A Survey of Mesh Partitioning Techniques for Irregular Grids

Michael M. Wolf

Overview

- Static Global Mesh Partitioning
  - Mesh Independent Partitioning
    - Random Partitioning
    - Scattered Decomposition
    - Regular Domain Partitioning
  - Geometric Partitioning Algorithms (1D/2D/3D)
    - Recursive Coordinate Bisection (RCB)
    - Recursive Inertial Bisection (RIB)
    - Hilbert Space-Filing Curve (HSFC)
  - Graph Partitioning Algorithms
    - Greedy Bisection
    - Recursive Layered (Graph) Bisection
    - ParMETIS
  - Problems with Standard Graph Partitioning Model
    - Hypergraph
- Local Refinement
  - Kernighan-Lin Algorithm
  - Helpful Sets

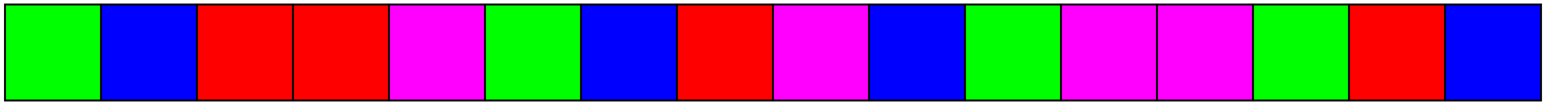
Global Static Partitioning Algorithms

- Partition entire mesh
- Partition once
- Not concerned with refining or evolving partition as the simulation progresses
- Algorithms may be applicable to dynamic partitioning schemes

Geometry Independent(?) Partitioning Algorithms

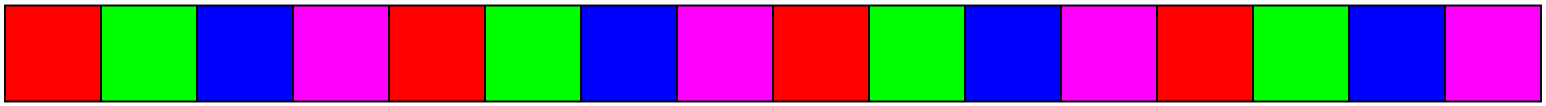
- Kind of geometry independent
- Based on the order on which the elements are operated
- Ignores x,y,z position of elements
- Ignores mesh connectivity

Random Partitioning (Geometry Independent)



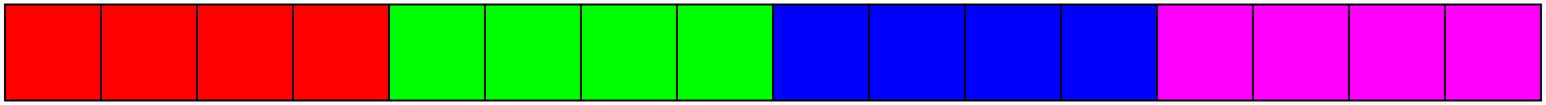
- Each element is distributed to a randomly chosen processor
- On average, work is well balanced
- No grouping by mesh connectivity
- No grouping by mesh locality
- Communication is thus BAD!!!

Scattered Partitioning (Geometry Independent)



- Each element is distributed in order to the processor with the current smallest subdomain
- Work is well balanced
- Neighbor elements won't be on same processor.
- Communication is thus BAD!!!

Regular Domain Partitioning (Geometry Independent?)

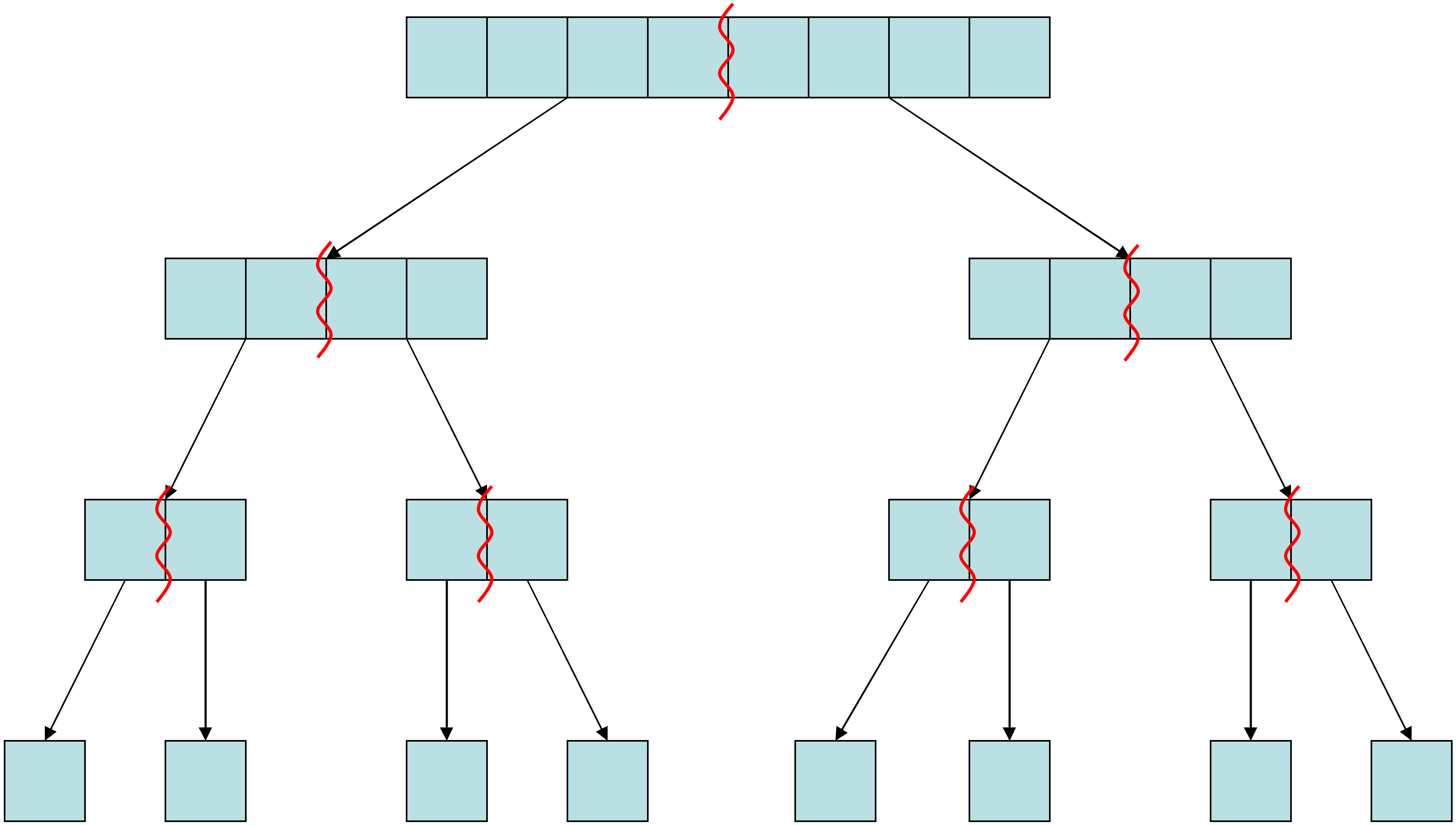


- First n/p elements given to proc 0.
- Second n/p elements given to proc 1.
- ... etc.
- Data Locality if numbering supports.
- Communication is better but still possible problems

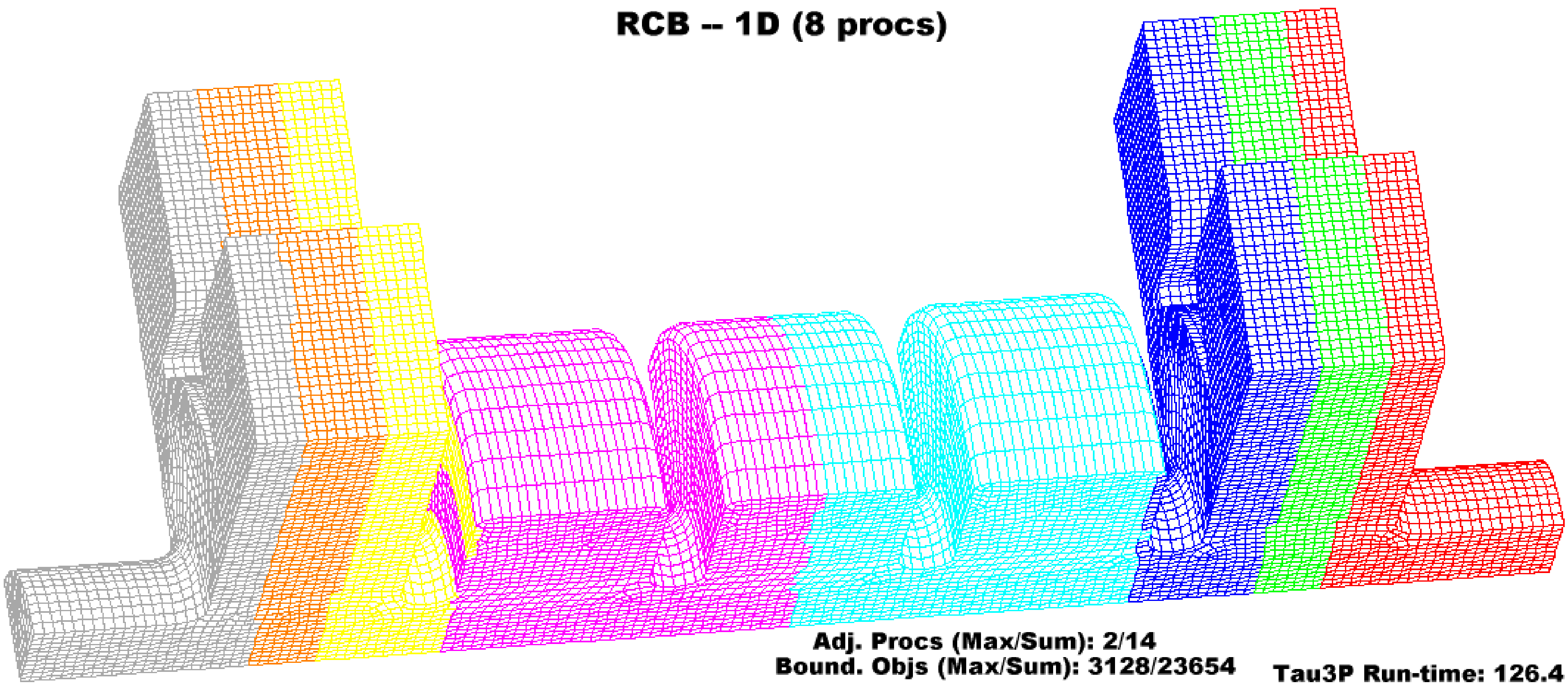
Geometric Partitioning Algorithms

- Elements grouped by geometric region
- Based on x,y,z position of elements
- Ignores element adjacency
- 1D, 2D, or 3D
- Fast
- Load Balance (at least in terms of elements) can be guaranteed

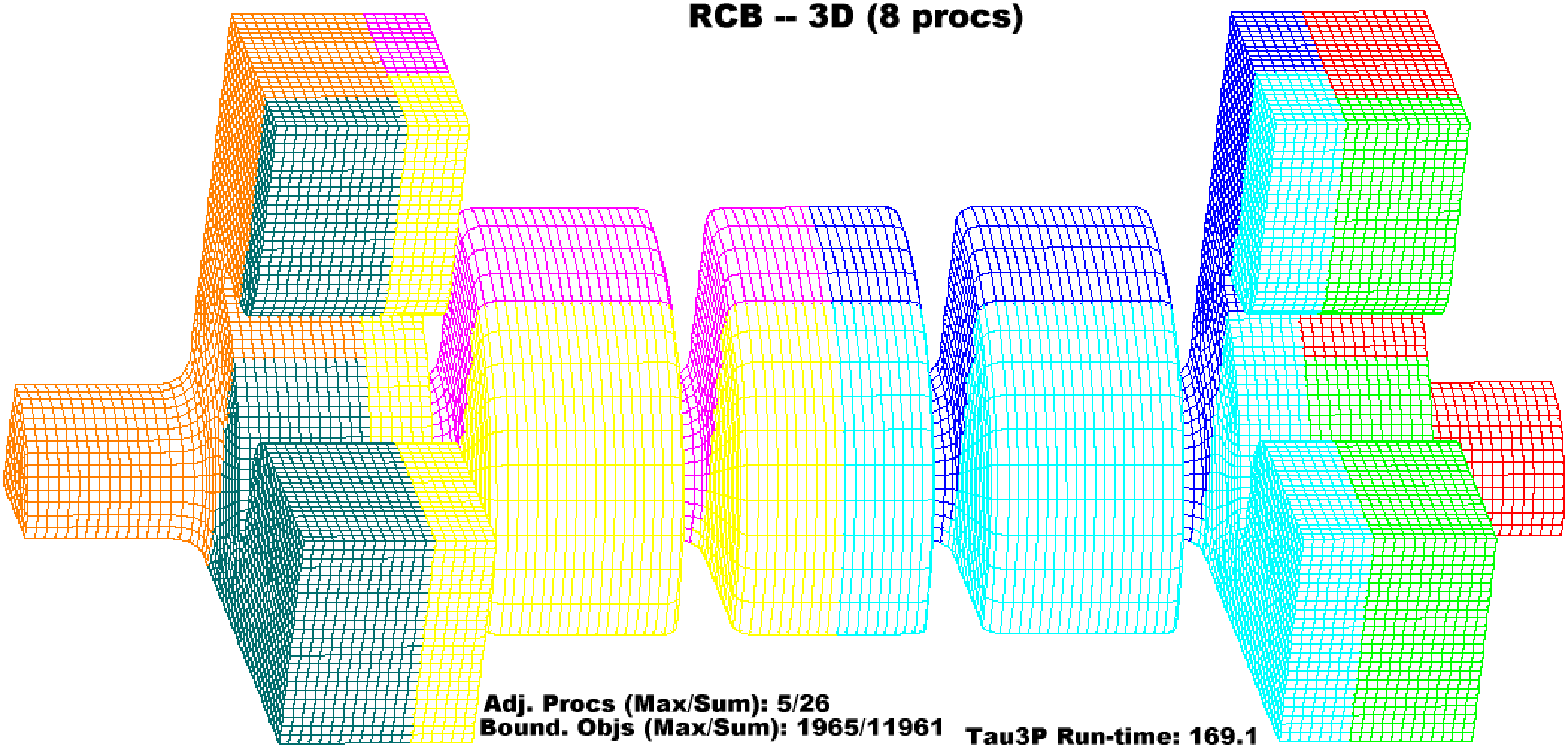
Recursive Coordinate Bisection (Geometric)



RCB-1D Partition



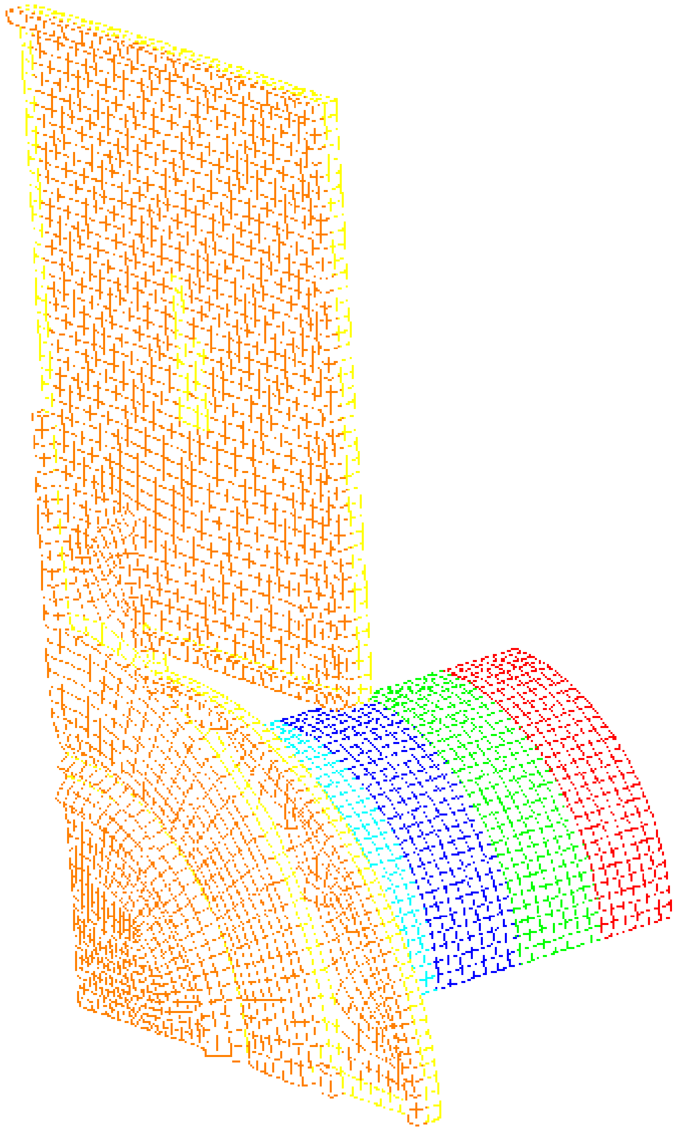
RCB-3D Partition



1D vs. 2D vs. 3D Partitioning

- The higher the dimension, the lower the surface/volume ratio.
  - Lower bandwidth
- The higher the dimension, the more neighboring subdomains each subdomain will bound.
  - More communications, more total latency.

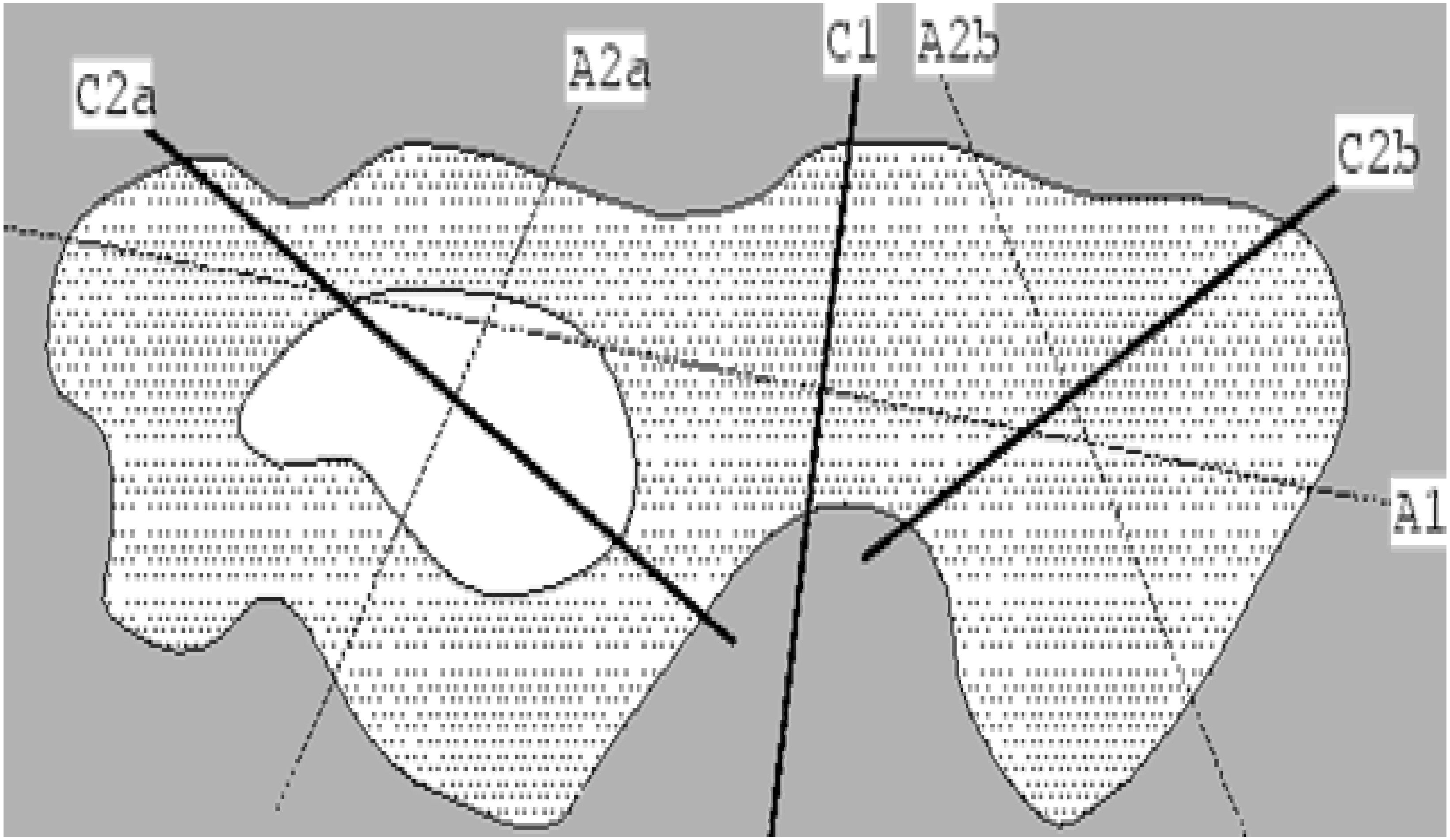
RCB 1-D Scalability Leveling Off



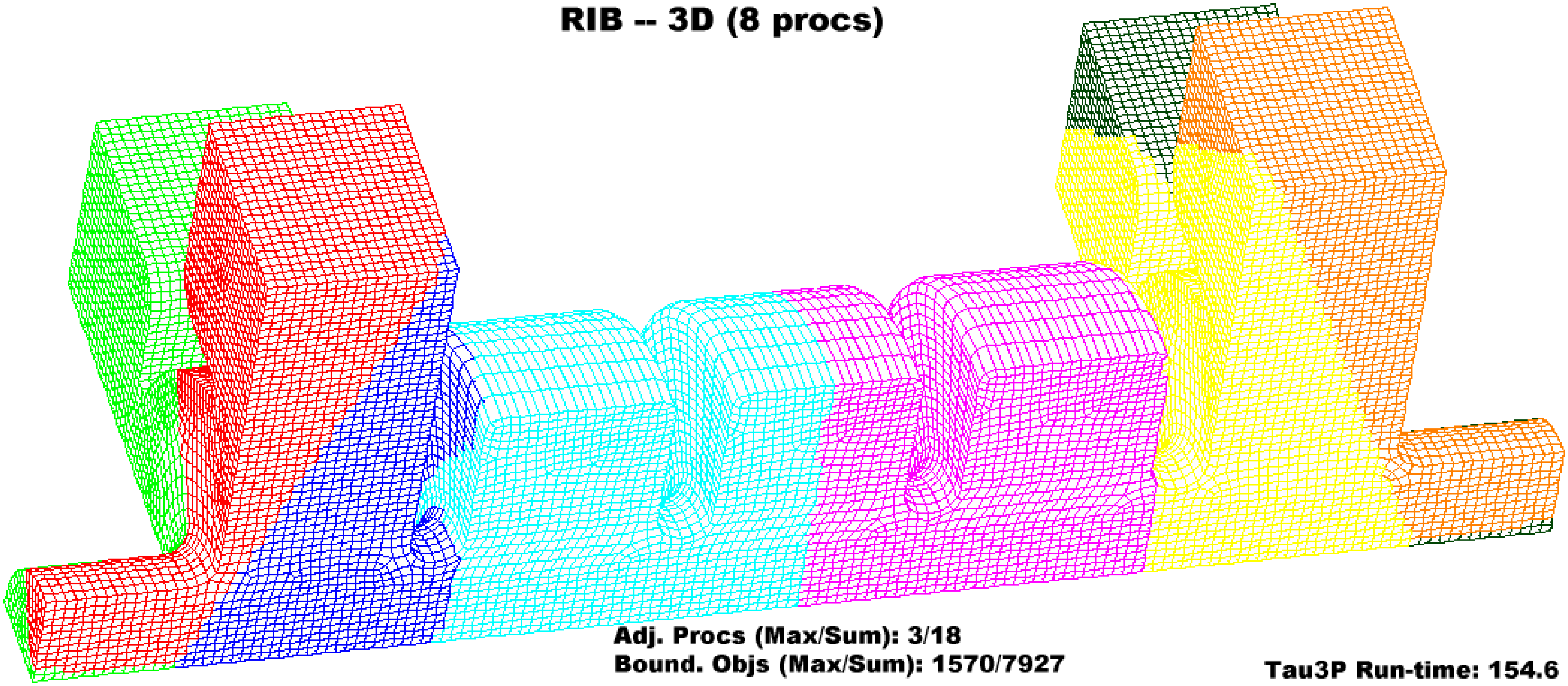
Recursive Inertial Bisection (Geometric)

- RCB problem when mesh not aligned with XYZ axes
- RIB uses idea of inertia to improve upon RCB

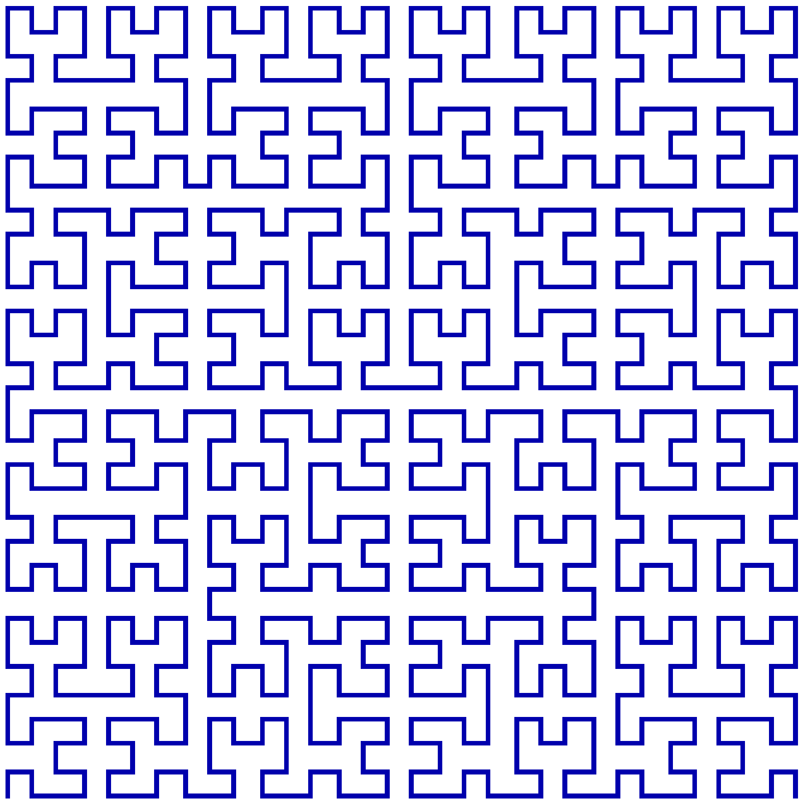
Recursive Inertial Bisection (Geometric)



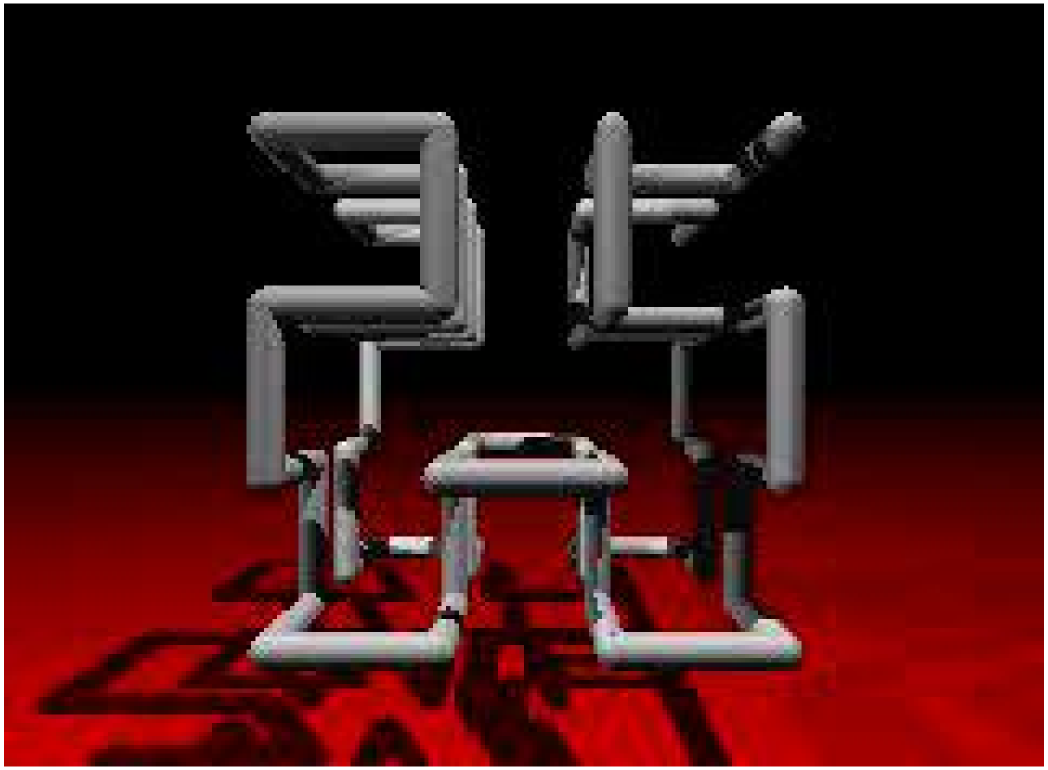
RIB-3D Partition



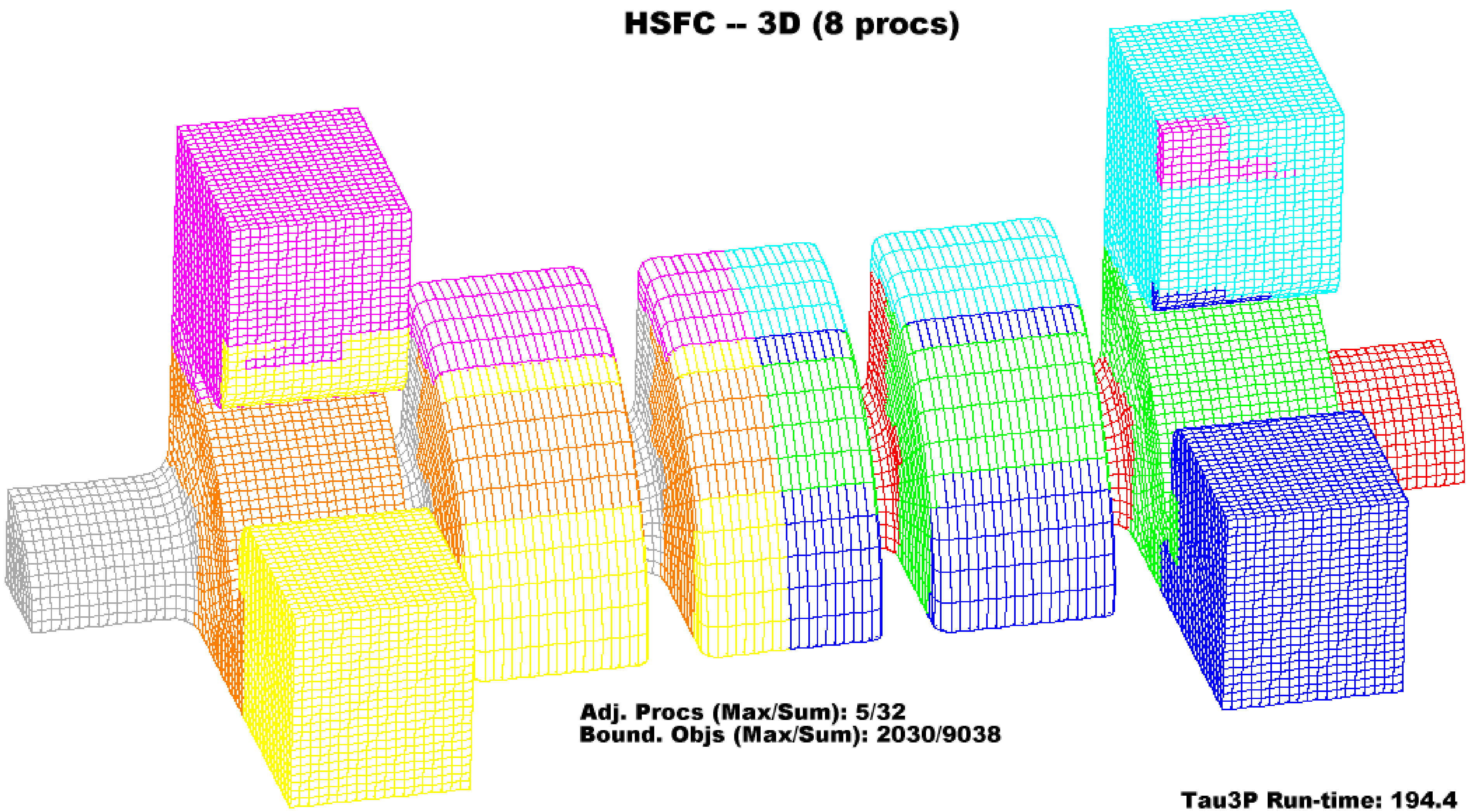
Hilbert Space Filling Curve (Geometric)



Hilbert Space Filling Curve (Geometric)



RDDS (5 cell w/ couplers) HSFC-3D Partition



Graph Partitioning Algorithms

- Build graph out of mesh elements
- $G = (V,E)$
- Elements are graph vertices
- Graph vertices of adjacent elements connected by edges in graph

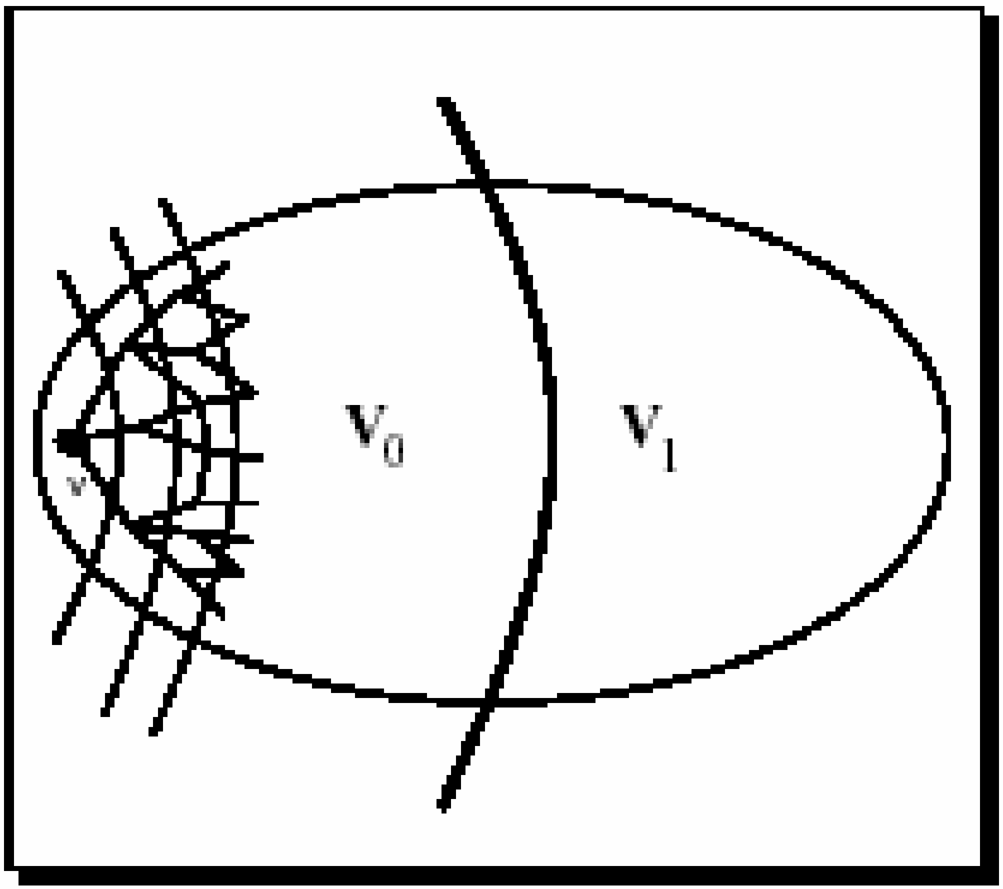
Graph Partitioning Properties

- Each Partition “should” be continuous.
- Uses element connectivity so partitions have little discontinuity.
- Slower than basic geometric methods

Greedy Bisection (Graph)

- Build graph
- Start at vertex of lowest degree
- Find neighboring layer.
- Repeat with vertices in that layer to find next layer, etc.
- Stop when n/p vertices are found
- Repeat process

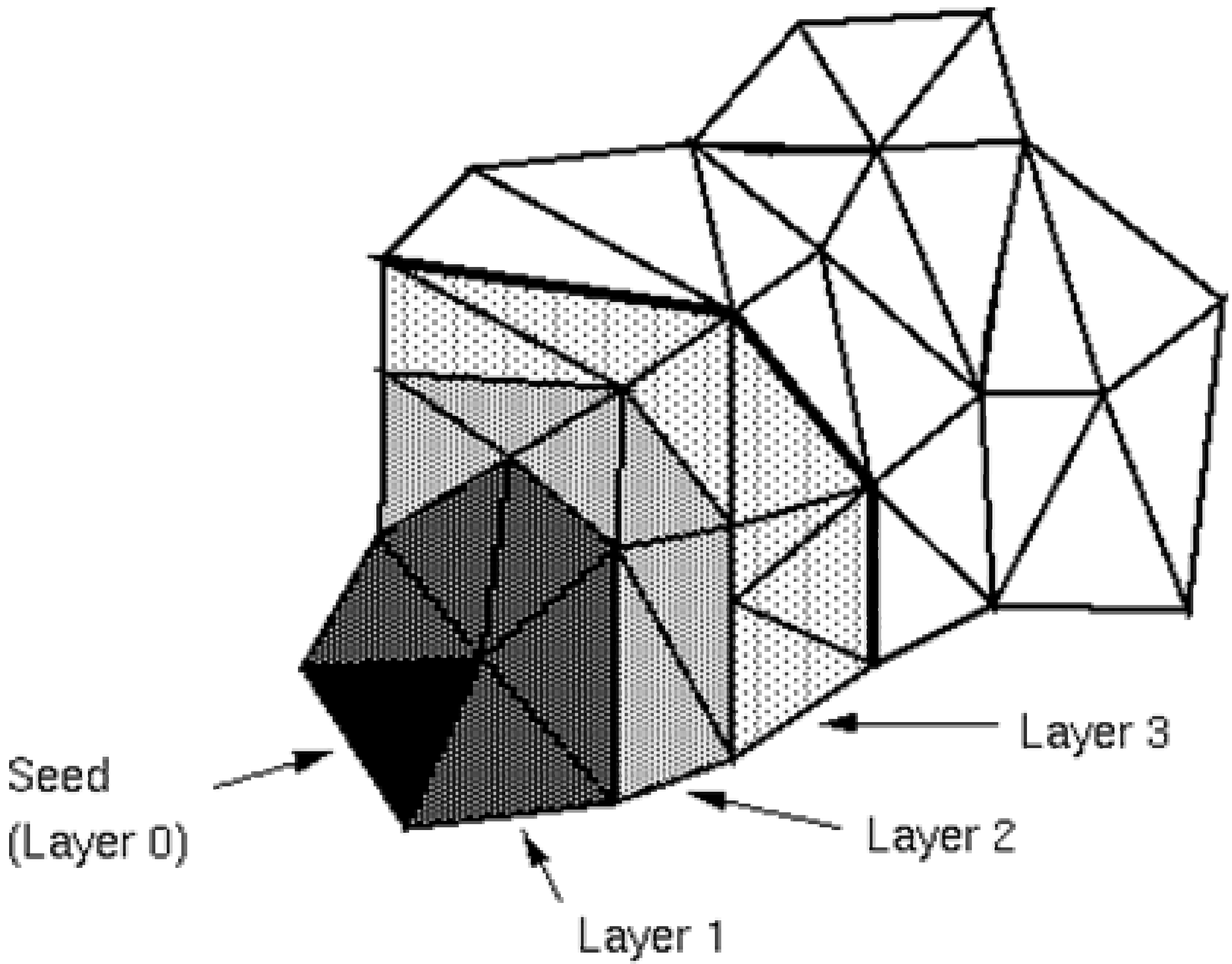
Greedy Bisection (Graph)



Recursive Layered (Graph) Bisection

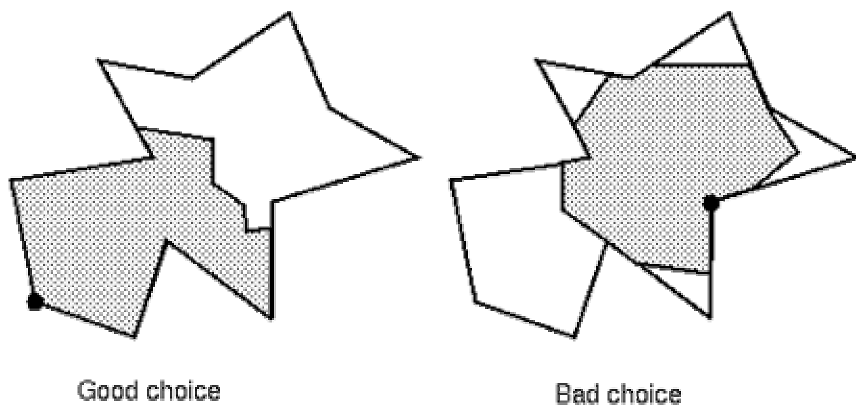
- Build graph
- Start from a seed vertex,
- Find neighboring layer.
- Repeat with vertices in that layer to find next layer, etc.
- Stop when number of vertices in layers reaches half.
- Now have 2 sets
- Recursively Repeat.

Recursive Layered (Graph) Bisection



Choosing Seed Points

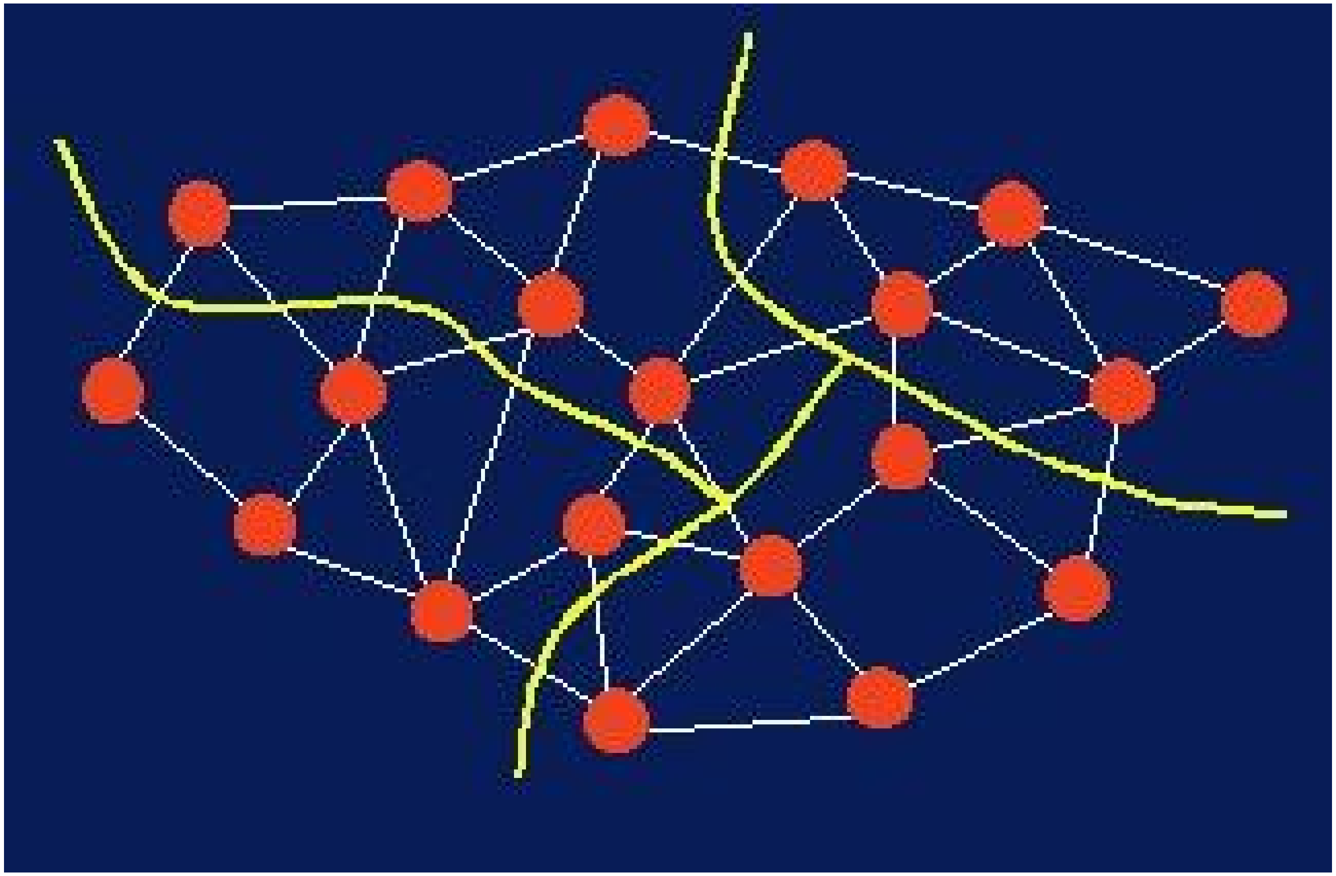
- Choice of Seed Points are Important
- Boundary of domain can be good choice.
- 1 of 2 points maximum distance apart.



ParMETIS (graph)

- Uses a standard graph approach
- Partition the vertices of the graph
- Minimize the (weighted) edge cut
- NP-hard problem
- Uses heuristics to generate approximate solutions

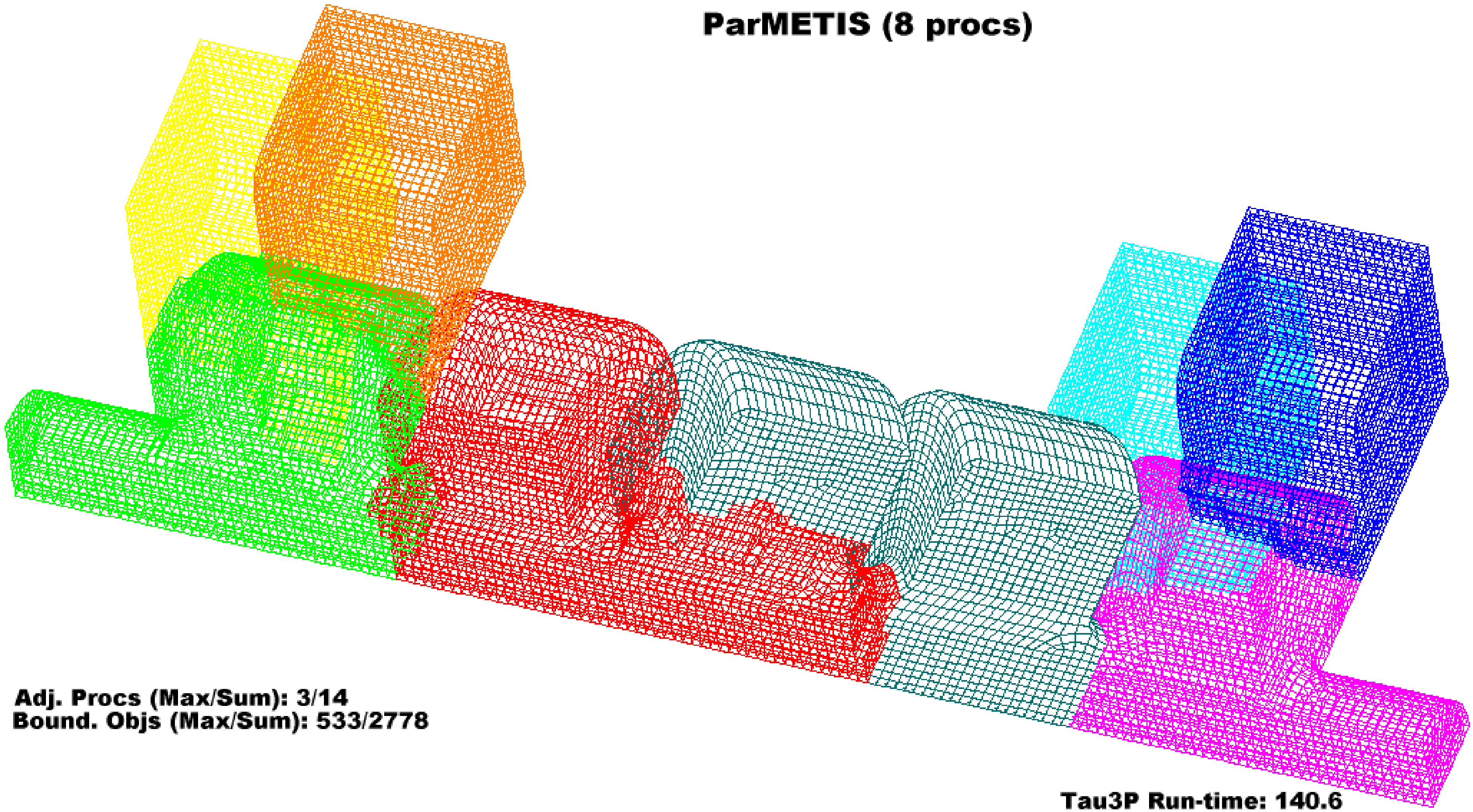
ParMETIS (Graph)



ParMETIS

- Main ParMETIS initial partition algorithm called ParMETIS_PartKway.
- Multi-level k-way partitioning algorithm
- Step 1: Graph gradually coarsened down to graph of a few hundred vertices.
- Step 2: k-way partition of coarse graph computed
- Step 3: Graph projected back to original graph by periodically refining partition.

RDDS (5 cell w/ couplers) ParMETIS Partition



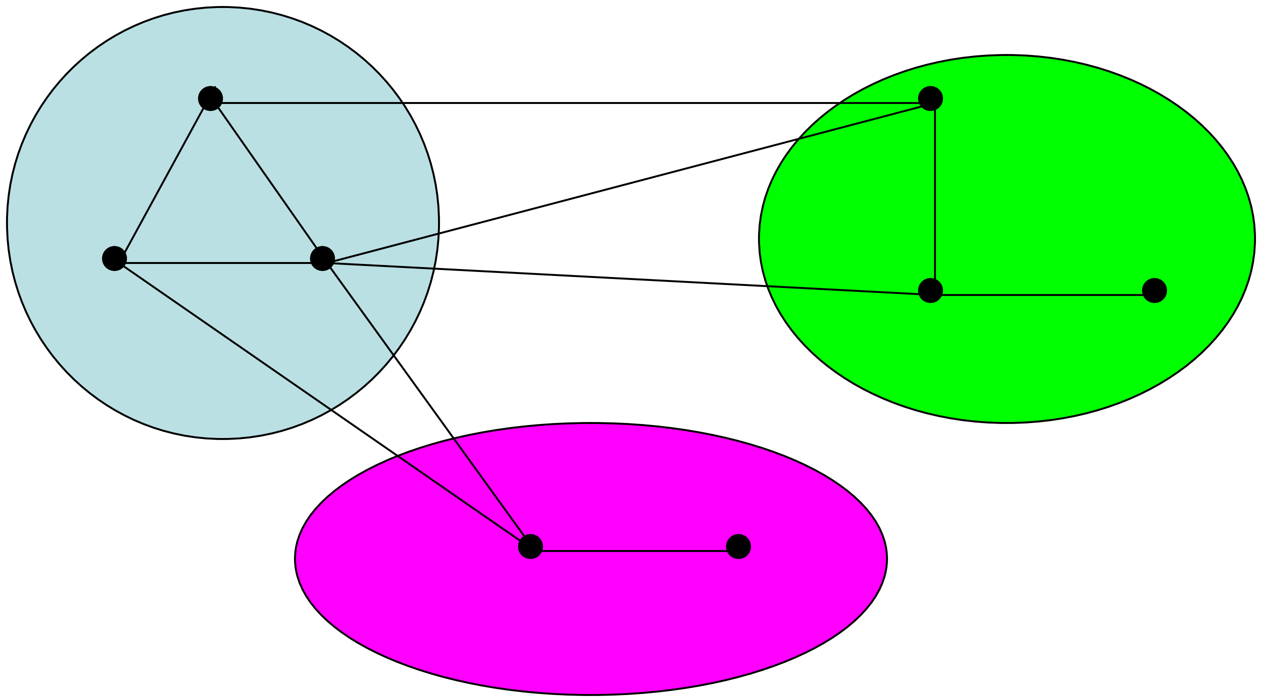
ParMETIS

- Multilevel makes k-way graph partitioning algorithm more acceptable.
- Does a great job of minimizing cut (i.e. bandwidth).
- However, pieces can be disconnected.
- Not as load balanced as geometric methods.
- More neighbors than 1D geometric methods (larger number of communications required).

Complications in Graph Partitioning

- Several potential shortcomings in standard graph partitioning model.
- Incorrect edge cut metric
- Limited in the scope of problems that can be naturally expressed (problem for other parallel partitioning problems)

Metrics



- Edge cuts not proportional to total volume
- Overcounting

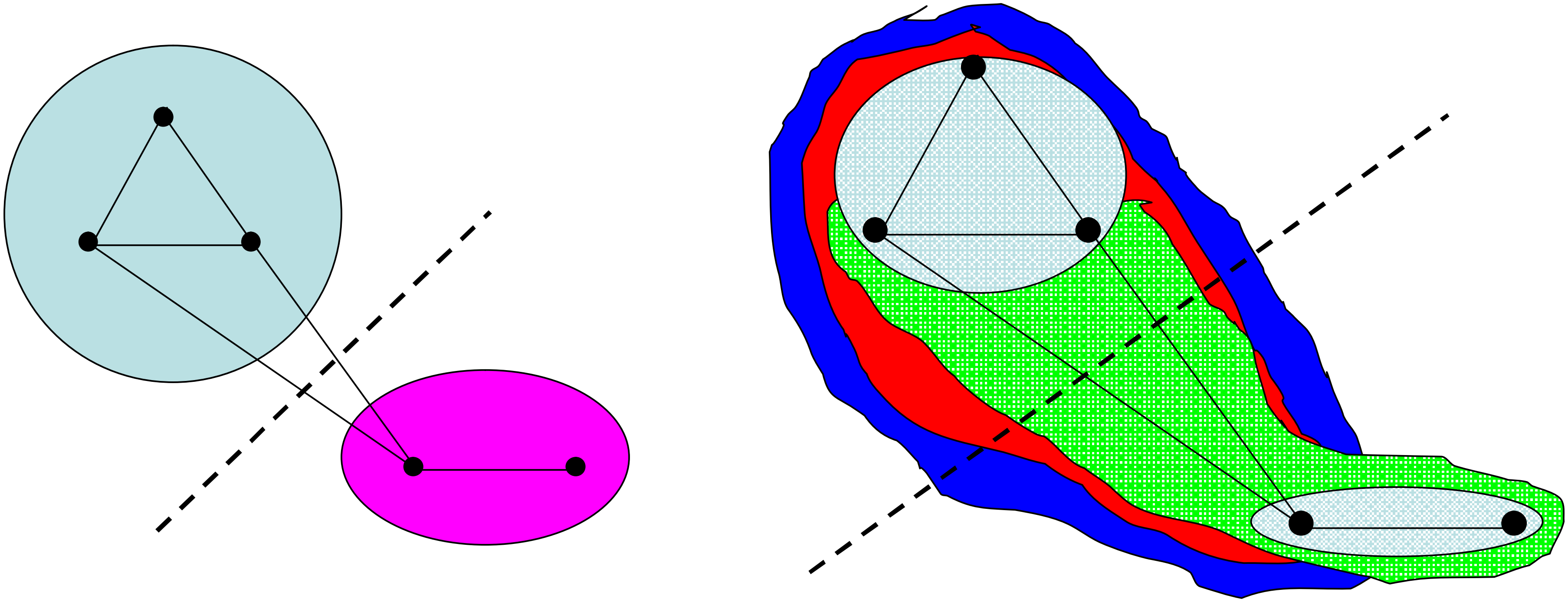
Metrics

- Communication costs are dependent on latency (total number of messages sent) as well as bandwidth
- Slowest process often most important
- Limited in the scope of problems that can be naturally expressed
- May want to limit communication to nearby processors
- Want to minimize objective function based on all of these, weighted by importance

Hypergraphs

- Hypergraphs can be used to better minimize communication in standard graph problem.
- Minimizes number of boundary cuts
- Build graph out of mesh elements
- $G = (V,H)$
- Elements are graph vertices
- A hyperedge exists for each vertex
- Hyperedge H_1 contains V_1 and its neighboring vertices

Hypergraph Partitioning



- Standard Graph Model (Cut=4)
- Hypergraph Model (Cut=3)

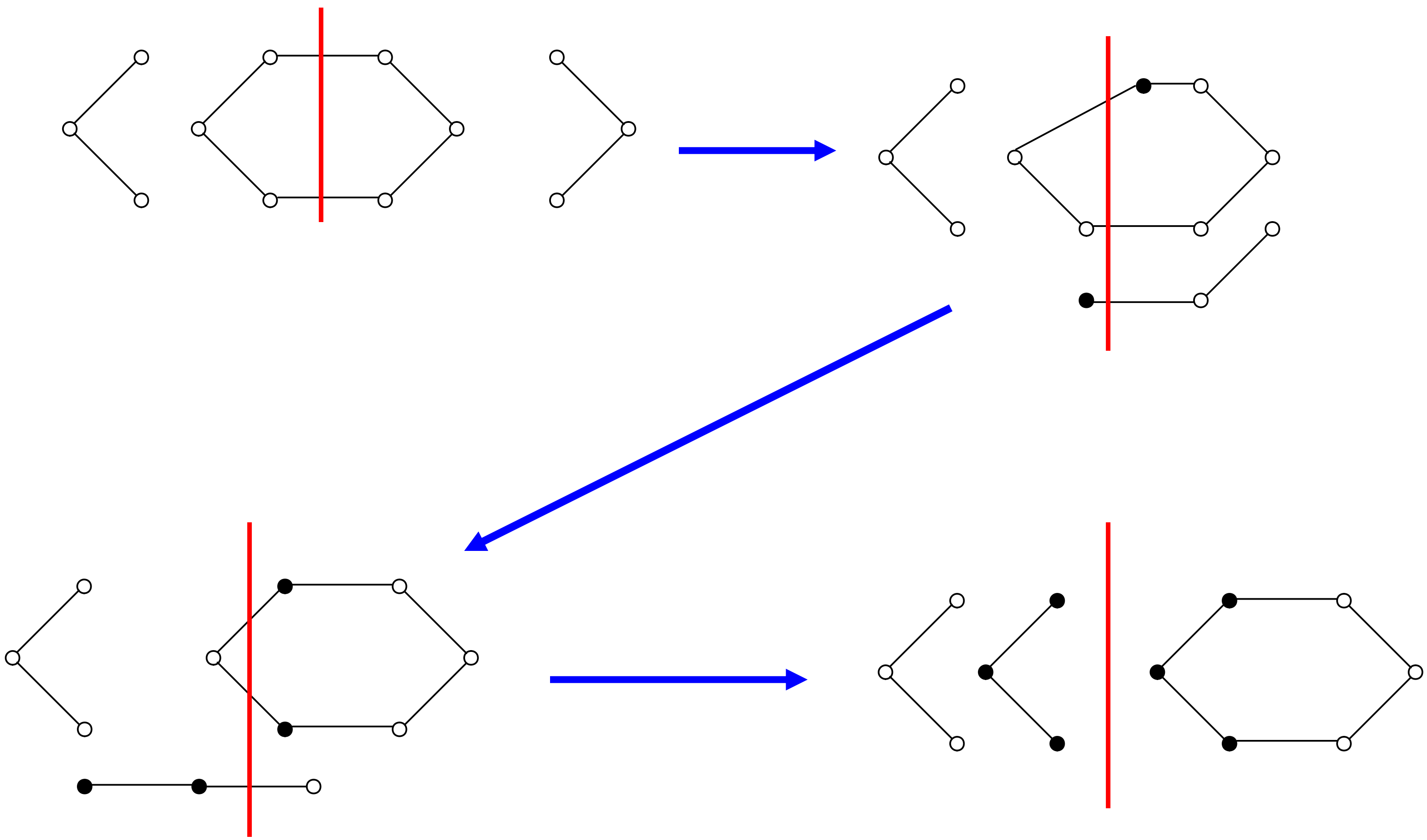
Local Refinement

- After initial partition, make small local changes to improve partition quality
- Swap small number of elements across process boundaries

Kernighan-Lin Algorithm

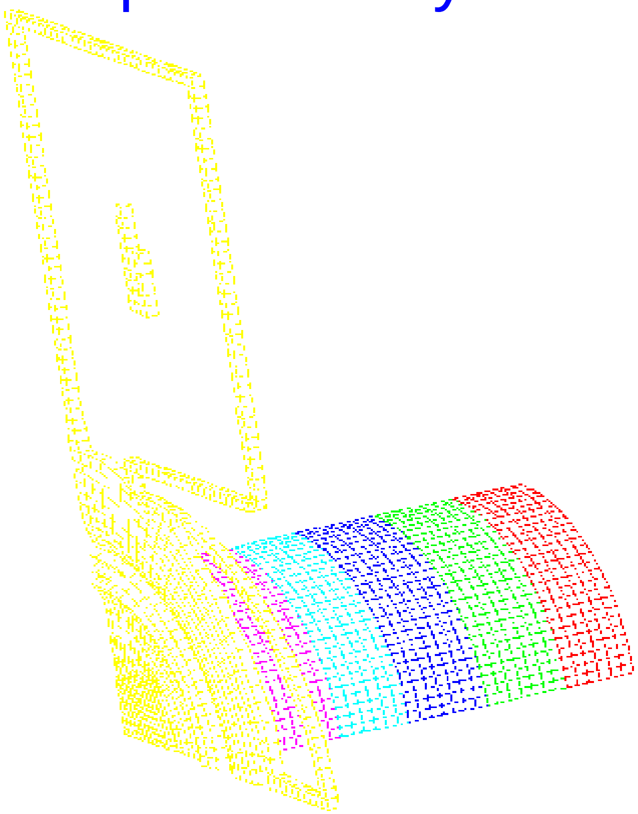
- Swap pairs of nodes to decrease the cut
- Allow intermediate increases in the cut size to avoid local minima
- Loop
  - Logically exchange pair of nodes with largest gain from swapping
  - lock those nodes
  - until all nodes are locked
- If new partition is better than old, save.
- Perform the swaps for real to obtain final partition on the best partition found
- Different heuristics used to improve speed of algorithm.

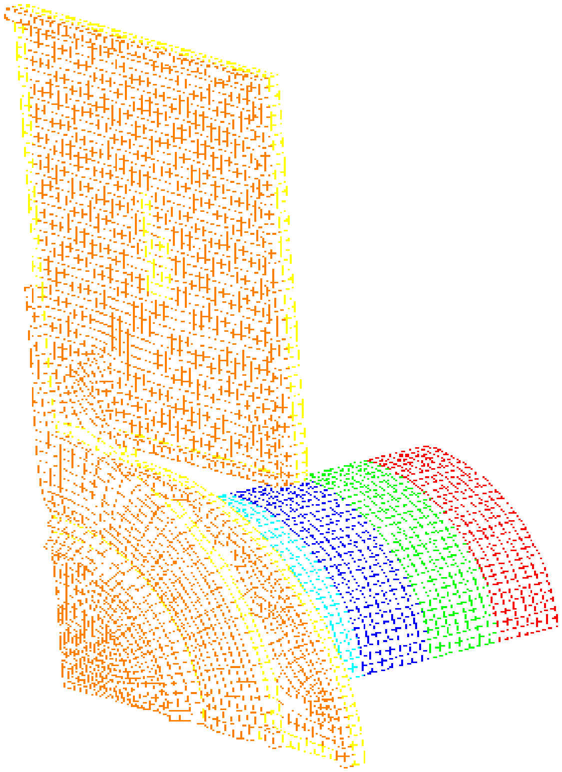
Kernighan-Lin Algorithm



Kernighan-Lin Algorithm

- Does not partition poorly partitioned meshes well.
- Often used in conjunction with a very computationally “cheap” global partitioning method.

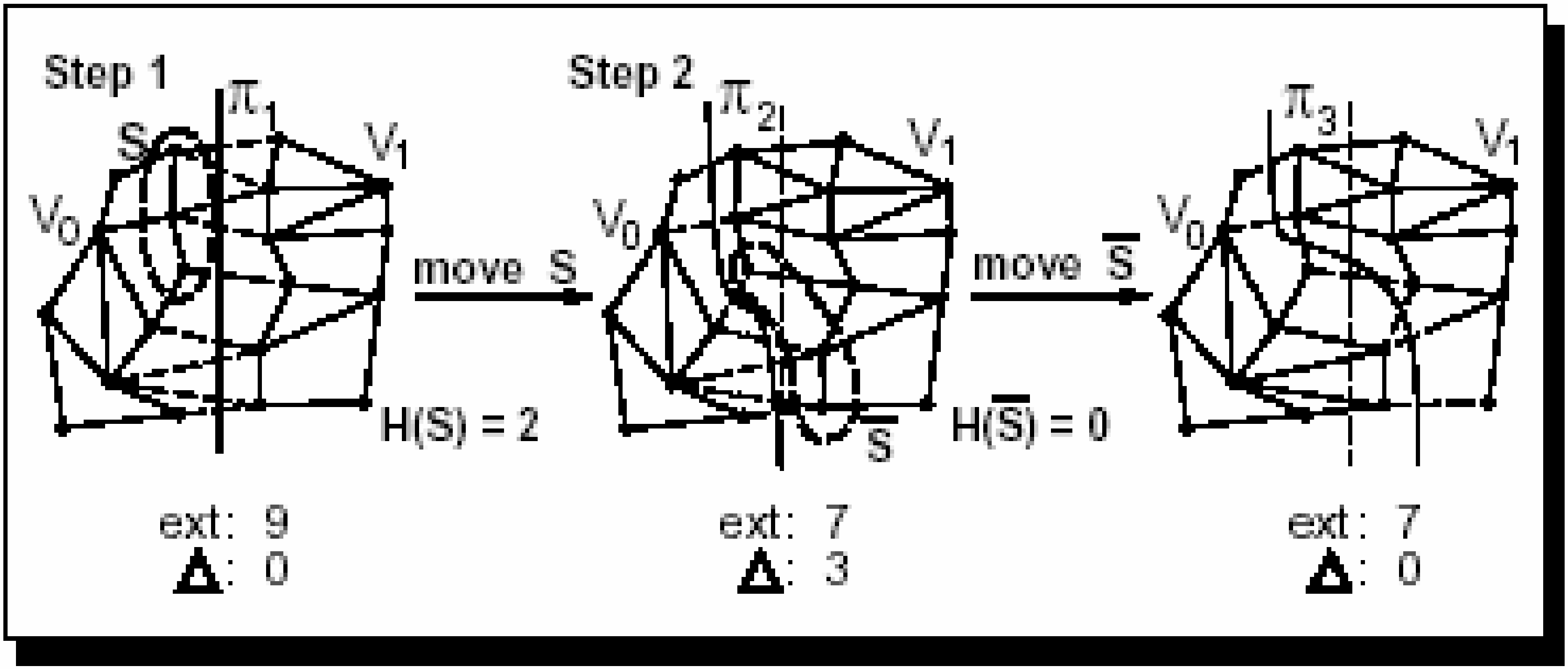




Helpful Sets

- A set of nodes is helpful if moving it from one processor to another (and rebalance) reduces the cut size.
- Step 1. Find a set of nodes in one partition and move it to the other partition to decrease the cut size
- Step 2. Rebalance the load
- Must be a net reduction in cut size after the two steps.

Helpful Sets



Acknowledgements

K. Devine, B. Hendrickson, E. Boman, M. St.John, and C. Vaughan. “Zoltan: A Dynamic Load-Balancing Library for Parallel Applications; User's Guide.” Sandia National Laboratories Tech. Rep. SAND99-1377, Albuquerque, NM, 1999.

Graph Partitioning Models for Parallel Computing, Bruce Hendrickson and Tamara G. Kolda, Parallel Computing. 26:1519-1534, 2000.

Overview

- Static Mesh Partitioning
  - Recursive Spectral Partitioning
  - Greedy Bisection
  - Recursive Spectral Bisection
  - Graph Partitioning Algorithms
    - Jostle
  - Geometric Partitioning Algorithms (1D/2D/3D)
    - Octree Partitioning (various traversal schemes including HSFC)
- Multi-level Hybrid Methods
- Dynamic Load Balancing/Data Migration
- Dynamic Load Balancing
  - Centralized
  - Decentralized
  - Fully Distributed
- Diffusion
- Dimension Exchange
- Advancing Front Algorithm
- Hypergraph
- Greedy algorithm

TODO

Hypergraph

- Kway graph partitioning?
- Understand Kernighan Lin Algorithm