SAND2018-1183PE

Opportunities for Concurrency in a Domain Decomposition Preconditioner

Clark Dohrmann

Computational Solid Mechanics & Structural Dynamics Department

Applied Computer Science Meeting
Sandia National Laboratories
February 12-16, 2018

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Thanks

Sandia National Laboratories

- Mark Hoemmen
- Paul Lin
- Kendall Pierson

Outline

- **BDDC* Preconditioner:**
  - Introduction
  - Multilevel Approach
- **Concurrency Opportunities:**
  - Computational
  - Algorithmic
  - Numerical Results
- **MPI Considerations:**
  - Communication Avoidance
  - Number of ranks
- **Recap**

*Balancing Domain Decomposition by Constraints

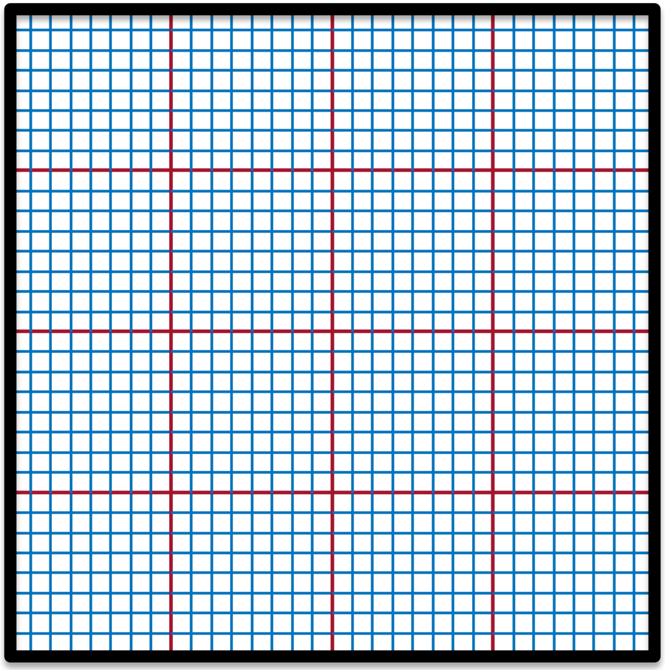
BDDC Introduction

- **Extension of BDD Method (Mandel, 1993)**
  - additive coarse correction
  - sparser coarse matrix
  - no singular problems if careful
- **Primal counterpart of FETI-DP (Farhat, et al., 2000)**
  - eigenvalues essentially the same
  - multilevel extensions & use of inexact solvers
- **Nice Physical Interpretation (Li & Widlund, 2006)**
  - Partially assembled system
  - Coarse basis functions minimize energy

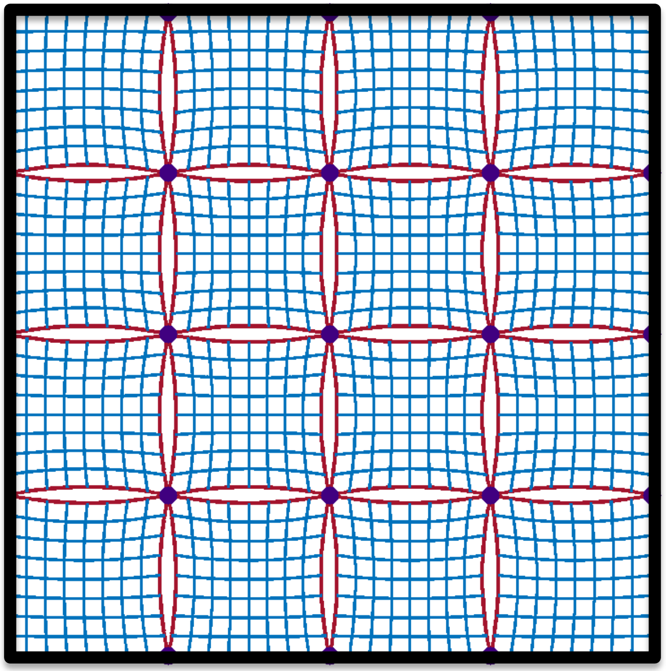
Some BDDC Refs: *SIAM Sci. Comput*. (2003), 25(1), 246-258.
Appl. Numer. Math. (2005), 54, 167-193. See extra slides for additional refs.

BDDC Introduction

A



fully assembled model



M

partially assembled model*

Basic components: decomposition, constraints, weights

$\lambda(M^{-1}A)$, FETI-DP connection, constraints/coarse space connection

Note: solving partially assembled problem equivalent to solving independent local subdomain problems and adding coarse solution } concurrency opportunity

*problem on right is easier to solve than one on left, see LW06 for additional details

BDDC Introduction

- **Computational Kernels:**
  - **Sparse matrix-vector multiplication**
    - **Apply operator/coarse interpolations**
  - **Sparse Linear Solvers**
    - **KNL: looking good with threads**
    - **GPU: uncertain future**
      - **May need to replace with inexact solves**
  - **Dense linear algebra**
    - **Iterative solution acceleration**
      - **Subspace recycling (projections)**
    - **Sparse direct solvers (supernodal variants)**
    - **Well-suited for vectorization**

Multilevel BDDC*

- **Within levels:**
  - **Solve problem directly if small enough**
    - **Threaded or distributed memory solver options**
  - **If not, decompose problem into subdomains**
    - **Graph-based or geometric partitioning**
  - **Coarsen each subdomain to a single element**
    - **Null space preservation for scalability**
  - **Pass coarse problem data to next coarser level**

- **Across levels:**
  - **Add contributions together in apply phase**
  - **Concurrency possible in both initialize and apply phases****
  - **Concurrency enabled using disjoint MPI communicators****

*see references in extra slides for related theory and practice

**see BMP16 for some details

Concurrency Opportunities

Sandia National Laboratories

- **Computational:**
  - **Distributed memory**
  - **Shared memory**
    - **Multiple subdomains for each MPI rank**
    - **Threading of sparse direct solvers**
  - **SIMD (vectorization)**
    - **BLAS kernels for sparse direct solvers**
    - **BLAS kernels for iterative solution acceleration**

Common to many DD preconditioners

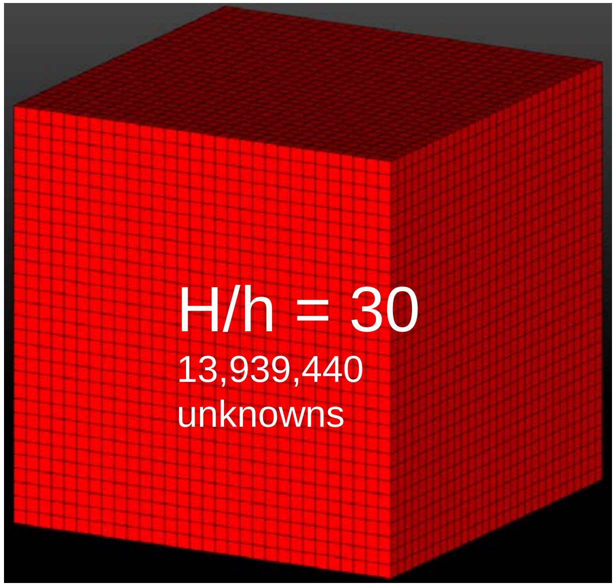
- **Algorithmic:**
  - **Concurrent work across levels**
    - **Solve phase: residual reductions prior to solves**
    - **Initialization phase: concurrent factorizations**
  - **Reduction of idle time across levels**

More specific to BDDC

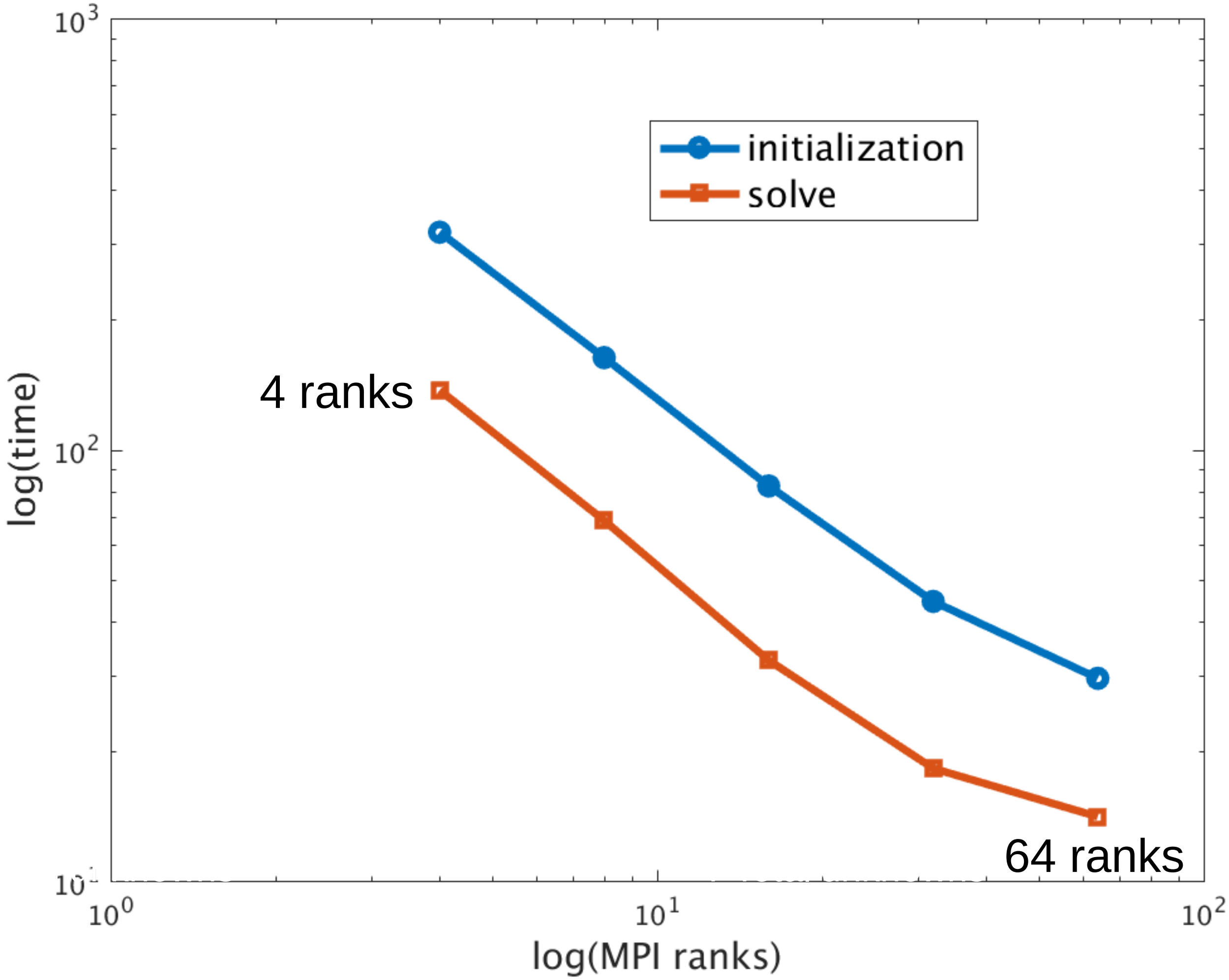
Numerical Results

- **Example Problem:**
  - **Poisson equation on unit cube, 512 cubic subdomains**
  - **Strong scaling (same problem size, more MPI ranks)**
    - **N = 1, 2, 4, 8, 16, 32, 64 ranks**
      - **512/N subdomains per rank, no threading**
    - **30 elements along each subdomain edge**

Each subdomain looks like this



Numerical Results



Numerical Results

- **Example Problem:**
  - **Unit cube domain, 1000 subdomains per MPI rank**
  - **Two-level preconditioner, 4 solves**
  - **Not fully optimized, best times from 4 different runs**

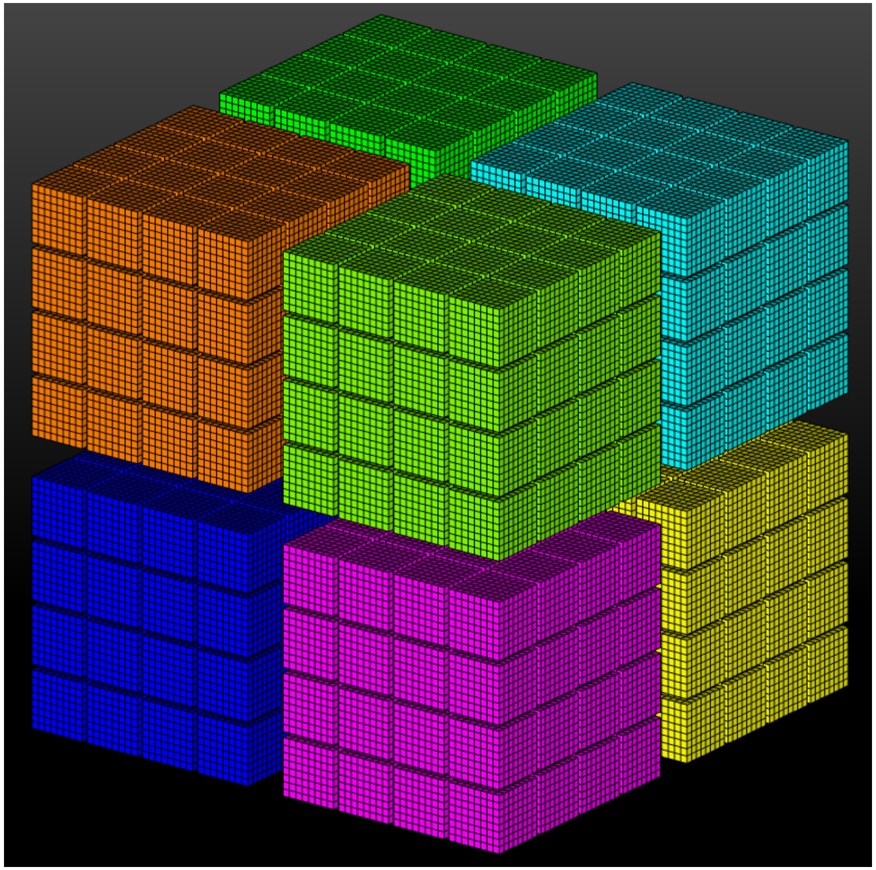
	27 ranks	28 ranks	speedup
	H/4 = 4		
Init time	8.23	6.97	1.18
Solve time	7.42	5.28	1.41
	H/h = 8		
Init time	26.1	21.5	1.21
Solve time	28.3	23.9	1.18

Some additional details: 27,000 subdomains, 75,690 coarse problem unknowns, and 1,756,920 and 13,939,400 problem unknowns for H/h = 4 and H/h = 8, respectively.

MPI Considerations

Sandia National Laboratories

- Simple example for discussion:
  - Each KNL has 1 MPI rank and 64 subdomains
  - 8000 KNLs ⇒ 8000 MPI ranks (compare 512,000 ranks)



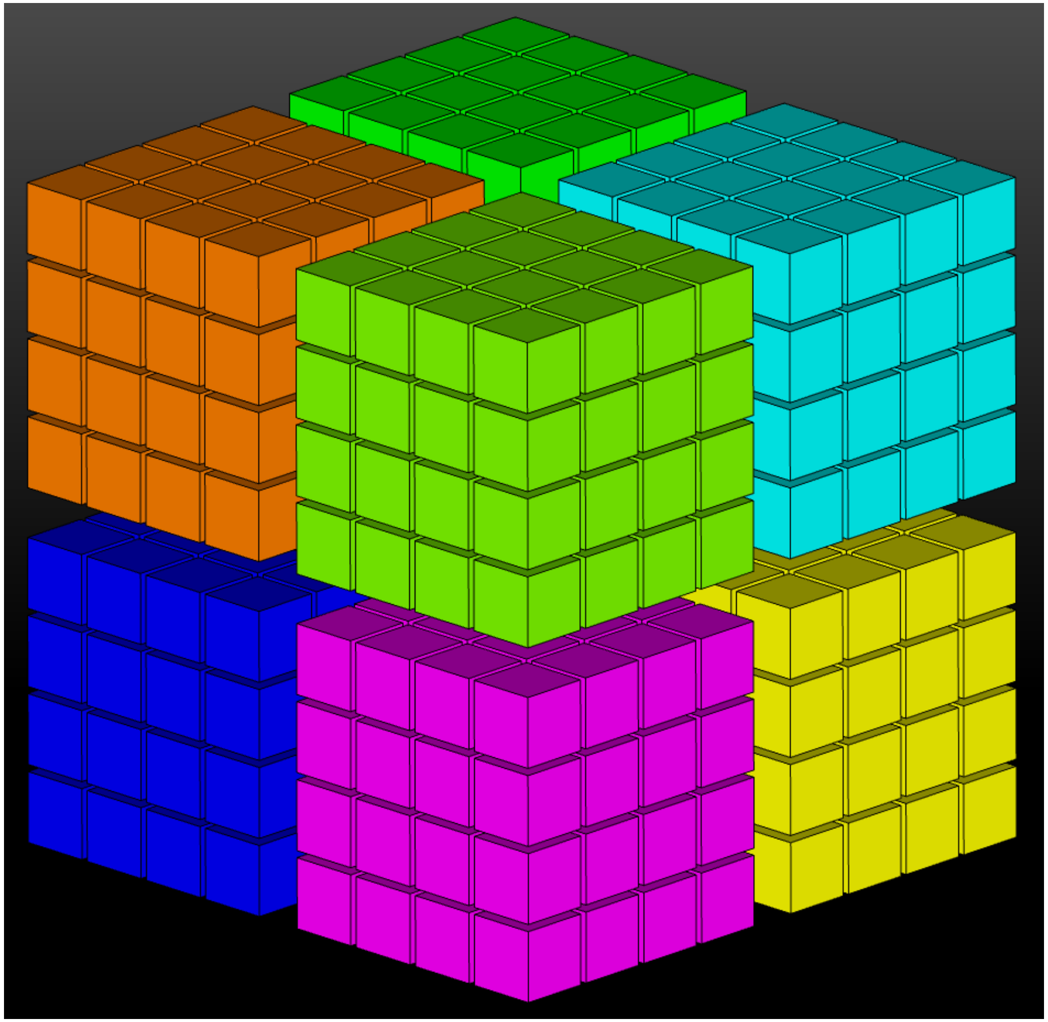
Decomposition shown for 8 of 8000 KNLs.

Total number of subdomains is 64*8000 = 512,000

Total number of unknowns for scalar problem is about $512{,}000 \cdot n^3$, where n is 8 in picture (n would be much larger in practice)

MPI Considerations

- **First level of coarsening:**



Decomposition shown for 8 of 8000 KNLs.

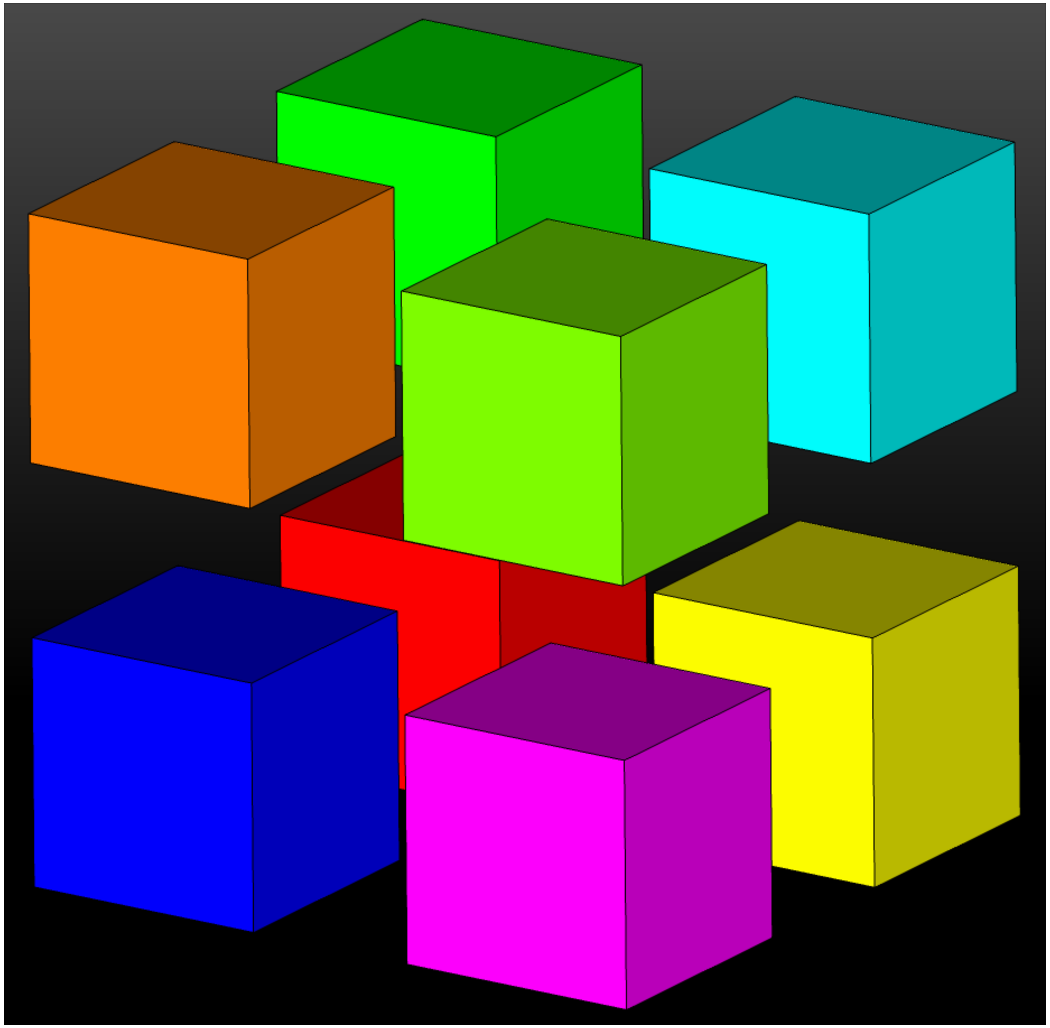
8x coarsening in each direction

No MPI communications needed to obtain this coarse level

About 512,000 unknowns at this level

MPI Considerations

- **Second level of coarsening:**



Decomposition shown for 8 of 8000 KNLs.

4x coarsening in each direction. Each subdomain has at most 8 unknowns

No MPI communications needed to obtain this coarse level

About 8000 unknowns at this level

Assembly requires MPI communications, but quite modest

Recap

- **Lots of Concurrency Opportunities:**
  - **Distributed memory**
  - **Shared memory**
  - **Vectorization**
  - **Across levels of hierarchy**
- **MPI Considerations:**
  - **Can reduce number of ranks by threading**
    - **Success depends on effective threading at node level**
    - **Domain decomposition algorithms like BDDC make this easier**
  - **Can reduce communications by "smart" coarsening**
    - **Simple example showed that only modest communications required during initialization phase**
    - **Application interface changes likely needed**

Thank You!

Extra Slides

Selected References

- BMP16: S. Badia, A. Martin and J. Principe, “Multilevel balancing domain decomposition at extreme scales”, *SIAM J. Sci. Comput*., Vol. 38, No. 1, pp. C22-C52, pp. 159-179, 2016.
- LW06: J. Li and O. Widlund, “FETI-DP, BDDC, and block Cholesky methods”, Int. J. Numer. Meth. Engng, Vol. 66, pp. 250-271, 2006.
- Some related links:
  - BDDC intro: http://epubs.siam.org/doi/abs/10.1137/S1064827502412887
  - BDDC theory: https://www.sciencedirect.com/science/article/pii/S0168927404001795
  - BDDC for "extreme" computing: http://epubs.siam.org/doi/abs/10.1137/15M1013511
  - Multilevel BDDC: http://epubs.siam.org/doi/abs/10.1137/050629902 https://link.springer.com/article/10.1007/s00607-008-0014-7

- Many other references to different problem types, theory, and practical implementations. Active area of research in domain decomposition.