SAND2015-2061C

Exceptional service in the national interest

Sandia National Laboratories

Strategies for Reducing Setup Costs in Algebraic Multigrid

Jonathan Hu

SIAM CS&E, March 14-18, 2015

U.S. DEPARTMENT OF ENERGY

NNSA National Nuclear Security Administration

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000. SAND NO. 2015-XXXXP

Introduction

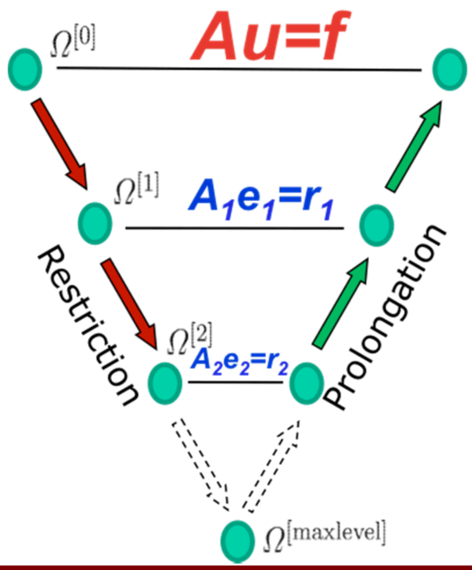
Sandia National Laboratories

- Algebraic multigrid (AMG) used by many applications at Sandia and elsewhere to great success
- As core counts increase, greater demand for good performance (total wall clock time)
- Setup cost of AMG has nontrivial setup cost
- Can cost be reduced by reusing information from previous setup?

Algebraic multigrid (AMG)

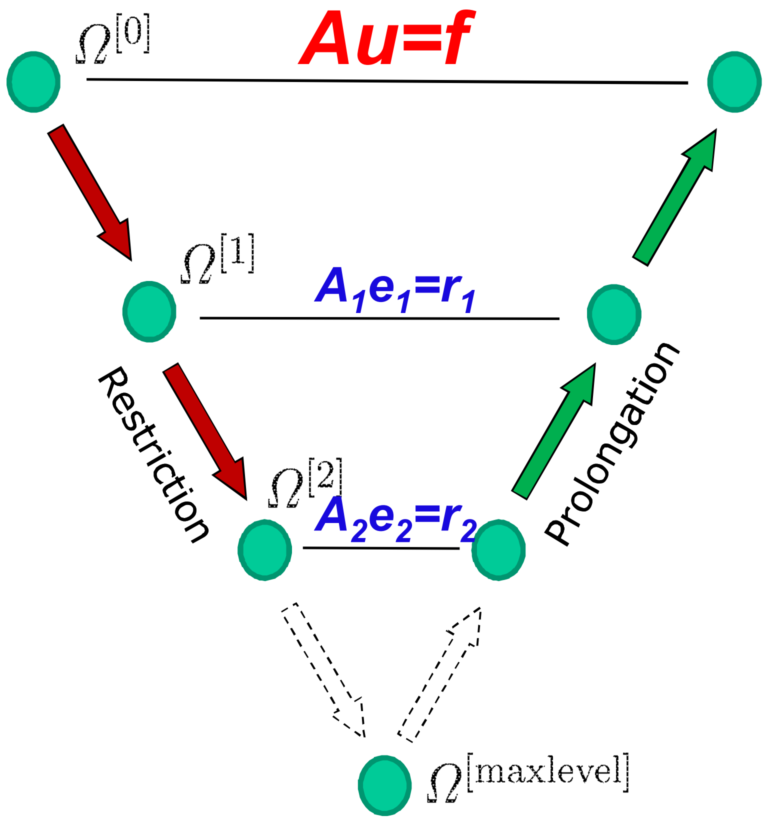
- Scalable solution method for elliptic PDEs
- Typically used as preconditioner to Krylov method
- Idea: capture error at multiple resolutions:
  - **Smoothing** reduces oscillatory error (high energy)
  - **Coarse grid correction** reduces smooth error (low energy)

- Two main variants
- Classical (Ruge-Stuben) AMG
  - Coarse grid DOFs are subset of fine DOFs
- Smoothed aggregation ←
  - Coarse grid DOFs are groups of fine DOFs



Smoothed Aggregation (SA) Main Kernels

- Setup
  - Form coarse unknowns (aggregation)
  - Prolongator creation
    - $P=(I-\omega D^{-1}A)P^{(tent)}$
  - Matrix matrix multiply
    - $A_k = R\,A_{k-1}\,P$
  - Load balancing of A_k's
  - (Smoother initialization)
- Apply
  - Matrix-vector multiply
  - (Triangular solves)

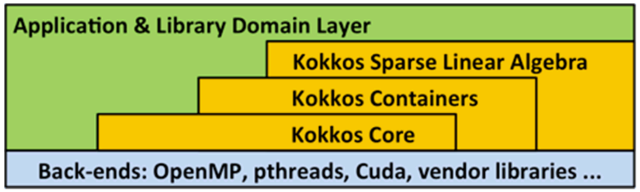


MueLu: Trilinos Multigrid Library

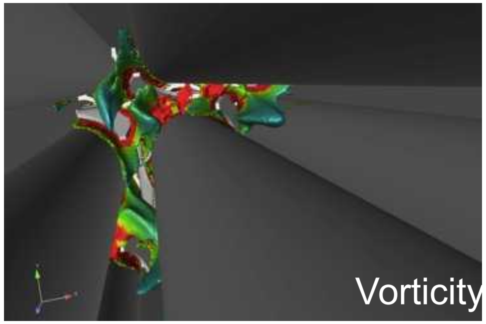
- Compatible with either 32-bit (Epetra) or templated (Tpetra) Trilinos linear algebra stack.
- Coarsening algorithms
  - smoothed aggregation (SA), Petrov-Galerkin SA, Maxwell, emin
- Smoothers: Jacobi, GS, *l*1 GS, polynomial, ILU
- Direct solvers: Trilinos native, SuperLU
- Load balancing: multijagged, RCB

Tpetra

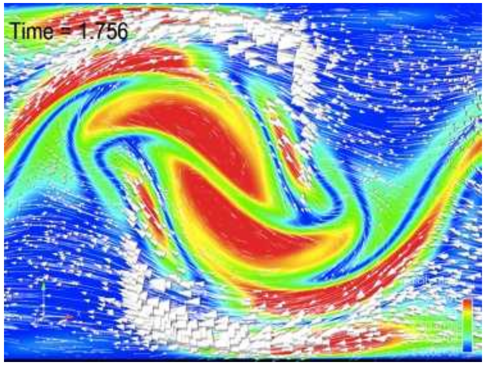
- Indirectly uses Kokkos (high-performance node-level kernels)
  - SPMV
- More details in poster session



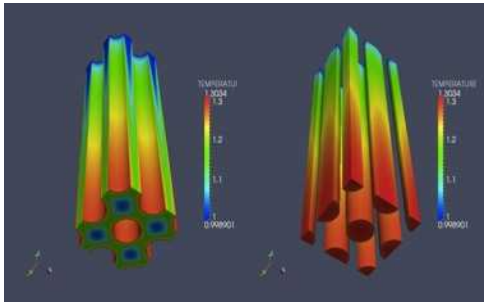
Drekar (J Shadid, R Pawlowski, E Cyr, T Smith, T Wildey)



LES: Flow over spacer grid



MHD: Hydromagnetic Kelvin-Helmholtz

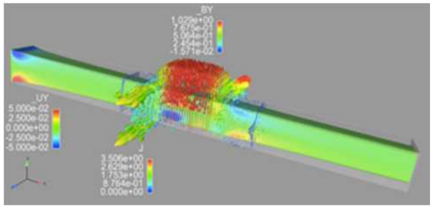


Conjugate Heat Transfer

Scalable parallel implicit FE code

- Includes: Navier-Stokes, MHD, LES, RANS
- Architecture admits new coupled physics
- Support of advanced discretizations
  - mixed, physics compatible and high-order basis functions
  - multi-physics capable (conjugate heat transfer)
- Advanced UQ tools/techniques
  - Adjoint based sensitivities and error-estimates
- Advanced solution methods
  - Parallel solvers from SNL's Trilinos framework
  - Physics-based preconditioning
  - Fully-coupled multigrid for monolithic systems

Some motivation for reuse



MPI tasks	DOFs	AMG (ILU) (s)
128	845k	9.5 (7.2)
1024	6.5m	10.8 (8.7)
8192	51m	12.2 (9.4)
65,536	401m	25.5 (10.4)
524,288	3.2b	1312 (452)

Weak scaling, BG/Q, *MHD generator**
AMG and ILU(0) smoother setup time

- AMG setup times growing with #MPI tasks
- Main offenders: matrix matrix multiply, smoother setup, prolongator setup

*Data courtesy P. Lin

Reuse: tentative prolongator

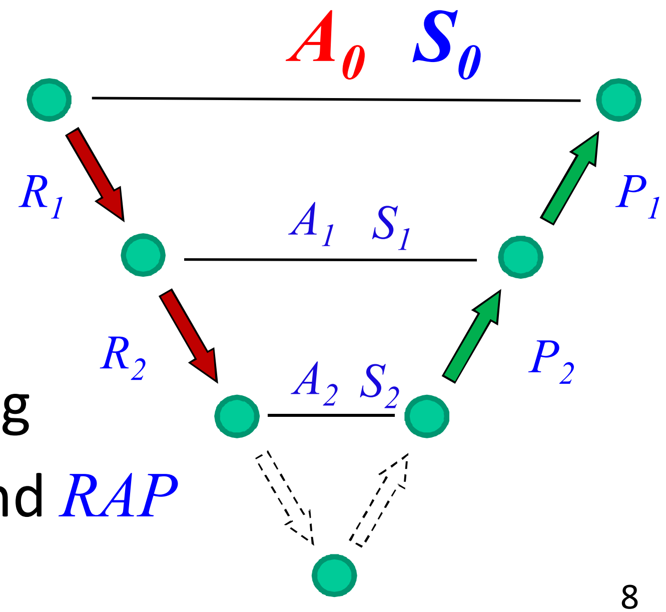
- Many possibilities for data reuse between multigrid setup phases. What works best is problem dependent.

- Reuse
  - Tentative interpolants $P^{(tent)}, R^{(tent)}$
- Recompute
  - Final prolongators $P=(I-\omega D^{-1}A)P^{(tent)}$
  - Matrices A_i , $i > 0$
  - Smoothers S_i

✓ Avoids construction of tentative ops.

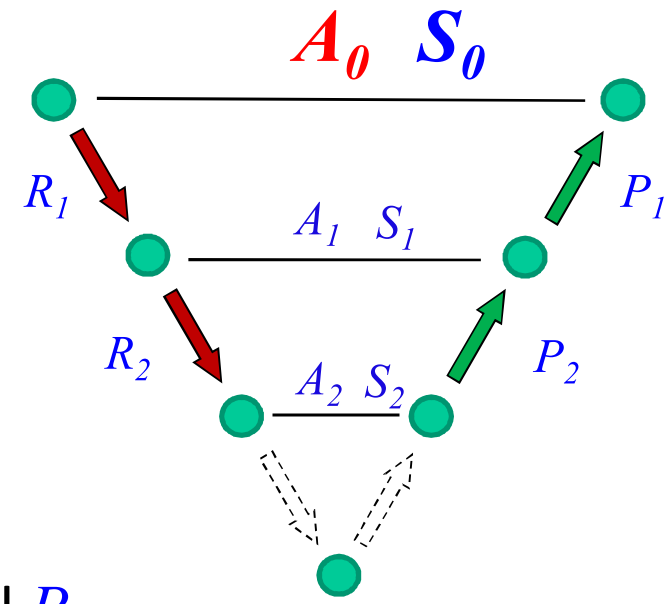
✓ Preserves import object for rebalancing

✗ Requires matrix-matrix product for P and RAP



Reuse: smoothed prolongators

- Reuse
  - Prolongators/restricters P_i, R_i
- Recompute
  - Matrices A_i , $i > 0$
  - Smoothers S_i



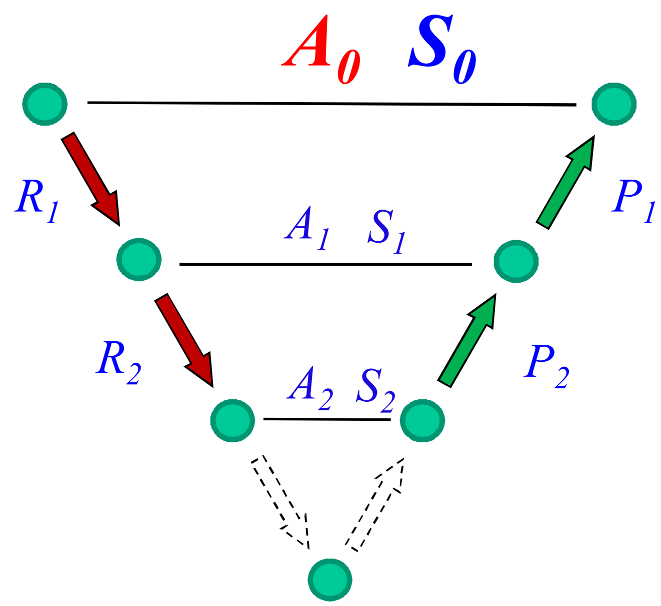
✓ Avoids matrix-matrix product for final P_i

✓ Preserves import object for rebalancing $A_i, i>0$

✗ Requires matrix-matrix product RAP

Reuse: all but fine grid smoother

- Reuse
  - Prolongators/restricters P_i, R_i
  - Matrices A_i , $i > 0$
  - Smoothers $S_{i,}$ $i>0$
- Recompute
  - Smoother S_0



✓ No matrix-matrix products required

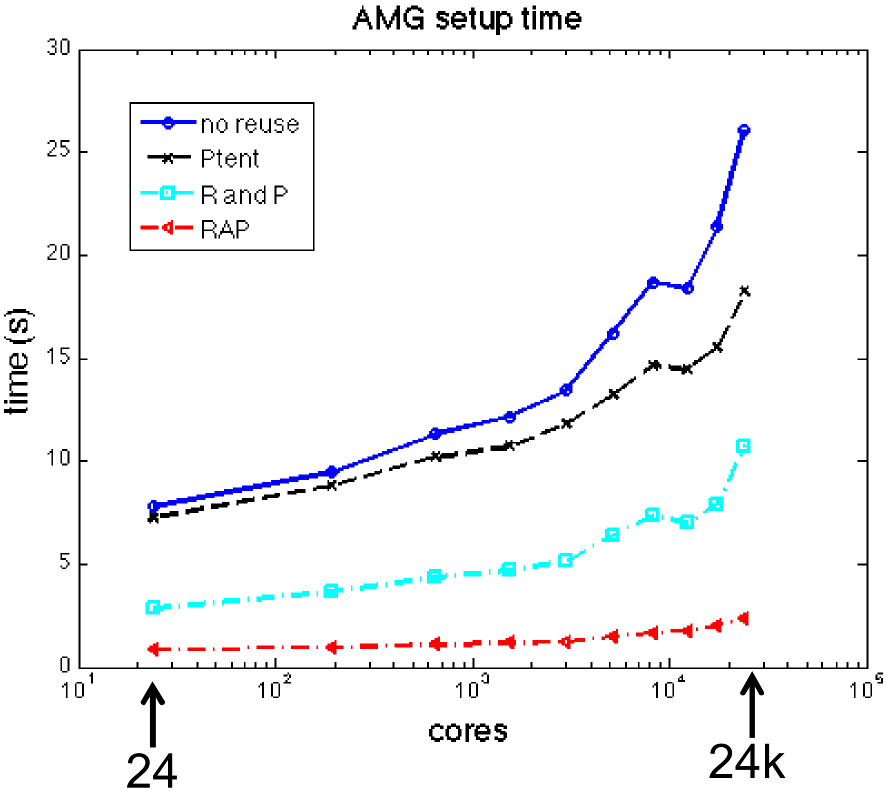
✓ Preserves coarse smoother data

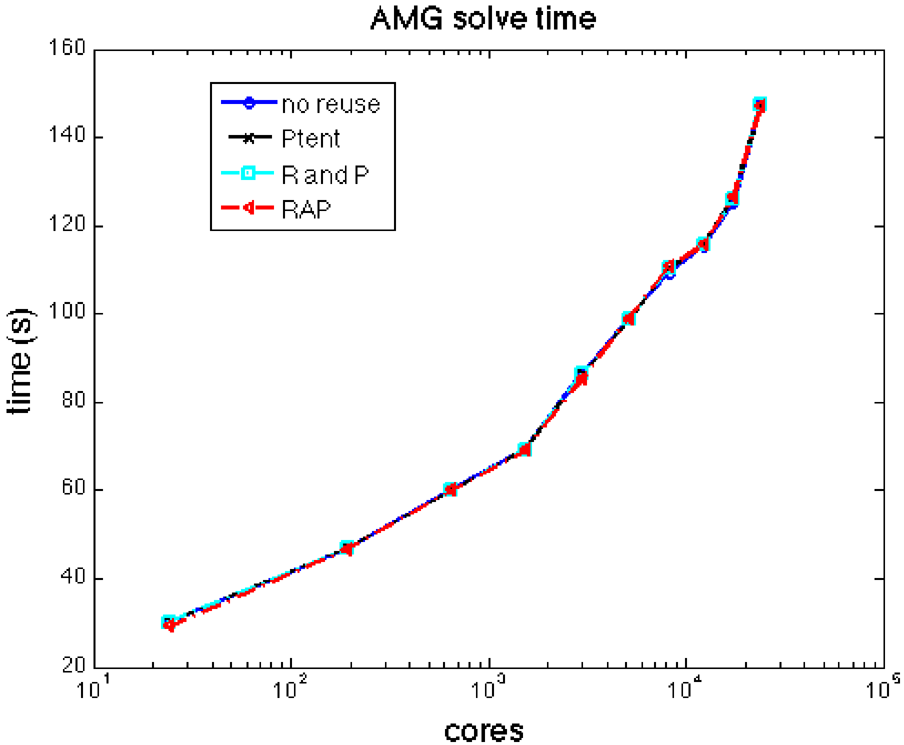
✓ Preserves rebalancing information

✗ Least likely to converge

Reuse experiments: Jet problem

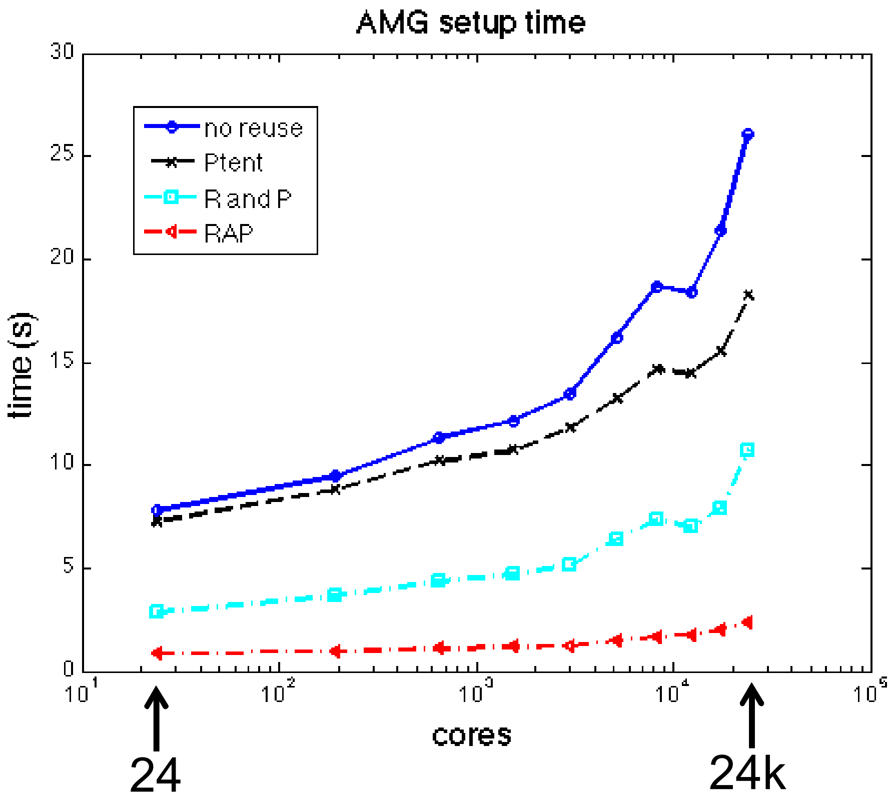
- Drekar; 3D Jet, Re=10^6, CFL ~0.25, no slip BCs
- SA AMG, V(3,3) symmetric Gauss-Seidel smoothing





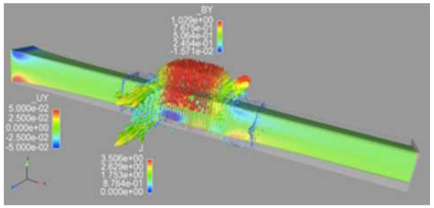
Reuse experiments: Jet problem

- Drekar; 3D Jet, Re=10^6, CFL ~0.25, no slip BCs
- SA AMG, V(3,3) symmetric Gauss-Seidel smoothing

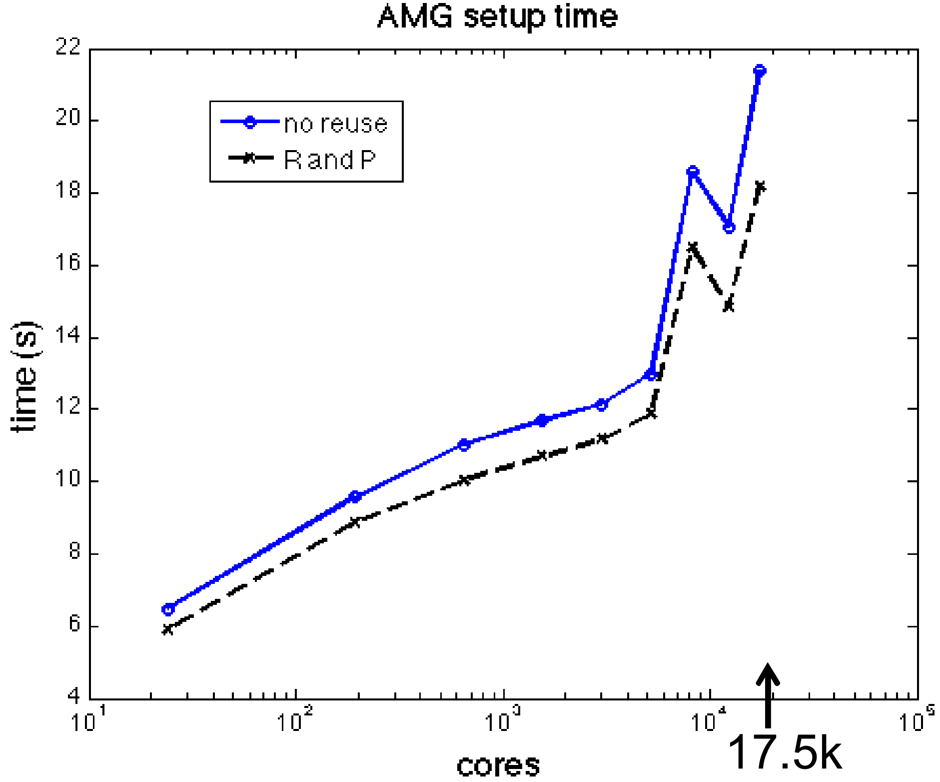


- Setup cost almost entirely
  - Smoothed prolongator
    - $P=(I-\omega D^{-1}A)P^{(tent)}$
  - Galerkin product
- For this particular problem, convergence maintained

Reuse experiments: MHD

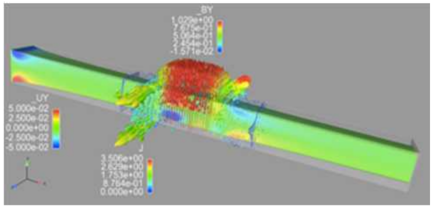


- Steady-state 3D MHD generator, Drekar
  - Resistive MHD model
  - stabilized FE; Newton-Krylov solve
  - 8 DOFs/mesh node

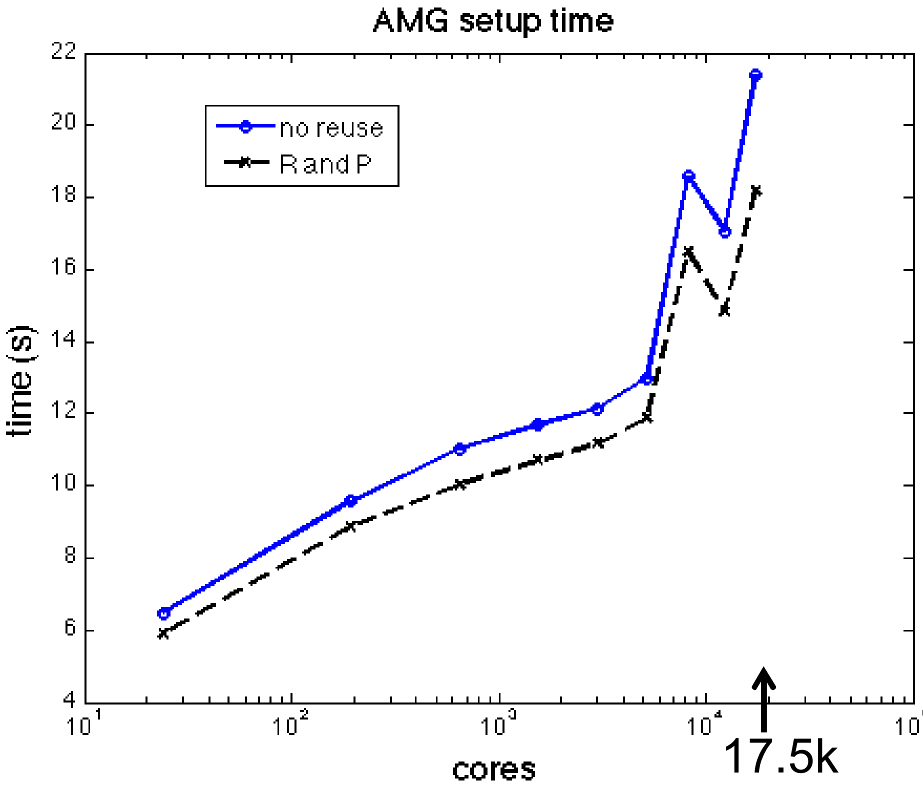


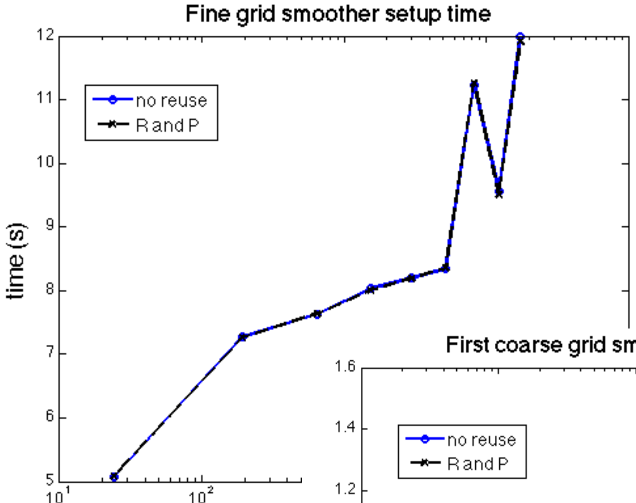
- Monolithic preconditioner
- Prolongator is unsmoothed
  - Requires more robust smoothing
  - In this case, DD/ILU(0)
- *P* cheap to construct compared to smoothers
- Cannot reuse coarse A_i's
  - Linear solve does not converge

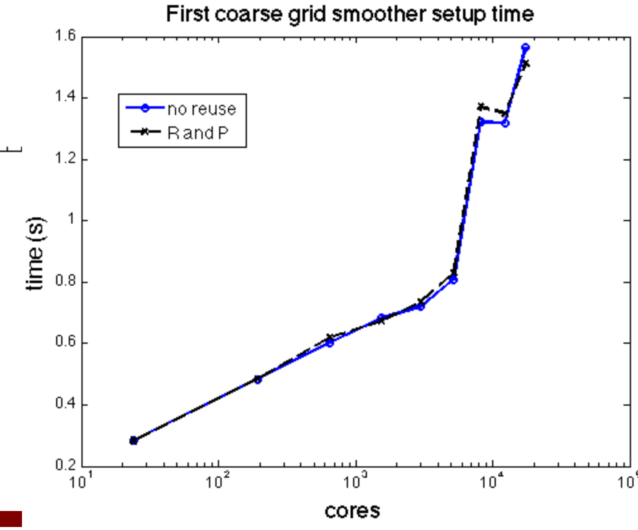
Reuse experiments: MHD



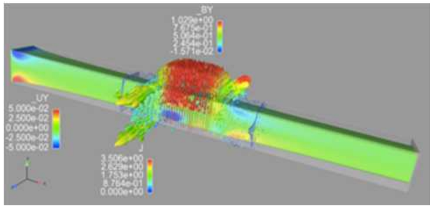
- Steady-state 3D MHD generator, Drekar
  - Resistive MHD model
  - stabilized FE; Newton-Krylov solve
  - 8 DOFs/mesh node



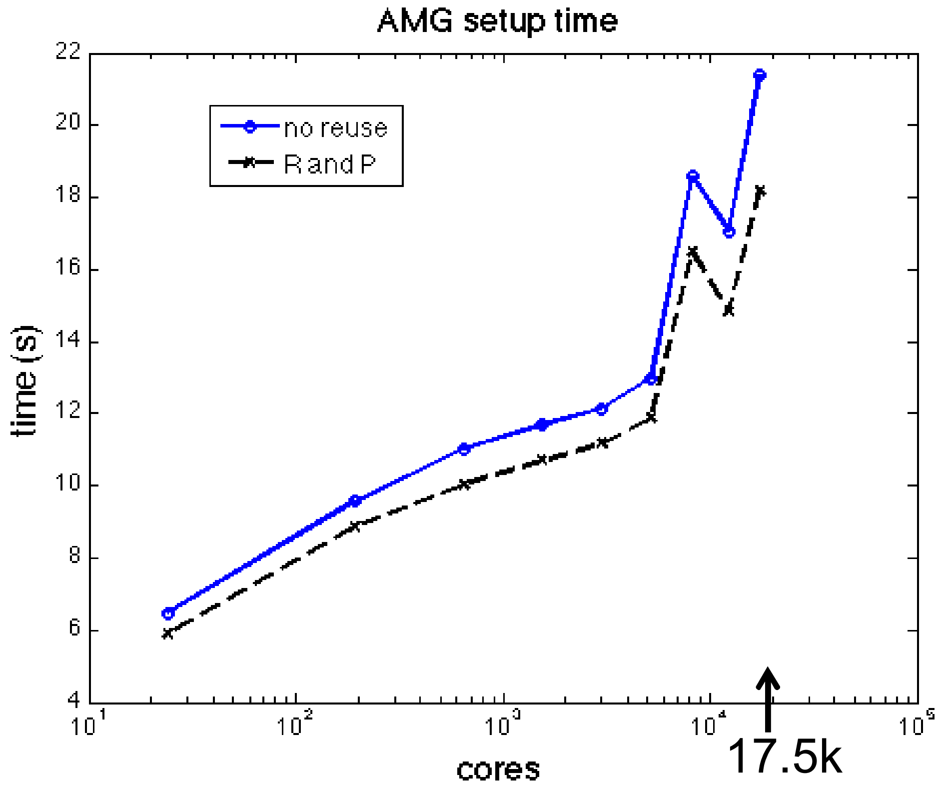




Reuse experiments: MHD



- Steady-state 3D MHD generator, Drekar
  - Resistive MHD model
  - stabilized FE; Newton-Krylov solve
  - 8 DOFs/mesh node



- Time dominated by smoother
  - Opportunity to reuse local graph
  - No reuse results yet for this…
- As problem grows, expect additional comm. costs
  - matrix-matrix product

Future: reuse within smoothers

- Additive Schwarz/subdomain ILU
  - ✓ Data import infrastructure
  - ✓ Local symbolic factorizations
  - ✗ Data transfers unavoidable
- Polynomial smoothers
  - reuse eigenvalue estimate
  - Reuse initial guesses for eigenvalue estimates (reduce matvecs)
- Jacobi, Gauss-Seidel, etc. – no reuse

Future: Reuse within Sparse Matrix Matrix Multiply

- SPGEMM is a dominant kernel in AMG setup
  - $A_C = R A_F P, \;\; (I - \omega D^{-1} A) P^{(tent)}$
  - First product required even if reuse grid transfers
- Trilinos implementation extends Gustavson algorithm [1]
- At least two reuse opportunities in Trilinos algorithm by keeping graph of product matrix
  - Eliminate "neighbor of neighbor" discovery
  - Eliminate memory reallocation and lookups required in serial phase

[1] Gustavson, ACM TOMS, 1978

Future: Greater nonlinear solver control of reuse within linear solver

- Currently, AMG reuse is either on or off.
  - If reuse is requested as AMG option, it will happen at every time step and at every Newton step.
  - May not want to reuse early in simulation due to startup conditions
- Features we'd like
  - Ability to enable/disable AMG reuse for particular nonlinear solves
  - Ability to vary how aggressive AMG reuse is
- Requires greater information exchange between nonlinear and linear solve
- Overall strategy for controlling this?

Conclusions

- Reducing AMG setup times important for large-scale applications
- Effectiveness of reusing setup information is problem dependent. Preliminary results show some potential.
- Opportunities for reducing cost through reuse
  - Grid transfers
  - Heavy weight smoothers
  - Matrix-matrix mult., nonlinear/linear interactions
- Still have some low-hanging fruit to pick