Multigrid Scaling Results on BlueGene/L

Robert D. Falgout

Center for Applied Scientific Computing

Lawrence Livermore National Laboratory

Fall Creek Falls Conference

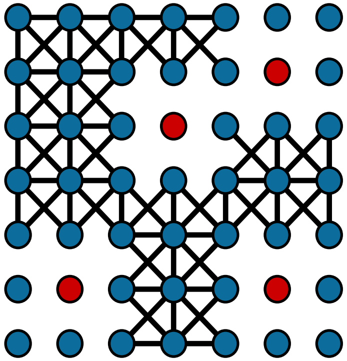
October 16-18, 2005

Efficient linear solvers are critical to good performance for many applications

- The solution of a large linear system of equations is central to most PDE-based applications
  - Large, sparse, and ill-conditioned
  - Often embedded within a nonlinear solver
  - Occur at every time step
- Multigrid methods can be scalable and efficient
  - Choice of interpolation and prolongation determine algorithmic scalability
  - Parallel implementation is a challenge, especially on thousands of processors!
- We have developed a scalable algebraic multigrid solver for use on BG/L

New coarsening and interpolation approaches improve complexity

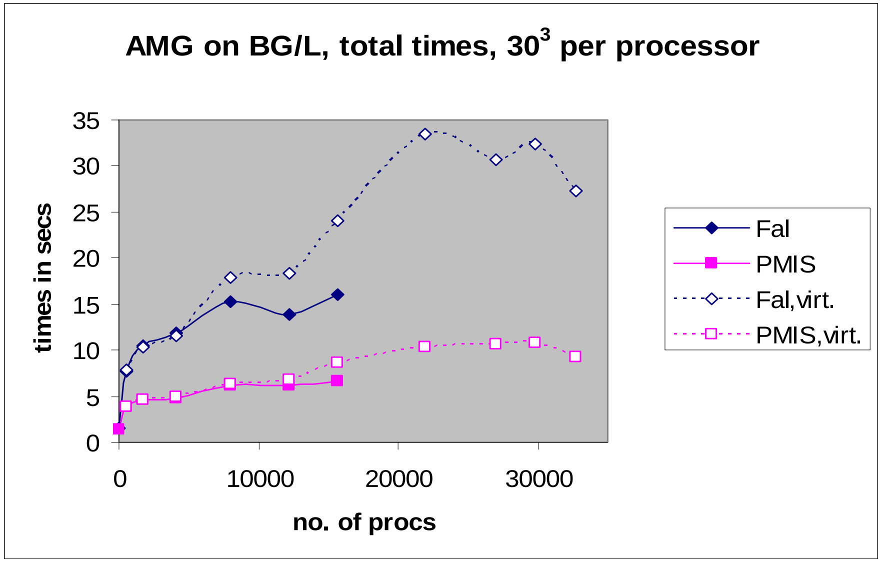
- BoomerAMG is our parallel AMG code
- One of the first parallel AMG codes (first to develop the necessary parallel coarsening algorithms)
- Used in the major ASC codes at LLNL

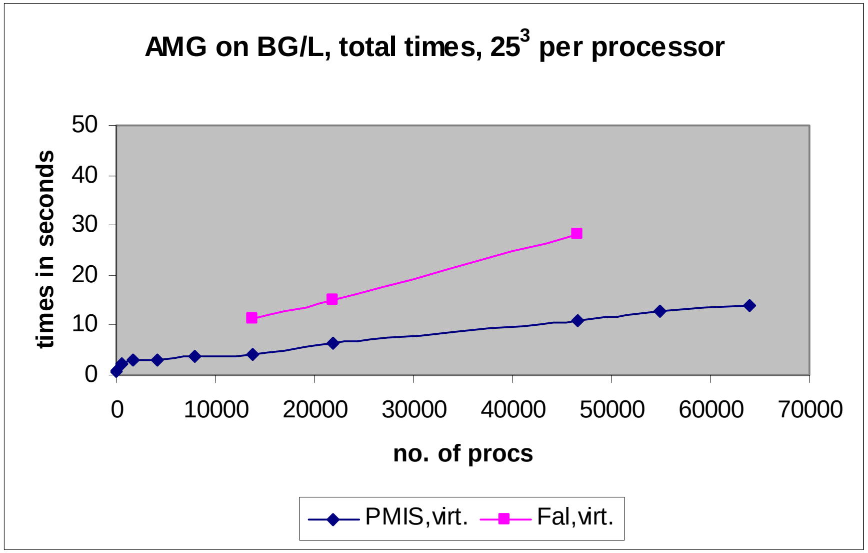


- Issue: Complexity (storage and comm) can grow significantly in parallel
- Currently no definitive solutions!

- New PMIS coarsening algorithm (with aggressive coarsening and multipass interpolation) helping to ameliorate complexity and setup costs
  - More than 2X less storage
  - Up to 2X faster solution times

BG/L results for hypre's BoomerAMG algebraic multigrid code





- More than 1B unknowns on 64K processors
- Problem size limited by 32-bit integer
- PMIS coarsening algorithm better than “Falgout” alg
- Virtual node mode not as efficient as co-proc

Focus: scalable interfaces in *hypre*

high performance preconditioners

- *hypre* software library:
  - linear solvers
  - interface between simulation codes and solvers

Both must be scalable! (BGL)

- Problem:
  - data is in distributed form
  - solvers need “nearby” data from other processors

Determine neighbor data efficiently!

Neighbor algorithm: previous approach

- Goal: neighbor information
  - “receive” processors – receive data from
  - “send” processors – send data to

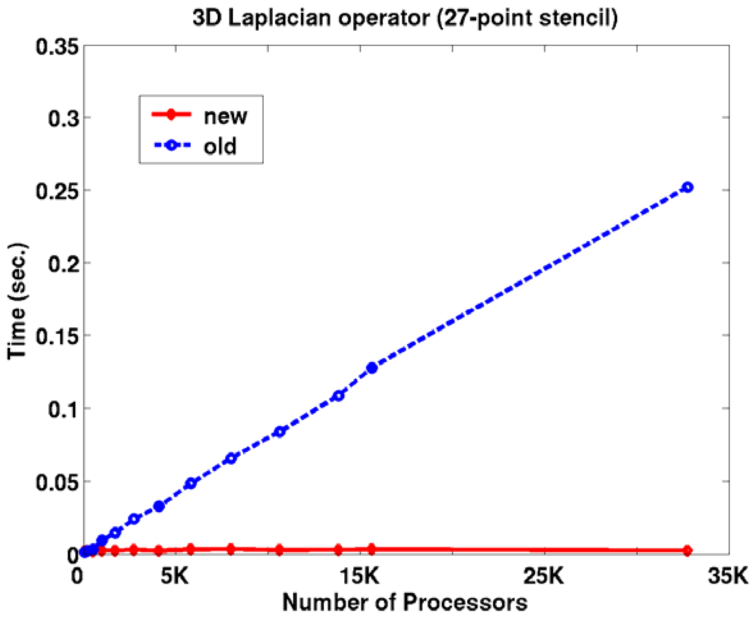
- Recall: processor only knows its own problem data

- Straightforward approach: all processors construct and store the global partition

- Cost dependency on # of processors (P):
  - communication: O(log(P))
  - computation and storage: O(P) ← not good enough!

New assumed partition algorithm enables scaling to 100K procs on BG/L

- **Answering global data distribution queries currently requires $O(P)$ storage and computational cost** (e.g., `MPI_Allgatherv`)
- **On BG/L, storing $O(P)$ data may not be practical or possible**
- **New algorithm employs the concept of an assumed partition to answer queries through a kind of rendezvous algorithm**
- **Reduces storage costs to $O(1)$ and computational costs to $O(\log(p))$**
- **Developed and demonstrated for *hypre*'s IJ and SEMI interfaces**
- **Algorithm and code are useful in more general contexts**



Multigrid methods can be efficiently implemented on BG/L

- We have demonstrated algorithmic and parallel scalability on the full LLNL BG/L system

- Future improvements
  - Algorithmic
  - Virtual node

- Software is available via *hypre* library via the SciDAC TOPS ISIC