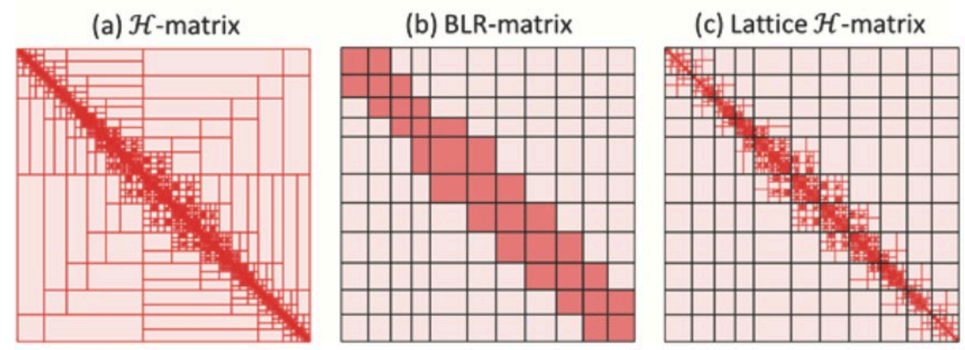
Optimizations of Hierarchical matrix Library for Boundary Element Methods

Tetsuya Hoshino (Information Technology Center, Nagoya University,
E-mail: hoshino@cc.nagoya-u.ac.jp)

Lattice $\mathcal{H}$-matrix

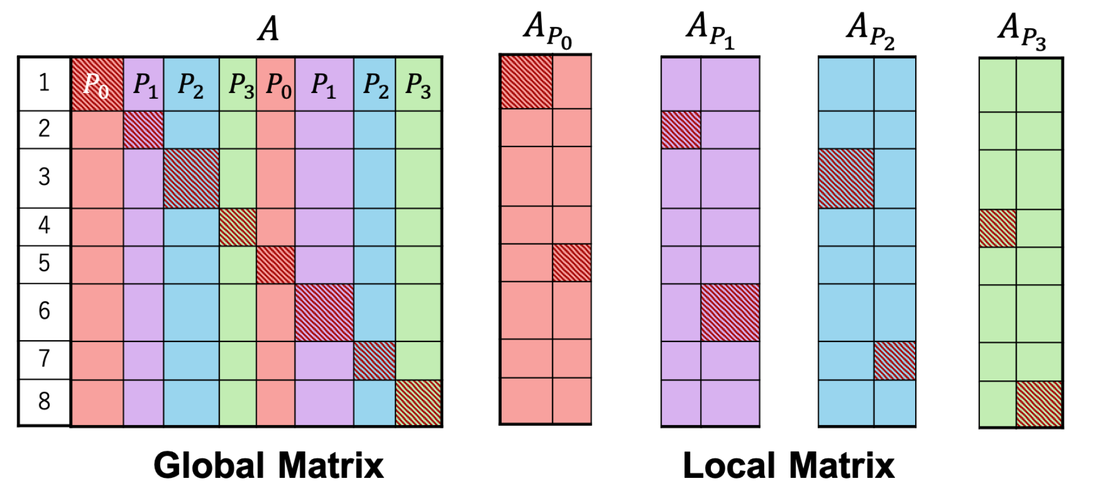
- Hierarchical ($\mathcal{H}$-) matrices are commonly used with the Boundary Element Method (BEM)
- Reduce memory from $\mathcal{O}(N^2)$ (dense) to $\mathcal{O}(N \log N)$ ($\mathcal{H}$-matrices).
- Lattice $\mathcal{H}$-matrices are hybrid of BLR and $\mathcal{H}$-matrices
  - Lower compression than pure $\mathcal{H}$-matrices
  - Block structure eliminates complex communication, making it suitable for large-scale distributed systems
- Near the diagonal is heavier; effective load balancing with communication hiding is crucial for high performance

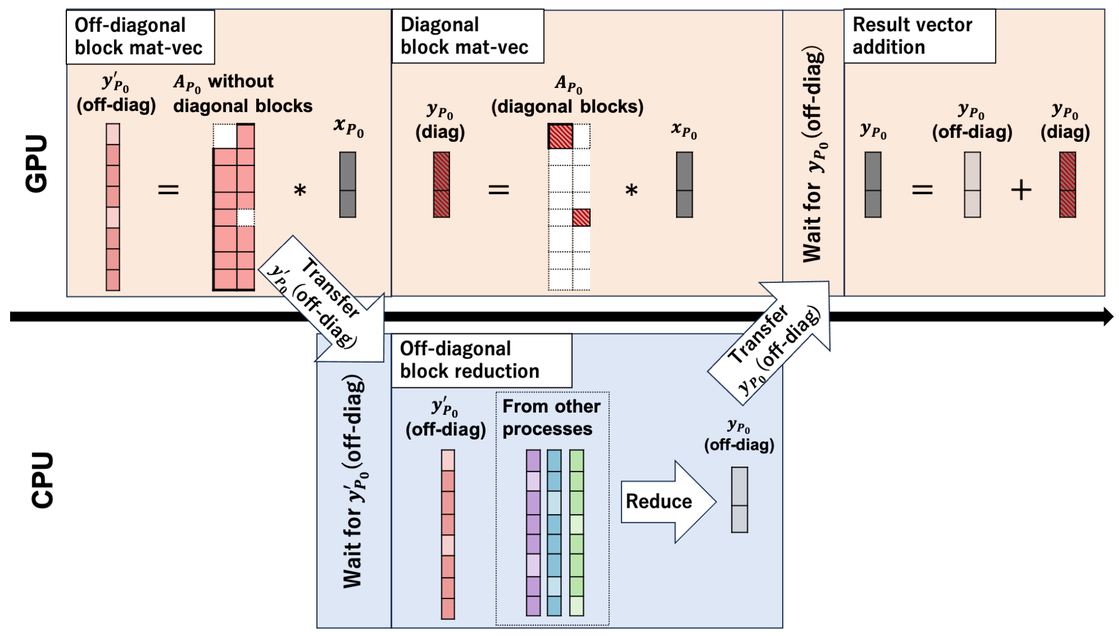


The dark red regions represent dense sub-matrices, while the light red regions represent low-rank sub-matrices.

Load balancing and communication overlapping methods for GPU systems

- Speeding up the most frequently used kernel: iterative solvers based on Lattice $\mathcal{H}$-matrix–vector products.
  - Each submatrix–vector product can be executed independently, but their results must be accumulated
    - Row-wise reduce is required.
  - The left-hand-side vector at iteration t becomes the right-hand-side vector at $t + 1$, communication must account for this dependency
    - Column-wise broadcast is required.
- Load balancing: 1D cyclic partitioning along columns to evenly split heavy diagonal blocks; requires row-wise reduce across processes but no column-wise communication
- Communication hiding: split work into off-diagonal and diagonal phases; overlap the reduce of off-diagonal matvec results with diagonal-block computation; finally accumulate diagonal results
- Communication path: CPU-side communication (no GPUDirect) to avoid performing reductions on the GPU

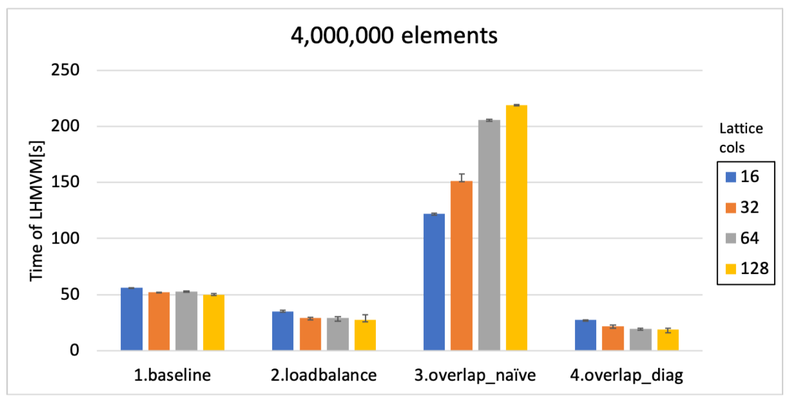


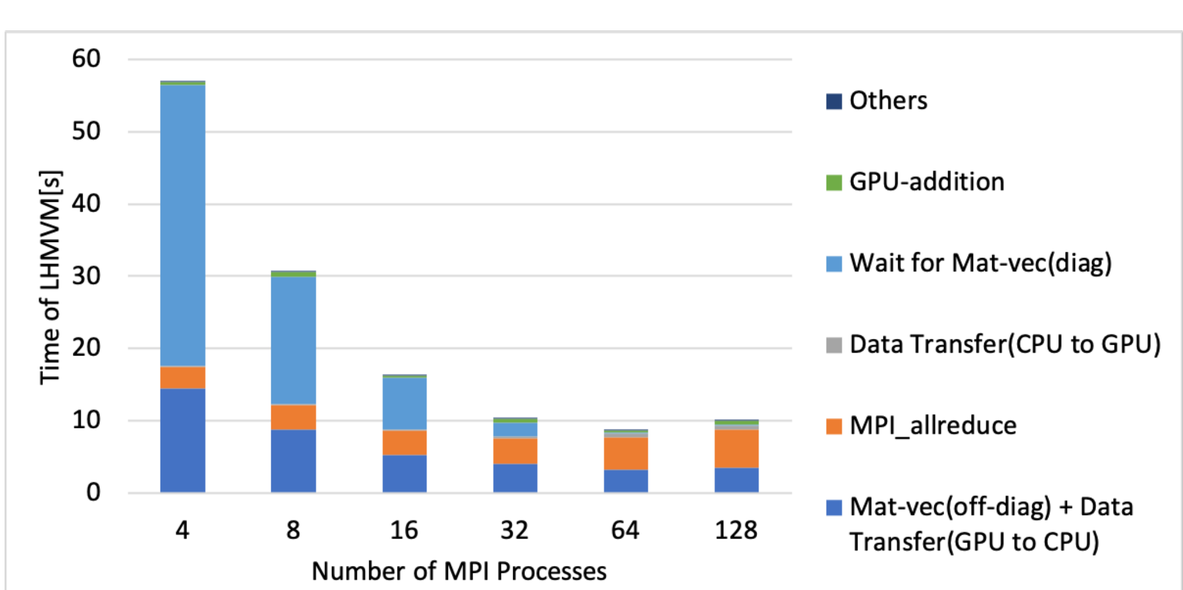


Experimental results

- Target: scaling evaluation on an earthquake-cycle simulation.
- Platform: up to 128 GPUs on a GH200 cluster.
- Strong scaling: from 4 → 64 GPUs, ~6.5× speedup.

16-GPU evaluation: ~2× speedup over the baseline implementation via load balancing and communication hiding.





Strong scaling result on GH200 cluster.