Computer Physics Communications 183 (2012) 1225–1236

Contents lists available at SciVerse ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

Development of a parallel semi-implicit two-dimensional plasma fluid modeling code using finite-volume method

K.-M. Lin [a], C.-T. Hung [a], F.-N. Hwang [b], M.R. Smith [c], Y.-W. Yang [a], J.-S. Wu [a,c,*]

[a] Department of Mechanical Engineering, National Chiao Tung University, Hsinchu, Taiwan
[b] Department of Mathematics, National Central University, Taoyuan, Taiwan
[c] National Center for High-Performance Computing, Hsinchu, Taiwan

ARTICLE INFO

Article history:
Received 12 July 2011
Received in revised form 30 January 2012
Accepted 2 February 2012
Available online 6 February 2012

Keywords:
Additive Schwarz method (ASM)
Finite-volume method
Local mean energy approximation (LMEA)
Parallel computing
Plasma fluid modeling

ABSTRACT

In this paper, the development of a two-dimensional plasma fluid modeling code using the cell-centered finite-volume method and its parallel implementation on distributed memory machines is reported. Simulated discharge currents agree very well with the measured data in a planar dielectric barrier discharge (DBD). Parallel performance of simulating helium DBD solved by the different degrees of overlapping of additive Schwarz method (ASM) preconditioned generalized minimal residual method (GMRES) for different modeling equations is investigated for a small and a large test problem, respectively, employing up to 128 processors. For the large test problem, almost linear speedup can be obtained by using up to 128 processors. Finally, a large-scale realistic two-dimensional DBD problem is employed to demonstrate the capability of the developed fluid modeling code for simulating the low-temperature plasma with complex chemical reactions.

1. Introduction

Low-temperature plasma has been widely used in various industrial fields such as display technologies, materials processing, and fabrication of electronic devices, among others [1,2]. Understanding of low-temperature plasma had strongly relied on experimental observations or trial-and-error approaches, which are expensive and time-consuming due to its complex physical and chemical processes. Recently, numerical simulation has become an indispensable method for revealing the underlying physics and chemistry of plasmas since direct quantitative measurements are either prohibitively costly or difficult. Hence, developing a fast, robust, and efficient plasma solver, which can handle the large-scale multidimensional plasma problems with many species and complex chemistry reactions involved, is eagerly needed in the community of plasma physics and engineering.

Particle-in-Cell with Monte Carlo collision (PIC/MCC) and fluid modeling are two of the major approaches that have been developed for the simulation of low-temperature plasma. On one hand, PIC/MCC approach is suitable for discharges at very low pressure, in which the energy distribution is far from equilibrium so that the physics of plasmas cannot be modeled properly by the assumption of a continuum fluid. However, the kinetic particle-based PIC/MCC requires tracking of a large number of pseudo particles to reduce statistical uncertainties of the simulation, which is often very time-consuming and was thus rarely used to treat the discharges with complex plasma chemistry. Hybrid modeling is another alternative numerical technique by taking advantage of both the PIC/MCC and fluid modeling method [3]. Similarly, this method requires tracking of a sufficiently large number of particles, which could result in unacceptable runtime for a practical problem.

On the other hand, fluid modeling of low-temperature plasma is based on the governing equations derived from the velocity moments of the Boltzmann equation with appropriate assumptions [2]. Fluid modeling is suitable for low-temperature plasmas in a wide range of pressures (from low pressure to atmospheric pressure). Generally, there are two types of approximations used in the fluid modeling: 1) local field approximation (LFA) and 2) local-mean-energy approximation (LMEA). The former assumes the locally absorbed electric power is fully balanced by the power dissipated through ionization, while the latter solves the electron energy density equation to obtain the electron temperature which is related to the evaluation of reaction rate constants and other transport properties associated with electrons. LMEA has been shown to be more accurate than LFA in fluid modeling of low-pressure gas discharges [4]. For wider future applications, the LMEA is adopted in the current study to consider non-local effect of electron energy distribution that LFA generally lacks.

Fluid modeling generally requires less computational time as compared to other methods. Nevertheless, it is still an issue for large-scale multidimensional problems with many species and complex chemistry, which could lead to unacceptable computational

* Corresponding author at: Department of Mechanical Engineering, National Chiao Tung University, Hsinchu, Taiwan. Tel.: +886 3 573 1693; fax: +886 3 611 0023.
E-mail address: chongsin@faculty.nctu.edu.tw (J.-S. Wu).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2012.02.001

time. Recently, a representative plasma simulation package *Plasimo* using the finite-volume method, developed by van Dijk et al. [5], has been demonstrated as a parallel version using symmetric multi-processing (SMP) with OpenMP protocol. To speed up the computation of large-scale plasma fluid modeling, parallel computing using very popular distributed memory machines with message passing interface (MPI) is required.

In the past, there are very few studies focusing on parallel implementation of low-temperature plasma fluid modeling on distributed memory machines, albeit the importance of reduced computational time cannot be overemphasized for practical applications. Among the very few, the parallel fully implicit Newton–Krylov–Schwarz (NKS) algorithm was employed to solve the coupled large sparse, algebraic nonlinear system of the discrete governing equations of fluid modeling derived from the fully implicit scheme [6,7]. Although the speedup of parallel computing is scalable up to hundreds of processors, the overall computational time is too large for realistic large-scale multidimensional problems. This obstacle could be overcome by the so-called semi-implicit method, which solves the fluid modeling equations independently with proper linearization of the source terms of the Poisson equation [8] and the electron energy density equation [9]. In this approach, the coupled nonlinear system of plasma fluid modeling equations become linear and decoupled so that they can be solved equation by equation. It was shown that much larger time step could be employed to greatly shorten the computational time in sequential implementation [8,9].

Thus, the major objective of this paper is to develop a parallel two-dimensional plasma fluid modeling code using the cell-centered finite-volume method with the semi-implicit approach. The resulting linear systems of discretized equations are solved by the parallel generalized minimal residual method (GMRES) [10] in conjunction with the parallel additive Schwarz method (ASM) [11] as the preconditioner to accelerate its convergence. The Schwarz type methods have been proved to be theoretically optimal for many types of problems, and practically powerful for solving large problems on computers with thousands of processors. The preconditioner is decomposed into several sub-domains by domain decomposition for parallel computing. The computational time could be dramatically reduced with the combination of preconditioning and linear matrix solvers for various modeling equations.

The remainder of the paper is organized as follows. The plasma fluid model and numerical method are introduced in Sections 2 and 3, followed by the results of parallel performance presented in Section 4. Finally, the major findings of the present study are summarized at the end of this paper.

2. Model description

2.1. Plasma fluid model

The governing equations for the plasma fluid model are similar to those solved by Hung et al. [12] and briefly described below for completeness. The general continuity equation for ion species can be written as

$$\frac{\partial n_p}{\partial t} + \vec{\nabla}\cdot\vec{\Gamma}_p = \sum_{i=1}^{r_p} S_{p_i} \quad p = 1,\ldots,K \tag{1}$$

where n_p is the number density of ion species p, K is the number of ion species, r_p is the number of reaction channels that involve the creation and destruction of ion species p and $\vec{\Gamma}_p$ is the particle flux that is expressed, based on the drift–diffusion approximation, as

$$\vec{\Gamma}_p = \mathrm{sign}(q_p)\mu_p n_p \vec{E} - D_p \vec{\nabla} n_p \tag{2}$$

$$\vec{E} = -\vec{\nabla}\phi \tag{3}$$

where q_p, $\vec{E}$, ϕ, μ_p, and D_p are the ion charge, the electric field, the electric potential, the ion mobility, and the ion diffusivity respectively. Note that the form of the source term S_{p_i} can be modified according to the modeled reactions describing how the ion species p is generated or destroyed in reaction channel i.

The continuity equation for electron species can be written as

$$\frac{\partial n_e}{\partial t} + \vec{\nabla}\cdot\vec{\Gamma}_e = \sum_{i=1}^{r_e} S_{e_i} \tag{4}$$

where n_e is the number density of electrons, r_e is the number of reaction channels that involve the creation and destruction of electrons and $\vec{\Gamma}_e$ is the corresponding particle flux that is expressed, based on the drift–diffusion approximation, as

$$\vec{\Gamma}_e = -\mu_e n_e \vec{E} - D_e \vec{\nabla} n_e \tag{5}$$

where μ_e and D_e are the electron mobility and electron diffusivity, respectively. These two transport coefficients can be readily obtained as a function of the electron temperature from the solution of a publicly available computer code for the Boltzmann equation, named BOLSIG$^+$ [13]. Similar to S_{P_i}, the form of S_{e_i} can also be modified according to the modeled reactions that generate or destroy the electron in reaction channel i.

The continuity equation for neutral species can be written as

$$\frac{\partial n_{uc}}{\partial t} + \vec{\nabla}\cdot\vec{\Gamma}_{uc} = \sum_{i=1}^{r_{uc}} S_{uc_i} \quad uc = 1,\ldots,L \tag{6}$$

where n_{uc} is the number density of uncharged neutral species uc, L is the number of neutral species, r_{uc} is the number of reaction channels that involve the generation and destruction of uncharged species uc and $\vec{\Gamma}_{uc}$ is the corresponding particle flux which can be expressed as

$$\vec{\Gamma}_{uc} = -D_{uc}\vec{\nabla} n_{uc} \tag{7}$$

where D_{uc} is the diffusivity of neutral species. It is noted that the convective effect is neglected in this study. Similarly, the form of S_{uc_i} can also be modified according to the modeled reactions that generate or destroy the neutral species in reaction channel i.

The electron energy density equation can be expressed as

$$\frac{\partial n_\varepsilon}{\partial t} + \vec{\nabla}\cdot\vec{\Gamma}_{n_\varepsilon} = -e\vec{\Gamma}_e\cdot\vec{E} - \sum_{i=1}^{s_c}\varepsilon_i k_i n_i n_e - 3\frac{m_e}{M} n_e k_B \nu_m (T_e - T_g) \tag{8}$$

where n_ε $(= \frac{3}{2} n_e k_B T_e)$ is the electron energy density, T_e is the electron temperature, ε_i and k_i are the energy loss and rate constant for the ith inelastic electron collision respectively, n_i is the number density of species related to the ith inelastic electron collision, S_c is the number of reaction channels of inelastic electron collision, k_B is the Boltzmann constant, ν_m is the momentum exchange collision frequency between the electron (mass m_e) and the background neutral (mass M), T_g is the background gas temperature and is assumed to be 400 K. $\vec{\Gamma}_{n_\varepsilon}$ is the corresponding electron energy density flux and can be expressed as

$$\vec{\Gamma}_{n_\varepsilon} = \frac{5}{2} k_B T_e \vec{\Gamma}_e - \frac{5}{2} D_e n_e \vec{\nabla}(k_B T_e) \tag{9}$$

The second term on the right-hand side of Eq. (8) represents the sum of the energy losses of the electrons due to inelastic collision with other species. The last term on the right-hand side of Eq. (8)

can be ignored for low-pressure gas discharges, while it is important for medium-to-atmospheric pressure discharges.

The Poisson equation for electrostatic potential can be expressed as

$$\vec{\nabla}\cdot(\varepsilon\vec{\nabla}\phi) = -\sum_{i=1}^{K}(qn)_i \tag{10}$$

where ϕ is the potential and the permittivity ε, is a function of position, is written as

$$\varepsilon = \varepsilon_r\varepsilon_0 \tag{11}$$

where ε_0 is the vacuum permittivity, and ε_r is the relative permittivity of each region. Several different regions including discharge, dielectric materials (such as alumina, substrate, and Teflon), and conductors are considered simultaneously in this study. All of these regions are meshed and solved to obtain the electrostatic potential distribution by using the Poisson equation.

2.2. Boundary conditions

The flux-type boundary conditions for the ions, electrons, and neutral species are employed on the solid surfaces (dielectric or electrode) as

$$\vec{\Gamma}_p = a\cdot\mathrm{sign}(q_p)\mu_p n_p\vec{E} - D_p\vec{\nabla}n_p \tag{12}$$

$$\vec{\Gamma}_e = -a\cdot\mu_e n_e\vec{E} - D_e\vec{\nabla}n_e + \frac{1}{4}n_e v_{th} \tag{13}$$

$$\vec{\Gamma}_{uc} = -D_{uc}\vec{\nabla}n_{uc} \tag{14}$$

where $a = 1$ if drift velocity $(\mathrm{sign}(q_p)\mu_p\vec{E})$ points toward the dielectric surface, and $a = 0$ otherwise. We assume that the ions and electrons accumulate and the neutral species quench at the dielectric surface in the present study. The thermal velocity of electron is

$$v_{th} = \sqrt{\frac{8k_BT_e}{\pi m_e}} \tag{15}$$

where m_e is the electron mass. Note that the effect of secondary electron emission is neglected. For all species, the fluxes at the boundaries of computational domain (except the dielectric surfaces) are assumed to be zero.

The boundary conditions of electron energy density flux at the dielectric surfaces are

$$\vec{\Gamma}_{n_\varepsilon} = 2k_BT_e\vec{\Gamma}_e \tag{16}$$

For the Poisson equation, the potentials of powered and grounded electrode are assigned with applied voltage and zero potential respectively. Neumann boundary conditions with zero gradients are applied to the other boundaries of the computational domain for the Poisson equation.

2.3. Plasma chemistry

For the case of parallel performance study, we consider a helium dielectric barrier discharge containing trace nitrogen impurity since it was reported that discharge current calculated from pure helium data does not quantitatively agree with experiment results [14]. Although the impurity level is typically less than 0.01%, it was reported that the impurity plays an important role in "pure" helium discharges [15,16]. In the plasma chemistry, we consider 10 species (e^-, He^+, He_2^+, N^+, N_2^+, N_4^+, He_m^*, He_{ex}^{**}, He_2^*, and N) and 43 reaction channels as listed in Table 1. Note consideration of the complex plasma chemistry of this level is not uncommon in practical applications (e.g., [20]). Reaction channels 1 to 27 consider chemistry for the pure helium discharge, and reaction channels 28 to 43 consider the effect of trace addition of N_2 and the interaction between helium and nitrogen as the mimics of impurity. Generally, the number of species considered (if not too small) does not influence the results of parallel performance which is presented later, because we have solved the continuity equations of species one by one. In addition, number of the reaction channels only influences the algebraic operations of the source terms of the species continuity equation originating from various reactions, which does not change the parallel performance of the problem very much.

The transport coefficients and the rate constants related to the electrons are calculated by solving the Boltzmann equation using BOLSIG$^+$ as mentioned earlier. Note that these coefficients are predicted and stored in a lookup table as a function of electron temperature prior to the simulation. The value of mobility for each ion species are taken from the literature [18,21–23], and the corresponding diffusivities are calculated using the Einstein relation. The diffusion coefficients of neutral species, such as those of He_m^*, He_{ex}^{**}, He_2^*, and N, are also taken from literature [18,24].

3. Numerical method

3.1. Semi-implicit treatment for the Poisson and electron energy density equations

It was reported that explicit evaluation of the source term of the Poisson equation leads to a very small time step due to the restriction of dielectric relaxation time [8]. The so-called semi-implicit treatment is thus applied on the source term of the Poisson equation to expand the time step by a Taylor's series expansion (TSE) in time. With some derivations based on a TSE in time and approximations, the Poisson equation, Eq. (10), can be rewritten as

$$\vec{\nabla}\cdot\left[\left(\varepsilon + \Delta t\sum_{i=1}^{K}\left(|q|\mu n\right)_i\right)\vec{\nabla}\phi\right] = -\sum_{i=1}^{K}q_in_i \tag{17}$$

Note the number densities and mobilities of semi-implicit term in Eq. (17) are approximated from the values of previous time level. Similar constraint on time step size can be found on the source term of the electron energy density equation, Eq. (8), and the energy source term is linearized by a TSE in electron energy with some approximations for increasing the time step size of the simulation [9]. Thus, the electron energy density equation can be rewritten as

$$\begin{aligned}&\frac{\partial n_\varepsilon}{\partial t} + \vec{\nabla}\cdot\vec{\Gamma}_{n_\varepsilon}\\ &= -e\vec{\Gamma}_e\cdot\vec{E} - n_e\sum_{i=1}^{s_c}\varepsilon_ik_in_i - 3\frac{m_e}{M}n_ek_Bv_m(T_e - T_g)\\ &\quad - \left[\frac{e}{n_e}\vec{E}\cdot\frac{\partial\vec{\Gamma}_e}{\partial\mu_e}\frac{\partial\mu_e}{\partial\bar{\varepsilon}} + \frac{e}{n_e}\vec{E}\cdot\frac{\partial\vec{\Gamma}_e}{\partial D_e}\frac{\partial D_e}{\partial\bar{\varepsilon}}\right.\\ &\quad \left. + \sum_{i=1}^{s_c}\varepsilon_i\frac{\partial k_i}{\partial\bar{\varepsilon}}n_i + 3\frac{m_e}{M}k_B\frac{\partial v_m}{\partial\bar{\varepsilon}}(T_e - T_g)\right](n_\varepsilon - n_e\bar{\varepsilon})\end{aligned} \tag{18}$$

where $\bar{\varepsilon} = 3/2k_BT_e$. The discretization form of $\frac{\partial\vec{\Gamma}_e}{\partial\mu_e}\frac{\partial\mu_e}{\partial\bar{\varepsilon}} + \frac{\partial\vec{\Gamma}_e}{\partial D_e}\frac{\partial D_e}{\partial\bar{\varepsilon}}$ has been derived by Hagelaar et al. [9], and the finite difference method is applied to evaluate $\frac{\partial k_i}{\partial\bar{\varepsilon}}$ and $\frac{\partial v_m}{\partial\bar{\varepsilon}}$. Details of the implementation can be found in Refs. [8,9], and are not described here for brevity.

Table 1
Summary of helium plasma chemistry with nitrogen impurity.

No	Reaction channel	Rate constant ($m^3\,s^{-1}$)	Threshold (eV)	Reference
01	$e + He \rightarrow e + He$	BOLSIG$^+$	0	[17]
02	$e + He \rightarrow e + He^*_m$	BOLSIG$^+$	19.82	[17]
03	$e + He \rightarrow e + He^*_m$	BOLSIG$^+$	20.61	[17]
04	$e + He \rightarrow e + He^{**}_{ex}$	BOLSIG$^+$	20.96	[17]
05	$e + He \rightarrow e + He^{**}_{ex}$	BOLSIG$^+$	21.21	[17]
06	$e + He \rightarrow e + He^{**}_{ex}$	BOLSIG$^+$	22.97	[17]
07	$e + He \rightarrow e + He^{**}_{ex}$	BOLSIG$^+$	23.7	[17]
08	$e + He \rightarrow e + He^{**}_{ex}$	BOLSIG$^+$	24.02	[17]
09	$e + He \rightarrow 2e + He^+$	BOLSIG$^+$	24.58	[17]
10	$e + He^*_m \rightarrow 2e + He^+$	BOLSIG$^+$	4.78	[17]
11	$e + He^*_m \rightarrow e + He$	2.9×10^{-15}	−19.8	[17]
12	$e + He^*_2 \rightarrow e + 2He$	3.8×10^{-15}	−17.9	[17]
13[a]	$He^+ + 2e \rightarrow e + He^*_m$	6×10^{-32}	−4.78	[17]
14[a]	$He^+_2 + 2e \rightarrow He^*_m + He + e$	2.8×10^{-32}	0	[17]
15[a]	$He^+_2 + e + He \rightarrow He^*_m + 2He$	3.5×10^{-39}	0	[17]
16[a]	$He^+_2 + 2e \rightarrow He^*_2 + e$	1.2×10^{-33}	0	[17]
17[a]	$He^+_2 + e + He \rightarrow He^*_2 + He$	1.5×10^{-39}	0	[17]
18	$He^{**}_{ex} + He \rightarrow He^+_2 + e$	1.5×10^{-17}	0	[17]
19	$He^*_m + He^*_m \rightarrow He^+_2 + e$	2.03×10^{-15}	−18.2	[17]
20	$He^*_m + He^*_m \rightarrow He^+ + He + e$	8.7×10^{-16}	−15.8	[17]
21[a]	$He^+ + 2He \rightarrow He^+_2 + He$	6.5×10^{-44}	0	[17]
22[a]	$He^*_m + 2He \rightarrow He^*_2 + He$	1.9×10^{-46}	0	[17]
23	$He^*_m + He^*_2 \rightarrow He^+ + 2He + e$	5×10^{-16}	−13.5	[17]
24	$He^*_m + He^*_2 \rightarrow He^+_2 + He + e$	2×10^{-15}	−15.9	[17]
25	$He^*_2 + He^*_2 \rightarrow He^+ + 3He + e$	3×10^{-16}	−11.3	[17]
26	$He^*_2 + He^*_2 \rightarrow He^+_2 + 2He + e$	1.2×10^{-15}	−13.7	[17]
27	$He^*_2 + He \rightarrow 3He$	4.9×10^{-22}	0	[17]
28	$He^*_m + N_2 \rightarrow e + N^+_2 + He$	7.0×10^{-17}	0	[18]
29	$He^*_2 + N_2 \rightarrow e + N^+_2 + 2He$	7.0×10^{-17}	0	[18]
30	$He^+ + N_2 \rightarrow N^+_2 + He$	5.0×10^{-16}	0	[18]
31	$He^+ + N_2 \rightarrow N^+ + N + He$	7.0×10^{-16}	0	[18]
32	$He^+_2 + N_2 \rightarrow N^+_2 + 2He$	5.0×10^{-16}	0	[18]
33	$He^+_2 + N_2 \rightarrow N^+ + N + 2He$	7.0×10^{-16}	0	[18]
34[a,b]	$2e + N^+_2 \rightarrow e + N_2$	$5.651 \times 10^{-39} T_e^{-0.8}$	0	[18]
35[b]	$e + N^+_2 \rightarrow 2N$	$2.540 \times 10^{-12} T_e^{-0.5}$	0	[18]
36[b]	$e + N_2 \rightarrow e + 2N$	$1.959 \times 10^{-12} T_e^{-0.7} \exp(-\frac{1.132\times10^5}{T_e})$	9.757	[18]
37[b]	$e + N \rightarrow 2e + N^+$	$8.401 \times 10^{-11} \exp(-\frac{1.682\times10^5}{T_e})$	14.5	[18]
38[b]	$e + N_2 \rightarrow 2e + N^+_2$	$4.483 \times 10^{-13} T_e^{-0.3} \exp(-\frac{1.81\times10^5}{T_e})$	15.6	[18]
39[b]	$e + N^+_4 \rightarrow 2N_2$	$2.0 \times 10^{-12} (\frac{T_g}{T_e})^{0.5}$	0	[19]
40[a]	$N^+_2 + 2N_2 \rightarrow N^+_4 + N_2$	1.9×10^{-41}	0	[19]
41[a]	$N^+_2 + He + N_2 \rightarrow N^+_4 + He$	1.9×10^{-41}	0	[19]
42	$N^+_4 + N_2 \rightarrow N^+_2 + 2N_2$	2.5×10^{-21}	0	[19]
43	$N^+_4 + He \rightarrow N^+_2 + He + N_2$	2.5×10^{-21}	0	[19]

[a] Rate constants are in $m^6\,s^{-1}$.
[b] T_e is the electron temperature, and T_g is the background gas temperature. Both are in Kevin.

3.2. Finite volume discretization with Scharfetter–Gummel flux approximation

In the present study, the above equations are discretized using the collocated cell-centered finite-volume method [25] as

$$\frac{\partial \eta}{\partial t} + \frac{F_{i+1/2,j} - F_{i-1/2,j}}{\Delta x_{i,j}} + \frac{G_{i,j+1/2} - G_{i,j-1/2}}{\Delta y_{i,j}} = S_{i,j} \qquad (19)$$

with

$$\eta = \begin{bmatrix} n_p \\ n_e \\ n_{uc} \\ n_\varepsilon \\ 0 \end{bmatrix}_{i,j}, \qquad F = \begin{bmatrix} \Gamma_p \\ \Gamma_e \\ \Gamma_{uc} \\ \Gamma_\varepsilon \\ \Gamma_\phi \end{bmatrix}_x$$

$$G = \begin{bmatrix} \Gamma_p \\ \Gamma_e \\ \Gamma_{uc} \\ \Gamma_\varepsilon \\ \Gamma_\phi \end{bmatrix}_y, \qquad S = \begin{bmatrix} S_p \\ S_e \\ S_{uc} \\ S_\varepsilon \\ S_\phi \end{bmatrix}_{i,j}$$

where the subscripts i and j represent the indices of cell in x- and y-direction respectively. For simplicity of presentation, the rectangular computational domain is assumed and a set of regular grids is considered. Δx and Δy are the cell width in x- and y-direction respectively. The fluxes in the continuity equations and the electron energy density equation are calculated with the Scharfetter–Gummel (SG) scheme [26]. After the backward Euler method is employed as a time-integrator, the discretized form of continuity equation is written as

$$\frac{n_{i,j}^{k+1}-n_{i,j}^{k}}{\Delta t}+\frac{\Gamma_{i+1/2,j}^{k+1}-\Gamma_{i-1/2,j}^{k+1}}{\Delta x_{i,j}}+\frac{\Gamma_{i,j+1/2}^{k+1}-\Gamma_{i,j-1/2}^{k+1}}{\Delta y_{i,j}}=S_{i,j}^{k} \tag{20}$$

with

$$\Gamma_{i+1/2,j}^{k+1}=-\frac{D_{i+1/2,j}^{k}}{x_{i+1,j}-x_{i,j}}\left[B(-X_{i+1/2,j})n_{i+1,j}^{k+1}-B(X_{i+1/2,j})n_{i,j}^{k+1}\right]$$

$$X_{i+1/2,j}=\frac{\mathrm{sign}(q)\mu_{i+1/2,j}^{k}}{D_{i+1/2,j}^{k}}(\phi_{i+1,j}-\phi_{i,j})^{k+1}$$

$$\Gamma_{i-1/2,j}^{k+1}=-\frac{D_{i-1/2,j}^{k}}{x_{i,j}-x_{i-1,j}}\left[B(-X_{i-1/2,j})n_{i,j}^{k+1}-B(X_{i-1/2,j})n_{i-1,j}^{k+1}\right]$$

$$X_{i-1/2,j}=\frac{\mathrm{sign}(q)\mu_{i-1/2,j}^{k}}{D_{i-1/2,j}^{k}}(\phi_{i,j}-\phi_{i-1,j})^{k+1}$$

$$\Gamma_{i,j+1/2}^{k+1}=-\frac{D_{i,j+1/2}^{k}}{y_{i,j+1}-y_{i,j}}\left[B(-X_{i,j+1/2})n_{i,j+1}^{k+1}-B(X_{i,j+1/2})n_{i,j}^{k+1}\right]$$

$$X_{i,j+1/2}=\frac{\mathrm{sign}(q)\mu_{i,j+1/2}^{k}}{D_{i,j+1/2}^{k}}(\phi_{i,j+1}-\phi_{i,j})^{k+1}$$

$$\Gamma_{i,j-1/2}^{k+1}=-\frac{D_{i,j-1/2}^{k}}{y_{i,j}-y_{i,j-1}}\left[B(-X_{i,j-1/2})n_{i,j}^{k+1}-B(X_{i,j-1/2})n_{i,j-1}^{k+1}\right]$$

$$X_{i,j-1/2}=\frac{\mathrm{sign}(q)\mu_{i,j-1/2}^{k}}{D_{i,j-1/2}^{k}}(\phi_{i,j}-\phi_{i,j-1})^{k+1}$$

where the superscripts k and $k+1$ represent properties of the previous and current time levels respectively and the Bernoulli function $B(X)=\frac{X}{e^{X}-1}$.

Similarly, the discretized form of electron energy density equation (8) is written as

$$\frac{n_{\varepsilon(i,j)}^{k+1}-n_{\varepsilon(i,j)}^{k}}{\Delta t}+\frac{\Gamma_{\varepsilon(i+1/2,j)}^{k+1}-\Gamma_{\varepsilon(i-1/2,j)}^{k+1}}{\Delta x_{i,j}}+\frac{\Gamma_{\varepsilon(i,j+1/2)}^{k+1}-\Gamma_{\varepsilon(i,j-1/2)}^{k+1}}{\Delta y_{i,j}}=S_{\varepsilon(i,j)}^{k} \tag{21}$$

with

$$\Gamma_{\varepsilon(i+1/2,j)}^{k+1}=-\frac{5}{3}\frac{D_{i+1/2,j}^{k}}{x_{i+1,j}-x_{i,j}}\left[B(-X_{i+1/2,j})n_{\varepsilon(i+1,j)}^{k+1}-B(X_{i+1/2,j})n_{\varepsilon(i,j)}^{k+1}\right]$$

$$\Gamma_{\varepsilon(i-1/2,j)}^{k+1}=-\frac{5}{3}\frac{D_{i-1/2,j}^{k}}{x_{i,j}-x_{i-1,j}}\left[B(-X_{i-1/2,j})n_{\varepsilon(i,j)}^{k+1}-B(X_{i-1/2,j})n_{\varepsilon(i-1,j)}^{k+1}\right]$$

$$\Gamma_{\varepsilon(i,j+1/2)}^{k+1}=-\frac{5}{3}\frac{D_{i,j+1/2}^{k}}{y_{i,j+1}-y_{i,j}}\left[B(-X_{i,j+1/2})n_{\varepsilon(i,j+1)}^{k+1}-B(X_{i,j+1/2})n_{\varepsilon(i,j)}^{k+1}\right]$$

$$\Gamma_{\varepsilon(i,j-1/2)}^{k+1}=-\frac{5}{3}\frac{D_{i,j-1/2}^{k}}{y_{i,j}-y_{i,j-1}}\left[B(-X_{i,j-1/2})n_{\varepsilon(i,j)}^{k+1}-B(X_{i,j-1/2})n_{\varepsilon(i,j-1)}^{k+1}\right]$$

$$S_{\varepsilon(i,j)}^{k}=-e\,\vec{\Gamma}_{e}\cdot\vec{E}-\sum_{m=1}^{S_e}\varepsilon_m k_{m(i,j)}^{k}n_{m(i,j)}^{k}n_{e(i,j)}^{k}$$
$$+3\frac{m}{M}n_e\nu_m\left(T_{e(i,j)}^{k}-T_{g(i,j)}\right)$$
$$=-e\left(\Gamma_{e,x(i,j)}^{k}\cdot E_{x(i,j)}^{k}+\Gamma_{e,y(i,j)}^{k}\cdot E_{y(i,j)}^{k}\right)$$
$$-\sum_{m=1}^{S_e}\varepsilon_m k_{m(i,j)}^{k}n_{m(i,j)}^{k}n_{e(i,j)}^{k}$$
$$+3\frac{m}{M}(n_e\nu_m)^{k}\left(T_{e(i,j)}^{k}-T_{g(i,j)}\right)$$

The Poisson equation (10) is discretized in a similar method as

$$\frac{1}{\Delta x_{i,j}}\left[\frac{\varepsilon'_{i,j}\varepsilon'_{i+1,j}}{\varepsilon'_{i,j}\Delta h_{x(i+1,j)}+\varepsilon'_{i+1,j}\Delta h_{x(i,j)}}\left(\phi_{i+1,j}^{k+1}-\phi_{i,j}^{k+1}\right)-\frac{\varepsilon'_{i-1,j}\varepsilon'_{i,j}}{\varepsilon'_{i-1,j}\Delta h_{x(i,j)}+\varepsilon'_{i,j}\Delta h_{x(i-1,j)}}\left(\phi_{i,j}^{k+1}-\phi_{i-1,j}^{k+1}\right)\right]$$
$$+\frac{1}{\Delta y_{i,j}}\left[\frac{\varepsilon'_{i,j}\varepsilon'_{i,j+1}}{\varepsilon'_{i,j}\Delta h_{y(i,j+1)}+\varepsilon'_{i,j+1}\Delta h_{y(i,j)}}\left(\phi_{i,j+1}^{k+1}-\phi_{i,j}^{k+1}\right)-\frac{\varepsilon'_{i,j-1}\varepsilon'_{i,j}}{\varepsilon'_{i,j-1}\Delta h_{y(i,j)}+\varepsilon'_{i,j}\Delta h_{y(i,j-1)}}\left(\phi_{i,j}^{k+1}-\phi_{i,j-1}^{k+1}\right)\right]$$
$$=-\left(\sum_{l=1}^{m}(qn)_l\right)^{k} \tag{22}$$

where $\Delta h_{x(i,j)}$ and $\Delta h_{y(i,j)}$ represents the half cell width of cell (i,j) in the x- and y-direction respectively. Note the effective local permittivity is defined as

$$\varepsilon'_{i,j}=\varepsilon_{i,j}+\Delta t\left(\sum_{l=1}^{m}\left(|q|\mu_{i,j}n_{i,j}\right)_l\right)^{k} \tag{23}$$

where the semi-implicit treatment is included.

3.3. Implementation of parallel fluid modeling code

At each time step, the resulting algebraic linear systems are solved equation by equation using parallel preconditioned Krylov subspace method provided by PETSc library [27] through domain decomposition technique on top of the MPI protocol. Fig. 1 shows the proposed flowchart of simulation. After the evaluation of transport properties and rate constants of reaction channels, the discretized governing equations are solved sequentially with acceptable time step size benefiting from the use of semi-implicit scheme.

Fig. 2 shows the computational domain of the test case. The electrodes are surrounded by Teflon, and the right-hand side domain is bounded by a substrate. Details of the experimental configuration can be found in Chiang et al. [28] and are not repeated here for brevity. The computational domain is decomposed with vertex-based partition [29] into several horizontal (or vertical) subdomains along the y- (or x-) direction. In our implementation, each sub-domain is assigned to a single processor. Such partition does not distinguish different types of physical regions such as electrodes, dielectric materials, and discharge region. Hence, the sub-domain of each processor may or may not contain a region of multi-physics.

Load balance is an important issue in parallel computing and indeed plays a great impact on the performance of a parallel code. Since the solution to the Poisson equation in the electrode region and the number densities of gas species to the continuity equations in the region of solid materials are known in priori but are all included in the solution process, it may potentially lead an issue of load imbalance. However, the load imbalance problem is not

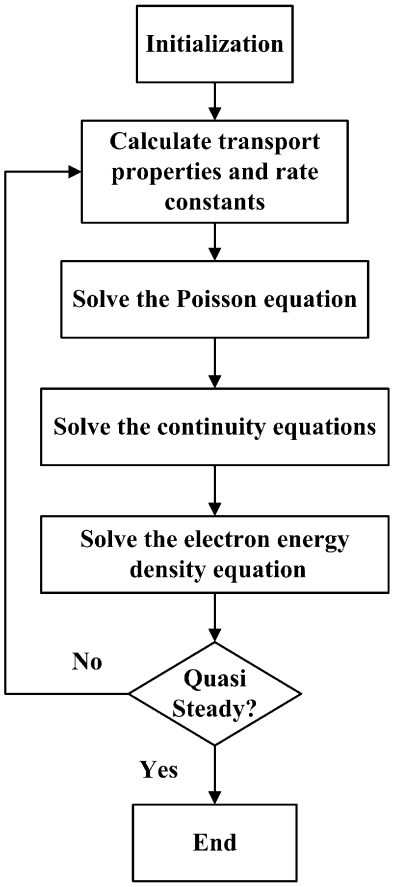

Fig. 1. The flowchart of fluid modeling simulation.

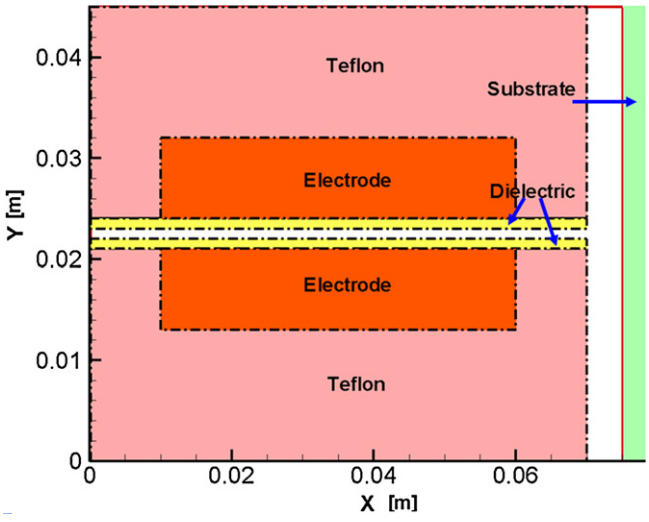

Fig. 2. Sketch of the simulation domain.

studied in the current paper but of course should be worthy of further investigation in the future.

The Poisson equation is an elliptic partial differential equation (PDE) while the continuity equations and the electron energy density equation are convection–diffusion–reaction equations either parabolic or hyperbolic types of PDE depending on the Peclet number (ratio of drift to diffusion fluxes). The continuity equations of species can be further classified into charged (such as electron and ions) and neutral species. The continuity equation of charged species consists of the mobility, the diffusivity, and the local distribution of electric field, which are varied both temporally and spatially. The coefficients of the matrices for these continuity equations need to be updated at each time step. It follows that the corresponding preconditioners of the continuity equation of charged species need to be reconstructed at each time step. On the other hand, the continuity equations of neutral species are diffusive equations and their diffusivities are treated as constant for most neutral species. Thus, the coefficients of these matrices are unchanged at each time step, leading to a constant preconditioning matrix for neutral species is sufficient.

The matrices resulting from the discretization are asymmetric because of the inclusion of the Dirichlet conditions, non-uniform cell size, and the use of SG scheme. We have employed GMRES due to its robustness for all cases tested. In addition, regarding preconditioning, we investigate the performance of additive Schwarz preconditioners, where either LU or incomplete LU decomposition without fill-in is used as a sub-domain solver. The effect of ASM with different degrees of overlapping is also discussed. Other preconditioners available in PETSc, for example, point Jacobi or Successive Over-Relaxation (SOR), are not efficient enough for the convergence of GMRES and thus excluded for further discussion. Note that GMRES without preconditioning is not worth to mention because it often leads to divergence based on our experience.

4. Results and discussion

To test the parallel performance and computational time of the continuity equations of charged and neutral species separately, we select the electron continuity equation since electrons are the most representative species among all charged species in a discharge. In addition, we only select one of the continuity equations of neutral species to test parallel performance since they have similar performance and convergence behavior. Accordingly, four types of model equations, (i) the Poisson equation, (ii) the electron continuity equation, (iii) the continuity equation of a neutral species, and (iv) the electron energy density equation are tested and discussed in this study. The selected representative equations are timed separately and the numbers of iterations of these equations are collected separately. Note that all equations are solved in the simulation since it is a self-consistent model.

The helium Dielectric Barrier Discharge (DBD) with trace nitrogen (100 ppm) between two parallel electrodes covered by ceramic materials (alumina) is simulated for testing the parallel performance of the developed code. The values of relative permittivity of each region modeled in this study are $\varepsilon_{r,\text{Discharge}} = 1.0$, $\varepsilon_{r,\text{Alumina}} = 11.63$ (measured), $\varepsilon_{r,\text{Teflon}} = 2.1$ [30] and $\varepsilon_{r,\text{Substrate}} = 10.0$ for some dielectric materials. The computational domain is shown in Fig. 2. Two problems with different sizes (501×310 cells and 1001×620 cells) are considered. Hereafter the former and the latter are referred to as the small and large problems respectively.

The data presented in the following are averaged from the results of 1000 time steps unless otherwise specified. The employed time step in the test cases was fixed at 5×10^{-10} seconds. The relative tolerance for GMRES is set as small as 10^{-7} to guarantee the physical correctness of the computed solution. In general, except for the Poisson equation, GMRES for solving the continuity equations and the electron energy density equation converge to this criterion in less than 30 iterations. The typical number of iterations required for solving the Poisson equation is generally few hundreds. All simulations were performed on the IBM-1350 supercomputing system at the National Center for High-performance Computing (NCHC) of Taiwan with 3.0 GHz of CPU speed and 4 GB of RAM per processor.

Fig. 3 shows the comparison of the simulated and measured discharge currents, which were sampled during the 5th cycle of the applied voltages in the simulation. Good agreement of discharge

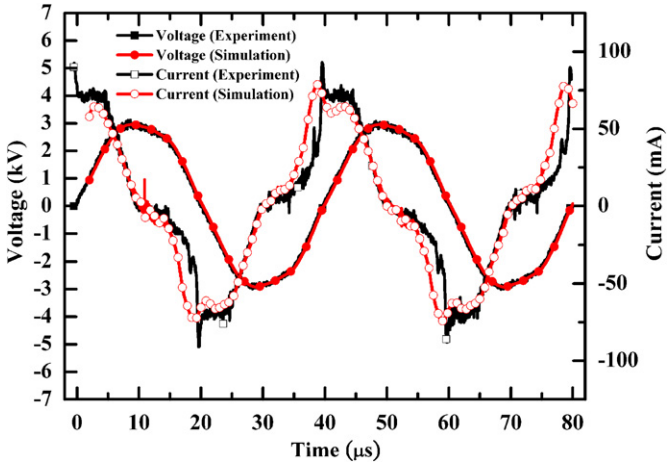

Fig. 3. Simulated and experimental discharge current for helium discharges with $V_{pp} = 6.0$ kV (peak-to-peak voltage). The frequency of power source is 25 kHz (1001×620 cells).

currents can be found between simulations and measurements. In a typical plasma simulation with ∼kHz frequency of voltage, the physical properties (such as species number density) reach the quasi-steady level within two or three cycles in the discharge region between electrodes; however, it may require 40–50 cycles for the discharge reaches the quasi-steady level in the post-discharge region near the substrate. The DBD problem investigated here is driven by nearly sinusoidal voltages with a frequency of 25 kHz. Input temporal voltages are fitted using a Fourier series expansion of 25 kHz as fundamental frequency with 15 terms in total. Two gas breakdowns during a cycle are reproduced in both phase and magnitude accurately in the simulation. Details of the other simulated discharge properties are not reported here since we only focus on discussing the parallel performance in the current study.

In the following, we present a systematic study for the parallel performance of the developed fluid modeling code for the small and large problems in turn.

4.1. Small problem case

Fig. 4 shows the averaged runtimes required for each time step for each of the different equation types solved by ASM preconditioned GMRES with respect to the number of processors for the small problem. The degree of overlapping for ASM is set to be one, and ILU and LU sub-domain solvers are employed. Table 2 summarizes the corresponding average numbers of iterations required for convergence for the equations solved. As shown from the figure and the table, the computational expense associated with each component of the plasma simulation can be sorted (in order from most expensive to least expensive) as the Poisson equation, the electron energy density equation, the electron continuity equation and the neutral-species continuity equation. Despite its mathematical simplicity, solving the Poisson equation requires most of the computational resources because of the general difficulties associated with the elliptic PDEs with discontinuous jump coefficients. Although both the electron energy density equation and the electron continuity equation are both convection–diffusion–reaction types of PDEs, the preconditioned GMRES for the former is slightly more difficult to converge than for the latter. GMRES for the neutral-species continuity equation converges fastest among all the equations and thus requires the lowest computational time. Note that the constant preconditioner of neutral species are constructed only at the first time step, which further reduces the computational time. The reasons leading to the above observations are explained next.

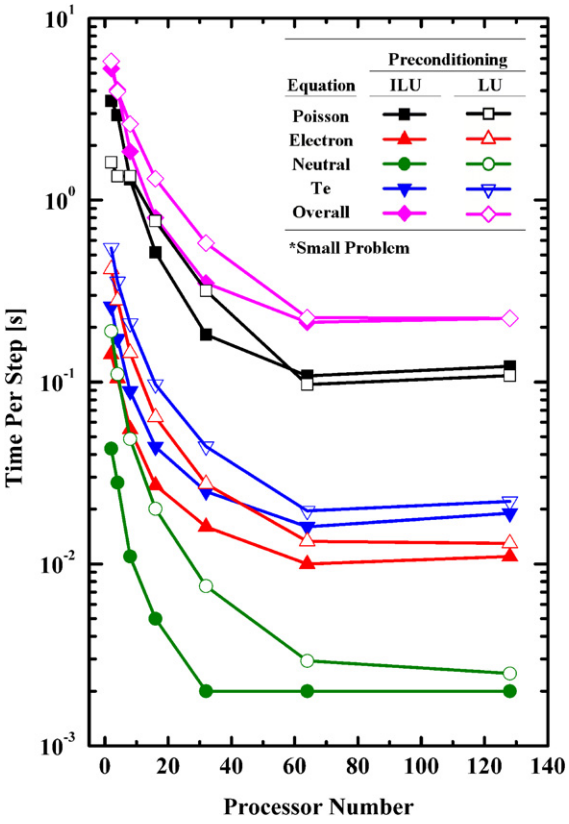

Fig. 4. Averaged time per time step of equations with sub-domain of ASM preconditioner solved by ILU and LU methods for small problem case (501×310 cells).

We have conducted some preliminary analyses, including the calculation of diagonal dominance and the eigenvalue spectrum analysis, to analyze the convergence rate of an iterative method for different type of model equations. Results (not shown in the study) show that the neutral continuity equation has the potential to converge faster than other types of equations, and the electron continuity equation and electron energy density equation have similar level of convergence speed, which are much faster than that of the Poisson equation. Although it is clearer that the convergence of an iterative method of the symmetric positive definite problems depends on the condition number of coefficient matrices, the convergence of an iterative method for solving asymmetric problems does not depend only on a single factor such as the distribution of eigenvalues, the magnitude of eigenvalues, or the diagonal dominance of matrices. It requires further investigation to fully understand the convergence behavior of GMRES for these plasma model equations.

The cases with preconditioning using the LU method required fewer number of iterations than those using the ILU method since the LU method obtains more accurate solutions in each sub-domain than the ILU method. In addition, the number of iterations increases with increasing number of processors. This is mainly because the corresponding preconditioner is divided into more sub-domains while more processors are used. Although each sub-domain is solved correctly with the LU method or the ILU method, the overall performance of the ASM preconditioning is not as good as that when fewer processors are used. This is caused by more erroneous inter-processor boundary data because of domain decomposition. In other words, the domain decomposition of the preconditioner induces slower convergence for solving the linear algebraic systems when using an iterative method as more processors are used. Resulting performance characteristics of the Poisson equation show a dramatic increase of number of iterations as more

Table 2
Averaged iteration number per time step of equations with sub-domain of ASM preconditioner with one-level overlapping solved by ILU and LU methods for small problem case (501 × 310 cells).

Poisson			Electron		
CPU	ILU	LU	CPU	ILU	LU
2	167.4	25.9	**2**	4.0	2.6
4	204.9	36.9	**4**	4.0	2.6
8	216.9	85.5	**8**	4.9	4.0
16	233.0	117.2	**16**	5.1	4.8
32	229.2	131.9	**32**	6.0	5.0
64	237.8	136.7	**64**	6.9	6.0
128	253.1	184.8	**128**	9.7	9.6
Neutral			Te		
CPU	ILU	LU	CPU	ILU	LU
2	2.0	2.0	**2**	8.5	4.4
4	2.0	2.0	**4**	8.5	4.3
8	2.0	2.0	**8**	10.0	7.9
16	2.0	2.0	**16**	11.4	9.2
32	2.0	2.0	**32**	13.5	11.7
64	2.0	2.0	**64**	15.6	13.6
128	2.1	2.4	**128**	24.6	24.5

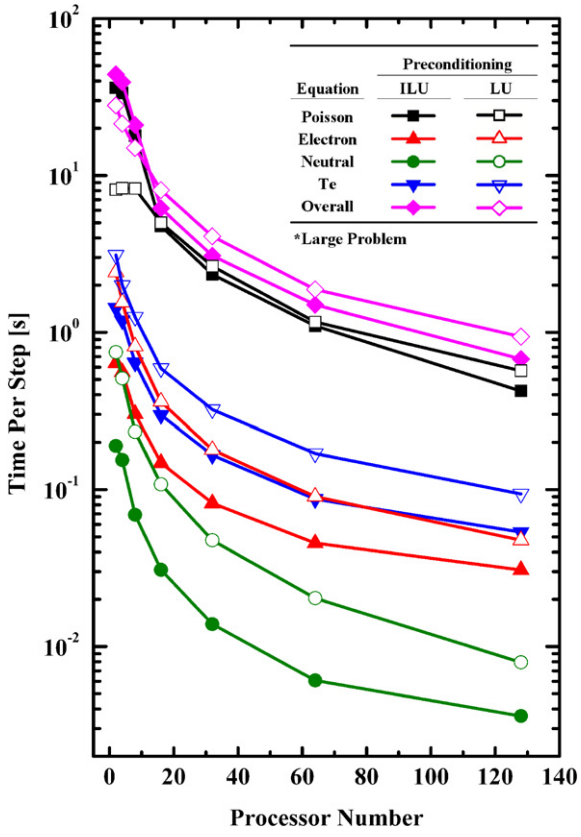

Fig. 5. Averaged time per time step of equations with sub-domain of ASM preconditioner solved by ILU and LU methods for large problem case (1001 × 620 cells).

processors are used, while only slight increases in number of iterations are experienced for the solution to the remaining equations, with increasing numbers of processors. The overall performance of the cases using the ILU and the LU methods are comparable and there is no advantage in time saving when more than 64 processors are employed for the small problem presented.

4.2. Large problem case

Fig. 5 shows the timing results for the large problem, and Table 3 summarizes the corresponding numbers of iterations for the same cases. Similar to the previous small problem, solving the Poisson equation takes most of the computational time. The solution of the electron energy density equation converges slower than the electron continuity equation, while the neutral-species continuity equation contributes the least computational time.

The number of iterations required for the convergence of the large problem is higher than that of the small problem, especially for the Poisson equation as shown in Table 3, because a much larger matrix system results from the former than the latter. The number of iterations of the large problem also increases while more processors are used. The overall parallel performance improves because of the increased grain size.

In brief summary, the computational time using the ILU method is faster than that using the LU method, mainly because the LU factorization is more costly in each sub-domain for the large problem.

4.3. Effect of ASM overlapping

To further explore the performance of GMRES with ASM preconditioner, we study the impact on the degrees of overlapping for ASM preconditioner for the cases of 32 processors or more. The results are summarized in Tables 4 and 5 for the ASM sub-domain problem solved by ILU and LU methods, respectively, for the large problem. The results for small problem cases are not shown here due to the similar trends of the results for the large problem cases. In general, the numbers of iterations of the cases with more overlapping degrees for both the ILU and the LU methods are generally reduced to some extent for all the equations with the same number of processors as shown in Tables 4(a) and 5(a) respectively. The more overlapping degree is used, the faster the convergence is because of increasing rate of information propagation among various sub-domains. However, this increase of convergence pays off with an increasing amount of data communication, which may in turn trade off the benefit of increasing rate of convergence. The overall runtime of test cases with different degrees of overlapping are summarized in Tables 4(b) and 5(b). As a result, for most of the test cases, the increased degree of overlapping does not reduce the runtime for all types of equations. It concludes that increasing overlapping degrees of ASM preconditioning has no advantage in terms of runtime for discharge simulations, at least, in the current test conditions.

4.4. Parallel performance

Figs. 6 and 7 show the speedup of the one-level overlapping ASM preconditioning solved by the ILU and LU methods for the

Table 3
Averaged iteration number per time step of equations with sub-domain of ASM preconditioner with one-level overlapping solved by ILU and LU methods for large problem case (1001 × 620 cells).

Poisson			Electron		
CPU	ILU	LU	CPU	ILU	LU
2	311.0	32.9	**2**	4.9	3.0
4	339.7	48.7	**4**	4.9	3.0
8	400.8	94.2	**8**	5.9	4.7
16	292.6	137.9	**16**	6.4	5.3
32	349.0	169.5	**32**	8.4	7.1
64	402.4	172.2	**64**	9.1	8.0
128	373.4	213.5	**128**	11.5	10.0
Neutral			Te		
CPU	ILU	LU	CPU	ILU	LU
2	2.2	2.0	**2**	12.3	5.6
4	2.1	2.0	**4**	12.3	5.6
8	2.1	2.0	**8**	13.7	10.0
16	2.1	2.0	**16**	15.0	11.5
32	2.1	2.0	**32**	19.5	16.1
64	2.1	2.0	**64**	22.4	19.6
128	2.1	2.0	**128**	29.2	26.6

Table 4
The parallel performance of different degree of overlapping ASM preconditioner with sub-domain solved by ILU method on the large problem (1001 × 620 cells). (a) Averaged iteration number per time step of equations. (b) The runtime per time step. "OL 1", "OL 2", and "OL 3" represent one-, two-, and three-level overlapping, respectively.

(a)

Poisson				Electron			
CPU	OL1	OL2	OL3	CPU	OL1	OL2	OL3
32	349.0	384.7	379.6	**32**	8.4	7.1	6.6
64	402.4	360.9	335.3	**64**	9.1	8.0	7.0
128	373.4	362.8	349.7	**128**	11.5	9.6	8.7
Neutral				Te			
CPU	OL1	OL2	OL3	CPU	OL1	OL2	OL3
32	2.1	2.2	2.2	**32**	19.5	16.7	15.3
64	2.1	2.2	2.2	**64**	22.4	19.0	17.1
128	2.1	2.2	2.6	**128**	29.2	24.0	20.4

(b)

CPU	OL1	OL2	OL3
32	3.08	3.43	3.67
64	1.50	1.60	1.59
128	0.68	0.77	0.88

Table 5
The parallel performance of different degree of overlapping ASM preconditioner with sub-domain solved by LU method on the large problem (1001 × 620 cells). (a) Averaged iteration number per time step of equations. (b) The runtime per time step. "OL 1", "OL 2", and "OL 3" represent one-, two-, and three-level overlapping, respectively.

(a)

Poisson				Electron			
CPU	OL1	OL2	OL3	CPU	OL1	OL2	OL3
32	169.5	149.0	132.5	**32**	7.1	5.6	4.9
64	172.2	144.5	128.6	**64**	8.0	6.1	5.1
128	213.5	205.0	155.2	**128**	10.0	8.0	7.9
Neutral				Te			
CPU	OL1	OL2	OL3	CPU	OL1	OL2	OL3
32	2.0	2.0	2.0	**32**	16.1	12.4	10.3
64	2.0	2.0	2.0	**64**	19.6	15.0	12.4
128	2.0	2.0	2.6	**128**	26.6	20.6	18.6

(b)

CPU	OL1	OL2	OL3
32	4.09	4.02	4.06
64	1.88	1.97	2.09
128	0.94	1.26	1.41

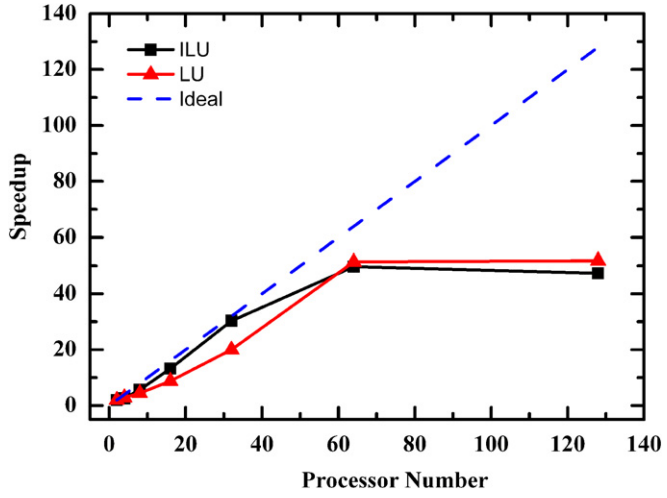

Fig. 6. Speedup of cases with sub-domain of ASM preconditioner solved by ILU and LU methods for small problem case (501 × 310 cells).

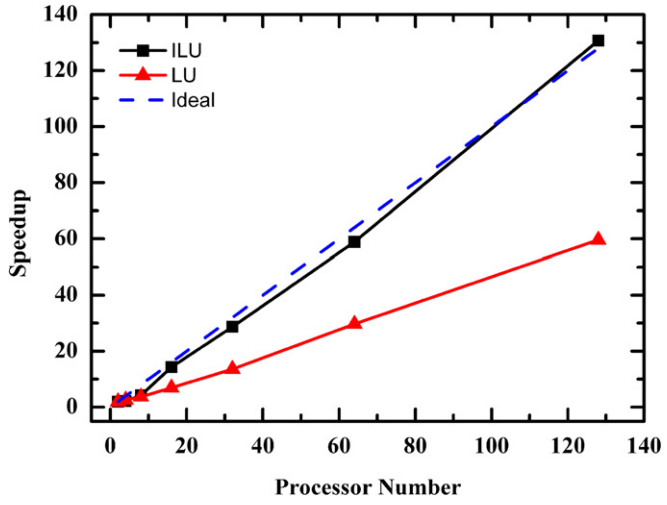

Fig. 7. Speedup of cases with sub-domain of ASM preconditioner solved by ILU and LU methods for large problem case (1001 × 620 cells).

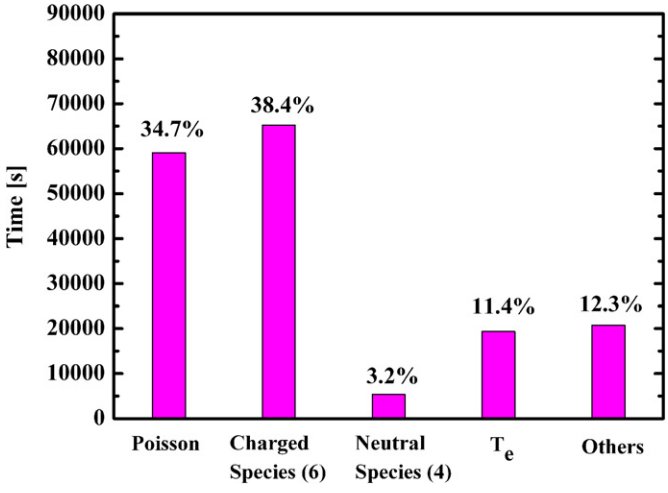

Fig. 8. Proportions of time consumed by different types of equations of the demonstration case (1001 × 620 cells). Note that "others" includes evaluation of transport properties and data communication among processors.

small and large problems respectively. In these two figures, we take timings using two processors as the baseline for calculating the speedup.

Results show the speedup achieves up to 50 times as 64 processors are used for the small problem (Fig. 6), and levels off afterwards with either ILU or LU solver for the sub-domain of ASM preconditioner for the small problem. For the large problem, the ASM with ILU sub-domain solver for GMRES is able to obtain up to 130 times speed up using 128 processors, which is better than that with LU method (Fig. 7). Meanwhile, the absolute runtime of cases for ASM-ILU preconditioner are also less than those of cases with ASM-LU preconditioner, e.g., 0.68 second/step vs. 0.94 second/step for the case of 128 processors. The above observations show that the developed parallel fluid modeling code can be very useful in practice for simulating low-temperature plasma discharges in greatly reducing the computational time up to two orders of magnitude with a limited number of processors (e.g., 60–128).

4.5. Demonstration of the results of the large problem in 5th cycle

To demonstrate the capacity of the developed parallel fluid modeling code for predicting complex plasma phenomena, we have conducted a complete simulation of the large problem case. The helium discharge (with nitrogen impurity of 100 ppm) in this simulation is driven by a nearly sinusoidal voltage with 3.0 kV in amplitude and 25 kHz in frequency under atmospheric-pressure condition. As mentioned earlier, there are six charged species, four neutral species, and 43 reaction channels involved in this demonstration. The complete simulation run for 5 cycles took about 48 hours by using 128 processors with the time size of 5×10^{-10} second. It is also noted that the convergence and computational time required for solving equations are varied in different phases of one cycle; therefore, the runtime of this demonstration case may not be consistent with the runtime given in Fig. 5.

The breakdown of computational time consumed by the solution of different type of equations is shown in Fig. 8. It takes 87.7% of the total time for solving all governing equations and 12.3% of that for data communication and other calculation such as evaluation of transport properties and rate constants. Fig. 9 shows several typical cycle-averaged spatial distributions of plasma properties such as the potential, the electron temperature (T_e), the number density of electron, and the number density of N_4^+, which is the dominant positive ion, near the exit of the parallel-plate DBD at the 5th cycle. The average plasma potential is calculated to be 140 volts in the bulk, which is higher than those of sheaths which are positively charged. The distribution of T_e shows that the average T_e is approximately 4–5 eV in both bulk and sheaths, which is generally high as compared to those driven by MHz-level atmospheric-pressure discharges. The electron number density is sustained at about 10^{16} m^{-3} and N_4^+ is sustained at the same order of magnitude (10^{16}). The detailed plasma physics and chemistry of this problem will be reported elsewhere in the near future.

5. Conclusion

In the current study, we have presented a parallel two-dimensional fluid modeling solver using the cell-centered finite-volume method. The simulated discharge currents are compared with the measured currents of helium DBD. The parallel performance of the four equation types present in these plasma simulations (the Poisson equation, the electron continuity equation, the neutral-species continuity equation, and the electron energy density equation) are presented for the cases solved by the parallel GMRES with parallel ASM preconditioning. Two problem sizes (small and large) are studied using the ILU and LU methods for

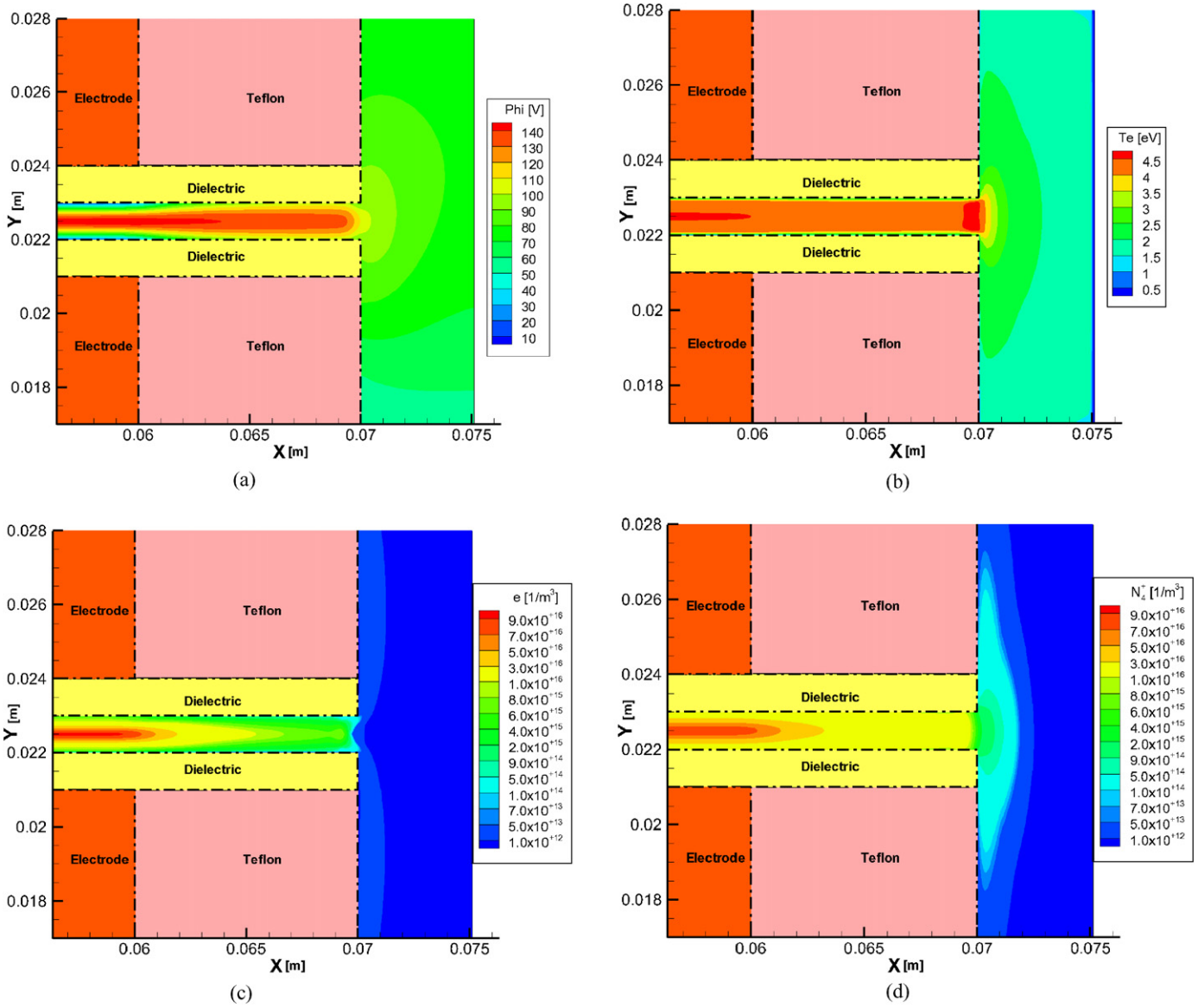

Fig. 9. Cycle-averaged spatial distributions of (a) potential, (b) electron temperature, (c) number density of electron, (d) number density of N_4^+ in the 5th cycle.

sub-domain solution in parallel ASM preconditioning with different overlapping degrees. The number of cells (i.e., problem size) required for simulation strongly depends on the geometrical size, numbers of charged and neutral species, and rarefaction of the problem. It is noted that the large problem presented in the current study is representative of actual applications of two-dimensional discharge simulation. The plasma chemistry investigated (helium with nitrogen impurity) includes 10 species and 43 reaction channels. A practical atmospheric-pressure discharge driven by a 25 kHz power source is also presented to demonstrate the capability of the developed parallel code.

In summary, the computational expense associated with each component of the plasma simulation can be sorted (in order from most expensive to least expensive) as the Poisson equation, the electron energy density equation, the electron continuity equation and the neutral-species continuity equation. Increasing degrees of ASM overlapping does not show any significant effect on the runtime though the number of iterations is generally reduced. Results show that an increase in performance of up to 50 times using 64 processors with either ILU or LU method is applied for the subdomain of ASM preconditioner for the small problem, while an increase in performance of up to 130 times with ILU method and 60 times with LU method using 128 processors for the large problem are obtained. The results demonstrate that parallel computing using domain decomposition with MPI for a two-dimensional fluid modeling code can be very useful in practice in greatly reducing the computational time up to two to three orders of magnitude with a limited number of processors.

Acknowledgements

The authors would like to express their sincere thanks to the computing resources provided by National Central University and National Center for High-Performance Computing of Taiwan through NSC-99-2221-E-009-056-MY2 and 97-NSPO(B)-SE-FA04-02.

References

[1] J. van Dijk, G.M.W. Kroesen, A. Bogaerts, J. Phys. D: Appl. Phys. 42 (2009) 190301.
[2] M.A. Lieberman, A.J. Lichtenberg, Principles of Plasma Discharges and Materials Processing, 1st ed., Wiley-Interscience, New York, 1994.

[3] M.J. Kushner, J. Phys. D: Appl. Phys. 42 (2009) 194013.
[4] G.K. Grubert, M.M. Becker, D. Loffhagen, Phys. Rev. E 80 (2009) 036405.
[5] J. van Dijk, K. Peerenboom, M. Jimenez, D. Mihailova, J. van der Mullen, J. Phys. D: Appl. Phys. 42 (2009) 194012.
[6] C.T. Hung, M.H. Hu, J.S. Wu, F.N. Hwang, Comput. Phys. Commun. 177 (2007) 138.
[7] C.T. Hung, Y.M. Chiu, F.N. Hwang, J.S. Wu, Comput. Phys. Commun. 182 (2011) 161.
[8] P.L.G. Ventzek, T.J. Sommerer, R.J. Hoekstra, M.J. Kushner, Appl. Phys. Lett. 63 (1993) 605.
[9] G.J.M. Hagelaar, G.M.W. Kroesen, J. Comput. Phys. 159 (2000) 1.
[10] Y. Saad, M.H. Schultz, SIAM J. Sci. Stat. Comput. 7 (1986) 856.
[11] X.C. Cai, W.D. Gropp, D.E. Keyes, R.G. Melvin, D.P. Young, SIAM J. Sci. Comput. 19 (1998) 246.
[12] C.T. Hung, Y.M. Chiu, F.N. Hwang, M.H. Chiang, J.S. Wu, Y.C. Wang, Plasma Chem. Plasma Process. 31 (2011) 1.
[13] G.J.M. Hagelaar, L.C. Pitchford, Plasma Sources Sci. Technol. 14 (2005) 722.
[14] F. Massines, A. Rabehi, P. Decomps, R.B. Gadri, P. Segur, C. Mayoux, J. Appl. Phys. 83 (1998) 2950.
[15] X.H. Yuan, L.L. Raja, Appl. Phys. Lett. 81 (2002) 814.
[16] Y.B. Golubovskii, V.A. Maiorov, J. Behnke, J.F. Behnke, J. Phys. D: Appl. Phys. 36 (2003) 39.
[17] T. Martens, A. Bogaerts, W. Brok, J. van Dijk, Anal. Bioanal. Chem. 388 (2007) 1583.
[18] X.H. Yuan, L.L. Raja, IEEE Trans. Plasma Sci. 31 (2003) 495.
[19] T. Martens, A. Bogaerts, W.J.M. Brok, J. van Dijk, Appl. Phys. Lett. 92 (2008) 041504.
[20] J. Waskoenig, K. Niemi, N. Knake, L.M. Graham, S. Reuter, V. Schulz-von der Gathen, T. Gans, Plasma Sources Sci. Technol. 19 (2010) 045018.
[21] L. Mangolini, C. Anderson, J. Heberlein, U. Kortshagen, J. Phys. D: Appl. Phys. 37 (2004) 1021.
[22] H.W. Ellis, R.Y. Pai, E.W. McDaniel, At. Data Nucl. Data Tables 17 (1976) 177.
[23] L.A. Viehland, E.A. Mason, At. Data Nucl. Data Tables 60 (1995) 37.
[24] T. Martens, A. Bogaerts, W.J.M. Brok, J.J.A.M. van der Mullen, J. Anal. At. Spectrom. 22 (2007) 1033.
[25] R.J. Leveque, Finite-Volume Methods for Hyperbolic Problems, Cambridge University Press, Cambridge, 2002.
[26] D.L. Scharfetter, H.K. Gummel, IEEE Trans. Electron Devices 16 (1969) 64.
[27] S. Balay, K. Buschelman, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, H. Zhang, PETSc library, http://www.mcs.anl.gov/petsc, 2011.
[28] M.H. Chiang, K.C. Liao, I.M. Lin, C.C. Lu, H.Y. Huang, C.L. Kuo, J.S. Wu, C.C. Hsu, S.H. Chen, Plasma Chem. Plasma Process. 30 (2010) 553.
[29] Y. Saad, Iterative Methods for Sparse Linear Systems, 2nd ed., Society for Industrial and Applied Mathematics, Philadelphia, 2003, pp. 453.
[30] H.F. Mark, D.F. Othmer, C.G. Overberger, Encyclopedia of Chemical Technology, vol. 11, 3rd ed., Wiley, New York, 1978, p. 14.