LBNL Visualization Research Program
High Performance Analytics
Highlights 2005-2007

E. Wes Bethel

Lawrence Berkeley National Lab

12 Feb 2008

Outline

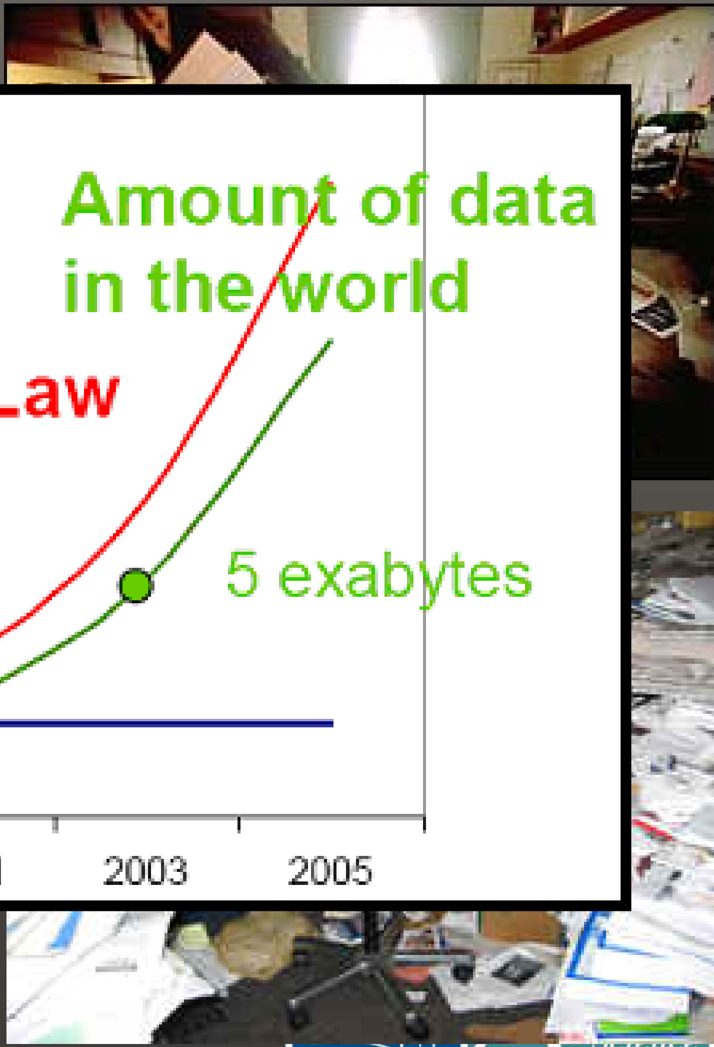
- Introduction
- Baseline performance for visualization apps
- Case Study – Cybersecurity: “Hero-sized” Network Traffic Analysis

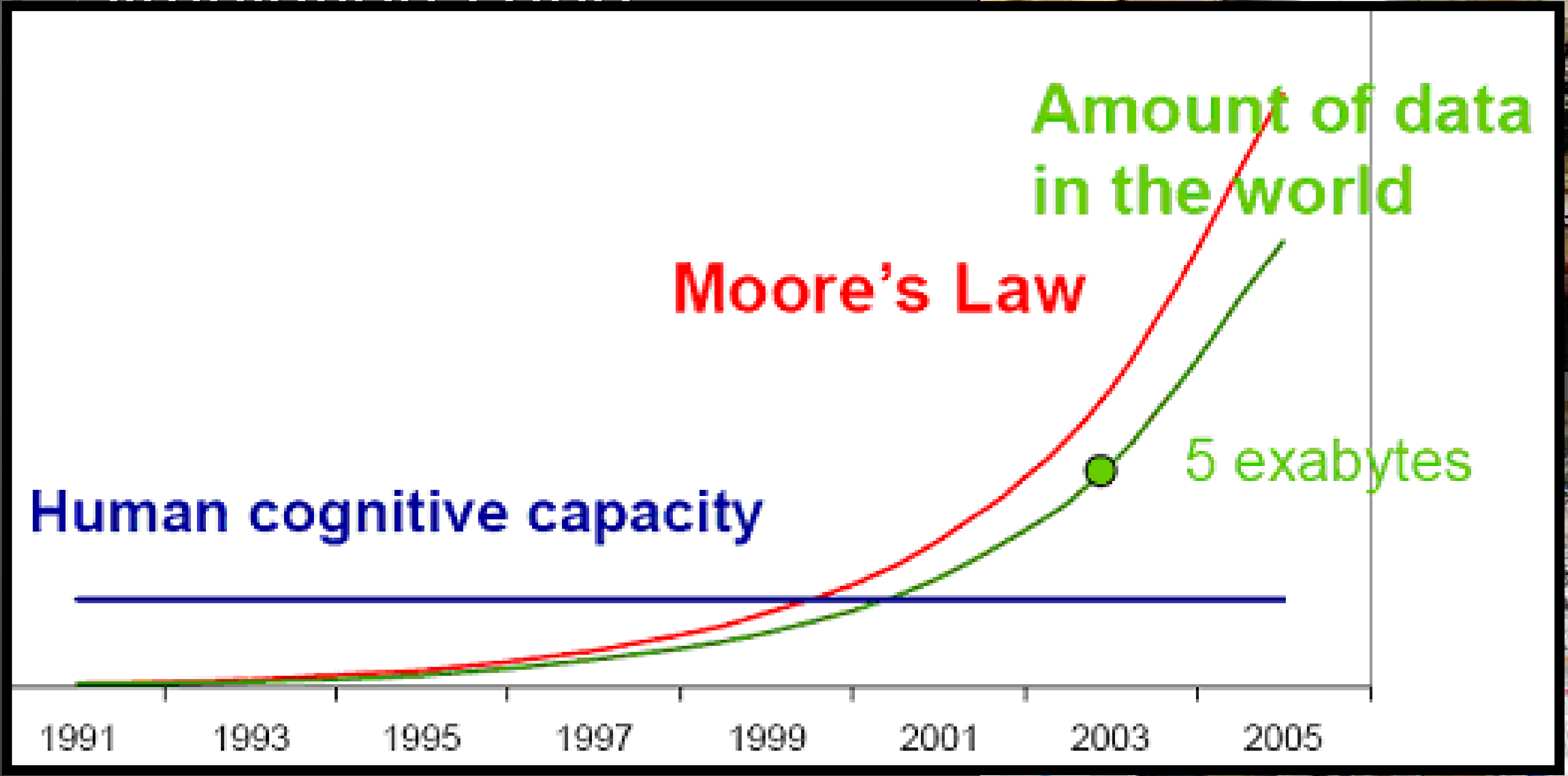
Motivation

- Simulations and experiments are generating data faster than it can be analyzed and understood.
- Science bottleneck: information analysis and understanding.

Motivation

- Simulations and

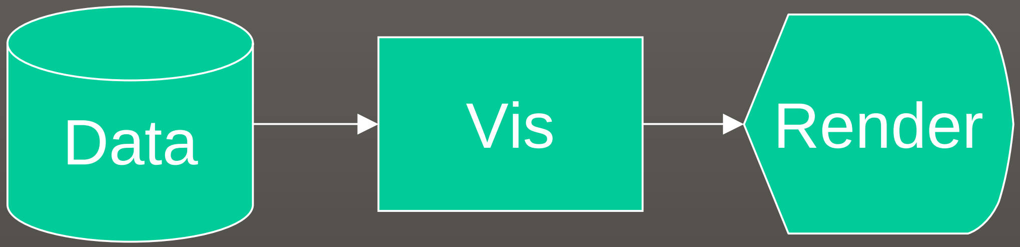




Query-Driven Visualization

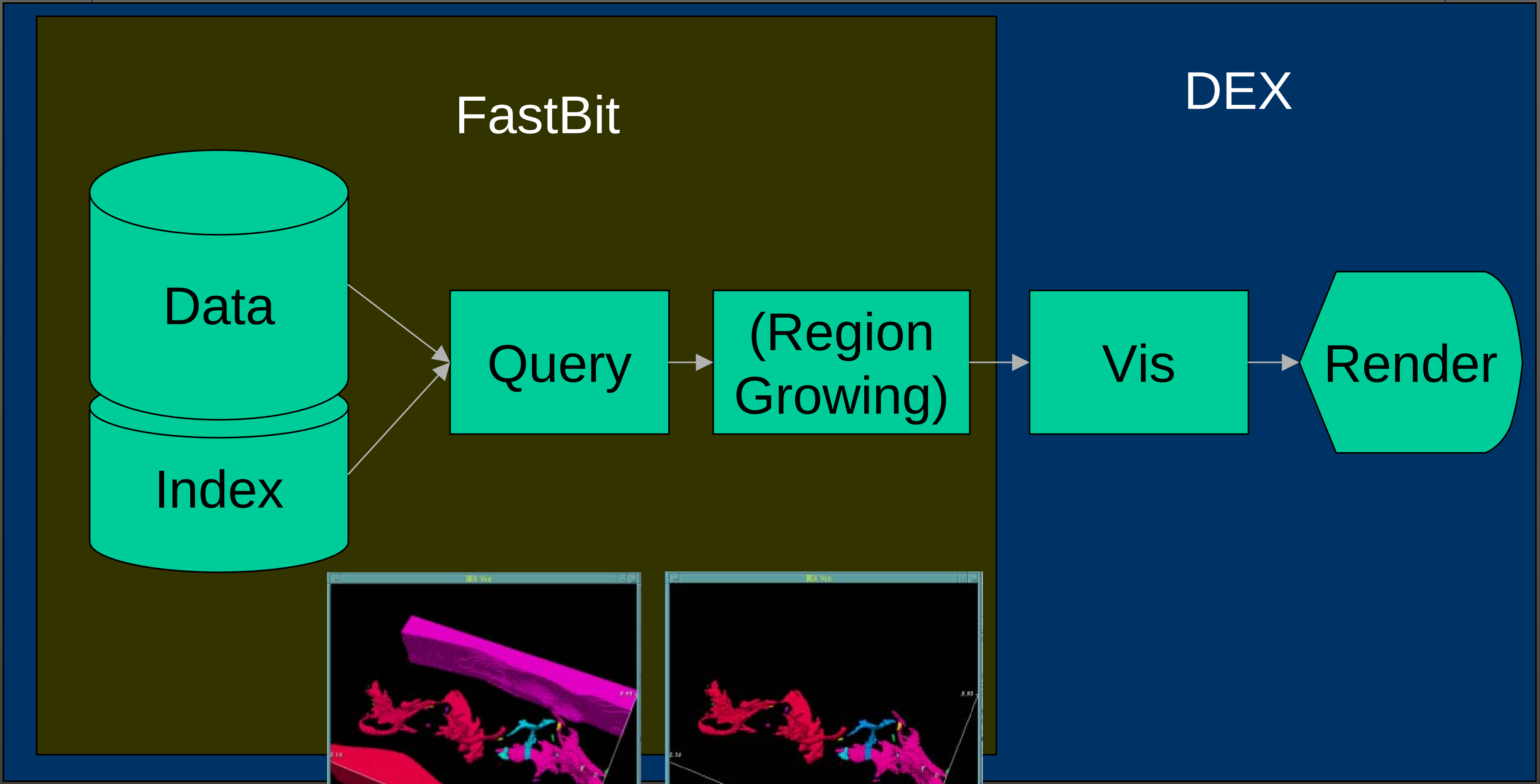
- What is Query-Driven Visualization?
  - Find “interesting data” and limit visualization, analysis, machine and cognitive processing to that subset.
- One way to define “interesting” is with compound boolean range queries.
  - E.g., ($CH_4 > 0.1$) AND ($T_1 < temp < T_2$)
- Quickly locate those data that are “interesting.”
- Pass results along to visualization and analysis pipeline.

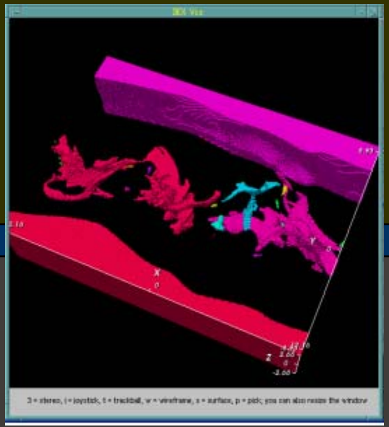
Query-Driven Visualization

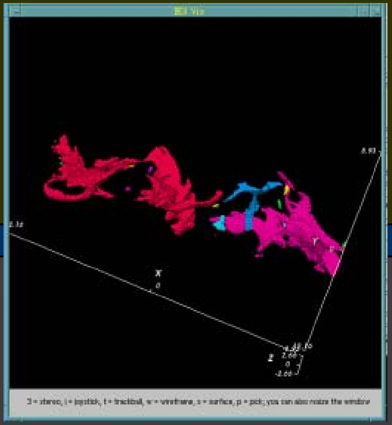


The Canonical Visualization Pipeline

Query-Driven Visualization

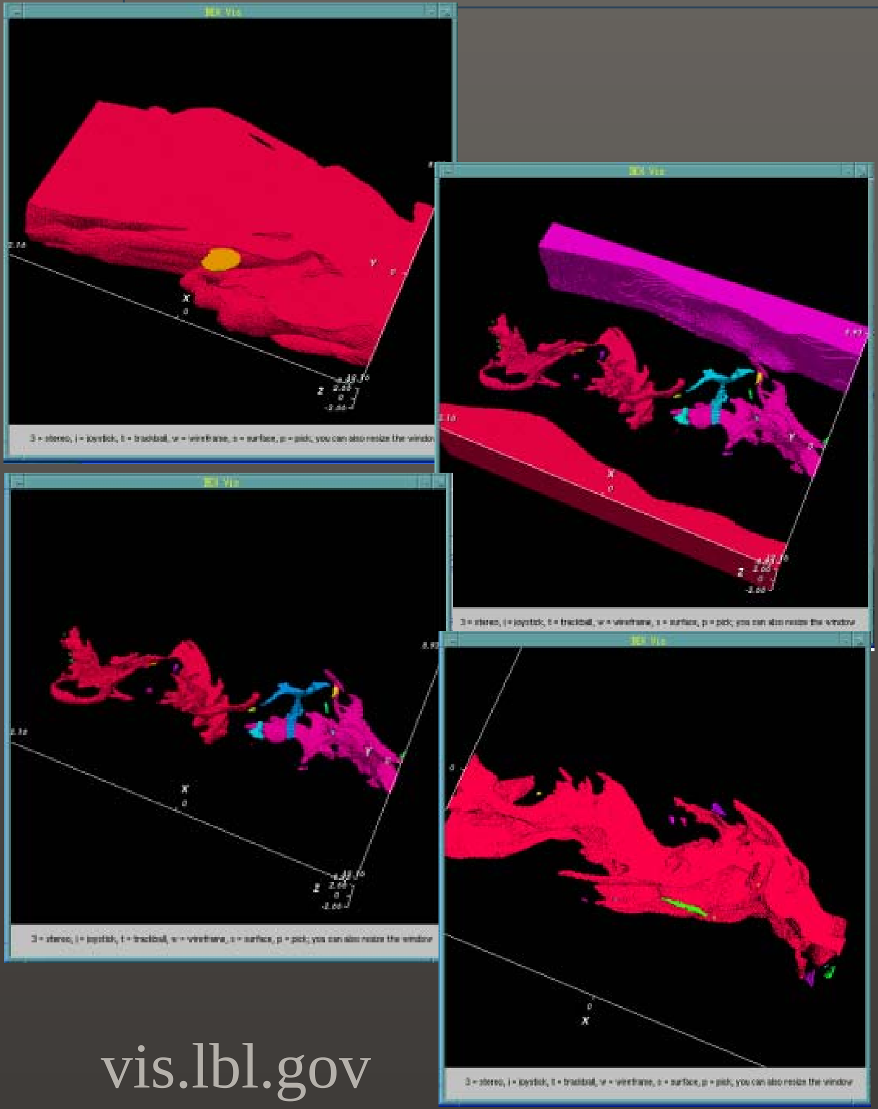






vis.lbl.gov

Query-Driven Visualization


3 = stereo, j = joystick, t = trackball, w = wireframe, s = surface, p = pick; you can also resize the window
x
y
z

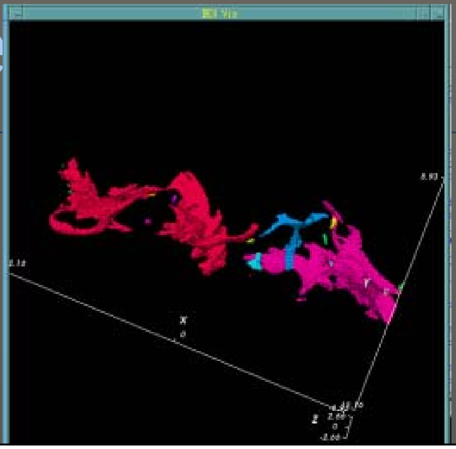
- $CH_4 > 0.3$
- $\text{Temp} < T_1$
- $CH_4 > 0.3$ AND $\text{temp} < T_1$
- $CH_4 > 0.3$ AND $\text{temp} < T_2$
  - $T_1 < T_2$

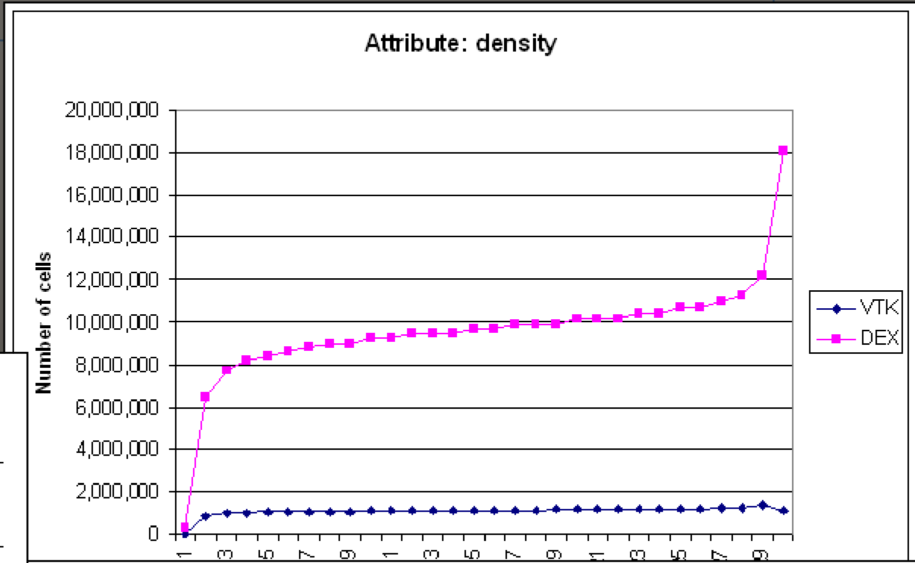
vis.lbl.gov

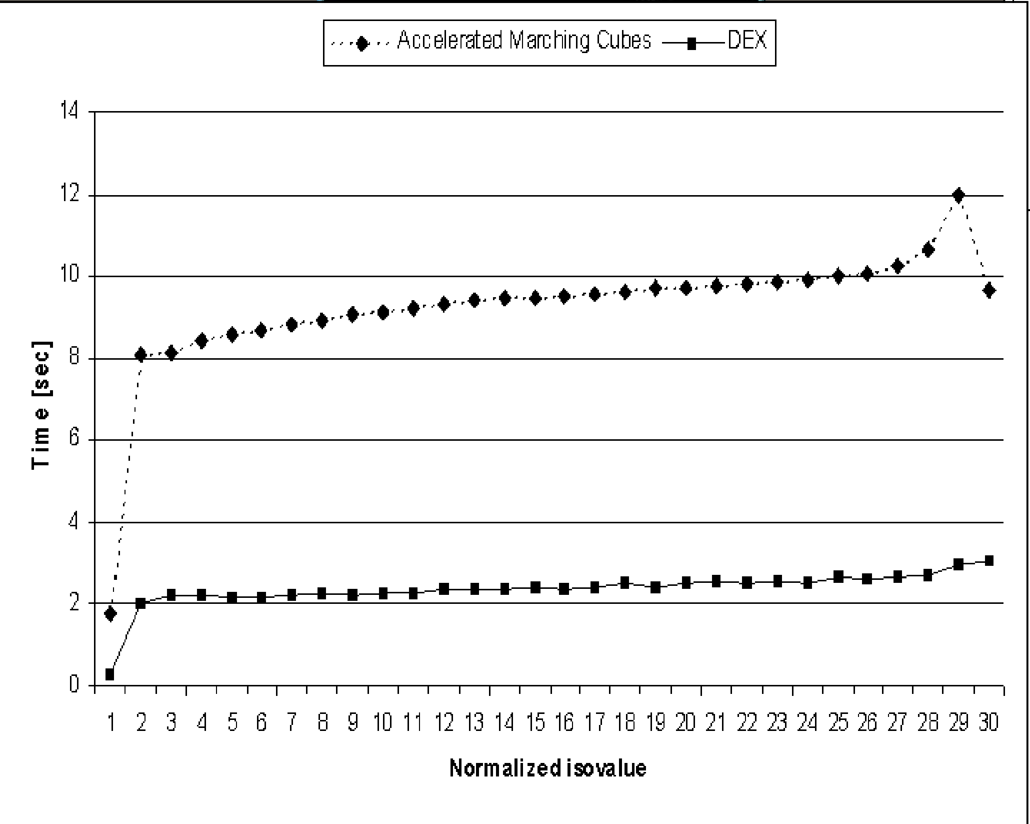
Query-Driven Visualization

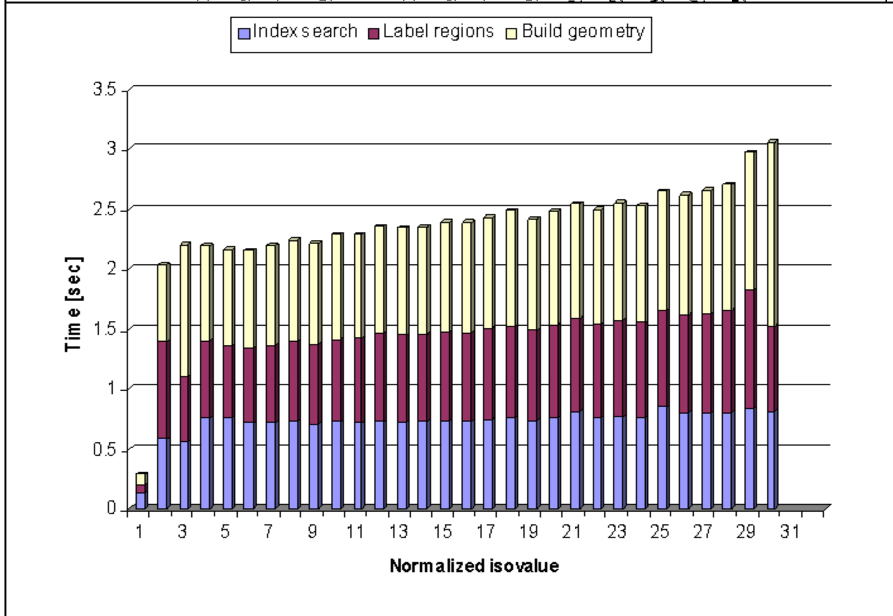
- How fast is it?
  - Comparison: Isosurface algorithms:
    - Nice summary in: Sutton et. al., “A Case Study of Isosurface Extraction Algorithm Performance,” *2nd Joint Eurographics-IEEE TCCG Symposium on Visualization*, May 2000
  - For *n* data values and *k* cells intersecting the surface:
    - Marching Cubes: O(*n*)
    - Octtree methods: O(*k* + *k* log (*n*/*k*))
      - Acceleration: pruning; sensitive to noisy data
    - Span-space methods:
      - NOISE: O(sqrt*(n*) + *k)*
      - ISSUE: O(log *(n/L*) + sqrt*(n)/L + k)*
        - » *L* is a tunable parameter
      - Interval Tree: O(log *n* + *k*)
- FastBit: <u>O(k) – the theoretical optimum.</u>
  - Profound performance gain for Petascale visualization!

Que sualization









Query-Driven Visualization

- What do these timing results mean?
  - In a one-sided matchup (DEX doing a lot more work), our performance results are markedly better for a given task than an industry-standard isocontouring implementation.
- These are single-valued queries.
  - FB capable of *n-dimensional* queries.
  - Tree-based indexing methods not practical for *n-dimensional* queries.
    - Trees: Curse of Dimensionality! (next slides)
    - O(N*D) vs. O(N**D)
- Why compare against isosurfacing?
  - Familiar to the visualization community.

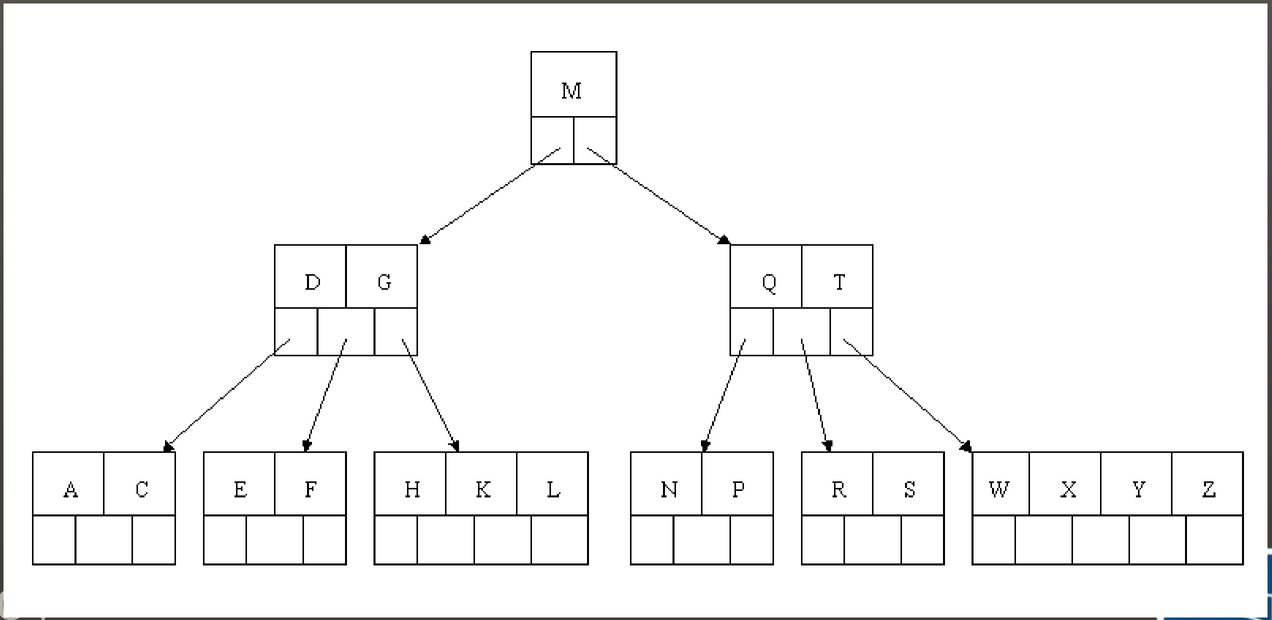
Compressed Bitmap Indices

Data values	b_0	b_1	b_2	b_3	b_4	b_5
0	1	0	0	0	0	0
1	0	1	0	0	0	0
5	0	0	0	0	0	1
3	0	0	0	1	0	0
1	0	1	0	0	0	0
2	0	0	1	0	0	0
0	1	0	0	0	0	0
4	0	0	0	0	1	0
1	0	1	0	0	0	0

- Compact: one bit per distinct value per object.
- Easy and fast to build: O(*n*) vs. O(*n* log *n*) for trees.
- Efficient to query: use bitwise logical operations.
  - $(0.0 < H_2O < 0.1)$ AND $(1000 < \text{temp} < 2000)$
- Efficient for multidimensional queries.
  - No "curse of dimensionality"
- What about floating-point data?
  - Binning strategies.

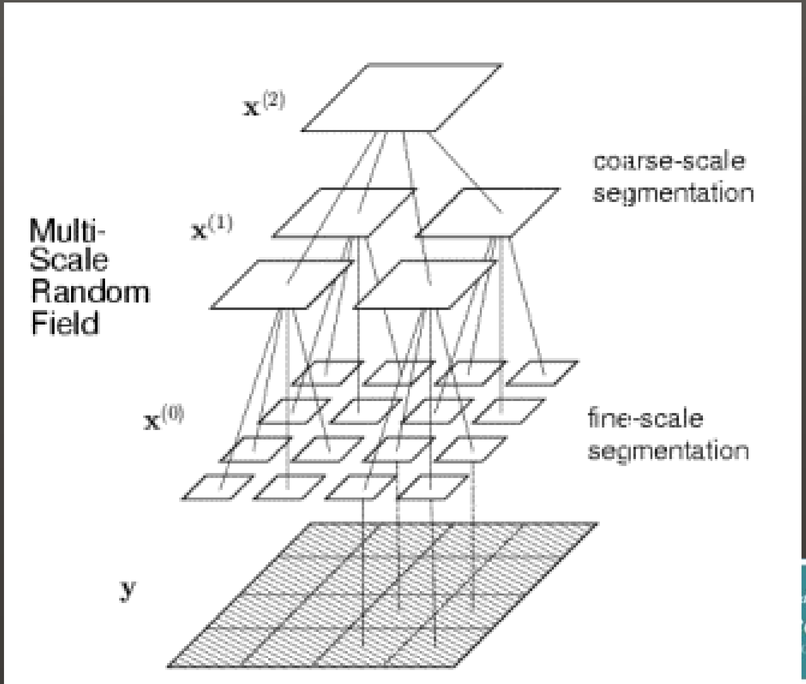
Compressed Bitmap Indices vs. Trees

- One-dimensional B-tree
  - O(*n log n)* for initial sort
  - O(*n log n)* for storage
  - O(*log n)* for search



Compressed Bitmap Indices vs. Trees

- Two-dimensional binary tree
- O(*n log n)* for initial sort in each dimension
- O(*(n log n)^d)* storage (d = dimensions)!
- O(*d log n)* for search



QDV Cybersecurity Case Study

- The next sequence of slides discusses application of the work to a cybersecurity application – proof that the idea is generally applicable to large data visualization.
- The team:
  - NERSC Network Security
  - ESnet Network Engineers
  - Scientific Data Management Research
  - Visualization Research

QDV – Detecting Distributed Scans

- The problem:
  - One day's worth of traffic consists of tens of millions of individual connections.
  - Traffic increasing by an order of magnitude every 48 months.
    - ESnet monthly traffic levels now exceed 1 PB.
  - The Internet is a hostile environment, and it will get worse.
  - Objective: enable rapid forensic data analysis (network flow records).

QDV – Detecting Distributed Scans

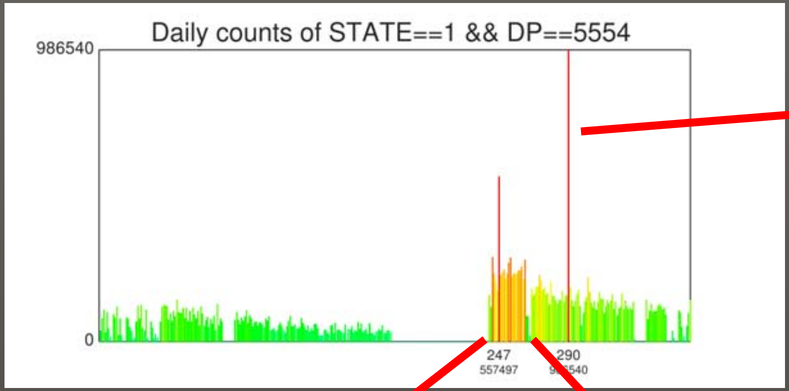
- The data:
  - 42 weeks' of connection records from Bro (NERSC).
  - 281GB for raw data, 78GB for compressed bitmap indices.

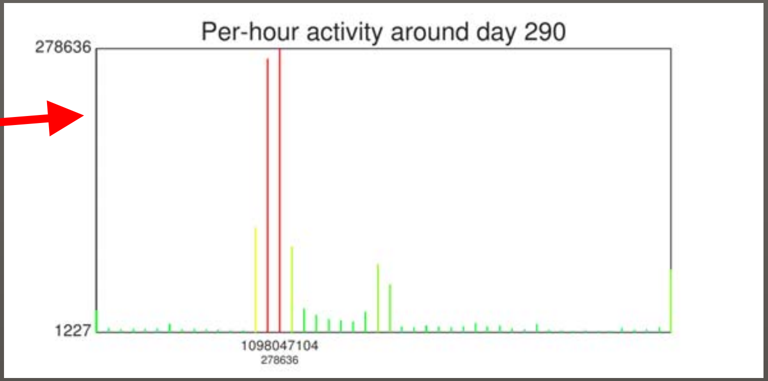
- "Hero-sized problem"
  - No previous network analysis work has ever attempted to perform interactive visual analytics on data of this scale.
  - Result: what once took days (if at all possible) now takes seconds.

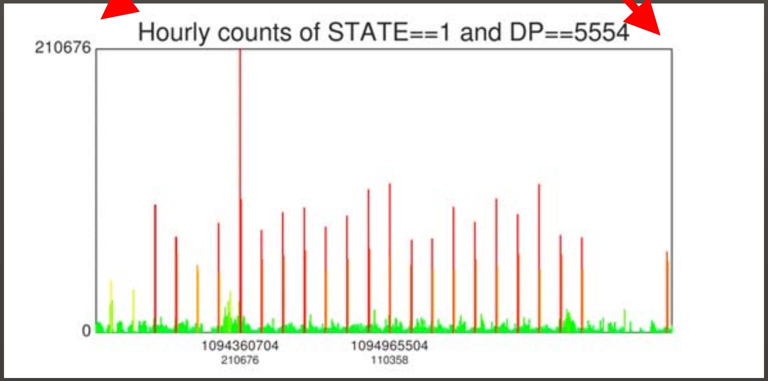
QDV – Detecting Distributed Scans

- The starting point:
  - – You are a network security analyst
  - – Your beeper goes off [at lunch, in the shower…]
  - – You receive an alert that "something odd is happening with the network…IDS showing unusual levels of activity on port 5554"
  - – Your job – answer questions:
    - What's going on now?
    - How long has this been happening?
    - Implications?

QDV – Detecting Distributed Scans

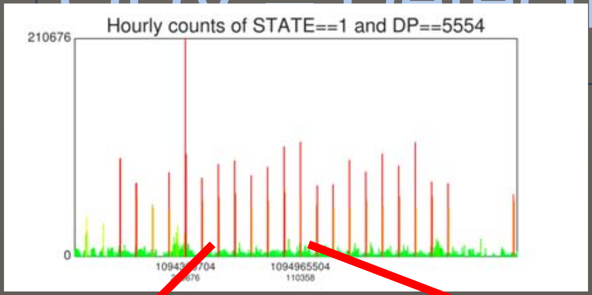


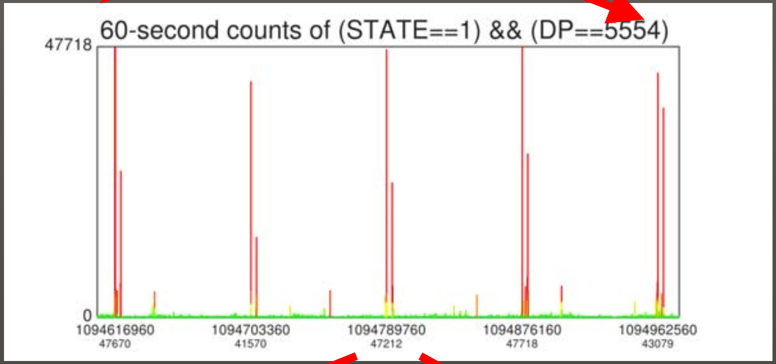


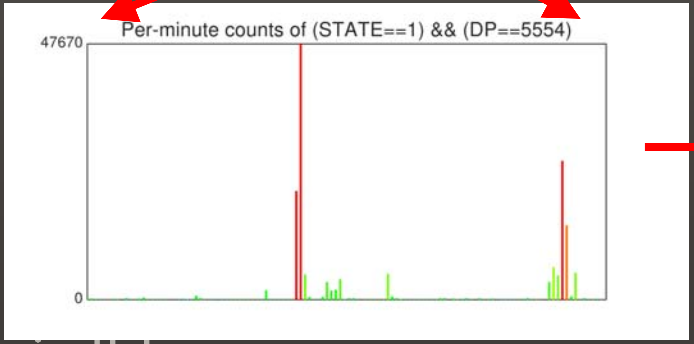


1. Query to produce a histogram of unsuccessful connection attempts over a 42-week period at one-day temporal resolution (upper left).
2. Drill into the data, query to produce a new histogram covering a four-week period at one-hour temporal resolution (lower left).
3. Generate a histogram of one-hour resolution over a two-day period around day 290 (upper right).

QDV – Detecting Dist



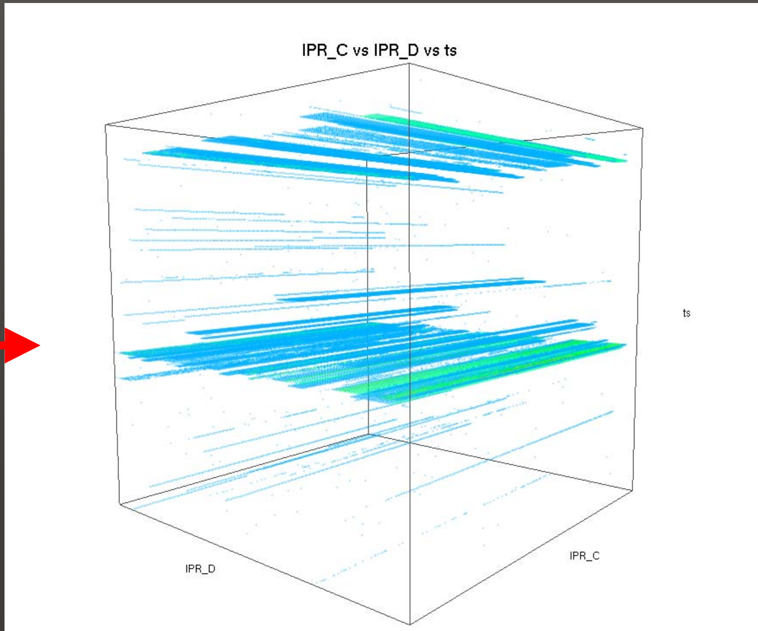




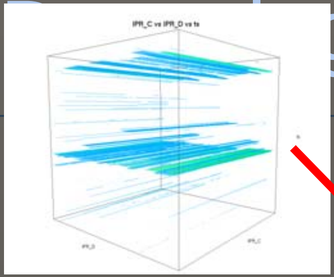
5. Query to generate a histogram of unsuccessful connection attempts over a five-day period sampled at one-minute temporal resolution (middle, left). Regular attacks occur at 21:15L, followed by a second wave 50 minutes later.

6. Query to generate histogram over a two-hour period at one-minute temporal resolution (lower left).

7. Query to generate a 3D histogram showing the coverage of attacks in destination address space (lower right).



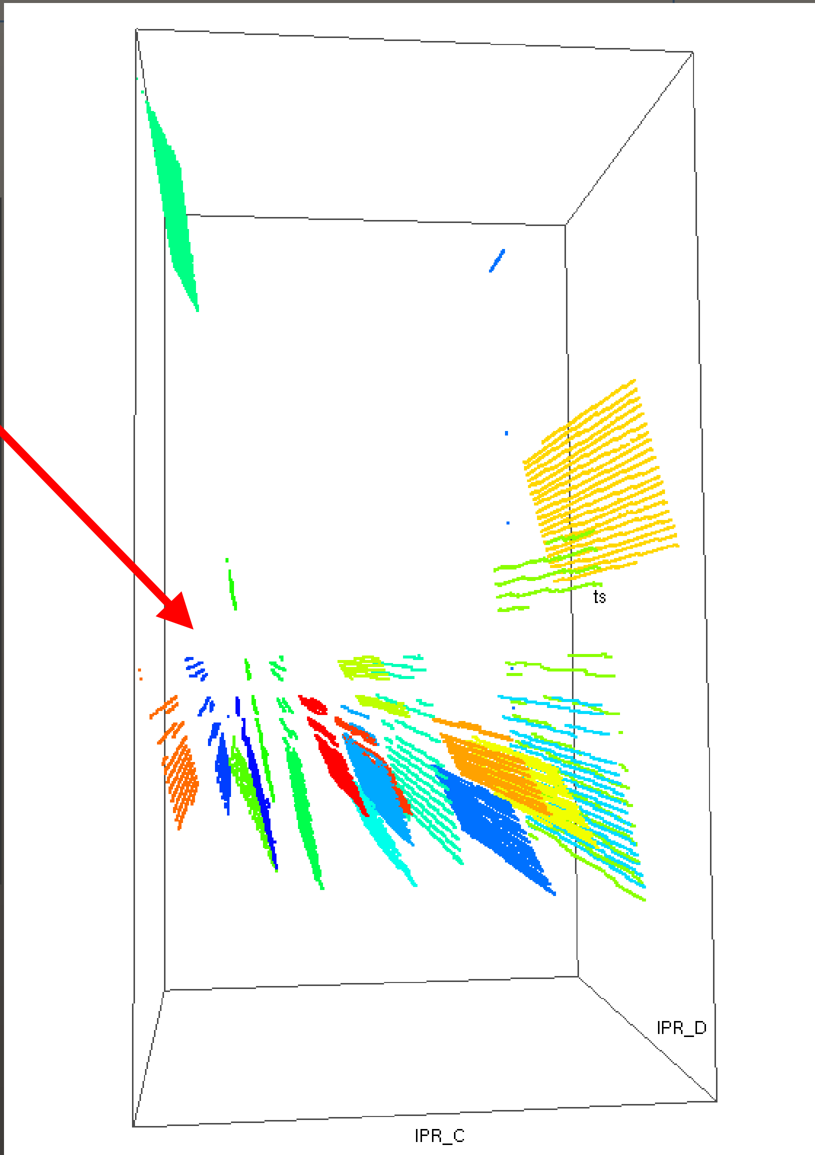
QDV – D[illegible]g Distributed Scans



After establishing that (1) a temporally regular activity is occurring, and (2) that it is in fact a systematic probe (scan) of entire blocks of network addresses, the next task is to determine the set of remote hosts participating in the attack.

Working backwards, we isolate the A, B, C and D address octets of the hosts participating in the attack.

This image shows a 3D histogram of the destination address space being attacked by each of 20 different hosts. The vertical axis is time – a seven-minute window at one-second temporal resolution.



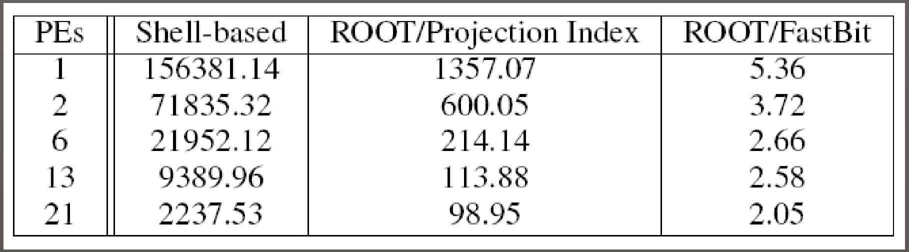
QDV – Detecting Distributed Scans

- Our analysis was performed in statistical space only.
  - We never accessed the raw data.
  - Our processing and visualization used only the index data.
- The same principles can be (and will be) applied to scientific data.
- Challenges:
  - How to define "interesting?"
  - Effective user interfaces for:
    - Support rapid interrogation, propagating query results from step to step in the analysis process.
    - Multivariate visualization
    - Drill-down (mining), linked/correlated views
  - Adapting, applying and deploying these principles to scientific data (e.g., AMR, distributed data/computing resources).

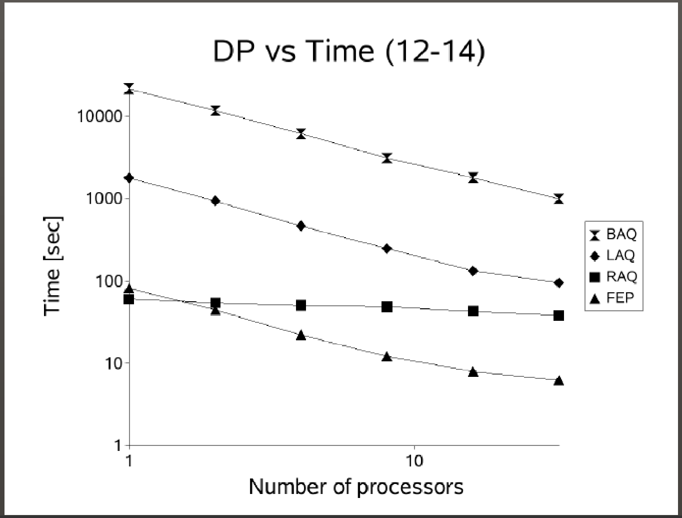
QDV – Detecting Distributed Scans

- How fast is it?
  - 3 to 6 orders of magnitude faster than shell-based tools.
  - 2 to 3 orders of magnitude faster than ROOT, the “gold standard” in the HEP community.
  - Shows favorable scaling characteristics up to 32P.

1D Histogram, per-bin queries



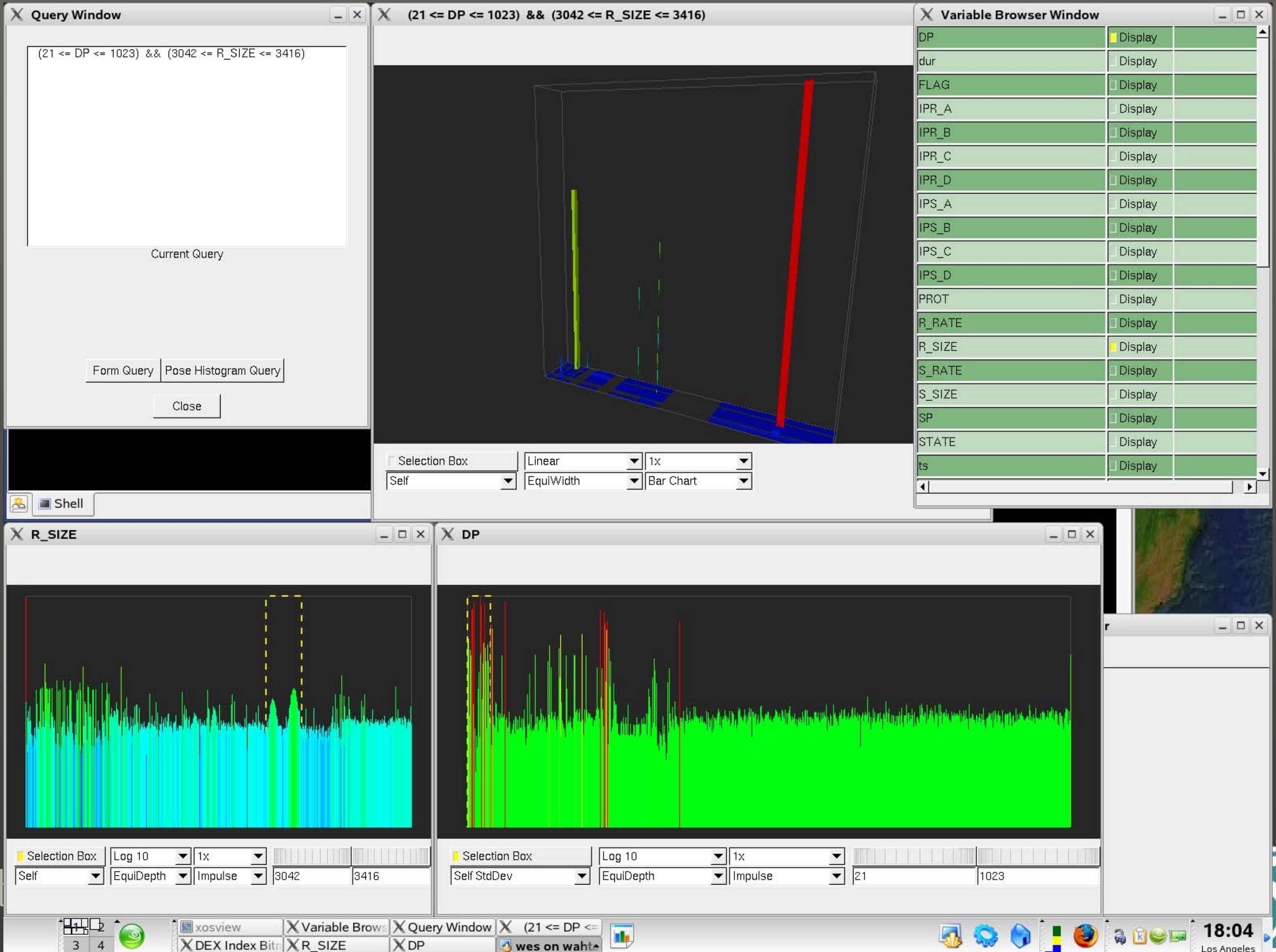
PEs	Shell-based	ROOT/Projection Index	ROOT/FastBit
1	156381.14	1357.07	5.36
2	71835.32	600.05	3.72
6	21952.12	214.14	2.66
13	9389.96	113.88	2.58
21	2237.53	98.95	2.05



Conditional 2D histogram processing time

Query: (1000 < DP < 11000) AND (50 <= tsyday <= 350) AND (state==1) AND (12 <= tshour <=14), 10K total bins.

QDV – Histogram “Cross-Products”



Query Window – Current Query:

```
(21 <= DP <= 1023)  &&  (3042 <= R_SIZE <= 3416)
```

3D histogram window: (21 <= DP <= 1023) && (3042 <= R_SIZE <= 3416)

Selection Box | Linear | 1x
Self | EquiWidth | Bar Chart

Variable Browser Window:

DP	Display
dur	Display
FLAG	Display
IPR_A	Display
IPR_B	Display
IPR_C	Display
IPR_D	Display
IPS_A	Display
IPS_B	Display
IPS_C	Display
IPS_D	Display
PROT	Display
R_RATE	Display
R_SIZE	Display
S_RATE	Display
S_SIZE	Display
SP	Display
STATE	Display
ts	Display

R_SIZE window:

Selection Box | Log 10 | 1x
Self | EquiDepth | Impulse | 3042 | 3416

DP window:

Selection Box | Log 10 | 1x
Self StdDev | EquiDepth | Impulse | 21 | 1023

Taskbar: xosview | Variable Brows | Query Window | (21 <= DP <= | DEX Index Bit | R_SIZE | DP | wes on waht

vis.lt

QDV – Summary

- New capability: ability to focus visualization and analysis processing on interesting data.
  - Orders of magnitude faster than previous approaches.
  - Directly responsive to needs of scientific researchers.
  - Quantifiable reduction in data understanding duty cycle.
- Leverages state-of-the-art Scientific Data Management technology to accelerate searches.
- QDV concepts are general-purpose and scalable to 1000s of PEs.