The Parallel Simulation Model for Thin Film Deposition Using the DSMC Method

R. Silapunt
Department of Electronics and Telecomunication
T. Achalakul and A. Banharnsakun
Department of Computer Engineering, King Mongkut's University of Technology, Thonburi, Bangkok, THAILAND.

Abstract

Thin film technology is currently used extensively in many applications including microelectronics, optics, magnetic, corrosion resistant coatings, and micro-mechanics. The technology also plays an important role in semiconductor fabrication industries. The goal of the deposition process is to selectively and controllably deposit thin films on silicon wafers. This paper presents the design and development of a parallel computational tool for deposition simulation that allows process engineers to optimize the deposition process as well as to predict the optimal sticking coefficients that are used to control the uniformity and surface growth, for arbitrary input conditions. In order to study the process, several inputs such as the type of gas, initial pressure, density, substrate temperature, reactor size and substrate size, can be parameterized. Our work focuses on the deposition processes in two-dimensional geometries. The simulation tool is developed based on the neutral flow Direct Simulation Monte Carlo (DSMC). As sited in previous literature, DSMC has a high computational cost when applies to a large substrate size. Therefore, a parallel and distributed programming technique is employed in order to keep the computational cost within an acceptable limit. The tool is designed to be executed in a grid environment where high computational power is readily available. Our parallel implementation utilizes the message passing technique as the communication paradigm. The problem is partitioned using the domain decomposition method. The simulation domain is divided into several cell-grids and each grid is assigned to a separate processor. Particle transport and collision are computed independently in each processor. The inter-processor communications only occur when particles move out of the cell-grid bound. When the steady state is reached, the output data from the simulation are collected, i.e., density, velocity, and temperature of gas molecules. These values are then used to calculate the sticking coefficient based on the Langmuir model. The model describes ideal chemical mechanical absorption and is used to study the surface growth. Our simulation produces the result in the previously published theoretical ranges. Finally, the parallel performance and scalability are

observed. From our experiment, the average processing time decreased as more computing nodes are added to the computation therefore it can be concluded that our simulation tool is fast, efficient, and accurate.

Key words: Thin Film Deposition / Parallel Simulation Model / Direct Simulation Monte Carlo (DSMC)

1. Introduction

The thin film deposition process has been indispensable in the semiconductor fabrication. There are several of deposition methods available and in use today, however mainly two types of deposition are employed: Physical Vapor Deposition (PVD) and Chemical Vapor Deposition (CVD). The CVD method is utilized in this research to simulate thin film deposition since the great advantage of CVD over any other deposition method is the uniformity that can be attained, particularly over patterned substrates. Whereas PVD techniques such as sputtering and evaporation can suffer from shadowing by substrate features, the CVD process is dependent only on chemical reactions at the substrate surface.

Direct Simulation Monte Carlo (DSMC) [1-2] has been used extensively in simulation of rarefied flows. DSMC associated with the CVD method is applied in this research for a simulation of the thin film deposition process under very low pressure conditions where the Boltzmann equation is utilized for solving the flow in this transition regime. Several difficulties exist in using simulation, including high computational cost associated with the large number of samples required for thin film deposition simulation. This can be alleviated by using efficient techniques in implementing DSMC together with parallelization in order to keep the execution time within acceptable limits.

In this work, the thin film deposition simulation is performed in two-dimensional geometries and the simulation is considered in the context of neutral flow DSMC and a parallel implementation of the DSMC method is developed for distributed memory parallel computers. Message passing [3] is a common communication paradigm that can be realized on distributed memory computers, and the domain decomposition method is used to parallelize the DSMC algorithm. The goal of this research is to implement a computational tool based on the DSMC method for simulating thin film deposition processes on the parallel system that can be used to optimize deposition processes and is capable of predicting surface growth for arbitrary input conditions of deposition processes.

When the steady state is reached, the output data from the simulation are collected i.e., density, velocity, and temperature of gas molecules. These values are

then used to calculate the sticking coefficient based on the Langmuir model [4-6]. The model describes an ideal chemical mechanical absorption that is used to study the surface growth. Our simulation produces results within previously published theoretical ranges. Finally, the parallel performance and scalability are observed. From our experiment, the average processing time decreased as more computing nodes are added to the computation. It can thus be concluded that our simulation tool is fast, efficient, and accurate.

2. Related Research

There are several related works in particle simulation based on the DSMC method and parallel computing techniques used to optimize the performance of this method. Wen *et.al*. [7] presented the design of a parallel programming environment based on message passing for multi-computer systems and described the details of the parallel programming steps, including processor allocation, application downloading, data exchanging through message passing, and gathering results from the parallel processing nodes. They developed some scientific parallel programs with a similar data partitioning nature as the DSMC method, such as fractal computation, image edge extraction, dominant evaluation, and linear equation evaluation. These programs were then run on a parallel environment. This study showed that the designed parallel programming environment based on message passing decreased the difficulty and the complexity of the parallel programming method.

LeBeau *et.al*.[8] proposed a parallel implementation of the DSMC method applied to several example problems such as the operation of flight vehicles in the upper atmosphere. They described the DSMC method that was used to track molecules. The method was applied to surface geometry applications resulting in an improvement of the computational speed. The parallel programming models used in their work included techniques such as domain decomposition, communication scheduling, and dynamic load balancing.

Dietrich et.al. [9] described the development of a large system, ‘‘MONACO’’, based on a new DSMC algorithm that used a localized data structure based on a computational cell to achieve high performance on workstation architectures. For this new concept, all data related to one module were kept strictly localized and could only be shared among subroutines in the same module, thus errors were easily tracked. Their research also focused on a new data structure using object-oriented design allowing a better localization of the data in memory. As are snit, cache management was improved.

The research by Antonoiu et.al. [10] focused on parallel domain decomposition and inter-process communication using the Message Passing Interface

(MPI) library applied to semiconductor device simulation. In this work, the use of functional decomposition was avoided, since the method was not scalable, and could cause a problem in overlapping parts. Using domain decomposition, the domain was divided into pieces and each sub-problem was solved locally on its sub-domain. Like any divide and conquer method, domain decomposition can be easily parallelized. The paper also introduced the issue of dynamic load balancing in semiconductor device simulation with good success.

3. Materials and Methods

In this section, a design framework and a sequential implementation of the thin film deposition algorithm based on the DSMC method are described. The proposed algorithm aims at simulating particle transport and collision as well as calculating the sticking coefficient from the simulation data based on the Langmuir model. Then, the parallel computing approach used to improve the system performance is presented.

A. The Design Framework

Our basic design framework for the thin film deposition system involves particle simulation and calculation of the sticking coefficient. The particle simulation is based on the DSMC method. DSMC decouples the continuous process of particles movement at each time step Δt into two consecutive phases: transport and collision. The Langmuir model is then applied to find the sticking coefficient based on the output obtained from DSMC. Particles are injected from inflow boundaries and move from the top to bottom wall of the reactor. The boundary conditions must be enforced at this step since some of the particles may collide with a wall or leave the computation domain through the outflow boundaries. After completion of particle movement, all particles must be indexed and sorted in each cell for a collision simulation via a probabilistic method. When a sampling time is reached, the macroscopic data are sampled from each cell in the computational domain and the average result is calculated after the last time step is over. This average result is then used to calculate the sticking coefficient based on the Langmuir model. The design framework is divided into 8 major phases: system initialization phase, particles injection phase, particles movement phase, particles sorting and indexing phase, particles collision phase, macroscopic sampling phase, average output data phase, and sticking coefficient calculation phase as illustrated in Fig. 1.

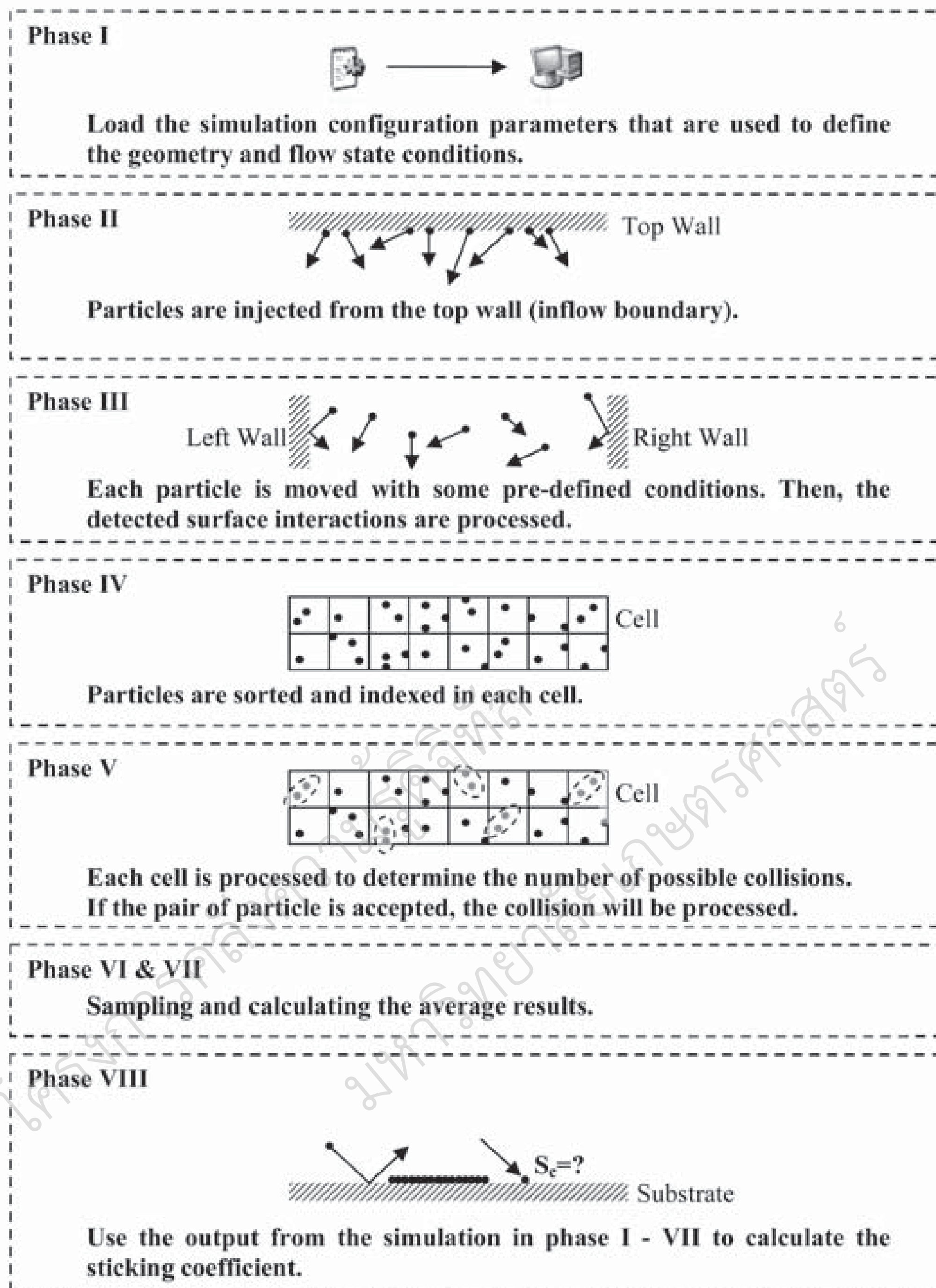


Figure 1 : The thin film deposition system design

B. Implementation Details

Step-by-step details of the particle simulation based on the DSMC method and calculation of the sticking coefficient for a thin film deposition system are described as follows.

1) System Initialization Phase

The simulation domain model illustrated in Fig. 2 is mapped from a physical domain space and divided into several cells, such that the cell size is smaller than a mean free path (λ) ; the mean free path is given by

$$\lambda = \frac{1}{\sqrt{2}\pi\sigma^2 n} \tag{1}$$

where

n = the number density of any gas.

σ = collision cross section.

A cell that is adjacent to the a substrate is called a "substrate cell". The domain simulation's height, width, substrate size, number of particles that are used to simulate, number of time steps, the time step for sampling, and the characteristic of the simulated gas as well as the flow state conditions are defined from input parameters in the a configuration file.

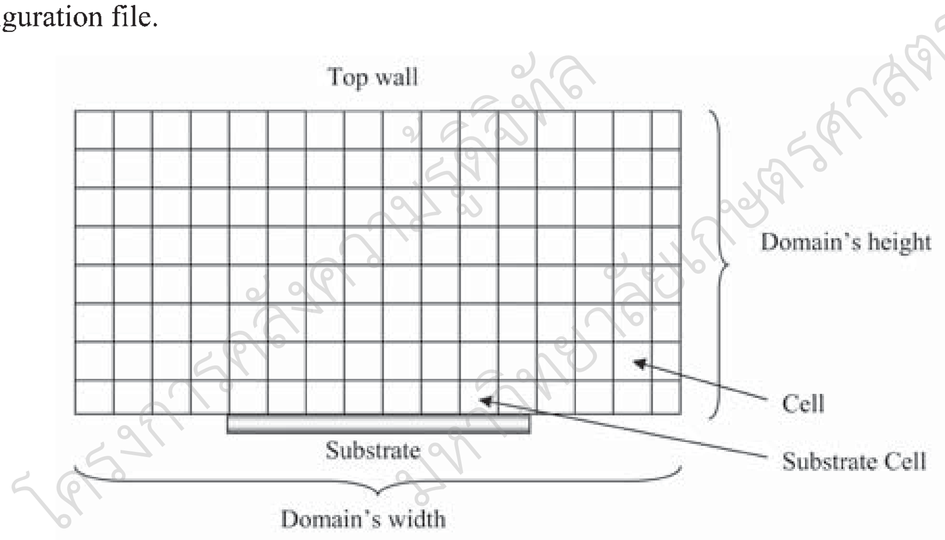


Figure 2 : The 2-dimension simulation domain model

2) Particles injection phase

The simulation particles injected from the top wall and inflow boundary,and initial velocity of each particle are assigned by following the Maxwell-Boltzmann distribution [11-12] written as

$$p(c) = 4\pi\left(\frac{m}{2\pi kT}\right)^{\frac{3}{2}} c^2 e^{\frac{-mv^2}{2kT}} \tag{2}$$

We use (2) to compute the average particle speed $\bar{c}$, root mean square particle speed c_{rms} and the most probable particle speed c_{mp} expressed by (3), (4), and (5), respectively.

$$\bar{c} = \sqrt{\frac{8kT}{\pi m}} \tag{3}$$

$$c_{rms} = \sqrt{\frac{3kT}{m}} \tag{4}$$

$$c_{mp} = \sqrt{\frac{2kT}{m}} \tag{5}$$

3) Particles movement phase

In this phase, a particle that moves without interactions with any boundary follows the equation:

$$R_j = R_i + c_{rms}\,\Delta t \tag{6}$$

where

R_i = initial coordinate of a particle.

R_j = new coordinate of a particle after movement.

c_{rms} = root mean square particle speed.

Δt = time step.

Specular surface is the boundary that is applied for top, right, and left wall to process the particle interaction in this simulation as shown in Fig. 3.

โครงการคลังความรู้ดิจิทัล มหาวิทยาลัยเกษตรศาสตร์

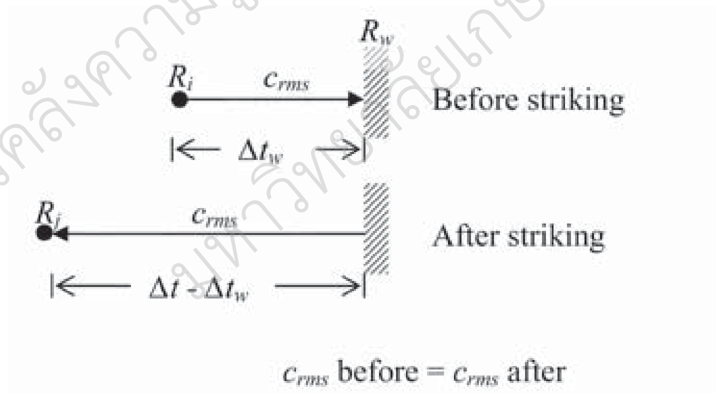


Figure 3 : Particle interactions with specular boundary.

The period of interactions after a particle strikes a wall is determined by tracing the straight-line trajectory from the initial location R_i to the point of impact, R_w. The time of flight from the particle's initial position to the point of impact is given by

$$\Delta t_w = (R_w - R_i)/c_{rms} \tag{7}$$

After striking the surface, the particle bounces off the wall with the same velocity before interactions but in a reversed direction and with remaining time Δt - Δt_w. So a new position of the particle is

$$R_j = R_w + c_{rms}(\Delta t - \Delta t_w) \tag{8}$$

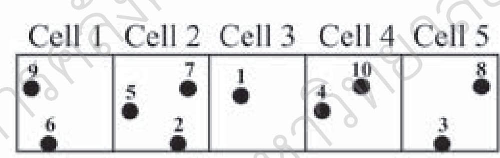
where
R_w = a point of impact.
Δt_w = the time of flight from the particle's initial position to the point of impact.
Particles that hit outflow surfaces (bottom of reactor) are simply removed from the simulation and re-initialized at the top wall with velocity given in (4). Finally, in this phase, the cell address for each particle is computed from its position.

4) Particles sorting and indexing phase

This phase prepares the sorted lists which are used to select random particles from an individual cell to calculate particle collision in the next phase. This phase involves three steps to perform the sorting and indexing:

1. Count the number of particles in each cell.
2. Build the index list as the cumulative sum of the number of particles in each cell.
3. Build a cross-reference list.

A pseudo-code function is written to create arrays that are used to select particles at random from cells. Fig.4 illustrates how the information is stored using three arrays.



Particle_Reference = [6, 9, 2, 5, 7, 1, 4, 10, 3, 8]
Number_Particle_In_Cell = [2, 3, 1, 2, 2]
Cumulative_Sum_Particle_In_Cell = [2, 5, 6, 8, 10]

Figure 4 : Illustration of particle sorting and indexing.

The cross-reference array, Particle_Reference contains particle names sorted by coordinate. Number_Particle_In_Cell array contains the number of particles in each cell and Cumulative_Sum_Particle_In_Cell is the cumulative sum of Number_Particle_In_Cell. Particle_Reference can be determined using Number_Particle_In_Cell and Cumulative_Sum_Particle_In_Cell when drawing particles randomly from a given cell.

Using the example in Fig. 4, suppose a random particle from cell 2 is requested, then the candidates are Particle_Reference [3] = Cumulative_Sum_Particle_In_Cell [2]) through Particle_Refercnce [5] 5 = Cumulative_Sum_Particle_In_Cell [2] + Number_Particle_In_
Cell [2] - 1), namely
Particle_Refercnce [3] = 2
Particle_Refercnce [4] = 5
Particle_Refercnce [5] = 7

5) Particles collision phase

The sorted particles and indexing array obtained from the previous phase are used to evaluate particle collision in this phase.

6) Macroscopic sampling phase

Each substrate cell condition is evaluated for the sampling phase Each substrate cell is sampled must contain least 20 particles. Macroscopic characteristics that are sampled and collected from these substrate cells include density, velocity in the x-and velocity in the x-and y- directions, and root mean square speed.

7) Average output data phase

After completion of the last time step, all sampled data from each time sampling are collected and used to calculate average macroscopic values in each cell, which include average density, velocity, temperature, pressure, and incident molecular flux.

8) Sticking coefficient calculation phase

Finally, the output from the particle simulating phase is applied to the calculation of the average sticking coefficient based on the Langmuir model. A sticking coefficient, $s(\theta)$, [4-6, 13] is defined by

$$s(\theta) = \frac{\text{Rate of adsorption}}{\text{The total flux of molecules onto the surface}} \tag{9}$$

where
θ = the fractional coverage of a surface

and
$$\theta = \frac{\text{number of occupied adsorption}}{\text{total number of possible sites}}$$

The next step is to define the behavior of the sticking probability as a function of coverage. In the Langmuir model, $s(\theta) = 1$ for empty sites and $s(\theta) = 0$ for filled sites Therefore, for nondissociative adsorption, the sticking probability is equal to the probability of striking to an empty site

$$s(\theta) = 1 - \theta \tag{10}$$

Several extensions to Langmuirian adsorption, the initial sticking probability need not be exactly unity; thus, s_0 is defined and stands for an initial sticking probability, so the sticking probability can be written as

$$s(\theta) = s_0(1-\theta) \tag{11}$$

4. The Parallel System

The particle simulation process is computationally intensive and requires large memory usage. In order to keep the execution time within an acceptable limit, an efficient parallel algorithm is designed for DSMC. In the algorithm, tasks are divided to a set of sub-tasks and each sub-task is performed concurrently by different processors. In this work, PC clusters are used as the experimental test bed. Fig. 5 illustrates the concurrent architecture.

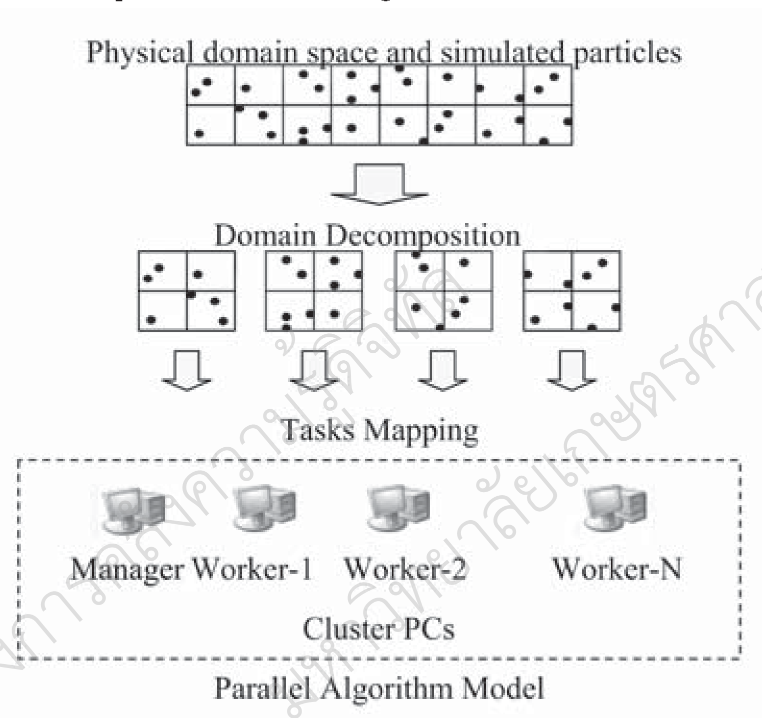


Figure 5 : The concurrent architecture.

The process of designing and implementing parallel algorithms involves three components:

- Domain Decomposition
- Tasks Mapping
- Parallel Algorithm Model

The Manager-Worker model [14] is employed in our parallel implementation. The model consists of one manager and multiple workers. The manager is responsible for initialization, load balancing, decomposing the data, distributing sub-tasks to workers, and collecting and displaying the simulation results. The workers are actual computing nodes where the sub-tasks are performed. Once finished, the partial results are returned to the manager. Inter-processor communications occur only between the manager and the workers based on MPI. The parallel DSMC algorithm model can be shown in Fig. 6.

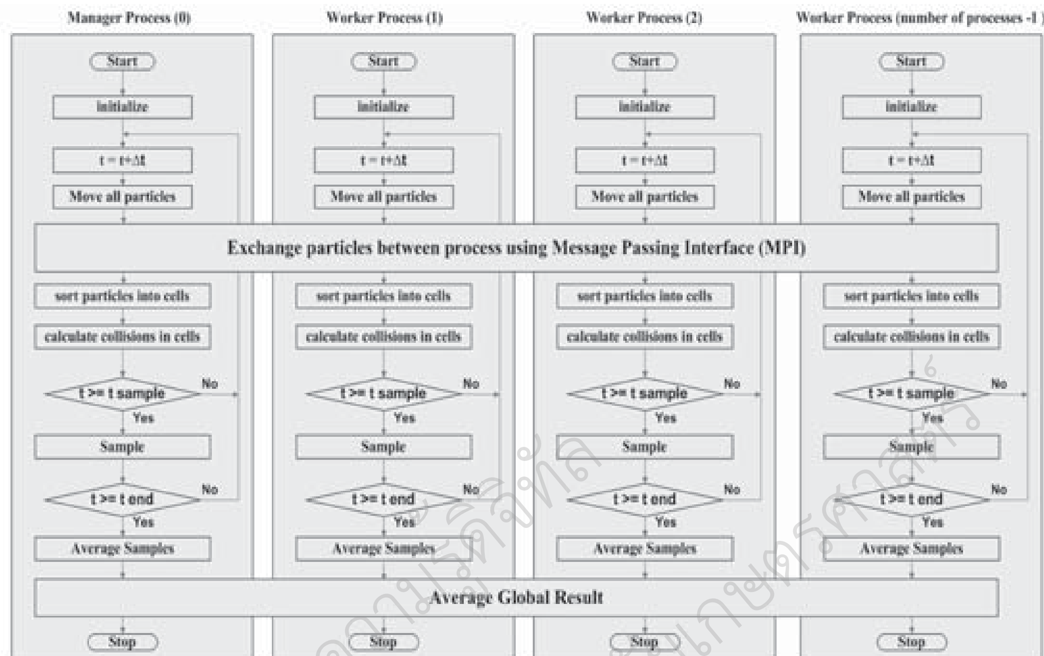


Figure 6 : Parallel DSMC algorithm flowchart.

5. Experimental Results

The experimental results are presented in perspectives of both simulation accuracy and parallel scalability.

A. Simulation Accuracy

In our thin film deposition experiment, CF_2 radical gas is applied under different conditions to investigate surface reactions. The experimental configuration is:

- The plasma chamber is cylindrical with a diameter of 30 cm.
- The plasma chamber height is 20 cm.
- The substrate dimension is 10 cm.

Several conditions can affect surface reactions directly. The most common ones are density, pressure, and substrate temperature [15]. Our system was tested under various conditions and collected data compared in terms of the accuracy compared to a similar measurement based on DSMC method and Langmuir model. In the first experiment, the growth of coverage with exposure in various pressure conditions is evaluated. The coverage and exposure are calculated as follows:

$$Coverage = \theta N_s \tag{12}$$

$$Exposure = I_z t \tag{13}$$

where N_s is the overall surface coverage and I_z is the exposure rate along the z-axis.

Applying (12) and (13) to our simulating data yields the coverage and exposure as illustrated in Fig. 7. The coverage increases with exposure and also with pressure.

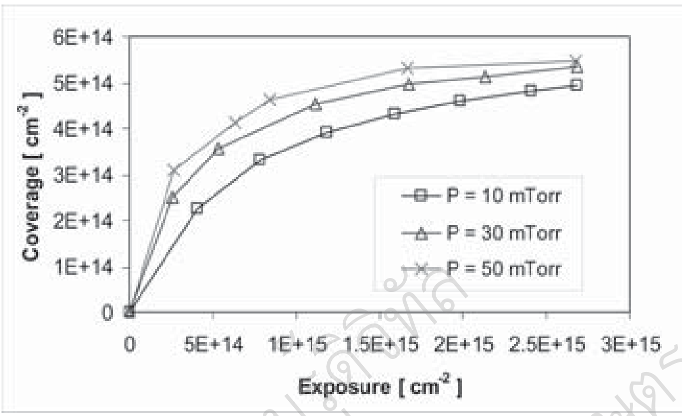


Figure 7 : Coverage and exposure in various pressure conditions.

Fig. 8 shows the simulation result of the variation in sticking coefficient at 10 mTorr, which decreases linearly with coverage as in Lanmuirian adsorption.

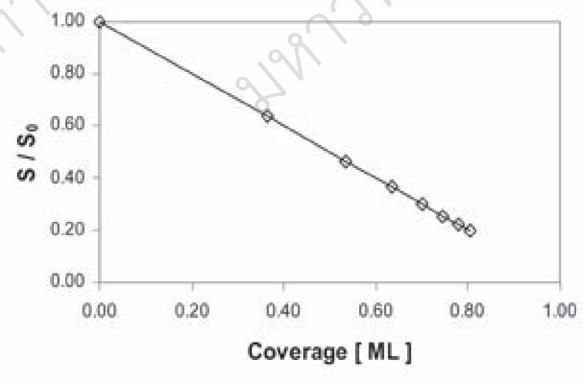


Figure 8 : The change in sticking coefficient with coverage.

โครงการห้องสมุดดิจิทัล มหาวิทยาลัยเกษตรศาสตร์

The effect of the substrate temperature on the coverage, shown in Fig. 9, illustrates the variation of coverage with exposure at different substrate temperatures. At the same exposure level, the coverage decreases with increasing substrate temperature.

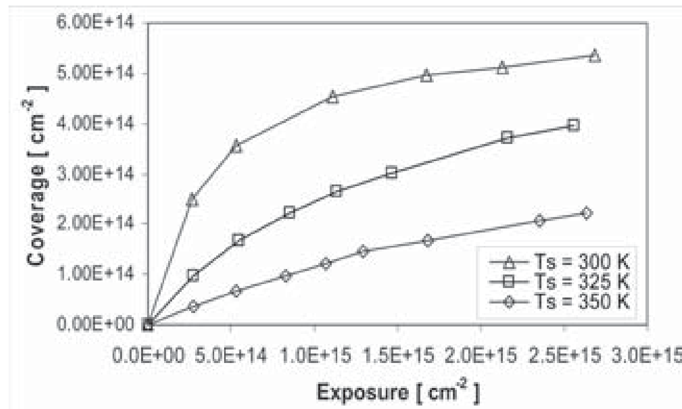


Figure 9 : Coverage and exposure in various substrate temperatures

Finally, Fig. 10 illustrates that the sticking coefficient drops with increasing CF_2 densities.

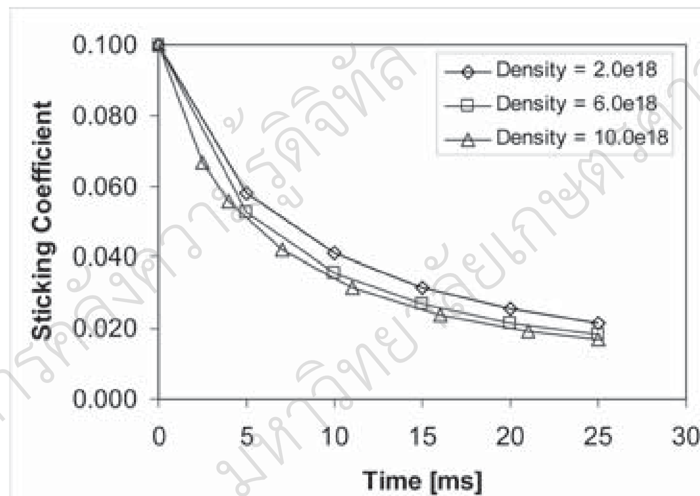


Figure 10 : The change in CF_2 density with sticking coefficient.

B) Algorithm Scalability

In this section, the parallel scalability analysis is discussed. The experiments were performed on a PC cluster each node has a 2.8GHz CPU and 3947 Mb of main memory. Our experimental test-bed represents a regular student cluster, where computers are not state of the art machines.

1) Variations in Problem Size

The problem sizes used in the experiments are 10^4, 10^5 and 10^6 particles. Fig. 11 to 13 illustrate the performance of the parallel algorithm versus for the three different cluster size particle counts.

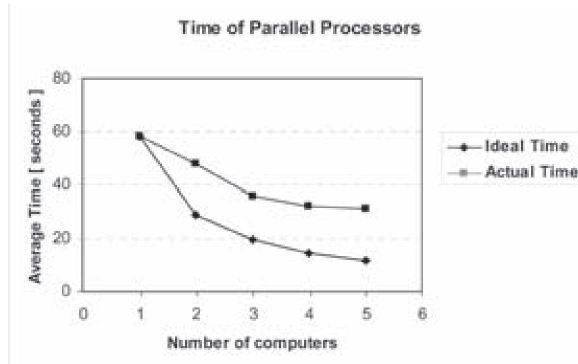


Figure 11 : Scalability plot (number of particles =10^4)

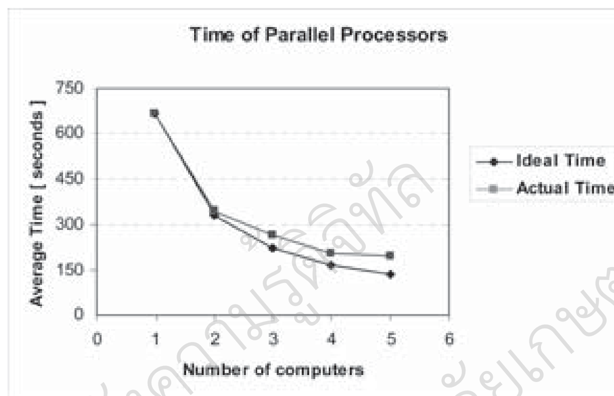


Figure 12 : Scalability plot (number of particles =10^5)

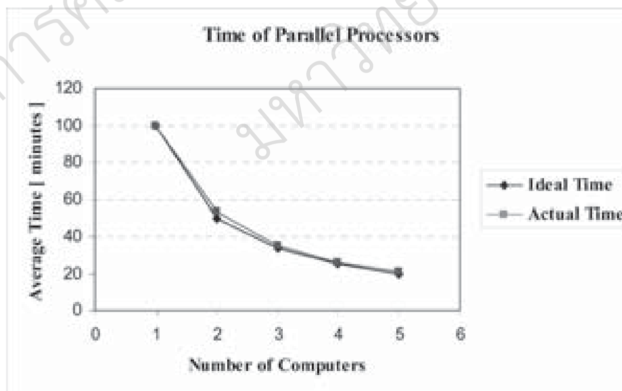


Figure 13 : Scalability plot (number of particles =10^6)

โครงการห้องสมุดดิจิทัล มหาวิทยาลัยเกษตรศาสตร์

These plots indicate that, for the larger problem size the parallel algorithm performs very close to the ideal. The results show that the algorithm is more scalable as the problem size grows. For small problem size, there is no sufficient computation to gain an impact from a large number of processors. Moreover, as a number of particles in the simulation decreases, a number of column cells in the sub-domain for each processor will also decrease since the number of particles is limited to only 20 particles per cell for the sampling process during simulation. Hence, the frequency of particle exchanges between processors rises, resulting in an increase in communication overhead.

The effect of gas density on the scalability is discussed next. Fig. 14 and 15 show the performance of our parallel algorithm for two different gas densities. The densities used in the experiments are 10^{11} molecules/cm^{-3} (Collisionless) and 10^{14} molecules/cm^{-3} (Collisional).

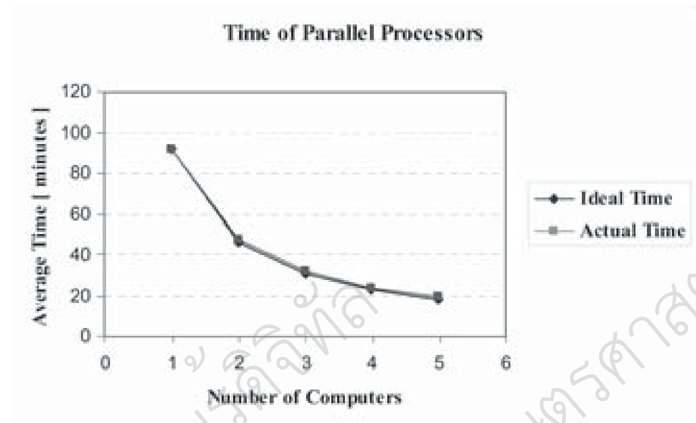


Figure 14 : Scalability plot (gas density 10^{11} molecules/cm^{-3})

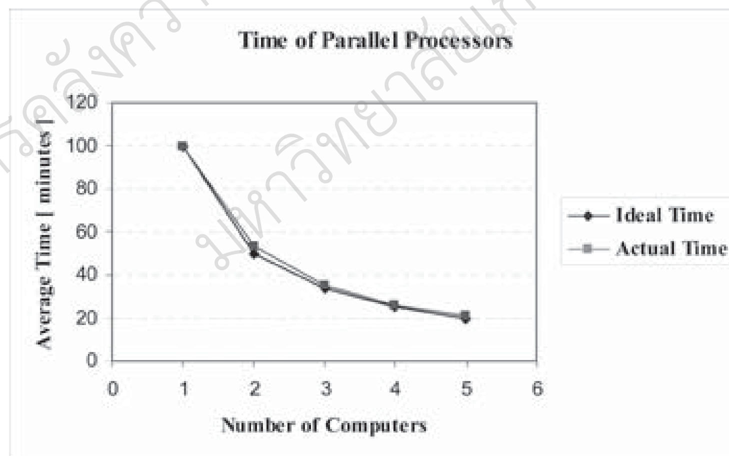


Figure 15 : Scalability plot (gas density 10^{14} molecules/cm^{-3})

The execution times drop nonlinearly as more processors are added in both cases. Therefore the gas density does not appear to have a significant effect on the scalability.

โครงการคลังความรู้ดิจิทัล มหาวิทยาลัยเกษตรศาสตร์

By adding more processors, a very large simulation can be conducted in a relatively short amount of time using a cluster of state of the art machines, and our model can be used as a simulation tool for simulating a vast number of particles in thin film deposition systems.

Due to the grid computing technology is emerging, a future version of the algorithm could be designed to run on a grid, with the load balancing and job distribution issues handled by the grid management layer. Our design framework is expected to be able to accommodate use for studying the gas flow and adsorption process based on Langmuir model in thin film deposition system.

In our experiments, the system is tested under several initial conditions: the growth of coverage with exposure in various pressure conditions, the change in sticking coefficient with coverage, the effect of substrate temperature on the coverage, and the sticking coefficient in various CF_2 densities. The system is able to accurately produce results which are consistent with the work presented in previous literature under all conditions.

Parallel computing is employed in order to keep the execution time within an acceptable limit. An efficient parallel algorithm is designed for DSMC to run on a cluster of PCs and the implementation is done based on the MPI. Static load balancing is also applied to eliminate the possibility of system imbalance. The parallel scalability is analyzed and shown that the average processing time decreased almost proportionally as computers are added to the system. For a large problem size, our parallel algorithm scales almost linearly.

6. Conclusions

This project described a design framework and a parallel simulation of the thin film deposition system using the DSMC method. The design framework of the thin film deposition system involves particle simulation and calculation of the sticking coefficient. The particle simulation is based on the DSMC method, which decouples the continuous process of particles movement into two consecutive phases: transport and collision at each time step. The Langmuir model is then applied to find the sticking coefficient based on the output obtained from DSMC.

In the experiment, the system is tested under several initial conditions: the growth of coverage with exposure in the various pressure conditions, the change in sticking coefficient with coverage, the effect of substrate temperature on the coverage, and the sticking coefficient in various CF_2 densities. The system is able to correctly produce results that follow the Langmuir model under all conditions.

An efficient parallel computing algorithm is designed for DSMC to run on a cluster of PCs using MPI. Static load balancing is also applied to eliminate the possibility of system imbalance. The parallel scalability is analyzed and it is shown that the average processing time decreased almost proportional to the number of computers in the cluster. Moreover, for a large problem size, our parallel algorithm performs very close to the ideal.

This system performs efficiently when a very large number of particles are used in simulation, and the computing time is in a reasonable scale. In future, implementations, this thin film deposition system will include real-time parameterized conditions and 3-D particle simulation as well as an adaptive domain decomposition that is one of the challenges to enhance system performance. Based on our results, we believed that a parallelized simulation using DSMC and the Langmuir model offer an effective approach to optimizing the deposition process in semiconductor fabrication.

7. Acknowledgements

This research is conducted in collaboration with the Thai National Grid Center (TNGC).

8. References

[1] Bird, G.A., 1976. Molecular Gas Dynamics, Clarendon Press, Oxford, 238 pp.
[2] Bird, G.A., 1994. Molecular Gas Dynamics and the Direct Simulation of Gas Flows, Clarendon Press, Oxford, 458 pp.
[3] Quinn, M.J., 2003, Parallel Programming in C with MPI and OpenMP, McGraw-Hill, Singapore.
[4] Kolasinski, K.W., 2002. Surface Science: Foundations of Catalysis and Nanoscience, John Wiley, Chichester, 305 pp.
[5] Masel, R.I., 1996. Principles of adsorption and reaction on solid surfaces, John Wiley, New York, 804 pp.
[6] McCash, E.M., 2001. Surface chemistry, Oxford University Press, New York, 177 pp.
[7] Wen, Y., Wang, D., Shen M. and Zhen, W., 1994, "A Parallel Programming Environment Based on Message Passing", Parallel and Distributed Processing, 724-729 pp.
[8] LeBeau, G.J., 1999. A parallel implementation of the direct simulation Monte Carlo method. Computer Methods in applied Mechanics and Engineering, vol. 174: 319–337 pp.
[9] Dietrich, S., and Boyd, I., 1996. Scalar and parallel optimized implementation of the direct simulation Monte Carlo method. Journal of Computational Physics, vol. 126: 328–342 pp.
[10] Antonoiu, G., Dima, G. and Profirescu, M.D., 1998, "Parallel Domain Decomposition. for 3D Semiconductor Device Simulation", Proceedings of the 21st International Semiconductor Conference, Sinaia-Romania, vol. 2, 371-374 pp.
[11] Lieberman, A. and Lichtenberg, J., 1994, Principles of Plasma Discharges and Materials Processing, Wiley-Interscience, New York.
[12] Garcia, A.L., 2000. Numerical Methods for Physics, Prentice Hall, New Jersey, 423 pp.
[13] Smith, D.L., 1995, Thin-film deposition: principles and practice, McGraw-Hill, New York.
[14] Grama, A., Gupta, A., Karypis, G., and Kumar, V., 2003, Introduction to parallel computing, Addison-Wesley, New York.
[15] Zhang, D., and Kushner, M.J., 2000, "Mechanisms for CF_2 radical generation and loss on surfaces in fluorocarbon plasma", J. Vac. Sci. Technol. A 18, 2661–2668 pp.