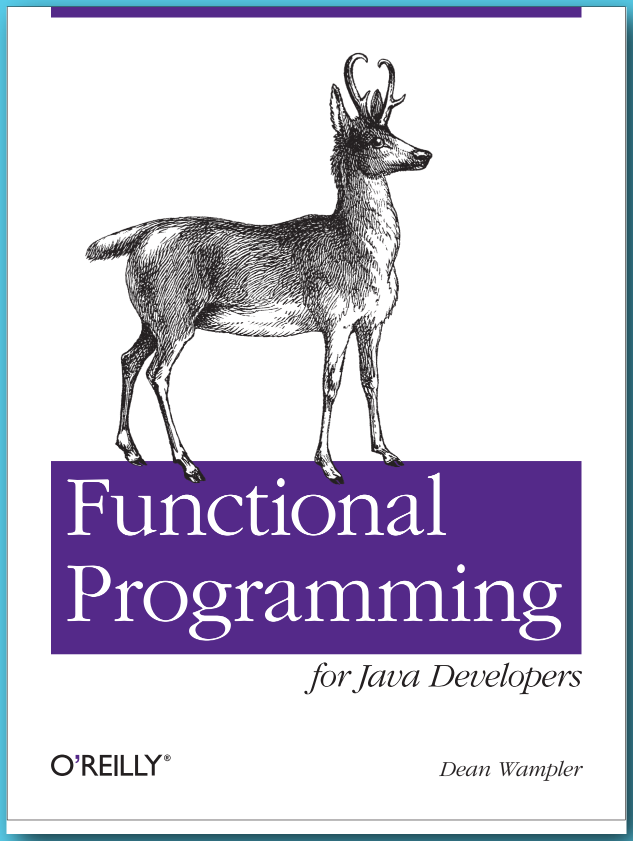
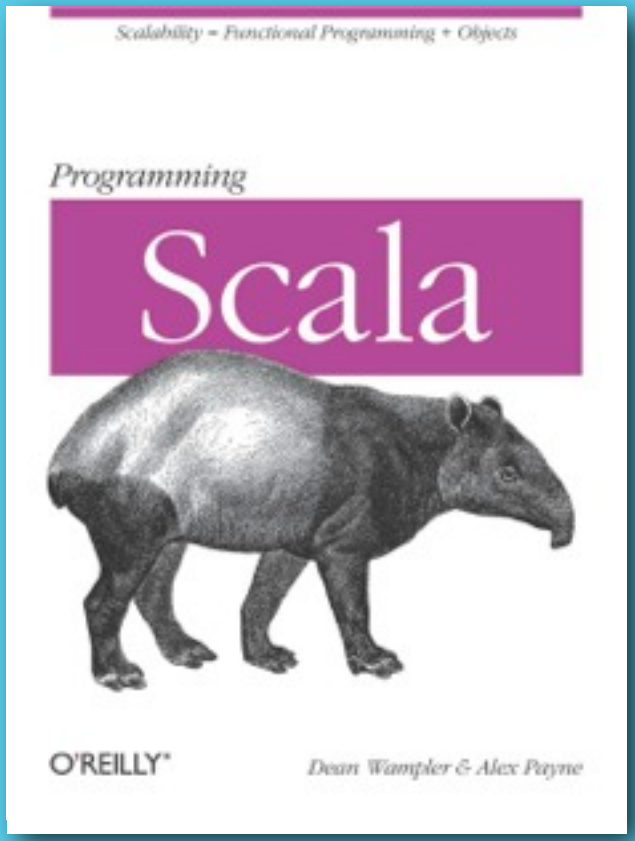
H2O: An open-source, in-memory prediction engine for data science

Typesafe

Some content is copyrighted from other sources, as noted.

Dean Wampler



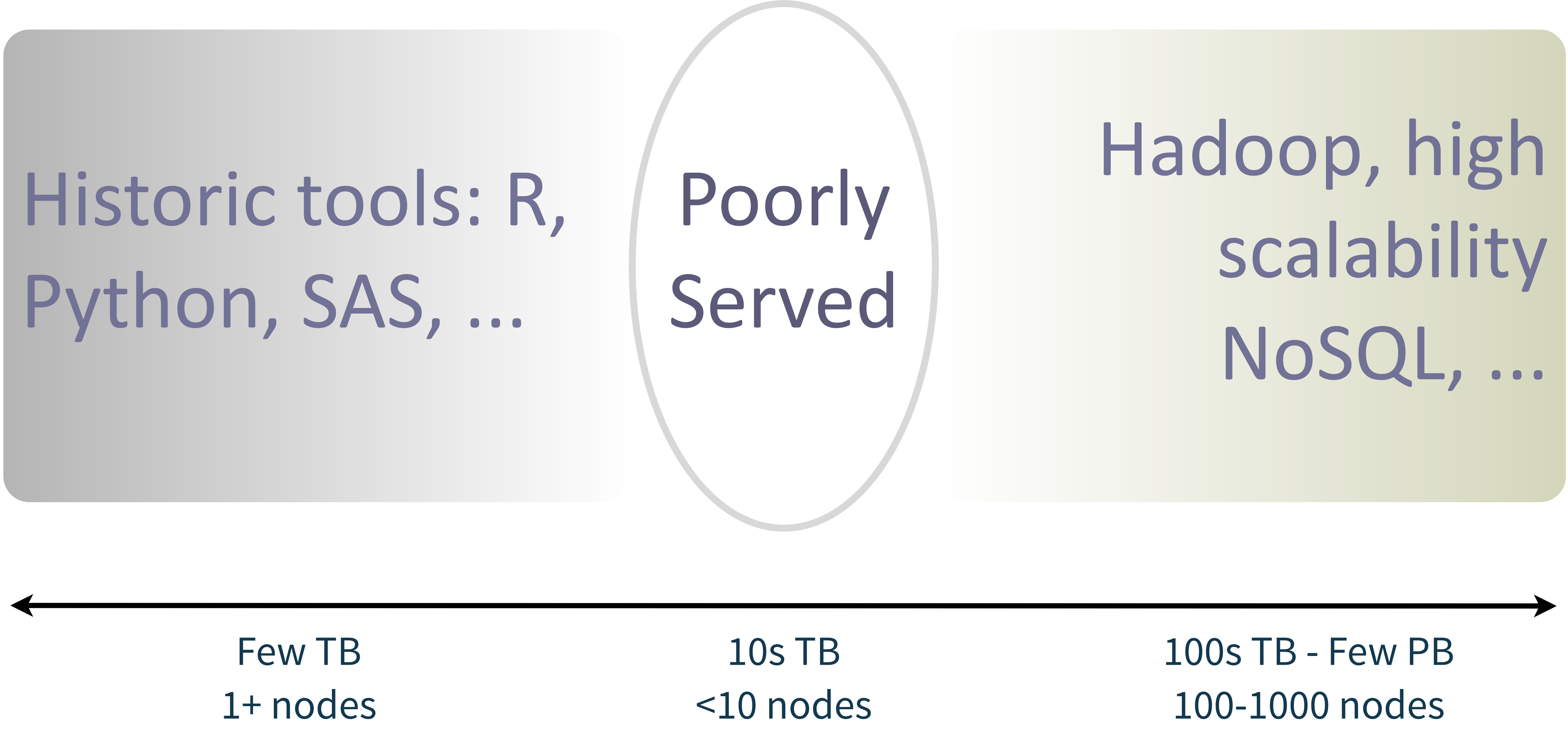

Scalability = Functional Programming + Objects
Programming
Scala
O'REILLY
Dean Wampler & Alex Payne

dean.wampler@typesafe.com
polyglotprogramming.com/talks
@deanwampler

About me. You can find this presentation and others on Big Data and Scala at polyglotprogramming.com.

H2O:
Why??

Spectrum of data set sizes, computation loads:



Historic tools: R, Python, SAS, ...

Poorly Served

Hadoop, high scalability NoSQL, ...

Few TB
1+ nodes

10s TB
<10 nodes

100s TB - Few PB
100-1000 nodes

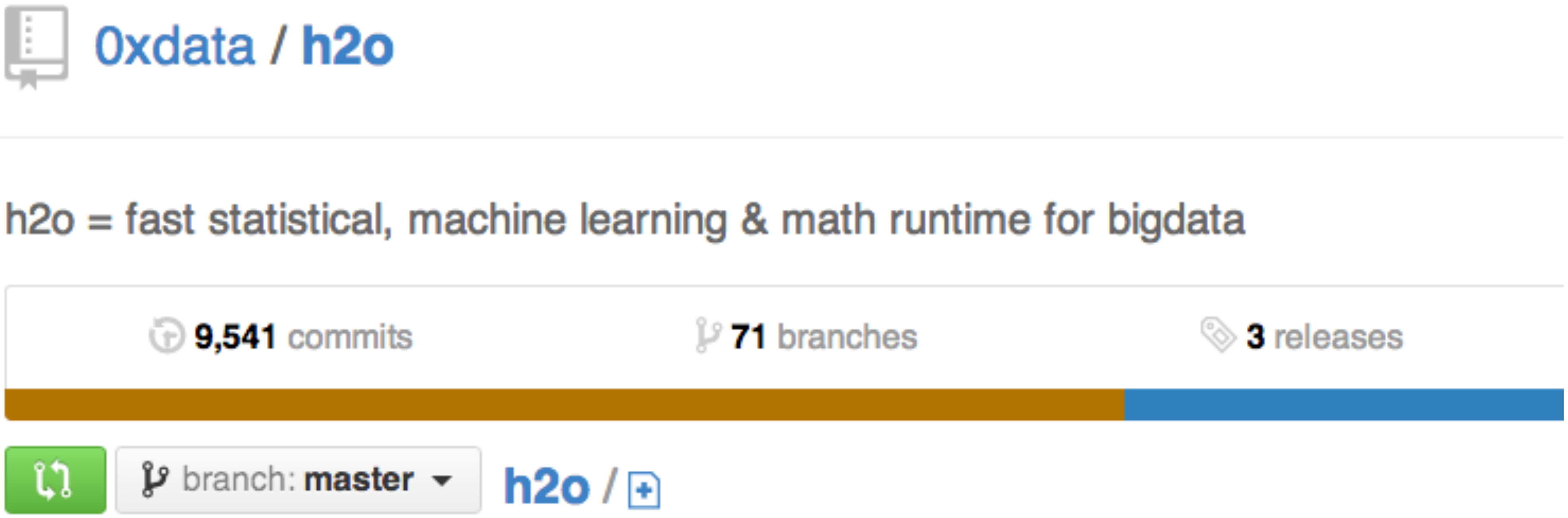
Tuesday, April 1, 14

H2O is a good fit for this middle, poorly-served area of data set sizes and distributed computation requirements.

H2O: An open-source, in-memory prediction engine for data science

H2O:
An open-source,
in-memory
prediction engine
for data science

https://github.com/0xdata/h2o



0xdata / h2o

h2o = fast statistical, machine learning & math runtime for bigdata

9,541 commits | 71 branches | 3 releases

Developed by 0xdata ("hexdata")

http://0xdata.com

H2O:
An open-source,
in-memory
prediction engine
for data science

H2O:

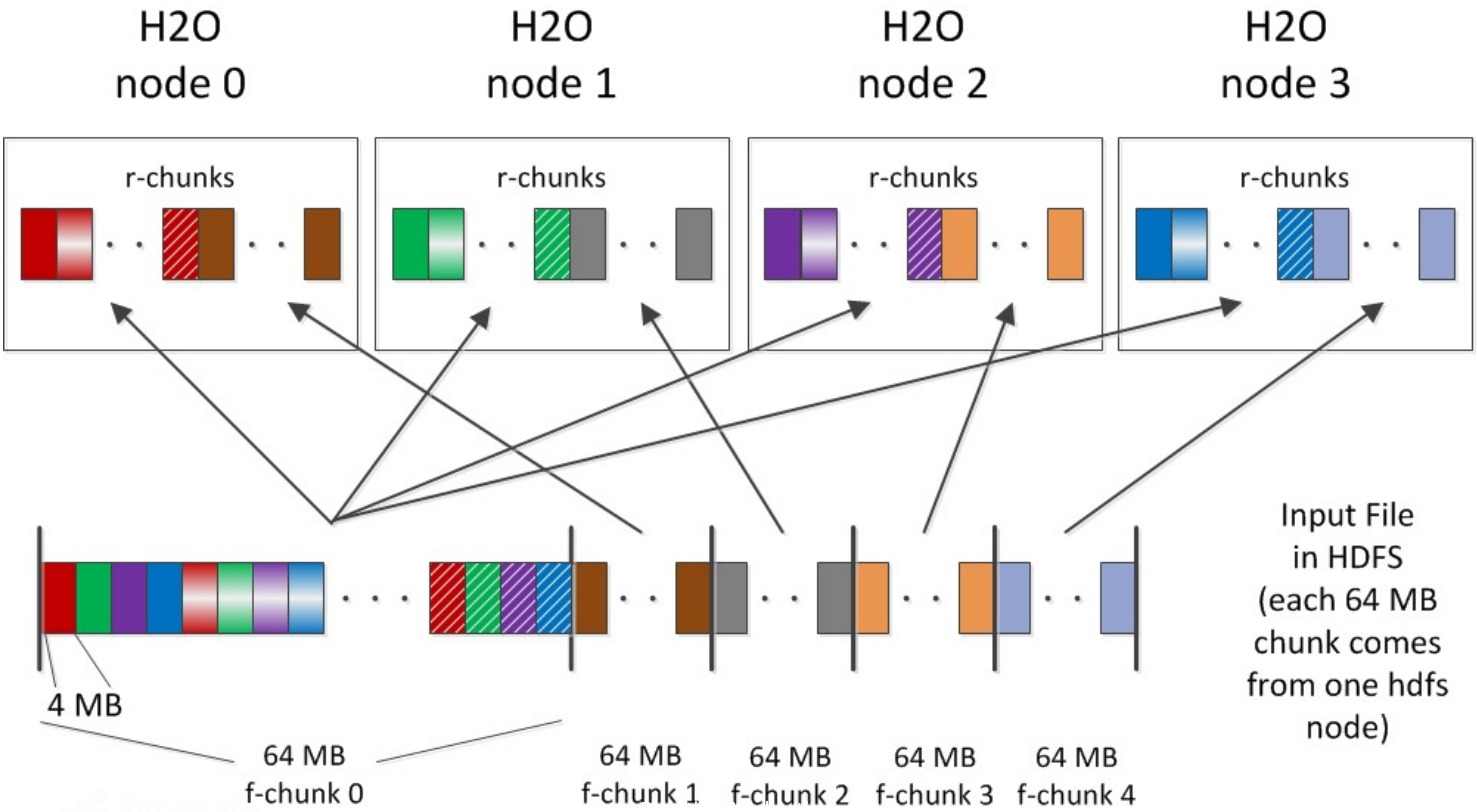
An open-source,

in-memory

prediction engine

for data science

Raw (Pre-Parse) Data Ingestion Pattern



H2O node 0

H2O node 1

H2O node 2

H2O node 3

r-chunks

r-chunks

r-chunks

r-chunks

4 MB

64 MB
f-chunk 0

64 MB
f-chunk 1

64 MB
f-chunk 2

64 MB
f-chunk 3

64 MB
f-chunk 4

Input File
in HDFS
(each 64 MB
chunk comes
from one hdfs
node)

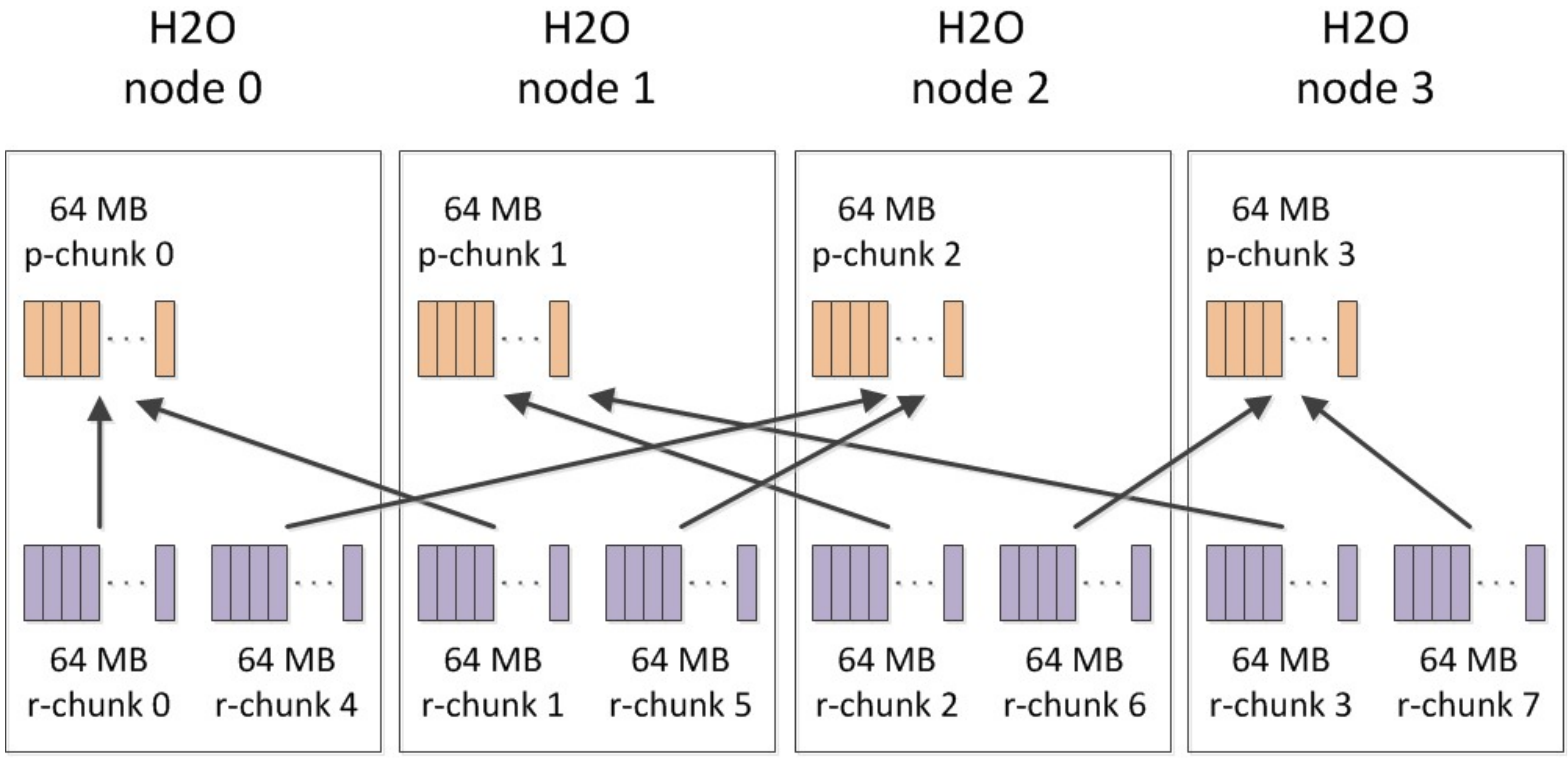
source: github.com/0xdata/docs/pictures/DataIngestion.png

In this example, HDFS files are read, where each HDFS block (called a “chunk” here and it might be a multiple of 64MB, depending on the cluster configuration). Each block is broken into 4MB raw chunks and a hash of the data is used to distribute these pieces uniformly around the H2O cluster (4 nodes, in this diagram).

Currently, if the H2O nodes are also running in the cluster, no attempt is made to colocate HDFS blocks on the same servers, so the full data set will likely be copied over the cluster’s network as it is ingested.

Source: The H2O github repo.

Parse Data Motion Pattern



(Note: all r-chunks and p-chunks live in Java heap memory)

source: github.com/0xdata/docs/pictures/Parse.png

Tuesday, April 1, 14

The parse step converts the in-memory raw data, such as CSV records, into the internal HEX format (key-value structure). This step is required before you can run algorithms on the data. If the raw data was plain text, the resulting HEX data will be smaller, but if it started out compressed, then the sizes will be roughly the same.

Source: The H2O github repo.

In memory

- A Key/Value Store: ~150nsec per get or put

When data doesn't fit in memory...

- It is spilled to disk as needed.

The in-memory storage is a distributed key-value store with spillage to disk.

Input file formats supported

- CSV and Gzip-compressed CSV
- MS Excel (XLS)
- ARRF (See http://weka.wikispaces.com/ARFF)
- Hive file format (Hadoop)
  - Also understands Hive files partitioned over a directory tree.
- NoSQL adapters & SQL/JDBC forthcoming
- Others...

H2O:

An open-source,
in-memory
prediction engine
for data science

H2O:

An open-source,
in-memory
prediction engine
for data science

Emphasis on Mathematics, esp. Statistics & Linear Alg.

- Linear algebra (Matrices), for example:
  - Dimensional reduction
  - Singular Value Decomposition
- Sampling
- Statistical distributions (Gaussian, Binomial, etc.)
- Efficient handling of sparse and asymmetric data sets
  - For outlier detection problems, like fraud detection
- Support for *streaming* applications.

Many ML algos require linear algebra support. Many are probabilistic, so stats. and probabilistic algos. required.
From the H2O document: 0xdata_H2O_Algorithms.pdf
The items in this and the next few slides are part of a roadmap and some may not be implemented yet.

Emphasis on Machine Learning Algorithms

- Classification
  - Distributed Random Forest and trees
  - GBM (Gradient Boosting Machines)

From the H2O document: 0xdata_H2O_Algorithms.pdf

Emphasis on Machine Learning Algorithms

- Regressions
  - GLM/GLMnet (Generalized Linear Models/R library)
  - Bayesian and Multinomial Regression
  - Parallel grid search on the parameter space of the regression method

From the H2O document: 0xdata_H2O_Algorithms.pdf

The last bullet means they can run the algos. in parallel, such as sampling, iterating to a solution for iterative algos, etc.

Emphasis on Machine Learning Algorithms

- Recommendation
  - Collaborative Filtering
  - Alternating Least Squares

Tuesday, April 1, 14

From the H2O document: 0xdata_H2O_Algorithms.pdf

The last bullet means they can run the algos. in parallel, such as sampling, iterating to a solution for iterative algos, etc.

Emphasis on Machine Learning Algorithms

- Neural Networks
  - Multi-layer Perceptron
  - Auto-encoder
  - Restricted Boltzmann Machines

From the H2O document: 0xdata_H2O_Algorithms.pdf

Emphasis on Machine Learning Algorithms

- Solvers and Optimization
  - Generalized ADMM Solver
  - L-BFGS (Quasi-Newton’s Method)
  - Least Squares
  - Stochastic Gradient Descent
  - Markov Chain Monte Carlo

From the H2O document: 0xdata_H2O_Algorithms.pdf

Emphasis on Machine Learning Algorithms

- Clustering
  - K-Means
  - K-Nearest Neighbors
  - Locality Sensitive Hashing

Tuesday, April 1, 14

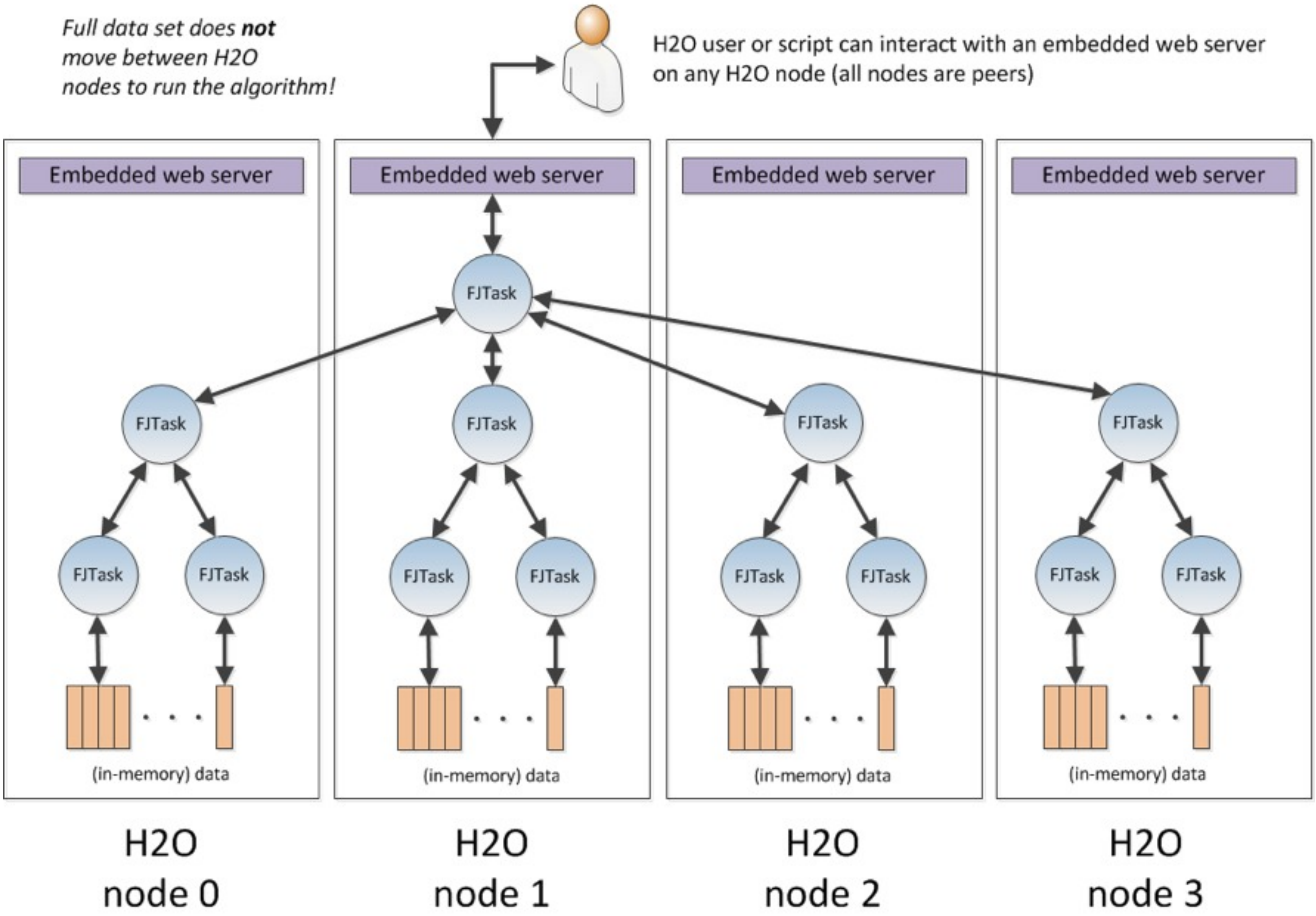
From the H2O document: 0xdata_H2O_Algorithms.pdf

Status of Algorithms

- H2O is relatively new. Its libraries are not as mature as those for R, Python, etc.
- However,
  - More algorithms are being added quickly.
  - H2O already provides very fast performance.

See for example: h2o/docs/RandomForestProductGaps.doc for a candid list of what they perceive is still needed.

GLM Algorithm Data Access Pattern



*Full data set does **not** move between H2O nodes to run the algorithm!*

H2O user or script can interact with an embedded web server on any H2O node (all nodes are peers)

source: github.com/0xdata/docs/pictures/GLMAlgoMem.png

Tuesday, April 1, 14

We already parsed the data into HEX format, now here is how an algorithm is run schematically. In this case, the user goes to the embedded webserver for one of the nodes (they're all equal), and starts a job. Lightweight Fork/Join tasks (a JVM concurrency primitive) are started to divide and conquer the data. There is no bulk movement of data necessary in this computation and because the data is in-memory, the results are computed very quickly.

Source: The H2O github repo.

Engine

- Distributed Fork/Join + Map/Reduce + Key/Value storage

H2O Clusters

H2O clusters

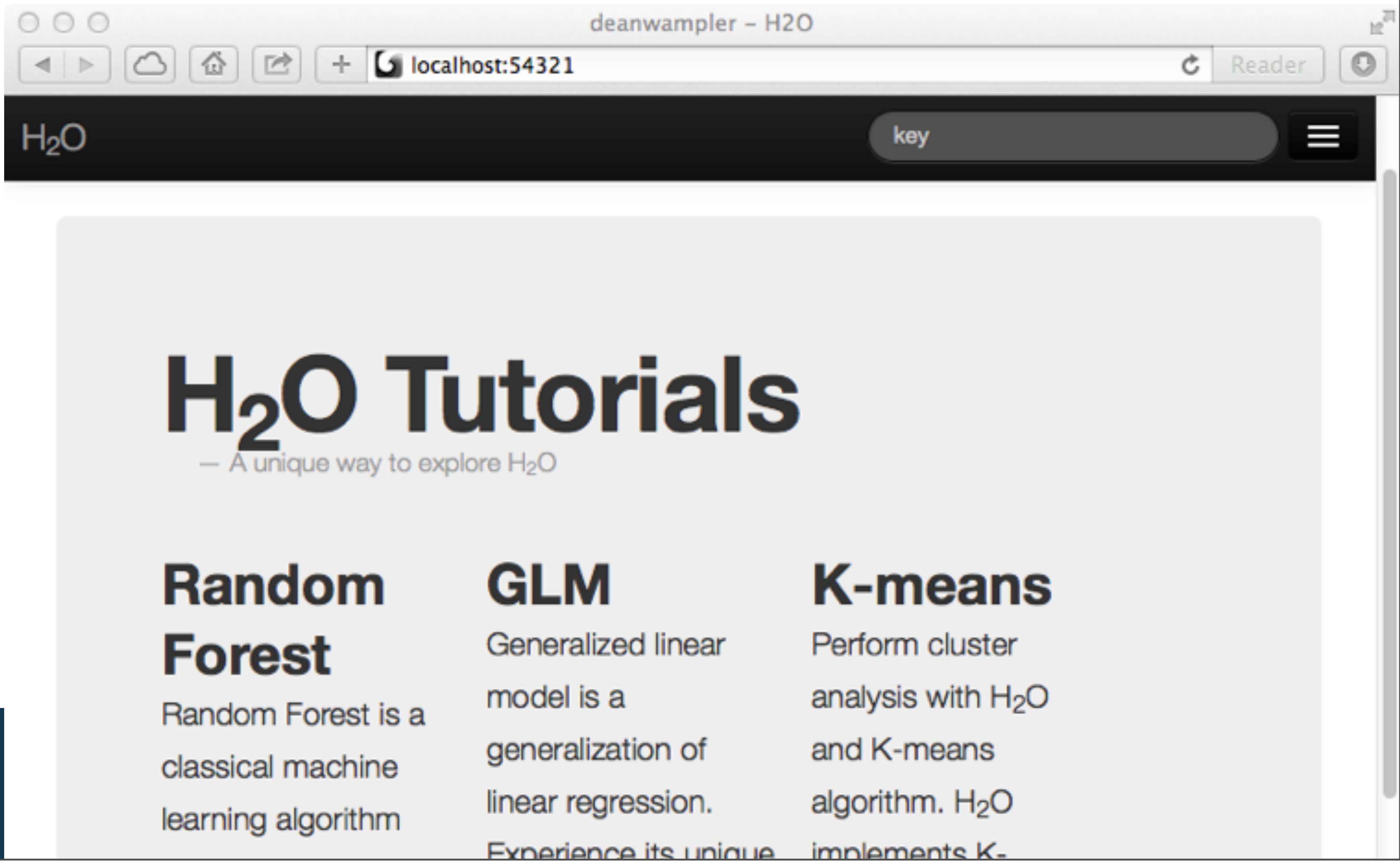
- Nodes discover each other at startup
  - Not a master/slave configuration.
- Set of nodes is fixed for the life of the cluster instance.
  - As of the moment the first *work item* is received.
- If a node fails, the cluster is dissolved.
  - As an in-memory system, it can't continue if part of the memory is "gone".
- But restart is fast; it's different than Hadoop.

H2O is a much lighterweight system. It will "fail fast" if a node goes done. Usually, it's relatively cheap to restart the cluster and the "work item".

Working with H2O

Run standalone. Use the Web UI

```
$ java -Xmx1g -jar target/h2o.jar
$ open http://localhost:54321
```



H_2O Tutorials

— A unique way to explore H_2O

Random Forest

Random Forest is a classical machine learning algorithm

GLM

Generalized linear model is a generalization of linear regression. Experience its unique

K-means

Perform cluster analysis with H_2O and K-means algorithm. H_2O implements K-

Tuesday, April 1, 14

Each node has an embedded web server; you can talk to any of them.

JSON/REST

- REST-API requests and JSON responses allows connecting via
  - Browser, curl/wget, programming REST libs.
  - MS Excel.
  - Integrated R environment for Data Analysis. R syntax is the default for statistical functions.

H2O.R

- An R module used in your friendly R environment.
- Uses the REST interface to communicate with a running H2O cluster.
- In Github repo, see R/README.txt for details.

Python

- Similar to the R support.
- Very poorly documented.
- This link seems to be a useful place to start:
  - https://github.com/0xdata/h2o/wiki/How-To-Run-Tests
- Also, look at the Github repo, “py” directory.

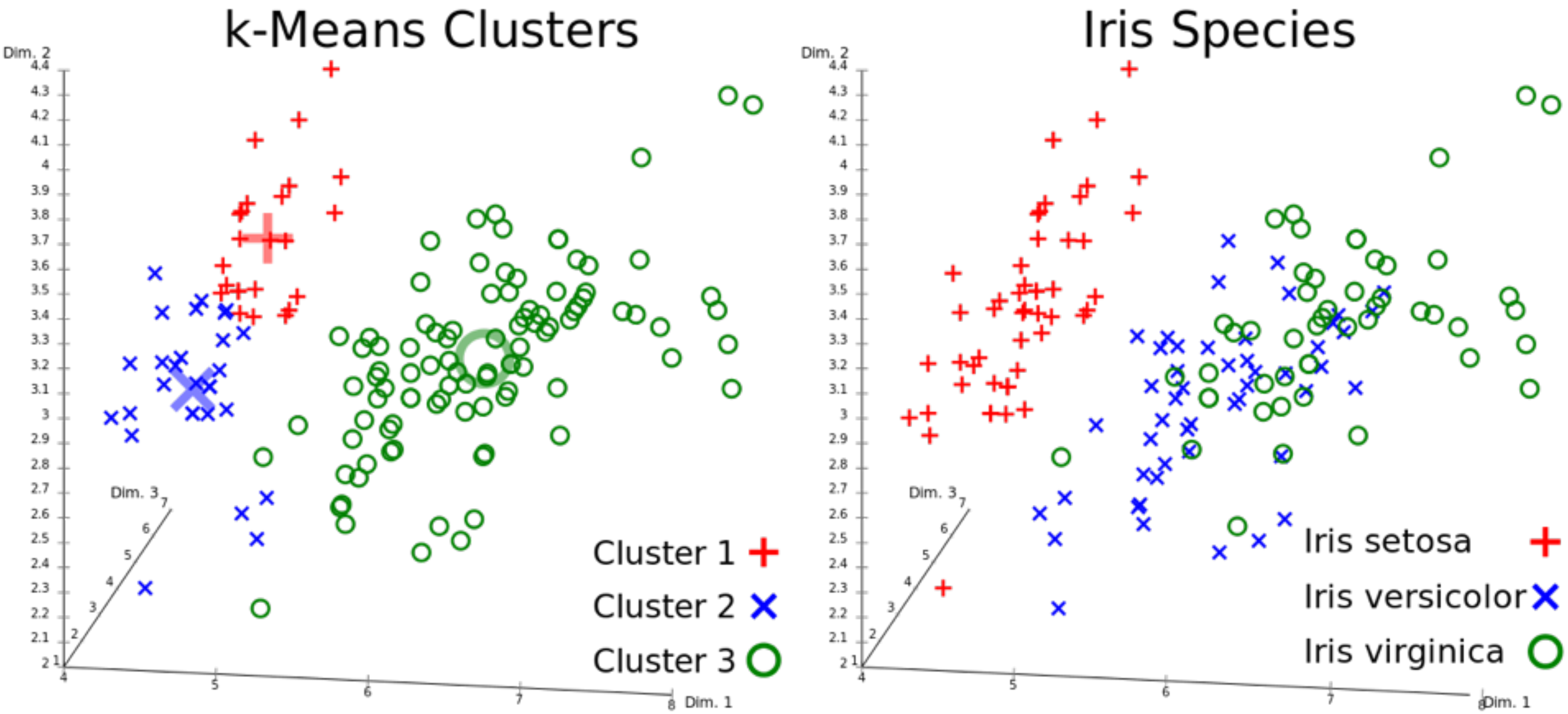
Java

- docs.0xdata.com/developuser/top_developer.html
- Can load H2O in Eclipse or IntelliJ IDEA.
- `h2o-samples` directory contains Java samples.

Demo of `MapReduceKMeans`

If you're reading the slides offline, the demo is a code walkthrough of h2o-samples/src/main/java/samples/MapReduceKMeans.java in the Github repo.

K-Means Clustering



en.wikipedia.org/wiki/File:Iris_Flowers_Clustering_kMeans.svg

K-Means Clustering

- In the `h2o-samples` directory.
- Run this sample within Eclipse.

Demo of `MapReduceKMeans`

If you're reading the slides offline, the demo is a code walkthrough of h2o-samples/src/main/java/samples/MapReduceKMeans.java in the Github repo.

Notes on Running `MapReduceKMeans`

- docs.0xdata.com/developuser/quickstart_eclipse.html has some errors. Rather than running the app, `Part05_KMeansNewAPI`, do the following:
  - Make sure that `h2o-samples/src/main/java` is a project source folder.
  - Edit `MapReduceKMeans.java` and change the path for `Key file` from `../lib/...` to `lib/....`
  - You have to stop the job with the red “kill” button.

Inside baseball on what I had to do to run the example in Eclipse, contrary to the documentation.

Hierarchy of Data Objects

- Frame – a collection of Vecs
  - Vec – a collection of Chunks
    - Chunk – a collection of 10^3 to 10^6 elems
      - elem – a Java double
- Row i – i^{th} elements of all the Vecs in a Frame

Example 2: Neural Network on MNIST dataset

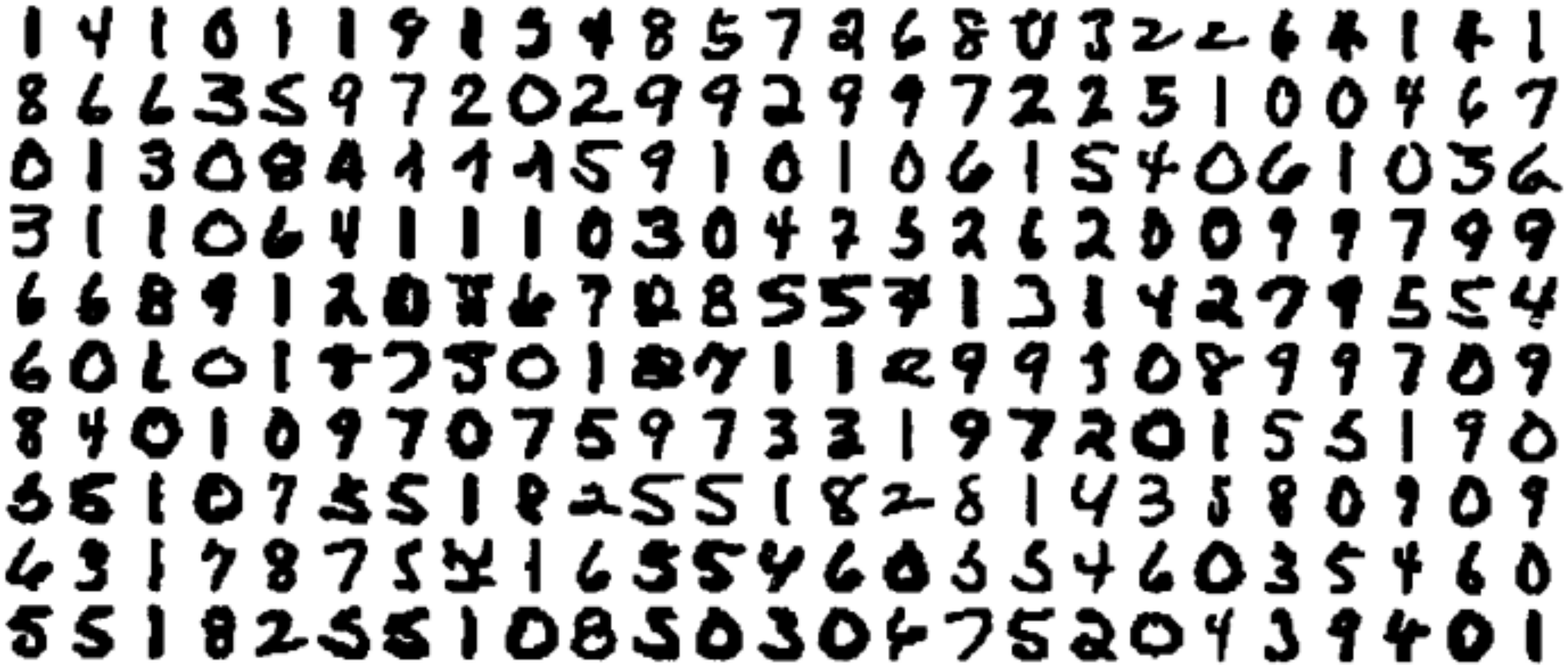


Figure 2: Examples of normalized digits from the testing set.

From *Handwritten zip code recognition with multilayer networks*, Y. LeCun, et al.,
http://yann.lecun.com/exdb/publis/pdf/lecun-90e.pdf

Tuesday, April 1, 14

If you’re reading the slides offline, the demo is a code walkthrough of h2o-samples/src/main/java/samples/NeuralNetMnist in the Github repo.
This one won’t run unless you add a JVM argument -Xmx4G. (A number less than 4, like 1 or 2 might work, but if you have the RAM...).

Example 2: Neural Network on MNIST dataset

- http://yann.lecun.com/exdb/mnist/
- Famous dataset of hand-written digits used to develop zip-code recognition software.
- I used `-Xmx4G` heap setting. (More typical of H2O apps.)

Demo of `NeuralNetMnist`

If you're reading the slides offline, the demo is a code walkthrough of h2o-samples/src/main/java/samples/NeuralNetMnist in the Github repo.
This one won't run unless you add a JVM argument -Xmx4G. (A number less than 4, like 1 or 2 might work, but if you have the RAM...).

Scala

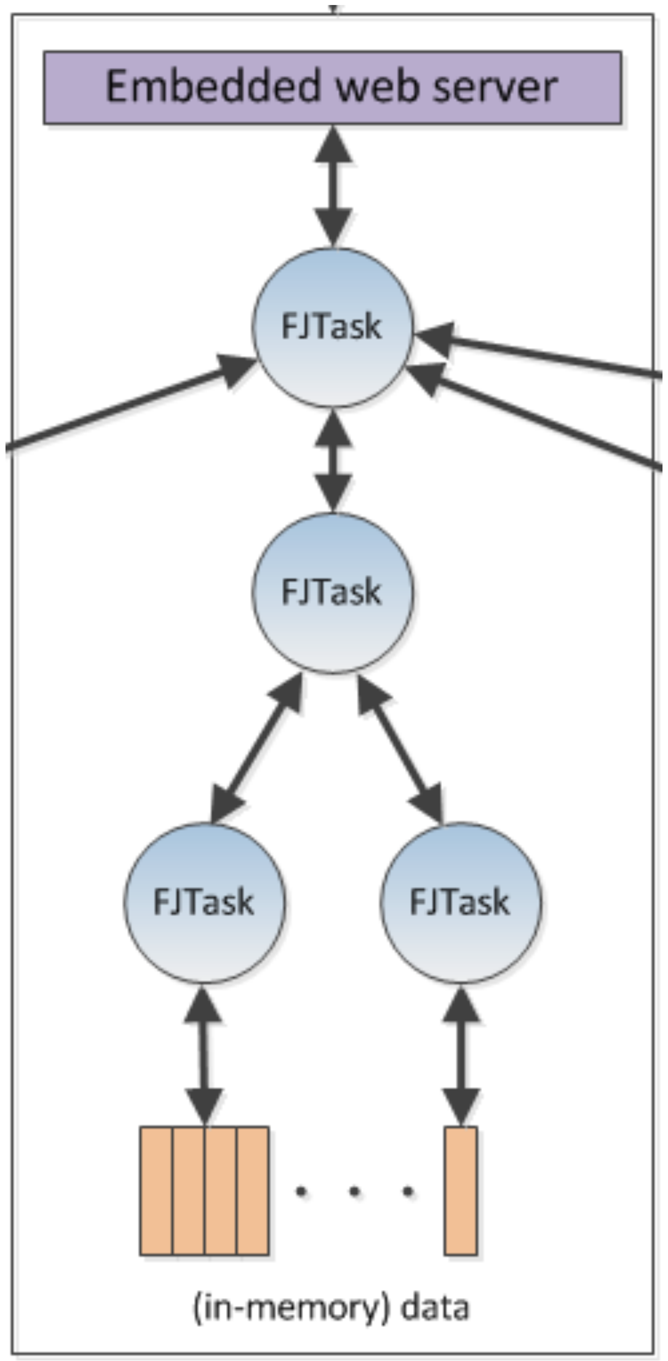
- docs.0xdata.com/developuser/quickstart_scala.html
- *shalala* shell - modified Scala REPL shell for interacting with H2O cluster.
- Scala “DSL” in subproject `h2o-scala`.

The Scala API is a work in progress, but other data-centric Scala APIs like Scalding and Spark have proven very popular because they are concise, yet powerful.

H2O Algorithms

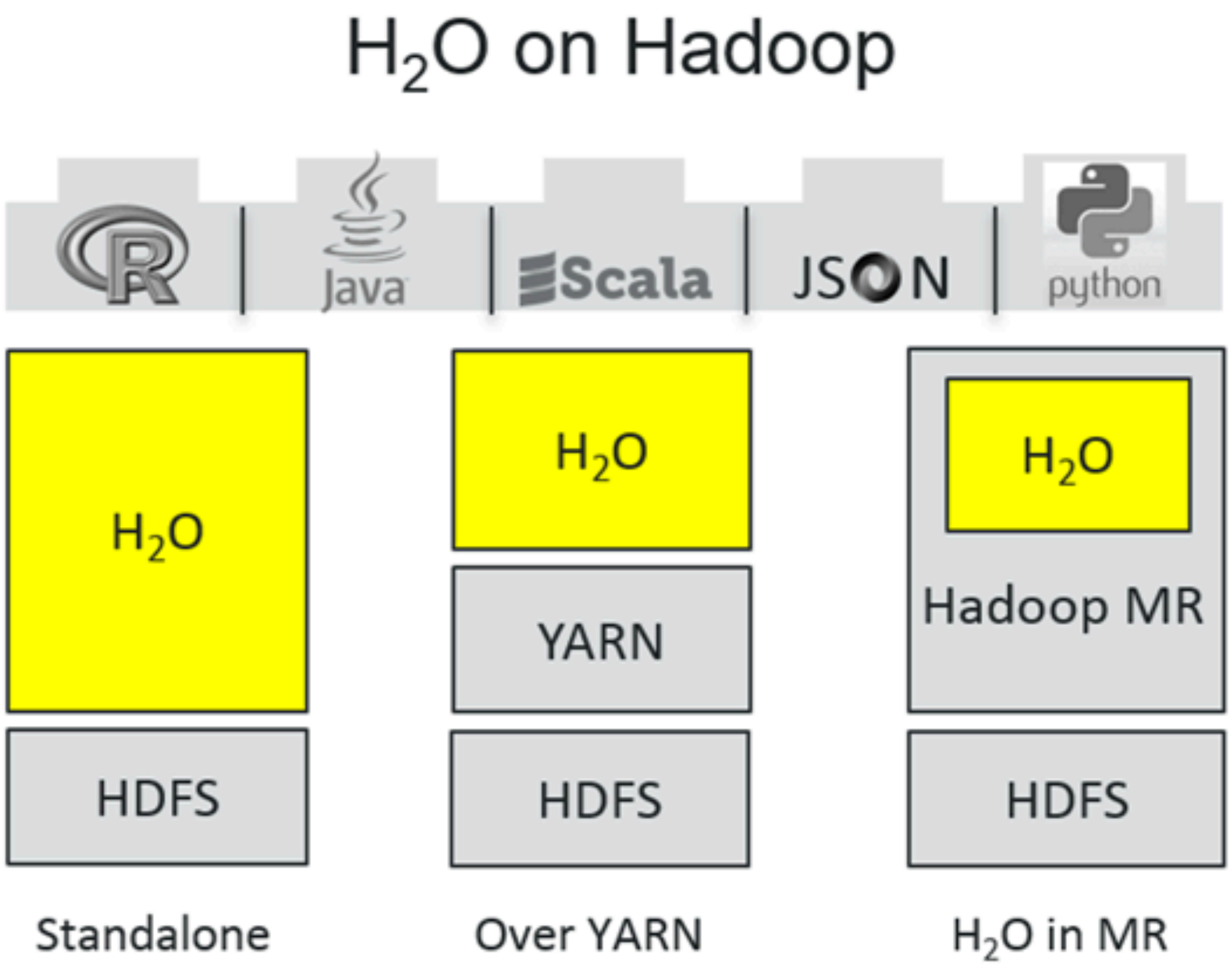
Algorithms

- Focus on Statistics and Machine Learning algorithms.
- Implemented on top of H2O's own versions of *map* and *reduce* primitives.
  - Not related to Hadoop's MapReduce.
- Work is decomposed using Java's Fork/Join framework into smaller units of work, one per 4MB data chunk.



Runs on Hadoop
or Standalone

Running on Hadoop



You have three options.

1. Run H2O by itself and talk to HDFS.
2. Run it on top of YARN.
3. Run it embedded as a MapReduce job

Running on Hadoop

- H2O nodes can be run as Java processes on “slave” nodes.
  - I.e., don’t run on the master nodes.
- For interactive use, a long-running H2O job runs as MapReduce *map* tasks.
  - Internally, H2O will do its own versions of *map* and *reduce* processing.
- For batch jobs, you submit a Hadoop job that builds an internal H2O set of nodes for the life of the job.

Long-running job for interactive use

```
$ cd $H2O_HOME/hadoop
$ hadoop jar h2odriver_cdh4.jar \
    water.hadoop.h2odriver \
    [-jt <jobtracker:port>] \
    -libjars ../h2o.jar \
    -mapperXmx 1g \
    -nodes 5 \
    -output hdfsOutputDirName
```

If you run this command on a node in the hadoop cluster, you usually don't need to specify the -jt jobtracker:port argument. If you clone the H2O repo, you'll need to cd to the "hadoop" directory and run "make" to create the Hadoop jars targeted for several platforms. Here I'm using the jar targeted for CDH4. This process will run inside MapReduce mapper tasks, so we give them plenty of memory (1GB). The nodes is the number of H2O nodes, which will usually be less than the number of Hadoop cluster nodes. Finally, we specify where to write output as we run H2O jobs from within this running MapReduce job!

To Learn More…

- Documentation: docs.0xdata.com/index.html
- Teh GitHubs: github.com/Oxdata/h2o
- Cliff Click's presentation at CodeMesh:
  - infoq.com/presentations/api-memory-analytics
- 0xdata.com

If you run this command on a node in the hadoop cluster, you usually don't need to specify the -jt jobtracker:port argument. If you clone the H2O repo, you'll need to cd to the "hadoop" directory and run "make" to create the Hadoop jars targeted for several platforms. Here I'm using the jar targeted for CDH4. This process will run inside MapReduce mapper tasks, so we give them plenty of memory (1GB). The nodes is the number of H2O nodes, which will usually be less than the number of Hadoop cluster nodes. Finally, we specify where to write output as we run H2O jobs from within this running MapReduce job!

Typesafe

dean.wampler@typesafe.com
polyglotprogramming.com/talks
@deanwampler