Divide & Recombine for Large Complex Data (a.k.a. Big Data)

Statistical framework requiring research in statistical theory and methods to make it work optimally

Framework is designed to make computation very simple, mostly embarrassingly parallel

Leads to a D&R computational environment

- R at the front end and
- the Hadoop distributed file system (HDFS) and parallel compute engine (MapReduce)

Divide & Recombine for Large Complex Data (a.k.a. Big Data): Goals

Provide the data analyst with statistical methods and a computational environment that enable deep study of large complex data. This requires application of analytic methods to the data at their finest level of granularity, and not just to summary statistics of the detailed data. This must include visualization methods.

The analyst uses, at the front end, an interactive language for data analysis that is powerful, so analysis can be tailored to the data and not just canned, and that enables highly time-efficient programming with the data.

At the back end, a distributed database and parallel compute engine running on a cluster that makes computation feasible and practical, and is easily addressable from within the language. The back end must provide a mechanism for fair sharing of the cluster resource by multiple analysts.

Within the environment, access to the 1000s of methods of statistics, machine learning, and visualization.

Write software packages that enable communication between front and back so that a variety of back-ends can be readily plugged in.

Use the system continuously to analyze large complex data in settings where results are judged by subject matter findings. This serves as a heat engine for generating new research ideas and for testing them.

D&R Statistics

The D&R Framework

Two types of analytic methods applied to subsets are treated differently

1. Number-category methods
   - outputs are numeric and categorical
   - e.g., logistic regression

2. Visualization methods
   - outputs are plots
   - e.g., scatterplot matrices

The D&R Framework

Division

- a **division method** divides the data into subsets
- a division persists and is used for many analytic methods

Number-category analytic methods are applied to each of the subsets

- no communication between the computations
- **embarrassingly parallel**

Visualization analytic methods are applied to each subset in a sample of subsets

- typically too many to look at plots for all subsets
- compute between-subset variables with one value per subset to enable rigorous sampling plans
- sampling plans: representative, focused, cognostics thanks to JWT

For each analytic method

- **recombination** method: subset outputs recombined, providing the D&R result for the analytic method
- often has a component of **embarrassingly parallel computation**
- there are many potential recombination methods, individually for analytic methods, and for classes of analytic methods

Computationally, this is a very simple.

Conditioning-Variable Division

In very many cases, it is natural to break up the data based on the subject matter in a way that would be done whatever the size

Data are embarrassingly divisible

Example

- 25 years of 100 daily financial variables for 10^5 banks in the U.S.
- division by bank
- bank is a conditioning variable

There can be multiple conditioning variables that form the division

The major division method used in our own analyses.

Conditioning-variable division has already been widely used in statistics, machine learning, and visualization for datasets of all sizes.

Analytic Recombination

For conditioning-variable division, typically the recombination depends mostly on the subject matter

Analytic recombination: a further analysis of the analytic method outputs

Example: hierarchical modeling

- subsets each with the same model that has parameters
- parameters are modeled as stochastic too: independent draws from a distribution
- recombination: analysis to build statistical model for the parameters using the subset estimated coefficients

Replicate Division

Observations are seen as exchangeable, with no conditioning variables considered

Division methods based on statistical matters, not the subject matter as in conditioning-variable division

Replicate Division

Consider logistic regression

- $n = 2^{30}$ observations = 1 gigaob
- $p = 2^7 - 1 = 127$ explanatory variables, all numeric
- one response of 0's and 1's
- $v = p + 1 = 2^7 = 128$ variables altogether
- memory size of each value is 8 bytes
- memory size of dataset is 2^{40} bytes = 1 terabyte

Suppose each subset has m observations, where $r = n/m$ is an integer

r subsets, so r logistic regressions

Outputs: r p-vectors of estimates of regression coefficients and associated statistical information

Give the outputs to a recombination method

Statistical Accuracy for Replicate Division

The statistical division and recombination methods have an immense impact on the accuracy of the D&R result

Examples for the logistic regression above

- division method: random
- recombination method: weighted by the inverses of estimates of the covariance matrices of the subset regression coefficient estimates

These are obvious choices; we know we can do better

AND, we need along with the final D&R estimates, characterizations of the statistical accuracy

We can do this, too, right now for the class of analytic methods with a certain structure like that of logistic regression

Statistical Accuracy for Replicate Division

The D&R result is not the same as that of the application of the method directly to all of the data

D&R research in statistical theory seeks to maximize the accuracy of D&R results

The statistical accuracy of the D&R result is typically less that that of the direct all-data result

Our results so far show that this is a small penalty to pay for the very simple, fast D&R computation that touches subsets just once for each analytic method.

D&R Computational Environment

D&R Computational Environment: Front End

R

Elegant design makes programming with the data very efficient

Saves the analyst time, which is more important than computer time

Very powerful, allowing the analyst to readily tailor the analysis to the data

Access to 1000s of analytic methods of statistics, machine learning, and visualization

Very large supporting community

Real big user community

D&R computational environment inherits this.

Back End

Hadoop running on a cluster

Distributed file system (HDFS)

Distributed parallel compute engine (MapReduce)

What the Analyst Specifies in R

D[dr], A[dr], and R[dr] Computations

D[dr]

- division method to divide the data into subsets using a division method
- structure of the R objects that hold the data and are written to disk

A[dr]

- analytic methods applied to each subset (number-category) or to each in a sample (visualization)
- structure of the R objects that hold the outputs

R[dr]

- for each applied analytic method, the recombination method and the structure of the R objects that hold the D&R result

What Hadoop Does with the Analyst R Commands

D[dr]
Computes subsets, forms R objects that contain them, and writes the R objects across the nodes of the cluster into the HDFS

A[dr]
Applies the analytic method to subsets in parallel on the cores of the cluster with no communication among subset computations, and if there is no accompanying recombination, writes the output to the HDFS

R[dr]
Takes outputs of the A[dr] computations and carries out the recombination method across them with the computation, and writes the results to the HDFS

The Hadoop Scheduler Enables Sharing Cluster

A[dr] computations run by mappers, each with a core

R[dr] computations run by reducers, each with a core

D[dr] typically uses mappers and reducers

These mappers and reducers are running micro-computations

If there is a single analyst, when one micro-computation ends, Hadoop schedules another

If analyst 2 appears, Hadoop begins scheduling cores to the micro-computations of analyst 2 as those for analyst 1 end until usage is fair

This is a beautiful thing

Three D&R Software Packages Between Front and Back Ends

What is this does

- provides communication between R and Hadoop
- makes D&R programming easy
- protects the analyst from the details of Hadoop computation

RHIPE

datadr

Trelliscope

The whole environment

linux, Hadoop, protocol buffers, datadr, Trelliscope RHIPE, R

It is all open source

RHIPE: The R and Hadoop Integrated Programming Environment

Merger of R and Hadoop that uses Hadoop streaming

Pronounced: hree-pay′

“In a moment” in Greek

Analyst gives R code for D[dr], A[dr], and R[dr] computations to RHIPE R functions, which then manage communication with Hadoop

First written by Saptarshi Guha in 2010 while a Purdue Stat PhD student

An R package available on Github (not CRAN)

Needs protocol buffers

It is all open source

datadr

Provides a simpler interface for

- division
- applying the analytic method
- recombination
- other data operations such as sample, filter, and join

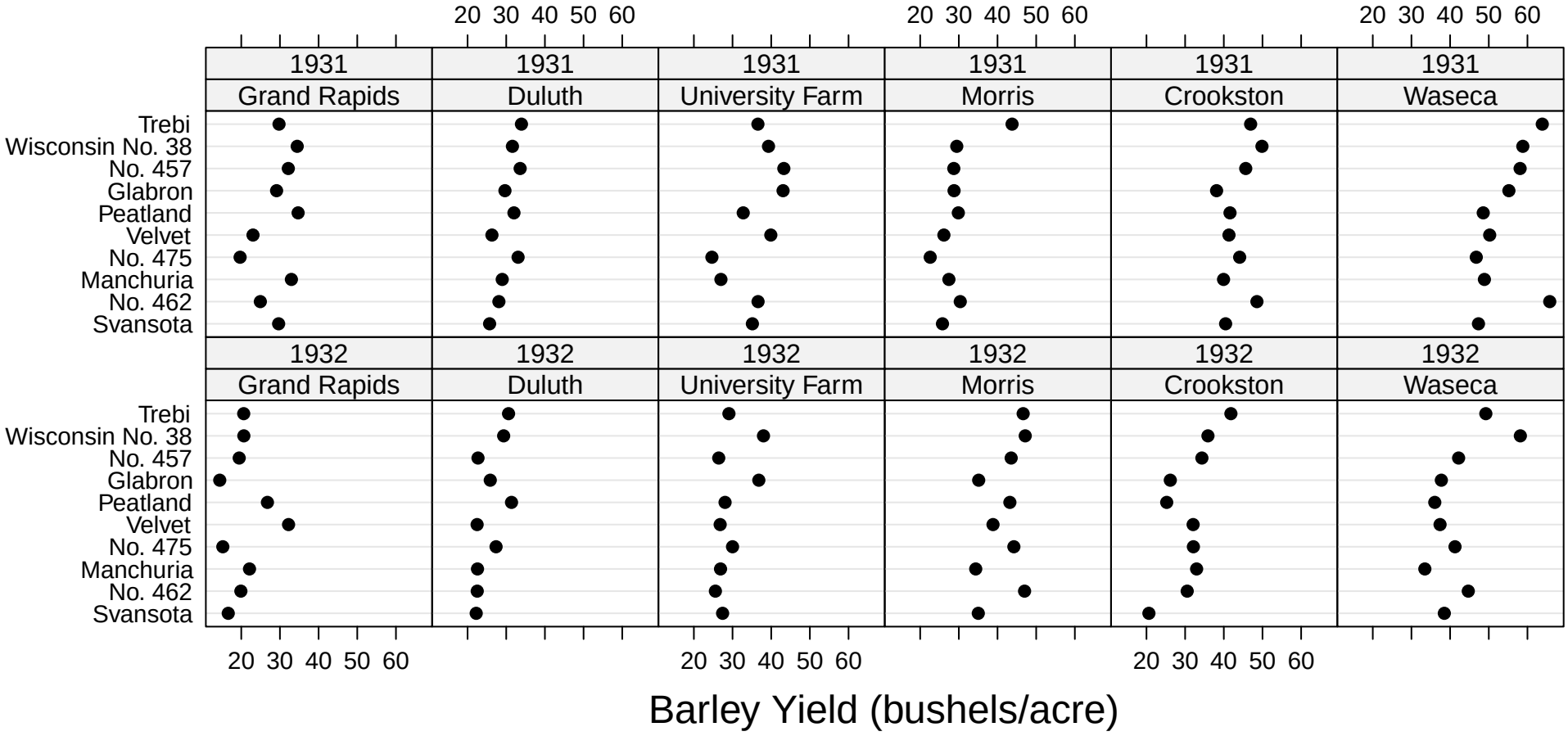
First written by Ryan Hafen at PNNL

Comes with a generic MapReduce interface

- fundamental data structure is key-value pairs
- designed to be easily extensible to any distributed key-value store and MapReduce system
- can support key-value pairs stored in memory, on local disk, or on an HDFS using `RHIPE`

Trelliscope

The trellis display visualization framework, implemented in R as the lattice graphics package, is very powerful because it is based on conditioning-variable division, good for data big and small.



Trelliscope

Extends trellis display to large complex data

Like `datadr`, is between `R` and `RHIPE`

Display panels have sampled subsets from rigorous sampling plans as described above

First written by Ryan Hafen at PNNL

How Fast is It?

Consider the above logistic regression

Observations $n = 2^{30}$

Variables $v = p + 1 = 2^7$

Number of subsets $r = 2^{20}$

1 TB of data

The subset logistic regressions were carried out in using the `R` function `glm.fit`.

The Cluster

11 nodes, each a Hewlett Packard ProLiant DL165 G7

Each node has

- Dual 2.1 GHz 12-core AMD Opteron 6172 processors (24 cores)
- 22 cores for Hadoop computations and 2 for the work in the R global environment used directly by users.
- 48 GB RAM
- 4x2 TB 7200 RPM SATA disks
- 10 Gbps Ethernet interconnect

Collectively, the 11 notes have

- $24 \times 11 = 264$ cores
- $22 \times 11 = 242$ Hadoop cores
- $48 \times 11 = 528$ GB total RAM
- $8 \times 11 = 88$ TB total disk

This is by today's standards a modest cluster. The disk is not particularly fast, a financial decision to have more disk with less speed.

1 TB data size about 2 times total cluster memory size.

How Fast is It?

Total elapsed time of the computation was 17.6 min

70% of this time was simply reading the subsets into memory with R

So `glm.fit` computations are 5.3 min

Easy way to drive down the total time would be to buy faster disk

The User Must Attend to Subset Size

Serial computation for the analytic methods of the analysis must be feasible and practical

Sizes best off being fraction of HDFS block size which for us is 128 MB

Very large number of small subsets makes Hadoop bookkeeping more time consuming

Must be careful of memory size of simultaneous subset computations

We are running multifactor designed experiments to optimize

- subset sizes
- Hadoop configuration parameters
- hardware design such as disk per core and server network speed

Benchmarks and one-factor at a time experiments of current Hadoop community inadequate.

Target Audience

Who can benefit from D&R computational environment?

“big data” and “large complex data” have no real meaning

It is all relative to the computational environment

Answer

You have cluster quite small or quite big

You install D&R environment

Sizes of the datasets on which you can carry out deep analysis can increase

Speed of computation for data sets you can analyze can increase

Increases can be dramatic

You can analyze datasets whose memory size is a lot bigger than your total cluster memory

D&R Team and Dissemination

The D&R Team

Purdue

- 4 faculty, 10 graduate students
- D&R statistics research
- D&R top-level computational environment
- D&R courses
- analyze large complex data.
- DARPA XDATA big data grant

PNNL

- Ryan Hafen
- D&R computational environment, front to back
- 2 Java developers
- 5 scientists doing D&R analyses and providing feedback
- bringing D&R to government organizations such as DHS
- DARPA XDATA big data grant

Mozilla

- Saptarshi Guha
- RHIPE
- analyzing large complex data.

Dissemination

R, RHIPE, and Trelliscope R packages on Github

datadr.org

Publications

Documentation

Building cloud service appliances, AWS and Google

DARPA XDATA program 8 week summer boot camp

- all out effort in technology transfer

Seeking to develop a consortium

- there is strength in numbers
- academic statistics departments
- industry analytic groups

Consortium info: wsc@purdue.edu

Tessera