Revisiting the MapReduce Paradigm: an R-Specific View

Norm Matloff and Alex Rumbaugh
University of California at Davis

UCB R Group
May 19, 2015
Updated, May 22

Norm Matloff and Alex Rumbaugh
University of California at Davis

Visual Summary

Norm Matloff and Alex Rumbaugh
University of California at Davis

Visual Summary

Norm Matloff and Alex Rumbaugh
University of California at Davis

Visual Summary

Norm Matloff and Alex Rumbaugh
University of California at Davis

Visual Summary

Norm Matloff and Alex Rumbaugh University of California at Davis

Visual Summary

My view: Plain Old R can work better in many situations

Norm Matloff and Alex Rumbaugh
University of California at Davis

Overview

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.
  - Not many ops.

Norm Matloff and Alex Rumbaugh
University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.
  - Not many ops.
- That time seems to have begun.

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.
  - Not many ops.
- That time seems to have begun.
- E.g. see *The Hadoop Honeymoon Is Over*, `http://smartdatacollective.com/martynjones/318406/hadoop-honeymoon-over`

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.
  - Not many ops.
- That time seems to have begun.
- E.g. see *The Hadoop Honeymoon Is Over*, `http://smartdatacollective.com/martynjones/318406/hadoop-honeymoon-over`
- There is a new kid on the block, Spark,

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.
  - Not many ops.
- That time seems to have begun.
- E.g. see *The Hadoop Honeymoon Is Over*, `http://smartdatacollective.com/martynjones/318406/hadoop-honeymoon-over`
- There is a new kid on the block, Spark, with an R interface, SparkR, a big improvement

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.
  - Not many ops.
- That time seems to have begun.
- E.g. see *The Hadoop Honeymoon Is Over*, `http://smartdatacollective.com/martynjones/318406/hadoop-honeymoon-over`
- There is a new kid on the block, Spark, with an R interface, SparkR, a big improvement
- But I will argue that for us R users, the utility of either Hadoop or SparkR is much more limited than many people realize.

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.
  - Not many ops.
- That time seems to have begun.
- E.g. see *The Hadoop Honeymoon Is Over*, `http://smartdatacollective.com/martynjones/318406/hadoop-honeymoon-over`
- There is a new kid on the block, Spark, with an R interface, SparkR, a big improvement
- But I will argue that for us R users, the utility of either Hadoop or SparkR is much more limited than many people realize.

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview

- When I was here one year ago, I speculated that Hadoop would start to lose popularity sometime in the future.
  - Too slow.
  - Not many ops.
- That time seems to have begun.
- E.g. see *The Hadoop Honeymoon Is Over*, `http://smartdatacollective.com/martynjones/318406/hadoop-honeymoon-over`
- There is a new kid on the block, Spark, with an R interface, SparkR, a big improvement
- But I will argue that for us R users, the utility of either Hadoop or SparkR is much more limited than many people realize.
- And I will present an alternative.

Norm Matloff and Alex Rumbaugh
University of California at Davis

Overview of MapReduce

Overview of MapReduce

- Parallel ops (cluster or multicore).

Overview of MapReduce

- Parallel ops (cluster or multicore).
- Work flow: map/sort/reduce.

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview of MapReduce

- Parallel ops (cluster or multicore).
- Work flow: map/sort/reduce.
- Example: word count.

Norm Matloff and Alex Rumbaugh
University of California at Davis

Overview of MapReduce

- Parallel ops (cluster or multicore).
- Work flow: map/sort/reduce.
- Example: word count.
  - Map: Read a line, break into words, emit one record for each (with count 1).

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview of MapReduce

- Parallel ops (cluster or multicore).
- Work flow: map/sort/reduce.
- Example: word count.
  - Map: Read a line, break into words, emit one record for each (with count 1).
  - Sort by word.

Norm Matloff and Alex Rumbaugh University of California at Davis

Overview of MapReduce

- Parallel ops (cluster or multicore).
- Work flow: map/sort/reduce.
- Example: word count.
  - Map: Read a line, break into words, emit one record for each (with count 1).
  - Sort by word.
  - Get counts by adding all the 1s for each unique word.

Overview of MapReduce

- Parallel ops (cluster or multicore).
- Work flow: map/sort/reduce.
- Example: word count.
  - Map: Read a line, break into words, emit one record for each (with count 1).
  - Sort by word.
  - Get counts by adding all the 1s for each unique word.
- Most famous example: Hadoop.

Norm Matloff and Alex Rumbaugh
University of California at Davis

What's Wrong with Hadoop

Norm Matloff and Alex Rumbaugh
University of California at Davis

What's Wrong with Hadoop

- SLOW.

What's Wrong with Hadoop

- SLOW. Like an elephant. :-)
  - Input to code must come from disk, output must be written to disk.

What's Wrong with Hadoop

- SLOW. Like an elephant. :-)
  - Input to code must come from disk, output must be written to disk.
  - Awful for iterative algorithms.

What's Wrong with Hadoop

- SLOW. Like an elephant. :-)
  - Input to code must come from disk, output must be written to disk.
  - Awful for iterative algorithms.
  - Sort phase (*shuffle*) is performed even if one's algorithm doesn't need it.

What's Wrong with Hadoop

- SLOW. Like an elephant. :-)
  - Input to code must come from disk, output must be written to disk.
  - Awful for iterative algorithms.
  - Sort phase (*shuffle*) is performed even if one's algorithm doesn't need it.
- Difficult to install/configure.

What's Wrong with Hadoop

- SLOW. Like an elephant. :-)
  - Input to code must come from disk, output must be written to disk.
  - Awful for iterative algorithms.
  - Sort phase (*shuffle*) is performed even if one's algorithm doesn't need it.
- Difficult to install/configure. Not everyone is a systems expert.

What's Wrong with Hadoop

- SLOW. Like an elephant. :-)
  - Input to code must come from disk, output must be written to disk.
  - Awful for iterative algorithms.
  - Sort phase (*shuffle*) is performed even if one's algorithm doesn't need it.
- Difficult to install/configure. Not everyone is a systems expert. Even worse when also need to install R interface.

What's Wrong with Hadoop

- SLOW. Like an elephant. :-)
  - Input to code must come from disk, output must be written to disk.
  - Awful for iterative algorithms.
  - Sort phase (*shuffle*) is performed even if one's algorithm doesn't need it.
- Difficult to install/configure. Not everyone is a systems expert. Even worse when also need to install R interface.
- Map and reduce ops too low-level.

What's Wrong with Hadoop

- SLOW. Like an elephant. :-)
  - Input to code must come from disk, output must be written to disk.
  - Awful for iterative algorithms.
  - Sort phase (*shuffle*) is performed even if one's algorithm doesn't need it.
- Difficult to install/configure. Not everyone is a systems expert. Even worse when also need to install R interface.
- Map and reduce ops too low-level. "Build a house from matchsticks."

Norm Matloff and Alex Rumbaugh
University of California at Davis

What Hadoop Gets Right

Norm Matloff and Alex Rumbaugh University of California at Davis

What Hadoop Gets Right

- Distributed file system (HDFS).

What Hadoop Gets Right

- Distributed file system (HDFS).
- "Move the computation to the data," rather than *vice versa*.

What Hadoop Gets Right

- Distributed file system (HDFS).
- "Move the computation to the data," rather than *vice versa*.
- Thus reduce time-consuming network communication time.

What Hadoop Gets Right

- Distributed file system (HDFS).
- "Move the computation to the data," rather than *vice versa*.
- Thus reduce time-consuming network communication time.
- Redundancy/fault tolerance, very important if have a huge cluster.

Norm Matloff and Alex Rumbaugh
University of California at Davis

Spark

Spark

- Extended map/reduce paradigm.

Spark

- Extended map/reduce paradigm.
- Cacheability of intermediate results, i.e. no costly writes to disk.

Spark

- Extended map/reduce paradigm.
- Cacheability of intermediate results, i.e. no costly writes to disk.
- *Lazy* computation: programmer's several specified ops automatically combined into faster coalesced code.

Spark

- Extended map/reduce paradigm.
- Cacheability of intermediate results, i.e. no costly writes to disk.
- *Lazy* computation: programmer's several specified ops automatically combined into faster coalesced code.
- Shuffle often avoided.

Spark

- Extended map/reduce paradigm.
- Cacheability of intermediate results, i.e. no costly writes to disk.
- *Lazy* computation: programmer's several specified ops automatically combined into faster coalesced code.
- Shuffle often avoided.
- Runs on top of HDFS or other DFS,

Spark

- Extended map/reduce paradigm.
- Cacheability of intermediate results, i.e. no costly writes to disk.
- *Lazy* computation: programmer's several specified ops automatically combined into faster coalesced code.
- Shuffle often avoided.
- Runs on top of HDFS or other DFS, so retain "move the computation to the data" philosoophy.

Norm Matloff and Alex Rumbaugh University of California at Davis

Spark

- Extended map/reduce paradigm.
- Cacheability of intermediate results, i.e. no costly writes to disk.
- *Lazy* computation: programmer's several specified ops automatically combined into faster coalesced code.
- Shuffle often avoided.
- Runs on top of HDFS or other DFS, so retain "move the computation to the data" philosoophy.
- Typically way faster than Hadoop.

Norm Matloff and Alex Rumbaugh University of California at Davis

Spark

- Extended map/reduce paradigm.
- Cacheability of intermediate results, i.e. no costly writes to disk.
- *Lazy* computation: programmer's several specified ops automatically combined into faster coalesced code.
- Shuffle often avoided.
- Runs on top of HDFS or other DFS, so retain "move the computation to the data" philosoophy.
- Typically way faster than Hadoop.
- Has various high-level ops, not just map and reduce.

Norm Matloff and Alex Rumbaugh University of California at Davis

Spark

- Extended map/reduce paradigm.
- Cacheability of intermediate results, i.e. no costly writes to disk.
- *Lazy* computation: programmer's several specified ops automatically combined into faster coalesced code.
- Shuffle often avoided.
- Runs on top of HDFS or other DFS, so retain "move the computation to the data" philosoophy.
- Typically way faster than Hadoop.
- Has various high-level ops, not just map and reduce.
- Elegant, sophisticated fault-tolerance mechanism.

Norm Matloff and Alex Rumbaugh
University of California at Davis

Drawbacks to Spark

Norm Matloff and Alex Rumbaugh University of California at Davis

Drawbacks to Spark

- Still have installation/configuration headaches, even worse than Hadoop.

Drawbacks to Spark

- Still have installation/configuration headaches, even worse than Hadoop. Ditto for SparkR.

Drawbacks to Spark

- Still have installation/configuration headaches, even worse than Hadoop. Ditto for SparkR.
- High-level ops are abstract, steep learning curve.

Drawbacks to Spark

- Still have installation/configuration headaches, even worse than Hadoop. Ditto for SparkR.
- High-level ops are abstract, steep learning curve. (Where have we heard that before?)

Norm Matloff and Alex Rumbaugh University of California at Davis

Drawbacks to Spark

- Still have installation/configuration headaches, even worse than Hadoop. Ditto for SparkR.
- High-level ops are abstract, steep learning curve. (Where have we heard that before?)
- Not clear that SparkR has much advantage over Plain Old R (POR).

Drawbacks to Spark

- Still have installation/configuration headaches, even worse than Hadoop. Ditto for SparkR.
- High-level ops are abstract, steep learning curve. (Where have we heard that before?)
- Not clear that SparkR has much advantage over Plain Old R (POR). See next slide.

Norm Matloff and Alex Rumbaugh
University of California at Davis

What's in It for R Users?

Norm Matloff and Alex Rumbaugh University of California at Davis

What's in It for R Users?

- Granted, Hadoop/Spark have automatic fault tolerance, and an efficient sort, both important.

Norm Matloff and Alex Rumbaugh University of California at Davis

What's in It for R Users?

- Granted, Hadoop/Spark have automatic fault tolerance, and an efficient sort, both important.
- But many apps don't need a sort, and many Hadoop users have small clusters (Hadoop Wiki, `https://wiki.apache.org/hadoop/PoweredBy`).

Norm Matloff and Alex Rumbaugh University of California at Davis

What's in It for R Users?

- Granted, Hadoop/Spark have automatic fault tolerance, and an efficient sort, both important.
- But many apps don't need a sort, and many Hadoop users have small clusters (Hadoop Wiki, `https://wiki.apache.org/hadoop/PoweredBy`).
- So, POR seems preferable for many users.

Norm Matloff and Alex Rumbaugh University of California at Davis

What's in It for R Users?

- Granted, Hadoop/Spark have automatic fault tolerance, and an efficient sort, both important.
- But many apps don't need a sort, and many Hadoop users have small clusters (Hadoop Wiki, `https://wiki.apache.org/hadoop/PoweredBy`).
- So, POR seems preferable for many users.
  - No Java/database/configuration issues.

Norm Matloff and Alex Rumbaugh
University of California at Davis

What's in It for R Users?

- Granted, Hadoop/Spark have automatic fault tolerance, and an efficient sort, both important.
- But many apps don't need a sort, and many Hadoop users have small clusters (Hadoop Wiki, `https://wiki.apache.org/hadoop/PoweredBy`).
- So, POR seems preferable for many users.
  - No Java/database/configuration issues.
  - No need to learn new abstractions.

What's in It for R Users?

- Granted, Hadoop/Spark have automatic fault tolerance, and an efficient sort, both important.
- But many apps don't need a sort, and many Hadoop users have small clusters (Hadoop Wiki, `https://wiki.apache.org/hadoop/PoweredBy`).
- So, POR seems preferable for many users.
  - No Java/database/configuration issues.
  - No need to learn new abstractions.
  - No forced shuffle.

What's in It for R Users?

- Granted, Hadoop/Spark have automatic fault tolerance, and an efficient sort, both important.
- But many apps don't need a sort, and many Hadoop users have small clusters (Hadoop Wiki, `https://wiki.apache.org/hadoop/PoweredBy`).
- So, POR seems preferable for many users.
  - No Java/database/configuration issues.
  - No need to learn new abstractions.
  - No forced shuffle.
  - POR at least as expressive as SparkR, and already familiar.

Norm Matloff and Alex Rumbaugh University of California at Davis

What's in It for R Users?

- Granted, Hadoop/Spark have automatic fault tolerance, and an efficient sort, both important.
- But many apps don't need a sort, and many Hadoop users have small clusters (Hadoop Wiki, `https://wiki.apache.org/hadoop/PoweredBy`).
- So, POR seems preferable for many users.
  - No Java/database/configuration issues.
  - No need to learn new abstractions.
  - No forced shuffle.
  - POR at least as expressive as SparkR, and already familiar.
- We've developed Snowdoop as an alternative:

Norm Matloff and Alex Rumbaugh
University of California at Davis

Snowdoop Overview

Norm Matloff and Alex Rumbaugh
University of California at Davis

Snowdoop Overview

- Pure POR!

Norm Matloff and Alex Rumbaugh University of California at Davis

Snowdoop Overview

- Pure POR! Uses the portion of R's **parallel** package adapted from the old **Snow**.

Norm Matloff and Alex Rumbaugh University of California at Davis

Snowdoop Overview

- Pure POR! Uses the portion of R's **parallel** package adapted from the old **Snow**.
- Retains the DFS philosoophy.

Snowdoop Overview

- Pure POR! Uses the portion of R's **parallel** package adapted from the old **Snow**.
- Retains the DFS philosoophy.
- Includes a distributed sort; could be optimized.

Snowdoop Overview

- Pure POR! Uses the portion of R's **parallel** package adapted from the old **Snow**.
- Retains the DFS philosoophy.
- Includes a distributed sort; could be optimized.
- So simple, it's embarrassing, tough to call it a "package."

Norm Matloff and Alex Rumbaugh
University of California at Davis

Word Count

Word Count

```
# prep: use SD filesplit(), load SD at nodes

# ndigs is number of digists in file suffix

fullwordcount <- function(cls,basename,ndigs) {
   counts <- clusterCall(cls,wordcensus,basename,ndigs)
   addlistssum <- function(lst1,lst2)
      addlists(lst1,lst2,sum) # SD
   Reduce(addlistssum,counts)
}

wordcensus <- function(basename,ndigs) {
   fname <- filechunkname(basename,ndigs) # SD
   words <- scan(fname,what="")
   tapply(words,words,length, simplify=FALSE)
}
```

Norm Matloff and Alex Rumbaugh
University of California at Davis

Word Count, cont'd.

Norm Matloff and Alex Rumbaugh University of California at Davis

Word Count, cont'd.

Test:

Norm Matloff and Alex Rumbaugh University of California at Davis

Word Count, cont'd.

Test:

```
library(partools)
cls <- makeCluster(2)
setclsinfo(cls) # SD
clusterEvalQ(cls,library(partools))
filesplit(cls,"x") # SD
fullwordcount(cls,"x",1) # SD
```

Norm Matloff and Alex Rumbaugh
University of California at Davis

Output

Norm Matloff and Alex Rumbaugh University of California at Davis

Output

```
> fullwordcount(cls,"x",1)
$a
[1] 2

$chuck
[1] 2

$Could
[1] 2

$How
[1] 1

$much
[1] 1

$wood
[1] 1

$woodchuck
[1] 2

$If
[1] 1

$`Wood?`
[1] 1
```

Norm Matloff and Alex Rumbaugh
University of California at Davis

Snowdoop vs. Hadoop Speed

Norm Matloff and Alex Rumbaugh University of California at Davis

Snowdoop vs. Hadoop Speed

Disclaimer: No serious experiments done yet, just some small, very preliminary simulations.

Snowdoop vs. Hadoop Speed

Disclaimer: No serious experiments done yet, just some small, very preliminary simulations.

k-Means Clustering:

Snowdoop vs. Hadoop Speed

Disclaimer: No serious experiments done yet, just some small, very preliminary simulations.

k-Means Clustering:

- Antonio, author of the R-Hadoop interface **rmr**, told me that the k-Means example is "just an example," not claimed to be fast.

Snowdoop vs. Hadoop Speed

Disclaimer: No serious experiments done yet, just some small, very preliminary simulations.

k-Means Clustering:

- Antonio, author of the R-Hadoop interface **rmr**, told me that the k-Means example is "just an example," not claimed to be fast.
- Our simulations (though not for large n, only 10^5) indicate that Snowdoop is about 100X faster than Hadoop.

Snowdoop vs. Hadoop Speed

Disclaimer: No serious experiments done yet, just some small, very preliminary simulations.

k-Means Clustering:

- Antonio, author of the R-Hadoop interface **rmr**, told me that the k-Means example is "just an example," not claimed to be fast.
- Our simulations (though not for large n, only 10^5) indicate that Snowdoop is about 100X faster than Hadoop.
- Same simulations show Snowdoop gives about a 50% speedup over R's serial **kmeans()** function, with 5 nodes.

Snowdoop vs. Hadoop Speed

Disclaimer: No serious experiments done yet, just some small, very preliminary simulations.

k-Means Clustering:

- Antonio, author of the R-Hadoop interface **rmr**, told me that the k-Means example is "just an example," not claimed to be fast.
- Our simulations (though not for large n, only 10^5) indicate that Snowdoop is about 100X faster than Hadoop.
- Same simulations show Snowdoop gives about a 50% speedup over R's serial **kmeans()** function, with 5 nodes. Note: **kmeans()** is written in C, not R.

Snowdoop vs. Hadoop Speed

Disclaimer: No serious experiments done yet, just some small, very preliminary simulations.

k-Means Clustering:

- Antonio, author of the R-Hadoop interface **rmr**, told me that the k-Means example is "just an example," not claimed to be fast.
- Our simulations (though not for large n, only 10^5) indicate that Snowdoop is about 100X faster than Hadoop.
- Same simulations show Snowdoop gives about a 50% speedup over R's serial **kmeans()** function, with 5 nodes. Note: **kmeans()** is written in C, not R.

sorting:

- This should be Hadoop's forte'.

Snowdoop vs. Hadoop Speed

Disclaimer: No serious experiments done yet, just some small, very preliminary simulations.

k-Means Clustering:

- Antonio, author of the R-Hadoop interface **rmr**, told me that the k-Means example is "just an example," not claimed to be fast.
- Our simulations (though not for large n, only 10^5) indicate that Snowdoop is about 100X faster than Hadoop.
- Same simulations show Snowdoop gives about a 50% speedup over R's serial **kmeans()** function, with 5 nodes. Note: **kmeans()** is written in C, not R.

sorting:

- This should be Hadoop's forte'.
- Yet we are finding Snowdoop 2X faster.

Norm Matloff and Alex Rumbaugh
University of California at Davis

Snowdoop vs. SparkR Speed

Snowdoop vs. SparkR Speed

- Haven't done a comparison yet.

Snowdoop vs. SparkR Speed

- Haven't done a comparison yet.
- But SparkR has no right to be faster than Snowdoop for nonsort apps.

Snowdoop vs. SparkR Speed

- Haven't done a comparison yet.
- But SparkR has no right to be faster than Snowdoop for nonsort apps.
- None of the apps in MLlib, Spark's machine learning (aka statistics) library uses sorting.

Snowdoop vs. SparkR Speed

- Haven't done a comparison yet.
- But SparkR has no right to be faster than Snowdoop for nonsort apps.
- None of the apps in MLlib, Spark's machine learning (aka statistics) library uses sorting.
- A BARUG speaker from Spark said that they haven't compared timings to non-MapReduce platforms.

Snowdoop vs. SparkR Speed

- Haven't done a comparison yet.
- But SparkR has no right to be faster than Snowdoop for nonsort apps.
- None of the apps in MLlib, Spark's machine learning (aka statistics) library uses sorting.
- A BARUG speaker from Spark said that they haven't compared timings to non-MapReduce platforms.
- With both Hadoop and Spark, is it really a matter of "It's not important how well the dog could walk on his hind legs, but that he could do it at all"?

Snowdoop vs. SparkR Speed

- Haven't done a comparison yet.
- But SparkR has no right to be faster than Snowdoop for nonsort apps.
- None of the apps in MLlib, Spark's machine learning (aka statistics) library uses sorting.
- A BARUG speaker from Spark said that they haven't compared timings to non-MapReduce platforms.
- With both Hadoop and Spark, is it really a matter of "It's not important how well the dog could walk on his hind legs, but that he could do it at all"?

Stay tuned.

Norm Matloff and Alex Rumbaugh
University of California at Davis

Obtaining Snowdoop

Norm Matloff and Alex Rumbaugh University of California at Davis

Obtaining Snowdoop

- Still under development.

Norm Matloff and Alex Rumbaugh University of California at Davis

Obtaining Snowdoop

- Still under development.
- In our **partools** package, on CRAN and GitHub.