# A parallel unidirectional coupled DEM-PBM model for the efficient solution and simulation of computationally intensive systems.

Chaitanya Sampat[a,*], Franklin Bettencourt[a,*], Yukteshwar Baranwal[a], Ioannis Paraskevakos[b], Anik Chaturbedi[a], Subhodh Karkala[a], Shantenu Jha[b], Rohit Ramachandran[a], Marianthi Ierapetritou[a,**]

[a] *Department of Chemical and Biochemical Engineering, Rutgers, The State University of New Jersey, Piscataway, NJ, USA-08854*

[b] *Electrical and Computer Engineering, Rutgers, The State University of New Jersey, Piscataway, NJ, USA-08854*

**Abstract**

The accurate modeling of the physics underlying particulate processes is complicated and requires significant computational capabilities to solve using particle-based models. In this work, a unidirectional multi-scale approach was used to model the high shear wet granulation process. A multi-dimensional population balance model (PBM) was developed with a mechanistic kernel, which in turn obtained collision data from the discrete element modeling (DEM) simulation. The PBM was parallelized using a hybrid OpenMP+MPI approach. The DEM simulations were performed using LIGGGHTS, which was parallelized using MPI. Speedups of approximately 14 were obtained for the PBM simulations and approximately 12 for the DEM simulations. The uni-directional coupling of DEM to PBM was performed using middle-ware components (RADICAL-Pilot) that did not require modifications of the DEM or PBM codes, yet supported flexible execution on high-performance platforms. Results demonstrate scaling from 1 to 128 cores for the PBM and up to 256 cores for the DEM. The proposed method, implementations and middle-ware enable the modeling of high shear wet granulation process faster than existing approaches in literature.

*Keywords:* Population balance model, Granulation, Discrete element method, MPI and OpenMP, Pharmaceutical process design

## 1. Introduction & Objectives

Half of all industrial chemical production is typically performed via the processing of particulate systems (Seville et al., 2012). These processes also account for approximate 70% of the industrially manufactured products like detergents, aerosols, fertilizers, and pharmaceuticals (Litster, 2016). They are widely popular as particulate products have advantages over liquid formulations such as better chemical and physical stability and reduced transportation cost. Despite the prevalence of these particulate processes, the underlying physics of these processes is poorly understood (Rogers et al., 2013). As a result industries that rely on these processes, especially pharmaceuticals, have to use expensive heuristic studies and inefficient operational protocols with high recycle ratios to meet strict regulatory standards (Ramachandran et al., 2009). This can increase costs and delay

[*] Equal contribution by C. Sampat and F. Bettencourt

[**] Corresponding author

*Email address:* marianth@soe.rutgers.edu (Marianthi Ierapetritou)

the release of new products. Furthermore these processes so challenging to design is that there are none or few governing equations to accurately predict their behavior (Sen et al., 2012).

Particulate processes are defined by chaotic micro-scale phenomena that result from the many particle-particle interactions inside these systems. These small scale phenomenon develop into the complex bulk behavior of these processes. To successfully predict the bulk behavior of these systems, a model needs to capture the particle-particle interactions and emergent meso-scale phenomena. Discrete element method (DEM) (Cundall and Strack, 1979)simulations are employed to obtain particle-level data, which helps describe the bulk properties of the particulate system accurately (Hancock and Ketterhagen, 2011). DEM uses Newton's equations of motion to model the forces on each particle in the system and it's interactions with the system geometry and other particles. Since DEM involves large amounts of interactions and force computation, it usually takes a significant amount of time to simulate. Thus, there is a need for a more efficient simulation technique for these particulate particles. One of the common methods used in the industry is the population balance model (PBM), which is more computationally efficient but, lacks the microscopic detail of the DEM.

PBM takes into account the changes in internal or spatial particle properties.This model is also unable to incorporate design parameters such as equipment geometry. As a result, it is semi-mechanistic in nature, that it uses population averages and collisional probability to capture bulk behavior but still seeks to capture some of the micro-phenomena of particle-particle interactions using correlations or empirically developed kernels to approximate those interactions. Although PBM is computationally much faster than DEM, a detailed PBM can still take a significant amount of time to solve (Barrasso et al., 2013). Though highly detailed PBMs (Immanuel and Doyle, 2005)(Poon et al., 2008) do take into account various dimensions, populating, averaging and contain semi-mechanistic kernels, these PBMs still can have difficulties in capturing the micro-scale phenomena that are crucial to predicting accurate dynamics of particulate processes (Christofides et al., 2007).

Thus, to complement the computational and mechanistic advantages of the PBM and DEM technique respectively, they can be coupled to produce a model which is more accurate than a PBM and also computationally more efficient compared to DEM. The typical work flow of such a DEM-PBM coupled model involves using a short DEM simulation to capture the inter-particle interaction of the system, which is fed into the PBM. This helps PBM accurately simulate the bulk system behavior (Goldschmidt et al., 2003) (Reinhold and Briesen, 2012)(Barrasso et al., 2013). Despite the performance benefits of these coupled DEM-PBM models, the simulations times are not short enough for practical use. In recent years, High-Performance Computing (HPC) has been employed to speed up these DEM and PBM simulations (Gunawan et al., 2008) (Prakash et al., 2013a) (Bettencourt et al., 2017).

The objective of this work is to employ HPC systems to solve the developed DEM-PBM coupled model such that it can be used to predict behavior of the wet granulation process accurately and in a computationally efficient manner. Usually such simulations with current techniques take upto several days to complete. This model could be used to design a control system around the granulation process to help control the product quality, which is of prime importance to the pharmaceutical industry. In the course of development of the model, the DEM simulation were performed on LAMMPS Improved for General Granular and Granular Heat Transfer simulations (LIGGGHTS) (Kloss et al., 2012) to model the micro-mechanics of the Lödige CoriMix CM5 high shear granulator. A 4-Dimensional, PBM coupled with DEM was developed that is parallelized using hybrid techniques (Message Parsing Interface (MPI) + Open Multi-Processing (OMP)) to model the bulk processes occurring during the granulation process and is discussed in section 3.1.3. Section 3.3 describes unidirectional coupling of DEM and PBM using RADICAL Pilot, which is a framework developed in Python. This provides task level parallelization, to help develop an

accurate model which can be run quickly on high performance computing systems. The speed improvements obtained for the parallel code and further improvements that can be implemented are discussed in Section 4

## 2. Background & Related Work

### *2.1. Particulate processes*

A system of discrete species exist in these particulate processes, that undergo changes in average composition, size, or other pertinent properties. Granulation is one such particulate process which is commonly found in the pharmaceutical industry. In this process, fine powders are converted to larger granules using a liquid binder. The rate processes governing granulation are wetting and nucleation, consolidation and aggregation, and attrition and breakage (Iveson et al., 2001) (Cameron et al., 2005). As liquid is added to the fine powder, it forms a porous nuclei that can coalesce, deform and break (Barrasso et al., 2015). As there is an alteration in the properties of these nuclei, they can take up additional liquid or breakdown to a finer powder.To understand how a granulation processes will behave with different design and operating parameter settings and formulations, experimental studies are performed which is a method referred to as Quality-by-Testing (QbT). This methodology is time consuming and expensive. Thus, newer research is focused on the more economical Quality-by-Design (QbD) concept i.e, using mathematical models to design these processes in silico.

### *2.2. Modeling*

The paradigm shift of the pharmaceutical industry towards continuous manufacturing, emphasizes the need for a more accurate model. These models further help develop better control strategies for the process. Such controllers require computationally efficient and precise models.

#### *2.2.1. Discrete Element Modeling (DEM)*

Discrete Element Method is a simulation technique used to monitor the behavior of each particle as a separate entity. This method tracks movement of each particle within the space, records the collisions of each particle with the geometry as well as with each other and it is also subject to other force fields like gravity (Barrasso and Ramachandran, 2015). This model is based on the Newton's laws of motion as shown (Cundall and Hart, 1992):

$$m_i \frac{dv_i}{dt} = F_{i,net} \tag{1}$$

$$F_{i,net} = F_{i,coll} + F_{i,ext} \tag{2}$$

where $m_i$ is mass of the particle, $v_i$ represents velocity of the particle, and $F_{i,net}$ represents net force on the particle. Forces on the particle due to collisions and other external forces are represented by $F_{i,coll}$ and $F_{i,ext}$ respectively.

Using this method, the distance between each particle is calculated at every time step and if this distance between two particles is less than the sum of the radii (for spherical particles) a collision between the two particles is recorded. The tolerance for overlap is low in the normal as well as the tangential direction. Micro-scale DEM simulations are computationally demanding and simulations may take up to several days to replicate a few seconds of real time experiments. Many methods have been implemented to increase the speed of these simulations, such as scaling by increasing the size of the particles. These approximations are useful in understanding the physics of the system especially velocity profiles (Gantt and Gatzke, 2005) but are not directly applicable to process-level simulations.

These granular particles usually have a visco-elastic type of behavior, thus the particle-particle and particle-wall collisions are modeled using a spring-dashpot rheological model (Dosta et al., 2018). The Hertz-Mindlin contact model (Mindlin and Deresiewicz, 1953) has been used in this work and it has been verified in earlier studies (Gantt et al., 2006)(Hassanpour et al., 2013).

### *2.2.2. Population Balance Model (PBM)*

Population balance models (PBM) predict how groups of discrete entities will behave on a bulk scale due to certain effects acting on the population with respect to time (Ramkrishna and Singh, 2014). In the context of process engineering and granulation, PBMs are used to describe how the number densities, of different types of particles, in a granulator change as rate processes such as aggregation and breakage reshape particles (Barrasso et al., 2013). A general form of population balance model is shown :

$$\frac{\partial}{\partial t}F(\mathbf{v},\mathbf{x},t)+\frac{\partial}{\partial \mathbf{v}}[F(\mathbf{v},\mathbf{x},t)\frac{d\mathbf{v}}{dt}(\mathbf{v},\mathbf{x},t)]+\frac{\partial}{\partial \mathbf{x}}[F(\mathbf{v},\mathbf{x},t)\frac{d\mathbf{x}}{dt}(\mathbf{v},\mathbf{x},t)] \\ = \Re_{formation}(\mathbf{v},\mathbf{x},t)+\Re_{depletion}(\mathbf{v},\mathbf{x},t)+\dot{F}_{in}(\mathbf{v},\mathbf{x},t)-\dot{F}_{out}(\mathbf{v},\mathbf{x},t) \tag{3}$$

In Equation (3), $\mathbf{v}$ is a vector of internal coordinates. For modeling a granulation process $\mathbf{v}$ is commonly used to describe the solid, liquid, and gas content of each type of particle. The vector $\mathbf{x}$ represents external coordinates, usually spatial variance. $F$ represents the number of particles present inside the system, $\dot{F}_{in}$ and $\dot{F}_{out}$ is the rate of particles coming in and going out the system respectively. $\Re_{formation}$ and $\Re_{depletion}$ are the rate of formation and depletion due various phenomena occurring in granulation. For a granulation process this account for spatial variance in the particles as they flow along the granulator.

### *2.2.3. Coupled DEM-PBMs*

The use of multi-physics models has recently been adapted to understand the behavior of particle systems. These models help understand the physics of the system at various scales *i.e.* micro, meso and macro scale (Sen et al., 2014). Particle process dynamics have been inferred from coupling of various physics models *viz.* CFD, DEM and PBM. Earlier works from Sen et al. (2014) & Barrasso and Ramachandran (2015) have successfully predicted process dynamics of the granulation process using such multi-physics models.

Initially, Ingram and Cameron (2004) coupled PBM and DEM using two different multi-scale frameworks which focused on methods of integration and information exchange required between these two methods. Later efforts on coupling of PBM and DEM were unidirectional in nature, where the collision data was obtained from the DEM and then used it in PBM. Gantt et al. (2006) used the DEM data to build a mechanistic model for the PBM. (Bouffard et al., 2012) evaluated spatial transfer in a compartmentalised PBM using DEM data, while Goldschmidt et al. (2003) solved a PBM using DEM by replacing smaller particles as they successfully coalesce with larger particles. Reinhold and Briesen (2012) replaced a mechanistic aggregation kernel with an empirical kernel in order to prove that, in certain cases DEM simulations may not be necessary for the development of kernels. A hybrid model for one-way coupling has been reported for continuous mixing (Sen et al., 2012) (Sen and Ramachandran, 2013) and is discussed in Section 3.1.3.

## *2.3. Computer Architecture and Parallel Applications*

An HPC node usually has two processors and at least 64GBs of RAM. Each processor has at least 16 cores. Main memory or RAM is divided between processors. As a result, each processor

has direct access to the part of the memory it considers local. However accessing the rest of the memory introduces delays (Jin et al., 2011). Therefore, managing data locality is very important.

Since the numerous cores of a CPU can communicate directly through memory they can do so by using shared memory methods. This is commonly done using OMP which uses threads to parallelize execution. The cores then communicate implicitly, through memory. This also allows processes to share common data automatically reducing communication overhead and the overall memory footprint required for computation.

When many CPUs are being used in parallel they cannot communicate with each other directly through memory which results in much slower communications. This type of system where processes cannot communicate directly through memory is called a distributed memory system. MPI is commonly used to enable communication of the processes in a distributed memory system. MPI allows the processes to communicate using explicit message passing methods. MPI will operate each process as a discrete entity with its own private copy of each variable it needs for computation which is how it avoids memory retrieval slowdowns when using many CPUs. For optimal parallel performance a programmer needs to consider the architecture of the cluster they are using and the features of the applications they are using such as MPI and OMP (Adhianto and Chapman, 2007).

### *2.4. Previous works using HPC methods to solve PBMs and DEMs*

Various researchers have employed parallel computing methods to solve DEM and PBM simulations. Gunawan et al. (2008) used high-resolution finite volumes solution methods for the parallelization of their PBM. They performed load balancing effectively by decomposing the internal coordinates of their PBM. They achieved speed improvements up to 100 cores on one system size, but was not tested for models with more dimensions. Moreover, they mentioned that parallelization could be improved using shared memory processing. In previous work, Bettencourt et al. (2017) adopted a hybrid approach towards the parallelization of the PBM using both MPI and OMP. This hybrid parallelization helped achieve a speed improvement of about 98% using 128 cores over the serial code. Prakash et al. (2013a) & Prakash et al. (2013b) used the inbuilt Parallel Computation Toolbox (PCT) in MATLAB (MathWorks$^{TM}$ Documentation, 2017b) to parallelize their PBM using less than 16 cores, but faced short comings due to the overhead of MATLAB software and therefore could not achieve the same level of performance as a program written in C/C++ or FORTRAN.

LIGGGHTS (Kloss et al., 2012), an open-source software used to perform DEM simulations has native support for MPI for parallelizing the simulation by static decomposition which partitions space such that the area of communication between the the MPI processes is minimized. Kačianauskas et al. (2010) used load balancing methods similar to a static decomposition and observed that this works well for a mono-dispersed system but the computational effort increases for simulations for poly-dispersed material. Gopalakrishnan and Tafti (2013) also reported a speed improvement and a parallel efficiency of about 81% in their CFD-DEM simulation. LIGGGHTS could not take advantage of shared memory interfaces since it did not support OMP. Berger et al. (2015) implemented hybrid parallelization methods for the particle-particle interaction and the particle-wall interaction modules in LIGGGHTS and also used the Zoltan library (Boman et al., 2012) developed by Sandia National Laboratories for dynamic load balancing. They achieved a speed improvement of about 44% for simulations performed on higher number of cores, but there was no significant speed improvement for smaller core counts.

### *2.5. Pilot abstraction and RADICAL-Pilot (RP)*

A primary challenge faced is the scalable execution of multiple (often two, but possibly more) heterogeneous simulations that need to run independently but have a need to communicate and

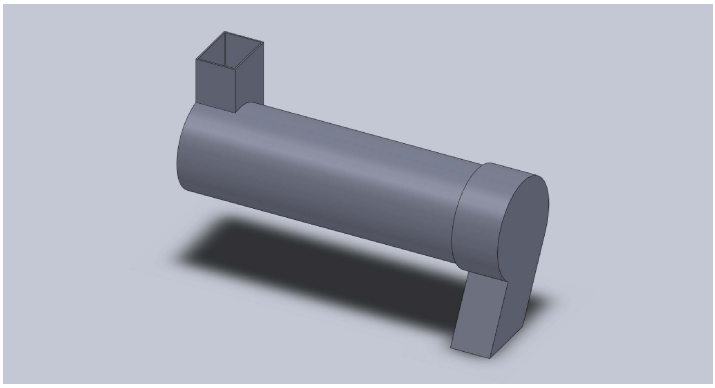


**Figure 1:** The isometric view of the Lödige CoriMix CM5 continuous high shear granulator casing.

exchange information. Traditionally each simulation is submitted as an individual job, but that invariably leads to a situation where each simulation gets through the batch-queue systems independent of the other. So although the first-through-the-queue is ready to run, it stalls fairly soon waiting for the other simulation to make it through the queue. On the other hand MPI capabilities can be used to execute both simulations as part of a single multi-node job. Whereas the former method suffers from unpredictable queue time for each job, the latter is suitable to execute tasks that are homogeneous and have no dependencies.

The Pilot abstraction (Luckow et al., 2012) solves these issues. The Pilot abstraction (i) uses a container-job as a placeholder to acquire resources, and (ii) decouples the initial resource acquisition from task-to-resource assignment. Once the Pilot (container-job) has acquired the resources, it can be populated with computational tasks. This functionality allows all tasks to be executed directly on the resources, without being queued individually. Thus, this approach can support the requirements of task-level parallelism and high-throughput as needed by science drivers.

RADICAL-Pilot is an implementation of the Pilot abstraction in Python, engineered to support scalable and efficient launching of heterogeneous tasks across different platforms.

## 3. Methods

### *3.1. Simulation Setup*

#### *3.1.1. Geometry and Meshing for DEM*

In this work, the Lödige CoriMix CM5 continuous high shear granulator has been studied. Its geometry was developed using the SolidWorks$^{TM}$ (Dassault Systèmes) software. The granulator consists of a high speed rotating element enclosed within a horizontal cylindrical casing. The casing (shown in Figure 1) consists of a cylinder with diameter of 120 mm at inlet and 130 mm at outlet and having a total length of 440 mm. A vertical inlet port is provided at one end of the casing and an angled outlet port is provided at the larger end of the case.

The impeller consists of a cylindrical shaft of length 370 mm and diameter 68 mm with four flattened sides 15 mm wide running along the axis. The blades are placed on these flattened sides as shown in Figure 2. There are three different blade elements on the shaft (Figure 3). At the granulator inlet, there are four paddle shaped feed elements following which there are twenty tear drop shaped shearing elements and finally four trapezoidal blades near the exit. All these elements are placed in a spiral configuration.

After the geometry was built in SolidWorks$^{TM}$ (Dassault Systèmes) the shell and impeller were exported as stereo-lithography (STL) files. The coarsest output option was used to keep the STL

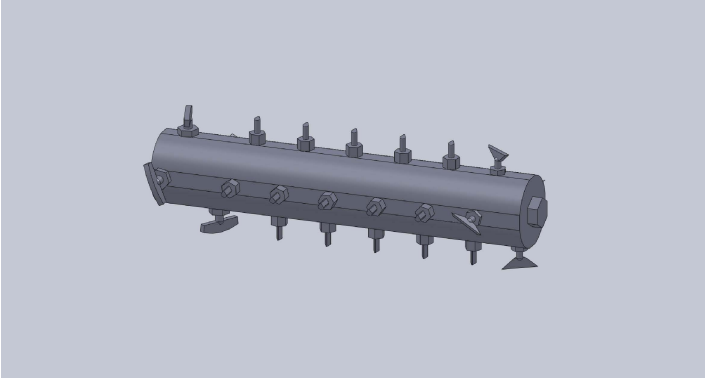
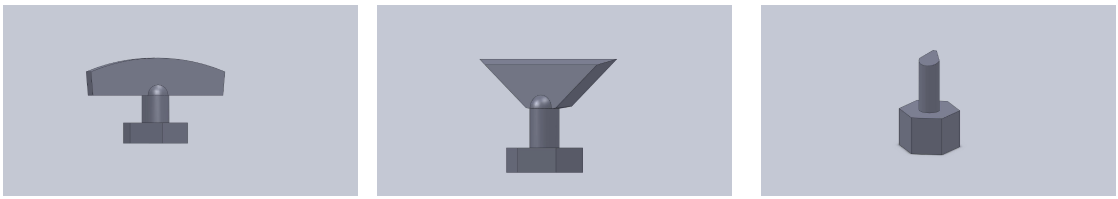


**Figure 2:** This shows the isometric view of the impeller inside the Lödige CoriMix CM5 continuous high shear granulator casing.



**(a)** Feed element **(b)** Exit element **(c)** Shear element

**Figure 3:** Components (a) and (b) help in the forward movement of the particles while (c) aids to direct the particles to the wall inside the Lödige CoriMix CM5 continuous high shear granulator.

files small for faster computation times. The origin of the geometry was not preserved while saving the STL files since it needs to be aligned in LIGGGHTS as per the process conditions. Meshlab was used to align the STL files for importing. No mesh treatments were used on the STLs. The meshes were then imported into LIGGGHTS using the write command in serial. The write exclusion list command was used and this exclusion list file is then used in the simulation to skip the highly skewed elements during the simulation.

*3.1.2. DEM Software and Input Setup*

LIGGGHTS v3.60 developed by DCS computing was used for all the DEM simulations performed in this study. Source code files were modified to obtain particle-particle collisions. This version of LIGGGHTS was compiled using the mvapich (mvapich2 v2.1) and intel (intel v15.0.2) compilers with the -O3 optimization option as well as an option to side load the process to the Xeon phi co-processors was added. The initial timing studies were performed on Stampede supercomputer located at TACC, University of Texas, Austin. The hardware configuration of each node consists of 2 8-core Intel Xeon E5-2680 processors based on the Sandy Bridge architecture, 32 GB of memory with QPI interconnects.

The DEM simulation in LIGGGHTS were setup using an input script which defines physical parameters of the particles, importing of the geometry, particle insertion commands, various physics models to be used during the simulation as well as various compute and dump commands to help print data required for post-processing. The particles were considered to be granular in nature. The Hertz-Mindlin model was used for non-adhesive elastic contact between the granular particles. Particles were inserted inside the granulator from the inlet at a constant mass flow rate of 15 kilograms per hour. Rotation speed of the impeller was kept throughout the study at 2000 rotations

**Table 1:** Physical Properties of the particle for the LIGGGHTS input script

| Parameter | Value | Units |
|---|---|---|
| Young's Modulus of particles | $8 \times 10^6$ | N m$^{-2}$ |
| Young's Modulus of Geometry | $1 \times 10^9$ | N m$^{-2}$ |
| Poisson's Ratio | 0.2 | – |
| Coefficient of restitution (constant for all collisions) | 0.4 | – |
| Coefficient of static friction | 0.5 | – |
| Coefficient of rolling friction | 0.2 | – |
| Density of the granules | 500 | kg m$^{-3}$ |

per minute. Such a high rotation speed was chosen since this would lead to high shear between the particles and walls of the shell resulting in better size control of the granules. There were two sets of simulations that were performed, one with mono-sized particles and second consisting of a distribution of sizes. The particle radii chosen for mono-sized simulation varied from 0.59 mm to 2 mm, consecutive particles radii had twice the volume of particles before them. The radii range of the distributed size simulation was 1mm to 3mm. The difference in mechanics of these two simulations is discussed in the Section 4. Physical constants used for the simulations are given in Table 1.

The simulation data was collected after every 50,000 time steps ($5\times10^{-3}$ sec) for the visualization of the particles inside the shell, further post processing . The collisions between each of the particles and collisions between particle and the geometry was collected and used in the PBM.

*3.1.3. Population Balance Modeling (PBM) development*

The population balance equation used in this work is expressed as (Ramkrishna, 2000):

$$\frac{d}{dt}F(s_1, s_2, x, t) = \Re_{agg}(s_1, s_2, x, t) + \Re_{break}(s_1, s_2, x, t) + \dot{F}_{in}(s_1, s_2, x, t) - \dot{F}_{out}(s_1, s_2, x, t) \quad (4)$$

where $F(s_1, s_2, x)$ is the number of particles with an active pharmaceutical ingredients (API) volume of $s_1$ and an excipient volume of $s_2$ in the spatial compartment $x$. The rate of change of number of particles with time in different size classes depend on the rate of aggregation $\Re_{agg}(s_1, s_2, x, t)$ and the rate of breakage $\Re_{break}(s_1, s_2, x, t)$. Also, the rate of particles coming into, $\dot{F}_{in}(s_1, s_2, x, t)$ and going out, $\dot{F}_{out}(s_1, s_2, x, t)$ of the spatial compartment due to particle transfer affect the number of particles in different size classes. The rate of change of liquid volume for a given time in each particle is calculated using:

$$\begin{aligned}\frac{d}{dt}F(s_1, s_2, x, t)l(s_1, s_2, x, t) = \Re_{liq,agg}(s_1, s_2, x, t) + \Re_{liq,break}(s_1, s_2, x, t) + \dot{F}_{in}(s_1, s_2, x, t)l_{in}(s_1, s_2, x, t) \\ - \dot{F}_{out}(s_1, s_2, x, t)l_{out}(s_1, s_2, x, t) + F(s_1, s_2, x, t)\dot{l}_{add}(s_1, s_2, x, t) \quad (5)\end{aligned}$$

where $l(s_1, s_2, x, t)$ is the amount of liquid volume in each particle with API volume of $s_1$ and excipient volume of $s_2$ in the spatial compartment $x$. $\Re_{liq,agg}(s_1, s_2, x, t)$ and $\Re_{liq,break}(s_1, s_2, x, t)$ are respectively the rates of liquid transferred between size classes due to aggregation and breakage. $l_{in}(s_1, s_2, x, t)$ and $l_{out}(s_1, s_2, x, t)$ are respectively the liquid volumes of the particles coming in and going out of the spatial compartment. $\dot{l}_{add}(s_1, s_2, x, t)$ is the volume of liquid acquired by each

particle in the compartment at every time step due to external liquid addition. Similarly, the rate of change of gas volume for a given time is calculated using:

$$\frac{d}{dt}F(s_1,s_2,x,t)g(s_1,s_2,x,t) = \Re_{gas,agg}(s_1,s_2,x,t) + \Re_{gas,break}(s_1,s_2,x,t) + \dot{F}_{in}(s_1,s_2,x,t)g_{in}(s_1,s_2,x,t) - \dot{F}_{out}(s_1,s_2,x,t)g_{out}(s_1,s_2,x,t) + F(s_1,s_2,x,t)\dot{g}_{cons}(s_1,s_2,x,t) \quad (6)$$

where $g(s_1,s_2,x,t)$ is the gas volume of each particle with API volume of $s_1$ and excipient volume of $s_2$ in the spatial compartment $x$. $\Re_{gas,agg}(s_1,s_2,x,t)$ and $\Re_{gas,break}(s_1,s_2,x,t)$ are respectively the rates of gas transferred between size classes due to aggregation and breakage. $g_{in}(s_1,s_2,x,t)$ and $g_{out}(s_1,s_2,x,t)$ are respectively the gas volume of the particles entering and leaving the spatial compartment. $\dot{g}_{cons}(s_1,s_2,x,t)$ is the volume of gas coming out of each particle at every time-step due to consolidation of the particles. The rate of aggregation, $\Re_{agg}(s_1,s_2,x,t)$ in Equation 4 is calculated as (Chaturbedi et al., 2017):

$$\Re_{agg}(s_1,s_2,x,t) = \frac{1}{2}\int_0^{s_1}\int_0^{s_2}\beta(s_1',s_2',s_1-s_1',s_2-s_2',x,t)F(s_1',s_2',x,t)F(s_1-s_1',s_2-s_2',x,t)ds_1'ds_2' - F(s_1,s_2,x,t)\int_0^{s_{max_1}-s_1}\int_0^{s_{max_2}-s_2}\beta(s_1,s_2,s_1',s_2',x,t)F(s_1',s_2',x,t)ds_1'ds_2' \quad (7)$$

where $s_1'$ & $s_2'$ represent for 2 different particles inside the powder granule. The aggregation kernel $\beta(s_1,s_2,s_1',s_2',x)$ is expressed as a function of the collision rate coefficient ($C$) and probability that a collision will result in agglomeration($\psi$) (Ingram and Cameron, 2004) and is shown:

$$\beta(s_1,s_2,s_1',s_2',x) = \beta_o C(s_1,s_2,s_1',s_2',x)\psi(s_1,s_2,s_1',s_2',x) \quad (8)$$

where $\beta_o$ is aggregation rate constant.
Collision rate coefficient ($C$) is a function of particle sizes and is calculated by normalizing the number of collisions between a group of particles (Gantt et al., 2006) and is obtained from a LIGGGHTS DEM simulation. Collision rate coefficient for every time step can be expressed as:

$$C(s_1,s_2,s_1',s_2') = \frac{N_{coll}(s_1,s_2,s_1',s_2')}{N_p(s_1,s_2)N_p(s_1',s_2')\Delta t} \quad (9)$$

In Equation 9, $N_{coll}$ is the number of collisions between two particles in time interval $\Delta t$ and $N_p$ is the number of particles of a particular size. The agglomeration ($\psi$) in Equation 8 can be determined by:

$$\psi((s_1,s_2,s_1',s_2') = \begin{cases}\psi_0, & LC((s_1,s_2) \geq LC_{min} \ \ and \ \ LC((s_1',s_2') \geq LC_{min} \\ 0, & LC((s_1,s_2) < LC_{min} \ \ or \ \ LC((s_1',s_2') < LC_{min}\end{cases} \quad (10)$$

In Equation 10, $LC$ is the liquid content of particles and $LC_{min}$ stands for minimum liquid content required for coalescence of particles.

The rate of increase of liquid volume of a particle, $\dot{l}_{add}(s_1,s_2,x)$ is defined as:

$$\dot{l}_{add}(s_1,s_2,x) = \frac{(s_1+s_2)(\dot{m}_{spray}(1-c_{binder}) - \dot{m}_{evap})}{m_{solid}(x)} \quad (11)$$

**Table 2:** Parameters used in the PBM simulation

| Parameter | Symbol | Value | Units |
|---|---|---|---|
| Initial time step | $\delta t$ | 0.5 | s |
| Mixing time | $T$ | 25 | s |
| Granulation time | $T$ | 75 | s |
| Velocity in axial direction | $v_{axial}$ | 1 | m $s^{-1}$ |
| Velocity in radial direction | $v_{radial}$ | 1 | m $s^{-1}$ |
| Aggregation constant | $\beta_0$ | $1 \times 10^{-9}$ | – |
| Initial particle diameter | $R$ | 150 | $\mu$m |
| Diameter of impeller | $D$ | 0.114 | m |
| Impeller rotation speed | $RPM$ | 2000 | rpm |
| Minimum granule porosity | $\epsilon_{min}$ | 0.2 | – |
| Consolidation rate | $C$ | 0 | – |
| Total starting particles in batch | $F_{initial}$ | $1 \times 10^6$ | – |
| Liquid to solid ratio | $L/S$ | 0.35 | – |
| Number of Compartments | $c$ | 16 | – |
| Number of first solid bins | $s$ | 16 | – |
| Number of second solid bins | $ss$ | 16 | – |

where $(s_1 + s_2)$ is the total solid volume of the particle; $\dot{m}_{spray}$ is the rate of external liquid addition, $c_{binder}$ is the concentration of binder in the external liquid (which is assumed to be zero in this case); $\dot{m}_{evap}$ is the rate of evaporation of liquid from the system (which is also assumed to be zero in this case) and $m_{solid}$ is the total amount of solid present in the compartment. The rate of decrease in gas volume per particle due to consolidation is calculated using the following expression: (Verkoeijen et al., 2002)

$$\dot{g}_{cons}(s_1, s_2, x, t) = c\nu^{\omega}_{impeller} V(s_1, s_2, x) \frac{(1 - \epsilon_{min})}{s} \left[ g(s_1, s_2, x, t) + l(s_1, s_2, x, t) - (s_1 + s_2) \frac{\epsilon_{min}}{1 - \epsilon_{min}} \right] \tag{12}$$

where $c$ and $\omega$ are the consolidation constants; $v_{impeller}$ is the impeller rotational speed; $V(s_1, s_2, x)$ is the volume of particle, $\epsilon_{min}$ is the minimum porosity; $g(s_1, s_2, x)$ and $l(s_1, s_2, x)$ are respectively the gas and liquid volumes of the particle.

Particle transfer rate, $\dot{F}_{out}(s_1, s_2, x)$ in Equation 4 is calculated as:

$$\dot{F}_{out}(s_1, s_2, x) = \dot{F}(s_1, s_2, x) \frac{\nu_{compartment}(x) * dt}{d_{compartment}} \tag{13}$$

where $\nu_{compartment}(x)$ and $d_{compartment}$ are respectively the average velocity of particles in compartment $x$ and the distance between the mid-points of two adjacent compartment, which is the distance particles have to travel to move to the next spatial compartment. $dt$ is the time-step. The process parameters and physical constants used in the PBM simulation are listed in Table 2.

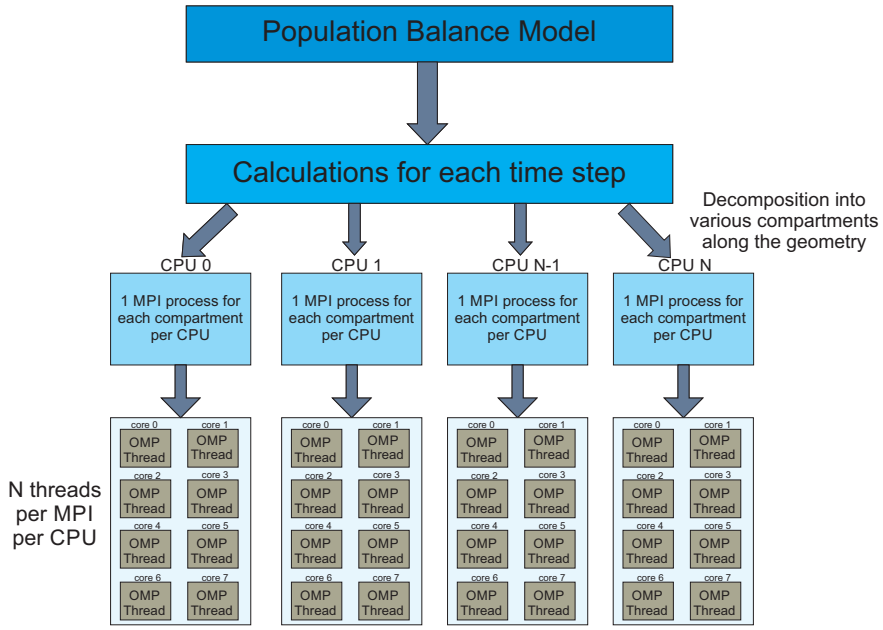



**Figure 4:** The distribution of the PBM calculations to utilize the MPI + OMP parallelization technique on the CPUs of each node

### *3.2. Development and Experiment Analysis*

#### *3.2.1. DEM post processing*

Post processing of data obtained from the DEM simulations was done using MATLAB. The first test run on the output data was to determine if the simulation had reached steady-state. The mass inside the granulator was found out by averaging it over 5 time steps and then compared to mass inside the granulator after every 10000 time steps (about $5\times10^{-4}$ sec) with a tolerance of 10%. If the mass was found to be constant for most of the iterations, it was considered to be at steady state. Another test to determine steady state was to monitor the number of collision inside the granulator. The visualization of the simulation data was done by running the LIGGGHTS post processing (LPP) script over the dump files to convert them into STL files. These files were then loaded in to Paraview (Ayachit, 2017) for a graphical representation of the simulation. It can be seen that the number of collision start to oscillate around a mean value. The number of collisions were then plotted and steady state time was determined. A precautionary script was also run to determine that no particles were lost due to overlap of the geometry with the particles as well as from particle-particle overlap.

#### *3.2.2. Parallel PBM approach*

The PBM was discretized by converting each of its coordinates in to discrete bins. For the spatial coordinates a linear bin spacing was used. For the internal coordinates, solid, liquid, and gas a non-linear discretization was used (Barrasso and Ramachandran, 2012). Once the PBM had been discretized, a finite differences method was used which created a system of ordinary differential equations (ODEs) (Barrasso and Ramachandran, 2015). The numerical integration technique used to evaluate the system of ODEs was first order Euler integration as it is commonly used to solve

these types of systems and (Barrasso et al., 2013) found it decreased computation time while having minimal impact on accuracy as opposed to other integration techniques. In order to avoid numerical instability due to the explicit nature of the Euler integration, Courant-Friedrichs-Lewis (CFL) condition must be satisfied (Courant et al., 1967). For the PBM model, a time-step was calculated at each iteration such that, the number of particles leaving a particular bin at any time was less than the number of particles present at that time (Ramachandran and Barton, 2010).

The model was parallelized such that it could take advantage of shared memory but still effectively run across a distributed system, thus employing a combination of both, MPI and OMP. An algorithm in the form of pseudo code is presented in Algorithm 1 illustrating how the calculations are distributed and carried out during the simulation. For each time step, the MPI processes are made responsible for a specific chunk of the spatial compartments. Subsequently, each OMP thread, inside of each MPI process, is allocated to one of the cores of multi-core CPU the MPI process is bound too. The OMP threads divide up and compute $\Re_{agg}$. After $\Re_{agg}$ is calculated the MPI processes calculate the new PSD value ($F_{t,c}$) for their chunk at that specific time step. The slave processes send their $F_{t,c}$ to the master processes which collects them into the complete PSD ($F_{t,all}$). The master process then broadcasts the $F_{t,all}$ value to all slave processes. This decomposition of the data into different CPUs and further into various threads is illustrated in Figure 4.

**Algorithm 1** Parallel Population Balance Model

**procedure** POPULATION BALANCE MODEL($N_{Comp}$,$N_{MPI}$,$N_{OMP}$) ▷ $N_{Comp}$ is the number of compartments
  Divide $N_{Comp}$ in $N_{MPI}$
  **while** $t < t_{final}$ **do**
    **for** $\forall n_{Comp}$ in 1 MPI process **do**
      **for** $\forall n$ in $N_{OMP}$ **do**
        Calculate $\Re_{agg}$ for solid bins $s_1$,$s_2$
      **end for**
      Calculate $n_{particles}$ using Euler's method
    **end for**
    Collect $n_{particles}$ from $N_{MPI}$ ▷ Master process collects all data
    Calculate *timestep* using *CFL* condition
    $t_{new} = t + timestep$
  **end while**
**end procedure**

The execution of the PBM was done on the newer Stampede2 supercomputer as it supported the developed PBM C++ code. Each of the compute node of the cluster consists of Intel Xeon Phi 7250 ("Knights Landing") which has 68 cores on a single socket clocked at 1.4 GHz, with 96 GB of DDR4 RAM. Each of the cores consists of 4 hardware threads. The simulations were then run by varying the the number of MPI processes from 1 to 16 and the number of OMP threads from 1 to 8, thus, using a maximum of 128 cores.

### *3.3. CyberInfrastructre: RADICAL-Pilot (RP) & Coupled DEM and PBM communication*

RADICAL-Pilot Merzky et al. (2018) defines a number of abstractions to describe resources, and computational tasks. The Pilot Description is the abstraction that describes resources by defining the number of cores, the HPC resource, and the total time that the resources are needed. Computational tasks are defined via the Compute Unit (CU) abstraction. A CU defines the executable, the necessary environment that is needed during execution, execution arguments, and any file dependencies the executable may have. RP also defines two managing components, the Pilot Manager, and Unit Manager. The Pilot Manager is responsible for submitting and monitoring a Pilot. The Unit Manager is responsible for placing CUs in active Pilots. As soon as a Unit is placed in a Pilot, it starts executing.

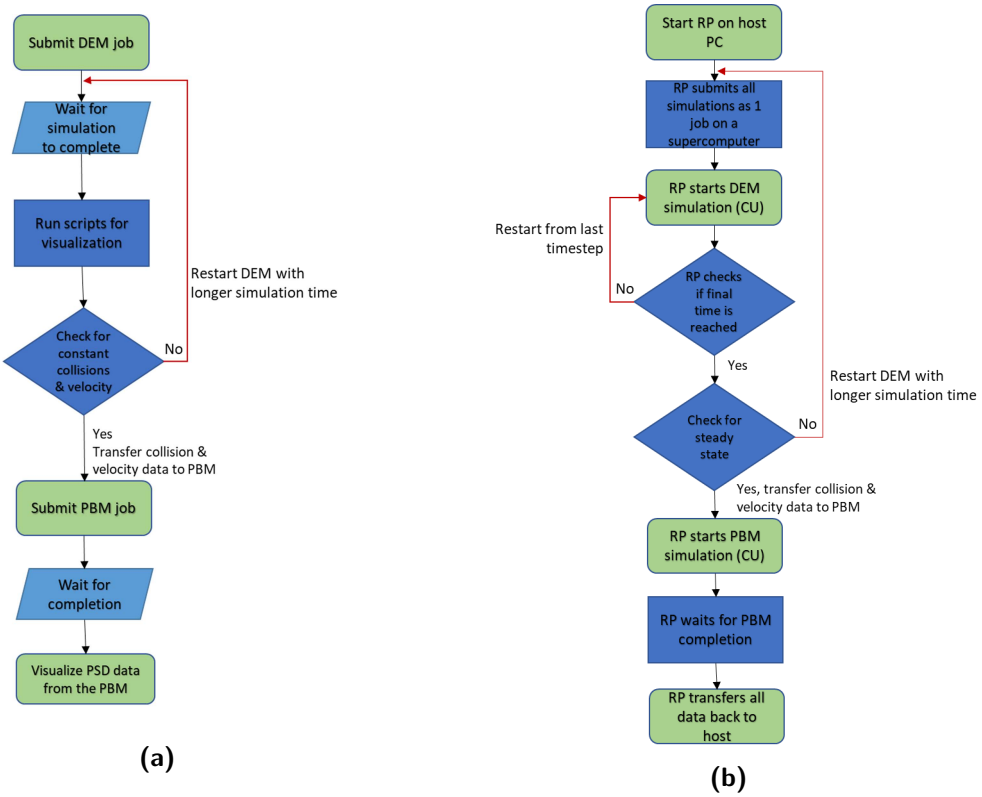



**Figure 5:** Flow charts representing differences in between the coupling of the DEM and PBM simulations manually and using RP respectively.

The DEM data was used to determine the steady state micro-mechanics of the powder mixture inside the high shear granulator. These simulations were also checked for particle loss as mentioned in Section 3.2.1. Physical quantities obtained from the DEM simulation like velocity, collisions were then used in the PBM simulation to give it a more mechanistic nature. Figure 5a represents the flow of data in the coupling process, where each of the DEM and PBM job was submitted manually to the supercomputer.

RADICAL-Pilot (Merzky et al., 2018) is utilized to execute and monitor DEM and PBM simulations. Initially, a pilot description is created to use several cores of a resource and submitted to the pilot manager. A CU description is created for the DEM simulation and submitted to the Unit Manager. As soon as the pilot is active, the DEM simulation starts executing. When it is finished the DEM output is linked to the new CU describing the PBM simulation. The PBM is submitted by the Unit Manager and starts executing. Any data dependency is satisfied by performing the necessary file linking through the CU description. A pictorial representation of the data flow using RP can be found in Figure 5b.

## 4. Results & Discussion

### *4.1. Discrete Element Modeling (DEM)*

#### *4.1.1. DEM Spatial Decomposition Studies*

LIGGGHTS as discussed previously, statically decomposes the work space and each section is sent to a MPI process for calculations. Thus, the division of the space became an important criteria

**Table 3:** Comparison of time taken for the DEM simulations using 128 and 256 core due to different spatial decomposition configurations.

| Number of cores used | Slices in x-direction | Slices in y-direction | Slices in z-direction | Time taken for a 10 second simulation (in minutes) |
|---|---|---|---|---|
| 128 | 16 | 2 | 4 | 264.67 |
| 128 | 16 | 4 | 2 | 247.2 |
| 256 | 16 | 2 | 8 | 271.5 |
| 256 | 16 | 8 | 2 | 252 |
| 256 | 16 | 4 | 4 | 265.32 |

for the simulation for efficient load balancing. The initial timing studies for the decomposition were performed using 64 cores. The time values indicated that dividing the x-direction in more number of compartments decreased the speed of the simulation. The extra slicing in the x-direction decreased the amount of computation carried out by a single process but it increased the number of communications in between the MPI slave and master processes. This led to slower simulations when the number of slices increased beyond 16 in the x-direction. The studies also indicated that if the geometry is divided into more than 2 compartments in either the y-axis or the z-axis the simulation time increased. This can be accounted to increased communication required to transfer the rotating impeller mesh from one compartment in the y-axis or the z-axis to the another compartment for each time step. Since MPI has to pass explicit messages in between the nodes for it to communicate data, there was a delay associated with the master process to obtain all the data. The master process needed to wait for all the other processes to complete their task before it received the data from all other processes. With the increased partitions in the y and the z-axes, the master process needed to wait for more processes, leading to delays before the next time step could be simulated which in turn made the simulation slower.

Following the results from the initial timing studies,the y and the z-axes were not divided in more than two compartments for 128 and 256 core simulation as well. This meant that the x-axis was divided into 32 and 64 compartments respectively.

In order to avoid the expensive communication between the processes, LIGGGHTS attempts to insert the particles towards the center of the compartment such that the number of ghost atoms are minimized. For simulations with higher number of cores the extra slicing in the x direction reduced the space available for the insertion of the particles. This resulted in more Monte-Carlo insertion re-attempts of the particles at the desired space thereby increasing simulation time. Thus, the number of partitions in the x-direction were reduced and a new set of simulations with different 3-D partition configuration were carried out. The comparison of the simulation times with the new configurations are shown in Table 3.

The simulation times illustrated in Table 3 show that incorrect slicing of the geometry also affected the performance of the system. It can be noted that slicing along y-axis is more favorable than slicing along the z-direction. The insertion of particles is hindered when the geometry is cut along the z-axis as the inlet is perpendicular to it. This made LIGGGHTS provision less space for the insertion of these particles, which increased the time of the simulation. Thus, slicing the geometry into 2 sections along the z-axis is preferred. The chosen configurations for the final simulations consisted of the geometry having only 2 sections in the z-direction and more slices along the y-axis.

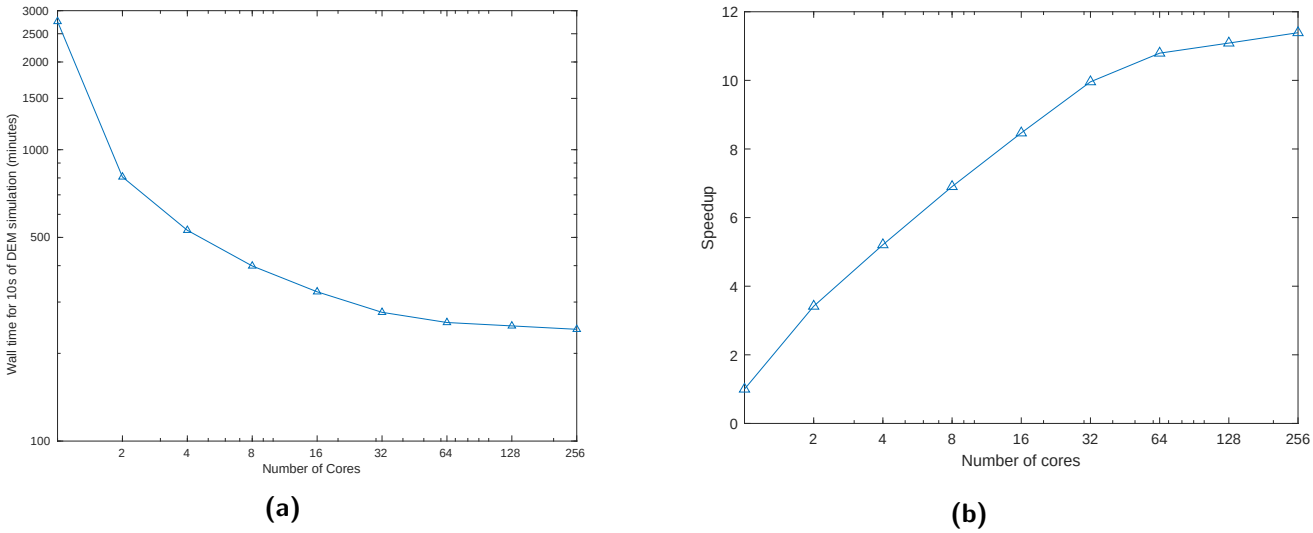



**Figure 6:** (a) The variation in the amount of time taken for the DEM simulation of 2 mm particles as a function of number of cores. The improvements in the speed of the simulation was higher when the cores were increased from 1 to 16 when compared to speed improvements obtained with a core count greater than 16.(b) The speedup achieved in the DEM simulations of 2 mm particles. There was a linear increase up to 16 cores while the speedup seems to plateau when more number of cores are used due to larger amount of communications.

*4.1.2. DEM Performance*

DEM simulation times were compared using a mono-disperse particles distribution of 2 mm. The times plotted in Figure 6a indicated that using lower number of cores is not feasible for long simulations. The time taken while using 1 core is about 11x times slower than a 128 core simulations. Thus, the simulations were carried out in core configurations of 64, 128 and 256 cores. The studies undertaken had 5 mono-sized population of particles of diameter 0.63, 1, 1.26, 1.59 and 2 mm simulations and one simulation consisted of particle size distribution. Figure 7 shows that the amount of wall clock time required for a 10 second simulation of the granulator. The post processing MATLAB script was run on the dump files obtained from the simulation and it was observed that the system reached a steady state about 4-6 seconds of the simulation time. Particles with larger diameter reached steady state at a faster rate with an average hold-up of particles of about 6500.

Pure timing studies are not the best representation for the parallel performance of a program. Speedup of a parallel program indicates the speed increase of the program when it is run on more compute cores compared to the wall clock time when it is run in serial. It is the most common way to represent the parallel performance. Speedup is the ratio of the time taken to run the program in serial to the time taken by the program to run in parallel as shown in Equation 14. For an ideally parallelized program, speedup is equal to $n$, where $n$ is the number of cores used.

$$Speedup = \frac{Wall\ Clock\ Time}{Parallel\ Wall\ Clock\ Time} \tag{14}$$

Speedup does not take into account number of processors used in the simulation. Another metric used to determine parallel performance is Parallel Efficiency. This metric is defined as speedup divided by the number of cores used. Thus, parallel efficiency normalizes speedup and

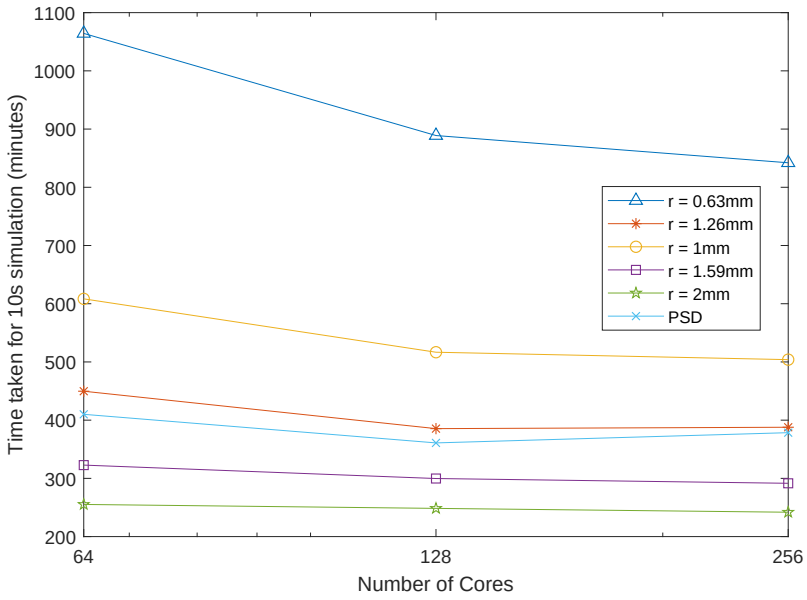



**Figure 7:** Time taken for a 10 second DEM simulation for radii ranging from 0.63 mm to 2 mm and a particle size distribution ranging from 1mm - 3mm. The speed improvements for the smaller particles was higher as the core count increased when compared to the larger particles. The larger number of smaller particles require more computational power thus benefit more as the number of cores are increased.

gives the fraction of the ideal speedup a program achieves increasing the number of cores.

$$Parallel\ Efficiency = \frac{Serial\ Wall\ Clock\ Time}{Parallel\ Wall\ Clock\ Time \times n} \tag{15}$$

The speedup of the DEM simulations using only MPI cores is shown in Figure 6b. There was a linear speed increase in the speedup up to 16 cores, after which the performance plateaued. Figure 6b indicates that there was not a significant amount of speed improvement for the simulation when 256 cores were used for the simulation over 128 cores. This speed decrease could be due to the communication time between the MPI processes. When the particles move from one section to another of the space, they are transferred as ghost particles from one process to another process as well as transfer the geometry for each time step. This led to large amounts of communications between the processes. One of the issues of using a cluster with no shared memory in between nodes is that it has to rely on the networking infrastructure of the cluster for communications leading to higher communication times. There are more sections present when 256 cores are used for the simulation, leading to more communication in this system when compared to the 128 core simulation. These excess communication made the simulation slower though there is more processing power and it required lesser time for other calculations. Another observation that was made from Figure 7 was that the particle size distribution simulation took more time compared to the simulations with mono-sized particles of 1.59 mm and 2 mm, though the mean size of particles in the distribution was 2 mm. The default profile provided by LIGGGHTS indicated that the time spent in calculating the particle-particle interaction forces was higher than the mono-sized simulations. The different diameters made the interaction forces more tedious thus making it computationally more expensive.

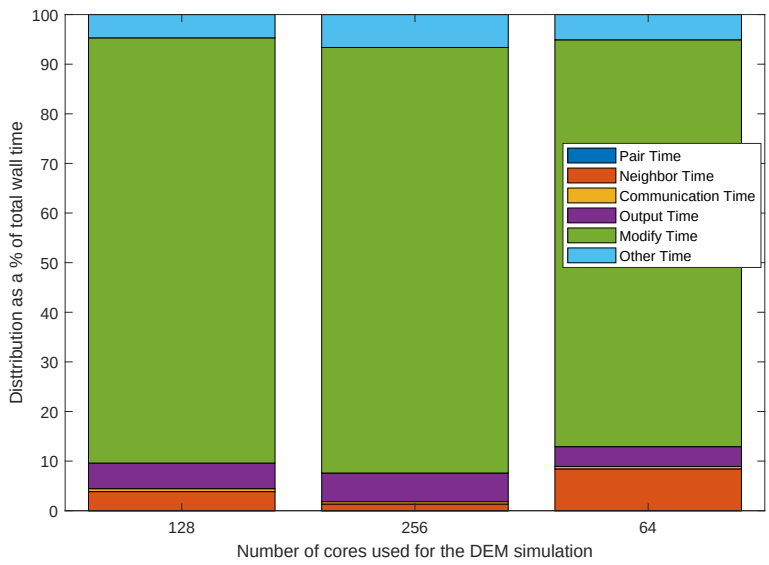



**Figure 8:** The distribution of time taken by each component of the DEM simulation with varying number of cores for the 2 mm particles. It can be observed that the percentage of time required to modify the geometry (indicated by the modify time), i.e. rotating the impeller at high speeds, was the highest.

The communication time in LIGGGHTS is indicated by the Modify time, which is the sum of the times required to transfer the rotating mesh from one process to the other. From Figure 8, it can be seen that the main portion of the simulation time is taken up by the modify time. This was expected since the impeller rotates at a very high speed of 2000 rpm. If the number of processes are increased the amount of time spent in transferring the mesh also increased. In the studies, the modify time as a percentage of the simulation increased from 82% to about 90%, when the core count was increased from the 64 cores to 256 cores. However, using the higher number of cores reduced the time taken to calculate particle-particle interaction as well as neighbor calculation. Thus, 128 cores was a better implementation for meshing and decomposition of the geometry for faster simulations.

### *4.2. Population Balance Modeling (PBM)*

#### *4.2.1. PBM model validation*

The population balance model implemented was considered to have an inlet flow of particles in the first compartment at a constant mass flow rate of 15 kilograms per hour. The particles were assumed to have a log normal distribution with minimum diameter of the particles being about 12 $\mu$m and the maximum of 2.1 mm with a mean of 0.453 mm and a standard deviation of about 0.62 mm.

The PBM used in this study employed an aggregation kernel that takes into account the formation of a larger particle from only two smaller particles. The ratio of the rate of formation and the rate of depletion due to aggregation helped us monitor whether the PBM satisfied the conservation of mass. Since this PBM took into account the aggregation of only 2 particles at once, the ratio of the particles was expected to have a value of 0.5 during the aggregation process. This ratio was reported by the PBM to be 0.5 throughout the simulation, validating the accuracy of the PBM used.

Figure 9 shows four different particle size distribution plots at four different time instants. Figure 9(a) shows the distribution of the particles at 30 seconds, where we expected to have the highest number of the smaller particles. Since the degree of aggregation that has occurred was very low, there was a jump in the number of particles of the smaller size. The particle size distributions at 2 intermediate times of 50 seconds and 75 seconds are plotted in Figures 9(b) & 9(c) respectively. These illustrate that there was an increase in the number of larger particles and a subsequent decrease in the number of smaller particles. Figure 9(d) shows the distribution of the particle size at the end of the granulation process. It can be seen that the number of particles in the higher diameter bins had increased.

*4.2.2. Parallel PBM performance studies*

The PBM was run for the results obtained from each of the aforementioned DEM simulations. Since the PBM has been parallelized using hybrid techniques, a combination of MPI and OMP cores were used to perform the simulations. Figure 10 shows the average time taken by a PBM simulation for a total of 100 seconds of the granulation process, which included 25 seconds of the mixing time and 75 seconds of granulation time. The time taken for simulating all DEM scenarios by a single set of core configuration of in less than 10% of each other, thus, an average time for a single core configuration was used to illustrate the performance. These simulations were run in a varied configuration of cores ranging from 1 to 128. The cores were initially increased to 16 by increasing only the number of MPI processes. To increase the number of cores used, 8 OMP threads were employed for a configuration of 32(4 MPI and 8 OMP), 64(8 MPI and 8 OMP) and 128 cores (16 MPI and 8 OMP). Figure 10 shows that the program scaled well to about 32 cores but then, the improvement in the performance is negligible. The scaling with the only MPI cores showed substantial increase in performance.

Figure 11a depicts the speed up achieved by the hybrid parallel PBM code. It can be seen when the MPI cores used were increased from 1 to 16 cores the speed up achieved was almost linear. This speed increase was due to the way MPI has been implemented inside the code. Each compartment of the granulator was offloaded on one MPI process, thus making 16 MPI processes the ideal since, the granulator had been divided into 16 compartments. When less than 16 cores were used, more than 1 compartment was sent to a single MPI process, which led to a decrease in performance. The implementation of OMP on top of the MPI parallelization helped improve the performance by the about 10%. The calculations inside the OMP parallel section of the code consisted of large number of nested loop which have been known to be difficult to parallelize (He et al., 2016) using the native *C++*'s OMP libraries. The amount of communication time spent in between these threads was much higher than the speed increase achieved by using higher number of cores. One OMP thread had to wait for another thread to complete processing the outer loops thus lower performance increase is achieved by using the OMP implementation.

A similar behavior as Figure 11b was observed where the parallel efficiency of the program was plotted against the number of the cores used. This efficiency decreased as the number of cores used increased. The decrease in efficiency initially up till 8 cores was steep as the communication time in between the MPI process decreased the efficiency, deviating its value from the ideal of 1 to about 0.4. The implementation of OMP decreased the efficiency further as there was no major increase in the performance when compared to the increase in the number of cores used. The efficiency fell to as low as 11% when 128 cores are used. Thus, there is scope for improvement in the parallel implementation of the program using OMP, especially in section of the code where nested loops are present.

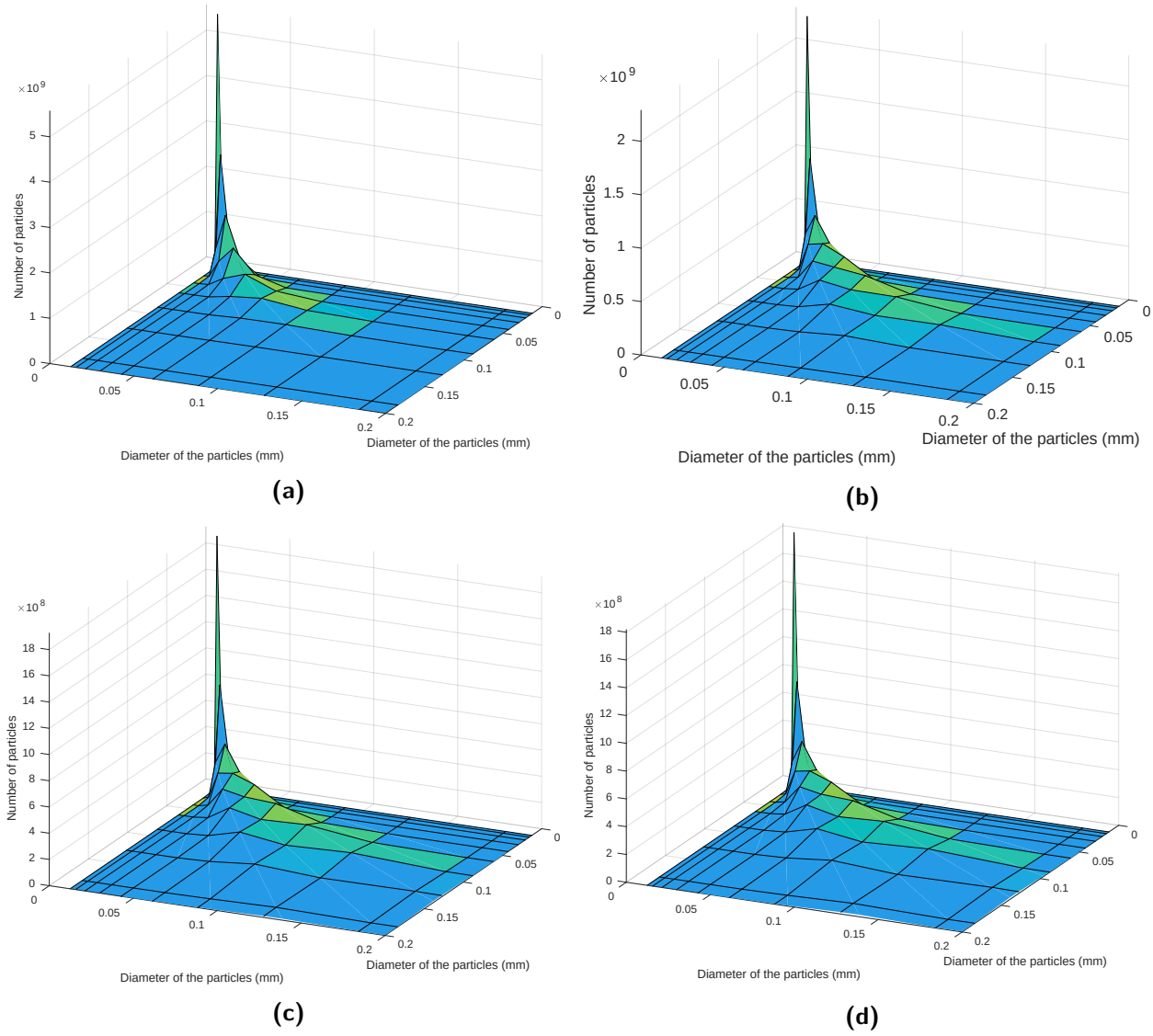



**Figure 9:** Representation of the total number of particles inside all compartments with diameters less than 0.2 mm in mm after (a) 30s (b) 50s (c) 75s and (d) 100s of PBM simulation (Mixing takes place for the first 25 seconds).

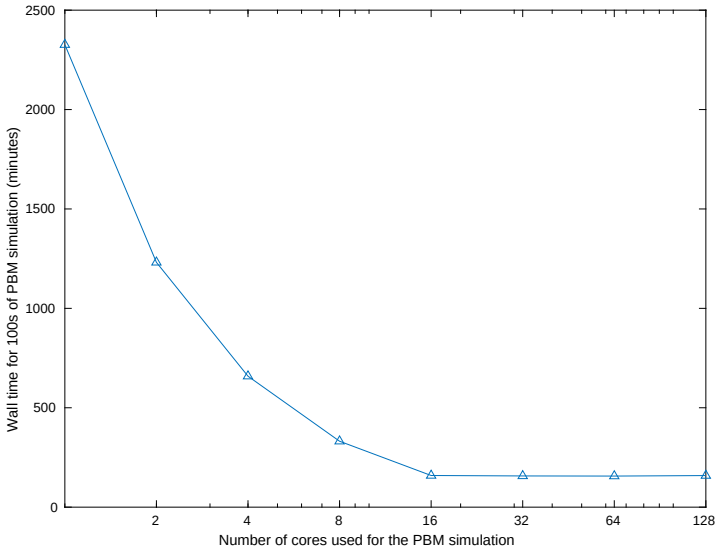
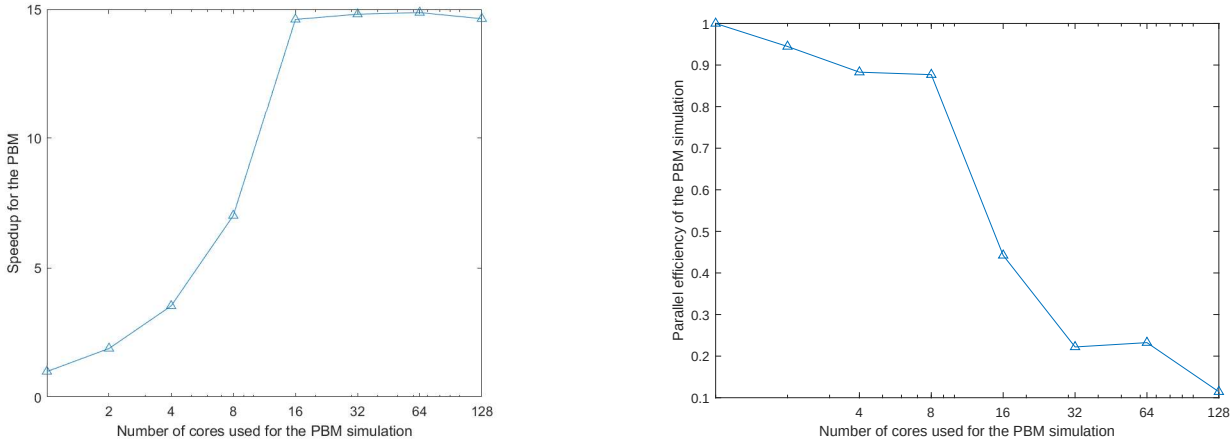



**Figure 10:** Average time taken to run the PBM simulation which consisted of 25 seconds of mixing time and 75 seconds of granulation time at different core configurations. There was a steady decrease as the number of MPI processes were increased, but the improvements on increasing the OMP threads were not that significant.




**(a)** **(b)**

**Figure 11:** (a) The speedup achieved by the hybrid parallel implementation of the PBM program. It can be seen the initial speedup up to 16 cores as expected as from an ideal parallel program where as it becomes almost constant from 32 cores to 128 cores(b) The parallel efficiency for the hybrid parallelized PBM code. The efficiency of the code decreased as the number of cores are increased. The higher number of cores have very low efficiency of about 11% which depicted that there is large time being spent in communication in between the cores.

### *4.3. Coupled DEM and PBM physics*

The micro-scale simulations provide an insight into the physics of the system usually by tracking each particle. This micro-scale simulation data is useful for the development of macro-scale mechanistic models which take into account the dynamics of bulk of the particles and not individual particles. A similar approach has been implemented in this work, where a mechanistic aggregation kernel was developed from the DEM particle-particle collisions. Thus, the aim of this section was to illustrate that the physics of the system does not change to a great extent with the change in the size of the particles or the distribution of the particles.

Two simulations from the study were compared, the Coupled DEM and PBM simulation of the 2 mm mono-sized particle and the second being the simulation where the inserted particles were in a size distribution. The ratio of rate of formation to the rate of depletion, both due to aggregation observed during these simulations was 0.5 which indicated that the both PBMs were stable.

One of the metrics to determine the physics of the system after a PBM simulation is to check the median diameter ($D_{50}$) plots of the system after the granulation process. $D_{50}$ indicates the maximum diameter of particles that constitute 50% of the total mass. These diameters vary along the length of the granulator since the granulated particles take time to pass through the granulator and that there is not enough liquid content in the later sections of the granulator to encourage the formation of granules. Thus, the $D_{50}$ for compartments in the latter section of the granulator is usually low. Figures 12a & 12b show the $D_{50}$ plots of the mono-sized and PSD respectively. It can be seen that both these plots have a similar behavior when it comes to the nature of the increase of the $D_{50}$ during the granulation process. There is a difference of about 20% in the final diameter of the particles predicted, which at the of micrometer scale does not affect the final product quality. This slight deviation was observed due to the sudden jump in the rate of the aggregation which increased when particles from one compartment with higher $D_{50}$ got transferred to the next compartment with a lower $D_{50}$. It can be seen from Figure 7 that the 2 mm mono-sized particle had a speed advantage over the PSD simulation as it took about 1.6 times less. Since the physics of both the systems were not different, using the mono-sized simulations for the DEM could help save time on the overall simulations.

### *4.4. One-way coupled DEM+PBM using RADICAL-Pilot and its performance*

With resources allocated, the pilot can execute multiple simulations at once. RADICAL-Pilot initiated the DEM simulation using the LIGGGHTS executable. As soon as DEM is complete, links are created to the outputed collision data and a PBM is started. Due to the similarity between the simulations – same executables and input files – we did not expect any changes in the physical system.

The DEM simulations were run for 64, 128 and 256 cores (MPI processes) and the PBM was run from 1, up to 16 MPI processes. We did not utilize the OpenMP capabilities of PBM, because the used version 0.47 of RADICAL-Pilot (Merzky et al., 2018) did not distinguish between processes and threads. Initially, DEM and PBM were submitted using custom batch scripts to provide with execution profiles that could be used as the basis of the analysis. Execution via RADICAL-Pilot required to break the DEM simulation to smaller simulations. This was due to some limitations RADICAL-Pilot presented when running Compute Units (CUs) for more than $\sim$ 6 hours. Thus, the execution included 10 CUs for DEM and a single CU for PBM. The same core configurations were used.

Figure 13 shows the comparison between the uncoupled execution of DEM and PBM and the one with RADICAL-Pilot. Both DEM and PBM provide profiling data in seconds. For RADICAL-Pilot, we aggregated the timings of the first 10 units, since it concerns only the DEM step of the

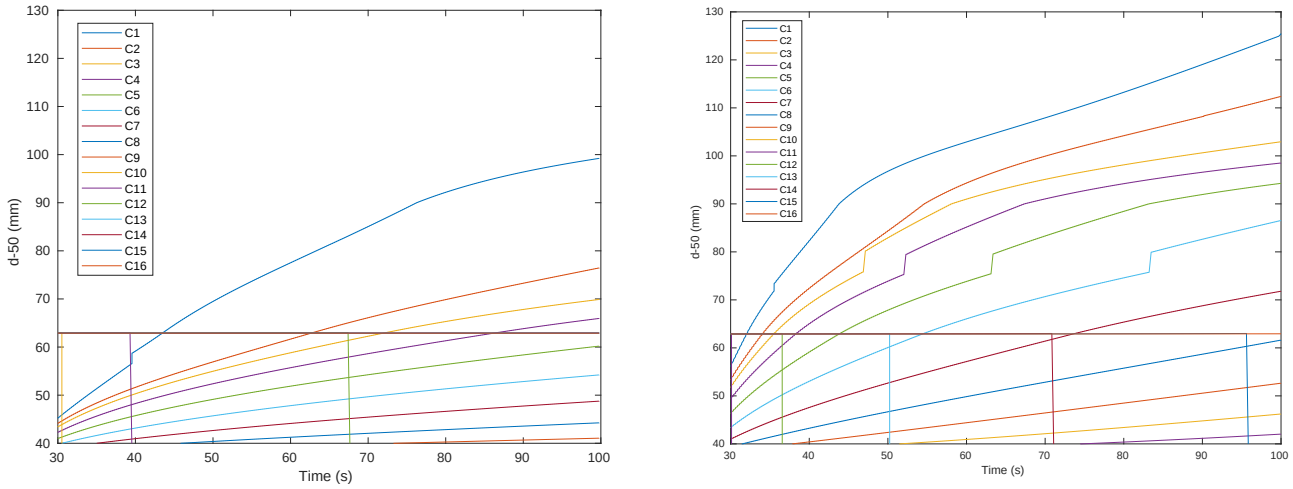



**Figure 12:** (a) $D_{50}$ of the particles obtained after 100 s of granulation time (25 s of mixing and 75 s of liquid addition) for the 2 mm mono-sized particle DEM simulation. (b) $D_{50}$ of the particles obtained after 100 s of granulation time (25 s of mixing and 75 s of liquid addition) for the distributed particle size DEM simulation. The trend observed was similar to the trend found in Figure 12a.

simulation. DEM's simulation is significantly larger when using RADICAL-Pilot compared to the single script execution. Restarting the DEM simulation add significant overheads to the overall execution. In addition, we verified that the difference cannot be attributed to RADICAL-Pilot. Figure 13 includes the overheads presented by RADICAL-Pilot, but they are so small compared to the overall execution that are barely visible. Despite the significant performance, we were able to multiple such simulation running concurrently via RADICAL-Pilot without the need to submit every simulation to the batch queue, thus reducing the wait time when executing multiple configurations.

## 5. Conclusions

A simple uni-directional DEM+PBM coupled study for a high shear granulator has been presented. DEM simulations were used to determine the movement of the particles inside the granulator and to obtain the collision data. This collision data was then used in the multi-dimensional PBM which was developed with 2 solid bins. Both these executables were parallelized to help them utilize the multi-core hardware of a cluster. The DEM was parallelized using MPI where as the developed PBM used MPI as well as OMP for a faster execution. The speed-up achieved for the DEM simulation was about 12 times using 64 cores, where as the PBM achieved a speed-up of 14 times when 16 cores were used. RP was utilized to execute the simulations for lower wait times. A more accurate model for the high shear granulator would be to couple the DEM and PBM bidirectionally i.e. these two methods are executed iteratively up till a steady state is reached. For the two-way coupling, the computation resources required would be really large as it would involve multiple DEM and PBM simulations. The role of RP in such a detailed simulation would be more crucial as it would need to handle more communications and various job submissions. This would also help reduce total time taken to complete the required runs.

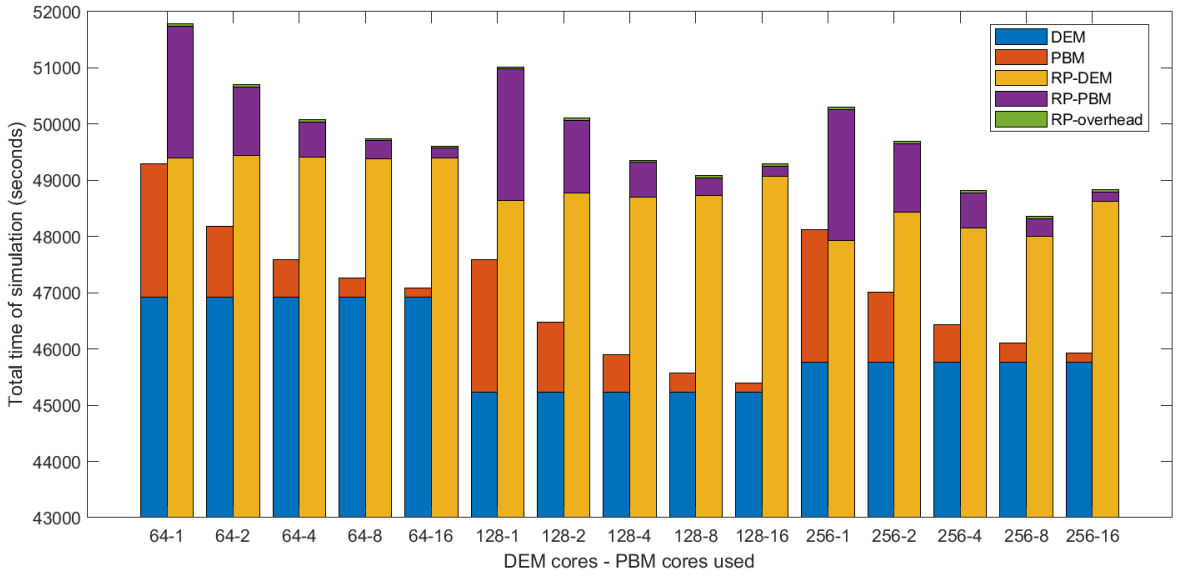



**Figure 13:** A comparison of times taken by 2 individual simulations (DEM and PBM) (without including queue time) to the time taken by each simulation in the RP job of the RADICAL-Pilot. A small communication time overhead can be seen which is a very small when compared to the time of the complete simulation.

## Software and Data

Source code and input scripts for reproduction of these simulations can be found at:
Cybermanufacturing: `https://github.com/radical-collaboration/CyberManufacturing`

## Acknowledgments

The authors would like to acknowledge National Science Foundation (NSF) for funding this project through the grant number: 1547171. Computational resources were provided by NSF XRAC award TG-MCB090174.

## References

Adhianto, L., Chapman, B., 2007. Performance modeling of communication and computation in hybrid MPI and OpenMP applications. Simulation Modelling Practice and Theory 15 (4), 481–491.

Ayachit, U., 2017. The paraview guide. Kitware, Inc.

Barrasso, D., Eppinger, T., Pereira, F. E., Aglave, R., Debus, K., Bermingham, S. K., Ramachandran, R., 2015. A multi-scale, mechanistic model of a wet granulation process using a novel bi-directional PBM–DEM coupling algorithm. Chemical Engineering Science 123, 500–513.

Barrasso, D., Ramachandran, R., 2012. A comparison of model order reduction techniques for a four-dimensional population balance model describing multi-component wet granulation processes. Chemical Engineering Science 80, 380–392.

Barrasso, D., Ramachandran, R., 2015. Multi-scale modeling of granulation processes: bi-directional coupling of PBM with DEM via collision frequencies. Chemical Engineering Research and Design 93, 304–317.

Barrasso, D., Walia, S., Ramachandran, R., 2013. Multi-component population balance modeling of continuous granulation processes: a parametric study and comparison with experimental trends. Powder Technology 241, 85–97.

Berger, R., Kloss, C., Kohlmeyer, A., Pirker, S., 2015. Hybrid parallelization of the LIGGGHTS open-source DEM code. Powder Technology 278, 234–247.

Bettencourt, F. E., Chaturbedi, A., Ramachandran, R., 2017. Parallelization methods for efficient simulation of high dimensional population balance models of granulation. Computers & Chemical Engineering 107 (Supplement C), 158–170.

Boman, E. G., Çatalyürek, Ü. V., Chevalier, C., Devine, K. D., 2012. The Zoltan and Isorropia parallel toolkits for combinatorial scientific computing: Partitioning, ordering and coloring. Scientific Programming 20 (2), 129–150.

Bouffard, J., Bertrand, F., Chaouki, J., 2012. A multiscale model for the simulation of granulation in rotor-based equipment. Chemical Engineering Science 81, 106–117.

Cameron, I., Wang, F., Immanuel, C., Stepanek, F., 2005. Process systems modelling and applications in granulation: A review. Chemical Engineering Science 60 (14), 3723–3750.

Chaturbedi, A., Bandi, C. K., Reddy, D., Pandey, P., Narang, A., Bindra, D., Tao, L., Zhao, J., Li, J., Hussain, M., Ramachandran, R., 2017. Compartment based population balance model development of a high shear wet granulation process via dry and wet binder addition. Chemical Engineering Research and Design 123, 187–200.

Christofides, P. D., Li, M., Mädler, L., 2007. Control of particulate processes: Recent results and future challenges. Powder Technology 175 (1), 1 – 7.

Courant, R., Friedrichs, K., Lewy, H., 1967. On the partial difference equations of mathematical physics. IBM Journal of Research and Development 11 (2), 215–234.

Cundall, P. A., Hart, R. D., 1992. Numerical modeling of discontinua. Engineering Computations 9 (2), 101–113.

Cundall, P. A., Strack, O. D., 1979. A discrete numerical model for granular assemblies. Geotechnique 29 (1), 47–65.

Dosta, M., Bröckel, U., Gilson, L., Kozhar, S., Auernhammer, G. K., Heinrich, S., 2018. Application of micro computed tomography for adjustment of model parameters for discrete element method. Chemical Engineering Research and Design 135, 121 – 128.

Gantt, J. A., Cameron, I. T., Litster, J. D., Gatzke, E. P., 2006. Determination of coalescence kernels for high-shear granulation using DEM simulations. Powder Technology 170 (2), 53–63.

Gantt, J. A., Gatzke, E. P., 2005. High-shear granulation modeling using a discrete element simulation approach. Powder Technology 156 (2), 195 – 212, particle Technology Forum Special Issue.

Goldschmidt, M., Weijers, G., Boerefijn, R., Kuipers, J., 2003. Discrete element modelling of fluidised bed spray granulation. Powder Technology 138 (1), 39–45.

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

Gunawan, R., Fusman, I., Braatz, R. D., 2008. Parallel high-resolution finite volume simulation of particulate processes. AIChE Journal 54 (6), 1449–1458.

Hancock, B. C., Ketterhagen, W. R., 2011. Discrete element method (DEM) simulations of stratified sampling during solid dosage form manufacturing. International Journal of Pharmaceutics 418 (2), 265–272.

Hassanpour, A., Pasha, M., Susana, L., Rahmanian, N., Santomaso, A. C., Ghadiri, M., 2013. Analysis of seeded granulation in high shear granulators by discrete element method. Powder Technology 238, 50–55.

He, J., Chen, W., Tang, Z., 2016. NestedMP: Enabling cache-aware thread mapping for nested parallel shared memory applications. Parallel Computing 51, 56–66.

Immanuel, C. D., Doyle, F. J., 2005. Solution technique for a multi-dimensional population balance model describing granulation processes. Powder Technology 156 (2), 213 – 225, particle Technology Forum Special Issue.

Ingram, G. D., Cameron, I. T., 2004. Challenges in multiscale modelling and its application to granulation systems. Asia-Pacific Journal of Chemical Engineering 12 (3-4), 293–308.

Iveson, S. M., Litster, J. D., Hapgood, K., Ennis, B. J., 2001. Nucleation, growth and breakage phenomena in agitated wet granulation processes: a review. Powder Technology 117 (1), 3–39.

Jin, H., Jespersen, D., Mehrotra, P., Biswas, R., Huang, L., Chapman, B., 2011. High performance computing using MPI and OpenMP on multi-core parallel systems. Parallel Computing 37 (9), 562–575.

Kačianauskas, R., Maknickas, A., Kačeniauskas, A., Markauskas, D., Balevičius, R., 2010. Parallel discrete element simulation of poly-dispersed granular material. Advances in Engineering Software 41 (1), 52–63.

Kloss, C., Goniva, C., Hager, A., Amberger, S., Pirker, S., 2012. Models, algorithms and validation for opensource DEM and CFD–DEM. Progress in Computational Fluid Dynamics, an International Journal 12 (2-3), 140–152.

Litster, J., 2016. Design and Processing of Particulate Products. Cambridge University Press.

Luckow, A., Santcroos, M., Merzky, A., Weidner, O., Mantha, P., Jha, S., 2012. P*: A model of pilot-abstractions. IEEE 8th International Conference on e-Science, 1–10.

Merzky, A., Turilli, M., Maldonado, M., Santcroos, M., Jha, S., 2018. Using Pilot Systems to Execute Many Task Workloads on Supercomputers.

Mindlin, R., Deresiewicz, H., 1953. Elastic spheres in contact under varying oblique forces. Adv Energy Syst Div, 327–344.

672 Poon, J. M.-H., Immanuel, C. D., Francis J. Doyle, I., Litster, J. D., 2008. A three-dimensional
673 population balance model of granulation with a mechanistic representation of the nucleation and
674 aggregation phenomena. Chemical Engineering Science 63 (5), 1315 – 1329.

675 Prakash, A. V., Chaudhury, A., Barrasso, D., Ramachandran, R., 2013a. Simulation of popula-
676 tion balance model-based particulate processes via parallel and distributed computing. Chemical
677 Engineering Research and Design 91 (7), 1259–1271.

678 Prakash, A. V., Chaudhury, A., Ramachandran, R., 2013b. Parallel simulation of population bal-
679 ance model-based particulate processes using multicore CPUs and GPUs. Modelling and Simu-
680 lation in Engineering 2013, 2.

681 Ramachandran, R., Barton, P. I., 2010. Effective parameter estimation within a multi-dimensional
682 population balance model framework. Chemical Engineering Science 65 (16), 4884–4893.

683 Ramachandran, R., Immanuel, C. D., Stepanek, F., Litster, J. D., Doyle, F. J., 2009. A mechanistic
684 model for breakage in population balances of granulation: Theoretical kernel development and
685 experimental validation. Chemical Engineering Research and Design 87 (4), 598–614.

686 Ramkrishna, D., 2000. Population balances: Theory and applications to particulate systems in
687 engineering. Academic press.

688 Ramkrishna, D., Singh, M. R., 2014. Population balance modeling: current status and future
689 prospects. Annual Review of Chemical and Biomolecular Engineering 5, 123–146.

690 Reinhold, A., Briesen, H., 2012. Numerical behavior of a multiscale aggregation model–coupling
691 population balances and discrete element models. Chemical Engineering Science 70, 165–175.

692 Rogers, A. J., Hashemi, A., Ierapetritou, M. G., 2013. Modeling of particulate processes for the
693 continuous manufacture of solid-based pharmaceutical dosage forms. Processes 1 (2), 67–127.

694 Sen, M., Barrasso, D., Singh, R., Ramachandran, R., 2014. A multi-scale hybrid CFD-DEM-PBM
695 description of a fluid-bed granulation process. Processes 2 (1), 89–111.

696 Sen, M., Dubey, A., Singh, R., Ramachandran, R., 2012. Mathematical development and compari-
697 son of a hybrid PBM-DEM description of a continuous powder mixing process. Journal of Powder
698 Technology 2013.

699 Sen, M., Ramachandran, R., 2013. A multi-dimensional population balance model approach to
700 continuous powder mixing processes. Advanced Powder Technology 24 (1), 51–59.

701 Seville, J., Tüzün, U., Clift, R., 2012. Processing of particulate solids. Vol. 9. Springer Science &
702 Business Media.

703 MathWorks$^{TM}$ Documentation, 2017b. Parallel computing toolbox - MATLAB®.
704 https://www.mathworks.com/products/parallel-computing.html.

705 Verkoeijen, D., Pouw, G. A., Meesters, G. M., Scarlett, B., 2002. Population balances for particulate
706 processes—a volume approach. Chemical Engineering Science 57 (12), 2287–2303.