Efficiency of linked cell algorithms

Ulrich Welling, Guido Germano

Department of Chemistry and WZMW, Philipps-University Marburg, 35032 Marburg, Germany

Abstract

The linked cell list algorithm is an essential part of molecular simulation software, both molecular dynamics and Monte Carlo. Though it scales linearly with the number of particles, there has been a constant interest in increasing its efficiency, because a large part of CPU time is spent to identify the interacting particles. Several recent publications proposed improvements to the algorithm and investigated their efficiency by applying them to particular setups. In this publication we develop a general method to evaluate the efficiency of these algorithms, which is mostly independent of the parameters of the simulation, and test it for a number of linked cell list algorithms. We also propose a combination of linked cell reordering and interaction sorting that shows a good efficiency for a broad range of simulation setups.

Keywords: molecular simulation, molecular dynamics, Monte Carlo, neighbour list, linked cell list, linked cell reordering, interaction sorting
31.15.xv

1. Introduction

A fundamental issue in the evaluation of non-bonded interactions in molecular simulation is to avoid the calculation of all pair distances because particles beyond a certain cutoff radius r_c can be neglected [1]. The two main methods to deal with this are the Verlet list [1, 2, 3] and the linked cell list (LCL) [1, 4, 5]. A further improvement uses a LCL to generate a Verlet list [6, 7, 8, 9]. These methods apply to both molecular dynamics (MD)

Email addresses: ulrich.welling@staff.uni-marburg.de (Ulrich Welling), guido.germano@staff.uni-marburg.de (Guido Germano)
URL: http://www.uni-marburg.de/fb15/ag-germano (Guido Germano)

and Monte Carlo simulations, but for convenience here we use only MD to illustrate them.

The reason to apply more complex algorithms than a plain LCL is that the majority of the computed pairwise distances is not within the cutoff radius. So, despite of the linear scaling with the number of particles, there is space for further optimizations. Verlet lists solve this accurately by taking into account only particles within a Verlet radius $r_{\mathrm{V}} > r_{\mathrm{c}}$. However Verlet lists have limitations for large scale simulations, as the total memory needed by them scales as the number of particles N times the average number of neighbours. Moreover they must be updated regularily and thus the same basic problem of determining neighbours within a cutoff needs to be solved.

Mason [10] introduced a variant of the LCL where, instead of dividing particles into cells with cubic shape, an arbitrary lattice is set up and particles are sorted into a container belonging to the closest lattice point. This can be seen as a generalization of the LCL by allowing all possible coordination polyhedra.

Gonnet [11] modified the LCL to avoid the majority of unnecessary distance calculations and thus gaining a significant speedup with respect to the standard algorithm by sorting particles according to their projection onto the vector that connects the cell centers; a schematic picture is given in Fig. 1. However in his publication he only investigated the use of this method for one specific case, i.e. the evaluation of the real space part of the Coulomb interaction of water molecules. We wished to investigate the performance of this algorithm for a wide area of simulation setups, taking into account the results of other publications concerning the optimization of linked cell algorithms.

To this aim a general method for comparing the performance of linked cell algorithms is needed. Sutmann [12] tested generalized LCLs for Lennard-Jones systems with different densities and cutoff radii. One of the results is that the ratio between the cell side L and the cutoff has an optimal value typically for a fraction of integer numbers, for example $1:1$ or $1:2$, which can be deduced by geometrical considerations, as the sphere that is built by the cutoff radius fits perfectly a cubic cell, or an array thereof, only in these cases. However the cutoff radius and the density of the system are connected by another parameter, which is the average number of neighbours each particle interacts with, i.e. the particles that reside within the cutoff radius around each particle. This single parameter is the most fundamental one for judging the performance of the different linked cell algorithms. As all linked cell

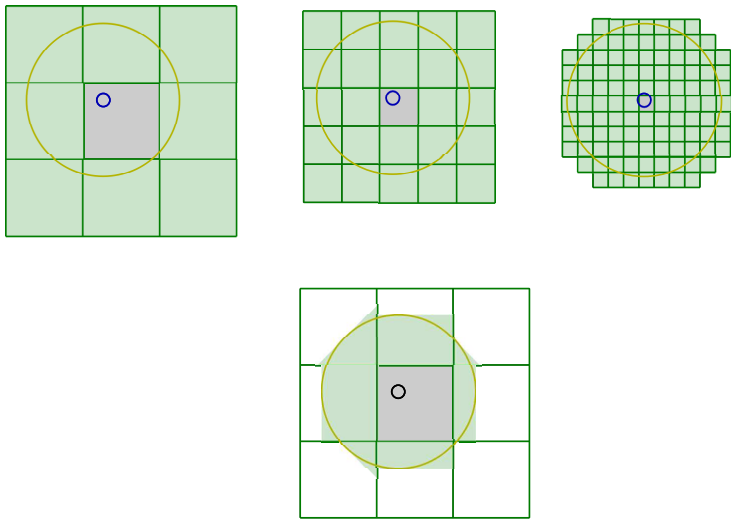

Figure 1: Reduction of the search space (grey/green) using smaller cells (top) or Gonnet's method (bottom).

algorithms scale more or less linearly with increasing number of particles, we can assume that an algorithm that works fastest for a given average number of interactions will do so regardless of the size of the simulation.

Apart from purely algorithmic considerations, there are publications addressing an effect that can be summarized as cache-optimized memory access [8, 13, 14]. The idea is to rearrange the particle information in order to ensure that the data representing particles which are close to each other in the simulated system is located as close as possible in memory, in order to achieve cache hits within the hierarchical memory typical of contemporary computers. Computer memory can be seen as a one-dimensional array, but typical molecular simulations are done in three spatial dimensions, so a perfect mapping is impossible. One early approach was to sort the particles into slabs along one of the three spatial axes, preferably the one corresponding to the largest box side [5, 8]. Meloni et al. [14] picked up a principle that had been used in multi-billion particle simulations [13] and demonstrated the gain in efficiency by reordering particles in memory by their linked cell index, which is referred to as linked cell reordering (LCR). A further improvement of data locality can be achieved doing this by means of the space-filling Peano or Hilbert curve [15]. However rearranging particles in memory is a hardware

specific optimization that will not necessarily be an advantage on all kinds of computers. Specifically, approaches that take advantage of coprocessors like graphic processing units [15, 16] or cell processors might need different adjustments.

To investigate the efficiency of the different linked cell algorithms we set up a small scalar MD program in C++ for the Lennard-Jones fluid and implemented the different linked cell algorithms, taking into account considerations of several publications concerning aspects for an efficient implementation.

2. Minimum image convention

As pointed out by Pütz and Kolb [17] as well as by Heinz and Huenenberger [18], the minimum image problem needs to be adressed on the linked cell level rather than for every particle pair. In the approach given in the book of Allen and Tildesley [1] the minimum image convention is computed for every calculated distance. This is done either with modulo operations or conditional tests, both of which are expensive in computation time. A better way is to compute a distance vector for an evaluated cell pair and to add it to the distance vector of molecules within these cells. This way the periodic images are resolved by three additions for any calculated distance. In this work we used the method introduced by Rapaport [19] and used by Pütz and Kolb [17], which uses copies of the linked cells at the borders of the unit box with translated coordinates. This way the evaluation of the pairwise forces does not need any additional operation. This is similar to domain decomposition, where the particles in the cells at the boundary of a domain are "exported", i.e. copied, to the neighbour domains for the calculation of the forces [19, 20, 21, 22, 23, 24]. Only interactions with the positive half shell of surrounding cells are computed, so Newton's third law is exploited to avoid the calculation of an interaction twice. The forces on the particles in the copied region are reassigned to the original particles. This reduces the effort of computing the minimum image convention from being proportional to the number of evaluated pair distances to the number of particles within the border region of the unit box. Gonnet [11] used the variant with an additional vector to resolve periodicity, thus making distance evaluations a little more expensive than necessary. It is important to avoid these superfluous computations, as it is possible that a performance gain of a different linked cell algorithm could be caused just by reducing the number of minimum image calculations, as can be seen in Mason's publication [10]. He explicitly

writes that in his approach minimum images are computed per particle pair and that this computation is time consuming.

3. Linked cell variants

In our test program we implemented several variants of linked cell algorithms. As basis we use a generalized LCR with an arbitrary relation between the cutoff radius and the cell side. During the initialization the program computes an interaction matrix of the linked cells given in relative coordinates as proposed by Mattson [6].

The LCR algorithm as given by Kadau [13] and Meloni et al. [14] consists in computing at every time step the scalar cell index i of each particle from its coordinates and then to sort the particles by i. The latter is computed e.g. as $i = i_z + i_y n_z + i_x n_y n_z$, where $i_\alpha = 0, \ldots, n_\alpha - 1$ are the Cartesian cell grid indices. An improvement computes i through the space-filling Peano or Hilbert curve [15], which results in better data locality. The sorting can be achieved by a hash sort which takes $O(N)$ operations, as the numbers of particles with the same cell index can be counted, and in a second loop all particles can be moved directly to a position according to their cell index. An advantage of the reordering method is that within one time step only a few particles change their cell, so the effort of rearranging the particles is small. Interestingly a quicksort [25] that exploits pre-existing order can be applied for sorting; in simulations with slow moving particles this is slightly faster than the direct way. We also generalized the LCR method for arbitrary fractions $m/n,\ m, n \in \mathbb{N}$, of the cutoff and the cell side. For this a list of relative cell coordinates that are within the cutoff radius around a central cell is generated and used to define the interacting cell pairs during the force evaluation. This list is reduced by a factor two exploiting Newton's third law as mentioned in the previous section.

On top of this we implemented Gonnet's interaction sorting method [11]. As it evaluates cell pairs, it can be used with all the linked cell algorithms, both with list and sorted variants and with an arbitrary cell-side to cutoff ratio. The full details are discussed in Gonnet's publication; the principle is that for each pair of cells all the particles are projected onto the vector that connects the cell centers. The projection can be simplified in the special cases where one or two components of the connecting vector are zero, which happens when the cells have one or two identical Cartesian grid indices.

However, the necessary additional conditional evaluations are slower than the saved multiplications by zero, and the code is more complex.

Afterwards the particles of both cells are sorted by the projected distance. Since this value is smaller than the distance itself, if the former is larger than the cutoff the latter will be larger too and this pair of particles does not need to be evaluated. So instead of computing all pairs between the two cells, which would be an operation of order $O(N_1 N_2)$, all the particles of both cells are projected and sorted, with an effort of $O(N_1 + N_2 + N_1 \log N_1 + N_2 \log N_2)$. The task to be done here is sorting a small set of particles as fast as possible. Gonnet used a typical implementation of the quicksort method that stops at a certain array size, usually smaller than 10, and handles the remaining subarray with an $O(N^2)$ method like insertion sort. However, for small sets the best solution are optimal sorting networks [25]. They result in the minimum number of comparisons and swaps, close to the theoretical limit of $N \log_2 N!$ operations. Thus in our implementation we resorted to a combination of the quicksort algorithm and, for each subarray size of 16 and smaller, a suited optimal sorting network. Gonnet's result happened to be faster than the generalized LC algorithm with $L = r_c/2$ though his sorting method was not optimal because in both cases he computed the minimum image convention in a fashion that was not optimal either. Speeding up the latter, it becomes essential to improve the sorting too in order to remain competitive with respect to the generalized LC algorithm without interaction sorting.

4. Computational details

To evaluate the efficiency we did a set of simulations with changing average number of interactions per particle. A typical setup of the Lennard-Jones fluid with a density $\rho = 0.72$ and a cutoff radius $r_c = 2.5$ has an average number of interactions of $(4/3)\pi r_c^3 \rho = 55$. Gonnet tested his modification of the linked cell algorithm for the real space part of the Coulomb interaction in water with $r_c = 10\,\mathrm{nm}$, resulting in an average number of interactions of 150. We are not aware of MD simulations with a considerably larger number of interactions. For each average number of interactions per particle we obtain the average time per integration step and particle.

We set up a small scalar MD program for the Lennard-Jones fluid using the velocity Verlet integrator [26]. Periodic boundary conditions are applied with the minimum image convention computed as described before. For the

tests we generated particles on a lattice with a density derived from the required number of interacting particles. For simplicity our unit box and therefore also its subcells were cubic. The velocities were randomized and the system was equilibrated for 20 000 steps of $\Delta t = 0.001$ at a temperature $T = 5$ using velocity scaling, followed by 4000 steps of time profiling in the microcanonical ensemble. We used the GNU C++ compiler version 4.3.2 to produce 64 bit binaries with double precision on an Intel Core2 Quad running at 1.6 GHz.

5. Results and discussion

First of all we compared the LCL, the LCR with a minimum image check for every calculated distance, and the LCR with resolved periodicity as described above (Fig. 2). The impact on performance of avoiding the evaluation of the minimum image convention is higher than changing from the list to reordering.

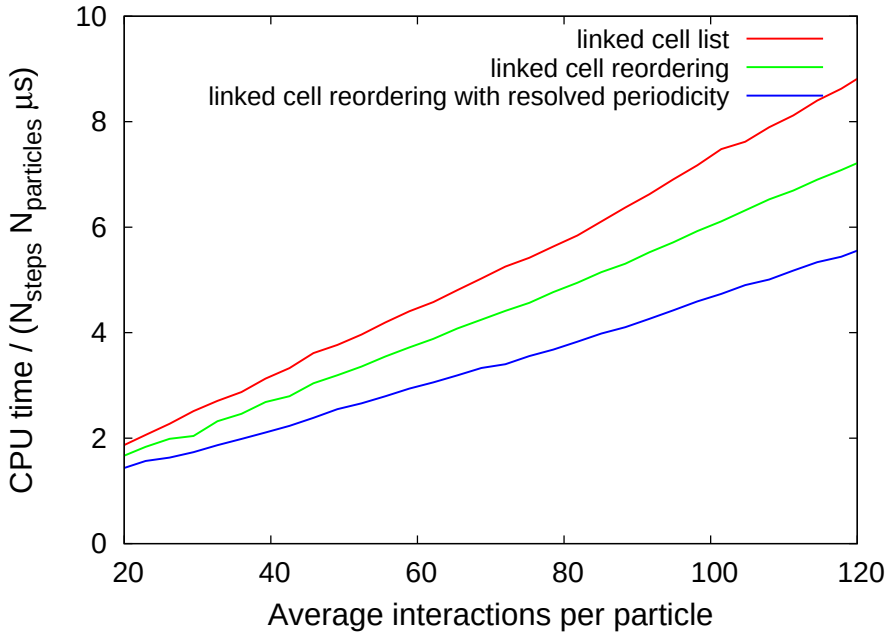

Figure 2: Performance of linked cell variants.

Knowing this we compared Gonnet's extention with the generalized LCR

variants to see which algorithm performs fastest for any given average number of interactions (Fig. 3). Gonnet's algorithm does perform faster than all the generalized LCR variants. The break-even between the LCR with a cell side equal to the cutoff radius and Gonnet's modified method is approximately at an average number of interactions equal to 82. This corresponds to an average number of 19 particles per linked cell.

Usual MD simulations do not set the side of the unit box as an integer multiple of the cutoff, so that the real size of the linked cells is generally slightly larger than the cutoff radius. We plotted the position of the crossover as a function of the ratio between the side of the linked cells and the cutoff (Fig. 4). Gonnet's approach tolerates larger linked cell sides much better than the standard linked cell algorithm. If we put the average number of particles per linked cell on the y-axis rather than the average interaction count per particle, we see that the crossover stays between 19 and 20 particles per linked cell (Fig. 5). This is almost a constant that is pretty much unaffected by all the simulation details. A MD software could automatically choose between the standard linked cell implementation and Gonnet's method just using the number of particles per linked cell as a criterium. This value depends on the compiler and a lot of hardware specific parameters like cache sizes and the CPU architecture; thus it must be determined separately for each system.

Even for a Lennard-Jones fluid simulation with $r_c = 2.5$ and $\rho = 1.0$ yielding an average interaction count of 65, Gonnet's method will be faster if the ratio L/r_c is larger than 1.08, which can easily happen for small-sized simulations.

If the minimum image convention is resolved in a more expensive way than we did in our test, Gonnet's method is faster for even a smaller number of interactions or particles per linked cell.

6. Summary

We investigated Gonnet's variant of the linked cell algorithm. We modified it in two ways, using linked cell reordering rather than a plain linked list and changing the sort method to a combination of quicksort and optimal sorting networks that are particularly suited for small arrays. Moreover we used the fastest known approach for the computation of the minimum image convention, which resembles the scheme of domain decomposition parallelism. Then we evaluated the performance of the modified Gonnet approach in comparison with the known linked cell algorithm variants, and introduced

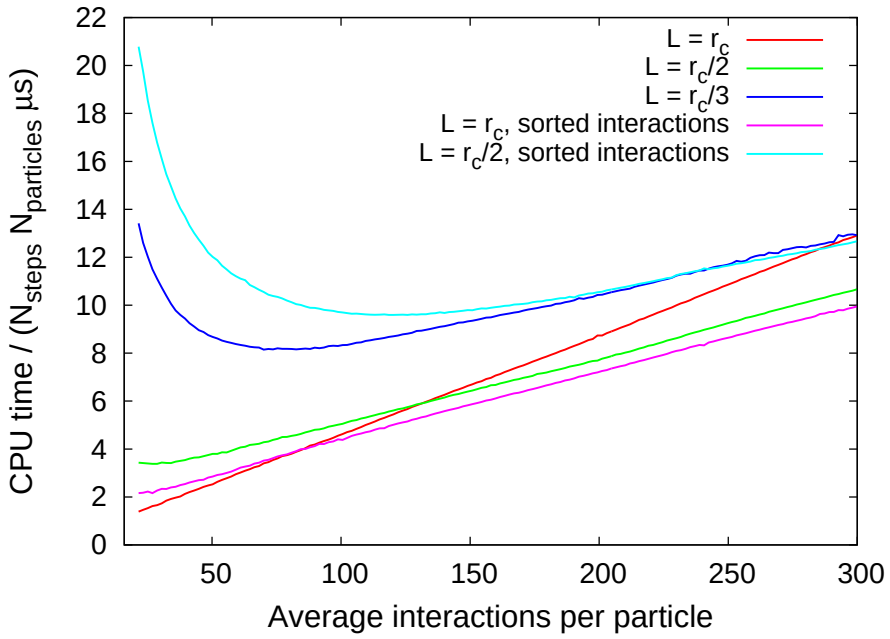

Figure 3: Performance of the generalized LCR with and without sorting and resolved periodicity, at different ratios between the cell side L and the cutoff radius r_c.

a general way to compare the performance of these algorithms independently of the simulation parameters. On our test systems the reordering version of the linked cell algorithm performed better than the list variant. This basically reproduces the work by Meloni et al. [14], however we investigated the behaviour for a broad range of possible simulation setups. Gonnet's modified variant always outperformes the generalized linked cell variants in our tests and is only slower than the linked cell algorithm for less than roughly 19 particles per linked cell. Our results show that even for Lennard-Jones fluids cases exist where the modified Gonnet variant of the linked cell algorithm is faster than the standard linked cell algorithm. These results enable molecular dynamics software to choose the faster method based on simple statistics that can be computed easily during a simulation.

Acknowledgments

We thank Pedro Gonnet for providing a reference implementation of his algorithm.

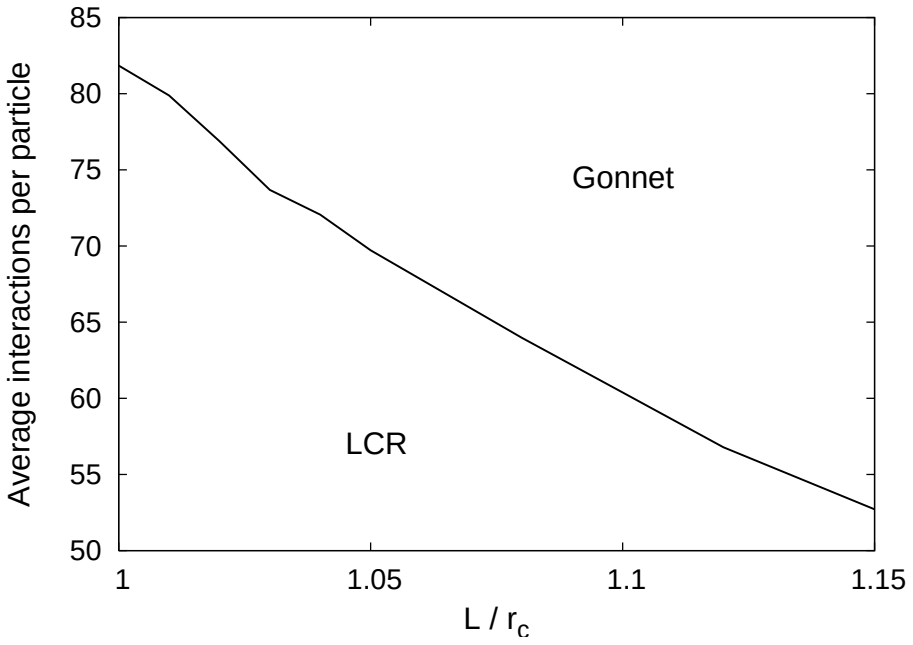

Figure 4: Performance crossover for simulations with cell sides L slightly larger than the cutoff radius r_c. Below LCR is faster, above the Gonnet method is faster.

[1] M. P. Allen, D. J. Tildesley, Computer simulation of liquids, Paperback Edition, Oxford University Press, Oxford, 1989.

[2] L. Verlet, Computer experiments on classical fluids, Phys. Rev. 159 (1967) 98–103.

[3] A. A. Chialvo, P. G. Debenedetti, On the use of the Verlet neighbor list in molecular dynamics, Comput. Phys. Commun. 60 (1983) 215–224.

[4] B. Quentrec, C. Brot, New method for searching for neighbors in molecular dynamics computations, J. Comput. Phys. 13 (1973) 430–432.

[5] R. W. Hockney, J. W. Eastwood, Computer simulations using particles, McGraw-Hill, New York, 1981.

[6] W. Mattson, B. M. Rice, Near-neighbor calculations using a modified cell-linked list method, Comput. Phys. Commun. 119 (1999) 135–148.

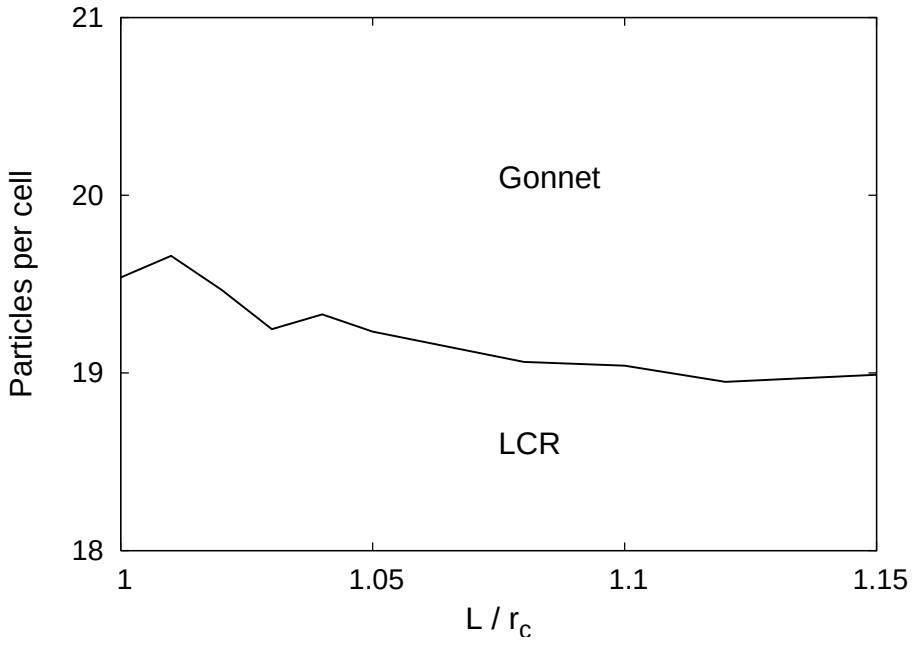

Figure 5: Performance crossover for simulations with cell sides L slightly larger than the cutoff radius r_c. Below LCR is faster, above the Gonnet method is faster.

[7] R. J. Petrella, I. Andricioael, B. R. Brooks, M. Karplus, An improved method for nonbonded list generation: Rapid determination of near-neighbor pairs, J. Comput. Chem. 24 (2003) 222–231.

[8] Z. Yao, J.-S. Wang, G.-R. Liu, M. Cheng, Improved neighbor list algorithm in molecular simulations using cell decomposition and data sorting method, Comput. Phys. Commun. 161 (2004) 27–36.

[9] D.-B. Wang, F.-B. Hsiao, C.-H. Chuang, Y.-C. Lee, Algorithm optimization in molecular dynamics simulation, Comput. Phys. Commun. 177 (2007) 551–559.

[10] D. R. Mason, Faster neighbour list generation using a novel lattice vector representation, Comput. Phys. Commun. 170 (2005) 31–41.

[11] P. Gonnet, A simple algorithm to accelerate the computation of non-bonded interactions in cell-based molecular dynamics simulations, J. Comput. Chem. 28 (2007) 570–573.

[12] G. Sutmann, V. Stegailov, Optimization of neighbor list techniques in liquid matter simulations, J. Mol. Liq. 125 (2006) 197–203.

[13] K. Kadau, T. C. Germann, P. S. Lomdahl, Molecular dynamics comes of age: 320 billion atom simulation on BlueGene/L, Int. J. Mod. Phys. C 17 (2006) 1755–1761.

[14] S. Meloni, M. Rosati, L. Colombo, Efficient particle labeling in atomistic simulations, J. Chem. Phys. 126 (2007) 121102.

[15] J. A. Anderson, C. D. Lorenz, A. Travesset, General purpose molecular dynamics simulations fully implemented on graphics processing units, J. Comput. Phys. 227 (2008) 5342–5359.

[16] W. G. Liu, B. Schmidt, G. Voss, W. Mueller-Wittig, Accelerating molecular dynamics simulations using graphics processing units with CUDA, Comput. Phys. Commun. 179 (2008) 634–641.

[17] M. Pütz, A. Kolb, Optimization techniques for parallel molecular dynamics using domain decomposition, Comput. Phys. Commun. 113 (1998) 145–167.

[18] T. N. Heinz, P. H. Huenenberger, A fast pairlist-construction algorithm for molecular simulations under periodic boundary conditions, J. Comput. Chem. 25 (2004) 1474–1486.

[19] D. C. Rapaport, Large-scale molecular dynamics simulation using vector and parallel computers, Comput. Phys. Rep. 9 (1988) 1–53.

[20] S. Plimpton, Fast parallel algorithms for short-range molecular dynamics, J. Comput. Phys. 117 (1995) 1–19.

[21] M. R. Wilson, M. P. Allen, M. A. Warren, A. Sauron, W. Smith, Replicated data and domain decomposition molecular dynamics techniques for simulation of anisotropic potentials, J. Comput. Chem. 18 (1997) 478–488.

[22] G. S. Heffelfinger, Parallel atomistic simulation, Comput. Phys. Commun. 128 (2000) 219–237.

[23] D. E. Shaw, A fast, scalable method for the parallel evaluation of distance-limited pairwise particle interactions, J. Comput. Chem. 26 (2005) 1318–1328.

[24] J.-S. Wu, Y.-L. Hsu, Y.-M. Lee, Parallel implementation of molecular dynamics simulation for short-ranged interaction, Comput. Phys. Commun. 170 (2005) 175–185.

[25] D. E. Knuth, The Art of Computer Programming, Vol. 3, Addison-Wesley, Reading, MA, 1973.

[26] W. C. Swope, H. C. Andersen, P. H. Berens, Computer-simulation method for the calculation of equilibrium-constants for the formation of physical clusters of molecules — application to small water clusters, J. Chem. Phys. 76 (1982) 637–649.