文章编号:1671-4598(2021)02-0144-06 DOI:10.16526/j.cnki.11—4762/tp.2021.02.029 中图分类号:TP311 文献标识码:A

# 基于异构平台的通量分裂格式性能研究

梁正虹,黄 俊,刘志勤,陈 波,杨 茂
(西南科技大学 计算机科学与技术学院,四川 绵阳 621010)

**摘要**:通量分裂是在方程组条件下实现迎风特性的主要手段,为了实现典型通量分裂格式在CPU/GPU异构平台的性能分析。在NVIDIA GTX1660super上,使用统一设备计算架构(CUDA)编程模型实现一维欧拉求解器;以激波管Riemann问题为算例,对矢通量分裂格式van leer、通量差分分裂格式Roe以及混合通量分裂AUSMPW+进行计算分析;数值结果表明,三种格式在异构计算体系能够得到合理且可用的计算结果;Roe格式激波分辨率最高且在CPU/GPU体系加速效果最好;Van Leer激波分辨率较低于Roe和AUSMPW+,计算效率高但其格式构造中存在大量判断分支,影响了加速性能;AUSMPW+格式激波分辨率与Roe相当,加速性能略好于Van Leer。

**关键词**:通量分裂格式;GPU;统一设备计算架构;异构并行计算

# Research on Performance of Flux Split Format Based on Heterogeneous Platform

Liang Zhenghong, Huang Jun, Liu Zhiqin, Chen Bo, Yang Mao
(School of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China)

**Abstract**: Flux splitting is the main method to realize the upwind characteristics under the condition of equations. In order to realize the performance analysis of typical flux splitting formats on CPU/GPU heterogeneous platforms. On NVIDIA GTX1660super, a one-dimensional Euler solver is implemented using the Compute Unified Device Architecture (CUDA) programming model. Taking the Riemann problem of shock tube as an example, van Leer, Roe and AUSMPW + are calculated and analyzed in vector flux splitting scheme, flux differential splitting scheme and mixed flux splitting scheme. The numerical results show that the three schemes can get reasonable and usable results in the heterogeneous computing system. Roe format has the highest resolution and the best acceleration effect in CPU/GPU system. The resolution of Van Leer shock wave is lower than Roe and AUSMPW+, and the calculation efficiency is high. However, there are a lot of judgment branches in the structure of Van Leer shock wave, which affects the acceleration performance. The AUSMPW+ format has a shock resolution comparable to Roe and a slightly better acceleration performance than Van Leer.

**Keywords**: flux split format; GPU; CUDA (compute unified device architecture); heterogeneous parallel computing

## 0 引言

计算流体力学(CFD, computational fluid dynamics)通过数值求解流体动力学控制方程,获取各种条件下的流动数据和作用在绕流物体上的力、力矩和热量等,从而达到研究各种流动现象和规律的目的。CFD是涉及流体力学、计算机科学与技术、计算数学等多个专业的交叉研究领域[1]。在航空航天领域,CFD已成为获取高超声速飞行器气动力和气动热数据、开展高超声速流动基础科学问题研究的三大手段之一[2]。

对于高超声速流动,保持格式的迎风特性对于计算结果十分有益。迎风格式主要沿着矢通量分裂(FVS, flux vector splitting)和通量差分分裂(FDS, flux difference splitting)两个路线发展。FVS的典型格式有Steger-Warming[2]和Van Leer[2];FDS最具代表性的是Roe[2]格式。FVS将对流项通量按其特征值的正负进行分裂,稳定性好但数值粘性较大;FDS通过近似求解Riemann问题获得控制单元通量,计算量较大且需要引入熵修正,增大了数值粘性[3]。20世纪90年代,Liou和Steffen提出兼具FVS和FDS优点的AUSM[4](advection upstream splitting method)格式,经过20多年的发展,其衍生格式已成为目前航

**收稿日期**:2020-12-05; **修回日期**:2020-12-24。
**基金项目**:国家自然科学基金(61802320);四川省教育厅项目(18TD0021)。
**作者简介**:梁正虹(1994-),男,四川绵阳人,硕士研究生,主要从事异构计算、可压缩流体力学方向的研究。
刘志勤(1962-),女,四川省绵阳人,硕士,教授,硕士研究生导师,主要从事高性能计算、数值模拟方向的研究。
陈 波(1963-),男,四川广汉人,工学博士,教授,硕士研究生导师,主要从事科学计算与嵌入式技术方向的研究。
**通讯作者**:黄 俊(1988-),男,四川省自贡人,博士,主要从事最优控制和气动数据库方向的研究。
**引用格式**:梁正虹,黄 俊,刘志勤,等.基于异构平台的通量分裂格式性能研究[J].计算机测量与控制,2021,29(2):144-149.

空航天领域应用最广泛的格式之一。

CPU/GPU 异构并行架构是当今构建大规模计算集群的主要架构之一，计算机技术的快速发展给 CFD 高性能计算领域带来前所未有的机遇与挑战。美国国家航天局（NASA）预测，21 世纪，高效能计算机和 CFD 技术的进一步结合将给各类航空航天飞行器的气动设计带来一场革命[5]。近十年来，国内外专家学者通过 CPU/GPU 异构平台加速 CFD 应用，取得一系列重要成果。D. A. Jacobsen[6] 等成功地将 CFD 应用从单 GPU 扩展至多 GPU 和集群；M. Aissa[7] 等在 GPU 平台上开展了基于结构网格的显格式隐格式加速性能研究；Y. Xia 等[8-9] 研究结构网格、非结构网格和混合网格在 GPU 上实现的差异；I. C. Kampolis 等[10] 分析了 GPU 架构单精度和双精度对仿真结果的影响。马文鹏[11] 等在 GPU 上实现了基于混合网格的非定常流动问题的求解；刘枫[12] 等建立了基于 MPI＋CUDA 的异构并行可压缩流求解器；李大力[13] 等在天河 2 超级计算机上实现了 GPU 的高阶格式隐式求解。国内外学者在 CPU/GPU 异构平台上开展大规模 CFD 并行计算做出大量相关工作，但鲜有文献对不同通量分裂方法的典型格式在 CPU/GPU 异构体系下的性能做出具体分析。

本文以一维激波管问题为算例，给出通量分裂方法的典型格式在 GPU 架构下的具体性能分析，为进一步在 CPU/GPU 异构平台上开展大规模 CFD 工程应用提供有益的指导和参考。

## 1　通量分裂方法

在进行 CFD 数值模拟中，差分格式的构造是 CFD 的关键之一。迎风类差分格式是从 20 世纪 80 年代开始兴起，在构造时就体现了方程在波动和流量等传播方向上的物理特性，与中心差分格式相比具有表现具体流动物理特征的天然优势。发展至今，各种迎风类格式已经成为工程界和学术界研究的主流。根据对方程物理特性的不同表现形式，迎风格式也可大致分为矢通量分裂（FVS）方法和通量差分分裂（FDS）方法。

### 1.1　矢通量分裂

FVS（flux vector splitting）方法，最初发展于 20 世纪 80 年代，它根据相关传播速度的正负对通量项进行分裂。矢通量分裂方法，简单、高效、鲁棒性高、理论上不会出现非物理解，并且具有较强线性波（如激波等）捕捉能力。根据不同的分裂方式，构造出不同的 FVS 格式，最具代表性是 Steger－Warming 格式和 Van Leer 格式。其中由于 Van Leer 格式在驻点、声速点等特征值变号的附近区域可以光滑过度，从而在航空航天工程领域得到广泛应用。

一维欧拉方程进行矢通量分裂：

$$\frac{\partial U}{\partial t}+\left(\frac{\partial \boldsymbol{F}}{\partial x}+\frac{\partial \boldsymbol{F}^{-}}{\partial x}\right)=0 \tag{1}$$

Van Leer 格式通量分裂表达式：

$Ma>1$：

$$\boldsymbol{F}^{+}=0,\ \boldsymbol{F}^{-}=\boldsymbol{F} \tag{2}$$

$|Ma|>1$：

$$\boldsymbol{F}^{\pm}=\begin{bmatrix} f_1^{\pm} \\ \dfrac{f_1^{\pm}}{\gamma}[(\gamma-1)u\pm 2c] \\ \dfrac{f_1^{\pm}}{2(\gamma^2-1)}[(\gamma-1)u\pm 2c]^2 \end{bmatrix} \tag{3}$$

其中：$f_1^{\pm}=\pm\rho c\left(\dfrac{Ma\pm 1}{2}\right)^2$

$Ma<-1$：

$$\boldsymbol{F}^{+}=\boldsymbol{F},\ \boldsymbol{F}^{-}=0 \tag{4}$$

### 1.2　通量差分分裂

通量差分分裂是迎风类格式的另一条发展路线，FDS 以 Godunov 方法为基本思路，但与 Godunov 精确求解黎曼问题的方法不同，FDS 近似求解黎曼问题，其对激波和接触间断都具有很高的分辨率。根据不同的解近似黎曼问题的方法，构造不同的 FDS 格式，最具代表性的是 Roe 格式。

将一维守恒形式欧拉方程表示为：

$$\frac{\partial U}{\partial t}+\bar{\boldsymbol{A}}\frac{\partial U}{\partial x}=0 \tag{5}$$

$\bar{\boldsymbol{A}}$ 是由左右状态变量（$u_L$，$u_R$）通过 Roe 平均组成的常矩阵。将矩阵进行特征分解：

$$\bar{\boldsymbol{A}}=\bar{\boldsymbol{S}}^{-1}\bar{\boldsymbol{\Lambda}}\boldsymbol{S} \tag{6}$$

Roe 格式通量表达式为：

$$F_{i+1/2}=\frac{1}{2}[F(U_L)+F(U_R)]-\frac{1}{2}S^{-1}\,|\,\Lambda\,|\,S(U_L-U_R) \tag{7}$$

其中：下标（$i+1/2$）为控制体表面；下标 $L$ 和 $R$ 为控制体表面左右两侧。

### 1.3　AUSM 分裂方法

在 20 世纪 90 年代初，Liou 和 Steffen 利用不同格式的优点，提出 AUSM（advection upstream splitting method）格式。它结合了矢通量分裂在非线性波捕捉上的鲁棒性和通量差分分裂在线性波上的高分辨率。从格式构造来讲，AUSM 是 Van Leer 格式的一种改进，但从其耗散项来分析，它是一种 FVS 与 FDS 的复合格式。目前，AUSM 格式经过 20 多年的发展，已经衍生出一系列格式。本文以高超声速领域使用较广的 AUSMPW＋格式为基础，研究其在 GPU 硬件架构下的计算性能。

AUSMPW＋格式具体表达式：

$$F_{i+1/2}=\bar{M}a_L^{+}a_{1/2}\Phi_L+\bar{M}a_R^{+}a_{1/2}\Phi_R+(p_L^{+}p_L+p_R^{+}p_R) \tag{8}$$

其中 $\Phi=\begin{pmatrix}\rho\\ \rho u\\ \rho H\end{pmatrix}$，$a_{1/2}^{L}$、$a_{1/2}^{R}$ 分别界面左右声速，网格界面统一声速和界面左右马赫数可表示为：

$$a_{1/2}=\frac{(a_{1/2}^{L}+a_{1/2}^{R})}{2},\ Ma_{L,R}=\frac{V_{L,R}}{a_{1/2}} \tag{9}$$

$$Ma_{L,R}^{\pm}=\begin{cases}\pm\dfrac{1}{4}(Ma_{L,R}\pm 1)^2, & |\,Ma_{L,R}\,|\leqslant 1\\ \dfrac{1}{2}(Ma_{L,R}\pm|\,Ma_{L,R}\,|), & |\,Ma_{L,R}\,|>1\end{cases} \tag{10}$$

$$p_{L,R}^{\pm}=\begin{cases}\dfrac{1}{4}(Ma_{L,R}\pm 1)^2(2\mp Ma_{L,R}), & |Ma_{L,R}|\leqslant 1\\ \dfrac{1}{2}[1\pm \mathrm{sign}(Ma_{L,R})], & |Ma_{L,R}|>1\end{cases} \tag{11}$$

$$Ma_{1/2}=Ma_L^{+}+Ma_R^{+},\ p_s=p_L^{+}p_L+p_R^{-}p_R \tag{12}$$

$$\overline{Ma}_L^{+}=\begin{cases}Ma_L^{+}+Ma_R^{+}[(1-f_w)(1-f_R)-f_L], Ma_{1/2}\geqslant 0\\ Ma_L^{+}f_w(1+f_L), Ma_{1/2}<0\end{cases} \tag{13}$$

$$\overline{Ma}_R=\begin{cases}Ma_R^{-}+Ma_L^{+}[(1-f_w)(1-f_L)-f_R], Ma_{1/2}<0\\ Ma_R^{-}f_w(1+f_R), Ma_{1/2}\geqslant 0\end{cases} \tag{14}$$

式中，$f$ 为基于压力的权函数，具体形式参考文献[2]。

## 2 Riemann 问题与计算模型

### 2.1 Riemann 问题

Riemann 问题是 CFD 的经典算例，它包含了 CFD 中的各种间断（膨胀波、激波及接触间断等）问题，对于这些间断问题的求解一直是 CFD 发展的难点和核心问题。Riemann 问题具有精确解，可以验证数值方法的精度，由此，一般差分格式都是建立在求解 Riemann 问题基础之上，之后再向多维扩展。

一维 Riemann 问题实质上是激波管问题。激波管是一根两端封闭、内部充满气体的直管。直管中一层薄膜将管隔开，在薄膜两侧充满均匀理想气体，薄膜两侧气体的压力、密度不同。当时，气体处理静止状态。当时，薄膜瞬时破裂，气体从高压端冲向低压端，同时在管内形成激波、稀疏波和接触间断等复杂波系。激波管问题初始条件简单，计算网格只需一维等距划分，计算过程不引入复杂处理、网格质量等非格式因素的影响。

激波管示意图如图 1 所示。

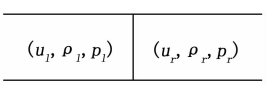


图 1 Sod 激波管问题

对于本次计算问题，计算区域为 $X=(0,1)$，边界条件在计算区域边界处取内点的一阶插值，初始间断分布为：

$$(u_l,\rho_l,p_l)=(0,1,1)\qquad 0\leqslant x<0.5$$

$$(u_r,\rho_r,p_r)=(0,0.125,0.1)\qquad 0.5\leqslant x\leqslant 1$$

### 2.2 控制方程

不考虑热源、热传导和体积力的影响，控制方程采用一维非定常欧拉方程的守恒形式：

$$\frac{\partial \boldsymbol{U}}{\partial t}+\frac{\partial \boldsymbol{F}}{\partial x}=0$$

式中：

$$\boldsymbol{U}=\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix},\ \boldsymbol{F}=\begin{pmatrix}\rho u\\ \rho u^2+p\\ (E+p)u\end{pmatrix} \tag{15}$$

补充理想气体状态方程使方程组封闭：

$$p=(\gamma-1)\left(E-\frac{1}{2}\rho u^2\right) \tag{16}$$

## 3 基于 GPU 的一维欧拉求解器

### 3.1 CPU/GPU 并行模式

GPU 不是一个独立执行的平台，GPU 的工作需要 CPU 协同，在异构计算体系中，CPU 与 GPU 通过 PCIE 总线连接在一起协同工作。CPU 称为主机端（host），GPU 称为设备端（device）。CPU 可以实现复杂的逻辑运算，但其计算单元较少；与此相反，GPU 包含大量的计算单元，相较于 CPU，GPU 线程是轻量级的，上下文切换开销小，比较适合执行大规模数据并行的计算密集型任务。

GPU 的核心组件是流多处理器（SM，streaming multiprocessor），SM 的核心组件包括 CUDA 计算核心、共享内存、寄存器等，SM 可以并发执行数百上千线程，并发能力依赖 SM 拥有的资源数。GPU 执行模式为单指令多线程（SIMT，single instruction multiple thread）模式，即一个 warp（通常为 32 个线程）执行同一条指令，遇见条件分支时，按分支顺序执行。

本文采用 CUDA 对 GPU 进行编程，CUDA 是 NVIDIA 公司推出的通用并行计算架构。一个典型的 CUDA 程序包括 Host 端和 Device 端两部分代码，Host 代码在 CPU 执行，Device 代码调用内核在 GPU 上执行。典型 CUDA 程序的执行流程如下：

1）分配 host 端内存，进行数据初始化；

2）分配 device 端内存，将 host 数据拷贝至 device 上；

3）调用 CUDA 核函数在 device 上执行运算；

4）将 device 上计算结果拷贝至 host；

5）释放 device 端和 host 端内存。

### 3.2 基于 CPU/GPU 的 CFD 并行算法

图 2 展示了一个典型的 CFD 算法流程，CFD 程序主要包括了 3 个关键部分，时间步长计算，算法核心求解步计算和进行边界条件处理。其中，算法的核心求解步占用了 CFD 计算的大量时间。在 CPU/GPU 异构并行计算体系如何分配这三大部分将会决定整个求解器的性能。

本文将时间步长计算分为两部分，全局时间步长的计算和系统迭代时间步的计算。迭代时间步的计算涉及到复杂的逻辑判断和循环过程。鉴于此，将迭代时间步的计算置于 CPU 上执行。针对如何分配 CFD 程序三个关键部分的计算，本文考虑两种并行算法。

算法一：对于算法核心步骤的计算，需要将流场变量 $u$ 从 CPU 传输至 GPU（主机端到设备端）；而边界条件的计算，需要用到核心算法求解器计算过后的流场变量 $d\_u$（在 GPU 上分配的变量数组之前添加 $d$，以区别 CPU 上具有相同功能的数组），$d\_u$ 需要从设备端传输至主机端。图 3 给出了算法一的示意图。在本模式中，整个计算过程只有核心求解器的计算置于 GPU，其余部分全部交由 CPU 完成。

算法二：图 4 给出了模式二的示意图，全局时间步长的计算、核心算法求解器的求解和边界条件的处理全部置

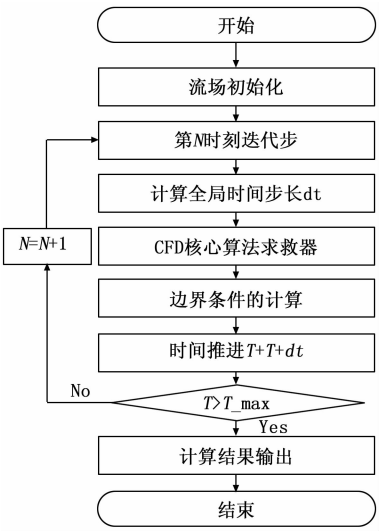


图 2　CFD 算法流程图

于 GPU 上执行，最小化 CPU 和 GPU 之间的数据传输。主机流场变量 $u$ 和 $d\_u$ 分别仅在循环之前和之后传输一次，而只有声速 $d\_a$ 需要从设备端单向传输至主机端进行迭代时间的计算。

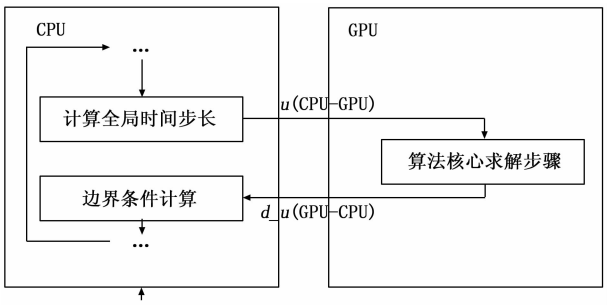


图 3　并行算法一

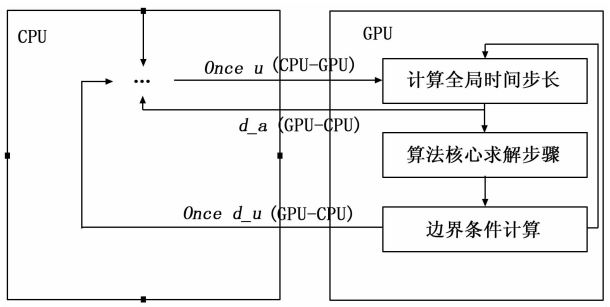


图 4　并行算法二

对于 GPU 上的大多数应用程序，限制程序性能瓶颈的是 CPU 和 GPU 之间的带宽[14-16]，而非 GPU 的浮点性能（单块 GPU RTX－3090 的浮点性能已达到 35.7 TFLOP/s）。此外，在许多内存绑定 CFD 程序中，带宽达到极限，浮点性能远远小于 GPU 峰值的情况经常出现。综合考虑，本文采用算法二进行求解器实现。

## 4　算例分析

本文建立一维欧拉求解器，对 Sod 激波管问题进行计算，分析了通量分裂方法的典型差分格式在 CPU/GPU 异构并行体系中的计算性能。本文主要是针对格式本身性能进行研究，故 Van Leer 格式、Roe 格式和 AUSMPW＋格式均采用一阶计算，计算精度为双精度。

### 4.1　GPU 计算结果

测试采用的 CPU 为 INTEL I7 处理器，8 核，主频为 3.2 GHz，主机内存 32 GB，GPU 为 NVDIA GTX 1 660 super，拥有 1 408 个 CUDA 计算核心，单精度浮点性能约 5 300 GFLOP/s，双精度浮点性能约 170 GFLOP/s。计算取时结果。

$t=0.2$ 时刻，激波管内流场结构如图 5 所示。激波管内流场被分为 A、B、C、D 共 4 个区域，A 区域和 D 区域中波未传播，未受到扰动干扰，保持着流动初始状态；B 区域为膨胀波扫过后的区域；C 区域为激波扫过区域。B 区域和 C 区域以一个接触间断分开，B、C 两区域压力相同、密度和温度不同。

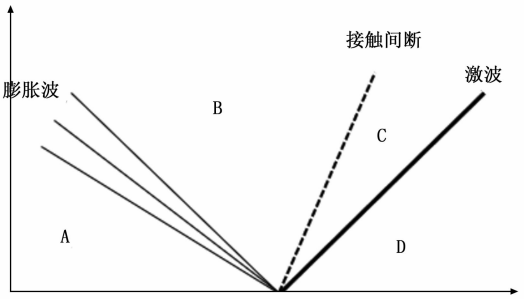


图 5　$t=0.2$ 波系结构图

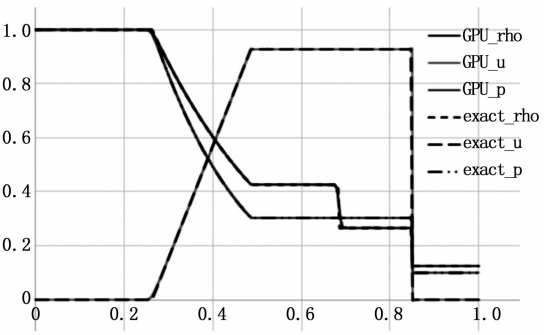


图 6　Van Leer 计算结果

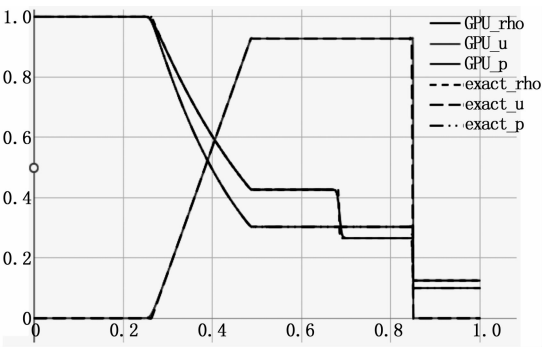


图 7　Roe 计算结果

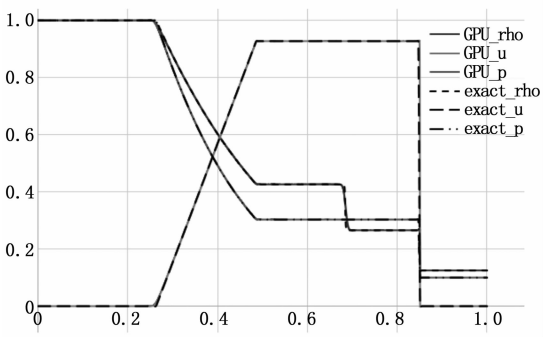


图 8 AUSMPW+计算结果

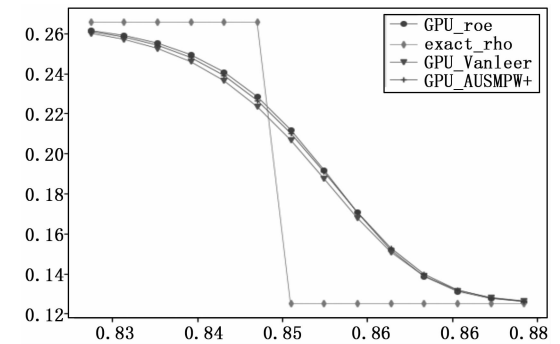


图 10 激波局部放大图

图 6～8 分别展示了 $t=0.2$ 时，3 种格式在 10 000 个网格点下的计算结果。如图所示，3 种格式都能清晰地分辨出所求时刻流场的波系结构。气体压力差产生的激波从 $x=0.5$ 处向右传播，$t=0.2$ 时刻传播至 $x=0.85$ 附近，产生速度、密度和压力的跃迁；$x=0.25$ 和 $x=0.5$ 之间 3 个变量缓慢变化，形成膨胀波；$x=0.68$ 处，速度和压力不变，密度产生跃迁，为接触间断。通过与精确值对比，3 种格式在 CPU/GPU 异构体系下得出合理且可用的计算结果。

### 4.2 格式效应分析

图 9 展示了 $t=0.2$ 时刻，3 种格式在 256 网格点下，流场的密度分布，为了便于比较格式对流场波系的分辨情况，给出 $t=0.2$ 时刻的流场密度精确解进行比较。可以看出，本文所采用的 3 种格式在 256 网格下都能表述所求时刻间断分布。格式本身对于间断的捕捉有差别，图 10 展示了激波附近局部放大图，Roe 激波分辨率最高，AUSMPW+与 Roe 激波分辨率相当，Van Leer 激波分辨率低于 Roe 和 AUSMPW+。值得注意的是 AUSMPW+格式，它是一种 FVS 和 FDS 的复合格式，兼有 FVS 的计算效率和 FDS 的间断高分辨率。计算结果表明，它在间断前后表现出与 Roe 格式相当的性能，同时又兼有与 Van Leer 格式相当的计算效率。

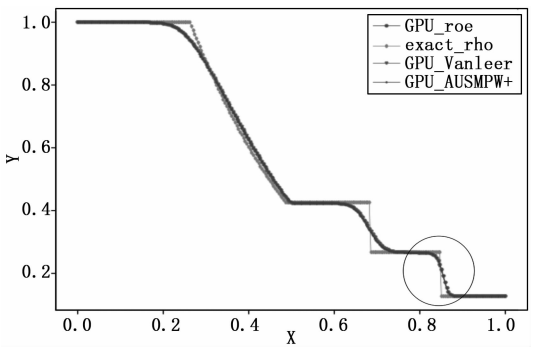


图 9 3 种格式计算对比

### 4.3 格式加速性能分析

本文研究格式本身在 CPU/GPU 上性能分析，加速比定义为算法核心求解步在 CPU 运行时间与相应 GPU 时间的比值，为排除额外因素干扰，时间步长取固定值 0.000 001。

表 1 为 3 种典型格式在 CPU 执行时间（100、1 000、10 000 共 3 种网格计算规模），由计算结果可知，Van Leer 计算效率最高，AUSMPW+计算所需时间略高于与 Van Leer，Roe 消耗时间最多。表 2 展示了 3 种格式在 GPU 执行时间。在本文的计算的网格规模中，GPU 硬件体系结构上，Van Leer 计算效率稍高于 Roe 和 AUSMPW+。表 3 体现了 3 种格式的加速比，在网格规模较小时，GPU 计算时间大于 CPU 计算。主要原因是网格规模小，无法体现 GPU 众核体系进行大规模计算的优势。加速比由高到低依次为 Roe、AUSMPW+、Van Leer。其影响因素有：Roe 格式构造中，需要对矩阵进行计算，其计算量远大于 Van Leer 和 AUSMPW+，而且 Roe 没有条件语句分支，控制流指令较少，运行中 warp 分支少，格式适用于 GPU 单指令多线程运行模式，故 Roe 加速效果最好；AUSMPW+和 Van Leer 两种格式存在大量条件分支，不利于 GPU 运行模式，影响了加速效果；AUSMPW+加速优于 Van Leer 的原因是 AUSMPW+计算量大于 Van Leer（从格式构造来看，AUSMPW+是 Van Leer 的改进，其计算步骤更多更复杂）。从以上分析结果来看，影响格式在异构平台的加速性能主要是格式存在的条件判断分支以及格式的运算量。

表 1 CPU 不同网格规模格式计算时间 s

| 网格规模 | Roe | Van Leers | AUSMPW+ |
|---|---|---|---|
| 100 | 5.340 | 3.168 | 3.732 |
| 1 000 | 80.058 | 28.924 | 35.033 |
| 10 000 | 856.254 | 285.033 | 362.598 |

表 2 GPU 不同网格规模格式计算时间 s

| 网格规模 | Roe | Van Leers | AUSMPW+ |
|---|---|---|---|
| 100 | 13.355 | 10.908 | 11.449 |
| 1 000 | 15.529 | 12.748 | 13.026 |
| 10 000 | 30.125 | 30.366 | 29.403 |

表 3 格式加速比 s

| 网格规模 | Roe | Van Leers | AUSMPW+ |
|---|---|---|---|
| 100 | 0.397 | 0.290 | 0.326 |
| 1 000 | 5.155 | 2.268 | 2.689 |
| 10 000 | 28.423 | 9.387 | 12.332 |

## 5　结束语

为了研究通量分裂格式在异构并行计算平台的加速性能，本文在 GTX 1660 super 上利用 CUDA 开发了一维欧拉求解器，并对当前航空航天领域常用的 3 种典型格式进行计算分析，结果表明：

1）Roe 格式对流场解析效果最好，且在异构平台加速性能最高，目前应用前景最好。

2）格式构造中存在的条件分支严重影响了格式的加速性能，构造数值格式时，合理地减少条件分支，将会大大提高其在异构平台的加速效果。

本文的工作对于在 CPU/GPU 异构并行计算平台开展 CFD 算法相关研究具有一定的指导意义。

**参考文献：**


[1] 徐传福，车永刚，李大力，等. 天河超级计算机上超大规模高精度计算流体力学并行计算研究进展 [J]. 计算机工程与科学，2020，42 (10)：1815－1826.

[2] 陈坚强，张益荣，郭勇颜，等. 高超声速流动数值模拟方法及应用 [M]. 北京：科学出版社，2019.

[3] 李艾挺，朱阳历，李　文，等. 跨声速压气机数值模拟格式效应研究 [J]. 推进技术，2020，41 (2)：70－78.

[4] Steffen L C J. A New Flux splitting scheme [J]. Journal of Computational Physics，1993，107 (1)：23－39.

[5] Slotnick A J，Khodadoust J，Alonso D，et al. CFD vision 2030 study：A path to revolutionary computational aerosciences：NASA/CRC2014－218178 [R]. Washington：NASA，2014.

[6] Jacobsen D A，Senocak I. Multi－level parallelism for incompressible flow computations on GPU clusters [J]. Parallel Computing，2013，39 (1)：1－20.

[7] Aissa M，Verstraete T，Vuik C. Toward a GPU－aware comparison of explicit and implicit CFD simulations on structured meshes [J]. Computers & Mathematics with Applications，2017，74 (1)：201－217.

[8] Xia Y，Lou J，Luo H，et al. OpenACC acceleration of an unstructured CFD solver based on a reconstructed discontinuous Galerkin method for compressible flows [J]. International Journal for Numerical Methods in Fluids，2015，78 (3)：123－139.

[9] Tsoutsanis P，Antoniadis A F，Jenkins K W. Improvement of the computational performance of a parallel unstructured WENO finite volume CFD code for Implicit Large Eddy Simulation [J]. Computers & Fluids，2018，173：157－170.

[10] Kampolis I C，Trompoukis X S，Asouti V G，et al. CFD－based analysis and two－level aerodynamic optimization on graphics processing units [J]. Computer Methods in Applied Mechanics and Engineering，2010，199 (9/10/11/12)：712－722.

[11] Ma W P，Lu Z H，Zhang J. GPU Parallelization of Unstructured/Hybrid Grid ALE Multi－Grid Solver for Moving Bodies [A]. International Conference on Parallel Computing in Fluid Dynamics [C]. Springer，Berlin，Heidelberg，2013.

[12] 刘　枫，李　桦，田正雨，等. 基于 MPI＋CUDA 的异构并行可压缩流求解器 [J]. 国防科技大学学报，2014 (1)：9－13.

[13] Li D L，Chuanfu X，Yongxian W，et al. Parallelizing and optimizing large－scale 3D multi－phase flow simulations on the Tianhe－2 supercomputer [J]. Concurrency & Computation Practice & Experience，2016，28 (5)：1678－1692.

[14] Parna P，Meyer K，Falconer R. GPU driven finite difference WENO scheme for real time solution of the shallow water equations [J]. Computers & Fluids，2018，161：107－120.

[15] Ha S，Park J，You D. A GPU－accelerated semi－implicit fractional－step method for numerical solutions of incompressible Navier－Stokes equations [J]. Journal of Computational Physics，2018，352：246－264.

[16] Zhu X，Phillips E，Spandan V，et al. AFiD－GPU：a versatile Navier－Stokes solver for wall－bounded turbulent flows on GPU clusters [J]. Computer physics communications，2018，229：199－210.


(上接第 143 页)


[4] Engel J，Koltun V，Cremers D. Direct sparse odometry [J]. IEEE Transactions on Pattern Analysis and Machine Intelligence，2018，40 (3)：611－625.

[5] Forster C，Pizzoli M，Sscaramuzza D. SVO：fast semi－direct monocular visual odometry [A]. 2014 IEEE International Conference on Robotics and Automation ICRA 2014 [C]. Hong Kong，2014：15－22.

[6] Raul M A，Tardos J D. Orb－slam2：An open－source slam system for monocular，stereo，and rgb－d cameras [J]. IEEE Transactions on Robotics，2017，33 (5)：1255－1262.

[7] Sola J. Quaternion kinematics for the error－state Kalman Filter [R]. 2017：50－59.

[8] 徐景硕，秦永元，彭　蓉. 自适应卡尔曼滤波器渐消因子选取方法研究 [J]. 系统工程与电子技术，2004，26 (11)：1552－1554.

[9] Xia Q J，Rao M，Ying Y Q，et al. Adaptive fading Kalman filter with an application [J]. Automatica，1994，30 (12)：1333－1338.

[10] Trawny N，Roumeliotiss I. Indirect Kalman filter for 3D attitude estimation [D]. Twin Cities：University of Minnesota，Department of Computer Science and Enginnering，2005：2－23.

[11] 秦永元. 卡尔曼滤波与组合导航原理 [M]. 第 3 版. 西安：西北工业大学出版社，2015.

[12] Gao B B，Gao S. Interacting multiple model estimation－based adaptive robust unscented Kalman filter [J]. Journal of Control，Automation and Systems，2017：1－13.

[13] 徐一鸣，李　笑，杨凯凯，等. 基于深度学习的四旋翼无人机控制系统设计 [J]. 计算机测量与控制，2020，28 (5)：123－127.

[14] 罗秋凤，高艳辉，张　锐，等. 轻型无人机飞行控制系统适航安全性研究 [J]. 计算机测量与控制，2020，28 (8)：139－143.