计算机科学
COMPUTER SCIENCE

DSMC/PIC耦合模拟的大规模高效混合并行计算研究

汪青松, 邱昊中, 林拥真, 杨富翔, 李洁, 王正华, 徐传福

引用本文

汪青松, 邱昊中, 林拥真, 杨富翔, 李洁, 王正华, 徐传福. DSMC/PIC耦合模拟的大规模高效混合并行计算研究[J]. 计算机科学, 2023, 50(11A): 230300146-9.

WANG Qingsong, QIU Haozhong, LIN Yongzhen, YANG Fuxiang, LI Jie, WANG Zhenghua, XU Chuanfu. Large-scale Efficient Hybrid Parallel Computing for DSMC/PIC Coupled Simulation[J]. Computer Science, 2023, 50(11A): 230300146-9.

计算机科学 COMPUTER SCIENCE
http://www.jsjkx.com
DOI:10.11896/jsjkx.230300146

DSMC/PIC 耦合模拟的大规模高效混合并行计算研究

汪青松[1] 邱昊中[1] 林拥真[1] 杨富翔[2,3] 李 洁[2] 王正华[4] 徐传福[1]
1 国防科技大学计算机学院量子信息研究所兼高性能计算国家重点实验室 长沙 410000
2 国防科技大学空天科学学院 长沙 410000
3 军事交通学院 安徽 蚌埠 233000
4 国防科技大学计算机学院 长沙 410000
(wangqs@nudt.edu.cn)

摘 要 DSMC/PIC 耦合模拟是一类重要的高性能计算应用。由于粒子动态注入、迁移等操作,传统 MPI 并行 DSMC/PIC 耦合模拟通常并行通信开销较大且负载不均衡。文中针对自主研发的 DSMC/PIC 耦合模拟软件,开展了大规模高效 MPI+OpenMP 混合并行及动态负载均衡研究。首先设计了基于嵌套双重非结构网格的 MPI 并行算法,实现了集中式和分布式两种并行通信策略,支持粒子在任意并行进程间的动态迁移。然后提出了加权负载性能模型,设计了动态负载均衡算法及高效网格重映射机制,大幅提升了耦合模拟并行效率,进一步设计了 MPI+OpenMP 混合并行算法,有效降低了纯 MPI 并行计算中动态负载均衡的网格重剖分和通信开销。在北京北龙超级云 HPC 系统上,针对 10 亿粒子规模脉冲真空弧等离子体羽流开展了数千处理器核心 DSMC/PIC 耦合并行模拟,验证了并行算法和动态负载均衡的效果。

关键词:DSMC/PIC 耦合;粒子模拟;分布式和集中式;动态负载均衡;MPI+OpenMP
中图法分类号 TP391

Large-scale Efficient Hybrid Parallel Computing for DSMC/PIC Coupled Simulation

WANG Qingsong[1], QIU Haozhong[1], LIN Yongzhen[1], YANG Fuxiang[2,3], LI Jie[2], WANG Zhenghua[4] and XU Chuanfu[1]
1 Institute for Quantum Information & State Key Laboratory of High Performance Computing, National University of Defense Technology, Changsha 410000, China
2 College of Aerospace and Engineering, National University of Defense Technology, Changsha 410000, China
3 Army Military Transportation University, Bengbu, Anhui 233000, China
4 College of Computer, National University of Defense Technology, Changsha 410000, China

Abstract DSMC/PIC coupled simulation is an important class of high-performance computing applications. Due to the dynamic particle injection and migration, the pure MPI parallelization of DSMC/PIC coupled simulation usually suffers from huge communications costs and load imbalance. In this paper, we present approaches to implement large-scale and efficient MPI+OpenMP hybrid parallelization and dynamic load balancing research for a self-developed DSMC/PIC coupled simulation software. Firstly, we propose a MPI parallel algorithm based on nested dual unstructured grid with two parallel communication strategies, centralized and distributed, to support the dynamic migration of particles between any parallel processes. Then, we present a weighted load performance model, and a dynamic load balancing algorithm and an efficient grid remapping mechanism are designed and implemented, which greatly improves the parallel efficiency of coupled parallel simulation. Furthermore, we design and implement a hybrid parallel algorithm of MPI+OpenMP for coupled simulation, which effectively reduces the grid redecomposition and communication overheads of pure MPI parallelization with dynamic load balance. On the BSCC HPC system, the DSMC/PIC coupled parallel simulation of thousands of processor cores is carried out for the billion particle scale pulsed vacuum arc plasma plume, and the effect of the parallel algorithm and dynamic load balancing has been verified.

Keywords Coupled DSMC/PIC, Particle simulation, Centralized and distributed communication strategies, Dynamic load balance, MPI+OpenMP

1 引言

粒子模拟是一类关键的高性能科学和工程计算应用,在很多领域应用前景广阔。当前主流的粒子模拟方法通常耦合了直接蒙特卡罗模拟(Direct Simulation Monte Carlo,DSMC)方法[1]和粒子网格(Particle in Cell,PIC)方法[2-3]。DSMC

基金项目:国家数值风洞工程(TC228S03J);四川省科技计划(2023YFG0152)
This work was supported by the National Numerical Windtunnel Project (TC228S03J) and Sichuan Science and Technology Program (2023YFG0152).
通信作者:徐传福(xuchuanfu@nudt.edu.cn)

可以对稀薄气体流动微观层面的问题进行建模，模拟粒子之间的运动和碰撞等。利用经典的 Bird 算法[4]，DSMC 可以得到与玻尔兹曼公式一致的结果；PIC 可以跟踪大量粒子的相互作用。耦合 DSMC 和 PIC 的方法可以在微观层面模拟大量的粒子及其行为。

等离子体羽流[5-6]模拟是 DSMC/PIC 的重要应用领域之一。等离子体羽流通常包括蒸汽、等离子体、分子簇和表面碎片，蒸汽和等离子体以每秒几十公里的超高速度反喷。脉冲真空电弧产生的等离子体羽流在高稳定性等离子体器件中有重要的应用价值，是航空航天推进[7-9]、核聚变反应堆[10-11]等领域研究的关键。脉冲真空弧等离子体羽流通常在毫米范围内流动，在微秒内引起一系列的复杂热化学非平衡反应和壁相互作用。由于蒸发的粒子数量多，粒子密度大（$10^{18}/m^3$ 到 $10^{22}/m^3$），相应的等离子体羽非常稀薄，克努森数大于 0.1，因此无法采用基于纳维-斯托克斯方程的连续介质计算流体动力学（Computational Fluid Dynamics，CFD）方法进行模拟。

虽然 DSMC/PIC 耦合方法可以实现精确的粒子模拟，但相对于传统的 CFD 模拟，大规模 DSMC/PIC 模拟通常需要更多的内存和计算资源[12-13]，高效并行计算至关重要。DSMC/PIC 并行模拟中，由于每个模拟时间步的粒子动态注入、迁移等操作，传统 MPI 并行 DSMC/PIC 耦合模拟通常并行通信开销较大且负载不均衡严重（请见第 5 节）。因此，高效 DSMC/PIC 耦合并行模拟通常需要设计动态负载均衡算法，在一定次数的时间步迭代后，对网格进行重新剖分并在进程之间重新分配仿真粒子。

本文针对自主研发的 DSMC/PIC 耦合模拟软件，开展了大规模高效 MPI+OpenMP 混合并行及动态负载均衡研究。首先设计了基于嵌套双重非结构网格的耦合模拟方法及其 MPI 并行算法，可有效支持复杂模拟外形，同时便于 MPI 并行实现；MPI 并行实现了集中式和分布式两种并行通信策略，支持粒子在任意并行进程间的动态迁移。然后针对 DSMC/PIC 并行模拟设计了一个动态负载均衡器，包括并行进程负载不均衡量化评估指标、指导网格重剖分的加权负载性能模型以及基于 Kuhn-Munkres（KM）[14-15]算法的高效网格重映射机制，动态负载均衡大幅提升了 DSMC/PIC 耦合模拟的并行效率。在此基础上，为了降低纯 MPI 并行中动态负载均衡的网格重剖分、重映射开销，同时减少结点间的并行通信，设计了 MPI+OpenMP 混合并行算法，进一步提升了 DSMC/PIC 耦合并行效率。在北京北龙超级云 HPC 系统上，采用脉冲真空弧等离子体羽流典型算例进行了 10 亿粒子规模、数千核并行规模的 DSMC/PIC 耦合模拟，充分验证了并行算法和动态负载均衡的效果。

2 相关工作

近年来，国内外开展了大量 DSMC/PIC 耦合方法应用及其并行计算的相关研究。Aleph[16]是一种用于模拟低温等离子体的 DSMC/PIC 耦合求解器，它可以使用非结构化四面体网格模拟三维模型[17-18]。文献[19]使用并行 DSMC/PIC 耦合算法来模拟稀薄条件下的活性等离子体流动和化学反应。文献[20]利用 MPI 实现了用一个并行的高阶 DSMC/PIC 求解器，模拟激光驱动的等离子体羽流的 250ps 扩散过程。该求解器基于三维非结构六面体网格，采用间断有限元法对电磁场进行离散[21]。SUGAR（自适应网格加密的可扩展非结构气体动力学）软件[22]也实现了用 MPI 进行并行计算。SUGAR 可以模拟离子推进器的喷流，包括中性粒子与带电粒子之间的 MEX（动量交换）碰撞和 CEX（电荷交换）碰撞。文献[23]采用 MPI+CUDA 设计了一种基于 GPU 的 DSMC/PIC 求解器。该方法在模拟过程中使用八叉树划分计算域，并根据粒子数进行负载均衡。

上文提到 DSMC/PIC 耦合求解器并行实现主要针对电推力器产生的等离子体羽流，它们大多只关注等离子体的产生而不是扩散过程。本文中 DSMC/PIC 耦合软件的物理化学模型、粒子种类更加丰富，可同时模拟扩散、碰撞和化学反应等，因此粒子行为变化更加动态，高效并行计算需要实现粒子在任意并行进程之间的迁移以及更加精细的动态负载均衡。此外，由于 DSMC/PIC 耦合模拟的复杂性，目前大部分求解器采用纯 MPI 并行实现，通常并行计算规模为数十到数百计算核心。本文采用 MPI+OpenMP 混合并行结合高效动态负载均衡算法，有效支撑了 10 亿粒子、数千核并行规模的 DSMC/PIC 耦合模拟。

3 DSMC/PIC 耦合求解器

本文采用的自主设计的 DSMC/PIC 耦合求解器包括 DSMC 求解器和 PIC 求解器两个相对独立的部分。DSMC 求解器主要由国防科技大学开发，该求解器除了实现文献[4]中的经典算法模型，同时集成了一些自主研发的模型，模拟流场中粒子的碰撞、壁面相互作用和化学反应等过程[24-25]。PIC 求解器主要由中国工程物理研究院开发，用于模拟电场作用下带电粒子的运动。由于独立的 DSMC 和 PIC 求解器的网格单元大小分别受不同类型粒子的限制，为了实现 DSMC 与 PIC 的耦合求解，设计了基于嵌套双重非结构网格的耦合方法。如图 1 所示，DSMC 模拟采用粗网格，网格单元尺度满足分子平均自由程约束（平均自由程是粒子（例如原子、分子或光子）运动时与其他粒子的一次或多次连续碰撞而显著改变其方向或能量的平均距离）；PIC 模拟采用细网格，网格单元尺度符合德拜长度要求（德拜长度也叫德拜半径，一般指宏观电场有意义的最小长度）。具体而言，在网格生成时首先生成 DSMC 粗网格，再将每个 DSMC 粗网格单元划分为 8 个 PIC 细网格单元，所有的 PIC 细网格单元完全嵌入一个 DSMC 粗网格单元中。

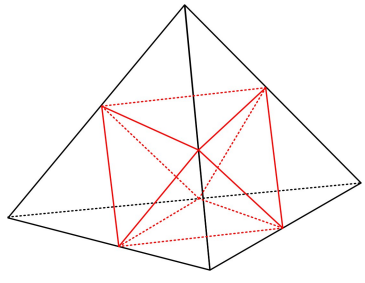
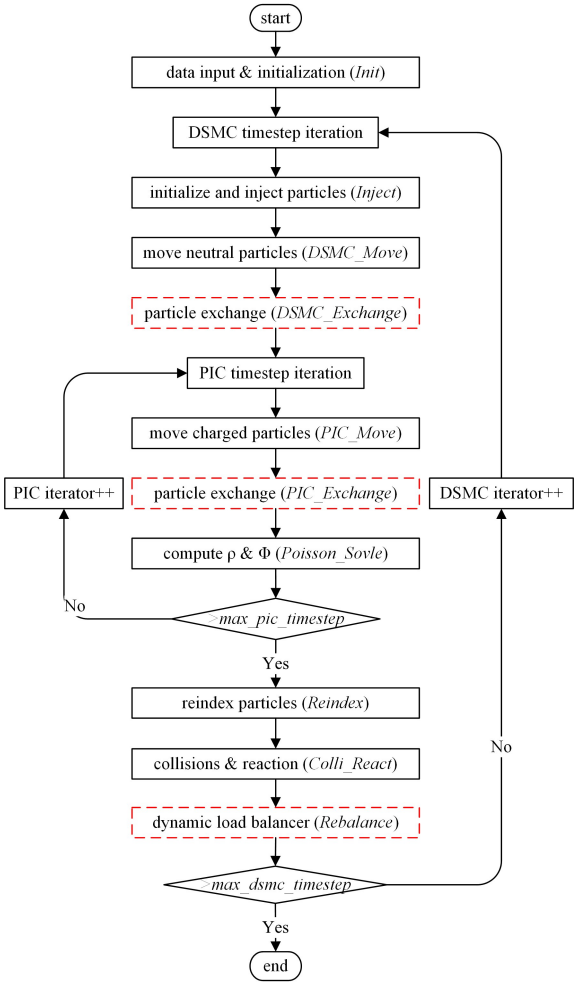


图 1 一个粗四面体 DSMC 网格单元有 8 个 PIC 细网格单元

Fig. 1 A coarse tetrahedral DSMC grid cell with 8 fine-grained grid cells for PIC

基于嵌套双重非结构，以 DSMC 求解流程为主集成 PIC 求解流程，实现了 DSMC/PIC 耦合求解。图 2 给出了耦合求解的整体工作流程。程序读入网格、模拟参数并初始化相关

数据结构后进入时间步迭代。



注：虚线矩形表示本文实现的并行通信和负载均衡部分。

图 2　DSMC/PIC 耦合求解器的整体流程（电子版为彩图）

Fig. 2　Workflow of coupled DSMC/PIC solver

DSMC 时间步主要包括：

1）粒子注入（Inject）：根据麦克斯韦分布和入口位置等条件注入并初始化模拟粒子。

2）模拟中性粒子运动（DSMC_Move）：每个 DSMC 时间步中性粒子会以恒定的速度直线移动，可以跨越不同的网格单元，甚至移动出计算域。

3）PIC 时间步迭代：一个 DSMC 时间步通常包含多个 PIC 时间步，DSMC 或 PIC 时间步的大小取决于粒子的平均自由程[26]，PIC 时间步迭代过程请参考后文。

4）粒子重编号（Reindex）：粒子位置很可能在迭代过程中发生变化，求解器以统一的方式对所有粒子重编号，同时删除移出计算域的粒子，以保证确保每个粒子都有唯一的全局索引。

5）碰撞和反应（Colli_React）：采用 NTC 方法[27]选择碰撞对，根据粒子类型和碰撞能量来判断两个粒子是否会发生化学反应，实现了各种碰撞和化学反应模型[25]。

PIC 时间步包括两个主要步骤：

6）模拟带电粒子运动（PIC_Move）：每个 PIC 时间步中，带电粒子由前一个时间步的电场驱动。

7）求解泊松方程（Poisson_Solve）：带电粒子的速度 v 和位置 r 可以通过求解等离子体物理动力学方程[25]得到：

$$\begin{cases}\dfrac{\mathrm{d}r}{\mathrm{d}t}=v\\ m\dfrac{\mathrm{d}v}{\mathrm{d}t}=q(\boldsymbol{E}+V\times B)\end{cases}\tag{1}$$

其中，m 为粒子的质量（由用户给出），q 为粒子的电荷，B 为磁场密度，$\boldsymbol{E}$ 为电场强度。本文只考虑静电场，假设不存在磁场（$B=0$）或恒定磁场（即 B 为用户给出的常数）。本文采用 Boris 方法[28]计算速度 v 的数值，然后通过向网格节点插值粒子电荷来计算电荷密度 q，跟踪带电粒子的运动。电场强度 $\boldsymbol{E}$ 通过求解静电的泊松方程来计算：

$$\nabla\boldsymbol{E}=\frac{\rho}{\epsilon_0}\tag{2}$$

其中，ρ 是一个总体积电荷密度，ϵ_0 是介质的介电常数，∇ 是拉普拉斯算子。电场强度 $\boldsymbol{E}$ 可以表示为电势的负梯度 $\varnothing$：

$$\boldsymbol{E}=-\nabla\varnothing\tag{3}$$

通过用 $-\nabla\phi$ 替换 $\boldsymbol{E}$，泊松方程转换为：

$$\nabla\cdot\boldsymbol{E}=\nabla\cdot(-\nabla\varnothing)=-\nabla^2\varnothing=\frac{\rho}{\epsilon_0}\tag{4}$$

在非结构网格上使用有限体积法[29]对式（4）进行离散求解需要构建线性系统：

$$\boldsymbol{K}\varnothing=b\tag{5}$$

其中，$\boldsymbol{K}$ 为根据网格拓扑构造的对角占优刚度矩阵，b 由边界条件和电势推导而来。通过线性方程组求解获得 $\varnothing$ 后，根据式（3）计算 $\boldsymbol{E}$。

图 2 中标黄的虚线框给出了本文实现的 MPI 并行通信（PIC_Exchange 和 DSMC_Exchange）以及动态负载均衡（Rebalance）部分。

4　MPI 并行算法设计与实现

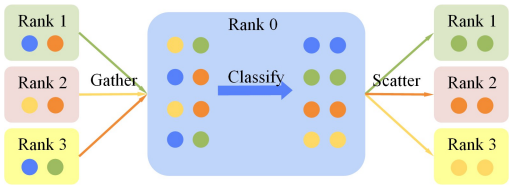
本文采用分区并行思想设计了 MPI 并行。在对原始网格进行剖分时，由于采用了基于嵌套双重非结构网格的设计，每个细网格单元属于一个粗网格单元，MPI 并行只需要对粗网格进行剖分，根据粗网格剖分结果可自动确定细网格的所属分区。实现中采用 METIS 软件[30]调用 METIS_PartGraphKway 对网格进行剖分。第一次剖分时无需指定网格单元权重数组；在本文实现的动态负载均衡过程中，通过粒子和网格单元的加权性能模型计算权重数组，需要调用 METIS 对网格进行重剖分（请见第 5 节）。

本文耦合求解及其应用场景中支持粒子长距离迁移，粒子迁移可能导致其跨越不同网格单元甚至其他并行区域，因此与传统 CFD 并行通常只涉及邻居区域通信不同，MPI 并行必须能够处理任意区域之间的通信。为此，本文提出了集中式和分布式两种并行通信策略，用于实现粒子在任意进程中的迁移，上述策略在图 2 中的 DSMC_Exchange 和 PIC_Exchange 上进行了实现。

4.1　集中式并行通信策略

图 3 给出了集中式并行通信策略示意图。首先选择 1 个进程（例如主进程，图 3 中的 rank 0）来集中管理粒子迁移的整个过程。粒子迁移过程包括 3 个阶段：收集、分类和分散。在收集阶段，每个进程将本区域的迁移粒子发送到集中进程。然后，集中进程收集所有需要迁移的粒子并根据它们的目的进程对它们进行分类，为减少通信次数，来自不同进程但目的地相同的粒子将被统一打包。最后，集中进程将打包的粒子

分散发送到相应的目的进程。



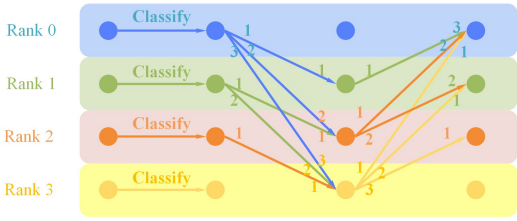
注：Rank 0 被选作集中进程，不同颜色的线表示不同过程之间的通信，不同颜色的圆圈代表要移动到不同目的地的粒子。

图 3 4 个进程使用集中式通信策略的简单说明(电子版为彩图)

Fig. 3 Illustration of 4 processes using centralized communication strategy

4.2 分布式并行通信策略

图 4 给出了分布式并行通信策略示意图。每个进程根据迁移粒子的目的进程对迁移粒子进行分类和打包，然后执行两轮同步的 MPI_send/MPI_recv 通信操作交换粒子。第一轮通信中，每个进程首先从进程号比自己小的进程中接收迁移粒子，然后将迁移粒子发送给进程号比自己大的进程。第二轮通信中，每个进程将首先从进程号大于自身的进程中接收迁移粒子，然后将迁移粒子发送给进程号比自己小的进程。为了在并行通信中避免死锁，需要注意的是：发送迁移粒子时，进程应首先发送给进程号较小的进程；接收迁移粒子时，应该首先从进程号较大的进程中接收。



注：不同颜色的线表示不同进程之间的通信，不同颜色的圆圈代表要移动到不同目的地的粒子，线上的数字表示通信(发送或接收)命令。

图 4 4 个进程使用分布式通信策略的简单说明(电子版为彩图)

Fig. 4 Illustration of 4 processes using distributed communication strategy

这里对上述两种 MPI 并行通信策略效率进行简单的理论分析。假设有 M 个迁移粒子在 N 个并行进程中迁移，每个进程都有粒子穿出和穿入。这种情况下，采用集中式通信策略时，通信次数为 $2N$，通信数据量大小与 $2M$ 成正比。采用分布式通信策略时，通信次数为 $N(N-1)$ 左右，通信数据量近似与 M 成正比。可以看出，集中式通信策略的通信次数相对较少，但通信数据量更大，此外集中式通信可能存在单点通信瓶颈；分布式通信策略的通信次数较多，但通信数据量更少。根据上述分析，上述两种策略在不同的 HPC 系统和模拟配置下可能表现出不同性能，后续在实际测试中将进一步进行对比分析。

4.3 PIC 中泊松方程的并行求解

组装和求解泊松方程是 PIC 计算中最耗时的过程。各进程内部网格节点的电荷密度可以根据其局部区域的带电粒子计算，对于边界网格节点，其电荷密度应为所有相邻进程的电荷密度之和，因此需要首先对边界节点的电荷密度进行归约求和。在得到电荷密度后，需要求解泊松方程获得网格节点上的电势。直接求解非线性泊松方程是一个复杂而耗时的过程，本文采用迭代法来实现对泊松方程的快速求解。首先将泊松方程转化为一个线性系统(见式(5))，其中 K 是一个全局稀疏矩阵；然后利用第三方线性系统求解库进行迭代求解。具体实现中，本文采用了 PETSc[31] 和 STRUMPACK[32] 两个求解库，前者支持纯 MPI 并行，后者支持 MPI+OpenMP 混合并行。为了减少内存占用，稀疏矩阵 $\boldsymbol{K}$ 采用 CSR 格式存储。

5 动态负载均衡

本文提出的 DSMC/PIC 模拟支持粒子在任意进程之间动态迁移，这可能导致并行进程间严重的负载不均衡，降低整体并行性能。此外，本文针对的脉冲真空弧等离子体羽流模拟等算例在每个时间步需要从入口注入新的粒子，粒子在网格单元间的初始分布极不均匀。图 5 给出了在没有动态负载均衡情况下，采用自主 DSMC/PIC 耦合软件运行 24 个并行进程模拟 200 步迭代时的粒子分布和计算通信时间分布。可以看出，粒子分布极不均衡，最终表现为各进程计算通信时间差别较大，一些进程通信时间超过总时间的 50%。DSMC/PIC 耦合模拟的负载不均衡在大规模非定常模拟时更加严重，因此需要设计高效的动态负载均衡方法。

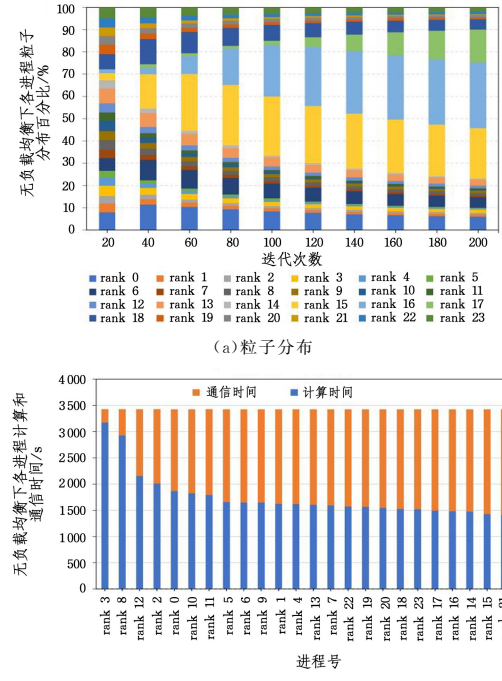


图 5 24 个并行进程模拟 200 步迭代无动态负载均衡时的粒子分布和计算通信时间分布

Fig. 5 Distribution of particles as well as computation and communication times among 24 MPI ranks for 200 steps of coupled DSMC/PIC simulations

本文设计的 DSMC/PIC 耦合并行模拟的动态负载均衡器的总体流程如算法 1 所示。模拟中每隔若干迭代步检查各进程的负载不均衡指示器，当该指标超过设定阈值时，基于加权负载模型对原始网格进行重剖分并重新将其映射到并行进程，从而实现动态负载均衡。

算法 1 动态负载均衡算法

Input: cellnum, procsnum, T, Treshold, OriginalMapping

Output: FinalMapping

1. compute lii according to(6)
2. iterator←iterator + 1
3. if iterator< T || lii<Threshold then

```
4.      return
5. end if
6. for i=0→cellnum do
7.      compute wlm_i according to(7)
8.      wlm←wlm_i
9. end for
10. NewPartition ← METIS _ PartGraphKway ( cellnum, procsnum,
    wlm)
11. FinalMapping←Kuhn_Munkras(OriginMapping,NewPartition)
12. returnFinalMapping
```

这里负载不平衡指示器(load imbalance indicator，lii)根据不同并行进程的执行时间最大值和最小值来计算，计算式如下：

$$lii=\frac{Time_{max\_total}-Time_{max\_pm}-Time_{max\_poi}}{Time_{min\_total}-Time_{min\_pm}-Time_{min\_poi}} \tag{6}$$

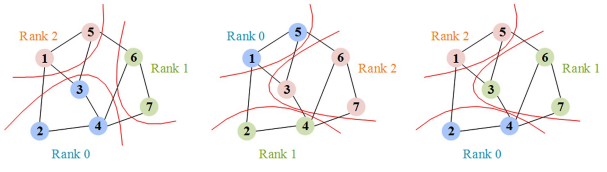
其中，下标“max”和“min”分别表示各进程中的最大和最小执行时间。$Time_{max\_total}/Time_{min\_total}$ 是总的执行时间，$Time_{max\_pm}/Time_{min\_pm}$ 和 $Time_{max\_poi}/Time_{min\_poi}$ 分别为粒子迁移(DSMC_Exchange 和 PIC _Exchange)和泊松方程求解(Poisson_Solve)时间(两者时间基本恒定，因此无需考虑)。模拟中设置每 T 次 DSMC 迭代检查 lii，如果 lii 大于阈值 Threshold，则执行动态负载均衡操作。

动态负载均衡需要重新剖分网格。求解器初始运行时网格剖分仅考虑了网格单元数量，没有提供权重数组给 METIS 工具，重剖分需要综合考虑 DSMC/PIC 耦合模拟中的各类粒子、网格单元等对负载的影响。例如，在耦合求解中，DSMC 只模拟中性粒子，PIC 只处理带电粒子，而一个 DSMC 时间步可以包含多个 PIC 时间步，因此中性粒子和带电粒子应赋予不同权重。除了粒子，还有一些计算(如图 2 中的 Colli_React 和 Poisson_Solve)是在网格单元上进行的，也应该考虑网格单元权重。本文设计了一个加权负载模型(weighted load model，wlm)，对于第 i 个网格单元的 wlm_i 定义为：

$$wlm_i=N_i+RC_i+W_{\text{cell}} \tag{7}$$

其中，N_i 和 RC_i 分别为第 i 个网格单元中的中性粒子和带电粒子个数，W_{cell} 为网格单元的权值，以中性粒子的权值为基线。根据式(7)可以计算每个网格单元格的组合权重，提供给 ETIS_PartGraphKway。

网格重剖分后需要重新映射到并行进程继续模拟迭代，相应的粒子也需要根据重剖分结果进行重新迁移分布。因此随机重映射网格剖分可能会产生较大的额外开销，影响动态负载均衡器的效果，应根据原始的映射优化网格重剖分后的映射，尽量减少粒子的迁移。本文将网格重映射问题转化为二分图的最大权值匹配问题，使用经典 KM 算法获得优化映射。不同网格映射策略实例如图 6 所示。



(a)Original　(b)Random　(c)Optimized

注：(a)为原始网格分解和映射，其中圆表示网格单元格；(b)是重新分解后可能的随机网格重映射；(c)是本文方法给出的优化后的重映射。

图 6　不同网格映射策略实例

Fig. 6　Example of different grid mapping strategies

如图 6(c)所示，此时只需要将单元 3 中的粒子从图 6(a)中的 rank 0 移动到图 6(c)中的 rank 1。而图 6(b)的随机映射需要在多个 MPI 进程中移动所有的粒子，因此粒子移动开销更大。

6　MPI＋OpenMP 混合并行

由于粒子动态迁移，纯 MPI 并行 DSMC/PIC 耦合模拟通常并行通信较大，动态负载均衡尽管提升了并行性能，但也引入了额外的通信开销。随着当前计算结点内处理器核数的增加，仅仅依赖纯 MPI 并行开发应用并行性已不切实际。本文在纯 MPI 并行 DSMC/PIC 耦合模拟基础上，设计了 MPI＋OpenMP 混合并行开发多层次并行性，其中 MPI 用于结点间的并行通信，OpenMP 用于开发共享存储线程级并行。OpenMP 并行不仅减少了 MPI 并行通信开销，也在同样并行规模下减少了 MPI 进程规模，一定程度地缓解了进程间的负载不均衡。

在整个 DSMC/PIC 耦合模拟计算过程中，简单的计算过程可以通过直接添加 OpenMP 编译指导语句实现共享存储线程并行，这里重点介绍如何高效实现共享存储线程针对某些全局数据结构的并发操作及相关的数据冲突。例如，模拟中粒子通常存储在全局链表中，粒子状态需要不断动态更新(例如删除、添加等)；在非结构网格中有些物理量(例如电荷密度)存储在网格结点上，邻居网格单元进行并发更新时可能存在数据冲突。上述情况主要出现在粒子注入、运动等过程中，为了解决上述问题，本文设计了一种复制 & 规约(Replication&Reduction，R&R)方法。算法 2 和算法 3 分别针对带电粒子注入和运动，给出了基于复制 & 归约方法实现的 OpenMP 并行。这里首先修改了存储带电粒子的数据结构，每个线程都存储一部分粒子数据的副本，从根本上消除多线程并发更新全局数据结构导致的数据冲突。在粒子注入时按照均分的方式在每个数据结构副本中分别追加粒子数据；在模拟粒子运动之前，对网格节点电荷密度这个共享变量创建每个线程私有的副本，并在函数返回前将私有副本值归约到最终的共享变量，从而避免网格顶点计算电荷密度和删除粒子时的数据冲突，各共享线程根据自身线程号来确定其操作的数据对象。

算法 2　基于复制 & 规约方法的粒子注入 OpenMP 并行

```
Input:inlets
Output:particles[]
1. n_threads ← omp_get_num_threads()
2. particles ← new double[n_threads]
3. #pragma omp parallel for
4. for i=0 → inject_num do
5.      thrade_num ← omp_get_thread_num()
6.      计算其他参数
7.      从入口网格编号获取粒子所注入的网格
8.      particles[thread_num]. push_back(part)
9. end for
10. return particles
```

算法 3　基于复制 & 规约方法的粒子运动 OpenMP 并行

```
Input:particles[n_threads],n_nodes
Output:dens[]
1. 创建临时二维数组 den_t[n_threads][ n_nodes]
2. #pragma omp parallel
3. while iterator ! = particles[my_thread_rank]. end() do
```

```
4.      计算其他参数
5.      if is_delete then
6.          particles[my_thread_rank]. erase( * iterator)
7.          continue
8.      end if
9.      if is_find_cell then
10.         den_t[my_thread_rank][i_node] += ( * iterator). den
11.         iterator ←iterator + 1
12.         continue
13.     end if
14. end while
16. for i=0 →n_nodes do
17.     dens[i]← sum(den_t[j][i])(j=0 →n_threads)
18. end for
19. return dens
```

此外，在纯 MPI 并行版本中，本文最初采用 PETSc 库实现泊松方程的全局并行求解。由于 PETSc 不支持 MPI+OpenMP 混合并行求解，我们在混合并行版本中采用了美国劳伦斯伯克利国家实验室开发的 STRUMPACK 库。

7 实验结果

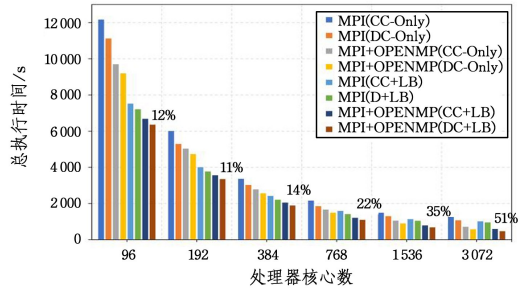
7.1 实验设置

本文测试平台为北京北龙超级云计算(BSCC)有限公司基于 x86 架构的超级计算机，目前我们只在 CPU 主机上运行求解器。每个 BSCC 计算节点包含 2 个 48 核 2.3 GHz Intel Xeon Platinum 9242 处理器，共享 384 GB 内存。计算节点间通过 InfiniBand 连接，点对点带宽为 100 Gbps。本文 DSMC/PIC 耦合求解器采用 C++语言实现，采用 GCC v7.5 以及 O3 优化选项编译，采用 MPICH v3.4.2 实现 MPI 并行通信，采用 METIS v5.1.0 进行剖分，采用 PETSc v3.15.0 和 STRUMPACK v6.3.1 的 KSP 求解器实现泊松方程并行求解。

本文中的测试算例模拟了脉冲真空电弧在三维圆柱喷管内等离子体羽流的扩散、碰撞和反应过程。测试算例包含 2 242 948 个细四面体单元，用于 PIC 模拟(对应 268 479 个粗四面体网格单元，用于 DSMC 模拟)。算例主要关注 H 粒子离解和 H^+ 粒子重组，粒子速度为 10 000 m/s，壁温为 300 K。根据 DSMC/PIC 模型的物理约束，H 和 $H+$ 的粒子密度分别设置为 1.4×10^{20} 和 1.25×10^{12}。仿真粒子数为：数据集 1 中 DSMC 处理 10^{10} 个 H 模拟粒子，PIC 处理 10^9 个 H+模拟粒子；数据集 2 的粒子规模缩小为原来的 1/10。我们根据多次实际测试选择合适的动态负载均衡参数：*Threshold* 设置为 2.0，R 设置为 2，T 设置为 20，W_{cell} 设置为 1。DSMC 和 PIC 的时间步长分别设置为 4.525×10^{-7} s 和 2.2625×10^{-7} s(每个 DSMC 时间步包含 2 个 PIC 时间步)，性能数据通过模拟运行 100 个 DSMC 时间步获得(每次模拟运行 5 次，取性能的平均值)。

7.2 实验结果分析

图 7 和表 1 分别给出了 DSMC/PIC 采用不同通信方式实现的纯 MPI 并行、MPI+OpenMP 混合并行的整体性能以及动态负载均衡效果。这里测试采用了数据集 1，以单个计算结点(96 个处理器核)作为性能基准，扩展到 32 个计算结点(3072 个处理器核)，混合并行时每个结点有 4 个 MPI 进程，每个进程启动 24 个 OpenMP 线程。可以看出，所有 8 个版本都表现了较好的并行可扩展性，加速比为 7.4～15.8。对于纯 MPI 并行，分布式通信相对于集中式通信 MPI 并行性能提升多达 18%；动态负载均衡对集中式通信 MPI 并行性能提升多达 38%，对分布式通信 MPI 并行性能提升多达 35%。同时，无论采用集中式通信还是分布式通信，MPI+OpenMP 混合并行性能都优于纯 MPI 并行性能：集中式通信时混合并行提升了 11%～44%，分布式通信时混合并行提升了 11%～51%。而动态负载均衡对混合并行性能也有提升，集中式通信时和分布式通信时的最大提升均达到 31%。



注：条形图上的百分比表示 DC+LB 下 MPI+OpenMP 混合并行相对于纯 MPI 的性能提升；CC 表示集中式通信，DC 表示分布式通信，LB 表示本文实现的动态负载均衡。

图 7 不同实现的总执行时间

Fig. 7 Total execution times of different implementations

表 1 可扩展性测试的总执行时间

Table 1 Total time for scalability tests

(单位：s)

	处理器核心数						加速比
	96	192	384	768	1536	3072	3072/96
MPI(CC-Only)	12168.9	6011.5	3370.2	2168.4	1492.7	1277.0	9.5
MPI(DC-Only)	11126.2	5294.5	3030.5	1858.9	1303.1	1075.8	10.3
MPI+OPENMP(CC-Only)	9694.7	5039.7	2786.3	1669.1	1056.4	714.7	13.6
MPI+OPENMP(DC-Only)	9197.1	4733.5	2566.0	1496.8	908.9	582.6	15.8
MPI(CC+LB)	7519.8	4006.9	2422.3	1594.4	1137.9	1020.4	7.4
MPI(DC+LB)	7209.5	3778.5	2211.4	1417.1	1049.4	959.0	7.5
MPI+OPENMP(CC+LB)	6680.0	3564.9	2054.1	1211.4	792.1	591.7	11.3
MPI+OPENMP(DC+LB)	6358.2	3350.4	1895.3	1099.3	685.5	473.1	13.4

由于在本文测试的 HPC 平台上，分布式并行通信性能更好，后续测试将主要采用分布式通信策略。图 8 给出了两种粒子规模下动态负载均衡对混合并行性能的影响。可以看出，动态负载均衡器对混合并行效果明显，尤其是模拟粒子规模较大、使用进程数较少的情况，可获得最多达 31%的性能提升，这是因为使用较少的进程时，我们并行模拟使用的

测试用例中的负载不均衡现象更为严重，在这种情况下，我们的动态负载均衡策略对重新平衡模拟负载起到了较好的效果，从而提高了整体性能和可扩展性。由于负载不均衡主要影响 DSMC_Move 和 PIC_Move 的性能，表 2 列出了动态负载均衡对这两个过程的影响。可以看出，在使用动态负载均衡时 DSMC_Move/PIC_Move 总的执行时间缩短到原来的 1/3，甚至 1/4，充分表明了动态负载均衡的有效性。

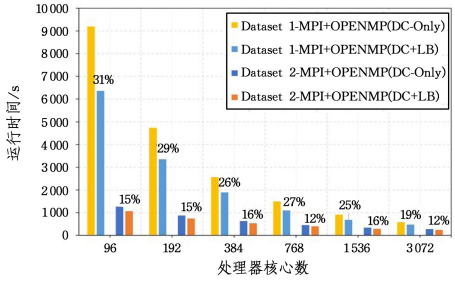


图 8　分布式通信策略下，负载均衡器对总执行时间的影响

Fig. 8　Impact of balancer on total execution time with distributed communication strategy

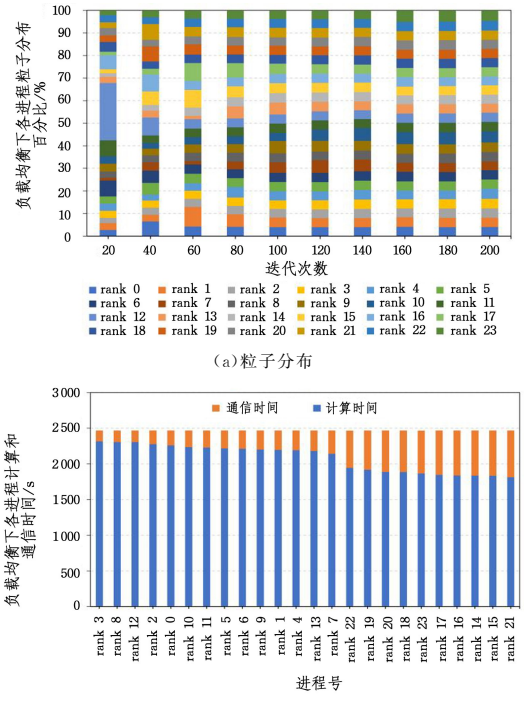
表 2　粒子迁移的总时间

Table 2　Total times for particle migration

（单位：s）

	处理器核心数					
	3072	96	192	384	768	1536
Dataset 1-MPI+OPENMP(DC-Only)	2706.8	1000.0	540.1	264.9	131.5	60.0
Dataset 1-MPI+OPENMP(DC+LB)	387.0	192.6	112.3	63.6	40.8	22.1
Dataset 2-MPI+OPENMP(DC-Only)	98.0	75.9	47.3	23.6	14.0	8.2
Dataset 2-MPI+OPENMP(DC+LB)	38.5	21.4	12.9	8.0	3.6	2.5

与图 5 对应，图 9 给出了实现动态负载均衡后各进程的粒子及计算通信时间分布。与图 5 相比，可以看出此时粒子和计算通信时间的分布明显更加均衡。



(a)粒子分布

(b)计算通信时间分布

图 9　使用动态负载均衡后 24 个并行进程模拟 200 步迭代的粒子分布和计算通信时间分布

Fig. 9　Distribution of particles as well as computation and communication times among 24 MPI ranks for 200 steps of coupled DSMC/PIC simulations

表 3 进一步对比了 MPI＋OpenMP 混合并行与纯 MPI 并行 DSMC/PIC 耦合模拟中各计算过程的开销。可以看出，除了 DSMC_Move 和 PIC_Move，其他过程混合并行版本都有着明显的优化效果。值得注意的是，MPI＋OpenMP 混合并行的一个主要目的是缓解多进程间的负载不均衡并减少总体通信开销，表 3 的最后一行和图 10 的结果显示本文算法符合预期效果。

表 3　两个版本在北京超算平台上主要程序的执行时间

Table 3　Total execution times for main procedures on BSCC

		处理器核心数					
		96	192	384	768	1536	3072
DSMC_Move	MPI(DC＋LB)/s	277.6	164.8	67.3	39.0	21.6	12.9
	MPI＋OPENMP(DC＋LB)/s	281.1	168.9	71.1	43.0	23.9	12.4
	性能提升/%	−1.3	−3.0	−5.6	−10.1	−10.7	3.7
Inject	MPI(DC＋LB)/s	6105.8	3017.3	1545.4	770.8	385.7	191.9
	MPI＋OPENMP(DC＋LB)/s	5281.4	2641.2	1337.9	666.2	326.3	170.4
	性能提升/%	13.5	12.5	13.4	13.6	15.4	11.2
PIC_Move	MPI(DC＋LB)/s	114.1	46.6	38.0	29.0	20.4	11.1
	MPI＋OPENMP(DC＋LB)/s	105.9	53.7	41.3	20.6	16.9	9.8
	性能提升/%	7.3	−15.3	−8.7	28.8	17.0	11.7
DSMC_Exchange	MPI(DC＋LB)/s	61.8	34.1	27.8	35.7	28.0	41.1
	MPI＋OPENMP(DC＋LB)/s	68.7	32.1	25.4	20.7	18.4	12.4
	性能提升/%	−11.2	5.9	8.5	42.1	34.4	69.8
PIC_Exchange	MPI(DC＋LB)/s	259.4	152.5	129.1	109.3	129.5	192.2
	MPI＋OPENMP(DC＋LB)/s	201.2	112.8	92.3	63.9	53.3	47.0
	性能提升/%	22.4	26.0	28.5	41.5	58.8	75.5
Poisson_Solve	MPI(DC＋LB)/s	368.9	379.8	390.0	415.6	434.7	453.6
	MPI＋OPENMP(DC＋LB)/s	399.6	359.9	319.2	276.5	237.8	209.2
	性能提升/%	−8.3	5.2	18.2	33.5	45.3	53.9
Rebalance	MPI(DC＋LB)/s	3.8	4.8	7.0	11.7	24.2	50.9
	MPI＋OPENMP(DC＋LB)/s	3.6	4.4	4.0	4.6	6.2	9.6
	性能提升/%	5.6	9.0	42.7	61.0	74.5	81.1

注：两个版本均使用分布式通信策略和动态负载均衡。

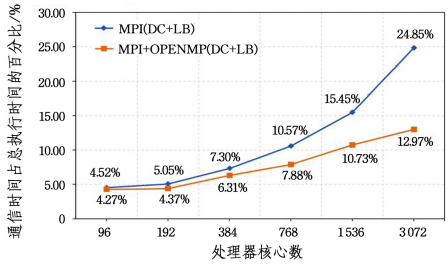


图 10 两个版本的通信时间占总执行时间的百分比

Fig. 10 Percentage of communication cost to total execution time of the two versions

图 11 给出了 MPI+OpenMP 混合并行不同进程数和线程数配置对整体性能的影响。我们在单个节点 96 个核上分别使用 96×1,48×2,32×3,24×4,16×6,12×8,8×12,6×16,4×24,3×32,2×48,1×96 共 12 种不同的进程线程组合来进行测试。可以看出,在一个节点上运行 24 个进程和 4 个线程效果最佳,因此上面的所有测试均是在这一配置下进行的。

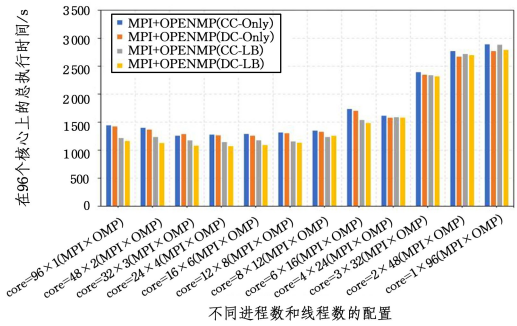


图 11 在单个节点 96 个核心上,使用不同进程数和线程数的配置的总执行时间

Fig. 11 Total execution time on a single node using configurations with different number of processes and threads

结束语 本文针对自主研发的 DSMC/PIC 耦合模拟软件,开展了大规模高效 MPI+OpenMP 混合并行及动态负载均衡研究,设计了两种 MPI 并行通信策略,用于支持粒子在任意并行进程间的动态迁移,提出了动态负载均衡算法,大幅提升了耦合模拟并行效率,设计了 MPI+OpenMP 混合并行算法,有效降低了纯 MPI 并行计算中动态负载均衡的网格重剖分和通信开销。针对 10 亿粒子规模脉冲真空弧等离子体羽流开展了数千处理器核心 DSMC/PIC 耦合并行模拟,验证了并行算法和动态负载均衡效果。后续一方面将在国产超算平台上开展 DSMC/PIC 耦合模拟并行计算和性能优化研究,另一方面将开展 GPU 异构移植工作。

参 考 文 献

[1] BIRD G A. Molecular Gas Dynamics and the Direct Simulation of Gas Flows[M]. Oxford, 1976.

[2] KBIRDSALL C, LANGDON A B. Plasma Physics Via Computer Simulation[J]. Computer Physics Communications, 1986, 42: 151-152.

[3] WHOCKNEY R, EASTWOOD J W. Computer simulation using particles[J]. Institute of Physics, 1988, 76.

[4] BIRD G A. Direct simulation of the boltzmann equation[J]. Physics of Fluids, 1970, 13(11): 2676-2681.

[5] COPPLESTONE S, ORTWEIN P, MUNZ C D, et al. Coupled PIC-DSMC Simulations of a Laser-Driven Plasma Expansion [M]// High Performance Computing in Science and Engineering '15. Springer International Publishing, 2016, 15: 689-701.

[6] COPPLESTONE S, MUNZ C D, PFEIFFER M. PIC-DSMC simulations of plasma plume expansions with ionization and recombination processes[C]// IEEE International Conference on Plasma Science. IEEE, 2016: 1-1.

[7] SMITH B D, BOYD I D, KAMHAWI H, et al. Hybrid-picmodeling of a high-voltage, high-specific-impulse hall thruster[C]// AIAA/ASME/SAE/ASEE. 2013.

[8] KORKUT B, LI Z, LEVIN D A. 3-d simulation of ion thruster plumesusing octree adaptive mesh refinement[J]. IEEE Transactions on Plasma Science, 2015, 43(5): 1706-1721.

[9] BRIEDA L, TAI S Z, KEIDAR M. Near plume modeling of amicro cathode arc thruster [C] // AIAA/ASME/SAEE/ASEE Joint Propulsion Conference. 2013.

[10] TACCOGNA F, MINELLI P, BRUNO D, et al. Kineticdivertor modeling[J]. Chemical Physics, 2012, 398(none): 27-32.

[11] GLEASON-GONZALEZ C, VAROUTIS S, HAUER V, et al. Simulationof neutral gas flow in a tokamak divertor using the direct simulationmonte carlo method[J]. Fusion Engineering & Design, 2014, 89(7/8): 1042-1047.

[12] XU C, ZHANG L, DENG X, et al. Balancing cpu-gpu collaborative high-order cfd simulations on the tianhe-1a supercomputer [C]// IEEE International Parallel & Distributed Processing Symposium. 2014.

[13] XU C, DENG X, ZHANG L, et al. Collaborating cpu and gpu for largescale high-order cfd simulations with complex grids on the tianhe-1a supercomputer[J]. Journal of Computational Physics, 2014, 278: 275-297.

[14] HYA W. The hungarian method for the assignment problem [J]. Naval Research Logistics, 1955, 2: 83-97.

[15] MUNKRES J. Algorithms for the Assignment and Transportation Problems[J]. Journal of the Society for Industrial and Applied Mathematics, 1957, 5(1): 32-38.

[16] HOPKINS, MATTHEWM, BOERNER, et al. Addressing challenges to simulating breakdown and arcevolution in vacuum and low pressure systems[C]// International Conference on Numerical Simulation of Plasmas. 2013.

[17] MHOPKINS M, BOERNER J J, MOORE C H, et al. Ppps-2013: Accommodating large temporal, spatial, andparticle weighting demands for simulating vacuum arc discharge[C]// Abstracts IEEE International Conference on Plasma Science. 2013.

[18] HOPKINS M M, MANGINELL R P, BOERNER J J, et al. Fully kinetic simulation of atmospheric pressuremicrocavity discharge device[C]// IEEE International Conference onPlasma Sciences. 2015: 1-1.

[19] ORTWEINP, BINDERT, COPPLESTONES, et al. Parallel performance of a discontinuous galerkin spectral elementmethod based pic—dsmc solver [C]// High Performance Computing in Science and Engineering' 14: Transactions of the high performance computing center, STUTTGART(HLRS) 2014. 2015: 671-681.

[20] COPPLESTONES, ORTWEINP, MUNZC D, et al. Coupled pic-

dsmc simulations of a laser-driven plasma expansion[C]// High Performance Computing in Science and Engineering '15. 2016: 689-701.

[21] JOHNSON C, PITKGRANTA J. An analysis of the discontinuousgalerkin method for a scalar hyperbolic equation[J]. Mathematics of Computation, 1986, 46(173): 1-26.

[22] KORKUT B, LEVIN D A. Three dimensional dsmc-pic simulations ofion thruster plumes with sugar[C]// AIAA/ASME/SAEE/ASEE Joint Propulsion Conference. 2014.

[23] REVATHIJ, LEVIND A. Chaos: An octree-based pic-dsmc code formodeling of electron kinetic properties in a plasma plume using mpicuda parallelization[J]. Journal of Computational Physics, 2018, 373: 571-604.

[24] LI J, INGHAM D, MA L, et al. Numericalsimulation of the chemical combination and dissociation reactions ofneutral particles in a rarefied plasma arc jet[J]. IEEE Transactions on Plasma Science, 2017, 45(3): 461-471.

[25] YS U, LI J, WANG H, et al. Numerical simulation of chemicalreactions on rarefied plasma plume by dsmc method[J]. IEEE Transactionson Plasma Science, 2021, 49(3): 1-13.

[26] BIRD G A. Definition of mean free path for real gases[J]. Physics of Fluids, 1983, 26(11): 3222-3223.

[27] BIRD G A. Molecular gas dynamics and the direct simulation of gas flow[J]. Clarendon Press, 1994.

[28] BORISJ P. Relativistic plasma simulations-optimization of a hybridcode[C]// International Conference on Numerical Simulation of Plasmas. 1970.

[29] LIMA E, TAVARES F W, JR E B. Finite volume solution ofthe modified poisson-boltzmann equation for two colloidal particles [J]. Physical Chemistry Chemical Physics, 2007, 9(24): 3174-3180.

[30] K L U. of Minnesota. Metis-serial graph partitioning andfill-reducing matrix ordering[OL]. http://glaros. dtc. umn. edu /gkhome/metis/metis/overview, 2013.

[31] LABORATORY A N. Petsc 3. 16—petsc 3. 16. 0 documentation [OL]. https://petsc. org/release/, 2021.

[32] STRUMPACK: STRUctured Matrices PACKages[OL]. http://portal. nersc. gov/project/sparse/strumpack/.

WANG Qingsong, born in 1999, postgraduate. His main research interests in high-performance computing applications.

XU Chuanfu, born in 1980, Ph.D, associate researcher, master supervisor. His main research interests include parallel computing and large-scale science and engineering computing.