Combinatorial Algorithms for Parallel Sparse Matrix Distribution

Erik Boman

Sandia National Laboratories, NM, USA

PMAA, Rennes, France, Sept. 2006.

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.

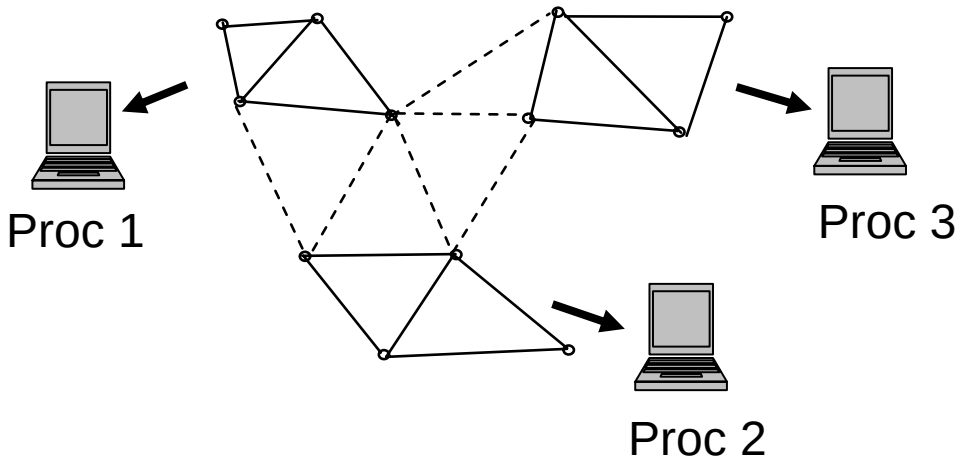
Sandia National Laboratories

Outline

- Load-balancing & partitioning
- Flaws in traditional approach
- Sparse matrix-vector multiplication
- Cost models: graph, hypergraph
- 1d and 2d distributions
- Software
- Results
- Open problems & Future work

Load Balancing, Graph Partitioning

- Load Balancing
  - Assign work to processors to distribute work evenly and minimize communication
- Graph partitioning
  - Vertices (weighted) = computation
  - Edge (weighted) = data dependence



Flaws in Traditional Models

- Flaw 1: Edge cut = comm. volume = comm. cost
  - Graph partitioning: edge cuts do NOT accurately represent communication volume
  - Communication volume is NOT the cost
    - latency, #messages
- Flaw 2: Single, known computational weight
  - Real world: Multiphase simulations
    - Need multiple weights, dependencies
  - Total work is NOT always a linear sum
    - Computation may depend on parallel distribution

Does it Matter?

- Mesh-based applications: No, not much
  - Simple graph partitioning works OK
  - Geometric structure ensures
    - Small separators and good partitions
    - Low vertex degrees give small error in graph model

- Irregular applications: Yes
  - Graph model is poor, need better model
  - Ex: circuit simulation, non-pde optimization, data mining
  - Nonsymmetric and rectangular matrices

Parallel Sparse Matrix-Vector Multiply

- Compute y=Ax, where A is large and sparse
  - A is distributed among processors
- Kernel in scientific computing
  - Iterative methods (Ax=b)
  - Eigenvalue computations
    - Google's PageRank
- Nice model problem
  - Many other applications have similar computation and communication pattern

Parallel Sparse Matrix-vector Algorithm

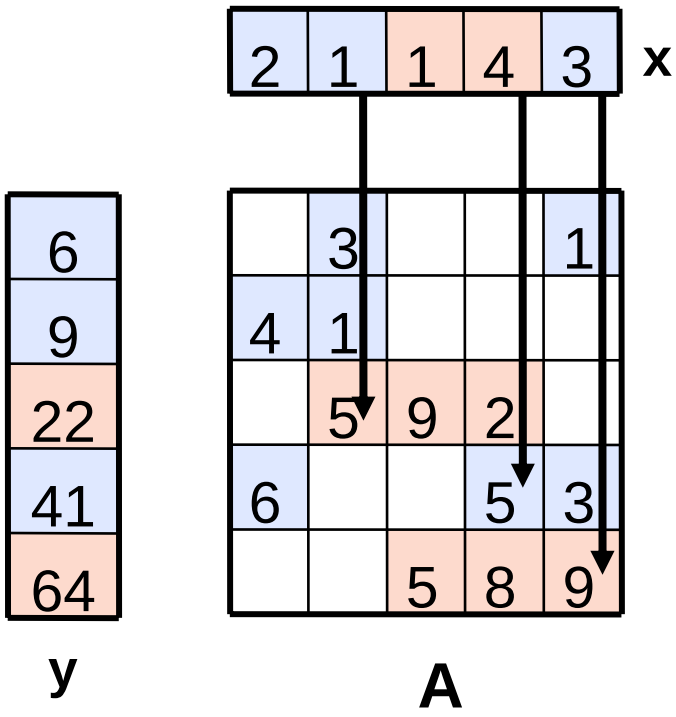
- Step 1: Fan-out
  - Send x_j to processors with nonzero in column j
- Step 2: Local multiply
  - y_i += A_{ij} x_j
- Step 3: Fan-in
  - Send partial results of y to relevant processors
- Step 4: Accumulate partial results

Two communication phases,

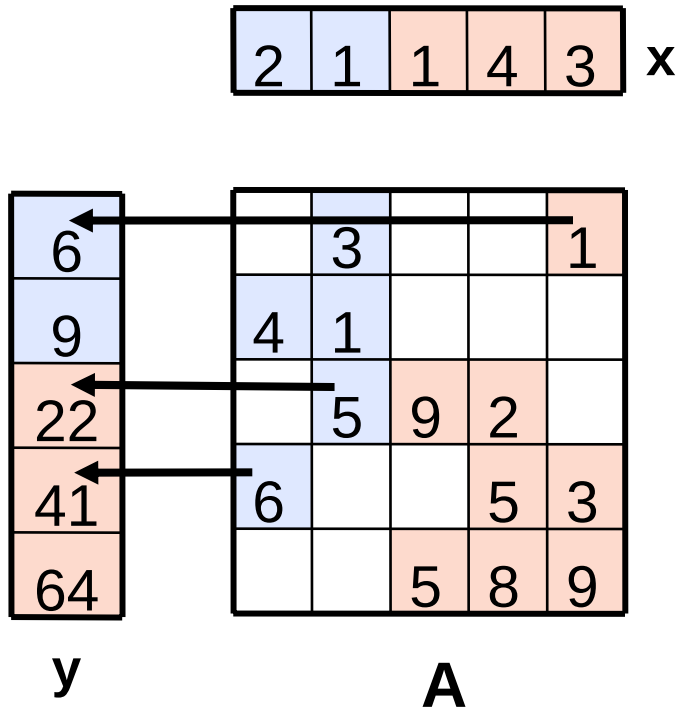
but often only need one.

Communication (p=2)

- Row distribution



- Column distribution



Combinatorial Models

- Models to represent a sparse matrix and 1d partitioning
  - Graph
  - Bipartite graph
  - Hypergraph
- Want accurate model for communication in parallel computing
  - communication volume (for now)

Graph Partitioning

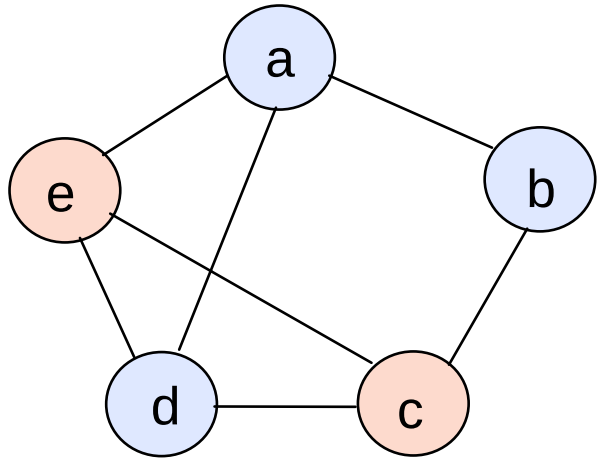
- Suppose A is a symmetric matrix
- Let G=(V,E) be the graph of A
- Partition the vertices V into k equal sets
  - Such that the number of cut edges is minimized
  - Optional weights on vertices and edges
- Widely used model, but
  - Does not accurately reflect communication volume
  - Requires symmetry (or symmetrization)

Graphs: Edge Cut Flaw

- Original matrix
- Comm volume = 3

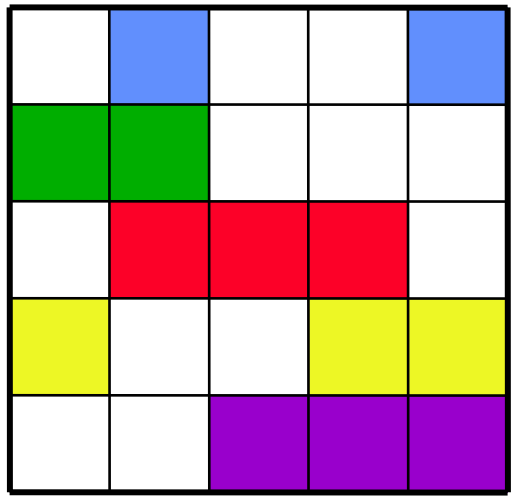
a		3			1
b	4	1			
c		5	9	2	
d	6			5	3
e			5	8	9

- Symmetrized graph
- Edge cut = 4

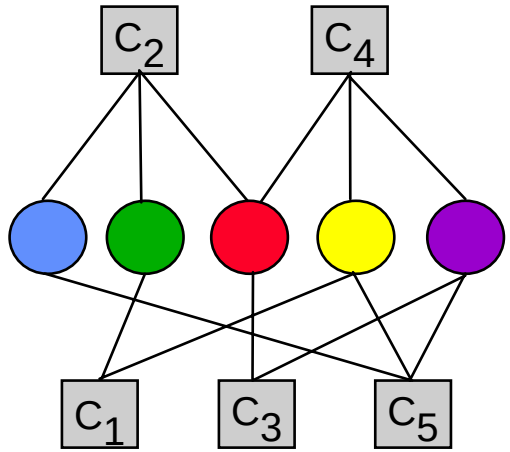


Bipartite Graph Model

- G=(R,C,E), where
  - R are row vertices
  - C are column vertices
- Partition both R and C
  - Minimize edge cut
  - But only use R for row distribution

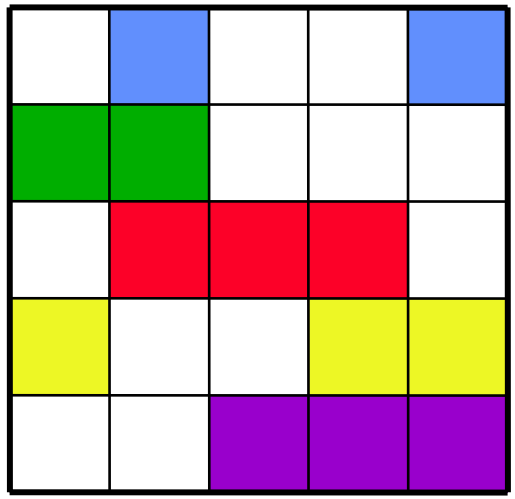


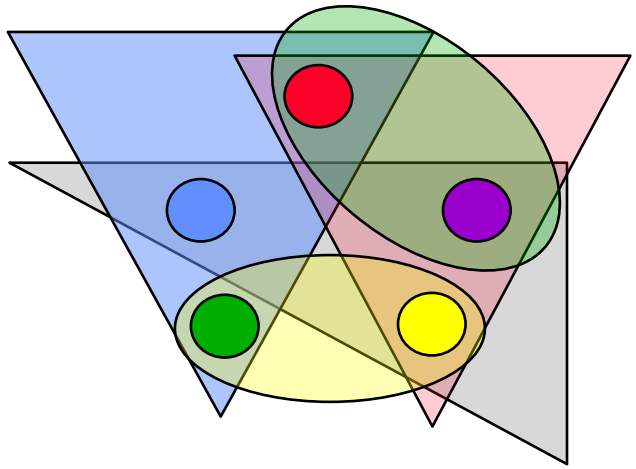
- Works in nonsym. case
- Edge cut approximates comm. volume
  - Is NOT exact



Hypergraph Model

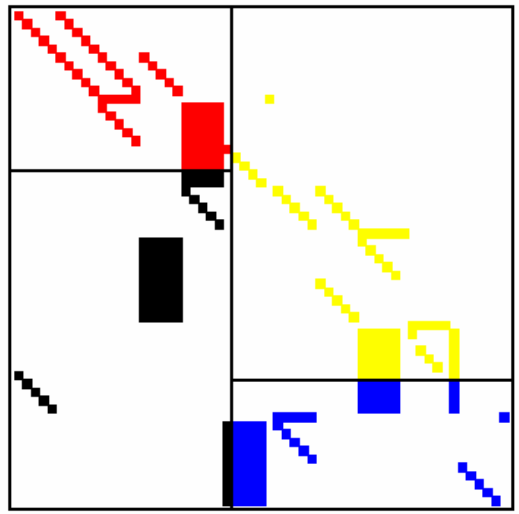
- Hypergraph
  - Hyperedge is a set of vertices (1 or more)
  - Rows = vertices
  - Columns = hyperedges
- Partition
  - Minimize hyperedges cut
- Edge cut is exactly comm volume
  - Aykanat & Catalyurek ('96, ’99)





2D Sparse Matrix Partitioning

- 2D partitioning methods
  - More flexibility
  - Reduces communication volume further
- 2D Cartesian (checkerboard)
  - First partition rows
  - Then partition columns
    - Via multiconstraint hypergraph partitioning
    - Catalyurek & Aykanat ('01)
  - Hard to get good balance
- 2D Mondriaan
  - Recursive hypergraph bisection
  - Bisseling & Vastenhouw ('04)



Courtesy: Rob Bisseling

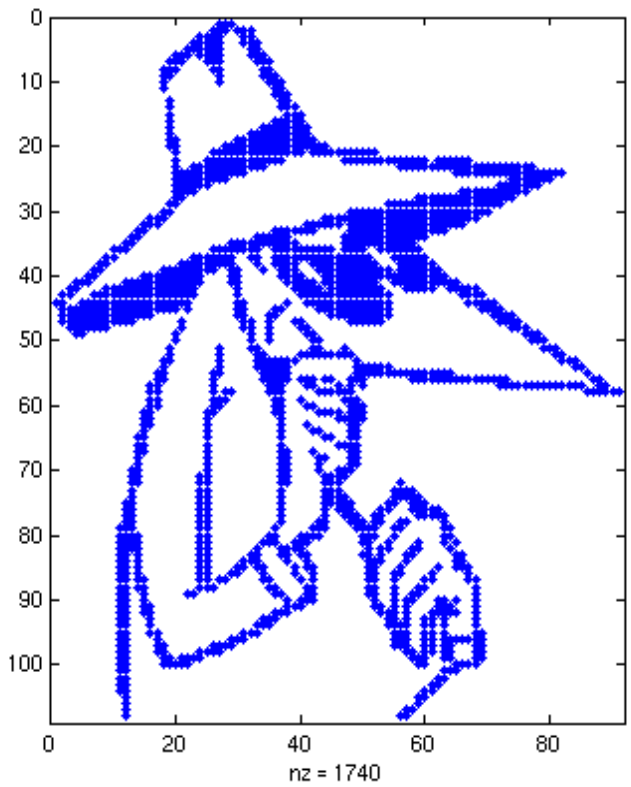
Fine-Grain Matrix Partitioning

- Fine-grain partitioning
  - Assign each nonzero separately
  - Ultimate flexibility

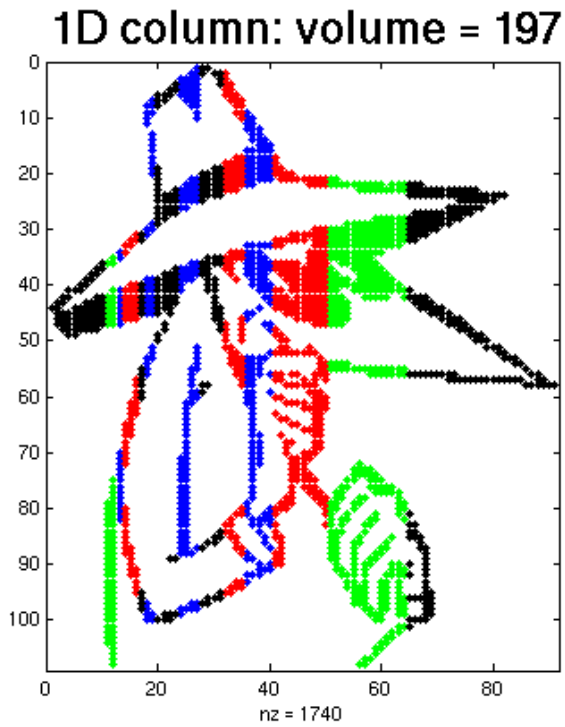
- Fine-grain hypergraph model
  - Each nonzero is a vertex
  - Each row and column is a hyperedge
  - Larger hypergraph than 1d model
  - Exact model for comm. volume
    - Catalyurek & Aykanat '01

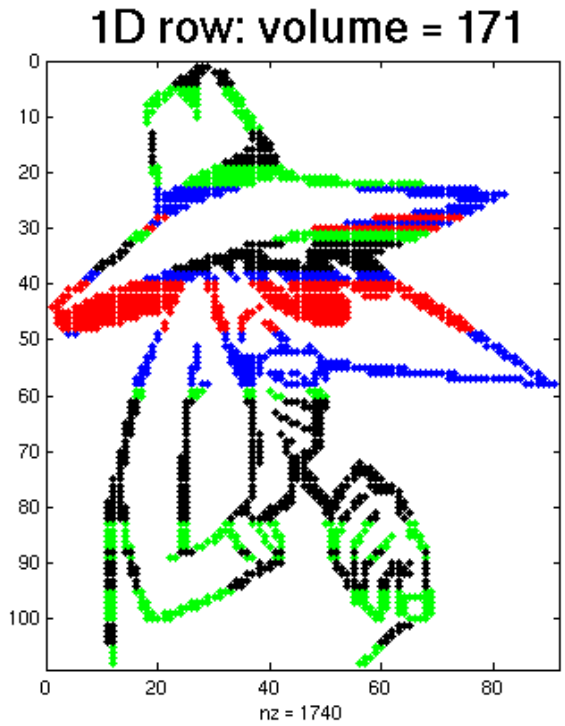
Matrix Example

- Spy matrix
- Partition into p=4
- Balance nonzeros
- Compare
  - 1D column
  - 1D row
  - 2D Mondriaan
  - 2D fine-grain

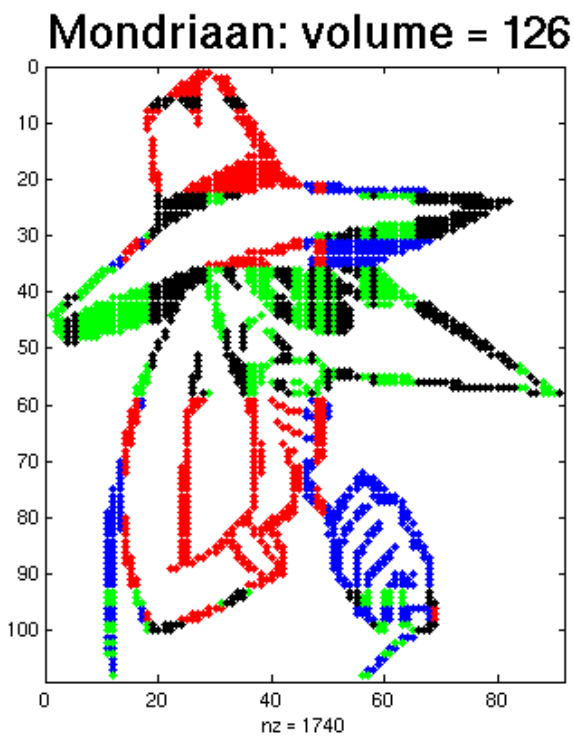


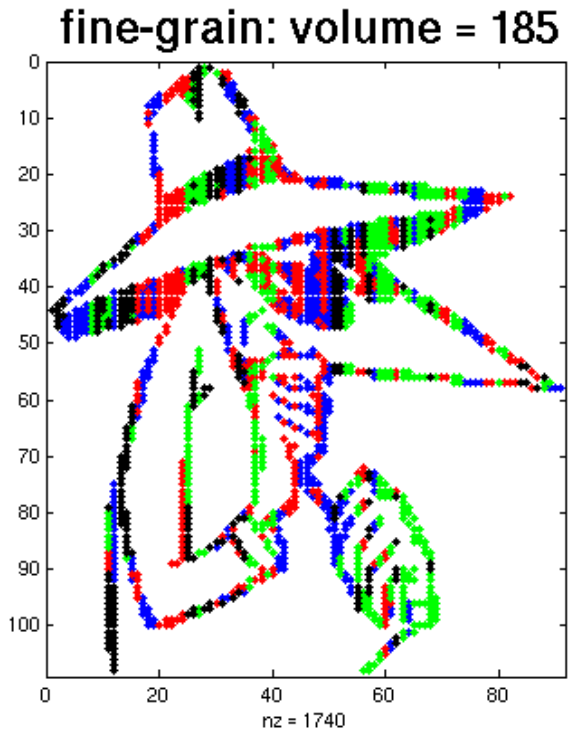
Example, 1D


1D column: volume = 197
0
10
20
30
40
50
60
70
80
90
100
0
20
40
60
80
nz = 1740


1D row: volume = 171
0
10
20
30
40
50
60
70
80
90
100
0
20
40
60
80
nz = 1740

Example, 2D


Mondriaan: volume = 126
0
10
20
30
40
50
60
70
80
90
100
0
20
40
60
80
nz = 1740

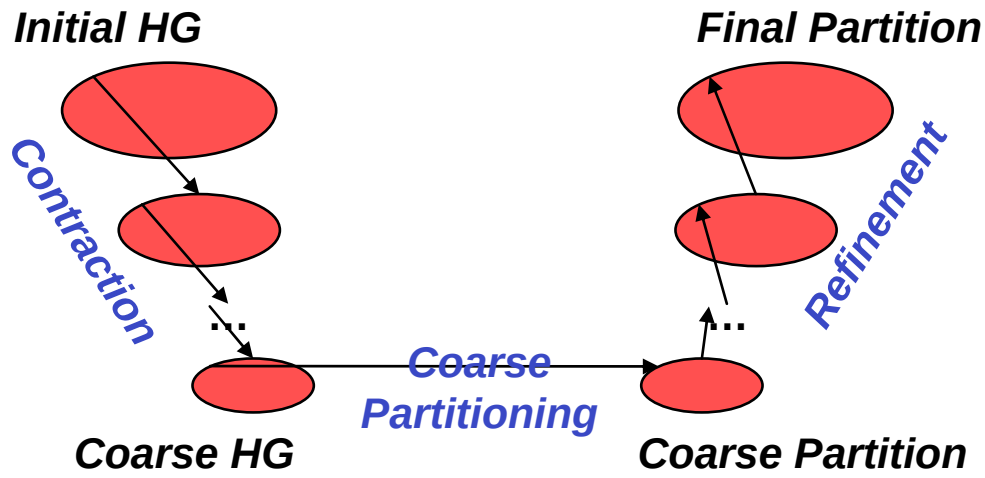

fine-grain: volume = 185
0
10
20
30
40
50
60
70
80
90
100
0
20
40
60
80
nz = 1740

Algorithms for Partitioning

- NP-hard problem (graph and hypergraph)
- Good heuristics available
- Multilevel methods most effective
  - Graph partitioning
    - Bui&Jones, Hendrickson&Leland, Karypis&Kumar
  - Hypergraph partitioning
    - Aykanat &Catalyurek, Karypis
    - More expensive than graph partitioning

Multilevel Scheme

- **Multilevel partitioning (graph or hypergraph)**
  - **Contraction**: reduce HG to smaller representative HG.
  - **Coarse partitioning**: assign coarse vertices to partitions.
  - **Refinement**: improve balance and cuts at each level.



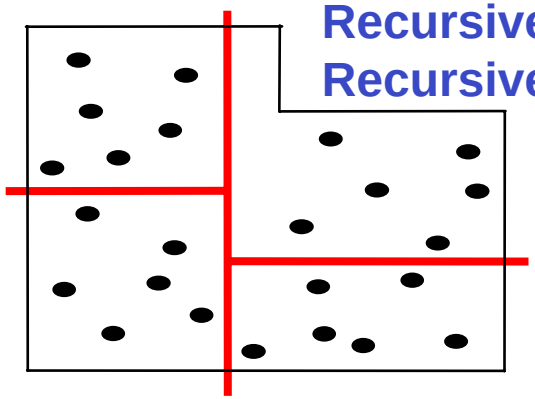
Multilevel Partitioning V-cycle

Software

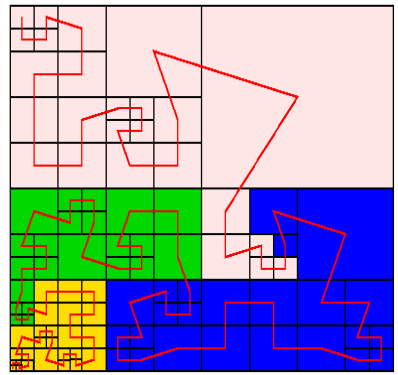
	Serial	Parallel
Graph partitioner	Chaco, Metis, Jostle, Scotch	ParMetis, PJostle
Hypergraph partitioner	hMetis, PaToH, Mondriaan	Parkway, Zoltan

Zoltan Toolkit: Suite of Partitioning Algorithms

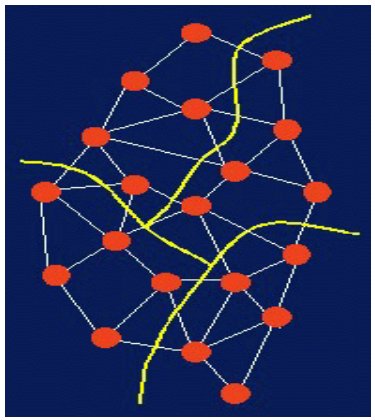
Sandia National Laboratories



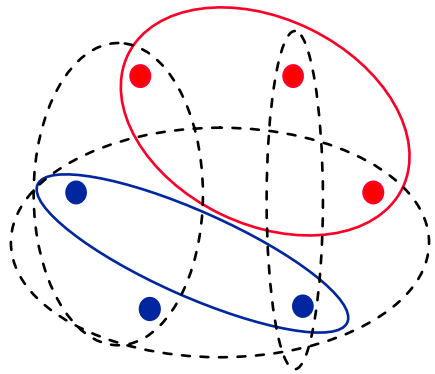
Recursive Coordinate Bisection
Recursive Inertial Bisection



Space Filling Curves
Refinement-tree Partitioning
Octree Partitioning



Graph Partitioning
ParMETIS , Jostle



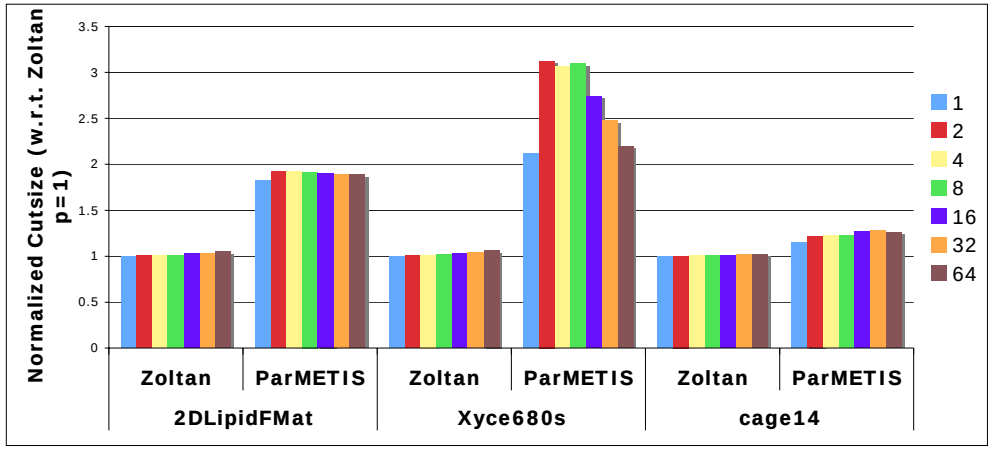
Hypergraph Partitioning
NEW!

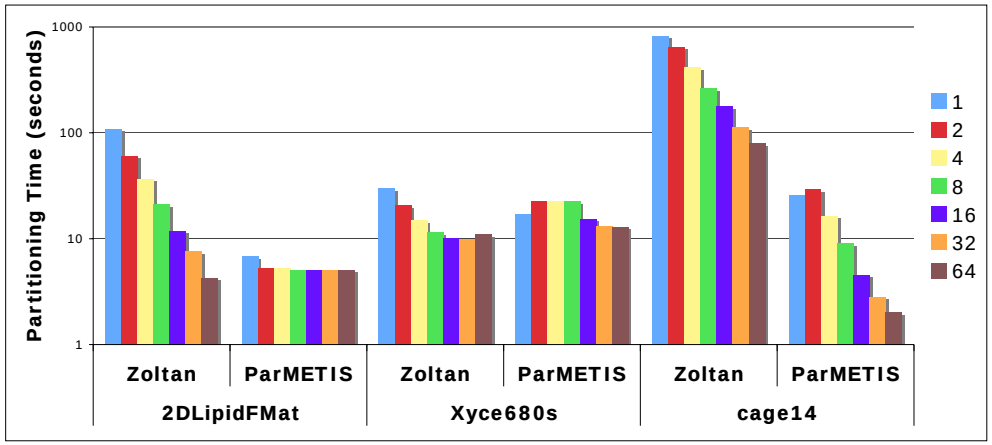
Zoltan

- Zoltan 2.0
  - Available now: www.cs.sandia.gov/Zoltan
  - Open source (LGPL)
- Parallel hypergraph partitioner
  - One of several packages within Zoltan
  - Multilevel method (like hMetis & Patoh)
  - Designed for large-scale problems
  - Parallel, distributed memory (MPI)
- Future features
  - Repartitioning
  - 2D matrix partitioning

Partitioning Results

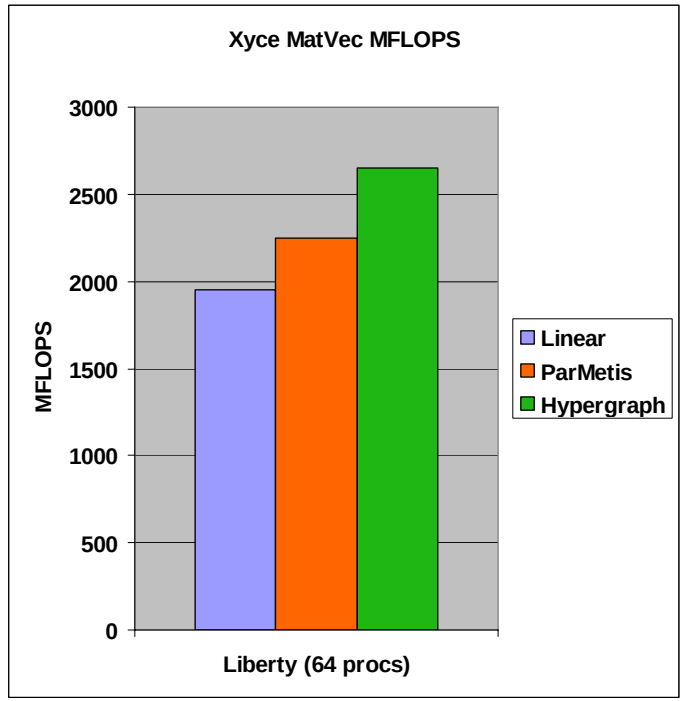
- Zoltan hypergraph partitioner vs. ParMetis graph partitioner
  - 3 applications:
    - Polymer DFT
    - Circuit simulation
    - DNA electrophoresis
  - 64 partitions; 1-64 processors
  - Communication volume (hyperedge cut) is up to three times lower with hypergraph partitioning





Application Results

- Parallel matrix-vector performance for two Sandia applications:
  - Tramonto (polymer DFT) and Xyce (circuit simulation)
- Zoltan hypergraph partitioning gave highest MFLOPS
  - 13-18% improvement over graph partitioning



Results courtesy of Mike Heroux, SNL.

Prior Results in the Field

- Hypergraph vs. graph
  - 5-10% volume reduction for mesh-based applications
  - 30-40% on rectangular or irregular matrices (LP)

- 2D vs 1D
  - 5-50% volume reduction for Mondriaan vs 1D
    - Small difference for FEM/mesh matrices
    - Big improvement on rectangular LP matrices
  - Fine-grain is similar

Dynamic Load-Balancing

- Adaptive simulations
  - Matrix structure changes over time
  - Usually small perturbation
- Repartitioning: 3 goals
  - Load balance
  - Small communication cost (edge cut)
  - Small migration cost
    - New partitions similar to old partitions
  - Complex trade-offs; many heuristics
    - Partially implemented in Zoltan; hypergraph soon

Recent Research: Communication Cost Models

- Minimize #messages (and keep volume low)
  - Message cost = $\alpha + \beta$ *length
  - Hypergraph model insufficient
- Minimize time for slowest processor
  - Balance communication cost among processors
  - Multiple communication costs (objectives)
  - Solve as two hypergraph problems (Ucar & Aykanat, ‘04)
  - Vector partitioning (Bisseling, ’05)

Open Problems: Partitioning for Preconditioning

Sandia National Laboratories

- Typical set-up:
  - Partition matrix (mesh)
  - Form local preconditioner on each subdomain
  - Possibly multilevel scheme
- Preconditioner depends on partitioning
  - Must take numerical properties into account
    - Matrix entries = edge weights?
    - Talk by Sosonkina, Saad, Ucar
  - Computation depends on partitioning
    - Ex: Domain decomposition, direct or ILU on subdomains

Complex Objectives

- Partitioning objective can only be evaluated after the partitioning is known
- Algorithm: (Pinar & Hendrickson ’01)
  - Partition for a simplified objective
  - Evaluate real objective. Update weights.
  - Repartition. Repeat until good balance & cost.
- Also called Predictor-Corrector method
  - Moulitsas & Karypis ('05)
  - Domain decomposition (FETI)
    - Balance fill in each subdomain

Conclusions

- Graph partitioning is not the answer to everything
- Hypergraph model and software available
  - Big improvement for some applications
- 2D data decompositions useful for sparse data
  - But can software (applications) handle this?
- Still many unsolved problems!

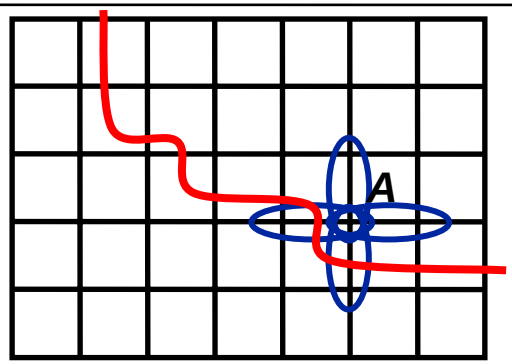
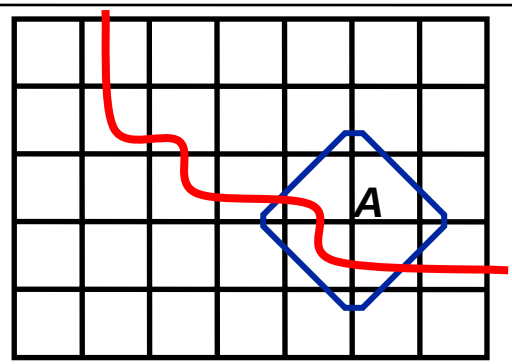
Thanks

- Collaborators:
  - Karen Devine (Sandia)
  - Bruce Hendrickson (Sandia)
  - Umit Catalyurek (Ohio State)
  - Rob Bisseling (Utrecht)
- Zoltan software:
  - www.cs.sandia.gov/Zoltan
- CSC Workshop
  - Attached to SIAM CS&E, Costa Mesa, Feb. 2007

Sandia National Laboratories

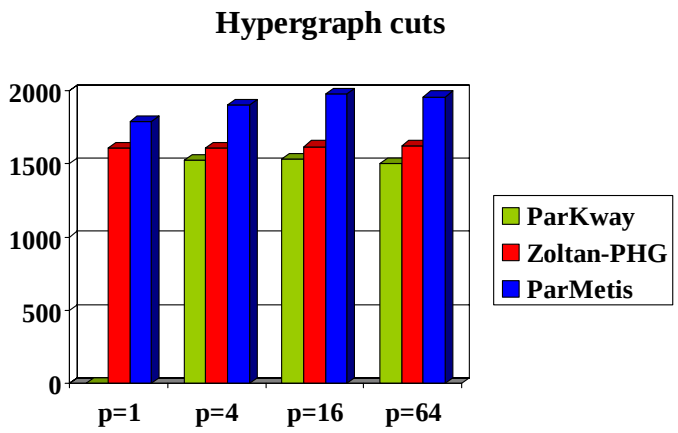
Graph Partitioning vs. Hypergraph Partitioning

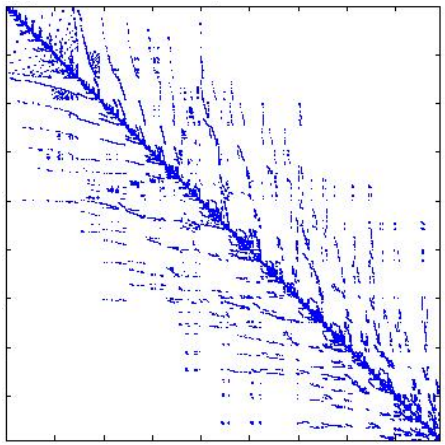
Sandia National Laboratories

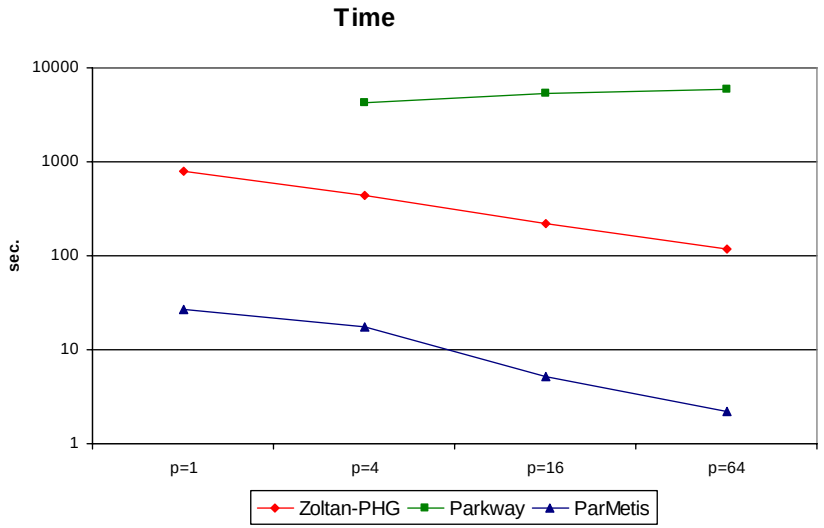
Graph Partitioning Kernighan, Lin, Schweikert, Fiduccia, Mattheyes, Pothen, Simon, Hendrickson, Leland, Kumar, Karypis, et al.	Hypergraph Partitioning Kernighan, Alpert, Kahng, Hauck, Borriello, Aykanat, Çatalyürek, Karypis, et al.
Vertices: computation.	Vertices: computation.
Edges: two vertices.	Hyperedges: two or more vertices.
Edge cuts approximate communication volume.	Hyperedge cuts accurately measure communication volume.
Assign equal vertex weight while minimizing edge cut weight.	Assign equal vertex weight while minimizing hyperedge cut weight.
	

Results

- **Cage14: Cage model of DNA electrophoresis (van Heukelum)**
  - **1.5M rows & cols; 27M nonzeros.**
  - **Symmetric structure**
  - **64 partitions.**
- **Hypergraph partitioning reduced communication volume by 10-20% vs. graph partitioning.**

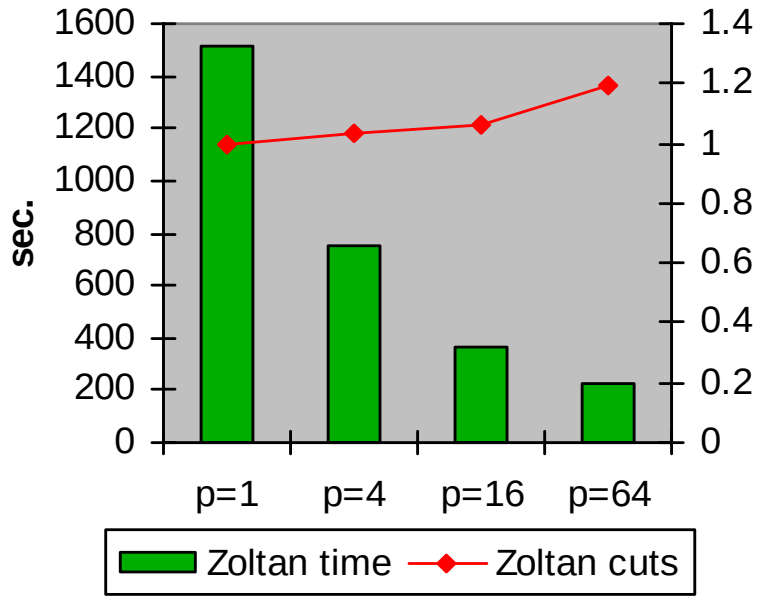




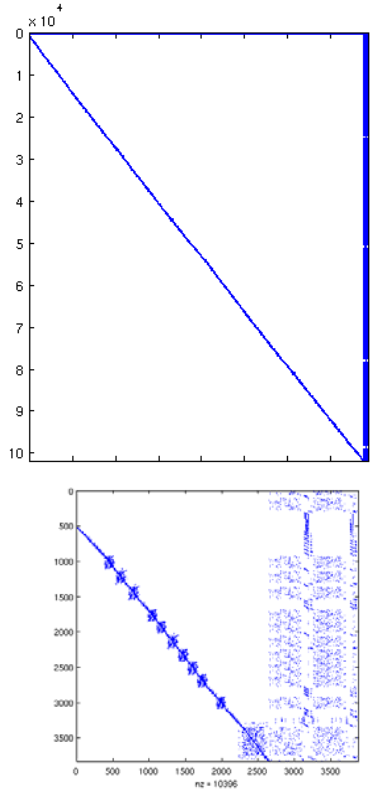


More Results

- **Sensor placement – IP/LP model**
  - **5M rows, 4M columns**
  - **16M nonzeros**



- **ParKway ran out of memory**
  - **1d with ghosting, not**



1-d matrix distribution

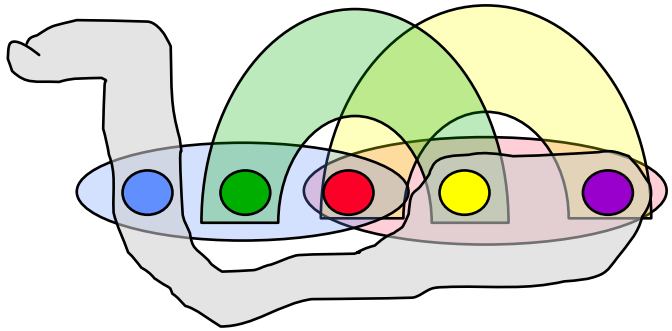
- Partition matrix along rows

x		x			x
	x		x		
x					x
		x		x	
			x	x	x
	x	x		x	
	x			x	

- After row and column permutations

x	x	x			
x		x			
	x		x		
	x		x	x	
		x	x	x	
			x		x
				x	x

- Turtle Hypergraph



- Line Hypergraph