BRIEFINGS IN BIOINFORMATICS. VOL 15. NO 4. 637–647
Advance Access published on 7 February 2013
doi:10.1093/bib/bbs088

Survey of MapReduce frame operation in bioinformatics

Quan Zou, Xu-Bin Li, Wen-Rui Jiang, Zi-Yu Lin, Gui-Lin Li and Ke Chen

Submitted: 26th October 2012; Received (in revised form): 17th December 2012

Abstract

Bioinformatics is challenged by the fact that traditional analysis tools have difficulty in processing large-scale data from high-throughput sequencing. The open source Apache Hadoop project, which adopts the MapReduce framework and a distributed file system, has recently given bioinformatics researchers an opportunity to achieve scalable, efficient and reliable computing performance on Linux clusters and on cloud computing services. In this article, we present MapReduce frame-based applications that can be employed in the next-generation sequencing and other biological domains. In addition, we discuss the challenges faced by this field as well as the future works on parallel computing in bioinformatics.

Keywords: *MapReduce; bioinformatics; Hadoop*

INTRODUCTION

Bioinformatics researchers have continuously and progressively helped biologists in solving computational biology problems, particularly in dealing with large amounts of genomic data. Computing and sequencing capability has recently improved fast. Given the urgency of establishing a new computational framework, high-performance computing has become extremely important for large-scale data analysis. The message passing interface (MPI) and graphics processing unit (GPU) are pioneer programming application programming interfaces (APIs) for parallel computing. More recently, the cloud computing concept, which utilizes a parallel distributed computing framework with computer clusters and a web interface, has been presented. Cloud Web services such as the Amazon Elastic Compute Cloud (EC2) and Amazon Elastic MapReduce are commercially available, whereas the IBM/Google Cloud Computing University Initiative and the United States Department of Energy's Magellan service are free. Users generally upload their data by using a Web interface, after which they can perform operations on a remote client webpage.

MPI is a widely used traditional parallel programming model. Although MPI provides a powerful API for the general programmers, researchers with a biology background still consider the program complicated. Hadoop is written in Java with high-level programming and provides a streaming mode for easier script language calling. MPI passes messages fast with minimal delay, but it lacks a distributed file system such as the Hadoop Distributed File System (HDFS), which significantly improves data access efficiency. Given that the number of nodes may increase up to a certain threshold, the network incurs huge data transfer costs. However, the strong extensibility of Hadoop mitigates this

Corresponding author. Zou Quan, School of Information Science and Technology, Xiamen University, PR China.
Fax: +86-592-2580033; E-mail: zouquan@xmu.edu.cn

Quan Zou is an assistant professor in Xiamen University. His research interests include bioinformatics, scale data mining and parallel computing application.

Xu-Bin Li is a third-year undergraduate student in Xiamen University. His research interests include CUDA and Hadoop computing, probabilistic graphical model and bioinformatics.

Wen-Rui Jiang is a third-year undergraduate student in Xiamen University. His research interests include Hadoop computing and web data mining.

Zi-Yu Lin is an assistant professor in Xiamen University. His research interests include cloud database and real-time active data warehouses.

Gui-Lin Li is an assistant professor in Xiamen University. His research interests include database and wireless sensor network.

Ke Chen is an associate professor in Guangdong University of Petrochemical Technology. His research interests include web data mining and bioinformatics.

Table 1: Related software and projects on MapReduce

Software	Description
YARN	Next Generation Apache Hadoop MapReduce Framework. URL: http://hadoop.apache.org/docs/r0.23.0/hadoop-yarn/hadoop-yarn-site/YARN.html
MAPR	A complete distribution for Apache Hadoop and HBase™ that includes Pig, Hive, Mahout, Cascading, Sqoop, Flume and more. URL: http://www.mapr.com/
Hadoop Common	The common utilities that support the other Hadoop subprojects. URL: http://hadoop.apache.org
Hadoop Distributed File System (HDFS™)	A distributed file system that provides high-throughput access to application data. URL: http://hadoop.apache.org/hdfs/
Hadoop MapReduce	A software framework for distributed processing of large data sets on compute clusters. URL: http://research.google.com/archive/mapreduce.html
Avro™	A data serialization system URL: http://avro.apache.org
Cassandra™	A scalable multi-master database with no single points of failure. URL: http://assandra.apache.org
Chukwa™	A data collection system for managing large distributed systems. URL: http://incubator.apache.org/chukwa/
HBase™	A scalable, distributed database that supports structured data storage for large tables. URL: http://hbase.apache.org
Hive™	A data warehouse infrastructure that provides data summarization and ad hoc querying. URL: http://hive.apache.org
Mahout™	A Scalable machine learning and data mining library
Pig™	A high-level data-flow language and execution framework for parallel computation. URL: http://pig.apache.org
ZooKeeper™	A high-performance coordination service for distributed applications. URL: http://hadoop.apache.org/zookeeper/
Dryad	An implementation of extended MapReduce from Microsoft and Azure. URL: http://research.microsoft.com/en-us/projects/dryad/
Haloop	A modified version of the Hadoop MapReduce framework. It does not only extend MapReduce with programming support for iterative applications but also dramatically improves their efficiency by making the task scheduler loop-aware and by adding various caching mechanisms. URL: http://code.google.com/p/haloop/
Pregel	A system for large-scale graph processing, similar with Haloop and Twister. URL: http://kowshik.github.com/JPregel/pregel_paper.pdf
Twister	Efficient support for Iterative MapReduce computations, extremely faster than Hadoop. URL: http://www.iterativemapreduce.org/

problem. MPI cluster cannot deal with node failure. However, when a node fails in the Hadoop system, the framework will launch the same job in another normal node. Given the absence of load balancing, fault tolerance and a distributed file system, MPI is unreliable and insufficiently robust.

The compute unified device architecture (CUDA) programming model that is based on a GPU framework has recently facilitated the improvement of computing performance. The GPU architecture, in which one GPU chip contains hundreds of cores with threads that run in such cores, enables a GPU to execute multiple tasks simultaneously. According to the NVIDIA CUDA Programming Guide 4.0, the CUDA programming model is divided into three hierarchies, in which threads comprise a thread block and thread blocks comprise a grid. The parallel computing capability of the model is suitable for computation-intensive tasks. However, the memory size limitation of GPUs is a fatal flaw when faced with large-scale data.

MapReduce is an easy-to-use and general-purpose parallel programming model that is suitable for large data set analysis on a commodity hardware cluster developed by Google. On the other hand, the Apache Hadoop is a framework that enables the distributed processing of large data sets across clusters of computers based on the MapReduce model [1]. With the combination of MapReduce and the HDFS module, Hadoop aims to implement reliable, scalable and distributed computing.

Aside from the Hadoop framework, numerous other popular open source Apache projects are related to Hadoop, including HBase, Hive, Mahout, Pig and ZooKeeper. Dryad [2] is an extension of MapReduce from Microsoft and Azure is one of Microsoft's cloud technologies [3]. With Dryad, Microsoft proposes the use of a directed acyclic graph (DAG) to combine computational 'vertices' with communication 'channels' to model data flow graphs [4]. However, Dryad is a complicated program and employs a general execution layer-data flow with DAG, which incurs the risk that a new programming model cannot be expressed [5]. The related software and projects are shown in Table 1 and Figure 1.

Apache Hadoop and Microsoft Dryad/Azure are widely implemented in local systems, not only for their parallel capability but also for their easy deployment in a commodity hardware cluster. With only several PCs and a local network, a parallelized environment can be built based on Hadoop, and no other

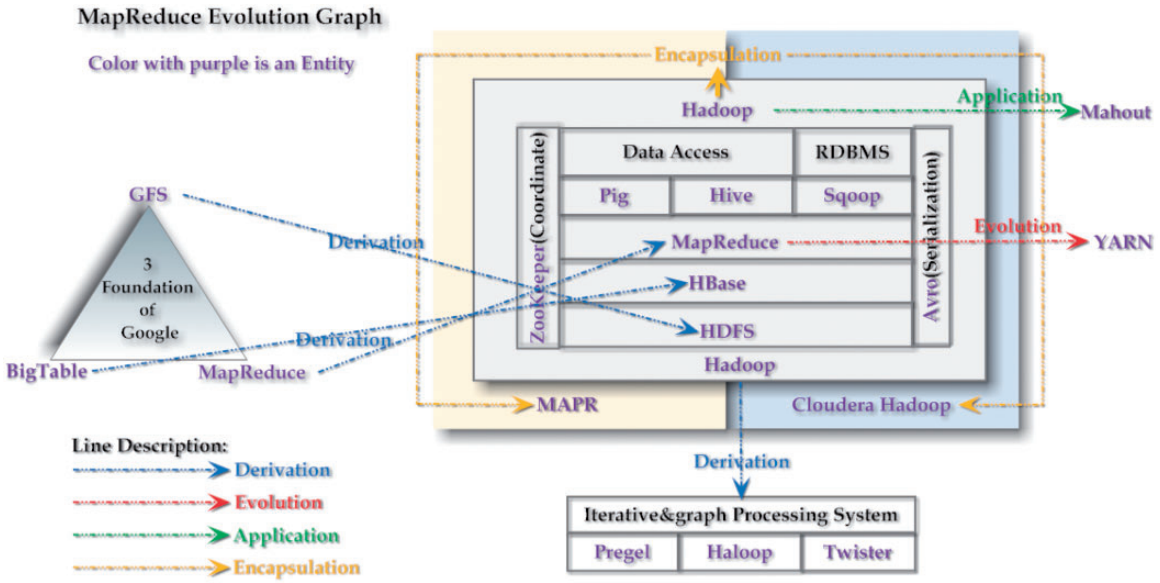

Figure 1: The relation of the MapReduce software and projects.

hardware is needed. This article discusses MapReduce applications in bioinformatics and gives suggestions for researchers.

SHUFFLE MECHANISM IN MAPREDUCE

MapReduce executes computations by two main functions, which are called Map and Reduce. A Map task takes a chunk of files as input and outputs a sequence of <key, value> pairs. The exact generation of <key, value> pairs is tailored by user preference. A Reduce task then turns the values associated to a same key into another value in certain user-defined manner.

The coordination of tasks and computing clusters is managed by the 'shuffle' mechanism of MapReduce. It detects the completion of all Map tasks, gets the data that need to be calculated, groups the values of the same key generated by each Map task and assigns the results to the correct computing cluster and Reduce task.

In order to observe the internal structure details of shuffle, we break it into two parts: 'shuffle@map' and 'shuffle@reduce'. The details of shuffle@map and shuffle@reduce are introduced in Figures 2 and 3.

As shown in Figure 2, the stage of shuffle@map can be decomposed to four finer granularities: Partition, Sort, Spill and Group. When a 'map' function finishes its work, the stream of <key, value> pairs are generated and buffered in memory. Before the data in memory can be written to local disk, the data are partitioned into r regions by a partitioning function. r is strictly equal to the number of reduce tasks. The partitioning function is responsible for appointing each reduce task a specific key to work on. The default partitioning function is simply a hash of key modulo r. But a user can replace this with another function if needed.

After the partition phase, each <key, value> pair has a partition label. Then the spill and group phases are implemented along with the sort phase. A spill thread starts flushing the data to the local disk, when the data in buffer reach the threshold. In the meantime, the data are sorted: first sorted by partition label and then sorted by key under the same partition label. After sorting, all pairs with same key are grouped together in the disk.

When the data on the disk reach a certain amount, they will be aggregated. All the <key, value> pairs having the same partition value will form a 'segment'. Within the segment, the spill files are merged into a single partitioned and sorted output file. All the pairs in a segment are sorted in ascending keys. When all the map tasks are finished, we can make an important conclusion: 'If the number of map tasks is m and the number of reduce tasks is r, then m big merged files should be created, while each one consists of r segments. These r segments are denoted by partition label from 0 to $r-1(0,1,2\ldots\ r-1)$, which represents the ith

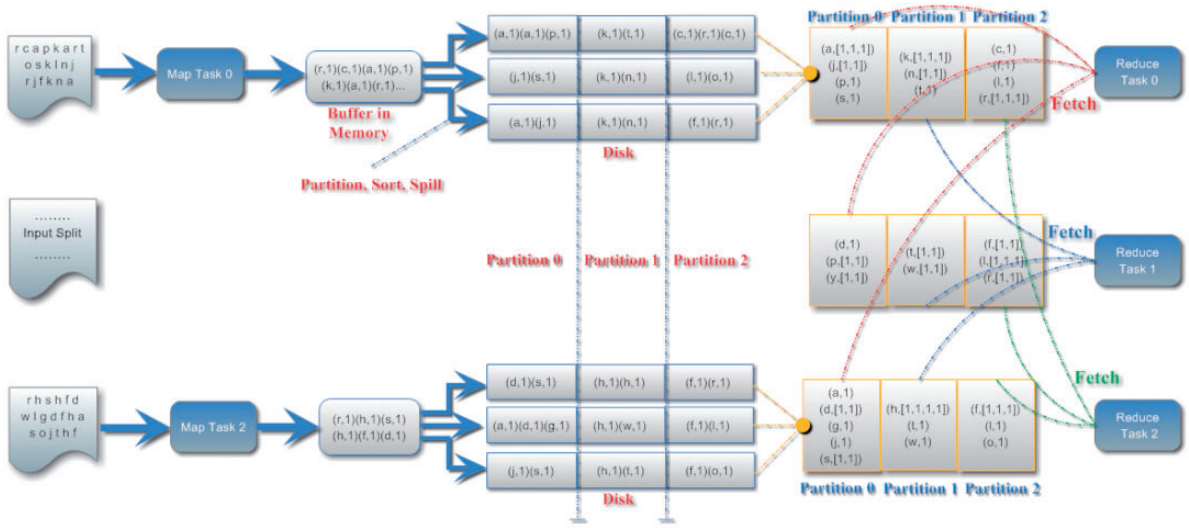

Figure 2: Illustration of 'shuffle@map'.

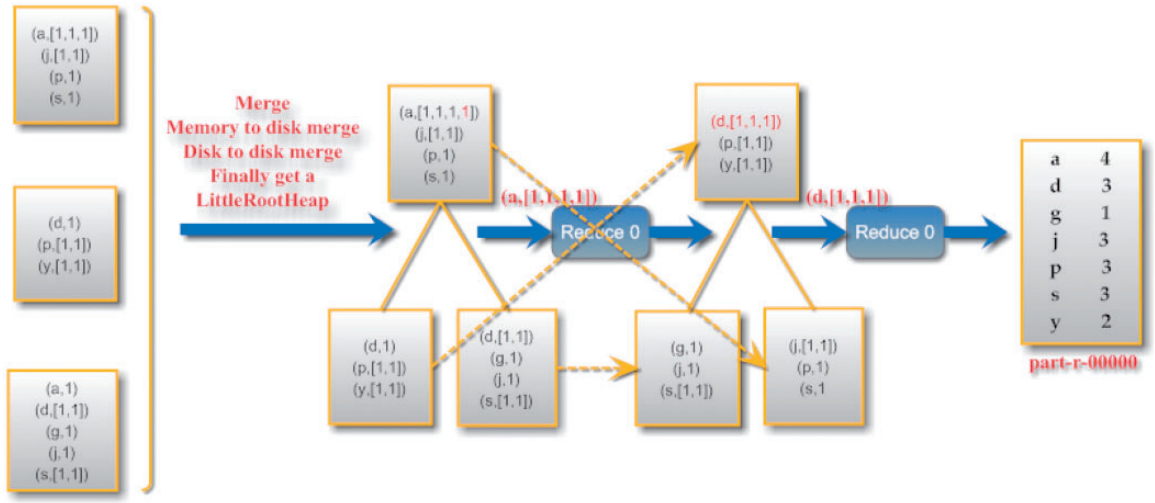

Figure 3: Illustration of 'shuffle@reduce'.

segment will be processed by the *i*th reduce task $(0 \leq i \leq r-1)$'.

We illustrate the partitioning function by a simple 'word counting' job. Suppose 'map tasks = reduce tasks = 3', single letter represents a word (there are 26 words totally). The partitioning function works are shown in Table 2. We can see from the table that the key c will be partitioned to the second segment and processed by the second reduce task.

When one of the map tasks is completed, the shuffle will move into the reduce stage: shuffle@reduce. Word counting in 'shuffle@reduce' is shown in Figure 3. As shown in Figure 3, shuffle@reduce consists of two stages: Copy and Merge. The goal of the copy phase is to get the data that need to be executed by the user-defined 'reduce' function. The copy thread of the *i*th reduce task will fetch all the segments with partition label *i*. A copy thread can be started as soon as each map task finishes. It does not need to wait until all map tasks have finished. And different copy threads work in parallel. As shown in Figure 2, three different copy threads are marked by different colors <red, blue and green>.

When the segments with the same partition label *i* flow to the *i*th reduce task, they will be stored. When the buffer reaches a threshold, these segments will be merged and spilled to disk. In this stage, there are two kinds of merge: memory-to-disk merge and disk-to-disk merge. The segments will be sorted in

Table 2: Partitioning function

Partitioning function		
0	1	2
a	b	c
d	e	f
g	h	i
j	k	l
m	n	o
p	q	r
s	t	u
v	w	x
y	z	

merge stage. The segments are stored in a heap structure, while each segment represents a tree node. When a reduce task fetches all the map outputs, the heap construction finishes. Then the reduce task starts executing user-defined 'reduce' function.

MAPREDUCE IN ALIGNMENT

The Basic Local Alignment Search Tool (BLAST) is widely used to search for similar regions between sequences. Before the Hadoop-based BLAST was created, mpiBLAST [6] and GPU-BLAST [7], which are, respectively, based on the MPI and GPU models, were in the spotlight. Given the lack of scalability and a well-designed framework that can hide system-level details for parallel programming, these two solutions are not adequately optimal and generalizable to perform the batch processing of huge amounts of sequence data [8]. By applying the Hadoop MapReduce framework, a parallel version of BLAST can be designed.

CloudBLAST [9] integrates MapReduce with virtual machine and virtual network technologies to deploy NCBI BLAST2 on a wide area network (WAN) between two distinct universities. CloudBLAST has been evaluated in both non-virtualized and local access network-based implementation. The comparison between mpiBLAST and CloudBLAST shows that the latter is more efficient than the former, which is the first time that Hadoop has achieved satisfactory WAN performance. bCloudBLAST [10] improves efficiency through the splitting of query sequences and of the sequence database for alignment. A traditional method such as CloudBLAST deals with query sequences by splitting and then sending these sequences to different nodes while copying a complete sequence database to each node. Therefore, the dual segmentation method presented by bCloudBLAST achieves efficient performance for a large sequence database.

Related works have been conducted by Gaggero *et al.* [11] and Leo *et al.* [12]. Gaggero *et al.* have parallelized BLAST and Gene Set Enrichment Analysis (GSEA) into Hadoop. A sequence alignment tool such as BLAST generally processes huge amounts of data, whereas a gene expression analysis tool such as GSEA generally deals with computation-intensive tasks with limited data. The experiments conducted by Gaggero *et al.* show that results are better by an order of magnitude for computationally intensive jobs even with a small number of nodes. Their work contributes to structural and genome-wide association studies that typically mix these two computational categories, such as GRAMMAR [13]. Leo *et al.* [12] tested the three aforementioned tools, and the results indicate that: (i) the Hadoop framework is capable of dealing with canonical (light computation and large data set) and non-standard problems (heavy computation and small data set), (ii) implementation performance is related to a number of parameters (e.g. number of nodes, number of map tasks and HDFS block size) and (iii) the use of virtual machines introduces a very small overhead (<5%).

Sadasivam *et al.* [14] proposed a novel approach that combines the dynamic programming algorithm with the computational parallelism of Hadoop data grids to improve accuracy and to accelerate of multiple sequence alignment (MSA). The new method reduces the time complexity of MSA from $O(N^n)$ (by using dynamic programming) to $O((n-1) \times N^2/b)$, where N represents the average length of 'n' sequences and b represents the block size.

Recently, Yang *et al.* [8] demonstrated a BLAST-like long-sequence alignment algorithm with MapReduce. They employed a database segmentation strategy that is similar to bCloudBLAST, which segments a large sequence database into small blocks of fixed size. Query sequences will be processed by using the following steps: the word list is built and the scanner is constructed; a scan is conducted for hits and such hits are extended during Map; results are sorted during 'Shuffle' and results are reported during Reduce. However, the fixed size for splitting a long sequence is difficult to define.

MAPREDUCE IN MAPPING AND ASSEMBLY

Next-generation technology, which features low cost and high throughput, generates an enormous amount of sequence data (reads). Next-generation equipments such as Illumina Solexa, HiSeq 2000, 454 Life Sciences sequencer (Roche, Applied Biosystems' SOLiD systems) and Ion Proton (Life Technologies) continue to be widely used. Given their short lengths (<500 bp), reads need to be pre-processed before further analysis. 'Mapping' and 'assembly' are the key steps. Mapping requires a pre-existing reference genome sequence that can be employed to align the reads. Biological analyses including gene expression, single-nucleotide polymorphism (SNP) discovery and genotyping are highly associated with mapping. It ought to be noticed that mapping in MapReduce is different from mapping in bioinformatics. The former is a program working mechanism, while the latter is a problem for reads assembly in bioinformatics. Assembly has two categories. The first category utilizes the output of mapping and employs assemble algorithms to obtain a new genome, called the reference assembly. The other category does not need a reference genome sequence but requires a process for constructing complex data structures, called *de novo* assembly. *De novo* assembly contributes to the discovery of new or unknown sequences and is currently indispensable in biological research.

MapReduce in mapping

BlastReduce with CloudBurst [15] was proposed by Schatz in 2009 to map short reads against a reference genome by using the MapReduce framework. Recently, the BlastReduce project has been discontinued and superseded by CloudBurst. BlastReduce uses the 'seed-and-extend' alignment algorithm, similar to BLAST, except that BlastReduce adopts the LandauVishkin algorithm to extend the exact seeds and to find alignments with a maximum of *k*-differences quickly.

CloudBurst employs an alignment algorithm modeled after RMAP [16] to map reads with any number of mismatches or differences. With the support of Hadoop, CloudBurst scales linearly as the number of reads increases and speeds up linearly as the size of the cluster increases. However, BlastReduce and CloudBurst have insufficient enrich features compared with traditional map/align tools such as SOAP, MAQ, ZOOM and RMAP. The most serious limitation is that CloudBurst does not support pair-end reads as well as bisulfite (BS) and fastq format input. SeqMapReduce [17] have achieved 57.9 times faster than CloudBurst and also presented a web server. Unfortunately, the web server is not accessible now.

SEAL [18] not only integrates BWA [19] to map pair-end reads but also removes duplicate reads according to the same criteria as Picard MarkDuplicates [20]. The two main programs of SEAL are 'PairReadsOseq' and 'Seqal'. PairReadsOseq converts qseq- or fastq-format files into prq-format files. Seqal implements read alignment and optional performs duplicate read removal. SEAL is deployed into the Hadoop framework by using Pydoop, which has satisfactory speed and scalability. Another workflow [21], also developed by the authors of SEAL, has an additional step that recalibrates base quality scores by using the genome analysis toolkit (GATK) [22].

A relatively complete tool for sequence mapping is CloudAligner [23], which supports BS, pair-end mapping, fastq format input, SAM/BED6 format output, Web interface and long reads with efficient performance. However, the limitations of CloudAligner included the incapability to operate in a Web server, lave of a README file for the project, lack of instructions on handling the software and lack of long-term updates.

MapReduce in assembly

Assembly algorithms can be classified into the overlap-layout-consensus (OLC) approach and the de Bruijn graph approach [24, 25]. The OLC approach is typically used to assemble long, high-quality reads. This approach has also been used in Celera (pipeline version: CABOG), Newbler (support Roche 454 Data), Arachne, EDENA (support Solexa and SOLiD data), etc. The de Bruijn approach, which was first proposed by Euler [26] in 2001, performs well in assembling short reads such as RNA-seq and resolves the 20-year-old 'repeat problem' in fragment assembly. Examples of the de Bruijn approach include Velvet, ALLPATHS, ABySS, SOAPdenovo and Contrail. Another prefix tree-based algorithm is introduced by SSAKE and is applied in SHARCGS and VCAKE.

Among the aforementioned tools, several assemblers are parallel powered. ABySS and Ray are MPI-based, whereas Contrail is Hadoop-based.

Based on implementations of Contrail, assemblers that are based on graphs have limited success in scaling to large mammalian-sized genomes because of the large memory that is needed to construct and manipulate graphs. To mitigate this problem, Contrail combines the de Bruijn graph algorithm with Hadoop for the *de novo* assembly of large genomes with short sequencing reads [27].

Apart from *de novo* assemblers, HPC-MAQ [28] is a short-read reference assembler based on Hadoop and is revised from MAQ to achieve higher efficiency. No additional information on HPC-MAQ is currently available.

MAPREDUCE IN GENE EXPRESSION ANALYSIS AND SNP ANALYSIS

RNA-seq data analysis is a method that is widely used to study gene expression levels and to identify variants that are important for the study of cell difference, medical diagnosis, phylogenies and drug design. Expression analysis is generally processed by using the following steps: First, the reads are aligned to the reference and then categorized according to the transcribed feature. Second, the distribution of reads that are assigned to each feature is normalized. Finally, a statistical analysis is conducted to identify which features exhibit differential abundance [29].

Myrna [29] is a cloud-computing tool for analyzing gene expression in large-scale RNA-seq data. A paper on Myrna recommends the use of a permutation model but not a single Poisson model, which is inappropriate for biological replicates. Given the different needs of users, Myrna provides three operating modes. 'Cloud mode', which is designed to operate under Amazon Elastic, is suitable for users without hardware support. 'Hadoop mode' requires an actual Hadoop cluster and is suitable for users who are proficient with Linux. 'Singleton mode', which is driven is a similar manner as multi-threading, can be run under single computer without a Hadoop environment. The primary limitation of Myrna is that expression signals may be lost in the absence of a process that can align reads across exon junctions.

Considering the difficulty in aligning spliced junction reads to a reference genome, a new method that maps such reads against previous known or predicted transcripts has been proposed [30]. Based on this approach, Hong *et al.* [31] developed an RNA-seq analysis tool called friendly gene eXpression analytic tool (FX) for gene expression level analysis and genomic variant calling. The output of FX contains gene expression profiles, SNP calling and short indels. Moreover, FX can be run on the cloud and a local Hadoop cluster with a friendly interface.

Crossbow [32] focuses on whole-genome resequencing and SNP detection by combining the aligner Bowtie and the SNP caller SOAPsnp [33]. To illustrate its efficiency, Crossbow aligns and calls SNPs from the 38-fold coverage of a human genome with ~2.66 billion reads in less than 3 h on a 320-core cluster (Amazon EC2). However, Crossbow has inherited a limitation from Bowtie, that is, only a maximum of three mismatches can be detected in its mapping stage [23]. Crossbow and Myrna both use the Hadoop streaming mode with Perl scripts as wrappers.

Kim *et al.* [34] have developed a cloud-computing tool for SNP detection as well as a visualization interface called Sequence_Analyzer. According to a previous paper, the SNP detection method considers the alignment scores, base qualities, allete call scores and sequencing errors. Moreover, results of SNP calls can be imported into Sequence_Analyzer for convenient verification. CloudTSS [35] proposes a Hadoop-based method, which is revised from traditional dynamic algorithms to detect haplotype blocks with improved computation efficiency compared with the sequential method. However, the two aforementioned works are not open source.

MAPREDUCE IN OTHER BIOLOGICAL APPLICATIONS

MapReduce in quality assurance of NGS data

Given the rapidly increasing rate of NGS data size, quality assurance (QA) tools show the importance of ensuring the minimum necessary standard for downstream analysis. Quake [36] is a QA tool that provides quality-aware detection and correction of sequencing errors by using the redundancy inherent in the sequence data. Experiments described in a paper show that: (i) Quake aids Velvet in producing better assemblies and can correct more reads than SOAPdenovo in a *de novo* assembly and (ii) Quake aids SNP detection tools such as Bowtie and SAMtools in identifying more true SNPs. Although Quake has achieved impressive performance and is capable of dealing with a billion reads, the

processing of billion-level reads is time-consuming because of the absence of parallel technical support.

SAMQA [37] is an efficient parallel tool that is used to identify errors in population-scale sequence data. The quality test includes the default technical test according to the SAM specification and the biological validation test through expert analysis. SAMQA takes the advantages of Hadoop-BAM to process input BAM files, those of the Picard toolset from SAMtools to conduct technical tests and those of the Hadoop MapReduce framework to achieve scalable and robust computing.

MapReduce in optimizing works

Sequence alignment is not only important in searching for similar regions among multiple sequences but also in terms of other aspects including gene expression analysis, microRNA study, mapping stage in NGS and *de novo* assembly. However, sequence alignment always needs a precomputed index of the sequence to improve efficiency. Suffix array (SA) and Burrows–Wheeler Transform (BWT) are popular index structures. Menon *et al.* [38] proposed a Hadoop-based algorithm for constructing SA and BWT structures. The new parallel algorithm significantly improves the performance of Vmatch (by using SA) and Bowtie (by using BWT).

The BAM format [39], a compact and indexed representation of nucleotide sequence alignments, is widely used to compress sequence alignment/map data into binary data. Analysis tools (e.g. GATK and SEAL) do not have efficient parallel access to BAM files because their low-level structure hinders adoption. Hadoop-BAM [40] is a novel library between BAM files and Hadoop-based analysis applications that provides two efficient modes of access to BAM files and an easy-to-use interface that is presented with a Picard-compatible Java API.

The GATK [22] is a structured and MapReduce-based programming framework that is designed for the easier development of analysis tools for NGS data. The architecture of GATK including traversal types, sharding, merging input files and parallelization separates the manipulation of massive data from specific analysis algorithms. Given GATK's robustness and efficiency, numerous GATK-based tools are being developed, such as depth of coverage, simple Bayesian genotyper and base quality score recalibration.

MapReduce in other works

The MapReduce framework is not only used in processing NGS data but is also applied in other aspects of bioinformatics. Numerous tools have been designed based on MapReduce, e.g. PeakRanger [41] (a peak caller for ChIP-seq data with a good performance), YunBe [42] (a gene set analysis tool for biomarker identification), RSD [43] (a reciprocal smallest distance algorithm for comparative genomics), BioVLAB-MMIA [44] (an integrated analysis tool for microRNA and mRNA expression data) and AutoDockCloud (the MapReduce version of AutoDock for virtual molecular docking). A number of special cases have also caught our attention:

- Efficient graph algorithms. Lin and Schatz presented three design patterns of MapReduce graph algorithms, which emerged as an enabling technology for large-scale graph processing. Their work is directly related to the module detection of protein–protein interaction networks and other topological analysis of biological networks.
- SeqWare Query Engine [45], which is structured with the scalable NoSQL HBase database from the Hadoop project, supports the databasing of information from thousands of genomes. This tool is a good example for building a scalable and efficient database while facing large-scale NGS data.
- Cloudera's Distribution for Hadoop (CDH) helps organizations in deploying a functional, scalable and flexible Hadoop distribution environment and in reducing their technical and administrative requirements. The CDH3 package includes almost entire popular Apache projects (e.g. Hadoop, Hive, HBase and Mahout).
- DOE's Cloud Computing: Magellan projects [46], which were awarded as 'Best use of HPC in the Cloud', benefit from the high-performance computing of Hadoop and HBase.

Table 3 is the conclusion of the parallel bioinformatics tools, which are free and also available from Internet.

DISCUSSION AND CONCLUSION

Hadoop with other Apache open-source projects and cloud computing will become more popular for its scalable, efficient and fault-tolerant/robust features. Moreover, bioinformatics will be continually

Table 3: Available and free parallel bioinformatics software Tools

Function	Name	URL	Reference
Mapping	CloudBurst	http://cloudburst-bio.sourceforge.net/	[15]
	SEAL	http://biodoop-seal.sourceforge.net/	[18]
	CloudAligner	http://cloudaligner.sourceforge.net	[23]
	Crossbow	http://bowtie-bio.sourceforge.net/crossbow	[32]
RNA-Seq data analysis	Myrna	http://bowtie-bio.sourceforge.net/myrna	[29]
	FX	http://fx.gmi.ac.kr/	[31]
	Eoulsan	http://transcriptome.ens.fr/eoulsan/	[47]
Dealing with sequence files	Picard	http://picard.sourceforge.net/	None
	Hadoop-BAM	http://sourceforge.net/projects/hadoop-bam/	[40]
	SeqWare	http://seqware.sourceforge.net/	[45]
Others	PeakRanger	http://ranger.sourceforge.net/	[41]
	YunBe	http://lrcv-crp-sante.s3-website-us-east-1.amazonaws.com/	[42]
	GATK	http://genome.cshlp.org/content/20/9/1297/suppl/DC1	[22]
	usm	http://usm.github.com/	[48]
GPU software	GPU-BLAST	http://eudoxus.cheme.cmu.edu/gpublast/gpublast.html	[7]
	SOAP3	http://soap.genomics.org.cn/soap3.html	[49]

confronted with large-scale data, not only from NGS but also from specific databases. The Hadoop framework has recently been found to be the most suitable method for handling bioinformatics data [48]. With cloud computing and some Web servers becoming more common and convenient, a number of problems must be considered, such as the time cost for large input data from local to remote servers under a slow network, charges for cloud computing, data security and privacy and problems with iterative structure, which are difficult for MapReduce to mitigate.

After the Jcuda (Java version of CUDA) was introduced, the study of CUDA on Hadoop clusters has been burgeoning. With the combination of the scalability of Hadoop and the super computational capability of CUDA, 'CUDA on Hadoop' shows promising performance. However, CUDA and Jcuda cannot be utilized so far in the Hadoop environment, since Hadoop is based on HDFS while CUDA should be driven in the local disk. In HDFS environment, the graphic memory could not be found since the NVIDIA graphic card is not driven. But we consider that this technique problem will be solved soon by the NVIDIA company and Apache, as the combine of GPU and MapReduce shows promising expectation.

In the short term, traditional bioinformatics resources, including tools and databases, will be redesigned or planted to support Hadoop MapReduce as an increasing number of researchers direct their attention toward high-performance computing.

Key Points

- The Apache Hadoop gives researchers a possibility of achieving scalable, efficient and reliable computing performance on Linux clusters and cloud computing services. We present MapReduce frame-based applications that can be employed in bioinformatics.
- We introduce the flow of shuffle in MapReduce, which can help the bioinformatics researchers to understand the mechanism of MapReduce.
- We discuss about the future research of parallel computation in bioinformatics and give our suggestion.

FUNDING

The work was supported by the Natural Science Foundation of China [61001013, 61102136 and 61100032]; the Natural Science Foundation of Fujian Province of China [2011J05158] and the Fundamental Research Funds for the Central Universities of China [2011121049 and 2010121072].

References

1. The Apache Software Foundation. *Apache Hadoop Home Page*. http://hadoop.apache.org.
2. Isard M, Budiu M, Yu Y, *et al*. Dryad: distributed data-parallel programs from sequential building blocks. *SIGOPS Oper Syst Rev* 2007;**41**:59–72.
3. Qiu X, Ekanayake J, Beason S, *et al*. Cloud technologies for bioinformatics applications. *Proceedings of the 2nd Workshop on Many-Task Computing on Grids and Supercomputers, Portland, Oregon, 2009*.
4. Yu Y, Isard M, Fetterly D, *et al*. DryadLINQ: a system for general-purpose distributed data-parallel computing using a high-level language. In: *Proceedings of the 8th USENIX*

Conference on Operating Systems Design and Implementation, San Diego, California, 2008.

5. Hindman B, Konwinski A, Zaharia M, *et al*. *Nexus: A Common Substrate for Cluster Computing*. University of California, Berkeley: EECS Department. UCB/EECS-2009-158, November 16, 2009.
6. Carey L, Darling A, Feng W. The design, implementation, and evaluation of mpiBLAST. In: *4th International Conference on Linux Clusters: The HPC Revolution 2003 in Conjunction with ClusterWorld Conference & Expo, 2003*.
7. Vouzis PD, Sahinidis NV. GPU-BLAST: using graphics processors to accelerate protein sequence alignment. *Bioinformatics* 2011;**27**:182–8.
8. Yang X-L, Liu Y-L, Yuan C-F, *et al*. Parallelization of BLAST with MapReduce for long sequence alignment. In: *2011 Fourth International Symposium on Parallel Architectures, Algorithms and Programming, Tianjin, 2011*.
9. Tsugawa M, Matsunaga A, Fortes J. CloudBLAST: combining MapReduce and virtualization on distributed resources for bioinformatics applications. *IEEE Fourth International Conference on eScience, Indianapolis, IN, 2008*.
10. Zhen M, Jianhui L, Yunchun Z, *et al*. bCloudBLAST: an efficient mapreduce program for bioinformatics applications. In: *Biomedical Engineering and Informatics (BMEI), 2011 4th International Conference, 2011, 2072–6*.
11. Gaggero M, Leo S, Manca S, *et al*. Parallelizing bioinformatics applications with MapReduce. In: *CCA-08: Cloud Computing and its Applications, Chicago, IL, 2008*.
12. Santoni F, Leo S, Zanetti G. Biodoop: bioinformatics on Hadoop. In: *2009 International Conference on Parallel Processing Workshops, Vienna, 2009*.
13. Aulchenko YS, de Koning DJ, Haley C. Genomewide rapid association using mixed model and regression: a fast and simple method for genomewide pedigree-based quantitative trait loci association analysis. *Genetics* 2007;**177**: 577–85.
14. Sudha Sadasivam G, Baktavatchalam G. A novel approach to multiple sequence alignment using hadoop data grids. *Int J Bioinform Res Appl* 2010;**6**:472–83.
15. Schatz MC. CloudBurst: highly sensitive read mapping with MapReduce. *Bioinformatics* 2009;**25**:1363–9.
16. Smith AD, Chung WY, Hodges E, *et al*. Updates to the RMAP short-read mapping software. *Bioinformatics* 2009;**25**: 2841–2.
17. Li Y, Zhong S. SeqMapReduce: software and web service for accelerating short sequence mapping. In: *Ninth International Conference for the Critical Assessment of Massive Data Analysis, Chicago, IL, USA, 2009*.
18. Pireddu L, Leo S, Zanetti G. SEAL: a distributed short read mapping and duplicate removal tool. *Bioinformatics* 2011;**27**: 2159–60.
19. Li H, Durbin R. Fast and accurate short read alignment with Burrows-Wheeler transform. *Bioinformatics* 2009;**25**: 1754–60.
20. brilliantred alecw, tfenne. *Picard MarkDuplicates Home Page*. http://picard.sourceforge.net.
21. Pireddu L, Leo S, Zanetti G. MapReducing a genomic sequencing workflow. In: *Proceedings of the Second International Workshop on MapReduce and its Applications, San Jose, California, USA, 2011*.
22. McKenna A, Hanna M, Banks E, *et al*. The genome analysis toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res* 2010;**20**: 1297–303.
23. Nguyen T, Shi W, Ruden D. CloudAligner: a fast and full-featured MapReduce based tool for sequence mapping. *BMC Res Notes* 2011;**4**:171.
24. Jourdren L, Bernard M, Dillies MA, Le Crom S. Eoulsan: a cloud computing-based framework facilitating high throughput sequencing analyses. *Bioinformatics* 2012;**28**:1542–3.
25. Schatz MC, Delcher AL, Salzberg SL. Assembly of large genomes using second-generation sequencing. *Genome Res* 2010;**20**:1165–73.
26. Pevzner PA, Tang H, Waterman MS. An Eulerian path approach to DNA fragment assembly. *Proc Natl Acad Sci USA* 2001;**98**:9748–53.
27. Schatz M, Gupta A, Gupta R, *et al*. Contrail Home Page. http://sourceforge.net/apps/mediawiki/contrail-bio/index.php?title=Contrail.
28. Prasad VVS, Loshma G. HPC-MAQ: a parallel short-read reference assembler. *CCSEA 2011, 2011*.
29. Langmead B, Hansen KD, Leek JT. Cloud-scale RNA-sequencing differential expression analysis with Myrna. *Genome Biol* 2010;**11**:R83.
30. Trapnell C, Williams BA, Pertea G, *et al*. Transcript assembly and quantification by RNA-Seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nat Biotechnol* 2010;**28**:511–5.
31. Hong D, Rhie A, Park SS, *et al*. FX: an RNA-Seq analysis tool on the cloud. *Bioinformatics* 2012;**28**:721–3.
32. Langmead B, Schatz MC, Lin J, *et al*. Searching for SNPs with cloud computing. *Genome Biol* 2009;**10**:R134.
33. Li R, Li Y, Fang X, *et al*. SNP detection for massively parallel whole-genome resequencing. *Genome Res* 2009;**19**: 1124–32.
34. Deok-Keun K, Jee-Hee Y, Jin-Hwa K, *et al*. Cloud-scale SNP detection from RNA-Seq data. In: *Data Mining and Intelligent Information Technology Applications (ICMiA), 2011 3rd International Conference, 2011*;321–323.
35. Hung C-L, Lin Y-L, Hua G-J, *et al*. CloudTSS: a TagSNP selection approach on cloud computing. *Grid and Distributed Computing* 2011;**261**:525–34.
36. Kelley DR, Schatz MC, Salzberg SL. Quake: quality-aware detection and correction of sequencing errors. *Genome Biol* 2010;**11**:R116.
37. Robinson T, Killcoyne S, Bressler R, *et al*. SAMQA: error classification and validation of high-throughput sequenced read data. *BMC Genomics* 2011;**12**:419.
38. Menon RK, Bhat GP, Schatz MC. Rapid parallel genome indexing with MapReduce. In: *Proceedings of the Second International Workshop on MapReduce and its Applications, San Jose, California, USA, 2011*.
39. Li H, Handsaker B, Wysoker A, *et al*. The sequence alignment/map format and SAMtools. *Bioinformatics* 2009;**25**: 2078–9.
40. Niemenmaa M, Kallio A, Schumacher A, *et al*. Hadoop-BAM: directly manipulating next generation sequencing data in the cloud. *Bioinformatics* 2012;**28**:286–7.
41. Feng X, Grossman R, Stein L. PeakRanger: a cloud-enabled peak caller for ChIP-seq data. *BMC Bioinformatics* 2011;**12**:139.

42. Zhang L, Gu S, Liu Y, *et al*. Gene set analysis in the cloud. *Bioinformatics* 2012;**28**:294–5.

43. Wall DP, Kudtarkar P, Fusaro VA, *et al*. Cloud computing for comparative genomics. *BMC Bioinformatics* 2010;**11**:259.

44. Hyungro L, Youngik Y, Heejoon C, *et al*. BioVLAB-MMIA: a reconfigurable cloud computing environment for microRNA and mRNA integrated analysis. In: *Bioinformatics and Biomedicine (BIBM), 2011 IEEE International Conference, 2011*;494–9.

45. O'Connor BD, Merriman B, Nelson SF. SeqWare Query Engine: storing and searching sequence data in the cloud. *BMC Bioinformatics* 2010;**11(Suppl 12)**:S2.

46. Argonne Leadership Computing Facility. *Magellan Project Home Page*. http://magellan.alcf.anl.gov/.

47. Taylor RC. An overview of the Hadoop/MapReduce/HBase framework and its current applications in bioinformatics. *BMC Bioinformatics* 2010;**11(Suppl 12)**:S1.

48. Almeida JS, Gruneberg A, Maass W, Vinga S. Fractal MapReduce decomposition of sequence alignment. *Algorithms Mol Biol* 2012;**7**:12.

49. Liu C-M, Wong T, Wu E, *et al*. SOAP3: ultra-fast GPU-based parallel alignment tool for short reads. *Bioinformatics* 2012;**28**:878–9.