Tom Peterka, Juliana Kwan, Adrian Pope,
Hal Finkel, Katrin Heitmann,
Salman Habib, Jingyuan Wang,
George Zagaris, Sergei Shandarin

U.S. DEPARTMENT OF ENERGY

Argonne NATIONAL LABORATORY

SDAV
Scalable Data Management, Analysis, and Visualization

Meshing the Universe:
Integrating Analysis in Cosmological Simulations

"Every particle, every field of force, even the space-time continuum itself—derives it function, its meaning, its very existence…from bits."

–physicist John Archibald Wheeler, 1989

Tom Peterka
tpeterka@mcs.anl.gov
Mathematics and Computer Science Division

SC Ultrascale
Visualization Workshop
Nov. 12, 2012

Executive Summary

We developed a prototype library for computing in situ tessellations from particle data and show its application in cosmology simulations.

Key Ideas

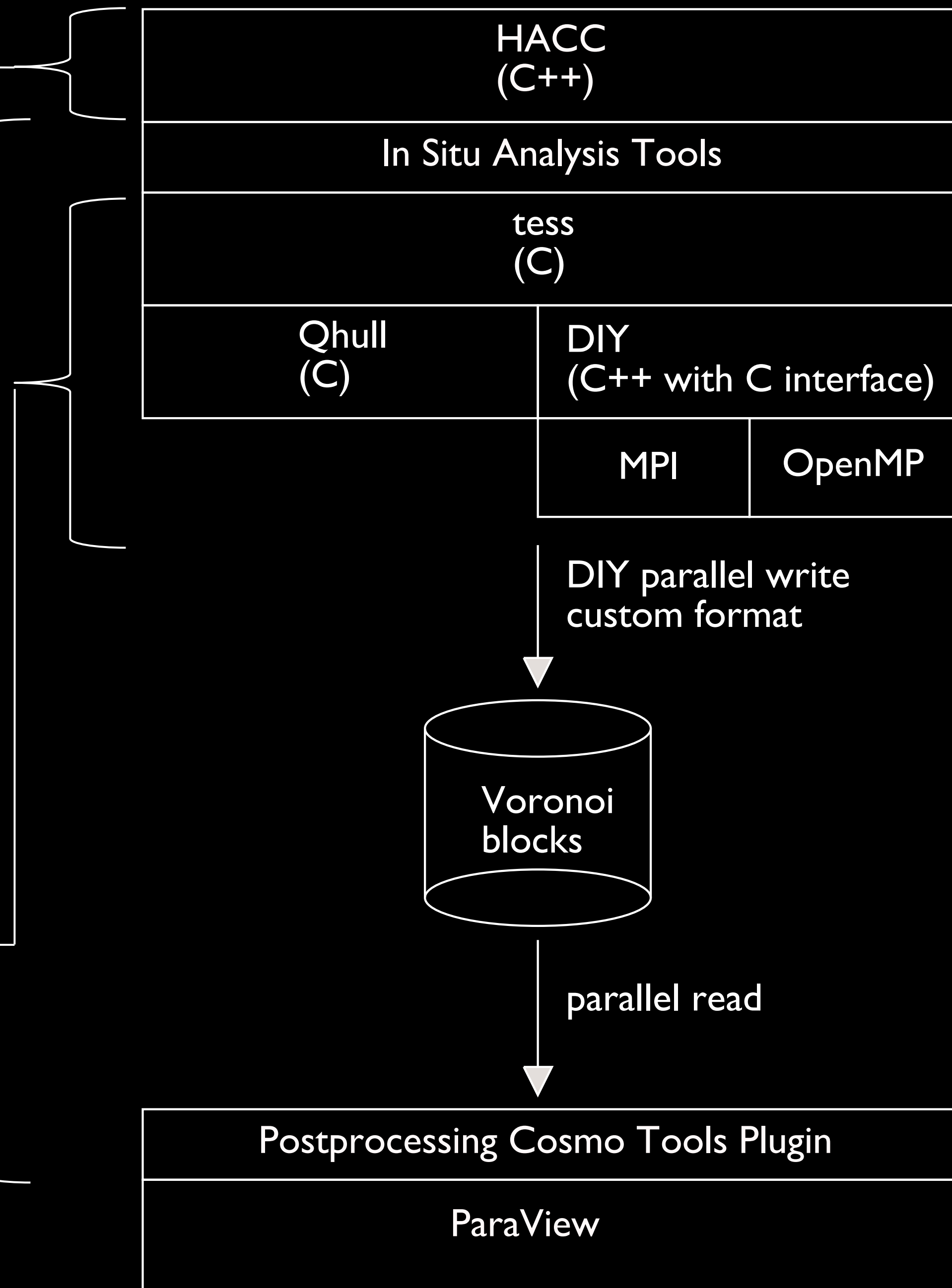
- Mesh tessellations convert sparse point data into continuous dense field data.
- Meshing output of simulations is data-intensive and requires supercomputing resources
- No large-scale data-parallel tessellation tools exist.
- We developed such a library, tess.
- Tess is part of a larger framework of in situ, coprocessing, and postprocessing cosmology tools.
- We demonstrate good parallel performance and scalability while running in situ with a cosmology code, HACC.
- Statistical analyses of cell distributions agree with underlying physics.

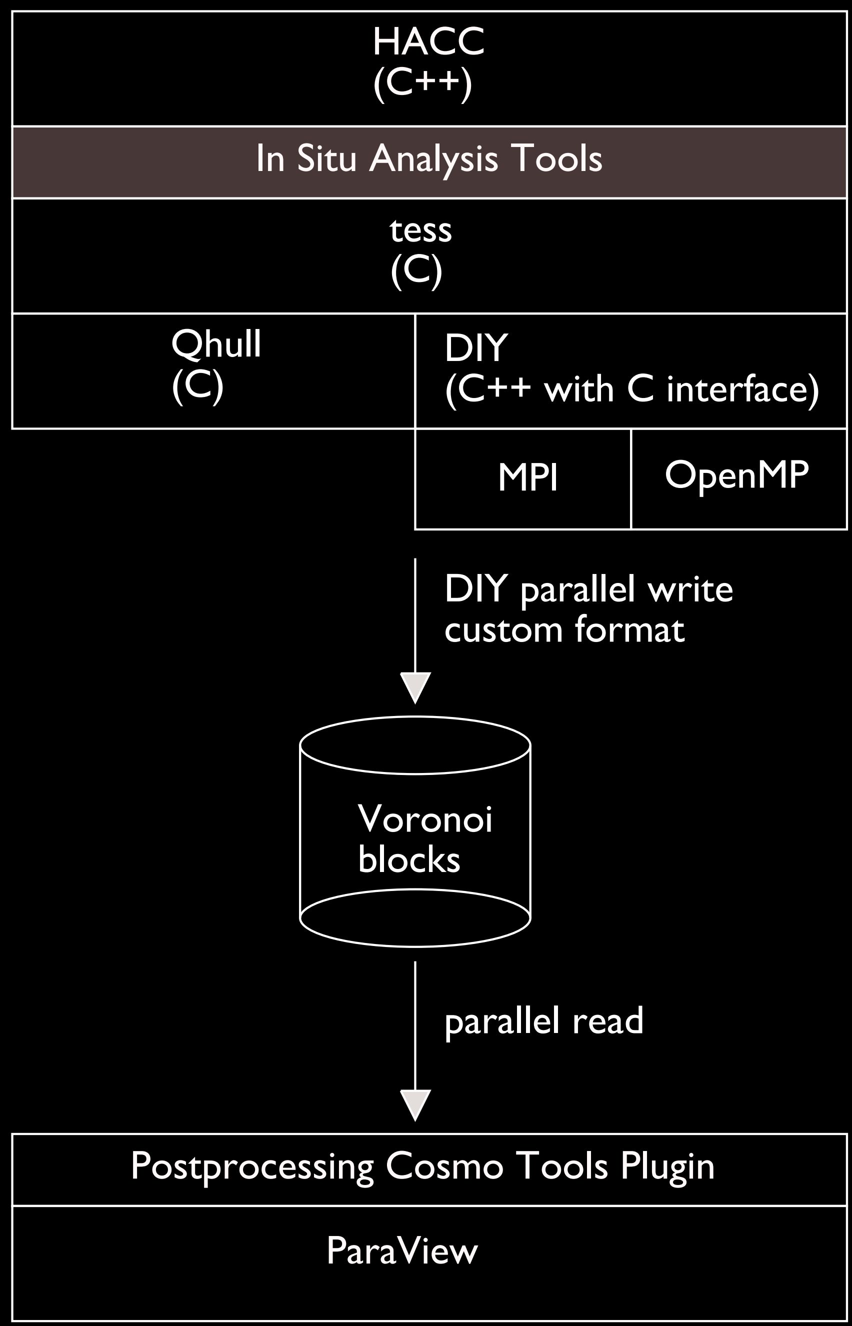
Overview

HACC is a petascale computational cosmology code and a 2012 Gordon Bell finalist.

This talk will summarize our work in the end-to-end analysis pipeline.

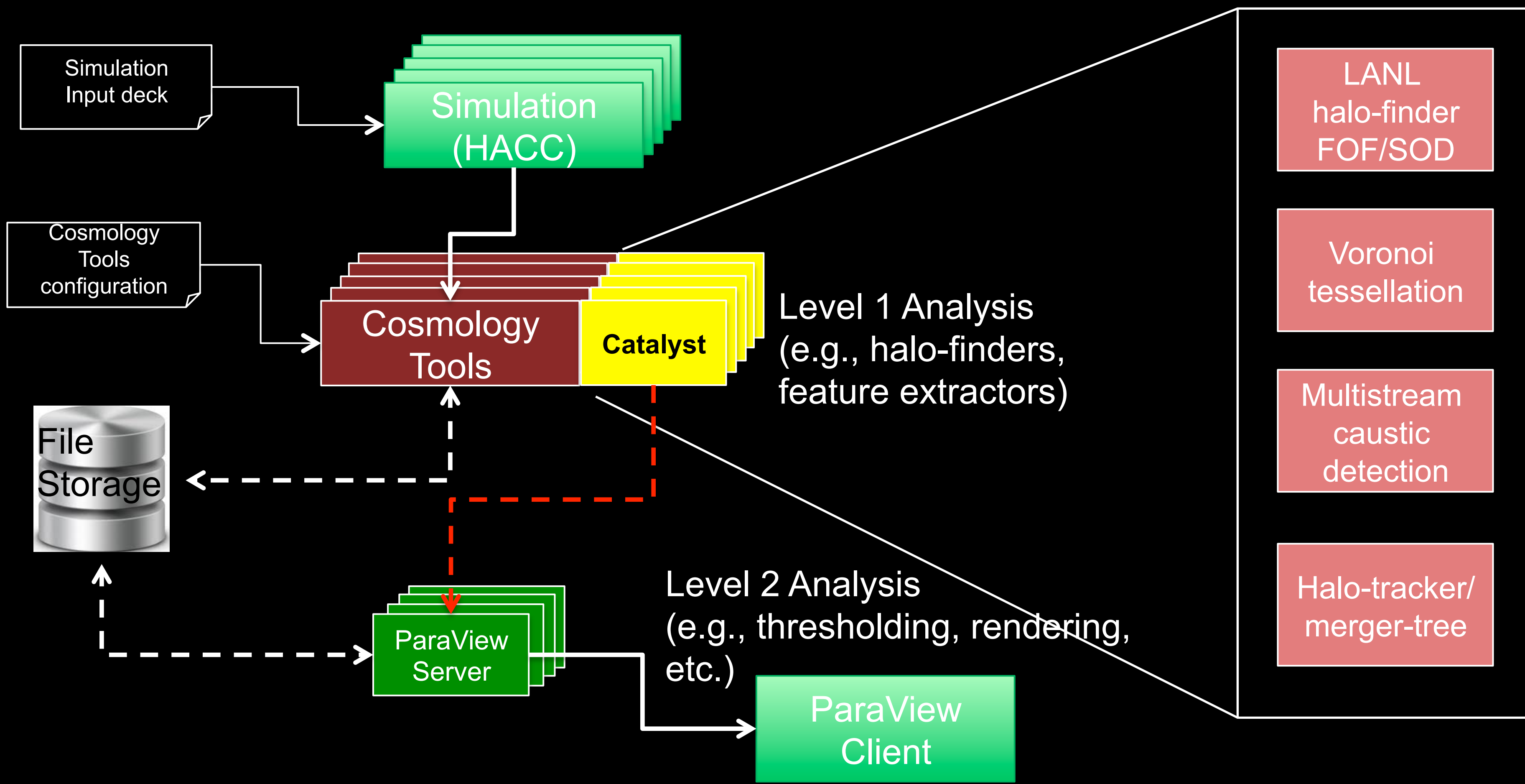
The focus will be on in situ computation of the Voronoi tessellation (tess).




HACC
(C++)
In Situ Analysis Tools
tess
(C)
Qhull
(C)
DIY
(C++ with C interface)
MPI
OpenMP
DIY parallel write
custom format
Voronoi
blocks
parallel read
Postprocessing Cosmo Tools Plugin
ParaView

In Situ Analysis Framework

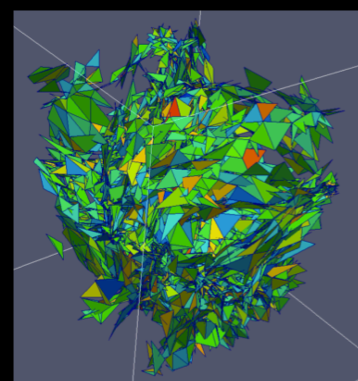
Collaboration with Kitware Inc.

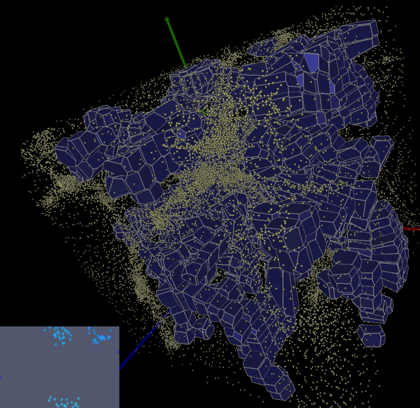


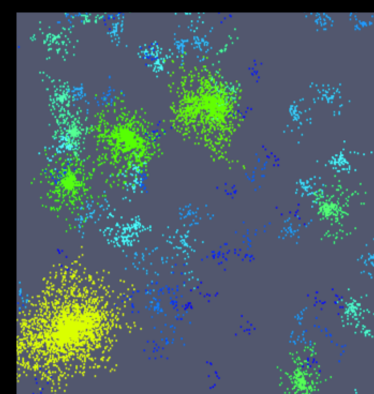
The cosmology tools framework provides the ability to apply various in situ analyses at selected timesteps in the simulation. It connects to ParaView via either Catalyst for run-time visualization or to a parallel file system for postprocessing. Existing analysis tools include halo finders, multistream feature classification, and a void finder based on Voronoi tessellation.

Calling Analysis Tools In Situ

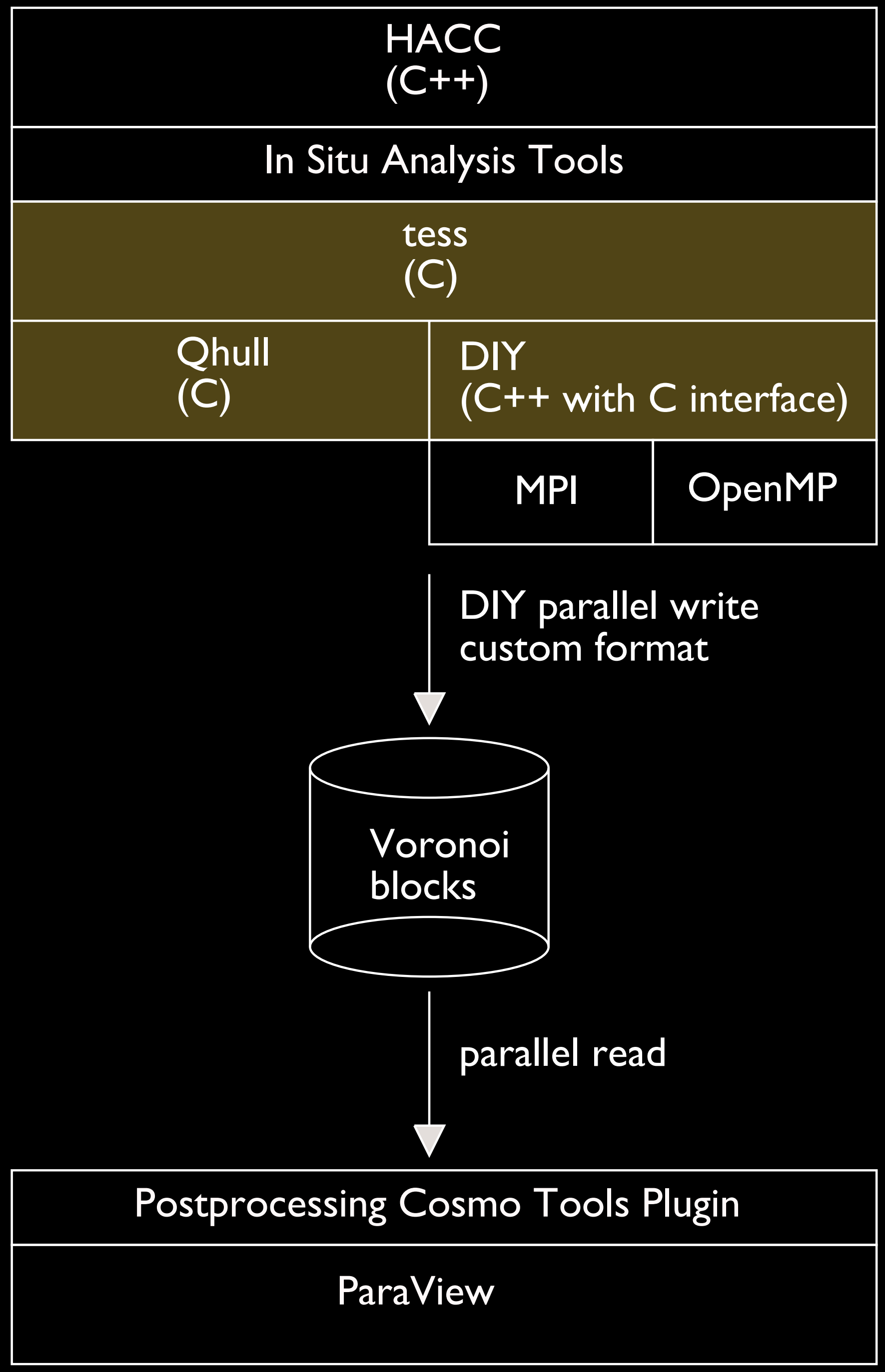
```
…
Initialize(comm);
SetAnalysisConfiguration(“cosmo-
config.dat”);
SetDomainParameters(rL,NG,NDIM)
…

for( int t=0; t < NSTEPS; ++t)
{
     …
   // Set particles at the given timestep/
redshift
   SetParticles(
          t, redshift,
               xx, yy, zz, vx, vy, vz,
          mass, potential,
          GlobalParticleIds,
          mask,
          state,
          N);
    CoProcess();
} // END for all timesteps
    caption
…

Finalize()
```







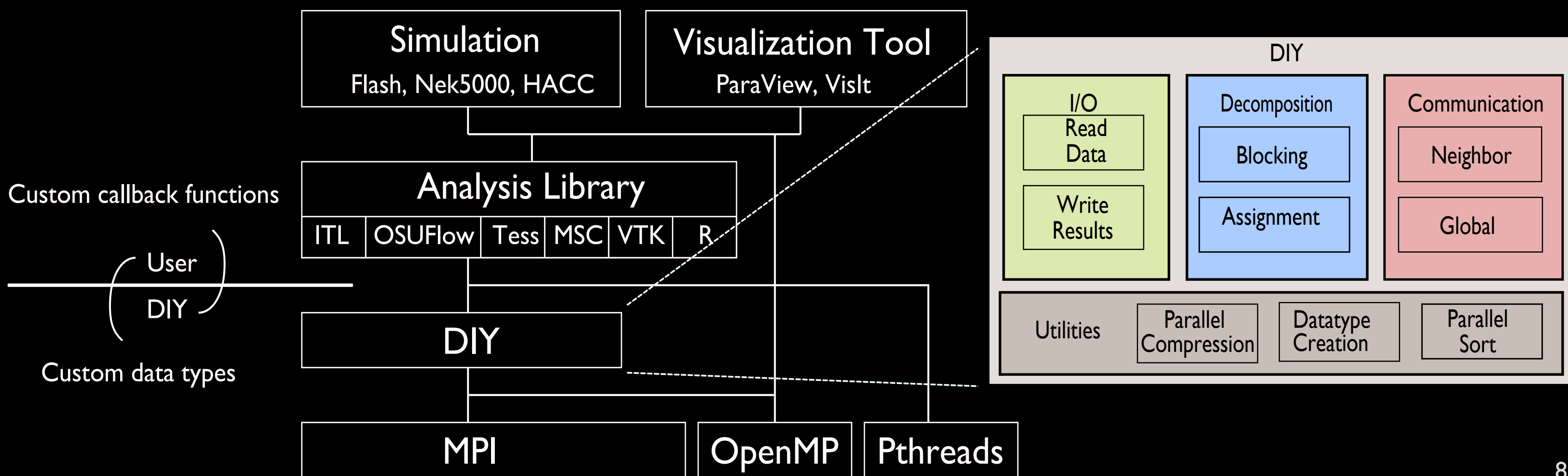
Tessellation is the first member of an in situ analysis framework that will include merger trees, multistream classification, and halo finding.


HACC
(C++)
In Situ Analysis Tools
tess
(C)
Qhull
(C)
DIY
(C++ with C interface)
MPI
OpenMP
DIY parallel write
custom format
Voronoi
blocks
parallel read
Postprocessing Cosmo Tools Plugin
ParaView

A Brief DIY Digression

- Separates user space and data space
- Lets users do what they do best
  - Custom serial analysis
- DIY does what it does best
  - Scalable data movement algorithms that run on Unix/Linux platforms, from laptop to supercomputer (including all IBM and Cray HPC leadership machines)
- Enables parallelization of new and existing libraries
- Lightweight, scalable, built for supercomputing applications

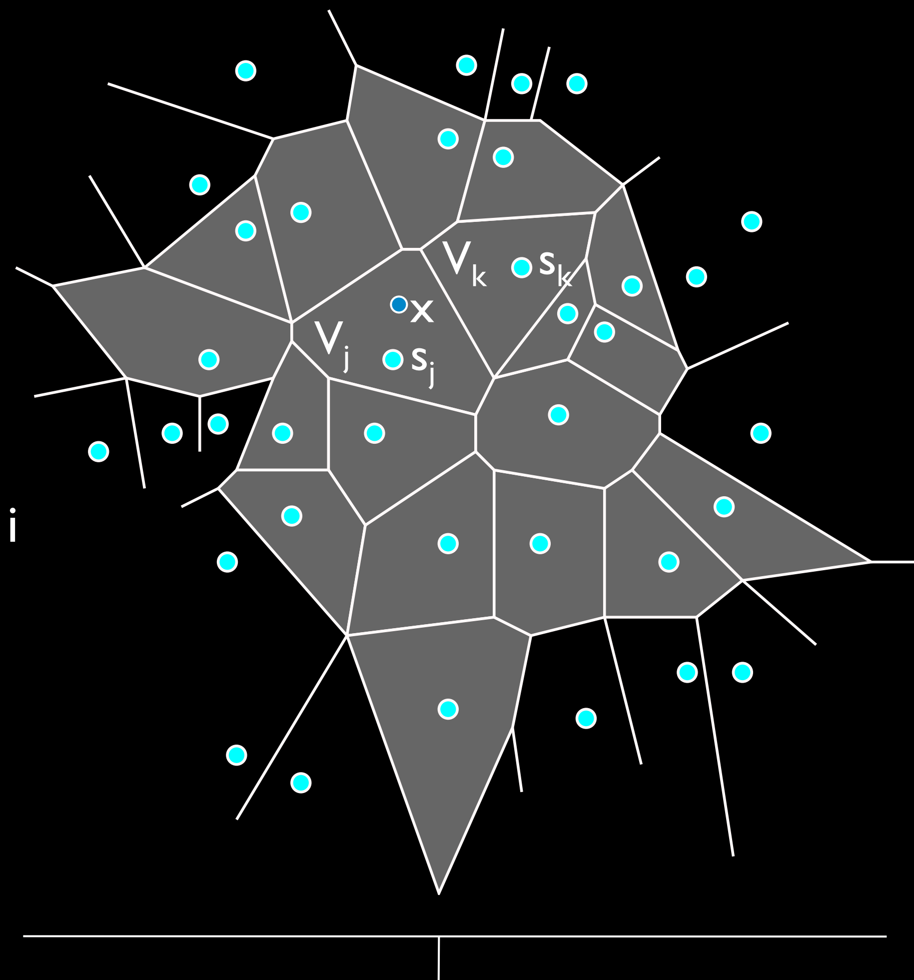
Peterka et al. Scalable Parallel Building Blocks for Custom Data Analysis, LDAV'11



Voronoi Tessellation Background

- Each Voronoi cell is associated with one input particle, the *site* of the cell
- A cell consists of the volume of all points closer to the site of that cell than to any other site
- Formally, each Voronoi cell

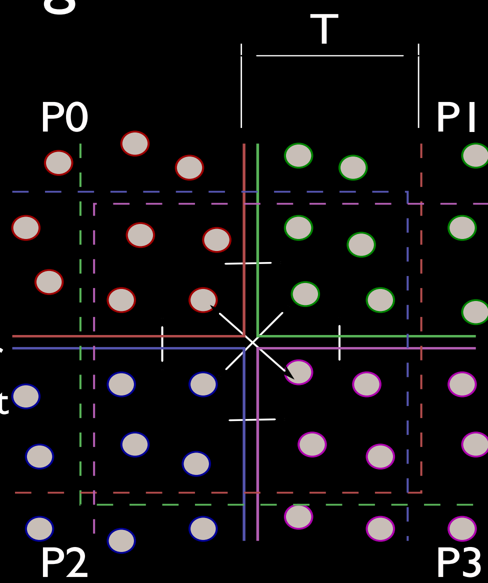
  $V_i = \{ \mathbf{x} \mid d(\mathbf{x}, \mathbf{s_i}) < d(\mathbf{x}, \mathbf{s}_k) \} \ \forall \ k \neq i$
- In 2D, Voronoi diagram of polygons, in 3D tessellation of polyhedra (used by Springel et al., Shandarin et al.)
- Dual is Delaunay triangulation (2D), tetrahedralization (3D) (used by Schaap et al.)



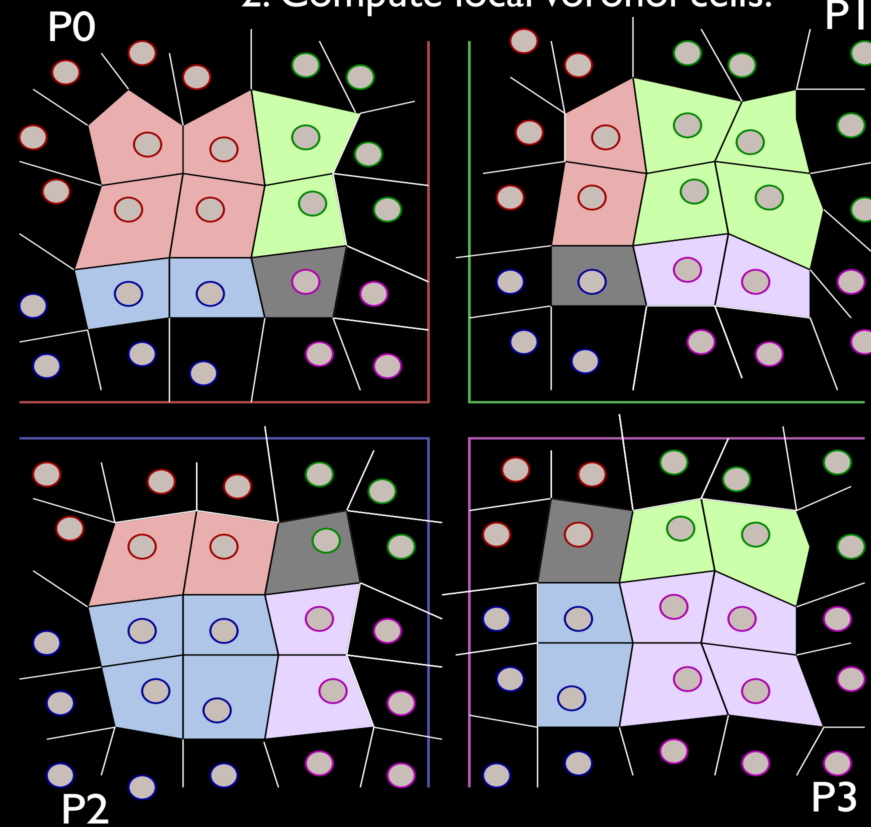
Example 2D Voronoi diagram

Parallel Algorithm

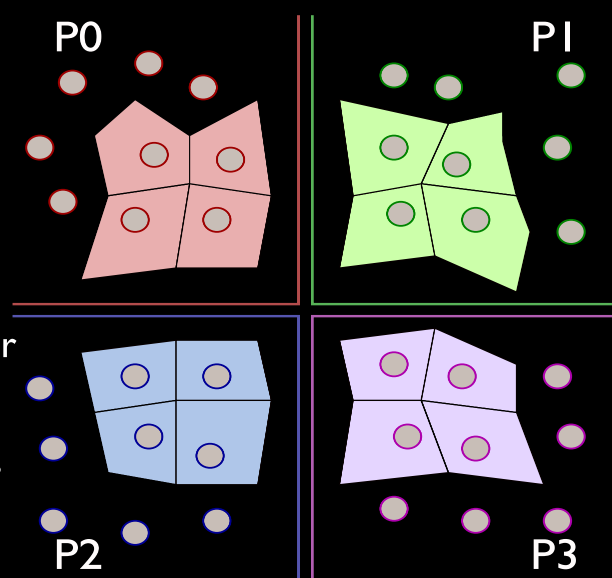
1. Exchange neighbor particles within ghost thickness T >= 2 * cell size.

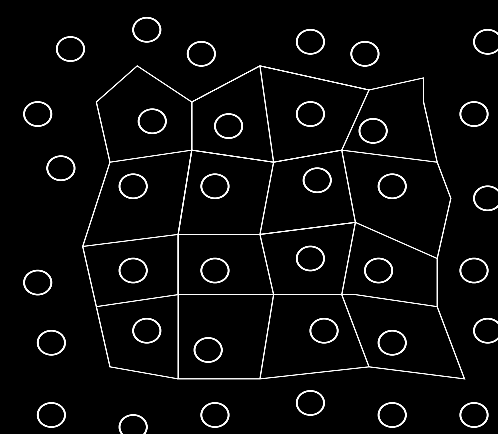


2. Compute local Voronoi cells.



3. Keep only cells sited at local particles. Delete incomplete cells and cells safely below volume threshold. Reorder cell vertices and compute volume. Delete other cells outside volume range.



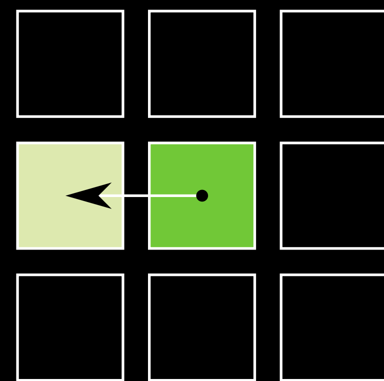


4. Write local sites and Voronoi cells to storage.

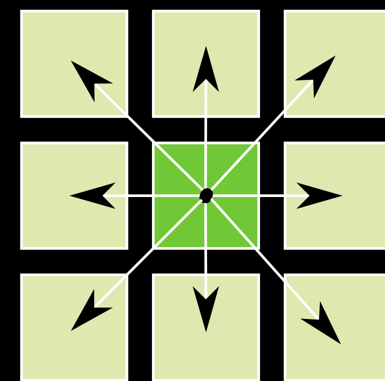
Neighbor Exchange Details

How to enqueue items for neighbor exchange

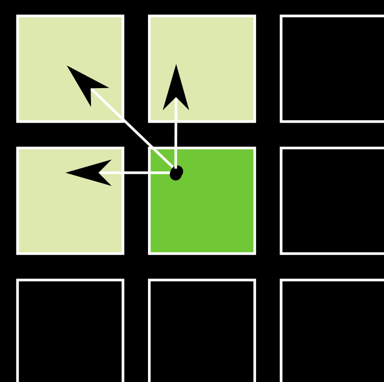
- DIY offers several options
- The patterns in the lower row were added as part of this research
- Support for periodic boundary conditions involves tagging which neighbors are periodic and calling user-defined transform on objects being sent to them



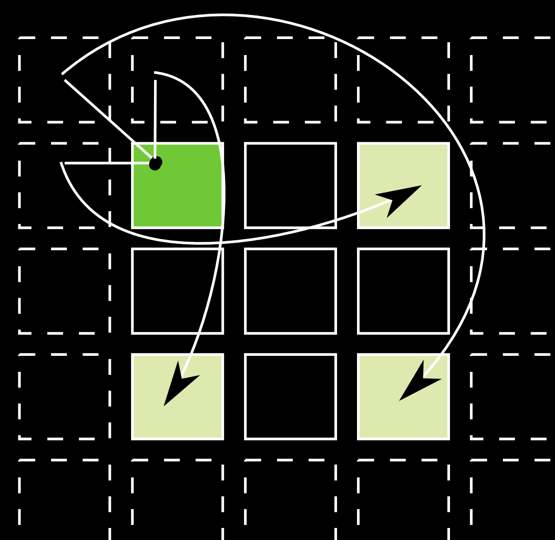
Send to only specific neighbors, indicated in various ways



Send to all neighbors



Send to all neighbors near enough to a target point



Support for wraparound neighbors (periodic boundary conditions)

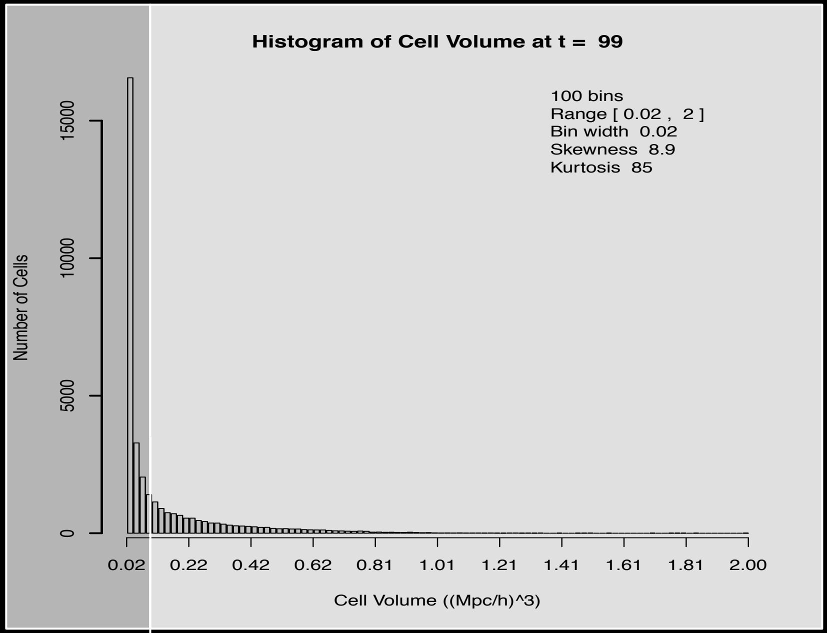
Parallel Accuracy

64^3 Particles				
Ghost size	# Cells in serial version	# Blocks	# Identical Cells	Accuracy
0	210181	2	201952	96.08%
		4	196803	93.64%
		8	192140	91.42%
1		2	209367	99.61%
		4	208564	99.23%
		8	207024	98.50%
2		2	210176	100.00%
		4	210155	99.99%
		8	210012	99.92%
3		2	210181	100.00%
		4	210180	100.00%
		8	210180	100.00%
4		2	210181	100.00%
		4	210181	100.00%
		8	210181	100.00%

Accuracy of parallel version with varying number of blocks is compared to the serial version with all particles in one block. Accuracy improves with increasing ghost size until the ghost region is sufficient.

Data Size

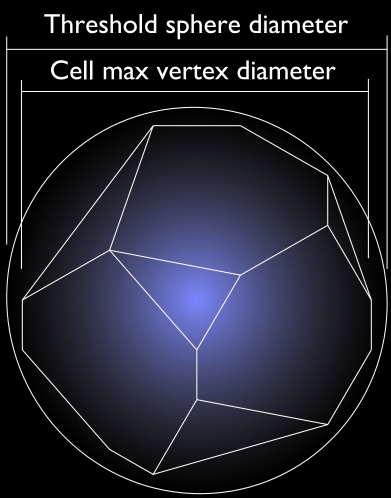
```
struct vblock_t {
   float mins[3], maxs[3];
   float *vertices;
   float *sites;
   float *areas, *vols;
   int *cells;
   int *num_cell_faces;
   int *num_face_verts;
   int *face_verts;
}
```



Classic 80/20 rule: 75% of cells are in the bottom 10% of volume. Thresholding in situ reduces storage size to ~100 bytes per particle.

Average cell statistics

- 15 faces per cell
- 5 vertices per face
- 35 vertices per cell
- 450 bytes per particle
- 7% floating point geometry
- 93% connectivity
- Simulation stores between 40-64 bytes per particle



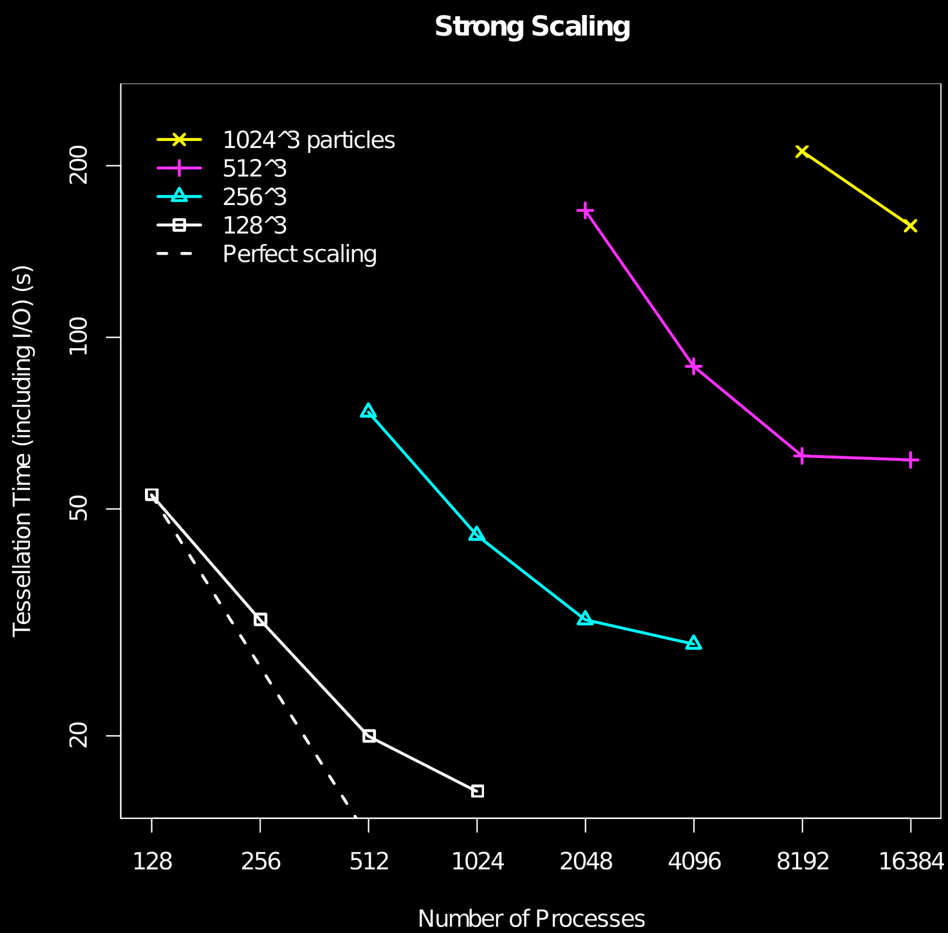
Cells are culled early by comparing maximum vertex diameter with diameter of sphere of threshold volume.

Performance

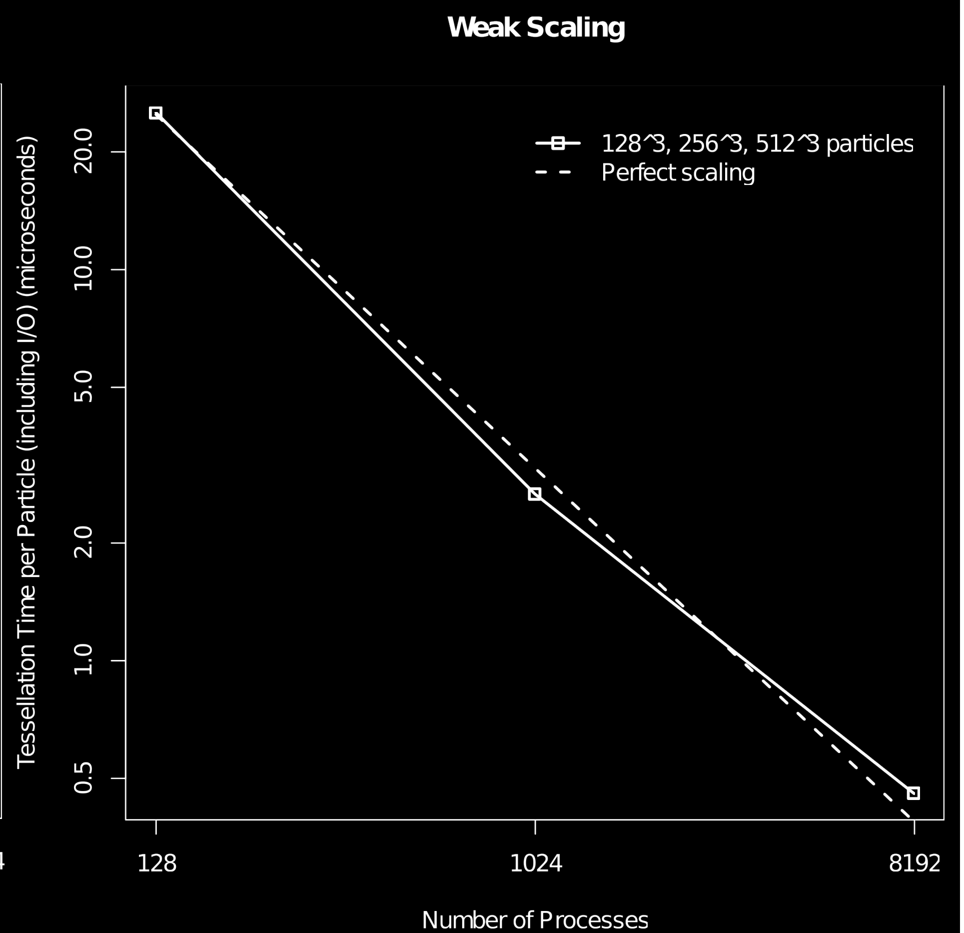
Particles	Time steps	Processes	Tot Time (s)	Sim Time (s)	Tess Tot Time (s)	Exchange Time	Tess Time (s)	Output Time (s)	Output Size(GB)
128^3	100	128	1862	1809	53	1	50	2	0.3
		256	1354	1322	32	1	29	2	
		512	1116	1096	20	1	17	2	
		1024	745	729	16	1	14	3	
256^3	100	512	3090	3016	74	2	69	3	1.7
		1024	2391	2346	45	2	39	4	
		2048	1861	1830	32	2	26	4	
		4096	1334	1305	29	2	15	12	
512^3	50	2048	3852	3684	167	4	157	6	14
		4096	2008	1918	89	3	77	9	
		8192	1784	1722	62	3	48	11	
		16384	1406	1344	61	2	32	27	
1024^3	25	8192	2331	2119	212	6	186	20	101
		16384	1446	1289	157	4	113	40	

All tests are tabulated above. Tessellation time scales comparably with simulation time and occupies a small fraction of the total time. Neighborhood exchange time is minimal. Output time is small in most cases, although it is significant on larger problem sizes.

Scalability

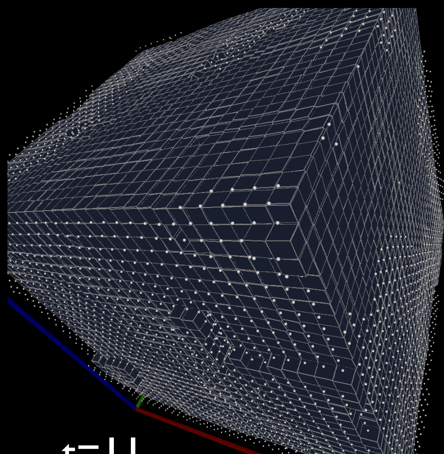


For 128^3 particles, 41 % strong scaling for total tessellation time, including I/O; comparable to simulation strong scaling.

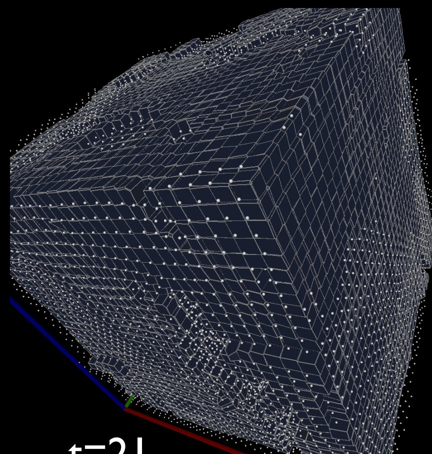


86 % weak scaling for total tessellation time, including I/O.

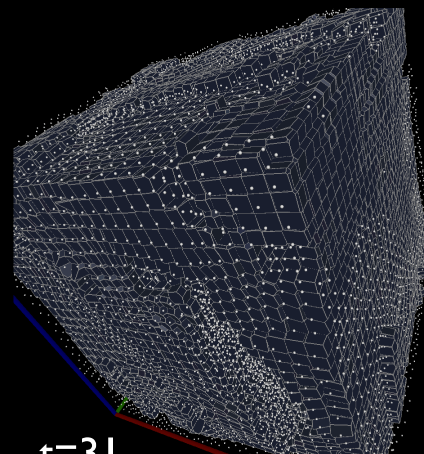
Temporal Cosmic Evolution



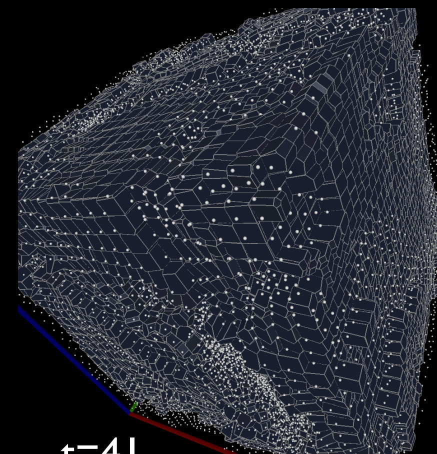
t=11



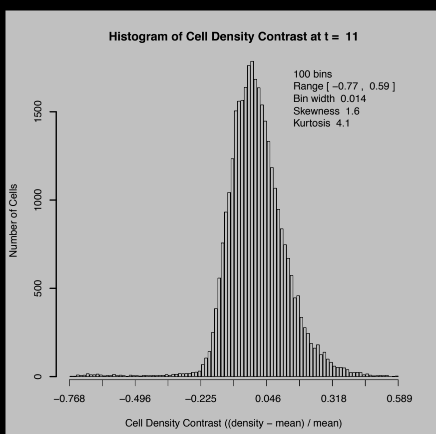
t=21

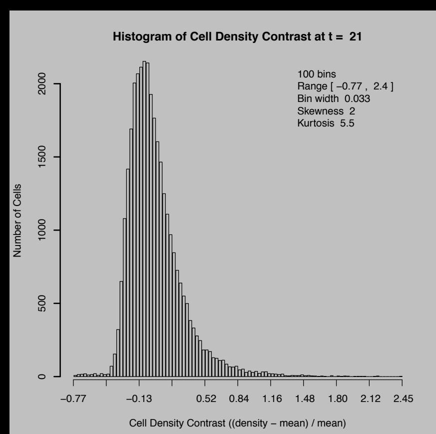


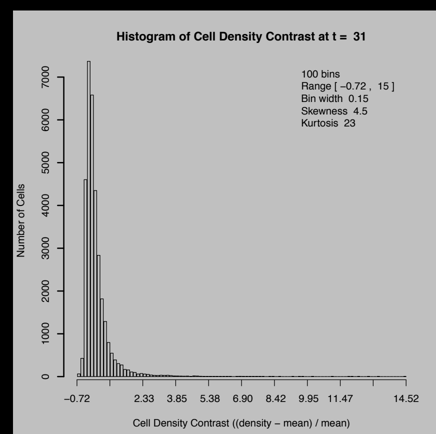
t=31

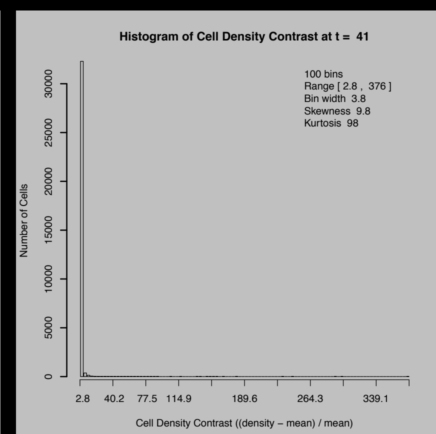


t=41

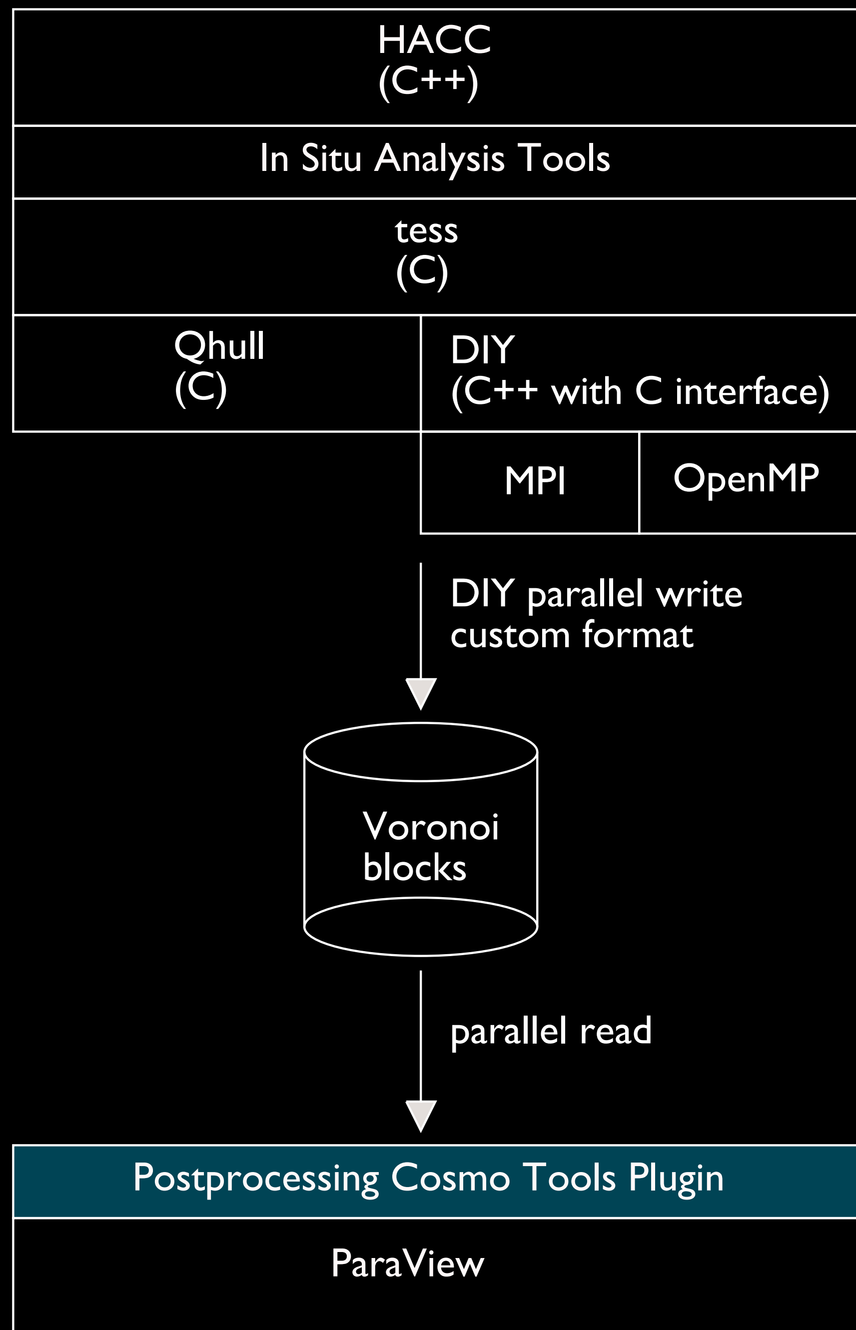








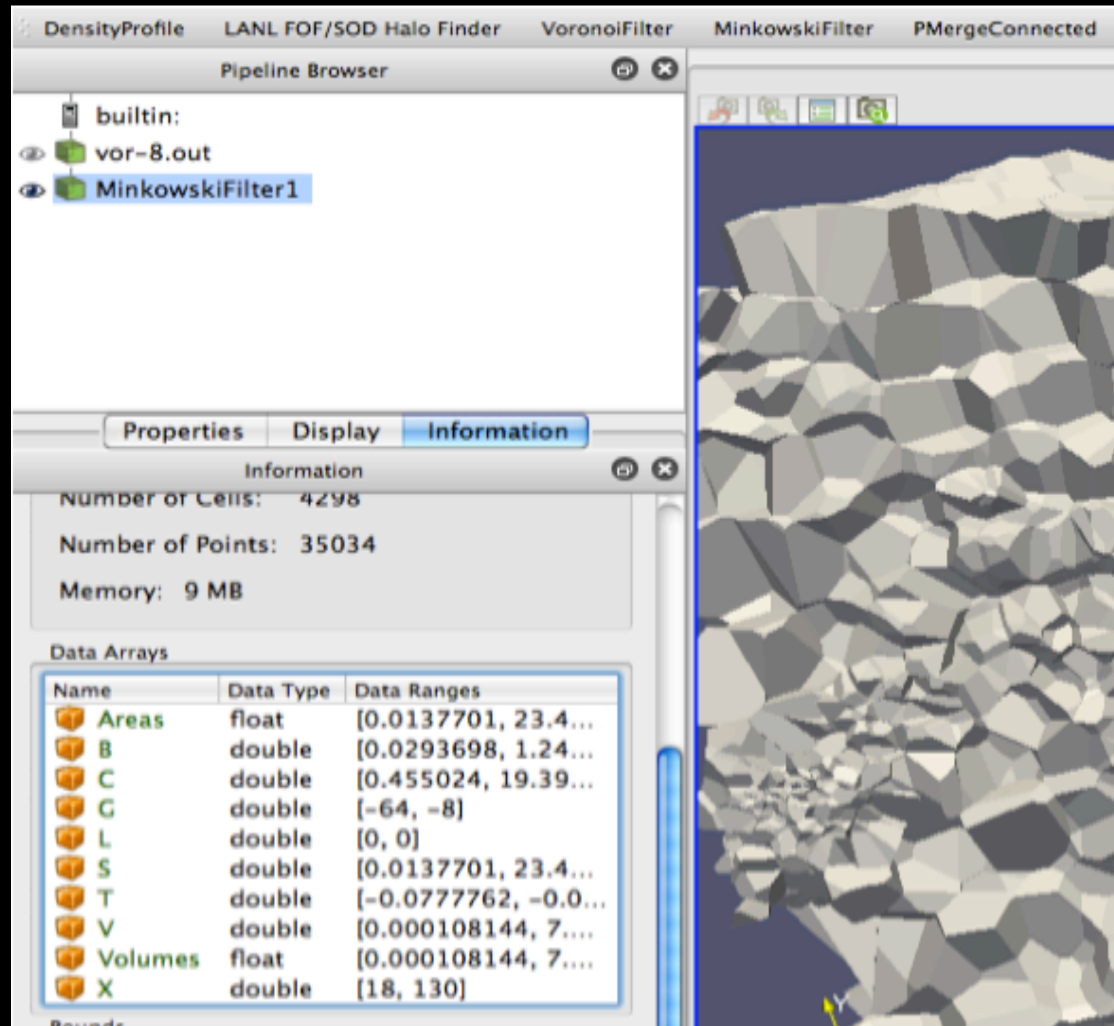
The early time steps begin with a normally distributed cell size and shape. As time progresses, the range of volume and density expands. The kurtosis increases as the distributions become more pointed, and skewness increases as well. These statistics are consistent with the governing physics that predict the formation of high- and low-density structures over time and can perhaps be used to summarize evolution at given time steps.


HACC
(C++)
In Situ Analysis Tools
tess
(C)
Qhull
(C)
DIY
(C++ with C interface)
MPI
OpenMP
DIY parallel write
custom format
Voronoi
blocks
parallel read
Postprocessing Cosmo Tools Plugin
ParaView

Visualization and Subsequent Analysis

Cosmo tools plugin in ParaView includes

- Parallel reader for Voronoi output
- Threshold filter (existing)
- Parallel connected component labeling filter
- 4 Minkowski functionals: volume, area, extrinsic curvature, genus are recognized by cosmologists for classifying structures



The cosmology tools ParaView plugin provides interactive feature exploration of Voronoi tessellation computed in situ in HACC.

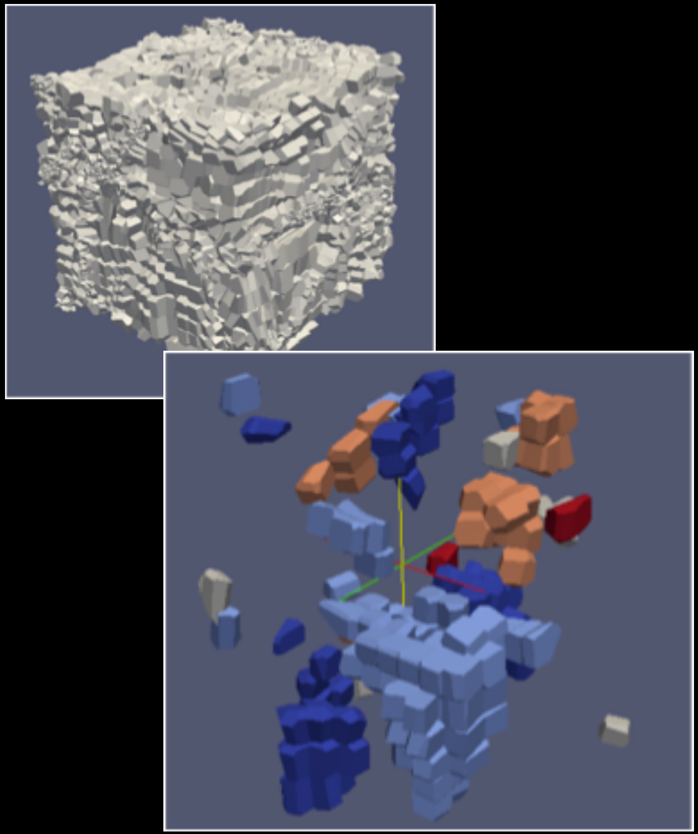
Connected Components of Voronoi Cells as Cosmological Voids

Recall

- Parallel reader for Voronoi output
- Threshold filter (existing)
- Parallel connected component labeling filter

Minkowski Functionals (MF)

- Basic
  - Volume *V*
  - Surface area *S*
  - Extrinsic curvature *C*
  - Genus *G*
- Derived
  - Thickness *T*
  - Breadth *B*
  - Length *L*



Connected components of Voronoi cells that have been filtered on cell volume are further characterized according to their Minkowski functionals.

Application of MF

- Compare different simulations
- Study percolation theory
- Find, characterize, and study shapes of clusters and voids

Sheth, Sahni, Shandarin, Sathyaprakash. Measuring the Geometry and Topology of Large-Scale Structure Using SURFGEN. 2003

Acknowledgments:

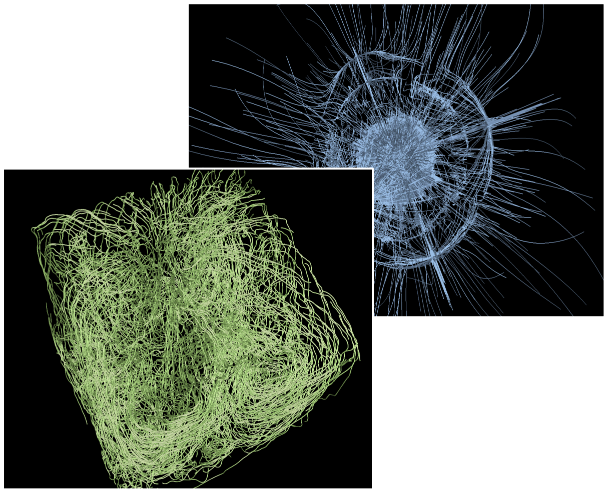
Facilities
Argonne Leadership Computing Facility (ALCF)

Funding
US DOE SciDAC SDAV Institute

People
Juliana Kwan, Adrian Pope, Hal Finkel, Katrin Heitmann, Salman Habib, Jingyuan Wang, George Zagaris, Sergei Shandarin

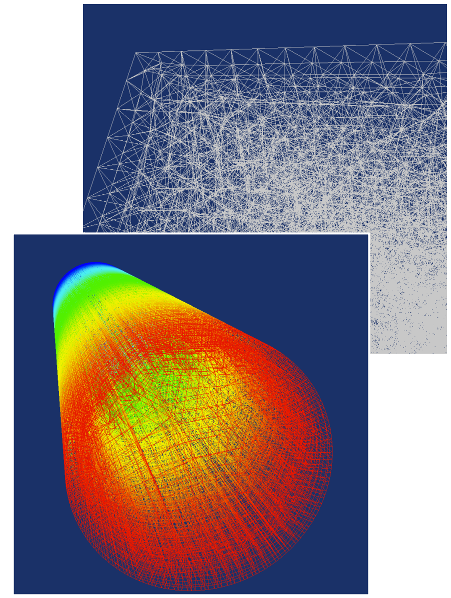
Tom Peterka
tpeterka@mcs.anl.gov
Mathematics and Computer Science Division

Postdoctoral Position in Data-intensive Analysis, Visualization, and Storage

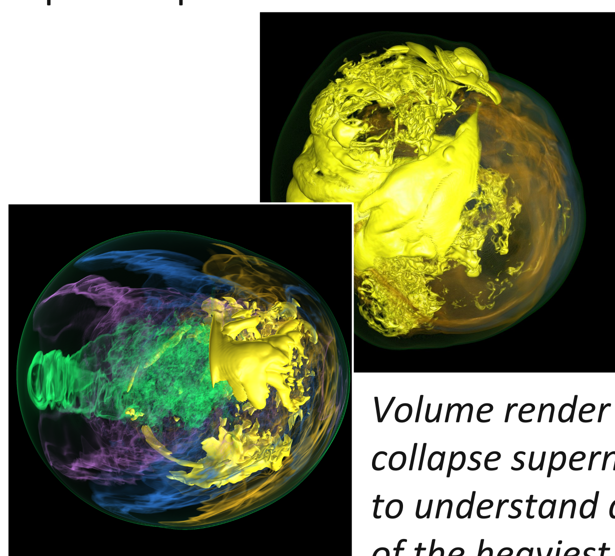


Integrate pathlines in parallel to help scientists understand complex flow behavior in time-varying vector fields.

Join a vibrant multidisciplinary research team of computer scientists and discover new parallel scalable algorithms for data analysis, visualization, and storage of scientific data. Integrate your research into parallel simulations running on some of the world's largest supercomputers.



Generate unstructured tetrahedral and spectral meshes to analyze nuclear reactor cooling.



Volume render core-collapse supernovae to understand death of the heaviest stars.

Contact: Tom Peterka
MCS Division
Argonne National Laboratory
tpeterka@mcs.anl.gov
For more information and to apply, visit www.mcs.anl.gov/~tpeterka and click on the Hiring link.