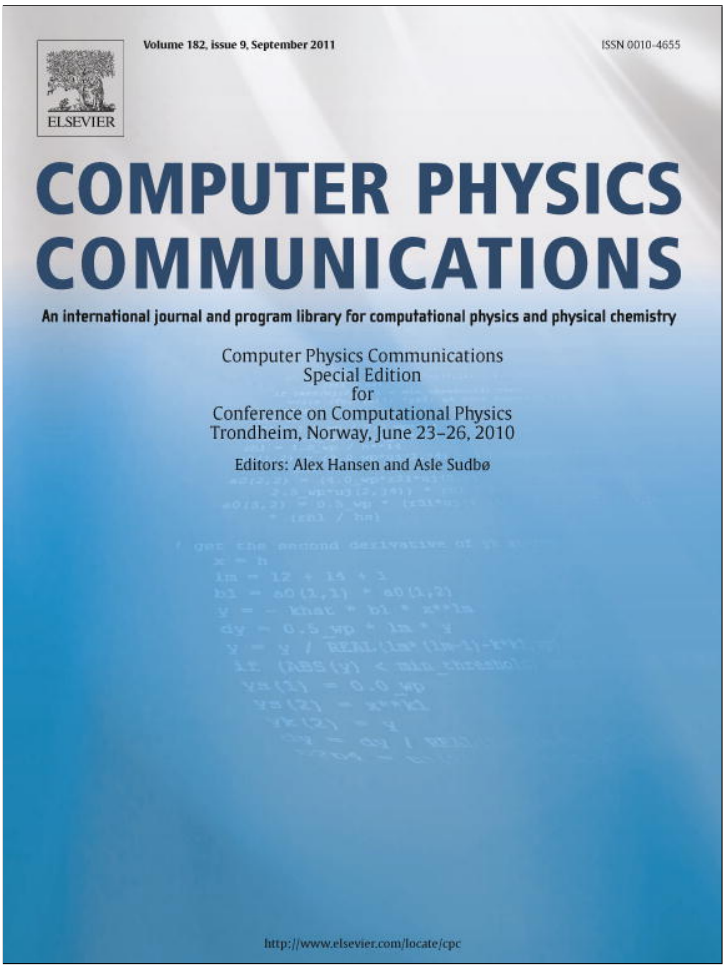

Computer Physics Communications 182 (2011) 1846–1850

Contents lists available at ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

Optimizing event-driven simulations

C. De Michele *

Dipartimento di Fisica, "Sapienza" Università di Roma, P.le Aldo Moro 2, 00185 Roma, Italy

ARTICLE INFO

Article history:
Received 30 July 2010
Received in revised form 4 November 2010
Accepted 10 November 2010
Available online 19 November 2010

Keywords:
Event-driven molecular dynamics
Molecular liquids
Hard rigid bodies
Sticky spots
Linked cell list
Nearest neighbor list
Computer simulations

ABSTRACT

Event-driven molecular dynamics is a valuable tool in condensed and soft matter physics when particles can be modeled as hard objects or more generally if their interaction potential can be modeled in a stepwise fashion. Hard spheres model has been indeed widely used both for the computational and theoretical description of physical systems. Recently further developments of computational techniques allow simulations of hard rigid objects of generic shape. In the present paper we will present some optimizations for event-driven simulations that offered a significant speedup over previous methods. In particular we will describe a generalization of the well-known linked cell list method and an improvement on the nearest neighbor list method recently proposed by us.

1. Introduction

Systems composed of many particles can be modeled as hard rigid bodies (HRB) if excluded volume interactions are dominant and despite the absence of any attraction they exhibit a rich phase diagram especially if their shape is non-spherical [1,2]. The spherical version of these models has already proven to be quite flexible and have been used to tackle, for example, several biological problems [3–5]. The generalization to non-isotropic objects increase even more their flexibility and applicability.

Furthermore also attractive interactions between HRBs, provided that they are short-ranged and/or localized, can be properly modeled employing sticky spots (SS) [6–8].

Several numerical techniques have been developed in the past to perform molecular dynamics simulations of particles interacting with only excluded volume interactions. Dealing with hard bodies, the system is propagated in configuration space from one event to next, giving rise to so-called event-driven molecular dynamics (EDMD). The essence of these EDMD numerical algorithms involves the evaluation of the overlap between different objects [9–12] or, equivalently, their geometrical distance [13].

Recently we proposed a novel method for performing EDMD of HRBs [13], that is summarized in Sections 2 and 3. In the present paper we discuss two optimizations applicable to such EDMD, namely: multiple linked cell lists (MLL) method and nearest neighbor lists with null sticky spots (SNL).

SNL method offers a significant speedup for simulating particles with complicated shapes, like hard-ellipsoids (HE) [1] or superquadrics (SQ) [10,13] and it is fully described in Section 4. In Section 5 we investigate the performance of the SNL method in the case of SQs. MLL method, that is discussed in Section 6, proved to be very useful in simulating mixtures of hard spheres with very different sizes and for such systems the performance of this novel method is analyzed in Section 7. Finally in Section 8 the conclusions are drawn.

2. Event-driven molecular dynamics of hard rigid bodies

In EDMD events, such as collisions between particles, cell-crossings (if one uses linked cell lists), saving of a system snapshot, output of measured quantities, etc. are processed serially one after another. Hence given the set of all possible events Σ, commonly called "the event calendar", it is crucial to adopt an efficient strategy in order to retrieve the next event to happen, to add new events to Σ or to delete events from Σ. In the present case such event calendar has been implemented using a hybrid approach as explained in [14], where a bounded priority queue is built on top of a binary tree implemented as described in Ref. [15]. All operations (insertion, deletion and retrieve of next event) with such priority queue have a complexity $O(1)$ with respect to particles number. In order to avoid round-off problems for events very close to each other, we shift forward time origin periodically.

A general algorithm to perform an EDMD of HRBs is outlined in Fig. 1, where after an initialization step at the begin of the simulation (step 1), events are serially retrieved from the event calendar (step 2) and processed accordingly (steps 4–6). In steps 4(f) and

* Tel.: +39 0649913524; fax: +39 064463158.
E-mail address: cristiano.demichele@roma1.infn.it.

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2010.11.012

1. Initialize the event calendar (predict collisions, cell-crossings, etc.).
2. Retrieve next event $\mathcal{E}$ and set the simulation time to the time of this event.
3. If final time has been reached terminate.
4. If $\mathcal{E}$ is the "NNL rebuild" event then:
   (a) remove all events from calendar,
   (b) update the system to current time,
   (c) evaluate the escape times t_i^{bb} of all particles from their BB and calculate the smallest escape time $t_r = \min_i\{t_i^{bb}\}$,
   (d) using LL check for overlaps between BBs and build the NNL accordingly,
   (e) predict all cell-crossings and schedule them,
   (f) predict all the collisions between HRBs using the NNL and schedule them,
   (g) schedule next "NNL rebuild" event,
   (h) schedule all other remaining events removed from calendar (output of summary, etc.).
5. If $\mathcal{E}$ is a collision between particles A and B then:
   (a) change angular and center-of-mass velocities of A and B according to conservation laws of momentum and energy,
   (b) remove from calendar all events (collisions, cell-crossings, etc.) in which A and B are involved,
   (c) evaluate new collision times t_A^{bb} and t_B^{bb} of A and B with their BBs and schedule a new "NNL rebuild" event if t_A^{bb} or t_B^{bb} are less than t_r,
   (d) predict and schedule the two cell-crossings events for A and B,
   (e) predict and schedule all possible collisions for A and B using the NNL of A and B.
6. If $\mathcal{E}$ is a cell-crossing of a certain rigid body A:
   (a) update LL data structures accordingly,
   (b) remove from calendar all events (collisions, cell-crossings, etc.) in which A is involved,
   (c) predict and schedule all possible collisions for A using NNL of A.
7. Go to step 2.

Fig. 1. Outline of EDMD algorithm for simulating HRBs.

5(e) the collision time between two particles must be evaluated and an algorithm for such purpose has been extensively discussed in Ref. [13]. In Ref. [13] we have also shown how to make use of linked cell lists (LL) and nearest neighbor lists (NNL) with generic HRBs. The remaining aspects of the algorithm presented in Fig. 1 are pretty standard and the interested reader can find all the details also in textbooks [15,16].

We will not provide all the details of the general EDMD developed in Ref. [13], but, since the two novel optimization techniques presented in this paper concern NNL and LL methods described therein, we believe that it is useful to discuss them briefly. NNL are widely used in time-driven molecular dynamics simulations but the implementation in an EDMD is not straightforward [17]. In [13] we proposed the following NNL implementation for EDMD: if one considers the oriented bounding box (BB) which is built around each particle and which encloses it completely, the nearest neighbor list for a given particle A is the set of particles having their BB overlapping with the BB of A. Moreover each BB associated to particle i is immobile and encloses only the particle i; if t_i^{bb} is the time when the particle i will collide with its BB (all the details for its calculation can be found in Ref. [13]), the NNL will have to be rebuilt not after the time $t_r = \min_i\{t_i^{bb}\}$. In addition, if a collision between two particles i and j occurs, then the new escape times t_i^{bb} and t_j^{bb} of these two particles with their BBs have to be evaluated and the new time $t_r' = \min\{t_r, t_i^{bb}, t_j^{bb}\}$ for rebuilding the NNL has to be set. If HRBs have a pronounced non-spherical shape, i.e. a large aspect ratio, it is mandatory to use NNL in order to minimize the number of collision predictions.

LL are also commonly employed in molecular dynamics simulations in order to avoid to check all the N^2 possible collisions among N simulated particles. In the LL method, the simulation box is partitioned into cells and only collisions between particles inside the same cell or its 26 adjacent cells are accounted for. This also means that, whenever an object crosses a cell boundary going from cell a to a new cell b, it has to be removed from the linked cell list of a and added to the linked cell list of b. In the algorithm illustrated in Fig. 1 LL are not directly used to calculate collision times between particles but they are used to rebuild the NNL instead. Given a BB corresponding to an HRB labelled by A, one searches for overlapping BBs in the same cell of A or in one of the 26 neighboring cells, assuming that the cells side length is greater than the length of the diagonal of the BBs. We note that if one wants to use only LL without NNL step 4 in Fig. 1 must be ignored and in steps 6(c) and 5(e) LL will be used instead of NNL to predict collisions between particles.

3. Simulating hard rigid bodies with sticky spots

Particles interacting through a hard core potential may be decorated with spherical sticky spots (SS) that interact via a square well potential [8]. In this case for predicting the next possible collision between two particles[1] we have to take into account both collisions between SSs and hard core collisions between HRBs in steps 4(f), 5(e) and 6(c) of Fig. 1. More specifically if t_{SW} is the next collision time between the SSs of two particles A and B and t_{HC} is the next collision time between their hard cores, the next event between A and B will occur at time $t_{next} = \min\{t_{SW}, t_{HC}\}$. All the details for an efficient algorithm to find collision times between SSs can be found in [6].

SSs belonging to a particle i have also to be considered in calculating the escape time from the BB[2] of i, i.e. in steps 4(c) and 5(c) of the algorithm illustrated in Fig. 1. If $t_{HC}^{bb}(i)$ is the escape time of the i-th HRB from its BB and $t_{SS}^{bb}(i)$ is the escape time of its SSs, then the escape time t_i^{bb} of such particle from its BB will be $t_i^{bb} = \min\{t_{HC}^{bb}(i), t_{SS}^{bb}(i)\}$. $t_{HC}^{bb}(i)$ can be calculated tracking the time evolution of the geometrical distances between the HRB and its BB planes $d_{i_p}(t)$, where $i_p = 1, \ldots, 6$ labels each of the 6 BB planes, as described in detail in Ref. [13]. In a similar fashion the escape time of the SSs can be calculated tracking the simultaneous evolution of all the geometrical distances $d_{i_s i_p}(t)$ between the SSs labelled by i_s and the 6 BB planes labelled again by i_p. An efficient strategy to find the zeros of the distances $d_{i_s i_p}(t)$, similar to the one discussed in Ref. [6] for collisions between SSs, is outlined in Fig. 2.

The algorithm consists in adaptively increasing the time looking for any sign change of the tracked distances $d_{i_s i_p}(t)$. Step 6 requires a short discussion because it corresponds to the possible occurrence of a "grazing collision" between an SS and a BB plane, i.e. a collision in which the SS and the BB plane touches slightly [9,13]. The quadratic interpolation, performed in order to find if the distance becomes negative within the time interval $[t, t + \Delta t]$, does not ensure that the "grazing" collision is not missed. Anyway to remedy this problem a simple and effective trick is to consider a BB whose dimensions (i.e. the sides lengths) are $2\epsilon_d$ smaller than the dimensions of the corresponding BB which is used to build the NNL by checking for overlaps with BBs associated to different particles.

4. A novel method to compute the escape time

In [13] the calculation of the escape time t_{HC}^{bb} of an HRB from its enclosing BB (steps 4(c) and 5(c) in Fig. 1) requires the evaluation of the collision time between the HRB and its BB. This is computationally quite expensive (although much faster than the prediction of the collision time between two HRBs) and a trick to reduce the cost of such calculation is to consider a polyhedron enclosing the HRB and to evaluate the escape time of such polyhedron from its BB (see Fig. 3). This polyhedron must fit the HRB, i.e. given a certain shape (e.g. a parallelepiped) it should have the smallest possible size in order to enclose completely the HRB.

[1] Particle here means "HRB plus its SSs".
[2] Note that in this case the BB must enclose both the HRB i and all its SSs.

1. Set $t = t_0^i$, where t_0^i is the current time of particle i.
2. Evaluate all the distances $d_{i_s i_p}(t)$ between the SSs (i_s) of particle i and the 6 planes (i_p) of its BB. Considering that the plane i_p divides the space in two half spaces, the distance $d_{i_s i_p}(t)$ is positive if the SS i_s is completely inside the half space which contains the particle i.
3. Choose a time increment Δt as follows:

$$\Delta t = \begin{cases} \min_{i_s i_p}\{d_{i_s i_p}(t)/\dot{d}^{\max}_{i_s i_p}\} & \text{if } d_{i_s i_p}(t) > \epsilon_d; \\ \epsilon_d / \min_{i_s i_p}\{\dot{d}^{\max}_{i_s i_p}\}, & \text{otherwise,} \end{cases} \tag{1}$$

   where ϵ_d is a parameter which has to be much smaller than the particle dimension and $\dot{d}^{\max}_{i_s i_p}$ is an overestimate of $\dot{d}_{i_s i_p}(t)$ (time derivative of the distance $d_{i_s i_p}(t)$) within the time interval $[t, t + \Delta t]$ (see Ref. [6] for an explicit expression for this quantity).
4. Evaluate the distances at time $t + \Delta t$, i.e. calculate $d_{i_s i_p}(t + \Delta t)$.
5. For every SS i_s and BB plane i_p such that $d_{i_s i_p}(t + \Delta t) < 0$ and $d_{i_s i_p}(t) > 0$ calculate their contact time with best possible accuracy.
6. For every SS i_s and BB plane i_p such that $0 < d_{i_s i_p}(t) < \epsilon_d$ and $0 < d_{i_s i_p}(t + \Delta t) < \epsilon_d$, evaluate the distances $d_{i_s i_p}(t + \Delta t/2)$, perform a quadratic interpolation of the points $(t, d_{i_s i_p}(t))$, $(t + \Delta t/2, d_{i_s i_p}(t + \Delta t/2))$, $(t + \Delta t, d_{i_s i_p}(t + \Delta t))$ and find the time $t_{\min}$ corresponding to the minimum of the resulting parabola. If $d_{i_s i_p}(t_{\min}) < 0$ calculate the contact time between i_s and i_p with best possible accuracy.
7. If no contact time has been found in steps 5 and 6 go to step 8, otherwise find the smallest contact time among the ones calculated in steps 5 and 6 and terminate, because this time is just the escape time t^{bb}_{SS}.
8. Increment time by Δt.
9. Go to step 2.

Fig. 2. Outline of the algorithm to find the escape time t^{bb}_{SS}.

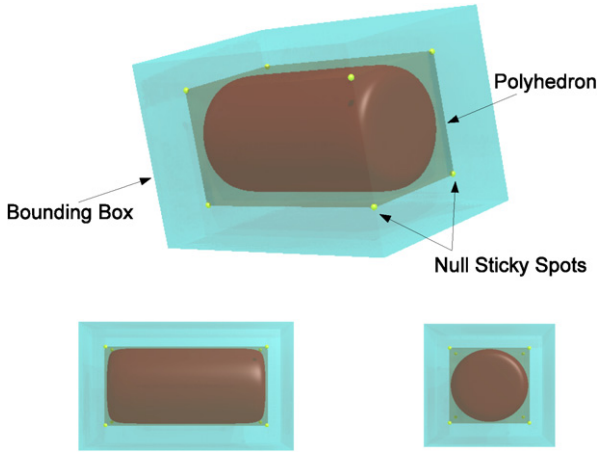

Fig. 3. SQ enclosed in its bounding box (cyan parallelepiped) with SSs of null diameter, represented here as finite size yellow spheres. (For interpretation of colors in this figure, the reader is referred to the web version of this article.)

Latter requirement will ensure that the escape time of such polyhedron t^{sbb}_{HC} will be an underestimate of the escape time t^{bb}_{HC}, i.e. $t^{sbb}_{HC} < t^{bb}_{HC}$. Evaluation of t^{sbb}_{HC} consists in calculating the smallest escape time of all vertices of the polyhedron. Because such vertices can be thought as SSs of null diameter, the same algorithm illustrated in Section 3 and used for calculating the escape time of SSs from the HRB BB can be employed to evaluate the escape time t^{sbb}_{HC}.

In Fig. 3 the polyhedron is a parallelepiped which encloses the given cylindrical-like HRB and whose vertices are represented as yellow finite size spheres. Note that if the shape of particles is more complicated than the one shown in Fig. 3 a polyhedron with a better fitting shape can be used. The improved efficiency of this novel method for evaluating the escape time of HRBs, which we called SNL method, is due to the fact that there is no need to calculate numerically any Jacobian and its inverse for determining distances and contact times, as in the method originally proposed in [13].

5. Nearest neighbor lists with null spots: Performance results

In this section we will test the performance of the novel method described in Section 4 within the algorithm discussed in Section 2. We consider here superquadrics (SQ), whose surface is defined as follows:

$$f(x, y, z) = |x/a|^n + |y/b|^m + |z/c|^p - 1 = 0 \tag{2}$$

where the parameters n, m, p are real numbers and a, b, c are the SQ semi-axes. A monodisperse system of $N = 512$ SQs has been simulated with $n = 8$, $m = p = 2$ and with two equal semi-axes, i.e. $b = c$. Such SQs can be characterized by the elongation $X_0 = a/b$ and if elongation $X_0 < 1$ particles are called "oblate", while if $X_0 > 1$ particles are called "prolate". For this test we have taken into account only prolate SQs, whose shape resembles that of a cylinder with smoothed edges (see Fig. 3). In particular elongations $X_0 = 1.0, 2.0, 3.0$ have been studied. The system of prolate SQs has been simulated in a cubic box of volume V with periodic boundary conditions at the volume fractions $\phi = 0.20, 0.30, 0.35, 0.40$. The length of the smallest semi-axes is chosen to be the unit of length ($b = 1.0$), the mass of the SQ is the unit of mass ($m = 1$) and the moment of inertia is spherically symmetric and equal to 1.0. To create the starting configuration at a desired ϕ, an extremely diluted crystal has been melted; afterwards, the particles have been grown independently up to the desired packing fraction (quench in ϕ at fixed N, X_0), similarly to what was done in [13].

To test the algorithm speed, we use the *CPU time*[3] per collision, i.e. $\tau_c = T_{tot}/N_{coll}$, where T_{tot} is the (real) time needed to perform N_{coll} collisions during a simulation.

We can define the speedup S_{SQ} as follows:

$$S_{SQ} = \tau_c^N / \tau_c^S \tag{3}$$

where N refers to simulations that use NNL originally proposed in [13] and S refers to simulations that make use of SNL discussed in Section 4. Fig. 4(a) shows S_{SQ} as a function of elongation X_0 for two different volume fractions and it is apparent that use of SNL offers a speedup around 2 independently of elongation. In a similar way Fig. 4(b) shows the speedup S_{SQ} for two elongations $X_0 = 1.0$, 3.0 as a function of ϕ and it turns out again that S_{SQ} is also independent of ϕ. Independence of X_0 and ϕ for the speedup S_{SQ} stems from the fact that the number of collision predictions for a given X_0 and ϕ is nearly independent of the method used to calculate the escape time (either SNL or NNL).

6. Polydisperse systems: Multiple linked cell lists

Several variants of original LL method have been proposed in literature [18] with a view to improving the original LL method (as discussed for example in [16]). All these improved LL methods are intended to avoid unnecessary distance calculations but they do not tackle the case of a system composed of different species having very different sizes. Here we consider the case of a system composed of N_s several species having different sizes $\{\sigma_1, \ldots, \sigma_{N_s}\}$ and for the sake of simplicity we assume that particles are spherical. We use again the EDMD algorithm discussed in Section 2 but without NNL. Therefore the prediction of collision times relies only on LL (or MLL).

For making use of the original LL method, the simulation box must be partitioned into cubic cells and each cell must have a side length greater than $\sigma_{\max} = \max_i\{\sigma_i\}$. In general, this restriction can considerably compromise the algorithm performance. To understand this, consider m_i particles of diameter σ_i inside a cell and

[3] *CPU time* means the real time spent by the CPU for calculations.

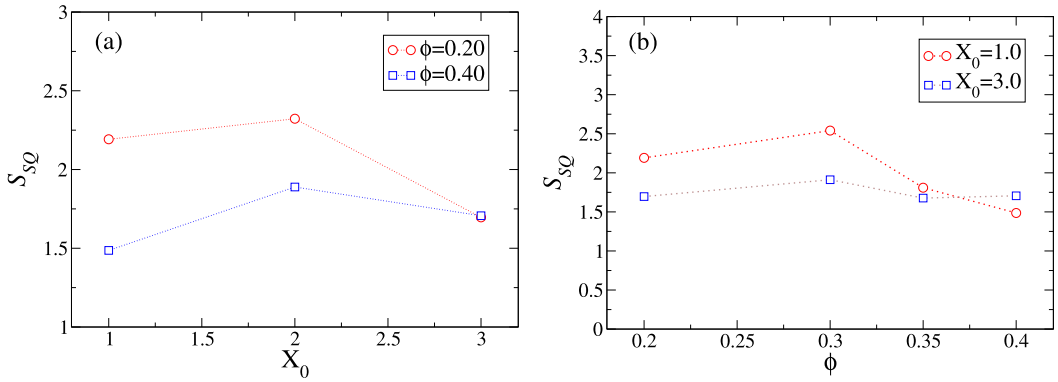

Fig. 4. (a) Speedup S_{SQ} versus elongation X_0 for $\phi = 0.20, 0.40$. (b) Speedup S_{SQ} versus ϕ for $X_0 = 1.0, 3.0$.

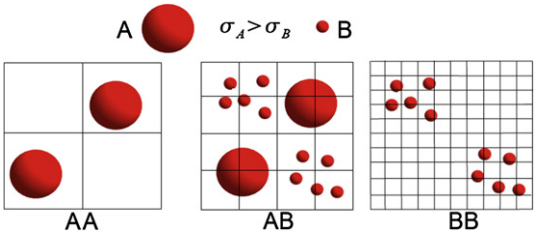

Fig. 5. Pictorial representation of Multiple Linked Lists for a binary mixture of two particles where diameter σ_A of particles A is bigger than diameter σ_B of particles B. For each possible interaction AA, AB and BB a different partitioning of simulation box into cells is employed, i.e. multiple linked cell lists are used.

$q_i = (\sigma_{\max}/\sigma_i)^3$, at a fixed volume fraction, if $q_i \to 0$ then $m_i \to \infty$ and the performance is severely compromised.

A possible generalization of the original LL method for dealing with such polydisperse system, which we called the MLL method, consists in using a different LL for each pair of species. To implement the MLL method, for each pair of species a different subdivision of the simulation box into cells is used, as illustrated pictorially in Fig. 5 for a binary mixture of hard spheres. In general if the number of species is N_s, $N_s(N_s+1)/2$ different subdivisions of the simulation box into cells are needed and the side length of cells associated to species l and m[4] must be greater than $(\sigma_l + \sigma_m)/2$. A cell associated with species l and m will be denoted by c_{lm} and each particle of species k belongs to all cells c_{lm} with $l = k$ or $m = k$, inside which center of A lies. The set of cells to which A belongs will be indicated by $\mathcal{C}(A)$. For predicting possible collisions of a particle A, all the particles inside the cells $\mathcal{C}(A)$ and in their neighboring cells have to be checked except for particles of the same species of A belonging to cells c_{lm} with $l \neq m$ in order to not take into account collisions between particles of same species twice.[5] As in the original LL method the program keeps track of cell changes for all cells to which a particle belongs and cell-crossings events are scheduled into the event calendar. The only remark for MLL method is that if particles, as they cross simulation box boundaries, are reinserted into the box with periodic boundary conditions, one has to ensure that reinsertion happens only once.

7. Speedup of multiple linked cell lists

In this section we will test performance of the MLL method for a binary mixture of spheres having a very different size. The two species (labeled by A and B) are characterized by a diameter ratio $q = \sigma_A/\sigma_B > 1$ and their masses are chosen to be equal and unitary. N_A and N_B will be the number of particles A and B respectively and $N = N_A + N_B$. The number of particles A will be kept fixed to 250, i.e. $N_A = 250$. We will investigate the algorithm performance varying N, the volume fraction ϕ and the size ratio q. The simulation box is cubic with periodic boundary conditions.

Again we define the speedup S_{SQ} as follows:

$$S_{BM} = \tau_c^{LL}/\tau_c^{MLL} \tag{4}$$

where LL refers to simulations that use the original LL method proposed in [13], MLL refers to simulations that employ the new MLL method described in Section 6 and τ_c is the *CPU time* per collision. Fig. 6(a) shows S_{BM} as a function of ϕ for two different size ratios q. It is remarkable that speedup reaches values up to 40.

The number of spheres of type B, N_B^{cell}, within a cell using conventional LL method is roughly:

$$N_B^{cell} = \frac{6q^3\phi}{\pi[q^3(1-\phi)N_A/N_B + 1]} \tag{5}$$

and an analytical estimate S_{BM}^{th} for the speedup turns out to be[6]:

$$S_{BM}^{th} = K N_B^{cell} \tag{6}$$

where K is an arbitrary constant. Fig. 6(a) and (b) shows also fits to simulations data of function S_{BM}^{th} defined in Eq. (6). Agreement between numerical data and S_{BM}^{th} proves that S_{BM}^{th} is a reasonable estimate.

8. Conclusions

In this paper two novel techniques easy to implement have been proposed for optimizing the EDMD algorithm summarized in Fig. 1 and extensively discussed in Refs. [6,13]. The SNL method offers a nearly constant speedup around 2 with respect to the old method proposed in [13] and it can be easily adapted to more complicated shapes of simulated particles. The SNL method is actually used for simulating a recently developed model of DNA duplexes (DNAD) [19] consisting in cylindrical-like SQs decorated with two sticky spots on their two bases in order to model stacking interactions between DNADs. The details of latter study will be given in a future publication.

MLL have been already used in [20] where spherical particles, evolving according to event-driven Brownian dynamics [21], can be absorbed by one fixed spherical sink whose size may be much greater than the diffusing particles. MLL are also currently used

[4] Assuming that they are additive.

[5] Possible collisions of A with other particles of same species k are already considered checking particles inside cells c_{kk}.

[6] For MLL method N_B^{cell} is of the order of 1.

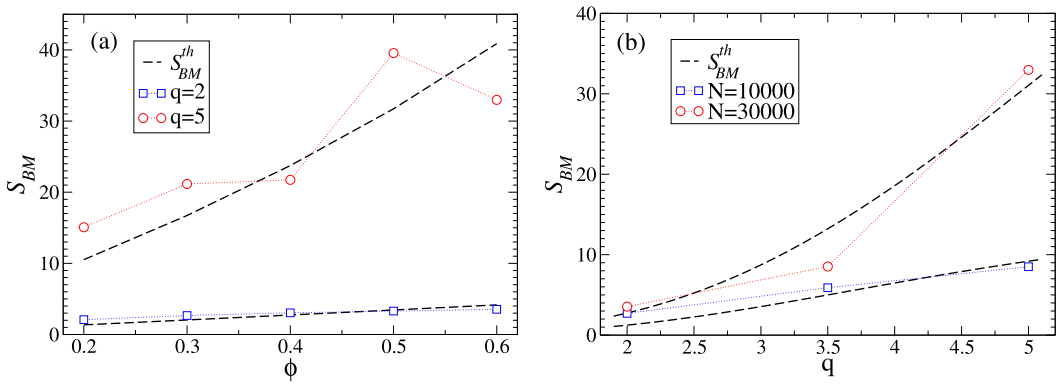

Fig. 6. (a) Speedup S_{BM} versus total volume fraction ϕ for binary mixtures for $q = 2, 5$ with $N = 30\,000$ for all points. (b) Speedup S_{BM} versus size ratio q for $N = 10\,000$, $30\,000$ with $\phi = 0.60$ for all points. Dashed lines are fits to function S_{BM}^{th} defined in Eq. (6).

for investigating the phase diagram of a binary mixture of hard spheres whose size ratio $q = 5$ upon changing the partial volume fractions of the two species. For this system it is possible to calculate a theoretical phase diagram with respect to glass transition within the framework of Mode Coupling Theory and recently jamming lines for such system have been evaluated both theoretically and numerically [22]. In view of growing interest for this system it is crucial to have an efficient algorithm to explore the whole phase diagram. It is worth noting that also conventional time-driven molecular dynamics may benefit from MLL. Huge increase in performance provided by MLL, when the length scales between the components is so important, could play a relevant role in multiscale simulations, a topic that is attracting a lot of interest for biological and material applications. Finally the two optimization methods, SNL and MLL, illustrated in this paper can be used together, like LL and NNL, by using MLL to generate SNL.

Acknowledgements

The author acknowledges support from ERC (226207-PATCHYCOLLOIDS) and warmly thanks Prof. G. Foffi for careful reading of the manuscript.

References

[1] C. De Michele, F. Sciortino, R. Schilling, Dynamics of uniaxial hard ellipsoids, Phys. Rev. Lett. 98 (2007) 265702.
[2] D. Frenkel, B.M. Mulder, J.P. McTague, Phase diagram of a system of hard ellipsoids, Phys. Rev. Lett. 52 (1984) 287–290.
[3] D.C. Rapaport, Role of reversibility in viral capsid growth: A paradigm for self-assembly, Phys. Rev. Lett. 101 (2008) 186101.
[4] A. Stradner, G. Foffi, N. Dorsaz, G. Thurston, P. Schurtenberger, New insight into cataract formation: Enhanced stability through mutual attraction, Phys. Rev. Lett. 99 (2007) 198103.
[5] J.M. Borreguero, B. Urbanc, N.D. Lazo, S.V. Buldyrev, D.B. Teplow, H.E. Stanley, Folding events in the 21-30 region of amyloid β-protein ($A\beta$) studied in silico, Proc. Natl. Acad. Sci. 102 (2005) 6015–6020.
[6] C. De Michele, S. Gabrielli, P. Tartaglia, F. Sciortino, Dynamics in the presence of attractive patchy interactions, J. Phys. Chem. B 110 (2006) 8064.
[7] C. De Michele, P. Tartaglia, F. Sciortino, Slow dynamics in a primitive tetrahedral network model, J. Chem. Phys. 125 (2006) 204710.
[8] S. Corezzi, C. De Michele, E. Zaccarelli, D. Fioretto, F. Sciortino, A molecular dynamics study of chemical gelation in a patchy particle model, Soft Matter 4 (2008) 1173–1177.
[9] A. Donev, S. Torquato, F.H. Stillinger, Neighbor list collision-driven molecular dynamics simulation for nonspherical hard particles, II. Applications to ellipses and ellipsoids, J. Comput. Phys. 202 (2005) 765–793.
[10] Y. Jiao, F.H. Stillinger, S. Torquato, Superdisks and the role of symmetry, Phys. Rev. Lett. 100 (2008) 245505.
[11] J. Vieillard-Baron, Phase transitions of the classical hard-ellipse system, J. Chem. Phys. 56 (1972) 4729–4744.
[12] J. Perram, M. Wertheim, Statistical mechanics of hard ellipsoids, I. Overlap algorithm and the contact function, J. Comput. Phys. 58 (1985) 409–416.
[13] C. De Michele, Simulating hard rigid bodies, J. Comput. Phys. 229 (2010) 3276–3294.
[14] G. Paul, A complexity $O(1)$ priority queue for event driven molecular dynamics simulations, J. Comput. Phys. 221 (2007) 615–625.
[15] D.C. Rapaport, The Art of Molecular Dynamics Simulation, Cambridge University Press, 2004.
[16] M.P. Allen, D.J. Tildesley, Computer Simulation of Liquids, paperback edition, Clarendon Press, 1989, 385 p.
[17] A. Donev, S. Torquato, F.H. Stillinger, Neighbor list collision-driven molecular dynamics simulation for nonspherical hard particles, I. Algorithmic details, J. Comput. Phys. 202 (2005) 737–764.
[18] U. Welling, G. Germano, Efficiency of linked cell algorithms, arXiv:1006.1239v1, 2010, 13 p.
[19] M. Nakata, G. Zanchetta, B.D. Chapman, C.D. Jones, J.O. Cross, R. Pindak, T. Bellini, N.A. Clark, End-to-end stacking and liquid crystal condensation of 6 to 20 base pair DNA duplexes, Science 318 (2007) 1276.
[20] N. Dorsaz, C. De Michele, F. Piazza, P.D.L. Rios, G. Foffi, Diffusion-limited reactions in crowded environments, Phys. Rev. Lett. 105 (2010) 120601.
[21] A. Scala, C. De Michele, Th. Voigtmann, Event-driven Brownian dynamics for hard spheres, J. Chem. Phys. 126 (2007) 134109.
[22] I. Biazzo, F. Caltagirone, G. Parisi, F. Zamponi, Theory of amorphous packings of binary mixtures of hard spheres, Phys. Rev. Lett. 102 (2009) 195701.