

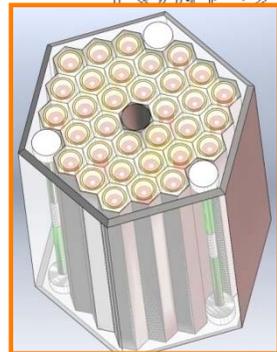
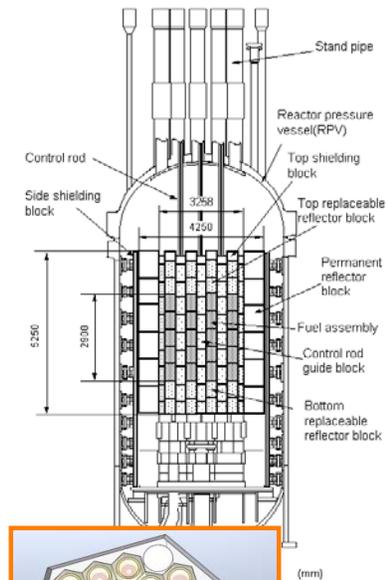
---

# Interoperable Mesh Components for Large-Scale, Distributed Memory Simulations

Lori Diachin, LLNL  
Representing the ITAPS Team



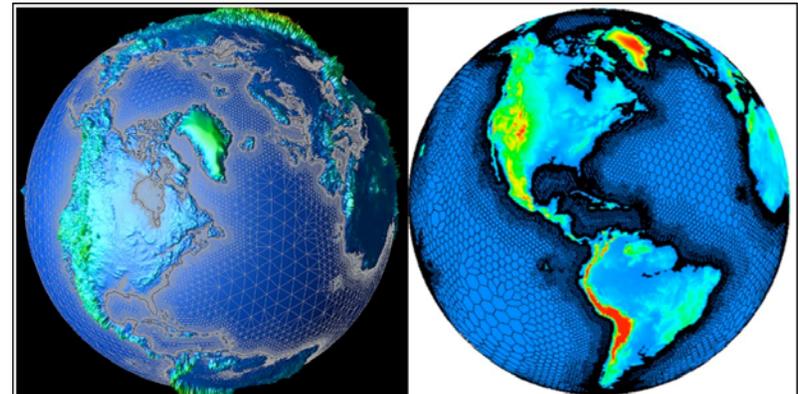
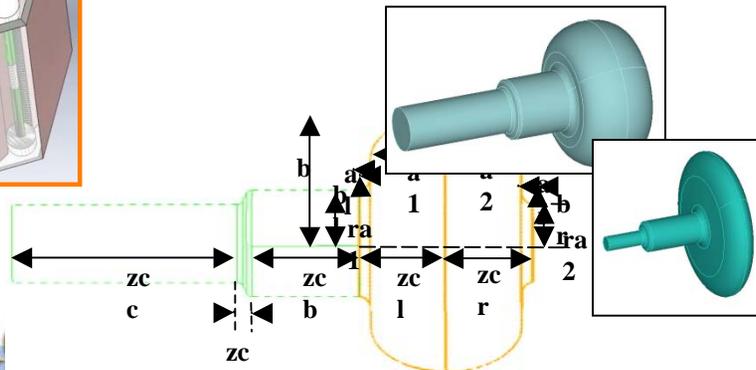
# DOE and SciDAC applications have sophisticated mesh and geometry needs



## Examples:

- Complex geometry in accelerator and nuclear energy applications require high order methods and interaction with the computational domain
- Multiple spatial scales in fusion, accelerator and climate apps require adaptive mesh refinement
- Multi-phase and multi-material models in fusion, groundwater, and nuclear energy apps require front tracking

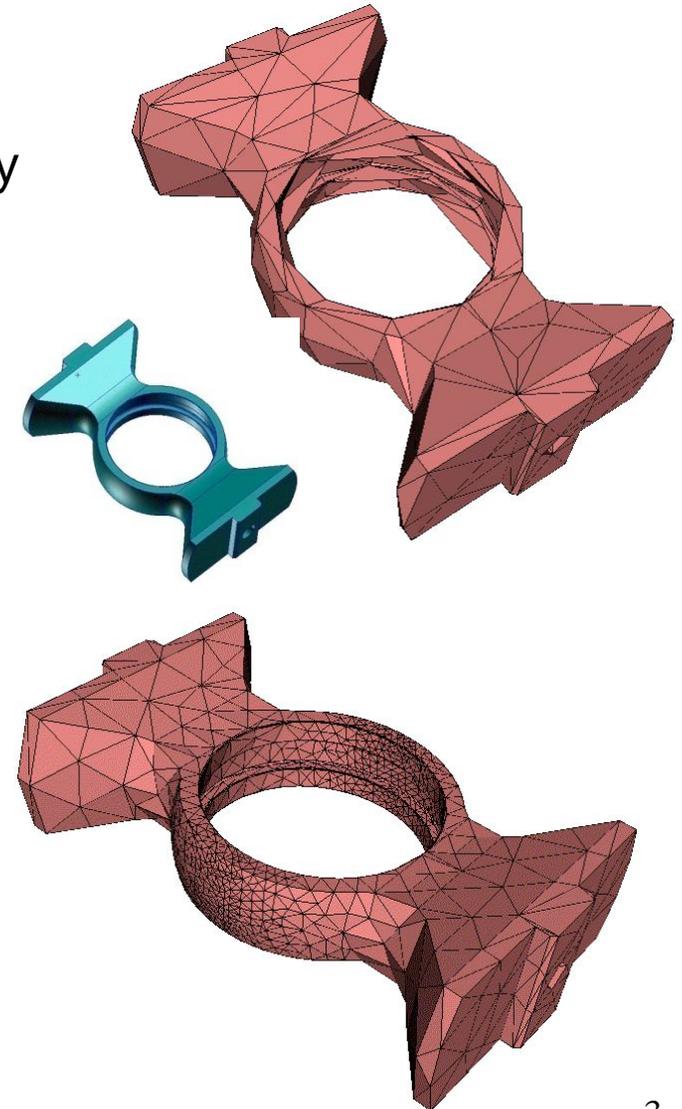
*All of this must be done on petascale-class computers*



# Unstructured mesh methods are commonly used for numerical simulation

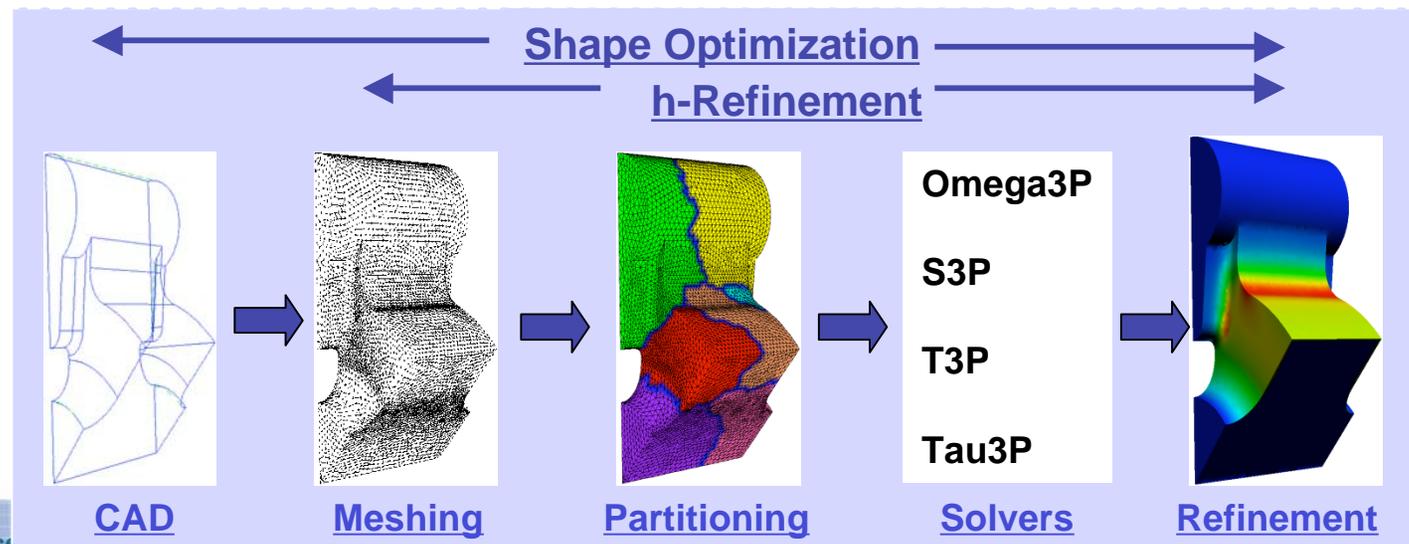
---

- Some Advantages
  - Meshes of mixed topologies and order easy
  - Mesh adaptation can account for curved domains
  - General mesh anisotropy can be obtained
  - Easy to create strong mesh gradations without special numerical techniques
  - Alignment with multiple curved geometric features
- Some Disadvantages
  - Data structures larger and more complex
  - Solution algorithms can be more complex



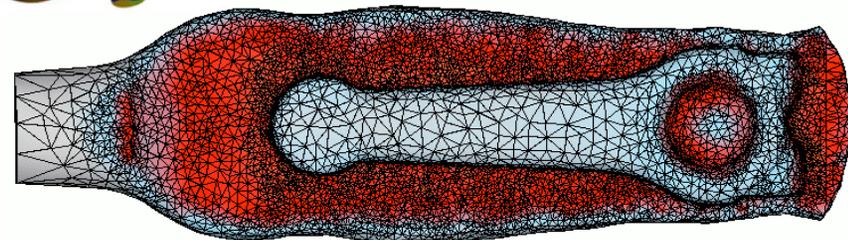
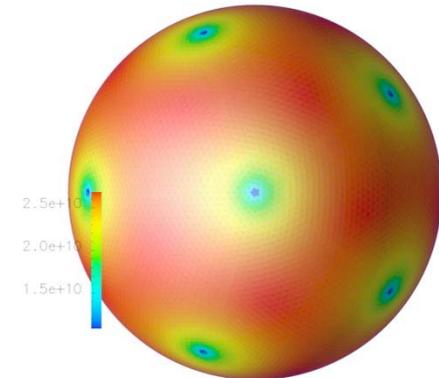
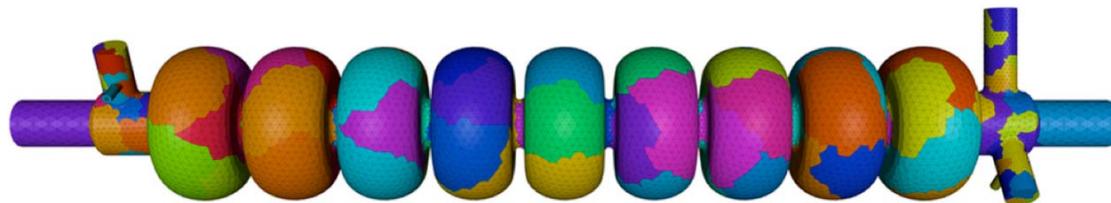
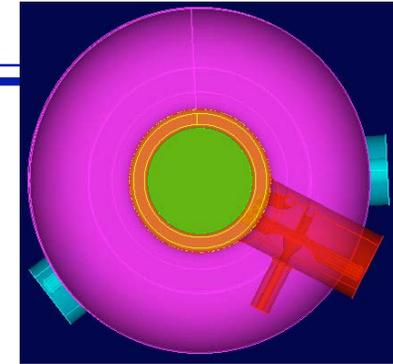
# Basic parallel solution on unstructured meshes has several key steps

- Construct** the initial mesh (serial or parallel)
- Improve the mesh using **smoothing** and **swapping**
- Partition** the mesh across processors
- Solve the PDE on mesh and estimate the error
- While error > tolerance
  - Refine, coarsen, improve and repartition** the mesh
  - Solve the PDE on the mesh and estimate the error
- End



# The ITAPS team has developed tools to address these steps

- CAD interaction: CGM
- Mesh generation: GRUMMP, NWGrid
- Mesh databases: FMDB, MOAB
- Mesh improvement: Mesquite, swapping tools
- Parallel Adaptive loops: MeshAdapt
- Front tracking: Frontier
- Partitioning: Zoltan



# While these tools exist, significant challenges remain

---

Developing and using these technologies requires significant software expertise from application scientists

- Difficult to improve existing codes
- Difficult to design and implement new codes

These tools all meet particular needs, but

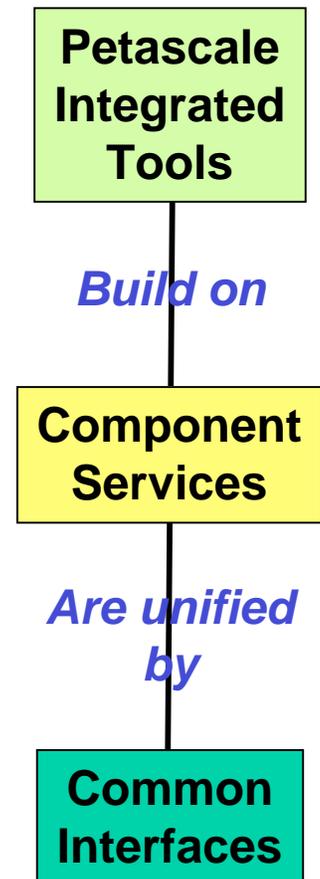
- They do not interoperate to form high level services
- They cannot be easily interchanged in an application

***The ITAPS center is developing key technologies to ease the use of advanced meshing tools on large scale parallel computers***

# ITAPS uses a component-based approach to address these challenges

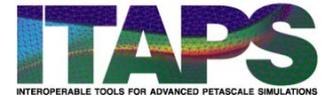
---

- Develop and deploy key mesh, geometry and field manipulation *component services* needed for petascale computing applications
- Develop advanced functionality *integrated services* to support SciDAC application needs
  - Combine component services together
  - Example: parallel AMR requires mesh database, geometry tools, local refinement/coarsening operations, load balance
  - Unify tools with *common interfaces* and *abstract data model* to enable interoperability
- Work with key application teams to insert ITAPS technologies into simulations



# There are several advantages of the component-based approach ITAPS uses

---

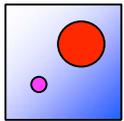


- Focus on *abstract data models* and *interfaces* not on data structures or file formats
- Use independent interfaces for distinct data model abstractions to make adoption easier
- Incremental adoption for applications; only service dependent interfaces need to be implemented
- Finer granularity of interoperable functionality reduces the need to mix huge libraries together

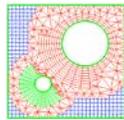
# The ITAPS data model abstracts PDE-simulation data hierarchy

---

- Core Data Types



- *Geometric Data*: provides a high level description of the boundaries of the computational domain; e.g., CAD, image, or mesh data (**iGeom**)



- *Mesh Data*: provides the geometric and topological information associated with the discrete representation of the domain (**iMesh**)



- *Field Data*: provides access to time dependent physics variables associated with application solution. These can be scalars, vectors, tensors, and associated with any mesh entity. (**iField**)

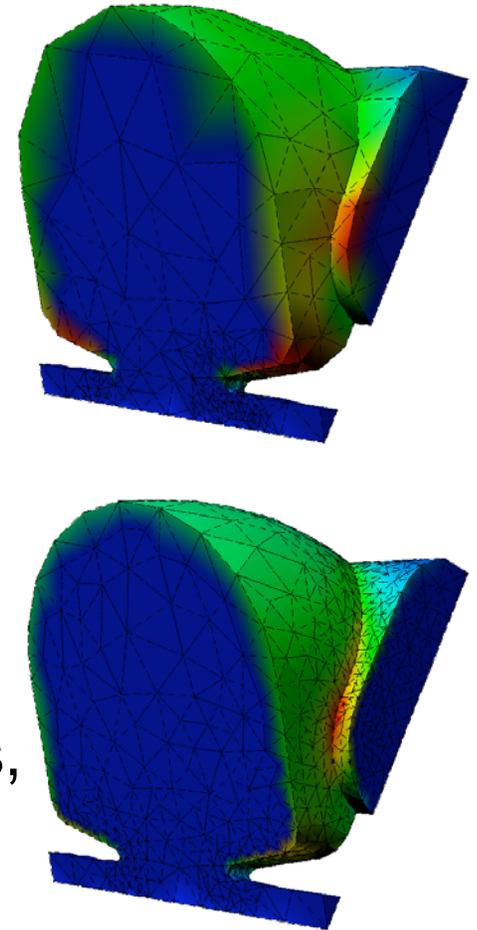
- Data Relation Managers (**iRel**)

- Provides control of the relationships among the core data types
- Resolves cross references between entities in different groups
- Provides functionality that depends on multiple core data types

# iMesh(P) provides access to the discrete representation of the domain

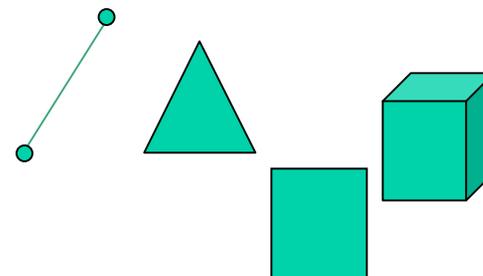
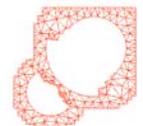
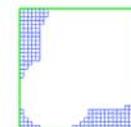
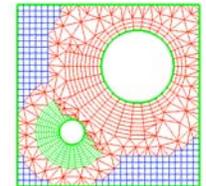
---

- iMesh supports local access to the mesh
- iMeshP complements iMesh with parallel support
- Required functionality:
  - Access to mesh geometry and topology
  - User-defined mesh manipulation and adaptivity
  - Grouping of related mesh entities together (e.g. for boundary conditions)
- Builds on a general data model that is largely suited for unstructured grids
- Implemented using a variety of mesh types, software, and for a number of different usage scenarios



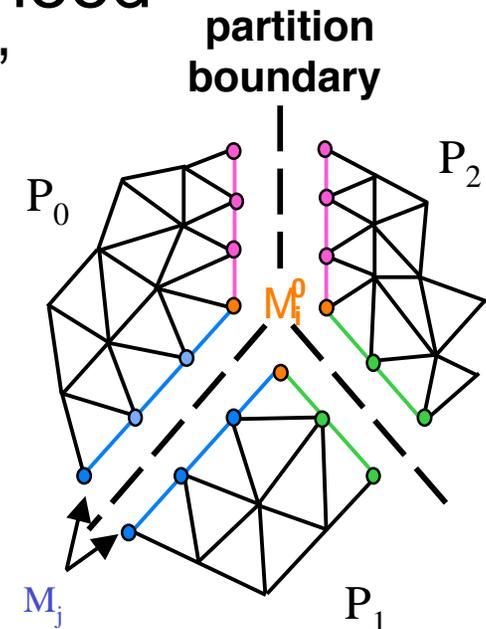
# The serial iMesh interface supports basic and advanced local operations

- Provides basic access to vertex coordinates and adjacency information
  - Mesh loading and saving
  - Global information such as the root set, geometric dimension, number of entities of a given type or topology
  - Access to all entities in a set as single entities, arrays of entities, or entire set
  - Set/remove/access tag data
- Set functionality
  - Boolean operations (union, subtract, intersect)
  - Hierarchical relationships
- Mesh modification
  - Adding / Deleting entities
  - Vertex relocation



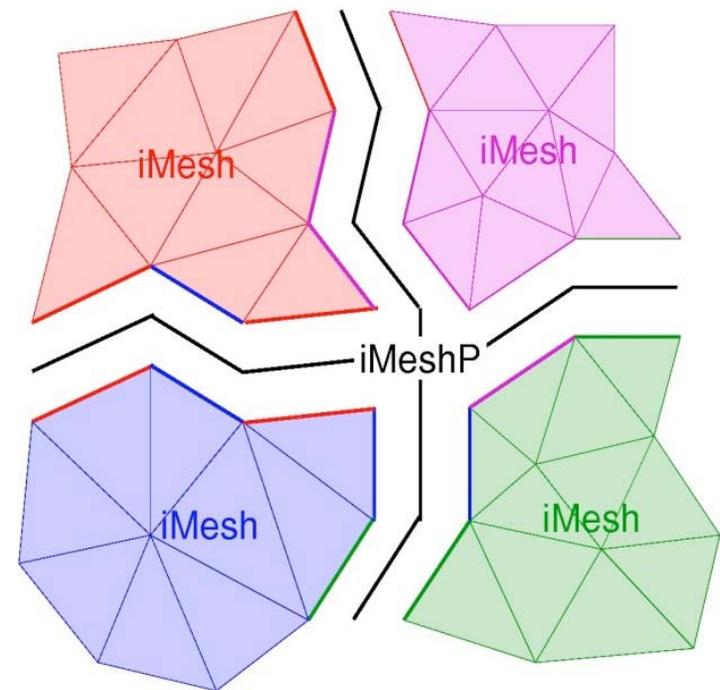
# Unstructured Meshes on Parallel Computers

- Typically the mesh is distributed over independent memories
- A mesh partition groups mesh entities and places them into parts
- Applications using a partitioned mesh need
  - Communication links for between “shared” mesh entities on neighboring parts
  - Ability to move mesh entities between parts (while maintaining links)
  - Algorithms to maintain load balance of parts which minimizing communications



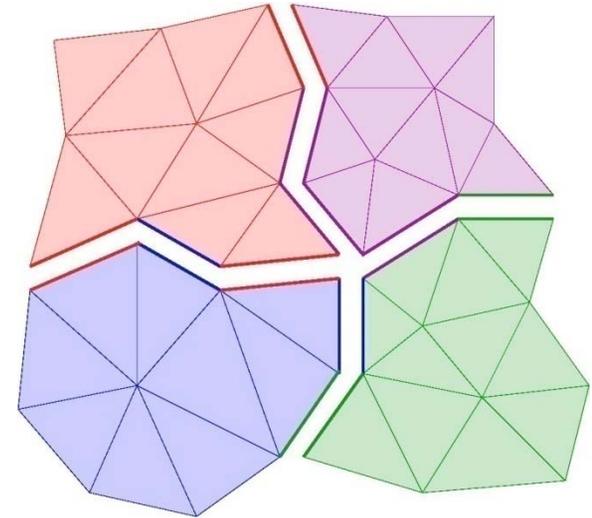
# iMeshP extends iMesh to support parallel computations

- Focus on distributed memory
  - For example: use application's MPI communicators
  - But allow use of global address space and shared memory paradigms
- Leverage serial iMesh
  - Works as expected within a process
  - Works as expected for global address space and shared memory programs



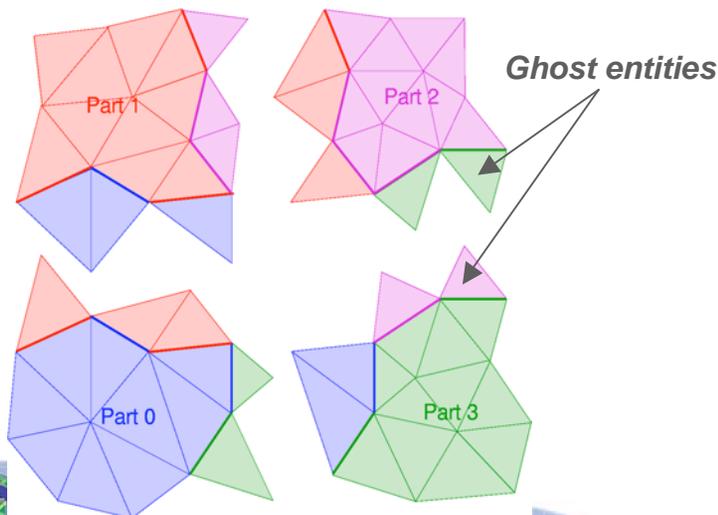
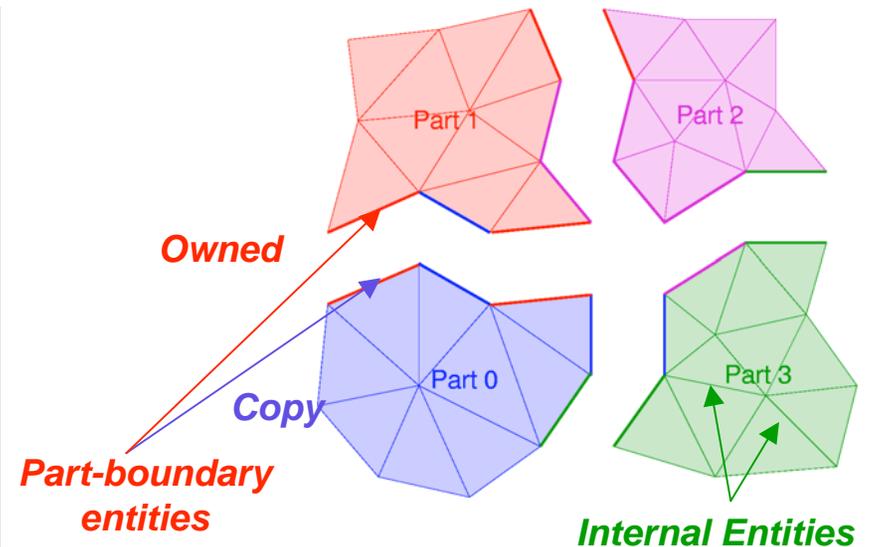
# Distributed mesh representations have several functional requirements

- Entity ownership
  - Each mesh entity is owned by exactly one part
  - Ownership imbues right to modify
  - Ownership is not static during the course of a simulation
    - Repartitioning
    - Local micro-migration
  - Some entities have read-only **copies** on other parts (e.g. along part boundaries and ghosts)
- Communication links
  - Efficient mechanisms to update mesh partitioning and keep the links between parts are mandatory



# ITAPS has a well-defined partition model data abstraction

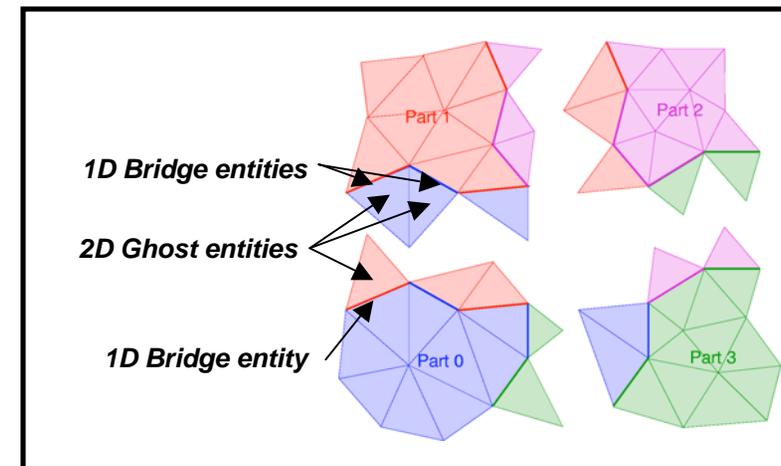
- *Process*: a program executing; MPI process
  - # of processes == MPI\_Comm\_size
  - Process number == MPI\_Comm\_rank
- *iMesh instance*: mesh database provided by an implementation
  - One or more instances per process
- *Partition*: describes a parallel mesh
  - Maps entities to subsets called *parts*
  - Maps parts to processes
  - Has a communicator associated with it



- *Ownership*: right to modify an entity
- *Internal entity*: Owned entity not on an interpart boundary
- *Part-Boundary entity*: Entity on an interpart boundary
- *Ghost entity*: Non-owned, non-part-boundary entity in a part
- *Copies*: ghost entities + non-owned part-boundary entities

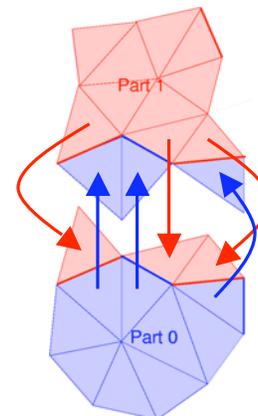
# The iMeshP interface supports the needs of parallel computation

- Build and maintain partition and entity ownership information
  - Partition creation and modification
  - Entity ownership status
  - Ghost entity creation
  - Tag data retrieval and exchange on owned and copy data
  - Information about part boundaries and neighboring parts
  
- Parallel operations
  - Large and small scale entity migration
  - Update of coordinate information
  - Coordination of new entities along part boundaries
  - Synchronous and asynchronous operations supported



One layer of ghost triangles for all boundary triangles sharing edges:

- Ghost-entity dimension = 2
- Bridge-entity dimension = 1
- Number of layers = 1



Ghost-entity updates of tag data from owners

# Simple Example: HELLO iMeshP

```

#include "iMesh.h"
#include "iMeshP.h"
#include <mpi.h>
int main(int argc, char *argv[]) {
    char *options = NULL;
    iMesh_Instance mesh;
    iMeshP_PartitionHandle partition;
    int num_vtx, num_parts, ierr, options_len=0;
    iBase_EntitySetHandle root;
    /* create the Mesh instance */
    iMesh_newMesh(options, &mesh, &ierr, options_len);
    iMesh_getRootSet(mesh, &root, &ierr);

    MPI_Init(&argc, &argv);
    /* Create the partition. */
    iMeshP_createPartitionAll(mesh, MPI_COMM_WORLD, &partition, &ierr);

    /* load the mesh */
    iMeshP_loadAll(mesh, partition, root, argv[1], options, &ierr,
        strlen(argv[1]), options_len);

    /* Report number of Parts in Partition */
    iMeshP_getNumParts(mesh, partition, &num_parts, &ierr);
    iMeshP_getNumOfTypeAll(mesh, partition, root, iBase_VERTEX, &num_vtx, &ierr);
    . . .

```

- 1) *Instantiates Partition*
- 2) *Reads mesh into mesh instance and Partition*
- 3) *Reports # parts in Partition*
- 4) *Get global number of vertices*

1

2

3

4

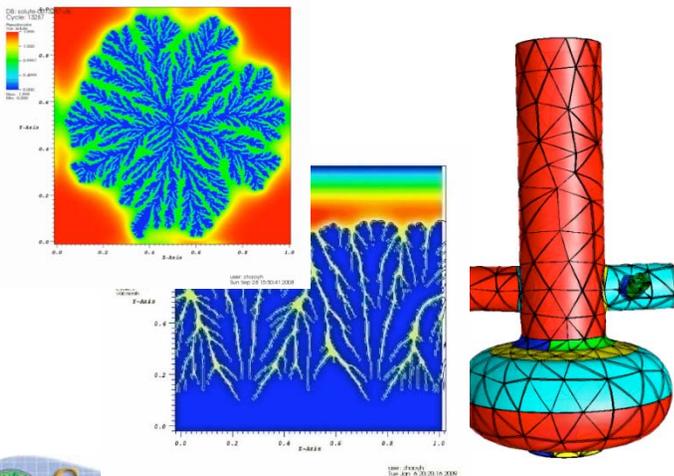
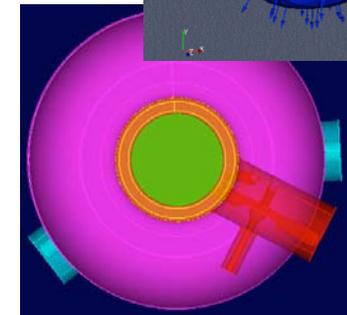
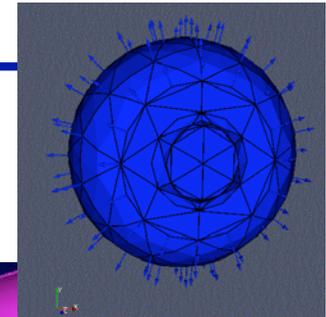
# Four ITAPS mesh components provide iMesh(P) functionality



iMesh Implementation	Emphasis	Parallel Capability	Applications
FMDB: Flexible Mesh DataBase	Entity addition/removal for adaptation	Scalable to at least 32K procs, 1B elements; uses iZoltan.	<i>fusion, accelerators, CFD, solid mechanics, multiphase flow</i>
MOAB: Mesh Oriented dAtaBase	Low memory usage first, then CPU time.	Up to 64 procs with flexible mesh loading based on geometric volume, material, partition	<i>nuclear reactors, accelerators, radiation transport, inertial confinement fusion, glacier dynamics</i>
GRUMMP: Generation & Refinement of Mixed-element Meshes in Parallel	Fast adjacency retrieval for mesh generation, improvement and adaptation.	In development.	CFD, biological systems, structural mechanics
NWGrid: NorthWest Grid Generation Code	Simplicial meshes; parallel generation of unstructured, hybrid, adaptive meshes.	Parallelism based on Global Arrays.	CFD, <i>subsurface transport</i>

# ITAPS services use iMesh(P) to support SciDAC application needs

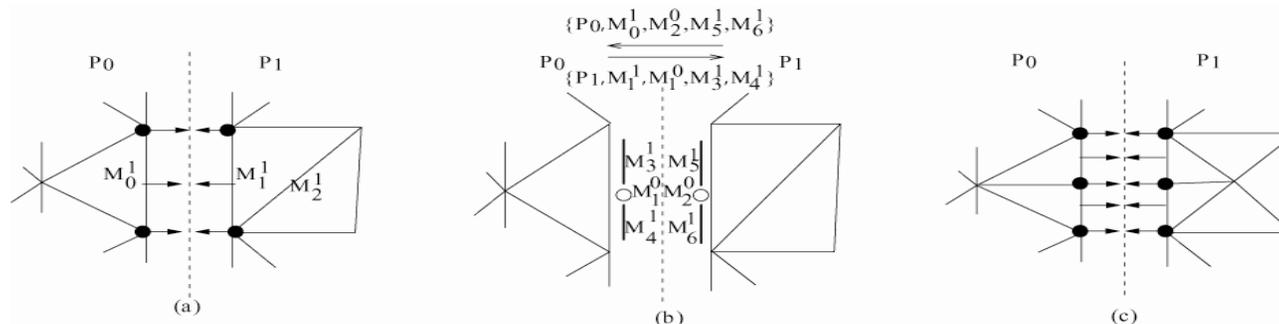
- Provides interfaces used by ITAPS services
  - Mesh and geometry tools
  - Mesh quality improvement (Mesquite/Swapping)
  - ➔ – Adaptive mesh refinement (MeshAdapt)
  - Front Tracking (FronTier)
  - Partitioning and load balancing (iZoltan)
  - Visualization and I/O (VisIt and iMeshIO)



	Coordinates	Adjacency	Other queries	Iterators	Modification	Basic Sets	Tags	Parallel	iGeom/iRel	Total
Mesquite	1	1	4	4	1	2	13		6	32
Swapping	1	1	4	4	2					12
Mesh Adapt	1	2	5	3	3		7	13		34
FronTier Lite	3	1	5	3	3		3			18
Zoltan	1	2	5					14		22
VisIt	1	1	9	3		1	16			31

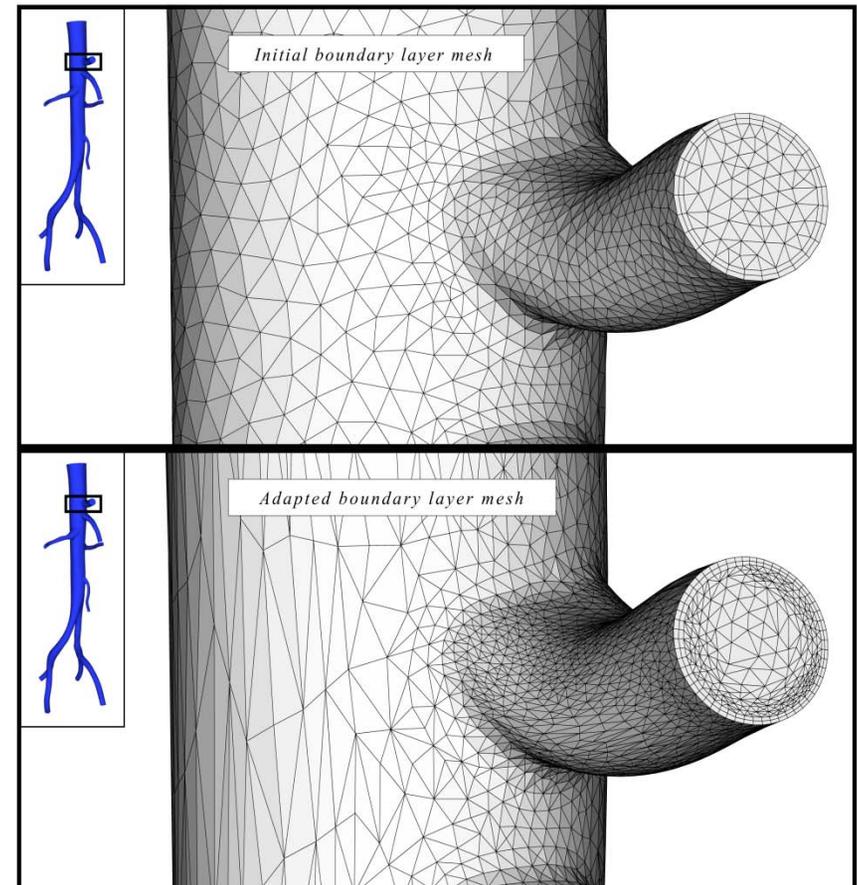
# Implementation of MeshAdapt procedure

- Given the “mesh size field”:
  - Examine element size and shape (in transformed space)
  - If mesh size field not met, select “best” modification
  - Split or swap out edges that are too long
  - Short edges eliminated
  - Operations must be sure to account for curved domain boundaries for placing new or moved vertices
- Determination of “best” mesh modification
  - Selection of modifications based on element properties
  - Appropriate consideration of neighboring elements



# MeshAdapt builds on ITAPS interfaces and services

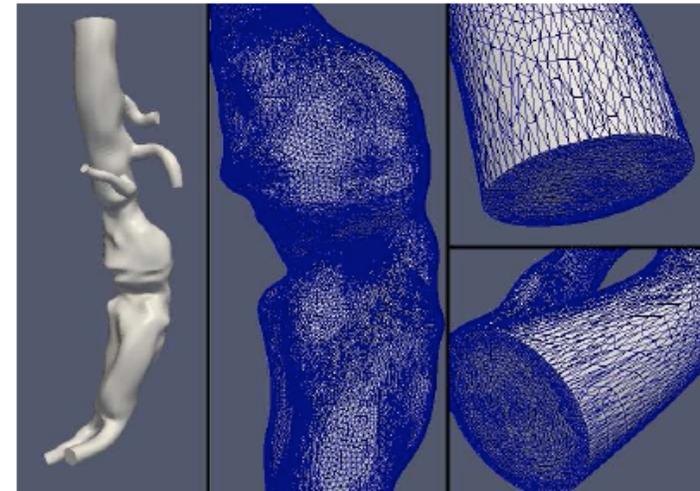
- iMeshP, iMesh supports mesh interactions in parallel
  - Entity ownership and communication link management
  - Entity migration
  - Tag data exchange
- iGeom provides interface to geometry
- iZoltan performs dynamic load balancing



# MeshAdapt and iZoltan used to prepare strong scaling study on 128K processors



- PHASTA CFD solver
  - Implicit time integration - iterative system solution at each time step
  - Employs the partitioned mesh for system formulation and solution
- PHASTA's work characterized as:
  - Organized and regular communication between parts that “touch” each other
  - A specific number of ALL-REDUCE communications also required
- ITAPS Services used
  - FMDB for the mesh database
  - MeshAdapt for refinement up to 32K
  - iZoltan to partition the mesh to 128K
- Strong scaling highlights need for advanced partitioning algorithms



1 billion element anisotropic mesh on Intrepid Blue Gene/P

#of cores	Rgn imb	Vtx imb	Time (s)	Scaling
16k	2.03%	7.13%	222.03	1
32k	1.72%	8.11%	112.43	0.987
64k	1.6%	11.18%	57.09	0.972
128k	5.49%	17.85%	31.35	0.885

# ITAPS is exploring two partition improvement procedures

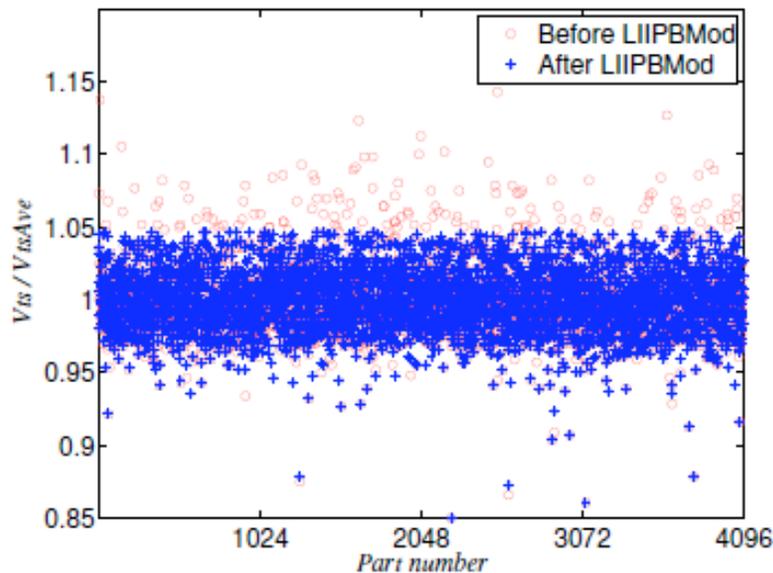
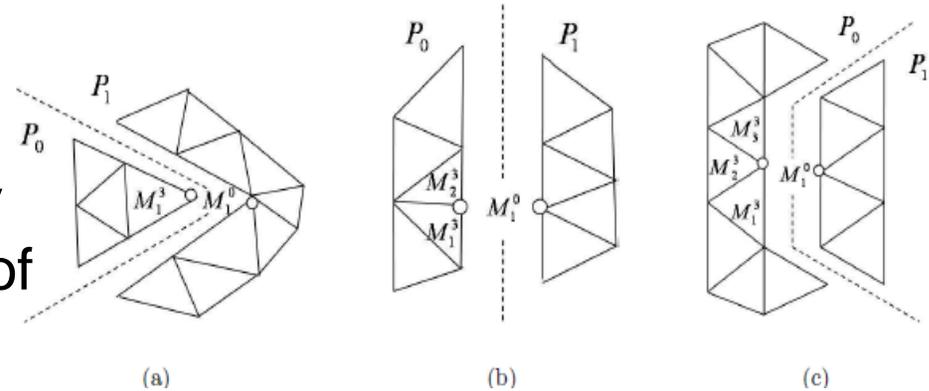
- Graph-based partitioners typically balance a single selected partition object (e.g. mesh regions)
- Objective: Incremental redistribution of mesh entities to improve overall balance
- Algorithms:
  - Local Iterative Inter-Part Boundary Modification (LIIPBMod)
  - Heavy Part Splitting (HPS)

Table: Region and vertex imbalance for a 8.8 million region mesh on a bifurcation pipe model partitioned to different number of parts

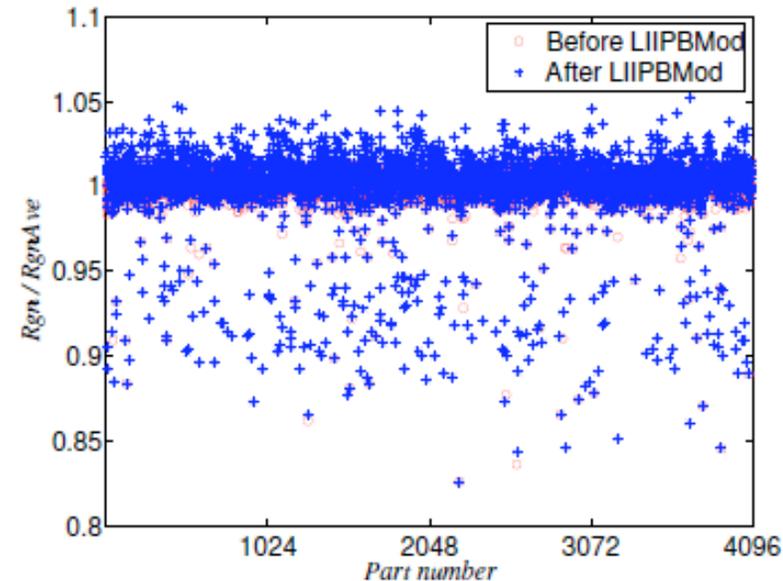
num of parts	128	256	512	1024	2048	4096	8192
Ave numRgns	68.856K	34.463K	17.231K	8.616K	4.308K	2.154K	1.077K
Ave numVts	14.02K	7.262K	3.780K	1.984K	1.050K	559.98	305.96
Region imbalance	1.022	1.019	1.021	1.014	1.024	1.021	1.029
Vertex imbalance	1.034	1.036	1.053	1.069	1.092	1.143	1.154

# LIIPBMod Algorithm

- Migrate vertices bounding a small number of adjacent elements
- Vertices with only one remote copy considered to avoid the possibility of creating nasty part boundaries



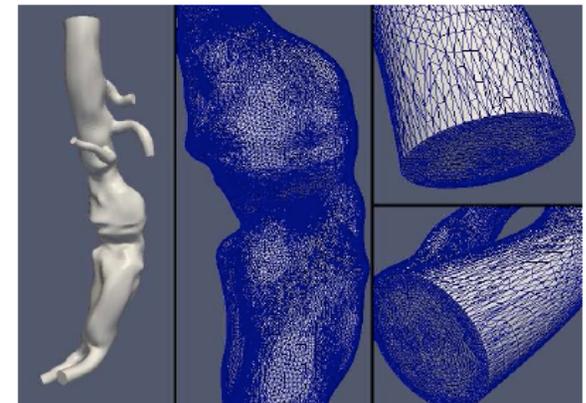
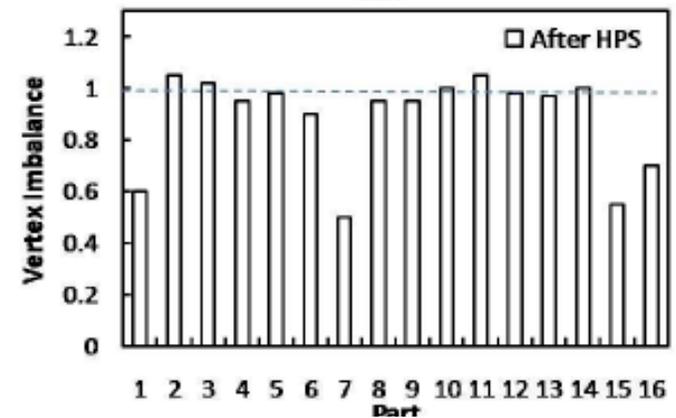
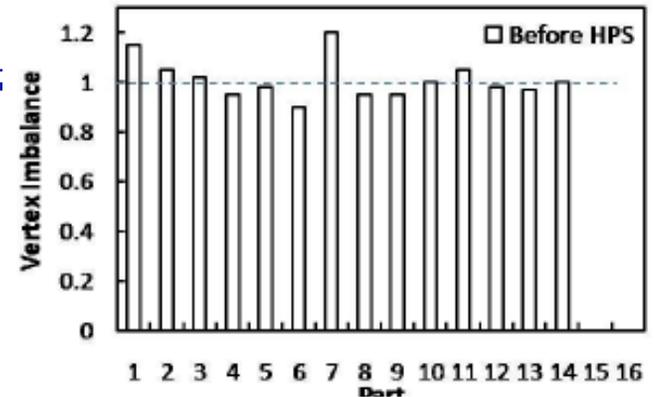
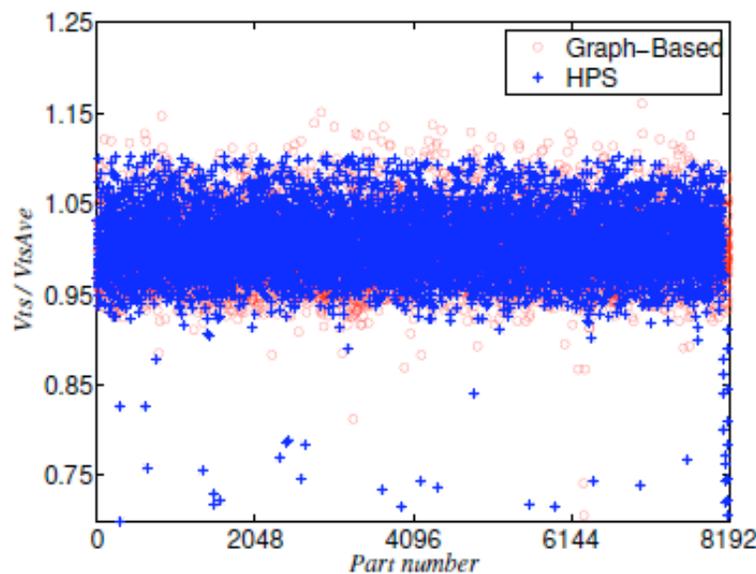
Vertex imbalance: from 14.3% to 5%



Region imbalance: from 2.1% to 5%

# HPS Algorithm

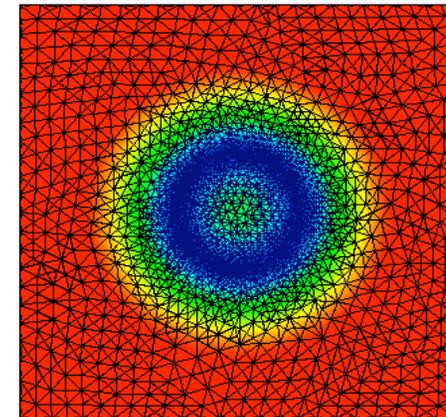
- Peak imbalance determines the scalability, HPS hence improves the performance
- Distribute the mesh to 99%\*numP by graph partitioner, leave 1%\*numP parts empty
- Select up to 1%\*numP parts with the highest vertex load and split into two parts



# Scaling studies of uniform refinement on up to 32K processors

- Weak scaling uniform refinement

# of Parts	Initial Mesh	Adapted Mesh	Time (s)	Scaling Factor
2048	17M	128M	5.0	1.0
4096	34M	274M	4.8	1.05
8192	65M	520M	5.1	0.97
16384	520M	1.1B	6.1	0.82
32768	274M	2.2B	7.4	0.68



- Mesh adaptation characterized by small, but variable, work per operation - “perfect scaling” too costly
  - Can run on the large numbers of parts used in the analysis
  - Will still be a small % of total solution time - the mesh adapt example given is 0.04% of the estimated solve time
- Improvements to message passing can improve scaling

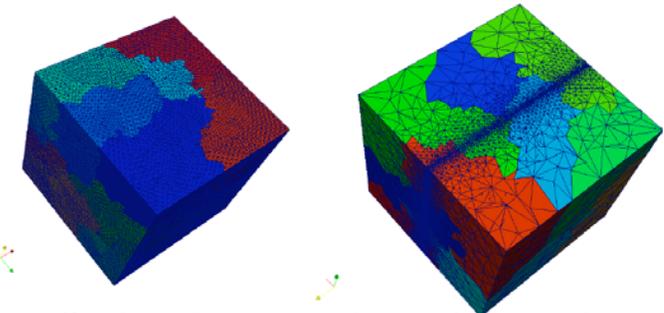
# IPComMan - Communication package to increase scalability of mesh-based operations



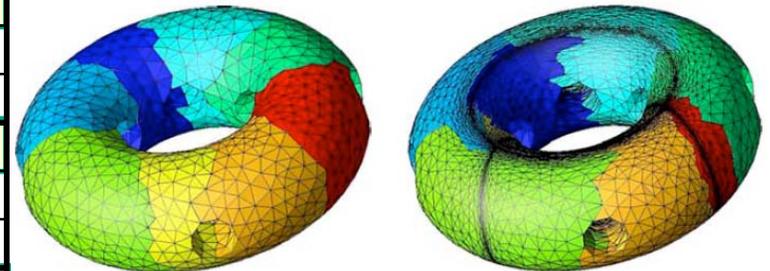
- For computations characterized by large numbers of messages of various sizes - automatic message packing, non-blocking
- Takes advantage of message-passing within neighborhoods, eliminates as many collective calls as possible
- Neighborhood in parallel applications
  - Subset of processors exchanging messages during a specific communication round
  - Bounded by a constant, typically under 40, independent of the total number of processors for many applications
  - Support for dynamically changing neighborhood during communication steps

# IPComMan Results for Mesh Migration

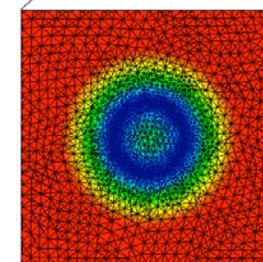
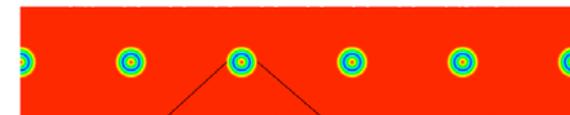
	N/proc	128	256	512	1024	2048	4096
C U B E	Autopack	90	75	73	82	96	142
	Scaling	1	0.60	0.31	0.14	0.06	0.02
	Migrate/Total	0.32	0.39	0.46	0.55	0.63	0.71
	IPComMan	75	41	34.5	32.5	26.5	21
	Scaling	1	0.91	0.54	0.29	0.18	0.11
	Migrate/tot	0.29	0.26	0.30	0.33	0.34	0.31
T O R U S	Autopack	385	291	120	117	125	146
	Scaling	1	0.66	0.80	0.41	0.19	0.08
	Migrate/Total	0.39	0.40	0.47	0.51	0.62	0.73
	IPComMan	310	166	75	50	37	25
	Scaling	1	0.93	1.03	0.78	0.52	0.39
	Migrate/tot	0.34	0.28	0.36	0.31	0.34	0.34
B U B L E	Autopack				116	99	102
	Scaling				1	0.59	0.28
	Migrate/Total				0.31	0.39	0.51
	IPComMan				75	64	41
	Scaling				1	0.59	0.46
	Migrate/Total				0.22	0.28	0.31



Cube isotropic adaptation



Torus Anisotropic adaptation



Adaptation for moving bubbles.

Maximum communication time in migration during adaptation

# Initial scaling studies of MeshAdapt with preliminary IPComMan



- Strong scaling of uniform refinement on Ranger 4.3M to 2.2B elements

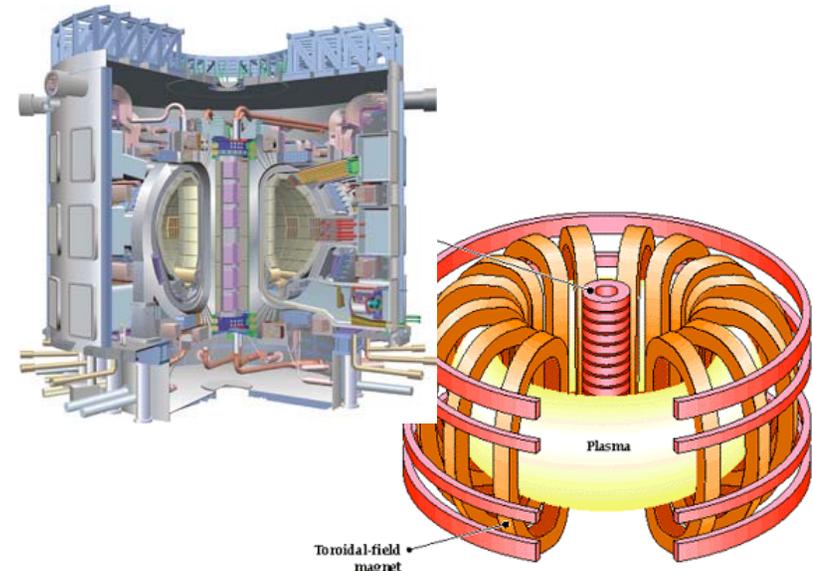
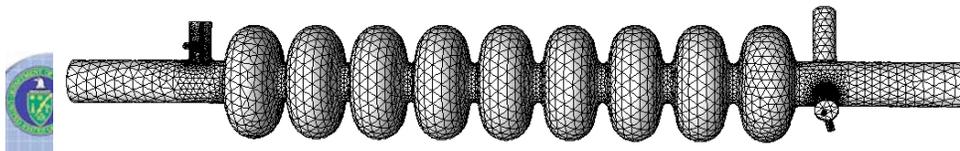
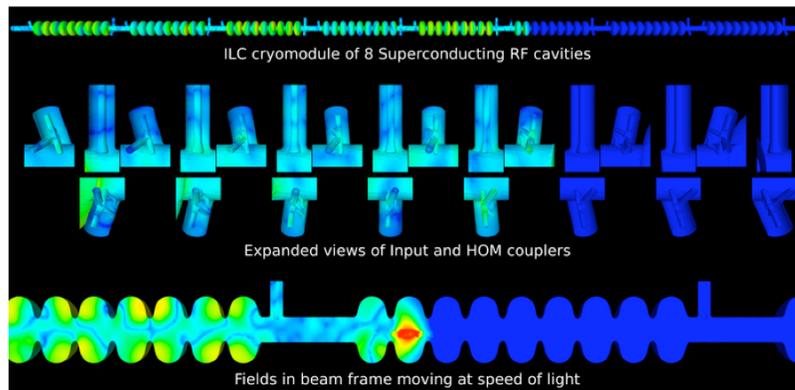
# of Parts	Time (s)	Scaling Factor
2048	21.5	1.0
4096	11.2	0.96
8192	5.67	0.95
16384	2.73	0.99

- Nonuniform field driven refinement (with mesh optimization operations) on Ranger 4.3M to 730 elements (time for dynamic load balancing not included)

# of Parts	Time (s)	Scaling Factor
2048	110.6	1.0
4096	57.4	0.96
8192	35.4	0.79

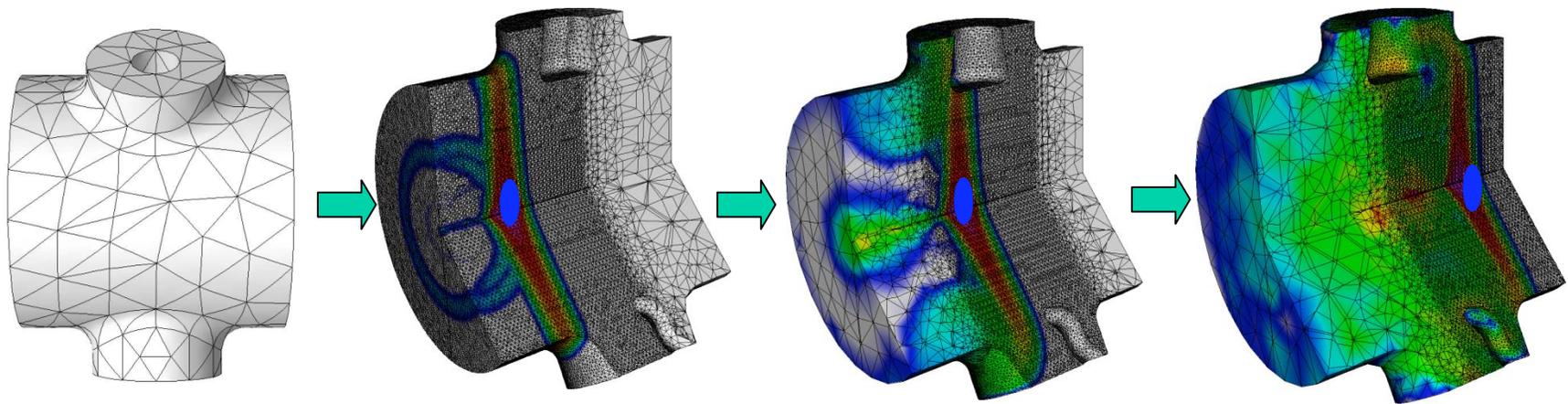
# MeshAdapt technologies are used by two key SciDAC application partners

- SLAC National Accelerator Laboratory as part of the ComPASS project working with Kwok Ko, Lie-Quan Lee, Lixin Ge and Cho Ng
- M3D-C1 development at PPPL as part of the CEMM project working with Steve Jardin, J. Chen and Nate Ferraro



# Impact of moving mesh adaptation for accelerator wakefield calculations

- Adaptively refined meshes have 1~1.5 million curved regions
- Reduced the number of elements needed by nearly an order of magnitude

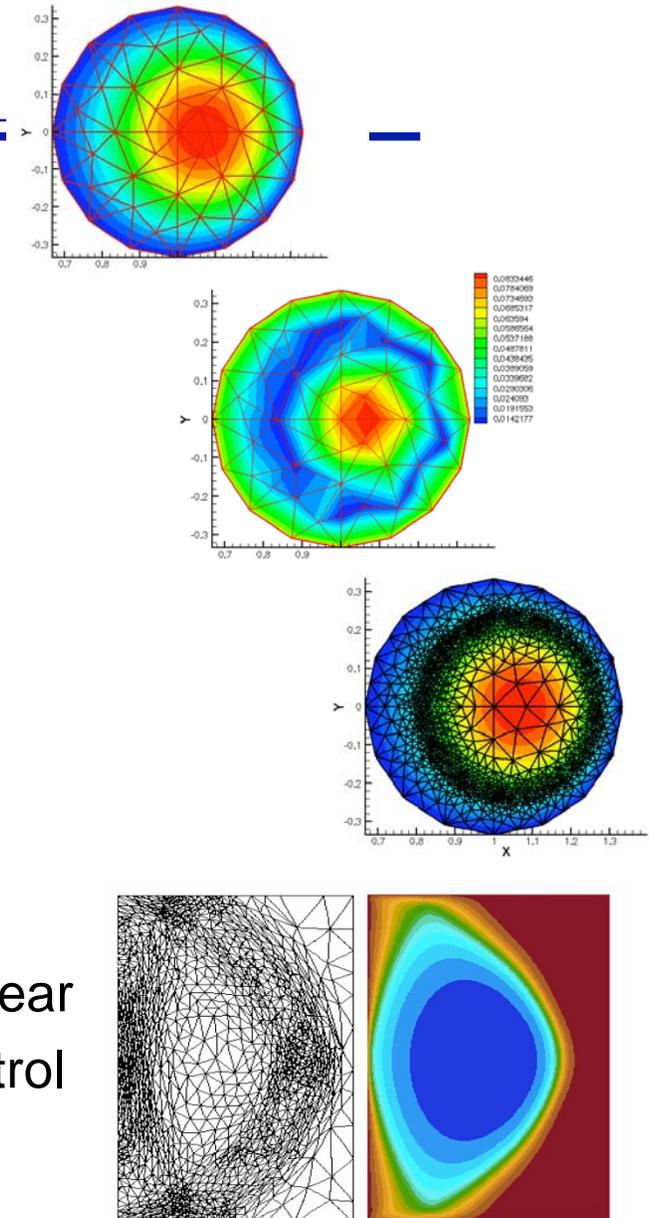


**Electric fields on the three refined curved meshes**

Best Meshing Technical Poster Award, 17th International Meshing Roundtable - X. Luo, M. Shephard, L.-Q. Lee, C. Ng, L. Ge, "Tracking Adaptive Mesh Refinement in 3D Curved Domains for Large-Scale Higher-Order Finite-Element Simulations"

# Integration of MeshAdapt with PPPL M3D-C1 MHD Code

- PPPL uses high order finite element methods
- PPPL and ITAPS jointly
  - Restructured code for integration with ITAPS technologies
  - Added new features such as ability to solve on curved domains
- Procedure:
  - Mesh size field construction
    - Hessian interpolation
    - Vorticity driven
  - Apply high-order local solution field transfer during mesh adaptation
- First results on curved domains obtained this year
- Progressing on anisotropic adaptive mesh control



## In summary, ITAPS is...

---

- Developing new interoperable technologies critical for high-fidelity single-physics, multi-physics, and multi-scale simulations
  - Common interface development
  - Services for complex mesh and geometry operations
- Working to ensure our technologies are effective on the current generation of petascale computers and conducting research on what is needed for the next generation
- Deploying key technologies for use by our SciDAC application partners
  - Accelerator design, fusion, nuclear reactors, groundwater, climate, and turbulence

# This work was accomplished by a strong multi-institutional team\*

- Karen Devine, SNL
- Lori Diachin, LLNL
- Jason Kraftcheck, Wisconsin
- Ken Jansen, RPI
- Vitus Leung, SNL
- Xiaojuan Luo, RPI
- Mark Miller, LLNL
- Carl Ollivier-Gooch, UBC
- Alex Ovcharenko, RPI
- Onkar Sahni, RPI
- Mark Shephard, RPI
- Tim Tautges, ANL
- Ting Xie, RPI
- M Zhou, RPI



Lori Diachin  
LLNL



Karen Devine  
SNL



Ken Jansen  
RPI



Carl Ollivier-Gooch  
UBC



Mark Shephard  
RPI

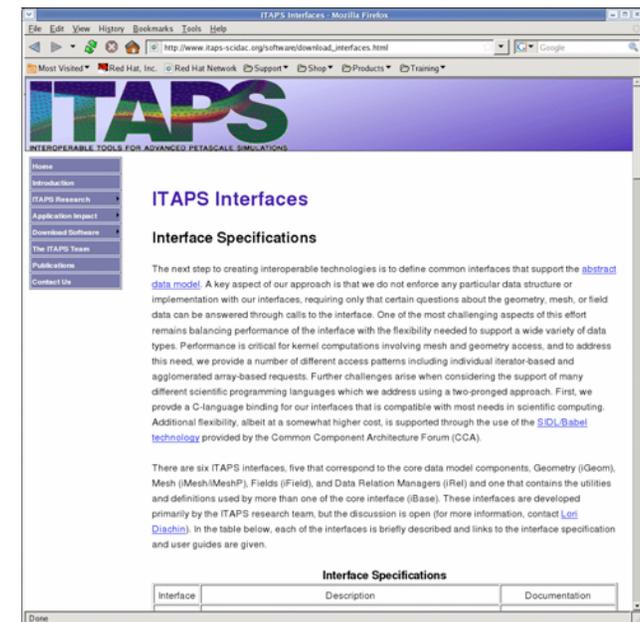


Tim Tautges  
ANL

\*This list represents the subset of the ITAPS team who were engaged in iMeshP definition, MeshAdapt, iZoltan, and work with fusion and accelerator scientists

# More Information

- Web Site: [www.itaps-scidac.org](http://www.itaps-scidac.org)
  - Comprehensive overview of ITAPS
  - Software available for download
  - Tutorial materials
- Email: [itaps-mgmt@lists.llnl.gov](mailto:itaps-mgmt@lists.llnl.gov)
- Lori Diachin: [diachin2@llnl.gov](mailto:diachin2@llnl.gov)



# Auspices and disclaimer

---

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

This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.