Using R with Hadoop

by Jeffrey Breen
Principal, Think Big Academy
Think Big Analytics
jeffrey.breen@thinkbiganalytics.com
Twitter: @JeffreyBreen
**Envision**
- **Think Big**
  - Priorities
  - Goal Architecture
  - Business Case
- **Capability Roadmap**
  - Proof of Concept

**Educate**
- **Start Smart**
  - Market Opportunities
  - Case Studies
  - Solutions
- **Products**
  - Best Practices
  - Techniques

**Engineer**
- **Scale Fast**
  - Software Engineering
  - Data Science
  - Iterative Delivery
- **Mentoring**
  - Trusted Advisor
Agenda

Why R?

What is Hadoop?

Counting words with MapReduce

RHadoop in Action
   Client scenario: Scaling legacy analytics with RHadoop
   Client scenario: Network security threat detection

Big Data Warehousing with Hive
   Client scenario: Using Hadoop as a “drop-in” database replacement
   Client scenario: Preprocessing data for R

Want to learn more?

Q&A
How many R Packages are there now?

At the command line enter:

> dim(available.packages())
Poll: What do you use for analytics?
Revolution R Enterprise has Open-Source R Engine at the core

3,700 community packages and growing exponentially
What is Hadoop?

An open source project designed to support large scale data processing

Inspired by Google’s MapReduce-based computational infrastructure

Comprised of several components

- Hadoop Distributed File System (HDFS)
- MapReduce processing framework, job scheduler, etc.
- Ingest/outgest services (Sqoop, Flume, etc.)
- Higher level languages and libraries (Hive, Pig, Cascading, Mahout)

Written in Java, first opened up to alternatives through its Streaming API

→ If your language of choice can handle stdin and stdout, you can use it to write MapReduce jobs
Poll: Have you used Hadoop?
Enter RHadoop

RHadoop is an open source project sponsored by Revolution Analytics

Package Overview

  rmr2 - all MapReduce-related functions
  rhdfs - interaction with Hadoop’s HDFS file system
  rhbase - access to the NoSQL HBase database

rmr2 uses Hadoop’s Streaming API to allow R users to write MapReduce jobs in R handles all of the I/O and job submission for you (no `while(<stdin>)`-like loops!)
RHadoop Advantages

Modular

- Packages group similar functions
- Only load (and learn!) what you need
- Minimizes prerequisites and dependencies

Open Source

- Cost: Low (no) barrier to start using
- Transparency: Development, issue tracker, Wiki, etc. hosted on github: [https://github.com/RevolutionAnalytics/RHadoop/](https://github.com/RevolutionAnalytics/RHadoop/)

Supported

- Sponsored by Revolution Analytics
- Training & professional services available
- Support included with Revolution R Enterprise
Hadoop cluster components

Key

*italics*: process

* : MR jobs
Hadoop’s distributed file system

Services
- **Name Node**
- **Data Nodes**
- 64MB blocks
- 3x replication

Cluster
- Primary Master Server
  - *Job Tracker*
  - Name Node
- Secondary Master Server
  - Secondary Name Node
- Client Servers
  - *Hive, Pig, ...*
  - *cron+bash, Azkaban, ...*
  - Sqoop, Scribe, ...
  - Monitoring, Management

Slaves
- Slave Server
  - *Task Tracker*
  - Data Node
- Slave Server
  - *Task Tracker*
  - Data Node
- Slave Server
  - *Task Tracker*
  - Data Node
Wordcount in Hadoop

(The “hello world” of MapReduce)
We need to convert the *Input* into the *Output*.
There is a Map phase

There is a Reduce phase

Input

Mappers

Hadoop uses MapReduce

(N, "...")

(N, "...")

(N, "")

(N, "...")

from Think Big Academy's Hadoop Developer Course
There is a Map phase

Hadoop uses MapReduce

There is a Reduce phase

Map phase

Reduce phase

from Think Big Academy’s Hadoop Developer Course
"I think you should be more explicit here in step two."
There is a Map phase

Hadoop uses MapReduce

(N, "...")

(N, "...")

(N, "")

There is a Reduce phase

(hadoop, 1)

(mapreduce, 1)

(uses, 1)

(is, 1), (a, 1)

(map, 1), (phase, 1)

(there, 1)

(phase, 1)

(is, 1), (a, 1)

(there, 1), (reduce, 1)

0-9, a-l

m-q

r-z

from Think Big Academy's Hadoop Developer Course
There is a Map phase

Hadoop uses MapReduce

(N, "...")

(Mappers)

There is a Reduce phase

(N, "...")

(Sort, Shuffle)

(N, "")

(Reducers)

0-9, a-l

(a, [1,1]),
(hadoop, [1]),
(is, [1,1])

m-q

(map, [1]),
(mapreduce, [1]),
(phase, [1])

r-z

(reduce, [1]),
(there, [1,1]),
(uses, 1)

from Think Big Academy’s Hadoop Developer Course
There is a Map phase

Hadoop uses MapReduce

(N, "...")

(hadoop, 1)

(mapreduce, 1)

(uses, 1)

(is, 1), (a, 1)

(map, 1), (phase, 1)

(there, 1)

There is a Reduce phase

(N, """")

(is, 1), (a, 1)

(phase, 1)

(Output)

There is a Reduce phase

(N, "...")

(mapreduce, 1)

(uses, 1)

(map, 1), (phase, 1)

(there, 1)

(Phase, 1)

(uses, 1)

(reduce 1)

0-9, a-l

(a, [1,1]),

(hadoop, [1]),

(is, [1,1])

m-q

(map, [1]),

(mapreduce, [1]),

(phase, [1,1])

r-z

(reduce, [1]),

(there, [1,1]),

(uses, 1)

Output

a 2

hadoop 1

is 2

map 1

mapreduce 1

phase 2

reduce 1

there 2

uses 1

from Think Big Academy's Hadoop Developer Course
There is a Map phase

Hadoop uses MapReduce

Map:
- Transform one input to 0-N outputs.

Sort, Shuffle

Reduce:
- Collect multiple inputs into one output.
library(rmr2)

map = function(k,lines) {
  words.list = strsplit(lines, '\\s')
  words = unlist(words.list)

  return( keyval(words, 1) )
}

reduce = function(word, counts) {
  keyval(word, sum(counts))
}

wordcount = function (input, output = NULL) {
  mapreduce(input = input ,
            output = output,
            input.format = "text",
            map = map,
            reduce = reduce)
wordcount: submit job and fetch results

Submit job

```r
> hdfs.root = 'wordcount'
> hdfs.data = file.path(hdfs.root, 'data')
> hdfs.out = file.path(hdfs.root, 'out')
> out = wordcount(hdfs.data, hdfs.out)
```

Fetch results from HDFS

```r
> results = from.dfs(out)
> results.df = as.data.frame(results, stringsAsFactors=F)
> colnames(results.df) = c('word', 'count')
> head(results.df)

word count
1 greatness     2
2    damned     3
3       tis     5
4      jade     1
5  magician     1
6     hands     2
```
Code notes

Scalable
  Hadoop and MapReduce abstract away system details
  Code runs on 1 node or 1,000 nodes without modification

Portable
  You write normal R code, interacting with normal R objects
  RHadoop’s rmr2 library abstracts away Hadoop details
  All the functionality you expect is there—including Enterprise R’s

Flexible
  Only the mapper deals with the data directly
  All components communicate via key-value pairs
  Key-value “schema” chosen for each analysis rather than as a prerequisite to loading data into the system
Data silos have developed by source, type, system, department...

- Customer Service, Call Center
- Website search, Clickstreams
- Operations & Finance
- Social Media
Poll: How do you intend to use R & Hadoop?
Scaling up a legacy modeling environment

Scenario

A financial services client has built a model to predict credit risk using a legacy system, but now wants to run the scoring algorithm more frequently than possible in existing hardware- (and license-) bound environment.

Solution

Model coefficients are exported from legacy system and written to HDFS as a text file so they are available for our R code to use while scoring the entire data set nightly.
"Side loading" data made easy

[...] 

# first load in our coefficients 
# Note that this data.frame will be
# _automatically_ distributed to each node

COEFFS.DF = as.data.frame( from.dfs(hdfs.coeffs, 
    format = make.input.format('csv', sep='\t')) )

out = mapreduce(input = hdfs.data, 
    output = hdfs.out, 
    input.format = make.input.format('csv', sep='\t'), 
    map = scoring.map, 
    verbose=T)
Logistic regression scoring “by hand”

```r
score.logit = function(coeffs, data, reverse.coeffs.signs=F)
{
  if( length(coeffs) != length(data) + 1 )
  {
    warning("Bad row: ", data)
    return(NULL)
  }

  # SAS may reverse the signs of the coefficients
  # (this assumes the intercept should be changed as well)
  if (reverse.coeffs.signs)
    coeffs = -1 * coeffs

  intercept = coeffs[1]
  coeffs = coeffs[-1]

  return( 1/(1 + exp(-intercept - sum(coeffs * data))) )
}
```
Network security threat detection

Scenario

A risk-conscious company with a high-profile web presence was seeking a way to detect potential attacks using captured network traffic (PCAP files).

Solution

Typical traffic was modeled using a combination of algorithms on the full data set. Baseline results from this “training” phase are stored on HDFS and can be refreshed periodically, as needed.

A detection job loads the baseline results and analyzes incoming network traffic looking for anomalies.
Big Data Warehousing with Hive

Hive supplies a SQL-like query language very familiar for those with relational database experience.

But Hive compiles, optimizes, and executes these queries as MapReduce jobs on the Hadoop cluster.

Can be used in conjunction with other Hadoop jobs, such as those written with rmr2.
Hive architecture & access

Hive
- Hive
- CLI
- HWI
- Thrift Server
- Driver (compiles, optimizes, executes)
- Metastore

Hadoop
- Master
  - Job Tracker
  - Name Node
- DFS

Access
- Terminal
- browser
- RODBC, RJDBC, etc.

Drivers
- JDBC
- ODBC

Data Access
- Terminal through Hive or browser through Hive or RODBC/RJDBC.
Hadoop as a “drop-in” database replacement

Scenario

A market research company had developed a large collection of mature R code to conduct mission-critical analysis. Reaching the limits of their legacy database infrastructure, the company has decided to move their data to Hadoop, but wants to avoid a “Big Bang” project to preserve existing functionality.

Solution

As a stop-gap solution, ODBC connections to Hive have replaced connections to the legacy relational database with minimal recoding.

Existing functionality was preserved while the team got up to speed with their new Hadoop environment and its capabilities.

Solution allows for parallel testing of two systems.
library(RJDBC)

# set the classpath to include the JDBC driver location, plus commons-logging
[...]  
class.path = c(hive.class.path, commons.class.path)  
drv = JDBC("org.apache.hadoop.hive.jdbc.HiveDriver", classPath=class.path, "\"")  

# make a connection to the running Hive Server:  
conn = dbConnect(drv, "jdbc:hive://localhost:10000/default")  

# setting the database name in the URL doesn't help,  
# so issue 'use databasename' command:  
res = dbSendQuery(conn, 'use mydatabase')  

# submit the query and fetch the results as a data.frame:  
df = dbGetQuery(conn, 'SELECT name, sub FROM employees LATERAL VIEW explode(subordinates) subView AS sub')
Preprocessing data with Hive

Scenario

A marketing analytics team had built their own segmentation model based on their own database of web visitors. A partnership with a new data provider brought an order of magnitude more visitors to segment.

Solution

A series of MapReduce jobs (Hive queries and RHadoop rmr2) was used to match, filter, and transform the data into a manageable size for interactive modeling with Revolution R Enterprise.
Other ways to use R and Hadoop

HDFS

Revolution Enterprise R can read and write files directly on the distributed file system
Files can include ScaleR’s XDF-formatted data sets

MapReduce

Many other R packages have been written to use R and Hadoop together, including RHIPE, segue, Oracle’s R Connector for Hadoop, etc.

Hive

Hadoop Streaming is also available for Hive to leverage functionality external to Hadoop and Java
RHive leverages RServer to connect the two: http://cran.r-project.org/web/packages/RHive/
Want to learn more?

Upcoming public *Getting Started with RHadoop* 1-day classes

- Hands-on examples and exercises covering rhdfs, rhbase, and rmr2
- Algorithms and data include wordcount, analysis of airline flight data, and collaborative filtering using structured and unstructured data from text, CSV files and Twitter
- February 25, 2013 - Palo Alto, CA
- March 13, 2013 - Boston, MA
- more to come: [http://www.eventbrite.com/org/3179781746?s=12196176](http://www.eventbrite.com/org/3179781746?s=12196176)
- (also available as private, on-site training)

Revolution Analytics Quick Start Program for Hadoop

- Private *Getting Started with RHadoop* training
- Onsite consulting assistance for initial use case
- Revolution R for Hadoop licenses and support
Thank you! Any Questions?