# Partitioning, ordering, and load balancing in a hierarchically parallel hybrid linear solver

Ichitaro Yamazaki
University of Tennessee, Knoxville
**Xiaoye Li**
Lawrence Berkeley National Laboratory
Francois-Henry Rouet
ENSEEIHT-IRIT and LIP
Bora Ucar
ENS-Lyon

Sparse Days, June 25th-26th, 2012, CERFACS, Toulouse

# OUTLINE OF THE TALK

- **Overview of DD Schur complement method**
  - Need for hierarchical parallelism
  - **PDSLin** (Parallel Domain decomposition Schur complement Linear solver)

    Download: http://crd-legacy.lbl.gov/FASTMath-LBNL/Software/
- **Combinatorial problems in hybrid sparse solvers**
  - K-way graph partitioning with multiple constraints
  - Sparse triangular solution with sparse right-hand sides

# Schur complement method

- **a.k.a iterative substructuring method**

  **or, non-overlapping domain decomposition**

- **Divide-and-conquer paradigm . . .**
  - Divide entire problem (domain, graph) into subproblems (subdomains, subgraphs)
  - Solve the subproblems
  - Solve the interface problem (Schur complement)

- **Variety of ways to solve subdomain problems and Schur complement … lead to powerful poly-algorithm or hybrid solver framework**

# Algebraic view

1. **Reorder into 2x2 block system, $A_{11}$ is block diagonal**

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

2. **Schur complement**

$$S = A_{22} - A_{21} A_{11}^{-1} A_{12} = A_{22} - (U_{11}^{-T} A_{21}^{T})^{T} (L_{11}^{-1} A_{12}) = A_{22} - W \cdot G$$

where $A_{11} = L_{11} U_{11}$

**S corresponds to interface (separator) variables**

**no need to form explicitly**

3. **Compute the solution**

(1) $x_2 = S^{-1}(b_2 - A_{21} A_{11}^{-1} b_1)$ ← iterative solver

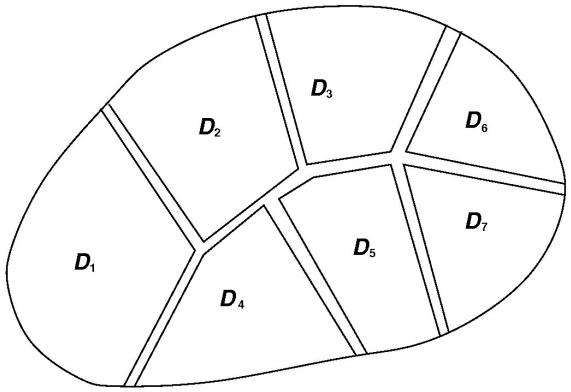
(2) $x_1 = A_{11}^{-1}(b_1 - A_{12} x_2)$ ← direct solver

# Parallelism – extraction of many subdomains

- **Partition adjacency graph of $|A|+|A^T|$**

  **Multiple goals: reduce size of separator, balance size of subdomains**

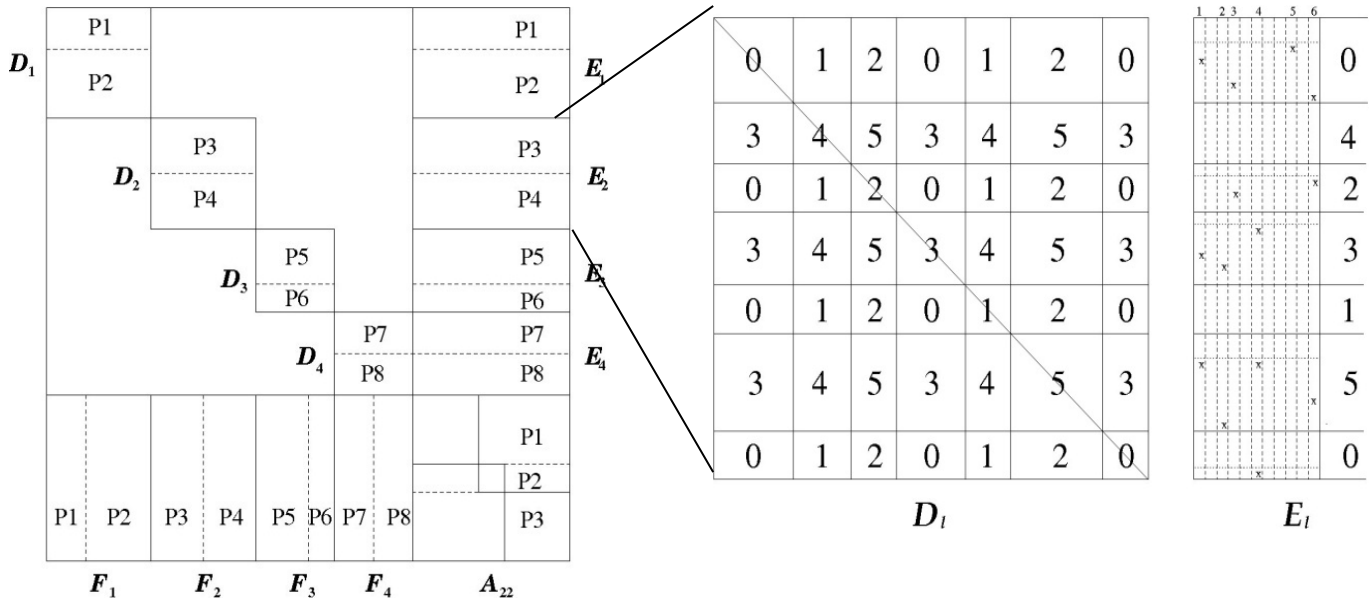
  - nested dissection (e.g., PT-Scotch, ParMetis)
  - k-way partition (preferred)




$$\left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right) = \left(\begin{array}{cccc|c} D_1 & & & & E_1 \\ & D_2 & & & E_2 \\ & & \ddots & & \vdots \\ & & & D_k & E_k \\ \hline F_1 & F_2 & \cdots & F_k & A_{22} \end{array}\right)$$

# Hierarchical parallelism

- **Multiple processors per subdomain**
  - E.g. one subdomain with 2x3 procs (e.g. SuperLU_DIST, MUMPS)




- **Load balance, communication**
  - Distinction between intra-group and inter-group
  - Intra-group relies on SuperLU_DIST, inter-group new design

# PDSLin software

- **Download: http://crd-legacy.lbl.gov/FASTMath-LBNL/Software/**
  - C and MPI, with Fortran interface.
  - Unsymmetric and symmetric, real and complex, multiple RHSs.
- **Requires following external packages:**
  - parallel graph partitioning:
    - **PT-Scotch (http://labri.fr/perso/pelegrin/scotch), or**
    - **ParMetis (http://glaros.dtc.umn.edu/gkhome/views/metis)**
  - subdomain solver options:
    - **SuperLU (http://crd.lbl.gov/~xiaoye/SuperLU)**
    - **SuperLU_DIST or SuperLU_MT: two levels of parallelization**
    - **MUMPS**
    - **PDSLin**
    - **Inexact subdomain solution: ILU**
  - Schur complement solver options:
    - **PETSc (http://mcs.anl.gov/petsc/petsc-as)**
    - **SuperLU_DIST**

# PDSLin encompass Hybrid, Iterative, Direct

**Hybrid**

Default
Subdomain: LU
Schur: Krylov

**Iterative**

Options
(1) num_doms = 0
Schur = A: Krylov

(2) FGMRES Inner-Outer:
Subdomain: ILU
Schur: Krylov

**Direct**

Options
(1) Subdomain: LU
Schur: LU
drop_tol = 0.0

(2) num_doms = 1
Subdomain: LU

$$\left(\begin{array}{cccc|c} D_1 & & & & E_1 \\ & D_2 & & & E_2 \\ & & \ddots & & \vdots \\ & & & D_k & E_k \\ \hline F_1 & F_2 & \dots & F_k & A_{22} \end{array}\right)$$

# Combinatorial problems . . .

- **K-way graph partitioning with multiple objectives**
  - Small separator
  - Similar subdomains
  - Similar connectivity

- **Sparse triangular solution with many sparse RHS**

$$S = A_{22} - \sum_l (U_l^{-T} F_l^T)^T \, (L_l^{-1} E_l) = A_{22} - \sum_l G \cdot W, \quad \text{where} \quad D_l = L_l U_l$$

- **Sparse matrix–matrix multiplication**

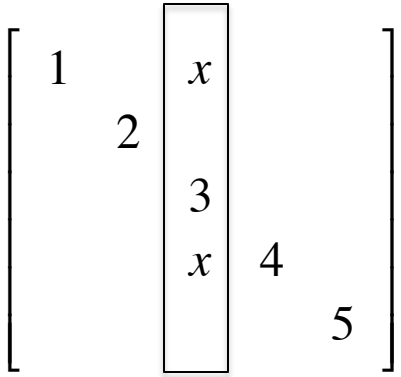
$$\widetilde{G} \leftarrow \text{sparsify}(G, \sigma_1); \quad \widetilde{W} \leftarrow \text{sparsify}(W, \sigma_1)$$

$$T^{(p)} \leftarrow \widetilde{W}^{(p)} \times \widetilde{G}^{(p)}$$

$$\hat{S}^{(p)} \leftarrow A_{22}{}^{(p)} - \sum_q T^{(q)}(p); \quad \widetilde{S} \leftarrow \text{sparsify}(\hat{S}, \sigma_2)$$

# Two graph models

- **Standard graph : G=(V, E)**
  - GPVS: graph partitioning with vertex separator
  - GPES: graph partitioning with edge separator
- **Hypergraph : H = (V, N), net = "edge", may include more than two vertices**
  - Column-net hypergraph: H = (R, C)
    rows = vertices, columns = nets
    $n_3$ = {1, 3, 4}
  - Row-net hypergraph: H = (C, R)
    columns = vertices, rows = nets



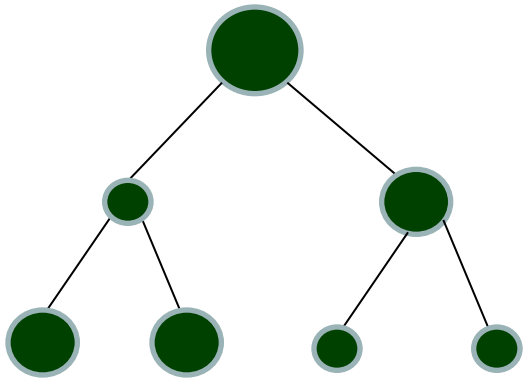

- **Partition problem: $\pi(V) = \{V_1, V_2, \ldots, V_k\}$, disjoint parts**
  - Graph: a cut edge connects two parts
  - Hypergraph: a cut net connects multiple parts

➔ Want to minimize the cutsize in some metric (Objective), and keep equal weights among the parts (Constraints).

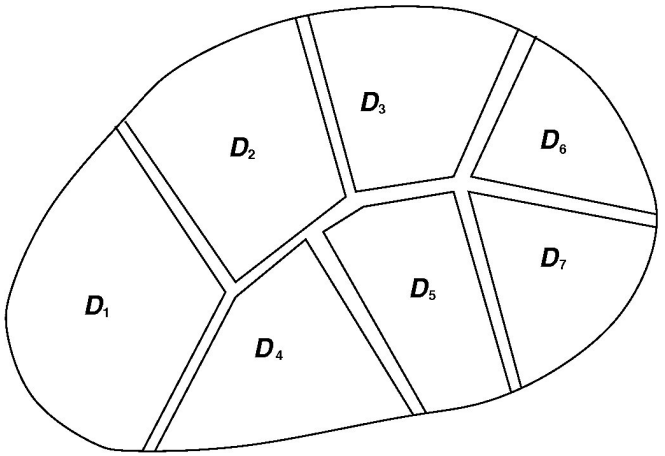
# 1. K-way subdomain extraction

- **Problem with ND:**

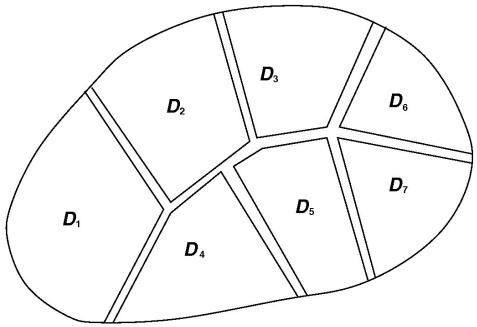
  **Imbalance in separator size at different branches → Imbalance in subdomain size**



- **Alternative: directly partition into K parts, meet multiple constraints:**
  1. **Compute k-way partitioning → balanced subdomains**
  2. **Extract separator → balanced connectivity**



# k-way partition: objectives, constraints




$$\begin{pmatrix} D_1 & & & & E_1 \\ & D_2 & & & E_2 \\ & & \ddots & & \vdots \\ & & & D_k & E_k \\ F_1 & F_2 & \dots & F_k & A_{22} \end{pmatrix}$$

- Initial partition to extract vertex-separator impacts **load balance:**

Objectives to minimize:

- number of interface vertices → separator size
- number of interface edges → nonzeros in interface

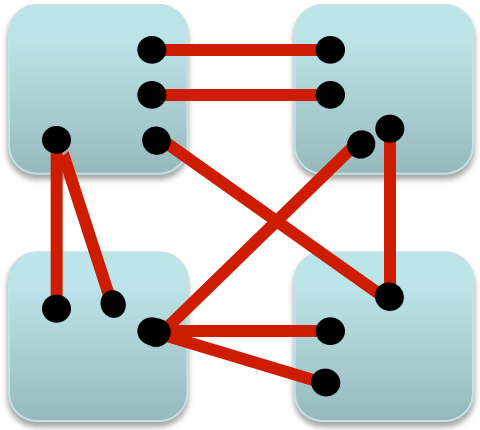
Balance constraints:

- number of interior vertices and edges → LU of $D_i$
- number of interface vertices and edges → local update matrix

$$(U_l^{-T} F_l^T)^T \, (L_l^{-1} E_l)$$

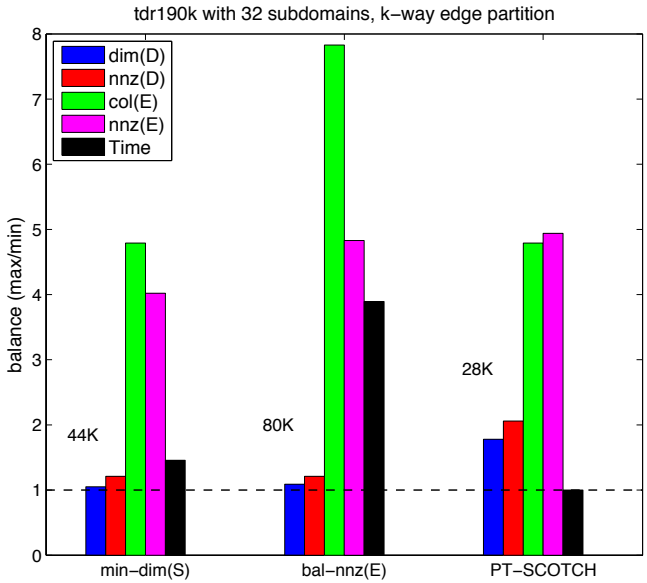
# K-way edge partition

- **Extract vertex-separator from k-way edge partition**
  - Compute k-way edge partition satisfying balanced subdomains (e.g., PT-Scotch, Metis)
  - Extract vertex-separator from edge-separators to minimize and balance interfaces (i.e. minimum vertex cover)
- **Heuristics to pick the next vertex: pick a vertex from largest subdomain to maintain balance among subdomains**
  - pick the vertex with largest degree to minimize separator size
  - pick the vertex to obtain best balance of interfaces (e.g., nnz).



# Balance & Solution time with edge part. + VC

- tdr190k from accelerator cavity design: N = 1.1M, k = 32
- Compared to ND of SCOTCH
- balance = **max value$_\ell$ / min value$_\ell$,** for $\ell$ = 1, 2, . . . , k
- Results
  - Improved balance of subdomains, but not of interfaces due to larger separator → total time not improved
  - Post-processing already-computed partition is not effective to balance multiple constraints



# Recursive Hypergraph Bisection

- Column-net HG partition to permute an m-by-n matrix M to a singly-bordered form (e.g., Patoh):

$$\mathrm{P}_r M P_c^T = \begin{pmatrix} M_1 & & & C_1 \\ & M_2 & & C_2 \\ & & \ddots & \vdots \\ & & M_k & C_k \end{pmatrix} \rightarrow P_c(M^T M)P_c^T = \begin{pmatrix} M_1^T M_1 & & & M_1^T C_1 \\ & M_2^T M_2 & & M_2^T C_2 \\ & & \ddots & \vdots \\ & & M_k^T M_k & M_k^T C_k \\ C_1^T M_1 & C_2^T M_2 & \cdots & \sum C_l^T C_l \end{pmatrix}$$

- Recursive hypergraph bisection (RHB)
  - Compute structural decomposition str(A) = str($M^T M$)
    - e.g., using edge clique cover of G(A) [Catalyurek '09]
  - Compute 2-way partition of M (with multiple constraints) into a singly-bordered form based on recursive bisection
  - Permute A as:

$$\mathrm{str}(\mathrm{P_c A P_c^T}) = \begin{pmatrix} D_1 & & E_1 \\ & D_2 & E_2 \\ F_1 & F_2 & C \end{pmatrix} = \begin{pmatrix} M_1^T & \\ & M_2^T \\ C & C_2^T \end{pmatrix} \bullet \begin{pmatrix} M_1 & & C_1 \\ & M_2 & C_2 \end{pmatrix}$$

# RHB: objectives, constraints

- **Objective: minimize cutsize**

  **cutsize metrics:**

  - Connectivity – 1, $\sum_{nj \in N} (\lambda(j) - 1)$ ( interface nz columns )
  - Cut-net, $\sum_{nj \in N, \lambda(j) > 1} 1$ ( separator size )
  - Sum-of-external degree (soed), $\sum_{nj \in N, \lambda(j) > 1} \lambda(j)$ ( sum of above )

  Where, j-th net $n_j$ is in the cut, and $\boldsymbol{\lambda}_j$ is the number of parts $n_j$ is connected to.

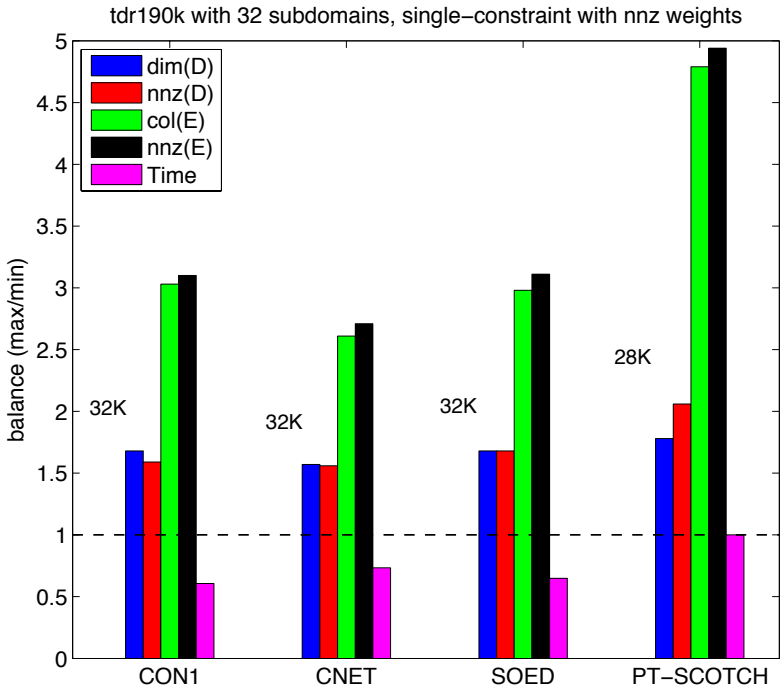
- **Constraints: equal weights of different parts**

  **i-th vertex weights (based on previous partition):**

  - unit weight ( subdomain dimension )
  - Nnz($M_k$(i, :)) ( subdomain nnz )
  - Nnz($M_k$(i, :)) + nnz($C_k$(i, :)) ( interface nnz )

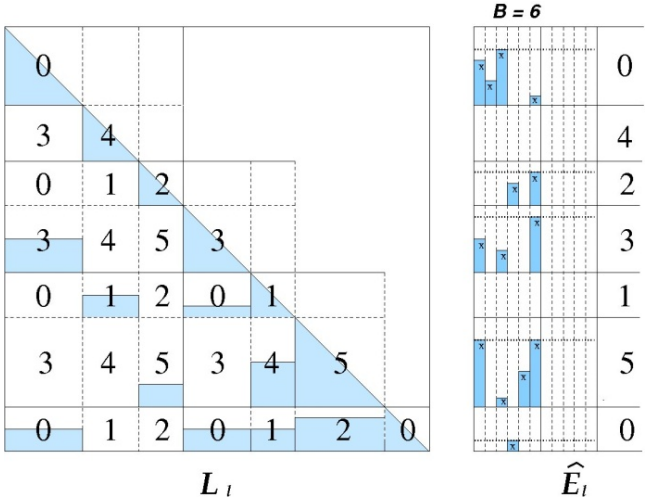
# Balance & Time results with RHB

- tdr190k from accelerator cavity design: N = 1.1M, k = 32
- Compared to ND of SCOTCH
- Results
  - Single-constraint improves balance without much increase of separator size → **1.7x faster than ND**
  - Multi-constraints improves balance, but larger separator



# 2. Sparse triangular solution with sparse RHS (intra-group within a subdomain)

- **RHS vectors $E_\ell$ and $F_\ell$ are sparse (e.g., about 20 nnz per column); There are many RHS vectors (e.g., O($10^4$) columns)**

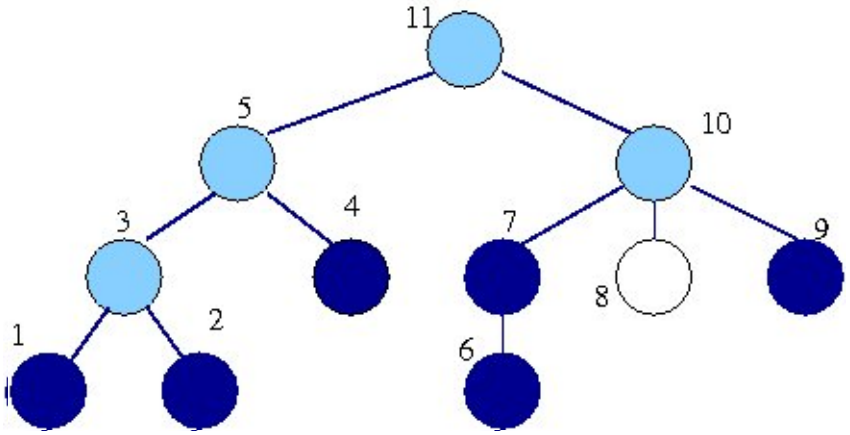


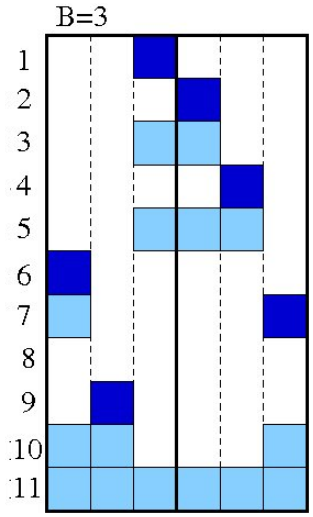

- **Blocking RHS vectors**
  - Reduce number of calls to the symbolic routine and number of messages, and improve read reuse of the LU factors
  - ➢ Achieved over 5x speedup
  - ✖ zeros must be padded to fill the block → memory cost !

# Sparse triangular solution with sparse RHSs

- **Objective: Reorder columns of $E_\ell$ to maximize structural similarity among the adjacent columns.**
- **Where are the fill-ins?**

  **Path Theorem** [Gilbert'94] **Given the elimination tree of $D_l$, fills generated in $G_l$ at the positions associated with nodes on the path from nodes of the nonzeros in $E_l$ to the root**

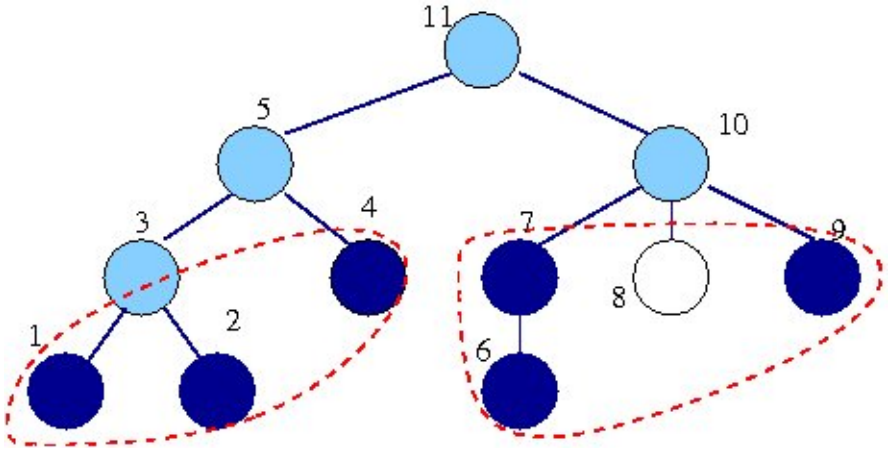


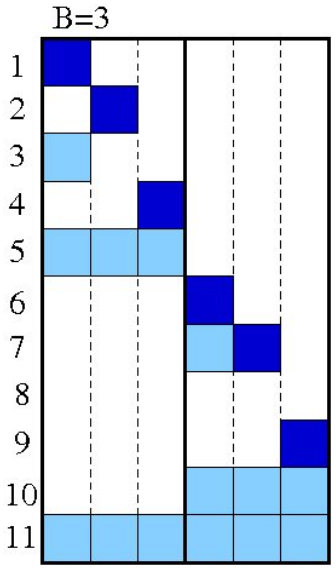




24 padded zeros

# Sparse rhs … postordering

- **Postorder-conforming ordering of the RHS vectors**
  - Postorder elimination tree of $D_l$
  - Permute columns of $E_l$ s.t. row indices of the first nonzeros are in ascending order
- **Increased overlap of the paths to the root, fewer padded zeros**
- **20–40% reduction in solution time over ND**



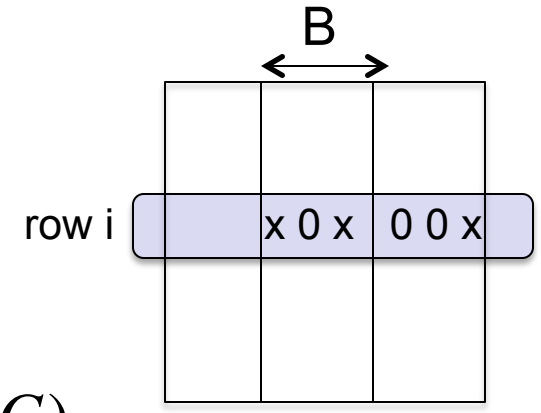

13 padded zeros

# Sparse triangular solution … Hypergraph

- **Partition/group columns using row-net HG**
- **Define a cost function ≈ padded zeros**

$$cost(\Pi) = \sum_{\text{row } i=1}^{n} \left( \lambda_i B - nnz(G_l(i,:)) \right)$$

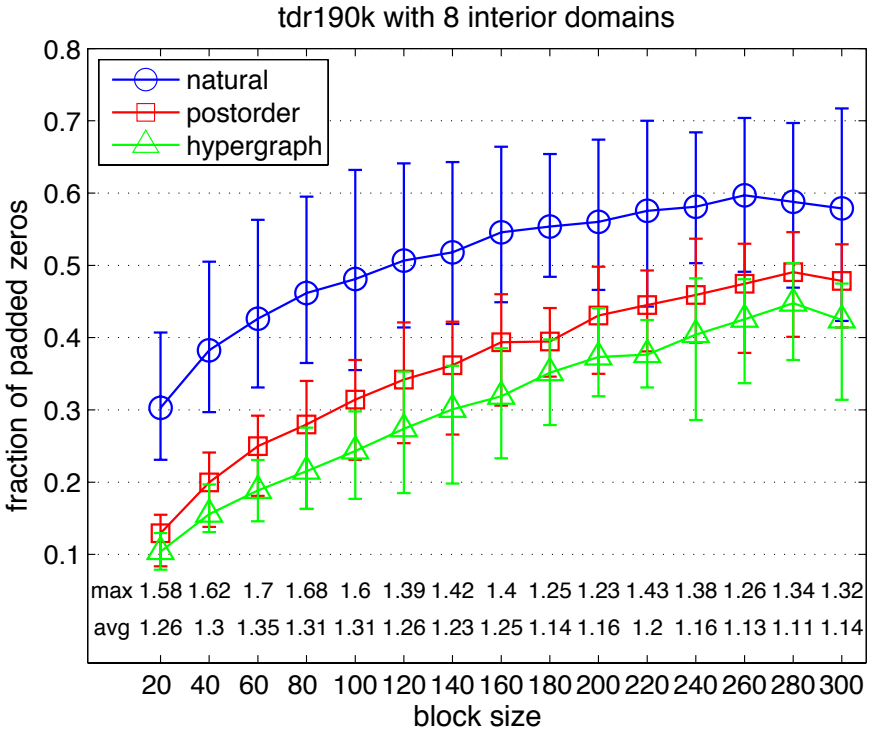
$$= \underbrace{\sum_{i=1}^{n} (\lambda_i - 1) B}_{\text{“connectivity-1” metric}} + \underbrace{\sum_{i=1}^{n} B + nnz(G)}_{\text{constant}}$$

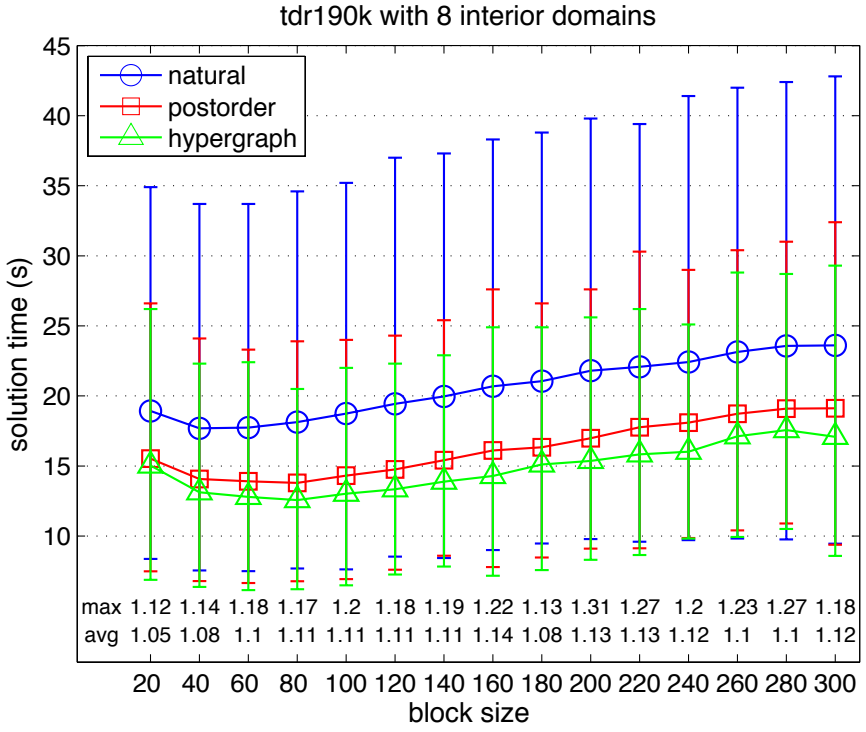



- **Minimize cost(π) using Patoh**
- **Additional 10% reduction in time**

# Sparse RHS: memory saving

- tdr190k from accelerator cavity design: N = 1.1M, k = 8
- Fraction of padded zeros, with different block size



# Summary

- **Graph partitioning**
  - Direct edge partition + min. vertex cover not effective
  - Recursive Hypergraph Bisection: 1.7x faster
- **Reordering sparse r.h.s. in sparse triangular solution**
  - postordering: 20-40% faster;  hypergraph: additional 10% faster
- **Future work**
  - Parallel hypergraph partitioning

# Extra Slides

# Parallelization with serial subdomain

- **No. of subdomains increases with increasing core count.**
  - ➔Schur complement size and iteration count increase

- **HIPS (serial subdomain) vs. PDSLin (parallel subdomain)**
  - M3D-C$^1$, Extended MHD to model fusion reactor tokamak, 2D slice of 3D torus
  - Dimension 801k, 70 nonzeros per row, real unsymmetric

| P | $N_S$ | HIPS 1.0 sec (iter) | PDSLin sec (iter) |
|---|---|---|---|
| 8 | 13k | 284.6 (26) | 79.9 (15) |
| 32 | 29k | 55.4 (64) | 25.3 (16) |
| 128 | 62k | -- | 17.1 (16) |
| 512 | 124k | -- | 21.9 (17) |

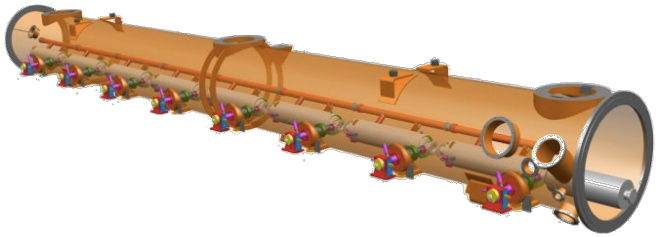
# PDSLin for largest system

- Matrix properties:
  - 3D cavity design in Omega3P, 3rd order basis function for each matrix element
  - dimension = 52.7 M, nonzeros = 4.3 B (~80 nonzeros per row), real, symmetric, highly indefinite
- Experimental setup:
  - PT-Scotch extracts 128 subdomains of size ~410k
  - SuperLU DIST factors each subdomain, computes preconditioner LU($\widetilde{S}$) of size 247k (32 cores)
  - BiCGStab of PETSc to solve Sy = d
- Performance:
  - Fill-ratio (nnz(Precond.)/nnz(A)): ~ 250
  - Using 2048 cores:
    - preconditioner construction: 493.1 sec.
    - solution: 108.1 second (32 iterations)

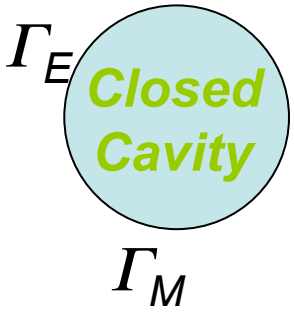
# Application 2: Accelerator cavity design

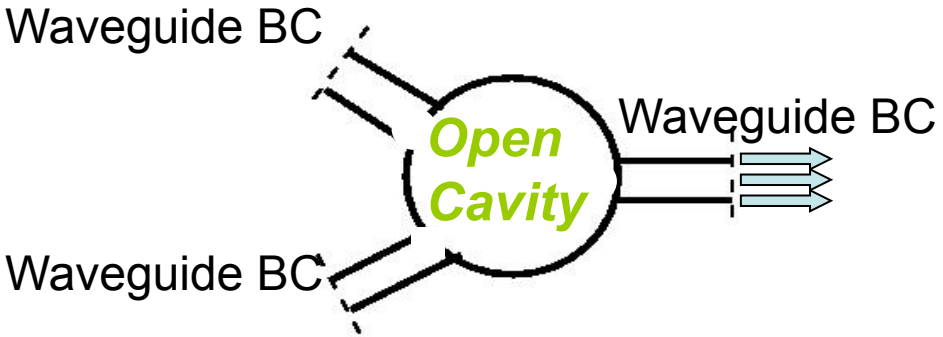
- DOE SciDAC: Community Petascale Project for Accelerator Science and Simulation (ComPASS), PI: P. Spentzouris, Fermilab
- **Development of a comprehensive computational infrastructure for accelerator modeling and optimization**
- **RF cavity: Maxwell equations in electromagnetic field**
- **FEM in frequency domain leads to large sparse eigenvalue problem; needs to solve shifted linear systems**

(L.-Q. Lee)



RF unit in ILC




linear eigenvalue problem

$$(K_0 - \sigma^2 M_0)\, x = M_0\, b$$

nonlinear complex eigenvalue problem

$$(K_0 + i\,\sigma W - \sigma^2 M_0)\, x = b$$