International Journal of Modern Physics C

A GENERALIZED CELL METHOD FOR HARD DISK MOLECULAR DYNAMICS SIMULATION OF POLYDISPERSE SYSTEMS

MEVLUT BULUT

Department of Physics, University of Alabama at Birmingham, Birmingham, Alabama 35294-1170,USA
mbulut@uab.edu

RENATO P. CAMATA

Department of Physics, University of Alabama at Birmingham, Birmingham, Alabama 35294-1170,USA
camata@uab.edu

Received 18 December 2006
Revised 14 March 2007
Accepted 19 March 2007

Hard Disk Molecular Dynamics (HDMD) techniques often exhibit significant loss in calculation speed when applied to the simulation of highly polydisperse particle systems. The collision rate of reported algorithms may be lower by as much as two orders of magnitude if compared to the collision rate of a monodisperse system of the same number of particles. This is mainly due to the fact that the rectangular cells in the simulation domain used in HDMD methods must meet a certain size criterion. In this paper, we introduce a cell technique that removes the requirement on the cell size enabling simulation of particles with sizes much larger than the cell size. This approach improves the collision rates in the simulation of tested polydisperse systems by factors ranging from 5.5 to 57 depending on the size distribution of the particle population simulated. This may enable the simulation of grand canonical systems in which the size and the number of particles can change throughout the simulation. The technique is compatible with the simulation of disk-like as well as irregularly shaped particles and can be extended to three dimensions.

Keywords: hard disk molecular dynamics; polydisperse particle systems; particle simulation; generalized cell method; event driven molecular dynamics;

PACS Nos.: 11.25.Hf, 123.1K

1. Introduction

Beginning with the work of Alder and Wainwright in the late 1950's[1] and in parallel with the rapid advances in computer technology over the past few decades, efficient algorithms have been developed to realize fast Hard Disk Molecular Dynamics (HDMD) simulations in 2D and Hard Sphere Molecular Dynamics simulations in 3D, also referred to as Discrete Molecular Dynamics by some researchers.[2,3]

The high performance of these algorithms is mainly a result of the utilization of event-driven simulations based upon dividing the simulation domain into rectangular cells,[4] event-scheduling with efficient queuing techniques such as Binary Search Tree,[5] Complete Binary Tree (CBT),[6] Red-black Complete Binary Tree, and Heap Structure[7] as well as the use of invalidation redundancy of obsolete events (also referred to as "check event" approach[7] or "lazy invalidation scheme"[8] in the literature). Some recent efforts to further improve the performance of HDMD include the work by Isobe[9] and Donev et al.[10] Isobe uses Extended Exclusive Particle Grid and realizes a HDMD simulation algorithm free of cell-crossing events at the expense of dynamically updated neighbor particle lists and upper cut-off time calculations.[9] Efficient simulations of particles that deviate slightly from a disk-like shape have also been reported recently by Donev et al.[10] In this case the simulation efficiency for systems of particles with relatively small aspect ratios is achieved by incorporating a Near-Neighbor-List approach using the background cell structure to update the lists of Near Neighbors.[11] These and other advances have enabled efficient simulation of monodisperse systems of particles.

The performance of current HDMD algorithms is considerably lower in the simulation of highly polydisperse particle ensembles. In general, reported algorithms and simulation approaches experience a dramatic speed loss when just a few large particles are added to the particle population.[12] This is mainly due to the fact that the cells in the simulation domain must meet a size criterion.[9,12] Hence, the addition of even a single large particle to the simulation domain may drastically reduce algorithm efficiency (in case commensurate cell structures are used for the sake of mitigating the efficiency degradation, code complexity is increased). Most importantly, the currently available cell-based techniques do not allow the dynamical addition of particles with sizes larger than the cell size unless the simulation is reinitialized.

In this paper we describe a method that removes the condition on the cell size and is compatible with simulation of arbitrarily shaped particles. All particles, regardless of their size or shape are handled in a conceptually identical way. This is achieved by carefully identifying and then generalizing the concepts regarding particle-cell interactions. Once the independence between cell size and particle size is achieved, new opportunities open up for HDMD simulations. Individual particle sizes can be changed at runtime without reinitializing the system, which is a necessary functionality to simulate numerous realistic particle systems that undergo particle growth, such as aerosols and colloids. In addition, this technique makes it straightforward to simulate arbitrarily shaped particles. Moreover, the overall cost of handling large particles in two dimensions grows linearly with their sizes instead of quadratically as in previous HDMD methods. This moves hard disk simulations one step closer to realizing efficient simulations of grand canonical systems in which the size and number of particles can change throughout the simulation as a result of nucleation, coagulation, coalescence, and particle flow in and out of the system. The obtained functionality is also very useful for the study of binary systems with

two populations having very different particle sizes as in the case of large particles undergoing Brownian motion in a colloidal suspension.

2. Preliminary Definitions

In this paper, we use the side of square cells in the simulation domain as our fundamental unit of length and set it equal to unity. We will refer to this unit of length simply as unit. The size of the particles, the size of the simulation domain, and all other distances in the simulation are expressed in terms of this unit. The size distribution of any given particle population must also be expressed in terms of this fundamental unit. This is done through the introduction of a scaling factor that is applied to the entire particle size distribution. Although the value of this scaling factor is in principle arbitrary, it can be optimized to yield the highest simulation speed. Because our method simulates particle ensembles with arbitrary size distributions, it follows naturally that some particles will be smaller than the cell size while others will be larger. This leads to the definition of two types of particles: small particles whose diameters are smaller than that of a cell and large particles whose diameters are larger than or equal to that of a cell.

3. Formalization of Concepts

In the standard implementations of the cell method, different cells play different roles depending on their location with respect to the particles being simulated. These cells may be classified in various categories according to these different roles. Although previous algorithms have made use of these categories implicitly, they have never been formalized in a systematic way. We formally classify the cells in four specific categories: Center Cells, Neighbor Cells, Host Cells and Projection Cells. These categories may be defined as follows.

3.1. *Center Cell and Neighbor Cells*

The Center Cell is the cell in which the coordinates of the center of a particle lie. Two particles can collide only if one occupies a Neighbor Cell of the other. Thus the Neighbor Cells of a given particle constitute the limited region of the simulation domain where collisions between the given particle and other particles are possible. Only cell boundary crossing events change the Neighbor Cells of a particle. The Neighbor Cells can efficiently be identified using the following criteria:

- The Neighbor Cells of a given particle should include all the cells in the simulation domain that other particles must register in before colliding with it,
- The Neighbor Cells of a particle should represent a contiguous region of the simulation domain.

The Neighbor Cells of a small particle comprise the Center Cell and the eight adjacent cells surrounding it. For small particles the number of Neighbor Cells is

therefore fixed and equal to nine.[7] For large particles, these criteria can be engraved into an algebraic expression to be used as the Neighbor Cell condition in a Neighbor Cell identification procedure. For this engraving, consider a simulation domain divided into square cells as illustrated in Fig. 1. Without loss of generality one may

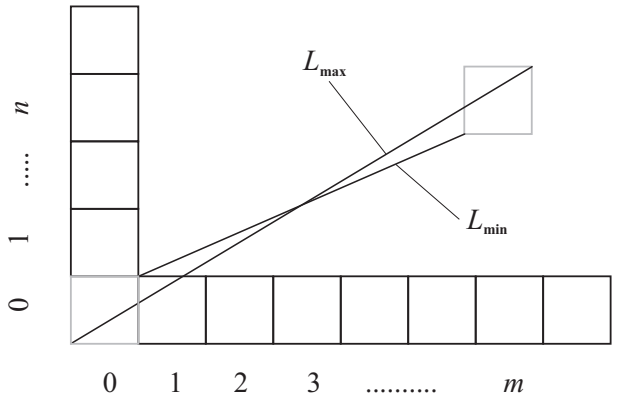

Fig. 1. Schematic diagram showing the distances involved in the identification of the Neighbor Cells using Eq.(1). The cell at coordinates $(0,0)$ is the Center Cell of a large particle. The cell at coordinates (m,n) represents an arbitrary cell that is being considered as a candidate for Neighbor Cell. $L_{\min}$ corresponds to the distance between the two nearest vertices of Center Cell and candidate cell while $L_{\max}$ represents the distance between the two vertices farthest apart. Large particle is not shown in the figure.

consider the cell with coordinates $(0,0)$ as the Center Cell of the large particle with radius R (particle not shown in Fig. 1). Using distances measured between cell vertices, it is clear that any generic cell with coordinates (m,n) may be characterized by two distances with respect to the Center Cell. A minimum distance L_{min} corresponding to the distance between the two nearest vertices, and a maximum distance L_{max} representing the distance between the two vertices farthest apart. In order to be considered as a Neighbor Cell, L_{min} and L_{max} of a cell must satisfy the conditions

$$\begin{aligned} L_{\min} &< R + 0.5 \\ L_{\max} &\geq R + R_0 \end{aligned} \tag{1}$$

where R_0 is the radius of the smallest particle in the system.

Because of the limited precision of numerical calculations, there is a finite probability that one particle will experience two successive binary collisions with zero time elapsed between them. Handling such events as true triple interactions would represent an inordinate use of computational resources since their occurrence is extremely rare. Hence they are handled in this work as two successive binary collisions.

However, if these events involve a binary collision between a small and a large particle, and the sum of their radii is equal to the distance between the farthest vertices of their Center Cells, a collision could be missed leading to particle interpenetration. This undesirable artifact can be avoided by including the equality sign in the $L_{\max}$ condition as shown in the Neighbor Cell identification criteria of Eq. (1). As an example of the accuracy of these identification criteria, Fig. 2 shows the Neighbor Cells identified using Eq. (1) for a large particle with a diameter $D_p = 22$ units.

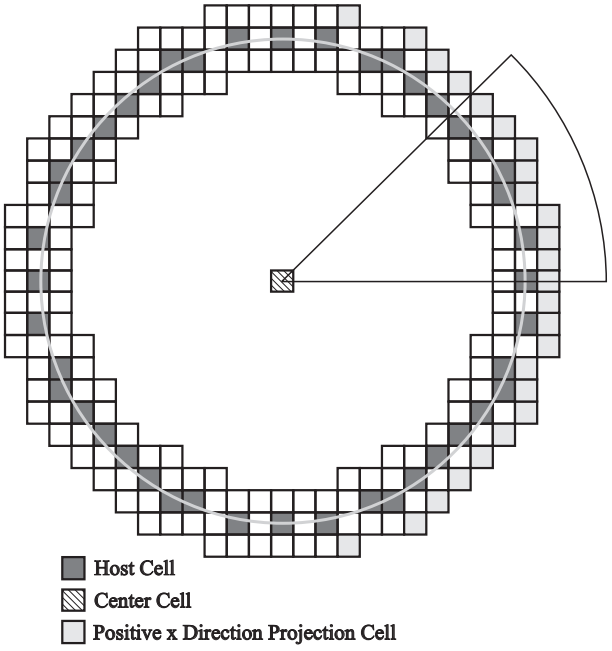

Fig. 2. Trace of a large particle of diameter $D_p = 22$ units with its Center Cell, Neighbor Cells, Host Cells and x direction Projection Cells. Neighbor Cells were identified using the criteria of Eq. (1). Host Cells satisfy the two general rules specified in the text. For a particle with circular symmetry, all the special cells can be obtained by symmetry operations following the identification of these special cells in the first octant.

3.2. *Host Cells*

The Host Cells are the cells in which a particle is registered. When other particles enter one of the Neighbor Cells of a given particle, they become aware of its existence by reading the registered particles in their Neighbor Cells, one of which is a Host Cell of this given particle. Hence the Host Cells are the means a particle uses to inform approaching particles of its presence. For small particles, there is only one Host Cell which is the Center Cell of the particle. Since large particles cannot be contained in a single cell, they must therefore be registered in more than one cell to

guarantee that every other approaching particle will be informed of its existence and the future collisions can be set up correctly. In general, the Host Cells are located along the perimeter of a given large particle and are a sub-set of the Neighbor Cells. Fig. 2 shows the Host Cells for a large particle with $D_p = 22$ units. When a particle undergoes a cell boundary crossing event, its ID is erased from the previous Host Cells and is registered in the new Host Cells. This registration update requires unambiguous identification of the previous Host Cells which will cease to carry the particle ID and the new Host Cells which will now hold the particle ID. The identification procedure for the Host Cells may be carried out in many different ways depending upon the specific implementation. If only a few large particles are entered in the simulation, one can manually identify the Host Cells following the identification of Neighbor Cells by the condition of Eq. (1). The identified Host Cells may be logged to a file to be read at the beginning of the simulation. When many large particles with various sizes (or shapes) are simulated, manual identification of Host Cells is not practical. In this case an algorithm capable of identifying the Host Cells must be generated and executed after the Neighbor Cell coordinates relative to the Center Cell are calculated. There are two general rules to identify the Host Cells:

- Host Cells should be immersed in the Neighbor Cell set by at least one cell. This is to prevent other particles from reading them without being in a Neighbor Cell,
- Host Cells should be as sparsely distributed as possible. This is to reduce the multiplicity of the particle in collision partners list of the neighborhood particles.

3.3. *Projection Cells*

The Neighbor Cell set of a given particle needs to be updated when the particle moves from one Center Cell to another as a result of a cell boundary crossing event. We refer to the cells that are added to or removed from the Neighbor Cell set as Projection Cells. Forward Projection Cells are added to the Neighbor Cell set and backward Projection Cells are abandoned. For the group of abandoned cells, nothing is done but the group of added cells should be processed, i.e., the particles registered in these cells are considered for possible future collisions. The group of newly acquired Neighbor Cells depends on the boundary crossing direction. There are four different groups for the four different crossing directions. Fig. 2 depicts the Projection Cells of a large particle for the positive x direction.

4. Collisions between Large Particles

Successful implementation of collisions between small particles is well established.[4,5] However, special considerations are needed for collisions between large particles. When two large particles approach each other, some of the Host Cells of one particle coincide with some of the Neighbor Cells of the other. This allows the setting up of possible future collisions between the particles. This is similar to the case when a

small particle reads its Neighbor Cells and becomes aware of the presence of a large particle in its vicinity. The marked difference in the case of collisions between large particles is the high multiplicity of one particle ID in the list of collision candidates of the other. This multiplicity can be eliminated by checking the ID of every new particle added to the list of collision candidates against the IDs already in the list. Alternatively, one can simply ignore the multiplicities and allow the simulation to proceed with multiple collision checking calculations for the same two particles. Evidently, both approaches ensue in some computation cost, and a decision must be made on which one is less costly depending on the details of the implementation.

5. Results

Our implementation of this method was developed in Microsoft Visual Studio 2005, Visual C++ programming environment. We used a Dell Precision 380 workstation with 2 GB RAM running XP Professional. Simulations were carried out with a variety of polydisperse particle configurations to assess the efficiency of our approach. In order to quantify the impact of using the new method introduced, we have carried out simulations using two different algorithms. In the case of the first algorithm, the cell size is larger than the largest particle in the simulation. Since this approach is similar to most cell method implementations, we refer to it as the standard cell technique. A second algorithm makes use of the ideas introduced in this paper. Since the new cell categories are in a sense generalized versions of the standard categories, we refer to this approach as generalized cell technique. Because all other aspects of these two algorithms are identical, a direct comparison of their speed in simulating a given system permits a quantitative assessment of the effect of the new approach in the simulation performance. Table 1 presents the results of this comparison.

For all the test simulations the temperature was set to T=300 K and the particle mass density ρ_p was arbitrarily chosen as 1 g/cm^3 (density of water). In the particle populations represented by the first four rows in Table 1, the diameter of the smallest particle was chosen as 10 nm. In the case of the size distribution on the last row of Table 1, the particle median diameter was set to 10 nm. The mass of any given particle, m_p, is thus evaluated from $\pi\rho_p D_p^3/6$, where D_p is the particle diameter. Each particle is given an initial velocity vector with a random direction and a magnitude calculated by $\sqrt{3k_BT/m_p}$. In order to guarantee that the total momentum of the system is zero, the procedure for assigning the initial velocities is carried out two particles at a time. Opposite velocity directions are assigned to each particle in a "pair" so that every pair will have zero linear momentum. Each simulation was initially allowed to run until the number of collisions processed exceeded the number of particles in the system in order to ensure that an equilibrium velocity distribution had been achieved. A simulation domain of 2.2×2.2 μm^2 was used throughout this work.

For each test system depicted in Table 1, simulations were initially run to determine the optimal value of the scaling factor. Subsequently, each system was run five

Table 1. Collision rates obtained for various particle populations by a Visual C++ implementation. The standard cell method refers to an algorithm using the cell size larger than the largest particle in the simulation whereas the generalized cell method is the technique introduced in this paper and uses the new cell categories. The performance ratio represents the ratio of the collision rates obtained with the generalized and standard cell methods. Domain size is equal to 220×220 unit^2 and the uncertainty in the collision rates is within 1%.

Particle Population		Collision Rate (min^{-1})		
Small Particles	Large Particles	Standard Cell Method	Generalized Cell Method	Performance Ratio
$N = 10,000$ Monodisperse $D_p = 0.99$ unit	None	6,776,000	6,776,000	1.0
$N = 10,000$ Monodisperse $D_p = 0.99$ unit	$N = 1$ $D_p = 50$ units	110,000	6,293,000	57
$N = 10,000$ Monodisperse $D_p = 0.99$ unit	$N = 30$ Linear distribution from $D_p = 1$ unit to $D_p = 25$ units	367,000	6,240,000	17
$N = 10,000$ Monodisperse $D_p = 0.99$ unit	$N = 200$ Linear distribution from $D_p = 1$ unit to $D_p = 25$ units	733,000	5,183,000	7.1
Log-normal distribution of 10,000 particles: $\overline{D}_{pg} = 1$ unit		798,000	4,405,000	5.5

times using the optimal scaling factor in order to obtain an average value for the collision rate, which we here use as a parameter to quantify the algorithm efficiency. For each particle population examined, Table 1 presents a comparison between the efficiencies of the standard cell technique and the generalized cell technique. The uncertainty in the values of the collision rates presented is about 1%. We initially performed simulations on a monodisperse system of 10,000 small particles with a 0.2 particle-per-cell density (first row in Table 1). As shown in the table, this system of monodisperse particles with a diameter $D_p = 0.99$ units is simulated at a rate of 6,776,000 collisions per minute (min^{-1}) by both the standard algorithm and the generalized cell technique. The addition of a single large particle with a diameter $D_p = 50$ units to the simulation domain reduces the collision rate by a dramatic 98% in the simulation by the standard algorithm. On the other hand, only a marginal loss of 7% in the collision rate is noted in the simulation carried out with the generalized cell technique. This illustrates the significant improvement in efficiency (by a factor of 57) achieved by the new method. Table 1 also shows the results for systems with linear distributions of large particles. Physically realistic particle systems often exhibit complex size distributions. In the case of aerosols, for example, log-normal size distributions are commonly observed.[14] The log-normal

distribution may be represented by:

$$n(D_p) = \frac{N}{\sqrt{2\pi} D_p \ln \sigma_g} \exp\left[-\frac{\left(\ln D_p - \ln \bar{D}_{pg}\right)^2}{2 \ln^2 \sigma_g}\right] \tag{2}$$

where $n(D_p)$ is the number density of particles with diameter D_p. The parameters $\bar{D}_{pg}$ and σ_g in Eq. (2) represent the median diameter and the geometric standard deviation of the particle population, while N is the total number of particles. In order to assess how the generalized algorithm fares on a direct comparison with the standard cell technique in simulating a particle system with more realistic size distributions, we simulated a system with $N = 10,000$, particles distributed in size according to the log-normal function of Eq. (2). $\bar{D}_{pg}$ was chosen as one unit of length while σ_g was set to 1.45, which is a typical value observed in coagulation aerosols.[15] Table 1 shows that the generalized method outperforms the standard approach in this case by a factor greater than five. Collision rate figures similar to those reported in Table 1 have been observed for simulation of systems containing up to N = 2,000,000. These results show that the presence of a few large particles in the simulation domain has only a marginal effect on the collision rate in the generalized cell technique whereas it may slow down the standard cell technique by as much as two orders of magnitude.

6. Conclusions

We introduced a generalized cell method that improves the efficiency of HDMD simulations of polydisperse particle systems. By generalizing concepts previously used in HDMD approaches, the collisions involving large particles can be reliably calculated. All particles with diameters less than one unit of length are handled as small particles whereas all particles with diameters equal to or greater than one unit of length are handled as large particles. Quantitative results of our implementation have been presented in the simulation of a system of ten thousand particles. Results show that with proper scaling, the new method brings significant performance gains compared to standard cell techniques when the simulated system is highly polydisperse. Collision rates higher by factors ranging from 5.5 to 57 have been observed depending on the size distribution of the particle population simulated. One important advantage of this generalized method is that one can change the particle sizes during the simulation without having to change the cell size and reinitialize the system. When the size of a particle is changed the only required task is to process this particle as if it has been involved in a collision plus an update of its Neighbor Cell, Host Cell, and Projection Cell lists. Application of this technique to irregularly shaped and non-symmetrical (concave or convex) particles is also possible by generalizing the collision-check and collision implementation methods so that every particle can use its own shape function.

Acknowledgments

This research was supported in part by grants from the National Aeronautical and Space Administration (NASA/EPSCoR, Grant Number NCC5-580) and the National Science Foundation (NSF-NIRT program, Grant DMR-0402891). M. B. also acknowledges support from the University of Alabama at Birmingham (Preparing Future Faculty Award). Computational resources for this work were provided by the School of Natural Sciences and Mathematics and the Department of Physics at the University of Alabama at Birmingham.

References

1. B. J. Alder and T. E. Wainwright, *J. Chem. Phys.* **31**, 459 (1959).
2. Y. Zhou, C. K. Hall, and M. Karplus, *Phys. Rev. Lett.* **77**, 2822 (1996).
3. B. Urbanc, J. M. Borreguero, L. Cruz, and H. E. Stanley, *Methods Enzymol.* **412**, 314 (2006).
4. J. J. Erpenbeck and W. W. Wood, *Molecular dynamics techniques for hard-core systems. In Statistical Mechanics, Part B, Bruce J. Berne* (Plenum, New York, 1977).
5. D. C. Rapaport, *J. Comput. Phys.* **105**, 367 (1993).
6. M. Marin, D. Risso, and P. Cordero, *J. Comput. Phys.* **109**, 306 (1993).
7. H. Sigurgeirsson, A. Stuart, and W. Wan, *J. Comput. Phys.* **172**, 766 (2001).
8. B. D. Lubachevsky, *Int. J. Comput. Simul.* **2**, 372 (1992).
9. M. Isobe, *Int. J. Mod. Phys. C* **10**, 1281 (1999).
10. A. Donev, S. Torquato, and F. H. Stillinger, *J. Comput. Phys.* **202**, 737 (2005).
11. K. Shida and Y. Anzai, *Comput. Phys. Commun.* **69**, 317 (1992).
12. M. P. Allen, D. Frenkel, and J. Talbot, *Comp. Phys. Rep.* **9**, 301 (1989).
13. A. Krantz, *ACM Trans. Model. Comput. Simulat.* **6**, 185 (1996).
14. J. H. Seinfeld, *Atmospheric Chemistry and Physics of Air Pollution* (John Wiley & Sons, New York, 1986).
15. M. K. Wu and S. K. Friedlander, *J. Aerosol Sci.* **24**, 273 (1993).