Divide & Recombine (D&R), RHIPE, and Riposte for Large Complex Data

Statistics, Purdue

Saptarshi Guha, now at Mozilla

Jin Xia, now at GE Research

Ryan Hafen, now at PNNL

Ashrith Barthur

Bill Cleveland

Jeff Li

Jeremiah Rounds

Bowei Xi

Yang Zhang

Computer Science, Stanford

Zack DeVito

John Gerth

Pat Hanrahan

Justin Talbot

Computer Science, Purdue

Jan Vitek

386 Gates

SriSatish Ambati

Large Complex Datasets

Ubiquitous today

Currently challenge everything involved in the analysis of data

- numeric methods of statistics and machine learning
- numeric models of statistics and machine learning
- visualization methods
- computational algorithms
- computational environments: hardware and software

Many references to issues, especially David Ebert, so far in FODAVA 2011 Annual Meeting

Defining the Community and the Analysis

Scientists, engineers, business analysts, etc.

Use of methods of statistics, machine learning, and visualization

Collectively, need to bring to bear the 1000s of methods of stat, ml, and vis

How can we develop an approach to analysis and a computational environment that serves them all

Two Goals

1. Deep Analysis

Definition

- comprehensive detailed analysis
- does not lose important information in the data

To achieve must use

- numeric methods of statistics and machine learning
- visualization methods
- need to visualize the detailed raw data through the whole analysis, starting with raw data

Two Goals

2. Interactive Language for Data Analysis (ILDA)

The data analyst and the methodologist developing methods for large data work within an ILDA

An example of a very effective ILDA is R

A very effectively designed ILDA provides very time-efficient programming for

- data analysis
- prototyping new methods and models

An ILDA can provide access to the 1000s of numeric and visualization methods, necessary for deep analysis

Size vs. Complexity

Sheer size, per se, is not the only challenge of large complex data to achieving deep analysis and using an ILDA

Size often not directly the challenge

With growing size typically comes a growing

- complexity of data structures
- patterns in the data
- the models needed to account for the patterns

It is the complexity that typically challenges methods, models, and computational environments

Data sets as small as 100s of gigabytes can challenge deep analysis

Achieving the Two Goals: Divide and Recombine (D&R)

D&R is an approach to the analysis of large complex data to meet the challenges

The data are parallelized: divided into subsets in one or more ways

Numeric and visualization methods are applied to each of the subsets separately

Then the results of each method are recombined across subsets

Computation is “embarrassingly parallel” and can be carried out on a cluster with nodes that can vary substantially in perfomance

Note: division + recombination is also an analysis “best practice”

- provides much insight, not just fast computation
- good for small data as well
- data are often embarrassingly divisible

D&R Enables Comprehensive Visualization

Visualization of the detailed raw data as well as summaries is critical for for deep analysis

Made feasible by using statistical sampling and experimental design

Choose a representative or focused sample of the subsets: apply visualization methods to each subset in the sample

The visualization is a visual recombination

Sampling guided by between-subset variables (each has one value per subset) that are computed

Can use the cognostics framework of Tukey to finds interesting subsets, which Lee Wilkinson has tailored to certain problems with great success

D&R Research

D&R research consists of the development of

- methods for division: statistics and machine learning
- methods for recombination: statistics and machine learning
- a new theory whose concept is optimality subject to the constraint of having to break up the data
- computational algorithms: computer science
- computational environments: distributed databases, distributed compute engines, cluster design, and interactive computer languages

Different data structures need different division and recombination methods

Today's Topic: Our Research in Computational Environments for D&R

Riposte and RHIPE

- novel software systems
- developed to meet the computational challenges
- make D&R computationally feasible

Brief overview of each now

For more detail, see the Riposte poster and the RHIPE poster

Hadoop Distributed Computational Environment

Distributed file system (HDFS)

Distributed compute engine (MapReduce)

Supported by Apache Software Foundation

Open source (Apache License)

History

- Google: released the architecture in a paper
- Yahoo: first public domain release of software
- Apache Software Foundation: signed on to develop and distribute

RHIPE: R and Hadoop Integrated Programming Environment

[hree-**pay**]: “in a moment” in Greek

First developed by Saptarshi Guha, formerly Purdue, now at Mozilla

Second member of core development team, Jeremiah Rounds, Purdue

A merger of R and Hadoop

Written in C, Java, and R

RHIPE: How It Works for Data Analysis or Methodology/Model Development

Program entirely within R

Saves time-costly programming in a lower-level language

RHIPE: How It Works for Data Analysis or Methodology/Model Development

Division: Creating Subsets in the HDFS

The data analyst or methodologist specifies subsets of the data using R commands

R code passed to RHIPE R commands, which create the subsets and tell Hadoop to write them to HDFS

In our work the number of subsets has ranged from 540 to 14,161,628

HDFS spreads subsets across the cluster in 128MB blocks transparent to user

These computations are elephants

RHIPE: How It Works for Data Analysis or Methodology/Model Development

Parallel Computing of R Code across Subsets

The data analyst or methodologist writes R code for a method to be applied to each subset

R code passed to RHIPE R commands that tell Hadoop to execute the code on each subset and write the results to te HDFS

Hadoop schedules subset computations optimally across cluster and writes results to HDFS, transparent to user

Results can stay in HDFS for further distributed computation

These computations are elephants

RECOMBINATION

Results from distributed computation that are data reductions often can be read into the standard R file system

Analyze the results using the traditional R computing

- the analyst or methodologist operates in a highly interactive mode
- issues sequences of commands
- many compute virtually instantaneously: output is needed virtually instantaneously

Recombination is almost always carried out by such computations

These computations are mice

A RHIPE R+Hadoop Computation

885,459,060 values of VoIP jitter

Subsets: 3,685,011 transmission intervals each with a varying number of values of jitter

For each transmission interval

- values of jitter are an ordered sequence J_i
- there is a cycle of 9 in the jitter whose pattern changes with the transmission interval
- to each subset, fitted cyclic regression to J_i: 9 predictor variables = 9 cosinusoids
- fitted by bisquare robust regression: least-squares fit + 3 weighted least squares fits
- computed fitted values and residuals of regressions
- computed maximum and minimum of the fitted values and then their maximum and minimum
- wrote the residuals and the maximum-minimum to the HDFS

Elapsed time = 12.50 min

Carried out on a modest cluster with about 60 processors whose age varied from 4 years to 1 year

The Delta-Rho Cluster Architecture

ELEPHANTS AND MICE

Today's standard high-performance massively-parallel computing environments

- not designed for elephant-mice computing tasks and do not work for them
- essentially batch environments

Developing a new parallel cluster architecture for D&R analysis of large complex data

Working with Purdue Rosen Center for Advanced Computation to develop Delta-Rho

No batch scheduling, thank you, because data analysis is a 24x7 occupation

The Hadoop scheduler allows sharing by intermingling subset computations of different users

Riposte: A Fast R Virtual Machine

[ri-**pohst**] “a quick, sharp return in speech or action; counter-stroke”

Justin Talbot, Stanford

Zack DeVito, Stanford

Pat Hanrahan, Stanford

R, like Javascript, is a very dynamic language

Many of the recent developments for Javascript can enable dramatic improvement in R performance

Testing has suggested 5-10 times performance improvement is possible

Riposte

New “from scratch” R virtual machine designed for performance

Key Design Insight

R contains two very different language components

- a high productivity scripting language (like JavaScript)
- a high performance vector language (like APL)

Riposte VM has 2 cooperative subVMs

- handle the scripting and vector portions of the language separately
- permits aggressive optimization of each

Riposte Performance

Early performance results

”Scripting-like” code

- 2-4x faster in Riposte than in R’s recent bytecode interpreter
- lots of low-hanging fruit left

Vector heavy code is 5-10x faster on a single core

Technology Transfer: RHIPE

Open source software, Apache license

R cran package: RHIPE

Google discussion group: groups.google.com/group/rhipe

Initial web site: rhipe.org

Code development site: code.google.com/p/rhipe

Technology Transfer: RHIPE + Riposte for D&R

Start-Up Company based on RHIPE+Riposte for D&R

Results of marketing survey of a companies, many in Silicon Valley

- very enthusiastic response to R+R
- today there are many, many companies whose competitive position depends heavily on the analysis of large complex data
- want to save analyst time

Two components: .org with open source, and .com with commercial products

Company name: 386 Gates (temporary, address of Justin and Zack in the Stanford CS building)

CEO: SriSatish Ambati