

Lawrence Livermore National Laboratory

Challenges to Effective Partitioning on BG/L (a work in progress)



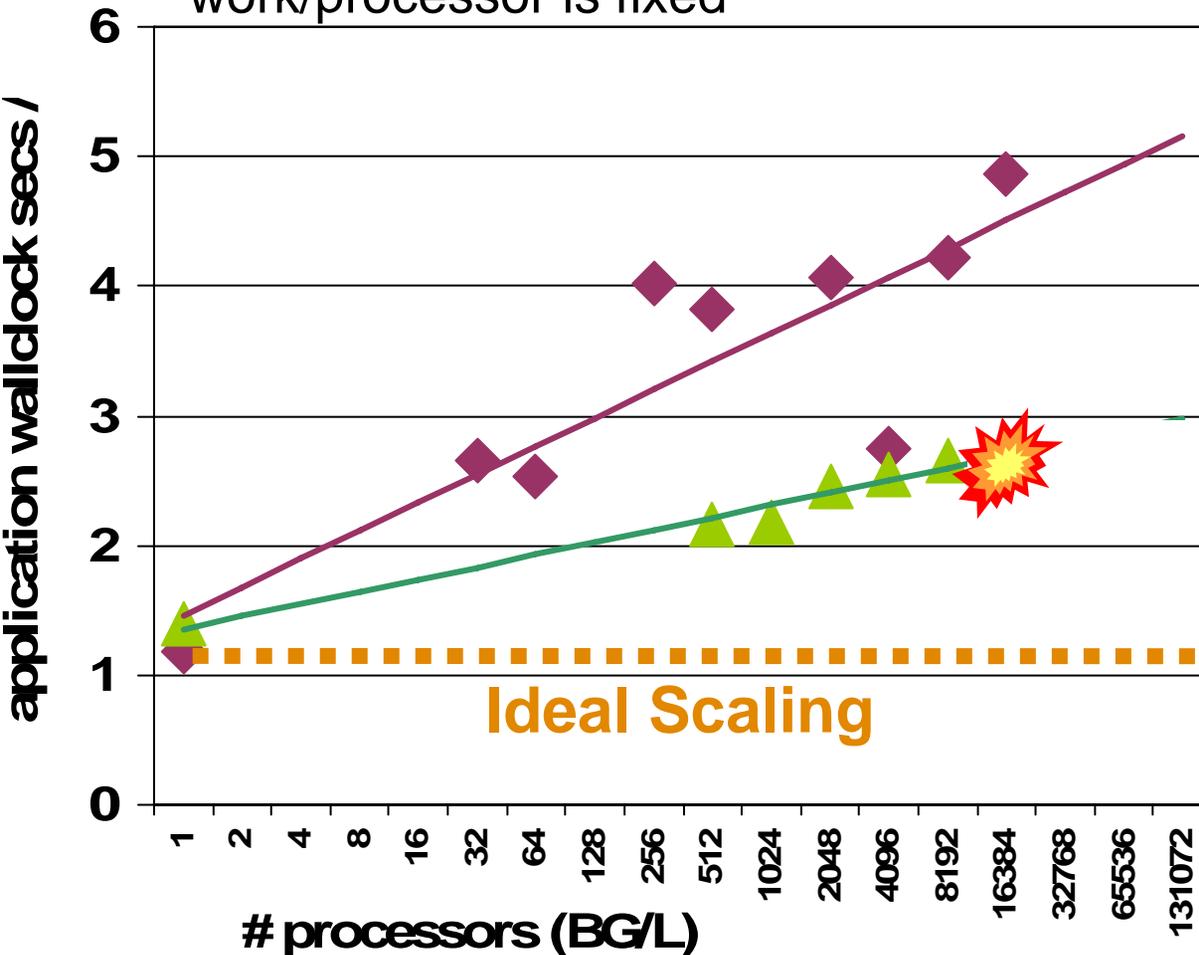
Gary Kumfert
Katie Lewis, Jim Reus

Lawrence Livermore National Laboratory, P. O. Box 808, Livermore, CA 94551
This work performed under the auspices of the U.S. Department of Energy by
Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344

LLNL-PRES-405052

Problem #1: Quality load balancers aren't scaling to full BG/L

“Weak Scaling” Study:
work/processor is fixed



- **Block Partitioner:** balances computation not communication
- **Multilevel Graph Partitioner: (ParMETIS)** balances computation and reduces communication
- **Problem:** Empirically not scaling.



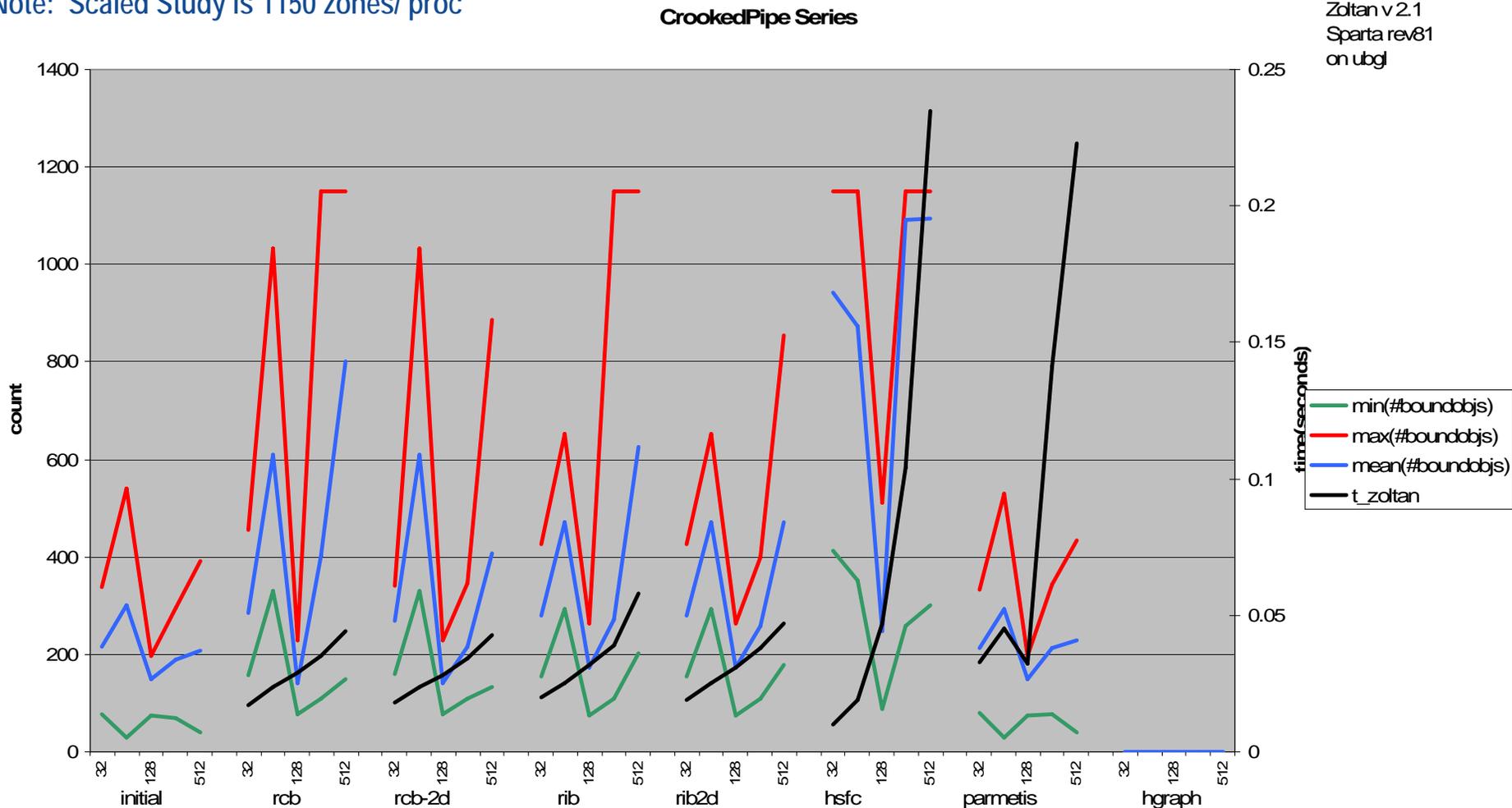
Problem #2: Production users never thrilled with ParMetis Anyway

- Pros
 - Been reliable “black box” for decade
- Cons
 - “My partitioned mesh looks like a map of the US”
 - “Slowest processor is always the domain with most neighbors”
 - “Doesn’t guarantee {connected components, smooth cuts, good aspect ratios} etc.”
- Translation
 - Edge cut is not their figure of merit
 - Little agreement what the figure of merit should be



Started "Feasibility Study" with Zoltan to explore alternative off-the-shelf algorithms

Note: Scaled Study is 1150 zones/ proc



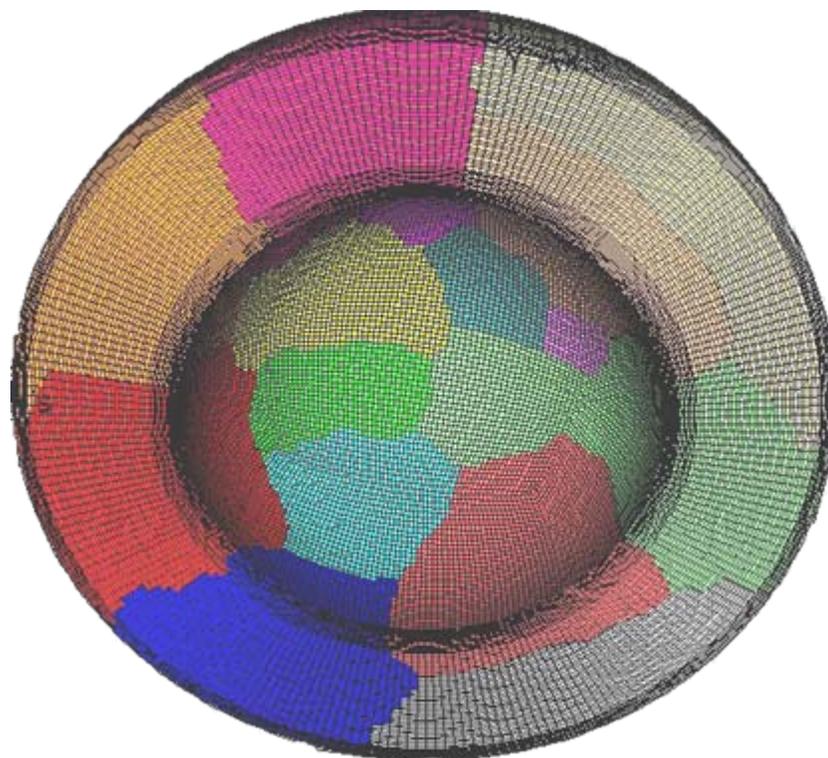
Lessons learned from feasibility study

- Customer gave us problems already partitioned by ParMetis
- Zoltan's multilevel graph partitioner is ParMetis
- Zoltan's hypergraph partitioner didn't work on BG/L
 - Fixed in Zoltan version 3.0
 - Hypergraph partitioner uses more memory than traditional multilevel graph partitioner
- Geometric algorithms generated partitions with at least one domain having no internal zones. (not good).
- Jostle's closed source model is non-starter.



Sparta is a new LDRD developing scalable partitioning algorithms

- SPARTA = Scalable PARTitioning Algorithms
- new = 7 months (of research funding)
- LDRD = Lab Directed R&D
- Funding is targeted for LLNL codes, not partitioning in general

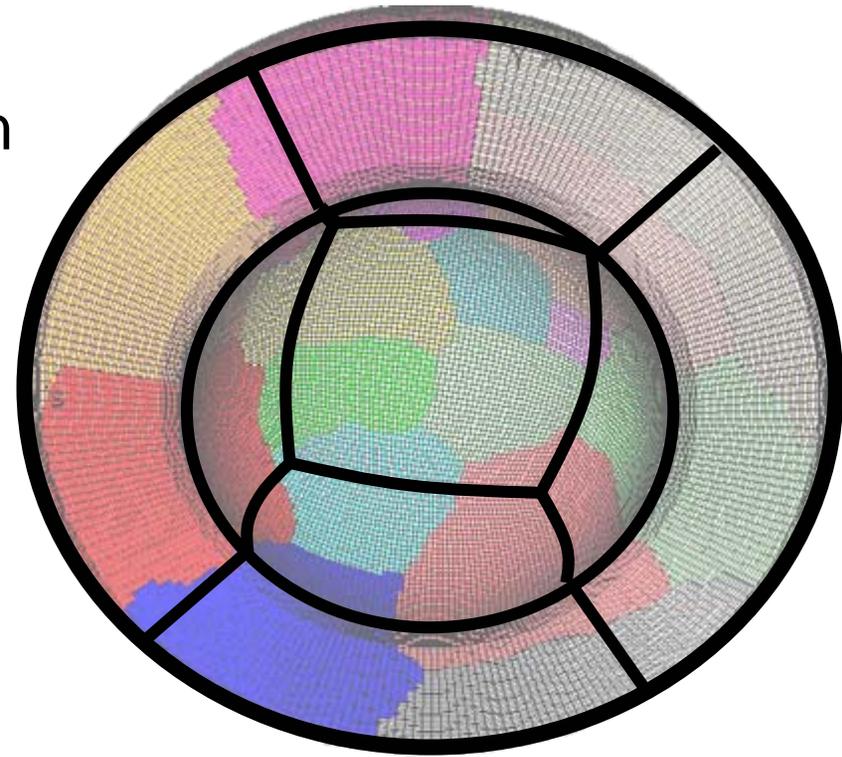


16-way decomposition of a hollow hemisphere, courtesy Jeff Keasler



Insight: Exploit structural artifacts of mesh generation that unstructured multi-physics codes discard

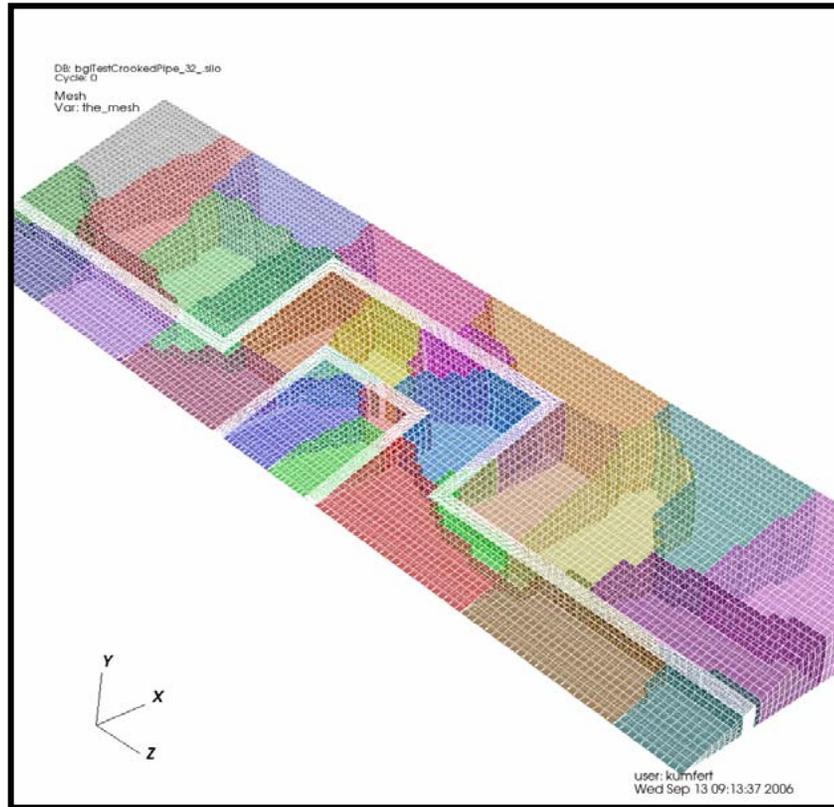
- Generating 3-D Hex Meshes On Rectangular Blocks is Completely Automatic
- Generating 3-D Hex Meshes On Complex Geometries Requires a Human Designer
 - Human manually fits blocks to complex geometries
 - Blocks can be stitched together irregularly
 - Internally, blocks are completely regular
- Many applications treat multiblock meshes as if they were completely unstructured



Same hollow hemisphere decomposed into 16 domains. Emphasis added to its five constituent logically rectangular blocks



Focus on blocks (not zones) reduces memory & computational complexity AND creates opportunity



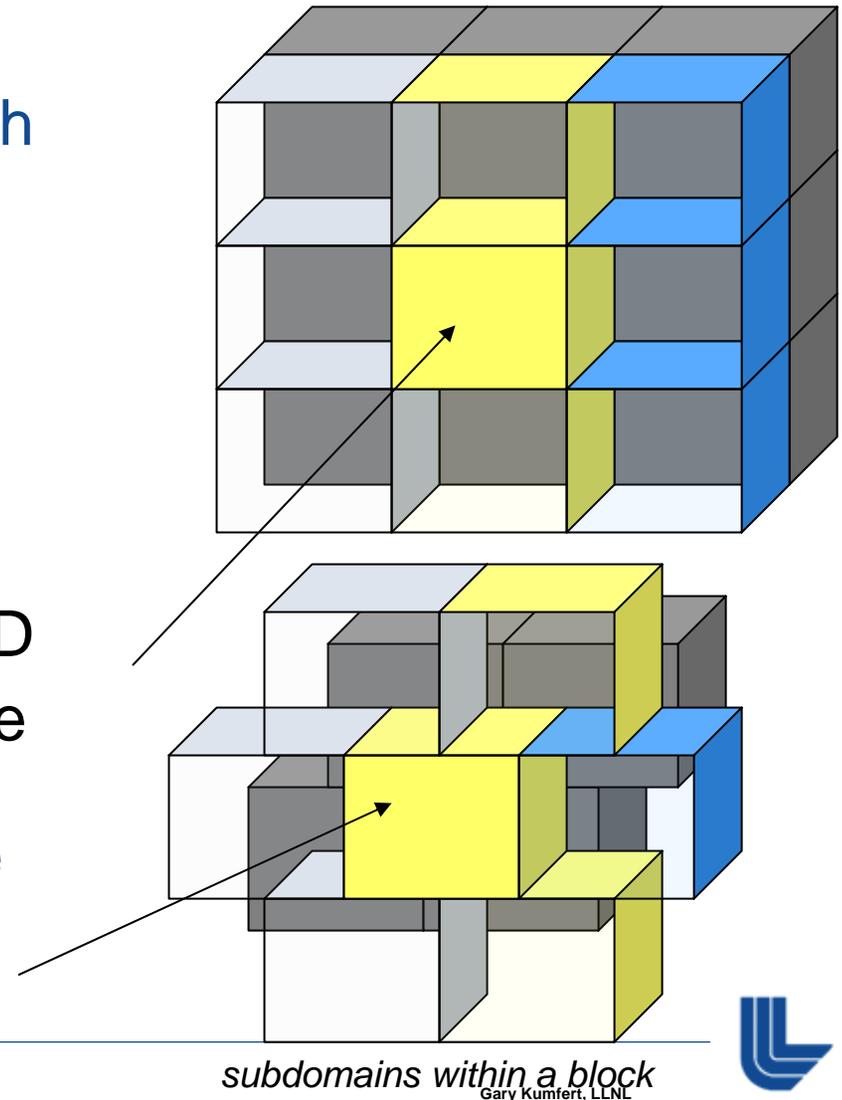
“Crooked Pipe” example composed of 26 blocks. 1150 zones/processor is “sweet spot” for this application. Therefore, this instance generated for 32 processors has 36,800 zones. Courtesy, Brian McCandless

- Can now develop efficient algorithms to:
 - Bound # hops for all neighbor communications on BG/L’s Torus network
 - Control “Smoothness” of the boundaries
 - Control aspect ratio of the subdomain
 - Reduce Variation in Communication Volume
 - Bound # neighboring domains

By bounding number of neighbors we expect a significant performance boost

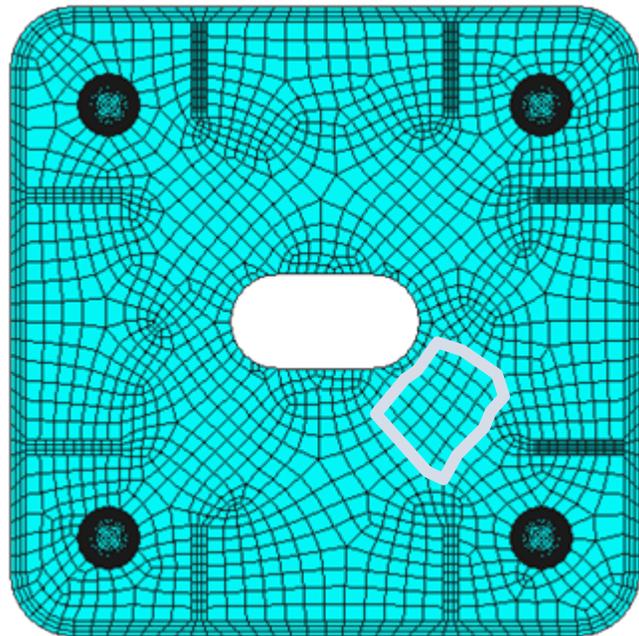
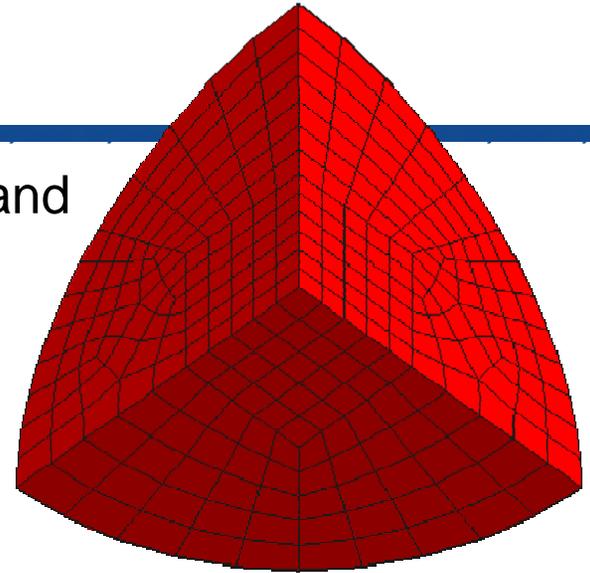
Year #1 Goal:

- Observation: “The slowest processor is usually the one with the most neighboring domains”
Keasler & McCandless
- Intuition: Fewer neighbors
→ fewer interprocess communications
→ faster physics
- 26 neighbors are expected in 3D
- With rectangular structure inside each block (and some tiling) we should be able to **guarantee** no more than 14 neighbors



Added wrinkle #1: Some codes have semi-structured regions.

- Example of structured and unstructured regions.



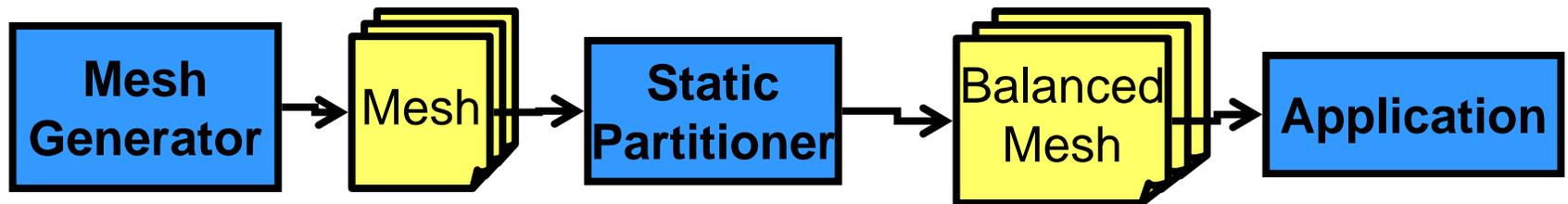
- 2-D quad meshing “paving” is fully automatic
- Can be “swept” in a structured way in the 3rd dimension
- Even then, there are accidentally regular regions

examples, courtesy Katherine Lewis

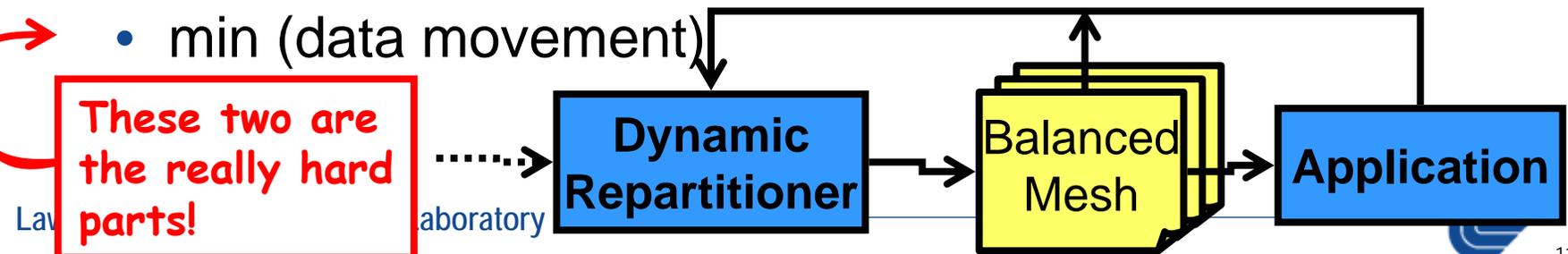


Added wrinkle #2: Customers are moving to dynamic repartitioning, but aren't there yet

- **Static Partitioning:** one-time process for application
 - max (load balance)
 - min (communication overhead)

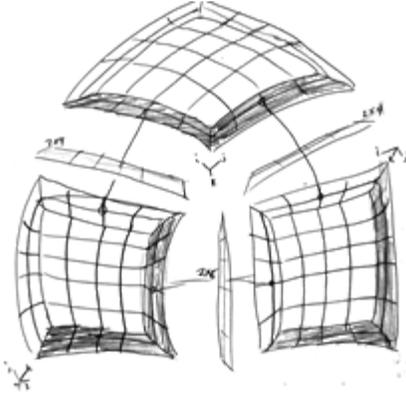


- **Dynamic Repartitioning:** Load balance is periodically adjusted by application
 - same constraints as static, plus...
 - min (calculation time)
 - min (data movement)

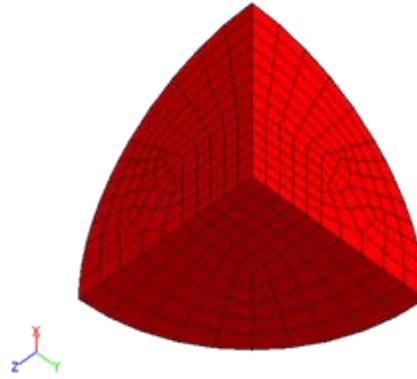


Significant coding ahead: Approach employs ALL THREE data representations for better memory scaling

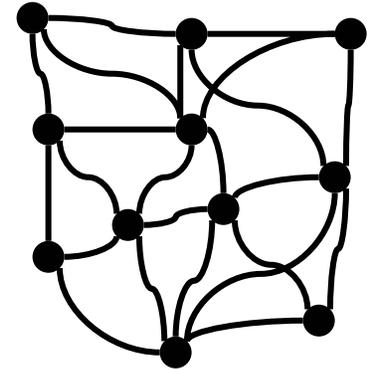
MultiBlock



General FE Mesh



Graph



- **Blocks, Interfaces, & discretization levels**
- **Typical for Mesh Generation & AMR**

- **Lists of nodes, zones, shells and beams**
- **Typical for Main Simulation**

- **Nodes & Edges**
- **Typical for Partitioners (e.g. ParMetis & Zoltan)**



Possible collaboration modes bwn Sparta & CSCAPES

- Need:
 - Partitioners with small memory footprint
 - New figures of merit for “good partitioning”
 - Identify maximal regular submeshes of an unstructured hex mesh
- Offer:
 - Have access to 100K processors
 - Test sets
 - Possible opening for summer intern
 - ?

