

Distributed Data Analysis with Hadoop and R

Jonathan Seidman and Ramesh Venkataramaiah, Ph. D.

StrangeLoop2011

September 20 | 2011



Flow of this Talk

- Introductions
- Hadoop, R and Interfacing the two
- Our Prototypes
- A use case for interfacing Hadoop and R
- Alternatives for Running R on Hadoop
- Alternatives to Hadoop and R
- Conclusions
- References

Who We Are

- Ramesh Venkataramaiah, Ph. D.
 - Principal Engineer, TechOps
 - rvenkataramaiah@orbitz.com
 - @rvenkatar
- Jonathan Seidman
 - Lead Engineer, Business Intelligence/Big Data Team
 - Co-founder/organizer of Chicago Hadoop User Group (<http://www.meetup.com/Chicago-area-Hadoop-User-Group-CHUG>) and Chicago Big Data (<http://www.meetup.com/Chicago-Big-Data/>)
 - jseidman@orbitz.com
 - @jseidman
- Orbitz Careers
 - <http://careers.orbitz.com/>
 - @OrbitzTalent

Launched in 2001

ORBITZ WTFHS Welcome to Orbitz [Sign In](#) | [Register now](#)
[Write A Review](#) | [My Trips](#) | [My Account](#) | [Traveler Update](#) | [Customer Support](#)

Quick Search: Vacation Packages | Hotels | Flights | Cars | Cruises | Activities | Deals

☒ Flight ☐ Flight + Hotel ☐ Flight + Car ☐ Hotel + Car ☐ Flight + Hotel + Car ☐ Activities ☐ Cruises

Summer Hotel Sale
SAVE up to 30%
[See offers](#)

☒ Round-trip ☐ One-way ☐ Multi-city

From City name or airport: To City name or airport:
Dallas, TX (DFW) PDX

Leave: Return:
Anytime Anytime

Travelers: [\(Children or seniors?\)](#)
Adult (18-64)

Flight preference: ☐ I prefer non-stop flights

[Expand search options](#) (Preferred airlines, first/business class, etc.)
[Flexible dates](#)

Orbitz on the go: Free mobile apps
The **Orbitz Hotels App for iPad** delivers the entire selection of Orbitz hotels, refined comparison tools & booking in 3 taps.
Plus, with the Orbitz app for **iPhone** & **Android**™ devices, you can book flights, hotels and cars and get updates on the go.

48-Hour Hotel Sale
[See offers](#)

Save up to 55%

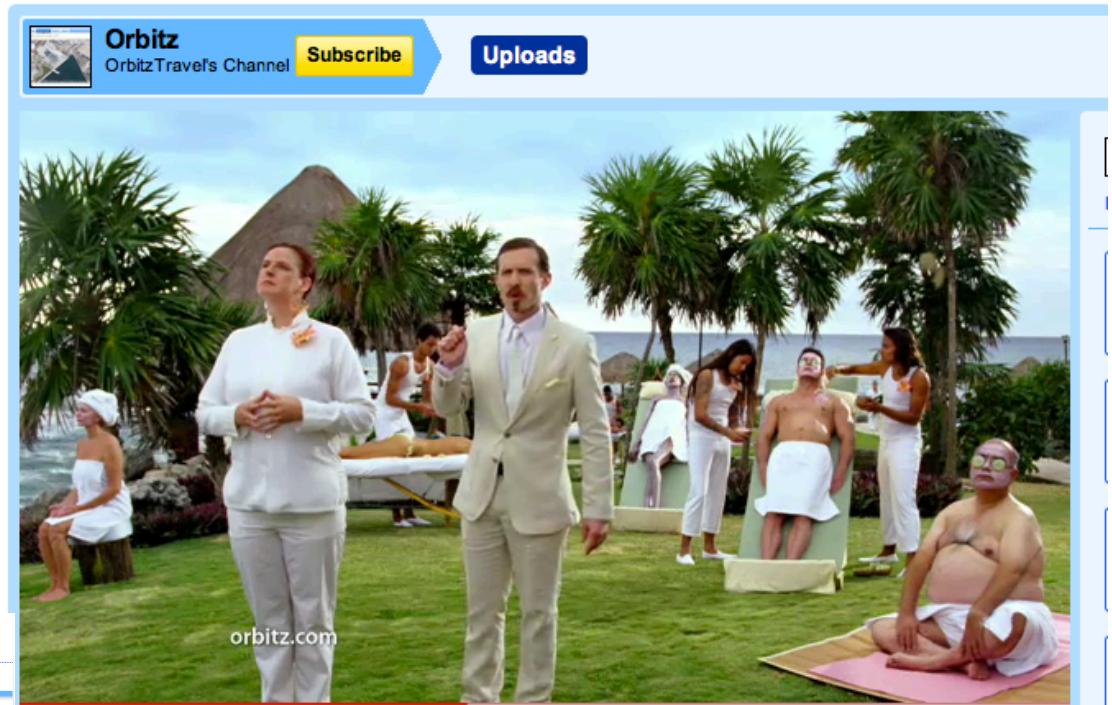
Top Destinations
[Caribbean](#)
[Florida](#)
[Hawaii](#)
[Las Vegas](#)
[Mexico](#)
[See more destinations](#)

Top Hotels
[Disney*](#)
[Hilton Worldwide](#)
[Marriott Hotels](#)
[Sandals Resorts](#)
[Starwood Hotels](#)
[See more hotels](#)

Top Interests
[All-inclusive](#)
[Beach](#)
[Family](#)
[Last-minute](#)
[LGBT](#)
[See more interests](#)

DEAL DETECTOR Set your price — We'll find a match and alert you.
[Start using DealDetector](#) | [Sign in to see your DealDetector trips](#)

More featured deals [View all deals](#)



Over 160 million bookings

7th Largest seller of travel in the world

Hadoop and R as an analytic platform?



What is Hadoop?

Distributed file system (HDFS) and parallel processing framework.

Uses **MapReduce** programming model as the core.

Provides **fault tolerant and scalable storage**
of very large datasets across machines in a cluster.

What is R? When do we need it?

Open-source stat package with visualization

Vibrant community support.

One-line calculations galore!

Steep learning curve but worth it!

Insight into statistical properties and trends...

or for machine learning purposes...

or Big Data to be understood well.

Our Options

- Data volume reduction by sampling
 - Very bad for long-tail data distribution
 - Approximation lead to bad conclusion
- Scaling R
 - Still in-memory
 - But make it parallel using segue, Rhipe, R-Hive...
- Use sql-like interfaces
 - Apache Hive with Hadoop
 - File sprawl and process issues
- Regular DBMS
 - How to fit square peg in a round hole
 - No in-line R calls from SQL but commercial efforts are underway.
- This Talk: How to bring Hadoop's parallel processing capability to R environment.

Our prototypes

User segmentations

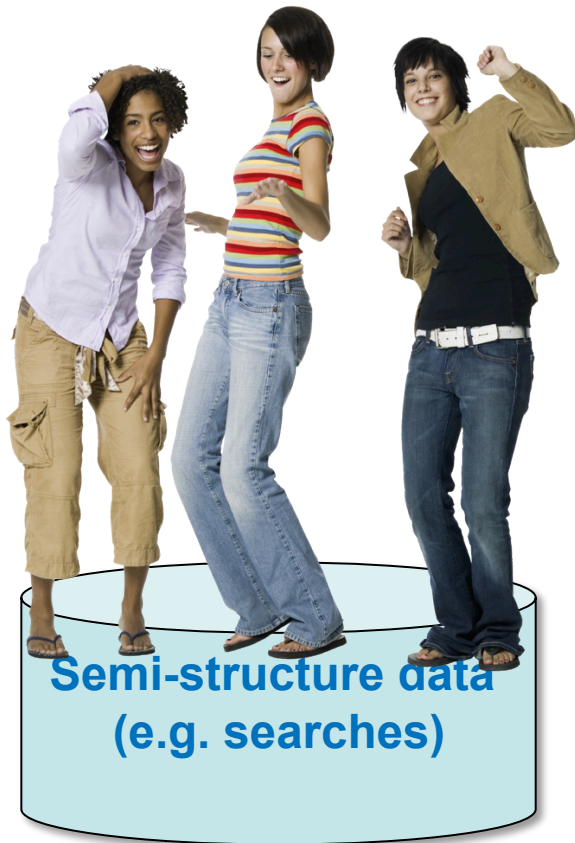
Hotel bookings

Airline Performance*

* Public dataset



We have two distinct dataspace serving different constituents

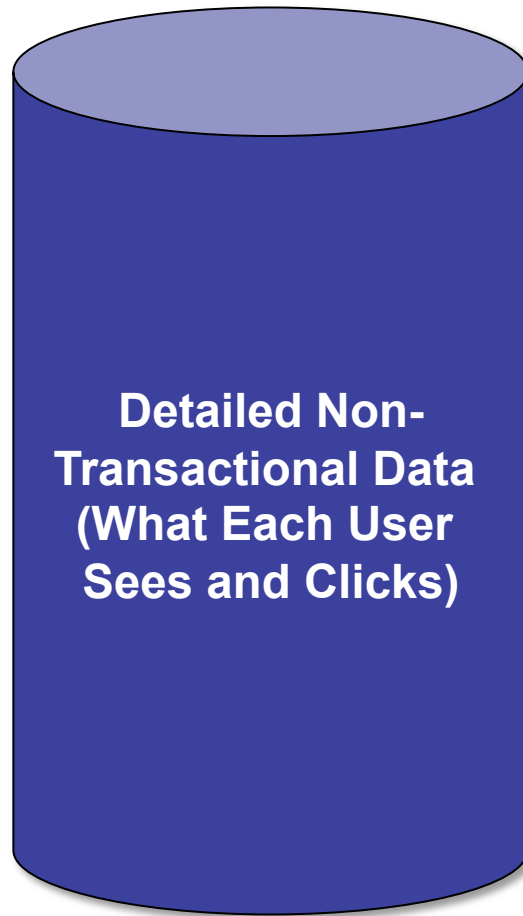


Hadoop Cluster

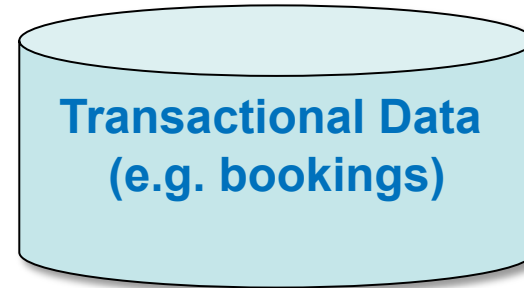


Data Warehouse

Our Hadoop infrastructure allows us to record and process user activity at the individual level



Hadoop



Data Warehouse

Getting a Buy-in

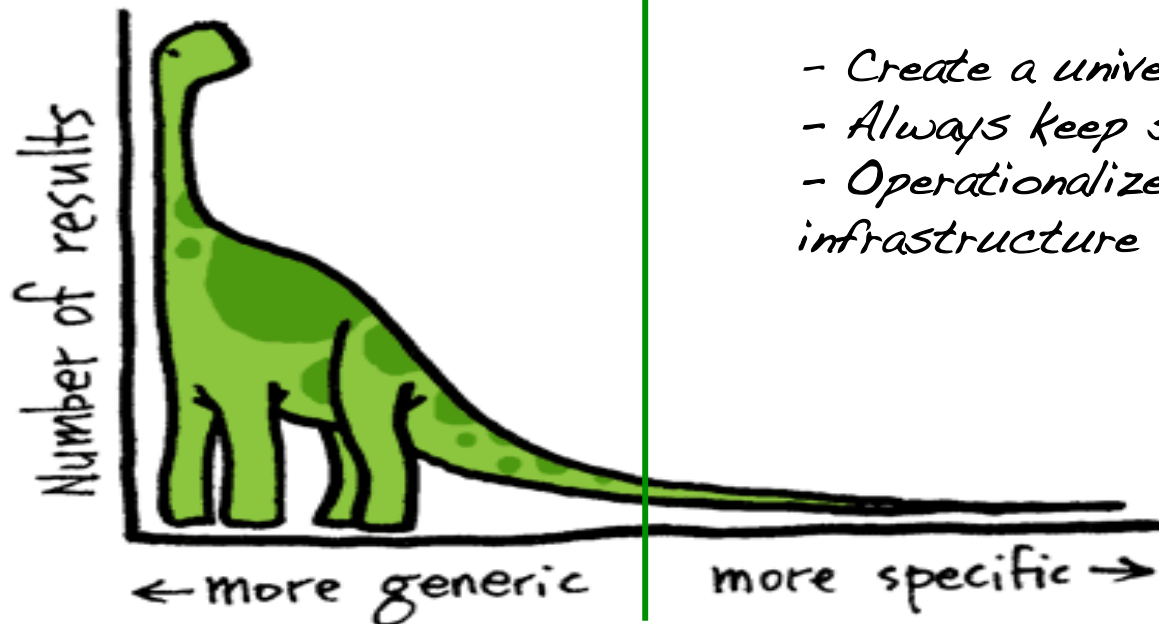
presented a long-term, semi-structured data growth story and explained how this will help harness long-tail opportunities at lowest cost.

- Traditional DW
- Classical Stats
- Sampling



- Big Data
- Specific spikes
- Median is not the message

- Create a universal key
- Always keep source data
- Operationalize the infrastructure



An example of “median is not the message”

- Positional Bias during Hotel Searches

[Change search](#) San Jose International Airport, San Jose, California, United States | Check-in: Tue, Apr 26, 2011 | Check-out: Sat, Apr 30, 2011 | Nights: 4 | Room(s): 1 | Guest(s): 2

Call us to book!
1-800-733-1297
(toll free)

Why book on Orbitz? Price Assurance + Low Price Guarantee + No Orbitz hotel change or cancel fees [Learn more](#)

Stars

☒ Any

☐ ★★★★★

☐ ★★★★☆

☐ ★★★☆☆

☐ ★★☆☆☆

☐ ★☆☆☆☆

Reviewer score

1.0 to 5.0

Amenities

☒ Any

☐ Airport Shuttle (7)

☐ Free parking (123)

☐ Pool (134)

☐ Spa services (4)

☐ Wireless Internet (59)

[See all](#)

Hotel chain

☒ Any

☐ All Seasons Europe (0)

☐ Aloft Hotels (0)


San Jose Airport Garden Hotel

★★★☆☆ [Save](#) [Review](#)

Nightly rates from ~~\$119~~ **\$83** [Select](#)

Total Price **\$94** [Price Assurance](#)

[Overview](#) [Description](#) [Photos](#) [Map](#) [Amenities](#)

 **Reviewer score** 3.4 out of 5 [66 reviews](#)

1740 North First St., San Jose, CA 95112-4584
1.2 miles North from the center of San Jose International Airport

[More hotel details](#) [Virtual tour](#)

Special Offer

• Book Now and Save 30%
[View all available promotions](#)


Arena Hotel

★★★☆☆ [Save](#) [Review](#)

Nightly rates from ~~\$66~~ **\$66** [Select](#)

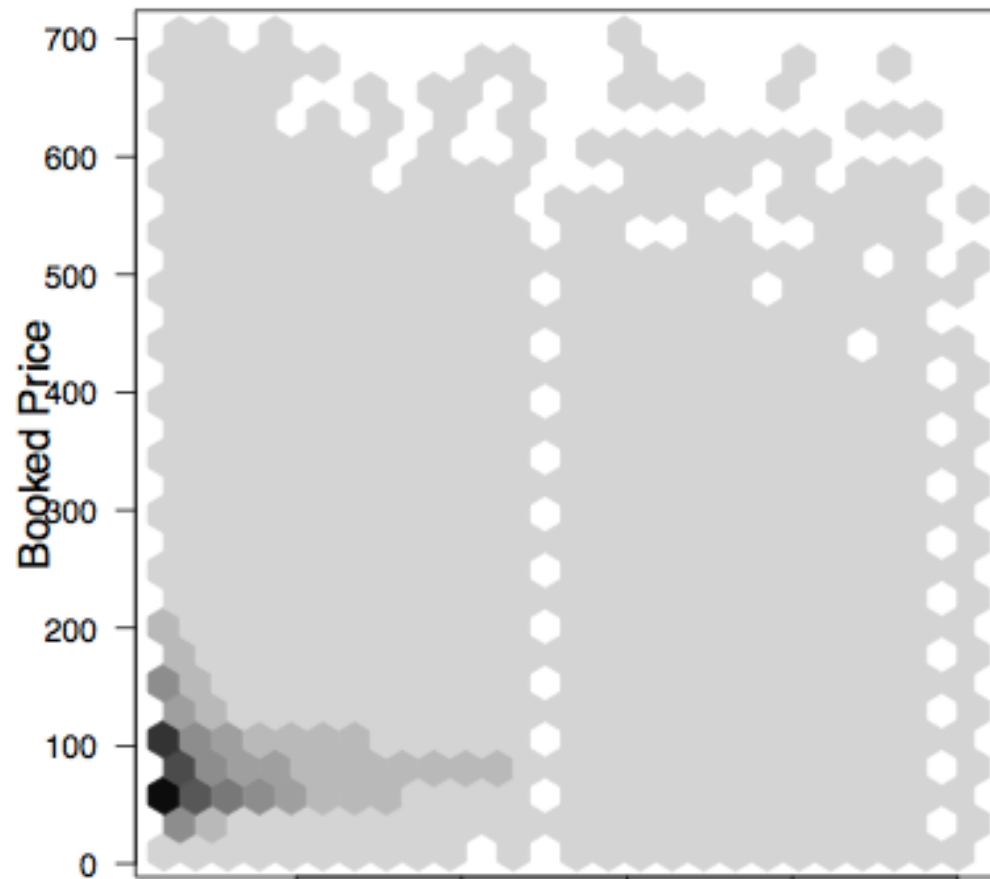
Total Price **\$73** [Price Assurance](#)

[Overview](#) [Description](#) [Photos](#) [Map](#) [Amenities](#)

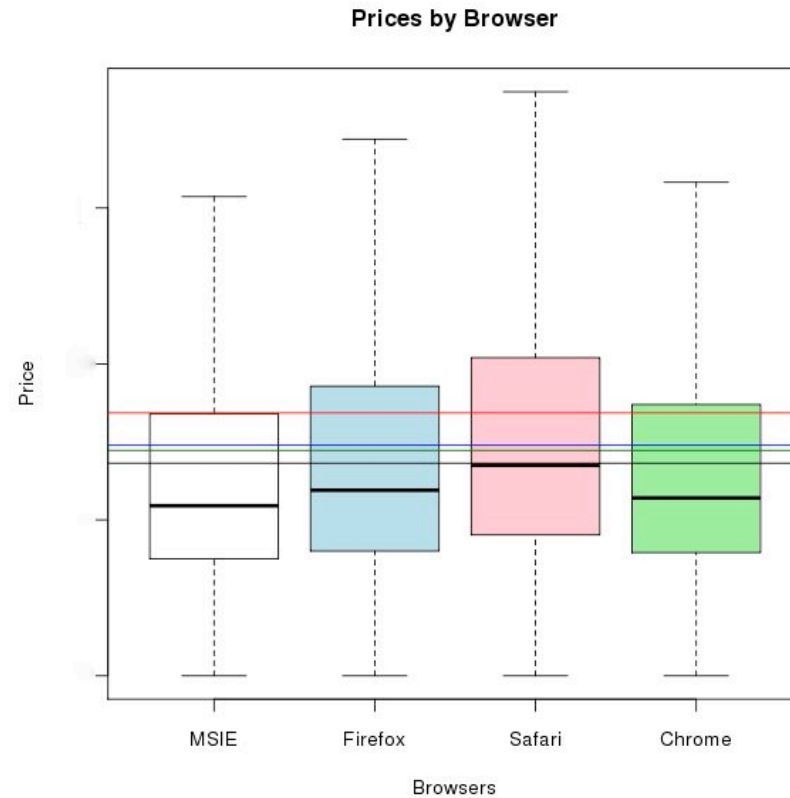
 **Reviewer score** 3.6 out of 5 [77 reviews](#)

817 The Alameda, San Jose, CA 95126
1.6 miles Southeast from the center of San Jose International Airport

Our Customers pick top positions the most...

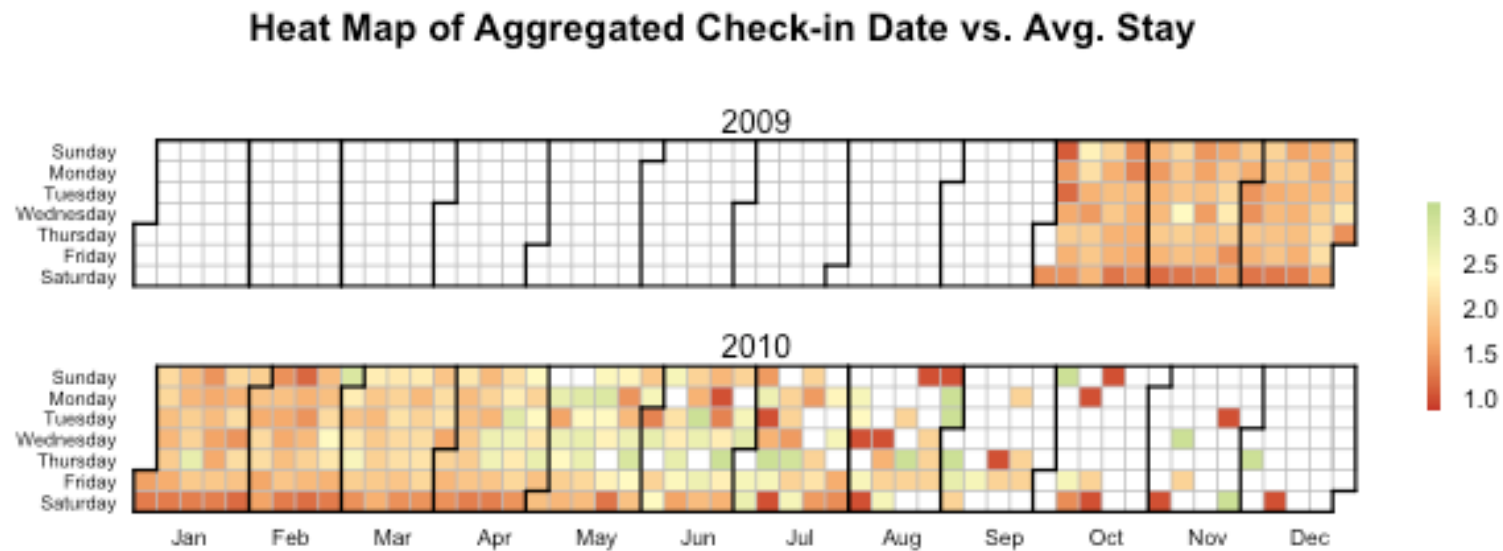


Safari Users Seem to be Interested in More Expensive Hotels



Seasonal variations

- Customer hotel stay gets longer during summer months
- Could help in designing search based on seasons.



Workload and Resource Partition

Purpose	Data Volume	Platform preference	Resource Level
Collection	Scalable, elastic GB to TB	Hadoop (cluster level)	Developers
Aggregation/ Summary	Large scale, Big data GB to TB	Rhipe Hadoop streaming Hadoop Interactive	Developers Analysts Machine Learning Teams
Modeling/ Visualization	Small datasets, In-memory, MB to GB	R (stand-alone)	Analysts Machine Learning Teams

Airline Performance



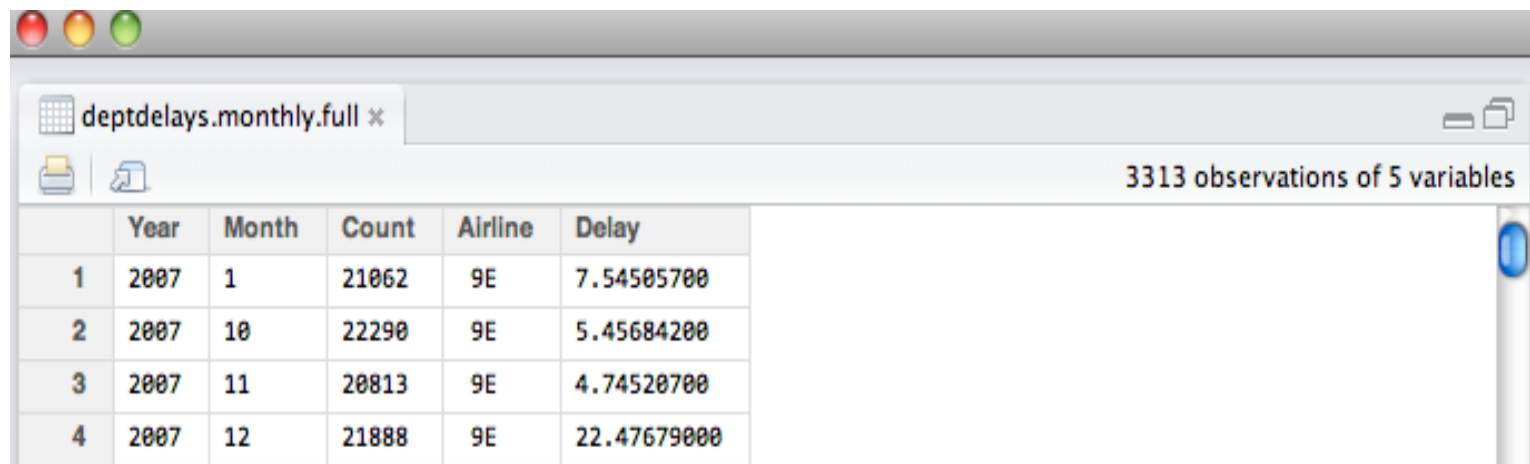
Description of Use Case

- Analyze openly available dataset: Airline on-time performance.
- Dataset was used in “Visualization Poster Competition 2009”
 - Consists of flight arrival/departure details from 1987-2008.
 - Approximately 120 MM records totaling 12GB.
- Available at: <http://stat-computing.org/dataexpo/2009/>

A sample of the text file

```
1987,10,23,5,1841,1750,2105,2005,PS,1905,NA,144,135,NA,60,51,LAX,SEA,954,NA,NA,0,NA,0,...
1987,10,24,6,1752,1750,2010,2005,PS,1905,NA,138,135,NA,5,2,LAX,SEA,954,NA,NA,0,NA,0,...
...
...
```

Our dataset



The screenshot shows a data viewer window with a title bar containing three colored buttons (red, yellow, green) and a tab labeled 'deptdelays.monthly.full x'. Below the tab bar, there are icons for a folder and a document, and a status bar indicating '3313 observations of 5 variables'. The main area displays a table with the following data:

	Year	Month	Count	Airline	Delay
1	2007	1	21062	9E	7.54505700
2	2007	10	22290	9E	5.45684200
3	2007	11	20813	9E	4.74520700
4	2007	12	21888	9E	22.47679000

Airline Delay Plot: R code

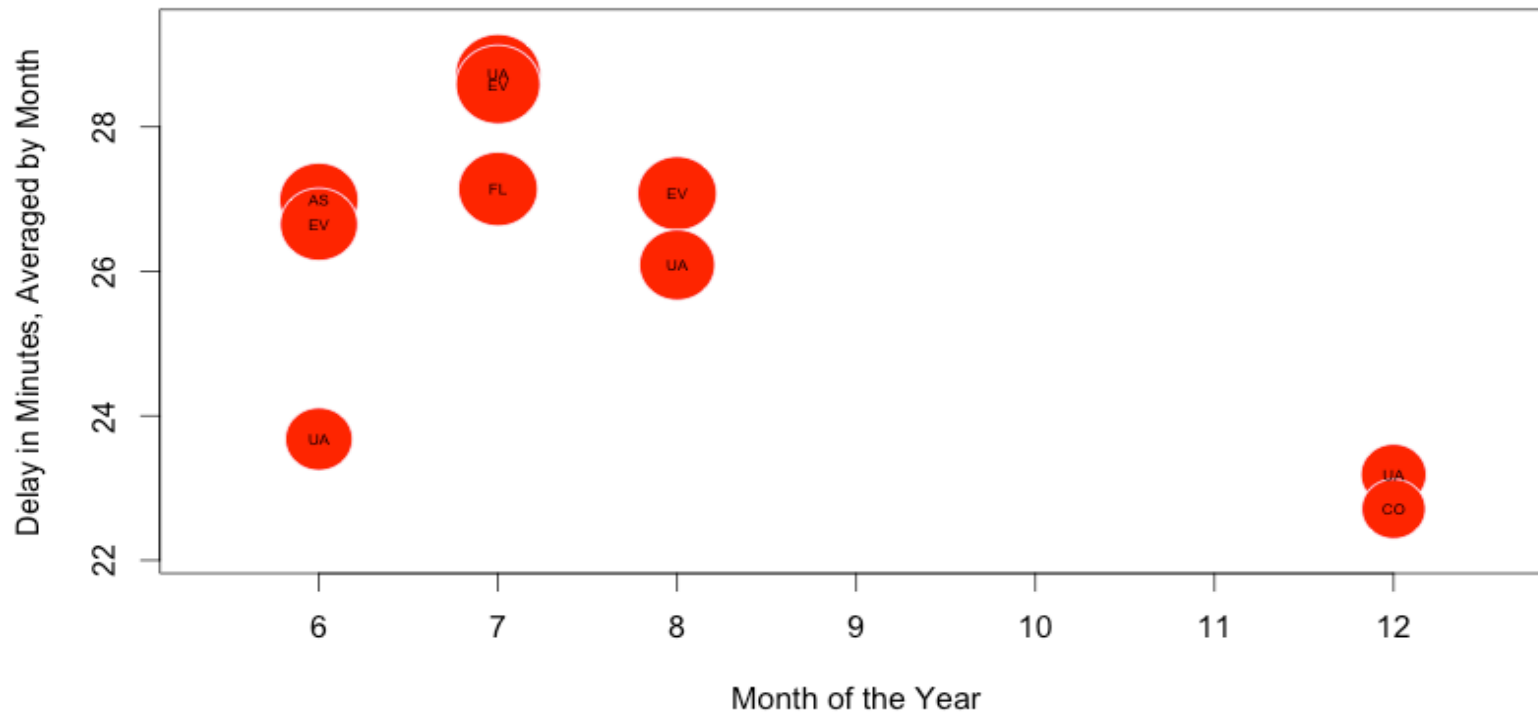
```
> deptdelays.monthly.full <- read.delim("~/OSCON2011/Delays_by_Month.dat", header=F)
> View(deptdelays.monthly.full)
> names(deptdelays.monthly.full) <- c("Year", "Month", "Count", "Airline", "Delay")
```

```
> Delay_by_month <- deptdelays.monthly.full[order(deptdelays.monthly.full
$Delay,decreasing=TRUE),]
```

```
> Top_10_Delay_by_Month <- Delay_by_Month[1:10,]
> Top_10_Normal <- ((Delay - mean(Delay)) / sd(Delay))
```

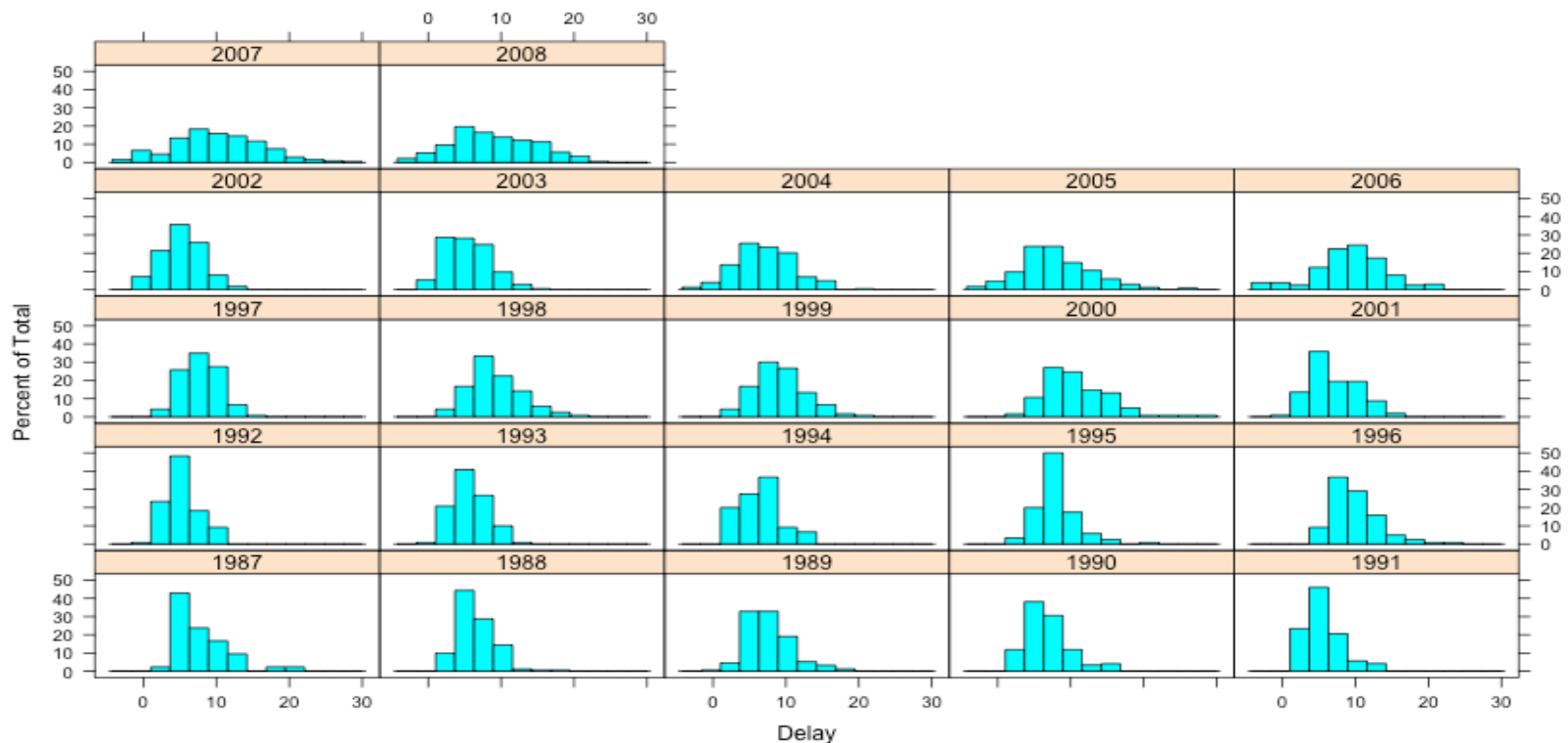
```
> symbols( Month, Delay, circles= Top_10_Normal, inches=.3, fg="white",bg="red",...)
> text(Month, Delay, Airline, cex= 0.5)
```

Airline delay



Multiple Distributions: R code

```
> library(lattice)
> deptdelays.monthly.full$Year <- as.character(deptdelays.monthly.full$Year)
> h <- histogram(~Delay|Year,data=deptdelays.monthly.full,layout=c(5,5))
> update(h)
```



Running R on Hadoop: Hadoop Streaming



Hadoop Streaming – Overview

- An alternative to the Java MapReduce API which allows you to write jobs in any language supporting stdin/stdout.
- Limited to text data in current versions of Hadoop. Support for binary streams added in 0.21.0.
- Requires installation of R on all DataNodes.

Hadoop Streaming – Dataflow

Input to map

```
1988,1,9,6,1348,1331,1458,1435,PI,942,NA,70,64,NA,23,17,SYR,BWI...
1988,1,17,7,1331,1331,1440,1435,PI,942,NA,69,64,NA,5,0,SYR,BWI...
1987,10,14,3,741,730,912,849,PS,1451,NA,91,79,NA,23,11,SAN,SFO...
1987,10,21,3,728,730,848,849,PS,1451,NA,80,79,NA,-1,-2,SAN,SFO...
1987,10,23,5,731,730,902,849,PS,1451,NA,91,79,NA,13,1,SAN,SFO...
1987,10,30,5,1712,1658,1811,1800,DL,475,NA,59,62,NA,11,14,LEX,ATL...
```

*

Output from map

PII1988I1	17
PII1988I1	0
PSI1987I10	11
PSI1987I10	-2
PSI1987I10	1
DLI1987I10	14

* Map function receives input records line-by-line via standard input.

Hadoop Streaming – Dataflow Continued

Input to reduce

DL 1987 10	14
PI 1988 1	0
PI 1988 1	17
PS 1987 10	1
PS 1987 10	11
PS 1987 10	-2

*

Output from reduce

1987	10	1	DL	14
1988	1	2	PI	8.5
1987	10	3	PS	3.333333

* Reduce receives map output key/value pairs sorted by key, line-by-line.

Hadoop Streaming Example – map.R

```
1 #!/usr/bin/env Rscript
2
3 # For each record in airline dataset, output a new record consisting of
4 # "CARRIER|YEAR|MONTH \t DEPARTURE_DELAY"
5
6 con <- file("stdin", open = "r")
7 while (length(line <- readLines(con, n = 1, warn = FALSE)) > 0) {
8   fields <- unlist(strsplit(line, "\\,"))
9   # Skip header lines and bad records:
10   if (!(identical(fields[[1]], "Year")) & length(fields) == 29) {
11     deptDelay <- fields[[16]]
12     # Skip records where departure delay is "NA":
13     if (!(identical(deptDelay, "NA"))) {
14       # field[9] is carrier, field[1] is year, field[2] is month:
15       cat(paste(fields[[9]], "|", fields[[1]], "|", fields[[2]], sep=""), "\t",
16         deptDelay, "\n")
17     }
18   }
19 }
20 close(con)
```

Hadoop Streaming Example – reduce.R

```
1 #!/usr/bin/env Rscript
2
3 # For each input key, output a record composed of
4 # YEAR \t MONTH \t RECORD_COUNT \t AIRLINE \t AVG_DEPT_DELAY
5
6 con <- file("stdin", open = "r")
7 delays <- numeric(0) # vector of departure delays
8 lastKey <- ""
9 while (length(line <- readLines(con, n = 1, warn = FALSE)) > 0) {
10   split <- unlist(strsplit(line, "\t"))
11   key <- split[[1]]
12   deptDelay <- as.numeric(split[[2]])
13
14   # Start of a new key, so output results for previous key:
15   if (!identical(lastKey, "") & (!identical(lastKey, key))) {
16     keySplit <- unlist(strsplit(lastKey, "\\|"))
17     cat(keySplit[[2]], "\t", keySplit[[3]], "\t", length(delays), "\t", keySplit[[1]], "\t", (mean(delays)), "\n")
18     lastKey <- key
19     delays <- c(deptDelay)
20   } else { # Still working on same key so append dept delay value to vector:
21     lastKey <- key
22     delays <- c(delays, deptDelay)
23   }
24 }
25
26 # We're done, output last record:
27 keySplit <- unlist(strsplit(lastKey, "\\|"))
28 cat(keySplit[[2]], "\t", keySplit[[3]], "\t", length(delays), "\t", keySplit[[1]], "\t", (mean(delays)), "\n")
```

Running R on Hadoop: Hadoop Interactive



Hadoop Interactive (hive) – Overview

- Very unfortunate acronym.
- Provides an interface to Hadoop from the R environment.
 - Functions to access HDFS
 - Control Hadoop
 - And run streaming jobs directly from R
- Allows HDFS data, including the output from MapReduce processing, to be manipulated and analyzed directly from R.
- Seems to still have some rough edges.

Hadoop Interactive – Example

```
#!/usr/bin/env Rscript

mapper <- function() {
  ...
}

reducer <- function() {
  ...
}

library(hive)
DFS_dir_remove("/dept-delay-month", recursive = TRUE, henv = hive())
hive_stream(mapper = mapper, reducer = reducer,
            input="/data/airline/", output="/dept-delay-month")
results <- DFS_read_lines("/dept-delay-month/part-r-00000", henv = hive())
```


Running R on Hadoop: RHIPE



RHIPE – Overview

- Active project with frequent updates and active community.
- RHIPE is based on Hadoop streaming source, but provides some significant enhancements, such as support for binary files (sort of).
- Developed to provide R users with access to same Hadoop functionality available to Java developers.
 - For example, provides `rhcounter()` and `rhstatus()`, analagous to counters and the reporter interface in the Java API.

RHIPE – Overview

- Can be somewhat confusing and intimidating.
 - Then again, the same can be said for the Java API.
 - Worth taking the time to get comfortable with.

RHIPE – Overview

- Allows developers to work directly on data stored in HDFS in the R environment.
- Also allows developers to write MapReduce jobs in R and execute them on the Hadoop cluster.
- RHIPE uses Google protocol buffers to serialize data. Most R data types are supported.
 - Using protocol buffers increases efficiency and provides interoperability with other languages.
- Must be installed on all DataNodes.

RHIPE – MapReduce

```
map <- expression({})
reduce <- expression(
  pre = {...},
  reduce = {...},
  post = {...}
)
z <- rhmr(map=map,reduce=reduce,
  inout=c("text","sequence"),
  ifolder=INPUT_PATH ,
  ofolder=OUTPUT_PATH,
  ...)
rhex(z)
```

RHIPE – Dataflow

Input to map

```
Keys = [...]  
Values =  
[1988,1,9,6,1348,1331,1458,1435,PI,942,NA,70,64,NA,23,17,SYR,BWI...  
1988,1,17,7,1331,1331,1440,1435,PI,942,NA,69,64,NA,5,0,SYR,BWI...  
1987,10,14,3,741,730,912,849,PS,1451,NA,91,79,NA,23,11,SAN,SFO...  
1987,10,21,3,728,730,848,849,PS,1451,NA,80,79,NA,-1,-2,SAN,SFO...  
1987,10,23,5,731,730,902,849,PS,1451,NA,91,79,NA,13,1,SAN,SFO...  
1987,10,30,5,1712,1658,1811,1800,DL,475,NA,59,62,NA,11,14,LEX,ATL...]
```

*

Output from map

PII1988I1	17
PII1988I1	0
PSI1987I10	11
PSI1987I10	-2
PSI1987I10	1
DLI1987I10	14

* Note that Input to map is a vector of keys and a vector of values.

RHIPE – Dataflow Continued

Input to reduce

DL 1987 10	[14]
PI 1988 1	[0, 17]
PS 1987 10	[1,11,-2]

*

Output from reduce

1987	10	1	DL	14
1988	1	2	PI	8.5
1987	10	3	PS	3.333333

* Input to reduce is a key and a vector containing a subset of intermediate values associated with that key. The reduce will get called until no more values exist for the key.

RHIPE – Example

```
1 #! /usr/bin/env Rscript
2
3 library(Rhipe)
4 rhinit(TRUE, TRUE)
5
6 # Output from map is:
7 # "CARRIER|YEAR|MONTH \t DEPARTURE_DELAY"
8 map <- expression({
9   # For each input record, parse out required fields and output new record:
10   extractDeptDelays = function(line) {
11     fields <- unlist(strsplit(line, "\\,"))
12     # Skip header lines and bad records:
13     if (!(identical(fields[[1]], "Year")) & length(fields) == 29) {
14       deptDelay <- fields[[16]]
15       # Skip records where departure delay is "NA":
16       if (!(identical(deptDelay, "NA"))) {
17         # field[9] is carrier, field[1] is year, field[2] is month:
18         rhcollect(paste(fields[[9]], "|", fields[[1]], "|", fields[[2]], sep=""),
19                   deptDelay)
20       }
21     }
22   }
23   # Process each record in map input:
24   lapply(map.values, extractDeptDelays)
25 })
26 .
```


RHIPE – Example

```
27 # Output from reduce is:
28 # YEAR \t MONTH \t RECORD_COUNT \t AIRLINE \t AVG_DEPT_DELAY
29 reduce <- expression(
30   pre = {
31     delays <- numeric(0)
32   },
33   reduce = {
34     # Depending on size of input, reduce will get called multiple times
35     # for each key, so accumulate intermediate values in delays vector:
36     delays <- c(delays, as.numeric(reduce.values))
37   },
38   post = {
39     # Process all the intermediate values for key:
40     keySplit <- unlist(strsplit(reduce.key, "\\|"))
41     count <- length(delays)
42     avg <- mean(delays)
43     rhcollect(keySplit[[2]],
44               paste(keySplit[[3]], count, keySplit[[1]], avg, sep="\t"))
45   }
46 )
47
```

RHIPE – Example

```
48 inputPath <- "/data/airline/"
49 outputPath <- "/dept-delay-month"
50
51 # Create job object:
52 z <- rhmr(map=map, reduce=reduce,
53           ifolder=inputPath, ofolder=outputPath,
54           inout=c('text', 'text'), jobname='Avg Departure Delay By Month',
55           mapred=list(mapred.reduce.tasks=1))
56 # Run it:
57 rhex(z)
58
59 library(lattice)
60
61 # Get the results from HDFS and use to create a dataframe:
62 results <- rhread(paste(outputPath, "/part-*", sep = ""), type = "text")
63 write(results, file="deptdelays.dat")
64 deptdelays.monthly.full <- read.delim("deptdelay.dat", header=F)
65 names(deptdelays.monthly.full) <- c("Year", "Month", "Count", "Airline", "Delay")
66 deptdelays.monthly.full$Year <- as.character(deptdelays.monthly.full$Year)
67
68 # Visualize results:
69 h <- histogram(~Delay|Year, data=deptdelays.monthly.full, layout=c(5,5))
70 update(h)
```

Running R on Hadoop: rmr



rmr Overview

- New project from Revolution Analytics introduced August 2011.
- Part of RHadoop, a suite of open-source projects which also includes:
 - rhdfs – functions to access and manage HDFS from within R.
 - rhbase – functions providing basic connectivity to HBase.
- Goals are to provide productive environment for MapReduce programming in an R-like way - "...stay true to map reduce and true to R ..."
- Reduce gets all intermediate values for each key (yay!).
- Like RHIPE, based on streaming source.

rmr – Example

```
1 #!/usr/bin/env Rscript
2
3 library(rmr)
4
5 csvtextinputformat = function(line) keyval(NULL, unlist(strsplit(line, "\\,")))
6
7 deptdelay = function (input, output) {
8   mapreduce(input = input,
9             output = output,
10            textinputformat = csvtextinputformat,
11            map = function(k, fields) {
12              # Skip header lines and bad records:
13              if (!(identical(fields[[1]], "Year")) & length(fields) == 29) {
14                deptDelay <- fields[[16]]
15                # Skip records where departure delay is "NA":
16                if (!(identical(deptDelay, "NA"))) {
17                  # field[9] is carrier, field[1] is year, field[2] is month:
18                  keyval(c(fields[[9]], fields[[1]], fields[[2]]), deptDelay)
19                }
20              }
21            },
22            reduce = function(keySplit, vv) {
23              keyval(keySplit[[2]], c(keySplit[[3]], length(vv),
24                                     keySplit[[1]], mean(as.numeric(vv))))
25            })
26 }
27
28 from.dfs(deptdelay("/data/airline/", "/dept-delay-month"))
29
```

Running R on Hadoop: Segue



Segue – Overview

- Intended to work around single-threading in R by taking advantage of Hadoop streaming to provide simple parallel processing.
 - For example, running multiple simulations in parallel.
- Suitable for ~~embarrassingly~~ pleasantly parallel problems – big CPU, not big data.
- Runs on Amazon's Elastic Map Reduce (EMR).
 - Not intended for internal clusters.
- Provides `emrapply()`, a parallel version of `lapply()`

Segue – Example

```
1 estimatePi <- function(seed){
2   set.seed(seed)
3   numDraws <- 1e6
4
5   r <- .5 #radius... in case the unit circle is too boring
6   x <- runif(numDraws, min=-r, max=r)
7   y <- runif(numDraws, min=-r, max=r)
8   inCircle <- ifelse( (x^2 + y^2)^.5 < r , 1, 0)
9
10  return(sum(inCircle) / length(inCircle) * 4)
11 }
12
13
14 seedList <- as.list(1:1000)
15 require(segue)
16 myCluster <- createCluster(20)
17 myEstimates <- emrapply( myCluster, seedList, estimatePi )
18 stopCluster(myCluster)
```


Performance Comparison: Streaming and RHIPE



Performance Testing – Environment and Setup

- Twenty-eight DataNodes:
 - Dual hex-core
 - 24GB RAM
 - Shared cluster.
- Data
 - Airline dataset
 - 22 input files
 - About 12GB uncompressed data

Performance Comparison

Number of Reducers	Streaming	RHIPE
264	246 seconds*	96 seconds*

*All numbers are an average of 3 runs.

Alternatives

Alternate languages/libraries:

- Apache Mahout
 - Scalable machine learning library.
 - Offers clustering, classification, collaborative filtering on Hadoop.
- Python
 - Many modules available to support scientific and statistical computing.

Alternatives

Alternative parallel processing frameworks:

- Revolution Analytics
 - Provides commercial packages to support processing big data with R.
- Other HPC/parallel processing packages for R, e.g. Rmpi or snow.

Alternatives

Apache Hive + RJDBC?

- We haven't been able to get it to work yet.
- You can however wrap calls to the Hive client in R to return R objects. See <https://github.com/satpreetsingh/rDBwrappers/wiki>

Alternatives

Interesting solutions that you can't have:

- Ricardo
 - Developed as part of a research project at IBM.
 - Interesting paper published, but apparently no plans to make available.

Conclusions

- If practical, consider using Hadoop to aggregate data for input to R analyses.
- Avoid using R for general purpose MapReduce use.

Conclusions

- For simple MapReduce jobs, or “embarrassingly” parallel jobs on a local cluster, consider Hadoop streaming.
 - Limited to processing text only.
 - But easy to test at the command line outside of Hadoop:
 - `$ cat DATAFILE | ./map.R | sort | ./reduce.R`
- To run compute-bound analyses with relatively small amount of data on the cloud look at Segue.

Conclusions

- Otherwise, your best bet is RHIFE, but definitely check out rmr.
- Also consider alternatives – Mahout, Python, etc.

Conclusions

On an operational note:

- Make sure your cluster nodes are consistent – same version of R installed, required libraries are installed on each node, etc.

Example Code

- <https://github.com/jseidman/hadoop-R>

References

- Hadoop
 - Apache Hadoop project: <http://hadoop.apache.org/>
 - Hadoop The Definitive Guide, Tom White, O' Reilly Press, 2011
- R
 - R Project for Statistical Computing: <http://www.r-project.org/>
 - R Cookbook, Paul Teetor, O' Reilly Press, 2011
 - Getting Started With R: Some Resources:
<http://quanttrader.info/public/gettingStartedWithR.html>

References

- Hadoop Streaming
 - Documentation on Apache Hadoop Wiki:
<http://hadoop.apache.org/mapreduce/docs/current/streaming.html>
 - Word count example in R :
<https://forums.aws.amazon.com/thread.jspa?messageID=129163>

References

- Hadoop InteractiVE
 - Project page on CRAN:
<http://cran.r-project.org/web/packages/hive/index.html>
 - Simple Parallel Computing in R Using Hadoop:
<http://www.rmetrics.org/Meielisalp2009/Presentations/Theussl1.pdf>

References

- RHIPE
 - RHIPE - R and Hadoop Integrated Processing Environment: <http://www.stat.purdue.edu/~sguha/rhipe/>
 - Code: <http://code.google.com/p/rhipe/>
 - Wiki: <http://code.google.com/p/rhipe/w/list>
 - Installing RHIPE on CentOS: <https://groups.google.com/forum/#!topic/brumail/qH1wjtBgwYI>
 - Introduction to using RHIPE: <http://ml.stat.purdue.edu/rhafen/rhipe/>
 - RHIPE combines Hadoop and the R analytics language, SD Times: <http://www.sdtimes.com/link/34792>

References

- RHIPE
 - Using R and Hadoop to Analyze VoIP Network Data for QoS, Hadoop World 2010:
 - video:
http://www.cloudera.com/videos/hw10_video_using_r_and_hadoop_to_analyze_voip_network_data_for_qos
 - slides:
http://www.cloudera.com/resource/hw10_voice_over_ip_studying_traffic_characteristics_for_quality_of_service
 - RHIPE examples (k-means, etc.):
http://groups.google.com/group/brumail/browse_thread/thread/e403db404f039e31?pli=1

References

- RHadoop (including rmr)
 - Github: <https://github.com/RevolutionAnalytics/RHadoop>
 - “Advanced ‘Big Data’ Analytics with R and Hadoop”
whitepaper:
<http://info.revolutionanalytics.com/R-and-Hadoop-Big-Data-Analytics-White-Paper.html>

References

- Segue
 - Project page: <http://code.google.com/p/segue/>
 - Google Group: <http://groups.google.com/group/segue-r>
 - Abusing Amazon's Elastic MapReduce Hadoop service... easily, from R, Jefferey Breen:
<http://jeffreymbreen.wordpress.com/2011/01/10/segue-r-to-amazon-elastic-mapreduce-hadoop/>
 - Presentation at Chicago Hadoop Users Group March 23, 2011:
<http://files.meetup.com/1634302/segue-presentation-RUG.pdf>

References

- Sawmill (A framework for integrating a PMML-compliant Scoring Engine with Hadoop).
 - More information:
 - Open Data Group www.opendatagroup.com
 - oscon-info@opendatagroup.com
 - Augustus, an open source system for building & scoring statistical models
 - augustus.googlecode.com
 - PMML
 - Data Mining Group: dmg.org
 - Analytics over Clouds using Hadoop, presentation at Chicago Hadoop User Group:
http://files.meetup.com/1634302/CHUG_20100721_Sawmill.pdf

References

- Ricardo
 - Ricardo: Integrating R and Hadoop, paper:
<http://www.cs.ucsb.edu/~sudipto/papers/sigmod2010-das.pdf>
 - Ricardo: Integrating R and Hadoop, Powerpoint presentation:
<http://www.uweb.ucsb.edu/~sudipto/talks/Ricardo-SIGMOD10.pptx>

References

- General references on Hadoop and R
 - Pete Skomoroch's R and Hadoop bookmarks:
<http://www.delicious.com/pskomoroch/R+hadoop>
 - Pigs, Bees, and Elephants: A Comparison of Eight MapReduce Languages:
<http://www.dataspora.com/2011/04/pigs-bees-and-elephants-a-comparison-of-eight-mapreduce-languages/>
 - Quora – How can R and Hadoop be used together?:
<http://www.quora.com/How-can-R-and-Hadoop-be-used-together>

References

- Mahout
 - Mahout project: <http://mahout.apache.org/>
 - Mahout in Action, Owen, et. al., Manning Publications, 2011
- Python
 - Think Stats, Probability and Statistics for Programmers, Allen B. Downey, O' Reilly Press, 2011
- CRAN Task View: High-Performance and Parallel Computing with R, a set of resources compiled by Dirk Eddelbuettel:
<http://cran.r-project.org/web/views/HighPerformanceComputing.html>

References

- Other examples of airline data analysis with R:
 - A simple Big Data analysis using the RevoScaleR package in Revolution R:
<http://www.r-bloggers.com/a-simple-big-data-analysis-using-the-revoscaler-package-in-revolution-r/>

And finally...

Parallel R (working title), Q Ethan McCallum, Stephen
Weston, O'Reilly Press, due autumn 2011

“R meets Big Data - a basket of strategies to help you use R
for large-scale analysis and computation.”