Title

Toward Polydisperse Flows With mfix - exa

Permalink

https://escholarship.org/uc/item/2cm5r06w

Journal

Journal of Fluids Engineering, 146(4)

ISSN

0098-2202

Authors

Lattanzi, Aaron M
Fullmer, William D
Myers, Andrew
et al.

Publication Date

2024-04-01

DOI

10.1115/1.4064533

Peer reviewed

Aaron M. Lattanzi[1]
Lawrence Berkeley National Laboratory,
Berkeley, CA, 94720,USA
email: amlattanzi@lbl.gov

William D. Fullmer
National Energy Technology Laboratory,
Morgantown, WV 26505 USA
email: william.fullmer@netl.doe.gov

Andrew Myers
Lawrence Berkeley National Laboratory,
Berkeley, CA, 94720,USA
email: atmyers@lbl.gov

Jordan Musser
National Energy Technology Laboratory,
Morgantown, WV 26505 USA
email: jordan.musser@netl.doe.gov

Towards polydisperse flows with MFIX-Exa

In the presence of large size disparities, single-grid neighbor search algorithms lead to inflated neighbor lists that significantly degrade the performance of Lagrangian particle solvers. If Eulerian–Lagrangian (EL) frameworks are to remain performant when simulating realistic systems, improved neighbor detection approaches must be adopted. To this end, we consider the application of a multi-grid neighbor search (MGNS) algorithm in the MFIX-Exa software package, an exascale EL solver built upon the AMReX library. Details regarding the implementation and verification of MGNS are provided along with speedup curves for a bidisperse mixing layer. MGNS is shown to yield up to 15× *speedup on CPU and* 6× *speedup on GPU for the problems considered here. The MFIX-Exa software is then validated for a variety of polydisperse flows. Finally, a brief discussion is given for how dynamic MGNS may be completed, with application to spatially varying particle size distributions.*

Keywords: MFIX-Exa, AMReX, bidisperse, polydisperse

[1] Corresponding Author.

1 Introduction

Many natural and industrial systems involve dynamic and heterogeneous particle-laden flows. Consequently, great effort has been devoted to the development of computational tools for the simulation of multi-phase flows. The Eulerian–Lagrangian (EL; commonly referred to as CFD-DEM) method has been established as a valuable intermediate description for particle-laden flows. More specifically, discrete particles are tracked in space and time while the fluid phase is described by continuum equations on a grid that is comparable or larger than the particle diameter—i.e., models are required for the inter-phase heat, mass, and momentum transfer. While EL methods require few closures and have shown good predictive capabilities, they become computationally expensive when one seeks to simulate large scale systems that may involve tens of millions or even billions of particles. Furthermore, many realistic granular solids exhibit significant polydispersity, which is known to dramatically reduce the performance of Lagrangian solvers [1–4]. Therefore, high-fidelity EL codes that can utilize exascale computing resources and efficiently handle polydisperse solids are of great importance.

The MFIX-Exa software (https://mfix.netl.doe.gov) is an exascale enabled EL solver that is built on the AMReX library [5] (https://amrex-codes.github.io/amrex/). Preliminary benchmarking of MFIX-Exa is provided in [6] while comprehensive descriptions of the numerical methods may be found in [7] and [8] for cold and reacting flows, respectively. The interested reader is referred to the aforementioned works for details regarding the governing equations and numerics employed by MFIX-Exa. While significant effort was devoted to the verification and benchmarking of MFIX-Exa, those prior works focused on single-sized (monodisperse) particles and did not consider performance or accuracy with polydisperse solids.

The present study details the implementation of a multi-grid neighbor search (MGNS) algorithm within the AMReX library; which is then utilized by MFIX-Exa to improve computational efficiency when simulating polydisperse solids. The MGNS algorithm described here most similarly parallels that proposed by [4] and supports both CPU and GPU computations through the AMReX backend. In § 2 the single-grid neighbor search (SGNS) and multi-grid neighbor search (MGNS) algorithms are discussed. Verification of MGNS is provided in § 3 followed by validation studies in § 4. Performance metrics with MGNS are provided in § 5 and potential future work to develop a dynamic MGNS, with application to clustering and segregated flows, is considered in § 6.

2 Neighbor Search Algorithms

The present study employs a 'soft-sphere' collision model where multi-bodied and enduring collisions are resolved. To compute the pair-wise collision forces and torques, for each particle a neighbor list is created that contains the number of nearest neighbors and the neighboring particle's local index in memory. As a result, particle-particle interactions are identified efficiently by looping over each particle and only testing for collisions (i.e., an overlap) with particles contained within its neighbor list. § 2.1–2.2 provide overviews for the SGNS and MGNS algorithms, respectively.

2.1 Single-Grid Neighbor Search. When constructing a neighbor list, most EL solvers utilize a single computational grid that is $\sim 3R_p$ to $6R_p$ in length, where R_p is the particle radius. For polydisperse solids, the grid size is constrained by the largest particle radius, $R_{p,\max}$, to avoid missing neighbors/collisions. Once particles are mapped to their ijk cell on the grid, the 27 surrounding cells $(3 \times 3 \times 3)$ are searched to determine which nearby particles are close enough to be considered a neighbor: $||\mathbf{r}_{ij}|| \leq \mathcal{D}$, where $\mathcal{D}$ is a chosen separation distance, often taken to be $3R_{p,\max}$. A cursory pseudo-code outlining the SG approach is given in Algorithm 1. The `DenseBins` function stores the local particle index in the ijk cell it resides, found from mapping particle positions $\mathbf{X}_p$ to the grid by way of the `BinMapper` functor. Subsequent loops over all particles are completed to count each neighbor and populate the neighbor list; the neighbor list population loop is omitted since it employs the same structure as the first loop for counting neighbors.

Using a single grid size and single separation distance $\mathcal{D}$ for polydisperse flows causes the performance to deteriorate as the particle size ratio $\lambda_i = R_{p,\max}/R_{p,i}$ and the solids volume fraction ϕ increase [9,10]. Specifically, if one considers a bidisperse mixture, it becomes immediately clear that the large grid and $\mathcal{D}$, which are constrained by $R_{p,\max}$, lead to many small particles in each cell, thereby artificially inflating the size of the neighbor list.

2.2 Multi-Grid Neighbor Search. In contrast to SGNS, MGNS allows for the use of multiple grids and multiple separation distances. The MGNS approach essentially splits the particle size distribution (PSD) into discrete bins (referred to as particle 'types'). A grid of size h_l is defined for each type and a separation distance is defined for each type interaction $\mathcal{D}_{lm} = (3/2)\left(R_{p,l} + R_{p,m}\right)$, where l and m denote the type index. Particle neighbors of the same type, such as $l - l$, may be found on the l grid in the same manner as the SGNS algorithm. Particle neighbors of a different type, such as $l - m$, must be found by searching on the grid for the other type. For the mixed type $l - m$ case, we map the smaller particles onto the larger grid. This choice is motivated by computational efficiency since large particles on a small grid will need to search more than the 27 surrounding cells and the number of particles in each cell (offsets) may be large, leading to noticeable slow down.

Algorithm 2 provides cursory pseudo-code for the MGNS approach that parallels the SGNS approach in Algorithm 1. One of the key differences between Algorithm 2 and Algorithm 1 is that the `BinMapper` functor accepts a new integer particle type input, from `PartData`, so that it may map the particle to the correct grid. Additionally, a loop from the current particle type to the max particle type is added for the mixed $l - m$ collisions and the separation distance $\mathcal{D}$ is a symmetric matrix of size $N_{\text{type}} \times N_{\text{type}}$.

The MGNS approach requires the number of types and the discrete size associated with each type to be specified. However, the selection of the bin sizes and number of bins impacts the computational efficiency of the MGNS algorithm and are fundamentally tied to the PSD [3,11]. Well established guidelines for dynamically setting bin sizes and bin counts to their ideal values are not readily available in the literature. To this end, it is worth noting that [3,11] provide valuable insight on the formulation of an optimization problem for the bin size and count in their MGNS algorithm and suggest that the particle per cell count on each grid should be equal. However, the present algorithm has notable departures from that given in [3] and evaluation of the analytical expression in [3] requires the particle data to be sorted by size, which is costly for large simulations. In § 6 we provide some basic estimates for the impact of the bin size/count on MGNS and motivate the potential for dynamic MGNS algorithm. However, development of such an algorithm will be the subject of future work.

3 Verification

MFIX-Exa employs a three-level regression suite to test code accuracy during development [7]. Here, the SG algorithm is assumed to be accurate and used as a benchmark to verify that the MG algorithm does not miss collisions. Output from a set of canonical bidisperse simulations discussed in § 2.1 are analyzed for consistency. Namely, the $\{i;\ j;\ \mathbf{F}_{\text{tot}} = \mathbf{F_n} + \mathbf{F_t}\}$ was written for each collision pair identified within an MFIX-Exa particle iteration. The output files were written in a block format where one block contains all the colliding pairs and their forces in one iteration. At each iteration, the block obtained with SG and MG were compared via a simple N^2 search algorithm to ensure that each collision pair in the SG block also resides in the MG block and that the collision was not already identified as a match (no double counting).

It should be noted that the neighbor list itself cannot be directly compared between the two algorithms for two reasons. First, the MGNS approach serves to reduce the neighbor list size by introducing more length scale criteria for determining if a particle is a valid neighbor. Therefore, the neighbor list with MGNS should be smaller than the neighbor list with SGNS and the extent of the reduction should scale with solids volume fraction and λ. Second, the MGNS approach may yield different ordering of particle indices within the neighbor list due to the modified loop structure; this is also why an N^2 search algorithm was necessary for comparing colliding pairs.

4 Validation

Again we rely on previous validation studies for the MFIX-Exa code as a whole, as this work focuses on the implementation of a new neighbor list algorithm. However, it is notable that all of the previous comparisons to experimental data have only considered monodisperse particles. Therefore, we add to MFIX-Exa's growing validation set by including two cases with bi- and tridisperse mixtures. It is assumed that statistical errors due to the particles are the largest source of numerical error in this work. Each simulation reported is repeated multiple times with slight differences, e.g., different random initial particle configuration, and the spread of the realizations is measured with an error bar signifying a 95% confidence interval from a *t*-test.

4.1 Bidisperse Fluidized Bed. It is natural to begin with the bench-scale fluidized bed of Goldschmidt et al. [12]. This fluidized bed is the most commonly referenced experiment for bidisperse mixtures because it was one of the first experiments set up specifically to generate validation data for the CFD-DEM numerical method. The two particle types are 2.5 mm and 1.5 mm glass beads. The properties of the bed and material (as modeled) are listed in Table 1. A few adjustments and simplifications have been made. First, the actual height of the test section was 70 cm is extended here to be an integer number of the bed width. The measured density of the two particles differed slightly in mean value. However, because they were very close with overlapping error bars, a single value is used here within the measurement error of both particles. The measured sphericity was reported to be approximately one and is therefore not considered here.

Although several conditions are considered in the original work of Goldschmidt [12], we focus on two here: (small particle) solids mass fractions of $x_1 = 0.25$ and $x_1 = 0.75$ with an inlet superficial gas velocity of $U_g = 1.15$ m/s. These conditions are of interest because very different results are obtained by simply changing the ratio of small-to-large particles. The particles are initialized in a statistically homogeneous (randomly distributed) array which is facilitated by a low particle volume fraction, approximately 20%. The number of large and small particles for the $x_1 = 0.25$ condition are: $N_2 = 28000$ and $N_2 = 18000$ and for the $x_1 = 0.75$ condition are: $N_2 = 6000$ and $N_1 = 83000$. The particles are allowed to settle for a $t \approx 2$ s initialization period into a packed bed. The inflow velocity is set to 0.8 which is insufficient to fluidize either particle or the mixtures. From the period $t = 1.8$ - 2.0 s the inlet velocity is abruptly and linearly increased to the desired value of $U = 1.15$ m/s. The spring constant, $k_n = 2290$ N/m is set to give large-large, large-small and small-small particle collision durations of $\tau_{coll} = 0.21$, 0.13 and 0.10 ms, respectively. The smallest of the three is used to set the solids timestep, $dt_s = \min(\tau_{coll})/20 = 5 \times 10^{-6} s$. The gas-phase is treated as an incompressible fluid with the properties of air at STP: $\rho_g = 1.2$ kg/m^3 and $\mu_g = 1.8 \times 10^{-5}$ kg/m-s. The fluid timestep is adjusted automatically by a Courant limit of 0.9. The Gidaspow [13] drag model is used to couple the phases.

Each condition is simulated for 60 s after the 2 s initialization period, which is neglected from analysis, and repeated five times. Each replicate uses a different seed in the random number generator used to generate the initial particle arrays, thereby creating a statistically unique condition. The quantity of interest is a segregation index, s, based on the average height of the two particles which could be measured by digital image analysis techniques. Following [12], the segregation index is defined by:

$$s = \frac{S - 1}{S\text{max} - 1}\ , \tag{1}$$

where $S = \overline{y_1}/\overline{y_2}$ is the ratio of the mean elevation of the small-to-large particles and

$$S\text{max} = \frac{2 - x_1}{1 - x_1}\ , \tag{2}$$

is the theoretical maximum degree of segregation. The simulation results are compared with the (digitized) experimental data in Fig. 1. The expected results are observed: with the small particle mass fraction is low, $x_1 = 0.25$, strong segregation is observed, but when the small particle mass fraction is high, $x_1 = 0.75$, essentially no segregation is observed. This is because the operating condition is set between the minimum fluidization of the two particle types; the measured minimum fluidization velocities were $U_{mf} = 0.80 \pm 0.02$ m/s and $U_{mf} = 1.25 \pm\ 0.01$ m/s for the small and large particles, respectively. Therefore, when there are fewer small particles, they tend to collect at the top of the bed and form a fluidized layer. On the other hand, the large particles collect at the bottom and form an under-fluidized or defluidized layer. However, when the large particles are relatively few, as in $x_1 = 0.75$, the small particles remain fluidized, transporting the large particles with them. This is an important behavior in the operation of industrial fluidized beds with difficult to fluidize material such as biomass [14]. It is a little surprising that the comparison is so good without adjusting or calibrating the drag model in any way, i.e., these are predictive results although the expected outcome was known to the authors. Finally, we note that the MGNS algorithm did not provide a speed up in this case due to the low particle size ratio, $\lambda < 2$.

4.2 Tridisperse Chute Flow. The second verification problem also looks at a segregation mechanism. Now, instead of being fluidized by a flowing gas-phase, the source of agitation is provided by gravity driven flow, down a rough inclined chute. The bench-scale experiment of Bhattacharya and McCarthy [15] is used to validate the results of granular (no gas phase) MFIX-Exa simulations. The experiment is enclosed at in a box with a length of $L_x = 1.2$ m, a depth of $L_x = 0.15$ m and modeled with a height of $L_y = 1.6$ m.

Twenty equally spaced bins line the bottom of the test section to collect particles. The bin height is equal to the bin length of 60 mm. A 0.6 m long ramp is hinged at the back of the test section. The ramp can be set at two different inclinations, $\theta = 30$ and 45 degrees from the horizontal. In the experiment, the elevation of the ramp, measuring from the floor to the hinge, can be set at three different levels. In this case, we are interested in comparing the tridisperse mixture at both ramp inclinations, for which the ramp height is 1.05 m. Sitting 0.1 m above the ramp (hinge) is the hopper holding the particle mixture. In the experiment the hopper contains a converging section and a larger upper cross-section. For simplicity, the model considers the hopper to have uniform cross-section equal to the discharge region of the experiment, which is a square 0.15×0.15 m^2 opening.

The particles are polystyrene beads with diameters of $d_3 = 14$, $d_2 = 7$ and $d_1 = 6$ mm and densities of $\rho_3 = 1170$, $\rho_2 = 1148$ and $\rho_1 = 952$ kg/m^3, for the large, medium, and small particles, respectively. The ramp, the collection bin dividers, and the hopper gate are modeled with "wall" particles having diameter and density $d_w = 5$ mm and $\rho_w = 1000$ kg/m^3. The remainder of the walls, e.g., domain boundaries and the hopper upright, is modeled as an embedded boundary (EB). All particle-particle (including wall particles) and particle-wall collisions are given the same contact properties: $e_{pp} = e_{pw} = 0.7$ and $\mu_{pp} = \mu_{pw} = 0.4$. The spring constant is set to $k_n = 35$ kN/m. The largest collision duration is between two large particles, $\tau_{3,3} \approx 5 \times 10^{-4}$ s. The shortest actual collision duration is between a small and wall particle, $\tau_{1,w} \approx 1.1 \times 10^{-4}$ s. However, the timestep calculation routine does not know that the wall particles will not collide with one another and $\tau_{w,w} \approx 9.7 \times 10^{-5}$ s is used to set the solids timestep, $dt_s = \tau_{w,w}/20$.

The tridisperse mixture is generated randomly and uniformly in the hopper section and allowed to settle for $t_0 = 1$ s. After the settling period, the wall particles defining the hopper gate are given a velocity of $u_w = -0.3$ m/s. This allows the hopper to fully open in 0.5 s. All other wall particles have velocity and angular momentum of zero. As the simulation progresses, the gate begins to open, particles fall from the hopper, onto the ramp, slide down and fall from the ramp, and collect in the bins at the bottom of the test section. The top row of Fig. 2 depicts this evolution for the $\theta = 30°$ inclination angle. Each of the two ramp angles is repeated ten times, each with a different randomized particle configuration in the hopper. The mass fraction of each particle size in each of the bins is collected and averaged to produce error bars (t-test 95%-confidence interval). The simulation results are compared to the experimental data in Fig. 2. For the lower angle of 30°, the simulation predicts that all particles travel slightly farther than observed experimentally. The higher angle of 45° shows excellent agreement.

Although the simulation results compare favorably to the experimental data, in neither instance are we able to see substantial evidence of segregation. To explore this further, we move away from the experimental geometry into a periodic configuration so that the material can flow continuously. The wall-particle defined chute is the same (length and width) and the depth walls are still defined by EBs. For simplicity in this case, the chute is laid flat and the gravity vector is rotated. The same tridisperse mixture is considered as shown in Fig. 2 with double the number of each particle type to fully cover the ramp in several layers of material. Inclination angles from $\theta = 21°$ to 35° are studied. Exploratory studies considered a wider range, however, this was determined to be the primary region of interest because a) below 21° the material does not readily flow down the chute, i.e., we have reached the angle of repose of the mixture and b) as $\theta \rightarrow 35°$ the mixture stops segregating. To quantify this later behavior, we consider a simple metric: $\bar{y}_3/\bar{y}_2$ and $\bar{y}_2/\bar{y}_1$, where $\bar{y}_i$ is the mean height of each particle type. The two ratios are provided in Fig. 3 for the inclination angles under consideration. Each case is simulated for 20 s which is sufficient to reach a fully developed mean elevations, although the mean streamwise velocity is still slightly increasing in some cases. The mean particle elevation ratios are taken at the end time, $t = 20$ s. Each case is repeated five times to compute error bars, again arising from unique initial particle configurations. Figure 3 shows that the segregation strength, higher $\bar{y}_{i+1}/\bar{y}_i$ values, is strongest at the lowest inclination angles. Beyond approximately 30°, both mean height ratios quickly decay to unity, indicating a well mixed state.

In interpreting these results, two points of further consideration are given. First, although the strongest segregation behavior occurs at 21°, this condition is very near the critical inclination angle for flow. Additionally, the mean particle velocity is quite low. Therefore, when considering an application where segregation is the desired result, practical considerations such as time or physical size limitations may result in a larger optimal angle of inclination. Second, in applications where uniformity is desired, Fig. 3 may be misleading at higher angles. Although the end state is well mixed, all cases up to and including $\theta = 35°$ went through a transient state where the large particles noticeably segregated. It is assumed that the initial segregation and then re-homogenization is a consequence of the acceleration of the mixture down the chute. At low speeds, the particles are able to segregate. However, as the mean velocity and granular temperature (agitation) continues to increase, the particles relax back into a well-mixed state. Therefore, a segregated mixture may be delivered even at high inclination angles if the chute length is too short.

Unlike the fluidized bed in § 4.1, the size ratio of the tridisperse material is at least two, $\lambda \geq 2$. Therefore, the time-to-solution of both the experimental setup and infinite chute was improved with the MGNS algorithm. Two bins with a refinement ratio of two were applied in all cases. To quantify the speedup, the first replicate of the infinite chute flow was repeated with the SGNS algorithm for all inclination angles. The speedup is given by the ratio of the average time-per-step of the SGNS simulation relative to the MGNS simulation. Figure 3 shows a speedup of approximately two for most cases. The performance is slightly better at lower inclination angles, which is likely a consequence of the denser packing of the granular layer. A more detailed assessment of the MGNS performance is the topic of the following section § 5.

5 Performance

To demonstrate the impact of polydispersity on the SGNS algorithm and test the speedup of the MGNS algorithm, we consider a granular, triply-periodic domain with edge length $L = 0.0128$ m. Large particles are seeded in a top and bottom layer while small particles are placed in the middle layer. The size of the small particles are varied to give specific $\lambda = R_{p,\max}/R_{p,\min}$ while large particles are held fixed at $R_p, \max = 200$ µm. In each simulation, the solids volume fractions are adjusted to reach specific particle counts, but are generally in the range of $\phi \in [0.2 \;\; 0.5]$. All particles are initialized in a hexagonal close-packed (HCP) lattice. The bidisperse solids mixture is given an initial granular temperature of $T_0 = 0.25$ m^2/s^2. The particles disperse through scattering into a random configuration from the HCP structure as the simulations evolve in time. The initial and final states for an example configuration in Fig. 4.

The simulations are carried out on OLCF's Summit, first considering the CPU (IBM POWER9 processors). As the size ratio λ increases, time-per-step begins to increase as shown in Fig. 5. Slowdown is computed in relation to a monodisperse simulation with the same particle count, where an average time per iteration is employed since the solids time step is a function of the particle size. The total time taken by the solids integrator is primarily associated with (~ 80%) evolving the particle states (computing forces/torques and particle update) and neighbor list construction (completed every 25 solids time steps in the present study). The ratio of time spent by the particle evolution (Evolve) and neighbor list (NL) routines remains relatively constant and of $\mathcal{O}(1)$ for all the λ considered here; see Fig. 6. Since the overall time spent by the solids solver increases with λ, from Fig. 5, the time spent evolving particles and making the neighbor list must be growing at similar rates. Since the size of the neighbor list strongly impacts the computational overhead of the force/torque computations in evolve, these results are consistent with an inflation of the neighbor list.

Now the canonical problem is simulated with the new MGNS algorithm. The performance (speedup) is assessed as the ratio of SGNS timing results to MGNS timing results. Figure. 7 shows that the the MGNS algorithm is capable of producing dramatic speedup on the CPU. As expected, the speedup with MGNS is greatest with large size ratios λ and large particle counts, since the reduction in the neighbor list is greatest for dense flows. Additionally, simulations were also carried out on a GPU (Nvidia Tesla V100) up to $N_p = 2 \times 10^5$ particles. The GPU results are provided in Fig. 8. The GPU results highlight a potential pitfall of the MGNS algorithm: the neighbor list is not significantly compressed if the size discrepancy between two types is not large enough. In other words, the additional overhead of the MGNS algorithm causes a performance degradation for $\lambda < 4$ and increasing with decreasing particle count.

6 Future Work: Dynamic Multi-Grid

In § 5, impressive speedup curves were generated from simulations of scattering in a bidisperse mixture with an initial granular temperature T_0. It should be noted though that the selection of bin counts and sizes is trivial for the simple bidisperse case considered. However in true polydisperse conditions, the degree of speedup with the MGNS algorithm is inherently a function of the PSD, the number of types, and sizes of the particle type bins. Therefore, the observed trends in Figs. 7–8 are not guaranteed. In fact, the slowdown observed in Fig. 8 for $\lambda < 4$ may be present and possibly enhanced for large λ if the nature of the PSD prevents the neighbor list size from being appreciably compressed.

Robust application of the MGNS algorithm to a general polydisperse flow, which involves local clustering and/or segregation, will likely require dynamic use of multiple grids when constructing the neighbor list; thereby making the number of particle types and the bins sizes locally computed quantities that are not known at run time. Under such a paradigm, dynamic MGNS becomes analogous to traditional adaptive mesh refinement (AMR). However, unlike traditional AMR, straightforward indicators for MGNS mesh refinement are not readily available. Additionally, efficient means for optimizing the number of types and bin locations, which would be time and processor/GPU specific, are not readily available.

While beyond the scope of the present work, it is worth noting that algorithms based upon linked-lists may alternatively be considered [16]. Such approaches reduce cell size sensitivity and may improve performance when adaptive meshing is employed.

As a first effort towards defining simple indicators for dynamic MGNS, we estimate the ratio of neighbor counts obtained with SGNS and MGNS in the bidisperse mixture (see Appendix A for more details)

$$\frac{N_{SG}}{N_{MG}} = \frac{\left(N_{p,1} + N_{p,2}\right)\left(\phi_1 + \phi_2 \lambda_2^3\right)}{\left(N_{p,1}\phi_1 + N_{p,2}\phi_2 + 2N_{p,2}\phi_1 \left(\frac{\lambda_2+1}{2\lambda_2}\right)^3\right)}. \tag{3}$$

For large N_{SG}/N_{MG} the MG algorithm is expected to provide appreciable speedup through the compression of the neighbor list while for small N_{SG}/N_{MG} the MG algorithm is expected to introduce uncompensated overhead. Recasting Figs. 7–8 in terms of Eq. (3), and adding additional GPU simulations (black diamonds), shows the expected trend; see Figs. 9–10. If we place a threshold of 2× speedup on CPU and 1.5× speedup on GPU, one would require $N_{SG}/N_{MG} > 4$ on CPU and $N_{SG}/N_{MG} > 10$ on GPU. While Eq. (3) provides a crude estimate, we may approximate the chute flow in § 4.2 as a bidisperse mixture and apply Eq. (3) in conjunction with Fig. 9. By approximating the solids volume fractions, $\phi_1 \approx \phi_2 \approx 0.5$, and summing together the two smaller particle counts, $N_{p,2} = 15791$, we have $\lambda_2 = 2$, $N_{p,1} = 237$, to obtain $N_{SG}/N_{MG} \approx 5$, which is consistent with a speedup of 2 in Fig. 9

Despite the simplifications made in deriving Eq. (3), it does provide a litmus test for dynamic MGNS that depends on low order statistics $N_{p,l}$, λ, ϕ_l and thus is efficient to evaluate. However, truly dynamic MGNS will inherently involve an optimization problem like that presented in [3] for the bin counts and sizes, given a local PSD. To this end, a generalized form of Eq. (3) may be derived and used as a criteria to solve an optimization problem like in [3] for redefining the bins. An algorithm along these lines will be the subject of future work.

7 Conclusions

The MFIX-Exa software targets high-fidelity EL simulations of pilot- and industrial-scale unit operations. Most realistic particle-laden flows exhibit significant polydispersity and broad particle size distributions. However, prior studies with MFIX-Exa have been confined to monodisperse solids. We consider here the performance and accuracy of the MFIX-Exa code for a variety of polydisperse flows.

It is well established that single-grid neighbor search (SGNS) algorithms significantly degrade the performance of Lagrangian solvers as the particle size ratio becomes large, due to artificial inflation of the neighbor list. To alleviate this computational bottleneck, we have implemented a multi-grid neighbor search (MGNS) algorithm into the AMReX library. The application code MFIX-Exa, specifically its EL model, was used to study the new MGNS algorithm. However, it should be noted that any previous or newly developed particle-based physics codes utilizing the AMReX framework have access to this new capability.

In this work, the MGNS algorithm was verified by bit-wise comparisons of collisions, forces and torques against the SGNS algorithm. Although MFIX-Exa has been previously validated on a number of problems [6,7,17], none of them had previously considered bi-, tri-, or polydisperse solids. Therefore, two benchmark cases were provided to extend the validation database of the code. The classic bidisperse fluidized bed of Goldshcmidt et al. [12] showed excellent agreement between simulation and experiment. Secondly, the granular chute flow experiment of Bhattacharya and McCarthy [15] was modeled, specifically the tridisperse mixture. The simulation results compared reasonably with the experiment and the problem was extended into an infinite-chute flow regime using periodic boundaries to investigate the relationship between the inclination angle and the segregation behavior. The performance of the MGNS algorithm was assessed on CPU and GPU for a bidisperse mixing layer. It is found that MGNS can provide significant speedup (up to 15× on CPU and 6× on GPU) for the cases considered here. Best performance is observed for large size ratios, λ, and large workloads, i.e., high N_p-count per processor.

Finally, future work to conceptualize dynamic MGNS is outlined, with application to clustered and segregated flows where the particle size distribution varies rapidly in space and time. Dynamic MGNS would require local specification of the number of grids and size of the grids for constructing the neighbor list. To this end, a neighbor count ratio is defined as a quick litmus test for the benefit of an additional grid. However, future work will be required to incorporate a generalized neighbor count ratio in conjunction with an optimization of local bin sizes and counts.

Appendix A: Neighbor Count Estimates

Consider a bidisperse mixture characterized by $R_{p,1} = R_{p,\max}$ and $R_{p,2} < R_{p,1}$ with corresponding solids volume fractions ϕ_1 and ϕ_2.

The SG algorithm utilizes $\mathcal{D} = 3R_{p,\max}$ as a cutoff distance for determining if a particle is a valid neighbor. Therefore, a characteristic volume may be defined in terms of the max particle volume $V_{p,\max} = 4/3\pi R_{p,\max}^3$ as

$$V_n = \frac{4}{3}\pi \left(3R_{p,\max}\right)^3 = 27V_{p,\max}. \tag{1}$$

Assuming spatial homogeneity, the number of neighbors of type l ($N_{n,l}$) and total number of neighbors for a particle ($N_{n,\text{tot}}$) within V_n are given by

$$N_{n,l} = \frac{V_n}{V_{p,l}}\phi_i = 27\phi_l\lambda_l^3, \tag{2}$$

$$N_{n,\text{tot}} = 27\left(\phi_1 + \phi_2\lambda_2^3\right), \tag{3}$$

where $\lambda_l = R_{p,\max}/R_{p,l}$. The overall neighbor count with SG is then estimated as

$$N_{SG} \approx \frac{1}{2}N_{p,\text{tot}}N_{n,\text{tot}} = \frac{27}{2}\left(N_{p,1} + N_{p,2}\right)\left(\phi_1 + \phi_2\lambda_2^3\right), \tag{4}$$

where the factor of $1/2$ corrects for double counting of neighbors.

If the bin locations with MGNS coincide with the two particle sizes, one will obtain $\mathcal{D}_{lm} = 3/2\left(R_{p,l} + R_{p,m}\right)$ as a cutoff distance for an $l - m$ pair. For M different types of particles, one has $\sum_{i=1}^{M} i = M(M+1)/2$ pairs. The bidisperse mixture in question has $M = 2$ and 3 collision pairs $\{1-1,\ 1-2,\ 2-2\}$. We may then define three characteristic volumes as

$$V_{n,lm} = 27V_{p,\max}F_{lm}^3, \tag{5}$$

$$F_{lm} = 1,\quad \frac{\lambda_2 + 1}{2\lambda_2},\quad \frac{1}{\lambda_2}. \tag{6}$$

The number of neighbors for each collision type within $V_{n,lm}$ is then given by

$$N_{n,11} = \frac{V_{n,11}}{V_{p,1}}\phi_1 = 27\phi_1, \tag{7}$$

$$N_{n,22} = \frac{V_{n,22}}{V_{p,2}}\phi_2 = 27\phi_2, \tag{8}$$

$$N_{n,21} = \frac{V_{n,21}}{V_{p,1}}\phi_1 = 27\phi_1\left(\frac{\lambda_2 + 1}{2\lambda_2}\right)^3. \tag{9}$$

The overall neighbor count with MGNS is then estimated as

$$N_{MG} = \frac{27}{2}\left(N_{p,1}\phi_1 + N_{p,2}\phi_2 + 2N_{p,2}\phi_1\left(\frac{\lambda_2 + 1}{2\lambda_2}\right)^3\right), \tag{10}$$

where the 2 on the third term on the RHS corrects for the fact that double counting has not occurred for the mixed $1-2$ type collisions.

Finally, the ratio of the neighbor counts with SG and MG is given by

$$\frac{N_{SG}}{N_{MG}} = \frac{\left(N_{p,1} + N_{p,2}\right)\left(\phi_1 + \phi_2\lambda_2^3\right)}{\left(N_{p,1}\phi_1 + N_{p,2}\phi_2 + 2N_{p,2}\phi_1\left(\frac{\lambda_2+1}{2\lambda_2}\right)^3\right)}. \tag{11}$$

Acknowledgment

This research was supported by the Exascale Computing Project (17-SC-20-SC), a collaborative effort of the U.S. Department of Energy Office of Science and the National Nuclear Security Administration. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725.

Funding Data

- U.S. Department of Energy, Office of Science, Award No. 17-SC-20-SC

References

[1] He, K., Dong, S., and Zhou, Z., 2007, “Multigrid contact detection method,” Physical Review E, **75**, p. 036710.

[2] in ’t Veld, P. J., Plimpton, S. J., and Grest, G. S., 2008, “Accurate and efficient methods for modeling colloidal mixtures in an explicit solvent using molecular dynamics,” Computer Physics Communications, **179**, pp. 320–329.

[3] Ogarko, V. and Luding, S., 2012, “A fast multilevel algorithm for contact detection of arbitrarily polydisperse objects,” Computer Physics Communications, **183**, pp. 931–936.

[4] Shire, T., Hanley, K. J., and Stratford, K., 2021, “DEM simulations of polydisperse media: efficient contact detection applied to investigate the quasi-static limit,” Computational Particle Mechanics, **8**, pp. 653–663.

[5] Zhang, W., Almgren, A., Beckner, V., Bell, J., Blaschke, J., Chan, C., Day, M., Friesen, B., Gott, K., Graves, D., Katz, M. P., Myers, A., Nguyen, T., Nonaka, A., Rosso, M., Williams, S., and Zingale, M., 2019, “AMReX: a framework for block-structured adaptive mesh refinement,” Journal of Open Source Software, **4**(37), p. 1370.

[6] Fullmer, W. D., Almgren, A. S., Rosso, M., Blaschke, J., and Musser, J., 2019, “Benchmarking of a preliminary MFiX-Exa code,” doi: 10.48550/arXiv.1909.02067, arXiv:1909.02067 [physics].

[7] Musser, J., Almgren, A. S., Fullmer, W. D., Antepara, O., Bell, J. B., Blaschke, J., Gott, K., Myers, A., Porcu, R., Rangarajan, D., Rosso, M., Zhang, W., and Syamlal, M., 2022, “MFIX-Exa: A path toward exascale CFD-DEM simulations,” The International Journal of High Performance Computing Applications, **36**(1), pp. 40–58.

[8] Porcu, R., Musser, J., Almgren, A. S., Bell, J. B., Fullmer, W. D., and Rangarajan, D., 2023, “MFIX-Exa: CFD-DEM simulations of thermodynamics and chemical reactions in multiphase flows,” Chemical Engineering Science, **273**, p. 118614.

[9] Srivastava, I., Roberts, S. A., Clemmer, J. T., Silbert, L. E., Lechman, J. B., and Grest, G. S., 2021, “Jamming of bidisperse frictional spheres,” Physical Review Research, **3**, p. L032042.

[10] Monti, J. M., Clemmer, J. T., Srivastava, I., Silbert, L. E., Grest, G. S., and Lechman, J. B., 2022, “Large-scale frictionless jamming with power-law particle size distributions,” Physical Review E, **106**, p. 034901.

[11] Krijgsman, D., Ogarko, V., and Luding, S., 2014, “Optimal parameters for a hierarchical grid data structure for contact detection in arbitrarily polydisperse particle systems,” Computational Particle Mechanics, **1**, pp. 357–372.

[12] Goldschmidt, M., Link, J., Mellema, S., and Kuipers, J., 2003, “Digital image analysis measurements of bed expansion and segregation dynamics in dense gas-fluidised beds,” Powder Technology, **138**(2-3), pp. 135–159.

[13] Lathouwers, D. and Bellan, J., 2001, “Modeling of dense gas–solid reactive mixtures applied to biomass pyrolysis in a fluidized bed,” International Journal of Multiphase Flow, **27**(12), pp. 2155–2187.

[14] Deza, M., Franka, N. P., Heindel, T. J., and Battaglia, F., 2009, “CFD modeling and X-ray imaging of biomass in a fluidized bed,” Journal of fluids engineering, **131**(11), p. 111303.

[15] Bhattacharya, T. and McCarthy, J., 2014, “Chute flow as a means of segregation characterization,” Powder technology, **256**, pp. 126–139.

[16] Chen, H., Lei, Z., and Zang, M., 2014, “LC-Grid: a linear global contact search algorithm for finite element analysis,” Computational Mechanics, **54**(5), pp. 1285–1301.

[17] Porcu, R., Musser, J., Almgren, A. S., Bell, J. B., Fullmer, W. D., and Rangarajan, D., 2023, “MFIX-Exa: CFD-DEM simulations of thermodynamics and chemical reactions in multiphase flows,” Chemical Engineering Science, p. 118614.

Algorithm 1 : SGbuild()

```
DenseBins($N_p$, *PartData, $N_{\text{bin}}$, BinMapper)
for n $\leq N_p$ do
    ip / jp / kp ← BinMapper(n)
    for i=ip-1; i $\leq$ ip+1; ++i do
        for j=jp-1; j $\leq$ jp+1; ++j do
            for k=kp-1; k $\leq$ kp+1; ++k do
                for p=offset[ijk]; p $\leq$ offset[ijk+1]; ++p do
                    if $||r_{ij}|| \leq \mathcal{D}$ then
                        count += 1
                    end if
                end for
            end for
        end for
    end for
    NbrCnt[i] = count
end for

NbrList ← resize from NbrCnt

REPEAT & POPULATE NbrList
```

Algorithm 2 : MGbuild()

```
DenseBins($N_p$, *particle, $N_{bin}$, BinMapper)
for n $\leq N_p$ do
    for type=type_i; type<n_type; ++type do
        ip / jp / kp ← BinMapper(n,type)
        for i=ip-1; i $\leq$ ip+1; ++i do
            for j=jp-1; j $\leq$ jp+1; ++j do
                for k=kp-1; k $\leq$ kp+1; ++k do
                    for p=offset[ijk]; p $\leq$ offset[ijk+1]; ++p do
                        if $||r_{ij}|| \leq \mathcal{D}$[type_i][type_j] then
                            count += 1
                        end if
                    end for
                end for
            end for
        end for
    end for
    NbrCnt[i] = count
end for

NbrList ← resize from NbrCnt

REPEAT & POPULATE NbrList
```

Table 1 Properties of the Goldschmidt fluidized bed.

property	value	units
bed width, L_x	15	cm
bed height, L_y	75	cm
bed depth, L_z	15	mm
inlet velocity, U_g	1.15	m/s
(large) particle diameter, d_2	2.49	mm
(small) particle diameter, d_1	1.52	mm
particle density, ρ_p	2525	kg/m^3
restitution coefficients, e	0.97	-
wall friction coefficient, μ_{pw}	0.10	-
particle friction coefficient, μ_{pp}	0.15	-
normal spring constant, k_n	2290	N/m

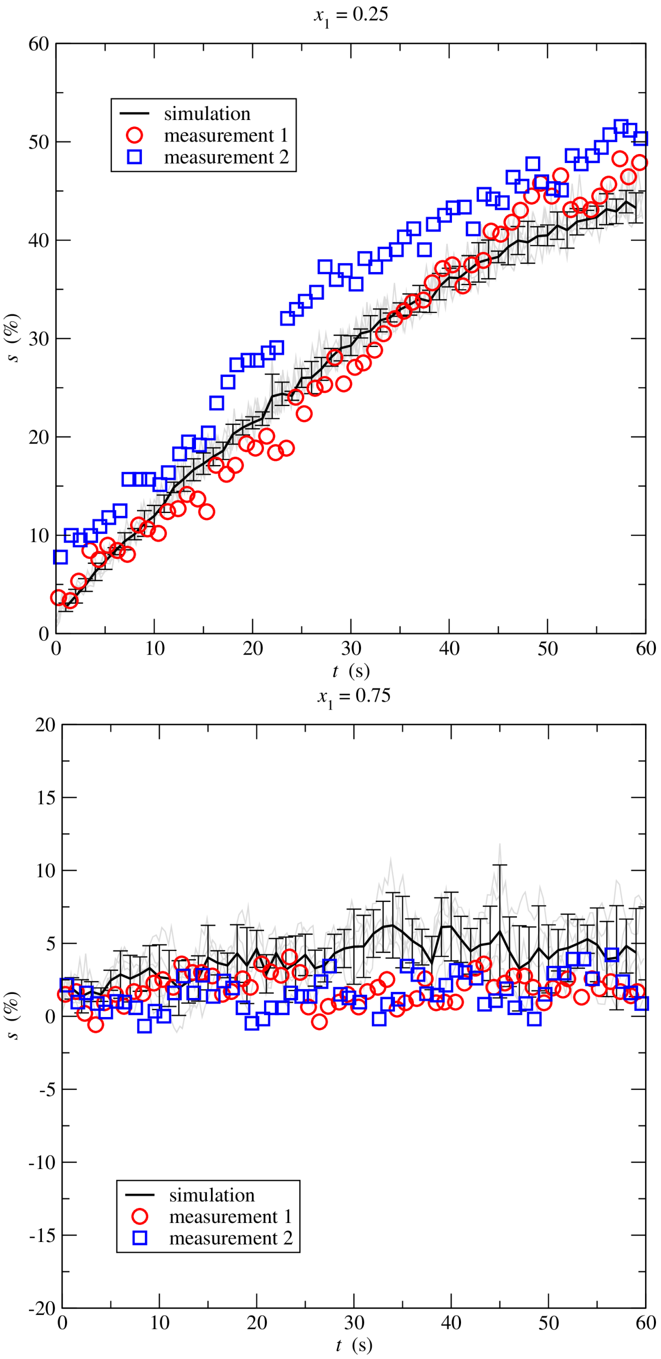

Fig. 1 MFIX-Exa simulation results compared to the original data of [12]. The experimental data has two measurements given by blue squares and red circles. The EL simulations were run five times, given by the grey curves, which are averaged to give mean and error bars (t-test 95%-confidence interval) at 1 s intervals.

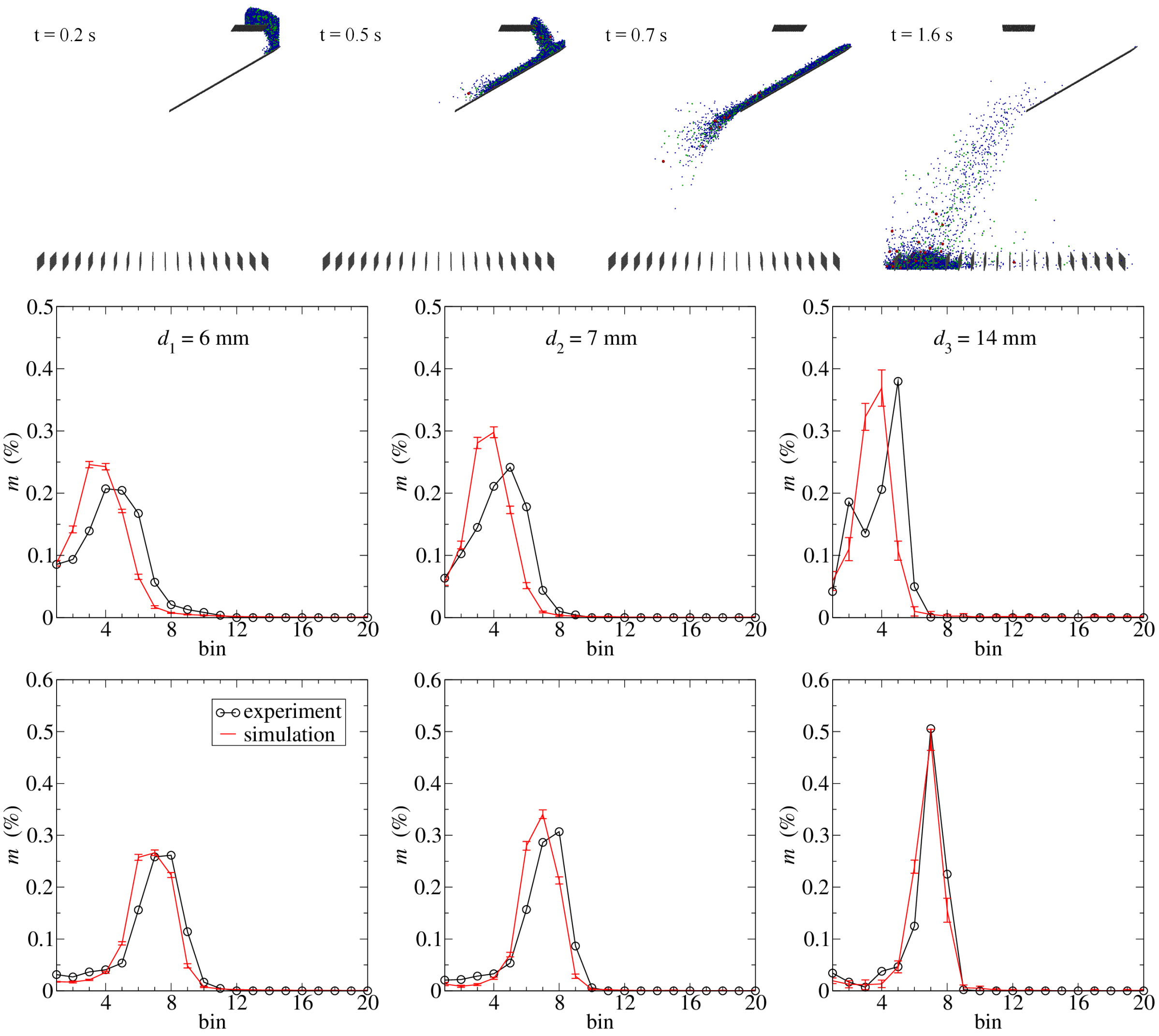

Fig. 2 Granular flow of a tridisperse mixture down an inclined chute after [15]. Top row depicts the hopper discharge and chute flow of the $\theta = 30°$ case. Middle and bottom rows show the mass fraction of each particle type for Inclination angles of $\theta = 30°$ and 45°, respectively. Simulation results (red) for the small, medium and large particles (left to right) are compared to the (digitized) experimental results (black).

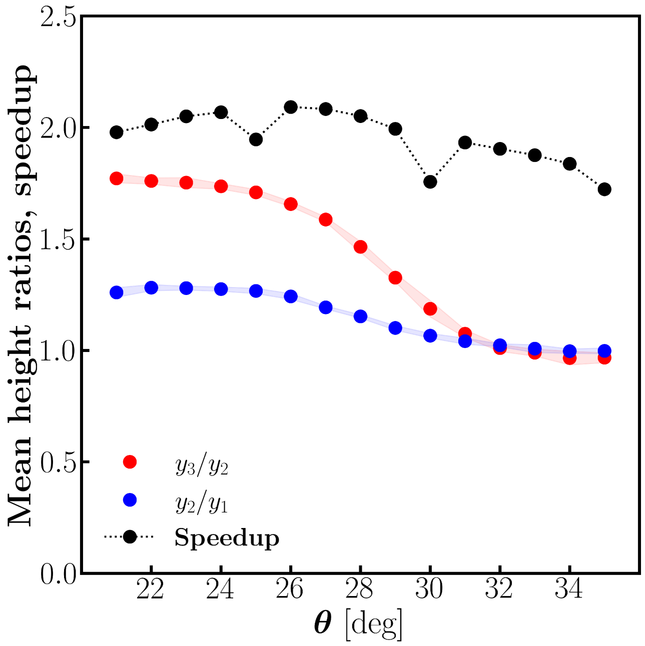

Fig. 3 Segregation of the Bhattacharya and McCarthy [15] material down an infinite (periodic) chute of varying inclination angles; filled regions around markers denote 95% confidence intervals. Speedup of the polydisperse algorithm for each case on CPU is also shown.

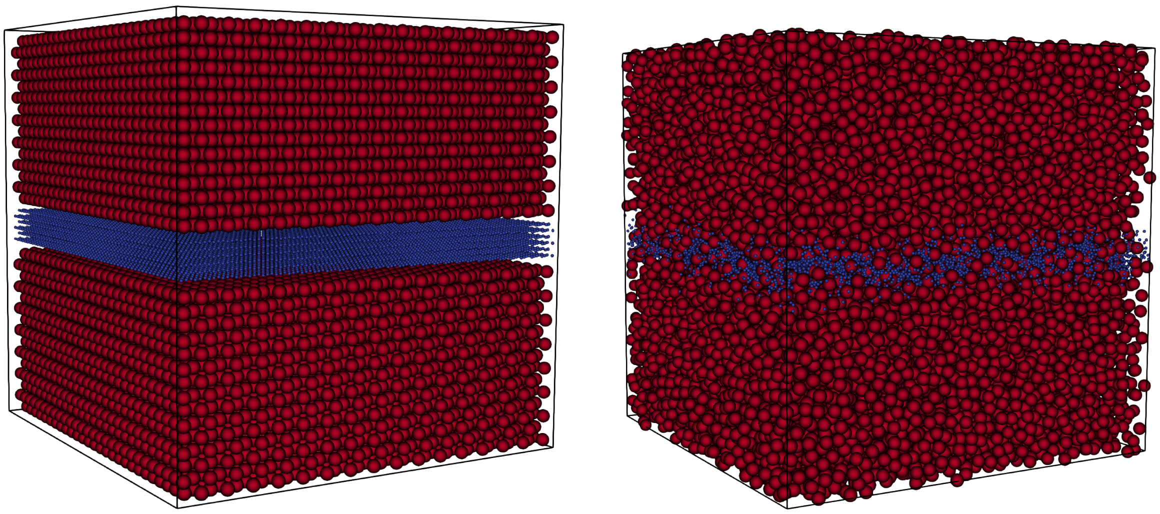

Fig. 4 Initial (left) and final (right) particle configurations of the performance test problem for a size ratio of $\lambda = 4$ and particle count of $N_p = 50$K.

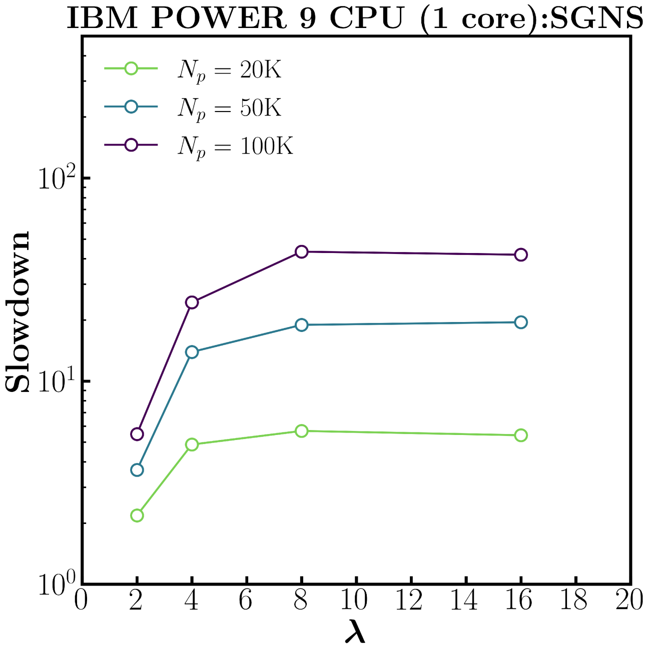

Fig. 5 Slowdown of bidisperse simulations on CPU with varying size ratio and particle count.

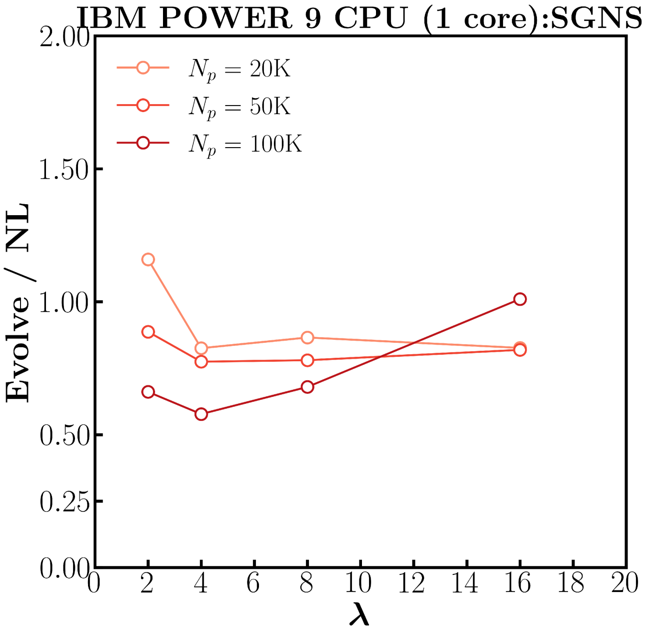

Fig. 6 Comparison of time associated with evolving particles states (Evolve) and neighbor list (NL) construction for SGNS algorithm on CPU.

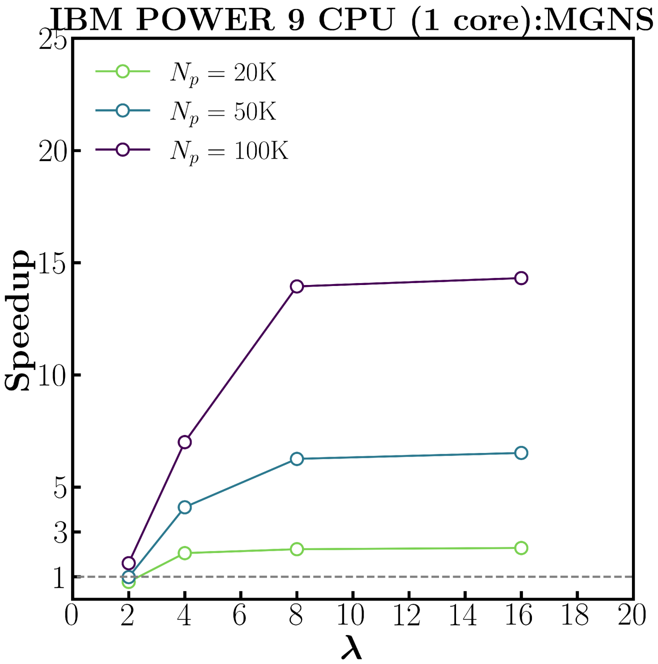

Fig. 7 Speedup of bidisperse simulations on CPU with varying size ratio and particle count.

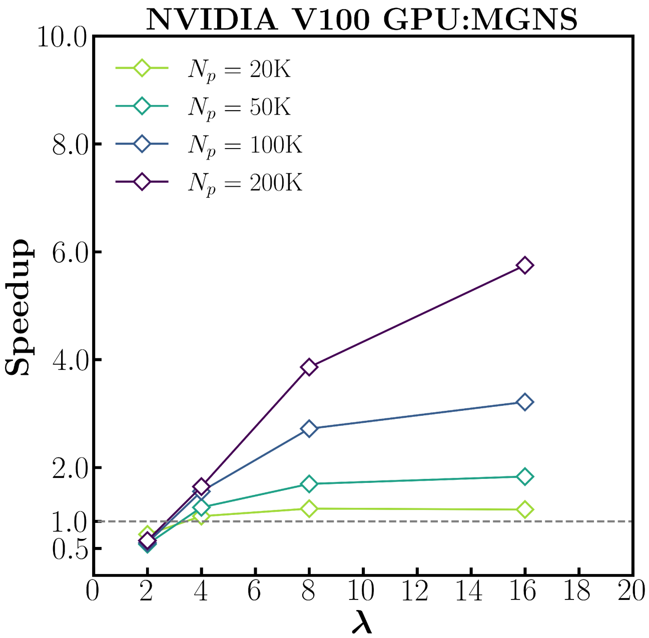

Fig. 8 Speedup of bidisperse simulations on GPU with varying size ratio and particle count.

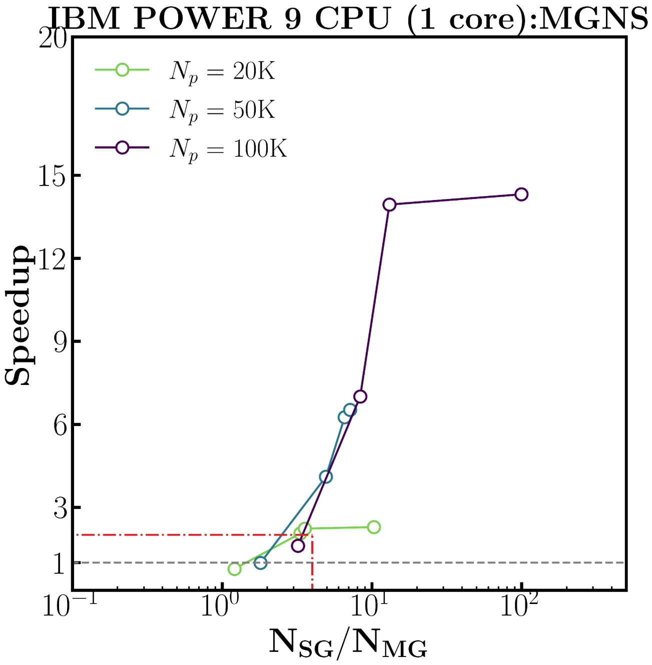

Fig. 9 Speedup of polydisperse simulations on CPU with varying N_{SG}/N_{MG}.

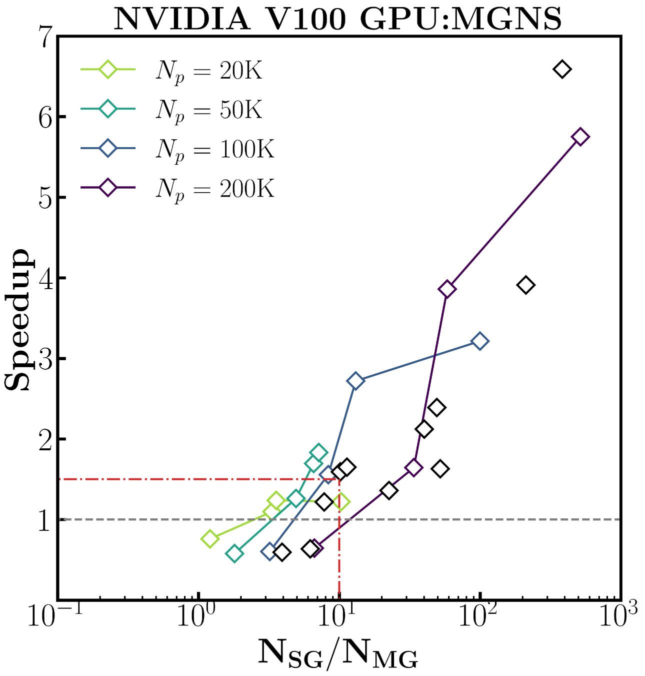

Fig. 10 Speedup of polydisperse simulations on GPU with varying N_{SG}/N_{MG}.

List of Figures

List of Tables