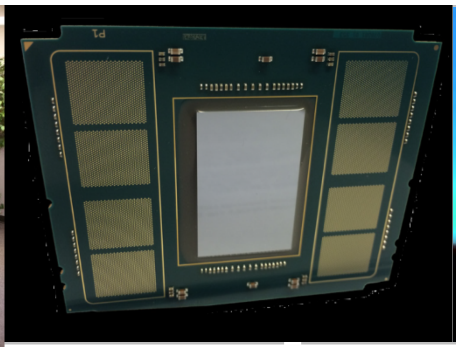
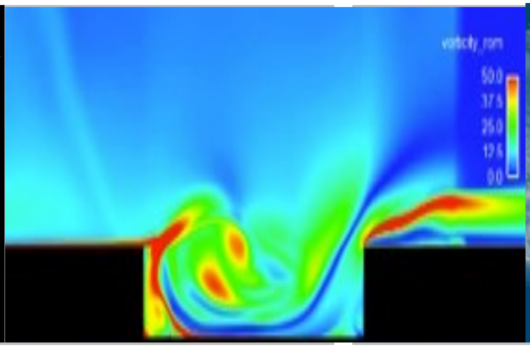
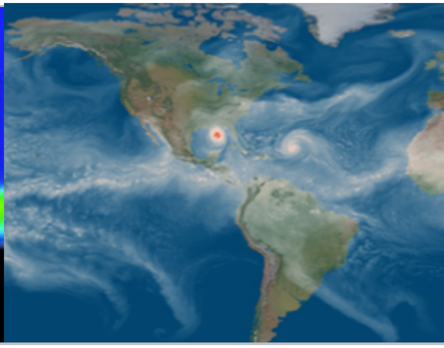
Sandia National Laboratories

A Parallel Hierarchical Low-Rank Solver for General Sparse Matrices

Erik Boman, Chao Chen, Eric Darve, Siva Rajamanickam, Ray Tuminaro,

Sparse Days, Sept 2017

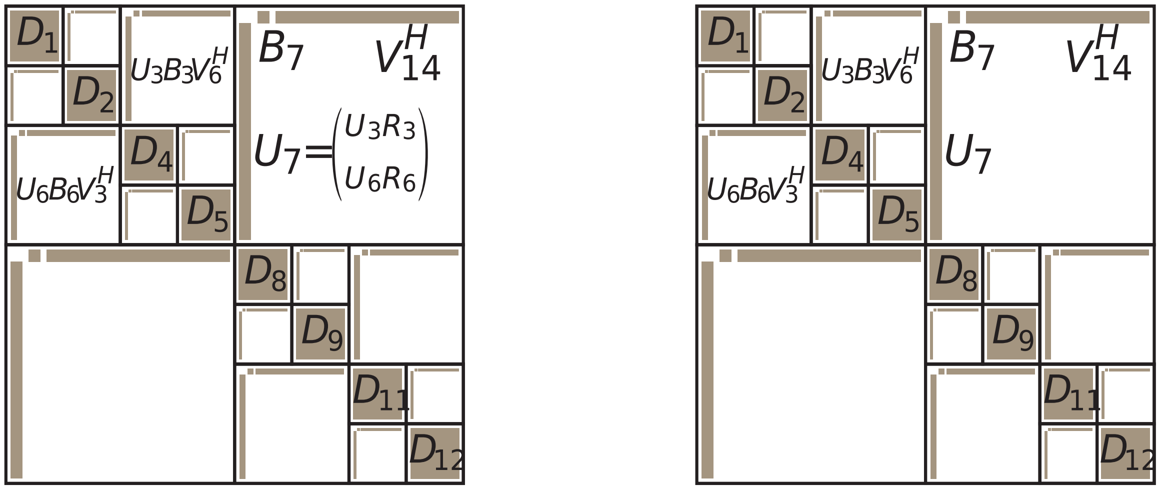
U.S. DEPARTMENT OF ENERGY NNSA National Nuclear Security Administration

Hierarchical Low-Rank (HLR) Matrices and Solvers

- Low-rank structure is common in applications
- Hierarchical (low-rank) matrices and solvers are *"hot"*
  - Early work by Hackbusch, 1999-2000 (H-matrix)
  - HSS and BLR formats most popular now
- Key insight: Many matrices have useful (rank) structure
  - Blocks far from diagonal can be approximated using low-rank
    - Holds for elliptic PDEs, some other (e.g., advection-diffusion problems)
    - Similar intuition as for Fast Multipole Methods (FMM)
    - May also apply to data science (e.g. covariance matrix)
- Our goals
  - Develop new solver/preconditioner with less memory (and flops)
  - Speed up sparse direct solvers (high accuracy)
  - Use as preconditioner (low accuracy)

Low-Rank Structure

- Low-rank structure often occurs in *off-diagonal blocks in*
  - The matrix A
  - The inverse of A
  - The LU factors of A
- Ex: Hierarchical low-rank structure (Figure from Wang et al.)

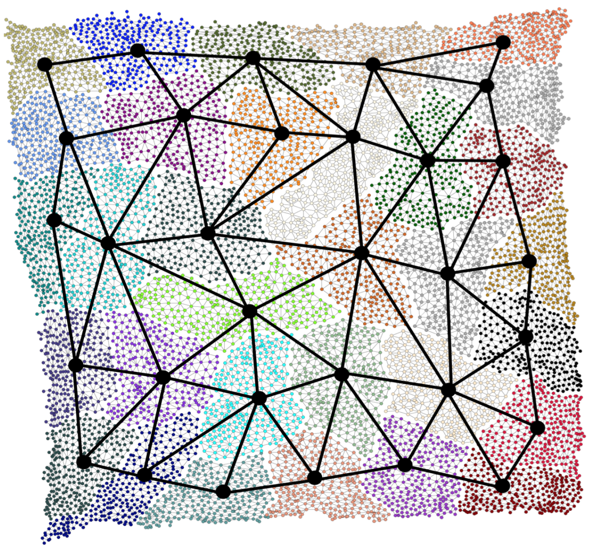


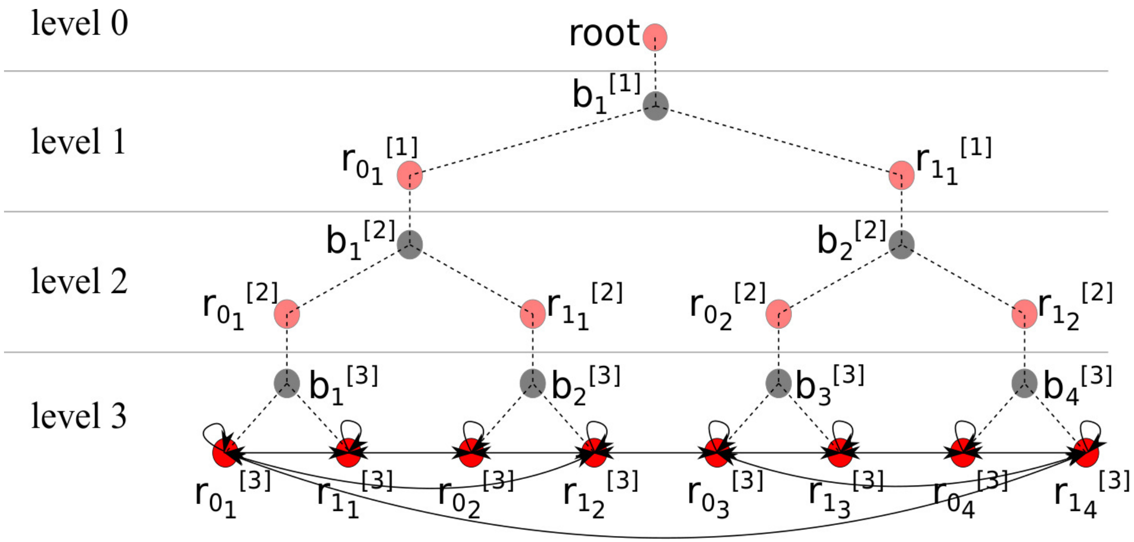
(a) HSS matrix. (b) HODLR matrix.

- Problem: rank growth in off-diagonal blocks
  - We use H2 format (allows further splitting, standard admissibility)

The LoRaSp/H2 Method

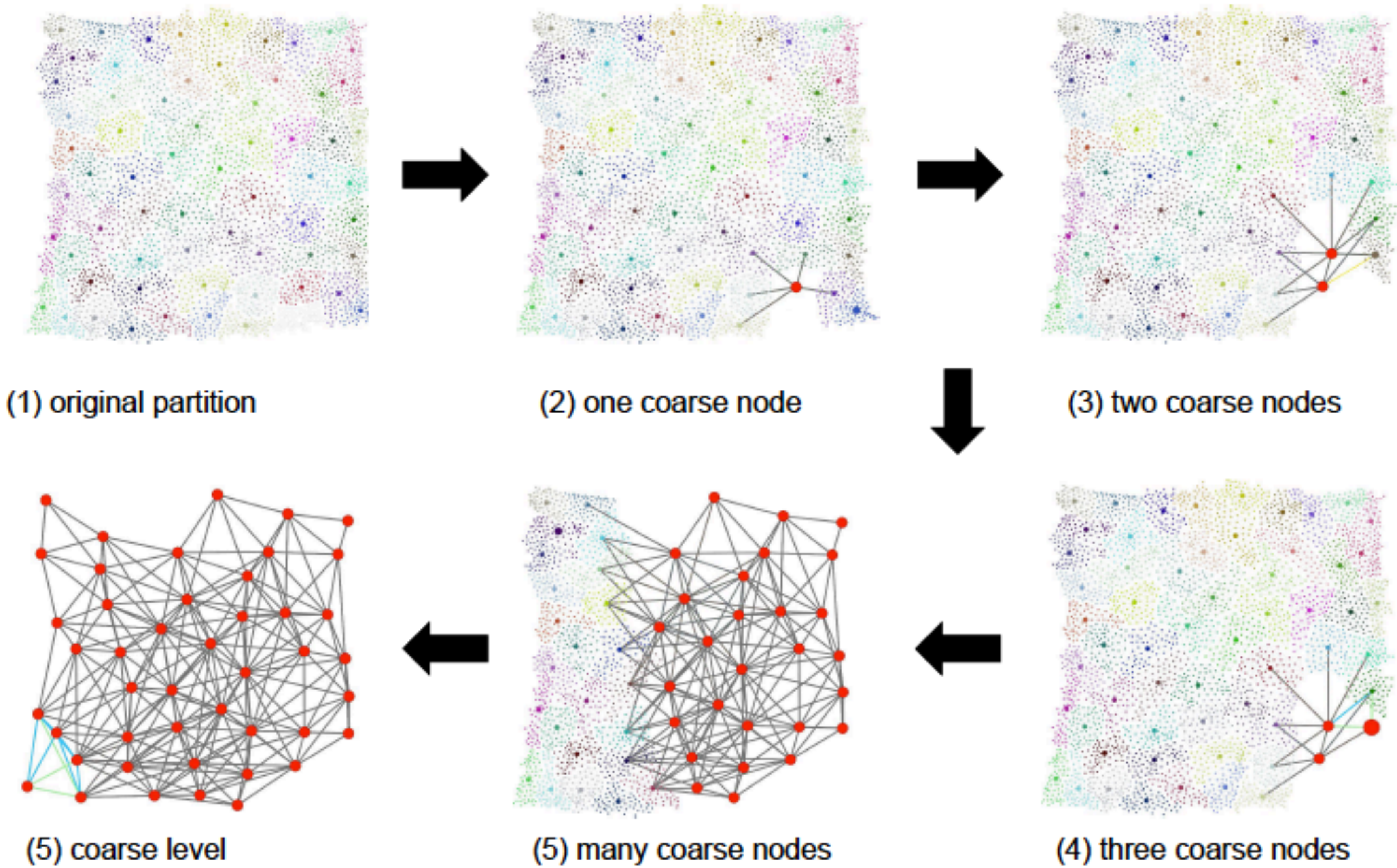
- Pouransari, Coulier, Darve, SISC 2017
- Partition graph via recursive bisection, construct tree
- Eliminate “clusters” (blocks) starting at leaf level
  - Approximate block LU factorization
  - New “coarse” dof via *extended sparsification*
- Merge neighbors, repeat for each level in the hierarchy (bottom up)





Figures courtesy Chen et al., and Pouransari et al.

Algorithm: Fine to Coarse Level



5

Multilevel Block Incomplete Factorization

- Strong ILU connection:
  - LoRaSp/H2 solver is really a Block ILU(0) factorization where the “dropped” blocks are approximated using low-rank method
    - A= LU+E, where E has low-rank blocks
  - We compensate for the dropped blocks by adding new rows/columns to the matrix (*extended sparsification*)
  - Typically, “fill” blocks have low-rank structure
- We eliminate the *fine* vertices (clusters)
  - The Schur complement for remaining *coarse* vertices is a much smaller matrix
  - Solve for this recursively
- Similarities to
  - ARMS (Saad, 2002)
  - AMG

6

Extended Sparsification

- For simplicity, assume symmetric A (can be extended)
- Suppose the off-diagonal blocks are (approx) low-rank:

$$\begin{pmatrix} A_1 & UV^T \\ VU^T & A_2 \end{pmatrix}$$

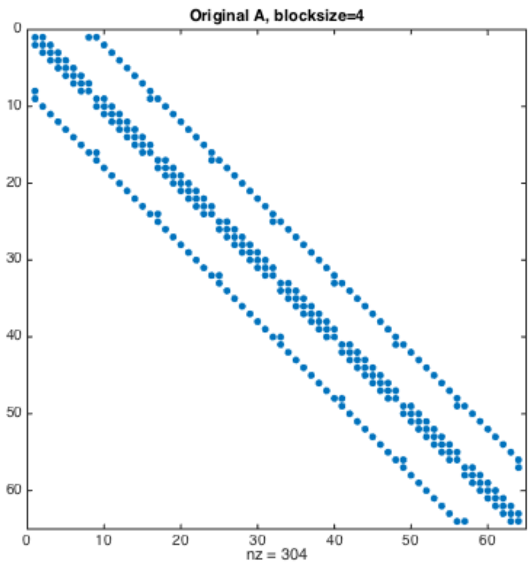
- We solve the equivalent extended system

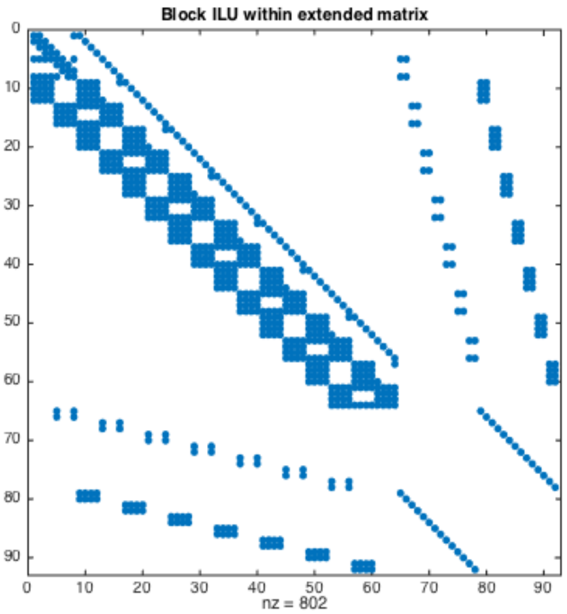
$$\begin{pmatrix} A_1 & 0 & U & 0 \\ 0 & A_2 & 0 & V \\ U^T & 0 & 0 & -I \\ 0 & V^T & -I & 0 \end{pmatrix}$$

- We sparsify the original matrix, but add extra rows/cols that also need to be factored.
- The lower the ranks of U,V, the smaller extended system

ILU and Extended Sparsification

- Note: Fill blocks in the block LU factors often have low rank
  - Schur complements in the Gaussian elimination
- Approach: Compute blocks in ILU(0) exactly, and
  - Approximate blocks in ILU(1) (not in ILU(0)) using low-rank
  - Extend matrix with new rows/cols

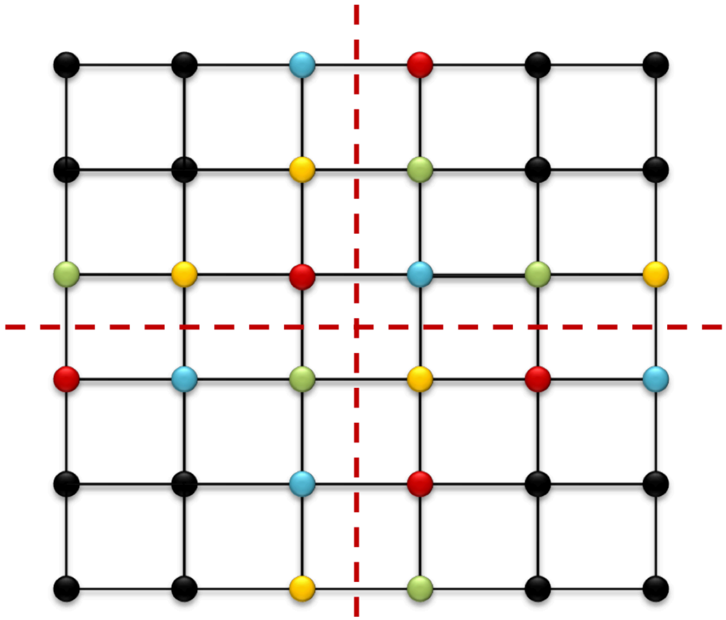




8

Our Parallel Algorithm

- Data parallel: Each processor works on a subgraph (subdomain).
- Consider the cluster graph:
  - Only boundary vertices need communication.
  - Interior vertices can be eliminated independently in parallel.
- Use graph coloring on the boundary to find concurrent work.
- #synchronization points = #colors
- Can overlap communication and computation.
- Repeat for each level.
- Future: Task-based; dynamic sched.



Example: 4 processors. Each vertex (node) corresponds to a cluster of variables.

Experiments

- Parallel H2 solver (and results) by Chao Chen
  - Parallel extension of LoRaSp serial code
  - MPI everywhere
  - Eigen library for dense linear algebra (on node)

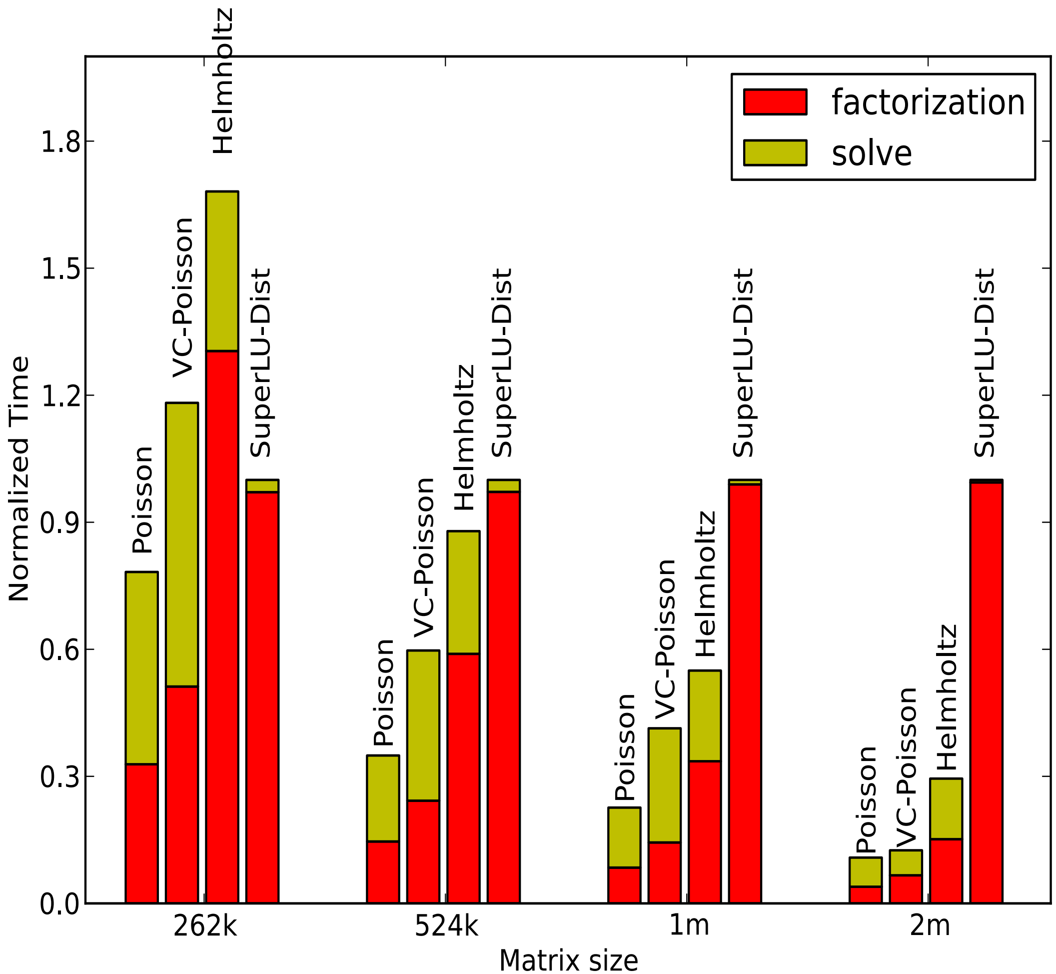
- SVD for low-rank compression
  - a) Fixed eps in matrix approx. (ranks vary)
  - b) Fixed rank in matrix approx. (quality varies)
    - used in most of the experiments

- Platform: Cray XC30 (Edison/NERSC)
  - Used 16 (out of 24) cores per node
  - Used up to 16 nodes (256 cores)

10

Results: Structured Mesh Case

Compare hierarchical solver as preconditioner vs. SuperLU-Dist direct solver on three 3D PDE model problems. 16 processors (MPI ranks).



(a) Total time

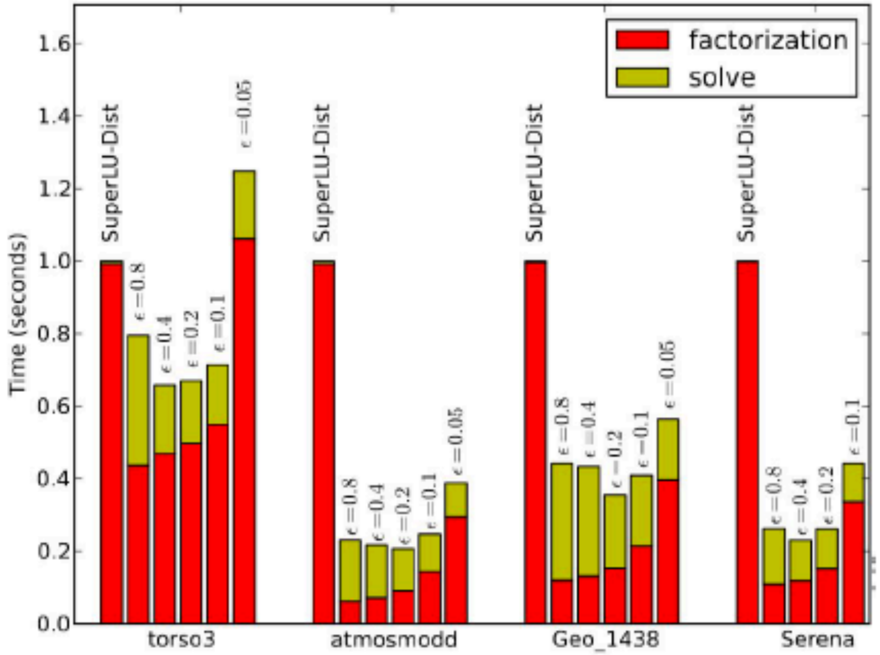
11

Results: Unstructured Case

Compare hierarchical solver as preconditioner vs. SuperLU-Dist on unstructured matrices, including nonsymmetric cases.

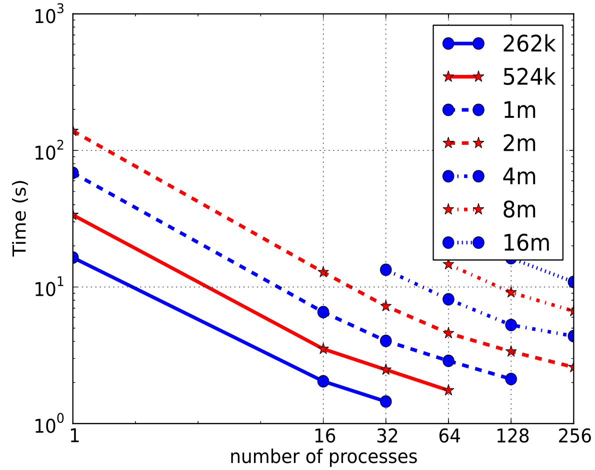
Table 1: General matrices from the University of Florida Sparse Matrix Collection

Matrices	size	# nonzero	symbolic pattern symmetry	numeric value symmetry	positive definite
torso3	0.26M	4.4M	no	no	no
atmosmodd	1.3M	8.8M	yes	no	no
Geo_1438	1.4M	60.2M	yes	yes	yes
Serena	1.4M	64.1M	yes	yes	yes

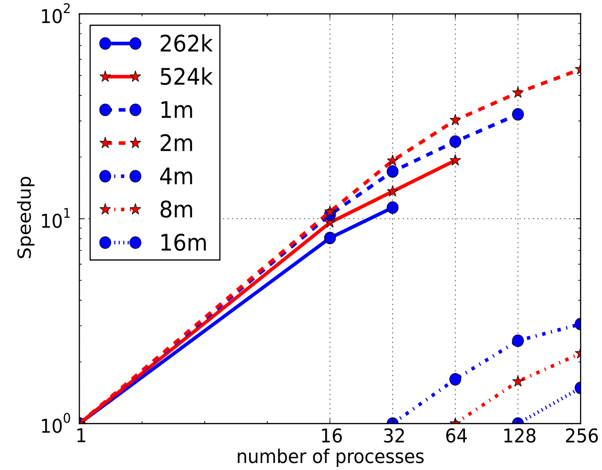


12

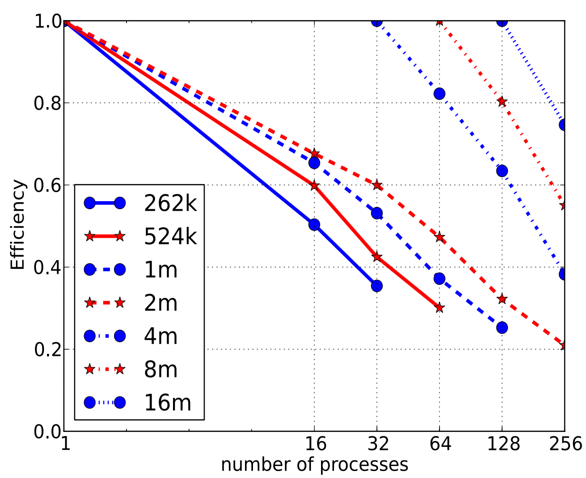
Results: 3D Poisson Eqn.



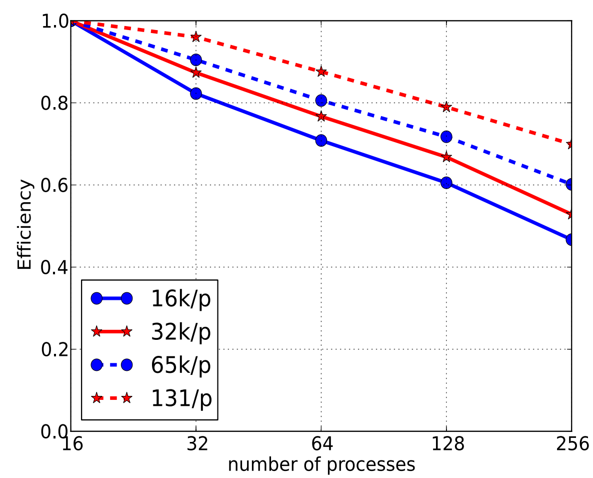
(a) Factorization time



(b) Factorization speedup



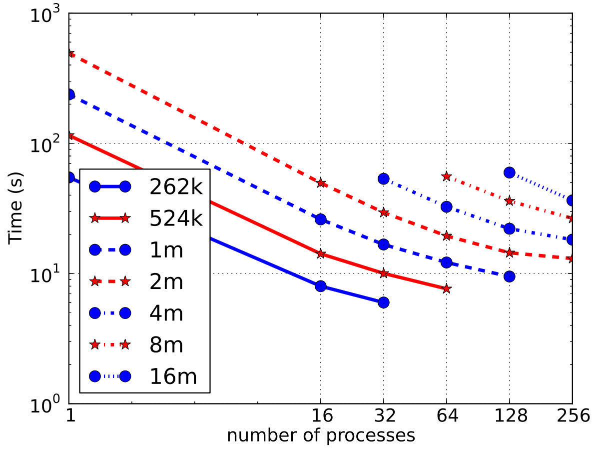
(c) Fixed total problem size



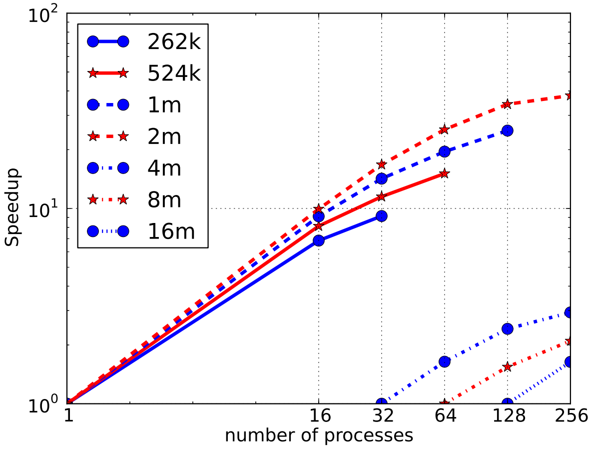
(d) Fixed problem size per processor

13

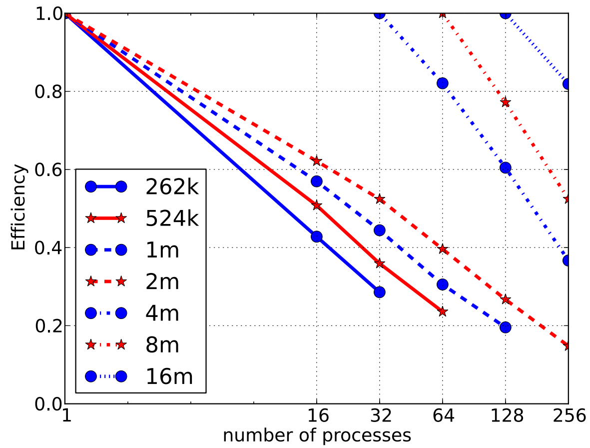
Results: Helmholtz eqn.



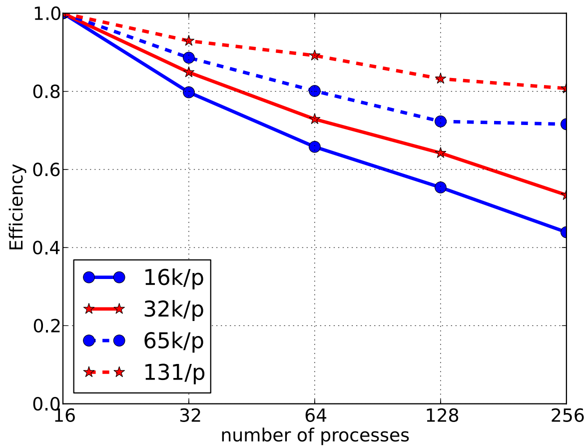
(a) Time



(b) Speedup



(c) Fixed total problem size



(d) Fixed problem size per processor

14

In Progress: Preserve Near-Null-Space

Laplace with Neumann bc's except Robin on bottom with $\beta = 100$ on ¼ of surface & $\beta = 0$ on the rest

standard method with vertical clusters

mesh	its	nnz/n
16 x 16 x 8	9	133.4
32 x 32 x 8	11	156.0
64 x 64 x 8	17	167.7
128 x 128 x 8	26	173.6

modified so 1st level compression accounts for constant

mesh	its	nnz/n
16 x 16 x 8	6	146.5
32 x 32 x 8	9	187.4
64 x 64 x 8	11	208.9
128 x 128 x 8	17	219.9
256 x 256 x 8	28	225.5

- Credit: Tuminaro (Matlab)
- Based on idea by Yang
- Only 1st level implemented so far

Conclusions (1)

- Hierarchical low-rank methods (HLR) augment the current solver/preconditioner ecosystem.
- Faster than sparse direct
- Most useful as preconditioner
- User must choose accuracy (input arg.)
- Setup phase can be expensive.
  - Can often amortize this cost over multiple rhs
- Theory promises near-linear complexity for many PDE problems
  - Potentially more robust than multigrid/AMG (at a cost)

16

Conclusions (2)

- Well suited for modern architectures
  - Most work is in dense linear algebra (even for sparse problems)
    - High arithmetic intensity
  - Many small dense matrices at same time
    - Could use batched BLAS/LAPACK
- Our hierarchical solver
  - Is motivated by H2 but can also be interpreted as multilevel ILU
  - Many options/variations still to explore
    - Low rank: SVD, RRQR, RRLU, ACA, ID, …
- Early days:
  - Several algorithm options, best choice unclear
  - Codes are still immature but rapidly improving

References

- Pouransari, Coulier, Darve, “Fast Hierarchical Solvers for Sparse Matrices using Extended Sparsification and Low-Rank Approximation”, SIAM J. Sci. Comp. 39(3), 2017.
- Chen, Pouransari, Rajamanickam, Boman, Darve, “A Distributed-Memory Hierarchical Solver for Sparse Linear Systems”, Parallel Computing, in revision.

18