BioRuby: Object Oriented Open Source Library for Bioinformatics
Toshiaki Katayama, Shuichi Kawashima, Naohisa Goto, Mitsuteru C. Nakao, Yoshinori K. Okuji, Minoru Kanehisa
Bioinformatics center, Kyoto University + various contributors on the Internet

Scope

BioRuby is a class library written in Ruby – an object oriented scripting language – for the daily bioinformatics tasks including biological database access, fetch and parse the entries, execute sequence analysis applications and parse the result in a unified way. Both of the Ruby and BioRuby projects are started in Japan.

History

1995/12/21 Ruby language (0.95) was opened to public [sources]
2000/11/21 BioRuby project started
2001/03/18 mailing list started
2001/06/21 bioruby-0.1
2001/07/13 http://gb.bioruby.org launched
2001/07/19 ISMB/BOSC2001 lightening talks
2001/10/24 bioruby-0.3 w/ CVS repository
2002/01/02 BioHackathon w/ BioFetch server
2002/07/18 http://q-p.bioruby.org launched
2002/08/02 BOSC Japan 2002 w/ BioBlog server
2002/11/22 http://gmod.bioruby.org launched
2002/12/xx bioruby-0.4 released

Open Bio* related projects

BioPerl by Ewan, Jason, Heikki et al.
BioPython by Andrew, Brad et al.
BioJava by Thomas, Matthew et al.
Omnigene by Brian et al.
BioMOBY by Mark et al.
GMOD by Lincoln et al.
BioSQL by Hilmar, Chris et al.
BioPipe by Eli et al.
BOS by Martin et al.
EMBOSS eTc...

BioHackathon

Invite only annual hacking event with commercial support where the Open Bio* developers meet face-to-face to develop common infrastructures for the Open Bio* and the bioinformatics community.

2002/01/26-28 the 1st BioHackathon held in Tucson, Arizona, U.S.A.
AZ portion - mainly sponsored by O'Reilly
2002/02/24-01 2nd part of the event held in Cape Town, South Africa
CT portion - mainly sponsored by Electric Genetics

Result:
Implement OBDA specs defined during the Hackathon.
- topics on ontology, graph, biblio are also discussed.

2003/02/xx-xx BioHackathon 2003 will be held in Singapore

OBDA - Open Bio* sequence Database Access

BioPerl
BioPython
BioJava
BioRuby
BioFlat
BioRegistry
Query
Directory service
Build
BioFetch
BioSQL
BioCorba
XEMBL
EBI server
EBI server
CGI
SOAP
BSML
AGAVE

User config file
http://open.bio-society.org/specification.xml

System config file
http://open.bio-society.org/specification.xml

Open Bio* sample
http://open.bio-society.org/specification.xml

Build
Flat file
index
bdl
indexer
CGI
db = embl
id = bnm
style = raw
style = fasta
RDB
MySQL
Pg
ORACLE
EBI server
BioRuby getting started

Stable release:
```bash
% ftp get ftp://ftp.bioruby.org/bioruby/bioruby-0.4.0.tar.gz
% tar xzf bioruby-0.4.0.tar.gz
% cd bioruby-0.4.0
% ruby install.rb config
% ruby install.rb setup
% su
% # ruby install.rb install
```

Cutting edge version:
```bash
% cvs -d :pserver:anonymous@cvs.bioruby.org/export/cvs/ login
CVS password: [ just hit return ]
% cvs -d :pserver:anonymous@cvs.bioruby.org/export/cvs/ checkout bioruby
% cd bioruby
following installation procedure is same as above
```

Bio::* classes

Bio::Sequence, Bio::Location, Bio::Feature classes
- treat nucleic acid and amino acid sequences, calculate compositions, reverse complement, translation, window search, splicing, sequence annotation from feature table, etc.

Bio::DB classes (GenBank/EMBL/KEGG etc.)
- parsers for various databases (currently supports >20 database formats)

Bio::Blast, Bio::Fasta, Bio::HMMER, Bio::EMBOSS classes
- application wrappers and report parsers (supports local and remote execution)

Bio::Reference, Bio::PubMed classes
- store and reformat the reference information, query the NCBI/PubMed database

Bio::Registry, Bio::SQL, Bio::Fetch, Bio::Flat classes
- OBDA sequence database accessing interfaces

Bio::Pathway, Bio::Relation classes
- graph computation and binary relations

Sample codes

Sequence manipulation
```ruby
#!/usr/bin/env ruby
require 'bio'
gene = Bio::Sequence::NA.new("ggtattctcg")
puts gene.complement # -> "ccagatacc"
puts gene.composition # -> {"A"=>1,"C"=>1,"G"=>4,"T"=>3}
puts gene.gc # -> 55.6
prot = gene.translate
puts prot # -> "GIW"
puts prot.molecular_weight # -> 374.45
```

Running Blast
```ruby
#!/usr/bin/env ruby
require 'bio'
fastafile = ARGV.shift
bl = Bio::Blast.local(\"blastp\", fastafile)
ff = Bio::FastaFormat.open(fastafile)
ff.each do |entry|
  puts entry.entry_id
  report = factory.query(entry)
  report.each do |h|
    puts h.query_id, h.target_id, h.evalue, \"(\"v\")\"
  end
end
```

Flat file database access
```ruby
#!/usr/bin/env ruby
require 'bio'
ff = Bio::GenBank.open("gbbic1.seq")
ff.each do |gb|
  gb.each_gene do |f|
    title = gb.entry_id + pos
    puts gb.seq.splice(pos).to_fasta(title)
  end
end
```

Entry fetch by OBDA
```ruby
#!/usr/bin/env ruby
require 'bio'
registry = Bio::Registry.new
db = registry.db(\"swissprot\")
puts db.fetch(\"TETW_BUTFI\")
```

`bioflat` command
```bash
% bioflat --makeindex gbphg4 \n  --format genbank gbphg4
% bioflat gbphg4 AB048798
```

`biofetch` command
```bash
% biofetch ecoB0001
```

Future work

Documentation
- Alignment classes
- NCBI / E-utils
- SOAP (DAS, XEMBL, BQS, NIG), CORBA
- Pfam, InterPro, GO, GFF, BSML, AGAVE
- Pathway computation / SSDB
- 3D structure / PDBj

What's coming next ...
- Demo in Tokutei symposium
- BioHackathon2003
- BIP2003
- BOSC2003

GiW 2002 : KATAYAMA Toshiaki <k@bioruby.org>