Advances in Engineering Software 41 (2010) 52–63

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

Parallel discrete element simulation of poly-dispersed granular material

Rimantas Kačianauskas [a,*], Algirdas Maknickas [b], Arnas Kačeniauskas [b], Darius Markauskas [a], Robertas Balevičius [c]

[a] Laboratory of Numerical Modelling, Vilnius Gediminas Technical University, Sauletekio al. 11, 10223 Vilnius, Lithuania
[b] Laboratory of Parallel Computing, Vilnius Gediminas Technical University, Sauletekio al. 11, 10223 Vilnius, Lithuania
[c] Department of Reinforced Concrete and Masonry Structures, Vilnius Gediminas Technical University, Sauletekio al. 11, 10223 Vilnius, Lithuania

ARTICLE INFO

Article history:
Available online 6 January 2009

Keywords:
Particle compacting
Discrete element method
Poly-dispersed granular material
Parallel computing
Spatial domain decomposition
Distributed memory PC clusters

ABSTRACT

The paper presents parallel 3D DEM simulation of poly-dispersed material described by the normal size distribution. Static domain decomposition and message passing inter-processor communication have been implemented in the DEM code. A novel algorithm for moving particles that exchange processors has been incorporated in the domain decomposition framework. Parallel performance of the developed algorithm and software has been investigated by a series of benchmark tests conducting tri-axial compaction of material with different numbers of particles, heterogeneity ratios and compaction durations. The speed-up equal to 8.81 has been obtained on 10 processors of the distributed memory PC cluster. It has been shown that a drastic increase of computational expenses of simulation for the poly-dispersed material in terms of CPU time is associated with the increase of its heterogeneity. A contribution of the temporal evolution of microscopic behaviour has also been illustrated.

1. Introduction

Among various numerical techniques, the discrete (distinct) element method (DEM) became widely recognised after the pioneering work published by Cundall [1] and later work of Cundall and Strack [2]. The representation of granular media as an assembly of contacting particles termed hereafter as discrete elements was seen as a more realistic approach compared to continuum models. In the DEM, the particles of granular media are treated as individual objects and all dynamical state variables of each particle are tracked during the simulation. The DEM allows simulation of motion and interaction between the particles, taking into account the microscopic geometry and various constitutive models.

The main advantage of the DEM is a possibility to model highly complex poly-dispersed systems using the basic data on individual particles without making oversimplifying assumptions. This makes DEM different from the conventional methods of the continuum mechanics, such as the finite difference, finite element and boundary element methods, and helps to avoid difficulties encountered in describing the microscopic nature of the granular media at the continuum level.

A series of comprehensive works and review papers discovered different aspects of the DEM and granular matter oriented issues. The theoretical frame of the DEM in terms of microscopic–macroscopic relationship is discussed by Luding [3] and Luding et al. [4]. It is widely recognised that the macroscopic properties of the particulate assemblies depend on their single particle properties and the interaction between contiguous particles, while modelling the mechanical behaviour of particulates can be reflected by inter-particle stiffness models. The-state-of-the-art in constitutive inter-particle models including particle adhesion and loading–unloading–reloading contact deformation paths applicable on micro and even nano-scale is presented by Tomas [5]. The influence of stiffness on the microscopic and macroscopic deformation characteristics of differently shaped particulate assemblies is considered by Moreno-Atanasio and Antony [6], while the effect of the surrounding liquid on their cohesive properties by Tüzün and Heyes [7]. By introducing the modified inter-particle models, the DEM may be easily employed for dynamic simulation of continuous media and multi-fracture behaviour, in particular [8–10]. Concerning a computational procedure, the most detailed and transparent presentation of algorithmic structure and details of the DEM are given by Džiugys and Peters [11]. Different programming approaches used for the development of sequential software codes for discrete element method and performance of separated procedures are investigated in [12].

Recently, the DEM has been extensively applied to examine different phenomena inside the granular materials, among which the compaction problem is of major importance in fundamental research and applications. Comprehensive reviews and fundamentals of multi-axial compacting are given by Martin et al. [13] and by Procopio and Zavaliangos [14], while some particular applications

* Corresponding author. Tel.: +370 274 48 53; fax: +370 274 48 44.
E-mail addresses: rkac@fm.vtu.lt (R. Kačianauskas), alm@sc.vtu.lt (A. Maknickas), Robertas.Balevicius@st.vtu.lt (R. Balevičius).

0965-9978/$ - see front matter © 2008 Civil-Comp Ltd. and Elsevier Ltd. All rights reserved.
doi:10.1016/j.advengsoft.2008.12.004

are presented in [15–18]. Comparison of DEM and FEM as well as combined approaches are investigated in [13–15]. It should be noted, that regardless of the specific nature of numerically simulated mechanical testing, the stage of an artificial contact-free particle compaction of initially packed configuration (or generation of physically adjustable initial conditions) is required [10,15–18].

The main disadvantages of the DEM technique, in comparison with the well-known continuum methods, are related to computational capabilities which are limited by a huge number of particles [19] and a short-time interval of simulations. The small time step imposed in the explicit time integration schemes gives rise to the requirement that a very large number of time increments should be performed. In addition, most of CPU time is spent on finding contacts between the particles. Considerable efforts have been expended by researchers to optimize this procedure for vector supercomputers [20] and even to design reconfigurable co-processors for DEM simulations [21].

Naturally, for the solution of industrial-scale problems, parallelization becomes an obvious option for significantly increasing computational capabilities. A fine-grain, shared memory parallelization is extensively used in the area of molecular dynamics [22,23]. Most of the authors [24] are convinced that a message passing model of programming [25] for distributed memory MIMD machines is the only model providing flexibility sufficient for implementing all the data structures and computational enhancements that are commonly exploited in DEM codes on sequential computers.

In recent years, there has been a considerable interest in developing parallel DEM codes. The design of parallel DEM algorithms presents a new challenge to computational scientists. The natural parallelism of DEM is that the force calculations and position updates can be done simultaneously for all particles. Two main ideas have been exploited to achieve this parallelism. In the first class of methods a pre-determined set of force computations is assigned to each processor. The assignment remains fixed for the duration of the simulation. The simplest way of doing this is to give a subgroup of particles to each processor. This method is called atom decomposition or force decomposition of the workload, since the processor computes forces on its particles no matter where they move in the simulation domain. Such methods have shown good performance for shared memory computers [26], but the global character of the employed algorithms produces inter-processor communication overhead on distributed memory machines [27].

In the second class of methods, the spatial decomposition [24], well-known as domain decomposition [28], is employed. The basic idea of this technique is the partitioning of the computational domain into sub-domains, each being assigned to a processor. The sub-domains exchange data with each other through their boundaries. If localized data dependency is not destroyed, time integration on the particles of a particular sub-domain can be sought entirely in parallel. Then, all required data will be local to the processor in control of that specific sub-domain. Inter-processor communication is necessary only when the data on neighbouring particles residing in other sub-domains is required for computations of contact forces. A parallel computer is used efficiently if load balancing is performed and none of the processors have to wait for information they need from other processors. Referring to continuum problems, two basic types, static or dynamic domain decomposition strategies, are extensively used in solving the time-dependent problems [29]. Static domain decomposition algorithms work by assuming the fixed inter-domain boundaries [30], while more flexible, but more complicated dynamic decomposition algorithms [31] allow us to move boundaries during the simulation keeping load balancing of individual processors. The cubic decomposition methods based on the hierarchical tree employ moving boundaries and also try to keep optimal shaped sub-domains [32–34]. As the degree of natural algorithmic concurrency inherent in explicit time integration procedures is high, domain decomposition-based parallel processing strategies can yield high efficiencies and speed-ups on both shared and distributed memory hardware configurations [23,8] and/or cluster systems [35].

The parallel DEM algorithms employing the domain decomposition differ from analogous parallel processing in the continuum approach. Moving particles dynamically change the workload configuration, making parallelization of DEM software much more difficult and challenging [36]. Essentially, the processor workload, inter-processor communication and data storage requirements undergo a continuous evolution during the simulation [37]. This problem poses considerable difficulties for distributed memory machines. Only recently, have some successful attempts emerged in tackling problems of a similar nature [36,38]. On the other hand, increasing efficiency of the parallel software may be achieved regarding particular features of the problem under consideration and proper handling of microscopic motions. In summary, a recent progress resulted in the dramatic increase of computational capacity, which could exceed millions of particles [32,36].

In spite of a considerable progress in developing parallel DEM software, their application to the solution of large-scale industrial problems is rather limited. The results presented by Landry et al. [39,40] may be considered as successful stories. In particular, granular packing up to 200,000 particles in a cylindrical container [39] and a rectangular box [40] was simulated on 50 processors. It should be noted that the majority of applications of the DEM are dealing with mono-disperse like material, where the ratio of particle diameters $\alpha = d_{\max}/d_{\min}$ is almost uniform $\alpha = 1.2 \times 1.0$. Some examples of simulations of 3D poly-disperse material characterised by a higher ratio α may be found in [11] ($\alpha = 10$, but the number of particles is relatively small, $n = 500$), [10] ($\alpha = 5$) and [40] ($\alpha = 2$). Limitations of poly-dispersed material probably result from the decreased efficiency of the contact search, caused by the increase of α. A brief explanation of the above-mentioned difficulties related to sub-division into cells and contact search may be found in the work of Kohring [41]. Attempts to overcome these difficulties by introducing modified sort-type contact detection algorithms were presented by Schinner [42] and Perkins and Williams [43], where the applications were limited to 2D polygonal particles.

In the present research, parallel DEM software based on the spatial domain decomposition strategy has been developed for simulating granular material on the distributed memory PC clusters. A particular manifest of this paper is twofold: to investigate computational performance of the developed algorithms and software, and to contribute to the understanding of algorithmic aspects related to poly-disperse properties of heterogeneous granular material. The numerical illustration addresses tri-axial compaction by rigid walls.

The paper is organised as follows. In Section 2, the methodology and governing relations of the discrete element method applied to the dynamic behaviour of non-cohesive visco-elastic frictional granular material are described. Section 3 discusses the implementation of parallel algorithms in the general DEMMAT code. In Section 4, the compaction problem and material data are described. Numerical results and computational performance are presented in Section 5, while the concluding remarks are given in Section 6.

2. Governing relations and computational procedure

Granular material is regarded as a system of the finite number N of spherical particles, characterised by radii R_i ($i = 1,\ldots,N$) and the prescribed material properties. The particles can change their position under the action of free rigid body motion or contact with the

neighbouring particles or walls. The deformation of the particles is replaced by the overlap of particles and it is assumed that impact between a pair of particles i and j has no influence on the contact between the impacting particles of another pair. A characteristic size of the overlap depth defined by h_{ij} must be much smaller than the particle size.

The dynamical behaviour of an individual particle i is considered by applying the Newton's second law. Three translations and three independent rotations expressed in terms of the forces and torques at the centre of the particle are as follows:

$$m_i \frac{\mathrm{d}^2 \boldsymbol{x}_i}{\mathrm{d}t^2} = \boldsymbol{F}_i \tag{1}$$

$$I_i \frac{\mathrm{d}^2 \boldsymbol{\theta}_i}{\mathrm{d}t^2} = \boldsymbol{T}_i \tag{2}$$

where m_i, and I_i are mass and inertia moments, while vectors $\boldsymbol{x}_i$ and $\boldsymbol{\theta}_i$ initiate the position of the particle centre and the orientation of particle i, respectively. Vectors $\boldsymbol{F}_i$ and $\boldsymbol{T}_i$ present the sum of external force $\boldsymbol{F}_{i,ext}$, contact force $\boldsymbol{F}_{i,cont}$, and gravity force $\boldsymbol{F}_{i,grav}$ as well as the corresponding torques.

The time-driven (TD) discrete element method as originally proposed in [1], is explored to simulate the behaviour of granular material. This method is better suited for a longer time of particle collision than for free pass of particles. The main focus of the TD methods is on evaluation of contact forces:

$$\boldsymbol{F}_i = \sum_{j=1, j\neq i}^{N} \boldsymbol{F}_{ij} + m_i \boldsymbol{g} \tag{3}$$

$$\boldsymbol{T}_i = \boldsymbol{T}_{i,contact} = \sum_{j=1, j\neq i}^{N} \boldsymbol{T}_{ij} = \sum_{j=1, j\neq i}^{N} \boldsymbol{d}_{cij} \times \boldsymbol{F}_{ij} \tag{4}$$

where $\boldsymbol{d}_{ij}$ is particle geometry-dependent vector, pointing from the particle centre to contact centre.

A methodology of calculating the forces (3) and (4) depends on the particle geometry and mechanical properties as well as on the constitutive model of the particle interaction. The presented inter-particle contact model considers a combination of elasticity, viscous damping and friction force effects. Actually, the contact between two material particles is modelled by a spring and dashpot in both the normal and tangential directions and an additional slider in tangential direction.

Hence, the inter-particle force vector $\boldsymbol{F}_{ij}$ describing the contact between the particles i and j may be expressed in terms of normal and tangential components $\boldsymbol{F}_{n,ij}$ and $\boldsymbol{F}_{t,ij}$, respectively. The normal component $\boldsymbol{F}_{n,ij}$ presenting, actually, a repulsion force comprises elastic and viscous ingredients. The tangential component $\boldsymbol{F}_{t,ij}$ reflects static or dynamic frictional behaviour. The static force describes friction prior to gross sliding and comprises elastic and viscous ingredients, while the dynamic force describes friction after gross sliding and is expressed by the Coulomb's law. Inter-particle friction is defined by internal friction coefficient μ which may be different, depending on whether particle–particle or particle–wall contact is considered.

For evaluating the contact forces (3), all contacts between the particles and their neighbours must be detected. Generally, contact detection problem is of the size $O(N^2)$ for the system containing N particles. In order to reduce the number of all particle pair combinations, a simple cellular decomposition known as *link-cell* method [44] was used for contact detection. A three-dimensional domain of the granular media is divided into cubic cells of the size slightly larger than the diameter of the largest particle. The neighbour-searching algorithm comprises referencing of individual particles to the cells and constructing of the *neighbour list of particles* by searching only over neighbouring cells. In a 3D case, the reference cell is surrounded by up to 27 neighbouring cells. The *boundary neighbour list of particles* was constructed in the manner described above. A detailed description of the DEM technique applied may also be found in [45,46].

The dynamical state of granular material is determined by numerical integration of Eqs. (1) and (2). In order to find a reasonable compromise between accuracy and computational efficiency, explicit one-step or predictor–corrector integration schemes are mainly used [47]. The earliest review of integration methods may be found in [11]. In order to achieve the accuracy of numerical integration schemes, unidirectional two particle motion tests were examined and the results were compared with analytical solutions. A comparison of different time integration schemes by using a relatively small time step [48], i.e. *Newmark*, *Taylor Expansion Series* expansion, *Velocity Verlet* and *5th-order Gear predictor–corrector*, showed that the highest accuracy was achieved by the 5th-order Gear predictor–corrector scheme. Later investigations [49] revealed, however, a tendency of higher-order schemes to require smaller time steps and be more sensitive to instabilities thought not necessarily computationally efficient.

The predictor–corrector scheme represents a two-step procedure. Let us denote the time-dependent variables, positions $\boldsymbol{x}_i$, velocities $\boldsymbol{v}_i = \mathrm{d}\boldsymbol{x}_i/\mathrm{d}t$, accelerations $\boldsymbol{a}_i = \mathrm{d}^2\boldsymbol{x}_i/\mathrm{d}t^2$ and the higher-order time derivatives $\boldsymbol{b}_{3i} = \mathrm{d}^3\boldsymbol{x}_i/\mathrm{d}t^3$, $\boldsymbol{b}_{4i} = \mathrm{d}^4\boldsymbol{x}_i/\mathrm{d}t^4$ and $\boldsymbol{b}_{5i} = \mathrm{d}^5\boldsymbol{x}_i/\mathrm{d}t^5$ of particle i by vector $\boldsymbol{y}_i = \{\boldsymbol{x}_i, \boldsymbol{v}_i, \boldsymbol{a}_i, \boldsymbol{b}_{3i}, \boldsymbol{b}_{4i}, \boldsymbol{b}_{5i}\}^{\mathrm{T}}$. The new value variables at time increment $t + \Delta t$ are predicted by a simple series expansion up to a desired order of accuracy

$$\boldsymbol{y}_i^p(t + \Delta t) = \boldsymbol{y}_i(t) + \Delta \boldsymbol{y}_i^p(\Delta t) \tag{5}$$

Here, incremental vector $\Delta \boldsymbol{y}_i^p(\Delta t)$ presents the required terms of the expansion series. Then, according to the new positions and velocities, the particle forces and accelerations are corrected and acceleration increment $\Delta \boldsymbol{a}_i$ is updated. Finally, the vector of particle variables is corrected as follows:

$$\boldsymbol{y}_i^c(t + \Delta t) = \boldsymbol{y}_i^p(t + \Delta t) + \Delta \boldsymbol{y}_i^c(c_j, \Delta t, \Delta \boldsymbol{a}_i) \tag{6}$$

Here, the correction vector $\Delta \boldsymbol{y}_i^p$ is calculated by using the given integration constants c_j. For details [11] may be referred.

3. Parallel algorithms

Parallel algorithms have been implemented in the FORTRAN 90 code DEMMAT_PAR developed in the Parallel Computing Laboratory of Vilnius Gediminas Technical University. Parallelization of the software is based on the domain decomposition, which has been established as one of the most efficient high level (coarse grain) strategies for scientific and engineering computations, also offering a generic solution for both shared and distributed memory computers. The solution domain is divided into nearly equal sub-domains. Each processor computes only the forces and updates the positions of particles in its sub-domain. In order to perform their computations, the processors need to share information about the particles which are near the division boundaries in nearby sub-domains. The communication required in the implemented algorithm is thus local in nature and a large portion of the sequential code can be used without any modification. Inter-processor communication is implemented in the DEMMAT_PAR by the subroutines of message passing library MPI [25]. Partitions containing a roughly equal number of particles ensure the static load balancing on the homogeneous PC clusters.

The parallel algorithm is designed as follows. Initially, a three-dimensional domain of the granular media is divided into cubic cells. The pre-processor assigns the cells containing particles to processors. The particle data is distributed among the appropriate processors and the setup of the problem is finished. The most CPU time consuming parts of the program are time integration and

computation of contact forces (3) and (4). The implemented spatial domain decomposition perfectly parallelizes the time integration performed in the time loop without any inter-processor communication. Each processor independently computes Gear's predictor (5) and Gear's corrector (6) steps by using locally stored data. The computation of contact forces needs contact detection as well as inter-processor communication. The link-cell method employs an efficient neighbour-searching algorithm and fast construction of the *neighbours list of particles*. Each particle i is referenced to the cell according to the position of the particle centre of gravity $\boldsymbol{x}_i$. After referencing the *neighbours list of particles* is constructed by assembling particles indices from the neighbouring cells that are around the reference cell. Some neighbouring cells might belong to another processor. The particle data from such buffer cells should be exchanged among neighbouring processors.

The initial portion of communications is performed after completing the Gear's predictor step, when processors exchange particles as they move from one sub-domain to another. The inter-processor communication (Fig. 1) is performed in the subroutine *Communicate*. The list of indexes of exchanging particles serves as the main input parameter. The communication is performed in two steps encoded by two subroutines: *InitialCommunication* and *MainCommunication*. The initial communication is carried out in two loops over the neighbouring processors. A number of exchanging particles *part_amount_to*(i) is sent to each of the neighbouring processors. It is received and used as the main parameter for the conditions governing the program flow and for the subsequent buffer preparation. The data buffers *Send_Buffers*(i) are formed and sent in another loop over the neighbouring processors. This work is optional, being performed only in the case of a non-zero number of the exchanging particles. The receive operations are performed in a similar manner. In such a way, the data of the exchanging particles, e.g. global index, radii, mass, co-ordinates, velocities, accelerations, shear slip and additional information are transferred to another processor and incorporated in its local data structure. The main part of the described inter-processor communication is optional, requiring sending–receiving a small amount of data, but it needs extensive manipulation on data structures.

The main portion of communications is performed prior to computing contact forces in order to exchange particle data from the buffer cells. The employed communication model is created for grid networks. First, processors communicate in left–right directions, then, in up–down directions. Each left processor sends particles data to the right processor, and, then, every left processor receives similar information from the right processor. Up and down processors communicate in the same way. The particles received from the buffer cells located in the sub-domain corners are also sent in the next direction, because they are necessary for all three neighbouring processors. The received particle co-ordinates are placed as contiguous data directly into the arrays containing local particles. No time is spent for rearranging the data, except for creating the buffered messages for inter-processor communication. The inherent synchronisation of this message passing algorithm ensures good performance of parallel computations on distributed memory PC clusters. The communications are performed by the MPI routines MPI_ISEND, MPI_REQUEST_FREE and MPI_RECEIVE. The previous version of the code [50] employed blocking MPI_SEND. The non-blocking communication routines significantly improve parallel efficiency of the code.

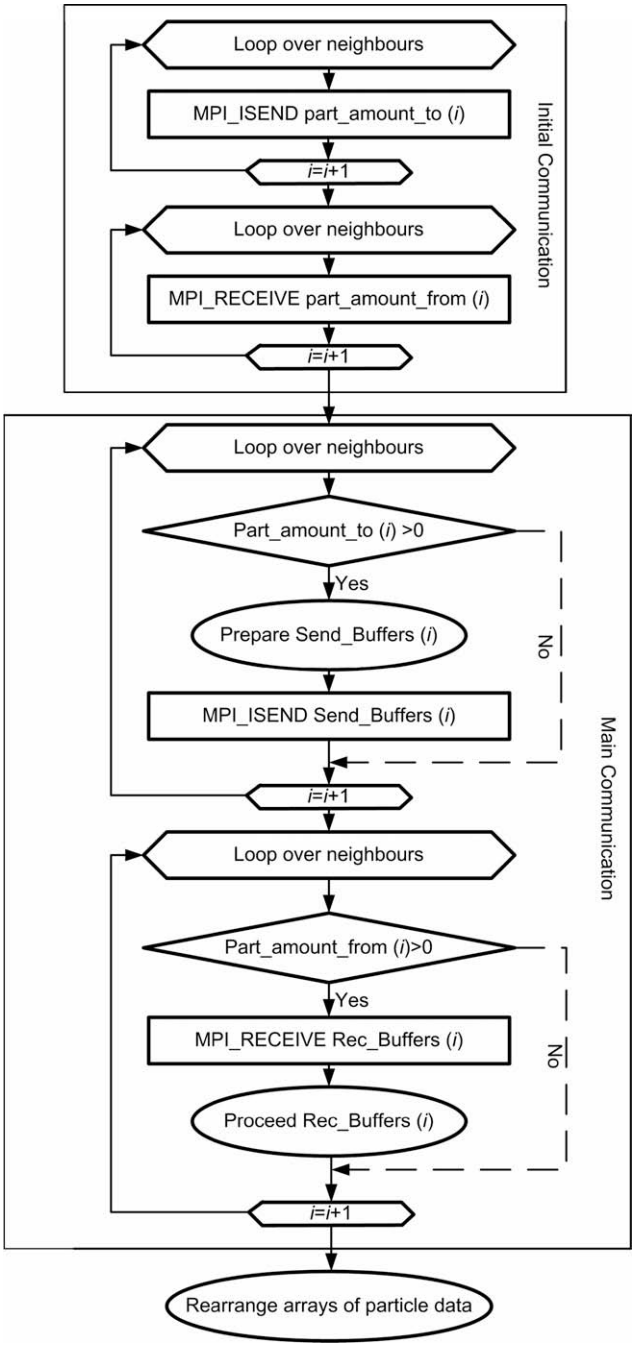

Fig. 1. The parallel algorithm for particles changing sub-domains/processors.

4. Compaction problem and material

The tri-axial compaction of poly-dispersed granular material is considered by the DEM. The three-dimensional computational domain imitates a representative macroscopic volume element containing particles and presents a box in the form of rectangular parallelepiped (Fig. 2). In particular, the dimensions of the box sides H and a are equal to those usually used for standard specimens, assuming that the ratio of longitudinal and cross-sectional dimensions $H/a = 4$ (hence, $H = 160$ mm, $a = 40$ mm). The compaction is performed by the motion of rigid walls and is controlled in time t by a constant rate v_n, $v_n = 0.01$ m/s.

The material represents an assembly of spherical non-cohesive visco-elastic frictional particles. Basic physical parameters of the particles used throughout simulation are presented in Table 1.

It should be noted, that the microscopic data of particles ρ, E and ν are assumed to be the same as the macroscopic values for cement pasta for refractory concrete. Here, the viscous damping coefficients γ also play the role of computational stabilization parameters, while their values are prescribed artificially on the

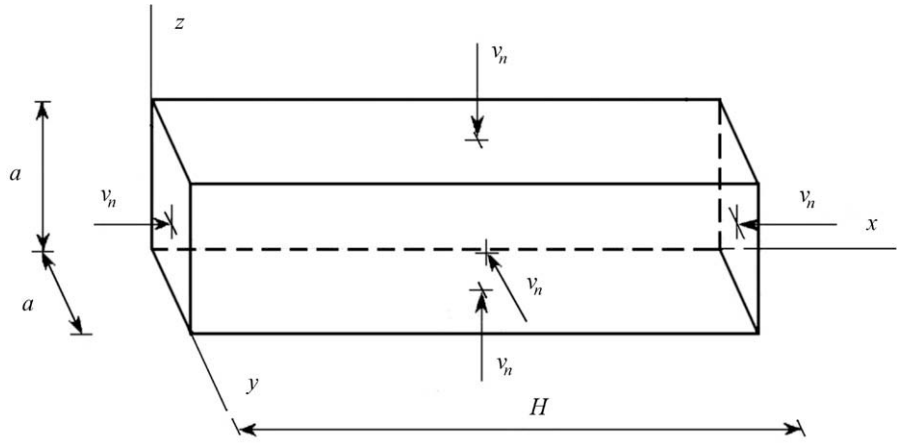

Fig. 2. Illustration of a compaction problem.

basis of values given in references and relying on personal computational experience. Frictional properties are defined by particle–particle friction coefficients μ exhibit a relatively rough material with the perfect sliding on wall.

In a framework of current investigation, poly-dispersive character of the composition of particles of different sizes is described by the heterogeneity ratio between maximal and minimal diameters of particles

$$\alpha = \frac{d_{\max}}{d_{\min}} \tag{7}$$

and the normal distribution law. Finally, material is defined by the up-scaled size distribution curve and given characteristic particle diameter.

Two poly-dispersed materials, *Poly*1 with $\alpha = 4.3$ and *Poly*2 with $\alpha = 9.0$ will be used in the below simulations. Size distribution curves used for particle design are plotted in Fig. 3.

Table 1
Physical data of particles.

Quantity	Value
Particle density (ρ)	2300 kg/m^3
Young's modulus (E)	13 GPa
Poisson's ratio (ν)	0.2
Normal damping coefficient (γ_n)	150 s^{-1}
Tangential damping coefficient (γ_t)	100 s^{-1}
Particle–particle friction coefficient (μ)	0.3
Particle–wall friction coefficient (μ_w)	0.0

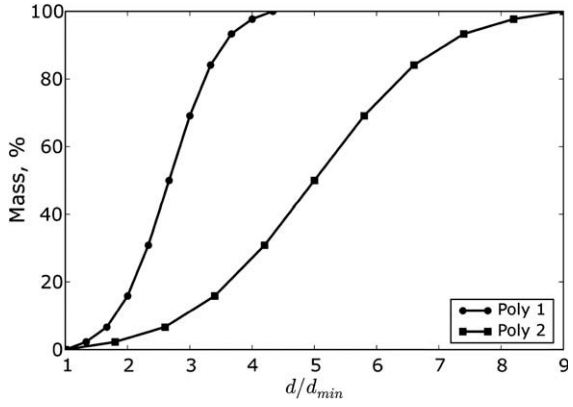

Fig. 3. Size distribution curves of poly-dispersed materials plotted in relative scale.

For generating the required particle composition and for locating particles in the given initial space the algorithm presented by Jiang et al. [51] was utilized. The algorithm works as follows. Each particle newly generated according to distribution is randomly placed in a specified volume. If a particle overlaps the neighbouring particles, then this particle is moved in the opposite direction to overlaps. The replacement and movement of particle is repeated until the particle does not overlap other particles. The algorithm runs iteratively and after some trial and error cycles a relatively dense contact-less particle composition is obtained. It should be noted that the presented algorithm is rather sophisticated and the appropriate experience is required to generate the required particle composition. The number of particles or the particle volume may be controlled within the prescribed limits.

5. Numerical results and discussion

The potential of the discussed algorithm and the developed parallel code DEMMAT_PAR as well as impact of poly-dispersive character of granular material were investigated by solving previously described tri-axial compaction problem. A series of benchmark tests examining parallel speed-up and computational performance were performed on the PC cluster VILKAS (NPACI Rocks Cluster, RedHat Linux Enterprise 3.0) at Vilnius Gediminas Technical University. The cluster consisted of 20 processors (Intel Pentium 4, 3.2 GHz, 1GB RAM for a processor). It was connected by D-Link DGS 1224T Gigabit Smart Switch (24-Ports 10/100/1000 Mbps Base-T Module).

5.1. Description of the benchmark problems

Two benchmark problems denoted hereafter by A and B, respectively, were solved. In particular, the solution domain of each individual problem is expected to be filled with approximately 20,000 and 100,000 particles. To ensure the dynamic character of moving particles and to avoid local effects, tests were executed by keeping the fixed relatively large number of time steps $N_t = 200{,}000$.

Two packed sets of particles representing poly-dispersed material *Poly*1, termed hereafter as *Poly*1A and *Poly*1B, were compiled in the box by applying the above algorithm. Because of the implicit character of the algorithm, the set *Poly*1A, containing 19,890 particles with $d_{\min} = 1.031$ mm and $d_{\max} = 4.466$ mm, and the set *Poly*1B, containing 100,037 particles with $d_{\min} = 0.603$ mm and $d_{\max} = 2.613$ mm, were composed. By assuming 15 particle fractions, the obtained size distributions for the both sets of particles are illustrated by the graphs, presented in Fig. 4.

Two sets of the mono-disperse material, termed hereafter as *Mono*A and *Mono*B, were also considered for the sake of comparison. They were compiled of 19,890 particles with $d = 2.22$ mm and of 100,037 particles of the diameter $d = 1.33$ mm, respectively. The initial composition of the particles presents a homogeneous orthogonal lattice-type structure, where particles are embedded into the centres of the cells to ensure that they are not in contact at the beginning of motion. Careful treatment of the initial configuration is obligatory because the number of particles is smaller compared to the number of cells in the grid.

Once the particles are generated, the user has the option to take a decision on this problem-dependent issue associated with the problem how to design link-cell grid and sub-domains and to assign sub-domains to different processors. Proper domain decomposition is a key task for DEM computations. As follows from the developed algorithm, the most important question for the static domain decomposition is how to divide the solution domain to sub-domains, ensuring the requirements of the load-balance and minimal inter-processor communication. The present approach explores a regular space-cell sub-division scheme and assigns particular cells to desired sub-domains. The size of the grid cells depends, however, on the composition of the material and is predicted by the maximal particle diameter. It should be noted, that the unique cell grid is explored for generation of a different number of sub-domains. Taking into account the rectangular shape of the investigated solution domain, the uniform grid structure of sub-domains is considered and sub-division of the column-like domain into sub-domains is chosen *a priori*. A maximal number of sub-domains considered here is 8 for the set A with approximately 20,000 particles and 10 for the set B with approximately 100,000 particles. Characteristic data of the cell grid and particles considered are given in Table 2.

In the case of mono-disperse material, the uniform cell grid geometry simply coincides with initially generated particles. On the contrary, for poly-disperse material it is a burden because the characteristic cell size is limited by the maximal particles diameter. It is easy to observe, that poly-disperse material is characterised by a significantly larger number of particles per cell.

The values of the required time integration step Δt, predefined by the size of the smallest particle, are also exhibited in Table 2. Generally, the required time step is relevant to the particle contact period. For the explicit integration schemes, it is obtained in terms of the particle mass m and stiffness k by equation

$$\Delta t = \frac{\pi}{\beta}\sqrt{\frac{m}{k}}$$

Regarding the suggestions of different authors summarised in [11], the factor β ranges between 10 and 50, while, in our examples, $\beta = 20$ is assumed. It is easy to prove that, for the linear contact of spherical particles, Δt is proportional to the particle size.

5.2. Parallel speed-up tests

The parallel performance of the code was evaluated by measuring the speed-up S_p:

$$S_p = \frac{t_1}{t_p} \tag{8}$$

where t_1 is the program execution time for a single processor; t_p is the wall clock time for a given job to execute on p processors. Parallel speed-up is measured by fixing the number of particles and increasing the number of the processors used.

The results of parallel performance tests are presented in Fig. 5. The program execution time and the speed-up as a function of the number of processors are shown here. When the number of processors is small, the speed-up is close to ideal for mono-disperse systems, or even higher (set *Poly*1B). The small reduction of the efficiency owing to communication overhead is obtained for a larger number of processors. Generally, the speed-up is largely determined by the ratio of local computations over inter-processor communications. As the number of processors increases, for a fixed size problem, the communication cost will eventually exceeds the local computation cost after a certain stage. This high ratio of communication to computation makes a further increase of efficiency very small. It was observed, that the communication cost is quite small for relatively large problems (set B with 100,037 particles), where a large number of particles and cells are used per processor. Compared to previous developments [50], the implemented non-blocking communication improved the speed-up from 7 to 9 for 10 processors. The presented results show that the implemented inter-processor communication algorithms are well designed for the distributed memory PC clusters.

In practice, ideal efficiency is naturally not attained because of an inherent sequential part of the algorithm, a load imbalance and parallel communication overhead. Good load balancing is insured by proper static domain decomposition discussed earlier. The number of particles is nearly equal in all sub-domains, therefore, initially, the processors perform nearly the same amount of work. During the investigated time interval, the numbers of particles remain almost constant in all processors. Thus, the load is well balanced during the performed benchmark tests.

In most of computer clusters, the inter-processor communication is performed significantly slower than local computations. In DEM computations, the communication routines should transfer a large amount of data because each processor needs information about all particles in buffer cells of its neighbours. Fig. 6 illustrates

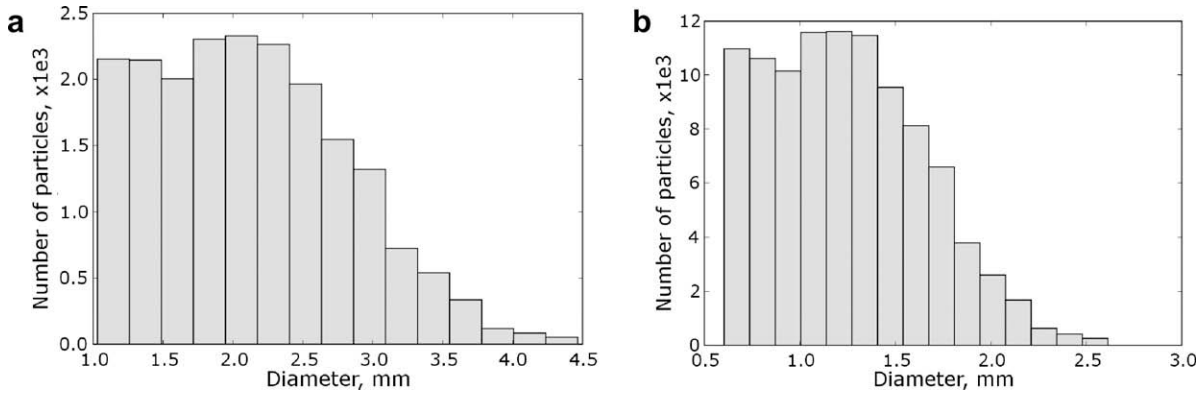

Fig. 4. Histograms of particle size distributions: (a) set *Poly*1A and (b) set *Poly*1B.

Table 2
Basic characteristics of the cell grid.

Material set	*Poly*1A	*Mono*A	*Poly*1B	*Mono*B
Number of particles	19,890	19,890	100,037	100,037
Total number of cells	2048	23,328	13,500	108,000
Number of cells in longitudinal direction	32	72	60	120
Number of cells in transverse direction	8	18	15	30
Number of buffer cells	64	324	225	900
Average number of particles in cell	9,71	0.85	7,41	1.0
Maximal number of particles in cell	20	2	15	2
Cell size (mm)	5.0	2.222	2.667	1.333
Time step Δt (10^{-7} s)	0.342	1.370	0.200	0.801

time history of data transfer during time period $T = 0.02$ s for poly-dispersed set *Poly*1A and a corresponding mono-dispersed set *Mono*A measured by solving the problem on 4 processors.

Variable D presents the data sent and received by each processor. Four processors divide the solution domain into four equal cubic sub-domains, which is a very convenient sub-division, employing a minimal area of sub-domain inter-connection. The number of particles, as well as the corresponding data transfer, are the same in the appropriate processors for mono-dispersed set MonoA, therefore, 0Mono curve represents the data transfer for one of the processors working on two boundary sub-domains, while 1Mono curve illustrates that for one of the processors working on two middle sub-domains. The number of particles is slightly different for poly-dispersed set *Poly*1A, therefore, the data transfer is represented by four individual curves. In two boundary" processors (0Poly, 3Poly), the data transfer is lower than in two processors working with the centre of the solution domain (1Poly, 2Poly). The data transfer of a smaller poly-dispersed system is significantly larger than that of the corresponding mono-dispersed system. The insignificant fall of speed-up (Fig. 4b) in the case of a poly-dispersed set *Poly*1A is caused by intensive inter-processor data transfer. Thus, heterogeneity of material can contribute to parallel efficiency of the code, but its influence was insignificant in the performed experiments.

The direct comparison of the obtained results with other data seems to be complicated. Most of the authors report parallel efficiency obtained on various supercomputers. Hustrulid [52] tested his original algorithm for contact detection including specific data structures on Transputer T805. The obtained speed-up ($S_p = 6$ on 16 processors and $S_p = 8$ on 32 processors) is significantly lower than that attained in the present work. The NSCD solver was developed for shared memory computers (SUNFIRE 880 and SGI origin 3800) and was described in the paper [38]. The obtained speed-up ($S_p = 11$ on 16 processors) can hardly be compared with the present results because different parallelization strategy and hardware are used. Qwen and Feng [8] presented the dynamic domain decomposition algorithm for shared memory machine SGI Origin 2000. Parallel efficiency was tested on 6 processors only ($S_p = 4.41$ or $S_p = 5$). Dowding et al. [36] developed an efficient NURBM3DP algorithm for distributed memory computers. The excellent speed-up ($S_p = 9.89$) was attained on 10 processors of IBM SP2 ($S_p = 8.81$ in the present work). The limited tests were carried out on 2 processors of DELL PC. The obtained speed-up $S_p = 1.89$ is comparable with the values ($S_p = 1.81$, $S_p = 2.02$ or $S_p = 2.23$) attained in the present work, where the speed-up equal to 8.81 has been obtained on 10 processors for poly-dispersed material and the speed-up equal to 9.22 has been attained for mono-dispersed material (also 10 processors). Thus, the parallel efficiency reported in the present paper can compete with that obtained on supercomputers and reported by other authors.

5.3. Computational performance

Poly-dispersive properties of material directly influence the computational performance. Fig. 5a illustrates that poly-dispersed systems, consisting of a smaller number of particles than mono-dispersed systems, require significantly larger computing time. Simulation of the poly-dispersed 100,037 particle system on one processor consumes much longer computing time than the mono-dispersed system (by 345%). The higher computational cost of poly-dispersed material may be attributed to a limited ability to generate a grid, consisting of cells of suitable size. The increased cell size reduces the number of buffer cells, therefore, the lists of possible contacts become significantly longer, as well as the com-

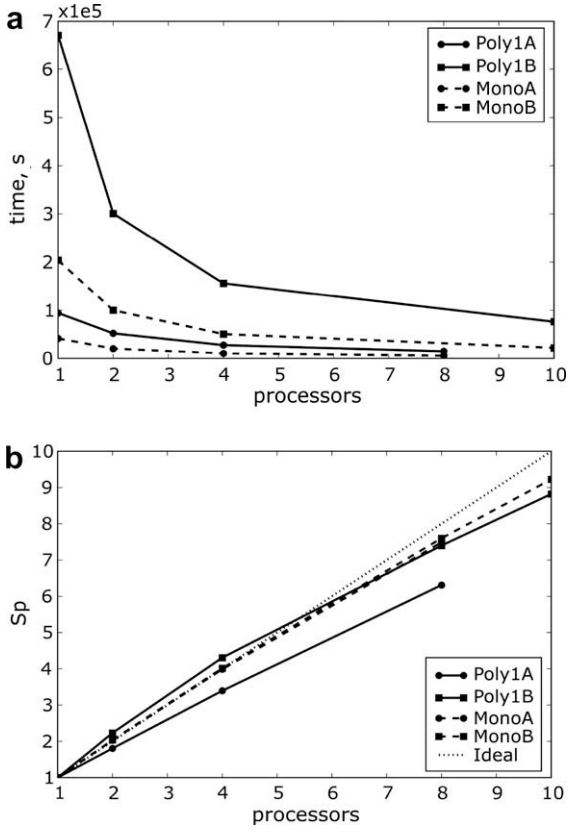

Fig. 5. Parallel performance obtained solving mono-dispersed particle systems (Mono) and poly-dispersed particle systems (Poly1), consisting of different numbers of particles: (a) the run time and (b) the speed-up.

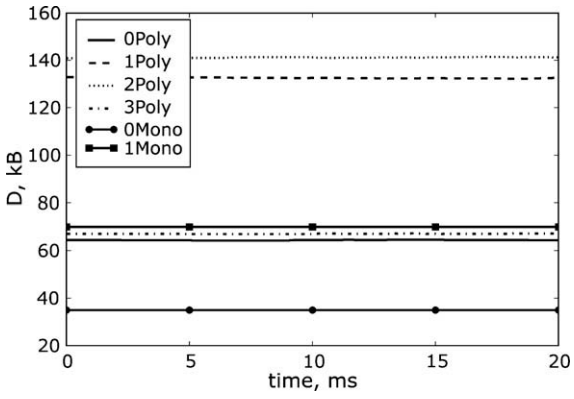

Fig. 6. Inter-processor data transfer D measured in solving a poly-dispersed set *Poly*1A and a corresponding mono-dispersed set *Mono*A by 4 processors.

puting time for contact detection. An indirect proof follows from the simple analysis of data presented in Table 2.

Additional expenses, reflecting particle heterogeneity, may be evaluated by the suggested increase factor

$$V_p = \frac{TP_p}{TM_p} \tag{9}$$

obtained for a different number of processors p. Here TP_p stands for the program execution time for poly-dispersed, while TM_p denotes mono-dispersed material. It can be easily seen that for the mono-dispersed material $V_p \equiv V_{Mp} = 1$. The variation of factor V_p against the different number of processors is plotted in Fig. 7. Examination of factor V_p shows, that expenses for poly-dispersed systems containing a smaller number of particles (19,890) increased up to 3 times ($V_p = 1.3$), while for a higher number of particles (100,037) this increase rose up 3 times ($V_p = 3.0$). This factor remains almost constant for the higher number of processors.

Thus, heterogeneity of material considerably contributes to computing time of the benchmark tests. The impact of the poly-dispersive properties is further examined by considering the material *Poly*2 with higher heterogeneity factor $\alpha = 9.0$. The size distribution law of particles was shown in Fig. 3, while the variation of particle diameters was limited by $d_{\min} = 0.55$ mm and $d_{\max} = 4.75$ mm. The packed set, termed hereafter as *Poly*2A, containing 20,004 poly-dispersed particles, was compiled and a benchmark test, keeping the fixed of time steps $N_t = 200{,}000$, was performed on 8 processors.

Rise up of computational expenses in terms of increase factor V_8 on 8 processors and its comparison with the corresponding sets *Mono*A and *Poly*1A are illustrated in Fig. 8a. Here, a significant increase (up to 7 times) compared to mono-dispersed material can be observed. Taking into account that benchmark simulations were performed with different time steps for different materials, the increase factor is not applicable to evaluate the increase of computational expenses in the real time scale. The new variable real time increase factor VR_p is defined as follows:

$$VR_p = \frac{TP_p}{N_t \Delta t} \tag{10}$$

and is obtained for a different number of processors p. It depends on the number of time steps N_t performed in the benchmark test and time strep Δt. A comparison of real time increase factor VR_8 for mono-dispersed set *Mono*A and poly-dispersed sets *Poly*1A, *Poly*2A obtained on 8 processors is graphically illustrated in Fig. 8b. Here, a dramatic increase, up to 31 times, compared to mono-dispersed material is shown.

The above result gives a rather pessimistic message, concerning applications of DEM to highly heterogeneous poly-dispersed materials in the nearest future. It seems reasonable to revise a standard DEM approach in the case of particles differing in the order of the size magnitude, to modify the most costly standard contact search link-cell procedure, to apply multiple time steps, etc. This task would be regarded as a future work. In spite of the appropriate pessimism, parallelization remains to be a very promising tool, drastically reducing the computing time by the factor almost proportional to the number of processors.

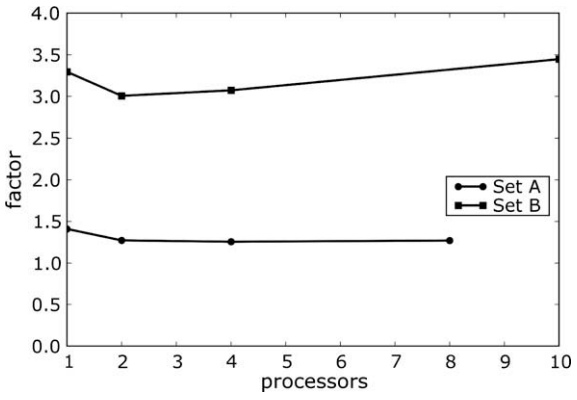

Fig. 7. Up-scaled computational expenses of poly-dispersed material.

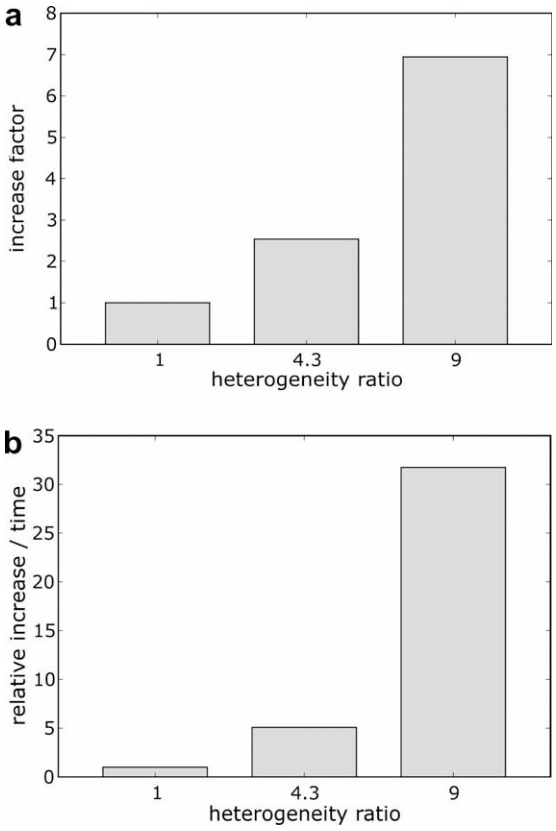

Fig. 8. Comparison of computational expenses for mono-dispersed set *Mono*A and poly-dispersed sets *Poly*1A and *Poly*2A obtained on 8 processors: (a) increase factor V_p and (b) real time increase factor VR_p.

5.4. The influence of the microscopic behaviour

The parallel performance results gained by the benchmark tests were obtained during short-term computations. The test methodology disregards the microscopic state of the particle behaviour. Actually, the measured performance is relevant to the contact-free state of particles. It is well known, that particle composition and internal field variables undergo dramatic changes during long-term action. A particular illustration reported in [12] exhibits variation of computational expenses against the number of possible contacts. Therefore, the influence of the microscopic behaviour should be considered in predicting long-term computational expenses, especially, those, regarding poly-dispersed material.

Long-term benchmark tests with newly generated sets of poly-disperse material *Poly*1 and mono-disperse material were conducted to examine the influence of the microscopic particle behaviour. To ensure physical relevance of both experiments, two sets of particles with equal volumes $V = 134\ \text{cm}^3$, providing the initial packing density (hereinafter D) or volume fraction of 0.52 were compiled. The packed set, termed hereafter as *Poly*1C, containing $N = 19{,}890$ particles with $d_{\min} = 1.031$ mm and $d_{\max} = 4.466$ mm, represented poly-dispersed material *Poly*1. The mono-disperse

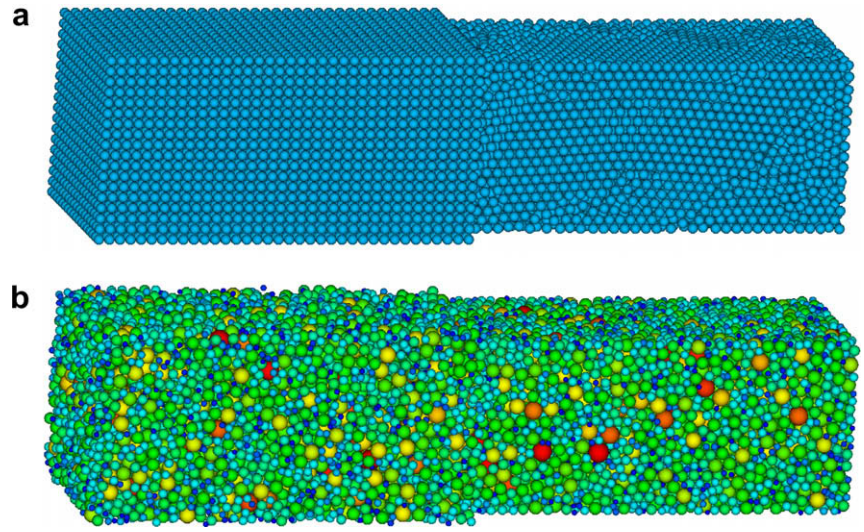

Fig. 9. Views of the initial (left side) and compacted (right side) specimens: (a) mono-disperse material and (b) poly-disperse material.

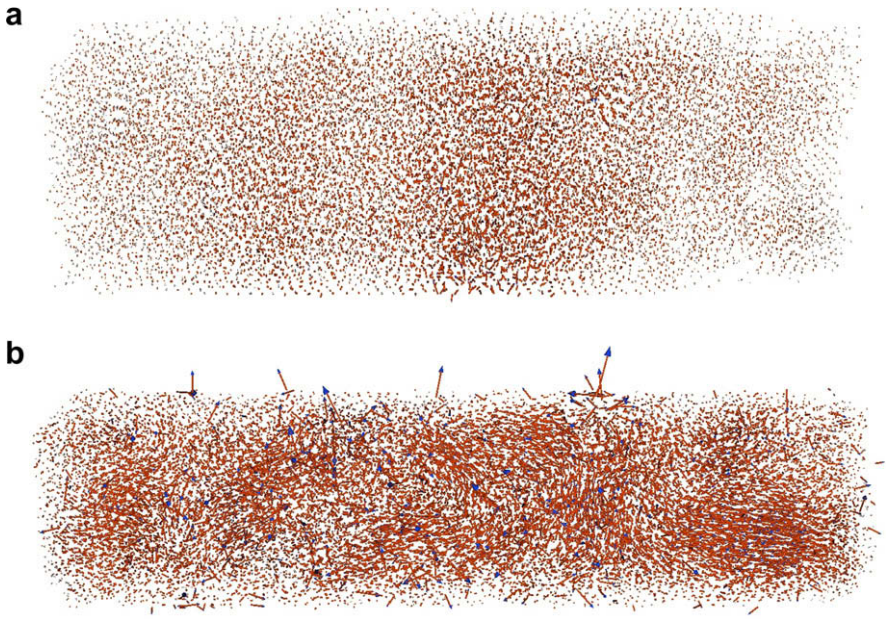

Fig. 10. Illustration of the particle velocity vectors: (a) mono-disperse material and (b) poly-disperse material.

material was represented by a set *Mono*C, containing $N = 23{,}350$ particles with $d = 2.22$ mm in the form of the 3D grid structure.

The total compaction simulation time T was equal to 0.28 s, while the motion of each of the walls was stopped by final displacement $u = 0.28$ cm responding to 5.60% of the lateral deformation of the initial material volume. To reach this state, $N_t = 4{,}080{,}000$ time integration steps with $\Delta t = 0.0685$ μs and $N_t = 2{,}040{,}000$ time integration steps with $\Delta t = 0.137$ μs were performed on 8 processors for *Poly*1C and *Mono*C sets, respectively.

The compaction process of particles may be qualitatively illustrated by the structure formed by particles. The views of the initial (left side) and compacted (right side) specimens containing the mono-disperse and poly-disperse materials are shown in Fig. 9.

Fig. 9a illustrates a regular initial configuration of mono-dispersed material. During the compaction process, the rectangular cells have a tendency to be transformed into the packed configuration, yielding several locally organised regions, involving the typical quasi-regular pyramidal (triangular) lattice of crystalline packing and the disordered structures. The disordered structure of poly-dispersed material is indicated in Fig. 9b).

During the compaction of the poly-disperse material smaller particles changes their position more intensively than larger particles. The so-called clusterisation phenomenon is more clearly indicated by the particle velocity vectors (Fig. 10). The graphs exhibit intensified motion of poly-dispersed material, where they are mainly concentrated at the locations involving small particles.

The physical nature of compaction may be quantitatively illustrated by the evolution of the coordination number (an average number of contacts per particle) and material packing density (the fraction of the volume occupied by the particles to the total volume of the region considered). The above variables have been extensively investigated in characterisation of the compaction of granular packing [13,14]. The evolution of the coordination number versus time is plotted in Fig. 11. The graphs shown clearly illustrate the microscopic behaviour of granular materials and phase change during compaction.

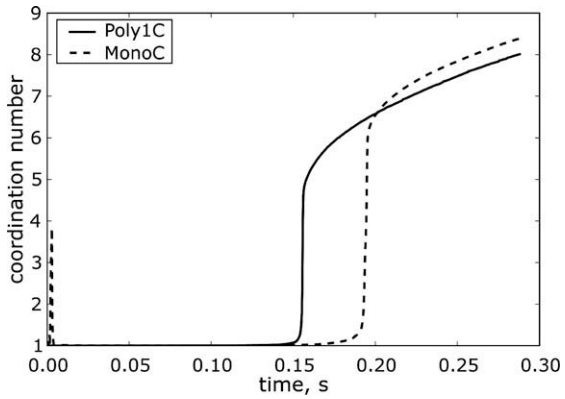

Fig. 11. Illustration of time histories of coordination number.

As a matter of fact, the entire compaction period T may be divided into two phases. The first phase, characterised here as a contact-free particulate phase, or gas-like phase, was initiated by packing density of the order of 0.52 and lasted till time t_1 was reached. In this phase, the densification is related to the chaotic motion of particles. The local peak of coordination number was observed for mono-dispersed material at the beginning of compaction (see Fig. 11) may be explained as the first attempt of granular material to build the contact network structure. However, due to the available free space, left after particles generation and located at the bottom wall, all particles undergo avalanching rearrangement, but the contact-free motion of the particles still remains. Thus, this peak is attributed to initial imperfection of material imposed by the initial conditions.

The time, t_1 was obtained by considering the jump in the coordination number of flux specifying phase transition. Thus, the obtained time $t_{1p} = 0.156$ s, required to reach the solid phase with $D_{poly} = 0.59$ for poly-disperse material, is considerably shorter than this time equal to $t_{1m} = 0.195$ s for mono-dispersed material with $D_{mono} = 0.62$. This difference mainly shows that the initial conditions, however, were more favourable for compacting of the poly-disperse particles than the mono-disperse ones.

The second phase, called the solid phase, lasts between t_1 and the end time of simulation. It is characterised as the bulk solid phase, where particles come into contact with each other, while densification is possible due to the particles' rearrangement and microscopic deformation of the contacting particles. The solid phase is characterised, finally, by a stable structure of particles forming a contact force network.

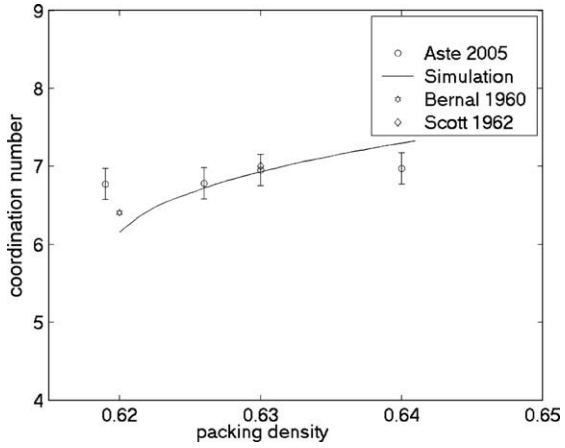

Fig. 12. Variation of the coordination number versus packing density.

Table 3
Relative values of the real time increase factor VR_8 obtained on 8 processors.

Type of tests	Poly-dispersed material *Poly*1	Mono-dispersed material
Short-term (contact-free particle behaviour)	5.08	1
Long-term (contacting with rearrangement)	6.53	1.32

It should be noted that well-defined mono-dispersed packing densities D, ranging from 0.58 to 0.64, can be reached by using simple techniques, involving the mono-dispersed particles purring into the container up to gently tapping or both tapping and slightly compressing [53]. The upper values of packing densities are addressed to the formation of crystalline-like structures and the exertion of the plastic deformations [53,54].

The quality of the numerical results was also examined by comparing them with the available experimental results. The experimental definition of the average number of the particles coordination is largely influenced by some degree of the uncertainty, since the exact information about the particles position and their shapes cannot directly be retrieved from the granular assembly. Recently, the most accurate studies on the structure of mono-disperse disordered packing at the grain level by means of the X-ray-tomography technique has been performed by Aste et al. [53,54]. They have statistically evaluated the particles coordination number. In Fig. 12, the comparison between various empirical data and the present simulation results is made for mono-disperse particles in the range of $D = 0.615$–0.64.

As it can be seen from this plot, the obtained values of the average coordination number correlate very well with the experimental values, indicating a realism of the numerical results for mono-sized material.

The influence of the microscopic behaviour on computational expenses is evaluated by comparing real time increase factor VR_8, obtained according to (10) in both short-term and long-term testing. Up-scaled relative comparison results are presented in Table 3. The results showed that the influence of the macroscopic behaviour is characterised by approximately 1.3 times increase of computational expenses required for real time simulations for both poly- and mono-sized materials. This increase is actually attributed to the increase of contacts during densification of material.

Detailed examination shows, however, that the instantaneous transformation of phases is not reflected in communication data, while a permanent increase may be relevant to evolutionary increase of possible contacts. Time variation of the average possible contact number relevant to link-cell algorithm is illustrated in Fig. 13.

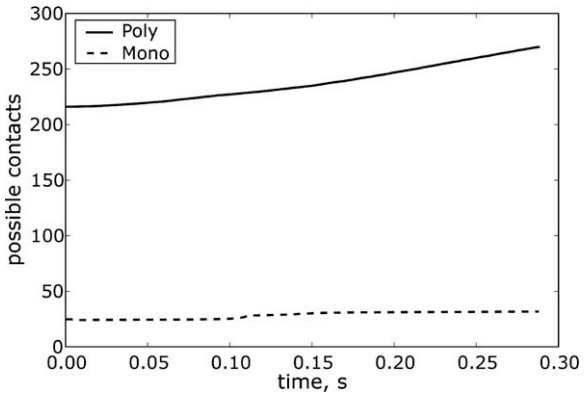

Fig. 13. Time histories of average possible contacts for one particle.

The curves (Fig. 13) show that the final increase of this quantity is approximately equal to the factor 1.3, correlating with the increase of real time simulation. The above numbers indicate, that the results of the numerical short-time benchmark tests have to be re-examined with respect to a physical nature of the simulation target.

6. Conclusions

The development of parallel DEM software and its application to the simulation of poly-dispersed granular material were considered. Based on the current investigation, some observations and concluding remarks may be drawn as follows:

- The measured parallel performance of the benchmark problems shows that the developed parallel algorithms are well designed for the distributed memory PC clusters. The speed-up equal to 8.81, obtained on 10 processors, can compete with that obtained by other researchers and reported in the literature.
- Increasing heterogeneity restricts, however, a sub-division of the computational domain into cells because a minimal cell size is limited by a maximal particle diameter. The obtained results show that the impact of the poly-dispersive character of material on parallel performance is not decisive. The increase of heterogeneity insignificantly affects the speed-up of parallel computations.
- On the contrary, the increase of heterogeneity as well as the number of particles and microscopic behaviour of material essentially contribute to computational expenses in terms of the CPU time. In summary, computational efficiency is relevant to the number of possible contacts, playing a decisive role in the link-cell contact search algorithm.
- A drastic increase of computational expenses still remains a serious obstacle in the simulation of large poly-dispersed material problems. However, intelligent handling of the particular problems, combining various simulation strategies, may improve numerical efficiency, while application of parallel technologies may be considered as the most attractive and realistic solution alternative.

Acknowledgement

Simulation and visualization of poly-dispersed granular material is partially supported by FP7 project BalticGrid-II (the contract number 223807).

References

[1] Cundall PA. A computer model for rock-mass behaviour using interactive graphics for the input and output of geometric data. In: Rep. AD/A-001 602: US National Technical Information Service; 1974.
[2] Cundall PA, Strack ODL. A discrete numerical model for granular assemblies. Geotechnique 1979;29(1):47–65.
[3] Luding S. From microscopic simulations to macroscopic material behaviour. Comput Phys Commun 2002;147:134–40.
[4] Luding S, Lätzel M, Volk W, Diebels S, Herrmann HJ. From discrete simulations to a continuum model. Comput Meth Appl Mech Eng 2001;191:21–8.
[5] Tomas J. Micromechanics of particle adhesion. In: CD-ROM proceedings of fifth world congress on particle technology WCPT2006; 2006.
[6] Moreno-Atanasio R, Antony SJ. Micromechanical behaviour of granular media: effects of contact stiffnesses. In: Topping BHV, Montero GP, Montenegro R, editors. Proceedings of the fifth international conference on engineering computational technology. Stirlingshire: Civil-Comp Press; 2006.
[7] Tüzün U, Heyes DM. Meso-scale modelling and measuring bulk cohesion in surface-wet granular assemblies. In: CD-ROM proceedings of fifth world congress on particle technology WCPT2006; 2006.
[8] Owen DRJ, Feng YT. Parallelized finite/discrete element simulation of multi-fracturing solids and discrete systems. Eng Comput 2001;18(3–4):557–76.
[9] Ibrahimbegovic A, Delaplace A. Microscale and mesoscale discrete models for dynamic fracture of structures built of brittle material. Comput Struct 2003;81:1255–65.
[10] Hentz S, Donzé FV, Daudeville L. Discrete element modelling of concrete submitted to dynamic loading at high strain rates. Comput Struct 2004;82:2509–24.
[11] Džiugys A, Peters BJ. An approach to simulate the motion of spherical and non-spherical fuel particles in combustion chambers. Granul Matter 2001;3(4):231–66.
[12] Balevičius R, Džiugys A, Kačianauskas R, Maknickas A, Vislavičius K. Investigation of performance of programming approaches and languages used for numerical simulation of granular material by the discrete element method. Comput Phys Commun 2006;175:404–15.
[13] Martin CL, Bouvard D, Shima S. Study of particle rearrangement during powder compaction by the discrete element method. J Mech Phys Solids 2003;51(4):667–93.
[14] Procopio AT, Zavaliangos A. Simulation of multi-axial compaction of granular media from loose to high relative densities. J Mech Phys Solids 2005;53:1523–51.
[15] Gethin DT, Ransing RS, Lewis RW, Dutko M, Crook AJL. Numerical comparison of a deformable discrete element model and equivalent continuum analysis for the compaction of ductile porpous material. Comput Struct 2001;79:1287–94.
[16] Markauskas D, Kačianauskas R. Compacting of particles for biaxial compression test by the discrete element method. J Civil Eng Manage 2006;12(2):153–61.
[17] Tordesillas A, Walsh SDC, Peters JF, Bosko JT, Muthuswamy M. Development of micromechanical models for particulate media: the role of mesoscale kinematics and non-affine motion in the transition from particle to bulk mechanical properties. In: CD-ROM proceedings of fifth world congress on particle technology WCPT2006; 2006.
[18] Guessasma M, Fortan J, Bellenger E. Numerical modelling of mechanical tests using the discrete element method. In: Topping BHV, Montero GP, Montenegro R, editors. Proceedings of the fifth international conference on engineering computational technology. Stirlingshire: Civil-Comp Press; 2006.
[19] Hoomans BPB, Kuipers JAM, van Swaaij WPM. Granular dynamics simulation of segregation phenomena in bubbling gas-fluidised beds. Powder Technol 2000;109(1–3):41–55.
[20] Shen M. Structure of a simple molecular dynamics Fortran program optimized for Cray vector processing computers. Comput Phys Commun 1989;52(2):175–85.
[21] Schfer BC, Quigley SF, Chan AHC. Acceleration of discrete element method (DEM) on a reconfigurable co-processor. Comput Struct 2004;82(20–21):1707–18.
[22] Yao Z, Wang J-S, Liu G-R, Cheng M. Improved neighbor list algorithm in molecular simulations using cell decomposition and data sorting method. Comput Phys Commun 2004;161:27–35.
[23] Hutter J, Curioni A. Dual-level parallelism for ab initio molecular dynamics: Reaching teraflop performance with the CPMD code. Parallel Comput 2005;31:1–17.
[24] Plimpton S. Fast parallel algorithms for short range molecular dynamics. J Comput Phys 1995;117(1):1–19.
[25] Pacheco P. Parallel programming with MPI. San Francisco: Morgan Kaufmann Publishers Inc.; 1997.
[26] Hendrickson B, Plimpton S. Parallel many-body simulations without all-to-all communication. J Parallel Distr Comput 1995;27(5):15–25.
[27] Komeiji Y, Haraguchi M, Nagashima U. Parallel molecular dynamics simulation of a protein. Parallel Comput 2001;27:977–87.
[28] Smith B, Bjorstad P, Gropp W. Domain decomposition: parallel multilevel methods for elliptic partial differential equations. Cambridge: Cambridge University Press; 1996.
[29] Topping BHV, Khan AI. Parallel finite element computations. Edinburgh: Saxe-Coburg Publications; 1996.
[30] Jabbarzadeh A, Atkinson JD, Tanner RI. A parallel algorithm for molecular dynamics simulation of branched molecules. Comput Phys Commun 2003;150:65–84.
[31] Hendrickson B, Devine K. Dynamic load balancing in computational mechanics. Comput Meth Appl Mech Eng 2000;184(2–4):485–500.
[32] Fleissner F, Eberhard P. Parallel load balanced particle simulation with hierarchical particle grouping strategies. In: International IUTAM2006 symposium multiscale problems in multibody system contacts; 2006.
[33] Miller S, Luding S. Event-driven molecular dynamics in parallel. J Comput Phys 2003;193:306–16.
[34] Trabado GP, Plata O, Zapata EL. On the parallelization of molecular dynamics codes. Comput Phys Commun 2002;147:711–5.
[35] Paik SH, Moon JJ, Kim SJ, Lee M. Parallel performance of large scale impact simulations on Linux cluster super computer. Comput Struct 2006;84:732–41.
[36] Dowding CH, Dymytryshyn O, Belytschko TB. Parallel processing for a discrete element program. Comput Geotech 1999;25(2):281–5.
[37] Brown K, Attaway S, Plimpton S, Hendrickson B. Parallel strategies for crash and impact simulation. Comput Meth Appl Mech Eng 2000;184(2–4):375–90.
[38] Renouf M, Dubois F, Alart P. A parallel version of the non smooth contact dynamics algorithm applied to the simulation of granular media. J Comput Appl Math 2004;168(1–2):375–82.
[39] Landry JW, Grest GS, Silbert LE, Plimpton SJ. Confined granular packing: structure, stress and forces. Phys Rev E 2003;67:1–9.
[40] Landry JW, Grest GS, Plimpton SJ. Discrete element simulations of stress distributions in silos: crossover from two to three dimensions. Powder Technol 2003;139:233–9.
[41] Kohring GA. Dynamic simulations of granular flow on multi-processor computers. In: Periaux J, editor. ECCOMAS '96: computational methods in applied sciences '96. New York: John Wiley & Sons; 1996.

[42] Schinner A. Fast algorithms for the simulation of polygonal particles. Granul Matter 1999;2:25–43.
[43] Perkins E, Williams JR. A fast contact detection algorithm insensitive to object sizes. Eng Comput 2001;18(1/2):48–61.
[44] Grest GS, Duenweg B, Kremer K. Vectorized. Link cell Fortran code for molecular dynamics simulations for a large number of particles. Comput Phys Commun 1989;55(3):269–85.
[45] Balevičius R, Džiugys A, Kačianauskas R. Discrete element method and its application to the analysis of penetration into granular media. J Civil Eng Manage 2004;10(1):3–14.
[46] Balevičius R, Kačianauskas R, Džiugys A, Maknickas A, Vislavičius K. DEMMAT code for numerical simulation of multi-particle dynamics. Inform Technol Control 2005;34(1):71–8.
[47] Allen MP, Tildesley DJ. Computer simulation of liquids. Oxford: Clarendon Press; 1991.
[48] Balevičius R, Kačianauskas R. DEM analysis of granular flow in pyramidal hoppers. In: Mota Soares CA, Martins JAC, Rodriques HC, Ambrosio JAC, editors. Proceedings of III European conference on computational mechanics, solids, structures and coupled problems in engineering. Dordrecht: Springer; 2006.
[49] Rougier E, Munjiza A, John NWM. Numerical comparison of some explicit integration schemes used in DEM, FEM/DEM and molecular dynamics. Int J Numer Meth Eng 2004;61:856–79.
[50] Maknickas A, Kačeniauskas A, Kačianauskas R, Balevičius R, Džiugys A. Parallel DEM software for simulation of granular media. Informatica 2006;17(2): 207–24.
[51] Jiang MJ, Konrad JM, Leroueil S. An efficient technique for generating homogeneous specimens for DEM studies. Comput Geotech 2003;30(7): 579–697.
[52] Hustrulid AI. Parallel implementation of the discrete element method. In: Report, Engineering Division, Colorado School of Mines, Golden, CO 80401; 1997.
[53] Aste T. Variations around disordered close packing. J Phys: Condens Matter 2005;17:S2361–90.
[54] Aste T, Saadatfar M, Senden TJ. Local and global relations between the number of contacts and density in monodisperse sphere packs. J Stat Mech: Theory Exp 2006;P07010:1–9.