Data Transfer Kit Summary

Stuart R. Slattery
Engineering Physics Department
University of Wisconsin - Madison

January 14, 2013

Outline

- Development Hisotry
- Domain Model and Geometric Rendezvous
- DTK Algorithms
- Code Examples

What is DTK?

- Collection of data mapping algorithms for shared domain probelems
- Data maps allow for efficient movement of data in parallel (e.g. between meshes of a different parallel decomposition)
- Ideally maps are generated at a desireable time complexity (logarithmic)
- Input mesh and geometry data drive the map generation
- Should be viewed as a service providing suite of concrete algorithm implementations

What DTK Doesn't Do

- Does not provide a general interface for all physics codes to couple to all other physics codes
- Does not provide discretization services (e.g. basis functions)
- Does not provide algorithm implementations for interface-based data transfer

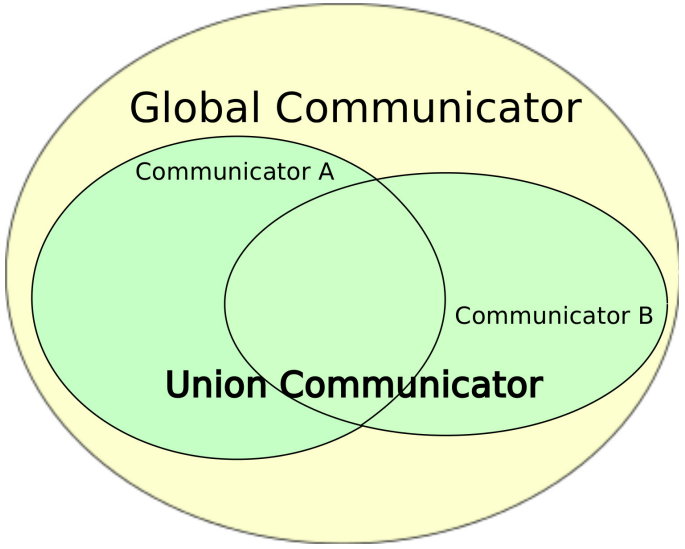
Software Overview

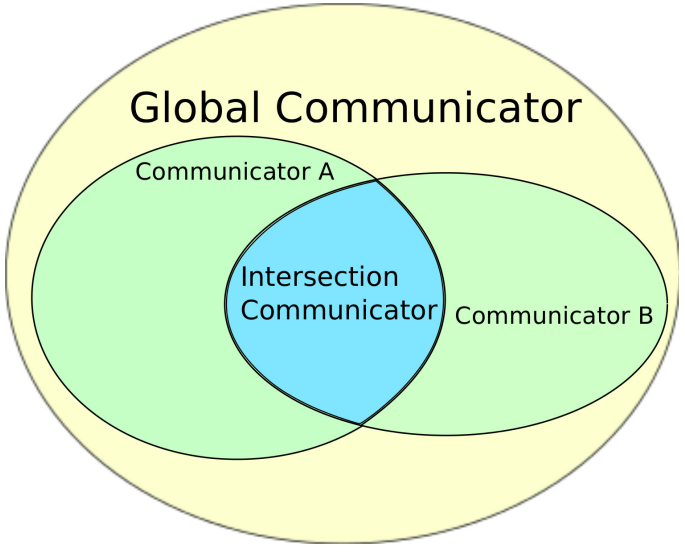
- Preliminary development of mesh-based capabilities during summer 2012 CASL internship at ORNL
- Additional development of geometry-based capabilities during fall 2012
- Implemented in C++
- Heavy use of the Trilinos scientific computing libraries
- Continuous and nightly testing as part of the CASL CDash system

Domain Model and Geometric Rendezvous

Communicators

- DTK handles source and target communicators of arbitrary relation
- Any amount of overlap or lack thereof supported
- A global communicator required (doesn't have to be MPI_COMM_WORLD)





Shared Domain Problems

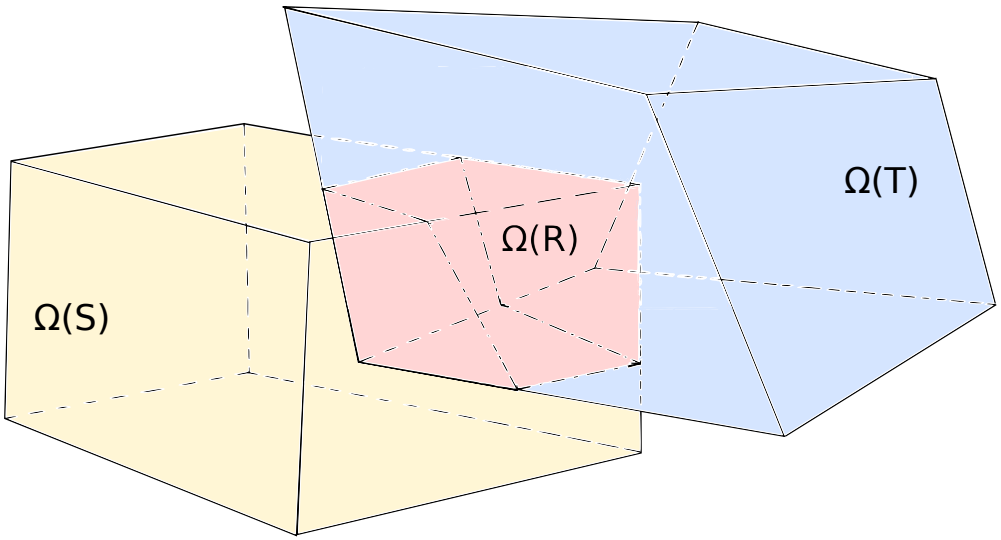


Figure: *Shared domain example.*

$\Omega(S)$ (yellow) is the source geometry, $\Omega(T)$ (blue) is the target geometry, and $\Omega(R)$ (red) is the shared domain.

- Defined over a communicator that encapsulates the union of the source and target communicators
- Source and target must be of same geometric dimension
- The rendezvous algorithm leveraged to map shared domains

The Rendezvous Algorithm

- Initially developed by the SIERRA team in the mid-2000's for parallel mesh-based data transfer [1]
- Creates a parallel topology map that can be used repeatedly for data transfer
- Map execution uses asynchronous strategy (posts and waits) with minimal messages
- Effectively $N * log(N)$ time complexity for parallel topology map generation
- Relies on the generation of a secondary decomposition of the source and target meshes with a geometric based paritioning (RCB)

[1] S. Plimpton, B. Hendrickson, and J. Stewart, A parallel rendezvous algorithm for interpolation between multiple grids, Journal of Parallel and Distributed Computing, vol. 64, pp. 266276, 2004

The Rendezvous Decomposition

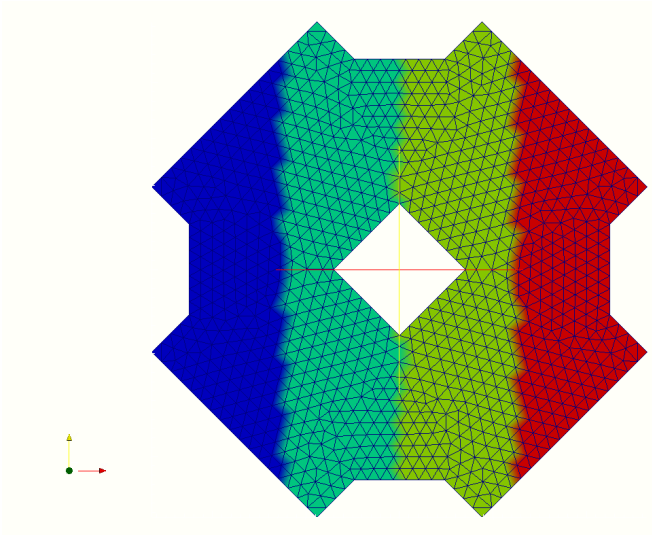


Figure: *Source mesh for 2D shared domain example.*

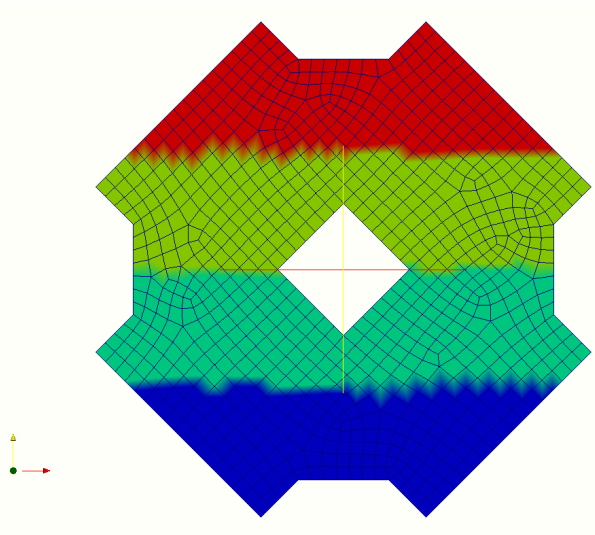


Figure: *Target mesh for 2D shared domain example.*

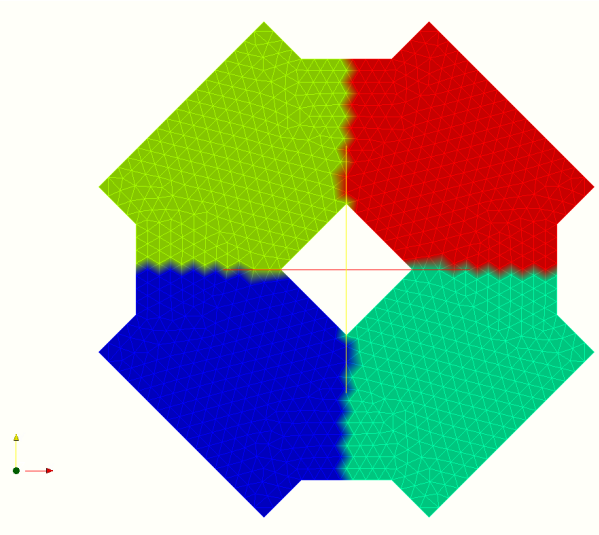


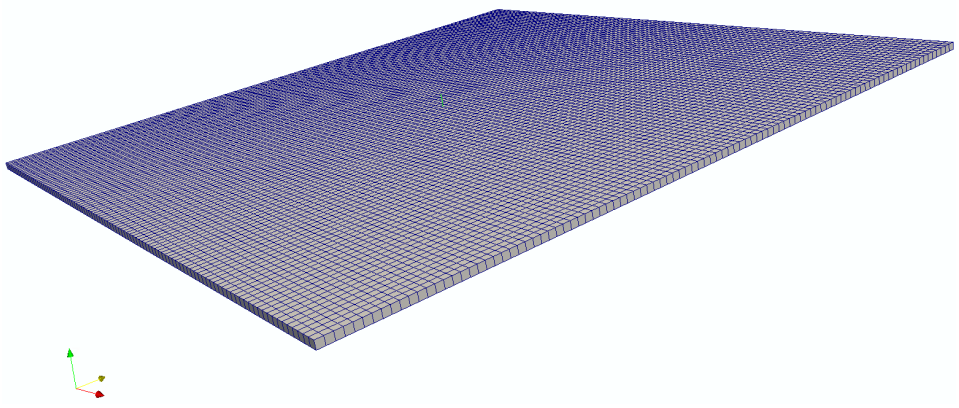
Figure: *Rendezvous decomposition for 2D shared domain example.*

Searching the Rendezvous Decomposition

- Hierarchical parallel search tree
- Rendezvous decomposition provides parallel search
- kD-tree provides on-process proximity search
- Newton iterations provide final point location
- Results in reasonable scalability

DTK Implementation Scaling Results

- Mesh-to-mesh transfer
- Worst case scenario study (all-to-all)
- Qualitatively similar to the SIERRA results
- Largest problems so for over 1.0E9 elements and 1.0E5 cores



Strong Scaling

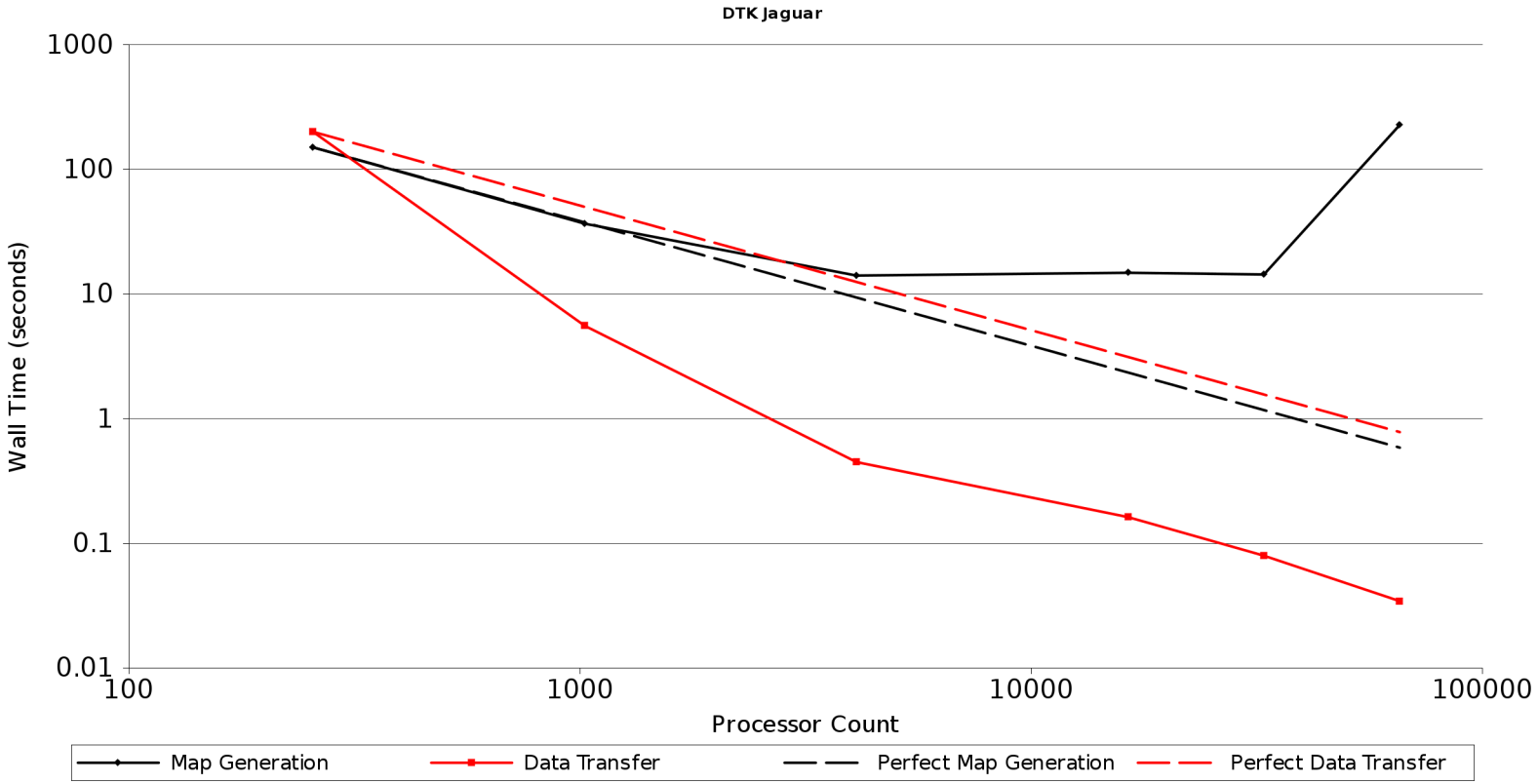


Figure: Strong scaling study results. The solid black curve reports the wall time to generate the mapping vs. number of processors while the solid red curve reports the wall time to transfer the data vs. number of processors. The dashed lines give perfect strong scaling the map generation (black) and the data transfer (red).

Weak Scaling

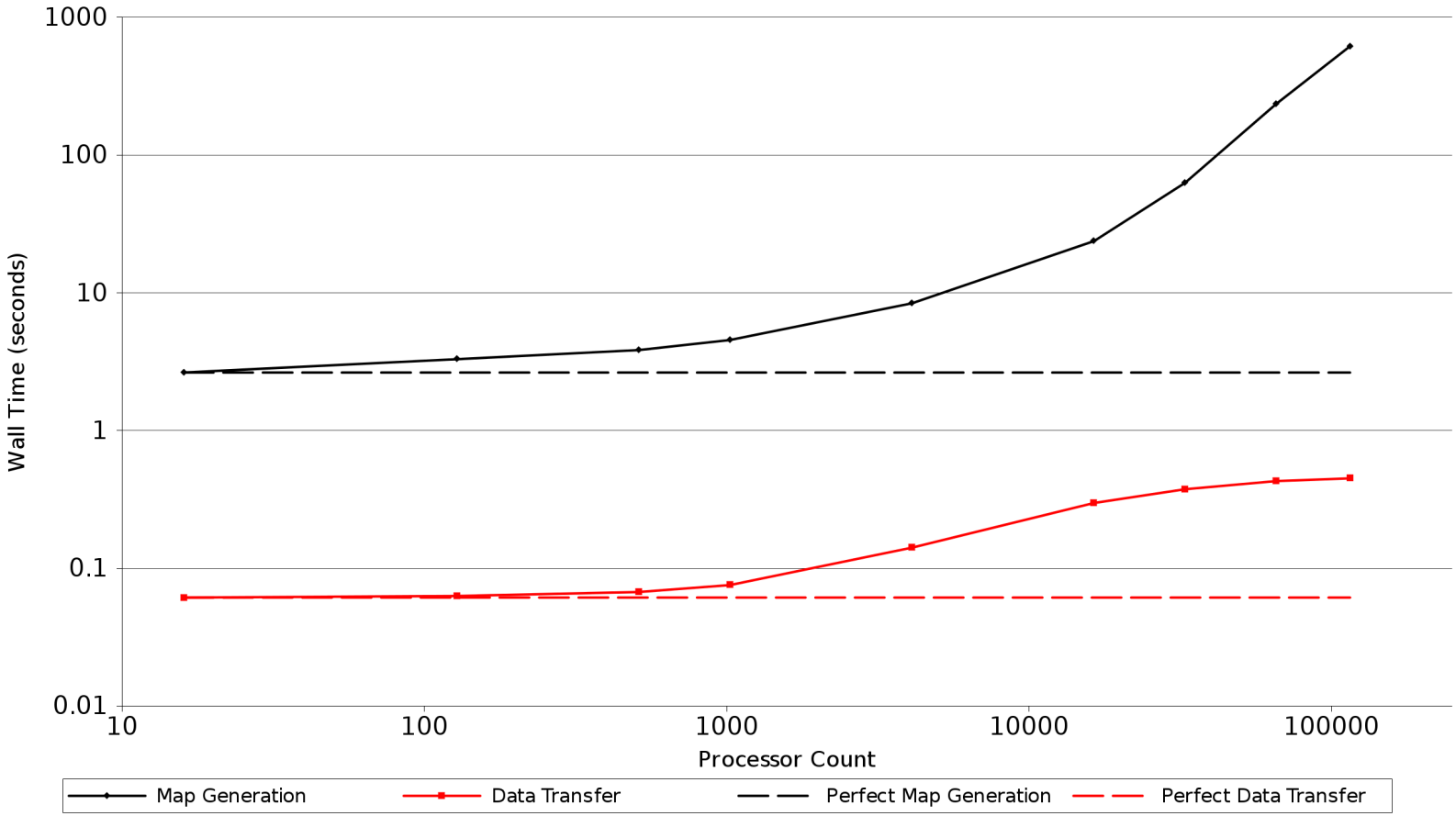


Figure: Weak scaling study results. The solid black curve reports the wall time to generate the mapping vs. number of processors while the solid red curve reports the wall time to transfer the data vs. number of processors. The dashed lines give perfect weak scaling the map generation (black) and the data transfer (red).

Getting Data into DTK: The Mesh

- Meshes are viewed as geometric structures
- A subset of total mesh information is needed:
  - Vertex coordinates
  - Element topology
  - Element connectivity
  - Connectivity permutation
- Parallel information is not required
- A communicator and global IDs are required

Mesh Vertices and Elements

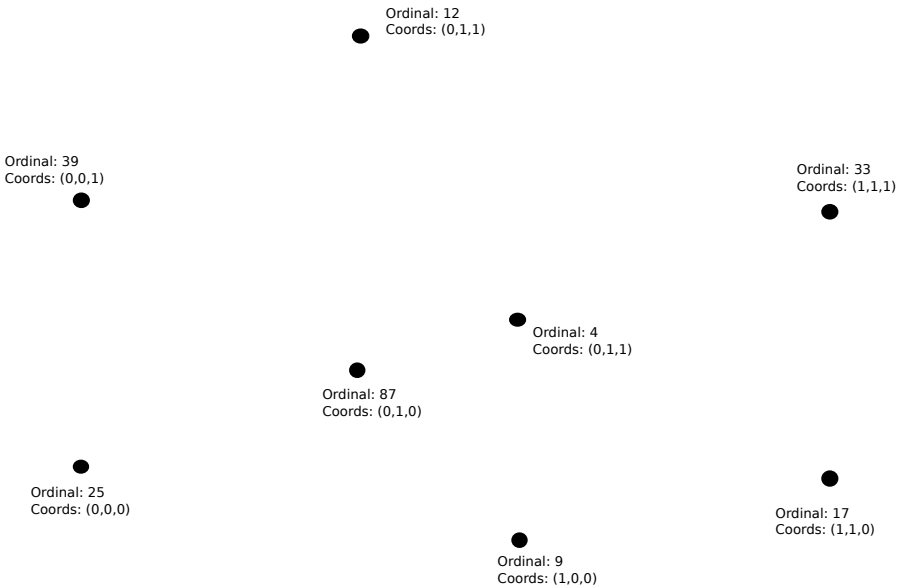


Figure: *Basic vertex description for a mesh.*

Each vertex is required to have a unique global ID

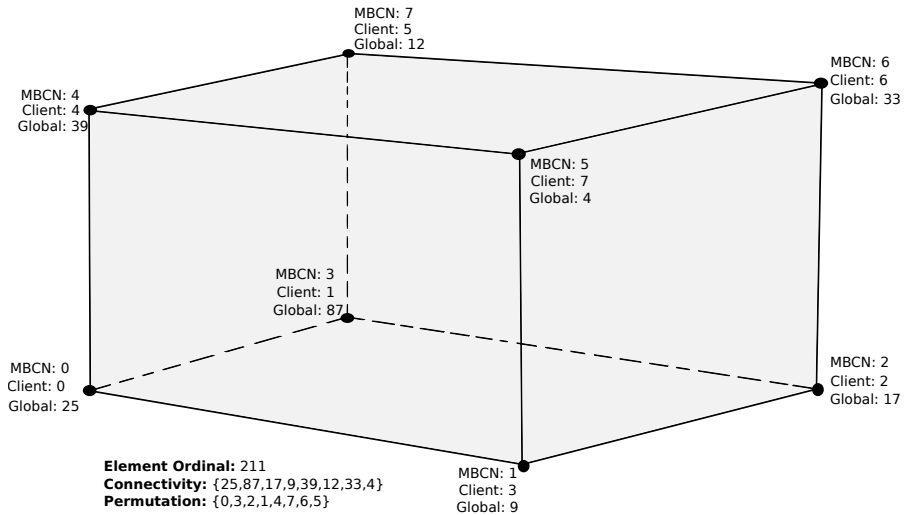
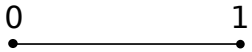


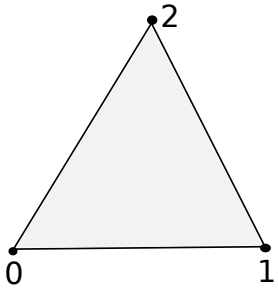
Figure: *Basic element description for a mesh.*

Each element is required to have a unique global ID

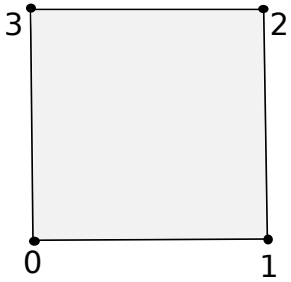
Connectivity Permutation



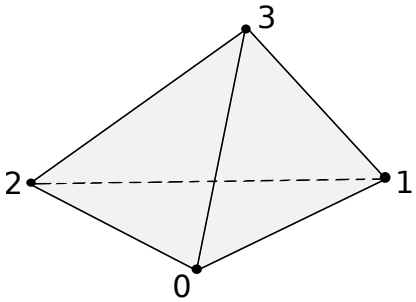
LINE SEGMENT



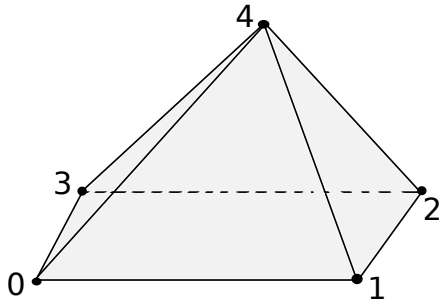
TRIANGLE



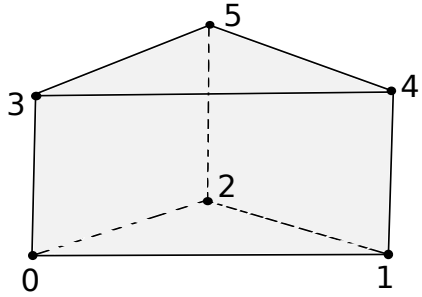
QUADRILATERAL

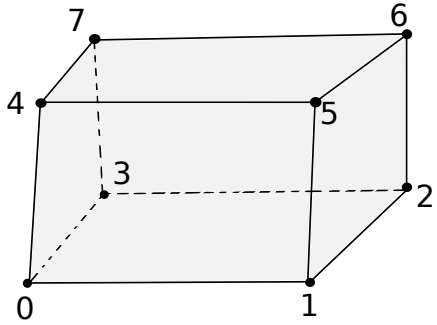


TETRAHEDRON



PYRAMID





- Allow for user specification of canonical ordering
- Permutation list specifies the variation in ordering for a specified topology
- Support for other mesh topologies currently not offered

Meshes of Multiple Topologies

- All topologies in a mesh must be of the same dimension
- Each topology contained in a block
- Can have multiple blocks of the same topology

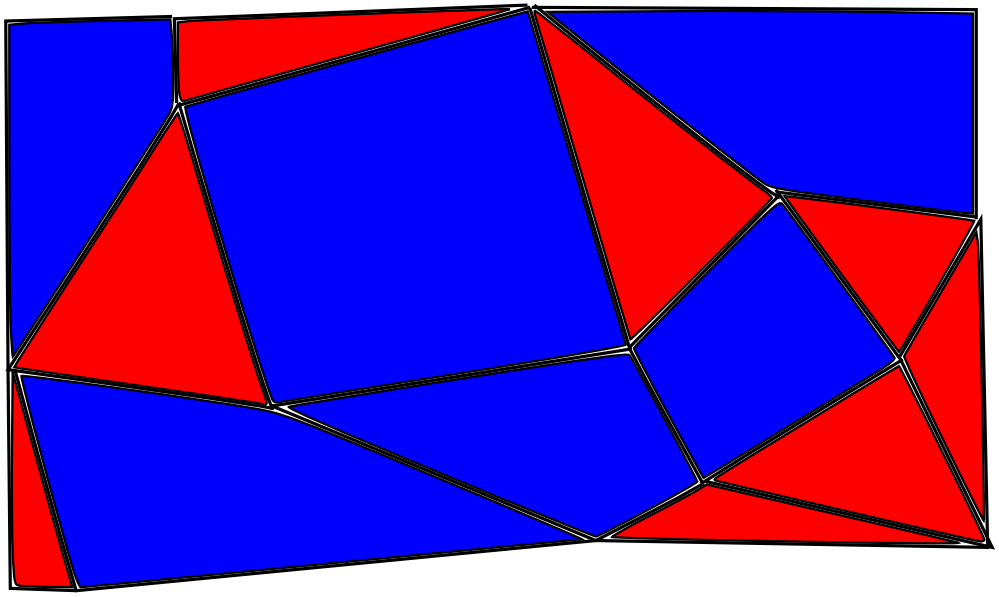


Figure: *Hybrid mesh example.*

Quadrilaterals (blue) must be specified in a different mesh block than the triangles (red). Both blocks can contain the mutual mesh vertices that construct their elements.