Improved $O(N)$ Neighbor List Method Using Domain Decomposition and Data Sorting

Zhenhua Yao*, Jian-Sheng Wang** and Min Cheng***

* National University of Singapore, Singapore 117576, smayzh@nus.edu.sg
** National University Singapore, Singapore 117543, wangjs@cz3.nus.edu.sg
*** Nanyang Technological University, Singapore 639798, emcheng@ntu.edu.sg

ABSTRACT

The conventional Verlet table neighbor list algorithm is improved to reduce the number of unnecessary interatomic distance calculation in molecular simulations involving many atoms. Both of the serial and parallelized performance of molecular dynamics simulation are evaluated using the new algorithm and compared with those using the conventional Verlet table and cell-linked list algorithm. Results show that the new algorithm significantly improved the performance of molecular dynamics simulation compared with conventional neighbor list maintaining and utilizing algorithms in serial programs as well as parallelized versions.

Keywords: Neighbor list, Molecular dynamics, Domain decomposition, Data sorting, Verlet table

1 Introduction

Some molecular simulation techniques such as molecular dynamics and Monte Carlo method are widely used to study the physical properties and chemical processes which contain a large number of particles at the atomic level in statistical physics, computational chemistry, and molecular biology field [1]. All these methods involve evaluation of the sum of total interatomic potential energy $V_{\rm tot}$ of N atoms and/or its gradients. In molecular dynamics simulation, this procedure needs $O(N^2)$ number of steps and usually costs most of CPU time. Obviously it is formidable to carry out such a calculation when there are many atoms in the system, and some methods are strongly needed to reduce the redundant computation.

A General way to reduce the calculation of potential/force is using a cutoff distance $r_{\rm cut}$ in potential functions, and assumes that both potential functions and gradients beyond the cutoff distance are zero. This treatment reduces the computing time greatly by neglecting all atoms beyond the cutoff distance, since interactions between these atoms are zero and needn't to be considered.

Effective reduction of redundant calculation of interatomic potential can be accomplished by conventional Verlet table algorithm and cell-linked list algorithm. However, basically there is a tradeoff between overhead for maintaining neighbor list table and reduction of calculation of unnecessary interatomic distance.

In this paper conventional Verlet table method is improved and the overhead to maintain the neighbor list table has been reduced to the order $O(N)$, and the efficiency of calculating interatomic distance is still as high as those of Verlet table and cell-linked list methods in almost all instances. Furthermore it is easy to parallelize on SMP platforms as well as on workstation clusters.

Conventional Verlet table algorithm and cell-linked list algorithm have been widely parallelized and have shown significant reduction in total computing time [2, 3]. In this work it is intend to optimize serial performance on single processor as well as parallel environment. The modification of the algorithm and demonstration of the improved performance on single processor computer and dual–processors are described in this paper.

2 Review of conventional algorithms and construction of improved neighbor list algorithm

The conventional Verlet table method and cell-linked list method have been introduced in a classical book about molecular simulation by Allen and Tildesley [4]. Some graphics in this section are drawn in two dimensions for convenience of illustration, however, all discussions are easy to be generalized to three dimensional systems.

2.1 Conventional Verlet table algorithm

The basic idea of Verlet table method is to construct and maintain a list of neighboring atoms for every atom in the system [4,5]. During the simulation, this neighbor list will be updated periodically for a fixed interval or reconstruct itself automatically when some atoms move too much and the list is going to be out-of-date [6].

In conventional Verlet table algorithm the potential cutoff sphere of radius $r_{\rm cut}$ is surrounded by a "skin", to give a larger sphere of radius $r_{\rm s}$ [4]. In the first step of simulation, a neighbor list is constructed for every atom in the system, and an atom is considered as a "neighbor"

if the distance between two atoms is equal to or shorter than r_s. Over the next few time steps this neighbor list is used in the force and potential evaluation routine. Each atom is assumed to interact only with those in its neighbor list, thus a huge amount of unnecessary interatomic distance calculation is eliminated and the overall performance is increased. In the following from time to time, the neighbor list is reconstructed and the similar procedure is repeated.

In conventional Verlet table algorithm, it is needed to evaluate the interatomic distances between all atom pairs, so the total steps to construct a neighbor list table are the order $O(N^2)$. But once the neighbor list is constructed and between the interval of updating, evaluation of the forces/potentials of the system is efficient because there are only atoms in the neighbor list, i.e., in the sphere of r_s as the radius, need to be evaluated the interatomic distances, and this procedure requires the order of $O(N \cdot N_{\text{neighbor}})$ steps, in which N_{neighbor} is the average number of neighbors in the material and won't change with the system size.

The Verlet table method has been proven to be efficient when a system contains a relatively small number of atoms and the atoms move slowly. Its disadvantage is the inefficiency of constructing neighbor list, as the procedure requires the order of $O(N^2)$ steps (more precisely, $N(N-1)/2$ steps). Moreover, as the atoms move quickly, either the "skin" must increase or the frequency of reconstructing neighbor list table must increase. Both of two requirements make the overall calculation increases dramatically.

2.2 Conventional Cell-linked list algorithm

The Conventional cell-linked list algorithm is another effective method to reduce the calculation of potential and force evaluation when the number of atoms is large [7]. In this method, the simulation space is partitioned into several cells, and each edge of cells is equal to or larger than cutoff distance of the potential function. All atoms are assigned to the cells according to their positions, and during this procedure a linked list of the atom indices is created. At the beginning of a simulation, an array that contains a list of cell neighbors for each cell is created, and this list remains fixed unless the simulation domain changes during the simulation [4].

The overhead to update the neighbor list, i.e., assigning each atom to corresponding cells, is very small, but a big number of interatomic distances still need to be evaluated in the potential/force calculation, and this makes cell-linked list method rather inefficient compared to Verlet table method when the number of atoms is small. A common choice of the cell edge is the potential cutoff distance r_{cut}, thus for each atom, all atoms in 27 cells, or in the volume of $27 \times r_{\text{cut}}^3$, will be evaluated the interatomic distances. Ideally, only atoms in the volume of $\frac{4}{3}\pi r_{\text{cut}}^3 \approx 4.189\ r_{\text{cut}}^3$ fall in the cutoff distance and need to be evaluated the interatomic distances.

Comparing with conventional Verlet table method, we can know that the advantage of cell-linked list method is the fast and efficient building of "neighbor list", and the disadvantage is that there are too many unnecessary atoms need to be evaluated the interatomic distances in the "neighbor list" and the improved methods seem to increase the complexity of constructing and using algorithm but the overall performance has little improvement only.

2.3 Improved neighbor list algorithm

In the section 2.1 one can know that the reason why the steps of maintaining Verlet table in conventional method is the order of $O(N^2)$ is enumerating every atom in the simulation domain for finding out the neighbors of an atom. And also in section 2.2, we know that cell-linked list method doesn't have this trouble so that its neighbor list constructing speed is higher, but the efficiency of utilizing neighbor list in force/potential evaluation is sacrificed. If the advantages of both methods can be combined together, the algorithm can be optimized.

2.3.1 Domain decomposition approach to search neighbors

In this work conventional Verlet table method and cell-linked list method are combined together, to prevent the constructing of the neighbor list table from over-counting too many atoms. Like the cell-linked list method, the whole simulation domain is partitioned into several cells, and the edges of these cells can be larger or smaller than the potential function cutoff distance r_{cut}, every time before constructing neighbor list table, each atom is assigned to these cells by their coordinates, and then Verlet table search algorithm is used to construct the neighbor list table, but only atoms in neighbor cells are needed to evaluate the interatomic distances, instead of all atoms in the system.

Because the searching of neighbors is limited to a fixed number of cells instead of whole simulation domain, the order of overall neighbor list construction becomes $O(c \cdot N)$ from $O(N^2)$. For a system contains more than 1000 atoms, this improvement is very significant and the overall performance is greatly boosted.

For gas and liquid simulation by using Lennard–Jones potential, we find that the best practice of cell edge is $\frac{1}{2}r_{\text{cut}}$ after testing on several computers with different architectures.

2.3.2 Acceleration of data access by data sorting

By considering the pipeline architecture of modern CPUs nowadays, further effort has been made to boost the computation performance: sorting the storage sequences of atoms in the memory and making atoms which in the same cell or neighbor cells also in adjacent memory locations, thus the data can be loaded and cached more efficiently.

For better understanding the reason why carrying out data sorting, we can consider the gas and liquid materials which have high atomistic mobility. In the beginning while the structure data is just generated, the data of position, velocity and acceleration are well sorted, and the memory location of an atom is near those of its neighbors, sometimes all neighbors' data can be loaded into limited number of cache lines if the original structure data is well organized. But as long as the simulation is going on and the atoms are moved here and there, this situation will change, the memory locations of every atom are far from each other, and seldom in the same cache line. Then CPU is hard to find the next neighbor's data in the cache, and has to stall the calculation and fetch it into the cache, however, the useless data long with the newly incoming data pollute the data cache, thus the data cache can never be well utilized. This situation can be detected in a long time simulation and a very significant performance degradation can be investigated.

The solution for this problem is rather straightforward, i .e. , sorting the data of atoms by their positions and making the memory locations of same atom's neighbors as near as possible. After this treatment, no distinct performance degradation can be investigated in the long time simulation.

Together with domain composition algorithm in section 2.3.1, the data locality is enhanced when the program is running. Thus when writing programs for SMP platforms, the data can be easily and well partitioned, and the searching of neighbors can be carried out by each CPU in the computers independently, thus the well parallelized execution can be achieved.

In this work, the overall procedure for constructing neighbor list table is shown in Algorithm 1.

Algorithm 1 Improved neighbor list algorithm

```
{Assigning all atoms into their appropriate cells}
for all atoms in the system do
  calculate the index i of its appropriate cell;
  append the index of atom into the list of cell i;
end for
{Sorting atoms by their coordinates}
{Now carry out conventional Verlet table procedure}
for all atoms i in the system do
  l ⇐ the cell number of atom i
  for all cells m among neighbors of cell l and cell l
  do
    for all atoms j in cell m do
      calculate interatomic distance r_ij;
      apply the periodic boundary condition;
      if r_ij < r_cut then
        append j into the neighbor list of atom i;
      end if
    end for
  end for
end for
```

3 Results

A molecular dynamics simulation program using Lennard-Jones $(12-6)$ two-body potential is developed to compare the performance of three different neighbor list algorithms. For measuring the performance quantitatively a new unit named *atom·step/second* is defined. It can be simply calculated by multiply number of atoms and number of steps simulated divided by number of seconds elapsed. The larger is this value, the better is the overall performance. For computers with same architecture, and the program is linearly scaled, this number is proportional to the CPU performance.

In the simulation, some uniformly distributed Argon atoms with random locations are placed in the domain firstly, and the density of gas is predetermined. Then simulation in canonical ensemble is performed, and the number of steps is 10^2 for 10^4 atoms and above, or 10^3 for $10^3 \sim 10^4$ atoms, or 10^4 for 999 atoms or less. The Nośe–Hoover thermostat is used to implement the canonical ensemble simulation, and the temperature of system is 300 K.

In order to verify the improved Verlet table algorithm, all neighbor lists were dumped to the disk files and compared with those in Verlet table algorithm for different system size. In the verification simulation the statistical quantities, such as total potential energy, total kinetic energy, transient temperature of system and the trajectory of atoms have been recorded for every 10 steps, and three sets of data generated from three algorithms are compared and ensured they differ in round-off errors only. A series of tests showed that neighbor lists from improved Verlet table method are exactly as same as those from conventional methods, and simulations with three different algorithms output exactly the same results.

For different simulation the volume of system is increased with constant density, thus the number of atoms is increased correspondingly, and the performance is calculated.

Comparison of performances of molecular dynamics simulation with different algorithms are shown in Fig. 1 (single processor results) and Fig. 2 (dual-processor re-

sults)

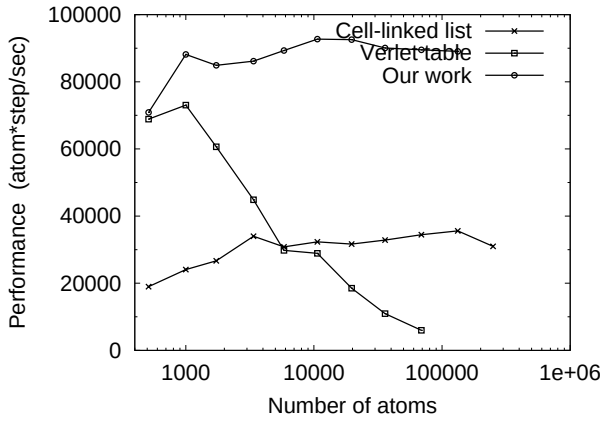

Figure 1: Comparison of three algorithms on single processor system. The performance is measured in the unit of "atom·step/second", and its value can be calculated by multiply number of atoms by number of steps and divided by the whole simulation time. The three curves from top to bottom stand for performances of our improved method, Verlet table and cell-linked list method, respectively.

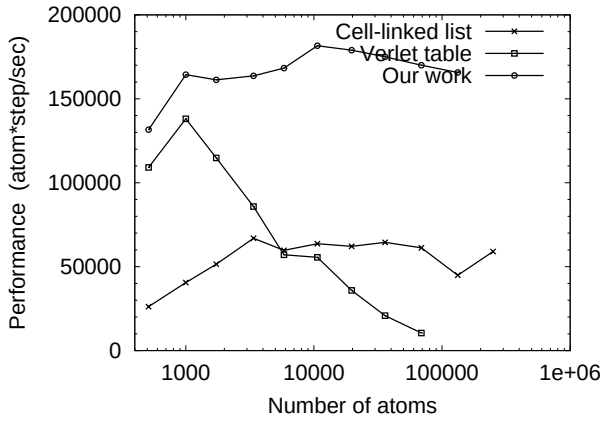

Figure 2: Comparison of three algorithms on dual-processor system. The three curves from top to bottom stand for performances of our improved method, Verlet table and cell-linked list method, respectively.

From the results, we can see that improved Verlet table method in our work makes very significant improvement to the overall simulation performance. When the system is small, the performance of new method is as high as those of conventional Verlet table method. While the system scale increases and there are more and more atoms, the performance of conventional Verlet table method decreases sharply, but the performance of new method becomes even better. As the system is as big as 7×10^5 atoms, the performance of conventional Verlet table method becomes very bad, which is far behind that of new method. On the other hand, though conventional cell–linked list method can handle a large system due to its small memory allocation, its performance is much lower than that of new method.

We can also see that in SMP platforms, the new method still exhibits higher performance than the other two methods. This advantage should be taken by the using of domain composition algorithm.

4 Conclusions

Nowadays the performance of CPU is increased follows Moore's law, and more and more powerful supercomputers emerge continuously, thus larger and complicated molecular simulations will be attempted which involve larger amount of atoms and more complex potential functions. The expectation of running molecular simulation faster and easier for larger systems on existing platforms make it important to improve the conventional neighbor list updating algorithm in order to reduce the unnecessary interatomic distance calculations. A significant improvement of molecular dynamics simulation performance has been shown in this paper by improved order $O(N)$ Verlet table algorithm both on single processor platforms and dual-processor platforms. The results have shown that the new algorithm is superior than conventional Verlet table and cell-linked list algorithm in serial programs as well as parallelized programs.

Acknowledgments

This work was supported by the Singapore–MIT Alliance.

REFERENCES

[1] J. M. Thijssen, Computational Physics, Cambridge University Press, Cambridge, 1999.

[2] W. Mattson, B. M. Rice, Computer Physics Communications, **119**: 135–148, 1999.

[3] J. H. Walther, P. Koumoutasakos, Journal of Heat Transfer, **123**: 741–748, 2001.

[4] M. P. Allen, D. J. Tildesley, Computer simulation of liquids, Oxford University Press, New York, 1990.

[5] L. Verlet, Physics Review, **159**: 98–103 1967.

[6] D. Fincham, B. J. Ralston, Computer Physics Communication, **23**: 127–134, 1981.

[7] B. Quentrec, C. Brot, Journal of Computational Physics, **13**, 430–432 (1975); R. W. Hockney, J. W. Eastwood, Computer simulation using particles, McGrow–Hill, New York, 1981.