The current issue and full text archive of this journal is available at
www.emeraldinsight.com/0264-4401.htm

Large-scale parallel discrete element simulations of granular flow

Jens H. Walther
Department of Mechanical Engineering, Technical University of Denmark, Kongens Lyngby, Denmark, and Institute of Theoretical Computer Science, ETH Zurich, Zurich, Switzerland, and
Ivo F. Sbalzarini
Institute of Theoretical Computer Science, ETH Zurich, Zurich, Switzerland

Abstract

Purpose – The purpose of this paper is to present large-scale parallel direct numerical simulations of granular flow, using a novel, portable software program for discrete element method (DEM) simulations.

Design/methodology/approach – Since particle methods provide a unifying framework for both discrete and continuous systems, the program is based on the parallel particle mesh (PPM) library, which has already been demonstrated to provide transparent parallelization and state-of-the-art parallel efficiency using particle methods for continuous systems.

Findings – By adapting PPM to discrete systems, results are reported from three-dimensional simulations of a sand avalanche down an inclined plane.

Originality/value – The paper demonstrates the parallel performance and scalability of the new simulation program using up to 122 million particles on 192 processors, employing adaptive domain decomposition and load balancing techniques.

Keywords Simulation, Particle size measurement, Meshes, Computer software, Programming and algorithm theory

Paper type Research paper

1. Introduction

Discrete element method (DEM) simulations constitute an important tool for the study of granular matter flows. Their fully resolved "atomistic" character allows to determine material properties (Hunt, 1997) and effective macroscopic dynamics under minimal assumptions. This renders DEM simulations invaluable in the search for continuum theoretic descriptions of granular matter (de Gennes, 1999; Douady *et al.*, 1999; Douady *et al.*, 2002; Douady *et al.*, 2006). DEM simulations have, for example, successfully been used to study the packing in a cylindrical container using up to 144,000 particles (Landry *et al.*, 2003). While such simulations are particularly well suited for the multiscale considerations (Heyes, *et al.*, 2004) of linking the single-particle description to the continuum description, the size of the computationally tractable systems is frequently limited to a few hundreds of thousands of grains (Richards *et al.*, 2004; Rycroft

Engineering Computations: International Journal for Computer-Aided Engineering and Software
Vol. 26 No. 6, 2009
pp. 688-697
© Emerald Group Publishing Limited
0264-4401
DOI 10.1108/02644400910975478

The authors gratefully acknowledge the experimental validation data provided by Stéphane Douady and Sylvain Courrech du Pont of École Normale Supérieure, Paris, France. They also thank the Swiss National Supercomputing Centre (CSCS) for providing the compute resources required by this work. The first author acknowledges additional support by the Myhrwold Foundation.

et al., 2006). This limitation currently hinders the use of fully resolved DEM simulations in studying the continuum limit of granular materials. One possible solution consists in using coarse-grained methods, such as the recently presented spot model (Rycroft *et al.*, 2006), in order to reach more realistic system sizes.

In the present work, we take a different approach based on high-performance computing. We present an efficient parallel DEM implementation on the basis of the general-purpose, scalable and portable parallel particle mesh (PPM) library (Sbalzarini *et al.*, 2006). While the PPM library was so far mostly used for simulations using continuous particle methods, it also supports fully discrete particle simulations (Altenhoff *et al.*, 2007). We have thus developed and implemented a specialized DEM client for the PPM library, taking advantage of method-specific optimizations. Within the PPM framework, the actual simulation client is written in a sequential way, using the parallelization primitives of the library. The library then automatically performs the parallelization, including domain decomposition, load balancing, communication scheduling and distributed file I/O.

Previous approaches to parallel DEM simulations include an adaptation of the molecular dynamics code DL_POLY (Todorov and Smith, 2004) to DEM simulations (Dutt *et al.*, 2005) and a specialized parallel code for finite and discrete element simulations based on domain-decomposition and dynamic load (re-)balancing (Owen and Feng 2001). Due to the limited scalability of these codes, the sizes of the simulated systems were, however, below 500,000 particles.

Using the present implementation, we perform DEM simulations of frictional, viscoelastic spheres using up to 122 million particles on 192 processors. These simulations consider a sand avalanche down an inclined plane (Balmforth and Kerswell, 2005) and allow us to study front propagation speeds and moving layer characteristics.

Large-scale DEM simulations have a significant potential in many areas of research and technology, including the environmental sciences to, e.g. simulate land slides or avalanches (Cleary and Prakash, 2004; Richards *et al.*, 2004), the study of complex self-organization phenomena as, e.g. in singing sand dunes (Douady *et al.*, 2006), and research toward answering open questions in the continuum modeling of granular matter (de Gennes, 1999).

2. Grain interaction model

In order to realistically model sand and prevent spurious crystallization artifacts, we consider spherical grains with randomly perturbed radii. Each particle is represented by the location of its center $\boldsymbol{r}_i$, its radius R_i, its mass m_i and its polar moment of inertia I_i. The collision of a particle i with particle j is modeled according to Silbert *et al.* (2001) (with the correction published later (Silbert *et al.*, 2002)), generalized to varying radii. The contact forces have a normal component due to elastic deformation of the spheres and a tangential component due to viscous friction. We use the Hertzian contact pressure model.

The normal elastic deformation of a contact is given by

$$\delta_{ij} = \left(R_i + R_j\right) - r_{ij}, \tag{1}$$

with $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$ the distance vector between the two particle centers and $r_{ij} = \|\boldsymbol{r}_{ij}\|_2$

its length. The normal and tangential components of the slip velocity at the point of contact are

$$\boldsymbol{v}_{n_{ij}} = \left(\boldsymbol{v}_{ij} \cdot \boldsymbol{n}_{ij}\right)\boldsymbol{n}_{ij}, \tag{2}$$

$$\boldsymbol{v}_{t_{ij}} = \boldsymbol{v}_{ij} - \boldsymbol{v}_{n_{ij}} - \left(\boldsymbol{\omega}_i R_i + \boldsymbol{\omega}_j R_j\right) \times \boldsymbol{n}_{ij}, \tag{3}$$

where $\boldsymbol{n}_{ij} = \boldsymbol{r}_{ij}/r_{ij}$ is the unit normal vector onto the contact plane, $\boldsymbol{\omega}_i$ the angular velocity of a particle and $\boldsymbol{v}_{ij} = \boldsymbol{v}_i - \boldsymbol{v}_j$ the relative velocity between the two particle centers. The evolution of the elastic tangential displacement $\boldsymbol{u}_{t_{ij}}$ is governed by

$$\frac{d\boldsymbol{u}_{t_{ij}}}{dt} = \boldsymbol{v}_{t_{ij}} \tag{4}$$

and is integrated over the duration of a contact with initial condition $\boldsymbol{u}_{t_{ij}}(0) = 0$ at first contact. The normal and tangential forces acting on the particles then become:

$$\boldsymbol{F}_{n_{ij}} = \sqrt{\frac{\delta_{ij}}{R_i + R_j}}\left(k_n \delta_{ij} \boldsymbol{n}_{ij} - \gamma_n m_{\text{eff}} \boldsymbol{v}_{n_{ij}}\right), \tag{5}$$

$$\boldsymbol{F}_{t_{ij}} = \sqrt{\frac{\delta_{ij}}{R_i + R_j}}\left(-k_t \boldsymbol{u}_{t_{ij}} - \gamma_t m_{\text{eff}} \boldsymbol{v}_{t_{ij}}\right), \tag{6}$$

where $k_{n,t}$ are the elastic constants in normal and tangential direction, respectively, and $\gamma_{n,t}$ the corresponding viscoelastic constants. The effective collision mass is $m_{\text{eff}} = m_i m_j/(m_i + m_j)$. At each interaction time step, the elastic tangential displacement $\boldsymbol{u}_{t_{ij}}$ is truncated in order to enforce Coulomb's law $\|\boldsymbol{F}_{t_{ij}}\|_2 < \|\mu \boldsymbol{F}_{n_{ij}}\|_2$. Truncating the elastic displacement if this condition is violated amounts to the two spheres slipping against each other without inducing further deformations. The truncation is performed by scaling the tangential force as

$$\boldsymbol{F}_{t_{ij}} \leftarrow \boldsymbol{F}_{t_{ij}} \frac{\|\mu \boldsymbol{F}_{n_{ij}}\|_2}{\|\boldsymbol{F}_{t_{ij}}\|_2} \tag{7}$$

and adjusting accordingly the stationary elastic displacement:

$$\boldsymbol{u}_{t_{ij}} = -\frac{1}{k_t}\left(\boldsymbol{F}_{t_{ij}} \sqrt{\frac{R_i + R_j}{\delta_{ij}}} + \gamma_t m_{\text{eff}} \boldsymbol{v}_{t_{ij}}\right). \tag{8}$$

The total resultant force on particle i is then computed by summing the contributions of all particles j with which it currently interacts, thus:

$$\boldsymbol{F}_i^{\text{tot}} = m_i \boldsymbol{g} + \sum_j \left(\boldsymbol{F}_{n_{ij}} + \boldsymbol{F}_{t_{ij}}\right), \tag{9}$$

where $\boldsymbol{g}$ is the acceleration due to gravity. The total torque acting on particle i is given by

$$\boldsymbol{T}_i^{\text{tot}} = -R_i \sum_j \boldsymbol{n}_{ij} \times \boldsymbol{F}_{t_{ij}}. \tag{10}$$

We integrate the equations of motion using the second-order accurate leap frog scheme

$$\boldsymbol{v}_i^{n+1} = \boldsymbol{v}_i^n + \frac{\delta t}{m_i} \boldsymbol{F}_i^{\text{tot}}, \quad \boldsymbol{r}_i^{n+1} = \boldsymbol{r}_i^n + \delta t \boldsymbol{v}_i^{n+1} \tag{11}$$

$$\boldsymbol{\omega}_i^{n+1} = \boldsymbol{\omega}_i^n + \frac{\delta t}{I_i} \boldsymbol{T}_i^{\text{tot}}, \tag{12}$$

where $(\boldsymbol{r}_i^n, \boldsymbol{v}_i^n, \boldsymbol{\omega}_i^n)$ denotes the state of the ith particle at time step n, and δt the time step size. The elastic tangential displacement of each particle–particle (PP) contact (Equation (4)) is integrated using the explicit Euler scheme with the same time step size.

3. Implementation using the PPM library

The present DEM implementation is based on the PPM library (Sbalzarini *et al.*, 2006), which provides an efficient and generic parallelization framework for numerical simulations using particle methods. The PPM library supports a wide range of static and adaptive domain decomposition schemes. These decompositions divide the computational domain into *sub-domains* with sufficient granularity to provide adequate load balancing. The concurrent presence of different decompositions allows to perform each step of the computational algorithm in its optimal environment with respect to load balance and the computation-to-communication ratio. For the actual computations, the individual sub-domains are treated as independent problems and extended with *ghost layers* to allow for communication between them. These ghost layers contain *ghost particles* that are copies of true particles that either reside on a neighboring processor, or account for periodic boundary conditions.

A PPM *topology* is defined by the decomposition of space into sub-domains with the corresponding boundary conditions, and the assignment of these sub-domains onto processors. Sub-domains can be assigned to the processors in various ways. The PPM-internal method assigns contiguous blocks of sub-domains to processors until the accumulated cost for a processor is greater than the theoretical average cost under uniform load distribution. The average is weighted with the relative processor speeds to account for heterogeneous machine architectures. In addition, four different Metis-based (Karypis and Kumar, 1998) and a user-defined assignment are available. Multiple topologies may coexist and library routines are provided to map particle data between them as described below.

In order to achieve good load balance, the *SAR heuristic* (Moon and Saltz, 1994) is used in the PPM library to decide when problem re-decomposition is advised, i.e. when the cost of topology re-definition is amortized by the gain in load balance. Moreover, all topology definition routines can account for the true computational cost of each particle, for example defined by the actual number of its interactions, and the effective speeds of all processors.

PPM topologies implicitly define a data-to-processor assignment. Mapping routines provide the functionality of sending particles to the proper processor, i.e. to the one that "owns" the corresponding sub-domain(s) of the computational space. Three different mapping types are provided:

(1) *global mapping*, involving an all-to-all communication among processors;

(2) *local mapping* for neighborhood communication; and

(3) *ghost mappings* to update the ghost layers.

The global mapping is used to perform the initial data-to-processor assignment or to switch from one topology to another (e.g. after problem re-decomposition for load balancing reasons), whereas the local mapping is mainly used to account for particle motion during a simulation. Communication is scheduled by solving the minimum edge coloring problem using the efficient approximation algorithm by Vizing (1964). Ghost mappings are provided to receive ghost particles and to send ghost contributions back to the corresponding real element.

For efficiency, all mapping types are organized as stacks. A mapping operation consists of four steps:

(1) defining the mapping;

(2) pushing data onto the send stack;

(3) performing the actual send and receive operations; and

(4) popping the data from the receive stack.

This architecture allows data stored in different arrays to be sent together to minimize network latency, and mapping definitions to be re-used by repeatedly calling the push/send/pop sequence for the same, persisting mapping definition.

The evaluation of PP interactions is a key component of DEM as the overall dynamics of the system are governed by local particle interactions. The PPM library implements symmetric and non-symmetric PP computations using a novel type of cell lists (Sbalzarini *et al.*, 2005) and Verlet lists (Verlet, 1967).

The present DEM implementation uses asymmetric PP interactions on the basis of Verlet lists (Verlet, 1967) that are constructed in linear time using an intermediate cell list. The size of the Verlet sphere is chosen slightly larger than the particle diameter to capture all possible contacts., such that all contact partners of a particle are inside. In order to prevent an update of the Verlet list at every time step, we enlarge the Verlet sphere by a safety margin, called the *skin*. Thus, the interaction lists only need to be reconstructed once a particle has traveled more than the skin thickness. In our implementation, we use the conservative lower bound

$$N_{Verlet} = \frac{c - d_{max}}{2\max(v_i)\delta t} \tag{13}$$

for the number of time steps between two list updates. Hereby, c denotes the size of the Verlet sphere including the skin, d_{max} the diameter of the largest particle in the domain and $\max(v_i)$ the magnitude of the largest occurring particle velocity. The serial implementation of the Verlet list is at most $3^4/4\pi \approx 6$ times faster than the corresponding cell list. However, the parallel implementation of the Verlet list offers the additional improvement that the particles need only be mapped onto the processors

after the Verlet list is updated. Between Verlet list updates the only required communication is the *state* of the *ghost particles* – the particular ghost mapping remains constant and is stored and used repeatedly until the next Verlet update.

4. Results for a sand avalanche

In order to simulate an avalanche down an inclined plane, we use the grain interaction model as described above, with the following parameter values: $k_n = 7{,}849\,\mathrm{N/m}$, $k_t = 2{,}243\,\mathrm{N/m}$, $\gamma_n = 3{,}401\,\mathrm{N/m}$, $\gamma_t = 3{,}401.0\,\mathrm{s}^{-1}$, and $\mu = 1$. The reference system contains 635,780 grains with uniformly distributed radii between 1.00 and 1.12 mm placed in a computational domain of size $1{,}800 \times 8.6 \times 250$ mm. The grains are initially placed on a regular lattice with a void fraction of 0.47 and confined in horizontal and vertical direction by solid walls modeled by two layers of immobile grains. Periodic boundary condition are imposed in the span-wise y-direction. The radius of the Verlet sphere is $c = 15$ mm, and the time step size is 10^{-6} s. These parameters require an update of the Verlet lists every 150 time steps.

All simulations are performed on the Cray XT3 computer at the Swiss National Supercomputing Centre (CSCS). This distributed memory machine has 1664 AMD Opteron processors at 2.6 GHz clock speed (5.2 GFlop/s peak performance). Each processor has 1 GB of main memory, and they are interconnected in a full 3D torus network topology through a six-way Cray SeaStar network. The machine is running version 1.4 of the UNICOS/lc operating system, and all codes are compiled using the Portland Group Fortran 90 compiler, version 6.1-4 (64 bit).

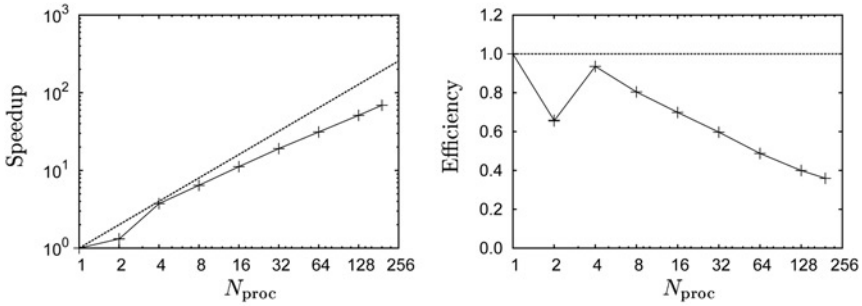
The performance of the simulations is tested for both a fixed-size and a scaled-size problems. In the fixed-size problem, the number of particles is kept constant, i.e. the work load per processor decreases with increasing number of processors. In the scaled problem, the particle number grows proportionally to the number of processors, resulting in a constant work load per processor. In each test, we measure the elapsed wall clock time t_{ij} for each time step j on each processor $i = 1, \ldots, N_{proc}$. To account for synchronous communication steps, we report the maximum of these times over all processors to compute speedup S and efficiency e:

$$S(N_{proc}) = \frac{t(1)}{\mathrm{mean}_j \max_i t_{ij}(N_{proc})} \cdot \frac{N(N_{proc})}{N(1)}, \tag{14}$$

$$e(N_{proc}) = \frac{S(N_{proc})}{N_{proc}}, \tag{15}$$

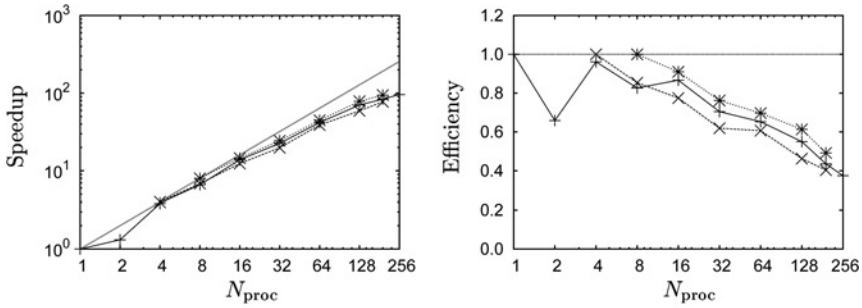
where $t(1)$ is the time on a single processor (linearly extrapolated if not measured), $t_{ij}(N_{proc})$ is the time on N_{proc} processors, $N(1)$ is the problem size on a single processor and $N(N_{proc})$ is the problem size on N_{proc} processors.

The performance of the code is shown in Figures 1 and 2 for the scaled and fixed size problems, respectively. We observe a good speedup up to 192 processors and a parallel efficiency of 40 percent for both the scaled and fixed-size problem on 192 processors. The scaled problem uses up to 122 million fully resolved particles and takes less than 2.6 s of wall clock time per time step. One time step of the fixed problem with 635,780 particles takes 1 s of wall clock time on a single processor and 0.011 s on 192 processors. For the ten-fold larger fixed-size problem, one time step takes 0.13 s on



Notes: The initial number of particles on two processors is 635,780, scalling up to 122 million particles on 192 processors. The PPM library decomposes the domin into 4,213 sub-domains that are automatically disributed among the processors. All timings are performed on the Cray XT3

Figure 1.
Parallel speedup and efficiency of the PPM DEM client for the scaled-size problem



Notes: The PPM ilbrary decomposes the domain into 4,213 sub-domainds that are automatically distributed among the processors. All timings are performed on the Cray XT3

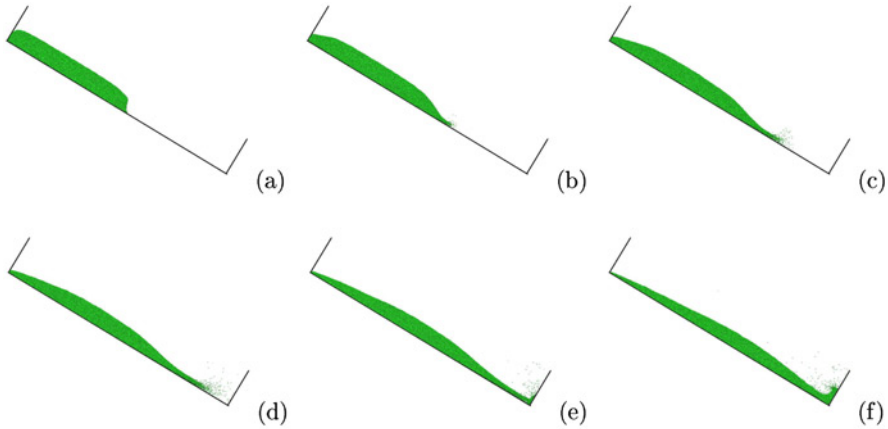
Figure 2.
Parallel speedup and efficiency of the PPM DEM client for fixed-size problems with 635,780 (+), 3,178,900 (×) and 6,357,800 (∗) particles on up to 192 processors

192 processors, thus demonstrating the good linear scaling of the simulation code with particle number.

Figure 3 shows snapshots from one of the avalanche simulations. The system is first equilibrated in the horizontal position for 400,000 time steps before the box is tilted and the right wall removed. The evolution of the avalanche is then simulated for 1.2 million time steps. The relatively low damping of the grains results in a spray-like front (Figures 3(b) and (c)). The front is bend upward and forms a jet as it impacts onto the lower wall Figures 3(e) and (f)).

5. Conclusions

We have presented an efficient parallel DEM implementation based on the PPM library. Using the particle interaction model by Silbert *et al.* (2001, 2002), the implementation has maintained a parallel efficiency of 40 percnt on up to 192 processors of a Cray XT3 machine, and we have presented results from simulations using up to 122 million fully resolved, frictional, viscoelastic particles. This constitutes the first fully discrete application of the PPM library and, to our knowledge, the largest DEM simulations performed so far. A simulation with 630,000 particles takes less than one second per



Notes: Initially, 120,000 grains with a void fraction of 0.468 and uniformly distributed radii between 1.00 and 1.12 equilibrated in a box of dimensions 740 × 200 × 8 mm. At time 0 the box is inclined at an angle of 31.2° and the right wall is removed. We show the flow of the avalanche down the plane 0.1, 0.2, ..., 0.6 seconds after removing the wall (a)-(f)

Figure 3.
Visualization of individual grains from direct DEM simulation of a sand avalanche down an inclined plane

time step on a single processor. Since the present simulation code scales well with both the number of particles and the number of processors, this translates to less than 1 s per time step using 40.7 million particles on 128 processors, or 0.017 s per time step using 635,780 particles on 128 processors.

We have applied the new code to the simulation of sand avalanches down an inclined plane. This system is frequently used as a model to study the macroscopic behavior of granular flows (Daerr and Douady, 1999; Douady *et al.*, 2002). Future work will include calibrating the model parameters by comparing the front propagation speeds to laboratory experiments of sand avalanches, and using the simulation code to help understand the global behavior of granular flows, including phenomena such as fingering (Pouliquen *et al.*, 1997), shear zones (Fenistein and van Hecke, 2003; Cheng *et al.*, 2006), and generation of sound (Douady *et al.*, 2006). The large number of particles that can be treated by the present implementation will allow the extraction of macroscopic material properties (Hunt, 1997) and the validation of the corresponding governing equations on the continuum level (Douady *et al.*, 1999).

References

Altenhoff, A., Walther, J.H. and Koumoutsakos, P. (2007), "A stochastic boundary forcing for dissipative particle dynamics", *Journal of Computational Physics*, Vol. 225, pp. 1125-36.

Balmforth, N.J. and Kerswell, R.R. (2005), "Granular collapse in two dimensions", *Journal of Fluid Mechanics*, Vol. 538, pp. 399-428.

Cheng, X., Lechman, J.B., Fernandez-Barbero, A., Grest, G.S., Jaeger, H.M., Karczmar, G.S., Möbius, M.E. and Nagel, S.R. (2006), "Three-dimensional shear in granular flow", *Physics Review Letters*, Vol. 96, p. 03001.

Cleary, P.W. and Prakash, M. (2004), "Discrete-element modelling and smoothed particle hydrodynamics: potential in the environmental sciences", *Philosophical Transactions of the Royal Society London, Series A*, Vol. 362, pp. 2003-30.

Daerr, A. and Douady, S. (1999), "Two types of avalanche behaviour in granular media", *Nature*, Vol. 399, pp. 241-3.

de Gennes, P.G. (1999), "Granular matter: a tentative view", *Reviews of Modern Physics*, Vol. 71 No. 2, pp. 374-82.

Douady, S., Andreotti, B. and Daerr, A. (1999), "On granular surface flow equations", *European Physical Journal B*, Vol. 11, pp. 131-42.

Douady, S., Andreotti, B., Daerr, A. and Cladé, P. (2002), "From a grain to avalanches: on the physics of granular surface flows", *Comptes Rendus Physique*, Vol. 3, pp. 177-87.

Douady, S., Manning, A., Hersen, P., Elbelrhiti, H., Protiére, S., Daerr, A. and Kabbachi, B. (2006), "Song of the dunes as a self-synchronized instrument", *Physics Review Letters*, Vol. 97, p. 018002.

Dutt, M., Hancock, B., Bentham, C. and Elliot, J. (2005), "An implementation of granular dynamics for simulating frictional elastic particles based on the DL_POLY code", *Computer Physics Communications*, Vol. 166, pp. 26-44.

Fenistein, D. and van Hecke, M. (2003), "Wide shear zones in granular bulk flow", *Nature*, Vol. 425, p. 256.

Heyes, D.M., Baxter, J., Tüzün, T. and Qin, R.S. (2004), "Discrete-element method simulations: from micro to macro scales", *Philosophical Transactions of the Royal Society London, Series A*, Vol. 362, pp. 1853-65.

Hunt, M.L. (1997), "Discrete element simulations for graular material flows: effective thermal conductivity and self-diffusivity", *International Journal of Heat and Mass Transfer*, Vol. 40 No. 13, pp. 3059-68.

Karypis, G. and Kumar, V. (1998), "A fast and high quality multilevel scheme for partitioning irregular graphs", *SIAM Journal on Scientific Computing*, Vol. 20 No. 1, pp. 359-92.

Landry, J.W., Grest, G.S., Silbert, L.E. and Plimpton, S.J. (2003), "Confined granular packings: structure, stress, and forces", *Physical Review E*, Vol. 67, p. 041303.

Moon, B. and Saltz, J. (1994), "Adaptive runtime support for direct simulation Monte Carlo methods on distributed memory architectures", *Proceedings of the IEEE Scalable High-Performance Computing Conference*, *IEEE*, pp. 176-83.

Owen, D.R.J. and Feng, Y.T. (2001), "Parallelised finite/discrete element simulation of multi-fracturing solids and discrete systems", *Engineering Computations*, Vol. 18 Nos 3/4, pp. 557-76.

Pouliquen, O., Delour, J. and Savage, S.B. (1997), "Fingering in granular flows", *Nature*, Vol. 386, pp. 816-7.

Richards, K., Bithell, M., Dove, M. and Hodge, R. (2004), "Discrete-element modelling: methods and applications in the environmental sciences", *Philosophical Transactions of the Royal Society London, Series A*, Vol. 362, pp. 1797-816.

Rycroft, C.H., Bazant, M.Z., Grest, G.S. and Landry, J.W. (2006), "Dynamics of random packings in granular flow", *Physical Review E*, Vol. 73, p. 051306.

Sbalzarini, I.F., Walther, J.H., Bergdorf, M., Hieber, S.E., Kotsalis, E.M. and Koumoutsakos, P. (2006), "PPM – a highly efficient parallel particle-mesh library for the simulation of continuum systems", *Journal of Computational Physics*, Vol. 215, pp. 566-88.

Silbert, L.E., Grest, G.S. and Plimpton, S.J. (2002), "Boundary effects and self-organization in dense granular flows", *Physics of Fluids*, Vol. 14 No. 8, pp. 2637-46.

Silbert, L.E., Deniz, E., Grest, G.S., Halsey, T.C., Levine, D. and Plimpton, S.J. (2001), "Granular flow down an inclined plane: Bagnold scaling and rheology", *Physical Review E*, Vol. 64, p. 051302.

Todorov, I.T. and Smith, W. (2004), "DL_POLY_3: the CCP5 national UK code for molecular-dynamics simulations", *Philosophical Transactions of the Royal Society London, Series A*, Vol. 362, pp. 1835-52.

Verlet, L. (1967), "Computer experiments on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules", *Physical Review*, Vol. 159 No. 1, pp. 98-103.

Vizing, V.G. (1964), "On an estimate of the chromatic class of a p-graph", *Diskret. Anal.*, Vol. 3, pp. 25-30 (in Russian).

Corresponding author
Jens H. Walther can be contacted at: jhw@mek.dtu.dk

To purchase reprints of this article please e-mail: **reprints@emeraldinsight.com**
Or visit our web site for further details: **www.emeraldinsight.com/reprints**