A parallel hybrid linear solver for EM simulations

Ichitaro Yamazaki, Xiaoye Li, and Esmond Ng

Lawrence Berkeley National Laboratory

ComPASS Collaboration meeting,
Boulder, Colorado, 10.05.09

In ComPASS EM simulations, solving linear systems is often memory bottleneck:

$$Ax = b,$$

where A is

- **sparse but large**
  direct methods are robust, but require large memory.
- **ill-conditioned and highly-indefinite**
  preconditioned iterative methods require less memory, but suffer from slow or no convergence.

Hybrid method has the potential of balancing robustness of direct methods with efficiency of preconditioned iterative methods.

Schur complement method: reorder A into a 2×2 block system:

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$$

where

- A_{11} is *interior domains*, A_{22} is *separators*, and A_{21} and A_{12} are the *interfaces* between A_{11} and A_{22} such that

$$\left(\begin{array}{c|c} A_{11} & A_{12} \\ \hline A_{21} & A_{22} \end{array}\right) = \left(\begin{array}{cccc|c} A_{11}^{(1)} & & & & A_{12}^{(1)} \\ & A_{11}^{(2)} & & & A_{12}^{(2)} \\ & & \ddots & & \vdots \\ & & & A_{11}^{(k)} & A_{12}^{(k)} \\ \hline A_{21}^{(1)} & A_{21}^{(2)} & \dots & A_{21}^{(k)} & A_{22} \end{array}\right).$$

- interior domains can be factored in parallel
  $\Longrightarrow$ great parallel performance!!

Schur complement method: with a block Gaussian elimination, we obtain

$$\begin{pmatrix} A_{11} & A_{12} \\ 0 & S \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} I & 0 \\ -A_{21}A_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix},$$

where $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ is the Schur complement.

Hence, the solution to the linear system can be computed by

1. solving $Sx_2 = b_2 - A_{21}A_{11}^{-1}b_1$
2. solving $A_{11}x_1 = b_1 - A_{12}x_2$

Most of fill occurs in S, while A_{11} is block diagonal.

Hybrid method: Schur complement method

With a block Gaussian elimination, we have

$$\begin{pmatrix} A_{11} & A_{12} \\ 0 & S \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} I & 0 \\ -A_{21}A_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix},$$

where $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ is the Schur complement.

Hence, the solution to the linear system can be computed by

1. solving $Sx_2 = b_2 - A_{21}A_{11}^{-1}b_1$ by an iterative method.
2. solving $A_{11}x_1 = b_1 - A_{12}x_2$

Most of fill occurs in S, while A_{11} is block diagonal.

Hybrid method: Schur complement method

With a block Gaussian elimination, we have

$$\begin{pmatrix} A_{11} & A_{12} \\ 0 & S \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} I & 0 \\ -A_{21}A_{11}^{-1} & I \end{pmatrix} \begin{pmatrix} b_1 \\ b_2 \end{pmatrix},$$

where $S = A_{22} - A_{21}A_{11}^{-1}A_{12}$ is the Schur complement.

Hence, the solution to the linear system can be computed by

1. solving $Sx_2 = b_2 - A_{21}A_{11}^{-1}b_1$ by an iterative method.
2. solving $A_{11}x_1 = b_1 - A_{12}x_2$ by a direct method.

Most of fill occurs in S, while A_{11} is block diagonal.

HIPS (Hybrid Iterative Parallel Solver):

- developed by P. Henon and Y. Saad, 2008.
- features:
  - each interior domain is factored by a single processor, number of interior domains $\geq$ number of processors.
  - fill in ILU is allowed between separators adjacent to same domain.
- advantage:
  time to compute the preconditioner scales well.
- disadvantage:
  to run on many processors, many interior domains are needed, which results in slow or no convergence
  - large Schur complement and poor preconditioner (fill is restricted within small blocks).

Our objectives:

to overcome the limitations of HIPS;

- factor each interior domain using multiple processors
  $\Longrightarrow$ two-level parallelization (each domains in parallel)
  more processors with a fixed Schur complement.
- improve flexibility and robustness of the solver
  $\Longrightarrow$ faster convergence to solve Schur complement system.
- exploit state-of-the-art techniques
  $\Longrightarrow$ superior parallel performance.

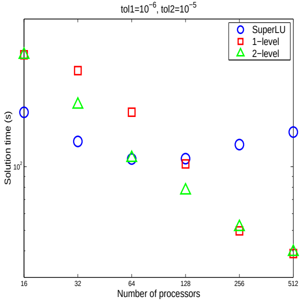
to provide the flexiblity and robustness to solve
large-scale highly-indefinite systems on many processors!!

New implementation.

- exploiting state-of-the-art software.
    - **PT-SCOTCH** to extract interior domains.
    - **SuperLU_DIST** to factor each interior domain.
    - **PETSc** to solve the Schur complement system.
- flexible and robust Schur complement solution.
  – efficient & stable formulation of approx. Schur
    - sparsity to reduce operation count and communication.
    - preprocessing for numerical stability and sparsity.
    - load-balancing and comm. strategies for good parallel performance.

  – choice of different preconditioners and solvers
    - SuperLU_DIST, Phidal, or any preconditioner from PETSc.
    - any solvers from PETSc

Preliminary results: tdr455k with $n = 2,738,556$.

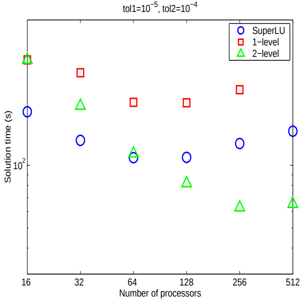


doms	16	32	63	128	256
itrs	11	15	15	17	16

- 1-level parallelization;
  one processor per domain.
- 2-level parallelization;
  multiple processors per domains.
  - 16 interior domains.
  - processors evenly distributed among interior domains.
  - 16 processors to solve Schur complement system.

- hybrid solver scaled better than direct solver.
- covergence (GMRES) only slightly degrades with more domains.
- HIPS required 150 iterations and took longer for 16 domains, and failed to converge within 1000 iterations for 32 domains.

Preliminary results: **tdr455k** with $n = 2,738,556$.



doms	16	32	63	128	256
itrs	32	60	116	205	290

- large drop tolerances to reduce memory cost.
- 1-level parallelization:
  \# of processors = \# of domains.

- 1-level needed more iterations with more processors.
- 2-level parallelization scaled better.
- working to improve the parallel performance.

Features of our implementation

- multiple processors to solve each interior systems:
  - more processors with fixed Schur complement.
- flexibility to solve Schur complement system:
  - choice of different preconditioners and iterative methods.
  - subset of processors to solve Schur complement system.
- better parallel performance than other state-of-the-art solvers.

Current work:

- improving scalability of parallel implementation.
  - improving load-balance to solve interior domains.
  - studying different communication strategies.
  - devleoping parallel ILU based on SuperLU_DIST.
- conducting further experimentation.
  - solving a large system on many processors.
  - profiling memory usage per processor.
- extending to solve complex systems.