GPU-capable ILU preconditioner in support of HMC computation and Chroma software

Sherry Li, Yang Liu, LBNL

Eloy Romero Alcalde, JLab

Andreas Stathopoulos, William & Mary

SciDAC PI Meeting, September 16-18, 2025

Optimizing SuperLU for Chroma – two usage scenarios

- Preconditioner for coarse operator at any multigrid level – global, strong scaling is critical (FY24 effort)

  Optimizing sparse triangular solve (SpTRSV); released in SuperLU_DIST 9.0.0

  - 3.5x speedups comparing to baseline 3D algorithm on NERSC Cori
  - 1 GPU to 64 GPUs (3D SpTRSV) with up to 6.5x speedups on Crusher (Frontier)
  - 4-8 GPUs (2D SpTRSV) to 256 GPUs (3D SpTRSV) on NERSC Perlmutter

- Domain decomposition preconditioner where domains can be on a single computing node – local, GPU throughout is critical (FY25 effort)
  - GPU kernel optimization
  - Batch organization to maximize GPU throughput
  - ILU(0) numerical factorization

1. Optimizing SpTRSV on single GPU node

- Optimized several kernels, especially for the common case NRHS=1

NRRHS	1	2	4	8	16
Total SpTRSV (ms)	19	27	43	75	146
Kernel time (ms)	15	19.8	30	45.8	78

- FY26 task
  – Make SpTRSV entirely GPU-resident, eliminate GPU-CPU transfer time

2. New feature: batched interface to improve GPU throughput

- Large overhead for execution of smaller operations
  - Kernel launches
  - Memory allocations and transfers

- Batch a large number of operations using a single kernel call
  - GPU libraries such as MAGMA and KBLAS
  - Variable size operations can be challenging to implement

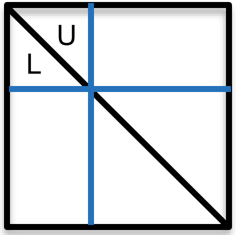
Fine-grain batched approach

Need to change internal solver

Unbatched

For k=1:N supernode-panels:

1. Diagonal block LU
2. TRSM to factor panels
3. GEMM to form Schur complement
4. Scatter Schur to destination

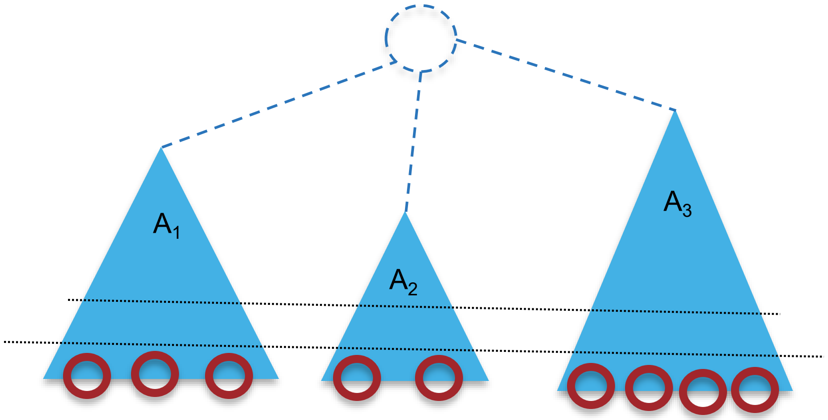


Batched

Input as a block diagonal form

$$A = \begin{bmatrix} A_1 & & \\ & A_2 & \\ & & A_3 \end{bmatrix}$$

Elimination forest-of-trees – based on struct(A+A')



Level-by-level batch

Leverage variable-size batched dense operations

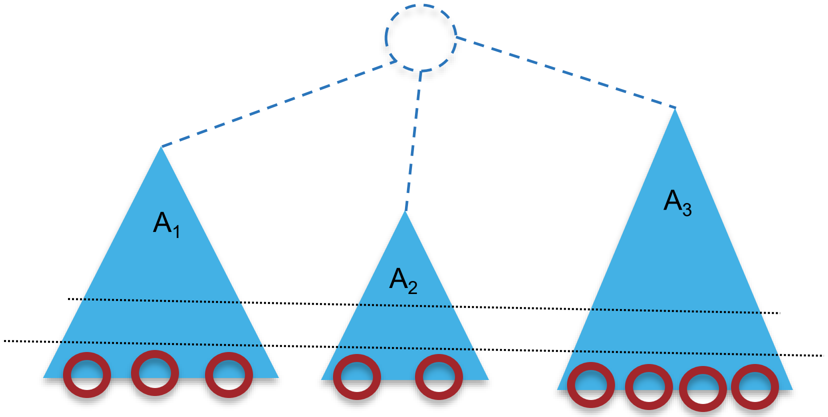
- Pre-pivoting done in preprocessing step
  - New non-uniform batched LU without pivoting introduced to MAGMA
  - Supports tiny diagonal element replacement
- Use MAGMA calls on GPU device

For level = 1 : root

- magma_dgetrf_vbatched ()
- magmablas_dtrsm_vbatched ()
- magmablas_dgemm_vbatched ()

- scatterGPU_batch<<< … >>>

Elimination forest-of-trees – based on struct(A+A')

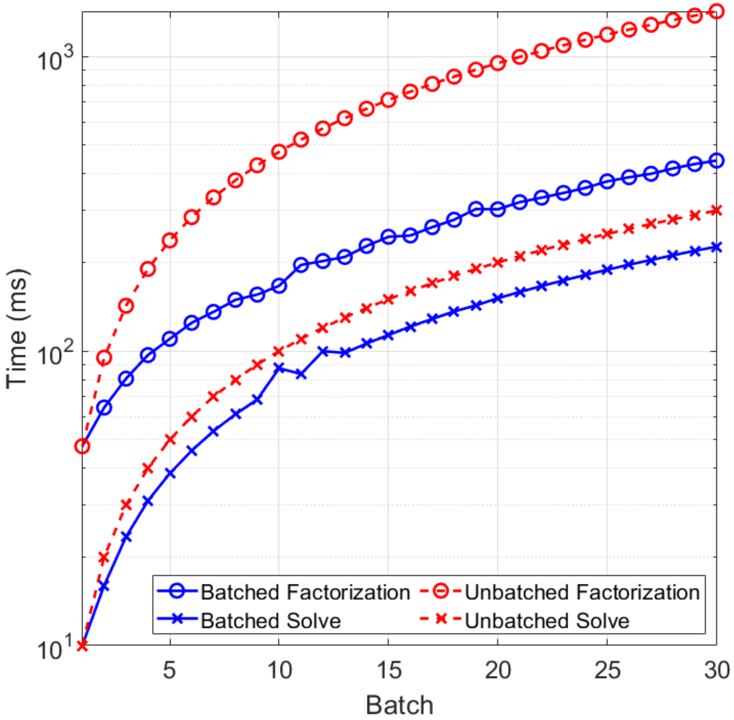


Level-by-level batch

Batch interface

```
pdgssvx3d_csc_batch
(
    superlu_dist_options_t *options, // options for algorithm parameters
    int batchCount, // number of systems in the batch
    int m, // matrix row dimension
    int n, // matrix column dimension
    int nnz, // number of non-zero entries
    int nrhs, // number of right-hand-sides
    handle_t  *SparseMatrix_handles, // array of sparse matrix handles, each pointing to
                                     // compressed storage
    double **RHSptr, // array of pointers to dense RHS
    int *ldRHS, // leading dimensions of RHS
    double **ReqPtr, // pointers to row scaling vectors
    double **CeqPtr, // pointers to column scaling vectors
    int **RpivPtr, // pointers to row permutation vectors
    int **CpivPtr, // pointers to column permutation vectors
    DiagScale_t *DiagScale, // indicate how equilibration is done for each matrix
    handle_t *F, // array of handles pointing to factored matrices
    double **Xptr, // pointers to dense solution X
    int *ldX, // leading dimensions of X
    double **Berrs, // pointers to backward errors
    gridinfo3d_t *grid3d,
    SuperLUStat_t *stat,
    int *info
)
```

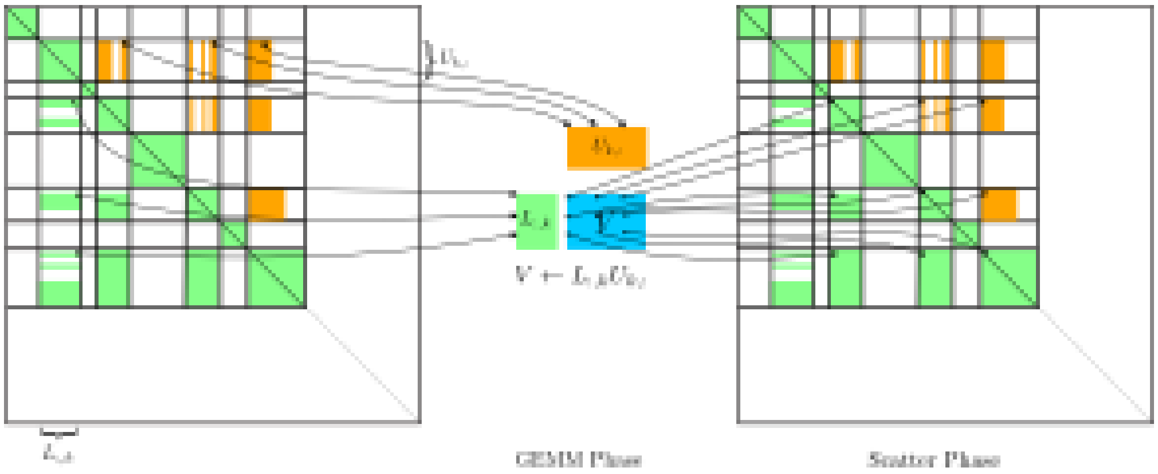
A matrix from SuiteSparse / ibm_matrix_2 (N=51,448)



Semiconductor Device Problem
Nonsymmetric

3. New feature: Compute numerical ILU(0) factor

- ILU(0) uses sparsity pattern of original matrix A – relatively easy
  - Symbolic factorization is trivial
  - Numerical factorization needs to skip the unwanted fill-in during Scatter



Next steps in FY26

- GPU resident SpTRSV
- Iterative construction of ILU
- ILU(k) if needed