8th. World Congress on Computational Mechanics (WCCM8)
5th. European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS 2008)
June 30 – July 5, 2008
Venice, Italy

Parallel Programming Techniques for DEM Models on Distributed Memory Machines

***Raju V. Kala, John F. Peters and Robert S. Maier**

U.S. Army Engineer Research and Development Center,
3909 Halls Ferry Road,
Vicksburg, MS 39180, USA
Raju.V.Kala@erdc.usace.army.mil
John.F.Peters@erdc.usace.army.mil
Robert.S.Maier@erdc.usace.army.mil

Key Words: *DEM, Parallel Programming, Particles, Computing Methods, MPI*

ABSTRACT

Since its introduction by Cundal and Strack [1] the discrete element method (DEM) has become a powerful tool to study the micromechanics of materials such as soil or rock systems [2]. DEM studies depend on high performance computer systems to accommodate the large number of particles involved in the systems. The advent of super computers and parallel programming had made these studies possible. This paper addresses one such parallel program (DEM-p) for discrete elements or particles. It is based on the code described by Horner and his colleagues [3][4] that was revised to improve modeling rigid objects, membranes (for modeling tri-axial tests), and contact laws that include resistance against particle rotations. This paper mainly addresses the computational or implementation issues to writing parallel DEM programs using the Message Passing Interface (MPI) Library for communication among the processors.

The principal component of the DEM system is the force calculated for each particle and the displacement of the particle due to this force. Accordingly, contact detection and contact resolution is the most critical and processor intensive operation of the DEM codes. It usually takes up about 65-80% of the total computing time. Contact detection identifies the potential contact pairs whereas contact resolution computes the actual penetration distance between the two particles. A naïve contact detection search would require a check between every particle pair in the system, leading to an algorithm that scales by $O(N^2)$. In larges particle systems, the contact detection is typically localized to immediate neighbors using neighbor cell methods. DEM-p program uses a hierarchal search method [5], to improve efficiency when particles are distributed over large size ranges. For parallel DEM code, the challenge is to distribute cell information among processors such that communication of neighbor cell information is minimized.

Data structures to store and manipulate the data efficiently are essential for good performance. They must accommodate parallelization with minimal communication overhead and without unduly complicating the algorithms. A three dimensional (3-D) cell structure is used to create neighbor lists. The cells are divided along the cell

boundaries and assigned to multiple processors for computations. The cell structure is constructed using a three dimensional array of pointers and facilitates storing the cell contents as a linked list. A hierarchical cell structure [5] is also used to maintain one or more nested 3-D cell structures or levels to store data of the particles of different sizes and is designed to avoid one-to-many and many-to-many potential contact situations. The particles and their contacts must be stored in a way that it is easier to access a particle's contact list to check if a contact had already existed before and update the contact force. Also, the data structures must accommodate movement of a particle from one processor to another with minimal overhead. In addition, the processors must be able to access the particles that are in the border cells (ghost cells) of the neighboring processor during contact detection and resolution.

The performance of the system depends on the number of particles per processor and load balance. DEM-p program uses recursive bisection method to achieve good load balance. The code has been tested for performance for up to 5 million particles on 256 processors and has been ported and tested on several platforms, including the Cray T3E, SGI Origin 3000 and Compaq SC45. The code scales well on all machines although as illustrated through examples, the performance depends on many variables such as particle distribution, problem nature, number of contacts, ratio of the biggest to the smallest particle and particles per processor. Many of these observations appear to be common to all DEM codes, not just the current implementation. All of the testing of DEM-p program was done in parallel using spherical particles, whose locations and radii were given as input in the beginning. The cell space technique is still valid to store the non-spherical data [6] via a bounding sphere that is used to assign these particles to the cells.

REFERENCES

[1] Cundall, P. A., Strack, O. D. L. (1979) A discrete numerical model for granular assemblies, Geotechnique, Vol. 29, No. 1, pp 47-65.

[2] Cundall, P. A. 2002, "A discrete future for numerical modeling" Discrete Element Methods: Numerical Modeling of Discontinua, ASCE Geotechnical Special Publication No. 117, 3-4.

[3] Carrillo, A.R., West, John E., Horner, David A., Peters, J. F. (1999) "Interactive Large-Scale Soil Modeling using Distributed High Performance Computing Environments", The International Journal of High Performance Computing Application, Vol. 13. No.1, pp.38-48.

[4] Horner, D. A., Peters, J. F., and Carrillo, A. J. (2001) "Large Scale Discrete Element Modeling of Vehicle-Soil Interaction," Journal of Engineering Mechanics, Vol. 127, No. 10, pp 1027-1032

[5] Peters, J. F., Kala, R., and Maier, R. S. "A Hierarchal Search Algorithm for DEM with Greatly Differing Particle Sizes", Proceedings of Discrete Element Methods 07.

[6] Peters, J. F., Hopkins, M. A., Kala, R., and Wahl, R. E. "A Polly-Ellipsoid Particle for Non-Spherical DEM", Proceedings of Discrete Element Methods 07.