doi:10.11887/j.cn.201401002 http://journal.nudt.edu.cn

基于 MPI + CUDA 的异构并行可压缩流求解器*

刘 枫[1,2],李 桦[1],田正雨[1],潘 沙[1]
(1.国防科技大学 航天科学与工程学院,湖南 长沙 410073;
2.中国空气动力研究与发展中心,四川 绵阳 621000)

摘 要:在CPU/GPU异构体系结构计算集群上,建立了基于MPI + CUDA的异构并行可压缩流求解器。讨论了异构结构上的可压缩流并行算法的并行模式,在CPU上执行计算密集度低、指令复杂的计算任务,在GPU上执行计算密集度高、指令单一的计算任务。通过数个算例,对比了异构并行计算和传统CPU并行计算计算结果和计算效率。将该算法运用于高超声速流动的数值模拟中,数值结果显示,基于MPI + CUDA的异构并行可压缩流求解器鲁棒性好,计算效率较CPU同构并行计算提高10倍以上。

关键词:消息传递接口;统一计算设备架构;异构计算;可压缩流
中图分类号:TP310 **文献标志码**:A **文章编号**:1001-2486(2014)01-0006-05

Heterogeneous parallel compressible flow solver based on MPI + CUDA

LIU Feng[1,2], *LI Hua*[1], *TIAN Zhengyu*[1], *PAN Sha*[1]
(1. College of Aerospace Science and Engineering, National University of Defense Technology, Changsha 410073, China;
2. Chinese Aerodynamic Research and Development Center, Mianyang 621000, China)

Abstract: A compressible flow heterogeneous parallel solver based on MPI + CUDA on CPU/GPU heterogeneous system was established. Then different parallel computing models and optimizing methods of compressible flow parallel computing algorithm were discussed. This solver runs different codes with difference resources: the codes which are complex or have low computing density are run on CPU, while the codes which are simple or have high computing density are run on GPU. The heterogeneous systems' computing results and the efficiencies with homogeneous systems were compared through several problems. Finally, the heterogeneous algorithm was applied to the hypersonic flow. The result shows that the algorithm is robust and the computing efficiency is improved ten times more than that of the homogeneous algorithm.

Key words: MPI(Message Passing Interface); CUDA(Compute Unified Device Architecture); heterogeneous computing; compressible flow

随着CPU/GPU异构体系结构的计算集群逐渐成为当今科学计算的主流,越来越多的研究者将由CPU执行的数值模拟算法"改造"为由GPU或者CPU/GPU执行。其目的是充分发挥GPU强大的浮点运算能力,以减少数值模拟所耗费的大量时间,提高计算效率。

复杂飞行器的可压缩流动数值模拟一直是困扰计算流体力学大规模应用于工程实践的难题。其主要原因在于:一方面,可压缩流动流场结构复杂,对于正确模拟包含激波、边界层的流动要求计算精度高,因此计算量巨大;另一方面,为了尽可能准确地获得流场信息,特别是非定常流场信息,需要耗费大量的迭代时间步,这就进一步增大了计算量。这就使得CFD在面对大规模工程计算,特别是飞行包线中大量计算状态的模拟时显得困难重重。

目前,主流计算机中的处理器主要是中央处理器CPU和图形处理器GPU[1]。传统上,GPU只负责图形渲染,而其他大部分工作都交给了CPU。然而相对CPU,GPU在处理能力(浮点运算能力)和存储器带宽上有明显优势。由于图形渲染的高度并行性,使得GPU可以通过增大并行处理单元和存储器控制单元的方式提高处理能力和存储器带宽。主流的GPU单精度浮点计算能力是同代CPU的数十倍,且其性能的提高速度远远超过了CPU所遵照的摩尔定律[2]。可惜的是,由于GPU架构和CPU架构有本质的不同,传统的基于CPU架构建立的算法,不能用于GPU的计算中。近年来,国内外一些文献针对GPU和CFD的计算特点作了一些有益的探讨。光滑流体动力学

* **收稿日期**:2013-05-09
基金项目:国家自然科学基金资助项目(91016010,91216117)
作者简介:刘枫(1984—),男,重庆万州人,博士研究生,E-mail:liufengmaple.33@gmail.com;
李桦(通信作者),男,教授,博士,博士生导师,E-mail:lihua_kd@tom.com

方法(SPH)、格子波尔兹曼方法(LBM)、不可压缩流动求解器、基于结构网格的可压缩流动求解器已经在 GPU 上得到了实现,分别得到了几倍至十几倍的加速效果,但已见报道的文献中往往仅仅单独使用 GPU 来进行数值模拟,鲜有同时发挥 CPU 和 GPU 能力进行模拟的解决方案。然而混合网格作为一种非结构网格,兼具各项同性网格和各项异性网格特性,可以快速生成复杂外形的高质量计算网格。在 GPU 实现中,由于缺乏高效的数据寻址手段和计算分配方法,难以得到较高的加速比,因此在国内外文献中尚未见报道。

为了充分发挥 CPU 和 GPU 各自的浮点运算能力,提高计算效率,本文建立了基于 MPI + CUDA 的异构并行可压缩流求解器。该求解器既可用于结构网格的异构并行实现,也可用于任意多面体的数值模拟,同时可以推广到大规模并行计算中。

1　并行数值方法

1.1　控制方程及离散方法

当不考虑外部加热和彻体力的影响时,三维直角坐标系下非定常守恒形式的 N - S 方程组如下:

$$\frac{\partial Q}{\partial t}+\frac{\partial E_c}{\partial x}+\frac{\partial F_c}{\partial y}+\frac{\partial G_c}{\partial z}=\frac{\partial E_v}{\partial x}+\frac{\partial F_v}{\partial x}+\frac{\partial G_v}{\partial x} \quad (1)$$

式中,x、y、z 分别为三个直角坐标的方向变量,t 为时间变量,Q 为流场守恒变量,E_c、F_c、G_c 分别为三个坐标方向上的无粘对流通量,E_v、F_v、G_v 分别为三个坐标方向上的粘性耗散通量,其具体表达式见文献[3]。

控制方程离散方法采用格子中心型有限体积法。无粘通量采用 AUSM + UP 格式,粘性通量采用中心格式,时间积分采用多步 Runge-Kutta 法,湍流模型采用 Kω-SST 模型。

$$\frac{Q^{n+1}-Q^n}{\Delta t}=-\frac{1}{V}\sum_{face}(F_c-F_v)\cdot nS \quad (2)$$

1.2　并行模式

在 CPU 集群上,传统的并行模式是基于区域分解的粗粒度并行解决方案。其基本思想是把计算区域分解为网格量大致相当的数个子区域,每一个 CPU 上负载一个或者数个子区域的计算量,从而实现各个 CPU 上的负载平衡,通过 MPI 等数据传递模式来实现子区域边界上的数据交换。该模式的优点在于并行代码可以大部分继承串行代码,只需要在数据交换上考虑并行的影响。图 1 给出了 CPU 集群上的并行计算模式示意图。

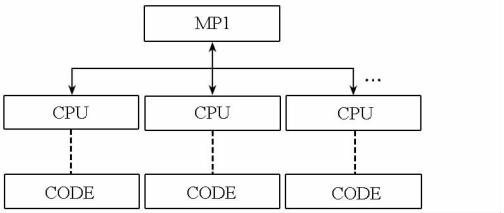

图 1　CPU 集群上的并行计算模式示意图

Fig. 1　Parallel computing model on CPU clusters

在 CPU/GPU 异构并行集群上,由于 CPU 和 GPU 构架存在本质的差异,CPU 适合进行复杂指令的计算任务,比如判断、选择等分支指令,而 GPU 对于分支指令的计算能力极差,对单一指令的计算能力强。因此,本文考虑以下两种并行模式。

模式一:在整个计算域上离散控制方程,然后根据计算流程将需要进行大量迭代计算的大型矩阵交给 GPU 并行执行,CPU 完成 GPU 间的数据交换、初始化、规约计算和后处理任务。图 2 给出了模式一的示意图。在本模式中,所有面通量的计算均由 GPU 完成。

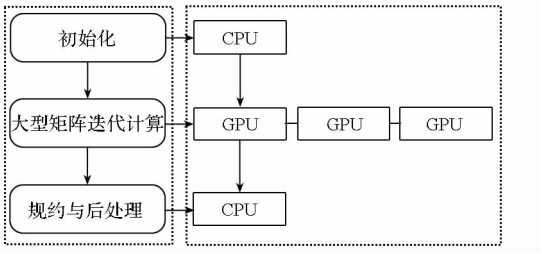

图 2　CPU/GPU 并行集群上的模式一示意图

Fig. 2　Parallel model 1 on CPU/GPU clusters

模式二:本文的模式二是在计算节点概念下提出的。这符合目前巨型机以及小型微机构建异构并行的通用模型。一个计算节点包括一块或者数块 GPU 和一个或者数个 CPU。通过网格分区算法把计算区域分解为网格量大致相当的数个子区域,每一个计算节点上负载一个或者数个子区域的计算量,从而实现各个计算节点上的负载平衡,并通过 MPI 等数据传递模式来实现子区域边界上的数据交换。在每一个计算节点上,将整个计算任务划分为分支指令任务(如边界条件的计算)、单一指令任务(如通量计算、规约计算和时间积分)、数据交换任务(如子区域之间的数据传递)。将分支指令任务和数据交换任务交予计算节点中的 CPU 完成,单一指令任务交予计算节点中的 GPU 来完成,从而实现了计算任务的细粒度分解,达到充分利用 GPU 计算资源的目的。图 3 给出了模式二的计算流程图。

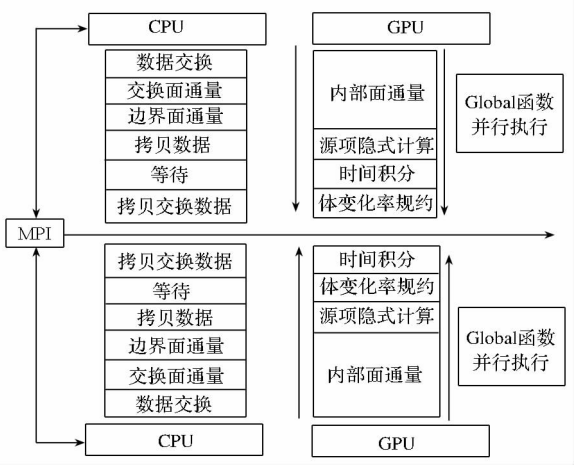

图3　CPU/GPU 并行集群上的模式二计算流程图

Fig. 3　Parallel model 2 on CPU/GPU clusters

对可压缩流动而言,一般内部面通量的计算量远远大于边界面通量和交换面通量的计算量,而且其计算指令单一,所有内部面计算具有高度的一致性。从图3上可以看出,由于模式二中的数据交换发生在计算节点中的CPU上,不同CPU接收数据的多少会有一定的差异,因此交换面通量交予计算节点中的CPU来完成,与此同时,由于CFD数值模拟中边界条件类型繁多,可能出现大量分支选择的情况,因此边界面通量的计算也交由CPU来完成。

从以上分析不难看出,模式一的并行算法简单而直接,容易实现,但不足之处在于难以推广到大规模乃至超大规模的并行计算中,不具有与计算规模无关的通用性。原因就在于大型矩阵的迭代求解需要把整个矩阵映射到每一块GPU的显存上去,当网格量急剧增大时,显存不足的矛盾就凸显了;模式二的并行算法兼容了CPU同构集群中的粗粒度算法的优点,同时充分发挥了GPU细粒度并行计算的特色。由于每一个节点仅负载计算网格的某一个子区域,显存不足的问题得到有效解决,从而形成与计算规模无关的通用异构并行算法。图4给出了采用模式一时网格增大时所需的显存图。当网格量大致为420万时,模式一所需的显存就超出了当前主流GPU的4G显存上限。

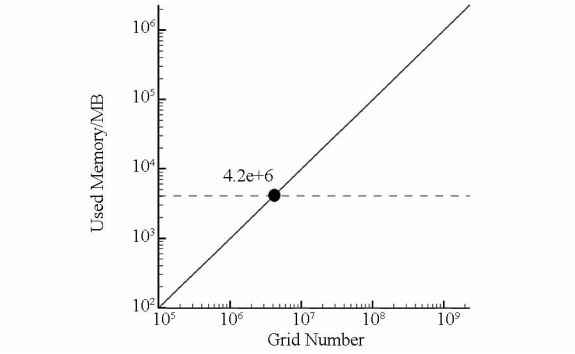

图4　模式一所需显存随网格量变化图

Fig. 4　Memory used by grid number in model 1

2　算例分析

2.1　Sod 问题

这是一个经典算例,如图5所示,在 $t=0$ 时刻,初始间断分为左右两部分;$t>0$ 时,间断分解为一簇向右的激波、接触间断和向左的膨胀波。该问题主要考察算法对激波、接触间断的分辨率。本文选取激波管问题的初始间断分布为($\rho=5.999\ 24$, $u=19.5975$, $p=460.894$),($\rho=5.992\ 42$, $u=-6.196\ 33$, $p=46.0950$),计算时间为0.035,考察范围 $x\in[0,1]$。

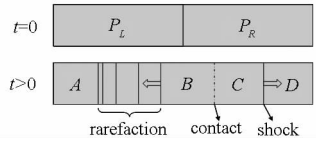

图5　激波管问题示意图

Fig. 5　Wave pattern and the corresponding distribution in the shock-tube.

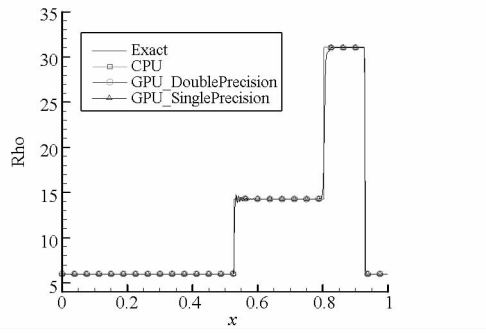

图6　密度分布计算结果同精确解对比

Fig. 6　The density comparing with exact solution

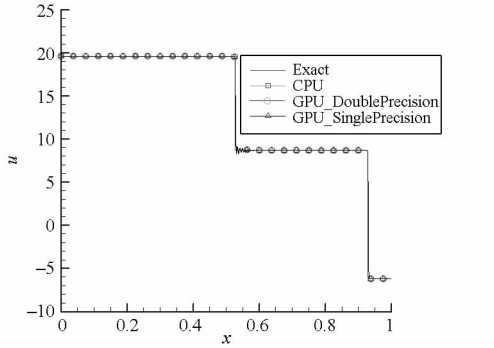

图7　速度分布计算结果同精确解对比

Fig. 7　The velocity comparing with exact solution

本文计算采用的CPU是INTEL I7处理器,主频为3.2GHz,拥有8个核心;采用的GPU为丽台Quadro2000,拥有192个计算核心,单精度计算峰

值为 362.4Gflops,双精度计算峰值为 175.7Gflops。通过如下方式定义加速比:单个 CPU(8 核)计算 100 步的计算时间除以 CPU 加 GPU 计算 100 步的计算时间。如图 6、图 7 所示,对比 CPU、GPU 计算结果和精确解,可以看出 CPU 和 GPU 计算结果几乎完全一致。GPU 单精度和双精度计算结果高度吻合,并且与精确解对比得知其计算精度与 CPU 结果相当。图 8 给出了 GPU 单精度与双精度计算相对于 CPU 计算的加速比。可以看出,随着网格量的增大,加速比快速增加,当网格量 20 000 左右时,单/双精度计算加速比分别达 20.5 和 12.2。单精度计算加速比高于双精度计算,原因在于目前技术水平下 GPU 的单精度浮点计算速度快于双精度浮点计算。

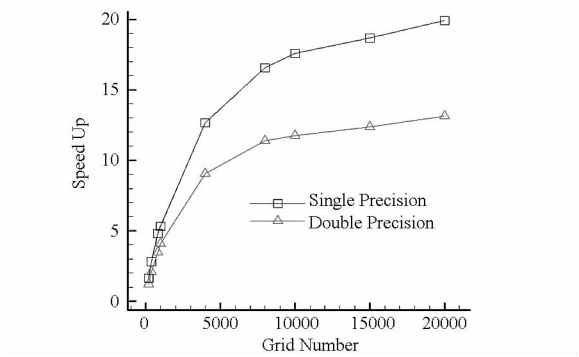

图 8　Sod 问题 GPU 单精度与双精度计算的加速比

Fig. 8　Single precision and double precision's speed up using GPU on Sod problem

2.2　双马赫反射问题

二维强激波双马赫反射问题,如图 9 所示。高速气流受到壁面压缩后,产生一斜激波与运动激波发生干扰,产生另一垂直于壁面的斜激波;同时,在斜激波与垂直于壁面激波之间形成一簇滑移线。

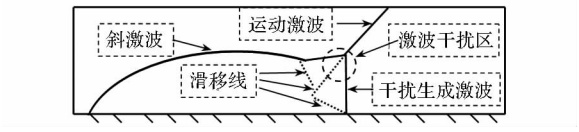

图 9　双马赫反射流场结构图

Fig. 9　Flow structure of double-Mach reflection problem

双马赫反射的问题描述如下:马赫数为 10 的激波,以 60°角在壁面上反射,激波前未扰动气流密度 $\rho = 1.4$,压力 $p = 1.0$;计算区域长 4,宽 1,反射壁面位于底边 $1/6 < x < 4$;计算域左边界及下边界($0 < x < 1/6$)赋以波后条件,右边界为出口条件,下边界($1/6 < x < 4$)为无粘壁面条件,上边界 $Ma = 10$ 的运动激波条件(即以激波速度移动),激波左侧赋以波后条件,右侧赋以波前条件。图 10 ~ 图 12 分别给出了 CPU、GPU 双精度和 GPU 单精度计算得到的密度等值线图。CPU、GPU 双精度和 GPU 单精度三种程序计算的结果从密度等值线图上来看基本没有差异。三者在不同的网格条件下表现出来的性能是一致的,对流场结构的刻画都是相同的。在大网格量的情况下,由局部放大的图来看,三者都可以比较清晰地捕捉出接触间断区域的波系结构。仅从波系结构上看,GPU 单精度的计算结果并不比双精度的计算结果差,对流场的刻画同样比较细致。

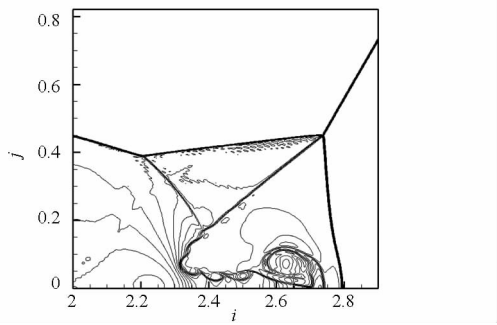

图 10　CPU 计算密度等值线图

Fig. 10　Density line(CPU)

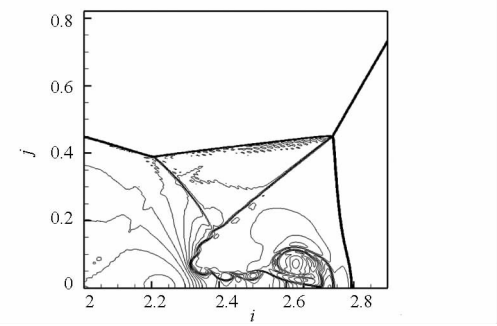

图 11　GPU 双精度计算密度等值线图

Fig. 11　Density line(GPU double precision)

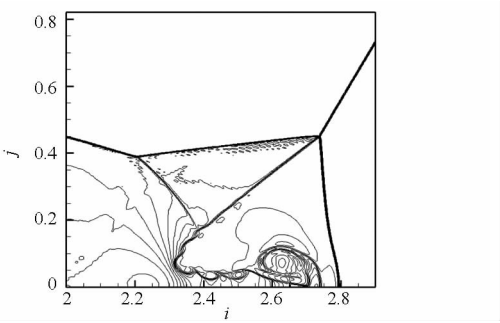

图 12　GPU 单精度计算密度等值线图

Fig. 12　Density line(GPU single precision)

图 13 给出了不同网格量下,GPU 双精度计算和单精度计算相对 CPU 计算的加速比。可以看出,随着网格量增加,二者相对于 CPU 计算的加速比都在增加,单精度计算加速比高于双精度计算加速比,这与一维问题得到的结论是一致的。当网格量约为 380 万时,单精度计算加速比达到 22,双精度计算加速比达到 15。

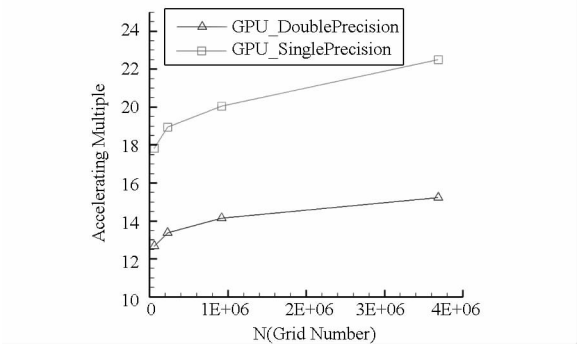

图 13 GPU 单精度和双精度运算加速比

Fig. 13 Single precision and double precision's speed up using GPU on double-Mach reflection problem

2.3 双椭球问题验证算例

文献[7]对双椭球模型进行了精细地实验研究,获得了清晰的纹影照片以及上下表面压力分布。图 14 给出了实验纹影图。图 15 给出了 GPU 双精度计算得到的密度梯度等值线图。图 16、图 17 给出了 GPU 双精度计算得到的上下表面压力系数分布及与实验值的对比。

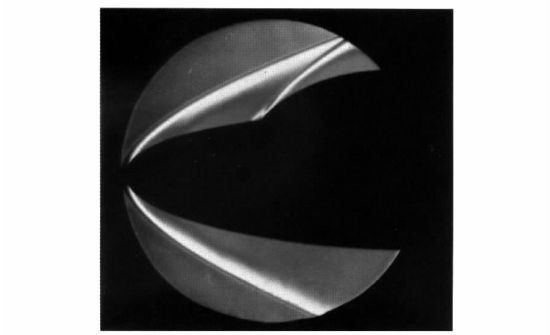

图 14 实验纹影图

Fig. 14 Experiment result

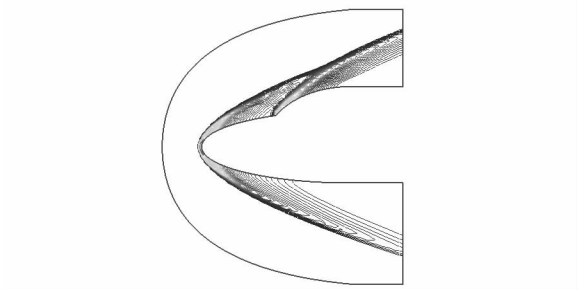

图 15 GPU 双精度计算得到密度梯度图

Fig. 15 Density's grads line of GPU double precision result

从对比结果可以看出,用 GPU 双精度计算得到的密度梯度流场与实验纹影结果符合较好,正确反映了流场的基本结构。从压力分布对比看,计算与实验压力分布符合较好。

3 结论

基于 MPI + CUDA 的异构并行可压缩流求解器能够发挥 GPU 的快速浮点计算能力,从算例对

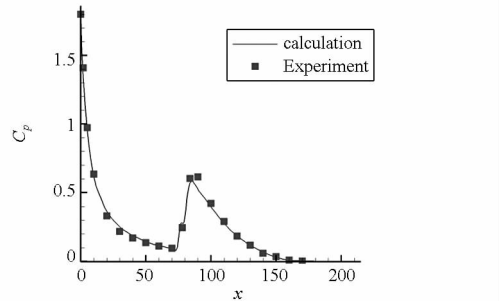

图 16 GPU 双精度计算得到上表面压力分布图

Fig. 16 Comparison of up surface's pressure between GPU double precision's computing and experiment

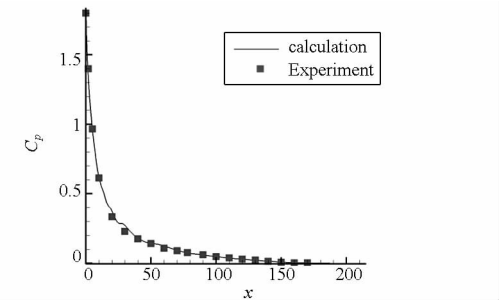

图 17 GPU 双精度计算得到下表面压力分布图

Fig. 17 Comparison of down surface's pressure between GPU double precision's computing and experiment

比来看,计算结果与实验值以及精确解符合较好,具有较好的准确性和鲁棒性,并且相对于 CPU 计算得到了 10 倍以上的加速比,下一步将推广到大规模异构并行计算中。

参考文献(References)

[1] Li S, Li X, Wang L, et al. Accelerating 2-Dimensional CFD on Multi-GPU supercomputer[C]//Proceedings of Solutions to Multiscale Problems, Springer, 2013.

[2] Bertolli A, Betts C. Compiler optimizations for industrial unstructured Mesh CFD applications on GPUs[C]//Proceedings of Solutions to Multiscale Problems, Springer, 2013.

[3] Zhang J, Wang Z J, Zhu S R, et al. GPU Accelerated CFD simulation in electronics cooling[J]. Applied Mechanics and Materials, 2013(1).

[4] 张舒,褚艳利. 高性能运算之 CUDA[M]. 北京:中国水利水电出版社,2009.
ZHANG Shu, ZHU Yanli. High performance computing for CUDA[M]. Chinese Water Press, Beijing, 2009. (in Chinese)

[5] 仇德元. GPGPU 编程技术 - 从 GLSL、CUDA 到 OpenCL[M]. 北京:机械工业出版社,2011.
QIU Deyuan. GPGPU programming: from GLSL CUDA to OpenCL[M]. China Machine Press, Beijing, 2011. (in Chinese)

[6] 阎超. 计算流体力学方法及应用[M]. 北京:北京航空航天大学出版社,2006.
YAN Chao. Computational dynamics: methods and applications[M]. Beihang Universit Press, Beijing, 2006. (in Chinese)

[7] 李素循. 典型外形高超声速流动特性[M]. 北京:国防工业出版社,2007.
LI Suxun. Typical shapes's hypersonic flow aerodymic characteristics[M]. National Defence Industry Press, Beijing, 2007. (in Chinese)