Chapter 2 Numerical method

2.1 Basic molecular dynamics

In this section, we introduce briefly for completeness the basis of the molecular dynamics simulations and the definition of the model used to simulate the behavior of a droplet impacting a liquid film laid on the solid substrate.

In a molecular dynamics simulation, all atoms are considered as point masses defined by a position and its time derivatives. Each particle is interacting with the other particles through interaction forces derived from the interaction potentials detailed in this section, and time evolution is governed by Newtonian mechanics. At each time step, particle accelerations are calculated using Newton's second law and their positions are then updated.

$$F_i = m\ddot{r}_i = \sum_{\substack{j=1 \\ (j \neq i)}}^{N} f_{ij}$$

where r_i is the position vector of molecule i.

Molecular dynamics method is used to solve the dynamics of a N-particle (atom or molecule) system by integrating the Newton's equation of motion, as shown in equations (2.1) and (2.2), for each particle to obtain the position of the particle as a function of time [23,24].

$$m_i \frac{d\vec{v}_i}{dt} = \sum_i F_2(\vec{r}_i, \vec{r}_j) + \sum_j \sum_k F_3(\vec{r}_i, \vec{r}_j, \vec{r}_k) + \ldots \qquad (2\text{-}1)$$

$$\frac{d\vec{r}_i}{dt} = \vec{v}_i \tag{2-2}$$

where m_i is the mass of atom i, $\vec{r}_i$ and $\vec{v}_i$ are its position and velocity vectors, respectively. F_2 is a force term, which describes pairwise interactions between atoms, F_3 is a force term, which describes three-body interactions, and many-body interactions can be added if needed. Force on each atom is the spatial derivative of potential energy that is generally written as a function of the position of itself and other atoms. The general procedure of MD simulation is shown in Fig 2.1.

Simulation programs are conventionally written so that all quantities are unitized that may avoid overflows of calculation by the computer. As units of distance and energy we use the potential parameters σ and ε, respectively, and as the unit of mass, that of one atom.

2.2 Equation of motion

The potential energy is a function of the atomic positions (3N) of all the atoms in the system. Due to the complicated nature of this function, there is no analytical solution to the equations of motion; they must be solved numerically. Numerous numerical algorithms have been developed for integrating the equations of motion.

One very simple numerical scheme that is widely used in MD is known as the leapfrog method; it is completely equivalent algebraically. In its simplest form the method yield coordinates that are accurate to third order in Δt. However, it tends to be considerably better than the higher-order methods from the viewpoint of energy conservation. In addition, its storage requirements are also minimal. The leapfrog method introduces in detail as follows:

The leapfrog method is equally simple to derive. Rewrite the Taylor expansion as

$$x(t+h) = x(t) + h[\dot{x}(t) + (h/2)\ddot{x}(t)] + O(h^3) \tag{2.6}$$

The term multiplying h is just $\dot{x}(t+h/2)$, so (.2.6) becomes (2.8) below. The result (2.7) is obtained by subtracting from $\dot{x}(t+h/2)$ the corresponding expression for $\dot{x}(t-h/2)$. The leapfrog integration formulate are then

$$\dot{x}(t+h/2) = \dot{x}(t-h/2) + h\ddot{x}(t) \tag{2.7}$$

$$x(t+h) = x(t) + h\dot{x}(t+h/2) \tag{2.8}$$

The fact that coordinates and velocities are evaluated at different times dose not present a problem; if an estimate for $\dot{x}(t)$ is required there is a simple connection that can be expressed in either of two ways:

$$\dot{x}(t) = \dot{x}(t \mp h/2) \pm (h/2)\ddot{x}(t) \tag{2.9}$$

The initial conditions can be handled in a similar manner, although a minor inaccuracy in describing the starting state, namely, the distinction between $\dot{x}(0)$ and $\dot{x}(h/2)$, is often ignored [23][24].

The Leapfrog method may directly find the new position of the atom next step; it does not have to revises. Therefore, it may simplify the process of calculation, its method as follows:

$$V_i(t+{dt}/{2}) = V_i(t-{dt}/{2}) + dt \times \frac{F_i(t)}{m_i} \tag{2.10}$$

$$r_i(t+dt) = r_i(t) + dt \times V_i(t+{dt}/{2}) \tag{2.11}$$

At first, we utilize the known velocity at $t-{dt}/{2}$ to find a velocity of an atom at $t+{dt}/{2}$, from (2.10), then utilizing the known position and velocity (2.11) to calculate a new position of an atom at $t+dt$. In order to reduce the error of calculation, using the velocity at $t+{dt}/{2}$ is better than at t. It also corresponds to utilizing the mean value of velocity at t and $t+dt$ to find the new position of an atom, of course, it is more accuracy.

2.3 Interaction potentials

Choice of potentials is a very important project in MD simulation. Potential energy can also be categorized as short-ranged and long-ranged interaction in nature. The former (e.g., L-J potential) [24] only considers the interaction between the atoms geometrically nearby the interested atom, while the later (e.g., Columb potential in ionic solid or biological system) needs to consider the interaction even from the atoms far away from the interested atom. Albeit its simplicity of the short-ranged interaction, it have been applied extensively in many MD simulations in the past, especially for liquids and solids. In contrast, long-ranged interaction models are not commonly used in classical MD simulation except for polarized molecules, such as water [23,24].

2.3.1 Lennard-Jones potential

In this study, we are only interested in dealing with classical MD using short-ranged interaction. The Lennard-Jones potential were used to He-He, Ar-Ar and Xe-Xe interaction whether the droplet, liquid film or solid substrate.

At first, the main step is to define the force field containing all potentials of interaction between atoms. The basic interaction applied between all atoms, solid as well as liquid, is the pairwise Lennard-Jones (12-6) interactions:

$$u_{ij}(r) = 4\varepsilon\left[\left(\frac{\sigma}{r_{ij}}\right)^{12} - \left(\frac{\sigma}{r_{ij}}\right)^{6}\right] \qquad r_{ij} \le r_c$$

where r is the distance between any pair of atoms i and j. The parameters ε and σ defines the strength of the interaction and the length scale, respectively.

Force resulting from LJ potential is written as

$$f = -\nabla u(r) \qquad \text{or}$$

$$f_{ij} = \left(\frac{48\varepsilon}{\sigma^2}\right)\left[\left(\frac{\sigma}{r_{ij}}\right)^{14} - \frac{1}{2}\left(\frac{\sigma}{r_{ij}}\right)^{8}\right] r_{ij}$$

provided $r_{ij} \le r_c$,zero otherwise and is shown in Fig. As r increases towards r_c the force drops to zero, so that there is no discontinuity at r_c; f and higher derivatives are discontinuous. The Lennard-Jones potential and force are shown in Fig. 2.2.

In general, to express these equations used in MD in dimensionless format. There are several reasons for doing this, not the least being the ability to work with numerical values not far from unity, instead of the extremely values normal associated with the atomic scale. Another benefits of using dimensionless units are that the equations of motion are simplified because some, if not all, of the parameters defining the model are absorbed into the units. From practical view point, the switch to such units removes any risk of encountering values lying outside the range that is represent by the computer hardware. The non-dimensional units relate to MD equations is listed in Table.2.1.

2.4 Force computations

2.4.1 All pairs

It is the simplest to implement, but extremely inefficient when the interaction range r_c is relatively small compared to the linear size of simulation region. All pairs of atoms must be examined, because it is not known in advance which atoms actually interact owing to the continual rearrangement that characterizes the fluid state. Although testing whether atoms are separated by less than r_c is only a part of the overall interaction computation, the fact that the amount of computation needed grows as O(N2), where N is the number of particles. This rules out the method for all but the smallest values of N. Two techniques for reducing the growth rate to a more acceptable O(N) level, often used in tandem to within a numerical factor. This clearly represents the lower bound for the amount of work required to process all N atoms. [23] (Fig. 2.3)

2.4.2 Cell subdivision

Cell subdivision provides a means of organizing the information about atom positions into a form that avoids most of the unnecessary work and reduces the computational effect. Imagine that the simulation region is divided into a lattice of small cells, and that the cell edges all exceed r_c. Then if atoms are assigned to cells based on basis of their current positions it is obvious that interactions are only possible between atoms that are either in the same cell or in immediately adjacent cells; if neither of these conditions are met, then the atoms must be at least r_c apart. Because of symmetry only half the neighboring cells need to be considered; thus a total of 14 neighboring cells must be examined in three dimensions (these numbers include the cell itself). The wraparound effect due to periodic boundaries is readily incorporated into the scheme. Clearly, the region size must be at least $4r_c$ for this method to be useful [23,24]. (Fig.2.3)

2.4.3 Neighbor lists

The neighbor list method is constructing the relation (neighbor lists) between every atom and surrounded atoms for a period of the assigned time step. As fig.2-3, take the atom i as the center, where r_L is the radius of the neighbor list, and $r_L = r_c + \Delta r$. In order to calculate the force of atom i due to the effect of the surrounded atoms for a period of assigned time step, it only judges by if these nearby atoms in the circle (sphere in 3-D) of radius r_L are in the circle (sphere) of radius r_c, and it dose not to calculate distances between all atoms. However, it only re-calculates the distribution of all atoms at the next of the assigned timestep. In order to allow this list to be useful over several successive timesteps we replay r_c in the test of interatomic separation by $r_n = r_c + \Delta r$, then it should be possible to benefit from this reduced neighborhood size. The success of the approach relies on the slowly changing microscopic environment, which implies that the list of neighbors remains valid over a number of time steps typically between 10 and 20 even for relatively small Δr (in general $\Delta r \approx$ 0.3-0.4) [23]. (Fig.2.3 and Fig.2.4)

2.4.4 Neighbor list + link-cell

In the current study, in combining with the neighbor list, we utilize the concept of link-cell that stores the atoms in each cell for the purpose of reducing the searching time as mentioned in the above. During the update of each neighbor list, we only search the atoms in the other 26 cells nearby and the cell where it locates. This can also reduce dramatically the time required to build up the neighbor list. Past results show that this combination represents the most efficient technique nowadays in classical molecular dynamics [23,24]. (Fig.2.3 and Fig.2.5)

2.5 Boundary conditions

2.5.1 Periodic boundary conditions

Unless the purpose of the MD simulation is to capture the physics near real walls, a problem that is actually of considerable importance, walls are better eliminated by using periodic boundary conditions (PBC). The introduction of PBC is equivalent to considering an infinite space-filling array of identical copies of simulation region [23,24].

2.5.2 Wall boundary conditions

To simulate the MD system, we would like to keep the wall isothermal. For the purpose, we define a corrected layer on wall. In the study, we use the rescaling method to modify corrected layer.

Rescaling method keep wall isothermal by modifying total kinetic energy. Under the micro-vision scale, the temperature is the performance of the atomic average kinetic. When we set the temperature of correction layer, it means to set the average kinetic energy of atoms on the correction layer, so we must keep the kinetic energy fixed. (Eq.2.18), so we have a reference valve. Then, use Eq.2.19 we compute the total kinetic energy of atoms. Finally, we start rescaling by using Eq.2.20 to make the total kinetic energy in the correction layer is the same as reference value which we computed in Eq.2.19 [25].

$$E_{kd} = \frac{3}{2} NKT_d \tag{2.18}$$

$$E_{Ka} = \frac{1}{2} m \sum_{i=1}^{N} V_i^{old^2} = \frac{3}{2} NKT_a \tag{2.19}$$

$$V_i^{new} = V_i^{old} \cdot \sqrt{\frac{E_{kd}}{E_{ka}}} = V_i^* \sqrt{\frac{T_d}{T_a}} \tag{2.20}$$

where

N : Total atoms in the correction layer

K : Boltzmann constant

E_{kd} : The total kinetic energy defines by T_d

E_{ka} : The total kinetic energy of atoms in the correction layer

T_d : The temperature of boundary, which we need

T_a : The average temperature of atoms in the correction layer before modification

v_i^{old} : The velocity of atom in the correction layer before modification

v_i^{new} : The velocity of atom in the correction layer after modification

2.6 Parallel molecular dynamics method

There is no doubt about that MD simulation is a useful and valuable tool. But MD simulation is very time-consuming due to large number of time steps and possibly large number of atoms required to complete a meaningful simulation. In liquids and solids, MD simulation is required to resolve the vibration of the atoms, which limits the time step to be on the order of fentosecond. Many hundreds of thousand or even millions of time steps are needed to simulate a nanosecond in "real" time scale. In addition, up to hundreds of thousand or millions of atoms are needed in the MD simulation, even for a system size in the nanometer scale.

In the past, there have been considerable effort that concentrated on parallelizing MD simulation on the memory-distributed machine by taking the inherently parallelism. Generally, parallel implementation of the MD method can be divided into three categories, including the atom decomposition, the force decomposition and the spatial decomposition among processors.

2.6.1 Atomic–decomposition algorithm

In the atom decomposition method, each processor, which owns nearly the same number of atoms as other processors and in which atoms are not necessarily geometrically nearby, integrates the Newton's equation for all atoms and moves the atoms of their owns. However, this method requires global communication at each time step, which becomes unacceptably expensive as compared with the "useful" MD computation when the number of atoms increases to a certain amount, since each processor has to know all information (position and velocities) of all atoms at each time step. Or equivalently, the communication is *O(N)*, where *N* is the number of atoms in the system that is independent of the number of the processors, P. Thus, the atom decomposition method is generally suitable for small-scale problem

2.6.2 Force–decomposition algorithm

In the force decomposition method, it is based on a block-decomposition of the force matrix rather than a row-wise decomposition in the atom-decomposition method. It improves the *O(N)* scaling to be $O(N/\sqrt{P})$. It generally performs much better that the atom decomposition method; however, there exists some disadvantages. First, the number of processors has to be square of an integer. Second, load imbalance may become an issue. From previous experience, it is suitable for small- and intermediate-size problems.

2.6.3 Spatial–decomposition algorithm

In the spatially static domain decomposition method, simulation domains are physically divided and distributed among processors. This method so far represents the best parallel algorithm for large-scale problem in MD simulation for short-ranged interaction ; however, it only works well for a system, in which the atoms move only a very short distance during simulation or possibly distribute uniformly in space. MD simulation of solids represents one of the typical examples. In contrast, if the distribution of the atoms tends to vary very often in the configuration space, then the load imbalance among processors develops very fast during simulation, which detriments the parallel performance. Thus, a parallel MD method capable of adaptive domain decomposition may represent a better solution for resolving this difficulty.

2.6.4 PCMD (Parallel Cellular Molecular Dynamics) algorithm

A new parallel algorithm for MD simulation, named parallel cellular molecular dynamics (PCMD), is developed by MuST (Multiscale Science & Technology) laboratory in NCTU in Taiwan, employing dynamic domain decomposition to address the issue of load imbalance among processors in the spatially static domain-decomposition method. We focus on developing a parallel MD method using dynamic domain decomposition by taking advantage of the existing link-cells as mentioned earlier. In this proposed method, not only are the cells used to reduce the cost for building up neighbor list, but also are used to serve as the basic partitioning units. Similar idea has been applied in the parallel implementation of direct simulation Monte Carlo (DSMC) method, which is a particle simulation technique often used in rarefied gas dynamics. Note that in the following IPB stands for interprocessor boundary. General procedures (Fig. 2.6) in sequence include:

1. initialize the positions and velocities of all atoms and equally distribute the atoms among processors;

2. Check if load balancing is required. If required, then first repartition the domain, followed by communicate cell/atom data between processors, renumber the local cell and atom numbers, and update the neighbor list for each atom due to the data migration;

3. Receive positions and velocities of other atoms in the neighbor list for all cells near the IPB;

4. Compute force for all atoms;

5. Send force data to other atoms in the neighbor list for all cells near the IPB;

6. Integrate the acceleration to update positions and velocities for all atoms;

7. Apply boundary conditions to correct the particle positions if necessary;

8. Check if preset total runtime is exceeded. If exceeded, then output the data and stop the simulation. If not, check if it is necessary to rebuild the neighbor list of all atoms using the most update atom information.

9. If it is necessary to rebuild the neighbor list ($N=8$ in the current study), then communicate atom data near the IPB and repeated the steps 2-8. If not necessary, then repeat steps 3-8.

In the above, in addition to the necessary data communication when atoms cross the IPB and particle/cell data near the IPB, there are two more important steps in the proposed parallel MD method as compared with the serial MD implementation. One is how to repartition the domain effectively and the other is the decision policy for repartitioning. These two steps are described next, respectively.