BIOINFORMATICS APPLICATIONS NOTE Vol. 28 no. 2 2012, pages 282–283
doi:10.1093/bioinformatics/btr646

Genome analysis Advance Access publication November 22, 2011

GenomicTools: a computational platform for developing high-throughput analytics in genomics

Aristotelis Tsirigos*, Niina Haiminen, Erhan Bilal and Filippo Utro
Computational Biology Center, IBM T.J. Watson Research Center, Yorktown Heights, NY 10598, USA

Associate Editor: Martin Bishop

ABSTRACT

Motivation: Recent advances in sequencing technology have resulted in the dramatic increase of sequencing data, which, in turn, requires efficient management of computational resources, such as computing time, memory requirements as well as prototyping of computational pipelines.

Results: We present *GenomicTools*, a flexible computational platform, comprising both a command-line set of tools and a C++ API, for the analysis and manipulation of high-throughput sequencing data such as DNA-seq, RNA-seq, ChIP-seq and MethylC-seq. *GenomicTools* implements a variety of mathematical operations between sets of genomic regions thereby enabling the prototyping of computational pipelines that can address a wide spectrum of tasks ranging from pre-processing and quality control to meta-analyses. Additionally, the *GenomicTools* platform is designed to analyze large datasets of any size by minimizing memory requirements. In practical applications, where comparable, *GenomicTools* outperforms existing tools in terms of both time and memory usage.

Availability: The *GenomicTools* platform (version 2.0.0) was implemented in C++. The source code, documentation, user manual, example datasets and scripts are available online at http://code.google.com/p/ibm-cbc-genomic-tools.

Contact: atsirigo@us.ibm.com

Supplementary information: Supplementary data are available at *Bioinformatics* online.

Received on October 7, 2011; revised on November 7, 2011; accepted November 16, 2011

1 INTRODUCTION

Advances in sequencing technology have led to an impressive increase in the production of experimental data in the form of RNA-seq, ChIP-seq and other types of sequencing data. Genomics studies now typically involve the analysis of dozens of sequencing datasets, or equivalently, hundreds of millions of sequenced reads. This amount of data requires an efficient management of computational resources such as time, memory and development time.

In an effort to address these issues, the BEDTools suite was developed (Quinlan and Hall, 2010) where UNIX command-line pipelines were used to effectively reduce the genomics analyses into a form of stream computing. BAMTools (Barnett *et al*., 2011) provides an API through which read alignments in BAM/SAM format (Li *et al*., 2009) can be manipulated sequentially. This computing model clearly makes a more efficient use of memory resources compared to loading entire datasets in memory as necessary in the Bioconductor environment (Gentleman *et al*., 2004).

The *GenomicTools* platform, while similar in motivation to BEDTools, it addresses several open issues. We summarize the novelty of *GenomicTools* below:

- Novel operations; for example, printing alignments to reference genome, scanning computations, shuffling within a reference set of regions.
- Extended support of input formats: all major fields for all supported formats are properly updated in all operations; for example, this leads to full use of all 'exons' in BED entries and it updates the thickStart and thickEnd variables.
- Full stream computing design: all input files can be processed as streams in order to minimize memory requirements and allow the simultaneous processing of several samples.
- C++ API: all command-line operations are implemented as C++ class methods in a convenient API, which can be used by developers to write new applications.
- Auxiliary tools: a set of auxiliary command-line tools (permutation_test, vectors and matrix) facilitates the construction of command-line pipelines.
- Performance: *GenomicTools* improves performance compared to similar tools, both in terms of time and memory requirements.

In the following sections, we describe the specific features of the *GenomicTools* platform and report on its performance.

2 FEATURES AND METHODS

Genome-wide data, such as transcripts/genes, exons/introns, promoter sites, alignments, binding sites, repeat elements, microarray probes, sequencing data (RNA-seq, ChIP-seq, DNA-seq, etc.), or chromosomal conformations (3C-seq, 4C-seq, etc.) can be represented as *genomic regions*, i.e. ordered sets of *genomic intervals*, which in turn are defined as tuples: <chromosome, strand, start position, end position>. Several different file formats are currently used to represent genomic intervals. The *GenomicTools* platform supports the standard GFF (http://www.sanger.ac.uk/resources/software/gff/spec.html), BED (Kent *et al*., 2002) and SAM (Li *et al*., 2009) formats (see user's manual, sections 1.3 and 1.4).

Examples of applications that can be built on top of the *GenomicTools* platform include standard analyses for high-throughput sequencing data, such as computing gene expression from RNA-seq data, constructing peak-finding algorithms for ChIP-seq data, computing read profiles across gene bodies, TSS regions, etc.

*To whom correspondence should be addressed.

© The Author 2011. Published by Oxford University Press. All rights reserved. For Permissions, please email: journals.permissions@oup.com

Downloaded from https://academic.oup.com/bioinformatics/article/28/2/282/199128 by guest on 25 April 2025

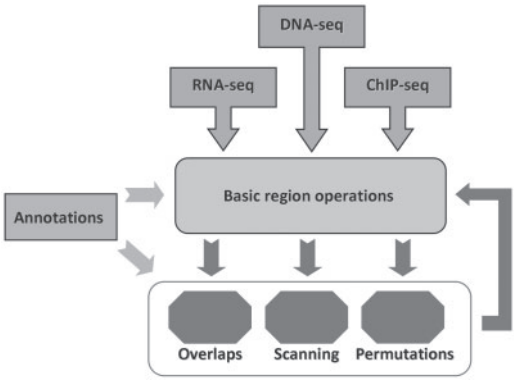

Fig. 1. Flow-chart of various functionalities: basic region operations are implemented in *genomic_regions* while complex operations are implemented in *genomic_overlaps*, *genomic_scans* and *permutation_test* (see text for details).

The *GenomicTools* operations are bundled in several command-line tools: *genomic_regions*, for basic genomic regions operations, *genomic_overlaps* for comparing sets of regions and computing offsets, *genomic_scans* for window-based operations and *permutation_test* for enrichment analyses (see Fig. 1 for a flow-chart).

2.1 The genomic_intervals library

The *genomic_intervals* library implements several C++ classes in order to retrieve, store and manipulate genome-wide data. The data is organized in three levels: interval, region and region set. Advanced operations such as sliding window scanning and overlap computations are implemented as separate classes that operate on top of the data classes. Access to full documentation is provided in the code repository and more details about the C++ classes can be found in the user's manual (chapter 2, page 6). The tools presented below are implemented using this library.

2.2 The genomic_regions tool

The *genomic_regions* tool is designed to perform basic operations on genomic region files. These are: (i) line-based operations, such as aligning, shifting, shuffling, sorting and modifying genomic regions, and (ii) file-based operations such as linking or inverting genomic regions. The reader is referred to Supplementary Table S1 for the complete list of operations, and user's manual chapter 4 for a detailed explanation of their function. The C++ API functions corresponding to the operations are listed in table 4 (page 8) of the user's manual and their documentation can be found in the code repository.

2.3 The genomic_overlaps tool

The *genomic_overlaps* tool allows the user to compute various measures of overlaps between sets of regions (see Supplementary Table S2 for the complete list of operations). This is achieved by providing a set of operations and a variety of options for: (i) finding regions that match or partially overlap, (ii) counting the number of matches, (iii) calculating densities of matched regions, and (iv) computing relative distances (i.e. offsets) between overlapping regions. These overlap computations rely on sorted input files for improved computational efficiency, although for users' convenience we have also implemented the genome binning approach proposed in (Kent *et al*., 2002). For details, see section 2 of Supplementary Material. Applications include computation of RPKMs (Mortazavi *et al*., 2008), construction of average read profiles or heatmaps across TSSs, enrichment analyses of virtually any genomic dataset, such as genes of specific functional categories, repeat types, SNPs, cancer-associated regions, etc. (see user's manual chapter 5 case study, page 57).

2.4 The genomic_scans tool

The *genomic_scans* tool performs window-based computations such as peak discovery (see Supplementary Table S3 for the complete list of operations). The command-line version offers several parameters for controlling the window size, statistical tests, etc. Users with basic C/C++ skills can easily modify the source code to perform the statistical test of their choice using the *GenomicRegionSetScanner* class described in the user's manual (page 9). Also a detailed command-line usage example is available as part of the case study in chapter 5 of the user's manual.

2.5 The permutation_test tool

This tool executes row permutations to determine *P*-values and q-values for all the categories contained in the input file given the statistic chosen by the user. In section 5.5 of the user's manual, this tool is used to determine Gene Ontology categories enriched in ChIP-seq peaks computed using the *genomic_scans* tool. The input format and options are described in the user's manual (page 53).

3 PERFORMANCE

Where directly comparable, we evaluated the time and memory performance of *GenomicTools* against (i) the IRanges Bioconductor package and (ii) the BEDTools suite. In overlap computations, *GenomicTools* outperforms IRanges and BEDTools with an average 9.1 and 3.6 speedup, respectively in sorted data with a minimal memory footprint (see Supplementary Material for details).

Conflict of Interest: none declared.

REFERENCES

Barnett,D.W. *et al*. (2011) BamTools: a C++ API and toolkit for analyzing and managing BAM files. *Bioinformatics*, **27**, 1691–1692.

Gentleman,R.C. *et al*. (2004) Bioconductor: open software development for computational biology and bioinformatics. *Genome Biol.*, **5**, R80.

Kent,W.J. *et al*. (2002) The human genome browser at UCSC. *Genome Res.*, **12**, 996–1006.

Li,H. *et al*. (2009) The Sequence Alignment/Map format and SAMtools. *Bioinformatics*, **25**, 2078–2079.

Mortazavi,A. *et al*. (2008) Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat. Methods*, **5**, 621–628.

Quinlan,A.R. and Hall,I.M. (2010) BEDTools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, **26**, 841–842.

Downloaded from https://academic.oup.com/bioinformatics/article/28/2/282/199128 by guest on 25 April 2025