Fast Dynamic Load Balancing for Extreme Scale Systems

Cameron W. Smith, Gerrett Diamond, M.S. Shephard

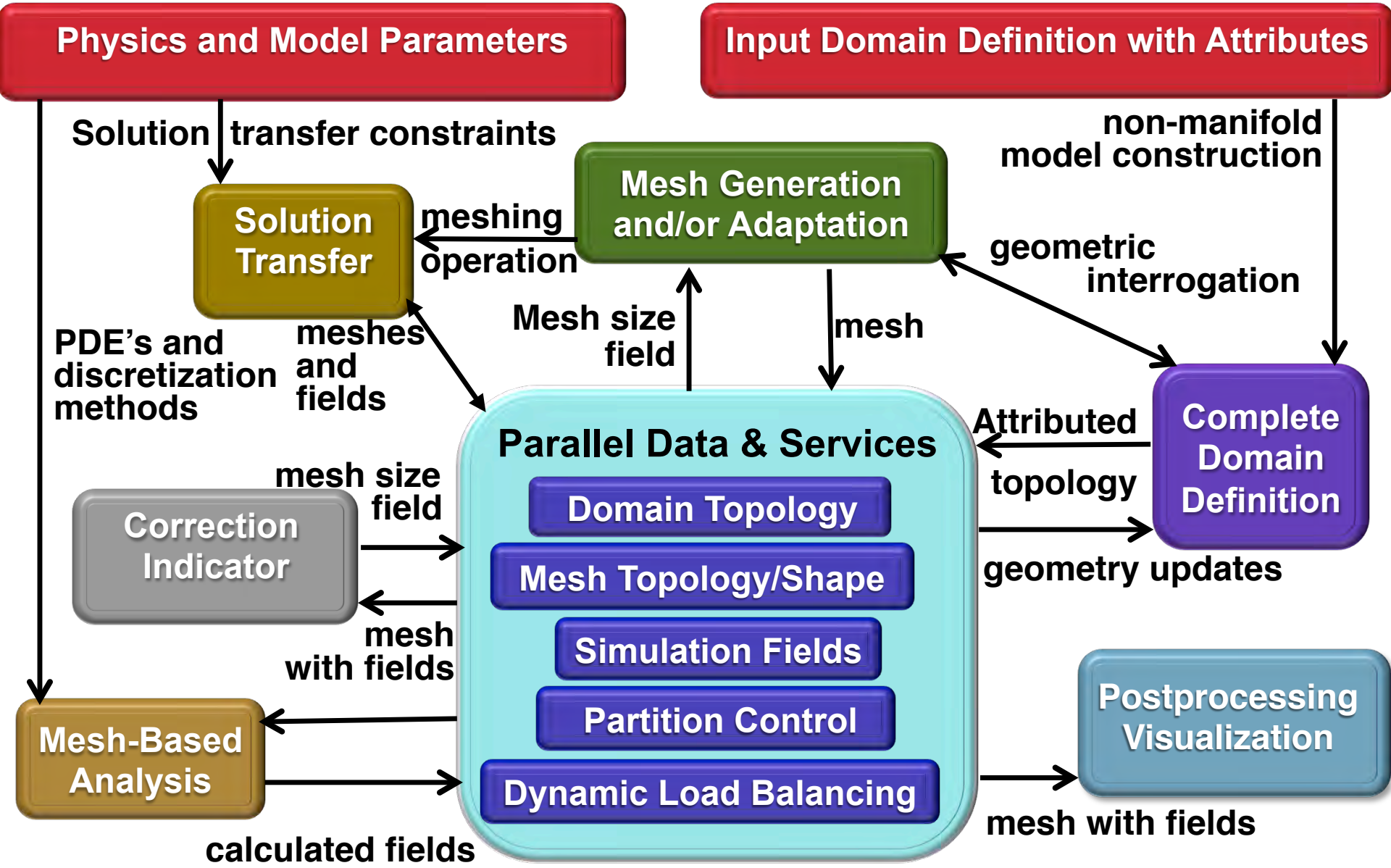
Computation Research Center (SCOREC)

Rensselaer Polytechnic Institute

Outline:

- Some comments on our tools for parallel unstructured mesh simulations
- Generalization of our multicriteria partition improvement procedures
- Applications being worked on

Geometry-Based Adaptive Simulation



Parallel Unstructured Mesh Infrastructure (PUMI)

PUMI Services:

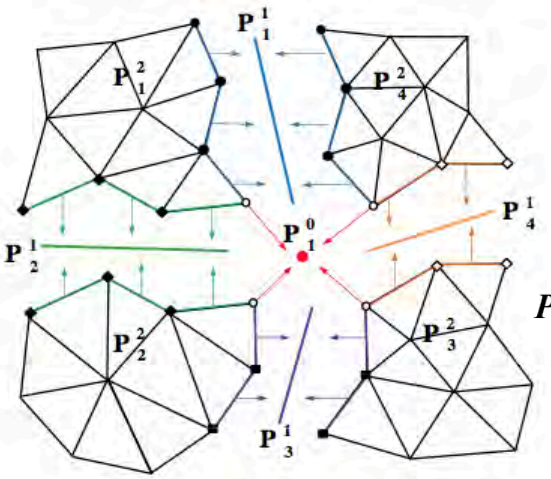
- Mesh and fields distributed across processes
  - Linked to geometry
  - Communication links
  - Ownership controls operations
- Entity migration
- Read only copies

MESH ADJACENCIES	GEOMETRIC DOMAIN ENTITIES
mesh region	region
mesh face	region or face
mesh edge	region, face or edge
mesh vertex	region, face, edge, or vertex

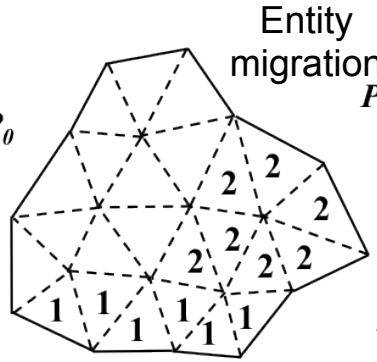
Geometric model

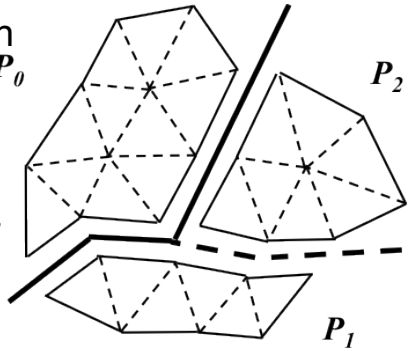
Partition model

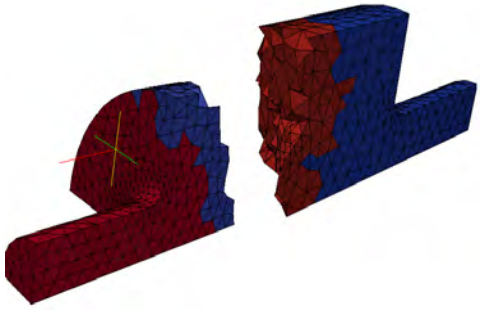
Distributed mesh



Communication links





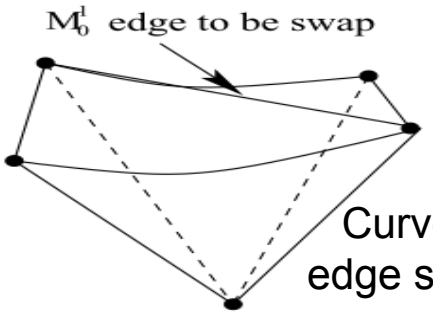


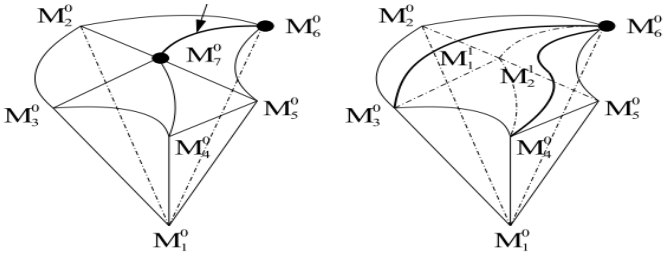
2 layers of read only copies

Parallel Curved Mesh Adaptation (MeshAdapt)

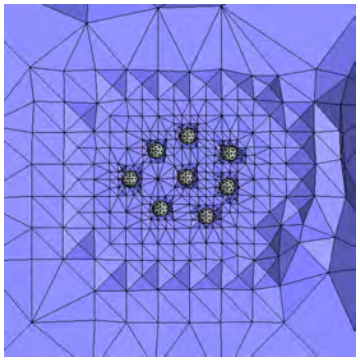
Fully parallel operating on distributed meshes

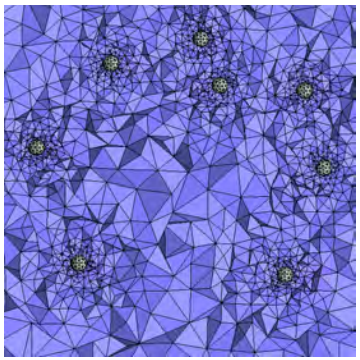
- General local mesh modification
- Adapts to curved geometry
- Driven by anisotropic mesh metric field
- Local on the fly solution transfer
- Supports curved mesh adaptation

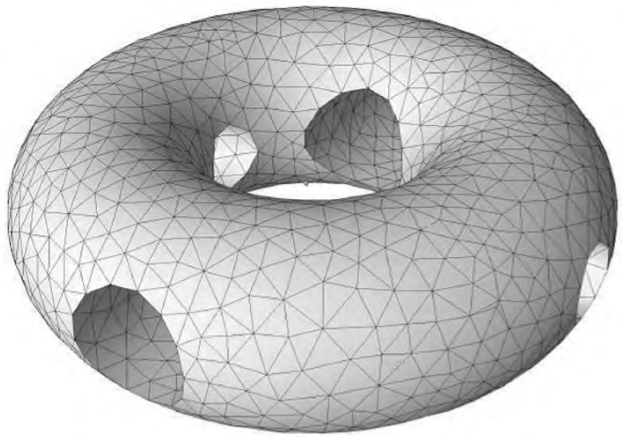


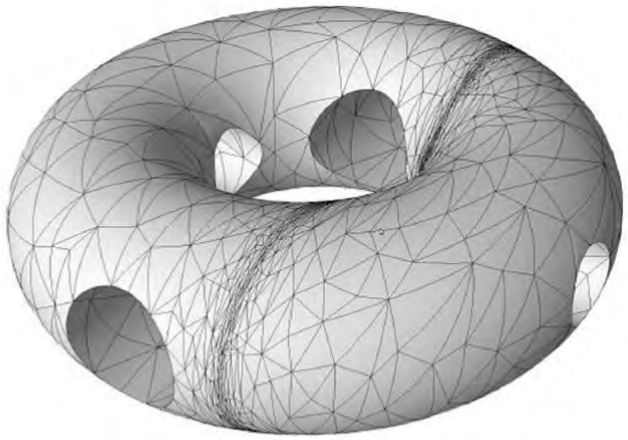


Curved edge collapse









Building In-Memory Parallel Workflows

A scalable workflow requires effective component coupling

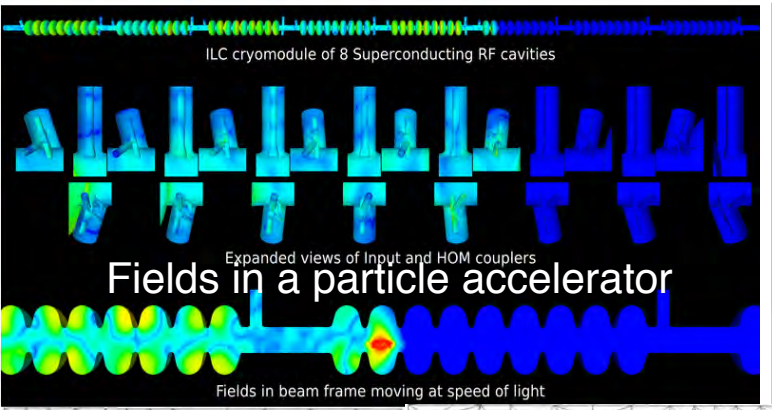
- Avoid file-based information passing
  - On massively parallel systems I/O dominates power consumption
  - Parallel filesystem technologies lag behind performance and scalability of processors
  - Unlike compute nodes, the file system resources are almost always shared and performance can vary significantly
- Use APIs and data-streams to keep inter-component information transfers and control in on-process memory
  - When possible, don't change horses
  - Component implementation drives the selection of an in-memory coupling approach
  - Link component libraries into a single executable

Parallel Unstructured Mesh Infrastructure

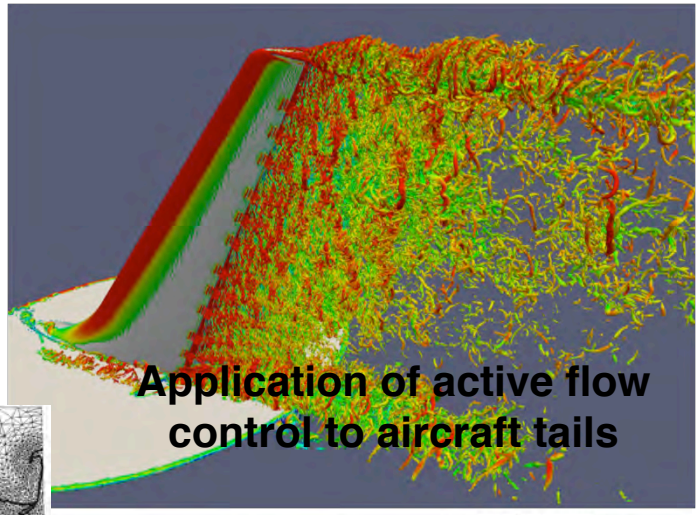
SCOREC unstructured mesh technologies:

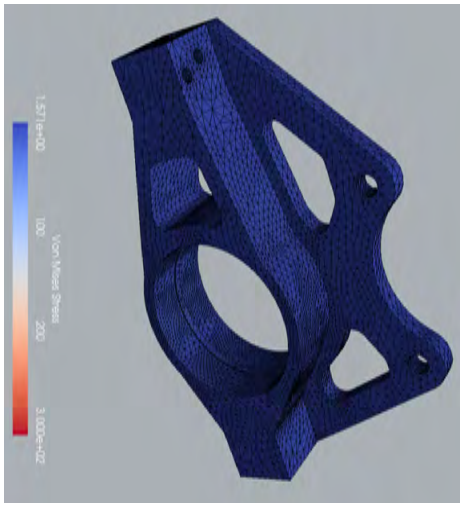
- PUMI – Parallel Unstructured Mesh Infrastructure (scorec.rpi.edu/pumi/)
- MeshAdapt – parallel mesh adaptation (https://www.scorec.rpi.edu/meshadapt/)
- ParMA (https://www.scorec.rpi.edu/parma/) and it generalization into EnGPar (http://scorec.github.io/EnGPar/) for multicriteria load balance improvement
- In-memory integration for parallel adaptive simulations for
  - Extended MHD with M3D-C1
  - Electromagnetics with ACE3P
  - Non-linear solids with Albany/Tirlinos multiphysics
  - FR fields in Tokamaks with MFEM multiphysics
  - CFD problems with PHASTA, Proetus, Fun3D, Nektar++

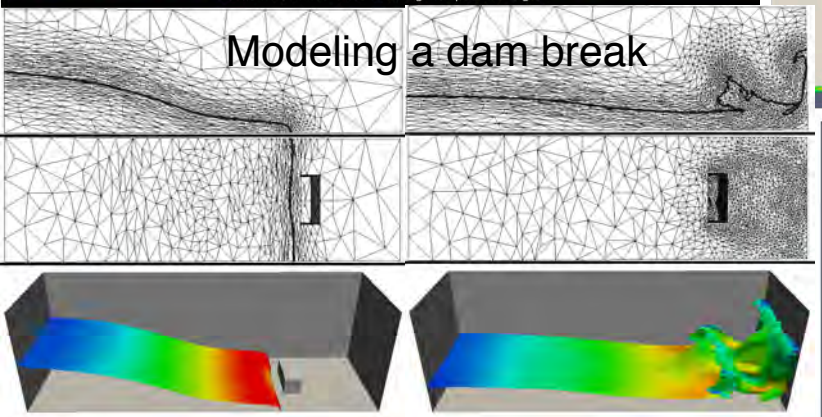
Application Examples

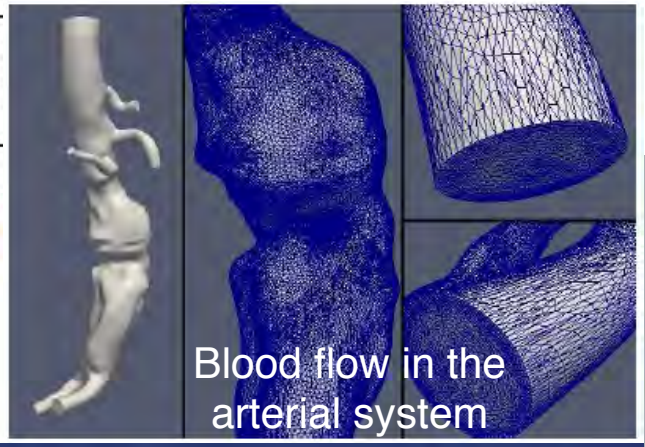


Fields in a particle accelerator

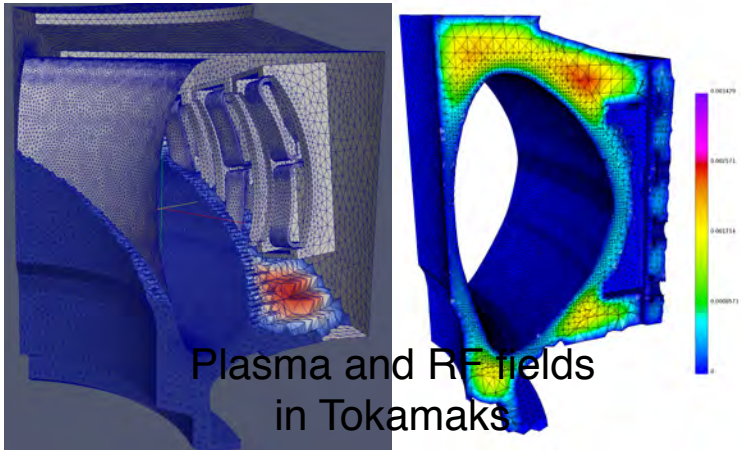

Application of active flow control to aircraft tails

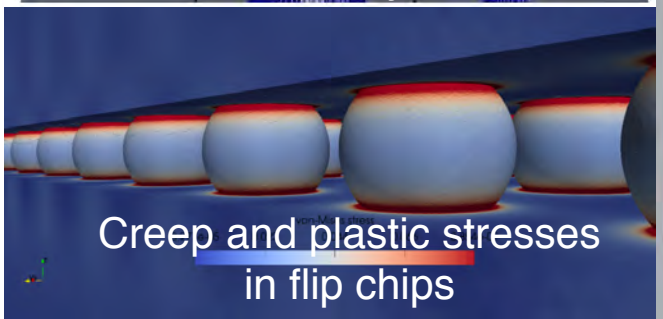


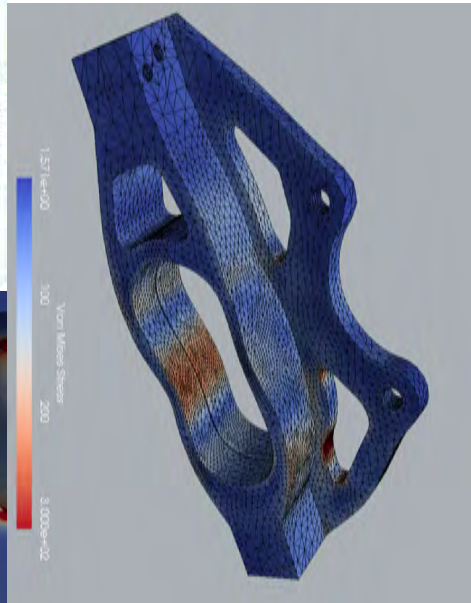

Modeling a dam break


Blood flow in the arterial system

Plastic deformation of a mechanical part


Plasma and RF fields in Tokamaks


Creep and plastic stresses in flip chips



Dynamic Load Balancing for Adaptive Workflows

At scale found graph and geometric based methods either consume too much memory and fail, or produce low quality partitions

Original partition improvement work focused on using mesh adjacencies directly to account for multiple criteria to

- ParMA partition improvement procedures that used diffusive methods
- Used in combination with various global geometric and local graph methods to quickly improve the partitions
- Account for dof on any mesh entity (balance multiple entity types)
- Produced better partitions (solved faster) using less time to balance

Goal of current EnGPar developments is generalization

- Take advantage of big graph advances and new hardware
- Broaden the areas of application to new applications (mesh based and others)

Partitioning to 1M Parts

Multiple tools needed to maintain partition quality at scale

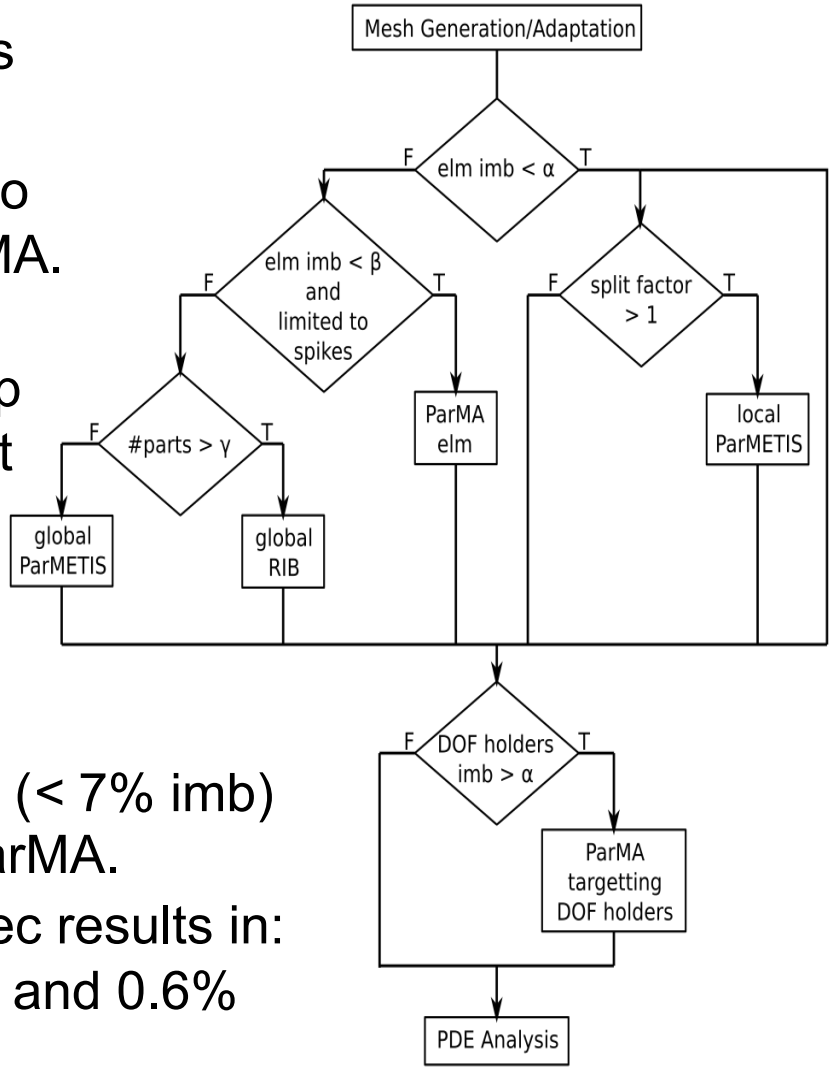
- Local and global topological and geometric methods
- ParMA quickly reduces large imbalances and improves part shape

Partitioning 1.6B element mesh from 128K to 1M parts (1.5k elms/part) then running ParMA.

- Global RIB - 103 sec, ParMA - 20 sec: 209% vtx imb reduced to 6%, elm imb up to 4%, 5.5% reduction in avg vtx per part
- Local ParMETIS - 9.0 sec, ParMA - 9.4 sec results in: 63% vtx imb reduced to 5%, 12% elm imb reduced to 4%, and 2% reduction in avg vtx per part

Partitioning 12.9B element mesh from 128K (< 7% imb) to 1Mi parts (12k elms/part) then running ParMA.

- Local ParMETIS - 60 sec, ParMA - 36 sec results in: 35% vtx imb to 5%, 11% elm imb to 5%, and 0.6% reduction in avg vtx per part



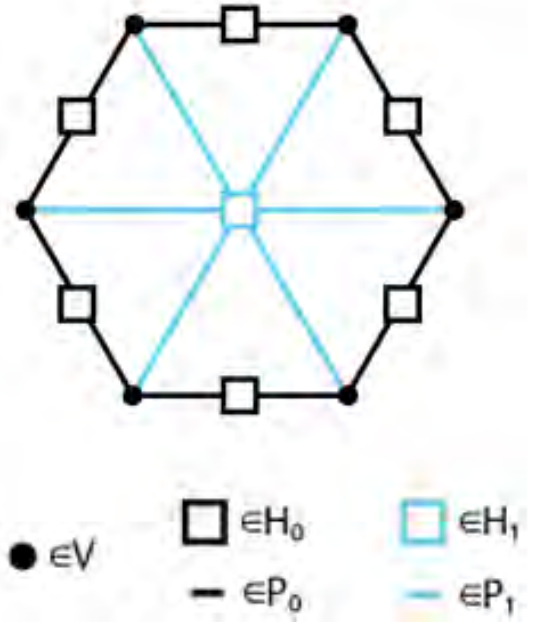
EnGPar: Diffusive Graph Partitioning

Employ an N-graph in the development of EnGPar

- Capable of reflecting multiple criteria which was the ParMA's advantage for conforming meshes
- Goal remains to supplement other partitioners to efficiently produce a superior partition of the parallel work

The N-graph, when considering multiple criteria, is:

- A set of vertices *V* representing atomic units of work.
- *N* sets of hyperedges, $H_0,\ldots,H_{n-1}$, for each relation type
- *N* sets of pins, $P_0,\ldots,P_{n-1}$, for each set of hyperedges
- Each pin in P_i connects a vertex, *v* in *V*, to a hyperedge, *h* in H_i

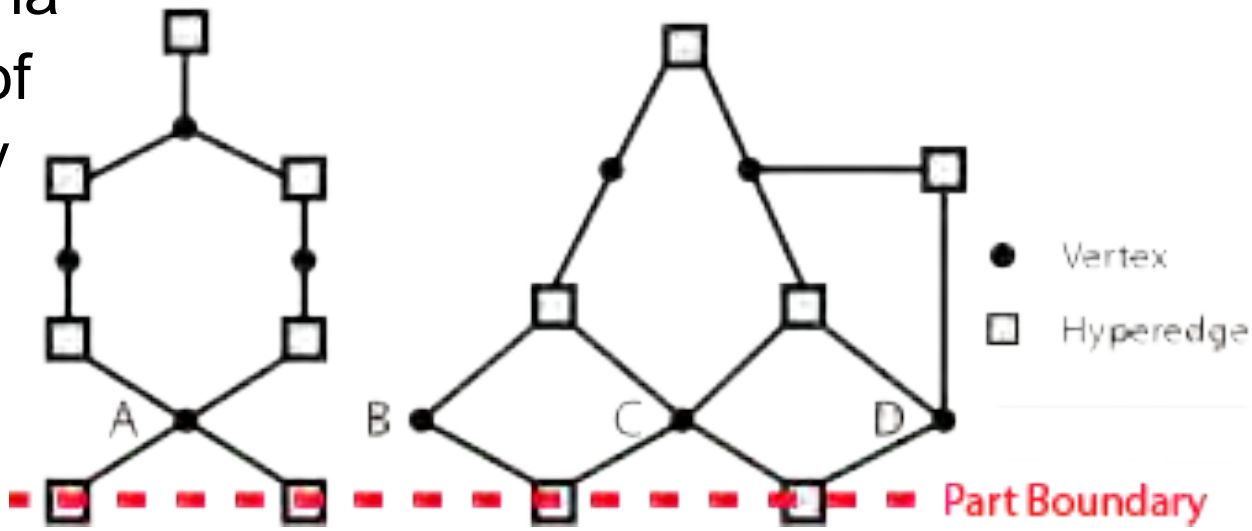


An N-graph with 2 relation types

EnGPar: Diffusive Graph Partitioning

To provide fast partition refinement

- Local decisions are made sending weights across part boundaries.
- Weight is sent from heavily loaded parts to neighbors with less weight
- Vertices on the part boundary (A,B,C,D) are selected in order to:
  - Reduce the imbalance of the target criteria
  - Limit the growth of the part boundary

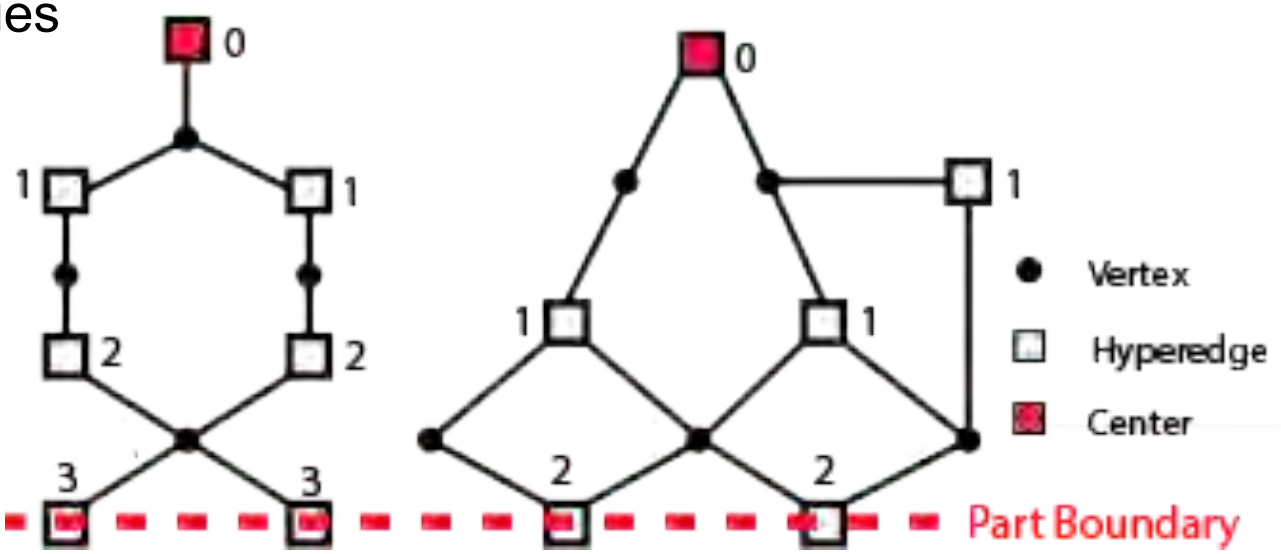


EnGPar: Diffusive Graph Partitioning

Order of migration controlled by graph distance calculations

Two steps to determine “Distance from Center”

- Breadth-first traversal seeded by the edges crossing the part boundary.
  - Determines the edges connected to part center (in red)
- Breadth-first traversal seeded by edges at the center of the part
  - Calculates distance of boundary edges from the center



Edges at part boundaries operated on to drive migration:

- First deal with disconnected and shallow components
- Then focus on edges with greater distance from the center

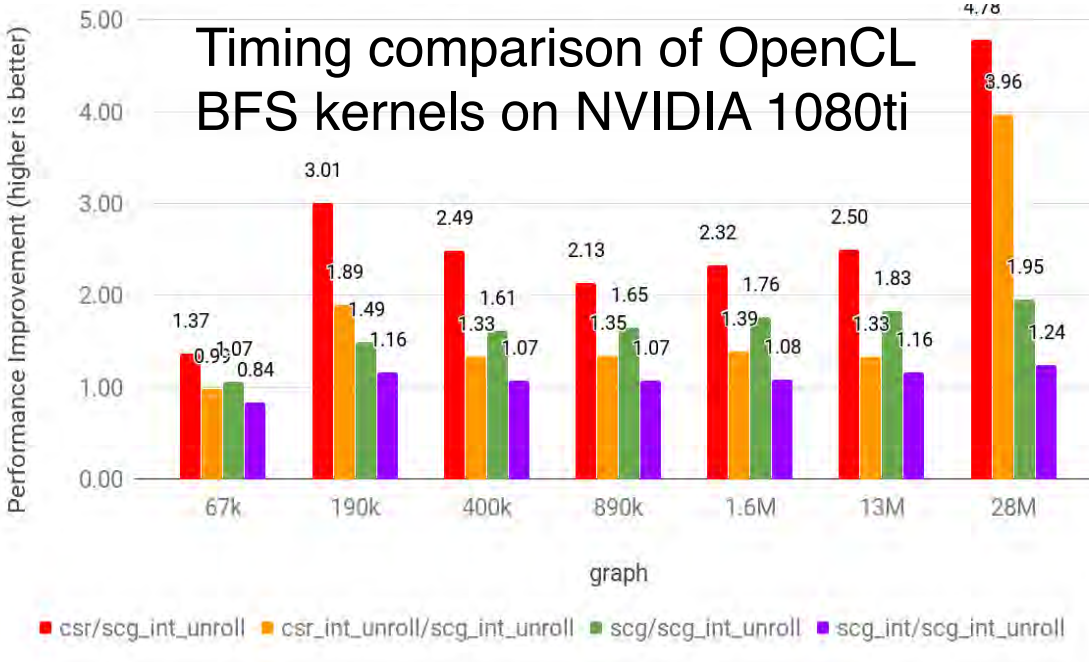
This ordering results in removing disconnected components faster and creating smaller part boundaries (less communication)

Toward Accelerator Supported Systems

EnGPar based on more standard graph operations than ParMA

- Take advantage of GPU based breath first traversals

scg_int_unroll is 5 times faster than csr on 28M graph and up to 11 times faster than serial push on Intel Xeon (not shown).



Continuing developments:

- Different algorithms and known techniques (unrolling loops, smaller data sizes)
- Different memory layouts (CSR, Sell-C-Sigma) Support migration – host communicates, device rebuilds (hyper)graph.
- Accelerate other diffusive procedures using data parallel kernels.
- Focus on pipelined kernel implementations for FPGAs.

EnGPar for Conforming Meshes

Applications using unstructured meshes exhibit several partitioning problems

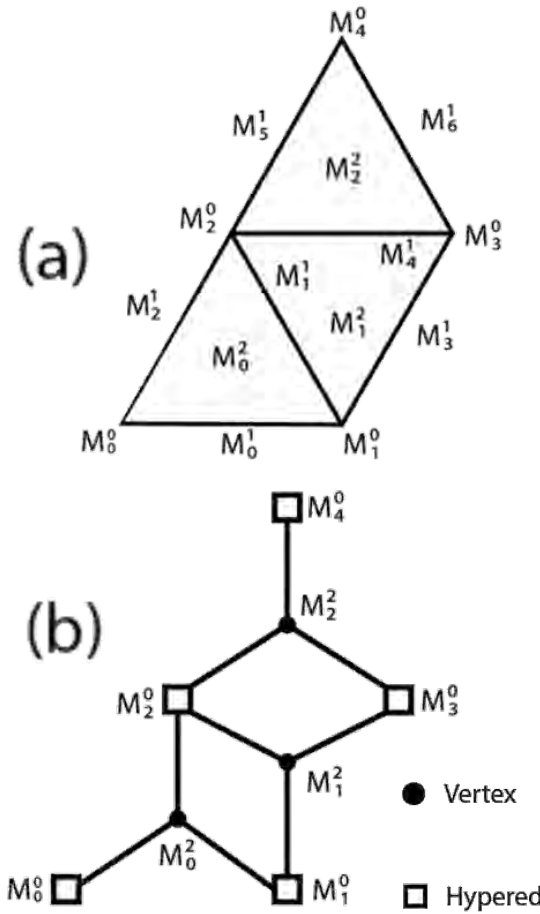
- Multiple entity dimensions important
- Complex communication patterns

To achieve the best performance require:

- Mesh entities holding dofs to be balanced
- Mesh elements to be balanced

N-graph construction includes

- Elements represented by graph vertices
- Mesh entities holding dofs represented by hyperedges
- Pins between graph vertex to hyperedge where the mesh element is bounded by the mesh entity



Mesh adjacencies (a) to N-graph (b)

EnGPar for Conforming FE Meshes

Tests run on billion element mesh

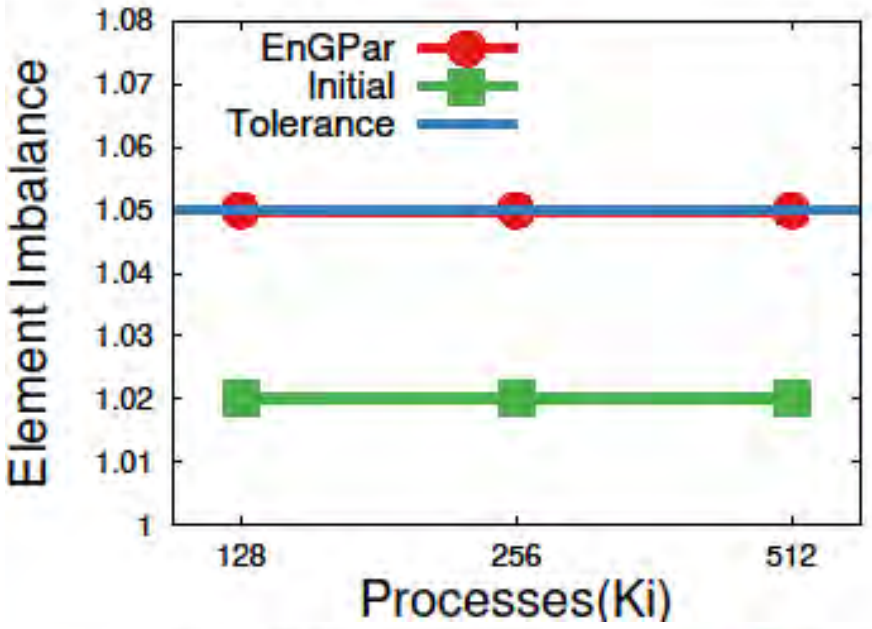
- Global ParMETIS part k-way to 8Ki
- Local ParMETIS part k-way from 8Ki to 128Ki, 256Ki, and 512Ki parts

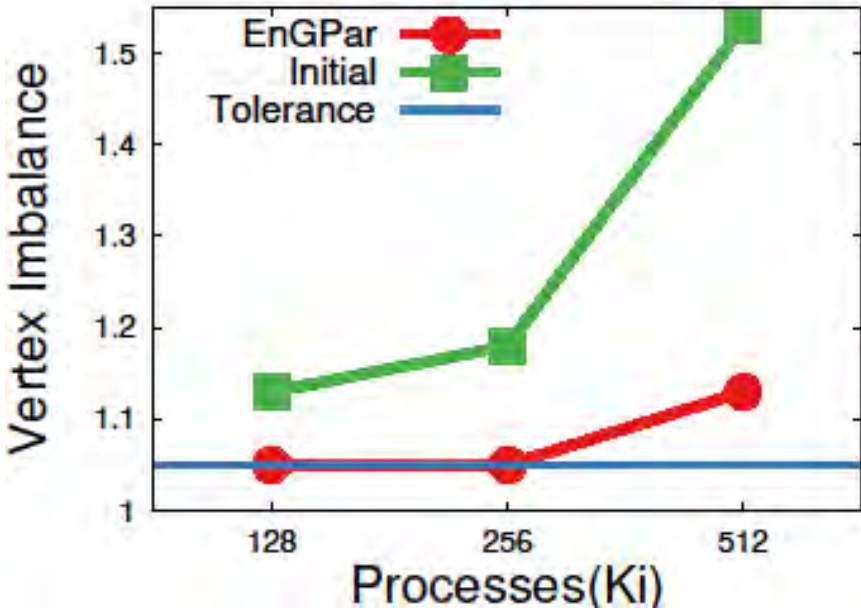
Resulting imbalances after running EnGPar are in the following figures

Accounting for multiple entities

- Creating the 512Ki partition from 8Ki parts takes 147 seconds with local ParMETIS (including migration)
- EnGPar reduces a 53% vertex imbalance to 13% in 7 seconds on 512Ki processes.

Results close to ParMA what was specific to this application





Mesh-Based Apps Suited to EnGPar (but not ParMA)

Overset grids

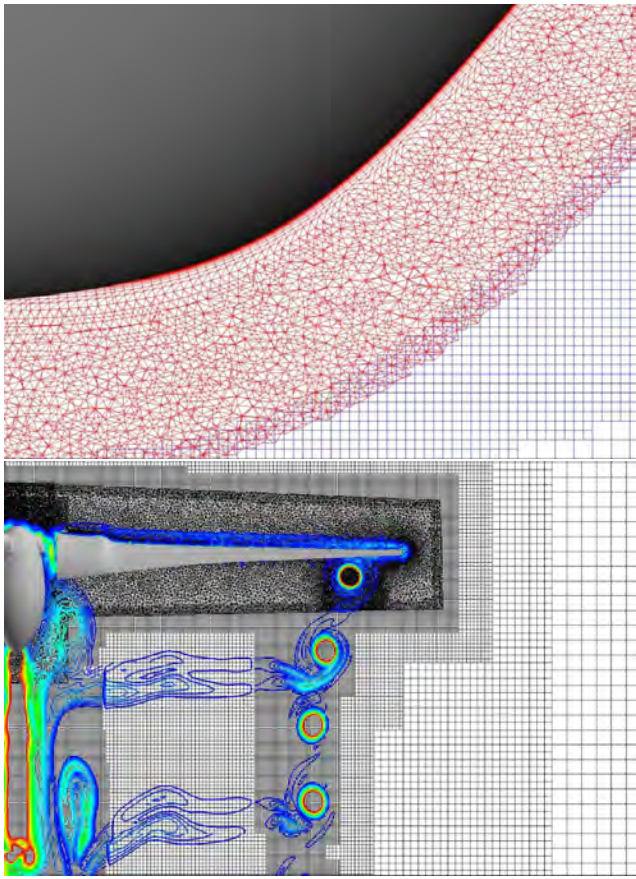
- Coupling between meshes
- More communication/part boundaries
- The N-graph construction includes:
  - Element for both meshes as vertices
  - Hyperedges for all dof holders
  - Hyperedges for overlap coupling

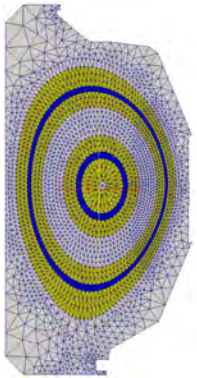
Non-conforming adaptive FV grids

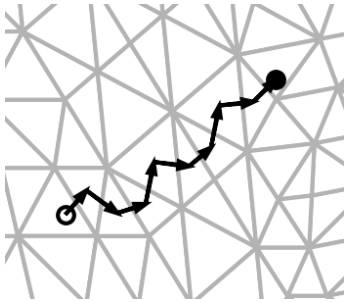
- Grid vertices as graph vertices
- Ghost layer related considerations
- Neighboring edges define edges

Unstructured mesh particle in cell for fusion

- Element define weights
- Partition must account for field following
- Particle drift slow – well suited for diffusive







EnGPar for Conforming FV Meshes

FV application (FUN3D)

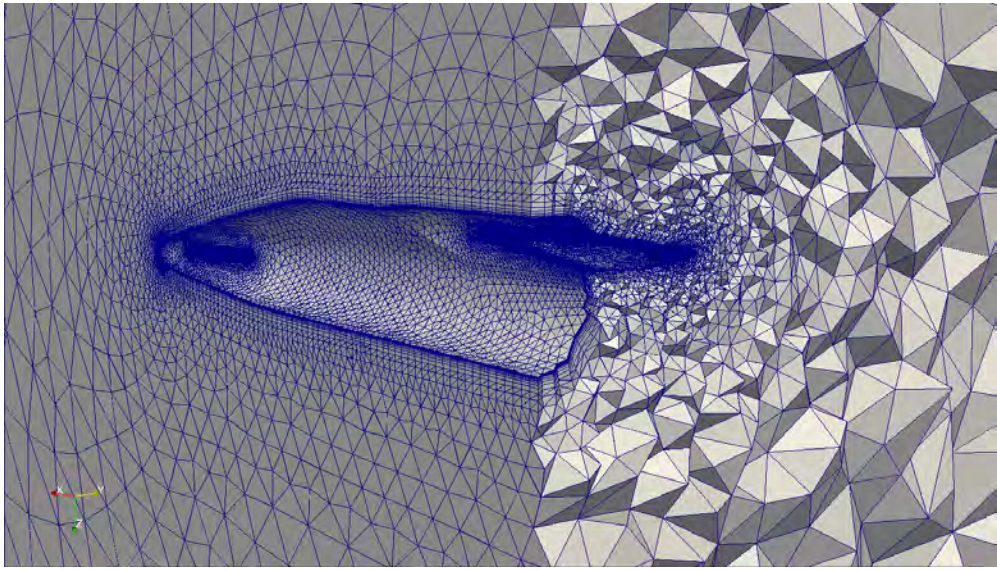
- Vertex partitioning to balance multiple entity types

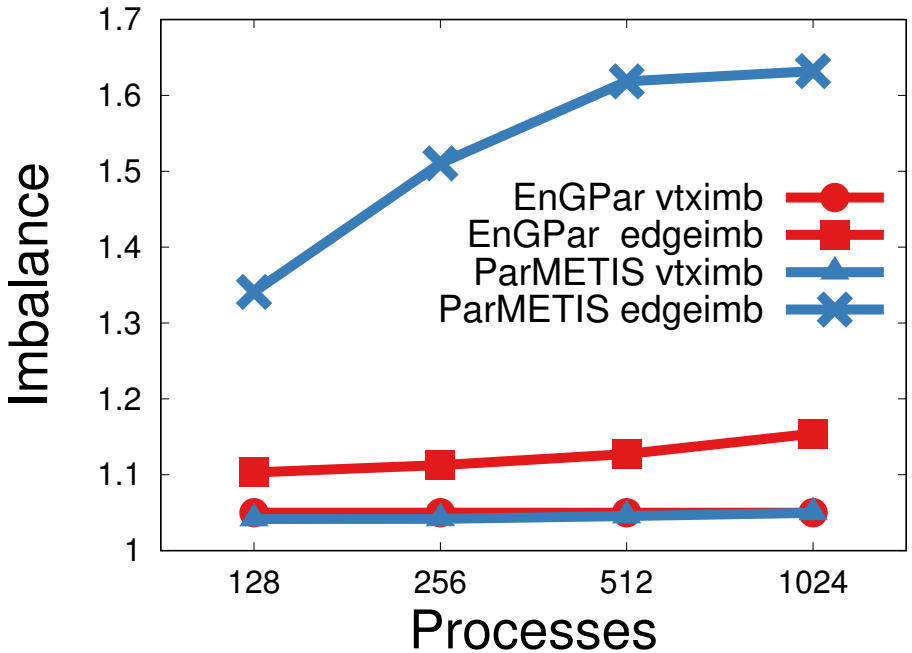
N-Graph construction includes:

- Graph vertices and mesh vertices
- Hyper edges for each element
- Pins between elements/vertices
- Ghosts vertices receive weights

3.6 million element mesh

- Partitioned to 1024 w ParMETIS
- EnGPar with vertex tolerance of 5% and edge at 10%
- Controlled growth on inter-part interfaces





Different Application: Discrete Event Simulation

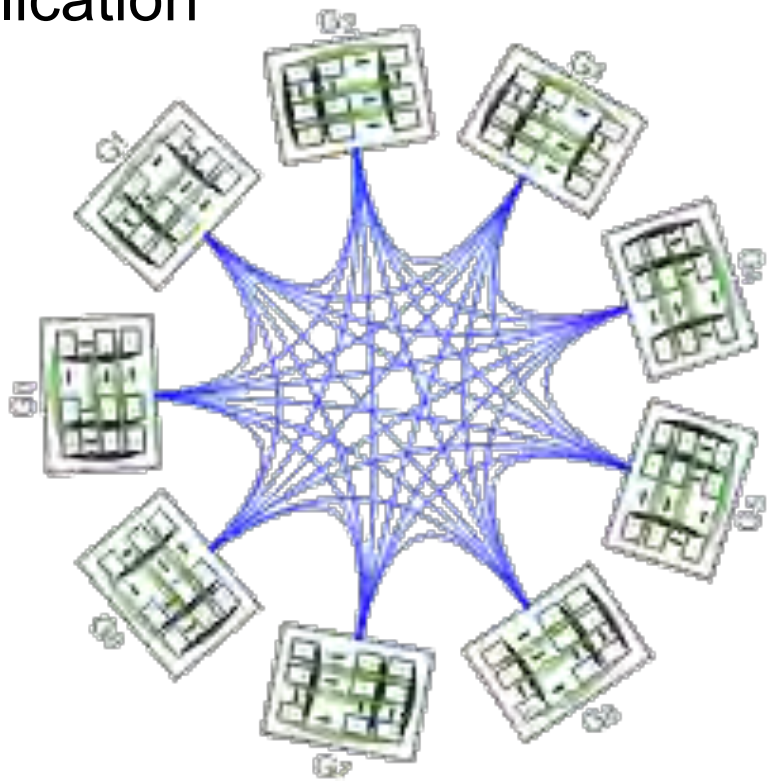
CODES simulates running an MPI application on a simulated hardware architecture.

The main CODES units are logical processes (LPs) which represent

- The hardware components
- The simulated MPI processes

The N-graph construction includes:

- Graph vertices for each LP
- Graph edges between LPs that have an event between them.



A dragonfly network

Closing Remarks

- The RPI SCOREC team has developed a number of parallel unstructured mesh tools used by DOE and DoD
- Tools can contribute to “A National Software Ecosystem”
- Dynamic load balancing work is an important component
- Current efforts are focused on:
  - Employing N-graph to meet the needs of multiple applications
  - Effective execution on new generation systems

Acknowledgements

- National Science Foundation Grant ACI 1533581
- DOE FastMath SciDAC Institute
- CEED ECP Co-Design Center
- DoD PETTT program