BIOINFORMATICS APPLICATIONS NOTE Vol. 30 no. 18 2014, pages 2652–2653
doi:10.1093/bioinformatics/btu343

Genome analysis Advance Access publication May 19, 2014

SparkSeq: fast, scalable and cloud-ready tool for the interactive genomic data analysis with nucleotide precision

Marek S. Wiewiórka[1,*], Antonio Messina[2], Alicja Pacholewska[3,4], Sergio Maffioletti[2], Piotr Gawrysiak[1] and Michał J. Okoniewski[5]

[1]Institute of Computer Science, Warsaw University of Technology, Warsaw, Poland, ICS 00-665 Warsaw (MW, PG), [2]Grid Computing Competence Center-GC3, University of Zurich, 8057 Zürich (SM, AM), [3]Swiss Institute of Equine Medicine, Vetsuisse Faculty, University of Bern and ALP-Haras, 3001 Bern (AP), [4]Institute of Genetics, Vetsuisse Faculty, University of Bern, Bern, 3001 Bern (AP) and [5]Functional Genomics Center Zurich, CH-8057 Zurich, Switzerland

Associate Editor: Inanc Birol

ABSTRACT

Summary: Many time-consuming analyses of next-generation sequencing data can be addressed with modern cloud computing. The Apache Hadoop-based solutions have become popular in genomics ***because of*** their scalability in a cloud infrastructure. So far, most of these tools have been used for batch data processing rather than interactive data querying.

The SparkSeq software has been created to take advantage of a new MapReduce framework, Apache Spark, for next-generation sequencing data. SparkSeq is a general-purpose, flexible and easily extendable library for genomic cloud computing. It can be used to build genomic analysis pipelines in Scala and run them in an interactive way. SparkSeq opens up the possibility of customized *ad hoc* secondary analyses and iterative machine learning algorithms. This article demonstrates its scalability and overall fast performance by running the analyses of sequencing datasets. Tests of SparkSeq also prove that the use of cache and HDFS block size can be tuned for the optimal performance on multiple worker nodes.

Availability and implementation: Available under open source Apache 2.0 license: https://bitbucket.org/mwiewiorka/sparkseq/.

Contact: marek.wiewiorka@gmail.com

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

Received on December 17, 2013; revised on April 18, 2014; accepted on May 12, 2014

In recent years, next-generation sequencing (NGS) has become the essential technology that is producing precise insight into the genomes and transcriptomes of living organisms. A standard sequencer run produces hundreds gigabase pairs of short reads. The experimental results (fastq or alignment files) can be regarded as big data, in particular with the high number of biological samples. This can be less overwhelming when lab techniques of preselection are applied or when the data are aggregated. Computationally, the reduction of the datasets is done by counting the reads in genes or finding genome variants. The resulting count tables or variant lists are then much smaller than the original BAM files and in most cases can be processed in memory. However, aggregation and summarization always result in the loss of information on such phenomena as novel expression regions, alternative splicing or low coverage areas. Sequencing can pinpoint various properties of genomes and transcriptomes (Frazee *et al.*, 2014; Leśniewska and Okoniewski, 2011) with a nucleotide precision, but often the size of datasets is prohibitive to study such details. The nucleotide-level analysis can be a driving force for the many new and additional applications of sequencing such as linc-RNA discovery, verification of gene and UTR boundaries in the species with imprecise annotation or studies of alternative splicing.

The developments in computer science, especially in the area of distributed and cloud computing, are trying to keep pace with the rapidly growing amount of experimental data. There are several applications already available (Langmead *et al.*, 2010; Schumacher *et al.*, 2014; Taylor, 2010). Currently there are many initiatives within the IT and bioinformatics community that can be used for creating a useful and scalable system for sequencing data analysis. One of the most frequently used parallel and distributed programming models is MapReduce, which has its open-source implementation—Apache Hadoop (Borthakur, 2007). As initially it was not possible to analyze NGS data stored in Hadoop Distributed File System (HDFS) without conversion to file formats supported by Apache Hadoop, a library Hadoop-BAM was developed for direct access and manipulation of formats commonly used in bioinformatics: Binary Alignment/Map format (BAM), FASTQ and Variant Call format (VCF) (Niemenmaa *et al.*, 2012).

Many analytical tools that process data within the Hadoop ecosystem have been developed recently in the open-source community: new high-level languages, database engines and application frameworks. They use the same Apache Hadoop execution model, primarily designed and optimized for one-pass batch processing of on-disk data but not for interactive and in-memory *ad hoc* data exploration. A good example of the successful combination of Hadoop-BAM and the Apache Pig language is SeqPig, an extension for processing of sequencing data (Schumacher *et al.*, 2014) similar to standard samtools (Li *et al.*, 2009). However, using high-level dataflow languages such as Apache Pig has one drawback: in many cases, data processing with them is far from optimal, as it is harder to do fine-grained code optimizations.

*To whom correspondence should be addressed.

A completely new MapReduce paradigm implementation, Apache Spark (Zaharia *et al.*, 2012), addresses both of those problems. It introduces a new Application Programming Interface (API), suitable for both high-level data processing workflow definition as well as low-level code tuning, using the concise Scala language. Apache Spark includes also a new storage primitive: resilient distributed datasets. It lets the users cache precomputed data in memory and/or disk across queries. Apache Spark has also much lower launching overheads, which is important for running *ad hoc* queries. Another advantage of Apache Spark is its REPL (read-eval-print-loop) environment: Spark-shell.

To study the utility of Apache Spark in the genomic context, we created SparkSeq. SparkSeq is a general-purpose tool for RNA and DNA sequencing analyses, tuned for processing in the cloud big alignment data with nucleotide precision. It combines Picard Java Development Kit for Sequence Alignment/Map format (SAM) (Picard SAM JDK) via Hadoop-BAM library and Apache Spark to introduce versatile sequencing analyses in MapReduce environment. It currently implements operations on many alignment files at a time, falling into three categories: filtering of reads, summarizing genomic features and statistical analyses. It can be run directly with Scala, or in R using the RSparkSeq package. The scalability study done with the prototype has proven that by using big enough computational infrastructure, it is possible to handle the constantly growing number of alignment files with base-pair resolution in a manageable time. Much shorter processing time due to scalability can lead to more interactive analysis with a deeper biological insight for the genomics researchers. The computing infrastructure for the tests included cluster nodes equipped with 8-core CPU and 16 GB RAM running Linux Ubuntu 12.04 virtual machines with Apache Hadoop 1.2.1, Apache Spark 0.8.0 and Scala 2.9.3 installed. Scalability and performance has been evaluated by increasing the number of worker nodes from 1 to 10 (8 to 80 cores in total) and comparing this with samtools and SeqPig on the coverage/pileup function of a multi-sample dataset. In separate tests, such parameters as HDFS block size and the use of caching have been examined to find the optimal processing time. Implemented analysis pipelines include the calculation of read coverage for all the nucleotides in the reference genome and the counts of reads within all exons, that can be processed then by the standard statistical methods for RNA sequencing (Anders *et al.*, 2013). Testing of SparkSeq has been done mainly using two datasets: one 30 GB BAM file from multiple-amplicon sequencing experiment and 32 BAM files (~40 GB) from a whole-transcript RNA-sequencing experiment.

The results presented in Figure 1 show that SparkSeq outperforms SeqPig in terms of speed (8.4–9.15 times) and that finding the coverage for all the nucleotides scales up well. The cache experiments described in detail in the Supplementary File show that using a fast data serializer (like KryoSerializer) instead of the Java built-in one together with LZF or Snappy compression can speed up multi-pass data querying up to 110 times and reduce memory consumption by a factor of 13. HDFS block size adjustment can also result in a performance boost.

The tests done on SparkSeq clearly prove the utility of Apache Spark/Hadoop-BAM-based solution for high-performance analysis of sequencing alignment files. The efficiency is comparable

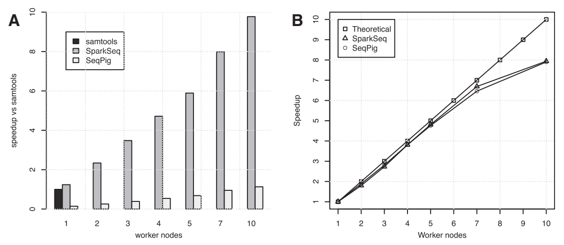

Fig. 1. Speedup (**A**) and scalability (**B**) of SparkSeq and SeqPig on 8 BAM files RNA-sequencing dataset (total 9 GB) and a different number of worker nodes. The benchmark consists of the filtering out duplicates and reads with low quality or gaps in CIGAR string, then calculating the coverage/pileup for all the samples. samtools used as a baseline

with standards (samtools) on a standalone machine and it offers the added value of horizontal scalability. Similar implementations can be done for other types of analyses and other type of genomic alignment files. The benefit for biology researchers will be the speeding up of the long analyses and getting even the nucleotide precision of multi-sample results in non-prohibitive time. This will enable unsupervised searches for novel genomics phenomena in many samples in parallel, as well as optimizing standard analyses by running them many times, tuning their parameters in an interactive way.

ACKNOWLEDGEMENTS

The authors thank Christian Panse, Riccardo Murri, Tyanko Aleksiev and Martin Ryan for the valuable discussions.

Funding: Scientific Exchange Programme NMS-CH (grant no. 12.289) and in part by the Swiss National Science Foundation (grant 310000–116502).

Conflicts of Interest: none declared.

REFERENCES

Anders,S. *et al.* (2013) Count-based differential expression analysis of RNA sequencing data using R and Bioconductor. *Nat. Protoc.*, **8**, 1765–1786.

Borthakur,D. (2007) *The Hadoop Distributed File System: Architecture and Design*. Hadoop Project Website 11 (2007): 21.

Frazee,A.C. *et al.* (2014) Differential expression analysis of RNA-seq data at single-base resolution. *Biostatistics*, **15**, 413–426.

Langmead,B. *et al.* (2010) Cloud-scale RNA-sequencing differential expression analysis with Myrna. *Genome Biol.*, **11**, R83.

Leśniewska,A. and Okoniewski,M.J. (2011) rnaSeqMap: a Bioconductor package for RNA sequencing data exploration. *BMC Bioinformatics*, **12**, 200.

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

Niemenmaa,M. *et al.* (2012) Hadoop-BAM: directly manipulating next generation sequencing data in the cloud. *Bioinformatics*, **28**, 876–877.

Schumacher,A. *et al.* (2014) Seqpig: simple and scalable scripting for large sequencing data sets in hadoop. *Bioinformatics*, **30**, 119–120.

Taylor,R.C. (2010) An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *BMC Bioinformatics*, **11** (**Suppl. 12**), S1.

Zaharia,M. *et al.* (2012) Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing. In: *Proceedings of the 9th USENIX Conference*. San Jose.