FASTMath Iterative Solver Technologies

Nonlinear, linear and eigenvalue iterative solvers are the key computational kernel in many application's simulations code. FASTMath has a robust research program in developing, implementing, and supporting a variety of iterative solvers for massively parallel computing systems.

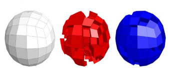
Reducing Communication in Algebraic Multigrid

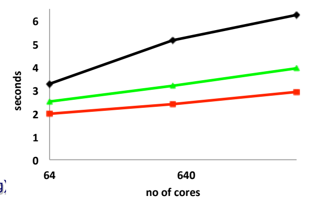
Objectives

- Algebraic multigrid (AMG) methods have shown excellent weak scalability on distributed-memory architectures, however the increasing fill-in and communication complexities on coarser levels have led to decreased performance on modern multicore architectures.
- The development of new methods with reduced communication is essential.

Additive AMG Variants

- Classical additive AMG methods have improved communication complexities per cycle, but converge significantly slower than multiplicative AMG. Mult-additive AMG, a new additive variant with reduced communication, in which the interpolation operator is replaced by a smoothed truncated prolongator, converges significantly faster than additive AMG.
- Further reductions in communication can be achieved by omitting most of the smoothing portion in the mult-additive V-cycle, leading to a simplified mult-additive variant.
- Solve times on a Linux cluster with Infiniband fat tree network for an unstructured 3D problem with jumps on a sphere



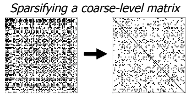


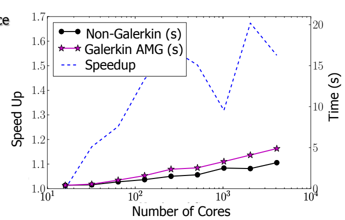
- multiplicative AMG
- mult-additive AMG
- simplified mult-additive AMG
(all methods used L1-Jacobi smoothing)

Non-Galerkin AMG

- Standard AMG suffers from fill-in on coarse-levels, which in turn induces increasing communication complexities at large scale.
- This project builds a mathematical and algorithmic framework to reduce this fill-in and hence also reduce parallel communication and runtime
- The approach safely eliminates matrix entries in the standard Galerkin coarse-level matrix to yield a non-Galerkin AMG method, while preserving important near null-space components to maintain good AMG convergence.
- Parallel test uses a set of best practices AMG parameters for the Galerkin AMG data, and then turns on non-Galerkin for a comparison.
  - Speedup is significant (~50%) and grows with core count.
  - Test uses a Linux cluster with Infiniband fat tree network.

Sparsifying a coarse-level matrix





Source: LLNL hypre Team

Computing many eigenpairs of sparse matrices

Application driver:

Material science and chemistry, especially excited state calculation

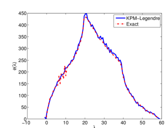
Limitations of the existing solver:

- Limited amount of parallelism in a standard Krylov subspace method (e.g. PARPACK)
- Rayleigh-Ritz (RR) procedure often the bottleneck

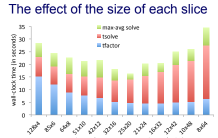
Alternative strategies:

1. Spectrum slicing

- More parallelism, no big RR calculation, but
- Need to estimate eigenvalue distribution
- Compute interior eigenvalues
  - Polynomial filtering (See Chelikowsky poster)
  - Shift-invert Lanczos
  - Contour integral method
  - Jacobi-Davidson
- Manage/optimize task distribution/load balancing

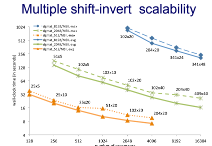


The effect of the size of each slice

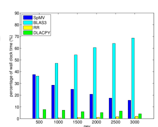


large → small

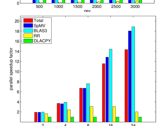
Multiple shift-invert scalability



2. Penalized trace minimization
   - formulation $\min_{X} X^T AX+\mu(X^T X-I)$
   - Algorithm characteristic
     - Preconditioned steepest descent
     - Barzilai Borwein line search
     - Few RR calculation
     - Block computation, more GEMMs
   - $R_j = AX_j + \mu X_j (X_j^T X_j - I)$
   - $X_{j+1} = X_j - \alpha M^{-1} R_j$, where $\alpha = \mathrm{trace}(\Delta R_j^T \Delta X_j)/\|\Delta R_j\|_F^2$



Performance profile



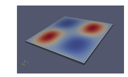
Scalability

Source: LBNL Arpack Team

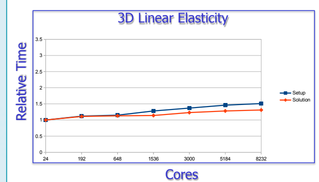
Parallel AMG Based on Energy Minimization

- Implemented in Trilinos/MueLu

$$\text{Find } P = \operatorname{argmin} \sum \|P_k\|_X$$
$$\text{s.t. sparsity}(P) \in S$$
$$N_f = PN_c$$



- Promising initial weak scaling



- Serial runs show benefit in simulations with multiple solves

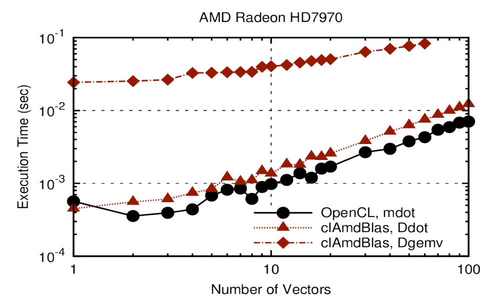
Ice Sheet Model (75x75x25)

Step	Emin(6)	Emin(1)	Emin(6,1)
2	17	30	17
8	16	32	17
12	17	33	18
18	17	36	18
23	17	36	18
28	17	34	18

Source: LLNL Trilinos Team

Broad Accelerator Support via OpenCL

Porting our existing CUDA-based operations to OpenCL allows for leveraging the performance of both current and future hardware from different vendors. By supplementing vendor-provided BLAS implementation by domain-specific kernels, high performance is obtained.



Source: ANL PETSc Team

More Information: http://www.fastmath-scidac.org or contact Lori Diachin, LLNL, diachin2@llnl.gov, 925-422-7130

SciDAC Scientific Discovery through Advanced Computing

Argonne National Laboratory · Berkeley Lab · Lawrence Livermore National Laboratory · Sandia National Laboratories · Berkeley · Columbia University · Rensselaer · The University of British Columbia