Advances in Engineering Software 84 (2015) 68–76

Contents lists available at ScienceDirect

Advances in Engineering Software

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

The comparison of two domain repartitioning methods used for parallel discrete element computations of the hopper discharge

Darius Markauskas [a], Arnas Kačeniauskas [b,*]

[a] Institute of Mechanics, Vilnius Gediminas Technical University, J. Basanavičiaus g. 28, 03224 Vilnius, Lithuania
[b] Laboratory of Parallel Computing, Vilnius Gediminas Technical University, Saulėtekio al. 11, LT-10223 Vilnius, Lithuania

ARTICLE INFO

Article history:
Available online 8 January 2015

Keywords:
Dynamic domain decomposition
Domain repartitioning
Discrete element method
Multilevel k-way graph partitioning
Recursive coordinate bisection
Hopper discharge

ABSTRACT

The paper presents an application of two domain repartitioning methods to solving hopper discharge problem simulated by the discrete element method. Quantitative comparison of parallel speed-up obtained by using the multilevel k-way graph partitioning method and the recursive coordinate bisection method is presented. The detailed investigation of load balance, interprocessor communication and repartitioning is performed. Speed-up of the parallel computations based on the dynamic domain decomposition is investigated by a series of benchmark tests simulating the granular visco-elastic frictional media in hoppers containing 0.3×10^6 and 5.1×10^6 spherical particles. A soft-particle approach is adopted, when the simulation is performed by small time increments and the contact forces between the particles are calculated using the contact law. The parallel efficiency of 0.87 was achieved on 2048 cores, modelling the hopper filled with 5.1×10^6 particles.

1. Introduction

The discrete element method (DEM) is a numerical technique originally developed by Cundall and Strack [1] for predicting the behaviour of soil grains. The Lagrangian approach is applied to a system of moving particles with a given shape and material properties. The state of the particles is obtained by time integration of the dynamics equations derived from the classical Newtonian mechanics. The Newton's second law is used for translation and rotation of each individual particle. All the forces and moments acting on each particle are evaluated at every time step. Thus, DEM enables us to investigate granular flow characteristics at the particle level, and to evaluate particle motion precisely. The increasing availability of computing power over the past three decades, along with refinements and adaptations of the original method, has made DEM applicable to a wide range of industries. Numerous examples include granular flows, milling, powder mixing, fluidized beds and fracture modelling [2].

The granular flow from hoppers and silos has a wide range of industrial applications. The conducted research is mainly focused on several aspects as follows: wall pressure, discharge rate and internal properties. The study of the bulk material pressure on the walls of a hopper plays a very important role in hopper design [3]. The prediction of the discharge rate is important for effective operation and control of the transport system [4]. The combined approach of DEM and the averaging method offers a possibility to link the fundamental understanding generated from DEM-based simulations to engineering application often achieved by continuum modelling [5].

The main advantage of using DEM for simulation of granular systems is that, by tracking the motion of each individual particle, the detailed information about the system behaviour across a range of time- and length-scales can be obtained. However, the simulation of systems at this level of detail has the disadvantage of making DEM computationally very expensive. DEM simulations on a single workstation or ordinary PC tend to be limited to systems of several tens of thousands of particles and short time intervals. The recent simulation of large-scale systems is performed by employing the parallel computation techniques, though the number of the used particles was still much smaller than that required in industry where typically over a billion particles are dealt with.

In spite of a progress in developing parallel DEM software and a high degree of natural algorithmic concurrency inherent in explicit time integration procedures, a detailed study of parallel performance is rather limited. It is difficult to find the quantitative comparison of speed-ups obtained employing different decomposition strategies and software libraries. Moreover, the presented speed-up analyses are rarely performed in solving the complex applications with a rapidly changing workload configuration. In the present research, two alternative repartitioning strategies for the

* Corresponding author. Tel.: +370 5 274 4913.
E-mail addresses: darius.markauskas@vgtu.lt (D. Markauskas), arnas.kaceniauskas@ vgtu.lt (A. Kačeniauskas).

http://dx.doi.org/10.1016/j.advengsoft.2014.12.002
0965-9978/

dynamic domain decomposition are considered to investigate the benefits of their application to hopper discharge flows. This paper is based upon Markauskas and Kaceniauskas [6], however, the current paper includes the following additional research: the detailed investigation of repartitioning strategies, load balance, interprocessor communication and the solution of a larger benchmark problem.

In Section 2, the related works are overviewed and discussed. Section 3 describes the governing relations of the discrete element method. In Section 4, the employed domain decomposition methods and their software implementation are presented. The measured parallel performance is evaluated and discussed in Section 5, while the concluding remarks are presented in Section 6.

2. The related works

Parallelisation of DEM codes has become an obvious option for rapidly increasing computational capability, along with recent remarkable advances in hardware performance. Early attempts to parallelize computations of particle systems were based on several main ideas, including force decomposition, particle decomposition and domain decomposition [7]. In the first class of methods, each processor holds the information of all the particles, while a predetermined set of force computations is assigned to each processor. Washington and Meegoda [8] divided the interparticle force computation among the processors but stored all particle information on each processor. They achieved a speed-up of 8.7 for 1672 particles with the help of 512 processors. Following the second class of methods, Darmana et al. [9] employed the particle decomposition approach and obtained speed-up of 20 for the simulation of the buoyancy driven flow in a bubble column with 105 bubbles, using 32 processors. The disadvantages of this method are associated with large memory requirements for holding all particle information. Kafui et al. [10] employed the particle subset method, where the particles are divided among the processors based on a graph partitioning algorithm. In this algorithm, a graph of particles connected by contacts is developed and the particles in close proximity are assigned to a single processor. They estimated the performance for the bubbling bed with 5.0×10^4 particles and reported a speed-up of 35 for 64 processors. Plimpton [7] concluded that particle decomposition and force decomposition are not optimal for large simulations due to high communication costs associated with the synchronization of data.

Recent advances in multithreading-type architectures encourage the development of DEM codes based on the above discussed methods for shared-memory parallel computers, particularly, to avoid the memory access conflicts. Attempts to perform straightforward parallelization of DEM codes by using OpenMP have not resulted in very high parallel efficiency and scalability. Frenning [11] obtained a speed-up of 3.75, simulating 700 particles only on 4 cores. Renouf et al. [12] parallelized the NSCD algorithm and attained a speed-up of 11 with 16 cores of SGI Origin 3800, simulating 1016 polydisperse disks. Shigeto and Sakai [13] performed exhaustive investigation and reported a speed-up of 5.4 measured in performing single-precision floating-point computations of 3.2×10^5 particles on Intel Xeon W5590 CPU with 16 logical cores. It is evident that the shared-memory configuration is limited to the number of processors that can efficiently access the common memory.

On the other hand, it is increasingly being found that graphics processing units (GPUs) have a much better performance to price ratio than architectures based on supercomputers and PC clusters. Radeke et al. [14] investigated the influence of the particle size on mixing statistics, modelling more than two million particles per Giga Byte of GPU memory. Shigeto and Sakai [13] also conducted a parallel DEM simulation on GPUs. Nishiura and Sakaguchi [15] developed several novel algorithms for shared-memory concurrent computation of particle simulations and measured their efficiency and scalability on various shared-memory architectures, including GPUs. He could observe the best performance, simulating up to 1.0×10^6 particles on the vector processor SX-9/E, but the quite close performance was attained on GPU Tesla C1060. Xu et al. [16] achieved quasi-real-time simulation of an industrial rotating drum, when about 9.6×10^6 particles were treated with 270 GPUs. In general, the performance of DEM simulation on a GPU was shown to be several dozen times faster than that on a single-thread CPU.

In the third class of methods, the domain decomposition [7] is employed. The basic idea of this technique is the partitioning of the computational domain into subdomains, each being assigned to a processor. Two basic types, static and dynamic domain decomposition strategies, are extensively used in solving time-dependent granular flows. Static domain decomposition works by assuming the fixed interdomain boundaries. Jabbarzadeh et al. [17] applied the parallel link-cells method based on the static domain decomposition to simulation of the short-range interaction of branched molecules. A maximum speed-up of 8.5 was reached by using 16 processors, simulating more than 51,864 molecules. Maknickas et al. [18] employed static domain decomposition with MPI in the parallelization of their DEM code. Static load balancing among the processors was enforced by employing regular partitions with nearly equal numbers of particles. Extensive use of local data structures kept interprocessor communications to a minimum. For 16 processors, a speed-up of 11 and the efficiency of 0.7 were obtained, simulating a 1.0×10^5 particle system. The effect of the material polydispersity on the performance of the static domain decomposition is presented in [19]. The analysis of tri-axial compaction with nearly 1.0×10^5 heterogeneous particles showed a speed-up of 8.81 for 10 processors. Parallel DEM computations on gLite grid infrastructure revealed similar speed-up [20]. A bubbling fluidized bed with 2.5×10^5 particles was simulated by the open source code MFIX based on the static domain decomposition [21]. The DEM part of the solver showed reasonable scalability up to 256 processors with an efficiency of 0.8. However, the dynamically changing workload configuration may lead to load imbalance and low parallel efficiency.

In the case of the hopper discharge problem, processor workload, interprocessor communication, and data storage requirements undergo continuous evolution during the simulation. More flexible, but more complicated dynamic domain decomposition is one of the solutions to this load balance problem that will allow higher scalability in parallel computing performance [22].

In the case of DEM, Owen and Feng [23] used a topological dynamic domain decomposition method based on a dynamic graph repartitioning. However, the parallel speed-up, equal to 4.41 or 5, was measured only on 6 processors of the shared-memory machine SGI Origin 2000. Zhang et al. [24] proposed a fast adaptive balancing method for particle-based simulations, in which a binary tree structure was used to partition the simulation region into subdomains. A higher efficiency of load balancing is obtained, adjusting the balance among the hierarchically grouped domains by compressing and stretching the cells in a group. Fleissner and Eberhard [25] applied an orthogonal recursive bisection of the simulation domain for recursive particle grouping and assignment to parallel processors. The parallel speed-up of 10.2 was achieved by using 16 nodes and simulating 1.0×10^6 of particles.

Walther and Sbalzarini [26] presented large-scale parallel simulations of granular flow, using novel, portable DEM software, employing adaptive domain decomposition based on a multilevel k-way graph partitioning [27] and load balancing techniques.