INFORMATICA, 2006, Vol. 17, No. 2, 207–224
© 2006 *Institute of Mathematics and Informatics, Vilnius*

Parallel DEM Software for Simulation of Granular Media

Algirdas MAKNICKAS, Arnas KAČENIAUSKAS
Parallel Computing Laboratory, Vilnius Gediminas Technical University
Saulėtekio al. 11, 10223 Vilnius, Lithuania
e-mail: alm@sc.vtu.lt, arnka@fm.vtu.lt

Rimantas KAČIANAUSKAS, Robertas BALEVIČIUS
Laboratory of Numerical Modelling, Vilnius Gediminas Technical University
Saulėtekio al. 11, 10223 Vilnius, Lithuania
e-mail: rkac@fm.vtu.lt, robertas.balevicius@st.vtu.lt

Algis DŽIUGYS
Laboratory of Combustion Processes, Lithuanian Energy Institute
Breslaujos 3, 44403 Kaunas, Lithuania
e-mail: dziugys@isag.lei.lt

Received: May 2005

Abstract. The paper describes the development and performance of parallel algorithms for the discrete element method (DEM) software. Spatial domain decomposition strategy and message passing inter-processor communication have been implemented in the DEMMAT code for simulation of visco-elastic frictional granular media. The novel algorithm combining link-cells for contact detection, the static domain decomposition for parallelization and MPI data transfer for processors exchanging particles has been developed for distributed memory PC clusters. The parallel software DEMMAT_PAR has been applied to model compacting of spherical particles in the rectangular box. Two benchmark problems with different numbers of particles have been solved in order to measure parallel efficiency of the code. The inter-processor communication has been examined in order to improve domain decomposition topology and to achieve better load balancing. The speed-up equal to 11 has been obtained on 16 processors. The parallel performance study has been performed on the PC cluster VILKAS of Vilnius Gediminas Technical University, Lithuania.

Key words: parallel computing, the discrete element method, visco-elastic frictional granular media, spatial domain decomposition, distributed memory PC clusters.

1. Introduction

The increasing capacity of the advanced computer technologies opens up new vistas for the discrete concept. Among various numerical techniques, the discrete element method (DEM) (Cundall and Strack, 1979) started with its first application to simulate the dynamic behaviour of granular media, which is presented as an assembly of particles. Contrary to the methods based on the continuum approach, where granular material is modelled by continuum media, in the DEM the particles of granular media are treated as individual objects and all dynamical parameters of each particle are tracked during the simulation. The dynamic motion of each particle is obtained by solving classical mechanics equations of Newton's laws.

Recently, the DEM has become a powerful tool for solving many scientific and engineering applications (Džiugys and Peters, 2001; Kuwagia and Horio, 2002; Cleary *et al.*, 2003). The main advantage of the DEM is a possibility to model highly complex systems using the basic data on individual particles without 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 granular media at the continuum level.

The main disadvantages of the DEM technique, in comparison with the above-mentioned methods, are related to computational capabilities which are limited by a huge number of particles (Hoomans *et al.*, 2000) and short time period 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 the contacts between the particles. Considerable efforts have been expended by researchers to optimize this procedure for vector supercomputers (Shöen, 1989) and even to build special-purpose hardware for performing molecular dynamics (MD) simulations (Bakker *et al.*, 1990) or designing reconfigurable co-processors for DEM simulations (Schäfer *et al.*, 2004).

Naturally, for the solution of industrial-scale problems, parallelization becomes an obvious option for significantly increasing computational capabilities. Relatively new DEM codes inherited the parallel algorithms from the software for MD simulations. The majority of the works that have included the implementation of MD parallel algorithms have been intended for single-instruction/multiple-data (SIMD) machines (Tamayo *et al.*, 1991) or for multiple-instruction/multiple-data (MIMD) architectures with a few dozens of processors (Rapaport, 1991). During the last decade there have been efforts to create scalable MD algorithms that work well on hundred to thousand processor MIMD machines (Esselink *et al.*, 1993). Most of the authors (Plimpton, 1995) are convinced that a message passing model of programming (Pacheco, 1997) for distributed memory MIMD machines is the only one that provides enough flexibility to implement 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 (Hendrickson and Plimpton, 1995), but the global character of the employed algorithms produces inter-processor communication overhead on distributed memory machines.

In the second class of methods, the spatial decomposition (Plimpton, 1995), well-known as domain decomposition (Smith *et al.*, 1996), 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 (Hendrickson and Devine, 2000) is performed and none of the processors have to wait for information they need from other processors. As the degree of natural algorithmic concurrency inherent in explicit time integration procedures is high, static domain decomposition-based parallel processing strategies can yield high efficiencies and speed-ups on both shared and distributed memory hardware configurations (Owen and Feng, 2001).

The DEM algorithms for the granular media considerably differ from analogous MD simulations. A distinguishing feature between molecular dynamics and dynamics of granular material is that granular flows directly depend on the interaction between particles. Unlike collisions between molecules, solid particle collisions dissipate energy. On a macro-scale granular material can behave like a liquid or can possess a transitional state between liquid and solid. The above features of granular material in comparison with MD simulations require the additional data transfer and appropriate domain decomposition. The dynamically changing configuration feature associated with the applications of granular media makes the parallelization of DEM software much more difficult and challenging (Dowding *et al.*, 1999). Essentially, processor workload, inter-processor communication and data storage requirements undergo continuous evolution during the simulation (Brown *et al.*, 2000). This problem poses considerable difficulties for distributed memory machines. Only recently, have some successful attempts emerged at tackling problems of a similar nature (Dowding *et al.*, 1999; Renouf *et al.*, 2004). The novel algorithm combining link-cells for contact detection, the static domain decomposition for parallelization and MPI data transfer for processors exchanging particles is presented in this paper. The measured inter-processor communication and parallel performance study provided interesting technological and scientific results for application of the code on distributed memory PC clusters.

In the present research, parallel DEM software based on the spatial domain decomposition strategy has been developed for simulation of granular material on the distributed memory PC clusters. In Section 2, methodology and governing relations of the discrete element method applied to the dynamic behaviour of visco-elastic frictional granular material are described. Section 3 discusses implementation of parallel algorithms in the general DEMMAT code. In Section 4, the numerical results are presented and the efficiency of the code is investigated on the distributed memory PC cluster, while the concluding remarks are included in Section 6.

2. Governing Relations for Dynamics of Granular Material

The granular media is assumed to be composed of discrete particles termed as discrete elements. It is regarded as a system of a finite number N of spherical particles with their geometric representation of surface by radii R_i ($i = 1, \dots, N$) and their physical behaviour is described as the state of visco-elastic material. The particles can change their position due to free rigid body motion or due to contacting with the neighbouring particles or walls defining computation domain. The deformation of the particles is replaced by overlap of particles shape and it is assumed that impact between one pair of particles i and j has no influence on the contact of other pair of impacting particles i and k. A characteristic size of the overlap depth defined by h_{ij} must be much smaller than the particle size (Fig. 1). The details needed for explicit evaluation of kinematics and contact geometry of spherical particles were considered in (Balevičius *et al.*, 2004).

The dynamical behaviour of the whole granular media is time-dependent and, therefore, is considered by applying the second Newton's law for translational and rotational motion for each the particle i:

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

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

where $\mathbf{x}_i$ and $\boldsymbol{\theta}_i$ are vectors of the position of the centre of gravity and orientation of the particle, m_i is the mass of the particle i, I_i is the inertia moment of particle.

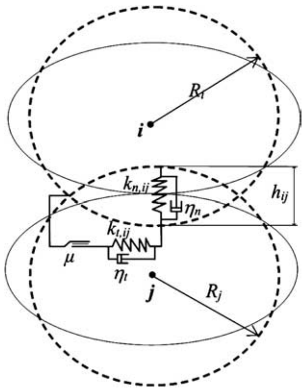


Fig. 1. A visco-elastic model of inter-particle contact.

Vectors $\mathbf{F}_i$ and $\mathbf{T}_i$ present the sum of gravity, contact forces $\mathbf{F}_{ij}$ and torques $\mathbf{T}_{ij}$, which act on the centre of gravity of the particle i, respectively:

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

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

where $\mathbf{g}$ is vector of gravity acceleration.

Several approaches to modelling contact forces are reported in the literature (e.g., Kohring, 1996). The presented visco-elastic inter-particle contact model considers a combination of elasticity, damping and friction force effects. The physical meaning of interaction stiffness $k_{n,ij}, k_{t,ij}$ and viscous damping coefficients, η_n and η_t in normal and tangential directions with respect to the contact surface is illustrated in Fig. 1.

In particular, particles normal interaction stiffness is defined by Hooke's law, $k_{n,ij} = \frac{2}{3} \cdot \frac{E}{(1-\nu^2)} R_{ij} (E, \nu$ are the elastic modulus and Poison‘s ratio of the particle material, respectively; $R_{ij} = \frac{R_i R_j}{R_i + R_j}$is the reduced radius of the colliding particles). Particles interaction stiffness in tangential direction is obtained by the simplified Mindlin's theory (Kohring, 1996), $k_{t,ij} = \frac{8}{3} \cdot \frac{G\sqrt{R_{ij}h_{ij}}}{(2-\nu)}$ (where G is the shear modulus of the particle material). The particle viscous damping coefficient coefficients η_n and η_t depends on the colliding particles reduced mass, $m_{ij} = \frac{m_i m_j}{m_i + m_j}$, normal and tangential dissipation coefficients, γ_n and γ_t, and, generally, are expressed by relation $\eta = \frac{m_{ij}}{2m_i}\gamma$.

A detailed description of these parameters can be found in (Balevičius *et al.*, 2004; Džiugys and Peters, 2001). The dynamical state of granular material is determined numerically solving the Eqs. 1–2 by the *5th-order Gear's predictor-corrector* (Allen and Tildesley, 1991) scheme.

3. Parallel Algorithms

Parallel software DEMMAT_PAR has been created in Parallel Computing Laboratory of Vilnius Gediminas Technical University. The parallel algorithms have been implemented in the sequential FORTRAN 90 code DEMMAT (Balevičius *et al.*, 2005) relying on the procedural programming approach. The code comprises subroutines, functions and module implementations as well as using intrinsic and supporting functions for vector algebra. All subroutines of the code have been written as external procedures. The interface blocks were used to define the argument details of the procedures. The sequential algorithm of DEM implemented in the code involves the following basic steps:

- problem setup (initial conditions of the particles and the walls);
- predict particles positions, velocities and accelerations by Gear's predictor;
- search for the contacts of particles-particles and particles-walls;
- compute all forces acting on the particles;

- correct particles positions, velocities and accelerations by Gear's corrector according to forces acting on the particles;
- increase simulated time by a time step;
- if simulation continues – go to the Gear's predictor, otherwise go to the next item;
- post processing, visualization.

The most CPU time consuming parts of the program are time integration and computation of contact forces. Time integration is performed by the two step Gear's predictor-corrector scheme (Allen and Tildesley, 1991). In the sequential implementation, approximately 60% of the CPU time is spent doing contact detection necessary for the calculation of contact forces.

In order to evaluate contact forces, 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, the link-cell algorithm (Grest *et al.*, 1989) was used for contact detection. The algorithm is constructed as follows. The neighbour-searching algorithm comprises referencing of individual particles to the cells and constructing of the *neighbours list of particles.* All simulation space is divided into equal cubic cells of the size not smaller then the diameter of the largest particle. Each particle i is referenced to the cell according to the position of the particle centre of gravity $\mathbf{x}_i$. After referencing the *neighbours list of particle* is constructed by assembling particles indices from the neighbouring cells which are around the reference cell. In a three-dimensional case, the reference cell includes 26 neighbouring cells, while for a two-dimensional case the reference cell has 8 neighbouring cells. The neighbouring cells also include the boundary zones. The *boundary neighbours list of particles* was constructed in the manner described above. The procedure of contact detection involves sequential checking of contacts of the type particle-particles and particle-walls and is performed by using particle indices obtained from the *neighbours list of particles* and the *boundary neighbours list of particles.* High efficiency performing sequential computations make the code to be a promising candidate for parallelization in order to become applicable for solving large industrial problems of granular media.

Parallelization of the DEMMAT_PAR 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, and also offers a generic solution for both shared and distributed memory computers. The solution domain is divided into nearly equal sub-domains (Fig. 2). 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 particles which are near the division boundaries in nearby sub-domains. The communication required in implemented algorithm is thus local in nature and a large portion of the sequential code can be used without modification. Inter-processor communication is implemented in the DEMMAT_PAR by the subroutines of message passing library MPI (Pacheco, 1997). Partitions containing a roughly equal number of particles ensure the static load balancing on the homogeneous PC clusters.

The parallel algorithm (Fig. 3) used in DEMMAT_PAR is similar to the described sequential one. The notable changes are made modifying problem setup (pre-processor

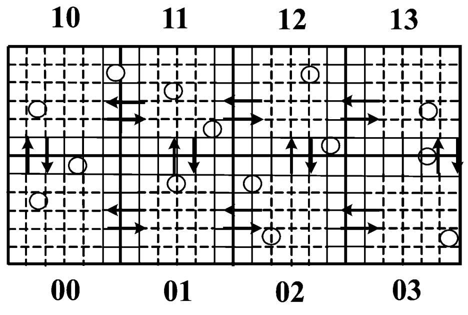


Fig. 2. The illustration of domain decomposition.

for parallel computations) and additionally implementing information exchange between neighbouring processors (concerning particles near the boundaries of sub-domains). Initially, a three-dimensional domain of the granular media is divided into cubic cells of the size approximately equal to the diameter of the largest particle. 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 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 and Gear's corrector steps by using locally stored data.

The initial portion of communications is performed after Gear's predictor when processors exchange particles as they move from one sub-domain to another. This is similar to the Eulerian approach for a fluid simulation where the grid remains fixed in space as fluid moves through it (Kačeniauskas, 2002). The inter-processor communication 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. The initial communication is carried out in the loop over the neighbouring processors. The number of exchanging particles is sent to each of the neighbouring processors. It is received and used as the main parameter for subsequent memory allocation and for conditions governing the program flow. If the number of exchanging particles is more than zero, the necessary memory buffers are dynamically allocated. The data buffers are formed and sent in another loop over the neighbouring processors. This work is optional and is performed only in the case of non-zero number of exchanging particles. The receive operations are performed in a similar manner. In such a way, the data of exchanging particles, e.g., global index, radii, mass, positions, orientations, velocities, accelerations, shear slip, integration parameters and additional information are transferred to another processor. 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 particle data should be extracted from the arrays of one processor and incorporated in the local data structure of another processor when the particle moves to another sub-domain.

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. At first, processors communicate in left-right direc-

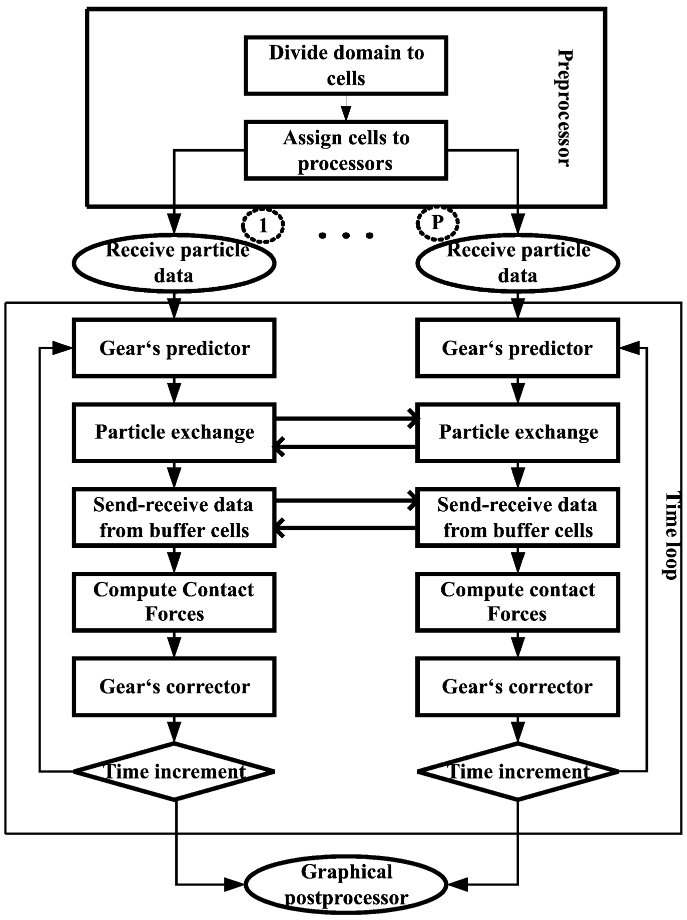


Fig. 3. The parallel algorithm.

tions, than 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. Each processor writes positions, orientations, velocities, accelerations of particles, forces, kinetic energy and other useful data in a separate result file, which can be stored in the global home directory as well as on the local hard disk. Thus, the influence of the output on parallel efficiency can be minimized if required. Finally, the results are read by the post-processor and sequentially prepared for graphical visualization.

4. Numerical Results and Discussions

The discussed algorithms have been designed for numerical simulation of granular materials by the DEM. Two benchmark problems were solved by the parallel code DEMMAT_PAR. All simulations were performed on the PC cluster VILKAS (OS Linux, OSCAR 3.0) of Vilnius Gediminas Technical University. The cluster consisted of 20 Intel®Pentium III processors (1.4GHz, 0.5GB RAM for a processor, Intel®Pro/1000T network card with 32 bit 33/66 MHz slot and 1000 Mbps). It was connected by HP ProCurve Switch 4104 GL (HP J4863A 100/1000 Base-TX Module, fabric speed 18.3 Gbps, throughput up to 35.7 million pps). The parallel performance of the developed code was evaluated by the measurements of speed-up S_p and the efficiency E_p:

$$S_p = \frac{t_1}{t_p}, \quad E_p = \frac{S_p}{p}, \tag{5}$$

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 efficiency is measured by fixing the number of particles and increasing the number of processors used. In practice, perfect efficiency is not naturally attained because of an inherent sequential part of the algorithm, parallel communication overhead and a load imbalance.

The first test simulates free compacting of granular material in a cubic box, which is assumed to be the computation domain. The side of the box is 2.0 m long. Granular material is presented by an assembly of 5000, 20000 or 100000 particles. The values of the particle radii R_i varying between 0.01 and 0.05 m are defined randomly with uniform distribution. The initial velocities of the particles are also imposed and defined in the same manner as the particle radii having their magnitudes over the range of 0–0.1 m/s.

The particles physical data for artificially assumed material are presented in Table 1. The elasticity and viscous damping parameters are chosen to avoid numerical difficulties as much as possible. A viscous damping coefficients in normal and tangential directions are assumed to be the same and equal to $\eta_n = \eta_t = 25/,\text{s}^{-1}$. The assumed values of normal viscous damping coefficient comprise by about 3% and 13% of critical damping coefficient in collision of two particles with minimal and maximal radii, respectively.

Thus, the space over the bottom is divided into cubic cells of the same size as an orthogonal and uniform grid. Cells are equal to 0.1 m (5000 particles), 0.05 m (20000 particles) and 0.025 m (100000 particles). Initially, the particles are distributed by embedding them into the centres of the cells to ensure that they are not in contact at the beginning of testing. The compacting of particles is driven by the particles settling under the force of gravity defined by gravity acceleration vector **g** ($g_x = g_y = 0$, $g_z = -9.8$ m/s^2). The box walls are treated as the fixed boundaries of the domain and the particles are initially contact-free. In the case of particle-wall contact, the latter is considered as a particle with the infinitive radius. The time integration of Eqs. 1 and 2 was carried out by the constant time integration step $\Delta t = 10^{-5}$ s.

The imposed initial distribution of the particles in the cubic box is illustrated in Fig. 4a. The simulation of the compacting process was interrupted after 1.5 s, when the

Table 1

Major data of the particles material

Quantity	Symbol	Value
Density, kg/m^3	ρ	1000
Poisson's ratio	ν	0.30
Elasticity modulus, Pa	E	$1 \cdot 10^6$
Shear modulus, Pa	G	$0.3 \cdot 10^6$
Friction coefficient	μ	1.0
Normal viscous damping coefficient, 1/s	η_n	25.0
Tangential viscous damping coefficient, 1/s	η_t	25.0

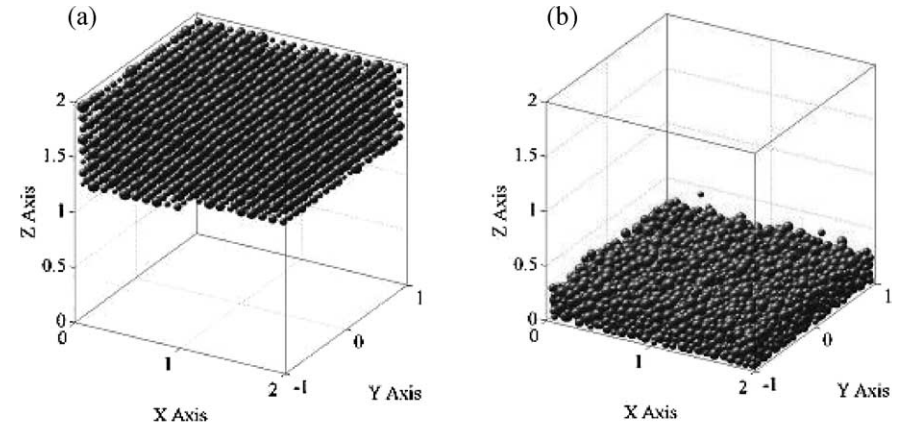


Fig. 4. A view of the granular media (5000 particles) referingto the first problem: (a) $t = 0.0$s, (b) $t = 1.5$s.

particles reached the state of rest (Fig. 4b). Physical nature and characteristic behaviour of granular flow may be investigated by considering the evolution of the total kinetic energy (Fig. 5), which in the compaction test is treated as an integral characteristic of the flow. The total kinetic energy is obtained by the summation of energy of individual particles due to rotational and translation motions. The graph obviously exhibits a different nature of the granular flow in characteristic time intervals. The increase of the total kinetic energy in the time interval reaching up to 0.35 s indicates the particle's contact-less falling period. The time interval within the next 0.35 and 0.45 s means that particles come into contact with each other, starting to rotate and, finally, coming into contact with the bottom. Hence, the increased number of the contacting particles as well as the sufficiently high coefficient of the particles friction and the coefficients of the damping predefine rapid dissipation of the total kinetic energy. The remaining time interval up to 1.5 s may be considered to be the motion stabilization interval. It reflects vanishing changes of the particles positions with respect to the bottom and their neighbors, approaching, finally, the state of the rest. It should be noted, however, that the state of the rest with negligibly small average accelerations is actually a theoretical case, because the corresponding zero

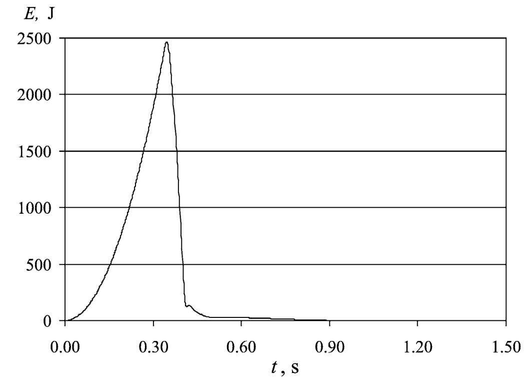


Fig. 5. The variation of the kinetic energy in the first problem.

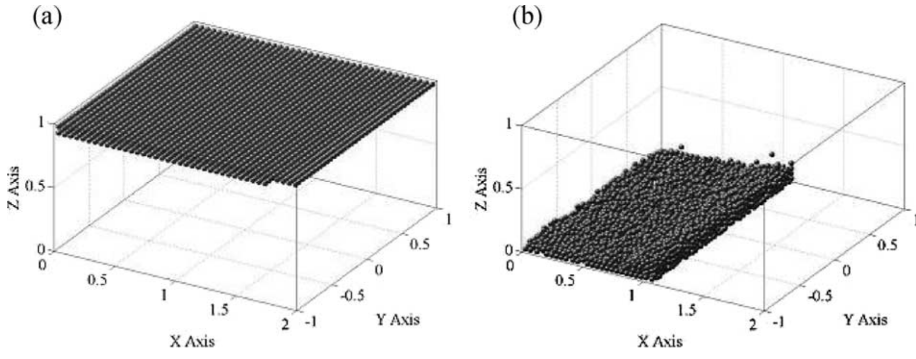


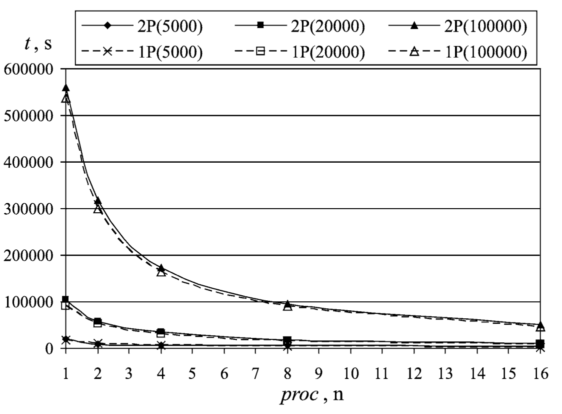
Fig. 6. A view of the granular media (5000 particles) referring to the second problem: (a) $t = 0.0$s, (b) $t = 1.5$s.

energy can be defined only within the limits of the prescribed tolerance.

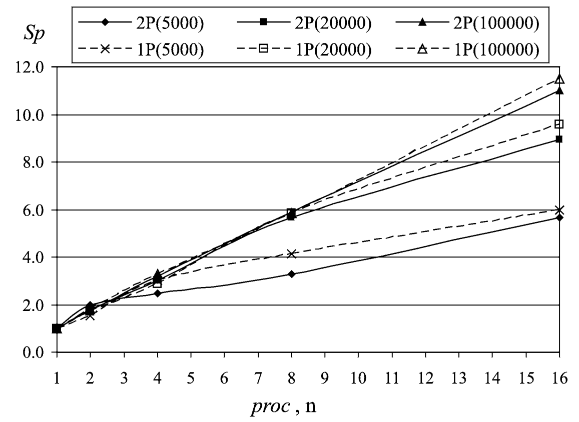
Compression of the granular material by a moving wall has been simulated in the second test case when the right wall begins to move pushing the settled material in x direction at constant velocity $v_x = -0.6$ m/s. As a result, the granular material was compressed. The simulation time of particles compression was limited to the time interval of 1.5 s. The state of granular material after 1.5 s was shown in Fig. 6. It illustrates the dynamically changing configuration of moving particles that forms a "wave" in front of the moving wall. This benchmark gives us another possibility to check the parallel efficiency of the code, when the data transfer among processors significantly varies in time.

The parallel performance tests carried out on the PC cluster VILKAS are presented in Fig. 7. The program execution time, the speed-up gained relative to a sequential run as a function of the number of processors and the parallel efficiency are shown for the systems consisting of 5000, 20000 and 100000 particles. When the number of processors is small, the speed-up is close to linear. The reduction of the efficiency owing to communication overhead is obtained for larger number of processors. The parallel efficiency is largely determined by the ratio of local computations over inter-processor communications. As the number of processors increases, for a fixed problem size, the communication cost will

(a)



(b)



(c)

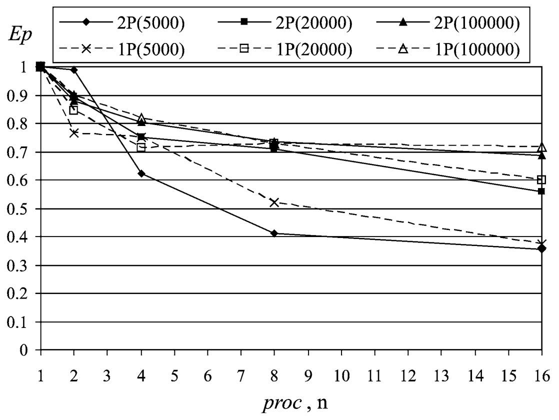


Fig. 7. Parallel performance obtained solving the first problem (1P) and solving the second problem (2P) based on different particle numbers: (a) the run time, (b) the speed-up, (c) the efficiency.

eventually become dominant over the local computation cost after a certain stage. This high ratio of communication to computation makes the influence of a further reduction in the local computation on the overall cost of running the application very small. The communication cost is quite small for relatively large problems (100000 and 20000 particles), where a large number of particles are used per processor. The results show that the implemented inter-processor communication algorithms are well designed for the distributed memory PC clusters. Fig. 7 also shows the results of the speed-up study solving the second test case. The unpredictable nature of particles moving between processors makes speed-up curves more chaotic than the curves obtained in parallel FEM computations (Kačeniauskas and Rutschmann, 2004). The plotted curves illustrate the superiority of the speed-up achieved for the first benchmark problem in comparison with that obtained solving the second benchmark.

Most of the authors report parallel efficiency obtained on various supercomputers. Hustrulid (Hustrulid, 1997) 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 presented work. The NSCD solver was developed for shared memory computers (SUN-FIRE 880 and SGI origin 3800) and described in the paper (Renouf *et al.*, 2004). The obtained speed-up ($S_p = 11$ on 16 processors) can hardly be compared with the present results, because of different parallelisation strategy and hardware. Qwen (Owen and Feng, 2001) presented dynamic domain decomposition algorithm for shared memory machine SGI Origin 2000. The parallel efficiency was tested on 6 processors only ($S_p = 4.41$ or $S_p = 5$). Dowding (Dowding *et al.*, 1999) developed efficient NURBM3DP algorithm for distributed memory computers. The excellent speed-up ($S_p = 9.89$) was attained on 10 processors of IBM SP2. 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 ($S_p = 1.79$ or $S_p = 1.97$) attained in the presented work. The parallel efficiency reported in the presented paper can compete with that obtained on supercomputers and reported by other authors.

Parallel computation based on the static domain decomposition starts from the geometrical subdivision of the solution domain. The pre-processor assigns different sub-domains containing cells and particles to different processors. The algorithm (Fig. 3) will be well load-balanced only if all sub-domains have a roughly equal number of particles. The most important question for the static domain decomposition is how to divide the solution domain ensuring the load-balance. In the case of unstructured sub-domains or irregular connectivity, the task of assigning particles to sub-domains and communication algorithm becomes more costly and complex. Taking into account the cubic shape of the investigated solution domains, the uniform grid structure of sub-domains is considered. Considering the types of models used in the DEM, the bodies tend to head for the bottom of the solution domain due to gravity. Dividing the domain into a 3D grid would leave areas without bodies, and processors without work to do. To avoid poor load balancing, the first problem is divided into 2D grid of columns (n subdivisions in x direction and m subdivisions in y direction) containing approximately equal number of particles for the whole time interval of simulation.

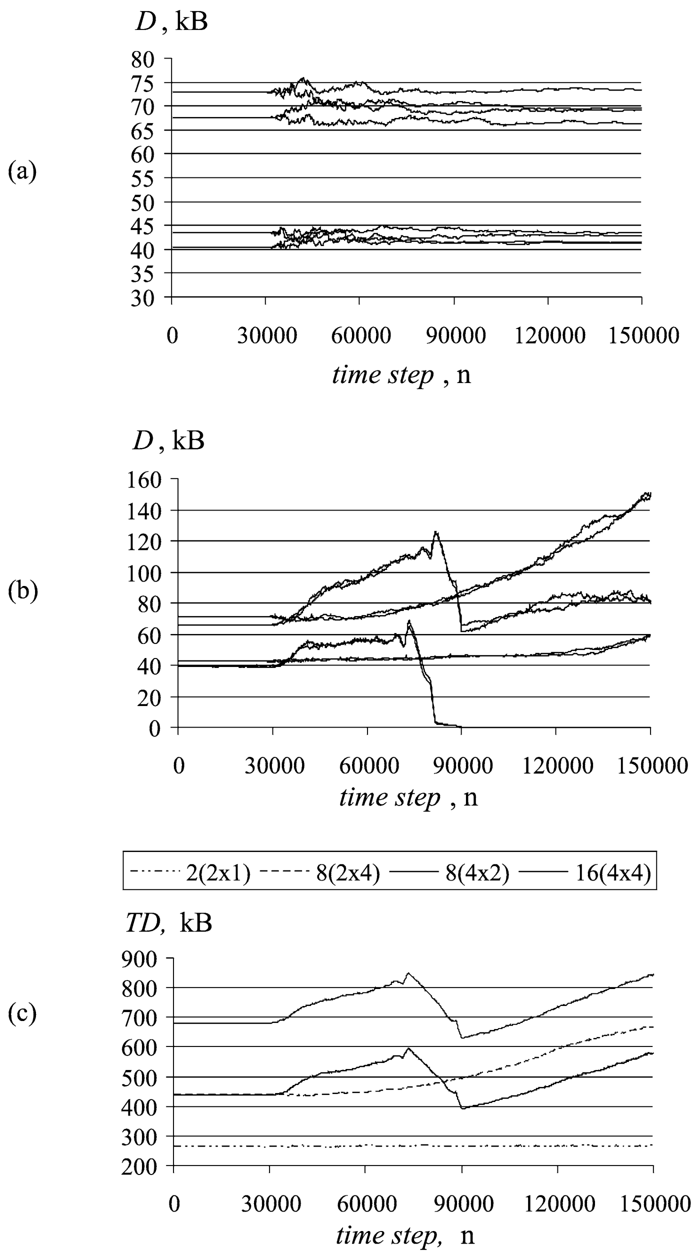


Fig. 8. Inter-processor communications: (a) the dependency of data transfer D on time step for each processor in one spatial decomposition 8 (4×2) of the first benchmark problem, (b) the dependency of data transfer D on time step for each processor in one spatial decomposition 8 (4×2) of the second benchmark problem, (c) the dependency of total data transfer *TD* on time step for four spatial decompositions.

A highly efficient parallel implementation requires two, often competing objectives to be achieved: a well balanced workload among the processors and a low level of inter-processor communication overhead. A difficult task is to find optimal domain subdivision for the second test case. The dynamic evolution of the particle configuration decreases the quality of the load-balance achieved by static domain decomposition. Any dynamic load balancing scheme requires additional computational overhead and data movement. Fig. 8 illustrates inter-processor communications in time. Data transfer D measured in eight processors is visualized by eight curves. The test case of 20000 particles employs spatial decomposition 8 (4×2) that means 8 working processors, 4 domain subdivisions in x direction and 2 domain subdivisions in y direction. Fig. 8a illustrates inter-processor data transfer for the first benchmark problem. At the beginning of computations ($t = 0.3$ s) the data transfer was almost constant for both problems. The data transfer for the first benchmark problem remained at the roughly same level in the whole time interval. The variation of the data transfer D had totally different character for the second benchmark problem (Fig. 8b). The dynamic changes of the particle configuration increased data transfer among processors. Two curves falling to zero showed, that, when $t = 0.9$s, two processors remained without data transfer, as well as without particles and load. The second problem employing spatial decomposition 8 (4×2) is purely load-balanced. Fig. 8c illustrates the total data transfer *TD* (the sum of D measured in each processor) between processors for the second test case of 20000 particles solved by different numbers of processors (2, 8 and 16) and different spatial decompositions $(2 \times 1$, 2×4, 4×2 and $4 \times 4)$. The curves show that purely load balanced applications like 8 (4×2) require less data transfer than a well-balanced one 8 (2×4). Despite of this, the second one runs faster, because it is better load-balanced. Considering this issue, the work load of the benchmark problem with moving wall was properly balanced with appropriate spatial decompositions. The small difference between the curves of the first benchmark problem and the second benchmark problem (Fig. 7) shows that the considered spatial domain decomposition algorithm is well designed for the distributed memory PC clusters.

5. Conclusions

In this paper, the development of parallel DEM software for simulation of granular visco-elastic frictional media has been described. The parallel algorithms have been based on the spatial domain decomposition strategy and message passing interface (MPI). The DEM code DEMMAT_PAR has been successfully parallelized designing data structures for the inherently synchronic message passing algorithm. The solution of two benchmark problems with a dynamically changing configuration has illustrated high efficiency of parallel implementation. The data transfer of inter-processor communication has been investigated in order to find suitable spatial decomposition topology for the dynamically changing configuration of the media. The obtained parallel performance shows that the applied spatial domain decomposition and the implemented inter-processor communication algorithms are well designed for the distributed memory PC clusters. The measured speed-up of the parallel computations can compete with that obtained by other researchers on supercomputers and reported in the literature.

References

Allen, M.P., and D.J. Tildesley (1991). *Computer Simulation of Liquids*. Clarendon Press, Oxford.

Bakker, A.F., G.H. Gilmer, M.H. Grabow and K. Thompson (1990). A special purpose computer for molecular dynamics calculations. *J. Computational Physics*, **90**(2), 313–335.

Balevičius, R., R. Kačianauskas, A. Džiugys, A. Maknickas and K. Vislavičius (2005). DEMMAT code for numerical simulation of multi-particle dynamics. *J. Information Technology and Control*, **34**(1), 71–78.

Balevičius, R., A. Džiugys and R. Kačianauskas (2004). Discrete element method and its application to the analysis of penetretion in granular media. *J. Civil Engineering and Management*, **10**(1), 3–14.

Brown, K., S. Attaway, S. Plimpton and B. Hendrickson (2000). Parallel strategies for crash and impact simullation. *Computer Methods in Applied Mechanics and Engineering*, **184**(2–4), 375–390.

Cleary, P.W., R. Morrisson and S. Morrell (2003). Comparison of DEM and experiment for a scale model SAG mill. *Int. J. Mineral Processing*, **68**(1–4), 129–165.

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

Dowding, C.H., O. Dymytryshyn and T.B. Belytschko (1999). Parallel processing for a discrete element program. *Computers and Geotechnics*, **25**(2), 281–285.

Džiugys, A., and B.J. Peters (2001). An approach to simulate the motion of spherical and non-spherical fuel particles in combustion chambers. *Granular Material,* **3**(4), 231–266.

Esselink, K., B. Smith and P.A.J. Hilbers (1993). Efficient parallel implementation of molecular dynamics on a toroidal network: I. Parallelizing strategy. *J. Computational Physics*, **106**(1), 101–107.

Grest, G.S., B. Duenweg and K. Kremer (1989). Vectorized link cell fortran code for molecular dynamics simulations for a large number of particles. *Computer Physics Communications*, **55**(3), 269–285.

Hendrickson, B., and K. Devine (2000). Dynamic load balancing in computational mechanics. *Computer Methods in Applied Mechanics and Engineering*, **184**(2–4), 485–500.

Hendrickson, B., and S. Plimpton (1995). Parallel many-body simulations without all-to-all communication. *J. Parallel and Distributed Computing,* **27**(5), 15–25.

Hoomans, B.P.B., J.A.M. Kuipers and W.P.M. van Swaaij (2000). Granular dynamics simulation of segregation phenomena in bubbling gas-fluidised beds. *Powder Technology*, **109**(1–3), 41–55.

Hustrulid, A.I. (1997). Parallel implementation of the discrete element method. *Report*, Engineering Division, Colorado School of Mines, Golden, CO 80401.

Kačeniauskas, A. (2002). Interface capturing techniques for tank sloshing problems. *J. Mechanics*, **37**(5), 14–17.

Kačeniauskas, A., and P. Rutschmann (2004). Parallel FEM software for CFD problems. *Informatica*, **15**(3), 363–378.

Kohring, G.A. (1996). Dynamical simulations of granular flows on multi-processor computers. In *Computational Methods in Applied Sciences'96*, John Wiley & Sons Ltd. pp. 190–196.

Kuwagia, K., and M. Horio (2002). A numerical study on agglomerate formation in a fluidized bed of fine cohesive particles. *Chemical Engineering Science*, **57**(22–23), 4737–4744.

Owen, D.R.J., and Y.T. Feng (2001). Parallelized finite/discrete element simulation of multi-fracturing solids and discrete systems. *Engineering Computations*, **18**(3–4), 557–576.

Pacheco, P. (1997). *Parallel Programming with MPI*. Morgan Kaufmann Publishers Inc., San Francisco.

Plimpton, S. (1995). Fast parallel algorithms for short range molecular dynamics. *J. Computational Physics*, **117**(1), 1–19.

Rapaport, D.C. (1991). Multi-milion particle molecular dynamics: II. design considerations for distributed processing. *Computer Physics Communication*, **62**(2–3), 217–228.

Renouf, M., F. Dubois and P. Alart (2004). A parallel version of the non smooth contacy dynamics algorithm applied to the simulation of granular media. *J. Computational and Applied Mathematics*, **168**(1–2), 375–382.

Schäfer, B.C., S.F. Quigley and A.H.C. Chan (2004). Acceleration of discrete element method (DEM) on a reconfigurable co-processor. *Computers & Structures*, **82**(20–21), 1707–1718.

Shöen, M. (1989). Structure of a simple molecular dynamics Fortran program optimized for Cray vector processing computers. *Computer Physics Communication*, **52**(2), 175–185.

Smith, B., P. Bjørstad and W. Gropp (1996). *Domain Decomposition*: *Parallel Multilevel Methods for Elliptic Partial Differential Equations.* Cambridge University Press, Cambridge.

Tamayo, P., J.P. Mesirov and R.M. Boghosian (1991). Parallel approaches to short-range molecular dynamics simulations. In *Proc. of Supercomputing`91*, IEEE Computer Society Press. pp. 462–470.

A. Maknickas – programmer in the Parallel Computing Laboratory of VGTU. Graduated in physics (as a theorist of the atom theory) from the Faculty of Physics of Vilnius University, Lithuania in 1986. Research interests are centered on parallel computing, the discrete element method and numerical simulation of time-dependent structures.

A. Kačeniauskas dr., principal researcher in the Parallel Computing Laboratory of Vilnius Gediminas Technical University (VGTU), Lithuania. Graduated in applied mathematics from the Vilnius University, Lithuania, in 1995. The MS degree in computer science received from the VGTU in 1996. Doctor thesis "Modelling of viscous incompressible flows and free surfaces by the finite element method" defended and PhD degree received from VGTU in 2000. Research interests comprise parallel and distributed computing, computational fluid dynamics, computational electromagnetics, the finite element method, the discrete element method, free surfaces and moving interfaces.

R. Kačianauskas – professor, dr. habil.-eng., head of the Department of Strength of Materials, senior researcher in the Laboratory of Numerical Modelling, Vilnius Gediminas Technical University, Lithuania. Graduated Vilnius Civil Engineering Institute, now Vilnius Gediminas Technical University (VGTU), Lithuania in 1975, PhD degree obtained in1982 at VGTU. Held DAAD scholarship (Germany) in 1986, 1992 and 2000. Visiting researcher in Sweden, UK, Portugal and Switzerland. Author of monograph "Computer Methods in Multi-Level Modelling of Beams and Shells" and over 100 publications. Research interests comprise computational mechanics, finite element method, discrete element method, fracture mechanics, modelling of structures, materials and multi-physical phenomena.

R. Balevičius dr., principal researcher in the Laboratory of Numerical Modelling of Vilnius Gediminas Technical University (VGTU), Lithuania. Received the MS degree in civil engineering science from VGTU in 1996. Doctor thesis "Evaluation of linear creep in analysis of prestressed concrete linear members" defended and PhD degree received from VGTU in 2000. Research interests comprise the discrete element method, time-dependent analysis of structures, and the finite element method.

A. Džiugys dr., principal researcher in the Lithuanian Energy Institute, Kaunas, Lithuania. Graduated in physics (as theorist of quantum electrodynamics) from the Faculty of Physics of the Vilnius University, Lithuania in 1986. Doctoral thesis "Research of heat and mass transfer in free shear layer" defended in 1992 and doctor of technical science received from Lithuanian Energy Institute. Research interests include methods of numerical simulation, discrete element methods, molecular dynamics simulation, hydrodynamics, computational fluid dynamics, free surfaces and moving interfaces, arterial blood flow, theoretical physics.

Lygiagreti DEM programinė įranga birioms medžiagoms modeliuoti

Algirdas MAKNICKAS, Arnas KAČENIAUSKAS, Rimantas KAČIANAUSKAS, Robertas BALEVIČIUS, Algis DŽIUGYS

Straipsnyje nagrinėjamas diskrečiųjų elementų metodo (DEM) programinės įrangos lygiagrečiųjų algoritmų kūrimas ir tobulinimas. DEMMAT programų pakete birioms klampiaitamprioms medžiagoms modeliuoti įdiegtos tarpprocesorinės komunikacijos, pagrįstos pranešimų perdavimu, bei srities skaidymo koncepcija. Paskirstytos atminties PK telkiniams sukurtas naujas algoritmas, jungiantis gardelių metodiką kontaktų paieškai, statinę srities skaidymo koncepciją ir dalelių perėjimų iš vieno procesoriaus į kitą, pagrįstų MPI pranešimų perdavimu. Sferinių dalelių suspaudimo stačiakampėje srityje uždavinys išspręstas lygiagrečių programų paketu DEMMAT_PAR. Lygiagrečiųjų skaičiavimų efektyvumas išmatuotas sprendžiant du testinius uždavinius su skirtingais dalelių skaičiais. Siekiant pagerinti srities skaidymo topologiją ir užtikrinti subalansuotą procesorių apkrovą, ištirtas tarp procesorių persiunčiamas duomenų kiekis. Naudojant 16 procesorių, nagrinėjamą uždavinį pavyko išspręsti 11 kartų greičiau. Visi lygiagrečiųjų skaičiavimų tyrimai atlikti Vilniaus Gedimino technikos universiteto PK telkinyje VILKAS.