

APPENDIX A

REPRODUCIBILITY DESCRIPTION APPENDIX: LARGE SCALE MPI-PARALLELIZATION OF LBM AND DEM SYSTEMS: ACCELERATING RESEARCH BY USING HPC

A. Abstract

This Appendix describes measurement of parallel speedup of the implementations of the discrete element method (DEM) and lattice-Boltzmann method (LBM) based algorithms. For the DEM, effects of the DEM parallelization schemes are shown. The DEM with particle partitioning will be referred to as the case A), while the DEM with space partitioning will be referred to as case B). Measurement of parallel speedup of the LBM based algorithm for solidification is also presented, referred to as the case C). Randomness of the resulting parallel speedup curves is assessed.

B. Description of the measurement of parallel speedup of the DEM and LBM implementations

Parallel speedup of the DEM algorithm was examined. Two schemes of DEM parallel partitioning are evaluated: spatial (case A) and particle based domain decomposition (case B). Parallel speedup of the LBM algorithm (case C) is also examined.

- **Algorithm:** Cases A) and B) DEM, case C) LBM
- **Compilation:** Case A) Intel 16 Fortran compiler, Intel MPI library, case B) Intel 16 C++ compiler, Intel MPI library, case C) Intel 16 Fortran compiler, Intel MPI and HDF5-1.8.20 library
- **Run-time environment:** SLES
- **Hardware:** Topaz@ERDC
- **Execution:** MPI-parallel
- **Output:** minimal status messages written to standard output
- **Experiment workflow:** multiple submissions to cluster queue
- **Publicly available:** Case C, 2D version

1) *Hardware:* Parallel performance was measured on Topaz supercomputers. Topaz is located at the U.S. Army Engineer Research and Development Center (ERDC). It is an SGI ICE X system with 3,456 standard computing nodes, each with 116 GB of DDR4-2133 RAM and two eighteen-core Intel Xeon E5-2699v3 processors. The nodes are connected by 4x FDR InfiniBand network with Hypercube topology. All login and compute nodes run SLES operating system. Full hardware configuration of Topaz is available at <https://www.erd.c.mil/hardware/index.html>.

2) *Software:* Case A) The present DEM code is written in Fortran 90 programming language. The code was parallelized using MPI library with particle-based domain decomposition. MPI capable parallel Fortran 90 compiler is required to compile and build the code.

Case B) An open source DEM code LIGGGHTS, a fork of LAMMPS, is written in C++ programming language. The code is parallelized using MPI and OpenMP with spatial domain decomposition. Speedup of the MPI-only parallelized version of LIGGGHTS was examined here.

Case C) The MSU LBM solidification code is written in Fortran 90 programming language. The code was MPI-parallelized with spatial domain decomposition imposing equal-size rectangular subdomains. MPI capable parallel Fortran 90 compiler is needed to compile and build the code. LBM

code can be optionally compiled with HDF5 support, taking advantage of parallel MPI output implemented in open source HDF5 library. To include HDF5 support, *h5pfc* compiler wrapper by HDF5 group is used to compile the code. Parallel HDF5 library allows writing a single binary output file from MPI parallelized code. Binary format enables exact restart of LBM simulations. Without any post-processing, the number of MPI processes for restarted simulation can differ from the number of MPI processors used for initial simulation.

3) *Experiment workflow:* For case A), initial 24,313 time steps of the lid-compaction phase of a settled cylindrical 9.5 million particle configuration was used as the benchmark to measure parallel DEM performance.

Portable Batch System (PBS) scripts were submitted to the cluster queue for each of the following number of MPI processes: 1, 3, 9, 18, 27, 36, 72, 144, 288, 576, 1152, 2304. Parallel speedup curves for DEM with increasing number of MPI processes using space and particle partitioning are shown in Figure 1 a). The scaling study used as many MPI processes per node as the number of cores available on a node, which is 36 on Topaz.

Load of the computational cluster interconnect network can vary depending on the number of parallel jobs executed and their inter-node communication demands. Effects of the network load from the traffic due to other jobs running concurrently on the cluster were assessed by performing two executions for each number of MPI processes. Parallel speedup was evaluated relatively to the faster of two executions using a single MPI process, which took 42 hours and 29 minutes CPU time. Execution using 1152 MPI processes on Topaz took 10 minutes and 15 s.

The present DEM particle partitioning technique was then compared with space partitioning as implemented in open source LIGGGHTS DEM package (case B). The benchmark examined settling of 9.5 million particles under gravity over 50000 time steps. The faster of two benchmark executions using a single core of Topaz took 59 hours and 2 minutes. Figure 1 a) shows that particle partitioning performs better with low number of MPI processes, while spatial partitioning is advantageous for large number of MPI processes, where particle partitioning curve exhibits a noticeable decrease in slope.

It is important to note that particles in the chosen benchmarks do not mix much. Mixing would decrease parallel efficiency of both partitioning schemes. An improvement could be achieved by dynamics re-assignment of particles to processors when the proximity of the particles on a processor is lost due to mixing.

A pre-grown configuration of two columnar dendrites with a grid dimensions of $120 \times 120 \times 2560$ served as a starting configuration for LBM speedup benchmark (case C). Twenty time steps of dendrite growth simulation were performed. Parallel performance of LBM solidification model was measured for the following number of MPI processes: 1, 8, 16, 32, 48, 64, 80, 96, 128, 144, 160, 192, 256, 288, 320, 384, 512, 576, 640, 768, 1152, 1536, 2304, 4608. The faster of two executions using a single core of Topaz took 8 hours and 19 minutes. Parallel speedup plot is shown in Figure 1 b).

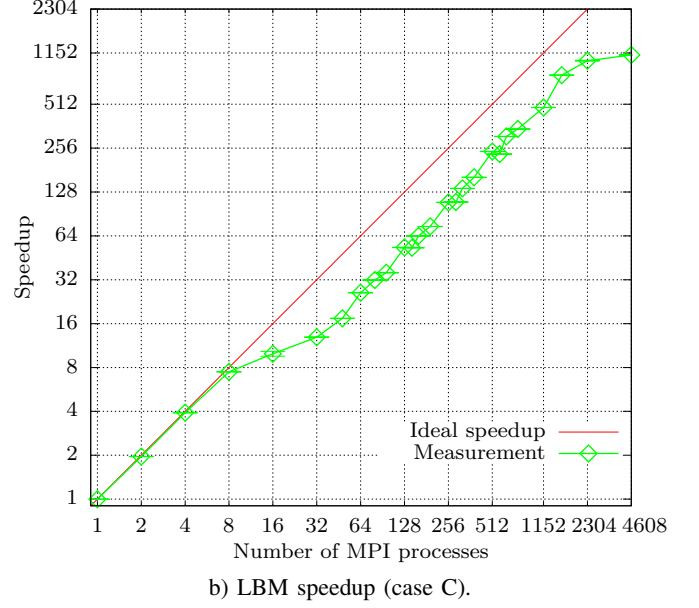
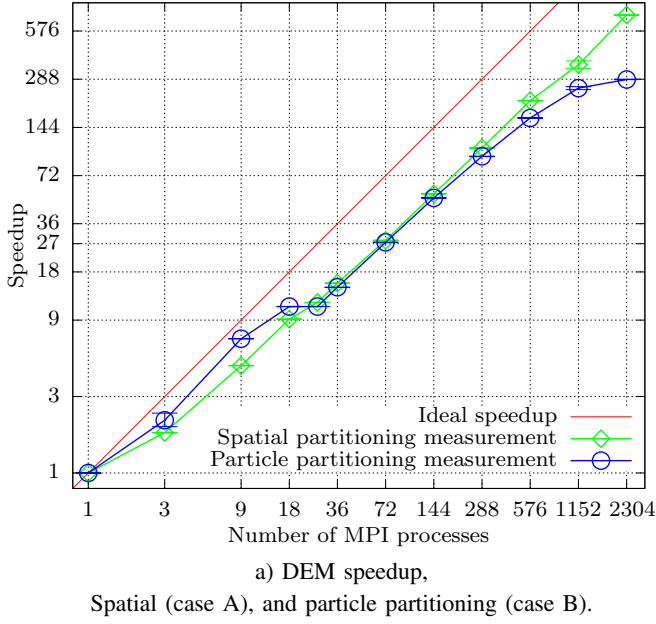


Fig. 1. Speedup plots. Range of two measurements at each number of MPI processes is indicated by horizontal bars.

4) *Evaluation and expected results:* Correctness of the DEM and LBM implementations was validated in earlier publications. Before parallel implementation, the correctness of serial calculation results was checked by two-particle and particle-rigid object interaction tests. The same results were validated in parallel implementation.

Then, the present benchmark cases were implemented. For cases A) and B), correctness of the results was validated by visual inspection of the initial, intermediate, and final particle configurations. In case A), visual inspection of particles confirmed that particles displaced in the direction of gravity. In case B), visual inspection of the particles confirmed that particles were compressed by the external pressure applied on the lid. The vertical stress value was found to correspond to the confining pressure. In case C), the expected dendrite growth was visually confirmed.

Results of parallel executions with varying number of MPI processes were identical within the limits of numerical differences of parallel executions due to varying order of parallel operations.

C. Summary

Each of the presented parallel benchmarks indicates a loss in parallel efficiency when all 36 cores of the nodes of the benchmarking supercomputer are used. Close-to-linear speedup is observed up to 8 MPI processes, followed by a decrease in parallel efficiency up to 36 MPI processes. This performance loss could be addressed by implementing hybrid OpenMP+MPI parallelization. Hybrid parallelization would require adding OpenMP directives to loops and executing a single MPI process per node, each MPI process using up to 36 OpenMP based threads.

No significant random variance of the two DEM executions on each number of MPI processes was observed. Presence of barely visible variability in execution time increases confidence in the measurement of speedup.

As seen in Figure 1 a), DEM with particle partitioning performs better with low number of MPI processes, while spatial partitioning is advantageous for large number of MPI processes-where particle partitioning curve exhibits a noticeable decrease in slope.

No significant random variance of the LBM speedup results was observed. For the 64, 128, 256, and 512 MPI processes, which are powers of 2, the measured speedup was marginally higher than for the remaining number of MPI processes. This can be attributed to MPI implementation taking advantage of the topology of the power of two communicator. LBM partitioning was found to perform best when the computational domain is split along the z -direction, which is expected because z is the largest dimension of the LBM computational domain.