www.biopieces.org

Biopieces: a bioinformatics toolset and framework

Martin A. Hansen*, Harald Oey, Selene Fernandez-Valverde, Chol-Hee Jung, and John S. Mattick

* Presenting Author: contact m.hansen@imb.uq.edu.au

The University of Queensland, Institute for Molecular Bioscience, 4072 St. Lucia, Brisbane, Queensland, Australia

Introduction

The Biopieces are a collection of bioinformatics tools that can be pieced together in a very easy and flexible manner to perform both simple and complex tasks.

The Biopieces work on a data stream that can be passed through several different Biopieces, each performing one specific task: modifying or adding records to the data stream and creating plots, or uploading data to databases and web services.

The Biopieces are executed in a command line environment where the data stream is initialized by specific Biopieces which read data from files, databases, or web services, and output records to the data stream that is passed to downstream Biopieces until the data stream is terminated at the end of the analysis as illustrated below:

```
read_data | calculate_something | write_results
```

The advantage of the Biopieces is that a user can easily solve simple and complex tasks without having any programming experience. Moreover, since the data format used to pass data between Biopieces is text based, different developers can quickly create new Biopieces in their favorite programming language - and all the Biopieces will maintain compatibility.

Data Stream

The data stream consists of Biopiece records that are passed from Biopiece to Biopiece using the standard streams of a Unix or Unix-like operating system. All Biopieces are able to parse Biopiece records from standard input and emit records to standard output, and thus the data stream may be passed through several Biopieces.

The data format used by the Biopieces is text based and consists of records delimited by lines with three dashes. Each record consists of lines with a key/value pair separated by a colon and a whitespace. When the records are parsed the key/value pairs are used to store the data in a hash structure. There are no constraints on the format of the keys and values.

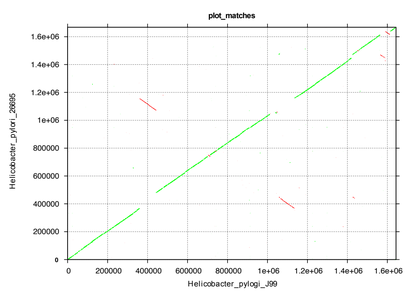
It is possible to read in data from several diffent sources at any point during an analysis resulting in a data stream with mixed Biopiece record types. The downstream Biopieces will parse all records and only act on them if the data content is of the right type for the task of that Biopiece. The Biopiece may emit processed or unprocessed records depending on the task - the behaviour follows the principle of least surprise.

Example

In order to compare the genomic layout between two strains of Helicobacter pylori a dotplot was created to yield information on location and size of genomic rearrangements.

```
read_fasta -i H.pyl_26695.fna |  #  Read the first genome sequences from a FASTA file.
read_fasta -i H.pyl_j99.fna |    #  Read the second genome sequence from a FASTA file.
match_seq |                      #  Find shared words between the sequences using MUMmer.
plot_matches -t post -x          #  Create a plot of the matches.
```

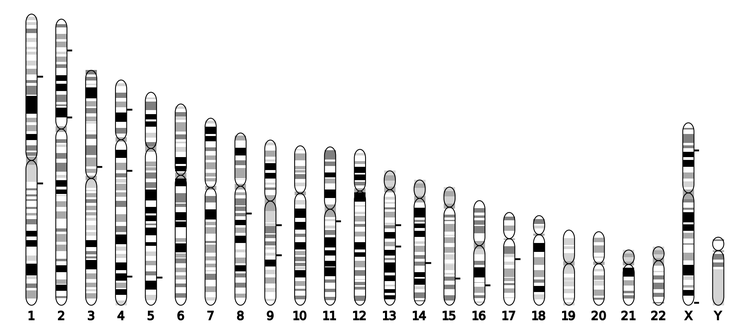
The resulting dotplot shows direct matches between the genome sequences in green and reversed matches in red.



Example

Here we demonstrate how a number of gene sequences can be used to search the human genome using BLAST, how to output the BLAST results to file, and at the same time create a karyogram of the BLAST hits.

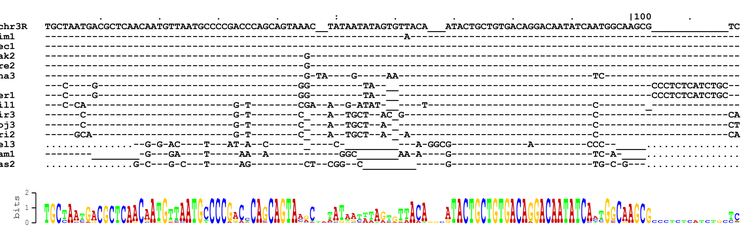
```
read_fasta -i genes.fna |     #  Read gene sequences from FASTA file.
blast_seq -g hg18 |           #  BLAST sequences against the human genome
write_blast -o genes.blast |  #  Write the BLAST results to file.
plot_karyogram -g hg18 -x     #  Create a human karyogram with the BLAST hits.
```



Example

Here we demonstrate a sequence conservation analysis for dme-mir-100 - a microRNA found in Drosophila.

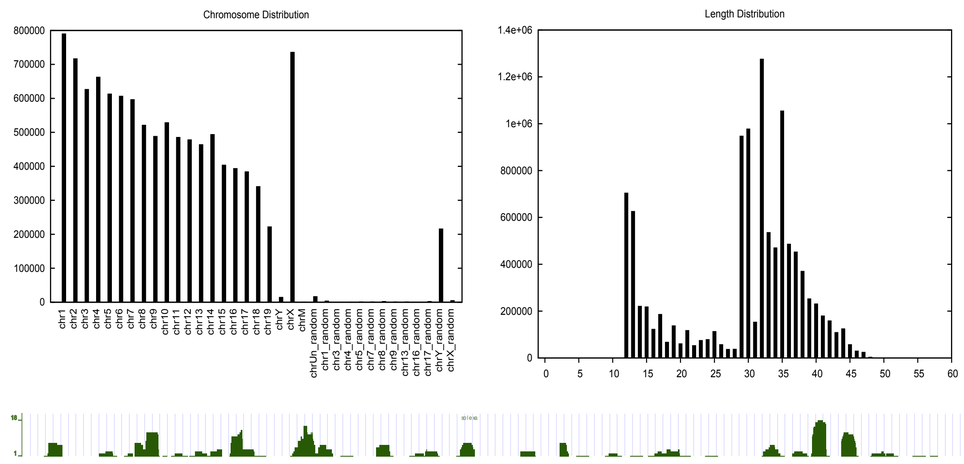
```
read_embl -i miRNA.dat.gz |       #  Read miRBase from the file in EMBL format.
grab -p dme-mir-100 |             #  Grab records maching miRNA of interest.
grab -i -p miRNA -k FEAT_TYPE |   #  Grab records that are not mature miRNAs.
rename_keys -k AC,SEQ_NAME |      #  Use accession number as sequence name.
vmatch_seq -g dm3 |               #  Map sequence to the Drosophila genome with Vmatch.
get_genome_align -g dm3 |         #  Get the multiple genome alignment for this loci.
reverse_seq |                     #  Reverse sequences (hit on complementary strand).
complement_seq |                  #  Complement sequences.
plot_seqlogo -o logo.svg |        #  Plot a sequence logo and save to file.
invert_align |                    #  Invert alignment to emphasize mismatches.
write_align -x                    #  Write alignment as blocks.
```



Example

The following example demonstrates how a Solexa deep sequencing experiment can be cleaned, mapped to the genome of interest, and displayed.

```
read_solexa -i data.solexa |          #  Read Solexa data from a file.
remove_adaptor -a TCGTATGCC -m 2 |    #  Remove adaptor sequence allowing for 2 mismatches.
grab -e 'ADAPTOR_POS > -1' |          #  Get all entries where an adaptor sequence was found.
count_vals -k SEQ |                   #  Determine the occurrences of all sequences.
uniq_vals -k SEQ |                    #  Get all entries with a unique sequence.
merge_vals -k SEQ_NAME,SEQ_COUNT |    #  Append the sequence count to the sequence name.
vmatch_seq -g mm9 |                   #  Map the sequences to the Mouse genome using Vmatch.
write_bed -o solexa_data.bed |        #  Write results to file.
plot_chrdist -t post -o chrdist.ps |  #  Plot chromosome distribution and save to file.
plot_lendist -t post -o lendist.ps |  #  Plot length distribution and save to file.
calc_fixedstep |                      #  Calculate a fixedStep Wiggle track.
upload_to_ucsc -d mm9 -t solexa -x    #  Upload the Wiggle data to the UCSC Genome Browser.
```



Developing Biopieces

Biopieces are easily developed because there is no requirement for any specific programming language as long as it functions on a Unix or Unix-like operating system, is able to read and write to the standard streams, and supports a hash data structure.

All Biopieces must be able to parse Biopiece records from the standard in stream and emit records to the standard out stream. These simple requirements enable Biopieces written by different developers in different programming languages to maintain compatibility.

Each Biopiece should perform one well defined task only - such as converting DNA sequence to RNA, initializing a BLAST search, or plotting a type of diagram. It is advisable to use the same argument names and record keys used in existing the Biopieces in order to maintain the high usability.

The documentation is based on wiki markup language and is independent of programming language. For each Biopiece a wiki page is available from the command line formatted as text and online at www.biopieces.org formatted as HTML. For the creation of a new Biopiece simply copy an existing wiki page and modify - and a specific Biopiece will render the content.

Finally, it is advisable to use the existing code base written in Perl, Python, Ruby, and C.

Download

There are currently 100 Biopieces available. The Biopieces code is open source under the GNU General Public License and can be downloaded freely from www.biopieces.org.

The University Of Queensland AUSTRALIA | IMB Institute for Molecular Bioscience

Australian Government
Australian Research Council
ARC Centre of Excellence in Bioinformatics

Danish Agency for Science Technology and Innovation
Ministry of Science Technology and Innovation