

A parallel dual-grid multiscale approach to CFD–DEM couplings

Gabriele Pozzetti^{a,*}, Hrvoje Jasak^b, Xavier Besseron^a, Alban Rousset^a, Bernhard Peters^a

^a Campus Belval, Université du Luxembourg 6, Avenue de la Fonte, L-4364 Esch-sur-Alzette, Luxembourg

^b Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb, Ivana Lucica 5, Zagreb, Croatia

ARTICLE INFO

Article history:

Received 6 August 2018

Accepted 20 November 2018

Available online 26 November 2018

Keywords:

Parallel interpolation

Dual-grid multiscale CFD–DEM

Parallel performance

ABSTRACT

In this work, a new parallel dual-grid multiscale approach for CFD–DEM couplings is investigated. Dual-grid multiscale CFD–DEM couplings have been recently developed and successfully adopted in different applications still, an efficient parallelization for such a numerical method represents an open issue. Despite its ability to provide grid convergent solutions and more accurate results than standard CFD–DEM couplings, this young numerical method requires good parallel performances in order to be applied to large-scale problems and, therefore, extend its range of application. The parallelization strategy here proposed aims to take advantage of the enhanced complexity of a dual-grid coupling to gain more flexibility in the domain partitioning while keeping a low inter-process communication cost. In particular, it allows avoiding inter-process communication between CFD and DEM software and still allows adopting complex partitioning strategies thanks to an optimized grid-based communication. It is shown how the parallelized multiscale coupling holds all its natural advantages over a mono-scale coupling and can also have better parallel performance. Three benchmark cases are presented to assess the accuracy and performance of the strategy. It is shown how the proposed method allows maintaining good parallel performance when operated over 1000 processes.

© 2018 Elsevier Inc. All rights reserved.

1. Introduction

CFD–DEM couplings are nowadays widely used for the numerical description of engineering and technical problems featuring the interaction between discrete entities and continuous fluid flows [1]. In recent years, several attempts have been made to extend their applicability to large-scale scenarios and more and more complex configurations [2–4]. The interaction between the discrete entities and the underlying fluid can be approached through direct numerical simulation [5,6], or via a so-called unresolved approach [7]. An unresolved or volume-averaged CFD–DEM coupling consists of the projection of Lagrangian fields into Eulerian ones by using volume-averaged variables [7]. In this kind of approaches, the coupling between CFD and DEM solution is performed using cell values or locally interpolated values for the solution of the fluid-particle interactions [8]. This significantly reduces the computational burden of the numerical scheme, and represents one of the main advantages of those approaches. For this reason, unresolved CFD–DEM couplings are the most used for the solution of large-scale scenarios.

* Corresponding author.

E-mail address: gabriele.pozzetti@uni.lu (G. Pozzetti).

When coupling DEM with complex fluids, mono-scale volume averaged CFD–DEM couplings can have severe limitations [4]. In particular, they can not always guarantee grid-convergent nor accurate solutions due to the fact that the CFD grids adopted for the averaging cannot ensure optimal properties for the fluid solutions.

A first attempt to introduce a two-grid formulation for CFD–DEM coupling was proposed in [9], where the authors addressed a problem featuring large particles by calculating the porosity field on a different grid, still without achieving grid-convergence of the solution finding that “A rigorous grid sensitivity analysis is difficult to perform in DEM–CFD framework due to excessive computational time requirement”. In [8], we showed how dual-grid multiscale approach based on a Stoke number criterion was indeed able to overcome the problem of grid convergence even in cases featuring very complex three-phase flows. This approach consists of the identification of two length-scales within the particle-laden flow: a bulk scale at which the fluid-particle interaction is resolved, and a fluid fine-scale at which the equations governing the fluid phase are discretized. Two grids are used in order to perform the two operations: a coarse CFD grid for the bulk scale, and a fine CFD grid for the fluid fine-scale. Eulerian fields are interpolated between the two grids in order to exchange information between the fine and the bulk scale.

As from the detailed comparison proposed in [8], the dual-grid multiscale approach offers significant benefits over the mono-scale one in sequential execution. This approach has been shown to restore grid convergence of both local and averaged properties of the flow where the mono-scale approach cannot. Additional works of [10], [11], [12,13] and [14] relied on this method to tackle very complex engineering problems. Nevertheless, in [8] we pointed out how the enhanced complexity of the multiscale scheme requires a thoughtful consideration for its parallel execution. In particular, it was underlined how the rise in complexity of the multiscale approach with respect to a standard CFD–DEM coupling could represent an issue for its parallelization, as the introduced layer of inter-scale communication could potentially negatively affect the parallel performance of the software.

The parallelization of CFD–DEM couplings is by itself a very difficult and open topic. First, the memory consumption of a coupled CFD–DEM simulation is normally important as, within a process, the information for both the CFD and DEM domain must be stored. Second, in addition to the intra-physic inter-process communication, that represents a major issue for the parallel optimization of the standalone CFD and DEM parts, a generic coupling features an additional inter-physic inter-process communication channel. This can lead to massive inter-process communication of the coupling, that can deeply penalize its performance.

In order to cope with the problem of memory consumption, in [15], the authors proposed an optimized parallelization strategy for the DEM part, and an independent parallelization for the CFD part, that allowed reducing memory consumption while keeping an intensive inter-physic communication. The resulting coupling was observed to scale better than what previously seen in the literature (~ 35 for 64 processes) still performing significantly worse than the sole DEM part. This proved how the inter-physic communication between the Eulerian and the Lagrangian part can induce major performance issues.

In [16], the authors gave further proof of this problem, focusing on a simulation featuring millions of particles, where the efficiency of the DEM part was approximately 81% on 256 processes, while the efficiency of the coupled solution was below 35%, due to the almost 40% of the total time spent in inter-physic communication. In [17], the authors tried to cope with this problem by adopting a communication strategy based on non-distributed memory. The results proved how the reduced communication cost can indeed help the parallel performance of the coupling, yet its advantages are limited to the usage of ~ 30 processes, and therefore, not suitable for very-large-scale simulations.

An attempt to tackle large-scale simulations with an Eulerian–Lagrangian coupling was proposed in [18]. In [18], the authors propose a coupling between the LAMMPS code, originally developed for molecular dynamics applications, and the OpenFOAM® libraries. In that work, the decomposition strategies for CFD and molecular dynamics code were completely independent from one another, and all the required data for the inter-physic exchange was provided through a complex communication layer. It was shown that the specific couple of codes can address large-scale problems and maintain good scalability properties when operated on one hundred of processes. On the other hand, when a larger number of processes is used, the inter-physic communication becomes very important, taking up to 30% of the computation time and creating a major bottleneck for the parallel execution.

In [19], we proposed a way to overcome such a bottleneck. This strategy consists of imposing a co-location constraint between the partitions of the CFD and DEM domains so to perform the inter-physic exchange locally. This strategy was proven to solve the bottleneck presented in [18]. Nevertheless, it reduces the flexibility in the partitioning of the domain, as configurations violating the co-location constraint are no longer possible. In [19], we pointed out how this limitation would be more severe in case of non-uniformly distributed loads.

In this contribution, we propose a parallelization strategy that takes advantage of a parallel implementation of the grid-interpolation executed in the dual-grid multiscale approach, to overcome the shortcomings of the co-located partitioning strategy while keeping its advantages.

The paper is structured as follows: First, the main structure of a dual-grid multiscale CFD–DEM coupling are proposed with focus on the communication required between the DEM element and the CFD grids. Second, the partitioning strategy for the parallel execution of the coupling is proposed and the different communication layers are identified. Third, the grid-based parallel interpolator is proposed in its simpler version. The CFD and DEM equations resolved in section 3 are then briefly recalled. Finally, three benchmark cases are proposed to assess the consistency and performance of the parallelization strategy.

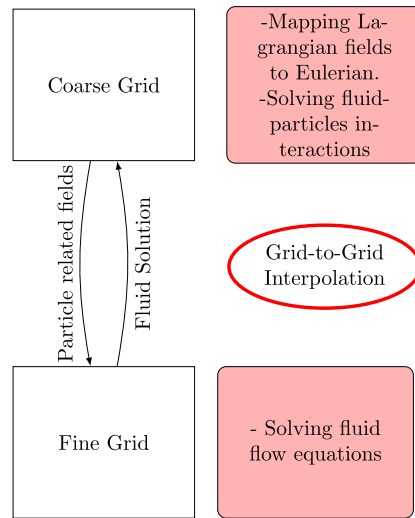


Fig. 1. Diagram of the solution procedure of the two different length scales for the simulation. The two boxes represent the different models, while the arrows show schematically the communication between the scales. A coarse CFD grid (top) is used to map the Lagrangian field of the DEM into an Eulerian reference and to solve the fluid-particle interaction. Particle-related fields are mapped to the supporting domain (bottom) where a finer grid is used to solve the fluid equations.

We claim that the proposed strategy solves the parallelization issue underlined in [8], and at the same time overcomes the lack of flexibility that was affecting [19].

2. Method

2.1. Dual-grid multiscale CFD–DEM coupling

The concept of dual-grid multiscale approach for CFD–DEM couplings was introduced in [8]. In a dual-grid multiscale CFD–DEM coupling, two length-scales are identified within the particle-laden flow: a bulk scale at which the fluid-particle interaction is resolved, and a fluid fine-scale at which the equations governing the fluid phase are discretized. As shown in Fig. 1, a coarse and uniform CFD grid is adopted to resolve the fluid-particle interactions at the bulk scale, and a fine CFD grid is used to solve the fluid flow equations at the fine-scale. An interpolation strategy ensures the communication between the two grids i.e. the two lengthscales.

The dual-grid multiscale approach was originally introduced with reference to the coupling between DEM and VOF as the fine resolution required by the VOF scheme was found to induce a marked separation between the two scales. Nevertheless, this coupling strategy can be applied to a generic CFD–DEM coupling being most beneficial when such a scale separation is present.

From a software point of view, the implementation of the dual-grid multiscale approach requires the presence of two computational grids and a routine to interpolate between them, as depicted in Fig. 1. The first implementation of the dual-grid multiscale approach used the *OpenFOAM-extend meshToMesh* library to perform this interpolation. In the present contribution, we adopt a modified version of the library as discussed in section 2.3 to perform the interpolation between grids in parallel.

In [8] it was pointed out how the coarse and fine CFD grids should be independent from one another in order to allow maximum flexibility of the approach. For instance, while an uniform grid is optimal when performing a Eulerian–Lagrangian coupling, a nonuniform, locally refined, computational grid provides several advantages for the solution of complex flows. For this reason, the two grids will not, in general be nested. When operated in parallel, the interpolation between the two grids will therefore require inter-process communication, as the coarse and fine domain partitions will not be, in general, perfectly aligned. This might represent a problem as the introduced layer of intra-physics communication (i.e. the interpolation between bulk and fluid fine scale) could potentially negatively affect the parallel performance of the software. As proposed in section 2.2, the main idea of the current work is to take advantage of this parallel communication so to avoid inter-physics inter-process communication completely.

2.2. Domain decomposition and multiscale parallel communication

In [19] was shown how using co-located partitions for the CFD and the DEM domains provides significant advantages in the parallel execution of the coupling in terms of both memory consumption and reduction of inter-process communication. In particular, it allows overcoming the inter-physics communication bottleneck that was identified in the literature [18].

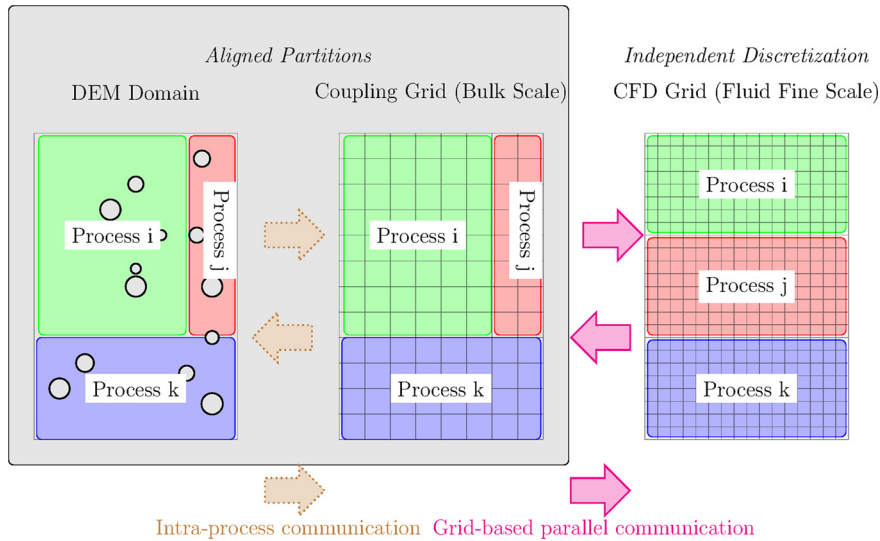


Fig. 2. CFD-DEM domain partitioning and communications.

This bottleneck was caused by the coupling operation that retrieves information from the CFD-grid to the particle position and averages the particle-related quantities to define equivalent Eulerian fields. Since the Lagrangian entities continuously change their position, the connectivity between the particles and the CFD grid needs to be updated at every coupling step. When CFD and DEM information is stored on different processes, this implies the creation of a new data structure that must be sent via MPI messages. When this is done for computationally expensive cases and a high number of processes, the coupling procedure can become the most expensive part of the simulation.

As shown in [19], this problem can be solved by partitioning CFD and DEM domain in such a way that all the information required for the inter-physics exchanges will always be local. This allows keeping the communication time negligible also when operating over hundreds of processes. Nevertheless, the approach proposed in [19] suffers from a lack of flexibility in the domain partitioning, that can represent a main issue for cases featuring a non-uniform distribution of the CFD and DEM parts.

As shown in Fig. 2, we propose to adopt a partitioning strategy based on perfectly aligned co-located partitions between the DEM domain and the coarse grid. In this way, we ensure that the exchange of information between Lagrangian and Eulerian fields will be performed within the same process using standard library calls as in [19].

On the other hand, we rely on a parallel communication for the interpolation between the coarse and the fine CFD grid. In this way, we allow a higher flexibility in the partitioning of the fine CFD grid, to which the computational load of the CFD is associated. This grid does not need to be aligned with the DEM one and will, therefore, allow more complex partitioning strategies. This introduces a higher communication cost if compared to the perfect alignment proposed in [19], but differently from what was happening in [18], the grid-based communication is much simpler than the Eulerian-Lagrangian one, as it relies on a static data-structure that, for non-deforming grids, can be calculated once per simulation.

2.3. Grid-based parallel communication

During a parallel execution, the coarse and fine CFD grids are partitioned into sub-domains that are distributed in different processes. At runtime, every process stores a sub-domain of the coarse grid and a sub-domain of the fine grid. For the sake of simplicity, we now refer to a one-way interpolation where one grid holds the updated information and one grid should receive the interpolated values. We will refer to the first grid as the sender grid and to the latter as the receiver grid. When performing the interpolation between grids, the information can be local (stored in the same process), non local (present in another process) or a combination of the two. We will here refer to a point-to-point communication assuming all the interpolations to retrieve the values to send, and to distribute the received values, to happen locally.

In Fig. 3a, we propose a simple example of two identical grids of 4 elements distributed on two processes i and j . In this case, the interpolation will consist of a simple map of the values from one cell of the sender grid to its twin cell of the receiver. In order to map all the 4 values from the sender grid (*Grid 1*) to the receiver grid (*Grid 2*), four communications are needed. For the communication 1 and 4, the information is local, therefore the operation can be done with simple library calls. On the other hand, for communication 2 and 3, the sender and receiver elements are located on different processes. Therefore, those communications must be performed via MPI.

In Fig. 3b, we propose a representation of the data-structure required for the grid-communication in a more general case of two meshes distributed over n processes. An element (i, j) of the matrix represents a dataset that needs to be communicated from process j to process i . The diagonal terms represent the intra-process communication that are executed

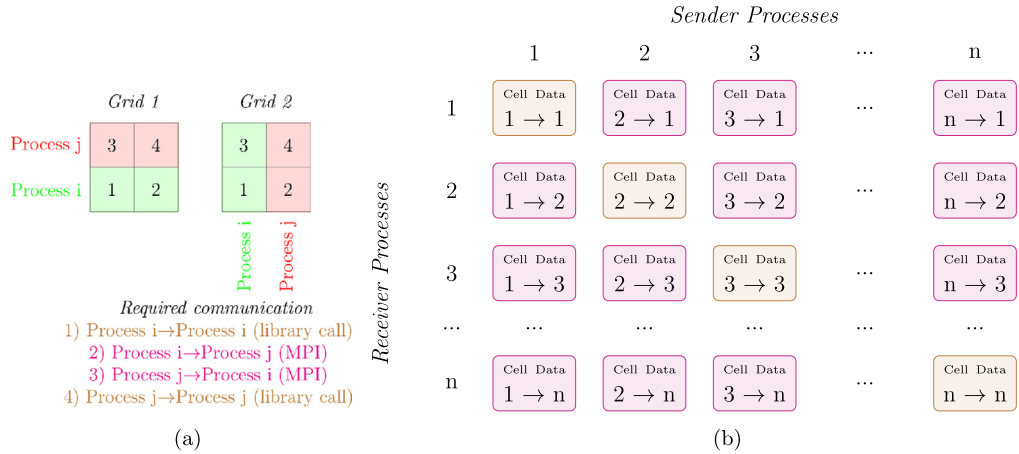


Fig. 3. Grid-based parallel communication: two identical grids of 4 elements distributed on two different partitions 3a, and data structure of a generic grid-based communication 3b.

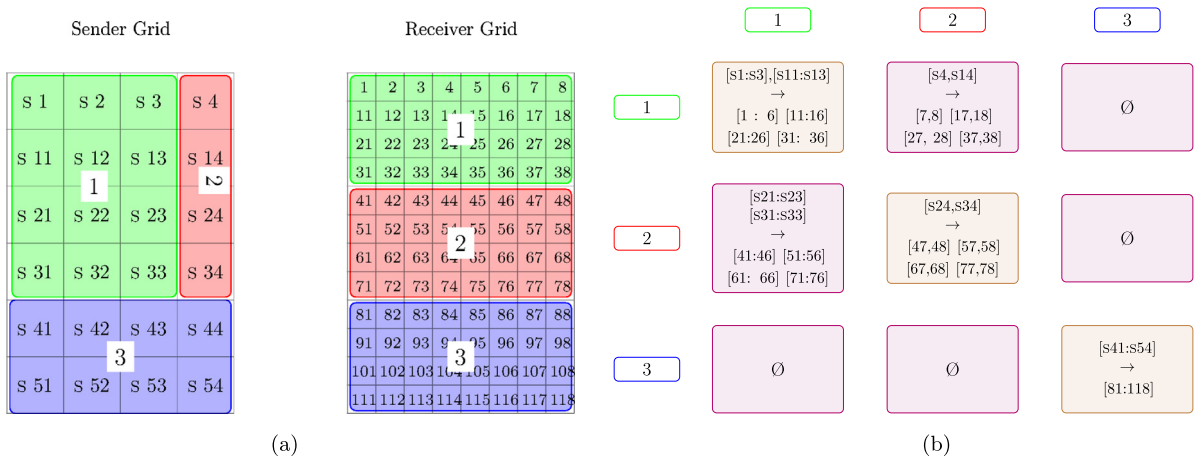


Fig. 4. Grid-based parallel communication: Example of a more complex grid partitioning 4a and the relative communication matrix 4b.

via library calls, while the non-diagonal terms require parallel communication. If the two meshes do not have relative movement over time (as is normally happening in CFD–DEM couplings), the same data-structure will hold during all the simulation, and must be calculated only once. An example of this data structure for a more realistic grid-partitioning is presented in Fig. 4 where it can be observed how, in general, not all the elements i,j will contain data that need to be communicated, and therefore the communication matrix will be sparse.

With the rise in the number of processes, the matrix generally becomes sparser and sparser and, therefore, not all the elements will, in general, contain a dataset. Furthermore, every process will only have to write the k th column and only have to read from the k th row, so that a global knowledge of the data contained in this object is not needed. In particular, the diagonal term (i,i) is never required by process j .

In this work, we compare two strategies for performing the parallel communication to which we will refer to as *gather-scatter* and *distributed*. In a *gather-scatter* strategy, every process holds knowledge of the non-diagonal part of the matrix of Fig. 3b. During the mapping operation, every process k writes the k th columns, then all the information is gathered from the master node and scattered back to all the processes. Afterwards, the process k reads the k th row and stores the interpolated values on the local sub-domain of the receiver grid. This strategy was chosen for its simplicity and to provide a reference to study the importance of the grid interpolation on the parallel performances of the coupling. In a *distributed* strategy, a process k only holds knowledge of the k -th column and the k -th row of the matrix and directly communicates the element k,j (if existing) and receive the element j,k (if existing) from the process j . In this way, the dataset (k,j) will be sent from process k to process j without passing from the master. In section 3.3 we compare the performance of the two methods to underline when a more complex parallel communication is needed.

Table 1
List of variables.

Symbol	Variable	Symbol	Variable
\mathbf{u}_f	fluid velocity	α	volume fraction of the liquid phase
μ_f	fluid viscosity	\mathbf{u}_c	compression velocity
ρ_f	fluid density	Γ	surface tension
\mathbf{F}_{fpi}	volumetric source of momentum due to interaction with particles	\mathbf{T}_Γ	surface tension force
\mathbf{F}_b	body force	ϵ	porosity field: $\sum_i V_{pi}/V_{\text{cell}}$ with $i = 1 \dots n_{pc}$
p	pressure	χ	phase indicator
ρ_p	particle density	\mathbf{F}_g	gravitational force
d_p	particle diameter	\mathbf{M}_{coll}	torque acting on a particle due to collision events
m	particle mass	\mathbf{M}_{ext}	external torque acting on a particle
\mathbf{I}	particle moment of inertia	\mathbf{u}_p	particle velocity
\mathbf{F}_c	contact force acting on a particle	A_p	particle surface area
\mathbf{F}_{ij}	contact force acting on the particle i due to the collision with particle j	ϕ	particle orientation
\mathbf{F}_{drag}	force rising from the particle-fluid interaction	ω	particle angular velocity
\mathbf{n}	particle-fluid interface normal vector	β	drag factor

2.4. Equations solved in the DEM domain

In this work, we rely on the dynamic module of the XDEM Platform [8,20,10], to evolve a set of discrete entities moving in the presence of a multiphase flow. In this section, we briefly recall the equations resolved by this module and the structure of domain-partitioning adopted within XDEM (for a list of variables, see Table 1). The positions and orientations of the particles are updated at every time-step according to

$$m_i \frac{d^2}{dt^2} \mathbf{x}_i = \mathbf{F}_{\text{coll}} + \mathbf{F}_{\text{drag}} + \mathbf{F}_g, \quad (1)$$

$$\mathbf{I}_i \frac{d^2}{dt^2} \phi_i = \mathbf{M}_{\text{coll}} + \mathbf{M}_{\text{ext}}, \quad (2)$$

where \mathbf{x}_i are the positions, m_i the masses, and ϕ the orientations of the entities. The term \mathbf{F}_c indicates the force arising from collisions

$$\mathbf{F}_{\text{coll}} = \sum_{i \neq j} \mathbf{F}_{ij}(\mathbf{x}_j, \mathbf{u}_j, \phi_j, \omega_j), \quad (3)$$

with \mathbf{u}_j the velocity of particle j , and ω the angular velocity. The term \mathbf{M}_{coll} represents the torque acting on the particle due to collisions

$$\mathbf{M}_{\text{coll}} = \sum_{i \neq j} \mathbf{M}_{ij}(\mathbf{x}_j, \mathbf{u}_j, \phi_j, \omega_j), \quad (4)$$

with \mathbf{M}_{ij} the torque acted from particle j to particle i . The term \mathbf{F}_{drag} takes into account the force rising from the interaction with the fluid, and \mathbf{F}_g corresponds to the gravitational force. For what concerns \mathbf{F}_{drag} a semi empirical model in the form:

$$\mathbf{F}_{\text{drag}} = \beta(\mathbf{u}_f - \mathbf{u}_p), \quad (5)$$

$$\beta = \beta(\mathbf{u}_f - \mathbf{u}_p, \rho_f, \rho_p, d_p, A_p, \mu_f, \epsilon), \quad (6)$$

is here chosen, with \mathbf{u}_f , \mathbf{u}_p the fluid and particle velocity respectively, ρ_f , ρ_p the respective densities, d_p , A_p the particle characteristic length and area, μ_f the fluid viscosity, and ϵ the porosity, defined as the ratio between the volume occupied by the fluid and the total volume of the CFD cell. For the sake of generality, we took β as described in [21].

As described in [19], in the parallel execution of XDEM, the simulation domain is geometrically decomposed in regularly fixed-size cells that are used to distribute the workload between the processes. In this way, every process will be assigned a set of cells that will define its sub-domain. Every process only performs the calculation and holds knowledge of the particles that belong to its sub-domain.

Different load-partitioning algorithms are available within the platform XDEM [19], among which a dedicated partitioner that is able to force a co-location constraint between XDEM and OpenFoam partitions as proposed in [19].

2.5. Equations solved in the CFD domain

For the solution of the fluid flow equations, we rely on the OpenFOAM-extend libraries. We rely on the solver presented in [8,22] for tackling the general case of an unsteady incompressible multiphase flow through porous media with forcing terms arising from the particle phase. We here briefly recall the sets of equations governing such a system. For an incompressible flow through porous media, the Navier–Stokes equations take the form

$$\frac{\partial \epsilon \rho_f \mathbf{u}_f}{\partial t} + \nabla \cdot (\epsilon \rho_f \mathbf{u}_f \mathbf{u}_f) = -\epsilon \nabla p + \nabla \cdot \left(\epsilon \mu_f \left(\nabla \mathbf{u}_f + \nabla^T \mathbf{u}_f \right) \right) + \mathbf{T}_\Gamma + \mathbf{F}_b + \mathbf{F}_{fpi}, \quad (7)$$

$$\nabla \cdot \epsilon \mathbf{u}_f = -\frac{\partial \epsilon}{\partial \tau}, \quad (8)$$

with \mathbf{u}_f the fluid velocity, p the fluid pressure, \mathbf{F}_b a generic body force, \mathbf{F}_{fpi} the fluid-particle interaction force, that is the counterpart of \mathbf{F}_{drag} , which is here treated with the semi-implicit algorithm proposed in [23]. Local density and viscosity are dependent on the fluid phase and can be written in the form

$$\rho_f(\mathbf{x}) = \rho_1 \alpha(\mathbf{x}) + \rho_2 (1 - \alpha(\mathbf{x})), \quad (9)$$

$$\mu_f(\mathbf{x}) = \mu_1 \alpha(\mathbf{x}) + \mu_2 (1 - \alpha(\mathbf{x})), \quad (10)$$

with α the volume fraction defined by

$$\alpha = \frac{1}{V} \int_V \chi(\mathbf{x}) d\mathbf{x}, \quad (11)$$

$$\chi = \begin{cases} 1 & \text{if first fluid,} \\ 0 & \text{if second fluid.} \end{cases} \quad (12)$$

The volume fraction is considered a scalar transported by the fluid flow for which

$$\frac{\partial \epsilon \alpha}{\partial t} + \nabla \cdot (\epsilon \alpha \mathbf{u}_f) + \nabla \cdot (\epsilon \alpha (1 - \alpha) \mathbf{u}_c) = 0, \quad (13)$$

must hold, with \mathbf{u}_c the relative velocity between the two-phases referred to as compression velocity. The third term is introduced in order to avoid an excessive numerical dissipation.

This set of equations is solved with the OpenFOAM libraries [24], which parallelization is based on domain decomposition. The CFD domain is split into sub-domains assigned to each process available at run time, over each of them a separate copy of the code is run. The exchange of information between processes is performed at boundaries through a dedicated patch class as described in [24]. According to [19], the partitioning of the OpenFOAM domain is performed through a unique XDEM-OF partitioner that aims to enforce the co-location constraint. In particular, as proposed in section 2.2, the coarse CFD grid is partitioned enforcing a perfect alignment with the XDEM sub-domains, while the fine CFD grid is partitioned independently.

3. Test cases

We here propose three benchmarks to assess the validity and the scalability of our approach. The results refer to the coupling between OpenFoam-extend 3.2 and XDEM as in [19].

The first benchmark, *One particle traveling across a processes boundary* is presented in section 3.1. It tests the equivalence of the results between the sequential run, a parallel run as proposed in [19], and a parallel run with the current approach featuring an arbitrary discretization of the CFD grid. This is done to assess the validity of our approach and implementation by checking the continuity of the results when the particle travels across a process boundary even without partitions alignment.

The second benchmark, *Three-phase dam-break* proposed in section 3.2, re-proposes a famous benchmark featuring complex multiphase interactions and a non-uniformly distributed computational load. This is done to assess how the present parallelization approach preserves the specific properties of the dual-grid multiscale DEM–VOF coupling in terms of accuracy and grid convergence. Also, it shows the benefit of the current strategy in case of non-uniform load distribution in the domain.

The third benchmark, *Ten Million Particles in ten Million cells* described in section 3.3, studies the parallel performance of a coupled solution in case of a heavy coupled case. This is done to show how our solution can handle highly costly simulations and at the same time allows resolving a main issue underlined in the literature linked to the inter-physics communication. Furthermore, the test-case aims to underline the importance of the parallel implementation of grid-based communication for large-scale computation.

The experiments were carried out using the *Iris* cluster of the University of Luxembourg [25] which provides 168 computing nodes for a total of 4704 cores. The nodes used in this study feature a total a 128 GB of memory and have two

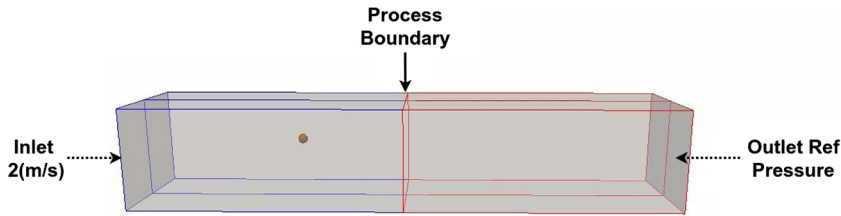


Fig. 5. One particle traveling across processes boundaries: Setup.

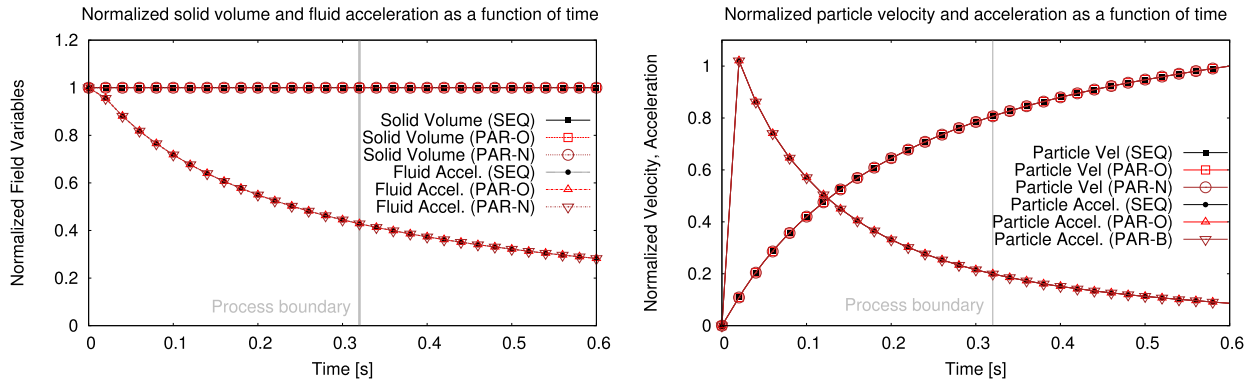


Fig. 6. One-Particle traveling across processes boundaries: comparison between the results obtained with sequential execution (SEQ), parallel execution with overlapping partitions (PAR-O), and parallel execution with non-overlapping partitions (PAR-N). Results for CFD (left) and DEM (right) variables. Continuity across processes boundary can be observed.

Intel Xeon E5-2680 v4 processors running at 2.4 GHz, that is to say a total of 28 cores per node. The nodes are connected through a fast low-latency EDR InfiniBand (100 Gb/s) network organized in a fat tree. We used OpenFOAM-Extend 3.2 and XDEM version b535f789736, both compiled with Intel Compiler 2016.1.150 and parallel executions were performed using Intel MPI 5.1.2.150 over the InfiniBand network. As proposed in [19] [26] the XDEM and OpenFoam libraries are here linked into one executable so that they share the local memory of the node, hence the local inter-physics data exchange can be performed by simple memory access.

To ensure the stability of the measurement, the nodes were reserved for an exclusive access. Additionally, each performance value reported in this section is the average of at least hundred of measurements. The standard deviation showed no significant variation in the results.

3.1. One particle traveling across process boundaries

A first test is proposed, featuring a particle initially at rest traveling across a boundary between processes. This is done in order to show the ability of the proposed approach to deal with arbitrary domain partitions in which the information related to CFD and DEM parts can be stored in different processes. The particle is initially at rest within the domain assigned to process 0. The CFD fine-grid partition overlapping with the DEM partition assigned to process 0 is here assigned to process 1 so to mimic a situation of complete non-overlap of the DEM and CFD local domains. The particle is accelerated by the fluid according to the law of equation 5. The resulting drag force pushes the particle across the boundary with process 1 causing it to be transferred from the sub-domain 0 to the sub-domain 1 of the DEM part. This represents the worse scenario for a coupling, in which none of the information required is local.

As shown in Fig. 5, the boundary conditions imposed on the fluid domain are an uniform Dirichlet at the inlet, no-slip at the wall, and reference pressure at the outlet. The CFD domain is a square channel of dimensions of 1m, 0.2m and 0.2m, and is discretized with 240 identical cubic cells.

In Fig. 6, the normalized particle velocity and acceleration, and the normalized solid volume and L_1 norm of the fluid acceleration are proposed as a function of time. All the quantities are normalized by dividing them for their maximum value, so that they can be displayed on the same plot. The particle crosses the process boundary at 0.32s leaving the DEM domain assigned to process 0 and entering into the domain assigned to process 1, and subsequently leaving the CFD domain assigned to process 1 and entering into the domain assigned to process 0.

It can be observed how the particle velocity and acceleration are continuous across the processes boundary. This shows how the information on the fluid velocity at the particle position is correctly exchanged between CFD and DEM code in the whole domain, including the regions between boundaries. Similarly, the porosity and acceleration fields projected by the particle into the Eulerian grid remain constant along time without suffering discontinuities when the particle switches between processes.

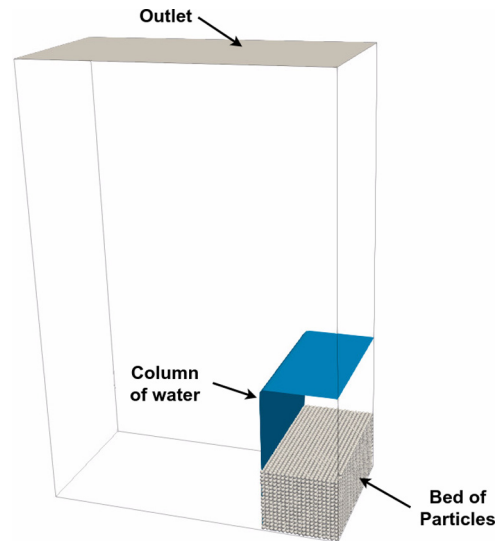


Fig. 7. Three-phase dam-break. Simulation domain setup.

In particular, the results are identical to the one obtained through a sequential execution and a parallel execution with co-located partitions [19]. This shows how the proposed method allows overcoming the limitation induced by the standard co-located partitions strategy extending its applicability to more complex scenarios where identical partitions for the two domains are not possible.

3.2. Three-phase dam-break

In this section, we propose the parallel execution of a benchmark for the multiscale DEM-VOF method. The aim is of showing how, with non geometrically uniform cases, the higher flexibility obtained by operating with two grid offers significant advantages in terms of parallel performance. With this purpose, we compare the parallel execution of a mono-scale DEM-VOF coupling using a co-located partitions strategy, with the parallel execution of a multiscale DEM-VOF coupling based on the parallelization strategy presented in this work.

We here re-propose the multiscale simulation of the three-phase dam break as in [8]. As shown in Fig. 7, a configuration without intermediate obstacle is adopted, in a box of dimensions $0.2 \text{ m} \times 0.1 \text{ m} \times 0.3 \text{ m}$. A column of water of extension $0.05 \text{ m} \times 0.1 \text{ m} \times 0.1 \text{ m}$ with a uniformly layered bed of approximately 10k spherical particles at the bottom, is left breaking and stabilizing. The spheres have a diameter of 2.7 mm. As already proposed in [8], we adopt a fine CFD grid discretized with 500k identical cubic cells while the coarse CFD grid discretized with 6k identical volumes. For the mono-scale approach a domain discretization of 48k cells is chosen, and no coarse CFD grid is used. The simulation parameters are the standard ones as follows. The liquid (heavy phase) density and viscosity are taken as 1000 kg/m^3 and 10^{-3} Pa s respectively, while for the gas (light phase) 1 kg/m^3 and 10^{-5} Pa s are chosen. The particle density is fixed at 2500 kg/m^3 . A linear dashpot impact model is used for both particle-particle and particle-container interactions with spring constant of 1000 N/m , a restitution coefficient of 0.9 and a friction coefficient of 0.3.

As already observed in Fig. 6, Fig. 9 confirms that the parallel execution with the strategy presented in this article, produces the same results as the sequential run and it is validated against experimental data. A further confirmation is provided in Fig. 10 that shows how also in the parallel execution, the kinetic energy reaches proper convergence.

In Fig. 8, the partitioning for the two cases is showed. One can observe how, in the mono-scale case, both the CFD and the DEM domains are partitioned uniformly in order to ensure perfect alignment between the partitions, according to a standard co-located partitions strategy. In this case, due to the non-uniform distribution of the particles, this constraint results in all the particles to be assigned to a single process. On the other hand, in the multiscale approach, the perfect alignment between the two domains is ensured by choosing the coarse CFD grid to be equal to the DEM grid. Therefore, the fine CFD grid can be distributed in an arbitrary way, trying to optimize the ratio between the load balance and the communication.

In [8] was discussed how, despite the usage of a finer CFD grid makes the multiscale approach more computationally expensive, the overall computation time is comparable with the one involved in a mono-scale simulation for what concerns the sequential execution. This is confirmed in Fig. 11, where the computational time of the simulation time is depicted as a function of the number of processes for both the mono-scale and the multiscale approaches. One can observe how the sequential run of the multiscale simulation requires more time. Nevertheless, the enhanced flexibility offered for the parallel execution, allows it to perform better when operated over multiple processes. In particular, when using more than 2 processes, the multiscale approach is indeed faster than the mono-scale. This can be explained by observing Fig. 12 where

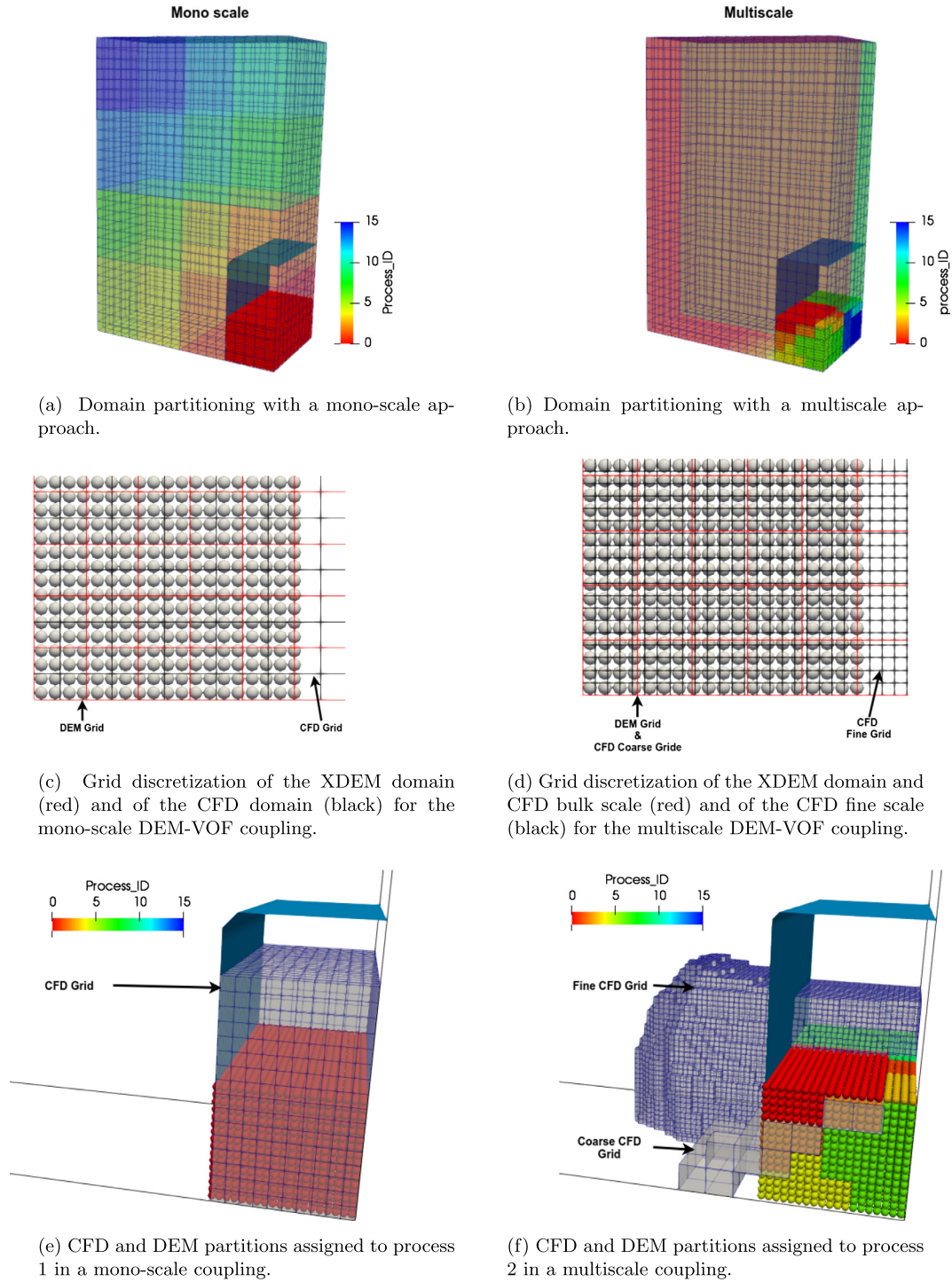


Fig. 8. Three-phase dam-break. Simulation setup. Different possible partitioning of the domain with mono-scale and multiscale approach.

the speedup of the two approaches as a function of the number of processes is proposed. It is clearly shown how the multiscale approach performs significantly better than the mono-scale one.

In conclusion, for this non-uniform case, the grid-based parallelization strategy proposed for the multiscale approach allows adopting a more efficient load distribution, leading to the counter-intuitive result of allowing a simulation that uses an heavier CFD grid to run faster than one that uses a coarser one. Therefore, within a co-located partitioning strategy, the

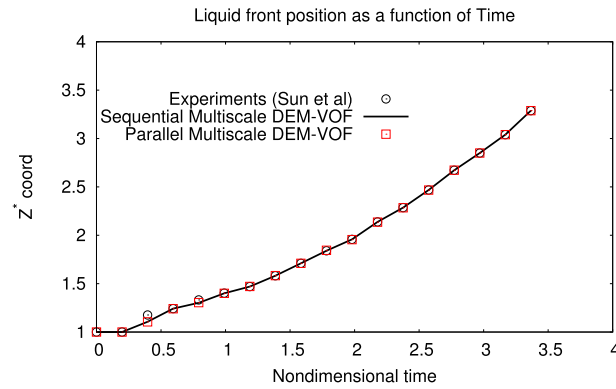


Fig. 9. Three-phase dam-break. Liquid front position as a function of time. Comparison between Experimental data, sequential multiscale simulation and parallel multiscale simulation. Identical results can be observed between parallel and sequential run, and good agreement is shown with the experimental results.

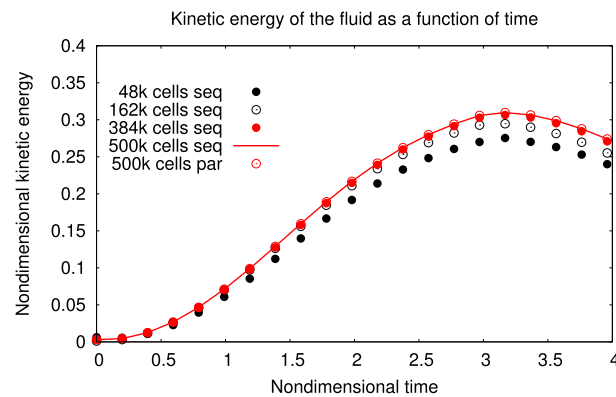


Fig. 10. Three-phase dam-break. Non-dimensional kinetic energy as a function of non-dimensional time. Comparison between sequential run with different fine grids resolution and parallel run with 500k discretization.

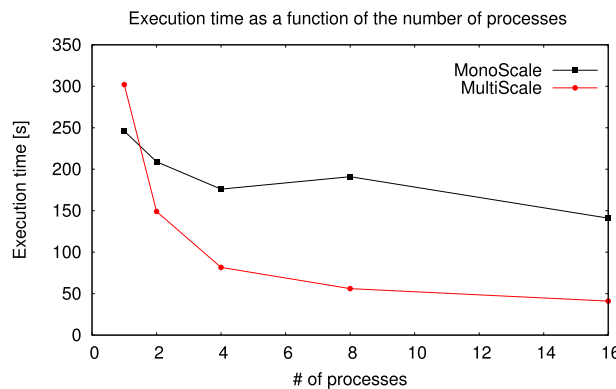


Fig. 11. Three-phase dam-break. Execution time as a function of the number of processes. Better parallel performance is observed for the multiscale approach.

multiscale approach is shown to provide not only a better accuracy but also a reduced computational time when parallel execution is involved.

3.3. 10 Million Particles in 10 Million Cells

In this section, a case consisting of a layered bed of ten million particles moving in the presence of a carrier gas is proposed. This case originally suggested in [18], we here propose the results concerning the parallel execution of the bench-

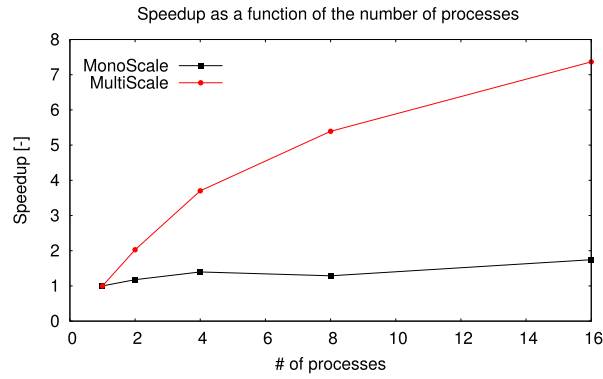


Fig. 12. Three-phase dam-break. Speedup as a function of the number of processes. Better parallel performance is observed for the multiscale approach.

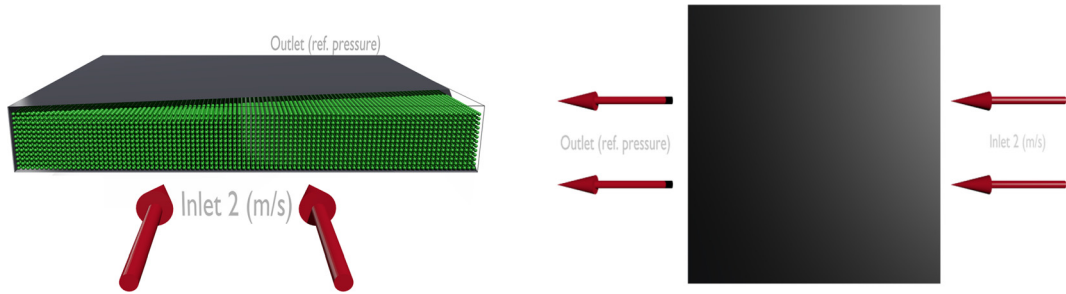


Fig. 13. 10 Million Particles in 10 Million cells. Simulation Setup front view (left) and top view (right).

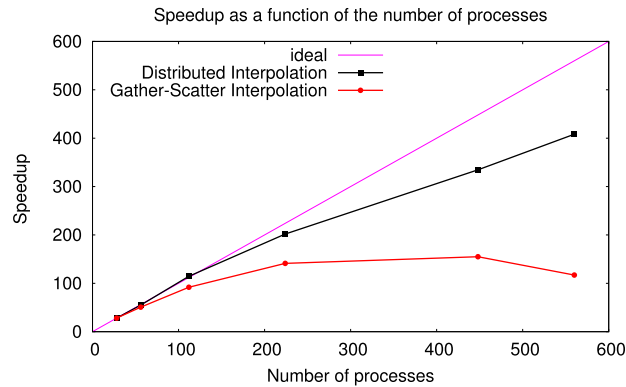


Fig. 14. 10 Million Particles in 10 Million cells. Speedup as a function of time when using gather-scatter and distributed communication for the grid-based interpolator: Significant influence of the communication strategy can be observed while running on more than 100 processes.

mark with a dual-grid coupling in order to test the performance of the parallelization scheme when applied to a large scale problem.

As shown in Fig. 13, the test case features a domain with sizes of $480 \text{ mm} \times 40 \text{ mm} \times 480 \text{ mm}$, as originally presented in [18]. A layered bed of 10 million of particles is moving under the action of an incompressible flow. The boundary conditions for the fluid solution are of constant Dirichlet at the inlet (2 m/s), non-slip at the wall boundaries and reference atmospheric pressure at the outlet.

In Fig. 14, the scalability performance of the coupled XDEM-OpenFOAM run is presented. One can observe how, differently from what observed in [18], good scalability performance when operating over more than 500 processes is achieved. This represents a key improvement for the application of the method to large-scale problems. Furthermore, it can be clearly noticed how the communication strategy chosen for the grid-based interpolator plays an important role in the performance of the overall code when an high number of processes is involved. In particular, while when using less than 100 processes, the gather-scatter strategy seems not to penalize the execution excessively, when adopting 500 processes, the time spent in grid-communication becomes the main bottleneck of the simulation.

This is better shown in Fig. 15, where the time spent in the mapping operation during the first 50 times-steps of the simulation is depicted as a function of the number of processes. Fig. 15 shows how a gather-scatter interpolation is always

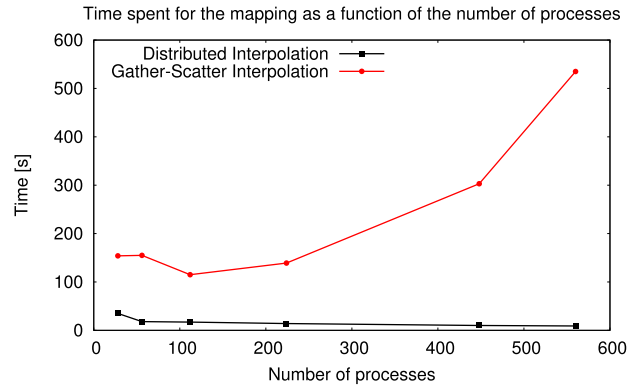


Fig. 15. 10 Million Particles in 10 Million cells. Time spent in the interpolation between grid as a function of the number of processes when using gather-scatter and distributed communication for the grid-based interpolator: significant difference can be observed for the two communication strategies.

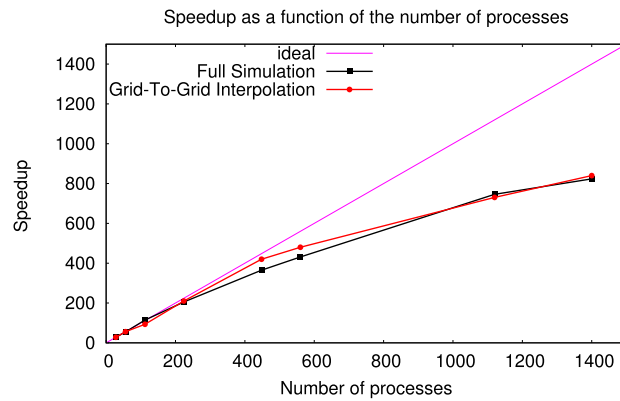


Fig. 16. 10 Million Particles in 10 Million cells. Speedup as a function of time of the full simulation and the grid-to-grid interpolation using a distributed communication for the grid-based interpolator: Good scalability properties of both can be observed when running up to 1400 processes.

worse than a distributed one when operating on more than 28 processes. Furthermore, when using more than 100 processes, the global time required by the interpolation increases with the number of processes when using a gather-scatter approach, while decreases when adopting a distributed one. This shows how, for the current approach, a gather-scatter communication for the grid interpolation is only acceptable when a low number of processes are involved, while for approaching large-scale problems, a more sophisticated communication strategy must be chosen.

In Fig. 16, we report the parallel performance of the coupled XDEM-OpenFOAM run over 1400 processes. Differently from Fig. 14, we here omit the results obtained with the gather-scatter approach as, when using more than 500 processes, the cost of this operation becomes prohibitive. For similar cost reasons, the scalability of the interpolation is obtained with reference to the 28 process run. It can be observed how, when using a distributed approach for the interpolation, good scalability performance can be obtained for both the coupled run as well as the sole interpolation operation. This represents an important result since it allows obtaining a speedup of ~ 830 that is approximately 3.5 times better than what obtained by the previous literature for this test case [19,18].

In Table 2, a weak scalability study is proposed in order to better investigate the parallel performance of the coupling. The study is performed by maintaining a constant number of particles and CFD cells per process and raising the number of processes until reaching the configuration proposed in Fig. 13. The time-step duration was obtained as an average over 100 time-step; the overhead corresponds to the percentage of increase of the time-step duration compared to case A.

One can observe how the timestep duration does not significantly increase with the number of nodes and, more importantly, the cost of the inter-physics exchange remains stable. In particular, case B has double the dimension (and number of processes) of case A and only suffers an overhead of 1%. In case C, the overhead increases again still remaining within acceptable values (2.3%). In order to have a comparison with previous literature, in [16] a total overhead of $\sim 160\%$ was observed on when running 10M particles on 256 processes and in [18] a total overhead of 28% was shown when running over 256 processes and of 100% over 500 processes. This shows that our approach can reduce by approximately two orders of magnitude the overhead when doubling [18] or quadrupling [16] the number of processes. It is important to notice that, in the selected configuration, the CFD part is run with fewer cells per process than what recommended for parallel executions, that is usually 20k cells per process [24]. This was also observed in Fig. 16 where the strong scalability was clearly penalized when less than 20k cells were present in the single process. Those results seem to suggest that in the present

Table 2

Weak scaling test with about 4464 particles and 4464 CFD cells per process. The overhead is calculated as the relative difference with the timestep duration of case A. A slight increase of the time-step duration is observed for bigger cases, while no significant difference in the inter-physics communication can be observed.

Case	Nodes	Total # processes	Total # particles	Total # CFD cells	Timestep duration [s]	Inter-Physics [s]	Total overhead	OF over-head
A	10	280	2.5M	2.5M	1.612	0.0007	0%	0%
B	20	560	5M	5M	1.618	0.0006	1%	0.06%
C	40	1120	10M	10M	1.65	0.0006	2.3%	15%

simulation the CFD is entering into a communication bottleneck that clearly becomes more important as the number of processes rises. For future contributions, it would be interesting to address this unbalance between CFD and DEM load with hybrid OpenMP+OpenMPI parallelizations based on the current scheme.

4. Conclusions

A parallel dual-grid multiscale approach to CFD–DEM couplings has been presented. It is based on the usage of two different CFD grids each associated with a different scale, and extends the multiscale DEM–VOF method to general couplings for large-scale scenarios. The innovative parallel implementation consists of adopting perfectly aligned co-located partitions between the DEM domain and the CFD domain associated with the bulk scale, while allowing independent discretization for the CFD domain associated with the fine scale. This allows avoiding inter-process communication between the CFD and the DEM part and, at the same time, keeping flexibility on the domain partitioning.

The communication between the grid associated with the bulk scale and the one associated with the fine scale is operated through a parallel grid-based interpolator. The data structure that requires being communicated by this interpolator has been presented and two possible solutions for the parallel communication discussed.

Three benchmark cases have been discussed to assess consistency and performances of the proposed strategy. Results showed how the higher flexibility, obtained by the independent partition of the fine mesh, allows the dual-grid multiscale approach to achieve better parallel performance than a single-grid CFD–DEM coupling for inhomogeneous cases. Furthermore, it was shown how the implementation of the parallel grid communicator plays a fundamental role in case of high computational loads. In particular, a gather-scatter communication strategy for the interpolator starts significantly affecting the performance of the overall numerical scheme when operated on more than 100 processes.

One of the main benefits of the current strategy consists of the possibility of adopting more complex partitioning algorithms for the DEM domain and the CFD fine grid, that can, in general, be completely independent from one another. Nevertheless, we want to point out how, using completely independent partitions for CFD and DEM increases the communication costs of the interpolation between grids. Therefore, in order to obtain an optimal partitioning algorithm, this communication must be taken into account. Finally, the definition of optimal partitioning algorithms and interpolations will represent an important topic for future studies.

Acknowledgements

This research is in the framework of the project DigitalTwin, supported by the programme “Investissement pour la compétitivité et emploi” – European Regional Development Fund (Grant agreement: 2016-01-002-06). The experiments presented in this paper were carried out using the HPC facilities of the University of Luxembourg [25].

References

- [1] H.P. Zhu, Z.Y. Zhou, R.Y. Yang, A.B. Yu, Discrete particle simulation of particulate systems: theoretical developments, *Chem. Eng. Sci.* (ISSN 0009-2509) 62 (13) (2007) 3378–3396, <https://doi.org/10.1016/j.ces.2006.12.089>, <http://www.sciencedirect.com/science/article/pii/S000925090700262X>. *Frontier of Chemical Engineering – Multi-scale Bridge between Reductionism and Holism*.
- [2] Bruno Blais, Manon Lassaing, Christoph Goniva, Louis Fradette, François Bertrand, Development of an unresolved cfd–dem model for the flow of viscous suspensions and its application to solid–liquid mixing, *J. Comput. Phys.* (ISSN 0021-9991) 318 (2016) 201–221, <https://doi.org/10.1016/j.jcp.2016.05.008>, <http://www.sciencedirect.com/science/article/pii/S0021999116301358>.
- [3] Tso-Ren Wu, Chia-Ren Chu, Chih-Jung Huang, Chung-Yue Wang, Ssu-Ying Chien, Meng-Zhi Chen, A two-way coupled simulation of moving solids in free-surface flows, *Comput. Fluids* (ISSN 0045-7930) 100 (2014) 347–355, <https://doi.org/10.1016/j.compfluid.2014.05.010>, <http://www.sciencedirect.com/science/article/pii/S0045793014002023>.
- [4] L. Jing, C.Y. Kwok, Y.F. Leung, Y.D. Sobral, Extended CFD–DEM for free-surface flow with multi-size granules, *Int. J. Numer. Anal. Methods Geomech.* (ISSN 1096-9853) 40 (1) (2016) 62–79, <https://doi.org/10.1002/nag.2387>, NAG-14-0182.R1.
- [5] Markus Uhlmann, An immersed boundary method with direct forcing for the simulation of particulate flows, *J. Comput. Phys.* (ISSN 0021-9991) 209 (2) (2005) 448–476, <https://doi.org/10.1016/j.jcp.2005.03.017>, <http://www.sciencedirect.com/science/article/pii/S0021999105001385>.
- [6] Amir Houshang Mahmoudi, Florian Hoffmann, Bernhard Peters, Semi-resolved modeling of heat-up, drying and pyrolysis of biomass solid particles as a new feature in [XDEM], *Appl. Therm. Eng.* (ISSN 1359-4311) 93 (2016) 1091–1104, <https://doi.org/10.1016/j.applthermaleng.2015.10.033>, <http://www.sciencedirect.com/science/article/pii/S1359431115010807>.

- [7] T.B. Anderson, Roy Jackson, Fluid mechanical description of fluidized beds. Equations of motion, *Ind. Eng. Chem. Fundam.* 6 (4) (1967) 527–539, <https://doi.org/10.1021/i160024a007>.
- [8] Gabriele Pozzetti, Bernhard Peters, A multiscale DEM–VOF method for the simulation of three-phase flows, *Int. J. Multiph. Flow* (ISSN 0301-9322) 99 (2018) 186–204, <https://doi.org/10.1016/j.ijmultiphaseflow.2017.10.008>, <http://www.sciencedirect.com/science/article/pii/S0301932216307327>.
- [9] Deb Surya, Danesh K. Tafti, A novel two-grid formulation for fluid–particle systems using the discrete element method, *Powder Technol.* (ISSN 0032-5910) 246 (2013) 601–616, <https://doi.org/10.1016/j.powtec.2013.06.014>, <http://www.sciencedirect.com/science/article/pii/S0032591013004221>.
- [10] Bernhard Peters, Gabriele Pozzetti, Flow characteristics of metallic powder grains for additive manufacturing, *EPJ Web Conf.* 140 (2017) 13001, <https://doi.org/10.1051/epjconf/201714013001>.
- [11] M. Baniasadi, M. Baniasadi, G. Pozzetti, B. Peters, A numerical study on the softening process of iron ore particles in the cohesive zone of an experimental blast furnace using a coupled CFD–DEM method, *arXiv e-prints*, 2018.
- [12] Gabriele Pozzetti, Bernhard Peters, A numerical approach for the evaluation of particle-induced erosion in an abrasive waterjet focusing tube, *Powder Technol.* (ISSN 0032-5910) 333 (2018) 229–242, <https://doi.org/10.1016/j.powtec.2018.04.006>, <http://www.sciencedirect.com/science/article/pii/S0032591018302729>.
- [13] Gabriele Pozzetti, Bernhard Peters, Evaluating erosion patterns in an abrasive water jet cutting using XDEM, *Adv. Powder Metall. Part. Mat.* (2017) 191–205, <http://hdl.handle.net/10993/31360>.
- [14] Liang Wu, Ming Gong, Jingtao Wang, Development of a dem–vof model for the turbulent free-surface flows with particles and its application to stirred mixing system, *Ind. Eng. Chem. Res.* 57 (5) (2018) 1714–1725, <https://doi.org/10.1021/acs.iecr.7b04833>.
- [15] D.K. Kafui, S. Johnson, C. Thornton, J.P.K. Seville, Parallelization of a Lagrangian–Eulerian dem/cfd code for application to fluidized beds, *Powder Technol.* (ISSN 0032-5910) 207 (1) (2011) 270–278, <https://doi.org/10.1016/j.powtec.2010.11.008>, <http://www.sciencedirect.com/science/article/pii/S0032591010005887>.
- [16] Pradeep Gopalakrishnan, Danesh Tafti, Development of parallel dem for the open source code mfix, *Powder Technol.* (ISSN 0032-5910) 235 (2013) 33–41, <https://doi.org/10.1016/j.powtec.2012.09.006>, <http://www.sciencedirect.com/science/article/pii/S0032591012006109>.
- [17] Amit Amritkar, Surya Deb, Danesh Tafti, Efficient parallel cfd–dem simulations using openmp, *J. Comput. Phys.* (ISSN 0021-9991) 256 (2014) 501–519, <https://doi.org/10.1016/j.jcp.2013.09.007>, <http://www.sciencedirect.com/science/article/pii/S0021999113006128>.
- [18] Rui Sun, Heng Xiao, SediFoam: a general-purpose, open-source CFD–DEM solver for particle-laden flow with emphasis on sediment transport, *Comput. Geosci.* (ISSN 0098-3004) 89 (2016) 207–219, <https://doi.org/10.1016/j.cageo.2016.01.011>, <http://www.sciencedirect.com/science/article/pii/S0098300416300218>.
- [19] G. Pozzetti, X. Besseron, A. Rousset, B. Peters, A co-located partitions strategy for parallel CFD–DEM couplings, *arXiv e-prints*, February 2018.
- [20] Mehdi Baniasadi, Maryam Baniasadi, Bernhard Peters, Coupled cfd–dem with heat and mass transfer to investigate the melting of a granular packed bed, *Chem. Eng. Sci.* (ISSN 0009-2509) 178 (2018) 136–145, <https://doi.org/10.1016/j.ces.2017.12.044>, <http://www.sciencedirect.com/science/article/pii/S0009250917307819>.
- [21] L. Schiller, Z. Naumann, A drag coefficient correlation, *VDI Zeitung* 77 (1935) 318–320.
- [22] Gabriele Pozzetti, Bernhard Peters, On the choice of a phase interchange strategy for a multiscale DEM–VOF method, *AIP Conf. Proc.* 1863 (2017), <http://orbi.lu.uni.lu/handle/10993/28863>.
- [23] Heng Xiao, Jin Sun, Algorithms in a robust hybrid CFD–DEM solver for particle-laden flows, *Comput. Phys. Commun.* 9 (2) (2011) 297–323.
- [24] H.G. Weller, G. Tabor, H. Jasak, C. Fureby, A tensorial approach to computational continuum mechanics using object-oriented techniques, *Comput. Phys.* 12 (6) (1998) 620–631, <https://doi.org/10.1063/1.168744>, <http://scitation.aip.org/content/aip/journal/cip/12/6/10.1063/1.168744>.
- [25] S. Varrette, P. Bouvry, H. Cartiaux, F. Georgatos, Management of an academic HPC cluster: the UL experience, in: *Proc. of the 2014 Intl. Conf. on High Performance Computing & Simulation (HPCS 2014)*, IEEE, Bologna, Italy, July 2014, pp. 959–967.
- [26] Gabriele Pozzetti, Xavier Besseron, Alban Rousset, Abdoul Wahid Mainassara Checkaraou, Bernhard Peters, On the performance of an overlapping-domain parallelization strategy for Eulerian–Lagrangian multiphysics software, *AIP Conf. Proc.* 1978 (1) (2018) 270002, <https://doi.org/10.1063/1.5043895>, <https://aip.scitation.org/doi/abs/10.1063/1.5043895>.