A CPU-GPU cross-platform coupled CFD-DEM approach for complex particle-fluid flows

Yi He[*], Frans Muller, Ali Hassanpour, Andrew. E. Bayly
School of Chemical and Process Engineering, University of Leeds, Leeds, LS2 9JT, UK

Abstract

High computational cost presents a significant barrier to the general application of coupled computational fluid dynamics and discrete element method (CFD-DEM) simulations, especially so for industrial systems with a large number of particles and complex geometries. In this study, a new cross-platform coupling approach is developed by integrating a CFD solver with a standalone GPU-based DEM solver via network communication. Consequently, the two modelling techniques benefit from the most appropriate hardware architecture. The developed coupling approach shows predictions comparable to experiments on a small-scale fluidized bed. Its computational performance is evaluated on a larger fluidized bed and shows superior performance over the CPU-based parallelization methods, making DEM calculation no longer the computational bottleneck. Its general applicability to handle complex geometrical domains is further demonstrated by simulations of a gas-solid cyclone separator. This work demonstrates the benefits of a novel coupling approach which enables efficient and robust solutions for industrial applications.

Keywords: CFD; DEM; CFD-DEM coupling; GPU; particle-fluid flow; ANSYS Fluent

[*] Corresponding author: Yi He (y.he1@leeds.ac.uk)

1. Introduction

Particle-fluid flow has been a subject of research for years due to its practical relevance to many industrial sectors, for example, chemical, pharmaceutical and mineral processing industries. However, understanding of these systems is often limited and empirical, although significant effort has been made in experimental studies using non-intrusive techniques. With the increasing availability of computing resources, the coupled CFD-DEM method has become an important tool to model these systems, as it provides detailed information at particle scale which is often difficult to obtain by the conventional experimental techniques (Hoomans et al., 1996; Tsuji et al., 1993; Xu and Yu, 1997). Typically, such modelling can be achieved using either a single application, such as MFIX (Garg et al., 2012; Li et al., 2012a) and most of the in-house codes or coupling of two applications, such as Fluent-EDEM (Kloss et al., 2009a), Fluent-Rocky and coupling between OpenFOAM and LIGGGHTS (Kloss et al., 2012b). However, practical applications of this method are largely hindered due to its associated computational cost. In addition, complex geometrical domain that is often encountered in industrial applications requires the method to be compatible with non-uniform unstructured CFD mesh. Consequently, there is an increasing interest to develop an efficient and robust coupling approach to handle industrially relevant problems.

In the classical coupled CFD-DEM approach, fluid flow is resolved by a grid-based CFD method while solid phase is modelled by DEM with each particle subjected to hydrodynamic forces, body forces(e.g. gravity), and contact forces and with its velocities and positions updated according to Newton's second law of motion (Hoomans et al., 1996; Tsuji et al., 1993; Xu and Yu, 1997). In principle, all CFD methods can be coupled with DEM. In fact, many CFD methods have been attempted, ranging from discrete to continuum methods, such as Lattice Boltzmann Method (LBM) (Han et al., 2013; Xiong et al., 2014), Smoothed Particle Hydrodynamics (SPH) (Cleary et al., 2006; He et al., 2018a), Finite Difference (Varas and Peters, 2017) and Finite Volume Methods (FVM) (Kuang et al., 2008; Xiao and Sun, 2011; Yang et al., 2015). These coupled CFD-DEM methods have been extensively used to model different systems, including fluidized bed (Jajcevic et al., 2013; Li et al., 2012b), high shear mixer (Washino et al., 2013), cyclone separators (Chu et al., 2009a; Chu et al., 2009b), pneumatic conveying (Kuang et al., 2013) and powder inhalers (Tong et al., 2013). However, most of the previous studies are limited to either quasi-2D or small-scale systems. The reason can be attributed to the large number of particles, the time-consuming contact detection and the small time steps required to resolve collisions in DEM calculations (Gopalakrishnan and Tafti, 2013; Liu and Hrenya, 2014). This is particularly true for concentrated systems and for systems with particles with a wide size distribution or with complex shapes. Furthermore, geometrical complexity of the computational domain and turbulence modeling of the fluid flow further make it challenging to accomplish the simulation in a reasonable time.

To accelerate CFD-DEM simulations, parallel computing has been increasingly used due to the increased availability of computing resources. Most efforts so far have been devoted to DEM parallelization, as it is often regarded as the computational bottleneck for CFD-DEM simulations.

Many algorithms have been proposed for this purpose, such as mirror domain technique (Darmana et al., 2006; Washington and Meegoda, 2003), particle subset method (Kafui et al., 2011) and domain decomposition methods (Amritkar et al., 2014; Tsuji et al., 2008). To implement these algorithms, the most suitable programming model is often determined by the hardware architecture. Parallelization on shared memory system is often achieved using OpenMP (Open Multi-Processing) while MPI (Message Passing Interface) is used for distributed memory systems (Rabenseifner et al., 2009). MPI can also be used for parallelization on shared memory systems by treating each CPU core as a separate entity. For example, Tsuji et al. (2008) parallelized their in-house CFD-DEM code using MPI to exchange data among CPUs and simulated a gas-solid fluidized bed with 4.5 million particles on 16 CPUs using a one-dimensional domain decomposition method. Amritkar et al. (2014) showed that OpenMP can achieve better performance than MPI in fluidized bed and rotary kiln simulations because of reduced communication needs and improved CPU load balancing. Recently, hybrid MPI/OpenMP parallelization is increasingly used. Yakubov et al. (2013) proposed a hybrid parallelization for the simulations of a cavitating flow, in which CFD calculation was distributed among CPU cores using MPI processes while OpenMP threads were initiated by each MPI process to perform Lagrangian calculation. To date, previous studies have mostly focused on coupling DEM with in-house CFD codes. However, the lack of capability to handle complex geometries and multiple physics limits their application, especially to industrial problems.

In recent years, coupling DEM with open source or commercial CFD solvers is gaining popularity. A coupled simulation can be executed either as a single executable or multiple executables (Chimakurthi et al., 2018). Kloss et al. (2009b) developed a separate coupling interface to handle the coupling between two commercial software codes EDEM and ANSYS Fluent. However, at the time EDEM was limited to a shared memory system. The coupling scheme was later extended to coupling open-source codes LIGGGHTS and OpenFOAM, which allows a coupled simulation to be parallelized on a distributed memory system (Kloss et al., 2012a). Gopalakrishnan and Tafti (2013) coupled DEM with an open-source code MFIX, in which CFD and DEM calculation were paralleled using a same three-dimensional domain decomposition method. Amritkar et al. (2014) coupled DEM with a CFD package GenIDLEST (Tafti, 2001) using a hybrid MPI/OpenMP parallelization method, in which domain decomposition was used for CFD calculation while thread-based OpenMP parallelization was used for DEM calculation. Wu et al. (2014) incorporated a DEM code into ANSYS Fluent through its UDFs, where the particle phase shares the same domain decomposition as the CFD solver. This is achieved by assigning particles to the same compute node of the cell in which the particle resides, resulting in a similar scalability as the CFD solver.

It is generally accepted that domain decomposition is the best parallelization technique for a grid-based CFD method while DEM is best to be parallelized over the number of particles (Amritkar et al., 2014). Previous studies have shown that using GPU-based DEM allows a simulation to be performed more time-efficient than CPU-based parallelization methods, because of the high floating-point performance and parallel architecture of GPU. On GPU, more transistors are devoted to data processing rather than data caching and flow control. The locality of particle-particle interaction makes it a feasible tool to achieve general applications of DEM. For example, Radeke et al. (2010) showed that mixing of several million of particles can be simulated on a single GPU. Xu et al. (2011) reported a quasi-real-time simulation of a rotating drum of industrial scale using more than 200 GPUs. In recent years, the development of GPU-based DEM has been extended to non-spherical particles (Govender et al., 2018), particles with large size ratio (He et al., 2018c), applications with complex geometries (Gan et al., 2016) and coupling with SPH for free-surface flows (He et al., 2018a).

However, to the best of our knowledge, there are only three studies reported in the literature focusing on the coupling between a GPU-based DEM and a grid-based CFD solver. Xu et al. (2012) are the first who reported a multi-scale CPU-GPU hybrid computing for simulations of gas-solid fluidization, in which gas flow is computed on CPUs using an in-house CFD code while particle phase is handled on GPUs. At the time of their implementation, mapping particles to cells, calculations of void fraction and drag forces are all handled by CPUs and copied to GPUs before updating particle status. Jajcevic et al. (2013) coupled a GPU-based DEM solver with a commercial CFD code (AVL-Fire$^{®}$), where a separate software component (i.e. AVL Code Coupling Interface) was used to handle data exchange between two codes. Data mapping is required between CFD mesh and a DEM grid (i.e. the linked cell space used for neighbour searching). In the work of Norouzi et al. (2017), a GPU-based DEM code was incorporated into an open-source code OpenFOAM as an integrated part, in which the coupling between two phases was calculated on CPU. In particular, a tree-based search algorithm provided by OpenFOAM was used to map particles to fluid cells. In spite of the parallelizing capability of GPU, in the previous studies, part or all of the calculations of interphase coupling parameters are handled on CPU, such as volume fraction and momentum source terms of each CFD cell, volume fraction and fluid forces acting on individual particle. Consequently, the parallel capability of GPU cannot be fully leveraged.

The objective of the present study is therefore to propose an efficient and robust coupling approach that combines the massive parallel capability of GPU for DEM calculation with the general applicability and flexibility of a commercial software for CFD calculation. Our coupling approach has the following significant features: (i) the DEM calculations are fully parallelized on GPU by a standalone in-house DEM solver (named as **HiPPS** (High Performance Particulate System)) (He et al., 2018a; He et al., 2018c), (ii) the DEM solver is coupled with a commercial CFD software, thus enabling a cross-platform hybrid CPU-GPU computing for coupled simulation while utilizing the flexibility of a commercial software for general-purposed industrial applications, (iii) data exchange between the CFD and DEM solver is handled by network communication, without relying on a separate coupling interface, (iv) only cell-based hydrodynamic information is transferred between two solvers, significantly reducing the amount of data being transferred, and more importantly, (v) calculations of the coupling parameters are all handled by GPU, largely enhancing the computational efficiency of the coupled simulation.

The paper is organized as follows: the methodology of the coupled simulation is first outlined and followed by a detailed description of the coupling scheme and its implementation. Thereafter, the coupling approach is validated against literature results on a small-scaled fluidized bed where quantitative experimental measurements are available. The performance of the coupling approach is evaluated on a larger fluidized bed and is compared with CPU-based parallelization methods. The capability of the present model in handling general industrial applications with complex geometrical domain is demonstrated by simulating a gas-solid cyclone separator where particles are generated continuously.

2. Model development

A typical coupled CFD-DEM approach solves volume-averaged Navier-Stokes equation using grid-based CFD methods while calculating the interactions between particles, particle and wall, particle and fluid flow by DEM. In this section, numerical models are first briefly outlined. This is followed by a detailed description of its implementations in the CFD solver, GPU-based DEM solver and communication between these two standalone solvers. Further details of force models can be found in our previous studies (He et al., 2018a; He et al., 2015).

2.1. Model description

The governing equations for the fluid phase follow the conventional Two Fluid Model (TFM) (Anderson and Jackson, 1967), given as,

$$\frac{\partial(\varepsilon\rho_{\mathrm{f}})}{\partial t} + \nabla\cdot(\varepsilon\rho_{\mathrm{f}}\mathbf{u}) = 0 \tag{1}$$

$$\frac{\partial(\varepsilon\rho_{\mathrm{f}}\mathbf{u})}{\partial t} + \nabla\cdot(\varepsilon\rho_{\mathrm{f}}\mathbf{u}\mathbf{u}) = -\varepsilon\nabla P + \nabla\cdot(\varepsilon\boldsymbol{\tau}_{\mathrm{f}}) + \varepsilon\rho_{\mathrm{f}}\mathbf{g} - \mathbf{S}_{\mathrm{p}} \tag{2}$$

where ρ_{f} is the fluid density, P is the pressure, $\boldsymbol{\tau}_{\mathrm{f}}$ is the fluid-phase shear stress tensor, $\mathbf{g}$ is the acceleration due to gravity, ε is the volume fraction of fluid in each cell, which account for the presence of the particle phase, $\mathbf{S}_{\mathrm{p}}$ is the momentum source term due to the effect of particulate phase.

In DEM, the translational and rotational motion of a given particle, *i*, with mass m_{i} and moment of inertia I_{i} are governed by the Newton's second law of motion,

$$m_{\mathrm{i}}\frac{d\mathbf{v}_{\mathrm{i}}}{dt} = \sum_{\mathrm{j}}\mathbf{F}_{\mathrm{c,ij}} + \mathbf{F}_{\mathrm{f}} + m_i\mathbf{g} \tag{3}$$

$$I_{\mathrm{i}}\frac{d\boldsymbol{\omega}_{\mathrm{i}}}{dt} = \sum_{\mathrm{j}}\left(\mathbf{T}_{\mathrm{t,ij}} + \mathbf{T}_{\mathrm{r,ij}}\right) + \mathbf{T}_{\mathrm{f}} \tag{4}$$

where $\mathbf{v}_i$ and $\boldsymbol{\omega}_i$ are the translational and rotational velocities, respectively; The forces acting on each particle consists of several contributions, including the collision component $\mathbf{F}_{\mathrm{c,ij}}$ arising from collisions with other particles or walls, the hydrodynamic component $\mathbf{F}_{\mathrm{f}}$ due to fluid-particle interactions and gravitational force, $m_i\mathbf{g}$. The collision between two particles is handled by a soft-sphere model which correlate the contact forces with the magnitude of interparticle overlap. The torques include the contribution of tangential contact force $\mathbf{T}_{\mathrm{t,ij}}$, the rolling resistance $\mathbf{T}_{\mathrm{r,ij}}$ caused by asymmetric distribution of the contact pressure and $\mathbf{T}_{\mathrm{f}}$ due to hydrodynamic effect.

The normal contact force between two particles is described by Hertzian contact theory while the Mindlin and Deresiewicz theory (Mindlin and Deresiewicz, 1953) governs the elastic frictional contact behaviour in the tangential direction. It should be noted that other non-contact forces, such as van der Waals force and capillary force, are required when dealing with fine particles or cohesive wet particles (He et al., 2018b).

Typical particle-fluid interaction forces include pressure gradient force, drag force, virtual mass force, and Basset force and lift forces. For gas-solid flows, the total fluid-particle interaction forces can be further simplified to,

$$\mathbf{F}_{\mathrm{f}} = -V_p\nabla P + \mathbf{F}_d \tag{5}$$

The first term on the right side is the pressure gradient force, where V_p represents volume of particle and ∇P is the pressure gradient at the particle position. $\mathbf{F}_d$ is the drag force determined by porosity, flow velocity and particle velocity at the position of the particle, which can be written as,

$$\mathbf{F}_d = \frac{\beta V_p}{1-\varepsilon}(\mathbf{u}-\mathbf{v}) \tag{6}$$

with β the interphase momentum exchange coefficient. In this study, the drag model of Ergun and Wen & Yu was adopted (Ergun, 1952; Wen and Yu, 1966).

To resolve the fluid phase, cell-based void fraction and volumetric particle-fluid interaction force are required. The momentum source terms $\mathbf{S}_{\mathrm{c}}$ can be calculated by summing up the drag force acting on each particle in a fluid cell so that Newton's third law of motion is satisfied (Xu and Yu, 1997), which is given by,

$$\mathbf{S}_{\mathrm{p}} = \frac{1}{V_{\mathrm{cell}}}\sum_{i=1}^{N_{\mathrm{c}}}\frac{\beta V_{\mathrm{i}}(\mathbf{u}-\mathbf{v}_{\mathrm{i}})}{1-\varepsilon_i} \tag{7}$$

with V_{cell} being the volume of a fluid cell and N_{c} the number of particles in a fluid cell.

2.2. Model implementation

The DEM calculation was parallelized on a single GPU while CFD calculation was carried out on CPU cores using a commercial CFD code, ANSYS Fluent v17.1, which is designed to efficiently solve Navier-Stokes equations on complex geometries and boundary conditions based on the Finite Volume Method (FVM). Communication between two solvers is enabled via network socket on top of the TCP/IP (Transmission Control Protocol/Internet Protocol) protocols, which provides a dedicated channel for communication via a computer network. Consequently, CFD and DEM calculations can be executed either on a same physical machine or on different machines as long as they are connected by a network.

It should be noted that the CFD solver follows a domain decomposition method for parallelization. The computational domain is divided into multiple subdomains, each assigned with its own compute process (or node). Each process executes the same instructions on its portion of mesh, following a SPMD (Single Process, Multiple Data) mode of computation. On the other hand, the DEM solver, HiPPS, was parallelized on a single GPU using C++ and Compute Unified Device Architecture (CUDA). It was formulated using a Single Instruction Multiple Threads (SIMT) model of parallel programming, where same instruction is executed by multiple threads simultaneously. Due to the discrete nature of particle method, GPU threads are assigned to each particle for the DEM calculations. Therefore, neighbor searching, force calculation and time integration of the equation of motion can be carried out independently for each particle using different GPU kernel functions. For the efficient use of GPU memory, parameters that remain unchanged during the calculation, such as material properties, are stored in the constant memory (a type of read-only memory on the GPU with fast data fetching) while other particle-related information, including positions, velocities, forces and contact histories, are stored in the global memory on GPU.

Fig. 1 shows the coupling architecture between the CPU-based CFD solver (i.e. ANSYS/Fluent) and the GPU-based DEM solver. In ANSYS Fluent, there is a process called *host* which does not store any mesh or solution data. Instead, the host process only interprets commands from the graphics-related interface, *Cortex* (a process responsible for user-interface and graphics-related functions). These commands (or data) are first passed to compute node-0 which is then responsible for distributing this information to other compute nodes. To reduce communication overheads, the DEM solver is directly connected to the compute node-0 instead of the *host* process in the present coupling scheme. As a result, the compute node-0 is responsible for gathering cell-based information from all compute nodes, distributing source terms received from the DEM solver to other computes nodes and sending/receiving data to/from the DEM solver.

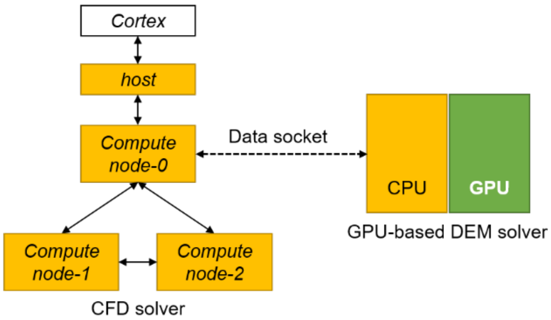


Fig. 1 Architecture of coupling ANSYS Fluent with GPU-based DEM solver (HiPPS).

2.2.1 Implementation on the CFD solver

The algorithm was implemented in ANSYS Fluent via its UDFs. As illustrated in Fig. 2, the UDFs are called at the end of each CFD time step through the macro DEFINE_EXECUTE_AT_END. After establishing a data socket, mesh-related information is sent from ANSYS Fluent to the GPU-based DEM solver, including number of CFD cells, coordinates of cell centres, volume of each CFD cell and CFD time step. Note that this is only done once for the case where the CFD mesh is not going to change during simulation. During each CFD time step, fluid velocity, velocity gradients and pressure gradient of each CFD cell are sent to the DEM. These fluid properties are later used to calculate the fluid force acting on each solid particle. After the DEM calculation, the updated void fraction and momentum source of each CFD cell are sent back to Fluent. The macro DEFINE_ADJUST is then used, responsible for assigning these source terms to each CFD cell incrementally at each iteration.

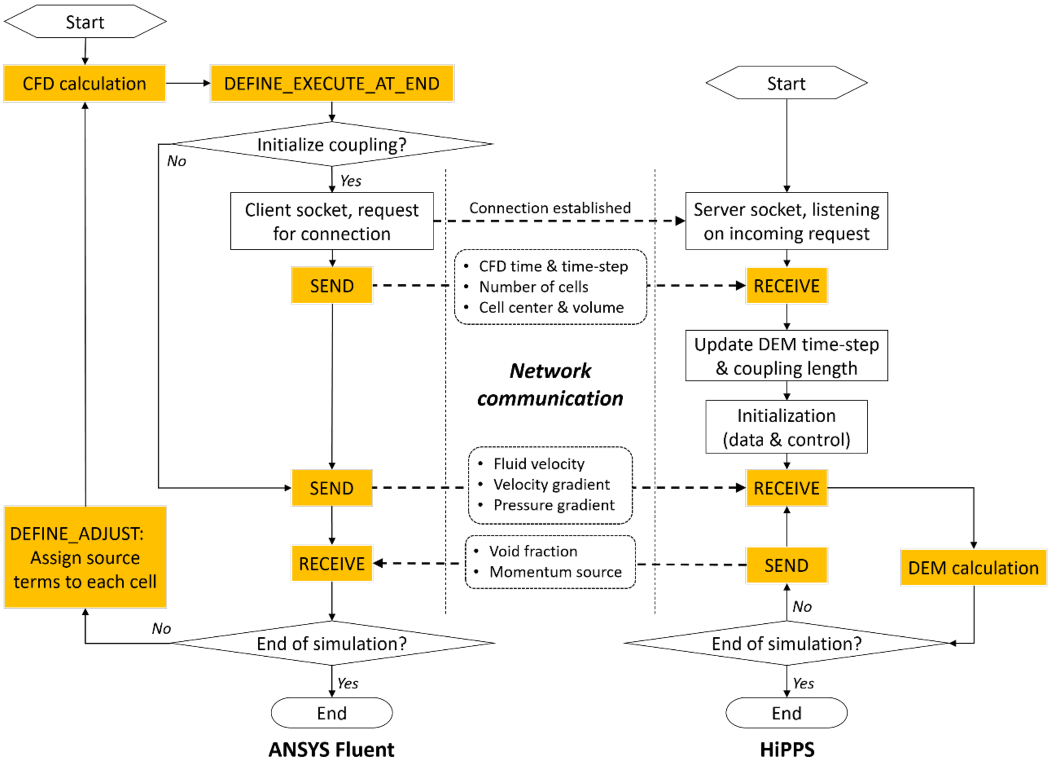


Fig. 2 calculation sequence of a coupled CFD-DEM simulation. Those yellow shaded boxes are the main loop of a coupled simulation.

Since all the required hydrodynamic information has been sent to GPU, calculations of all the coupling parameters can be handled by the DEM solver on GPU. The UDFs on the Fluent side are only responsible for communication with the DEM solver (i.e. gathering and distributing data across computing nodes). It should be emphasized that the data being exchanged between two solvers are all cell-based, significantly reducing the communication overhead compared with exchanging particle-based information. For example, if N_{cell} denotes the number of cells used for CFD calculation and N_p is the number of particles in the system, the amount of double-precision floating point numbers being transferred at each CFD time step is $19N_{cell}$. Whereas for particle-based communication, the amount of data to be exchanged is $21N_p$. Considering the fact that the number of CFD cells is much smaller than the number of particles for concentrated systems, the computational efficiency can be significantly enhanced by the present coupling approach and the communication load is more or less insensitive to the number of particles being simulated in the system.

2.2.2 A dual-grid approach

In a coupled CFD-DEM approach, fluid phase is resolved at the computational cell level in the Eulerian framework while particle's motion is tracked individually in the Lagrangian framework. To achieve phase coupling, it is therefore necessary to interpolate particle properties to the centroid of a given CFD cell and fluid properties to the position of a given particle. A dual grid searching approach is proposed here for this purpose. As schematically shown in Fig. 3, two axially aligned searching grids are created: the particle search grid (colored by yellow) and the fluid search grid (colored by blue). In the following, key steps relying on the dual search grids are outlined, including collision detection for particles, mapping CFD cells, calculations of fluid forces and void fraction for a given particle, and calculations of void fraction and momentum sources for each CFD cell.

- Collision detection for particles

Solid particles are mapped onto the particle search grid for collision detection, as shown in Fig. 3(a). The grid size is normally set as 1-2 times of the largest particle size, same as the conventional linked-cell method. All the potential particle-particle collisions can then be identified from the cell containing the given particle and its immediate surrounding search cells. If dealing with particles with large size ratio, a multi-grid searching method specially designed for the GPU architecture can be used to handle the neighbour searching (He et al., 2018c). For simplicity, mono-sized spherical particles are shown here. This step is performed on GPU, with each CUDA thread handling one particle. A radix sort algorithm from the *Thrust* library is applied to sort particles according to the cell indices of the particle search grid (Hoberock and Bell).

- Mapping CFD cells to search grid

CFD cells are mapped onto the fluid search grid, as shown in Fig. 3(b). In order to be compatible with unstructured CFD mesh, a coupling length $L_{i,\mathrm{cell}}$ is defined for each CFD cell. It is basically the volume equivalent sphere diameter of a CFD cell, calculated as,

$$L_{i,\mathrm{cell}} = \sqrt[3]{6V_{i,\mathrm{cell}}/\pi} \tag{8}$$

with $V_{\mathrm{i,cell}}$ the volume of a given CFD cell. The maximum value of $L_{\mathrm{i,cell}}$ is then used to determine the grid size of the fluid searching grid. It should be noted that mapping CFD cells onto the fluid search grid is only done once if dealing with stationary CFD mesh while repeated mapping is required if dynamic meshing is enabled for the CFD calculation. This mapping step is performed on GPU by assigning a CUDA thread to each CFD cell, with the aid of the radix sort algorithm (Hoberock and Bell).

- Calculation of fluid forces acting on each particle

To calculate fluid forces, fluid properties at particle's position are required, including pressure gradient, fluid velocities and velocity gradients for a gas-solid flow. These fluid properties are typically stored at the centroid of a CFD cell in an FVM calculation. Cell-based fluid information can be interpolated to the particle position by,

$$\phi_{\mathrm{p}} = \phi_{\mathrm{cell}} + \nabla\phi_{\mathrm{cell}} \cdot \mathbf{r}_{pc} \tag{9}$$

where ϕ_{p} and ϕ_{cell} are the fluid properties at particle position and the cell centroid, respectively; $\mathbf{r}_{pc}$ is the distance vector pointing from the cell centre to the particle position. This is achieved by mapping a given particle onto the fluid search grid, as schematically shown in Fig. 3(b). The nearest CFD cell can be identified by looping through the search cell hosting the given particle and its immediate search cells on the fluid search grid. This step is performed on GPU, with each CUDA thread handling one particle.

- Calculation of local void fraction of each particle

To calculate local void fraction for each particle, a method proposed by Kuang et al. (2008) is adopted here. Each particle is associated with a spherical sampling domain, with a size three times of the particle diameter. The local void fraction is calculated as,

$$\epsilon_{\mathrm{i,p}} = 1 - \frac{1}{V_{\mathrm{sample}}} \sum_{j}^{n_p} V_{j,sp} \tag{10}$$

where n_p is the number of particles within the sampling domain, $V_{j,sp}$ is the fractional volume of particle *j* inside the spherical sampling domain and V_{sample} is the volume of the sampling domain. For GPU implementation, each particle is assigned with a CUDA thread. As shown in Fig. 3(a), neighbour searching is conducted by looping through the cell hosting the given particle and its immediate three surrounding layers on the particle searching grid. Attention should be given to particles which are located near the boundary due to a truncated sampling domain.

- Calculation of void fraction and momentum source of each CFD cell

In analogy to the method mentioned above, void fraction of a CFD cell is calculated as the local void fraction at the cell centroid. The coupling length $L_{i,\text{cell}}$ is used as the sampling size (Eq. (8)). Each CFD cell is first mapped onto the particle search grid, as shown in Fig. 3(c). Then, the number of layers n_{layer} being searched in the particle grid is determined as,

$$n_{\text{layer}} = \lceil L_{i,\text{cell}}/L_p \rceil \tag{11}$$

where L_p is the cell size of the particle search grid, the nomenclature $\lceil x \rceil$ returns the smallest integer larger than *x*.

In a same sense, source terms $\mathbf{S}_p$ is calculated by summing up fluid drag forces of each particle within the sampling domain,

$$\mathbf{S}_p = \frac{1}{V_{\text{sample}}} \sum_{j}^{n_p} \alpha_j \mathbf{F}_{d,j} \tag{12}$$

where α_j is the fractional volume of each particle inside the sampling domain. n_{p} is the number of particles within the sampling domain, which is different from that in Eq. (10). For implementations on GPU, this is achieved by assigning a CUDA thread to each CFD cell.

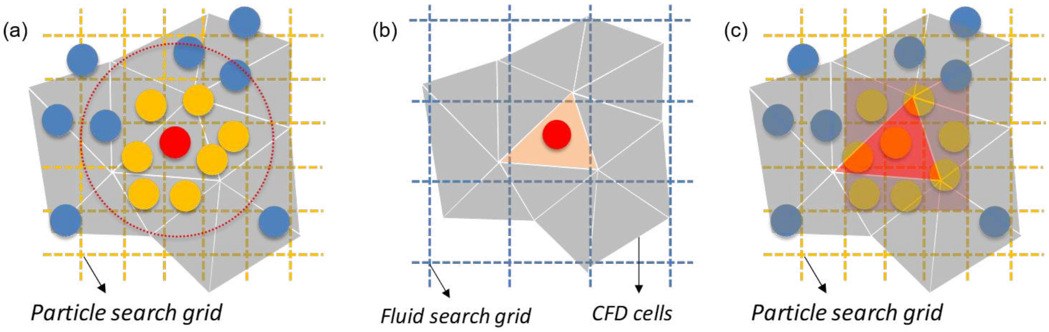


Fig. 3 Schematic illustration of dual-grid approach for data mapping and neighbor searching, in which (a) neighbor searching of the particle phase and calculation of local void fraction for a given particle, (b) mapping particle onto fluid searching grid to interpolate fluid information at the particle's position and (c) mapping CFD cells onto the particle search grid to calculate source terms for a given CFD cell.

2.2.3 HiPPS: a GPU-based DEM solver

Fig. 4 shows the flow chart of the algorithm of HiPPS that is fully executed on GPU. The algorithm has two major modules: phase coupling and calculations of the particle phase. It starts with data initialization by reading control parameters from input files. One of the key parameters is the port number which is required to establish data socket between the CFD and DEM solvers. The socket server is hosted by HiPPS while the socket client is hosted by UDFs of ANSYS Fluent. Once the connection is established, the current simulation time, time step and cell-related properties of CFD calculation are sent to HiPPS. The cell-related properties include the number of cells, coordinates of cell center and its volume. This step, however, is only done once at the beginning of a coupled simulation when dealing with stationary CFD mesh. For dynamic meshing, the cell-related information should be updated in the DEM solver when the CFD mesh is changed.

To ensure numerical stability, DEM calculation normally requires a time step that is much smaller than the CFD time step. The DEM time step is thus updated after receiving the CFD time step, so that,

$$\Delta t_{\mathrm{P,new}} = \Delta t_{\mathrm{F}} / \left\lceil \frac{\Delta t_{\mathrm{F}}}{\Delta t_{\mathrm{P,old}}} \right\rceil \quad (13)$$

in which Δt_{F} is the received CFD time step, $\Delta t_{\mathrm{P,old}}$ is the original DEM time step calculated according to a Rayleigh wave propagation criteria (Zhu et al., 2007). Multiple DEM iterations are thus executed during each CFD time step.

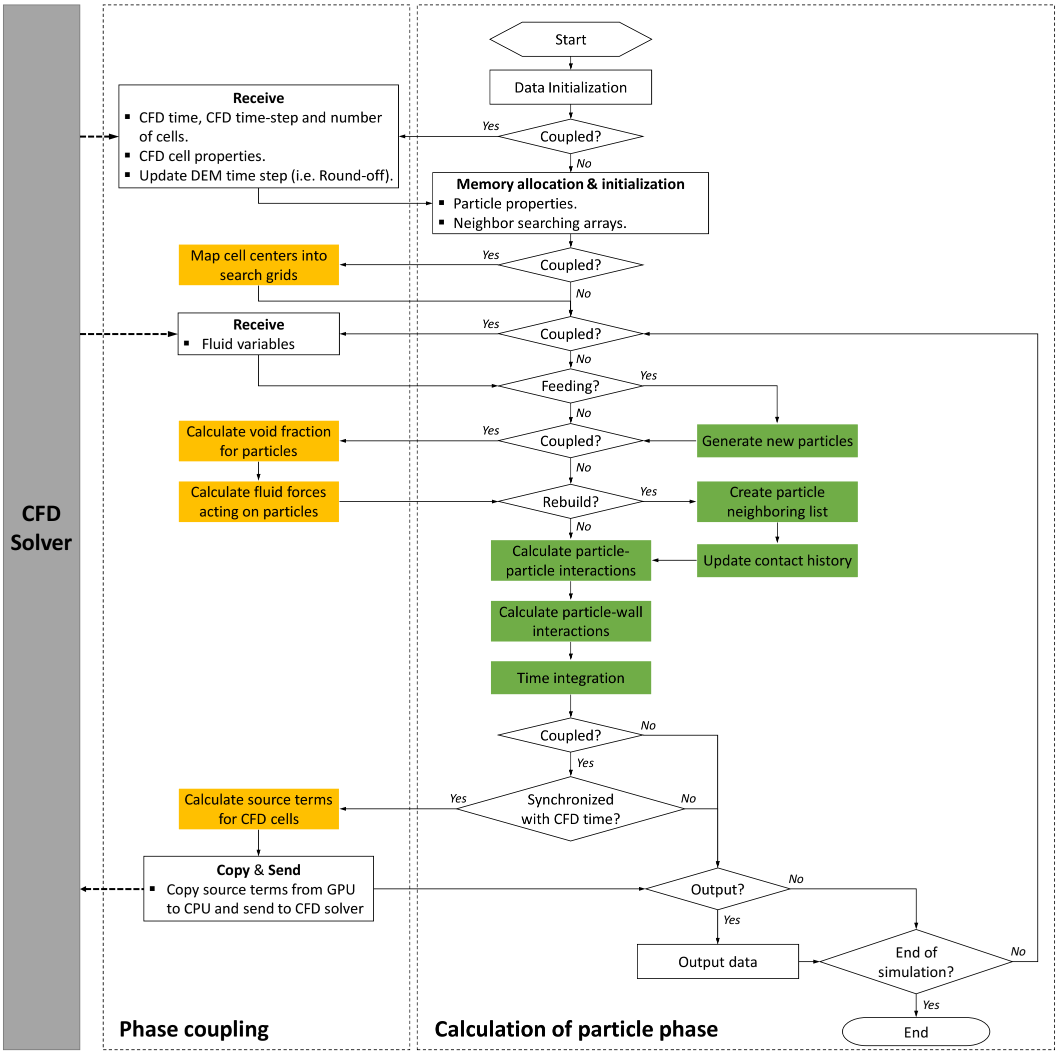

CFD Solver
Receive
CFD time, CFD time-step and number of cells.
CFD cell properties.
Update DEM time step (i.e. Round-off).
Map cell centers into search grids
Receive
Fluid variables
Calculate void fraction for particles
Calculate fluid forces acting on particles
Calculate source terms for CFD cells
Copy & Send
Copy source terms from GPU to CPU and send to CFD solver
Phase coupling
Start
Data Initialization
Coupled?
Yes
No
Memory allocation & initialization
Particle properties.
Neighbor searching arrays.
Feeding?
Generate new particles
Rebuild?
Create particle neighboring list
Update contact history
Calculate particle-particle interactions
Calculate particle-wall interactions
Time integration
Synchronized with CFD time?
Output?
Output data
End of simulation?
End
Calculation of particle phase

Fig. 4 Flow chart of the algorithm of HiPPS that is fully parallelized on GPU. The steps in the shaded boxes are paralleled on GPU, in which the yellow-shaded boxes represent the

calculations corresponding to phase coupling and green-shaded boxes are the main steps for the DEM calculation.

Memory allocation is conducted on both CPU and GPU for arrays related to particle properties and neighbor searching. The received CFD cells are mapped onto the fluid search grid according to the coordinates of cell centroid, which will be used to identify the CFD cell residing the particle of interest. This step is parallelized on the GPU by assigning a CUDA thread to each CFD cell and is only done once unless the computational domain is re-meshed.

After completing the initialization, the main loop of the coupling simulation starts. The data socket on the DEM side listens until all the necessary fluid information are received, including velocities, velocity gradients and pressure gradients of fluid flow. These received fluid variables are then copied to the GPU memory and are used later for the calculations of fluid forces acting on solid particles. The main steps of DEM calculation follows an order of generating new particles in the system (if necessary), calculation of local void fraction for solid particles, calculation of fluid forces, searching neighbors, update of the contact histories, calculations of interactions, and update of particle information. These procedures are repeated multiple times ($\Delta t_{\mathrm{F}}/\Delta t_{\mathrm{P,new}}$) until DEM time is synchronized with CFD time. After that, the void fraction and momentum exchange terms for each CFD cell are calculated on GPU by assigning a CUDA thread to each CFD cell. These updated source terms will be copied from GPU to CPU memory and sent to the CFD solver via data socket. The flow field can then be updated by the CFD solver. During the CFD calculation, the DEM solver can either halt to receive the updated flow field or write DEM results to hard drive if reaching specified output time. The output data includes particle positions, velocities and forces for post-processing purpose. In this sense, time cost of output DEM data is hidden by the CFD calculation, thus able to further enhance the computational efficiency of a coupled simulation.

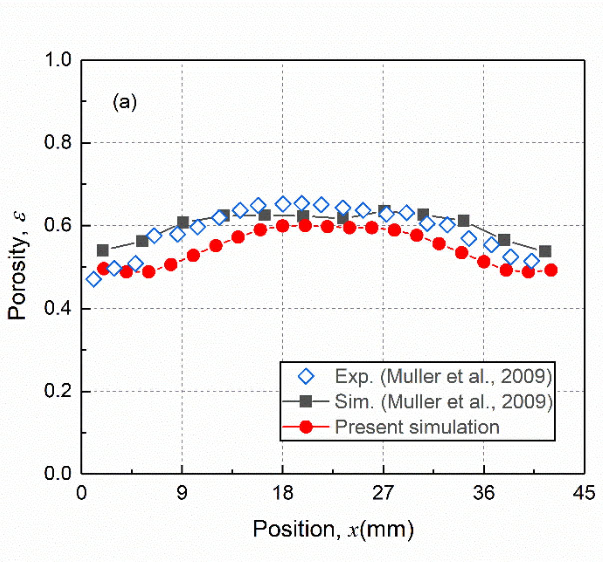
3. Results and discussion

In this section, the CFD-DEM model is first validated against literature data on a small-scale fluidized bed. To evaluate the efficiency of the proposed coupling scheme, comparisons with both MPI- and OpenMP-based parallelization are made. Scalability of the scheme is tested on a larger fluidized bed, with varying number of particles and different number of CPU cores for CFD calculation. The capability to handle industrial applications with complex geometries is demonstrated by a gas-solid cyclone separator with different levels of solid loading. All the simulations were run on ARC3 (a high-performance computing facilities at the University of Leeds), where a NVIDIA P100 card was used for the DEM calculation. The ARC3 system is based on CentOS 7 distribution, with Broadwell E5-2650v4 processor and 5.3 GB of memory/core.

3.1. Model validation: small fluidized bed

Simulations of a quasi-2D fluidized bed are performed to validate the proposed coupling scheme. The bed has a dimension of 44mm × 10mm × 120mm. Gas is supplied from the bottom distributor plate at a constant speed of 0.9 m/s. The simulation setup is the same as that used in the previous study (Muller et al., 2009). Velocity inlet and pressure outlet are used, with no-slip conditions applied to the side walls. A total physical time of 20s is simulated. Data is sampled every 10ms after the first 3s to avoid the start-up period to get statistically meaningful comparison with the experiments (Muller et al., 2009). Quantitative comparisons are made in terms of time averaged velocity profile and void fraction distributions.

Fig. 5 shows the time-averaged porosity profile at two different heights above the distributor plate: 16.4 mm and 31.2 mm. As shown in Fig. 5(a), the porosity is underestimated by the present simulation at the height of 16.4mm, with a maximum difference within 12% and an average difference within 8% relative to the experimental value. However, the general trend observed in the experiments is well captured, where a lower porosity is present in the vicinity of the side walls due to the arising air bubble in the middle of the bed. The present results also show a good agreement with the previous simulations (Muller et al., 2009). Compared with experimental data (Muller et al., 2009), at the height of 31.2mm, the porosity is slightly under-predicted in the middle region but is over-predicted when it is close to the side walls. The difference is smaller than 8% in the middle of the bed but slightly larger than 10% near the side walls compared to the experimental value. The present results are comparable to the simulation results of Muller et al. (2009) in the middle region, with a slightly better match with the experiments at two sides. Similar voidage profiles have also been reported by Li et al. (Li et al., 2012c). The deviation from experimental results can be attributed to several reasons. Frist, the way of calculating local void fractions of CFD cells and of solid particles affects the calculation of inter-phase drag force. Peng et al. (2014) found that the size of local sampling can affect accuracy of the calculations of void fraction. Secondly, spherical spheres were used in the simulation while the poppy seeds are non-spherical in the experiments, which may cause considerable difference in the drag force and contact mechanics.



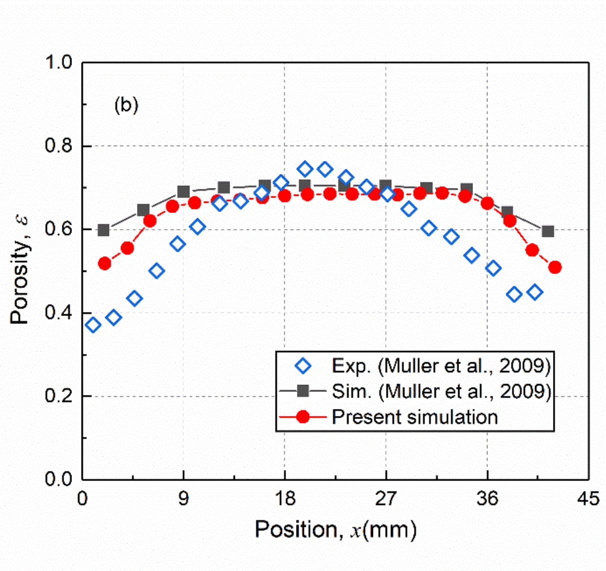
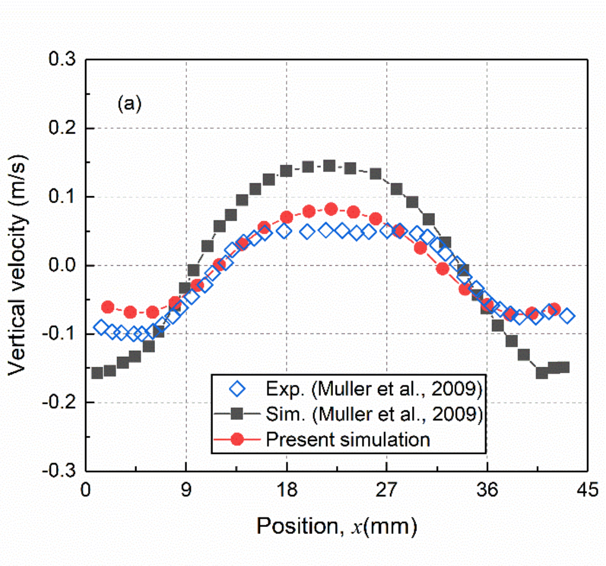
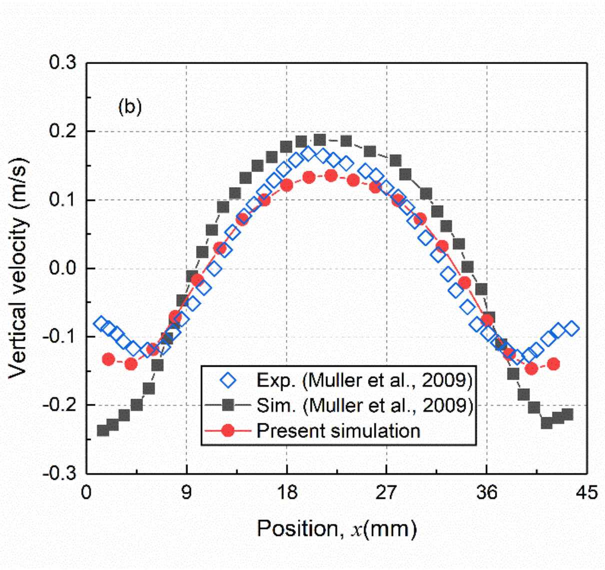


Fig. 5 Comparison of time-averaged porosity profile for a superficial velocity of 0.9m/s at different heights: (a) 16.4mm and (b) 31.2mm.

Three additional heights are also monitored to give time-averaged vertical particle velocity profiles: 15 mm, 25 mm and 35 mm. As shown in Fig. 6, a better match with the experiments can be obtained by the present simulation at the height of 15 mm and 25 mm than that of the previous studies. In particular, the up-turning tails of the velocity profile in the vicinity of the side walls is better captured. However, particle velocity at the height of 35 mm is underestimated in the central region of the bed, leading to an underestimation of the porosity as shown in Fig. 5. In general, the present simulation can produce qualitatively similar results as that observed in the experiments, with only minor difference in quantitative details.





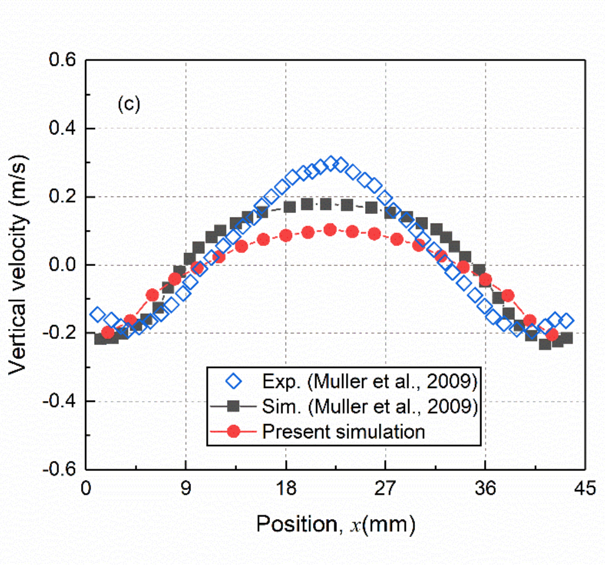


Fig. 6 Profile of vertical velocity of particles at different heights of the bed: (a) 15mm, (b) 25mm and (c) 35mm.

3.2. Model performance: large fluidized bed

The performance of the present coupling scheme is evaluated on a larger fluidized bed and compared with the results presented in the study of Amritkar et al. (Amritkar et al., 2014), where both MPI-based and OpenMP-based parallelization were performed over Intel Xeon X7542 processor and 5.3GB of memory per core. The scalability of the coupling scheme is also evaluated by varying the number of particles and the number of CPU cores used for the CFD calculation.

The fluidized bed has a dimension of 384*d* × 384*d* × 192*d*, with *d* the particle diameter. The CFD cell size is set as 3 times of the particle's diameter. Other simulation parameters are summarized in Table 1. The setup is similar to that of Amritkar et al. (Amritkar et al., 2014), except that Hertzian-Mindlin contact model is adopted here while linear contact model was used before

(Amritkar et al., 2014). For the gas phase, a no-slip boundary condition is applied at the bed walls and pressure outlet condition is applied to the top exit. Gas is supplied at a velocity of 2.6m/s from the bottom of the bed. A time step of 6×10^{-5}s is used for both the DEM and CFD calculations, which is consistent with the study of (Amritkar et al., 2014) in order to make a proper comparison. However, it should be noted that the DEM time step is normally smaller than the CFD time step, which is essentially a compromise between numerical accuracy and computing efficiency.

Table 1 Parameters used in the simulations of larger fluidized bed.

Parameter	**Value**
Particle diameter (mm)	4.0
Particle density (kg/m^3)	2700
Young's modulus (Pa)	1.0×10^7
Poisson's ratio	0.3
Sliding friction coefficient	0.3
Rolling friction coefficient	0.01
Restitution coefficient	0.9
Gas density (kg/m^3)	1.225
Gas inlet velocity (m/s)	2.6
Gas viscosity (kg/m/s)	1.789×10^{-5}
Number of particles ($\times 10^6$)	0.5, 1, 2, 4, 8, 10
Grid	128 × 128 × 64
Time step (s)	6.0×10^{-5}

To evaluate the performance, a total number of 1,327,104 particles are simulated, in accordance with the previous study (Amritkar et al., 2014). The computational cost is shown in Fig. 7. Here, the computational cost is averaged over 1000 CFD time steps. Due to limitations of the available computational resources, CFD calculations are only tested up to 24 cores.

As shown in Fig. 7(a), the present coupling scheme shows a performance comparable to the MPI parallelization in terms of the average computational cost of a coupled simulation (Amritkar et al., 2014). For the CFD calculation, ANSYS Fluent requires 2-3 times more time than that in the study of Amritkar et al. (Amritkar et al., 2014), despite it drops sharply with increasing number of CPU cores (see Fig. 7(b)). In contrast, the GPU-based DEM solver outperforms the CPU-based parallelization, providing 3.81 times speedup over OpenMP and 6.31 times speedup over MPI using 32 CPU cores. In particular, the DEM cost shows little change with the number of CPU cores used for the CFD calculation, as it has been fully parallelized on GPU. Here, the DEM cost includes the computational time of both the phase coupling and the particle phase.

Several possible reasons for the difference in the CFD cost between two studies can be identified, for example, the choices of pressure-velocity coupling scheme, convergence criteria and specification of CPUs. In fact, the Intel X7542 used in the previous study comes with a higher base frequency than the Intel Xeon E5-2650 v4 processor used in the present study. It should also be emphasized that the aim of the present study is not to benchmark the CFD calculation but instead focusing on the benefit of parallelizing the calculations of the particle phase and phase coupling on GPU. As illustrated by Fig. 7(d), the computational cost of DEM solver is less than 10% of the total cost of a coupled simulation, even when the CFD calculation is paralleled on 24 cores. The computational bottleneck is clearly shift to the CFD calculation, as opposed to the traditional coupled CFD-DEM simulation.

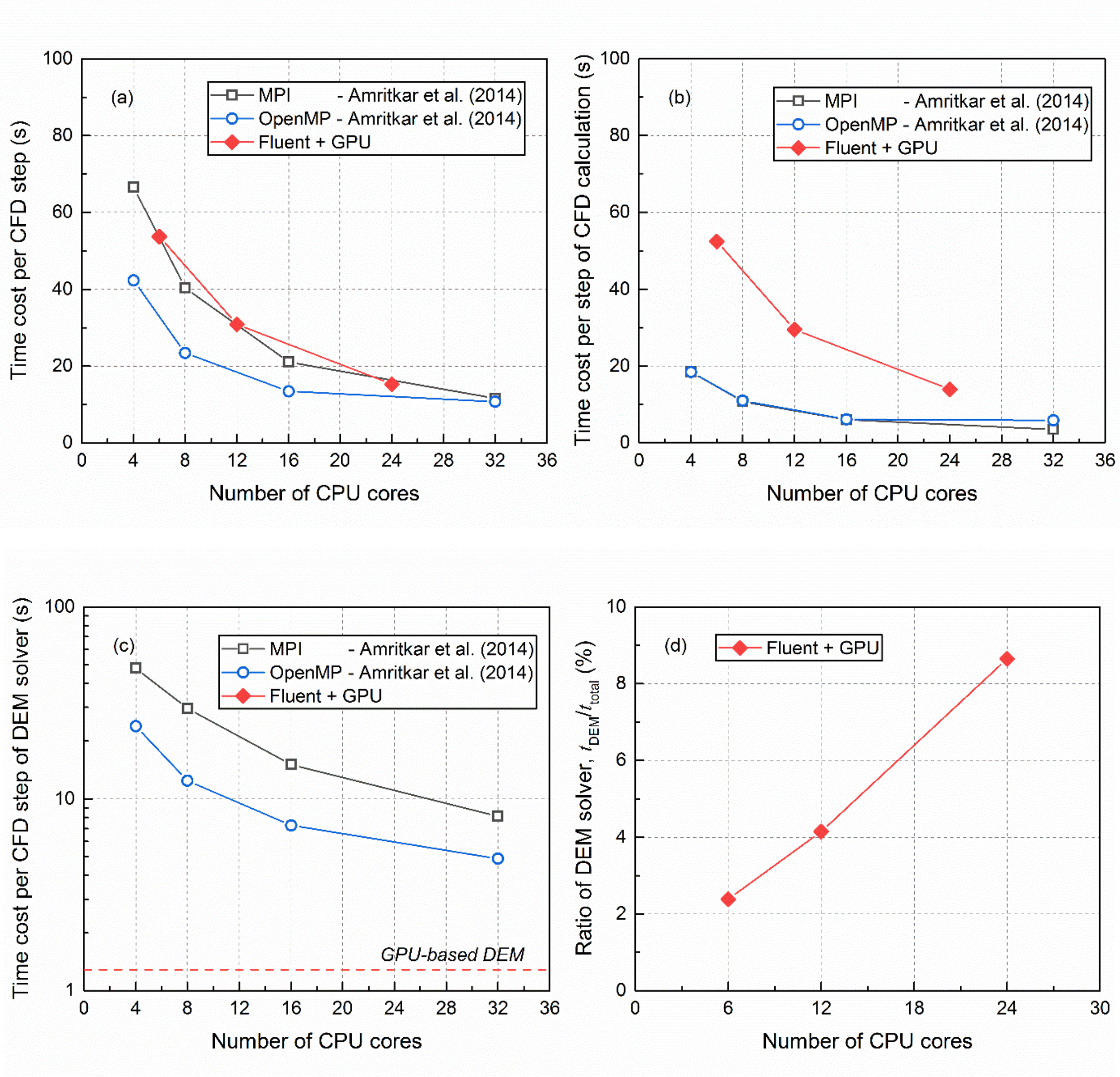


Fig. 7. Average computational cost per CFD step of (a) a coupled CFD-DEM simulation, (b) the CFD solver, (c) the DEM solver, including phase coupling and calculation of the particles, and (d) ratio of the computational cost of the DEM solver. The number of simulated particles is 1,327,104.

To evaluate scalability of the coupling scheme, we vary the number of particles, between 1 million and 8 million. The simulation setups are kept the same as that of the previous case of 1.3 million particles, except that 24 CPU cores were used for all the CFD calculations. Results in Fig. 8(a) indicates that there is a linear scalability of the average computational cost per CFD time step of a coupled simulation to the number of particles. For the major components of a coupled simulation, the majority of the computational time is due to CFD calculation (>60% for 8 million particles) followed by phase coupling (>5% for 1 million particles). The computational cost of particle phase is relative low, which only accounts for less than 1% of the total cost. The ratio of the time cost increases with particle number for the calculations of particle phase (Fig. 8(b)) and the phase coupling (Fig. 8(c)) but decreases for the calculation of fluid phase (Fig. 8(d)), suggesting that the limitation of the present coupling scheme lies in the CFD calculation.

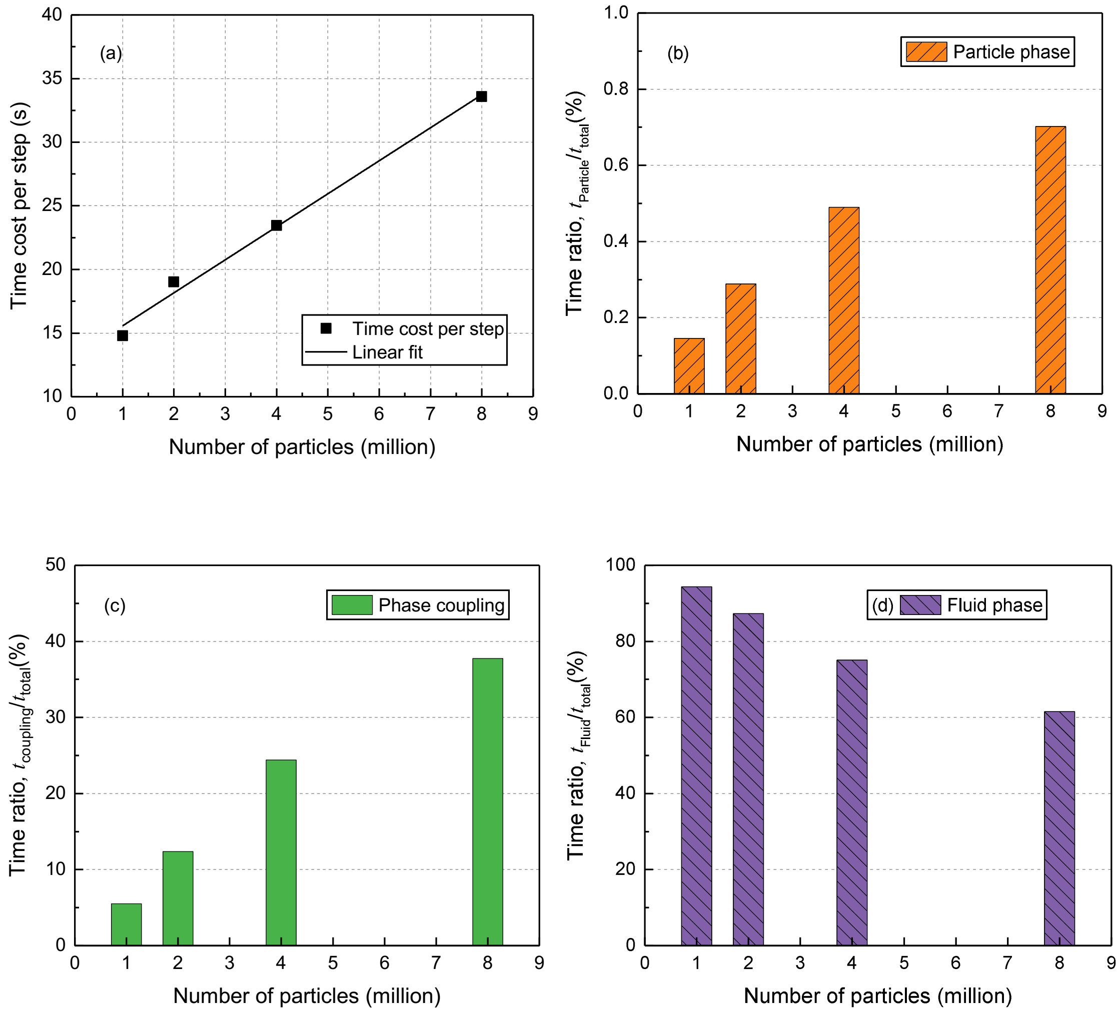
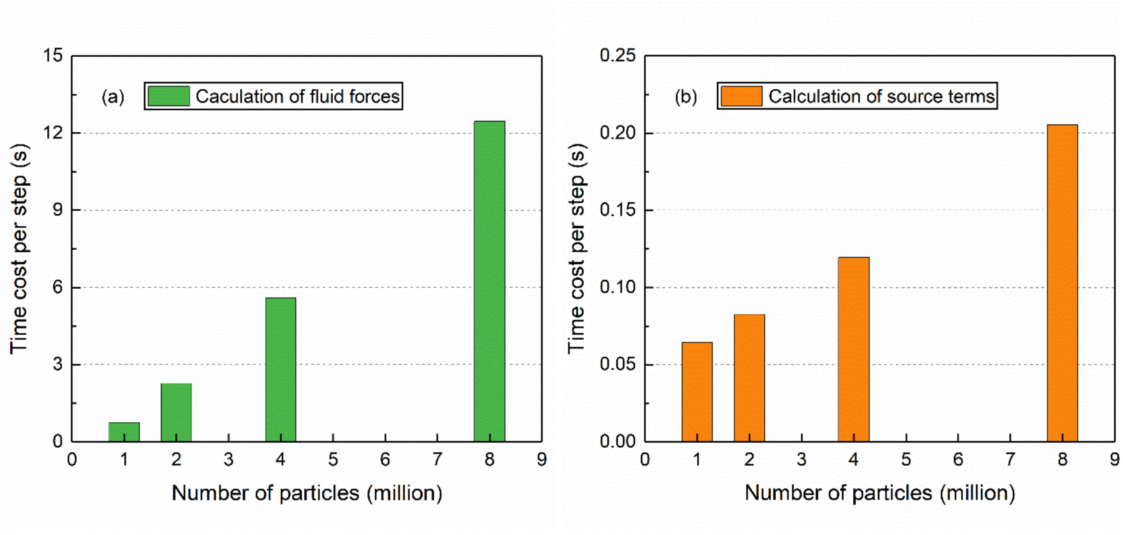


Fig. 8. Dependences of (a) average time per CFD step, computational ratio of (b) particle phase, (c) phase coupling and (d) fluid phase in a coupled CFD-DEM simulation of gas-solid fluidization on the number of particles.

There are two major calculations involved in phase coupling: fluid forces acting on solid particles and source terms for CFD cells. Both calculations are parallelized on GPU, with each CUDA thread assigned either to each particle or to each CFD cell, respectively. As shown in Fig. 9(a), the computational cost fluid-particle interaction forces scales almost linearly with the particle number, which accounts for a larger proportion of time compared to that of the source terms (Fig.

9(b)). This can be attributed to two reasons. First, the calculation of local void fraction at the particle position is issued as a separate kernel before calculating the fluid forces, as it is required in the calculation of the drag force. Secondly, it involves particle mapping, searching of neighboring particles and neighboring CFD cells. To calculate the local void fraction, each particle is mapped on the particle grid, with three layers of surrounding cells being searched. While for the fluid forces, solid particles are mapped onto the fluid grid and only one layer of surrounding cells are searched.

Fig. 9(c) and 9(d) shows the computational time of the main components of the DEM calculation in simulating gas-solid fluidization. The DEM calculations are grouped into four parts: reconstruction of the neighbor list (including update the contact history), particle-particle interaction, particle-wall interaction and time integration of particle movement (i.e. updating the particle positions and velocities). Creation of the neighbor list is the most time-consuming step and also scales almost linearly with the particle number. This is followed by the particle-particle interaction, the time cost of which also increases with the number of particles. However, it should be highlighted that updating the neighbor list is only triggered intermittently by the maximum cumulative particle displacement instead of updating at every DEM timestep. It includes creation of the neighbor list and update of the contact history for the calculation of tangential contact forces. The costs of resolving particle-wall interaction and updating particle positions are less than 0.1% of the total cost of a coupled simulation.



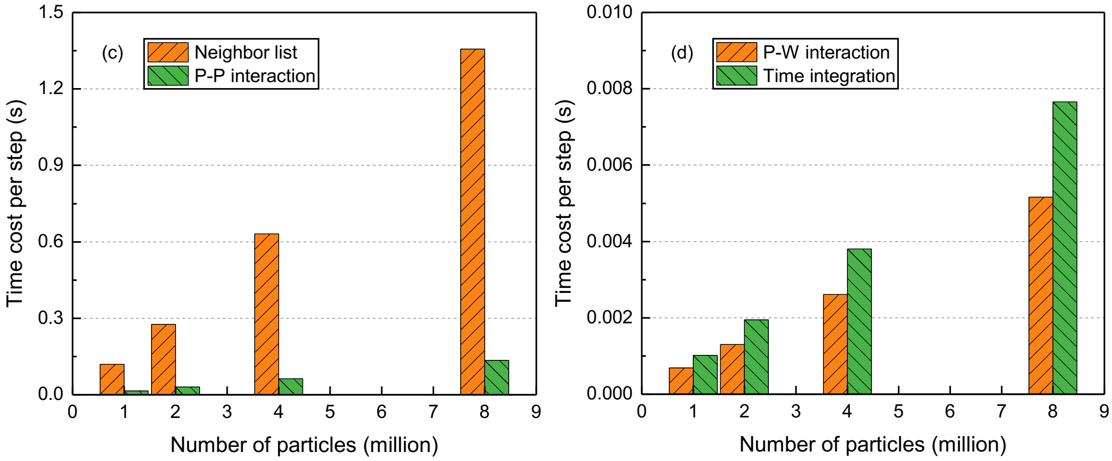


Fig. 9. Averaged time cost per step for the calculations of (a) fluid forces acting on each particle, (b) source terms for each CFD cells, (c) reconstruction of neighbor list and particle-particle interaction and (d) particle-wall interaction and time integration.

3.3. Model application: gas-solid cyclone

To demonstrate the general applicability of the developed coupling scheme, it is further applied to the simulations of gas-solid flows in a cyclone separator. Cyclone separators are often used to separate solid particles from gas in a variety of industrially important processes, such as dryers, circulating fluidized bed combustion and fluid catalytic crackers to recovery catalyst from gases. The gas enters the cyclone from a tangential inlet, generating a strong swirling gas motion. Particles are driven to the wall by the swirling gas flow and leave from the bottom while gas exists from the top through the vortex finder. The resulting flow dynamics is highly complex, making the numerical modelling very challenging. This is especially the case when dealing with high solid loadings. The associated high computational cost often limits the number of particles that can be simulated using a conventional coupled CFD-DEM method. For example, the previous study on the same geometry is limited to a mass ratio of solids to gas to less than 2.5 and particle size to 2mm (Chu et al., 2011). Some approaches, such as coarse-graining/parcel-particle, have been applied to accelerate the simulation but at a cost of reduced levels of accuracy, applicability and physical details that can be extracted from the simulations (Chu et al., 2016).

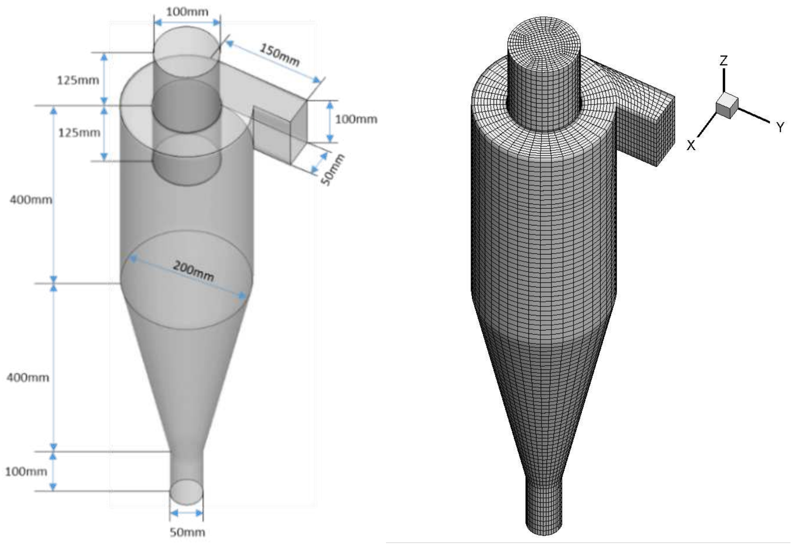


Fig. 10. (a) Geometry and (b) CFD grid of the cyclone separator.

The cyclone simulated in this work is the same as that used by (Wang et al., 2006) and (Chu et al., 2011). The geometry of the cyclone is shown in Fig. 10(a). The gas phase is solved by the ANSYS/Fluent, with the turbulence modelled by the Reynolds stress model and no-slip boundary condition applied to the solid walls. The particle phase is modelling using DEM. The cyclone is meshed into 87546 hexahedral CFD cells. The pressure outlet conditions are applied to both the top and the bottom exits, with an ambient atmospheric pressure of 1 atm. To test the performance of the proposed coupling scheme, the solid to gas mass ratio is varied while keeping the inlet gas velocity constant as 20 m/s. Up to 800,000 particles were simulated, with the parameters listed in Table 2. For convenience, the wall has the same properties as the particle but with an infinitely diameter. A physical time of 2s is simulated.

Table 2 Parameters used in the simulations.

Parameter	**Value**
Particle diameter (mm)	1.0
Particle density (kg/m^3)	2500
Young's modulus (Pa)	1.0 × 10^7
Poisson's ratio	0.3
Sliding friction coefficient	0.3
Rolling friction coefficient	0.002
Restitution coefficient	0.3
Particle inlet velocity (m/s)	3.0
Gas density (kg/m^3)	1.225
Gas inlet velocity (m/s)	20
Gas viscosity (kg/m/s)	1.789 × 10^{-5}
Solid loading ratio (kg/kg)	2.0, 4.0, 6.0, 8.0
Number of particles (× 10^3)	200, 400, 600, 800

Fig. 11 shows the flow pattern of particles in the cyclone with a solid loading ratio of 2.0. Particles are colored by the vertical velocity. It can be seen that the strand phenomenon is well captured by the simulation. The width of the strand shows little change in the cylindrical part but gradually expands after entering the conical region. A macroscopic steady state is reached after 0.7s. This is earlier than that observed for a particle size of 2mm. The descending velocity first increase during the cylindrical section and gradually decrease in the conical section, due to the supporting role of the conical wall. The particle flow patterns under different solid loading are shown in Fig. 12, where the strand pattern is more pronounced under higher loading ratio. The descending angle of the strand is found to increase with the solid loading ratio. The width of the strands increases with the solid loading ratio, consistent with that observed in the previous study (Chu et al., 2011).

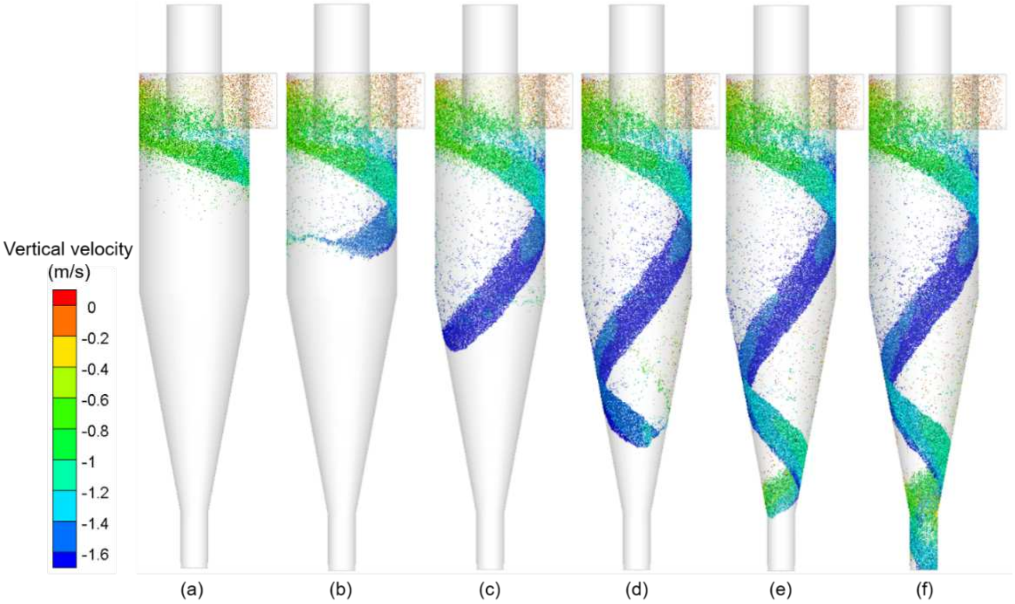


Fig. 11. Dynamic flow patterns of particles in the cyclone with a solid loading ratio of 2, in which (a) t=0.2s, (b) t = 0.3s, (c) t = 0.4s, (d) t = 0.5s, (e) t = 0.6s and (f) t = 0.7s. Particles are colored by the vertical velocity.

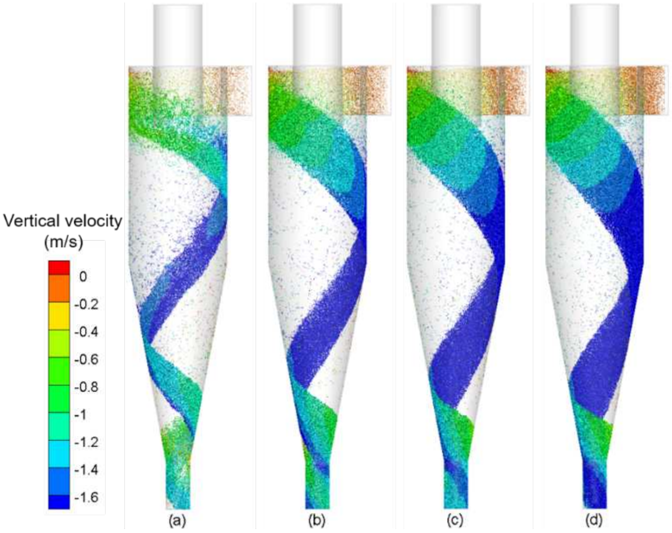


Fig. 12. Flow patterns of particles in the cyclone under different solid loading ratios: (a) MR=2, N_P=200,000, (b) MR=4, N_P=400,000, (c) MR=6, N_P=600,000 and (d) MR=8, N_P=800,000, where the MR stands for the solid loading ratio and N_P is the maximum number particles simulated in each case. Particles are colored by the vertical velocity.

Fig. 13(a) shows the variation of time cost in every 50 CFD time steps during the simulations. The surge in the time cost is due to the expense of saving both case and data files every 0.1s. Except the time used for data saving, the time cost increases almost linearly with time. This is due to the continuous feeding of particles into the system. With increasing solid loading ratio, the time cost also increases. Fig. 13(b) shows the ratio of time spent on the CFD calculation with different ratios of solid loading. As expected, the fraction of time used for CFD calculation decreases with the increase of solid loading due to increased number of particles in the system. A linear relationship can be found between the time cost ratio and the solid loading ratio, indicating a good scalability of the coupled simulations.

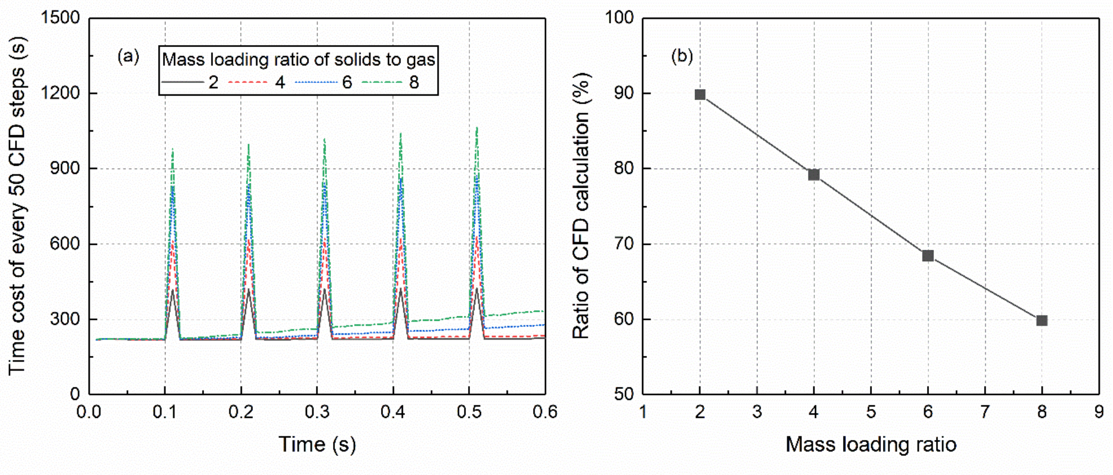


Fig. 13 (a) Average time cost of every 50 CFD time steps and (b) the ratio of time cost of CFD calculations.

4. Conclusion

In the present study, an efficient CPU-GPU cross-platform coupling approach and its implementation are presented to accelerate CFD-DEM simulations of particle-fluid flows. The particulate phase is simulated by a standalone, in-house GPU-based DEM solver while the fluid phase calculations are performed by a CFD solver, ANSYS Fluent, thus enabling the best use of different hardware architectures. On the CFD solver, a data socket is established and connected with the GPU-based DEM solver for data communication. The UDFs built for the CFD solver are only responsible for exchanging cell-based information with the DEM solver and distributing this information to computing nodes without conducting any calculations. Hence, the amount of data being transferred is independent of the number of particles in the system. On the DEM side, all the major components of the DEM calculations are parallelized on the GPU. A dual grid approach is proposed to aid the data mapping between Eulerian and Lagrangian properties, namely, the calculations of fluid forces acting on particle, void fraction and momentum source terms for each CFD cell. The following conclusions can be drawn,

- The model is validated on a small-scale fluidized bed, showing good agreement with experiments on the profiles of particle velocity and void fraction at different heights.
- The computational performance is evaluated on a larger fluidized bed by comparing with literature results where MPI and OpenMP were used to parallelize the coupled simulation. For a fluidization of 1.3 million particles, the GPU-based DEM provides 3.81 times speedup versus OpenMP and 6.31 times speedup versus MPI on 32 CPU cores.
- A linear scalability was obtained between the averaged calculation time and the number of particles. With the proposed coupling scheme, DEM calculation is no longer the computational bottleneck of a coupled simulation even for a fluidization of 8 million particles.
- Practical applicability to handle complex geometrical domains is demonstrated by simulations of gas-solid flows in a cyclone separator with varying ratios of solid loading. The experimentally observed strand phenomena can be well predicted. The continuous phase CFD calculations remained the rate determining factor controlling simulation time. The computational cost increases with the number of particles; the CFD calculation scales linearly with the solid loading ratio.

The proposed cross-platform approach has demonstrated the strong potential to contribute to an improved understanding of industrially important processes. It is worthwhile to mention that the present coupling scheme is generally applicable to any CFD solvers, although the ANSYS/Fluent is used here as an example for the CFD calculation.

Acknowledgments

The authors would like to thank the European Commission for supporting this work as part of the research project "Intensified by Design® for the intensification of processes involving solids handling" under the H2020 SPIRE programme (SPIRE-08-2015-680565) and the EPSRC-UK MAPP Future Manufacturing Hub (EP/P006566/1, www.mapp.ac.uk). This work was undertaken on ARC3, part of the high-performance computing facilities at the University of Leeds, UK.

Reference

Amritkar, A., Deb, S., Tafti, D., 2014. Efficient parallel CFD-DEM simulations using OpenMP. Journal of Computational Physics 256, 501-519.

Anderson, T.B., Jackson, R., 1967. A Fluid Mechanical Description of Fluidized Beds. Industrial & Engineering Chemistry Fundamentals 6, 527-&.

Chimakurthi, S.K., Reuss, S., Tooley, M., Scampoli, S., 2018. ANSYS Workbench System Coupling: a state-of-the-art computational framework for analyzing multiphysics problems. Engineering with Computers 34, 385-411.

Chu, K., Wang, B., Yu, A., Vince, A., Barnett, G., Barnett, P.J.M.E., 2009a. CFD–DEM study of the effect of particle density distribution on the multiphase flow and performance of dense medium cyclone. 22, 893-909.

Chu, K., Wang, B., Yu, A., Vince, A.J.P.T., 2009b. CFD-DEM modelling of multiphase flow in dense medium cyclones. 193, 235-247.

Chu, K.W., Chen, J., Yu, A.B., 2016. Applicability of a coarse-grained CFD-DEM model on dense medium cyclone. Minerals Engineering 90, 43-54.

Chu, K.W., Wang, B., Xu, D.L., Chen, Y.X., Yu, A.B., 2011. CFD-DEM simulation of the gas-solid flow in a cyclone separator. Chemical Engineering Science 66, 834-847.

Cleary, P.W., Sinnott, M., Morrison, R.J.M.e., 2006. Prediction of slurry transport in SAG mills using SPH fluid flow in a dynamic DEM based porous media. 19, 1517-1527.

Darmana, D., Deen, N.G., Kuipers, J.A.M., 2006. Parallelization of an Euler-Lagrange model using mixed domain decomposition and a mirror domain technique: Application to dispersed gas-liquid two-phase flow. Journal of Computational Physics 220, 216-248.

Ergun, S., 1952. Fluid Flow through Packed Columns. Chemical Engineering Progress 48, 89-94.

Gan, J.Q., Zhou, Z.Y., Yu, A.B., 2016. A GPU-based DEM approach for modelling of particulate systems. Powder Technology 301, 1172-1182.

Garg, R., Galvin, J., Li, T., Pannala, S.J.P.T., 2012. Open-source MFIX-DEM software for gas–solids flows: Part I—Verification studies. 220, 122-137.

Gopalakrishnan, P., Tafti, D., 2013. Development of parallel DEM for the open source code MFIX. Powder Technology 235, 33-41.

Govender, N., Wilke, D.N., Wu, C.Y., Rajamani, R., Khinast, J., Glasser, B.J., 2018. Large-scale GPU based DEM modeling of mixing using irregularly shaped particles. Advanced Powder Technology 29, 2476-2490.

Han, Y., Cundall, P.A.J.I.J.f.N., Geomechanics, A.M.i., 2013. LBM–DEM modeling of fluid–solid interaction in porous media. 37, 1391-1407.

He, Y., Bayly, A.E., Hassanpour, A., Muller, F., Wu, K., Yang, D.M., 2018a. A GPU-based coupled SPH-DEM method for particle-fluid flow with free surfaces. Powder Technology 338, 548-562.

He, Y., Evans, T.J., Shen, Y.S., Yu, A.B., Yang, R.Y., 2018b. Discrete modelling of the compaction of non-spherical particles using a multi-sphere approach. Minerals Engineering 117, 108-116.

He, Y., Evans, T.J., Yu, A.B., Yang, R.Y., 2018c. A GPU-based DEM for modelling large scale powder compaction with wide size distributions. Powder Technology 333, 219-228.

He, Y., Wang, Z., Evans, T.J., Yu, A.B., Yang, R.Y., 2015. DEM study of the mechanical strength of iron ore compacts. International Journal of Mineral Processing 142, 73-81.

Hoberock, J., Bell, N., Thrust: a C++ Template Libaray for CUDA. available from: https://github.com/thrust/thrust.

Hoomans, B.P.B., Kuipers, J.A.M., Briels, W.J., vanSwaaij, W.P.M., 1996. Discrete particle simulation of bubble and slug formation in a two-dimensional gas-fluidised bed: A hard-sphere approach. Chemical Engineering Science 51, 99-118.

Jajcevic, D., Siegmann, E., Radeke, C., Khinast, J.G., 2013. Large-scale CFD-DEM simulations of fluidized granular systems. Chemical Engineering Science 98, 298-310.

Kafui, D.K., Johnson, S., Thornton, C., Seville, J.P.K., 2011. Parallelization of a Lagrangian-Eulerian DEM/CFD code for application to fluidized beds. Powder Technology 207, 270-278.

Kloss, C., Goniva, C., Aichinger, G., Pirker, S., 2009a. Comprehensive DEM-DPM-CFD simulations-model synthesis, experimental validation and scalability, Proceedings of the Seventh International Conference on CFD in the Minerals and Process Industries, CSIRO, Melbourne, Australia.

Kloss, C., Goniva, C., Aichinger, G., Pirker, S., 2009b. Comprehensive DEM-DPM-CFD simulations - model synthesis, experimental validation and scalability. Proceedings Seventh International Conference on CFD in the Minerals and Process Industries, CSIRO, Melbourne, Australia, 9-11 December.

Kloss, C., Goniva, C., Hager, A., Amberger, S., Pirker, S., 2012a. Models, algorithms and validation for opensource DEM and CFD-DEM. Progress in Computational Fluid Dynamics 12, 140-152.

Kloss, C., Goniva, C., Hager, A., Amberger, S., Pirker, S.J.P.i.C.F.D., an International Journal, 2012b. Models, algorithms and validation for opensource DEM and CFD–DEM. 12, 140-152.

Kuang, S., Li, K., Zou, R., Pan, R., Yu, A.J.C.E.S., 2013. Application of periodic boundary conditions to CFD-DEM simulation of gas–solid flow in pneumatic conveying. 93, 214-228.

Kuang, S.B., Chu, K.W., Yu, A.B., Zou, Z.S., Feng, Y.Q., 2008. Computational investigation of horizontal slug flow in pneumatic conveying. Industrial & Engineering Chemistry Research 47, 470-480.

Li, T., Garg, R., Galvin, J., Pannala, S.J.P.T., 2012a. Open-source MFIX-DEM software for gas-solids flows: part II—validation studies. 220, 138-150.

Li, T., Gopalakrishnan, P., Garg, R., Shahnam, M.J.P., 2012b. CFD–DEM study of effect of bed thickness for bubbling fluidized beds. 10, 532-541.

Li, T.W., Garg, R., Galvin, J., Pannala, S., 2012c. Open-source MFIX-DEM software for gas-solids flows: Part II - Validation studies. Powder Technology 220, 138-150.

Liu, P.Y., Hrenya, C.M., 2014. Challenges of DEM: I. Competing bottlenecks in parallelization of gas-solid flows. Powder Technology 264, 620-626.

Mindlin, R.D., Deresiewicz, H., 1953. Elastic Spheres in Contact under Varying Oblique Forces. Journal of Applied Mechanics-Transactions of the Asme 20, 327-344.

Muller, C.R., Scott, S.A., Holland, D.J., Clarke, B.C., Sederman, A.J., Dennis, J.S., Gladden, L.F., 2009. Validation of a discrete element model using magnetic resonance measurements. Particuology 7, 297-306.

Norouzi, H.R., Zarghami, R., Mostoufi, N., 2017. New hybrid CPU-GPU solver for CFD-DEM simulation of fluidized beds. Powder Technology 316, 233-244.

Peng, Z.B., Doroodchi, E., Luo, C.M., Moghtaderi, B., 2014. Influence of void fraction calculation on fidelity of CFD-DEM simulation of gas-solid bubbling fluidized beds. Aiche Journal 60, 2000-2018.

Rabenseifner, R., Hager, G., Jost, G., 2009. Hybrid MPI/OpenMP parallel programming on clusters of multi-core SMP nodes, Parallel, Distributed and Network-based Processing, 2009 17th Euromicro International Conference on. IEEE, pp. 427-436.

Radeke, C.A., Glasser, B.J., Khinast, J.G., 2010. Large-scale powder mixer simulations using massively parallel GPU architectures. Chemical Engineering Science 65, 6435-6442.

Tafti, D.K., 2001. Genidlest - A scalable parallel computational tool for simulating complex turbulent flows. in: ASME-IMECE, American Society of Mechanical Engineering, New York, NY 10016-5990, United States., 347-356.

Tong, Z., Zheng, B., Yang, R., Yu, A., Chan, H.J.P.t., 2013. CFD-DEM investigation of the dispersion mechanisms in commercial dry powder inhalers. 240, 19-24.

Tsuji, T., Yabumoto, K., Tanaka, T., 2008. Spontaneous structures in three-dimensional bubbling gas-fluidized bed by parallel DEM-CFD coupling simulation. Powder Technology 184, 132-140.

Tsuji, Y., Kawaguchi, T., Tanaka, T.J.P.t., 1993. Discrete particle simulation of two-dimensional fluidized bed. 77, 79-87.

Varas, A.E.C., Peters, E.A.J.F., 2017. CFD-DEM simulations and experimental validation of clustering phenomena and riser hydrodynamics. Chemical Engineering Science 169, 246-258.

Wang, B., Xu, D.L., Chu, K.W., Yu, A.B., 2006. Numerical study of gas-solid flow in a cyclone separator. Applied Mathematical Modelling 30, 1326-1342.

Washington, D.W., Meegoda, J.N., 2003. Micro-mechanical simulation of geotechnical problems using massively parallel computers. International Journal for Numerical and Analytical Methods in Geomechanics 27, 1227-1234.

Washino, K., Tan, H., Hounslow, M., Salman, A.J.C.E.S., 2013. A new capillary force model implemented in micro-scale CFD–DEM coupling for wet granulation. 93, 197-205.

Wen, C.Y., Yu, Y.H., 1966. Mechanics of fluidization. Chemical Engineering Progress Symposium Series. 62, 100-111.

Wu, C.L., Ayeni, O., Berrouk, A.S., Nandakumar, K., 2014. Parallel algorithms for CFD-DEM modeling of dense particulate flows. Chemical Engineering Science 118, 221-244.

Xiao, H., Sun, J., 2011. Algorithms in a Robust Hybrid CFD-DEM Solver for Particle-Laden Flows. Communications in Computational Physics 9, 297-323.

Xiong, Q., Madadi-Kandjani, E., Lorenzini, G.J.C.M., Thermodynamics, 2014. A LBM–DEM solver for fast discrete particle simulation of particle–fluid flows. 26, 907-917.

Xu, B.H., Yu, A.B., 1997. Numerical simulation of the gas-solid flow in a fluidized bed by combining discrete particle method with computational fluid dynamics. Chemical Engineering Science 52, 2785-2809.

Xu, J., Qi, H.B., Fang, X.J., Lu, L.Q., Ge, W., Wang, X.W., Xu, M., Chen, F.G., He, X.F., Li, J.H., 2011. Quasi-real-time simulation of rotating drum using discrete element method with parallel GPU computing. Particuology 9, 446-450.

Xu, M., Chen, F.G., Liu, X.H., Ge, W., Li, J.H., 2012. Discrete particle simulation of gas-solid two-phase flows with multi-scale CPU-GPU hybrid computation. Chemical Engineering Journal 207, 746-757.

Yakubov, S., Cankurt, B., Abdel-Maksoud, M., Rung, T., 2013. Hybrid MPI/OpenMP parallelization of an Euler-Lagrange approach to cavitation modelling. Computers & Fluids 80, 365-371.

Yang, S.L., Luo, K., Zhang, K., Qiu, K.Z., Fan, J.R., 2015. Numerical study of a lab-scale double slot-rectangular spouted bed with the parallel CFD-DEM coupling approach. Powder Technology 272, 85-99.

Zhu, H.P., Zhou, Z.Y., Yang, R.Y., Yu, A.B., 2007. Discrete particle simulation of particulate systems: Theoretical developments. Chemical Engineering Science 62, 3378-3396.