BIOINFORMATICS APPLICATIONS NOTE Vol. 26 no. 20 2010, pages 2617–2619
doi:10.1093/bioinformatics/btq475

Sequence analysis

Advance Access publication August 25, 2010

BioRuby: bioinformatics software for the Ruby programming language

Naohisa Goto[1,†], Pjotr Prins[2,†], Mitsuteru Nakao[3], Raoul Bonnal[4], Jan Aerts[5] and Toshiaki Katayama[6,*]

[1]Department of Genome Informatics, Genome Information Research Center, Research Institute for Microbial Diseases, Osaka University, Japan, [2]Department of Nematology, Wageningen University and Groningen Bioinformatics Centre, The Netherlands, [3]Database Center for Life Science, Research Organization of Information and Systems, Tokyo, Japan, [4]Integrative Biology Program, Fondazione Istituto Nazionale di Genetica Molecolare, Milan, Italy, [5]Genome Dynamics and Evolution, Wellcome Trust Sanger Institute, Cambridge, UK and [6]Laboratory of Genome Database, Human Genome Center, Institute of Medical Science, University of Tokyo, Japan

Associate Editor: Dmitrij Frishman

ABSTRACT

Summary: The BioRuby software toolkit contains a comprehensive set of free development tools and libraries for bioinformatics and molecular biology, written in the Ruby programming language. BioRuby has components for sequence analysis, pathway analysis, protein modelling and phylogenetic analysis; it supports many widely used data formats and provides easy access to databases, external programs and public web services, including BLAST, KEGG, GenBank, MEDLINE and GO. BioRuby comes with a tutorial, documentation and an interactive environment, which can be used in the shell, and in the web browser.

Availability: BioRuby is free and open source software, made available under the Ruby license. BioRuby runs on all platforms that support Ruby, including Linux, Mac OS X and Windows. And, with JRuby, BioRuby runs on the Java Virtual Machine. The source code is available from http://www.bioruby.org/.

Contact: katayama@bioruby.org

Received on June 18, 2010; revised on August 11, 2010; accepted on August 12, 2010

1 INTRODUCTION

Research in molecular biology depends critically on access to databases and web services. The BioRuby project was conceived in 2000 to provide easy access to bioinformatics resources through free and open source tools and libraries for Ruby, a dynamic open source programming language with a focus on simplicity and productivity (www.ruby-lang.org).

The BioRuby software components cover a wide range of functionality that is comparable to that offered by other Bio* projects, each targeting a different computer programming language (Stajich and Lapp, 2006), such as BioPerl (Stajich *et al.*, 2002), Biopython (Cock *et al.*, 2009) and BioJava (Holland *et al.*, 2008). BioRuby software components are written in standard Ruby, so they run on all operating systems that support Ruby itself, including Linux, OS X, FreeBSD, Solaris and Windows. With JRuby, BioRuby also can run inside a Java Virtual Machine (JVM), allowing interaction with Java applications and libraries, like Cytoscape for visualization (Shannon *et al.*, 2003).

Both BioRuby and Ruby are used in bioinformatics for scripting (Aerts and Law, 2009), scripting against applications (Katoh *et al.*, 2005), modelling (Lee and Blundell, 2009; Metlagel *et al.*, 2007), analysis (Prince and Marcotte, 2008), visualization and service integration (Philippi, 2004). The web development framework 'Ruby on Rails' is used to create web applications and web services (Biegert *et al.*, 2006; Jacobsen *et al.*, 2010). BioRuby provides connection functionality for major web services, such as the Kyoto Encyclopedia of Genes and Genomes (KEGG; see example in Fig. 1) (Kanehisa *et al.*, 2008), and the TogoWS service, which provides a uniform web service front-end for the major bioinformatics databases (Katayama *et al.*, 2010).

The BioRuby source tree contains over 580 documented classes, 2800 public methods and 20 000 unit test assertions. Source code is kept under Git version control, which allows anyone to clone the source tree and start submitting. We have found that Git substantially lowers the barrier for new people to start contributing to the project. In the last 2 years, the source tree has gained 100 people tracking changes and 32 people cloned the repository.

The BioRuby project is part of the Open Bioinformatics Foundation, which hosts the project website and mailing list, and organizes the annual Bioinformatics Open Source Conference together with the other Bio* projects. A number of BioRuby features support Bio* cross-project standards, such as the BioSQL relational model for interoperable storage of certain data objects or their implementation is coordinated across the Bio* projects, including support for the FASTQ (Cock *et al.*, 2010) and phyloXML (Han and Zmasek, 2009) data exchange formats.

2 FEATURES

BioRuby covers a wide range of functional areas which have been logically divided into separate *modules* (Table 1).

BioRuby allows accessing a comprehensive range of public bioinformatics resources. For example, BioRuby supports the Open Biological Database Access (ODBA) as a generic and standardized way of accessing *biological data sources*. In addition, BioRuby

*To whom correspondence should be addressed.

†The authors wish it to be known that, in their opinion, the first two authors should be regarded as joint First Authors.

can directly process local database files in a variety of different flat file formats, including FASTA, FASTQ (Fig. 2), GenBank and PDB. BioRuby also allows querying and accessing remote online resources through their interfaces for programmatic access, such as those provided by KEGG, the DNA Databank of Japan (DDBJ), the National Center for Biotechnology Information (NCBI) and the European Bioinformatics Institute (EBI).

BioRuby has online *documentation*, tutorials and code examples. It is straightforward to get started with BioRuby and use it to replace, or glue together, legacy shell scripts or to mix Ruby on Rails into an existing web application.

BioRuby comes with an *interactive environment*, both for the command-line shell and in the browser. Ideas can be quickly prototyped in the interactive environment and can be saved as 'scripts' for later use. Such an interactive environment has shown to be especially useful for bioinformatics training and teaching (Fig. 1).

New features, and refinements of existing ones, are constantly being added to the BioRuby code base. Current development activity focuses on adding support for the semantic web, and on designing a plugin system that allows adding entirely new components in a loosely coupled manner, such that experimental new code can be developed without having an impact on BioRuby's core stability and portability.

3 CONCLUSION

The BioRuby software toolkit provides a broad range of functionality for molecular biology and easy access to bioinformatics

Table 1. BioRuby modules

Category	Module list
Object	Sequence, pathway, tree, bibliography reference
Sequence	Manipulation, translation, alignment, location, mapping, feature table, molecular weight, design siRNA, restriction enzyme
Format	GenBank, EMBL, UniProt, KEGG, PDB, MEDLINE, REBASE, FASTA, FASTQ, GFF, MSF, ABIF, SCF, GCG, Lasergene, GEO SOFT, Gene Ontology
Tool	BLAST, FASTA, EMBOSS, HMMER, InterProScan, GenScan, BLAT, Sim4, Spidey, MEME, ClustalW, MUSCLE, MAFFT, T-Coffee, ProbCons
Phylogeny	PHYLIP, PAML, phyloXML, NEXUS, Newick
Web service	NCBI, EBI, DDBJ, KEGG, TogoWS, PSORT, TargetP, PTS1, SOSUI, TMHMM
OBDA	BioSQL, BioFetch, indexed flat files

Refer to www.bioruby.org for an explanation of all acronyms.

```
% gem install bio
% bioruby

bioruby> puts keggapi.bfind("module citrate cycle")
md:M00009 Citrate cycle (TCA cycle, Krebs cycle)
md:M00010 Citrate cycle, first carbon oxidation
md:M00011 Citrate cycle, second carbon oxidation

bioruby> entry = keggapi.bget("md:M00010")
bioruby> md10 = flatparse(entry)

bioruby> entry = keggapi.bget("md:M00011")
bioruby> md11 = flatparse(entry)

bioruby> md10.pathways
  ==> {"ko00020"=>"Citrate cycle (TCA cycle)"}
bioruby> md11.pathways
  ==> {"ko00020"=>"Citrate cycle (TCA cycle)"}

bioruby> md10ko = md10.orthologs_as_array
bioruby> md11ko = md11.orthologs_as_array

bioruby> ko_list = md10ko + md11ko
bioruby> bg_list = ["#000099"] * md10ko.size + ["#660033"] * md11ko.size
bioruby> fg_list = ["#cccccc"] * ko_list.size

bioruby> url = keggapi.color_pathway_by_objects("path:map00020", ko_list, fg_list, bg_list)

bioruby> savefile "map00020.png", open(url).read
```

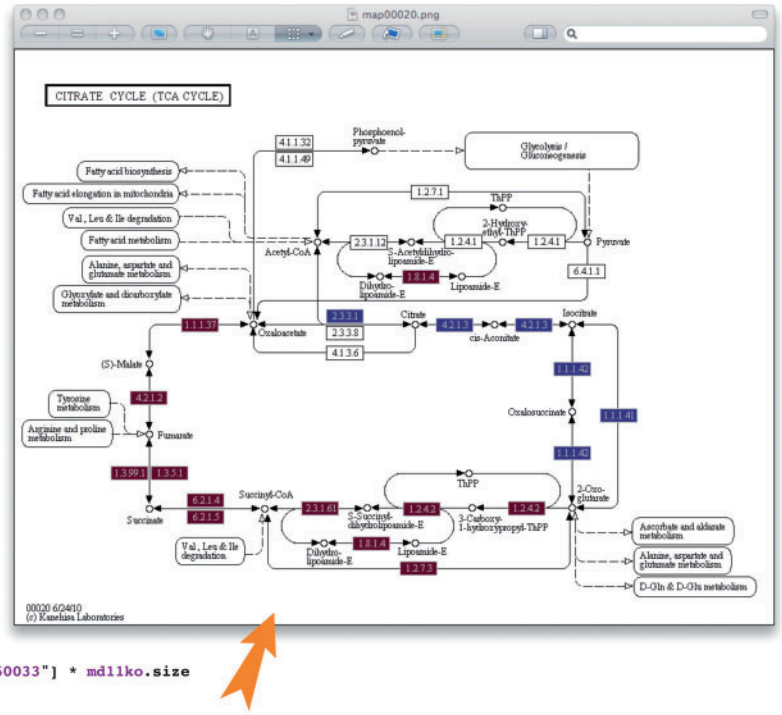

Fig. 1. BioRuby shell example of fetching a KEGG graph using BioRuby's KEGG API (Kanehisa *et al.*, 2008). After installing BioRuby, the 'bioruby' command starts the interactive shell. With the *bfind* command, the KEGG MODULE database is queried for entries involved in the metabolic 'cytrate cycle' or tricarboxylic acid cycle. The purple and blue colours, in input and output, reflect two modules in the carbon oxidation pathway. The user loads and confirms entries by using *flatparse* and *pathways* commands. Next, KEGG ORTHOLOGY database IDs are fetched and the colours are assigned to enzymes in each module. Finally, KEGG generates the coloured image of the 'cytrate cycle' pathway and the image is saved locally.

```
require 'bio'
quality_threshold = 60
Bio::FlatFile.open('sample.fastq').each do |entry|
  hq_seq = entry.mask(quality_threshold)
  puts hq_seq.output_fasta(entry.entry_id)
end
```

Fig. 2. BioRuby example of masking sequences from next generation sequencing data in FASTQ format using a defined quality_threshold, and writing the results in FASTA format.

resources. BioRuby is written in Ruby, a dynamic programming language with a focus on simplicity and productivity, which targets all popular operating systems and the JVM. The BioRuby project is an international and vibrant collaborative software initiative that delivers life science programming resources for those researchers who want to benefit from the productivity features of the Ruby language, as well as from the larger Ruby ecosystem of reusable open source components.

ACKNOWLEDGEMENTS

All individual contributors are credited on the BioRuby website. We thank Hilmar Lapp and Chris Fields for comments and review of the manuscript.

Funding: Information-technology Promotion Agency Japan (IPA); Database Center for Life Science (DBCLS) Japan; Human Genome Center, Institute of Medical Science, University of Tokyo; Open Bioinformatics Foundation (OBF); the National Evolutionary Synthesis Center (NESCent); Google Summer of Code for ongoing support.

Conflict of Interest: none declared.

REFERENCES

Aerts,J. and Law,A. (2009) An introduction to scripting in Ruby for biologists. *BMC Bioinformatics*, **10**, 221.

Biegert,A. *et al*. (2006) The MPI Bioinformatics Toolkit for protein sequence analysis. *Nucleic Acids Res.*, **34**, W335–W339.

Cock,P.J. *et al*. (2009) Biopython: freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, **25**, 1422–1423.

Cock,P.J. *et al*. (2010) The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Res.*, **38**, 1767–1771.

Han,M.V. and Zmasek,C.M. (2009) phyloXML: XML for evolutionary biology and comparative genomics. *BMC Bioinformatics*, **10**, 356.

Holland,R.C. *et al*. (2008) BioJava: an open-source framework for bioinformatics. *Bioinformatics*, **24**, 2096–2097.

Jacobsen,A. *et al*. (2010) miRMaid: a unified programming interface for microRNA data resources. *BMC Bioinformatics*, **11**, 29.

Kanehisa,M. *et al*. (2008) KEGG for linking genomes to life and the environment. *Nucleic Acids Res.*, **36**, D480–D484.

Katayama,T. *et al*. (2010) TogoWS: integrated SOAP and REST APIs for interoperable bioinformatics Web services. *Nucleic Acids Res.*, **38**, W706–W711.

Katoh,K. *et al*. (2005) MAFFT version 5: improvement in accuracy of multiple sequence alignment. *Nucleic Acids Res.*, **33**, 511–518.

Lee,S. and Blundell,T.L. (2009) Ulla: a program for calculating environment-specific amino acid substitution tables. *Bioinformatics*, **25**, 1976–1977.

Metlagel,Z. *et al*. (2007) Ruby-Helix: an implementation of helical image processing based on object-oriented scripting language. *J. Struct. Biol.*, **157**, 95–105.

Philippi,S. (2004) Light-weight integration of molecular biological databases. *Bioinformatics*, **20**, 51–57.

Prince,J.T. and Marcotte,E.M. (2008) mspire: mass spectrometry proteomics in Ruby. *Bioinformatics*, **24**, 2796–2797.

Shannon,P. *et al*. (2003) Cytoscape: a software environment for integrated models of biomolecular interaction networks. *Genome Res.*, **13**, 2498–2504.

Stajich,J.E. and Lapp,H. (2006) Open source tools and toolkits for bioinformatics: significance, and where are we? *Brief Bioinform.*, **7**, 287–296.

Stajich,J.E. *et al*. (2002) The Bioperl toolkit: Perl modules for the life sciences. *Genome Res.*, **12**, 1611–1618.