Hadoopizer: a cloud environment for bioinformatics data analysis

Anthony Bretaudeau (1), Olivier Sallou (2), Olivier Collin (3)

(1) anthony.bretaudeau@irisa.fr, INRIA/Irisa, Campus de Beaulieu, 35042, RENNES, Cedex, France
(2) olivier.sallou@irisa.fr, INRIA/Irisa, Campus de Beaulieu, 35042, RENNES, Cedex, France
(3) olivier.collin@irisa.fr, INRIA/Irisa, Campus de Beaulieu, 35042, RENNES, Cedex, France
Next generation sequencing (NGS)

Very cheap DNA sequencing

Compute and storage grow slower than sequence production

“Data deluge”

This is the first wave

- Proteomic
- Imaging
- ...
DNA

Alphabet = A, T, G, C (1 letter = 1 “base”)
Human genome = 3 Gb

Next generation sequencing (NGS)
1 run (11 days) = 6 Billion reads of 100 b

Many methods
De novo assembly of a genome
Mapping on a genome

...
Many algorithms (101*)

Sequencing errors, mutations, repetitions, gaps

Parallelization

Indexed genome

Each read can be mapped individually

* source: seqanswers.com wiki
Data management

Production facilities
  Sequencing centers
    Genoscope, BGI, companies, ...
  Sequencers in labs => disseminated

Computing resources
  Disseminated too => transfers
    in labs, on platforms (or none)
  Peaks of activity

=> Engineering challenges
Why the cloud

Extensible computing resources
  Amazon, Azure, private clouds
Parallelization
  SGE or Hadoop clusters
Reproducible results
  Shareable virtual machines
Hadoop

MapReduce framework
Distributed processing of large datasets
Scalability
  1 to thousands of nodes
Fault tolerance
  Detect node failure and retry
Hadoop-related projects
  Hbase, Hive, Cassandra, Pig, ...
Hadoop in a few words

1. Load input data as key/values
2. Distribute them to computing node
3. Map(): transform to new key/values pairs
4. Reduce(): combine values having the same key
5. Write to output file
Cloud computing and bioinformatics data:

Does it work?
Do we miss some tools?
Will it be efficient?

Private cloud setup: GenoCloud
Inventory of existing solutions
Development of Hadoopizer
Private cloud

Private cloud for test purpose
Based on OpenNebula
  240 cores, 940 Gb memory, 8 Tb storage
  EC2 compatibility
Ready to use images
  Sge cluster
  Hadoop cluster

http://genocloud.genouest.org
Already existing tools:

CloudAligner
  Mapping (specific algorithm)

CloudBurst
  Mapping (RMAP algorithm)

Contrail
  De novo assembler (specific algorithm)

Crossbow
  SNPs detection (uses Bowtie and SOAPsnp)

Myrna
  RNAseq, differential expression (bowtie, R/Bioconductor)
Inventory

Less specific tools:

Eoulsan
  Filtering, mapping, differential expression
  Pipelines
  Extendable (but not simple)

CloudMan
  SGE cluster (with console) + Galaxy frontend
Main limitations

Fast evolution of both data & algorithms
  Obsolescence of algorithms (myrna, cloudburst, contrail)

Evolution cost
  Test new algorithms
  Custom code/scripts (very common)

Incompatible dependencies

Missing
  Launching custom command line
  Splitters adapted to bioinformatics data formats
  Some glue to make life easier

=> Hadoopizer
Non parallelized treatment

> a sequence
AAAATGC\text{CGTACCGT}
> another sequence
TGTC\text{GTACTGGTGAC}
> third sequence
TGTC\text{GTACAACGTTCGA}
...

**EXECUTION**

of command line on all data

---

Big input file

---

Complete result file
Hadoopizer: how it works

Input chunks

Big input file

SPLITTING with specific splitter

EXECUTION of command line on received data

REDDUCING with specific reducer

Complete result file
Hadoopizer: using it

Xml example

```xml
<?xml version="1.0" encoding="utf-8"?>
<job>
  <command>
    bowtie -m 1 --best --strata -S ${genome} ${reads} > ${mapped}
  </command>

  <input id="genome">
    <url autocomplete="true">/home/example/indexed_genome</url>
  </input>

  <input id="reads" split="true">
    <url splitter="fastq">/home/example/reads.fastq</url>
  </input>

  <outputs>
    <url>/home/example/output_mapping/</url>
    <output id="mapped" reducer="sam" />
  </outputs>
</job>
```

Command line example

```
hadoop jar hadoopizer.jar -c job_config.xml -w hdfs://master_node_ip/bowtie_tmp/
```
Hadoopizer: using it

Xml example

```xml
<?xml version="1.0" encoding="utf-8"?>
<job>
  <command>
    bowtie -m 1 --best --strata -S ${genome} ${reads} > ${mapped}
  </command>

  <input id="genome">
    <url autocomplete="true">/home/example/indexed_genome</url>
  </input>

  <input id="reads" split="true">
    <url splitter="fastq">/home/example/reads.fastq</url>
  </input>

  <outputs>
    <url>/home/example/output_mapping/</url>
    <output id="mapped" reducer="sam"/>
  </outputs>
</job>
```

Command line example

```
hadoop jar hadoopizer.jar -c job_config.xml -w hdfs://master_node_ip/bowtie_tmp/
```
Hadoopizer: other features

Software deployment

    hadoop jar hadoopizer.jar -b binaries.tar.gz [...]  
    Extracted in work dir on each node

Supported formats

    Fasta, Fastq, Sam, (Bam)

Compression

    Input and output

Multiple input+output

    Support of paired sequences
Mapping example: first results

First results on mapping:

- Reference genome: 400 Mb
- Reads: 11 Gb

With hadoopizer

- 343 splits on 10 nodes
- 55 min for map step, 3h for reduce step

This is a test cloud, not optimized for performances
Configuration tuning, code improvements, ...

Same command on 1 machine (same config)

4h30
Benchmarks

The next step of the project

Comparison with:

- non parallelized
- SGE parallelized
- other implementations using Hadoop

Take into account the transfer time

Expected results:

- Overhead due to I/O on computing nodes
Coarse grain parallelism
For embarrassingly parallel problems
Works with any command line
Perspectives
  Support more data formats (bed, wiggle, gff, ...)
Performance issues
6 months work

1.0 released on github

    https://github.com/genouest/hadoopizer

Open position to continue the project

   Benchmarks

   Public clouds

   Support other formats, new features

   Real life applications
Thank you!

www.genouest.org
genocloud.genouest.org
github.com/genouest/hadoopizer

support@genouest.org