Divide & Recombine with Tessera: Analyzing Larger and More Complex Data

The D&R Framework

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 are applied by the analyst to each of the subsets

- when a method is applied, there is no communication between the subset computations
- embarrassingly parallel

Statistical recombination for an analytic method

- statistical recombination method applied to subset outputs providing a D&R result for the 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
- recombination is a very general concept

Analytic recombination for an analytic method

- the outputs of the method are written to disk
- they are further analyzed individually in a highly coordinated way
- hierarchical modeling

Computationally, this is a very simple.

Tessera Back End

A distributed parallel computational environment running on a cluster

Most use by Tessera so far is Hadoop, but other back ends are provided for

Creates subsets as specified by the analyst and writes them out across the cluster nodes on the Hadoop Distributed File System (HDFS)

Runs D&R analytic computations as specified by the user using the Hadoop MapReduce distributed parallel compute engine.

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

1st implementation Jan 2013

RHIPE: The R and Hadoop Integrated Programming Environment

When Hadoop is the back end, 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 at Purdue

1st implementation Jan 2009

What the Analyst Specifies With datadr

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

D[dr]

- division method to divide the data into subsets
- data structure of the subset R objects

A[dr]

- analytic methods applied to each subset
- structure of the R objects that hold the outputs

R[dr]

- for an analytic method, a statistical 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 Hadoop Distributed File System (HDFS)

Typically uses both the Map and Reduce computational procedures (see below)

A[dr]
Applies an analytic method to subsets in parallel on the cores of the cluster with no communication among subset computations

Uses the MAP procedure which carries out parallel computation without communication among the different processes

For an analytic recombination, writes outputs to HDFS

R[dr]
Takes outputs of the A[dr] computations, carries out a statistical recombination method, and writes the results to the HDFS

Uses the Reduce procedure which does allows communication between the difference output computations

The R Session Server

Analyst logs into it

Gets an R session going

Programs R/datadr the R global environment

Hadoop jobs are submitted from there

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

Break up the 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

There can be multiple conditioning variables that form the division

There can be a number of conditioning-variable divisions in the analysis

The heavy hitter in practice for D&R

This applies to all data in practice, from the smallest to the largest

Widely done in the past

Replicate Division: The Concept

Observations are seen as exchangeable, with no conditioning variables considered

Subsets are replicates

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

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

Now the statistical theory and methods kicks in

While a distant second in practice, still a critical division method

Statistical Accuracy for Replicate 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 accuracy of D&R results

The accuracy of the D&R result depends on the division method and the recombination

A community of researchers in this area is developing

Another Approach to Replicate Division

Distributed parallel algorithms 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)

Access data at each iteration

Critical notion here for computational performance is that the data are addressable in memory

Spark has this capability

Conditioning-Variable Division

Suppose we have 10 TB of data with a large number of variable and need to explain a binary variable

Do we want to do a logistic regression using all of the data?

We should not just drop a logistic regression on the data and hope for the best

Suppose there is a categorical explanatory variable

- its different values have different values of the regression parameters
- needs to be a conditioning variable

It is likely that there is much to be learned by conditioning

The premise of the trellis display framework for visualization: backed up by a large number of smaller datasets

Let's not use the poor excuse: "This is just predictive analytics. All I need to do is get a good prediction."

Conditioning-Variable Division: Recombination

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

If outputs are collectively large and complex and challenge serial computation

- a further D&R analysis

If outputs are collectively smaller and less complex

- written from HDFS to the analyst's R global environment on the R session server
- further analysis carried out there
- 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

Cyber Security: Spamhaus Blacklist Data

Collecting data from the Stanford mirror of the Spamhaus blacklist site at a rate of 100 GB per week.

13,178,080,366 queries over 8+ months

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

13 variables: 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 and Subset Data Structures

17TB of R objects in the Hadoop HDFS (Hadoop replicates data 3 times)

Cluster for analysis: 320 GB

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

This is conditioning-variable division

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

Bundle 6,000 queries on an R dataframe and make it Hadoop key-value pair

A[dr] code given to Tessera for these dataframes executes analytic method row-by-row

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

This is conditioning variable division

Now the number of variables for each subset goes from 13 to 11

Dataframe object for a blacklisted host 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 on 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

How Fast?

Logistic regression

Number of observations $N = 2^{30} \approx$ 1 billion

1 response and p = 127 explanatory variables, all numeric

Number of variables $V = p + 1 = 2^7$ = 128

8 bytes per observation

$2^{30}2^72^3$ bytes = 1 TB of data

Number of observations per subset, M, from $2^{18} \approx$ 128 thousand to $2^{22} \approx$ 4 million

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

The Rossman Cluster

11 nodes, each a Hewlett Packard ProLiant DL165 G7

Each node

- Dual 2.1 GHz 12-core AMD Opteron 6172 processors (24 cores)
- 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

1 TB data size about 2 times total cluster memory size

By today's standards, modest hardware power

The servers are getting old, so Purdue research computing insisting we and others replace

We get a $110 rebate for each node

How Fast is 1 TB Logistic Regression on Rossman

Minimum total elapsed time of the computation 17.6 min

Occurred at $M = 2^{20} \approx$ 1 million

70% of this time was simply reading the subsets into memory with R and forming the subset objects

So `glm.fit` computations and writing to the HDFS are 5.3 min

1 TB was the largest dataset size that “worked”

The Deneb/Mimosa Cluster

There are two nodes, each a Dell 1950. Each has

- Dual 2.33GHz 4-core Intel(R) Xeon(R) E5410 processors (8 cores)
- 32 GB memory
- 2TB disk in SAS-RAID
- 1 Gbps Ethernet interconnect

Collectively, the cluster has

- $8 \times 2 = 16$ cores
- $32 \times 2 = 64$ GB total memory
- $2 \times 2 = 4$ TB disk

Very small hardware power compared with Rossman

Logistic Regression with `glm.fit` on Deneb/Mimosa

Number of variables: $V = 2^6 = 64$

How big a logistic regression can be done serially in R?

The largest value of N, the number of observations, that "worked" was 2^{23}

The memory size of the data is $2^6 2^{23} 2^3 = 2^3 2$ = 4 GB

The run time was 16.52 min

How big a logistic regression can be done using D&R/Tessera

In doing this we varied, M, the number observations/subset from $2^8 \approx$ 256 to $2^{18} \approx$ 256,000

The largest value of N that worked was 2^{27}

The memory size of the data is $2^6 2^{27} 2^3 = 2^{36}$ = 64 GB

Minimum total elapsed time of the computation 16.71 min

Occurred at $M = 2^{11} \approx$ 2 thousand

”Big Data”?

A very poor term

A concept associated with the term: new computational technology is needed to make computational performance with them acceptable

(1) computation must be feasible; (2) if feasible, performance needs to be practical

For data analysis the term “big data” misses half the problem with computational performance

Size is a factor in performance

Complexity of the data is critical too

- really it is the computational complexity of analytic methods used in the analysis
- but complexity of the patterns on the data are positively correlated with the computational complexity of the analytic methods

The term “large and complex data” conveys this, “big and complex data” if you must

”Large and Complex Data”

Still a problem

When is a dataset large and complex

The term cannot be clearly defined without something else

That something else is your cluster

The data are too large and complex for your cluster when

- applying analytic methods to them is not feasible
- if feasible, execution time is so long, it is impractical

Larger and More Complex is the Key Concept

Let’s suppose the Deneb/Mimosa results are actually real data being analyzed in a technical discipline

Today, 4 GB and even 64 GB are not large datasets

The cluster power is low

This is irrelevant

What is relevant is that the D&R with Tessera enables analysis of 16 times more data

An increase by a factor of 16 in the amount of data that can be analyzed can have an immense impact on the discipline being studied

D&R with Tessera for Data of All Sizes and Complexity

D&R with Tessera can significantly increase the size and complexity of the data that can be analyzed, without increasing the power of the cluster hardware

For those who are currently analyzing 100’s of gigabytes of complex data

For those who are currently analyzing a few terabytes of complex data

For those who have even more data

B&R with Tessera has a very wide scope; it’s not just for “big”

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

A powerful way to carry out

- model diagnostics to make sure a model fits the data
- method diagnostics to make sure the method was sensible

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

Visualization for Large Complex Data

This is as true today, as in the past

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

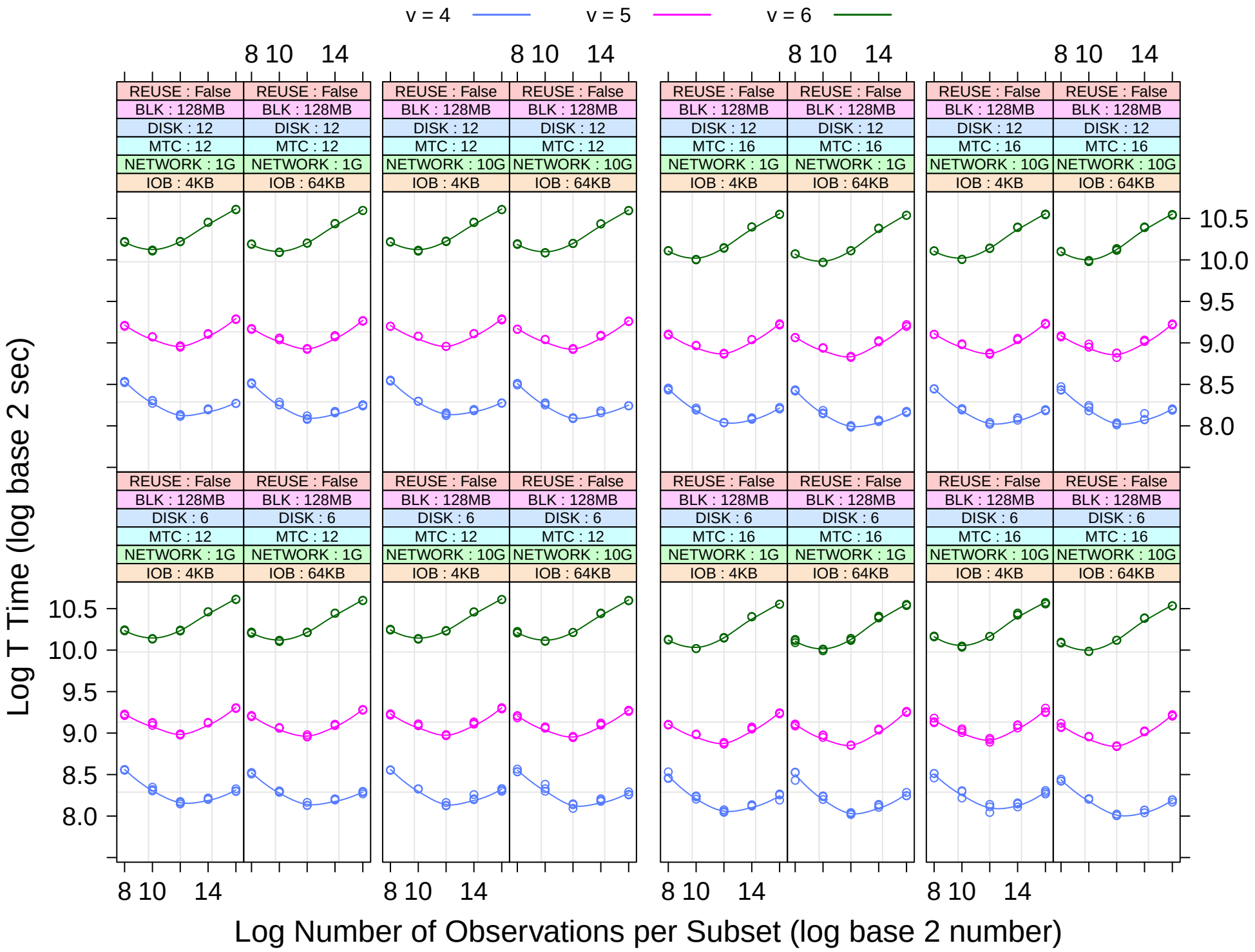
So the visualization method is applied to a sample of subsets

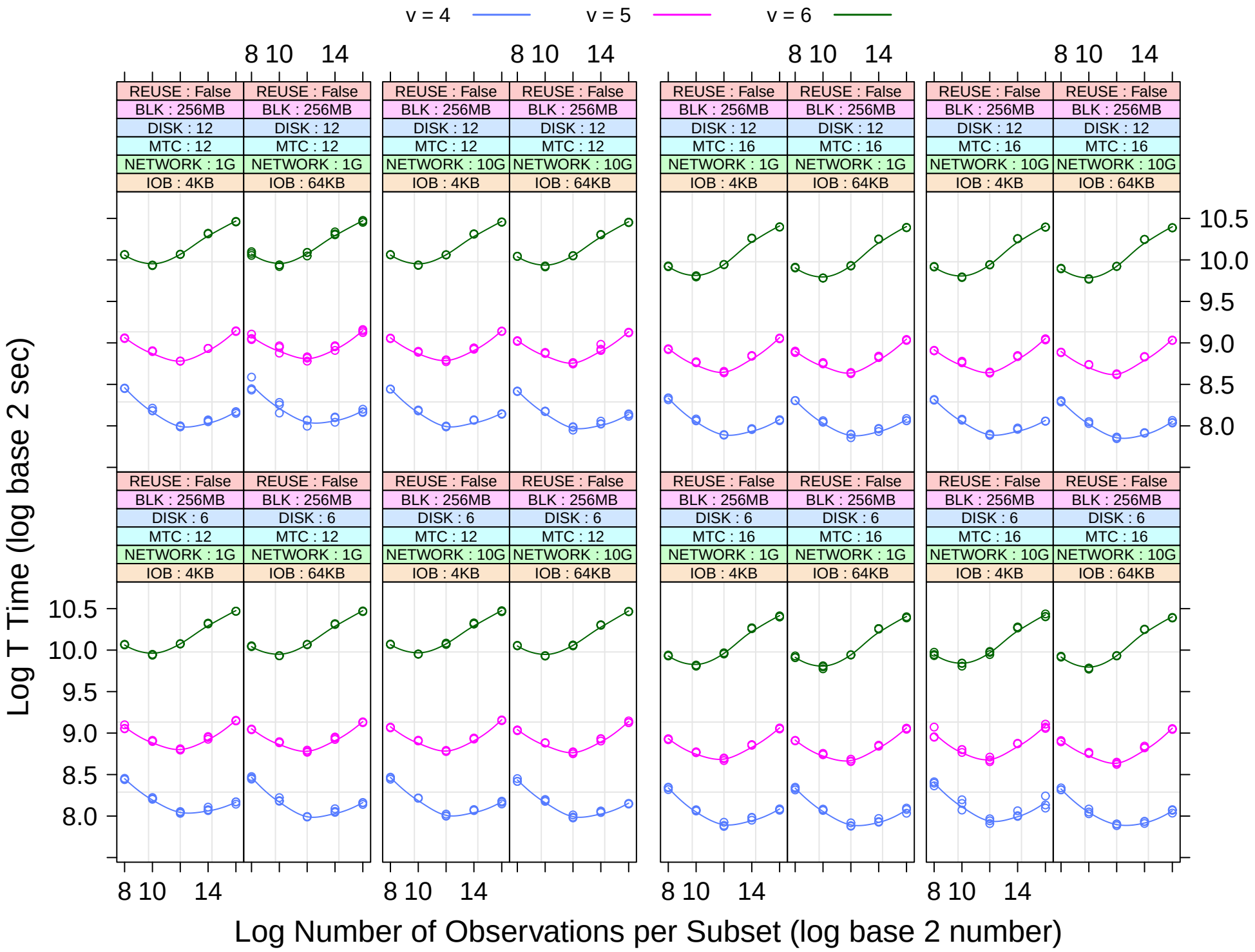
Such sampling can be very powerful and rigorous

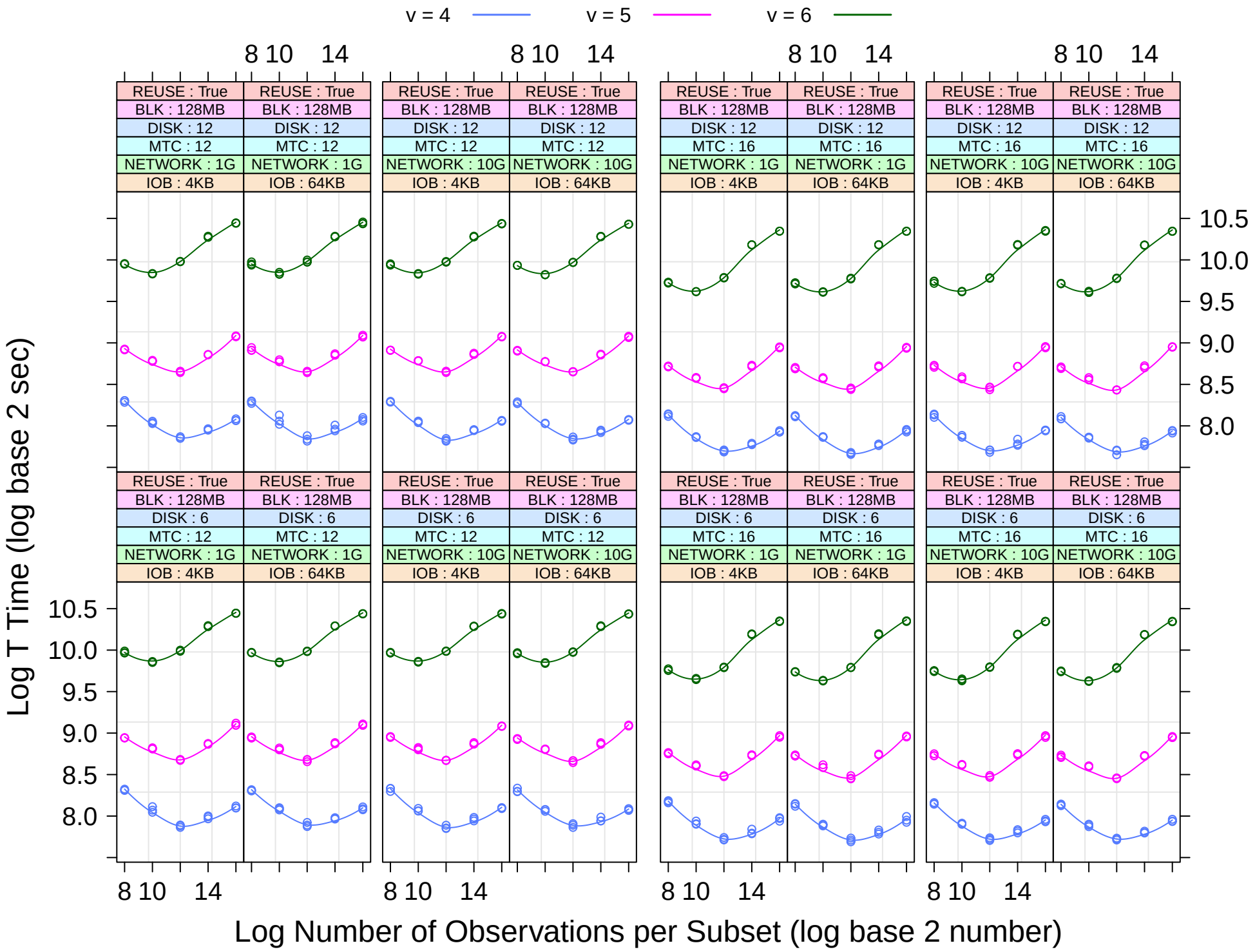
We can readily compute variables, each with one value per subset to enable rigorous sampling plans

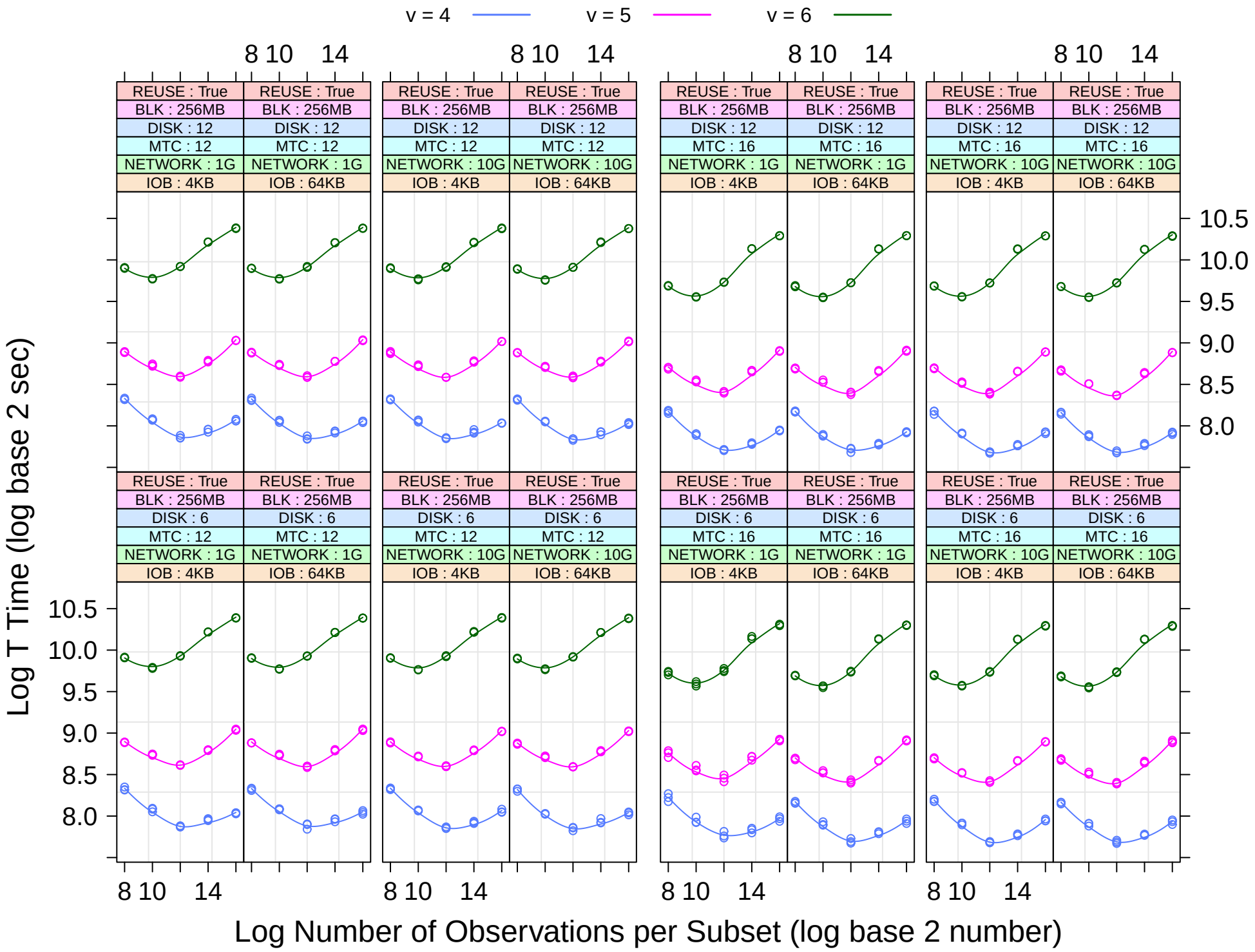
Sampling plans

- stratified sampling
- focus on a region of the sampling variables


v = 4
v = 5
v = 6
REUSE : False
BLK : 128MB
DISK : 12
DISK : 6
MTC : 12
MTC : 16
NETWORK : 1G
NETWORK : 10G
IOB : 4KB
IOB : 64KB
8 10 14
10.5
10.0
9.5
9.0
8.5
8.0
Log T Time (log base 2 sec)
Log Number of Observations per Subset (log base 2 number)


v = 4
v = 5
v = 6
REUSE : False
BLK : 256MB
DISK : 12
DISK : 6
MTC : 12
MTC : 16
NETWORK : 1G
NETWORK : 10G
IOB : 4KB
IOB : 64KB
8 10 14
10.5
10.0
9.5
9.0
8.5
8.0
Log T Time (log base 2 sec)
Log Number of Observations per Subset (log base 2 number)


v = 4
v = 5
v = 6
REUSE : True
BLK : 128MB
DISK : 12
DISK : 6
MTC : 12
MTC : 16
NETWORK : 1G
NETWORK : 10G
IOB : 4KB
IOB : 64KB
8 10 14
10.5
10.0
9.5
9.0
8.5
8.0
Log T Time (log base 2 sec)
Log Number of Observations per Subset (log base 2 number)


v = 4
v = 5
v = 6
REUSE : True
BLK : 256MB
DISK : 12
DISK : 6
MTC : 12
MTC : 16
NETWORK : 1G
NETWORK : 10G
IOB : 4KB
IOB : 64KB
8 10 14
10.5
10.0
9.5
9.0
8.5
8.0
Log T Time (log base 2 sec)
Log Number of Observations per Subset (log base 2 number)

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

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

It’s the key element in the chain

Requires deep knowledge of computational performance

Requires deep intuition and knowledge about what the data analyst needs

Question: How do you develop knowledge about what the data analyst needs?

datadr

Answer: You analyze a lot of data

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

The Tessera team is now about 25: PNNL, Purdue, Hafen Consulting LLC, and Mozilla

There many data analysis projects

The data lead the way

This is why S/R became so widely used

Summary: Important Points Covered

Division & recombination are very broad concepts

- combined with Tessera, cover what is needed for data analysis

Conditioning-variable division

- the division heavy hitter in practice
- a statistical best practice going on for decades

You can increase size and complexity of the data you analyze on your cluster

- for those currently analyzing 100's of gigabytes of complex data
- for those currently analyzing a few terabytes of complex data or even more

You can analyze a dataset whose memory size is larger than total cluster memory

- bigger than memory often happens
- not being able to do this is quite limiting

Visualization of the detailed data at their finest granularity

- critical, whatever the size of the data
- a statistical best practice

datadr at the front end meets the immense challenge

- based on immense experience with data analyses
- requires deep understanding of computational systems behind it