Interactive Data Science

Tim Kraska
I want to understand how our customers from CA differ from the ones from MA.

OK, give me a week.

I found a significant difference in the age between CA and MA.

Did you also look at the salary difference? And what about MA vs WA?

???????????
Interactivity allows for more Teamwork
Landscape

Interactive/Visual

- Tableau, imMens, ...
- GestureDB, IFocus, dbTouch, MapD, ...
- Database systems

Complex workflows

- BlinkDB
- MADlib, ...
- RapidMiner
- Analytical Frameworks (Hadoop, Spark, ...)

Not scalable
Scalable
Why is nobody in that space

We need new ways to do data science and no python is not the answer

Cutting latency is hard (especially if you do not know, what you are looking for)

A lot of interesting data is in the tail. Cutting latency for the tail is even harder
Limitations

Not for Big Big Data, but...
• 90% of jobs on a Facebook cluster have input sizes under 100 GB\(^2\)
• Cloudera Customers: typical job <10GB\(^1\)
• Microsoft/Yahoo!: median size <14GB\(^2\)

Not for All Domains, but...
• There is a lot of potential for building interactive infrastructure for other domains
• For example, we started to work with the Rhode Island Hospital to support medical image data

Why Does Latency Matter
Why Does Latency Matter

“Delays of 500ms incurred significant costs, decreasing user activity and data set coverage while reducing rates of observation, generalization and hypothesis.”

In Class Discussion

Where would you start cutting down latency?
Techniques

• Adaptive Indexing (Database Cracking, VizTrees,...)

• Pre-fetching/pre-computation (Dice, ForeCache,...)

• Sampling (M4, IDEA,...)

• Online Aggregation (RippleJoin,...)

• Recommender Systems (SeeDB, Data Polygamy,..)

• Auto-Tuning (Mlbase,...)
Brown’s Interactive Data Exploration Stack (BIDES)
Brown’s Interactive Data Exploration Stack (BIDES)

vizdom

[VLDB15]
Brown’s Interactive Data Exploration Stack (BIDES)
Problem with Existing Systems

Hadoop

Flink

Asterix DB

Spark

Myria

and many others....
Unique Opportunity: Enterprise Clusters

Most important differences:
• Cluster Size
  • Median Hadoop cluster: <10 nodes
  • 65% of Hadoop clusters: <50 nodes
• Memory Size 1-8 TB
• More reliable hardware
• More advanced CPU features
• Fast interconnects (e.g., Infiniband FDR 4x)

- Built from scratch for enterprise clusters
- No runtime, no fault-tolerance, no complex resource sharing,…
- Co-design of language, framework and compiler for a specific type of hardware
- Key ingredient: Tupleware compiles workflows into distributed programs
What makes Tupleware different from other Code Compiling Systems

We were the first to look at compiling complex workflows into distributed systems, but more importantly:

**DBMS**
- High-level rewrites
- Language semantics
- Data Statistics

**Compiler/HPC**
- Low-level information
- Minimize indirection
- Hardware-dependent
Architecture

1. Programming model with explicit shared state.

2. Code generation

3. Distributed program deployment instead of run-time
Tupleware Optimizations

New Hybrid Optimizations
1) Program Structure (H1)
2) Context Variables (H2)
3) Selection Strategies (H3)
....

DBMS
High-level rewrites
Language semantics
Data Statistics

Compiler/HPC
Low-level information
Minimize indirection
Hardware-dependent

```c
data[N];
while (converge()) {
    for (i = 0; i < N; i++) {
        dist = distance(data[i]);
        min = minimum(dist);
        reassign(min);
    }
    recompute();
}
```
data[N]; dist[N]; min[N];
while (converge()) {
    for (i = 0; i < N; i++)
        dist[i] = distance(data[i]);
    for (i = 0; i < N; i++)
        min[i] = minimum(dist[i]);
    for (i = 0; i < N; i++)
        reassign(min[i]);
    recompute();
}

Tiling helps with cache-locality but is not as good as pipelining for register locality

Hybrid Heuristic

```
data[N]; min[N];
while (converge()) {
    for (i = 0; i < N; i++) {
        dist = distance(data[i]);
        min[i] = minimum(dist);
    }
    for (i = 0; i < N; i++)
        reassign(min[i]);
    recompute();
}
```

**Algorithm:** For each UDF, partition if CPU cycles > load cycles.

*Why not a compiler?*
Tupleware Optimizations

New Hybrid Optimizations
1) Program Structure (H1)
2) Context Variables (H2)
3) Selection Strategies (H3)

DBMS:
- High-level rewrites
- Language semantics
- Data Statistics

Compiler/HPC:
- Low-level information
- Minimize indirection
- Hardware-dependent

1. Programming model with explicit shared state.

2. Code generation

3. Distributed program deployment instead of run-time
Tupleware Achieves Orders-Of-Magnitude Performance Improvements

- Amazon EC2 (10 x c3.8xlarge, 600GB memory)
  - Does not include RDMA benefits to be more fair
- Common machine learning workflows
- Important: (1) same algorithm in all systems; (2) log scale

[VLDB15, CIDR15]
Brown’s Interactive Data Exploration Stack (BIDES)
Brown’s Interactive Data Exploration Stack (BIDES)

- IDEA
  Interactive Data Exploration Accelerator

- Tupleware

- Legacy Systems

References:
- CIDR15, VLDB15, EngBull
- HILDA@SIGMOD, several under submission
- VLDB15
Brown’s Interactive Data Exploration Stack (BIDES)

IDEA
Interactive Data Exploration Accelerator

[-VLDB15, HILDA@SIGMOD, several under submission]

[CIDR15, VLDB15, EngBull]
IDEA Design Goals

- Just connect (no pre-computed indexes, samples, etc.)
IDEA Design Goals

Just connect (no pre-computed indexes, samples, etc.)

First response <500ms
IDEA Design Goals

Just connect (no pre-computed indexes, samples, etc.)

First response <500ms

See results unfold/
Progressive results
IDEA Design Goals

Just connect (no pre-computed indexes, samples, etc.)

First response <500ms

See results unfold/
Progressive results

Quantify risk
A lot of challenges

1) Fast progressive computation of histograms and other aggregates
2) Analytics in the tail of the distribution
3) Re-use of results / Consistency
4) Auto-tuning of Stats/ML algorithms
5) Adaptive index structures for visualizations/ML-pipelines
6) Biased samples (e.g., a database index can highly bias the data stream you get from it)
7) Extensibility
8) Integration of 3\textsuperscript{rd}-party code
9) Expressivity of the operations
10) Quantifying and visualizing error/risk (more on that later)

…
A lot of challenges

1) Fast progressive computation of histograms and other aggregates
2) Analytics in the tail of the distribution
3) Re-use of results / Consistency
4) Auto-tuning of Stats/ML algorithms
5) Adaptive index structures for visualizations/ML-pipelines
6) Biased samples (e.g., a database index can highly bias the data stream you get from it)
7) Extensibility
8) Integration of 3rd-party code
9) Expressivity of the operations
10) Quantifying and visualizing error/risk (more on that later)

…
Achieving Sub-Second Response-Times

A single histogram or many other more complex operations

➔ Usually not a problem thanks to online aggregation/sampling
Achieving Sub-Second Response-Times

A single histogram or many other more complex operations

→ Usually not a problem thanks to online aggregation/sampling

Having several ops

→ Usually not a problem thanks to shared scans
Achieving Sub-Second Response-Times

A single histogram or many other more complex operations

→ Usually not a problem thanks to online aggregation/sampling

Having several ops

→ Usually not a problem thanks to shared scans

Linking and selecting

→ Not a problem if done in the mass of the distribution for the same reasons
Achieving Sub-Second Response-Times

A single histogram or many other more complex operations → Usually not a problem thanks to online aggregation/sampling

Having several ops → Usually not a problem thanks to shared scans

Linking and selecting → Not a problem if done in the mass of the distribution for the same reasons

Selecting in the tail → A problem
Nominal: Re-organize the data

Salary

Count

Main Memory

Data Warehouse

Price

Count

Main Memory

Data Warehouse

Data Warehouse
Nominal: Re-organize the data

Why not re-organize for continuous attributes

- Bucket-based sorting insufficient for zoom-in/out
- Potentially introduces hard-to-control sample bias for later stages in the pipeline
Continuous Attributes

B+-Tree

Leaves contain the pointers to the data and are connected to facilitate range-scans
Continuous Attributes

Annotated B+-Tree

Leaves contain the pointers to the data and are connected to facilitate range-scans

Designed for range-requests, not histograms.
Other approaches, like imMens cubes, do not help for later stages in the pipeline (e.g., to get a sample for training) and require pre-processing of the data.
VizTrees – Visual Balanced Tree

Visual Balanced Nodes

Normal Balanced Nodes

1 2 3 4 5 6 6 7 8 10 11 12 13 16 18 20
VizTrees – Visual Balanced Tree

Visual Balanced Nodes

Optimization I: Remove common branches and replace them with scans
VizTrees – Visual Balanced Tree

Optimization I: Remove common branches and replace them with scans
Optimization II: Relax the visual-balance property for the top nodes
Multi-Dimensional VizTree

Instead of being based on B+ Trees, they are based on annotated R Trees

1st vis-level (l=0)

2nd vis-level (l=1)

Other levels (l>1)

Use think-time between interactions to dynamically build the tree
Incremental Loading

![Diagram showing age, salary, and taxes distributions]

- Age: Count
- Salary: Count (incremental loading)
- Taxes: Count (incremental loading)

Data visualization with age on the x-axis, salary on the y-axis, and taxes on the z-axis.
## Multi-Dimensional VizTree

**Compression of Visual Balanced nodes**

Below is a table and a diagram illustrating the compression of visual balanced nodes in a multi-dimensional context. The table and graph represent the distribution and visualization of data in a two-dimensional space, with axes for Salary and Age, showcasing different salary levels and age ranges.

### Table

<table>
<thead>
<tr>
<th>Salary</th>
<th>Age</th>
</tr>
</thead>
<tbody>
<tr>
<td>1k</td>
<td>300</td>
</tr>
<tr>
<td>5k</td>
<td>400</td>
</tr>
<tr>
<td>10k</td>
<td>500</td>
</tr>
<tr>
<td>100k</td>
<td>1k</td>
</tr>
<tr>
<td>1M</td>
<td>10k</td>
</tr>
</tbody>
</table>

### Diagram

The diagram visualizes the data distribution on a Salary-Age grid, with different color intensities indicating varying counts of data points within each cell.
Multi-Dimensional VizTree

Compression of Visual Balanced nodes

Merge regions along the hilbert-curve as long as the fill-grade is below a threshold (e.g., the leaf-fill grade)
Scanning a VizTree

1st vis-level (l=0)

2nd vis-level (l=1)

Other levels (l>1)

Buckets (Vis.-aligned)

Buckets (Vis.-aligned)

Visually-balanced

Normally-balanced
Scanning a VizTree

Count: 1M+100k+10k
Max-Error: 0
Min-Error: 0
Scanning a VizTree

Count: 1M+100k+10k
Max-Error: +\(\alpha\)
Min-Error: -\(\alpha\)
Hoeffding's inequality
Scanning a VizTree

Count: 10k + 1k + 1/3 * 1200
Max-Error: 10k + 1k + 500
Min-Error: 10k + 1k + 300
Scanning a VizTree

Count: 10k + 1k + 1/3 * 1200
Max-Error: 10k + 1k + 500 + α
Min-Error: 10k + 1k + 300 - α
Scanning a VizTree

Stop Traversing and/or Remove Error Bars if

\[ JND = \frac{\text{Stimuli Change } \Delta I}{\text{Stimuli}} < 1.6\% \]

Just-Noticeable-Difference (JND)\(^1\)

---

VisTrees Achieve Interactive Performance

**Uniform**

- Ann.Tree/Trad.Lookup
- Ann.Tree/Hist.Lookup (10%)
- Vis.Tree/Hist.Lookup (10%)

**Skewed (Zipf, s=2)**

- Ann.Tree/Trad.Lookup
- Ann.Tree/Hist.Lookup (10%)
- Vis.Tree/Hist.Lookup (10%)

4 Dimensions
10% Selectivity
Intel Xeon E5-2660 v2, 256GB RAM
VisTrees Can Leverage the User’s Think-Time
(Just one example)

1% Selectivity
600 Million data points
2 Dimensions
Just noticeable difference stopping condition
A lot of challenges

1) Fast progressive computation of histograms and other aggregates (under submission)
2) Analytics in the tail of the distribution
3) Re-use of results / Consistency (under submission)
4) Auto-tuning of Stats/ML algorithms (MLBase/ TUPAQ [CIDR13,SOCC15])
5) Adaptive index structures for visualizations/ML-pipelines (VisTrees under revision for [SIGMOD16])
6) Biased samples
7) Extensibility (MLI [ICDM13])
8) Integration of 3rd-party code
9) Expressivity of the operations
10) Quantifying and visualizing error/risk (more on that later) (Unknown Unknowns [SIGMOD16], Cleaning Error [SIGMOD14])

…
Brown’s Interactive Data Exploration Stack (BIDES)

IDEA
Interactive Data Exploration Accelerator

vizdom

Legacy Systems

[HILDA@SIGMOD, several under submission]

[CIDR15, VLDB15, EngBull]
Brown’s Interactive Data Exploration Stack (BIDES)

IDEA
Interactive Data Exploration Accelerator

Python
Tupleware

vizdom

Legacy Systems

BigDAWG
Brown’s Interactive Data Exploration Stack (BIDES)

IDEA
Interactive Data Exploration Accelerator

Python
Tupleware

vizdom

Mlbase (TuPAQ)
[SOCC16]

BigDAWG

Legacy Systems
Brown’s Interactive Data Exploration Stack (BIDES)

- QUDE: Quantifying the Uncertainty in Data Exploration
- IDEA: Interactive Data Exploration Accelerator
- BigDAWG
- Tupleware
- Legacy Systems
- Mlbase (TuPAQ [SOCC16])

Python

vizdom
Brown’s Interactive Data Exploration Stack (BIDES)

QUDE
Quantifying the Uncertainty in Data Exploration

IDEA
Interactive Data Exploration Accelerator

Python

Tupleware

Mlbase
(TuPAQ)
(SOCC16)

Legacy Systems

vizdom

BigDAWG
SeeDB on Survey Data

Top ranked SeeDB Recommendation
QUDE

Multi-Hypothesis Control

Other common errors (e.g., Simpson Paradox, Missing Data, etc.)
Paper Discussions

Systems for Interactive Data Exploration

Carsten Binnig
Brown University
Updates for Class

• Papers are presented only by one student (not two)

• Projects will be implemented in groups of two

• Presentation Schedule is online on class website

• First presentations: next week!

• Please send us your paper preferences (ignore “Student 2” field in form)
Outline

Review Processes (general)

What is a good paper?

What is a good review?

Review Process (for our class)
Outline

Review Processes (general)

What is a good paper?

What is a good review?

Review Process (for our class)
Review Process

1. Define your **program committee (PC)**
2. Paper **submissions by authors**
3. Paper **assignment to reviewers** in PC
4. Discuss reviews for Papers
5. Make decisions (Accept / Reject)
6. Paper presentation by authors (conference)
Review Roles

Meta reviewer

• **Summarizes** the highlights of the paper and the reviews
• **Leads the discussion** of the paper during the group meeting
• Makes **final decision**

Reviewer

• **Writes a review** and **recommends a decision**
• Needs to **defend his review** in discussion
Outline

Review Processes (general)

What is a good paper?

What is a good review?

Review Process (for our class)
Key Choices and Trade-Offs

- off the beaten path (low difficulty, low popularity)
  - exciting: trendsetting opportunity
- mainstream (high difficulty, high popularity)
  - still interesting: be early & fast to beat the crowds (or be better)
- low difficulty, low popularity
  - underexplored but perhaps boring
- high difficulty, high popularity
  - too easy
Criteria for Good Papers

1. Problem Validity

2. Major Contribution

1. Openness

1. Presentation

2. Evaluation

3. Related Work
(1) Problem Validity

Do people care about solution?  Does it solve a “real” problem?
(2) Major Contribution

The way to go:

• Discovers “new land” or solves a well-known “hard problem”
• Thorough study or evaluation

Things to avoid:

• Avoid minor-incremental work
• Avoid potpourri papers
(3) Openness

Solution is not directly clear:

Variety of approaches conceivable

Need for thorough evaluation
(4) Presentation

Presentation should be adequate:

Do not make papers unnecessarily complicated

- **Formalisms** are nice if they help
- **Intuition and Examples** are equally important

Use a **professional consultant** for spell checking
(5) Evaluation

Should support paper claims

Study a broad range important parameters / scenarios

Explain setup (HW + SW) and workload / data
(6) Related Work

• Clearly state what is different from existing solutions

• ... and why proposed solution is better!
Outline

Review Processes (general)

What is a good paper?

What is a good review?

Review Process (for our class)
At 10.000 feet

At a high level, your review should:

• be at least 4-5 paragraphs in length
• critique the paper constructively
• communicate to the authors that you read their paper with a positive attitude
• convey clearly the rationale for your criticisms
Key Questions to Answer

Q1: Overall rating: Accept or Reject

Q2: Briefly summarize your review, and rationale for the chosen rating

Q3: List at least 3 strong points, numbered S1, S2, S3, ...

Q4: List at least 3 weak points, numbered W1, W2, W3, ...

Q5: Paper review
   • Use this section to provide authors with your detailed feedback, and suggestions on how to improve the paper
   • Comment on novelty, depth, presentation quality, and soundness and thoroughness of experimental evaluation.
More Information

Review guidelines (adopted from ACM SIGMOD 2016):
https://brownbigdata.github.io/db-read/guidelines.html

Example: http://cs.brown.edu/courses/csci2951-v/pdf/Review.PDF

• **Reviewer 1: bad** and superficial review
  – Ignores most contributions of the paper
  – Does not list any detailed comments

• **Reviewer 3: good** and very detailed review
  – Constructive feedback which asks for clarification
  – Makes suggestions to enhance the paper
Outline

Review Processes (general)

What is a good paper?

What is a good review?

Review Process (for our class)
Review Process in Class

Everybody in class reviews every paper that is presented on a day

Submit reviews to: http://cs2951v.cs.brown.edu

Presenters do not need to review papers on that day (see Presentation Schedule)

However, presenters need to moderate discussion (i.e., they are meta-reviewers)
Questions?