Dynamic Partitioning Using Mesh Adjacencies

FASTMath Team Members: *Cameron W. Smith* [1], Michel Rasquin [2], Dan A. Ibanez [1], Gerrett Diamond [1], Kenneth E. Jansen [2], and Mark S. Shephard [1]

[1]Rensselaer Polytechnic Institute, USA [2]University of Colorado Boulder, USA

Parallel unstructured mesh-based applications running on the latest petascale systems require partitions optimizing specific balance metrics. Methods combining the most powerful graph based and geometric methods with diffusive methods directly operating on the unstructured mesh are discussed. Partitions with over one million parts for meshes of several billion elements were generated on ALCF's Mira Blue Gene/Q.

Dynamic Partitioning of Unstructured Meshes

Tools for re-partitioning an unstructured mesh due to changing work loads or communication patterns are required to [1]:

- Balance work, reduce communications, output distribution, execute in parallel quickly, use little memory, and provide API

Graph and hypergraph based partitioners

- Produce balanced partitions with low cuts but have limited scalability
- Use one order of mesh entity as the graph nodes, hence the balance of other mesh entities may not be optimal

Geometric partitioners

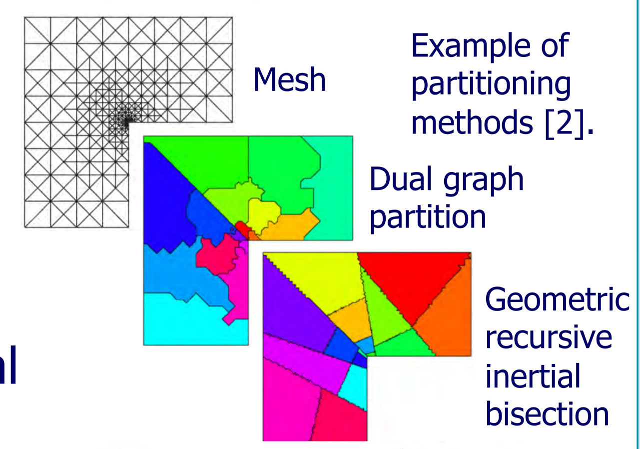
- Inexpensive and scalable vs (hyper)graph at cost of larger cuts

Diffusive partitioners

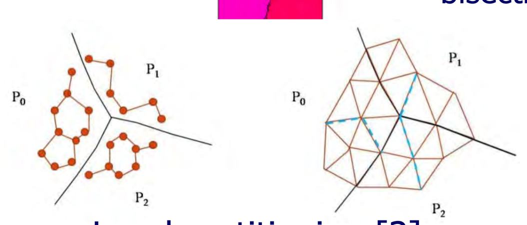
- Quickly reduce small imbalances

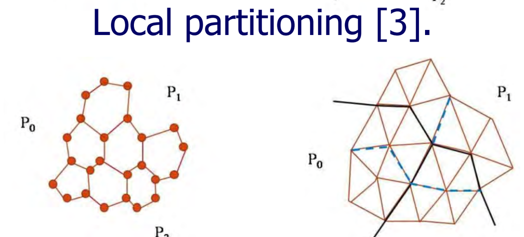
Local partitioners

- Consider intra-process relations only



Example of partitioning methods [2].


Local partitioning [3].


Global partitioning [3].

3 - "Controlling Unstructured Mesh Partitions for Massively Parallel Simulations", Min Zhou, et al., 2010

ParMA: Partitioning using Mesh Adjacencies

Guide partitioning decisions with mesh adjacency information

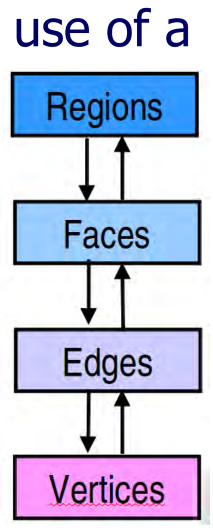
- Mesh and partition model adjacencies represent application data more completely then standard (hyper)graph-partitioning models.
- All mesh entities can be considered, while graph-partitioning models use only a subset of mesh adjacency information.
- Any adjacency can be obtained in O(1) time with the use of a complete mesh adjacency structure.

Advantages

- Avoid graph construction
- Directly account for multiple entity types important for the solve process - typically the most computationally expensive step
- Easy to use with diffusive procedures

Disadvantage

- Lack of well developed algorithms for global parallel partitioning operations directly from mesh adjacencies



Complete mesh adjacency structure.

Diffusive Improvement

Approach

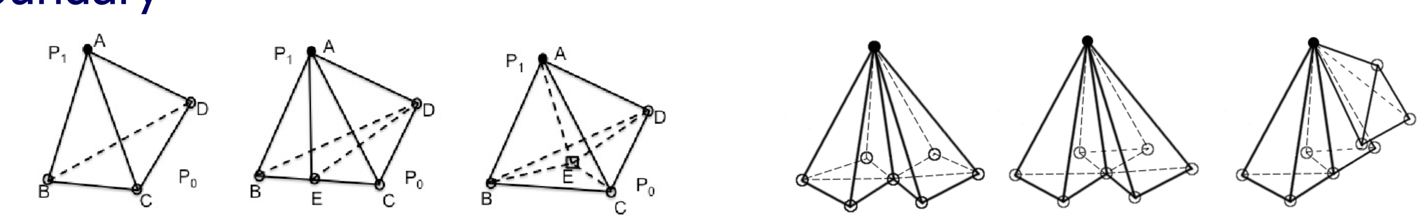
- Iteratively migrate small sets of elements from imbalanced parts to less imbalanced parts to reduce the peak imbalance.
  - Stop when improvements to the imbalance and cut are small
- Select elements for migration that will reduce the imbalance and reduce the number of mesh entities on the part boundaries.

Iteration Stages

- Weight computation – compute weights and exchange with peers
- Targeting – determined how much weight each peer can accept
- Element selection – select elements for migration
- Migration – move elements to peers

Element Selection

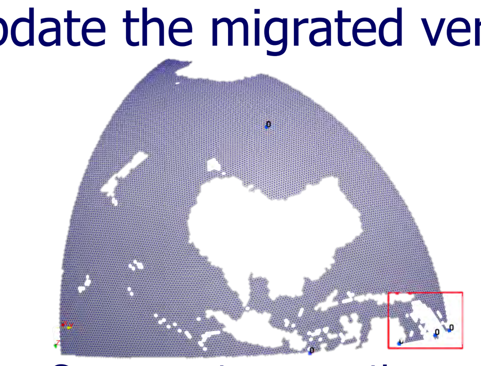
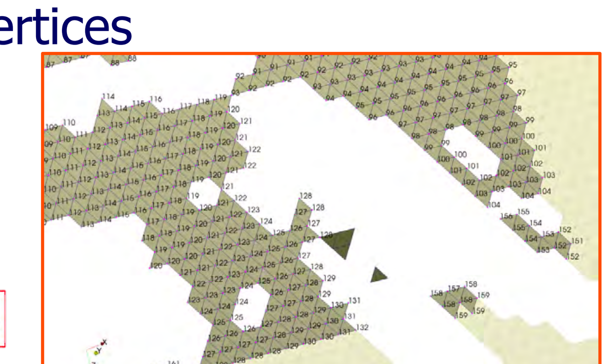
Selects small groups of elements bounded by a vertex on the part boundary


Vertex bounded elements selected for migration. A circle marks vertices on the part boundary, a square marks interior vertices, and a disc marks the element bounding vertex.

Evaluate vertices in descending order of distance from the parts topological center

- The elements in a part are not necessarily connected – sets of elements may not be reachable from other sets via adjacencies
- Connected components are identified and sorted in descending order of their depth – as determined by a breadth-first traversal from its boundary vertices
- Dijkstra's algorithm is ran from one of the max depth vertices of each component to determine the graph distance to each vertex
- Distances are offset to avoid overlapping ranges
- During the first iteration distances are computed – subsequent iterations simply update the migrated vertices

Connected components in one of four parts of the MPAS 60km ocean mesh. Dark shaded elements are isolated and light shaded elements are on a different part.



Component core vertices.

Partitioning to One Million Parts

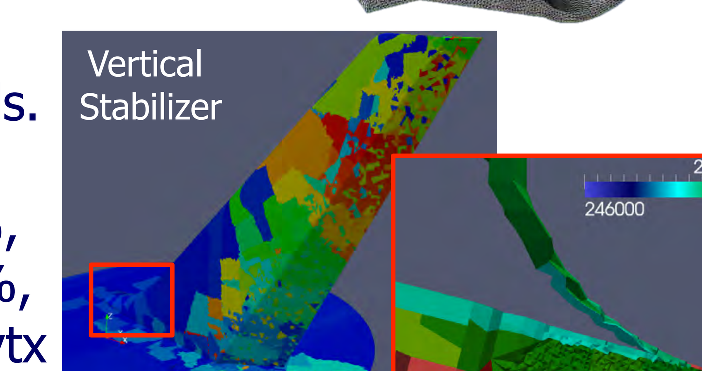
Multiple tools needed to maintain partition quality at scale

- Local and global topological and geometric methods

ParMA quickly reduces large imbalances and improves part shape

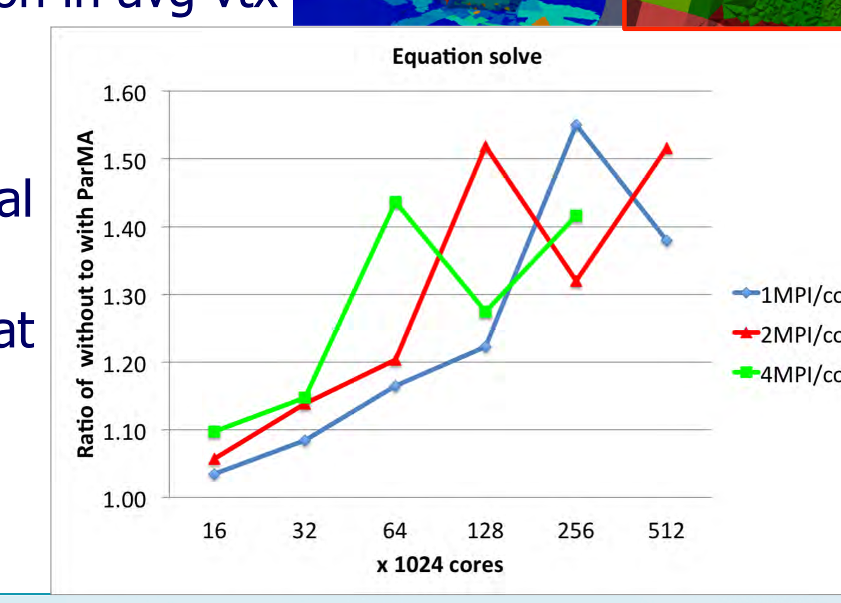
- Partitioning 1.6B element mesh from 128K to 1M parts then running ParMA. 128K partition has less than 7% imbalance for all entity orders.
  - Global RIB – 103 seconds ParMA – 20 seconds to: 209% vtx imb reduced to 6%, perfect elm imb increased to 4%, and 5.5% reduction in avg vtx per part
  - Local ParMETIS – 9.0 seconds. ParMA – 9.4 seconds to: 63% vtx imb reduced to 5%, 12% elm imb reduced to 4%, and 2% reduction in avg vtx
- Partitioning 12.9B element mesh from 128K (< 7% imb) to 1M parts then running ParMA.
  - Local ParMETIS – 60 seconds. ParMA – 36 seconds to: 35% vtx imb reduced to 5%, 11% elm imb reduced to 5%, and 0.6% reduction in avg vtx

Coarse mesh of RPI Formula Hybrid 2014 suspension upright.



ParMA improves strong scaling of PHASTA

- 1.2B elements, vertical stabilizer geometry
- >50% improvement at 128K and 256K cores
- 35% improvement at 1M cores



Closing Remarks

ParMA diffusive improvement combined with local and global graph and geometric partitioners provides a scalable partitioning solution for meshes with over one million parts and several billion elements.

Ongoing efforts

- Controlling partition model topology – elimination of small part boundaries, gradient diffusion.

1 - "Dynamic load balancing in computational mechanics", Bruce Hendrickson and Karen Devine, 2000

2 - "A refinement-tree based partitioning method for dynamic load balancing with adaptively refined grids", William F. Mitchell, 2007

More Information: http://www.scorec.rpi.edu/parma or contact Cameron Smith, smithc11@rpi.edu