

Connecting big data to R, using SparkR, parquet and jupyter - a genomic example

Michał Okoniewski, Scientific IT Services, ETH Zurich

Questions

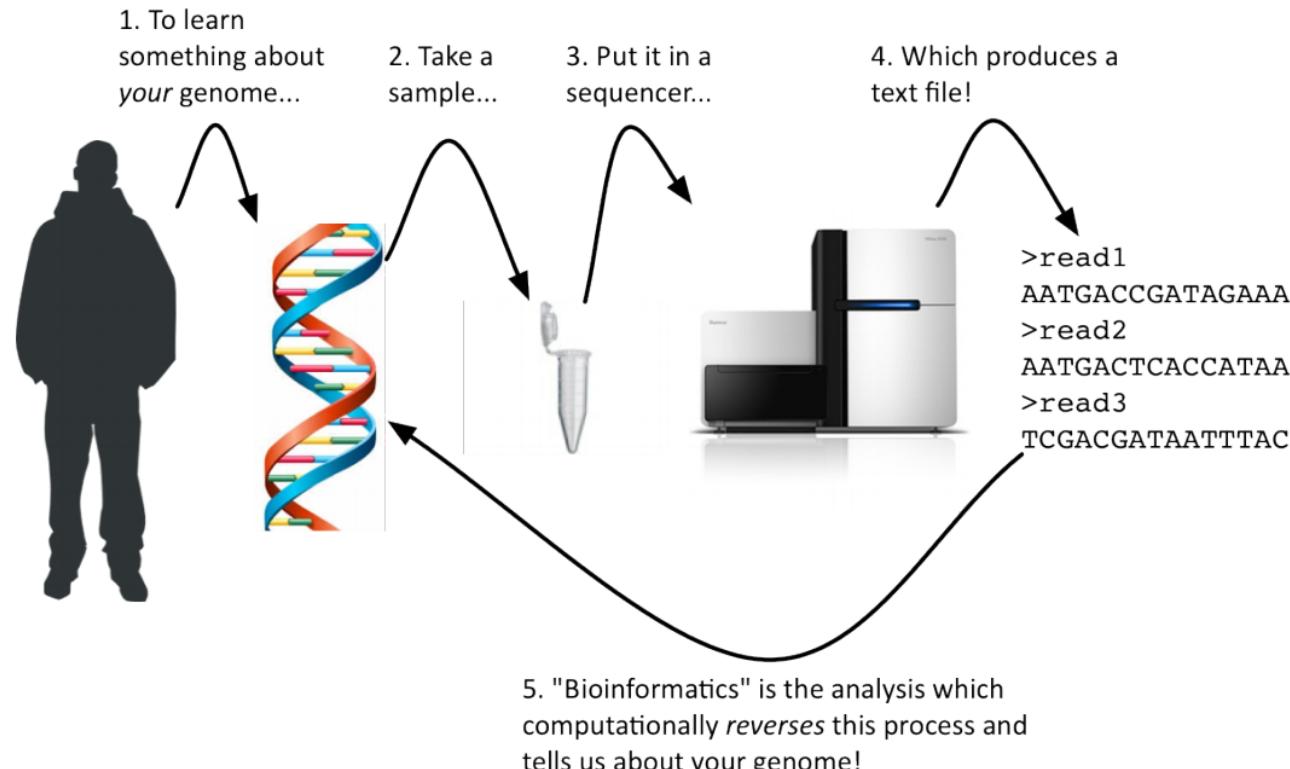
Can we analyze **BIG DATA** with R?

Are we technically ready for personalized,
genomics-based medicine?

Is R useful for it?

Next-generation sequencing (NGS)

A One-Slide Introduction to Genomics



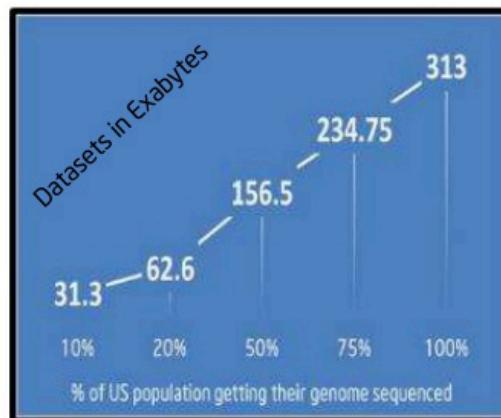
NGS as a big data problem

- How many full genomes may be sequenced with 30x coverage



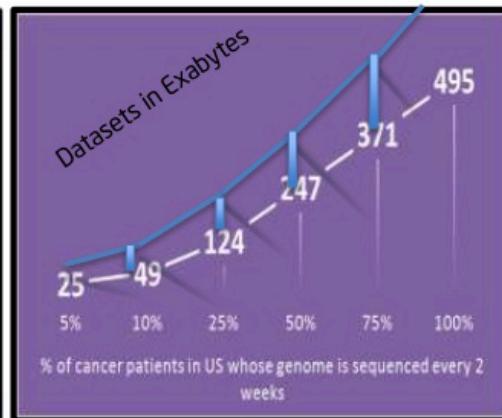
NGS as a big data problem

The day when every newborn gets their DNA sequenced is not far away: <http://www.nih.gov/news/health/sep2013/nhgri-04.htm>.



313 Exabytes

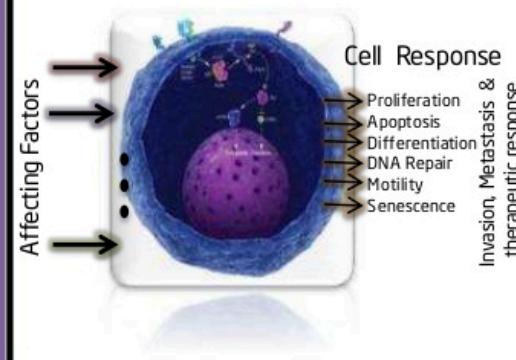
if everyone in the US has
their genes sequenced



495 Exabytes

if every cancer patient in the US has
their genes sequenced every 2 weeks.

**Images, Assays and Drug
response data will push it
further up as shown in Blue line**



Complex interaction of
varied & changing intrinsic
and extrinsic factors
determine cell response

With Genomic Data growing rapidly, hospitals and research centers need to access the local data (the ones not shared) and the centralized public/private data for various analysis and analytics for Genomic Research/Development/Medicine.

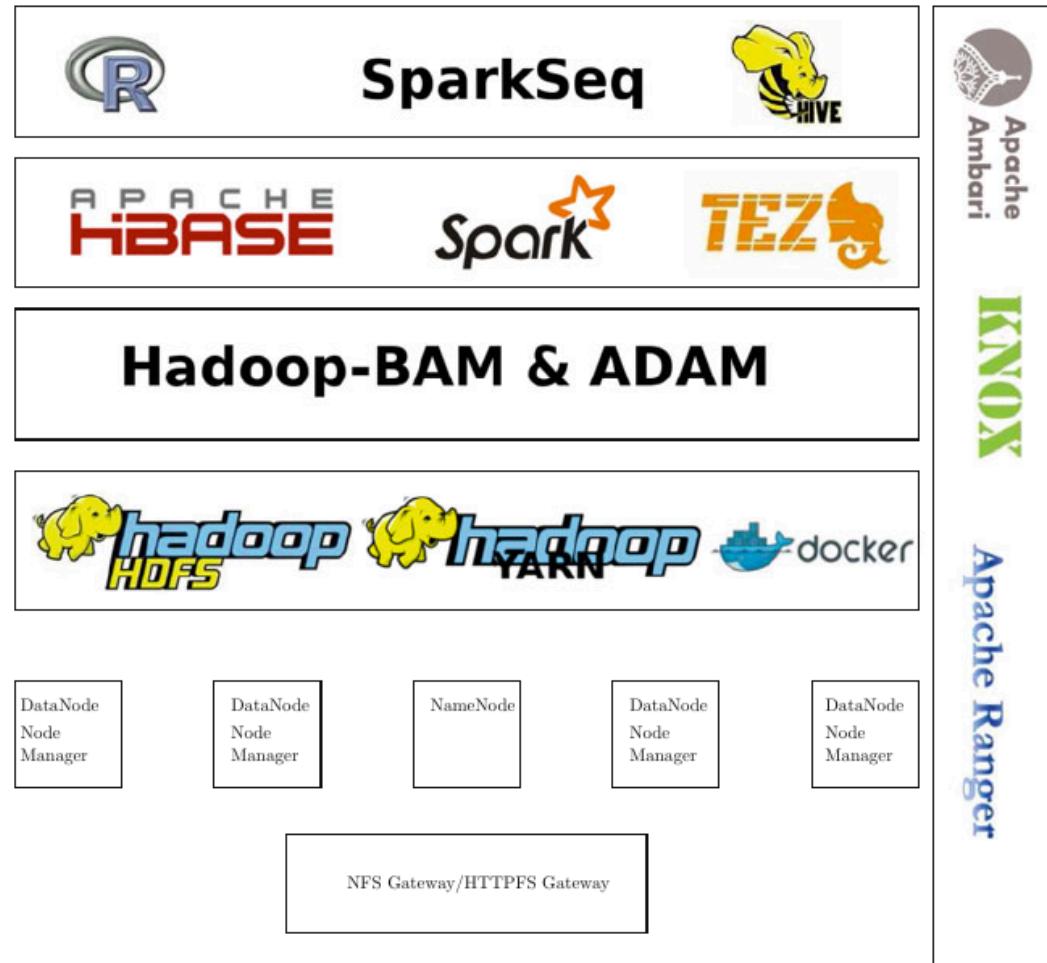
**Compute has to be done "where data is" and need to be consistent locally and in the cloud.
Energy, Total Cost of Operation are key**

Source: Knights Cancer Institute, Oregon Health Sciences University & Intel

NGS data analysis – clinical applications

- Personalized/precision medicine
- Examples:
 - Personalized cancer therapy
 - Other “systemic diseases”
 - Newborn screening
 - Emergency medicine
 - Personalized food and diet
 -

Big data architecture for NGS



Proof-of-concept paper

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 Messina², Alicja Pacholewska^{3,4}, Sergio Maffioletti²,
Piotr Gawrysiak¹ and Michał J. Okoniewski⁵

Big Data Genomics project and ADAM standard

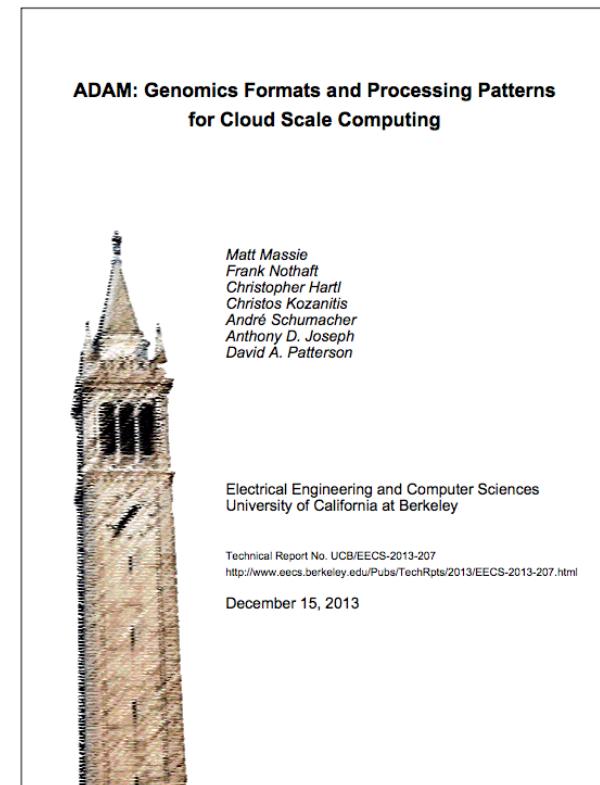
Collaboration of the AMPLab at UC Berkeley,
Genome Informatics Lab at UC Santa Cruz,
Icahn School of Medicine at Mount Sinai,
Microsoft Research, Cloudera, and the Broad
Institute.

<http://bdgenomics.org/>

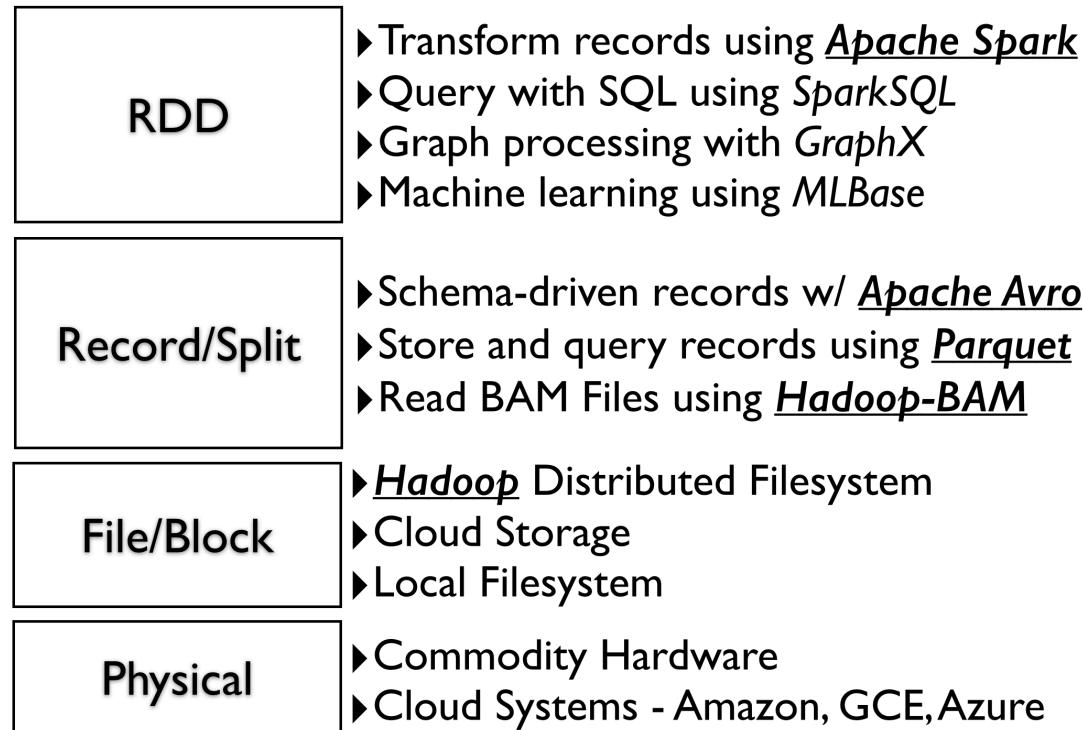
Matt Massie

<http://www.hammerlab.org/>

Jeff Hamerbacher



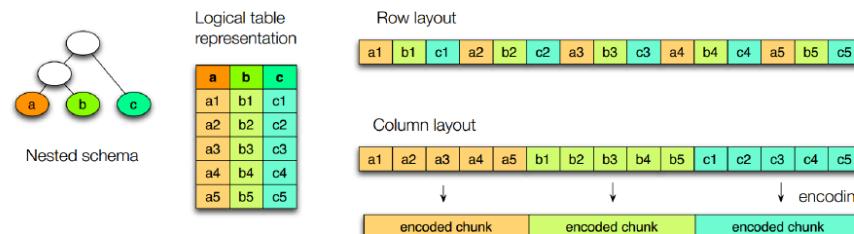
Big Data Genomics project and ADAM standard



ADAM: Avro + Parquet

```
{"namespace": "example.avro",
"type": "record",
"name": "User",
"fields": [
    {"name": "name", "type": "string"},
    {"name": "favorite_number", "type": ["int", "null"]},
    {"name": "favorite_color", "type": ["string", "null"]}
]
}
```

Columnar storage



<http://www.slideshare.net/cloudera/hadoop-summit-36479635>

Running ADAM: adam-shell

```

michalo - tmux - 192x70 [1670/5733]
16/05/27 10:46:03 INFO metastore.HiveMetaStore: Added admin role in metastore
16/05/27 10:46:03 INFO metastore.HiveMetaStore: Added public role in metastore
16/05/27 10:46:03 INFO metastore.HiveMetaStore: No user is added in admin role, since config is empty
16/05/27 10:46:04 INFO metastore.HiveMetaStore: 0: get_all_databases
16/05/27 10:46:04 INFO HiveMetaStore.audit: ugi=michalo ip=unknown-ip-addr    cmd=get_all_databases
16/05/27 10:46:04 INFO metastore.HiveMetaStore: 0: get_functions: db=default pat/*
16/05/27 10:46:04 INFO HiveMetaStore.audit: ugi=michalo ip=unknown-ip-addr    cmd=get_functions: db=default pat/*
16/05/27 10:46:04 INFO dataNucleus.Datastore: The class 'org.apache.hadoop.hive.metastore.model.MResourceURI' is tagged as "embedded-only" so does not have its own datastore table.
16/05/27 10:46:04 INFO session.SessionState: Created local directory: /tmp/aa792807-6010-4b15-97be-416de7e09c7_resources
16/05/27 10:46:04 INFO session.SessionState: Created local directory: /tmp/michalo/aa792807-6010-4b15-97be-416de7e09c7
16/05/27 10:46:04 INFO session.SessionState: Created local directory: /tmp/michalo/aa792807-6010-4b15-97be-416de7e09c7/_tmp_space.db
16/05/27 10:46:04 INFO repl.SparkLoop: Created sql context (with Hive support)...
SQL context available as sqlContext.

scala> sc
res0: org.apache.spark.SparkContext = org.apache.spark.SparkContext@c58c560

scala> import org.bdgenomics.adam.rdd.ADAMContext
import org.bdgenomics.adam.rdd.ADAMContext

scala> import org.bdgenomics.adam.projections.AlignmentRecordField, Projection
import org.bdgenomics.adam.projections.AlignmentRecordField, Projection

scala> val ac = new ADAMContext(sc)
ac: org.bdgenomics.adam.rdd.ADAMContext = org.bdgenomics.adam.rdd.ADAMContext@5cd3eaf

scala>

scala> val reads = ac.loadAlignments(
|   "hdfs://user/michalo/L085_13.adam",
|   projection = Some(
|     Projection(
|       AlignmentRecordField.sequence,
|       AlignmentRecordField.readMapped,
|       AlignmentRecordField.mapq
|     )
|   )
| )
16/05/27 10:46:48 INFO rdd.ADAMContext: Loading hdfs://user/michalo/L085_13.adam as Parquet of AlignmentRecords.
16/05/27 10:46:48 INFO rdd.ADAMContext: Reading the ADAM file at hdfs://user/michalo/L085_13.adam to create RDD
16/05/27 10:46:48 INFO rdd.ADAMContext: Using the specified projection schema
16/05/27 10:46:48 INFO storage.MemoryStore: Block broadcast_0 stored as values in memory (estimated size 262.2 KB, free 262.2 KB)
16/05/27 10:46:49 INFO storage.MemoryStore: Block broadcast_0_pieceed stored as bytes in memory (estimated size 263.3 KB, free 262.5 KB)
16/05/27 10:46:49 INFO storage.BlockManagerInfo: Added broadcast_0_pieceed in memory on 10.201.2.36:55184 (size: 26.3 KB, free: 511.0 MB)
16/05/27 10:46:49 INFO spark.SparkContext: Created broadcast 0 from newAPIHadoopFile at ADAMContext.scala:167
res0: org.bdgenomics.adam.rdd.AlignmentRecordRDD[0] = AlignedReadRDD(MapPartitionsRDD[1] at map at ADAMContext.scala:176, SequenceDictionary[1])
1->248956422, 0
1->133797422, 1
1->135986622, 2
1->133275389, 3
1->114364028, 4
1->107843718, 5
1->101991189, 6
1->98338345, 7
1->83257441, 8
1->68025619, 9
1->56317616, 10
2->42193529, 11
2->64444167, 12
21->46709983, 13
22->50816468, 14
3->198295559, 15
4->198214555, 16
5->181536259, 17
6->1788085979, 18
7->159345973, 19
8->145138636, 20
[2] 0:[tmux]*2

```

"michalo@aa9053:~/adam/" 12:03 27-May-16

adam-submit: ADAM command line tools

```
[michalo@a9078 bin]$ ./adam-submit --help
Using ADAM_MAIN=org.bdgenomics.adam.cli.ADAMMain
Using SPARK_SUBMIT=/cluster/apps/spark/spark-current/bin/spark-submit
Picked up _JAVA_OPTIONS: -XX:ParallelGCThreads=1
Picked up _JAVA_OPTIONS: -XX:ParallelGCThreads=1
16/05/25 15:04:00 INFO cli.cli.ADAMMain: ADAM invoked with args: "--help"

      e     888~-      e      e   e
      d8b    888  \      d8b      d8b  d8b
      /Y88b   888 |      /Y88b      d888bdY88b
      / Y88b   888 |      / Y88b      / Y88Y Y888b
      / Y88b   888 /      / Y88b      / YY  Y888b
      / Y88b   888_-~    / Y88b      /     Y888b

Usage: adam-submit [<spark-args> --] <adam-args>

Choose one of the following commands:

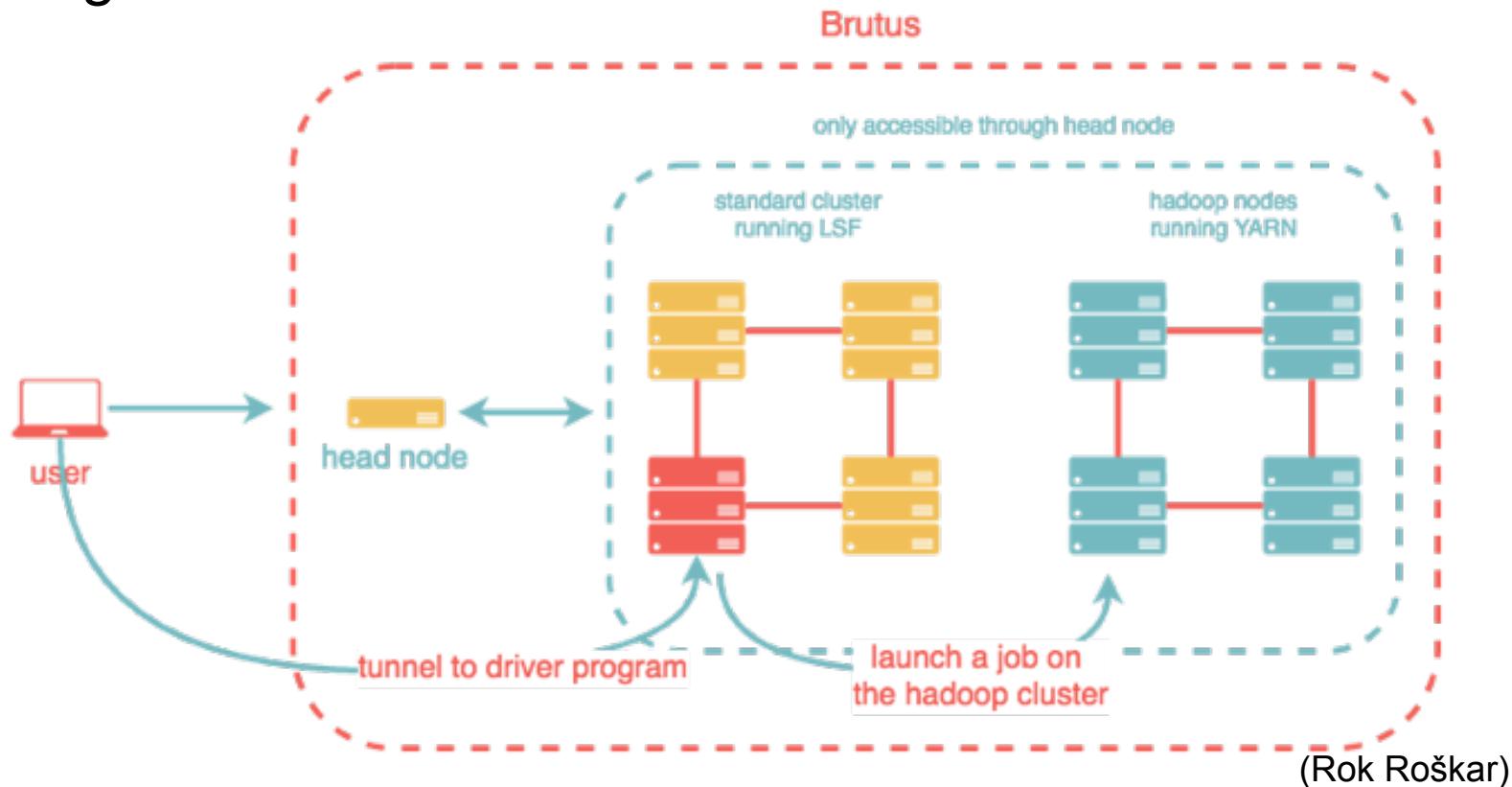
ADAM ACTIONS
  depth : Calculate the depth from a given ADAM file, at each variant in a VCF
  count_kmers : Counts the k-mers/q-mers from a read dataset,
  count_contig_kmers : Counts the k-mers/q-mers from a read dataset,
  transform : Convert SAM/BAM to ADAM format and optionally perform read pre-processing transformations
  adam2fastq : Convert BAM to FASTQ files
  plugin : Executes an ADAMPlugin
  flatten : Convert a ADAM format file to a version with a flattened schema, suitable for querying with tools like Impala

CONVERSION OPERATIONS
  vcf2adam : Convert a VCF file to the corresponding ADAM format
  adam2vcf : Convert an ADAM variant to the VCF ADAM format
  anno2adam : Convert a annotation file (in VCF format) to the corresponding ADAM format
  fasta2adam : Converts a text FASTA sequence file into an ADAMNucleotideContig Parquet file which represents assembled sequences.
  adam2fasta : Convert ADAM nucleotide contig fragments to FASTA files
  features2adam : Convert a file with sequence features into corresponding ADAM format
  wigfix2bed : Locally convert a wigFix file to BED format
  fragments2reads : Convert alignment records into fragment records.
  reads2fragments : Convert alignment records into fragment records.

PRINT
  print : Print an ADAM formatted file
  print_genes : Load a GTF file containing gene annotations and print the corresponding gene models
  flagstat : Print statistics on reads in an ADAM file (similar to samtools flagstat)
  print_tags : Prints the values and counts of all tags in a set of records
  listdict : Print the contents of an ADAM sequence dictionary
  allelecount : Calculate Allele frequencies
  view : View certain reads from an alignment-record file.
```

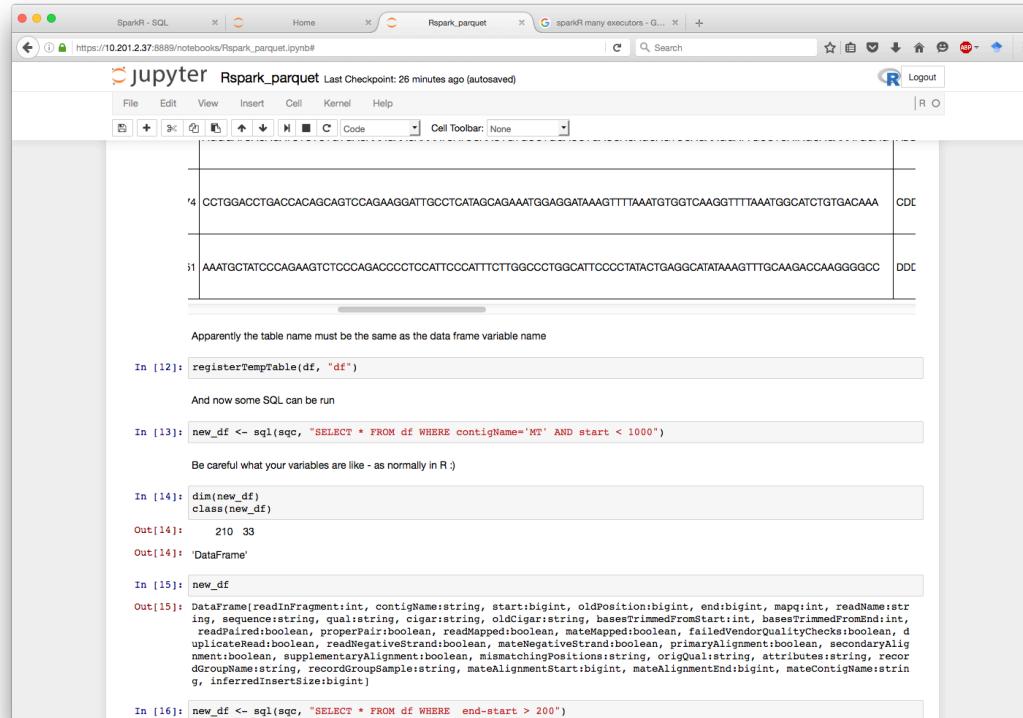
Hadoop cluster at ETH

- Fragment of the Brutus cluster



Using SparkR for secondary analyses

- Via Parquet files and SQL



The screenshot shows a Jupyter notebook interface running on a web browser. The title bar indicates it's a SparkR - SQL notebook. The main area displays two rows of DNA sequence data:

Contig	Sequence
14	CCTGGACCTGACCACAGCAGTCAGAAGGATTGCCTCATAGCAGAAATGGAGGATAAAGTTTAAATGGTGTCAAGGTTTAAATGGCATCTGTGACAAA
31	AAATGCTATCCCAGAAGTCTCCAGACCCCTCACTCCATTCTTGCCCCCTGGCATCCCCCTACTGAGGCATAAAAGTTGCAAAGCCAAGGGCC

Below the table, a message states: "Apparently the table name must be the same as the data frame variable name". The notebook then contains the following R code:

```
In [12]: registerTempTable(df, "df")
And now some SQL can be run
In [13]: new_df <- sql(sqcl, "SELECT * FROM df WHERE contigName='MT' AND start < 1000")
Be careful what your variables are like - as normally in R :)
In [14]: dim(new_df)
class(new_df)
Out[14]: 210 33
Out[14]: 'DataFrame'
In [15]: new_df
Out[15]: Dataframe[readFragment,int,contigName:string,start:int,oldPositon:bigint,endPosition:int,mateStart:bigint,rename:string,
isReadStart:Boolean,operaPair:Boolean,oldPartner:bigint,primaryVendor:Boolean,baseQualityCheck:Boolean,
readPaired:Boolean,readPairedOpéraPair:Boolean,readDapped:Boolean,mateMapped:Boolean,failedVendorQualityCheck:Boolean,
duplicatedRead:Boolean,readNegativeStrand:Boolean,mateNegativeStrand:Boolean,primaryAlignment:Boolean,secondaryAlignment:Boolean,
supplementaryAlignment:Boolean,mismatchingPositions:String,originalString:String,attributes:String,recordGroupSample:String,
mateAlignmentStart:bigint,mateAlignmentEnd:bigint,mateContigName:String,inferredInsertSize:bigint]
In [16]: new_df <- sql(sqcl, "SELECT * FROM df WHERE end-start > 200")
```

The same thing in scala

```
val df = sqlContext.read.parquet("hdfs://user/michalo/cc5Ago2KO_2MAaligned.out.sorted.adam")
df.registerTempTable("df")
df.printSchema
val myout = sqlContext.sql("SELECT count(*) FROM df WHERE `end` - `start` > 90")
myout.collect
```

Alternative to SparkR: Rstudio sparklyr

- `plyr => dplyr => sparklyr`
- Connect to Spark from R — like dplyr backend
- Filter and aggregate Spark datasets then bring them into R for analysis and visualization.
- distributed machine learning from R with Spark MLlib or H2O Sparkling Water
- Create extensions that call the full Spark API and provide interfaces to Spark packages

Info release yesterday....

The screenshot shows a web browser window with the following details:

- Title Bar:** RStudio Preview – RStudio
- Address Bar:** https://www.rstudio.com/products/rstudio/download/preview/
- Bookmarks Bar:** Apps, paper | ZSI Bio Slack, scikit-learn: machin..., SparkR @ ZGM, Schema Document..., rraureservation ETH, Other Bookmarks
- Header:** Michal, R Studio, Products, Resources, Pricing, About Us, Blog, Search icon
- Content:**
 - Section:** RStudio v1.0.39 Preview — Release Notes
 - Text:** A preview release of RStudio v1.0.39 is now available for testing and [feedback](#). Highlights include:
 - Authoring tools for [R Notebooks](#).
 - Integrated support for the [sparklyr](#) package (R interface to Spark).
 - Enhanced data import tools based on the [readr](#), [readxl](#) and [haven](#) packages.
 - Performance profiling via integration with the [profvis](#) package.
 - Authoring tools for R Markdown [websites](#) and the [bookdown](#) package.
 - Many other miscellaneous enhancements and bug fixes.
 - Text:** See v1.0.39 [Release Notes](#) for full details on all of the changes in this release.
 - Section:** Desktop Version
 - Table:** A table showing download links for various operating systems and architectures. The columns are: Installers, Size, Date, and MD5.
 - Section:** Zip/Tarballs

Summary

- Genomics started generating real big data
- Mechanisms for storing and processing are being developed
- Connecting to “good-old-memory-limited” R is needed
- With SparkR it is possible
 - E.g. to query parquet files on R level and get manageable data frames
- RStudio has just mimicked the recipe described here
- It is not only for genomic use, but many big data cases

Acknowledgements

- Rok Roškar – setting up the Spark and jupyter configuration
- HPC group, Scientific IT Services ETH
- Tomasz Gambin, Marek Wiewiorka– Politechnika Warszawska, Instytut Informatyki

**Thank you for your attention!
Questions/comments?**

