Dynamic Partitioning of Unstructured Meshes

Tools for re-partitioning an unstructured mesh due to changing workloads or communication patterns are required to:
- Balance work, reduce communications, output distribution, execute in parallel quickly, use little memory, and provide API

Common Methods
- Graph, hypergraph, geometric, and local

Examples will demonstrate that specific combinations are needed to reach really large part counts (> 500K).

ParMA: Partitioning using Mesh Adjacencies

Guide partitioning decisions with mesh adjacency information
- Mesh and partition model adjacencies represent application data more completely than standard (hyper)graph-partitioning models.
- All mesh entities can be considered; (hyper)graph uses a subset of the adjacency information.
- Complete mesh - O(1) time adjacency queries.

Advantages
- Avoid graph construction and directly account for multiple entity types important for the solve process
- Easy to use with diffusive procedures - compliments other partition methods via incremental refinements

Diffusive Approach
- Iteratively migrate small sets of elements to (1) reduce the peak imbalance and (2) reduce the number of entities on the part boundaries.

Diffusive 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

Partitioning to One Million Parts

Multiple tools needed to maintain partition quality at scale [2]
- Local and global topological and geometric methods via Zoltan and Zoltan2 integration

ParMA quickly reduces large imbalances and improves part shape
- 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
- 209% reduction in avg vtx per part

Supporting Next Generation Systems

Next generation systems will have nodes with accelerators - FPGAs and GP-GPUs - and many-core processors
- Partition to multiple devices within a node
- Ex) 72 core Knights Landing, intra-processor mesh network, NUMA clustering modes
- Partitioning implemented using data-parallel programming model
- Assume that implementation language/library will change as devices are released and mature
- Some diffusive procedures are well suited to bulk operations - greedy quality metric computation and element selection
- Investigate effective utilization of additional memory level hierarchies - HBM, DRAM, processor-device shared memory

More Information: [http://www.scorec.rpi.edu/parma](http://www.scorec.rpi.edu/parma)
Contact: Cameron Smith smithc11@rpi.edu and Mark Shephard shephard@rpi.edu

Access to Software

ParMA: [https://github.com/SCOREC/core](https://github.com/SCOREC/core)
Zoltan2: [https://trilinos.org/packages/zoltan2/](https://trilinos.org/packages/zoltan2/)

Extending Application Support

Goal
- Support applications operating on graph-like structures. Three demonstrations:
  - Finite element unstructured mesh-based analysis
  - Hierarchic multi-scale for soft tissue modeling
  - Scale-free graphs for discrete event simulations

Approach
- Define N-graph abstraction to directly represent both graph and mesh relational structures:
  - defined as $G_i(V, E_i, E_{i-1})$ where:
    - $V_i$ atomic units $u_i$ of the domain $\Omega$ which uniquely exist on one part $s_t$. $\Omega = \cup u_i 
    - $E_i$ relations $e_i(u,v)$ of type $i$ between vertices $u$ and $v$ where $u,v \in V$ and $u \neq v$
  - Optionally, vertices and edges may be assigned associated weights $w$

- Implement EnGPar - N-Graph based versions of ParMA methods and interfaces to graph and geometric methods

Hierarchical structure for soft tissue modeling. N-graph vertices and edges represent the macro-scale mesh (left) vertices and bounded regions. Vertex weights represent the micro-scale computational load.

High degree vertex splitting for discrete event network simulations. The original high degree vertex in (a) the split N-graph representation in (b), and the scatter and gather update protocol in (c) and (d) respectively.

Collaboration Development
- NASA and DoD have contacted us for partitioning support
- Interested in addressing your group's load balancing needs