U.S. DEPARTMENT OF ENERGY

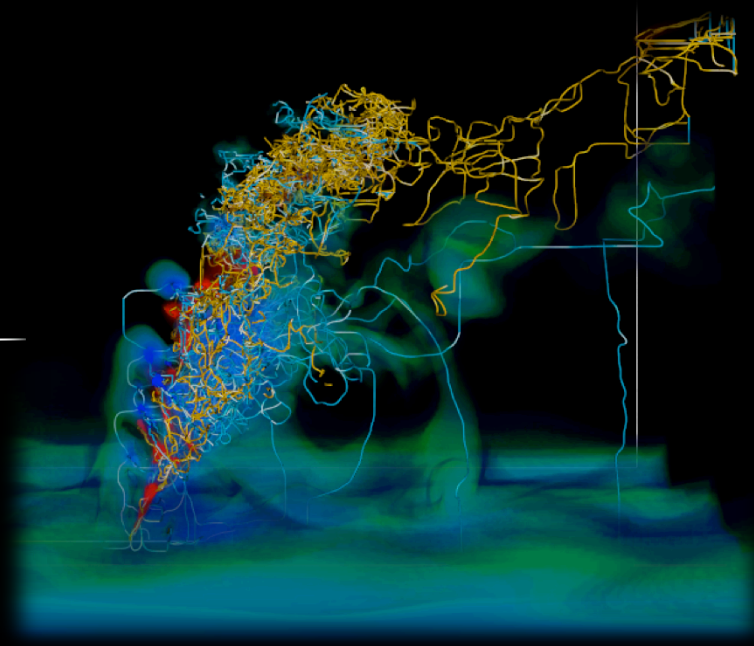
Argonne NATIONAL LABORATORY

"I have had my results for a long time, but I do not yet know how I am to arrive at them."

–Carl Friedrich Gauss, 1777-1855

Visualization and Data Analysis Tools

Morse-Smale Complex of combustion in the presence of a cross flow (image courtesy Attila Gyulassy)



Tom Peterka

tpeterka@mcs.anl.gov

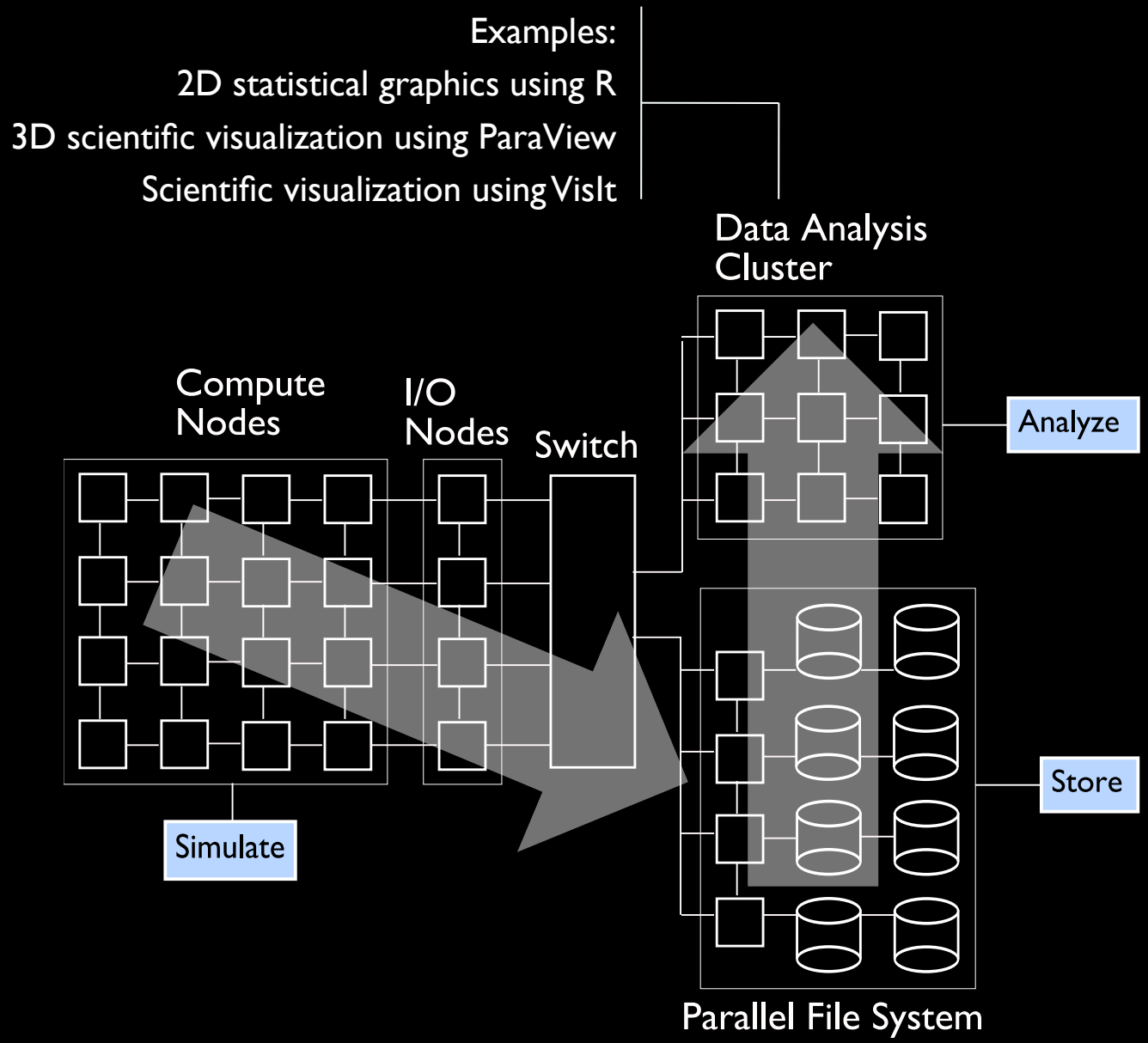
Mathematics and Computer Science Division

Executive Summary

-Postprocessing tools
    -Interactive (+)
    -Full-featured (+)
    -Scalability limited by data size (-)
    -High latency results (-)
    -Examples: R, ParaView, VisIt
-Run-time tools
    -Coprocessing
    -In Situ
    -Direct data access (+)
    -Low latency results (+)
    -Interaction more difficult (-)
    -Limited features (-)
    -Examples: ADIOS, GLEAN, DIY

Postprocessing Tools

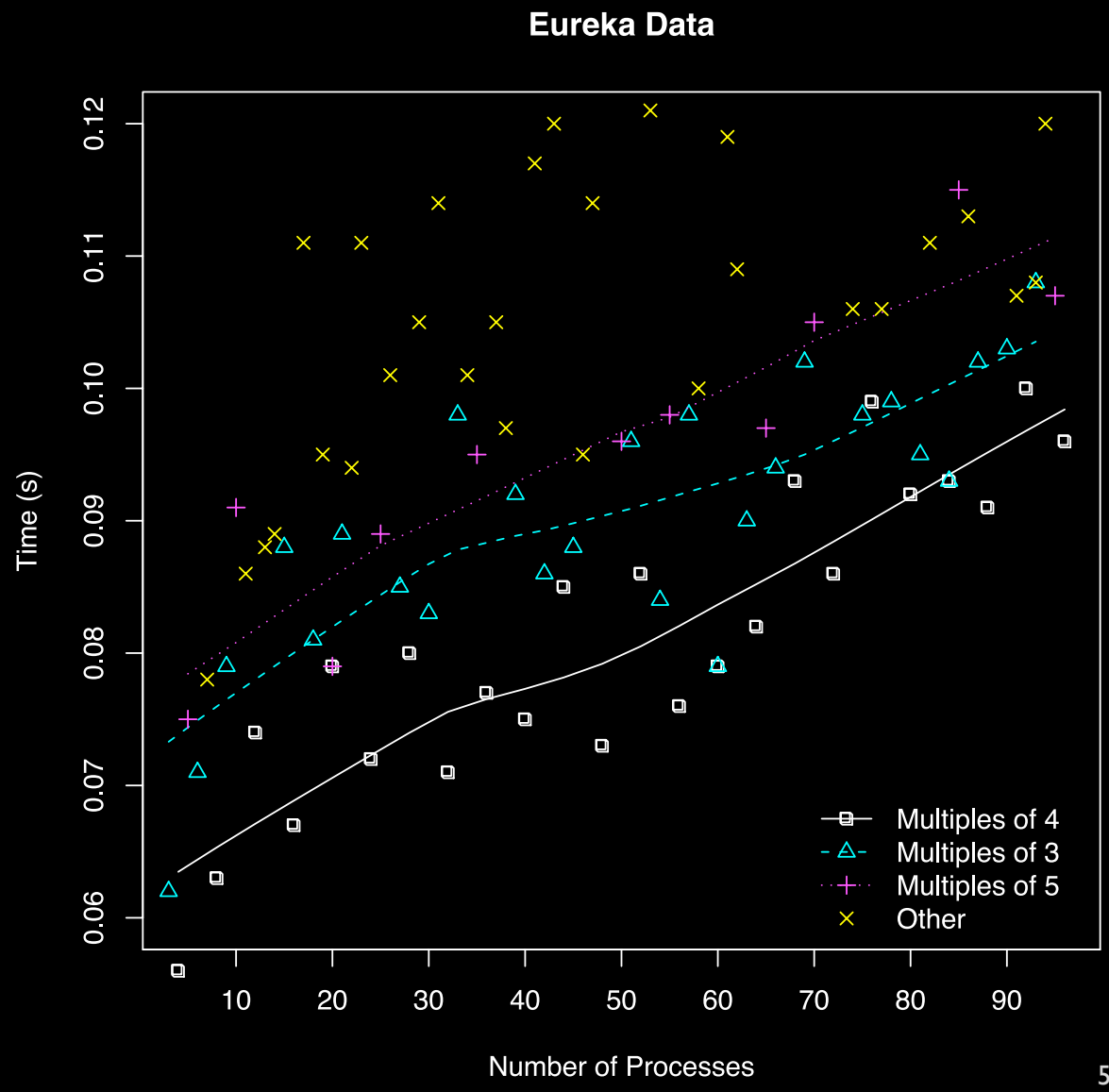
Postprocessing Scientific Data Analysis in HPC Environments



Statistical Graphics:

http://r-project.org/

-S (1976) John Chambers, Bell Labs
-R (1993) R. Gentleman and R. Ihaka, Auckland
-~250K – 1M users
-Steep learning curve (3000-page manual)
-Merges statistics with plotting
-Powerful plotting features



Data Pipeline Tools

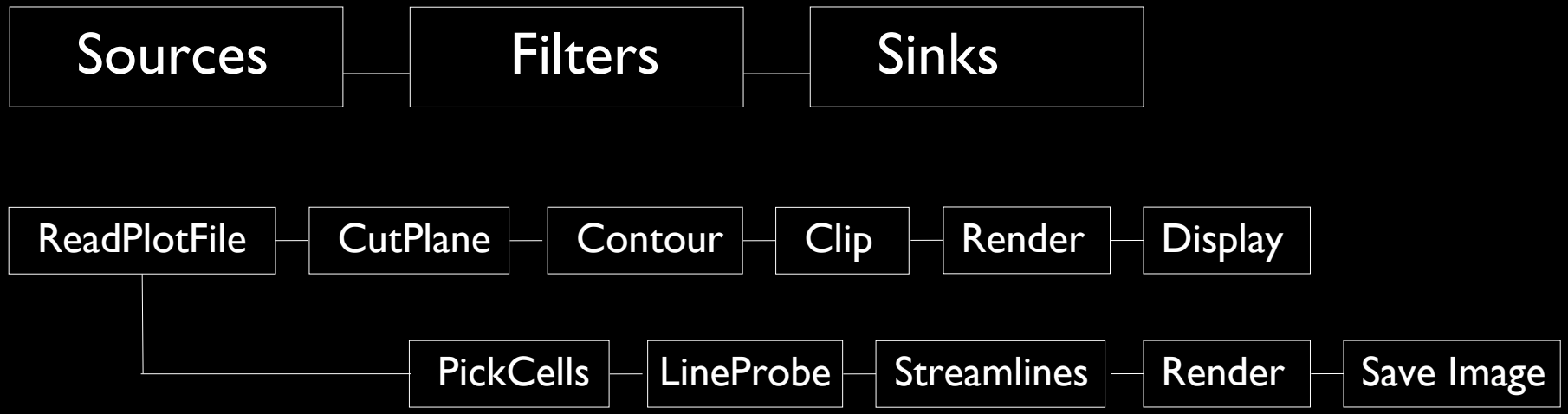
Data analysis as a series of transformations

-Source, filters, and sink

-VTK (Schroeder, Martin, Lorensen 1993)

-Many tools on top of VTK: ParaView, VisIt, VisTrails

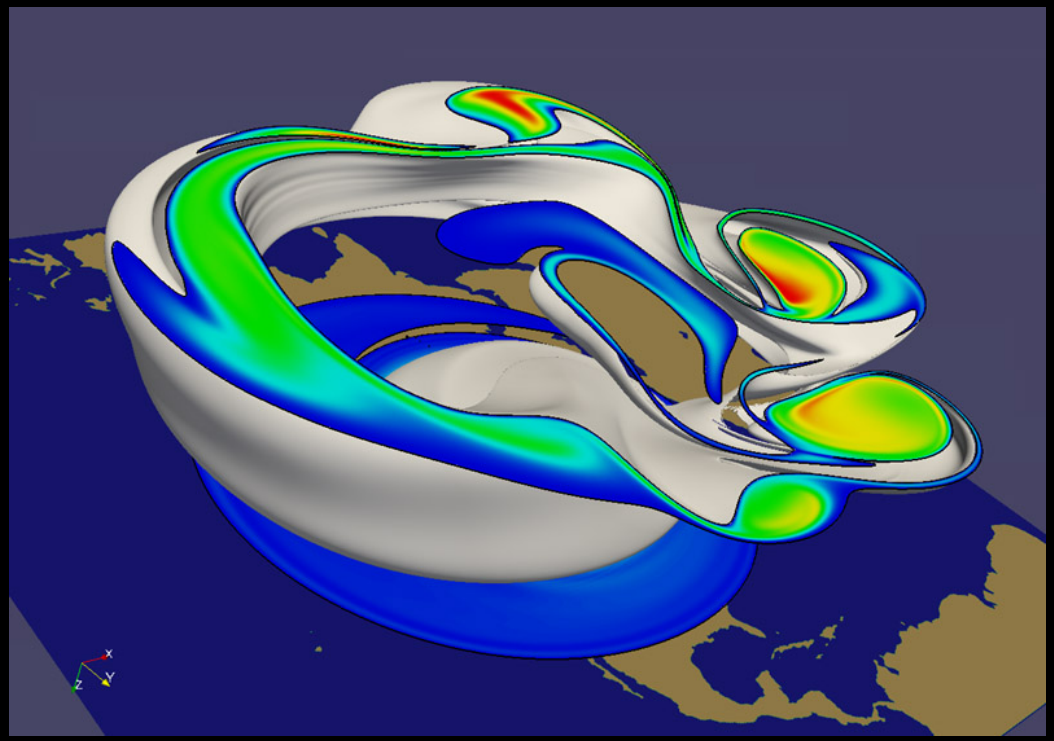
-Code reuse, portability, standardization



3D & 4D Scientific Visualization: ParaView

http://www.paraview.org

-Started in 2000 with Kitware and LANL, later included SNL and ARL
-VTK engine
-Qt interface
-Contacts: Ken Moreland (SNL), Berk Geveci (Kitware)
-Tutorials at SC, SciDAC, elsewhere
http://www.itk.org/Wiki/ParaView_2.X_documentation_and_tutorials



0.5 billion-cell weather visualization courtesy Ken Moreland

Advanced ParaView: Client-Server Mode

On Eureka:
-Add a few one-time items to .softenvrc, .bashrc
-Grab nodes in interactive mode for a time:

```
qsubi -n 4 -t 60
```

-Start the pvserver:

```
mpirun -np 4 -machinefile $COBALT_NODEFILE /soft/apps/
paraview-3.4.0-mpich-mx/bin/pvserver
```

On local machine:
-Setup a tunnel:

```
ssh -NL 11111:vs37:11111 username@eureka.alcf.anl.gov
```

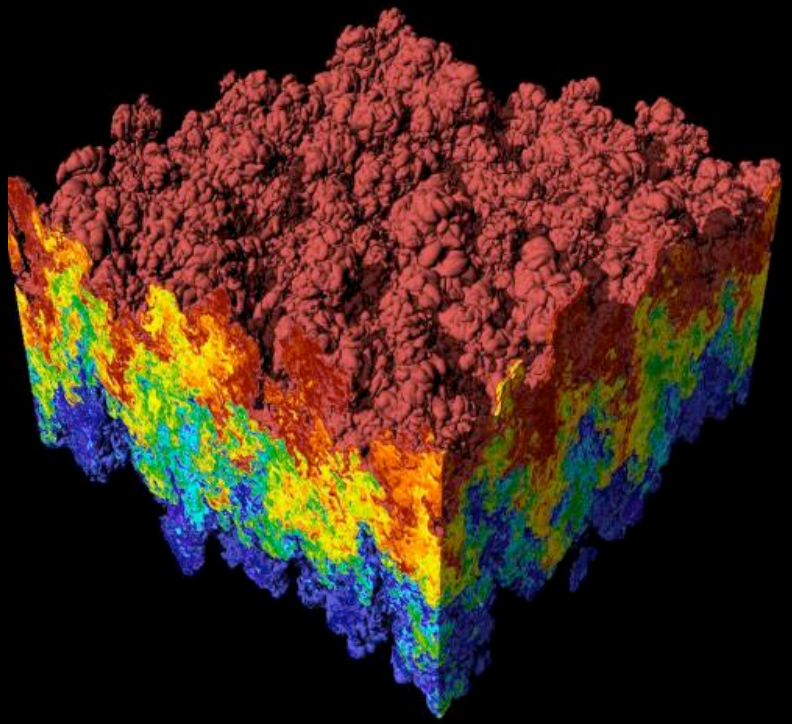
-Start ParaView, configure connection, connect
-Beware to have matched ParaView versions between client and server
http://paraview.org/paraview/resources/software.html

Eureka setup instructions at
https://wiki.alcf.anl.gov/index.php/Paraview_on_the_Data_Analytics_Cluster

3D & 4D Scientific Visualization:

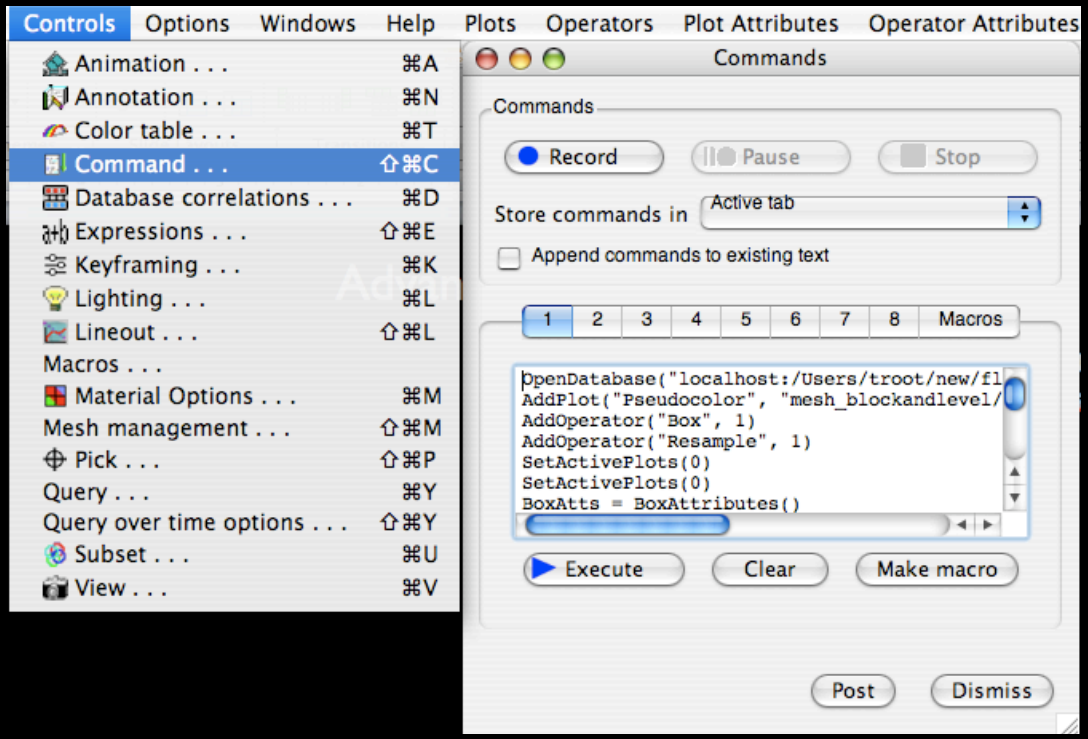
https://wci.llnl.gov/codes/visit/home.html

-Started in 2000 at LLNL as an ASCI-funded program
-VTK-like engine
-Qt interface
-Contacts: Hank Childs (BNL, UC-Davis), Jeremy Meredith (ORNL)
-Tutorials at SC, SciDAC, elsewhere
https://wci.llnl.gov/codes/visit/1.4.1/VisualizationWithVisIt.pdf



Rayleigh-Taylor Instability visualization courtesy Hank Childs

Advanced VisIt: Scripting Mode



Capture the script with Controls | Command and record

Save in script.py.

Run with:

visit –cli –nowin –s script.py

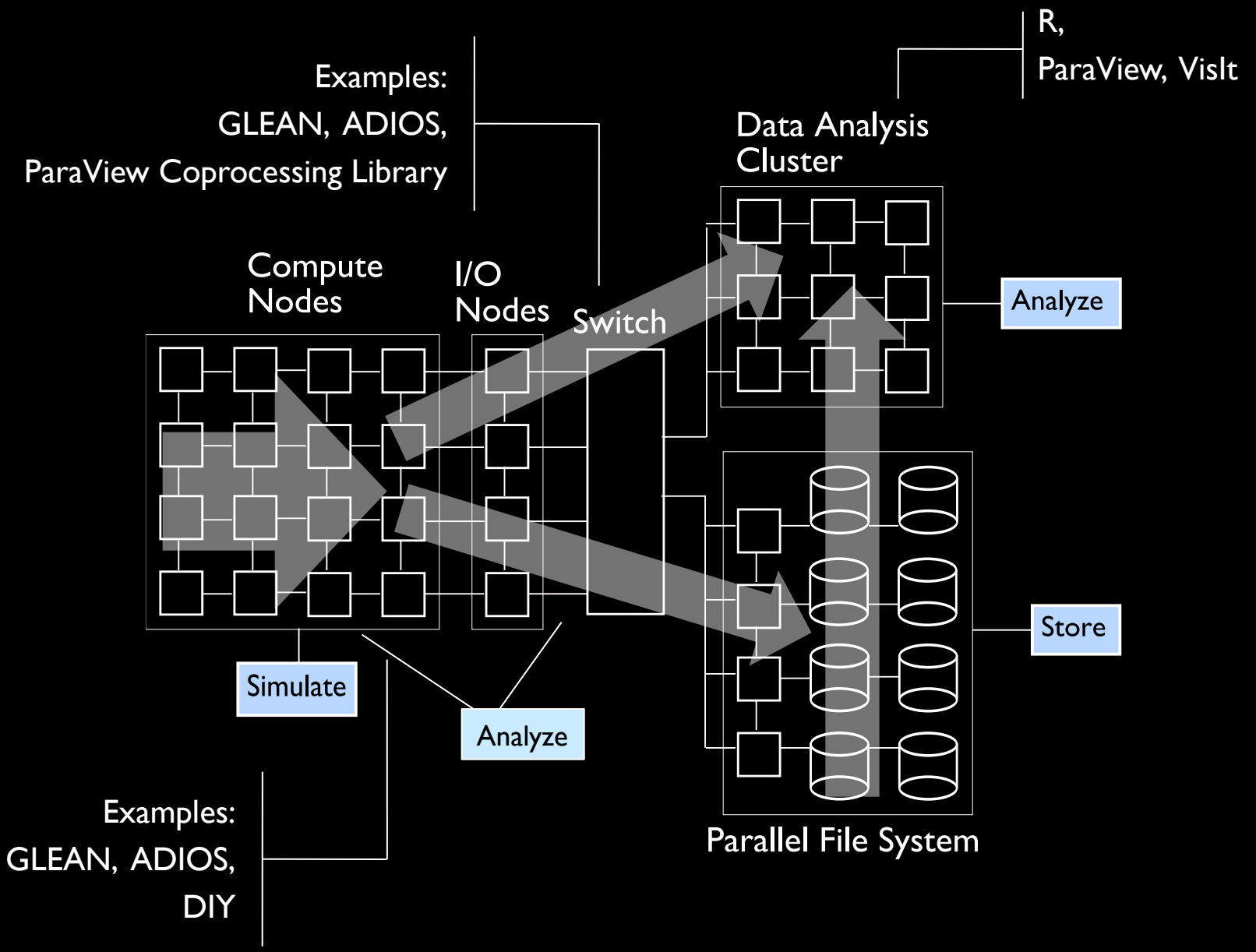
```
OpenDatabase("localhost:/filename", 0)
AddPlot("Pseudocolor", "velx", 1, 1)
AddOperator("Box", 1)
AddOperator("Resample", 1)
SetActivePlots(0)

SetActivePlots(0)
BoxAtts = BoxAttributes()
BoxAtts.amount = BoxAtts.Some
BoxAtts.minx = -0.4
BoxAtts.maxx = 0.4
BoxAtts.miny = -0.4
BoxAtts.maxy = 0.4
BoxAtts.minz = -0.4
BoxAtts.maxz = 0.4
SetOperatorOptions(BoxAtts, 1)

DrawPlots()
ExportDBAtts = ExportDBAttributes()
ExportDBAtts.db_type = "BOV"
ExportDBAtts.filename = "0.x"
ExportDBAtts.dirname = "."
ExportDBAtts.variables = "velx"
ExportDBAtts.opts.types = ()
ExportDatabase(ExportDBAtts)

quit()
```

What We Learned So Far

-No one tool can do it all
-Tools have steep learning curves, may require expert assistance
-Competing tools are often quite similar, choices often are not that critical
-There will always be a role for postprocessing tools

Run-time Tools

Run-time Scientific Data Analysis in HPC Environments



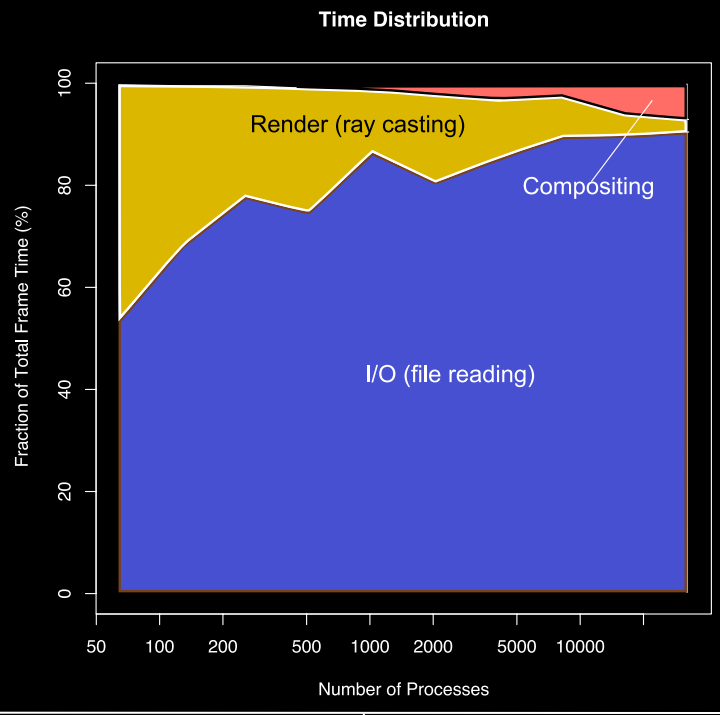
The Data-Intensive Nature of Computing and Analysis

"Analysis and visualization will be limiting factors in gaining insight from exascale data."
–Dongarra et al., International Exascale Software Project Draft Road Map, 2009.

Normalized Storage / Compute Metrics

Machine	FLOPS (Pflop/s)	Storage B/W (GB/s)	Bytes comp. per byte stored
LLNL BG/L	0.6	43	O(10^3)
Jaguar XT4	0.3	42	O(10^3)
Intrepid BG/P	0.6	50	O(10^3)
Roadrunner	1.0	50	O(10^4)
Jaguar XT5	1.4	42	O(10^4)

-In 2001, bytes computed per bytes stored was approximately 50.
Refs: John May, 2001, Murphy et al. ICS'05.



The relative percentage of time in the stages of volume rendering as a function of system size. Large visualization is dominated by data movement: I/O and communication.

Scalable Analysis & Visualization: The Data Parallel Approach

Treat analysis as any other parallel computation

-Decompose the domain

-Assign to processors

-Combine local and global operations

-Use parallel I/O, MPI, other programming models

-Balance load, minimize communication

-Measure strong, weak scaling, efficiency

Integrate with simulation

"The combination of massive scale and complexity is such that high performance computers will be needed to analyze data, as well as to generate it through modeling and simulation."

–Lucy Nowell, Scientific Data Management and Analysis at Extreme Scale, Office of Science Program Announcement LAB 10-256, 2010.

Tackling the Data-Intensive Part of Data Analysis

DIY: help the user write own data-parallel analysis algorithms.

Main ideas and Objectives

-Large-scale parallel analysis (visual and numerical) on HPC machines

-Scientists, visualization researchers, tool builders

-In situ, coprocessing, postprocessing

-Data-parallel problem decomposition

-Scalable data movement algorithms

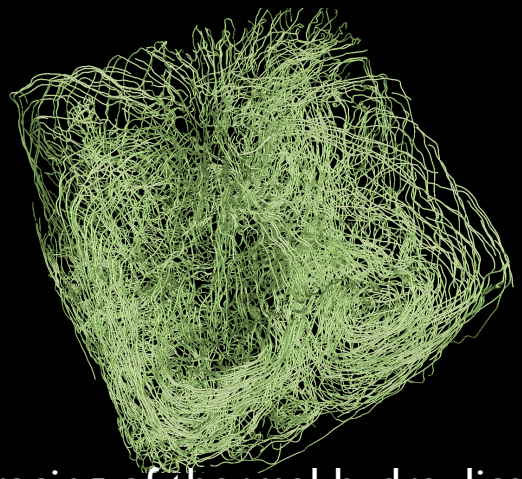
Benefits

-Researchers can focus on their own work, not on parallel infrastructure

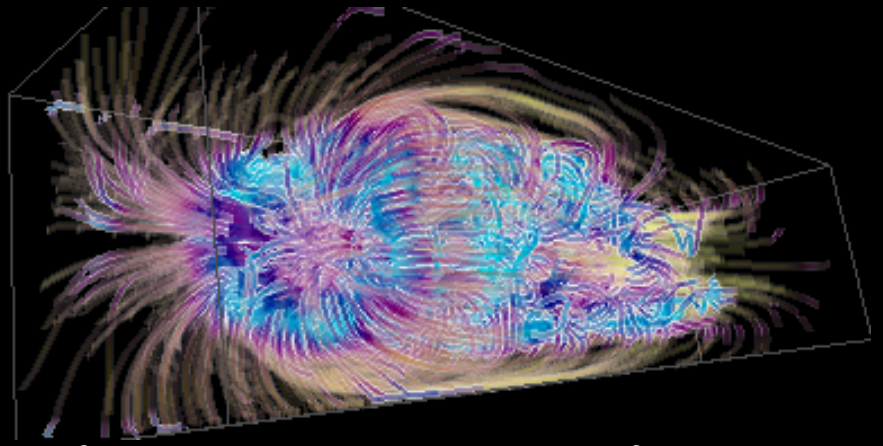
-Analysis applications can be custom

-Reuse core components and algorithms for performance and productivity

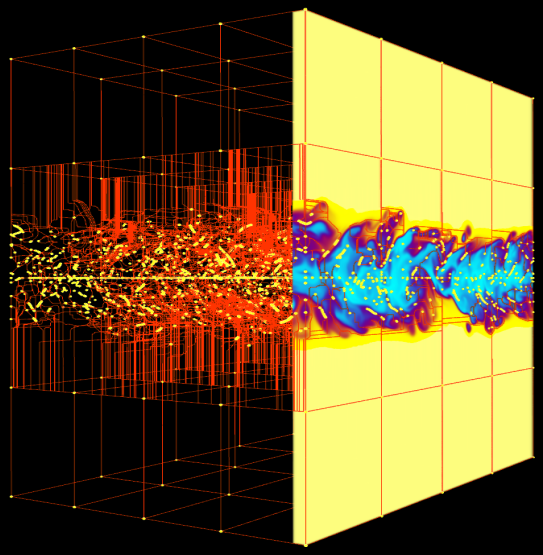
Data Analysis Comes in Many Flavors



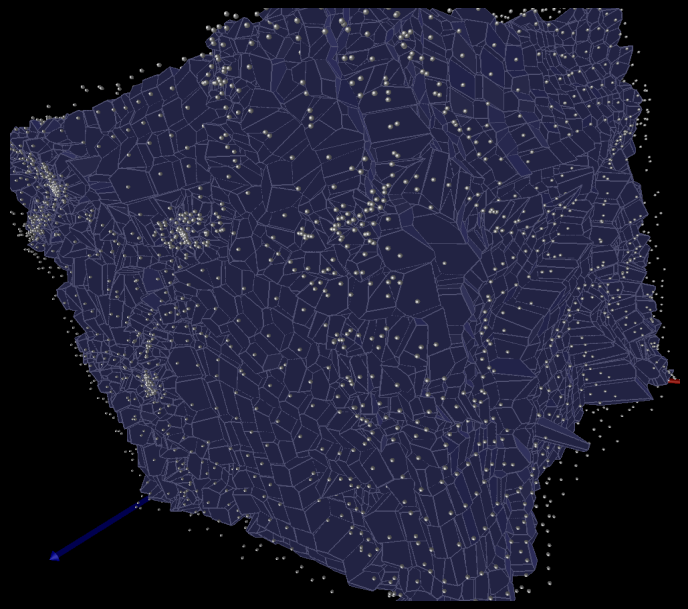
Particle tracing of thermal hydraulics flow



Information entropy analysis of astrophysics



Morse-Smale complex of combustion



Voronoi tessellation of cosmology

Separate Analysis Ops from Data Ops

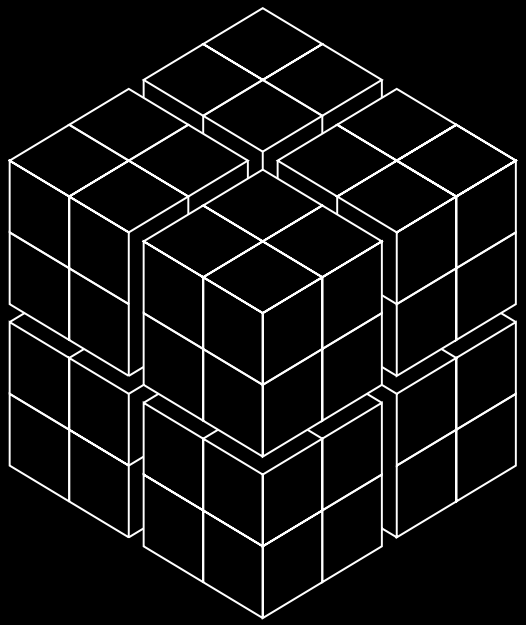
Analysis	Application	Application Data Model	Analysis Data Model	Analysis Algorithm	Communication	Additional
Particle Tracing	CFD	Unstructured Mesh	Particles	Numerical Integration	Nearest neighbor	File I/O, Domain decomposition, process assignment, utilities
Information Entropy	Astrophysics	AMR	Histograms	Convolution	Global reduction, nearest neighbor	
Morse-Smale Complex	Combustion	Structured Grid	Complexes	Graph Simplification	Global reduction	
Computational Geometry	Cosmology	Particles	Tessellations	Voronoi	Nearest neighbor	

You do this yourself

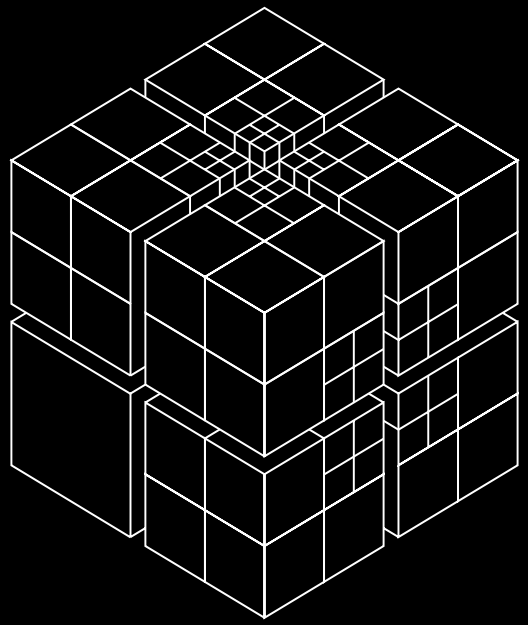
Can use serial libraries such as OSUFlow, Qhull, VTK (don't have to start from scratch)

DIY handles this

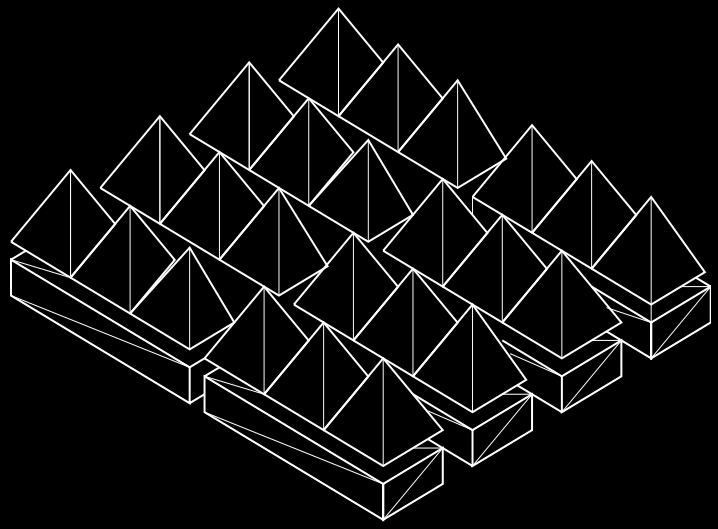
Group Data Items Into Blocks



Structured Grid



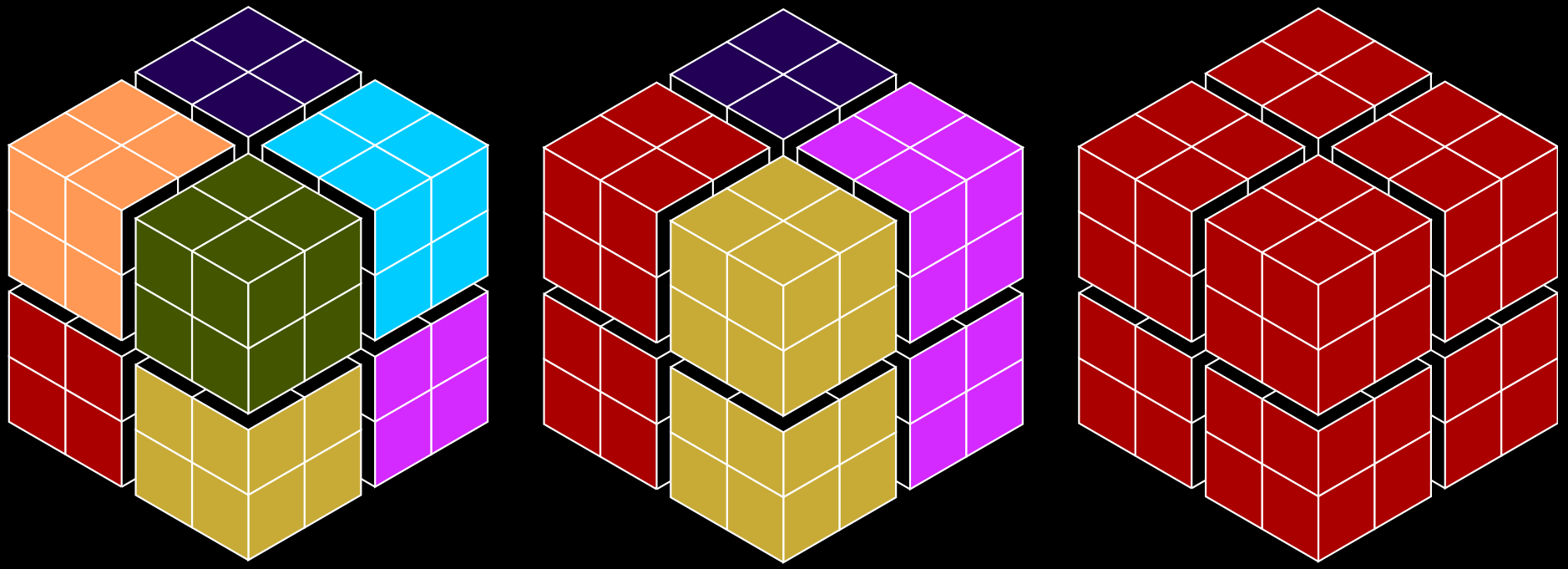
AMR Grid



Unstructured Mesh

The block is DIY's basic unit of data. Original dataset is decomposed into generic subsets called blocks, and associated analysis items live in the same blocks. Blocks contain one or more instances of the data type described earlier.

Block ≠ Process



8 processes

4 processes

1 process

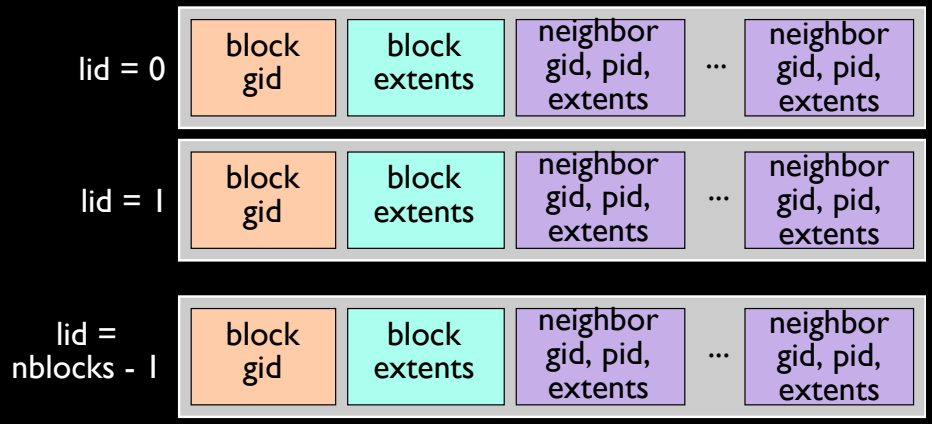
All data movement operations are per block; blocks exchange information with each other using DIY's communication algorithms. DIY manages and optimizes exchange between processes based on the process assignment. This allows for flexible process assignment as well as easy debugging.

Group Blocks into Neighborhoods

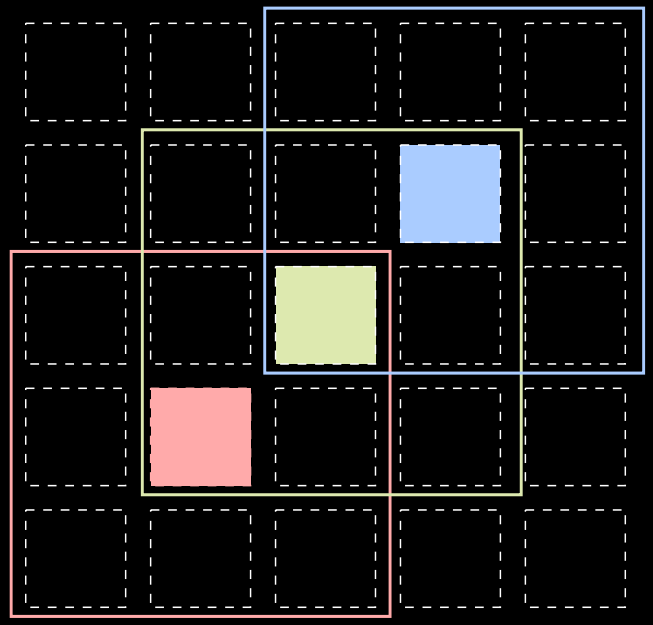
-Limited-range communication

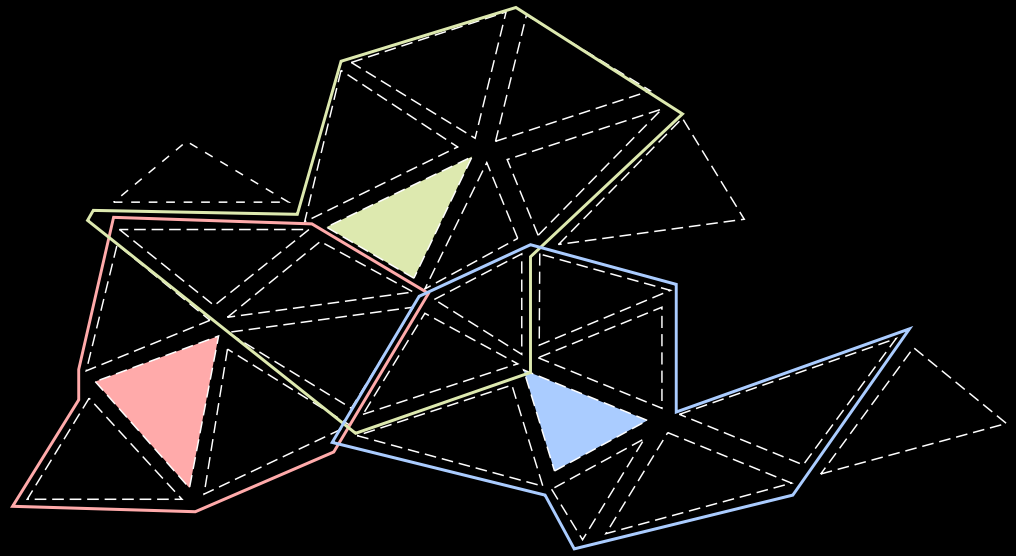
-Allow arbitrary groupings

-Distributed, local data structure and knowledge of other blocks (not master-slave global knowledge)



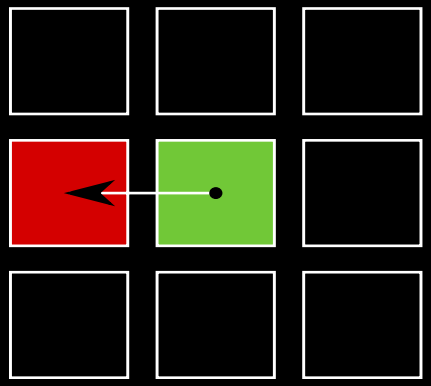
gid = global block identification
lid = local block identification
pid = process identification



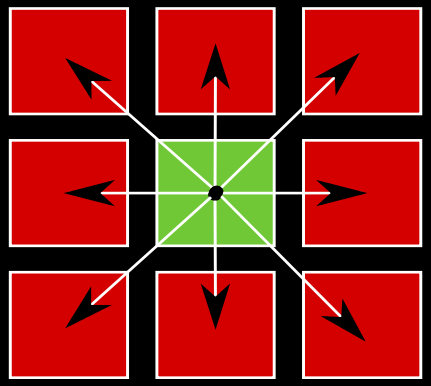


Two examples of 3 out of a total of 25 neighborhoods

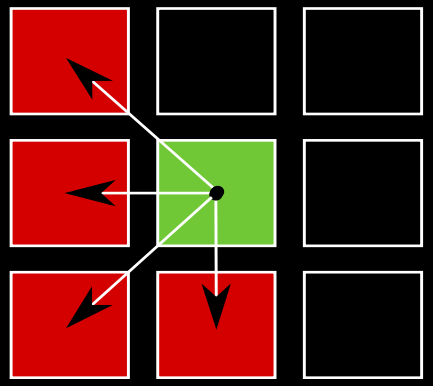
Provide Different Neighborhood Communication Patterns



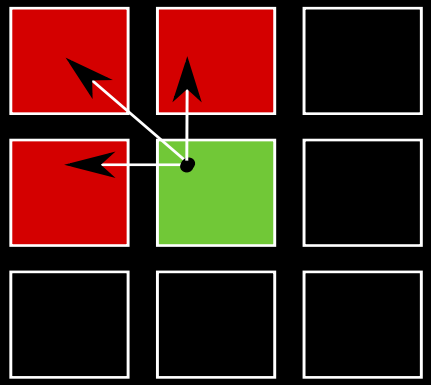
DIY_Enqueue_item_pt()
DIY_Enqueue_item_mask()



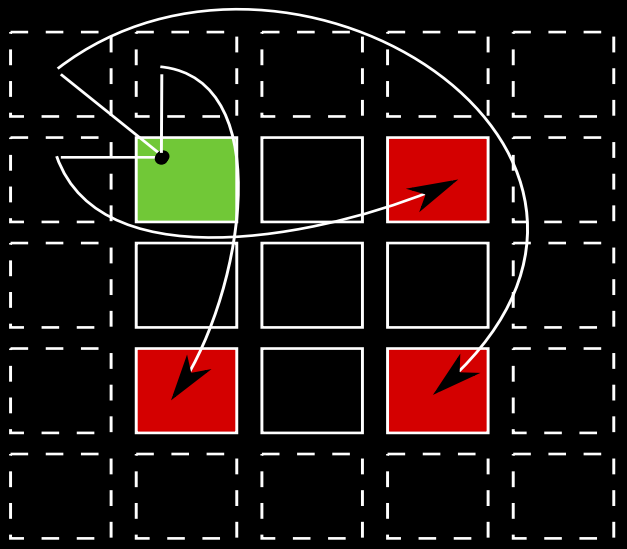
DIY_Enqueue_item_all()



DIY_Enqueue_item_half()



DIY_Enqueue_item_all_near()
DIY_Enqueue_item_half_near()



Support for wraparound neighbors
(repeating boundary conditions)

DIY provides point to point and different varieties of collectives within a neighborhood via its enqueue_item mechanism. Items are enqueued are subsequently exchanged (2 steps).

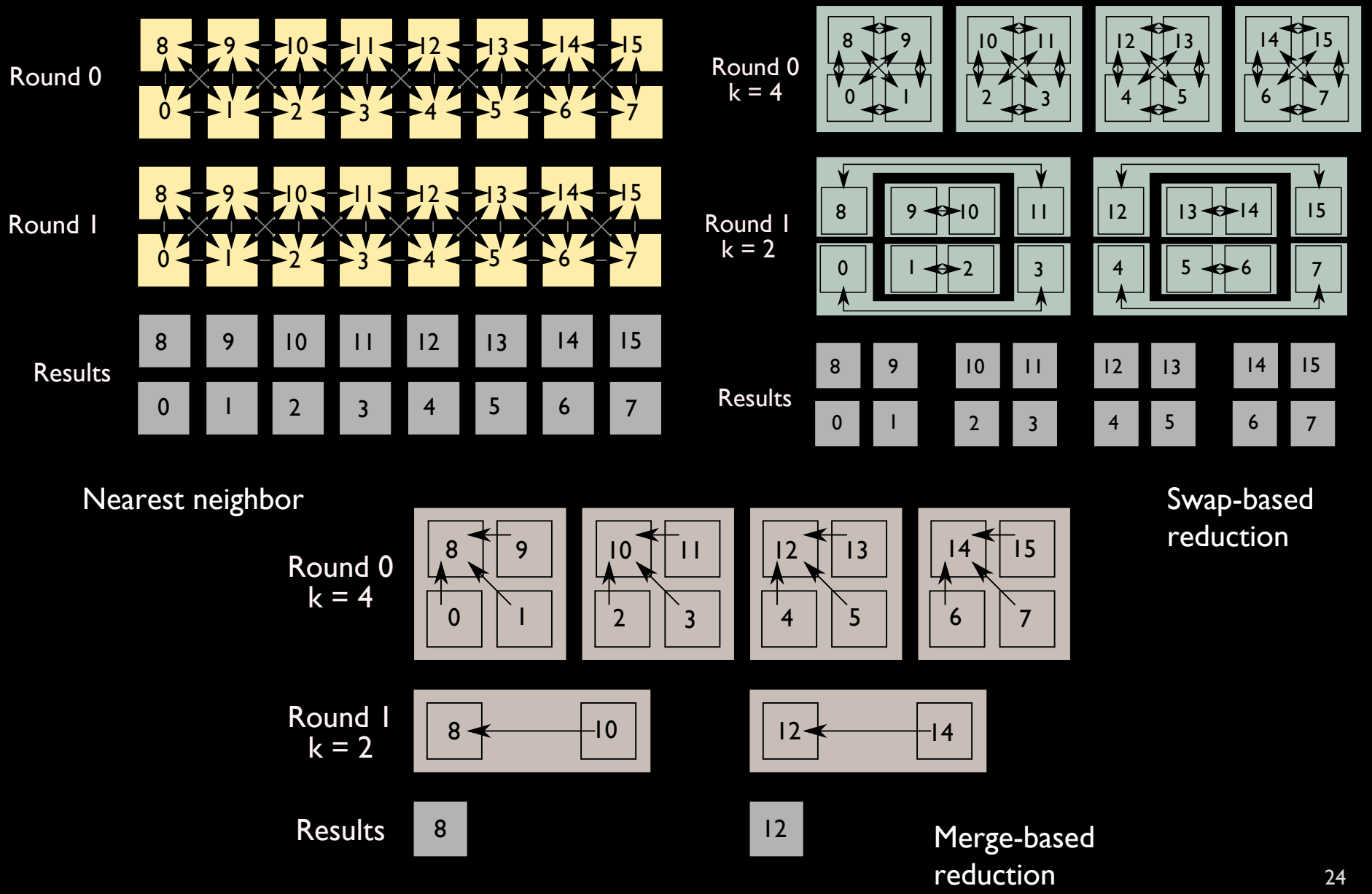
Make Global and Neighborhood Communication Fast and Easy

Analysis	Communication
Particle Tracing	Nearest neighbor
Global Information Entropy	Merge-based reduction
Point-wise Information Entropy	Nearest neighbor
Morse-Smale Complex	Merge-based reduction
Computational Geometry	Nearest neighbor
Region growing	Nearest neighbor
Sort-last rendering	Swap-based reduction

Factors to consider when selecting communication algorithm:
-associativity
-number of iterations
-data size vs. memory size
-homogeneity of data

DIY provides 3 efficient scalable communication algorithms on top of MPI. May be used in any combination.

3 Communication Patterns



Data Input

Multiblock and Multifile I/O

-Application-level two-phase I/O

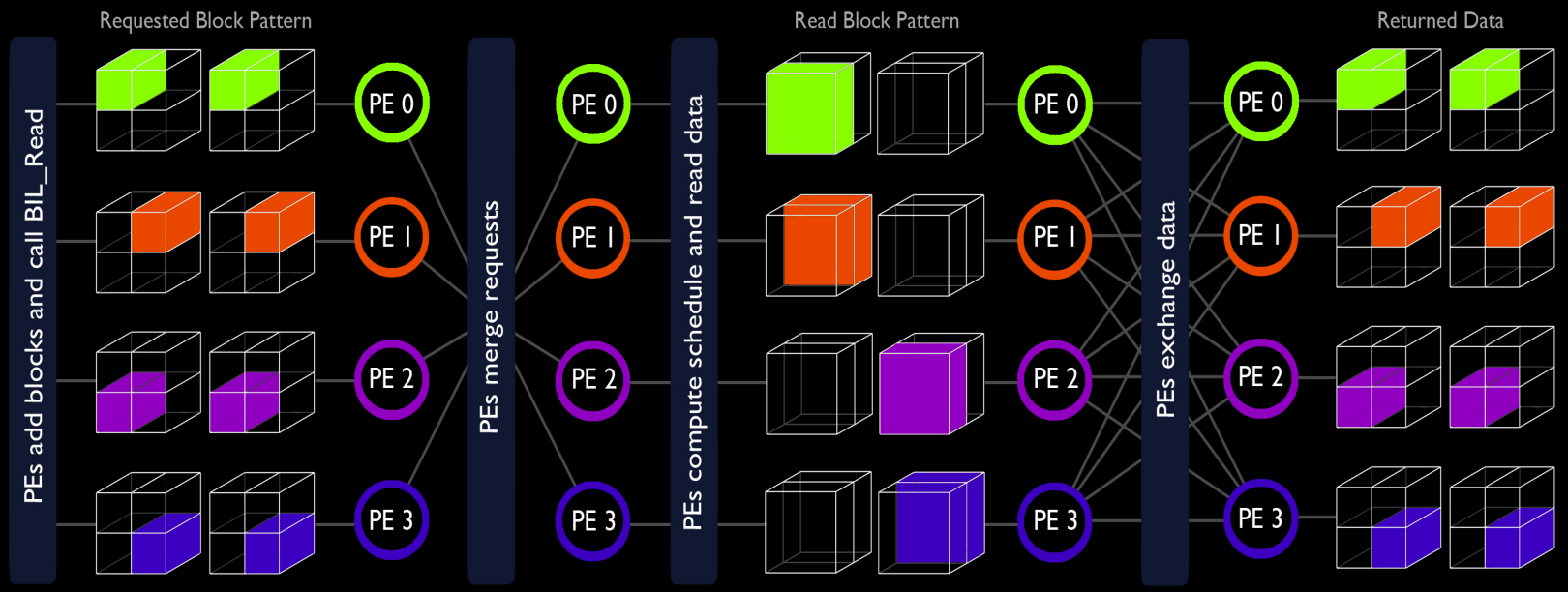
-Reads raw, netCDF, HDF5 (future)

-Read requests sorted and aggregated into large contiguous accesses

-Data redistributed to processes after reading

-Single and multi block/file domains

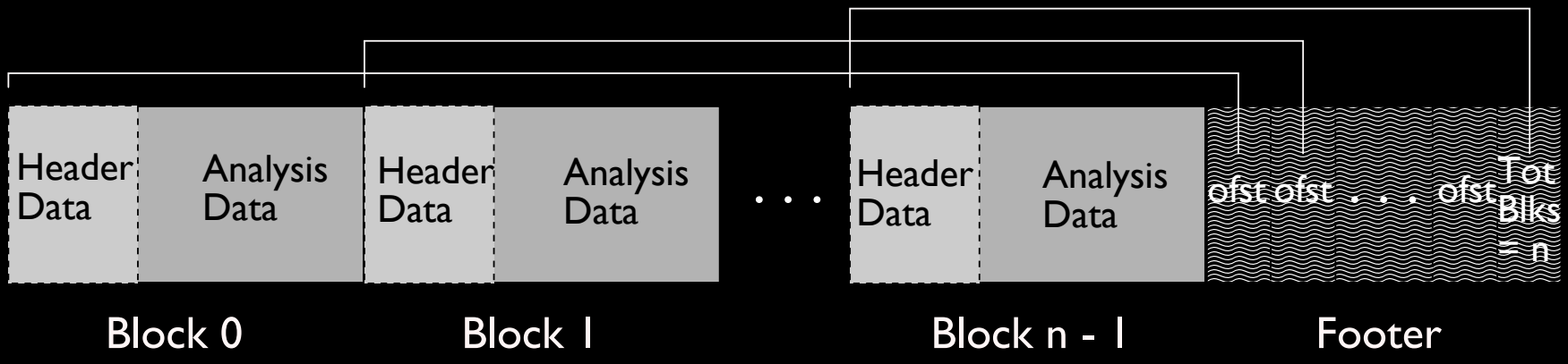
-75% of IOR benchmark on actual scientific data



Input algorithm

Kendall et al., Towards a General I/O Layer for Parallel Visualization Applications, CG&A '11

Analysis Output



Output file format

Features

- Binary
- General header/data blocks
- Footer with indices
- Application assigns semantic value to DIY blocks
- Written efficiently in parallel
- Parallel block-wise compression

Implement Data Operations in a Library with a small ℓ

Features

Parallel I/O to/from storage
- -MPI-IO, BIL

Domain decomposition
- -Decompose domain
- -Describe existing decomposition

Network communication
- -Global reduction (2 flavors)
- -Local nearest neighbor

Library

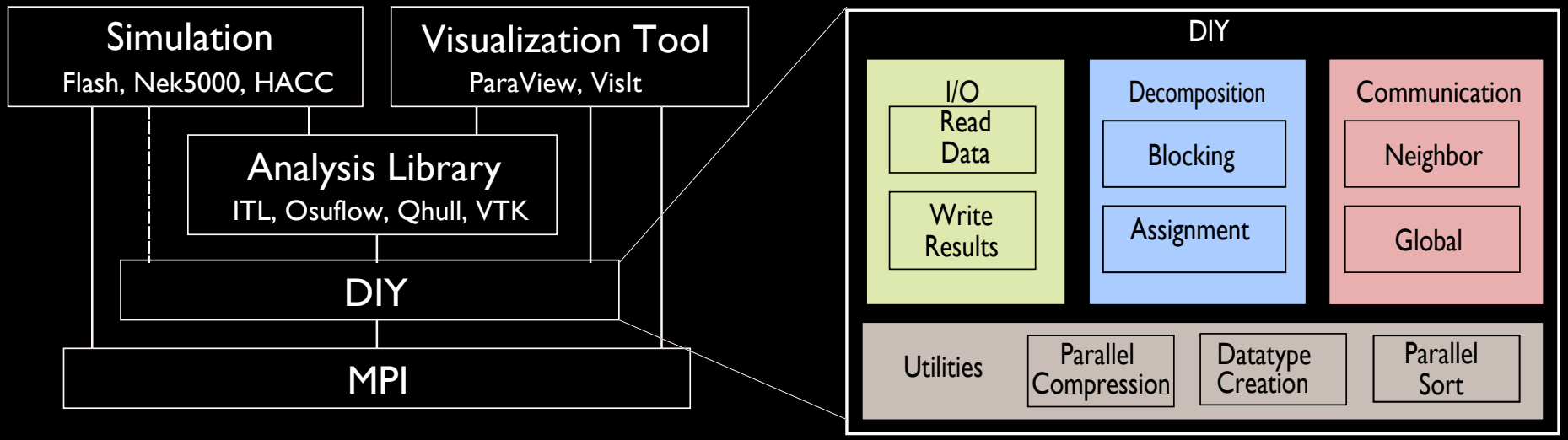
Written in C++

C bindings, future Fortran bindings

Autoconf build system (configure, make, make install)

Lightweight: libdiy.a 800KB

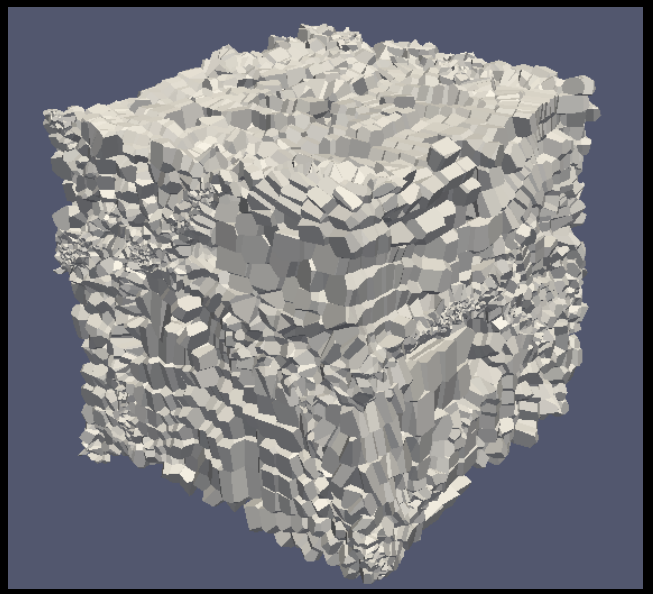
Maintainable: ~15K lines of code, including examples

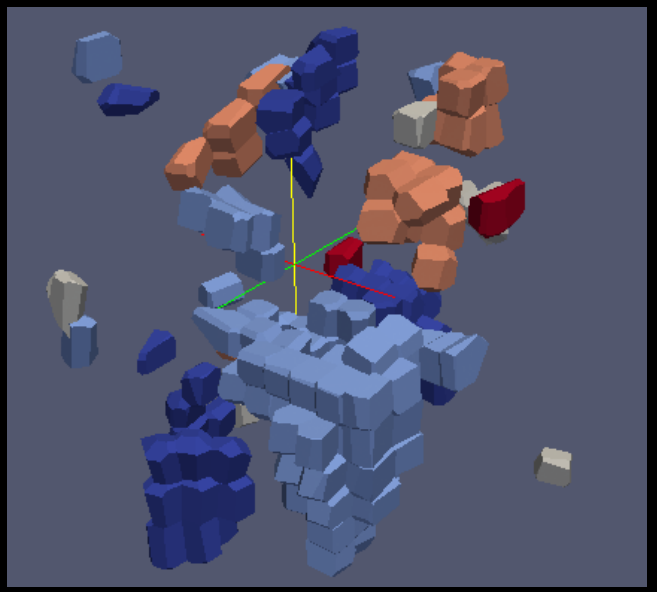


DIY usage and library organization

Work With Postprocessing Tools

Interactive Voronoi Exploration in ParaView





-ParaView reader for Voronoi output from previous slide

-Plugin as part of suite of cosmo tools in ParaView

-Threshold filter

-Connected component labeling

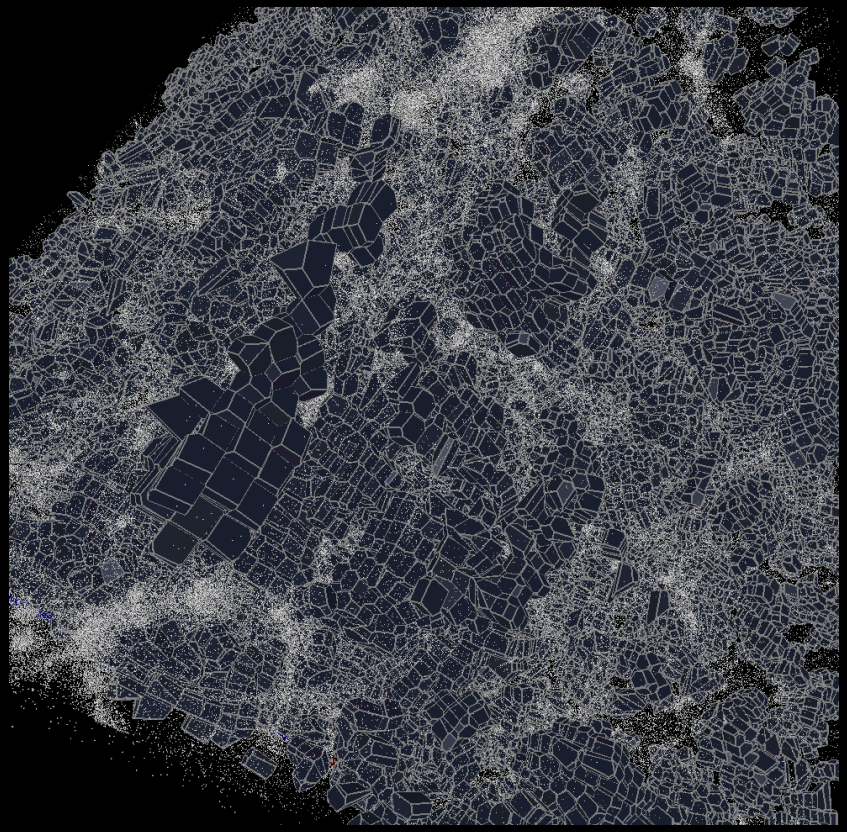
-Analysis of connected components

Run-time Tools Recap

-In situ and coprocessing (terms can be confusing and often used differently)
-We covered in situ analysis with a library called DIY
-Perform some amount of analysis in the simulation and write those results for later postprocessing

Applications of Run-time Analysis

Computational Geometry for Cosmology



-Cosmology N-body simulations produce a sparse particle field

-Can cluster to find halos

-Finding voids more difficult

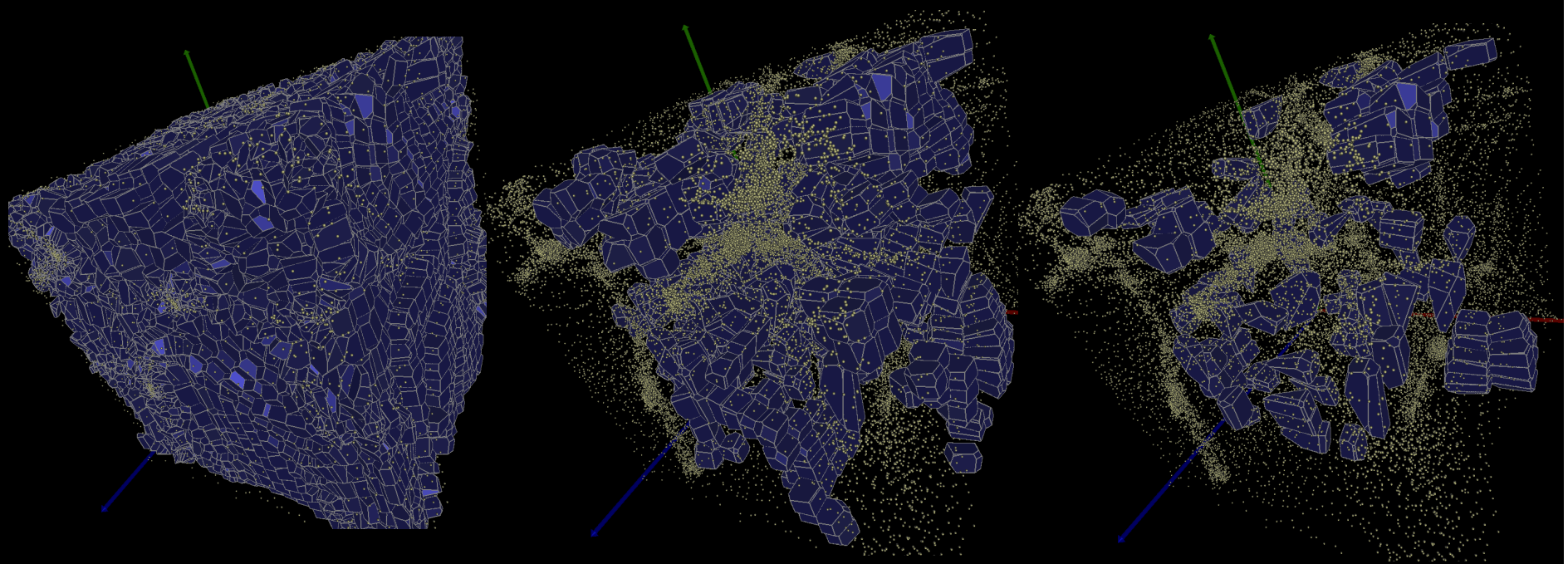
-Dense geometric field is useful (tessellation)

-Voronoi tessellation is ideal because it adapts automatically to particle distribution and assumes little about cell size and shape

-Idea: Generate Voronoi cells from particle data in situ, threshold filter, and store for later postprocessing analysis.

Peterka et al., Meshing the Universe: Identifying Voids in Cosmological Simulations Through In Situ Parallel Voronoi Tessellation, Submitted to LDAV'12

Parallel Voronoi Tessellation



Thresholding cell volume to reveal cosmological voids

Particles	Processes	Total Time (s)	Simulation Time (s)	Tessellation Time (s)
512^3	2048	3852	3684	167
	4192	2008	1918	89
	8096	1784	1722	62
	16384	1406	1344	61

Subset of strong and weak scaling test results shows good scalability and relatively small fraction of total run time for in situ analysis

Parallel Time-Varying Flow Analysis

Collaboration with the Ohio State University and University of Tennessee Knoxville

Approach

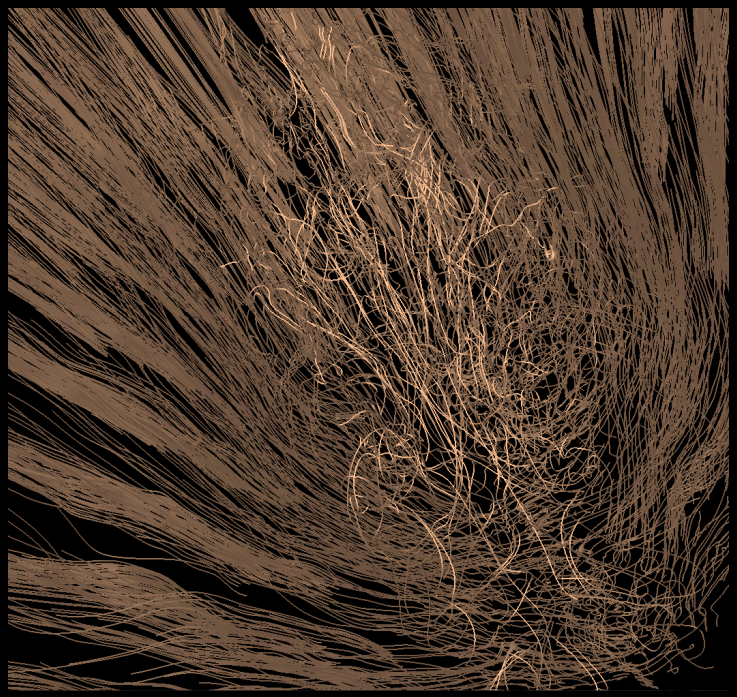
-In core / out of core processing of time steps

-Simple load balancing (multiblock assignment, early particle termination)

-Adjustable synchronization communication

Algorithm

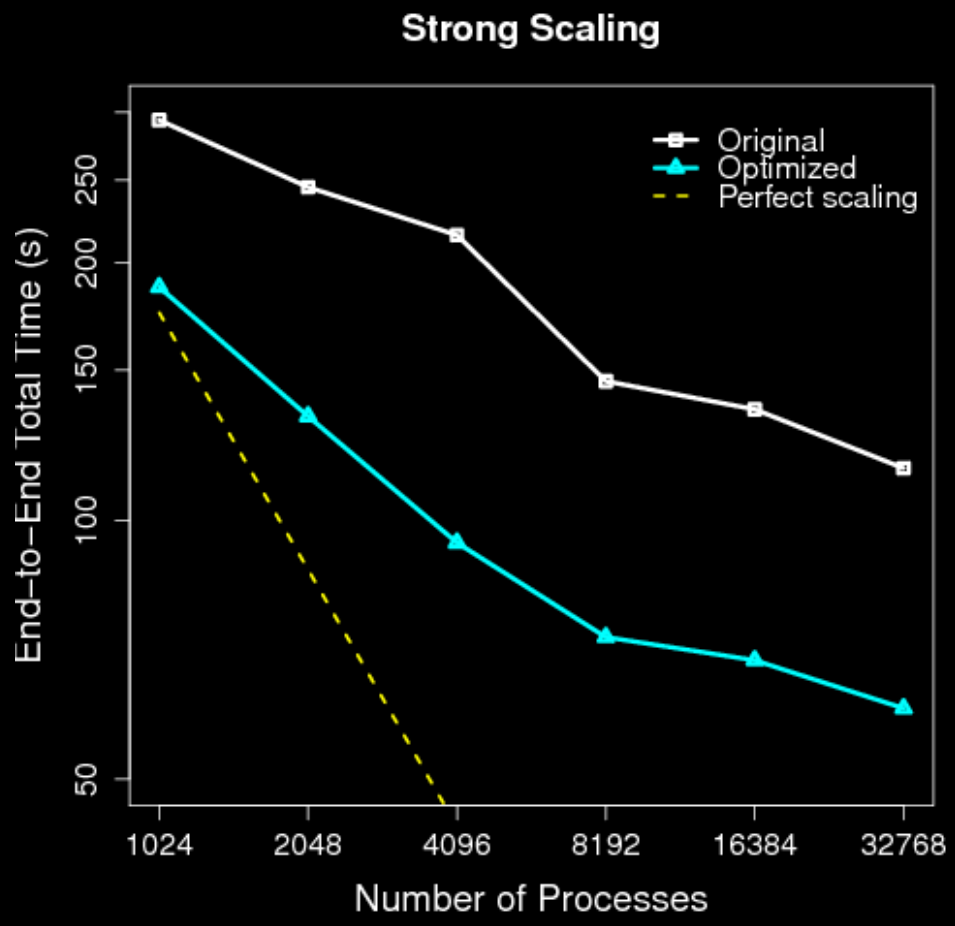
```
for (epochs) {
  read my process' data blocks
  for (rounds) {
    for (my blocks) {
      advect particles
    }
    exchange particles
  }
}
```

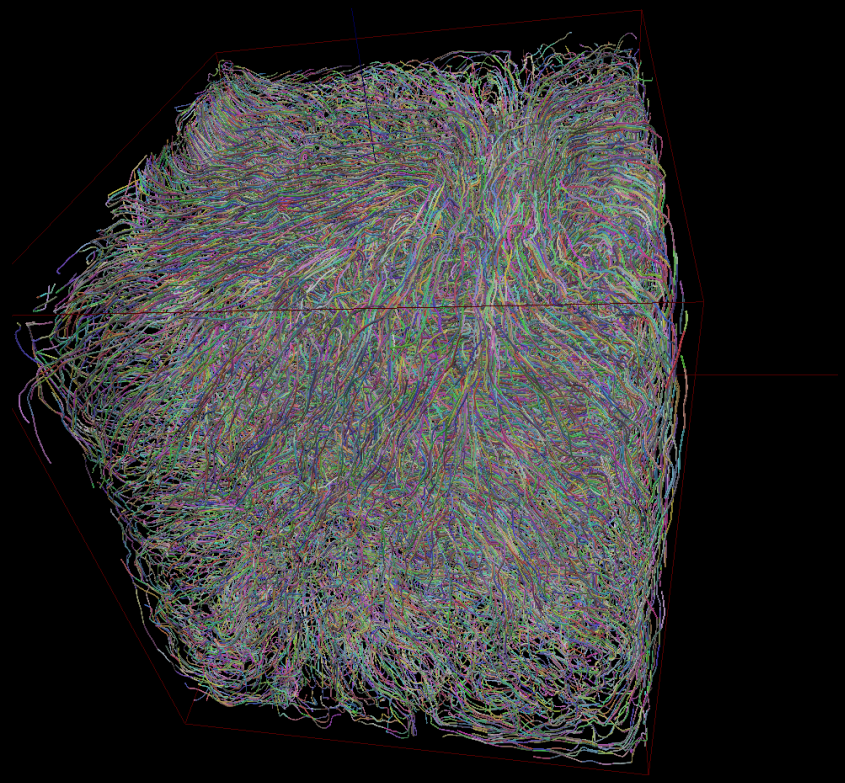


Pathline tracing of 32 time-steps of combustion in the presence of a cross-flow

Peterka et al., A Study of Parallel Particle Tracing for Steady-State and Time-Varying Flow Fields, IPDPS '11

Parallel Particle Tracing





Particle tracing of ¼ million particles in a 2048^3 thermal hydraulics dataset results in strong scaling to 32K processes and an overall improvement of 2X over earlier algorithms

Parallel Information-Theoretic Analysis

Collaboration with the Ohio State University and New York University Polytechnic Institute

Objective

-Decide what data are the most essential for analysis

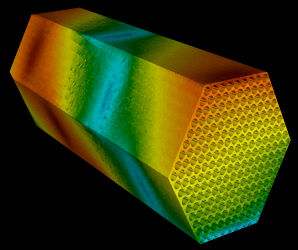
-Minimize the information losses and maximize the quality of analysis

-Steer the analysis of data based on information saliency

Information-theoretic approach

-Quantify Information content based on Shannon's entropy

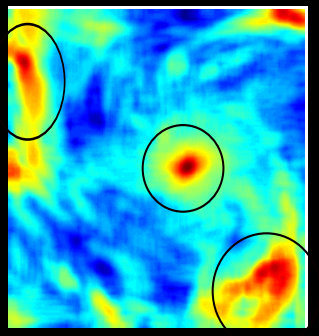
-Use this model to design new analysis data structures and algorithms



Nek5000
CFD model

Information-
theoretic
algorithms

Areas of high information entropy--turbulent regions in original data--are the interesting regions in simulating coolant flow in a nuclear reactor.



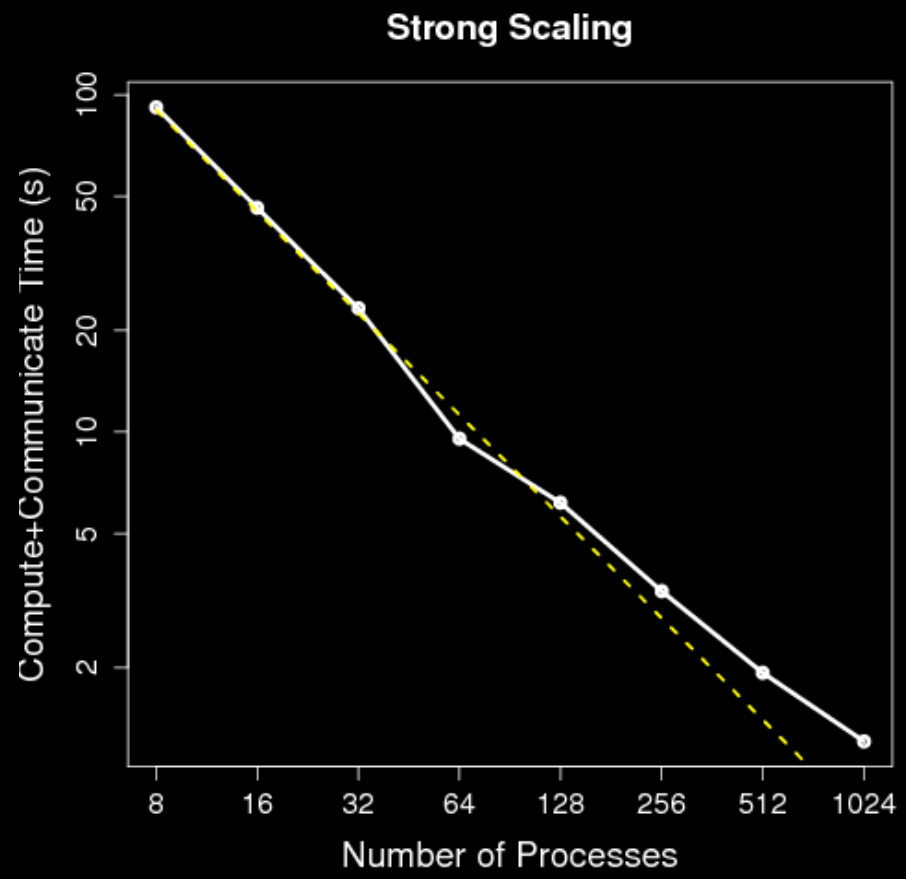
Section of information entropy field

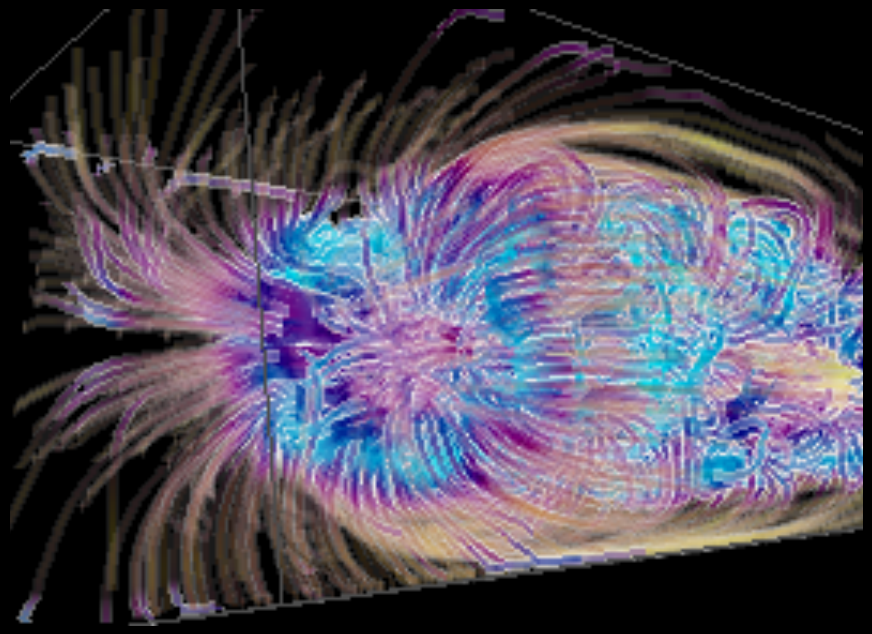
Shannon's Entropy

The average amount of information expressed by the random variable is

$$H(x) = -\sum_{i=1}^{n} p_i \log p_i$$

Information Entropy Performance and Scalability



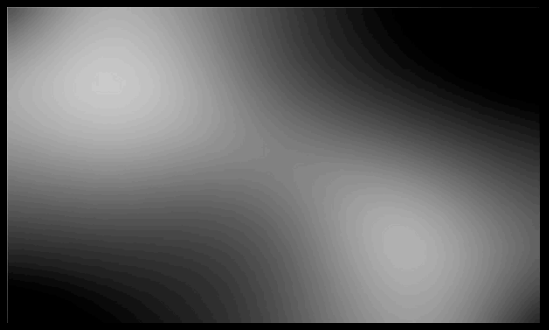


Computation of information entropy in 126x126x512 solar plume dataset shows 59% strong scaling efficiency.

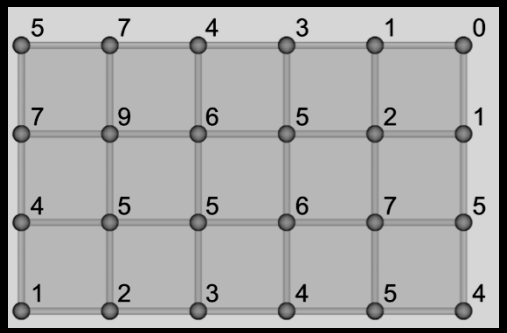
Parallel Topological Analysis

Collaboration with SCI Institute, University of Utah

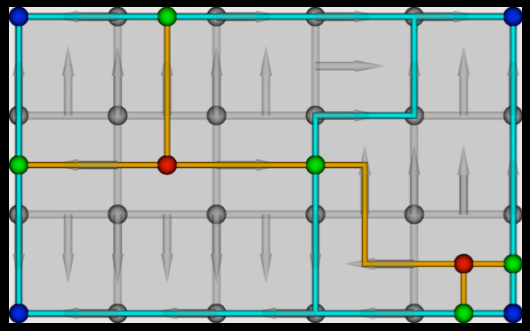
-Transform discrete scalar field into Morse-Smale complex
-Nodes are minima, maxima, saddle points of scalar values
-Arcs represent constant-sign gradient flow
-Used to quickly see topological structure



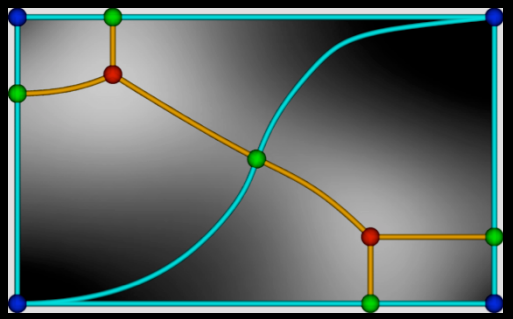
1



2

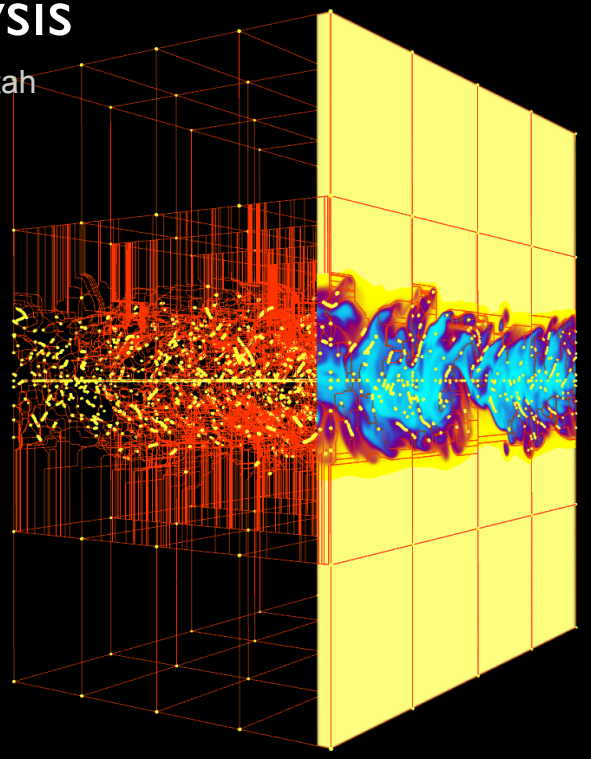


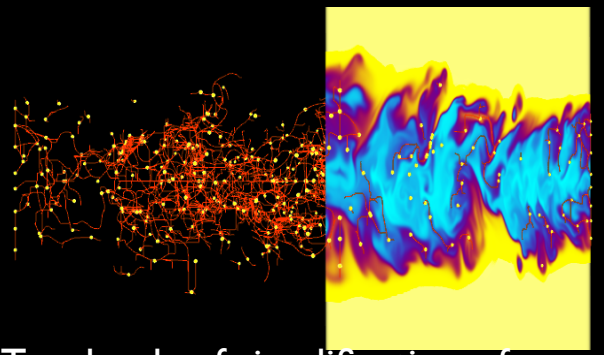
3



4

Example of computing discrete gradient and Morse-Smale Complex

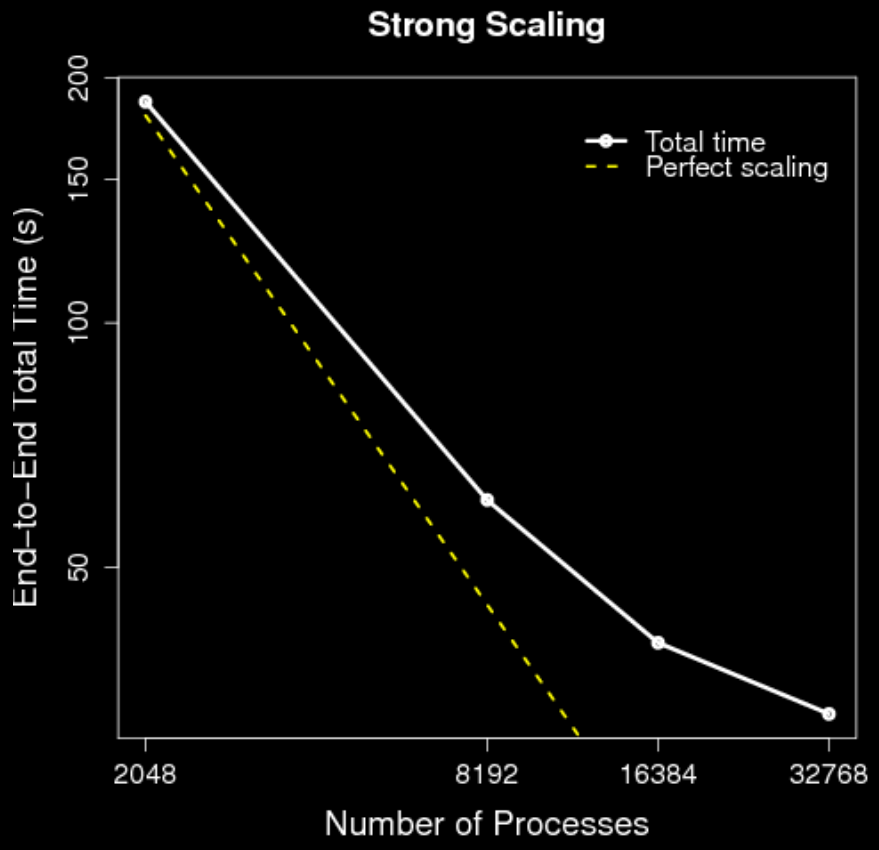


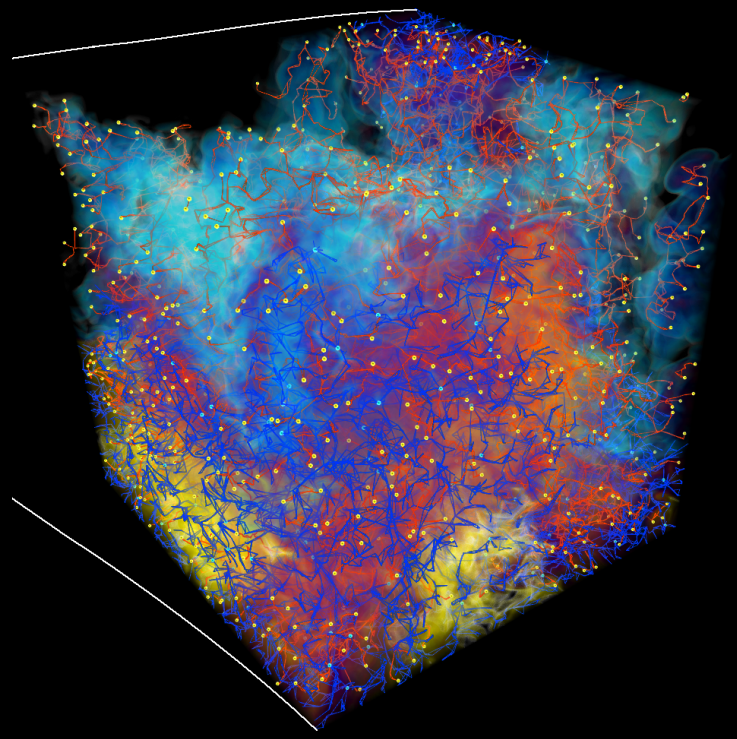


Two levels of simplification of the Morse-Smale complex for jet mixture fraction.

Gyulassy et al., The Parallel Computation of Morse-Smale Complexes, Submitted to IPDPS '12

Morse-Smale Complex Performance and Scalability





Computation of Morse-Smale complex in 1152^3 Rayleigh-Taylor instability data set results in 35% end-to-end strong scaling efficiency, including I/O.

Summary

-Consider data and data movement as first-class citizens

-Tools needed both for run-time as well as postprocessing analysis

-Analysis is any sequence of operations on data that hopefully reduces its size and/or improves its understandability

-Much more work to be done!

U.S. DEPARTMENT OF ENERGY

Argonne NATIONAL LABORATORY

"The purpose of computing is insight, not numbers."

–Richard Hamming, 1962

Acknowledgments:

Facilities

Argonne Leadership Computing Facility (ALCF)

Oak Ridge National Center for Computational Sciences (NCCS)

Funding

DOE SDMAV Exascale Initiative

DOE Exascale Codesign Center

http://www.mcs.anl.gov/~tpeterka/software.html

https://svn.mcs.anl.gov/repos/diy/trunk

Tom Peterka

tpeterka@mcs.anl.gov

Mathematics and Computer Science Division

CScADS Summer Workshop 7/26/12