Offloading the IBM Workloads for Efficient LBM Fluid Simulations on Grace Hopper

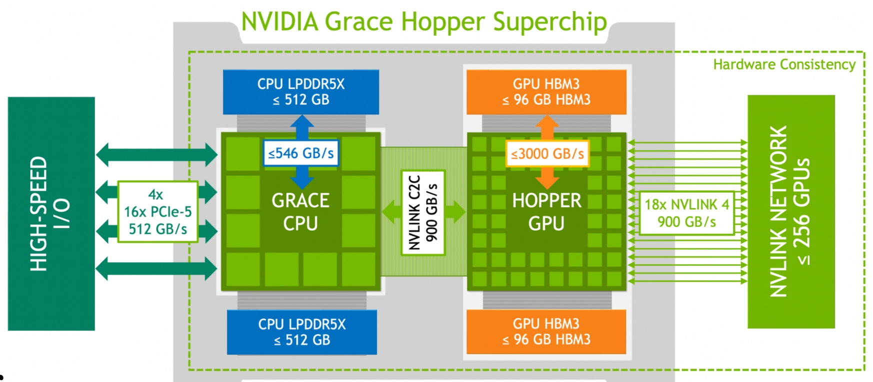
Yize Yang[1], Takashi Shimokawabe[2]

[1]Graduate School of Engineering, The University of Tokyo, Tokyo, Japan

[2]Information Technology Center, The University of Tokyo, Tokyo, Japan

Introduction

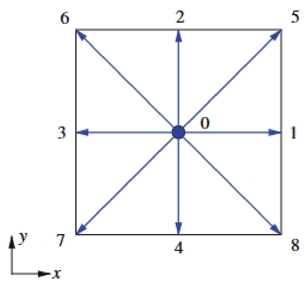
- Hardware Trend: Tightly Coupled CPU–GPU Architectures
  - Modern HPC systems are moving toward tightly coupled CPU–GPU architectures.
  - Faster CPU–GPU communication enables finer-grained workload distribution.
  - Tightly coupled architectures enable new collaboration execution strategies.
  - Workloads can be assigned based on their computational characteristics.

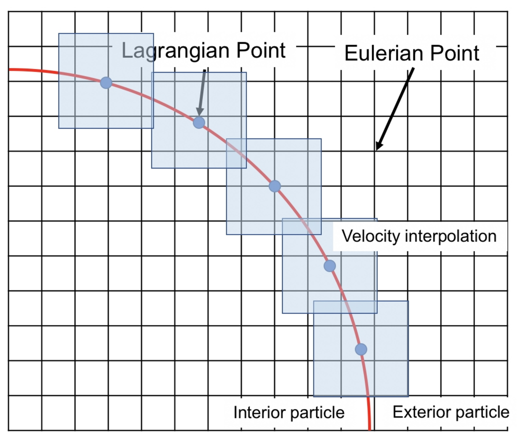


Source: NVIDIA

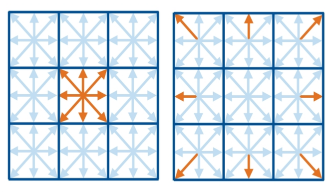
- NVIDIA Grace Hopper Overview
  - NVIDIA Grace Hopper is a tightly coupled CPU–GPU superchip.
  - It integrates a Grace CPU and a Hopper GPU in a single package.
  - The two processors are connected via NVLink-C2C.

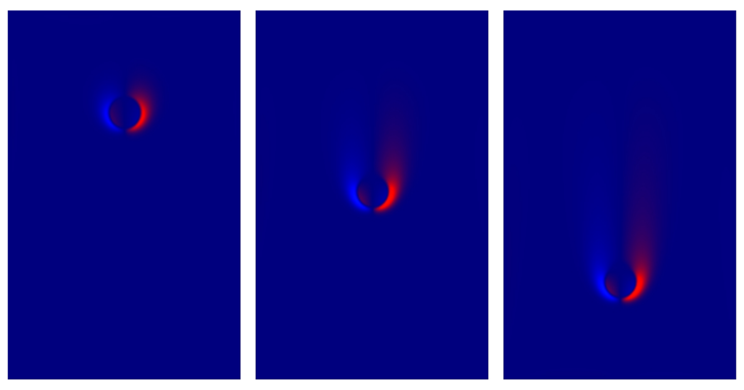
- Lattice Boltzmann Method (LBM)
  - LBM solves fluid flow using mesoscopic particle distributions.
  - The method operates on a fixed Eulerian lattice.
  - It is highly parallel and well suited for GPU acceleration.





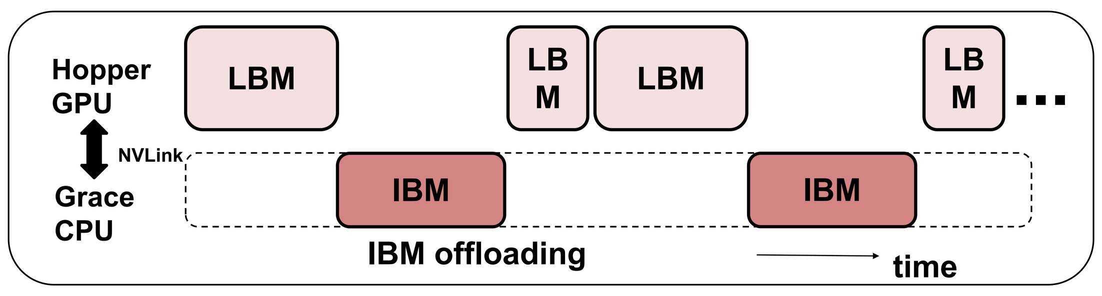
- Immersed Boundary Method (IBM)
  - IBM represents solid boundaries using Lagrangian points.
  - IBM couples LBM Eulerian fluid grids with Lagrangian boundaries.
  - Involves irregular memory accesses that challenge performance on GPUs.



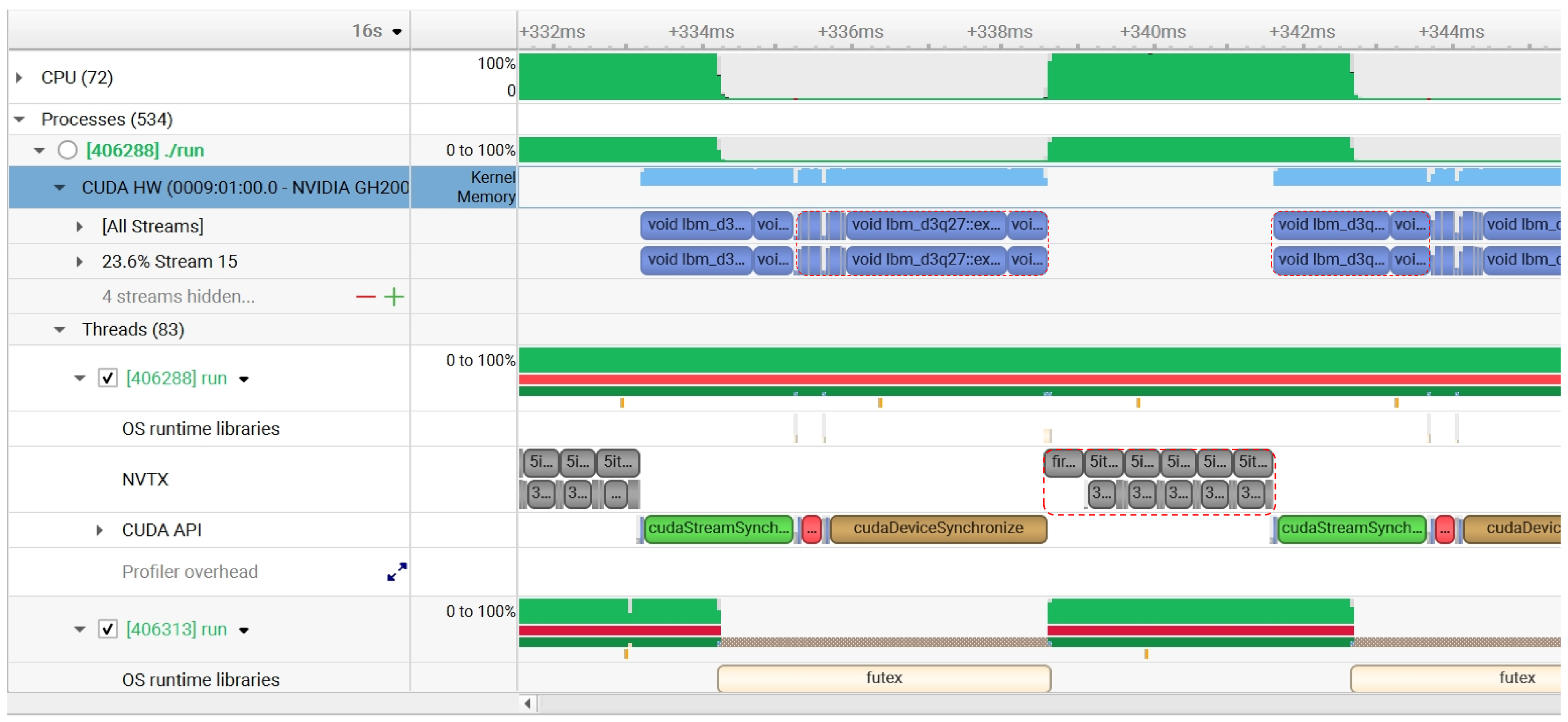


- Platform: Miyabi-G Supercomputer at JCAHPC
  - Designed for large-scale scientific simulations.
  - The node of Miyabi-G system features NVIDIA Grace Hopper GH200.
  - Experiments are conducted on the Miyabi-G.

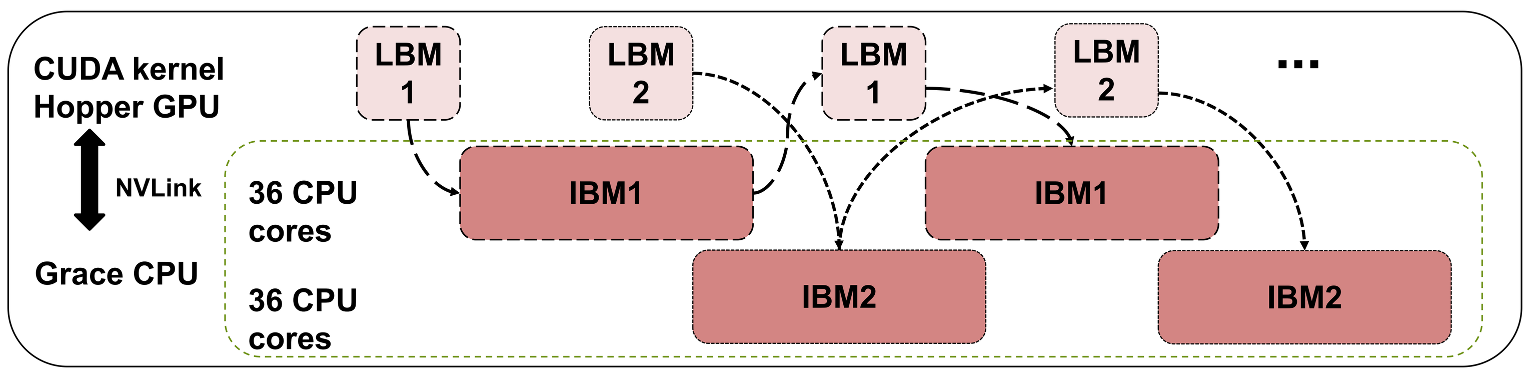
Methodology



- We study LBM–IBM coupling simulations on Grace Hopper GH200.
- The fluid solver (LBM) is kept on the Hopper GPU.
- The IBM is selectively (vforce) offloaded to the Grace CPU.

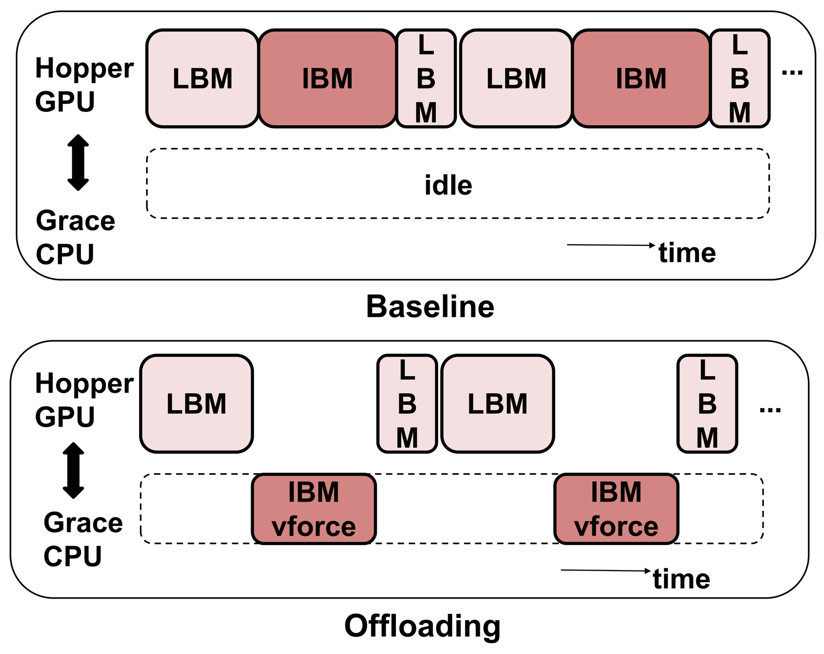


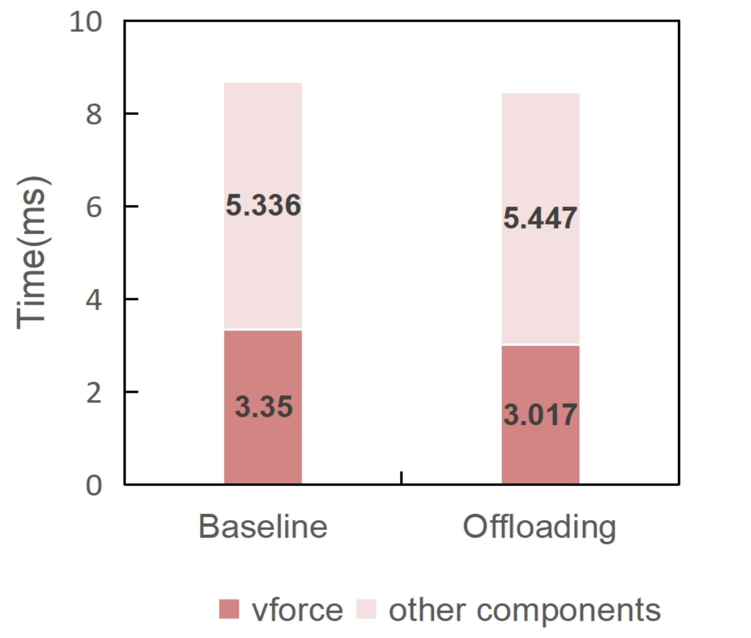
- NVIDIA Nsight system is used to profile program execution.
- NVTX annotations are inserted to mark CPU offloaded IBM execution in the Nsight profiler.
- Memory Optimization
  - IBM vforce related variables are allocated in CPU memory.
  - Data memory placement is shown to have a significant impact on performance.
- OpenMP is used to exploit the 72-core capability of the Grace CPU.
  - The offloaded IBM computation is parallelized across CPU cores.
  - This improves the performance of irregular IBM workloads.
- CPU–GPU synchronization is used to ensure data dependencies between LBM and IBM.



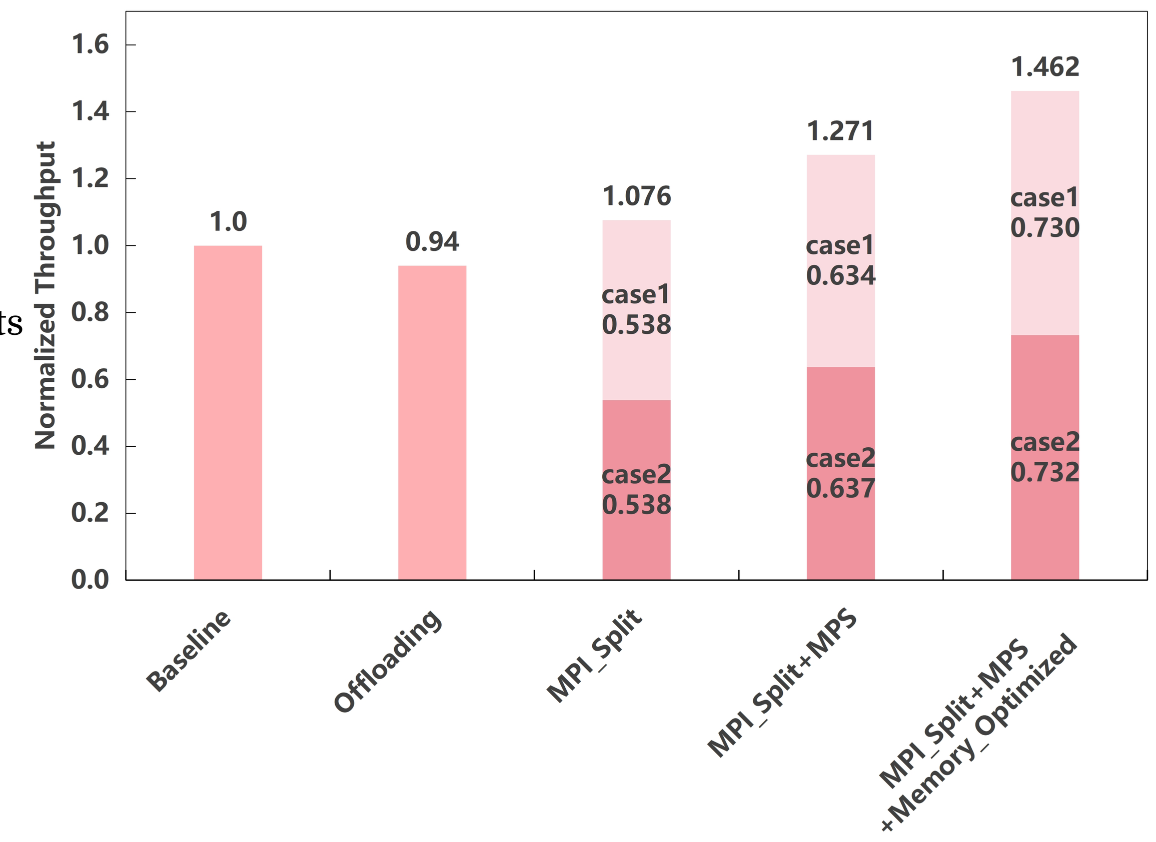
- Extending IBM Offloading with MPI-Split Double-Case Execution
  - MPI_Split is used to run two processes at the same time.
  - Each process executes one simulation case.
  - Idle CPU–GPU bubbles caused by offloading are filled after 2 case interleaving
  - Overall system throughput is improved.
- NVIDIA Multi-Process Service (MPS)
  - NVIDIA MPS is used to enable two processes to share one GPU, improving overall system throughput furthermore.
  - GPU resources are better utilized under multi-process workloads when using MPS.

Results

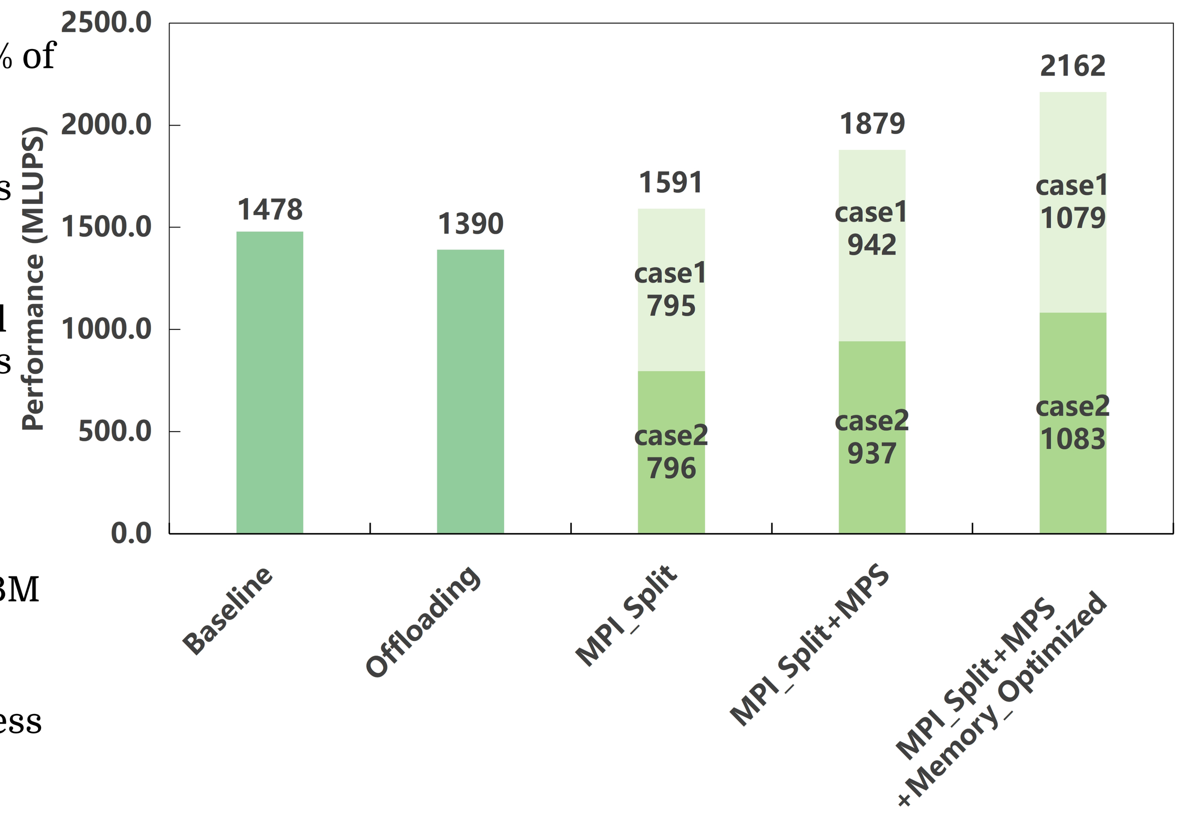




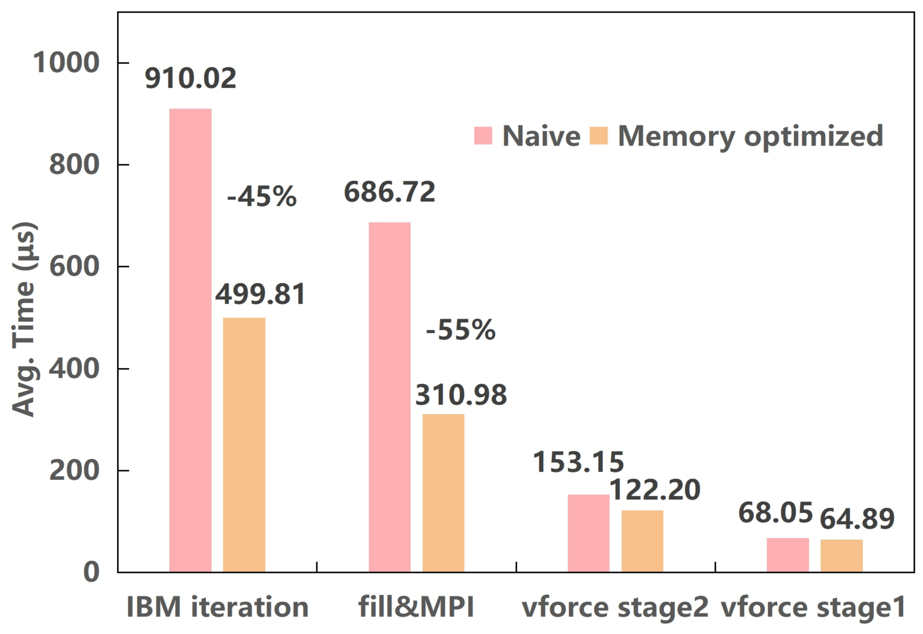
- IBM Time Comparison after IBM Offloaded to CPU
  - IBM vforce runtime decreases from 3.35 ms to 3.017 ms compared with GPU baseline.
  - Confirms the effectiveness of IBM offloading to CPU.

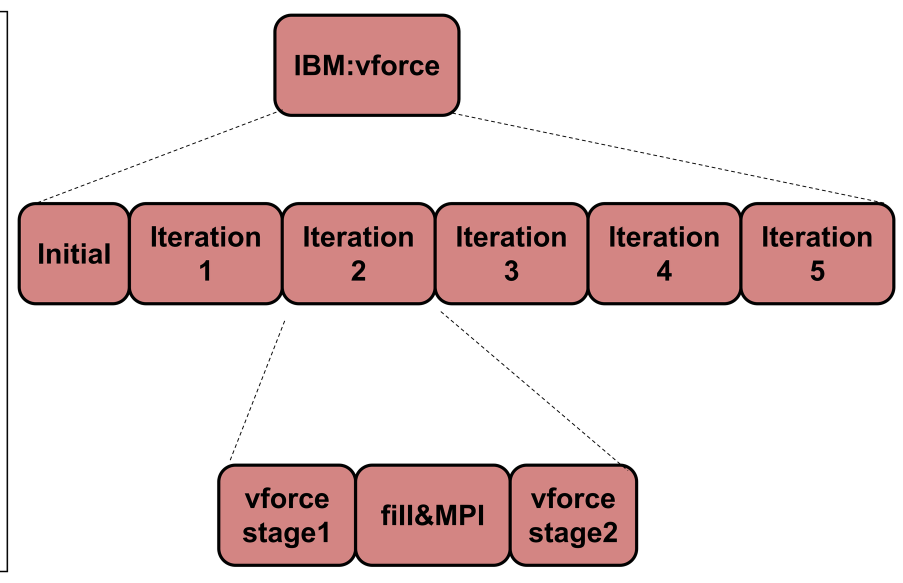


- System-Level Performance Results
  - System-level performance is evaluated using normalized throughput and MLUPS (Million Lattice Updates Per Second).
  - Baseline: fully GPU-Based execution as baseline.
  - Offloading: throughput is reduced due to the overhead introduced by CPU–GPU synchronization.
  - MPI_Split: overall system throughput increases by 7.6% of 2 cases.
  - MPI_Split + MPS: overall system throughput increases by 27.1% of 2 cases.
  - MPI_Split + MPS + Memory_Optimized: overall system throughput increases by 46.2% of 2 cases.



- Performance Improvement of Memory Optimization
  - Average execution time of IBM iterations reduced 45% after memory optimization.
  - Improved CPU memory access locality contributes to the performance gain.





Conclusion

- Tightly coupled CPU–GPU systems such as Grace Hopper unlock new potential for high-performance computing beyond LBM–IBM fluid simulations.
- Idle processing resources can be effectively utilized to improve overall system efficiency.
- Matching workloads to the most suitable computing device is critical for performance.
- System-level CPU–GPU cooperation can significantly enhance overall throughput.

Future Work

- Workload balance to maximize CPU core utilization while keeping GPU efficiency.
- Scalability: Extend IBM-LBM cases performing on multiple GPUs.

Acknowledgements

This work was partly supported by JSPS KAKENHI Grant Number JP24K02947 and JHPCN project jh250037.

Email: yang-y@g.ecc.u-tokyo.ac.jp