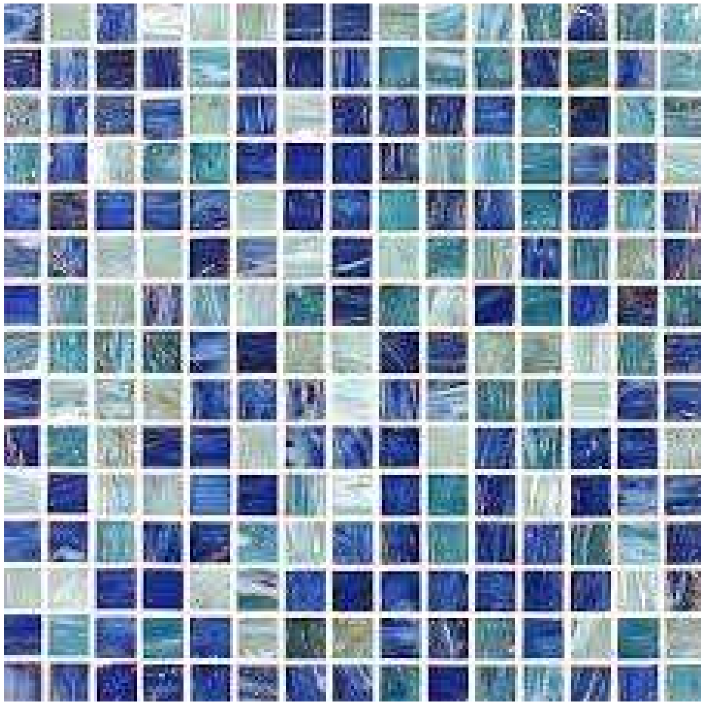
Divide & Recombine (D&R) with Tessera: High Performance Computing for Data Analysis



www.tessera.io

Big Data?

A big term

An associated concept

- new computational methods and systems are needed to make computational performance feasible and practical

Data analysis

- there is a tremendous demand to carry it out for big data

Big Data?

“big data” is an insuffient term for data analysis

Computational performance for data analysis also depends very heavily on other factors

(1) Computational complexity of the data analytic methods used in the analysis

- datasets that have big computational complexity challenges have a very wide range of sizes, from small to big

(2) Hardware power available to the data analyst also a critical factor

- more hardware power typically yields better computational performance

We need more than just a more powerful cluster to succeed

A most critical matter is that we need to compute in parallel

High Performance Computing for Data Analysis (HPC-DA)

Divide & Recombine (D&R) with Tessera

D&R is a statistical approach

The Tessera D&R software system

- makes programming D&R easy
- provides for parallel distributed computing
- protects the data analyst from details of parallel computing

HPC-DA provided for wide ranges of

- dataset size, big and small
- computational complexity
- hardware power

Deep Analysis

Millions of data analysts around the world

Working in fields where data analysis is needed

Seeking to analyze the data comprehensively, and in detail at their finest granularity

Scientific discoveries

Engineering design and performance

Collectively, thousands of analytic methods are being used

- statistics
- machine leaning
- data visualization

The D&R Framework

Statistical Division

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

Analytic Methods

- each method applied by the analyst to each of the subsets
- no communication among the subset analytic-method computations
- embarrassingly parallel, the simplest parallel computation

Statistical Recombination

- for each method, statistical recombination method applied to subset outputs
- provides a D&R result for the method
- often has a component of embarrassingly parallel computation
- very general, e.g., applies to visualization methods

Computationally, this is a very simple.

Tessera Front End: datadr R package

The analyst programs in R and uses the datadr package

A language for D&R

First written by Ryan Hafen at PNNL (former grad student in Purdue Statistics)

1st implementation Jan 2013

Analyst R code specifies divisions, analytic methods to be used, and recombinations

Tessera Back End

A distributed parallel computational environment running on a cluster

So far, backend has been Hadoop

Hadoop runs the analyst’s R code for divisions, analytic methods, and recombinations

As part of this, Hadoop writes subsets and outputs of analytic methods to the HDFS as specified in the analyst R code

datadr is written to be back-end agnostic, so there are other possibilities for back ends

RHIPE: The R and Hadoop Integrated Programming Environment

Provides communication between datadr and Hadoop

Also provides programming of D&R but at a lower level than datadr

First written by Saptarshi Guha while a grad student in Purdue Statistics

1st implementation Jan 2009

Why R?

Provides analyst with very powerful, extensible language for data analysis

Elegant design makes programming with the data very efficient

Makes available vast number of statistical, machine learning, and visualization methods, more than any other system

Open source

Parent S won 1999 ACM Software System Award joining pretty good company like Unix, VisiCalc, Make, TCP/IP, World Wide Web, Java, . . .

“John M. Chambers. For The S system, which has forever altered how people analyze, visualize, and manipulate data.”

Conditioning-Variable Division

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

Divide by conditioning on the values of variables important to the analysis

Based on subject matter knowledge

Example

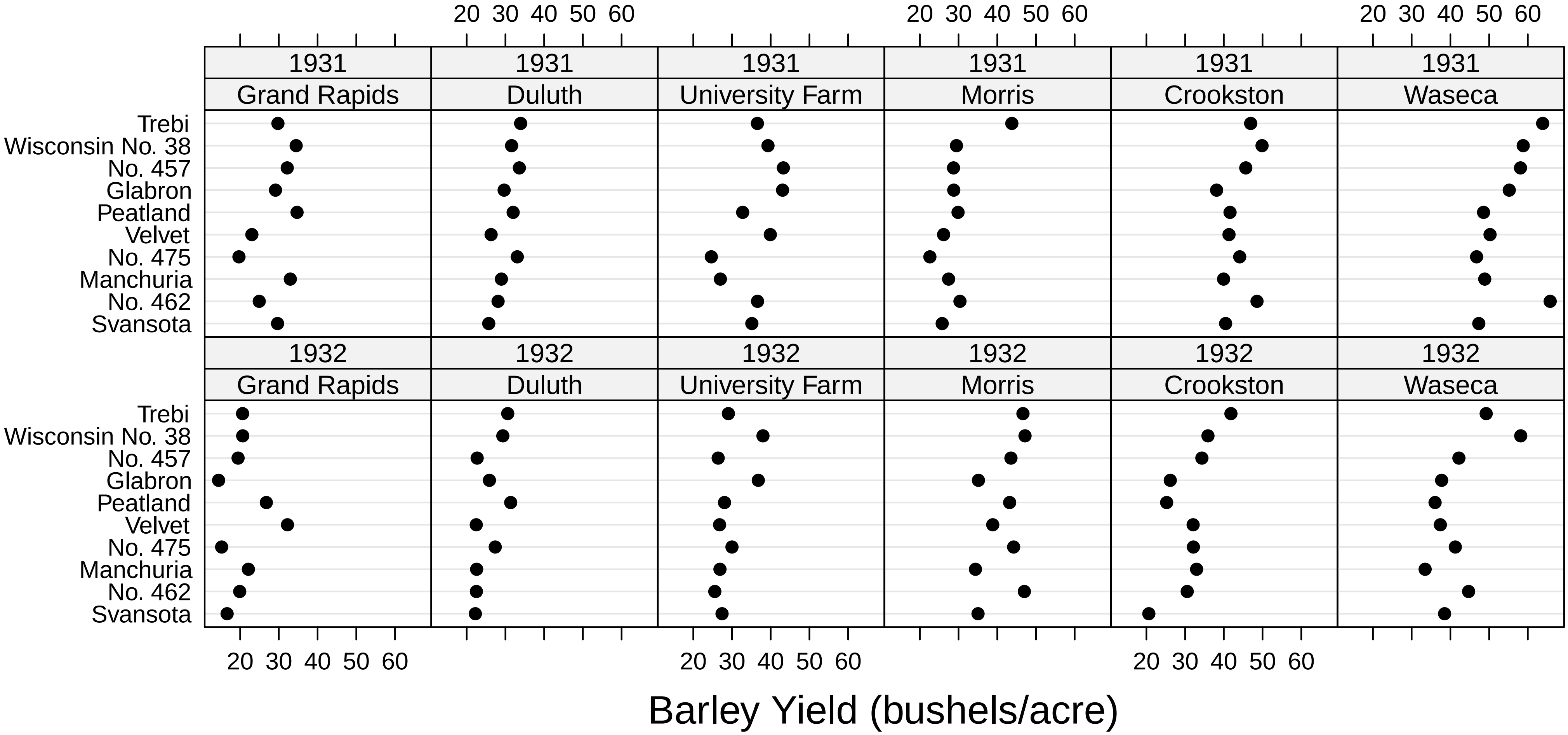
- 25 years of 100 daily financial variables for 10, 000 banks in the U.S.
- division by bank
- bank is a conditioning variable

Just as valid for small datasets

- widely practiced in the past
- a statistical best practice

D&R with Tessera takes advantage of this best practice for computational gain as well

Barley Yield vs. Variety Conditional on Year and Site



Cyber Security: Spamhaus

Spamhaus classifies IP addresses and domain names as blacklisted or not

Based on many sources of information and many factors such as being a major conduit for spam

Blacklisting information is used very widely, for example, by mail servers

Querying host sends query to Spamhaus to determine if an IP address or domain name is blacklisted

Spamhaus Data

We are collecting data from the Stanford mirror of the Spamhaus site at a raw data rate of 100 GB per week.

Current dataset has

- IP address queries for 8+ months: 10,615,054,608
- number of querying IP addresses: 982,293
- number of queried IP addresses: 207,081,108
- number of queried IP addresses with at least one blacklist result: 59,435,635

Our analyzed dataset has 13 variables for each IP address query

- e.g., timestamp, querying host IP address, queried host IP address at least one blacklist variable is blacklist or not, generic spam or not

Data Size and Hardware Memory Size

For the first division, dataset has 5.71 TB memory size of R objects in the Hadoop HDFS

Hadoop replicates the data for recovery from damage to disks: 3 copies of the data

Cluster for analysis: 315 GB memory

First Division: Conditioning-Variable

First D&R division: each query is a subset in terms of D&R analysis

This is conditioning-variable division, as are all divisions of the analysis

This means 13,178,080,366 subsets

Hadoop does not perform optimally when there are a very large number of small subsets, key-value pairs in Hadoop parlance

Need to optimize performance

First Division: Conditioning-Variable

Put q queries on a dataframe

- make each dataframe a Hadoop key-value pair, or in D&R parlance, a computational subset
- on each dataframe timestamp increases with row number
- dataframes ordered by timestamp, first one accessible in dataframe attribute
- Write R code for each computational subset to apply analytic method to each row of the dataframe

We ran performance tests and found 6,000 queries per dataframe was nearly optimal

Blacklisted Queried Host Analysis

Study all the queries of each queried IP address (queried host) with at least one query that has a blacklist result

Create a new division where each subset is data for each blacklisted queried host

Each subset is a marked point process: the point process is the query times and the marks for each time are 11 variables, down from 13

This is conditioning-variable division

Blacklisted Queried Host Analysis

Number of subsets: 59,435,635

Dataframe object for each subset has 11 columns and each row is a query

This time profile of the host is a marked point process

Consecutive blacklistings for the process are an “on” interval

Consecutive whiltelisting for the process are an “off” interval

We study the on-off process

Our ability to look at these blacklisted queried host time profiles in immense detail, has led to a big discovery

Sampling Division: The Concept

Each subset is seen as a sample of the data

Subsets are replicate samples, or replicates

For example, we can carry out random replicate division: choose subsets randomly

We seek a single result for all of the data

One place this arises is when subsets from conditioning variable division are too large

Statistical Accuracy for Sampling Division

There is a statistical division method and a statistical recombination method

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

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

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

The accuracy depends on the division method and the recombination method

A community of researchers in this area is developing

Another Approach to Sampling Division

Distributed parallel algorithms (DPAs) that compute on subsets

Like D&R, apply an analytic method to each subset

Unlike D&R, iterate and have communication among subset computations

A well-known one is ADMM (Alternating Direction Method of Multipliers)

Read data and apply analytic method at each iteration

DPAs vs. D&R

D&R has just one step, read and apply analytic method to each subset

DPAs do this at each iteration

Critical notion for computational performance of DPAs

- data are addressable in memory to make reads faster
- Spark has this capability

All data in memory

- this is a major limitation
- D&R with Tessera allows study of data whose memory size is greater than physical memory
- those millions of deep analysts are hardware poor, relative to the data size (including our D&R with Tessera team)

Accuracy

- distributed parallel algorithms seek numerical accuracy
- D&R seeks statistical accuracy

What Matters?

In practice, conditioning-variable division is the heavy hitter

Sampling division is important, well worth much research, but a distant second place holder

Conditioning-Variable Division: Recombination

Almost always, there is a analytic recombination: outputs are further analyzed

If outputs are collectively large in size and computational complexity, and challenge serial computation

- a further D&R analysis of outputs

If outputs are collectively smaller in size computational complexity, and can be analyzed serially

- written from HDFS to the analyst's R global environment on the R session server
- further analysis carried out there in serial
- this happens a lot

So D&R analysis is not just a series of Hadoop jobs

A significant amount of analysis is done in the classical R serial way

Visualization

Visualization of the detailed data at their finest granularity is critical, whatever the size of the data

A powerful way to understand the details of patterns in the data

This serves as guide to what models to try or machine learning methods to use

This has been done routinely for decades for analyses of smaller data

There are many examples where data are analyzed, and later someone uses visualization to shows the analysis missed important happenings in the data

This true for all datasets, small and big

Visualization for Large Complex Data

Some think large complex data are too big to visualize in detail at the outset

Do a data reduction first, and then visualize only the summary and not the detail

Visualization of summaries is very useful indeed, but not close to enough

This is a step backwards

A surrendering to the size and complexity of the data

How are we going to pull off detailed visualization?

Put our statistician hats on

D&R Visualization

Visualization analytic method is applied at the subset level

Get to see the detail for the subset

The number of subsets typically too large to look at plots for all of them

So the visualization method is applied to a sample of subsets

Such sampling can be very powerful and rigorous

We can readily compute variables, each with one value per subset, to enable rigorous sampling plans

A single Trellis display with one subset per panel, many pages, and many panels per page is a visualization recombination

datadr

Interface is abstracted from the different back end choices, so that commands are the same whatever the back end

This enables datadr to be the D&R domain specific language for back ends other than Hadoop

Large distributed data objects behave like native R objects

Tools for division-independent methods that can compute things like aggregations, quantiles, or summaries across the entire dataset

The Tessera Software Chain for Hadoop

R/datadr ↔ RHIPE ↔ Hadoop

Tremendous effort has gone into optimizing computational performance for this chain

Optimizing D&R programming performance is at least as important

datadr

It is a programming language for D&R designed and implemented by Ryan Hafen

Beautiful design enables very efficient programming of D&R by the data analyst

It is the key element in the chain

What was needed to develop datadr

- knowledge of computational systems sitting behind it
- intuition and knowledge about what the data analyst needs

D&R Tessera Team and Data Analyses

The D&R Tessera team is now about 25:

- PNNL: statisticians and computer scientists
- Purdue: Statistics faculty and students
- Hafen Consulting LLC: Ryan Hafen
- Mozilla: Saptarshi Guha

Since 2009, data analysis projects have been an integral part of what we now call Tessera

There many data analysis projects carried out by team members

The data analyses lead the way: heat engine for ideas, and a test bed

RHIPE got started in 2009 to analyze packet-level data on millions of SSH connections