramandeep Kaur[2], Alexander Levitsky[2], Jie Li[2], Qiang Sun[2], Wei Liang[2], Nick Weber[2] and Yentram Huyen[2]
[1] Centre d'Informatique en Biologie, Institut Pasteur, Paris, France,
[2] Bioinformatics and Computational Biosciences Branch, Office of Cyber Infrastructure and Computational Biology, National Institute of Allergy and Infectious Diseases, National Institutes of Health, Bethesda, MD 20892, USA
Presenting author email : hmenager@pasteur.fr

Mobyle Website	:	`https://projets.pasteur.fr/wiki/mobyle`
Mobyle Release Downloads	:	`ftp://ftp.pasteur.fr/pub/gensoft/projects/mobyle/`
Mobyle SVN Repository	:	`https://projets.pasteur.fr/svn/mobyle`
BMID/BMPS SVN Repository	:	`https://projets.pasteur.fr/svn/mobyle-niaid`
Open Source License	:	GNU GPLv2

Mobyle is an open-source framework and web portal specifically designed to facilitate the integration of bioinformatics software and databanks. Using Mobyle, researchers and bioinformaticians can leverage command-line applications from a web browser to seamlessly perform bioinformatics analyses on remote computing resources, such as high performance computing clusters. Here we present the improvements of the latest version of Mobyle, version 1.5.

The **BCBB Mobyle Pipeline System (BMPS)** is a new web application distributed with the Mobyle framework to create automated analysis pipelines by linking multiple applications such that the output of one application becomes the input of the next. BMPS provides a graphical interface for creating user-defined scientific pipelines by dragging-and-dropping multiple applications onto a drawing canvas. The typing mechanism of Mobyle data and parameters allows to filter only the compatible services for a selected task output, guiding users in the construction of the workflow. BMPS also provides an interface to manage and run pipeline jobs in remote compute environments. The workflow jobs created with BMPS can be re-executed directly in the Mobyle portal through a simple form.

Other highlights include improvements to the **workspace management**, which facilitates common tasks such as jobs and data renaming. Custom **tutorials** can now be dynamically added to the portal using XML files similar to the ones used to publish command line-based programs, web-based widgets and workflows. These tutorials include HTML-based contents and provide a simple way for Mobyle portal owners to document the services they provide. The Mobyle **execution engine** now support the MODULES (`http://modules.sf.net`) system, a solution used by multiple platforms to facilitate version management and traceability for bioinformatics programs. It has also been extended to enable a per-user limit on the number of simultaneous jobs running.

Title: C-based Bioinformatics Library and Web Application Framework

Authors, with the presenting author's name underlined: Clemens Broger, Detlef Wolf
1) brogerc@bluewin.ch 2) F.Hoffmann-La Roche AG, Basel, Detlef.Wolf@Roche.com
URL for the overall project web site: http://bioinfoc.ch
URL for accessing the code: https://sourceforge.net/p/bioinfoc
The particular Open Source License being used: LGPL

Since 1996 members of the F. Hoffmann-La Roche Bioinformatics community have developed a software library and framework for rapidly building bioinformatics and other applications. These include a web interface for nucleic and protein sequence analysis, as well as a generic data management web application framework with controlled vocabulary support. The main programming language used is C, resulting in superior performance, robustness, excellent scalability and modest hardware requirements. Core parts have been released under LGPL license earlier and are used in academia [2]. We now substantially expand on this by releasing two subsystems (1) Parts of BioinfoLib and (2) Webfile.

BioinfoLib

As the standard C-library is very bare bone, everybody doing major software development in C needs to implement dynamic strings, dynamic arrays, hashes, sparse arrays and the like. The same holds for bioinformatics objects like sequences, assemblies, BLAST outputs etc. The BioinfoLib is a library of shared functions grouped into modules, with encapsulation and instantiation where needed. It makes heavy use of an object-based memory management pattern found also in ACeDB [1] and includes (1) HTML header & form handling, (2) Dynamic strings & arrays, (3) File handling (4) Hyperlink builder (5) Generic SequenceObject/Container with EMBOSS Adapter (6) Various event-based parser modules (Blast, Fasta, ...) (7) Efficient relational database access (postgres, oracle) (8) TCP/IP based clients and servers communicating with low overhead (9) interface to R (“Ribios”).

WebFile

WebFile is a customizable framework for creating WebFile instances. Each WebFile instance is a multi-user, access controlled web application for entering, importing, querying and exporting records consisting of data fields and managing a controlled vocabulary. WebFile’s built-in data types include text, text from controlled vocabulary, number, picture, typed binary file, user ID, and Gene ID. The built-in controlled vocabulary manager supports maintaining synchronized copies of selected controlled vocabulary domains provided by a shared controlled vocabulary server. Examples of applications implemented using the WebFile framework include (1) chip experiment directory (2) formulation management, (3) compound name resolution (4) target lists.

Many individuals contributed to this code, its concepts and its publication, besides the authors of this abstract especially Bjoern Gaiser, Axel Klenk, Martin Neeb, Martin Strahm, Martin Ebeling, Laura Badi, Isabelle Wells, Guido Steiner, Yuan Wang, Said Aktas, Marco Berrera, Roland Schmucki, Jitao David Zhang, Klaus Weymann, Xing Yang, Liping Jin, Sittichoke Saisanit and Michael Braxenthaler.

[1] THE ACEDB GENOME DATABASE, Richard Durbin and Jean Thierry-Mieg; Computational Methods In Genome Research, page 45-55. Edited by S. Suhai, Plenum Press, New York, 1994
[2] RSEQtools: a modular framework to analyze RNA-Seq data using compact, anonymized data summaries. Habegger L, et al. Bioinformatics (Oxford, England). 2011 Jan 15; 27(2):281-3.