

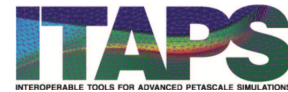


Dynamic Load Balancing and Partitioning using the Zoltan Toolkit

Karen Devine, Erik Boman, Vitus Leung, Lee Ann Riesen
Sandia National Laboratories



Umit Çatalyürek
Ohio State University



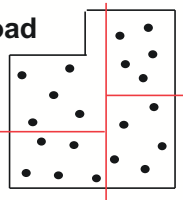
Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.



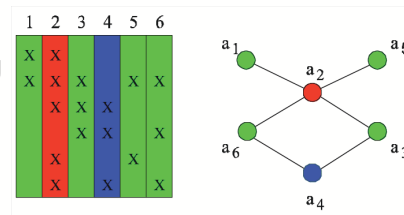
The Zoltan Toolkit

- Library of data management services for unstructured, dynamic and/or adaptive computations.

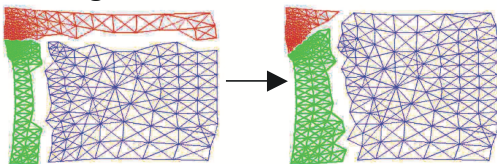
Dynamic Load Balancing



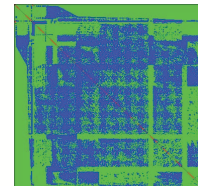
Graph Coloring



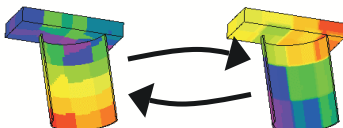
Data Migration



Matrix Ordering



Unstructured Communication



Distributed Data Directories

A	B	C	D	E	F	G	H	I
0	1	0	2	1	0	1	2	1

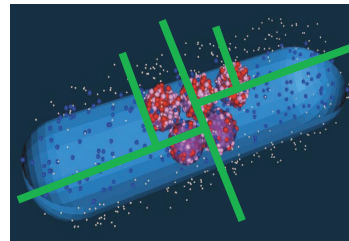
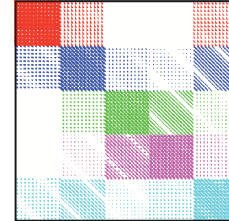
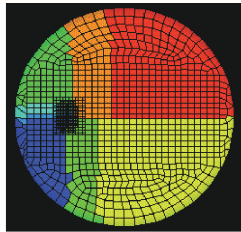
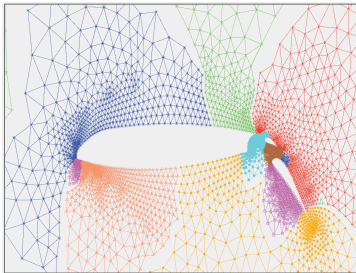
Dynamic Memory Debugging





Partitioning and Load Balancing

- Assignment of application data to processors for parallel computation.
- Applied to grid points, elements, matrix rows, particles,



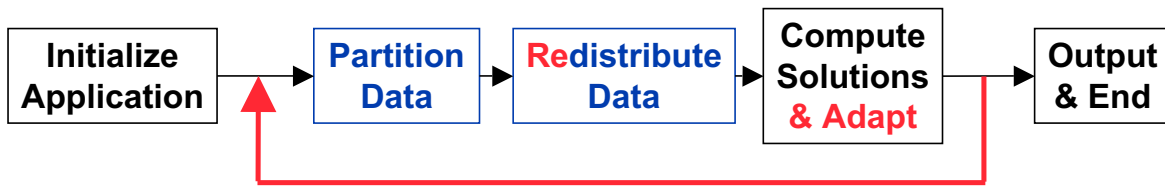
Static Partitioning



- **Static partitioning in an application:**
 - Data partition is computed.
 - Data are distributed according to partition map.
 - Application computes.
- **Ideal partition:**
 - Processor idle time is minimized.
 - Inter-processor communication costs are kept low.



Dynamic Repartitioning (a.k.a. Dynamic Load Balancing)



- Dynamic repartitioning (load balancing) in an application:
 - Data partition is computed.
 - Data are distributed according to partition map.
 - Application computes **and, perhaps, adapts**.
 - **Process repeats until the application is done.**
- Ideal partition:
 - Processor idle time is minimized.
 - Inter-processor communication costs are kept low.
 - **Cost to redistribute data is also kept low.**



What makes a partition “good,” especially at petascale?



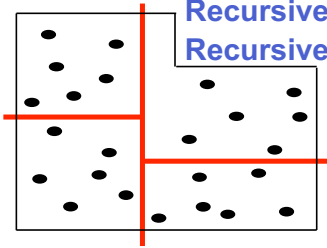
- **Balanced work loads.**
 - Even small imbalances result in many wasted processors!
 - **50,000 processors with one processor 5% over average workload is equivalent to 2380 idle processors and 47,620 perfectly balanced processors.**
- **Low interprocessor communication costs.**
 - Processor speeds increasing faster than network speeds.
 - Partitions with minimal communication costs are critical.
- **Scalable partitioning time and memory use.**
 - Scalability is especially important for dynamic partitioning.
- **Low data redistribution costs (for dynamic partitioning).**
 - Redistribution costs must be recouped through reduced total execution time.



Partitioning Algorithms in the Zoltan Toolkit



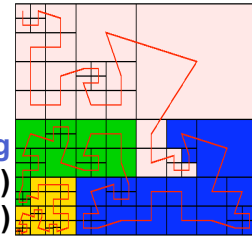
Geometric (coordinate-based) methods



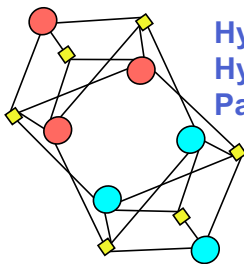
Recursive Coordinate Bisection (Berger, Bokhari)
Recursive Inertial Bisection (Taylor, Nour-Omid)

Space Filling Curve Partitioning
(Warren&Salmon, et al.)

Refinement-tree Partitioning (Mitchell)

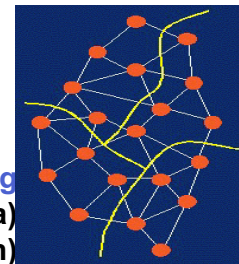


Hypergraph and graph (connectivity-based) methods



Hypergraph Partitioning
Hypergraph Repartitioning
PaToH (Catalyurek & Aykanat)

Zoltan Graph Partitioning
ParMETIS (U. Minnesota)
Jostle (U. Greenwich)



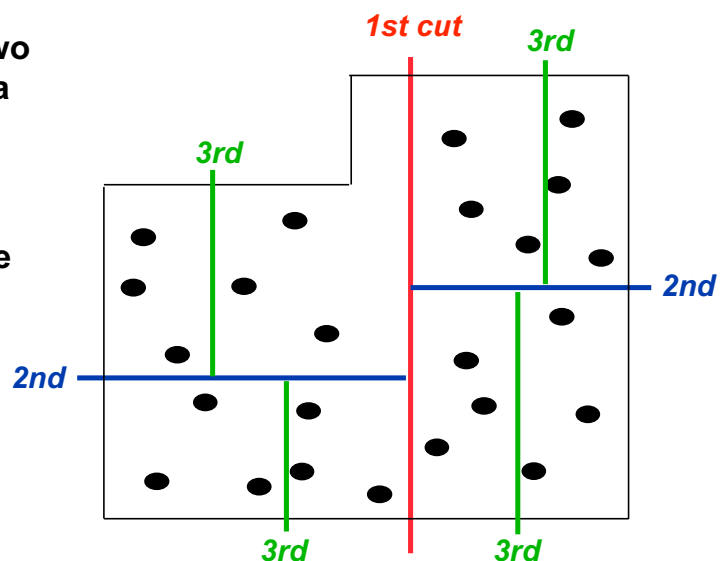
Geometric Partitioning: RCB



- Recursive Coordinate Bisection: Developed by Berger & Bokhari (1987) for Adaptive Mesh Refinement.

- Idea:

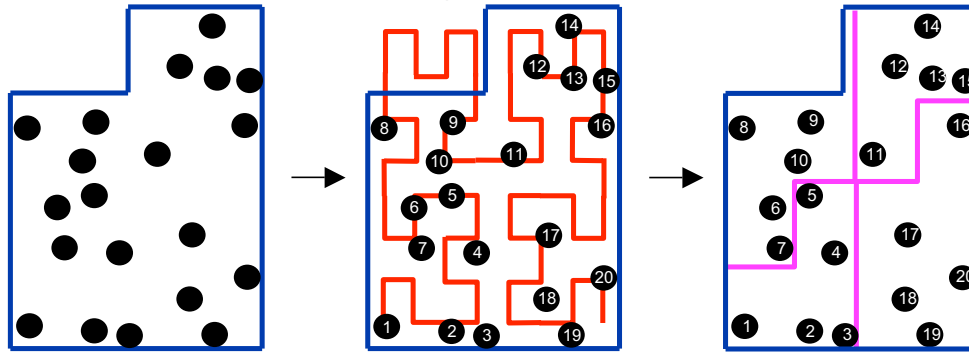
- Divide work into two equal parts using a cutting plane orthogonal to a coordinate axis.
- Recursively cut the resulting subdomains.



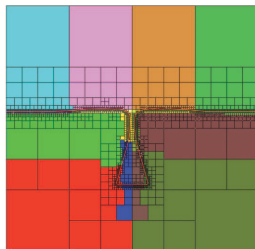


Geometric Partitioning: SFC

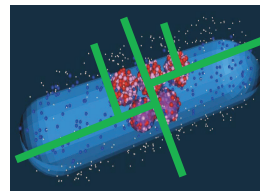
- **Space-Filling Curve Partitioning**
 - Gravitational simulations (Warren & Salmon, 1993)
 - Smoothed particle hydrodynamics (Pilkington & Baden, 1994)
 - Adaptive mesh refinement (Patra & Oden, 1995).
- **SFC Partitioning Algorithm:**
 - Run SFC through domain (mapping from R^3 to R^1).
 - Order objects according to position on curve.
 - Perform 1-D partitioning of curve.



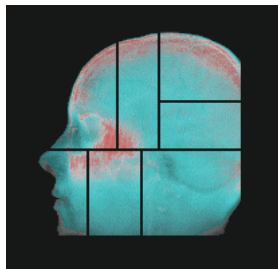
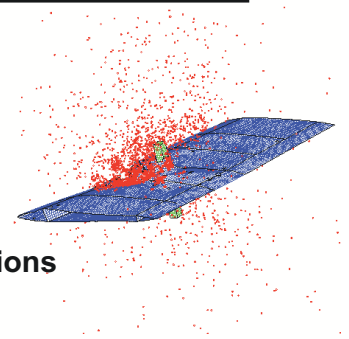
Applications of Geometric Methods



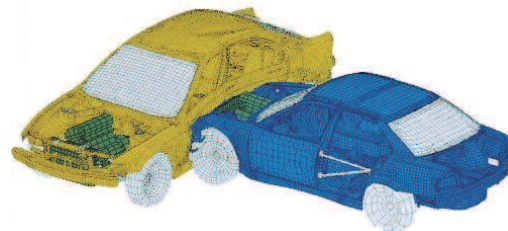
Adaptive Mesh Refinement



Particle Simulations



Parallel Volume Rendering



Crash Simulations and Contact Detection



Geometric Methods

Advantages and Disadvantages



- **Advantages:**
 - Conceptually simple; fast and inexpensive.
 - All processors can inexpensively know entire partition (e.g., for global search in contact detection).
 - No connectivity info needed (e.g., particle methods).
 - Good on specialized geometries.



*SLAC'S 55-cell Linear Accelerator with couplers:
One-dimensional RCB partition reduced runtime up
to 68% on 512 processor IBM SP3. (Wolf, Ko)*

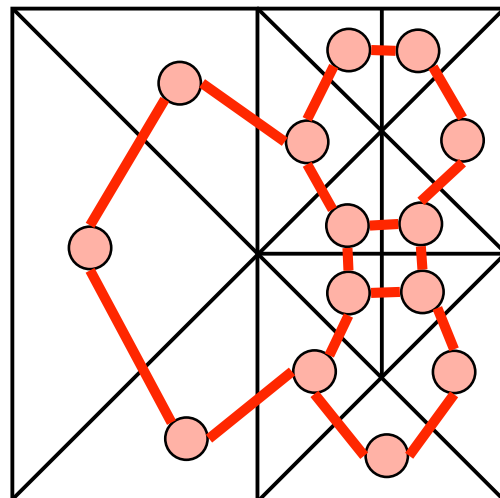
- **Disadvantages:**
 - No explicit control of communication costs.
 - Mediocre partition quality.
 - Need coordinate information.



Graph Partitioning

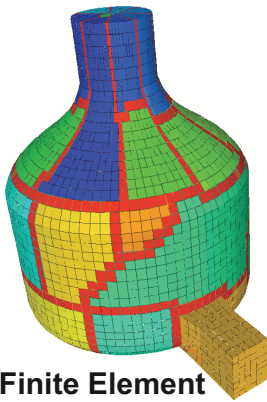


- **Kernighan, Lin, Schweikert, Fiduccia, Mattheyes, Simon, Hendrickson, Leland, Kumar, Karypis, et al.**
- **Represent problem as a weighted graph.**
 - Vertices = objects to be partitioned.
 - Edges = dependencies between two objects.
 - Weights = work load or amount of dependency.
- **Partition graph so that ...**
 - Parts have equal vertex weight.
 - Weight of edges cut by part boundaries is small.

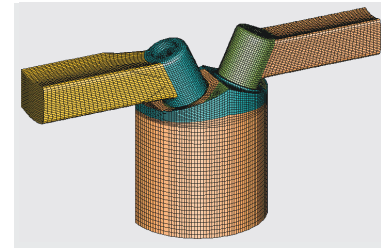




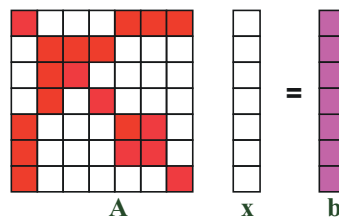
Applications using Graph Partitioning



Finite Element Analysis



Multiphysics and multiphase simulations



Linear solvers & preconditioners
(square, structurally symmetric systems)



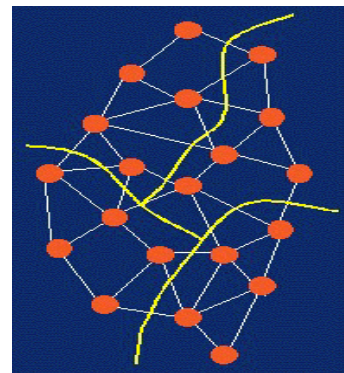
Graph Partitioning: Advantages and Disadvantages



• Advantages:

- Highly successful model for mesh-based PDE problems.
- Explicit control of communication volume gives higher partition quality than geometric methods.
- Excellent software available.

- Serial:
 - Chaco (SNL)
 - Jostle (U. Greenwich)
 - METIS (U. Minn.)
 - Party (U. Paderborn)
 - Scotch (U. Bordeaux)
- Parallel:
 - Zoltan (SNL)
 - ParMETIS (U. Minn.)
 - PJostle (U. Greenwich)



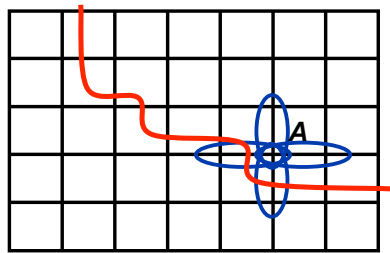
• Disadvantages:

- More expensive than geometric methods.
- Edge-cut model only approximates communication volume.

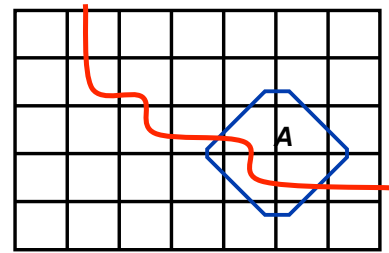


Hypergraph Partitioning

- Alpert, Kahng, Hauck, Borriello, Çatalyürek, Aykanat, Karypis, et al.
- Hypergraph model:
 - Vertices = objects to be partitioned.
 - Hyperedges = dependencies between two or more objects.
- Partitioning goal: Assign equal vertex weight while minimizing hyperedge cut weight.



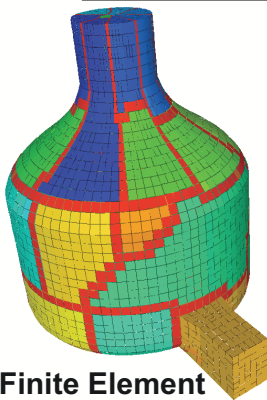
Graph Partitioning Model



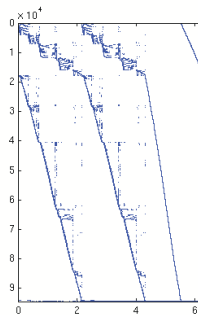
Hypergraph Partitioning Model



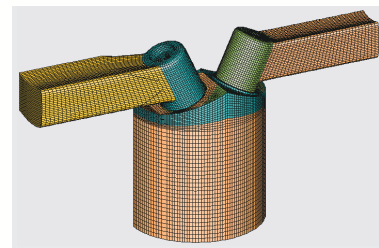
Hypergraph Applications



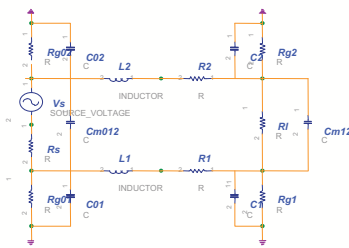
Finite Element Analysis



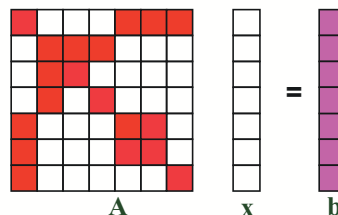
Linear programming for sensor placement



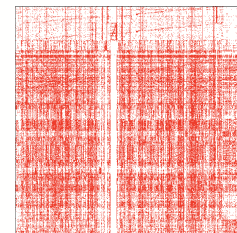
Multiphysics and multiphase simulations



Circuit Simulations



Linear solvers & preconditioners (no restrictions on matrix structure)



Data Mining



Hypergraph Partitioning: Advantages and Disadvantages



- **Advantages:**
 - Communication volume reduced 30-38% on average over graph partitioning (Catalyurek & Aykanat).
 - 5-15% reduction for mesh-based applications.
 - More accurate communication model than graph partitioning.
 - Better representation of highly connected and/or non-homogeneous systems.
 - Greater applicability than graph model.
 - Can represent rectangular systems and non-symmetric dependencies.
- **Disadvantages:**
 - More expensive than graph partitioning.



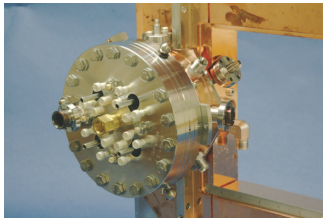
Performance Results



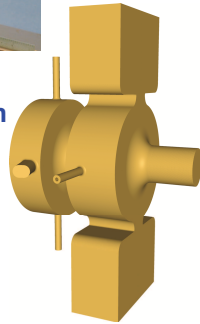
- **Experiments on Sandia's Thunderbird cluster.**
 - Dual 3.6 GHz Intel EM64T processors with 6 GB RAM.
 - Infiniband network.
- **Compare RCB, SFC, graph (ParMETIS) and hypergraph methods.**
- **Measure ...**
 - Amount of communication induced by the partition.
 - Partitioning time.



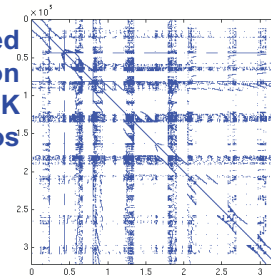
Test Data



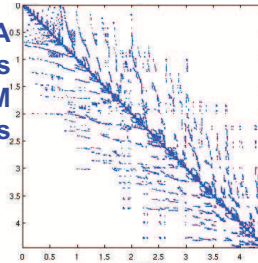
**SLAC *LCLS
Radio Frequency Gun**
6.0M x 6.0M
23.4M nonzeros



**Xyce 680K ASIC Stripped
Circuit Simulation**
680K x 680K
2.3M nonzeros



**Cage15 DNA
Electrophoresis**
5.1M x 5.1M
99M nonzeros

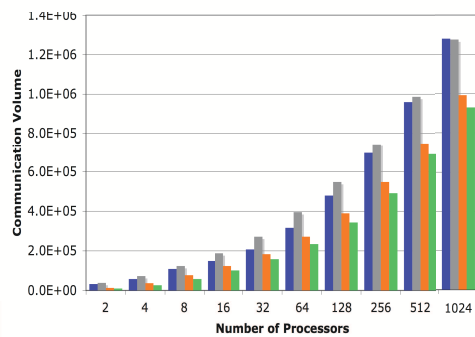


SLAC Linear Accelerator
2.9M x 2.9M
11.4M nonzeros



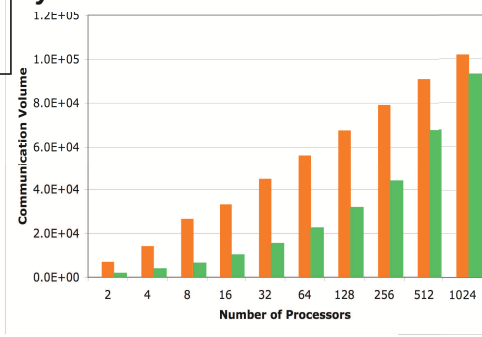
Communication Volume: Lower is Better

SLAC 6.0M LCLS

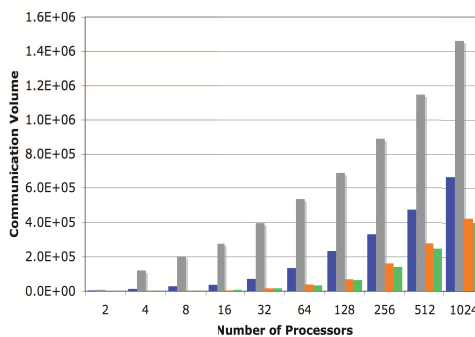


*Number of parts
= number of
processors.*

Xyce 680K circuit

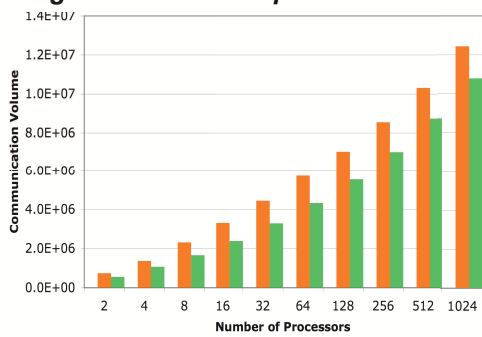


SLAC 2.9M Linear Accelerator



■ RCB
■ SFC
■ Graph
■ Hypergraph

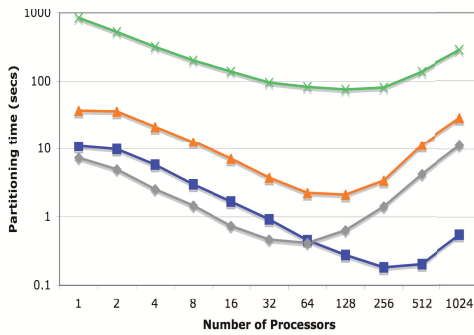
Cage15 5.1M electrophoresis



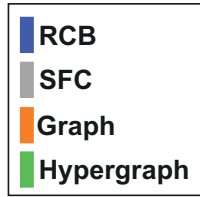


Partitioning Time: Lower is better

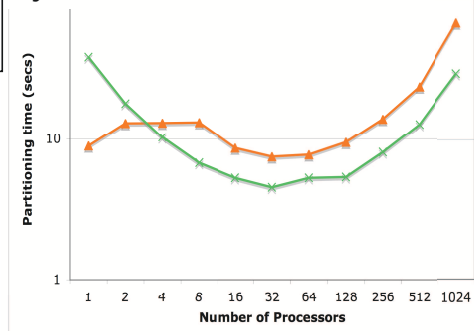
SLAC 6.0M LCLS



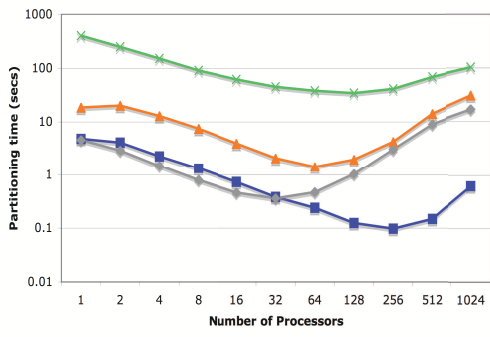
1024 parts.
Varying number of processors.



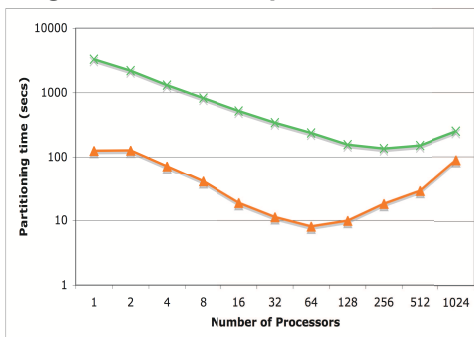
Xyce 680K circuit



SLAC 2.9M Linear Accelerator



Cage15 5.1M electrophoresis



Repartitioning Experiments

- Experiments with 64 parts on 64 processors.
- Dynamically adjust weights in data to simulate, say, adaptive mesh refinement.
- Repartition.
- Measure repartitioning time and total communication volume:

$$\text{Data redistribution volume} + \text{Application communication volume} = \text{Total communication volume}$$

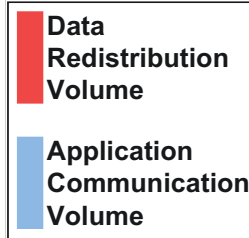
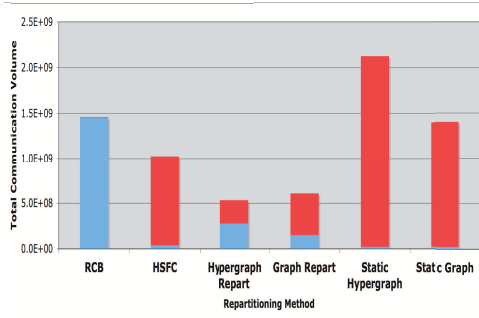
Best Algorithms Paper Award at IPDPS07
 "Hypergraph-based Dynamic Load Balancing for Adaptive Scientific Computations"
 Catalyurek, Boman, Devine, Bozdag, Heaphy, & Riesen



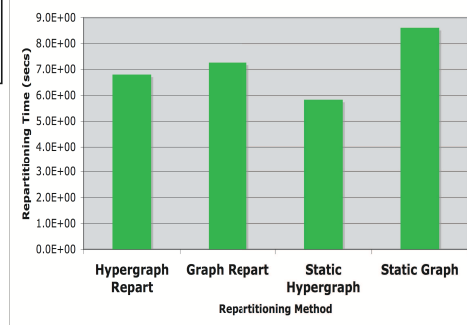
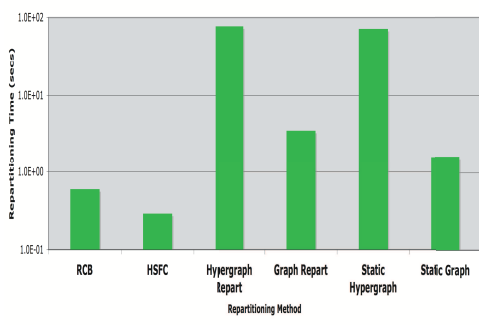
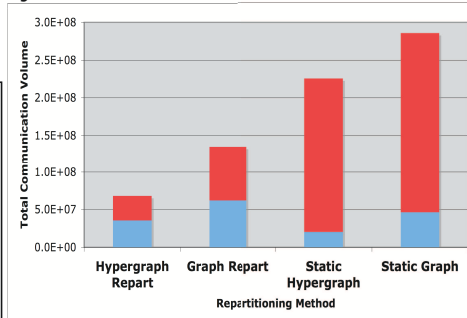
Repartitioning Results: Lower is Better



SLAC 6.0M LCLS



Xyce 680K circuit



Zoltan Toolkit: Suite of Partitioners



- **No single partitioner works best for all applications.**
 - Trade-offs:
 - Quality vs. speed.
 - Geometric locality vs. data dependencies.
 - High-data movement costs vs. tolerance for remapping.
- **Application developers may not know which partitioner is best for application.**
 - Suite of partitioners allows experimentation, comparisons.