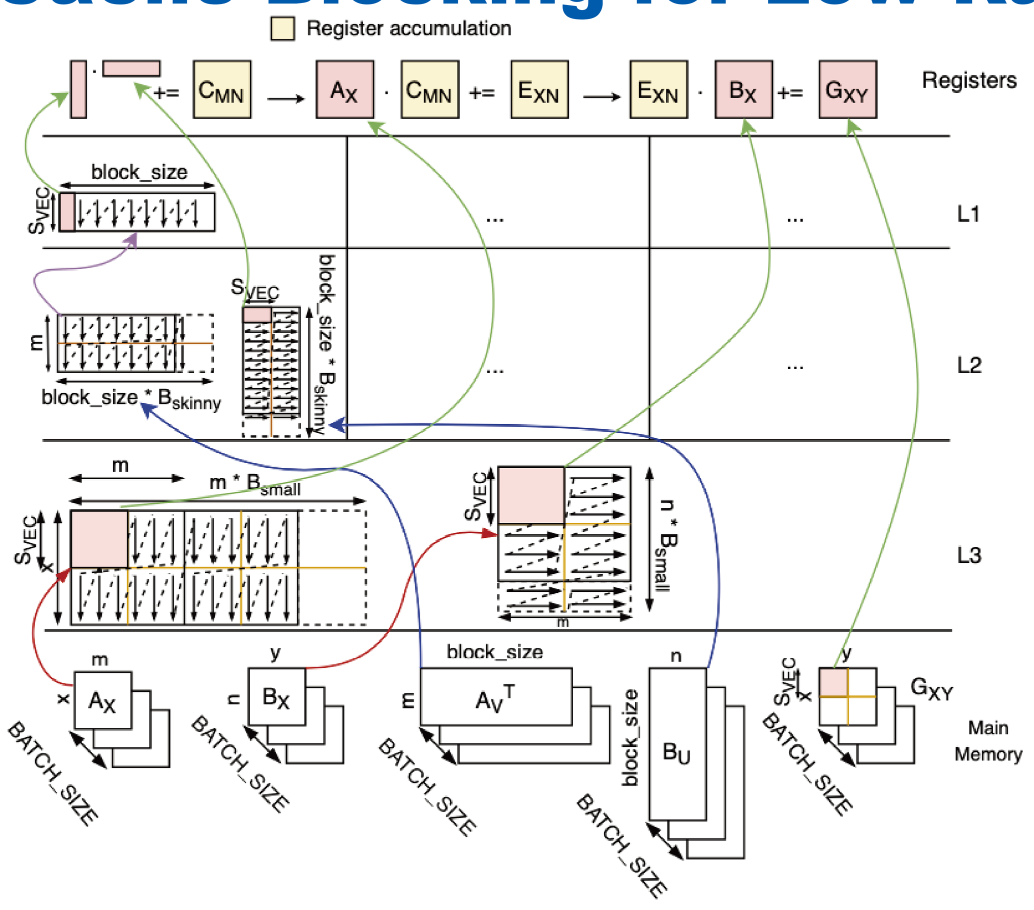
Applications on TSUBAME3.0

Fast Algorithms and Large Scale CFD

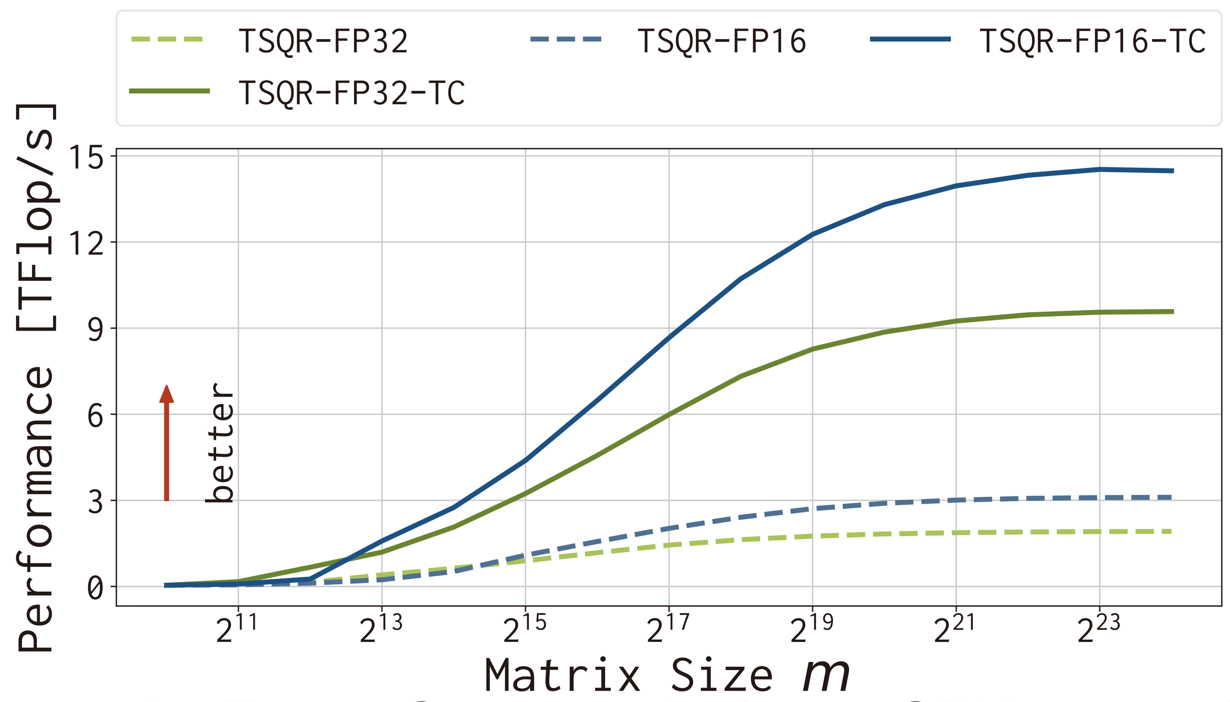
Fast Algorithms for HPC and Deep Learning

Cache Blocking for Low-Rank Matrices



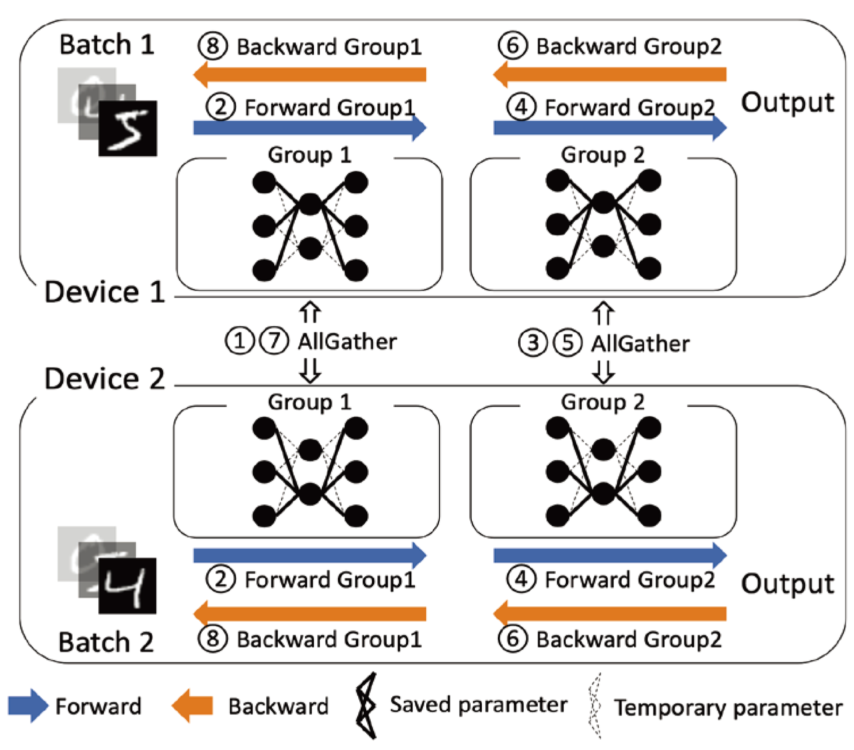
Structured low-rank matrix operations result in a batch of small and rectangular matrices. This causes standard BLAS libraries to perform poorly. Even batched MKL is suboptimal when the small matrices are skinny. We develop a highly optmized inner kernel for such matrices and show that we can outperform batched MKL by 2x, which can be used in structured low-rank matrix operations.

TSQR on TensorCores



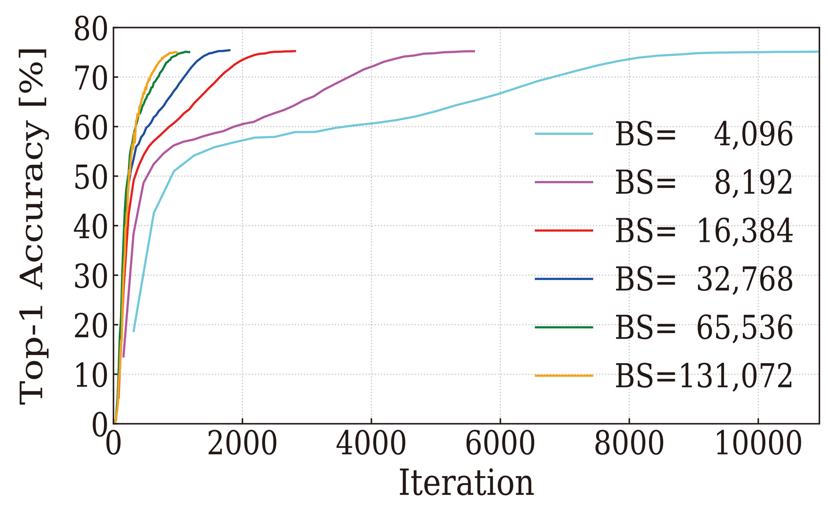
The compression of H-matrices involves a QR decomposition of tall and skinny matrices --TSQR. Since the goal of the compression is to perform a low-rank approximation of the original dense matrix, the accuracy required for this operation is not so high. This means that we can use the TensorCores on the latest GPUs, to perform the TSQR much faster than usual. We investivate the amount of performance we can achieve when using different precision (FP16 and FP32) along with the TensorCore. We find that we can get close to 15 TFlops for tall matrices.

Memory Efficient Deep Learning



The size of deep neural networks continues to grow rapidly, where models like GPT-3 and Megatron-LM exceed the memory capacity of a single GPU. The trend in computer architecture where the arithmetic throughput is growing faster than memory capacity, suggests that memory consumption is a critical issue in deep learning. We reduce the memory usage of deep neural networks by scattering, recomputing, and offloading the model parameters and activations.

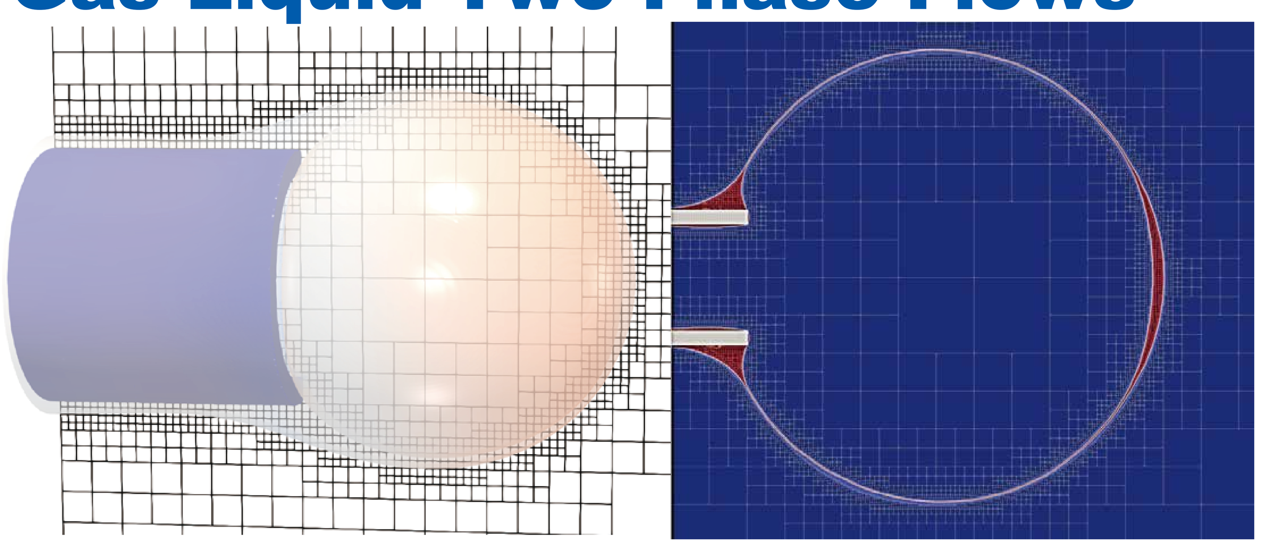
Training ImageNet in 2 minutes on 2048 GPUs



Data-parallel training of deep neural networks suffers from the large-batch problem, where the generalization gap increases as the mini-batch size increases proportional to the number of GPUs. Using the Kronecker Factorization mention on the right, we are able to use a second order optimization method with minimum overhead, which allows us to retain the convergence even for extremely large batch sizes. We trained ImageNet in 2 minutes on 2048 GPUs, using a batch size of 131,072.

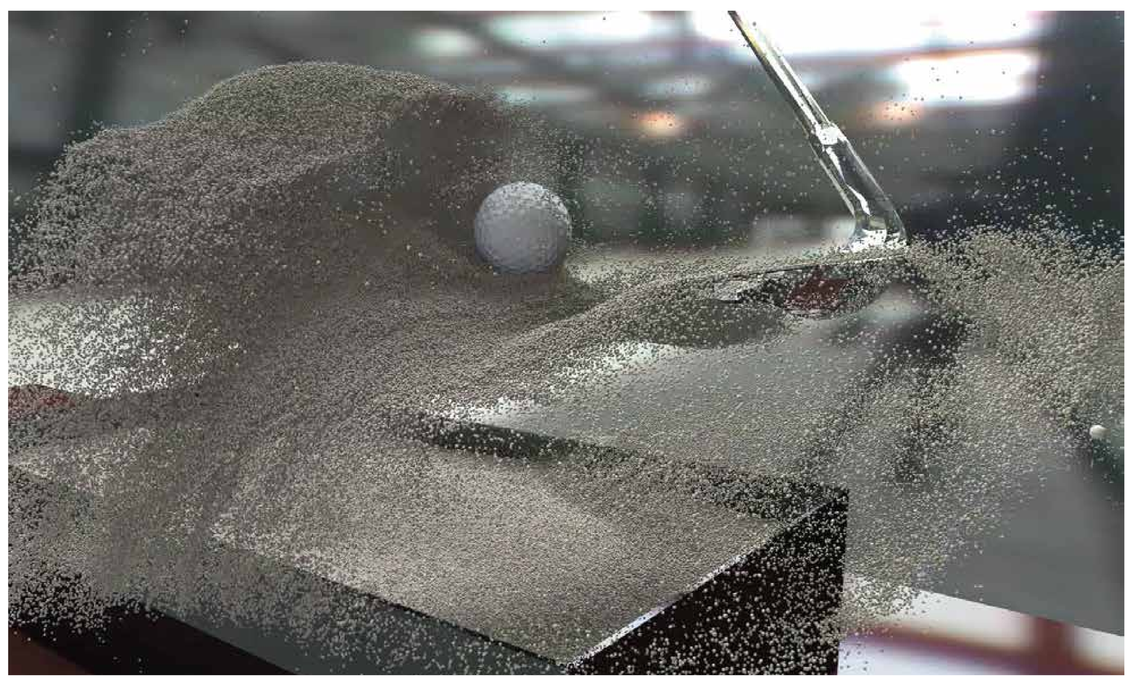
Large-scale Mesh-based and Particle-based Simulations

Weak-Compressible Flow Computations for Gas-Liquid Two-Phase Flows



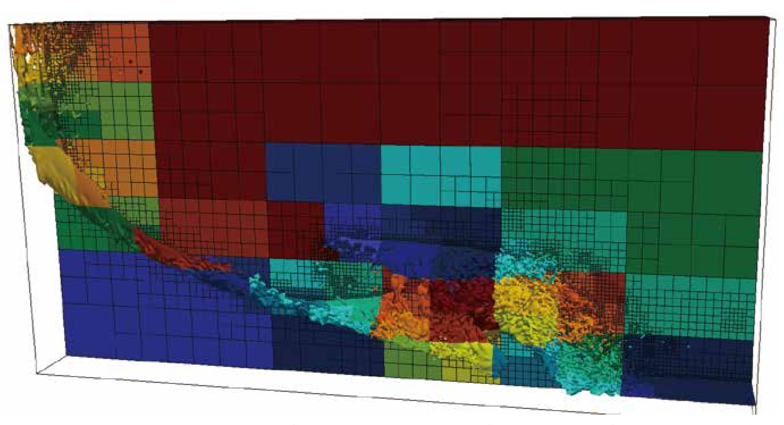
A weakly compressible scheme with interface-adapted AMR method for incompressible gas-liquid two-phase flows without solving the Poisson equation has been developed. The fully explicit time integration is achieved by solving pressure evolution equation derived from compressible Naiver-Stokes equation under the condition of isothermal and low-Mach number. To reduce volume oscillation, the conservative Allen-Cahn equation coupled with Level-set method is solved. The results of benchmarks agree well with those of semi-implicit incompressible solvers and simulated 2D/3D soap bubble forming problem.

Large-scale Granular Simulation



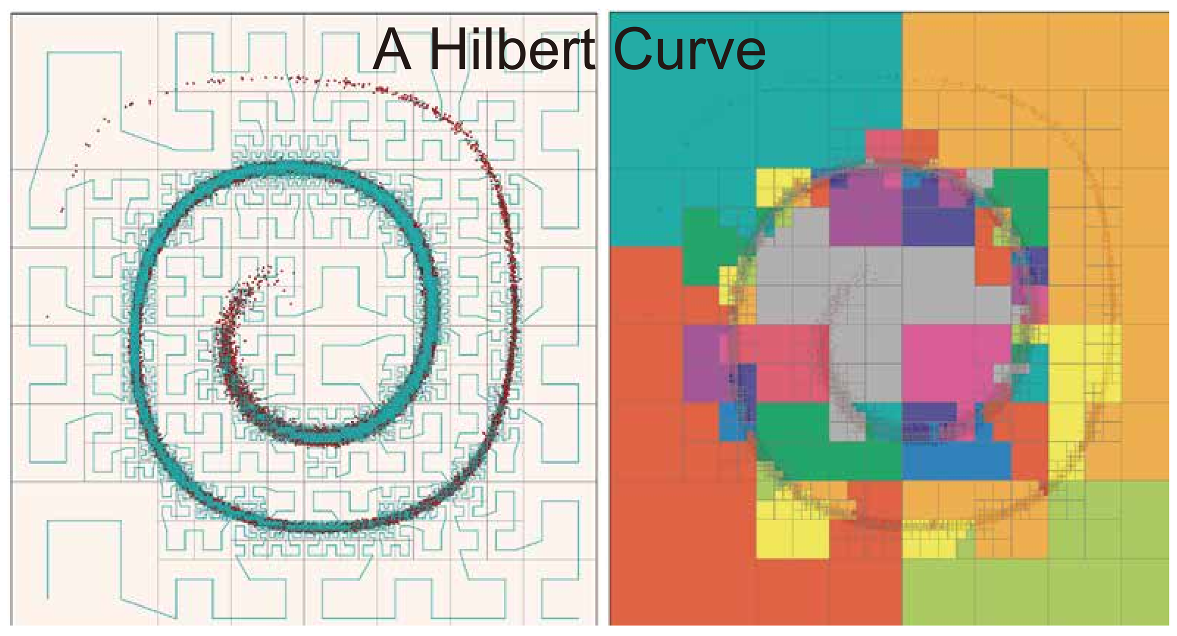
Discrete element method (DEM) is often used to simulate granular dynamics and its simple algorithm with the contact interaction is suitable for GPU computing. However so many particles are included and the particle distributions are changing in time and space. A dynamic domain decomposition has to be introduced for multiple-node computing. In a bunker shot, the sand wedge does not hit to the golf ball directly and transferring the force through the sand to the ball in order to reduce the impact. In this simulation, 16.7 millions of DEM particles are used to represent the dynamics of the sands with 256 GPUs.

Adaptive Mesh Refinement for Multi-phase Flows



Simulations for multi-phase flows require high-resolution grids to capture phenomena at the interface. By using the adaptive mesh refinement (AMR) method, which dynamically adapts high-resolution grids to interfaces, computational cost and memory usage are educed. The spatial distribution of a computational load change in time; therefore, dynamic domain partitioning using a space-filling curve is introduced for multi-GPU computing to assign an equal number of grid points to each GPU. The figures show the large-scale free-surface flow simulations for the dam-breaking process and corresponding domain decomposition for 64 GPUs.

Dynamic Load Balancing using A Space-filling Curve



For large-scale particle-based simulation and Adaptive Mesh Refinement (AMR), it is a critical issue to achieve computational load balance and equal memory usage on multiple compute nodes. A domain partitioning in terms of a space-filling curve(SFC) is one of promising candidates and it is recognized that a 1-dimensional mapping of 3-dimensional space by cutting the equal length. Due to low cost of SFC domain partitioning, it is suitable for frequent re-partitioning in the simulations of unsteady phenomena.