PAVE: More Intuitive Performance Analysis

Martin Schulz
in collaboration with Todd Gamblin and Peer-Timo Bremer

Lawrence Livermore National Laboratory

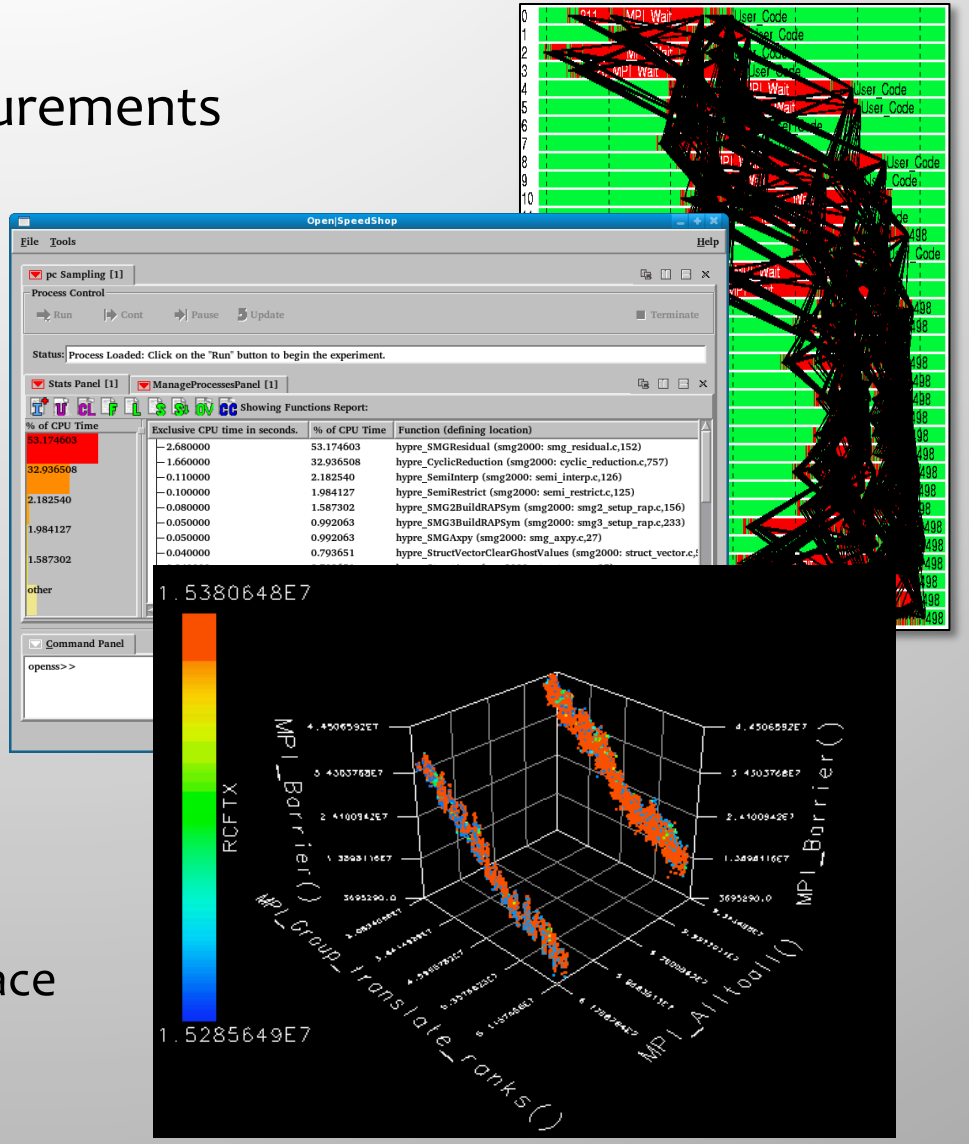
CScADS Workshop on Tools 2011

LLNL-PRES-491808

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

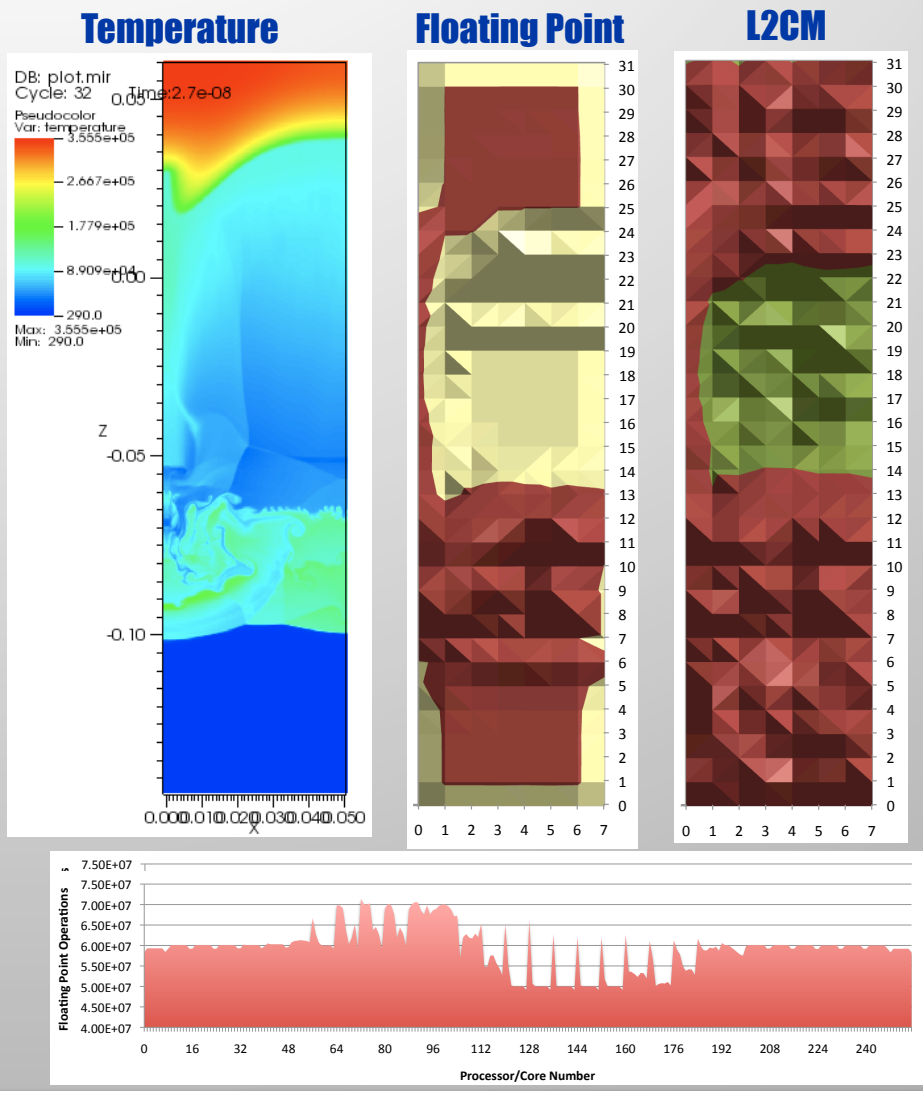
Versatility of Current Performance Analysis Tools

- Large variety of efficient measurements
  - Sampling/Tracing
  - Timings/Counters
- Easy instrumentation
  - Source code transformation
  - Binary rewriting
- Attribution to source code
- Sophisticated multi-metric visualization tools
- But: Interpretation often hard
  - Manual analysis by experts
  - Perspective is often MPI rank space
  - Little connection to applications



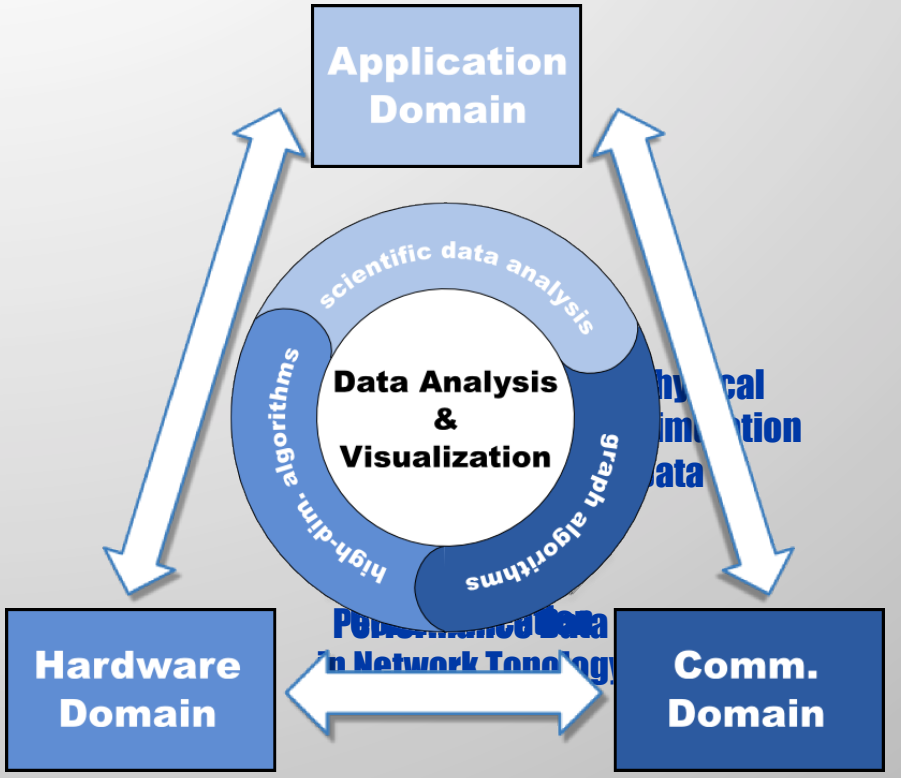
How to Make Analysis Easier/More Intuitive? Taking the Application Developer's Perspective

- Example: 256 core run of a CFD application
  - Floating point operations
- Application developers think in the app domain
- Simple step:
  - Map floating point ops onto the application domain
  - Similar L2 cache misses
- Clear correlations
  - Explains performance
  - Helps establish a baseline



A New Generation of Analysis Tools

- Tools must consider several domains / perspectives on data
  - Application domain
  - Hardware domain
  - Communication domain
- Visualize in the domains
- Inter domain mappings
  - Enable new perspectives
  - Analysis across domains
  - Use data analysis techniques

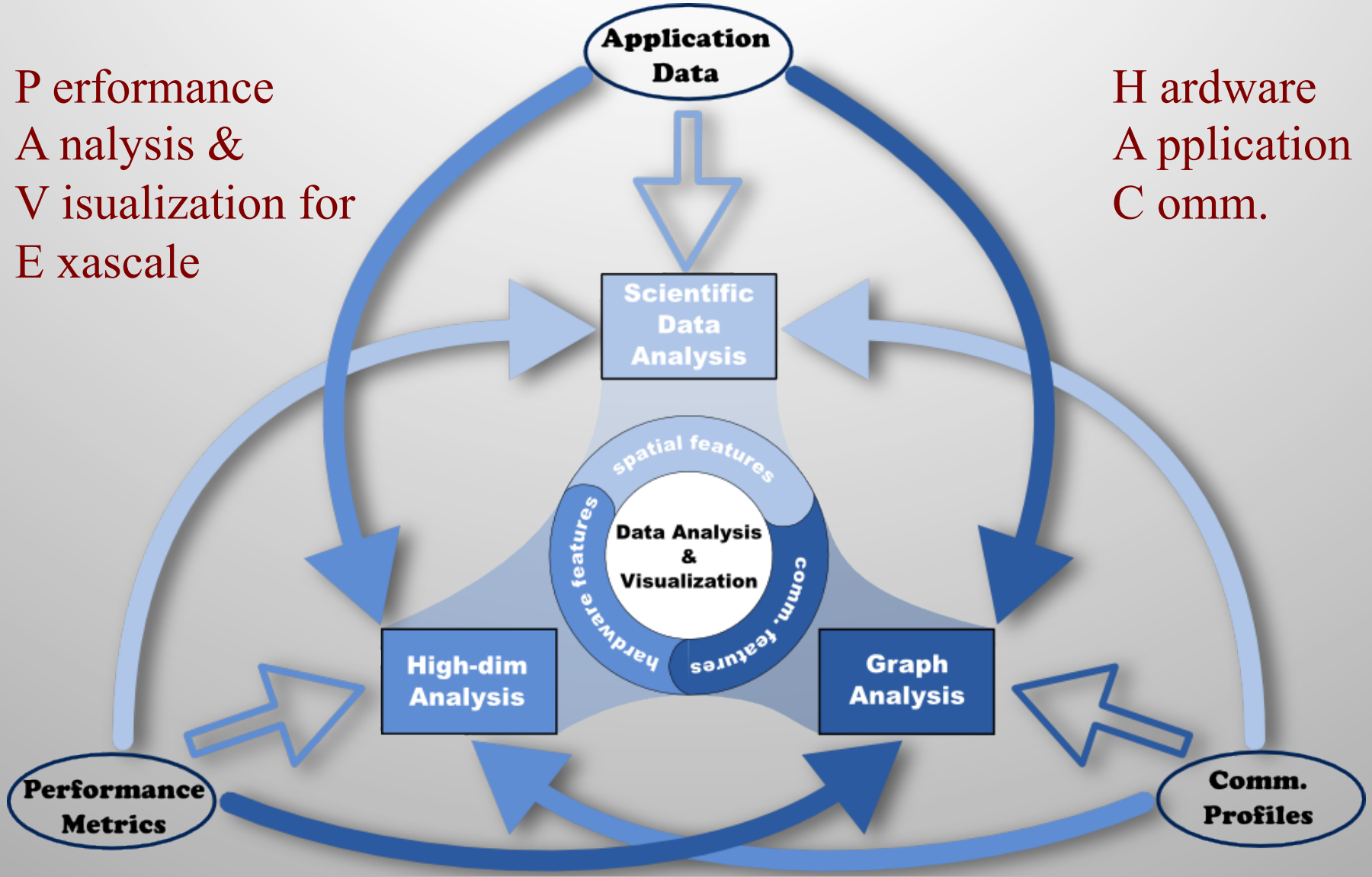


- PAVE: Performance Analysis & Visualization for Exascale
  - Joint project with Todd Gamblin and Timo Bremer
  - Collaboration with SCI@Utah (Valerio Pascucci and Joshua Levine)
  - Bridge between Performance & Visualization/Analysis

The PAVE/HAC Model

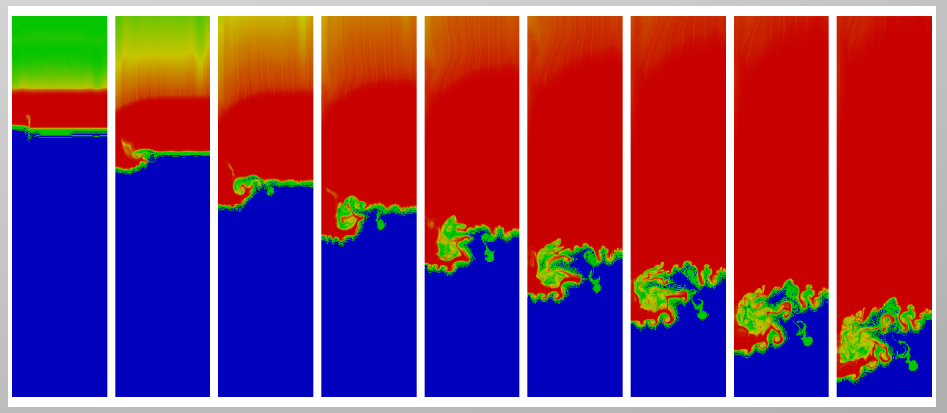
P erformance
A nalysis &
V isualization for
E xascale

H ardware
A pplication
C omm.

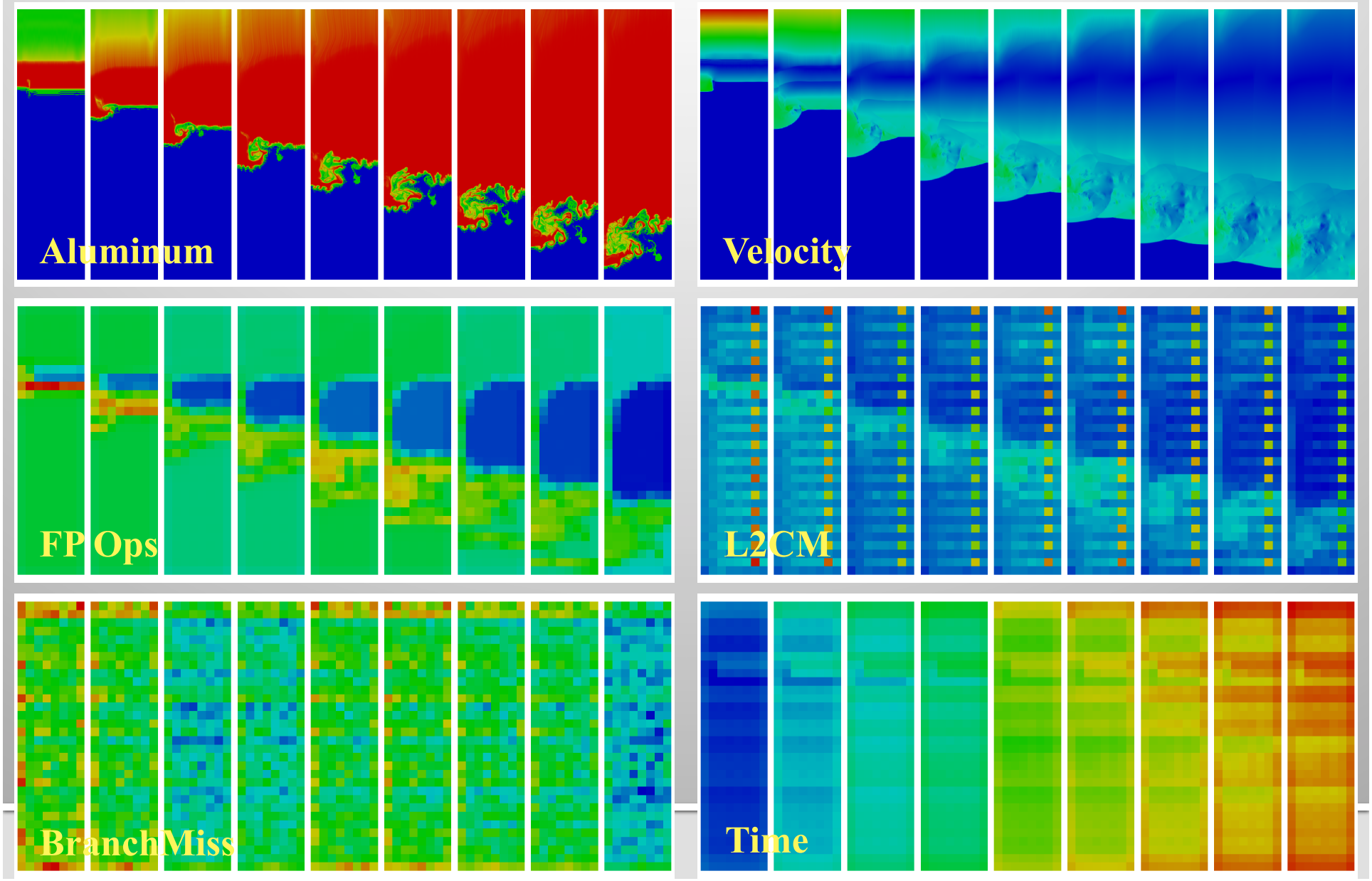


Mapping Measurements into the Application Domain

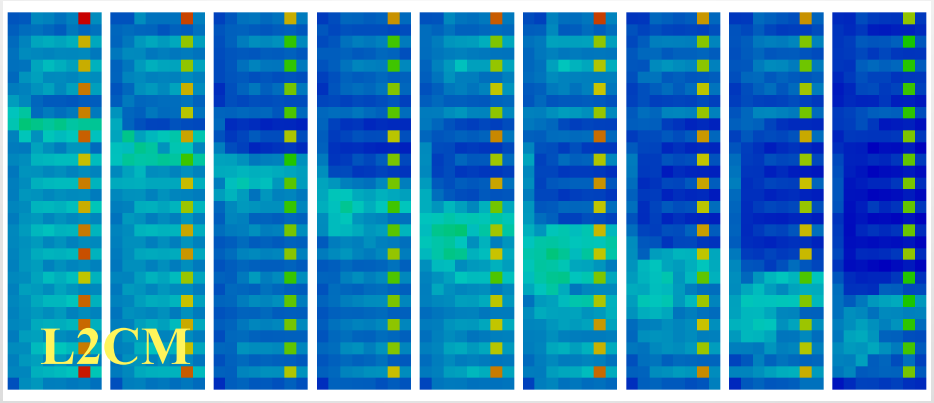
- Performance measurements acquired in the hardware domain
  - Map into the application's physical space
  - Visualize data in the application domain that is familiar to the developer
  - Ability to use existing and proven visualization techniques
- Requirements
  - Applications need to expose process ID -> grid mapping
  - In some cases we extract this automatically
  - Application independent API would be helpful
- Case study
  - CFD application
  - Shock wave caused by Aluminum jet
  - 2D version, 32x8 CPUs
  - 9 time steps



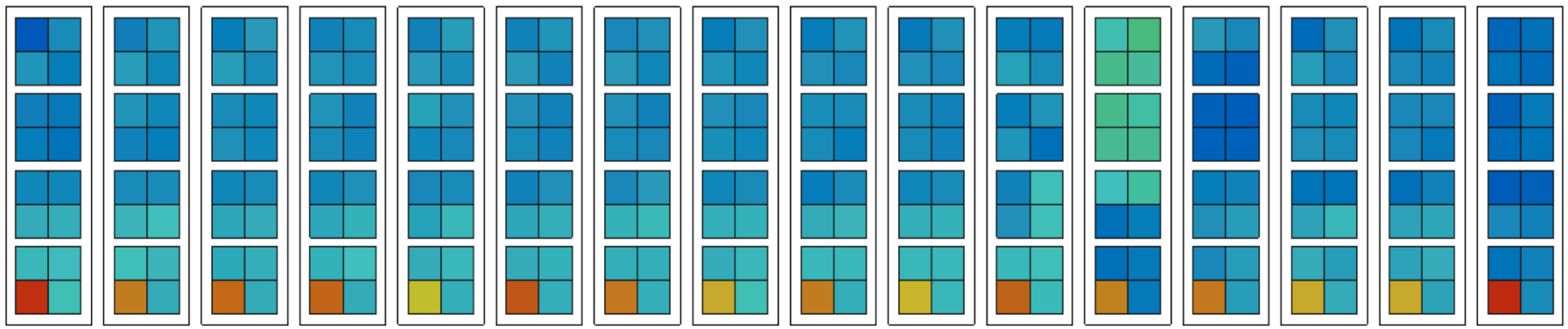
CFD Code, 9 Time Steps, 32x8 Processor Grid



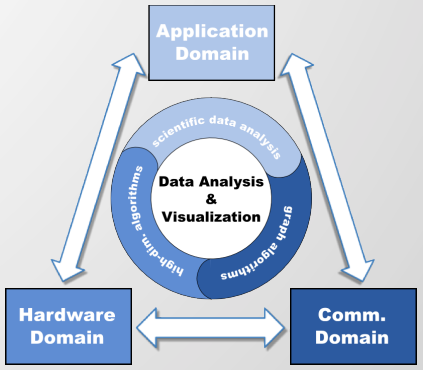
Disambiguating Effects



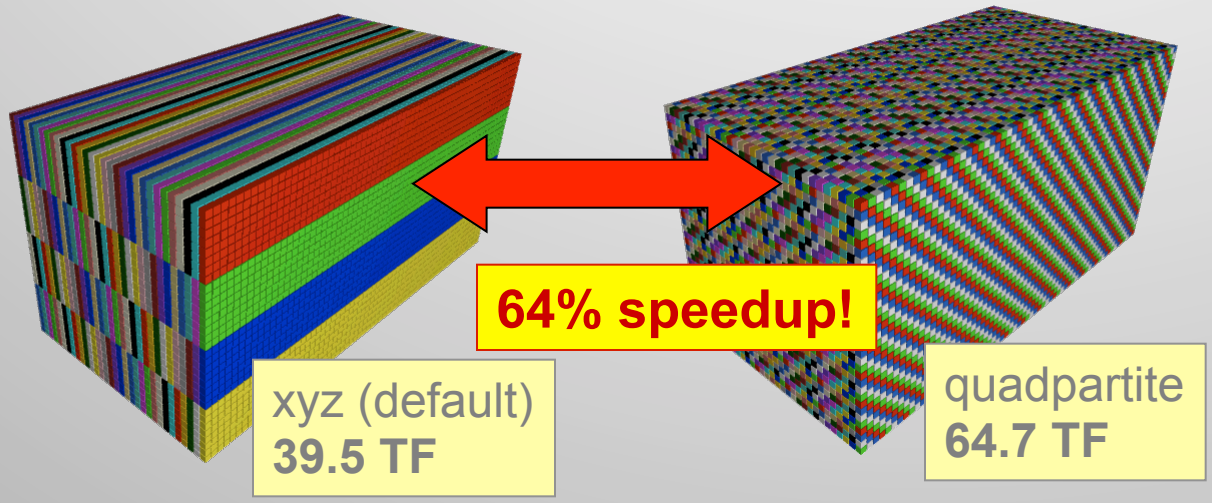
- Secondary, independent effect for L2 misses
  - Single core per socket creates more cache misses
  - Caused by the execution of collective MPI operations
  - Shows that we need different perspectives to disambiguate causes



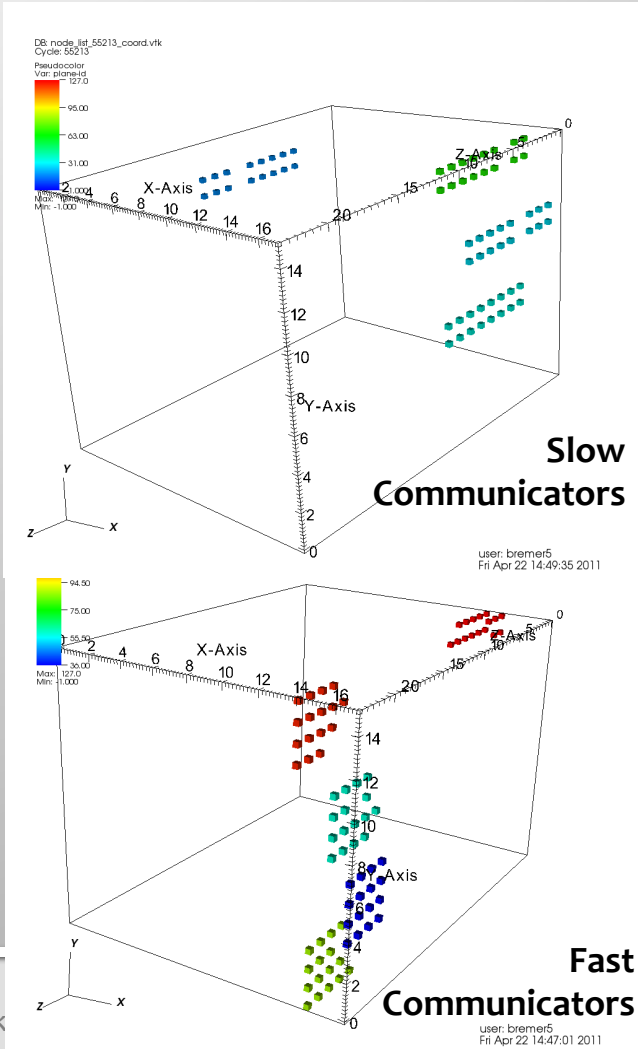
Hardware vs. Communication Domains



- Complex network topologies
  - Interactions with communication topology
  - Node placement performance critical

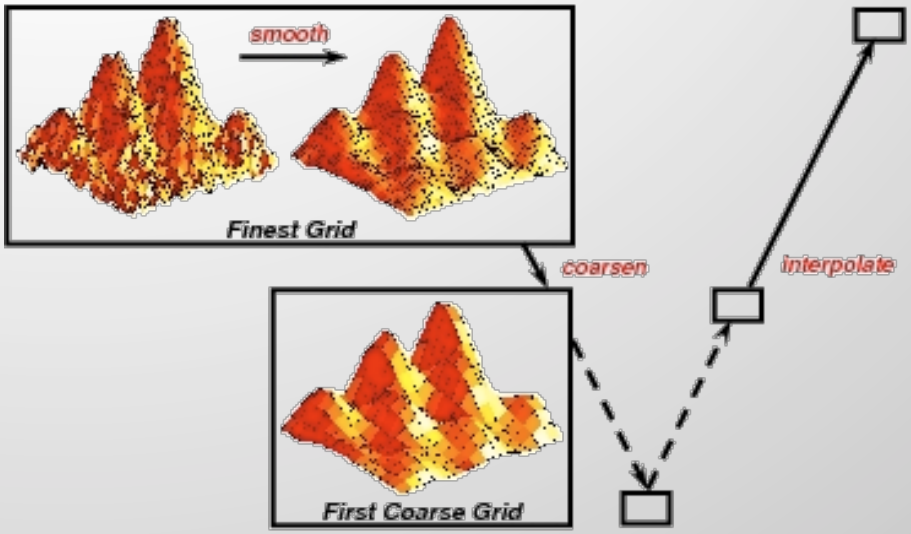


- Non compact mappings
  - Example: 3D plasma physics codes on XE-6
  - 1D communicators doing FFTs



Case Study: Algebraic Multigrid Solver (AMG)

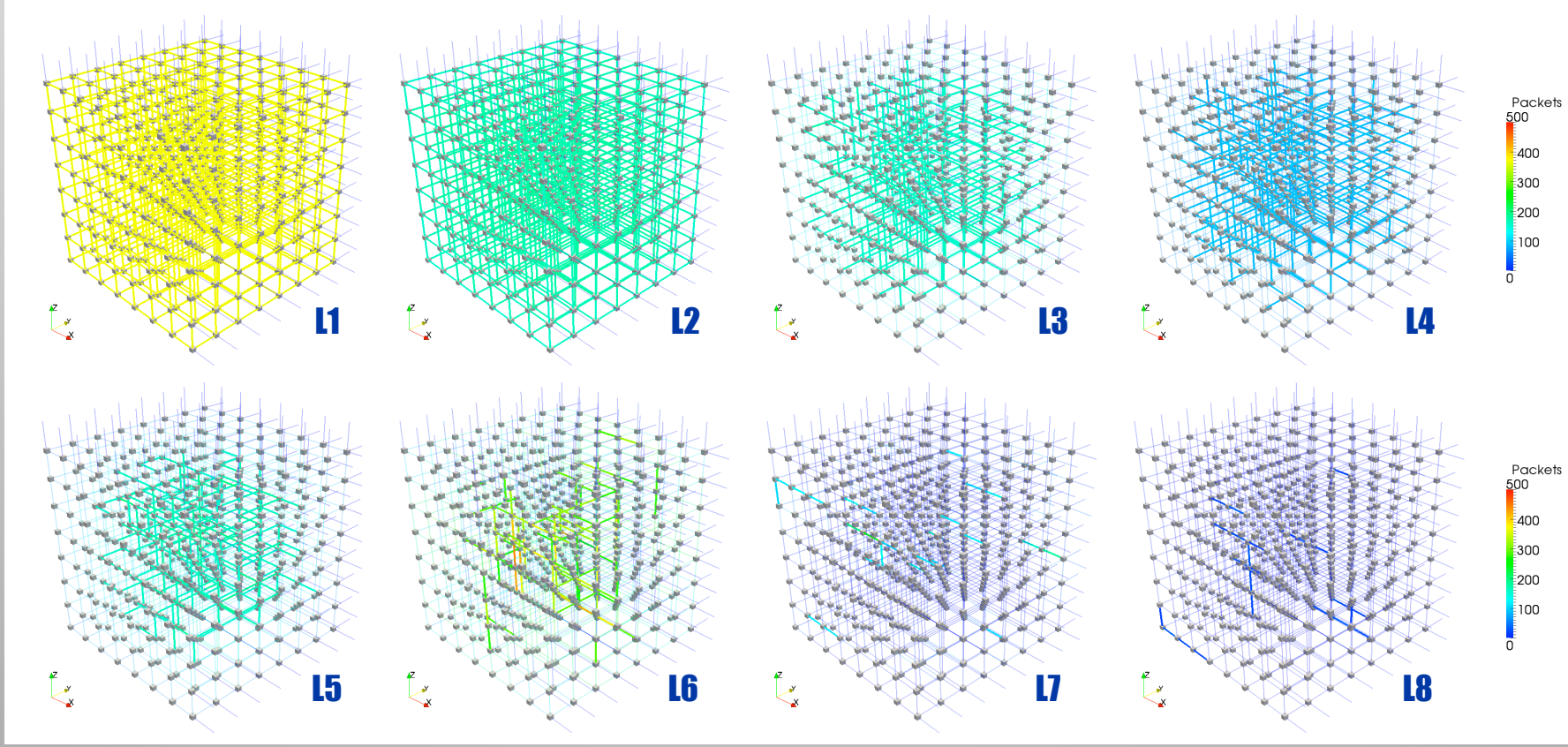
- Essential component for many applications
- Series of V Cycles
  - Coarsening
  - Direct solve
  - Interpolation



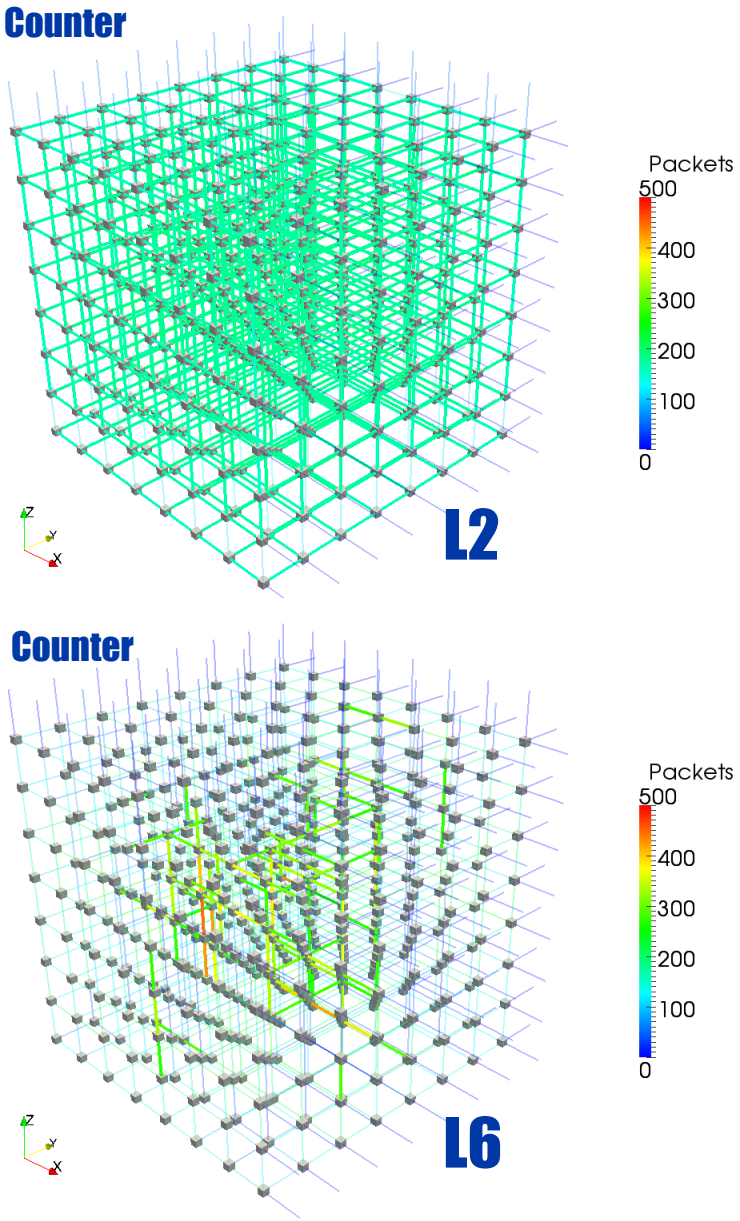
- Communication requirements change between layers
  - Fine layers have nearest neighbor communication
  - Coarse layers have more and less communication partners
  - Potential for link contention
- Experimental setup
  - AMG2006 on BG/P, 512 nodes/tasks
  - Measurements of X+/X-, Y+/Y-, Z+/Z- link activity

AMG on BG/L, 8x8x8 HW Torus, 8x8x8 virtual topology

- Communication counters for all eight levels of AMG
  - Mapped/Aggregated to the edges in a torus display

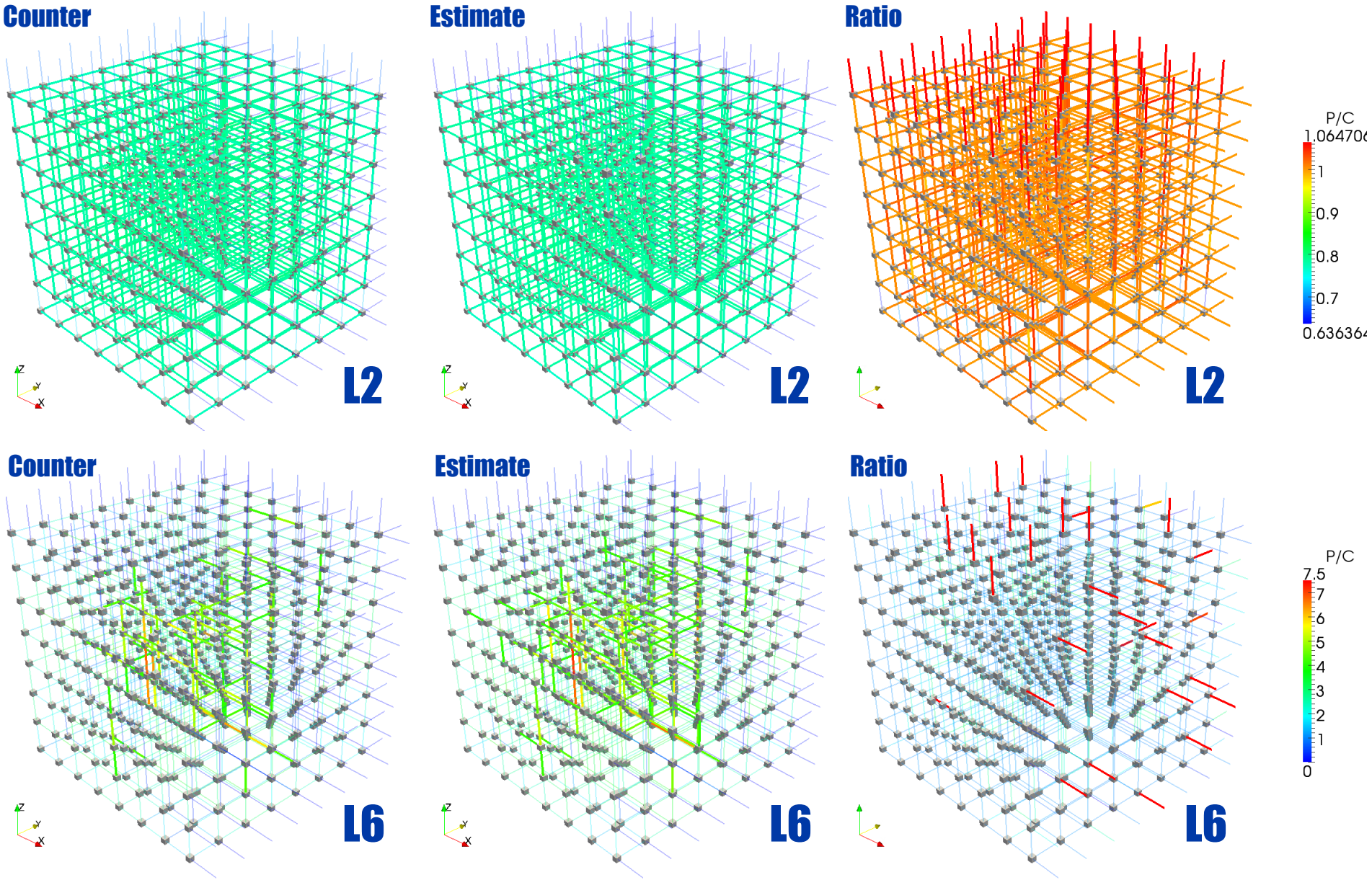


AMG on BG/L, 8x8x8 HW Torus, 8x8x8 virtual topology



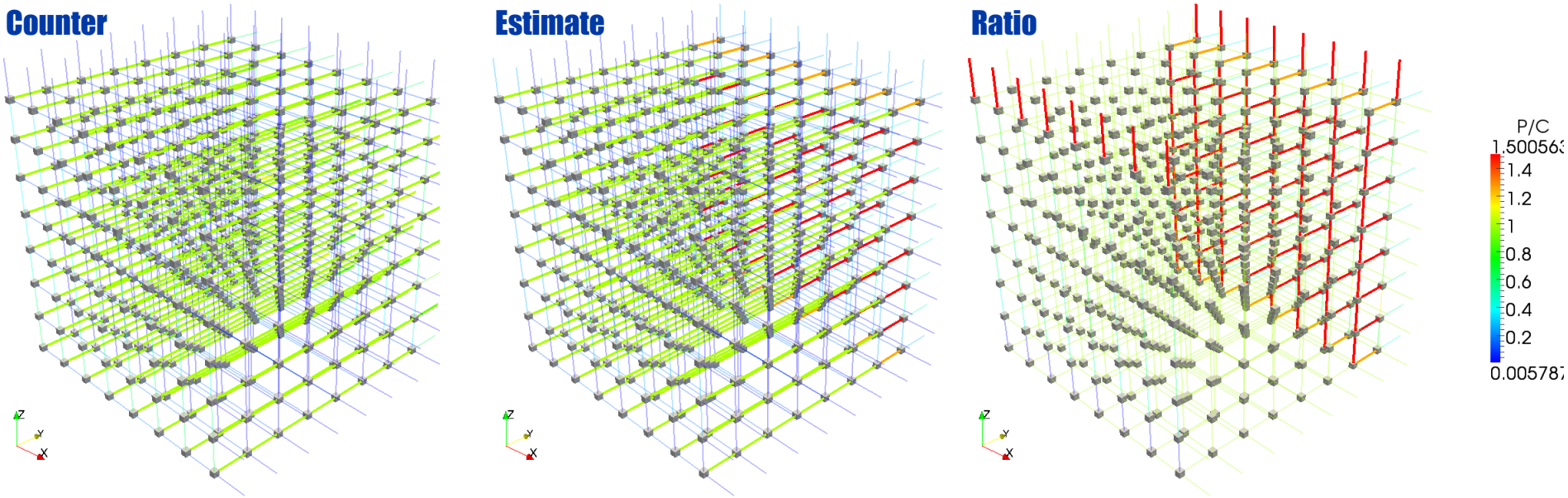
- New visualization that shows the hardware topology (level 2 vs. 6)
  - Finer layers are more global
  - Coarse layers have fewer partners
- How can we interpret the data?
  - Need connection to MPI communication
  - Need a baseline to compare to
- Map Communication to HW domain
  - Gather full MPI communication matrix
  - Emulate each message based on observed patters and aggregate
  - Contrast estimate with measurements
  - Ability to detect hotspots/contention

AMG on BG/L, 8x8x8 HW Torus, 8x8x8 virtual topology



Observation

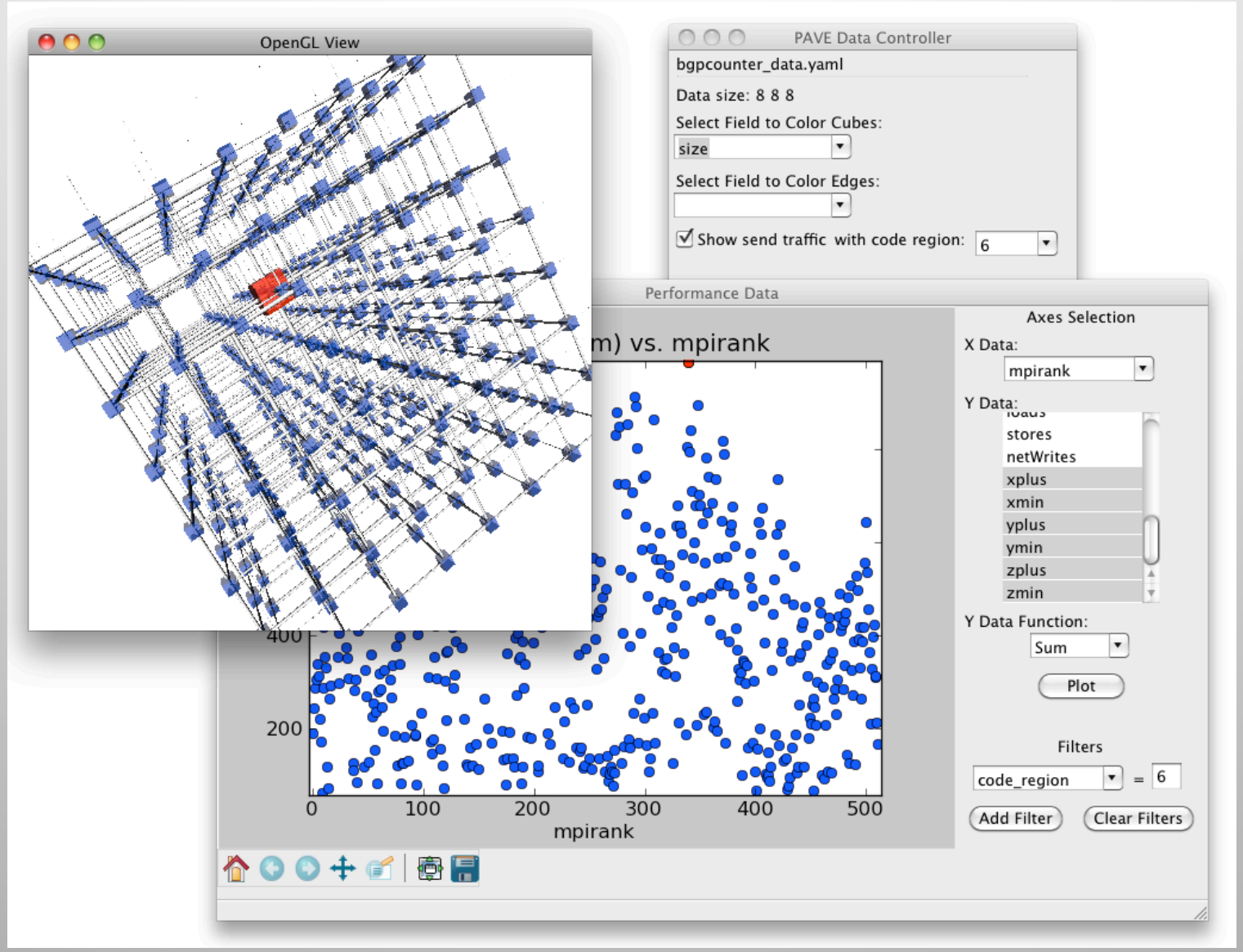
- Identify communication sparsity
  - Communication displays provide good insight
  - Leverage data analysis and visualization techniques
- Experiments with non optimal decompositions
  - BG/L, 8x8x8 HW Torus, 2x4x64 virtual topology
  - Results show more potential bottlenecks, but ratio is small



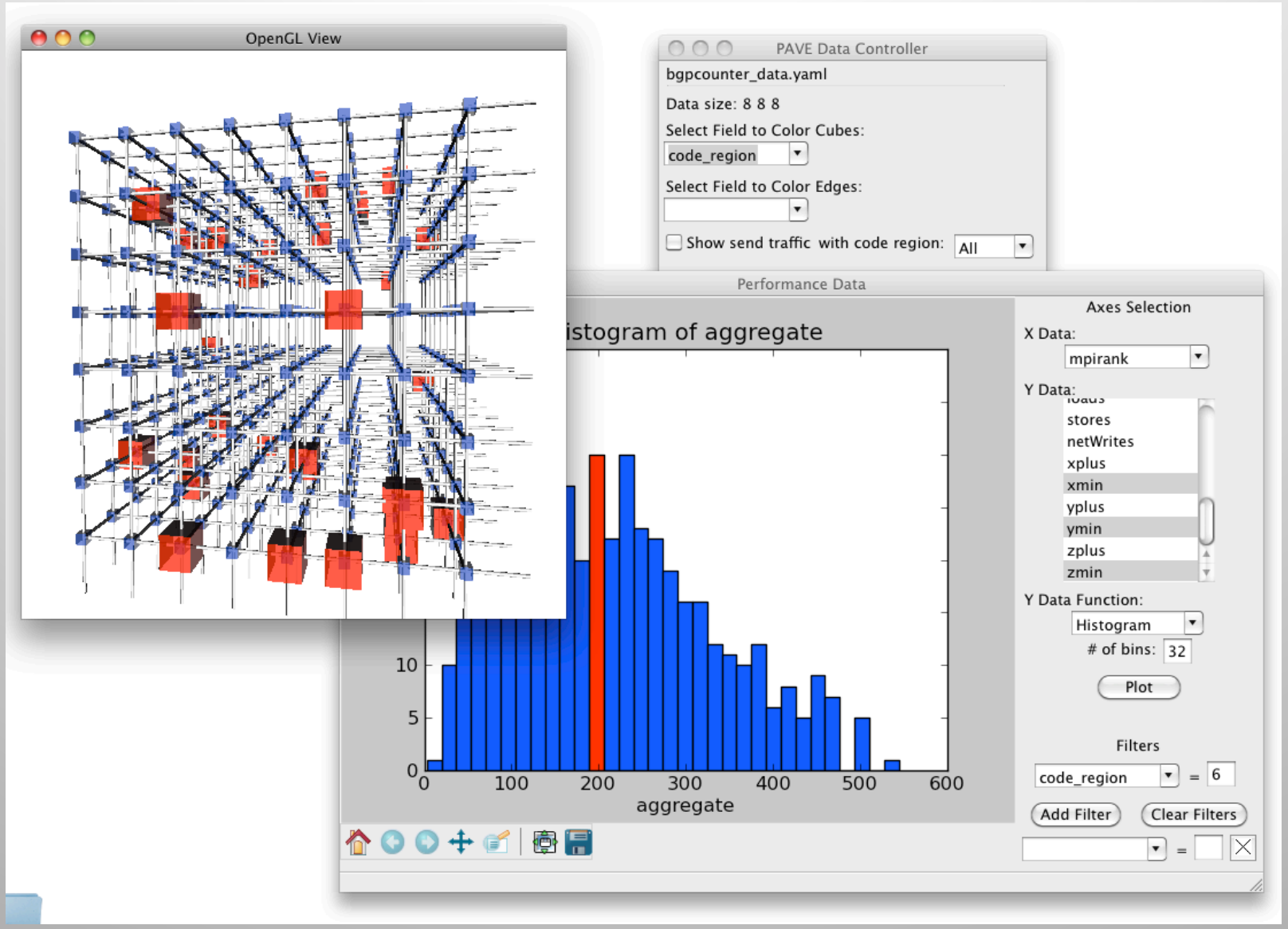
Next Steps: New Visualizations in the Hardware Domain

- Improved network emulation
  - Addition of collective communication
  - Support for multiple (sub)communicators
- Interactive node identification
  - Select and highlight node groups
  - Flexible selection criteria based on statistical analysis
  - Interactive node selection and manipulation
- Integrated analysis
  - Statistical evaluation of communication patterns
  - Integration of application patterns and groups
  - Reverse mapping to communication domain

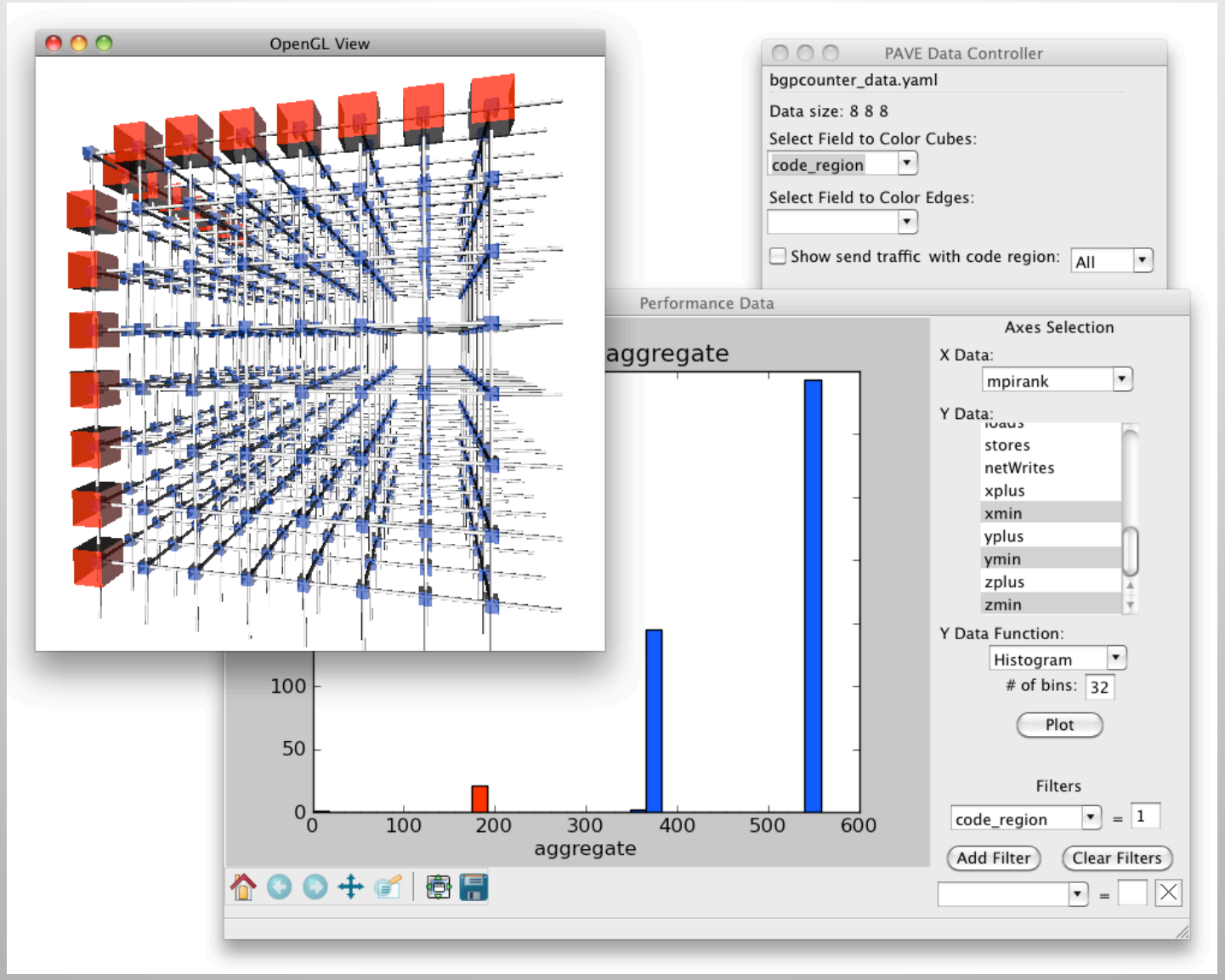
Early View on a Prototype Tool



Early View on a Prototype Tool



Early View on a Prototype Tool



New Perspectives Lead To New Insights

- New and easy insights by using multiple domains
  - Makes performance results tangible
  - Enables multiple perspectives to disambiguate effects
- Long term directions for PAVE
  - Using C->H mapping for topology optimizations
  - Integration of feature detection in the application domain
  - Exploiting graph-based techniques in the communication domain
- Multiple pieces of the puzzle have to come together
  - Information interfaces from all system layers
  - Export ability of application specific information
  - Interfaces with existing and future tools
  - Integration with a modular measurement environment
- **We need more intuitive performance analysis to help users understand the performance of their codes in their domains**