基于多块结构网格的并行计算及负载平衡研究

李桂波，杨国伟
（中国科学院力学研究所高温气体动力学重点实验室，北京 100190）

摘 要：从并行计算流体力学程序的稳定性和效率两大问题入手，针对多块结构网格的通用数据传输方法和基于遗传优化算法的负载平衡方法，并在已有串行多块结构网格程序基础上发展了相应的并行程序。该并行程序以物理区域分割为基础，采用 MPI 实现消息传递，适用于各种不同的并行机体系结构，具有很好的可移植性。大量数值实验证明，本文发展的并行程序具有良好的稳定性和并行效率，可以进一步应用于大规模实际工程计算。

关键词：计算流体力学；多块结构网格；并行计算；遗传算法；负载平衡
中图分类号：V211.3；O35 **文献标识码**：A **文章编号**：1000-1328(2011)06-1224-07
DOI：10.3873/j.issn.1000-1328.2011.06.003

Study on Parallel Computation and Load Balance Strategy Based on Multiblock Structured Grid

LI Gui-bo, YANG Guo-wei
(Key Laboratory of High Temperature Gas Dynamics, Institute of Mechanics, Chinese Academy of Sciences, Beijing 100190, China)

Abstract: In order to improve stability and efficiency of computational fluid dynamics (CFD) program, a general data transfer method and a genetic algorithm load balance strategy-based, are presented in this paper, A parallel program is developed on the basis of existing serial program for the multiblock structured grid. This parallel program whose message transport mechanism is based on the MPI can be run on different parallel computer architectures and has better portability. Many numerical experiments show that the parallel program is stable and efficient and can be further applied to massively engineering computations.

Key words: CFD; Multiblock structured grid; Parallel computation; Genetic algorithm; Load balance

0 引 言

计算流体力学经过近几十年的发展，已经在航空、航天、船舶、汽车、气象等领域获得了广泛的应用。现代流体力学所要解决的工程问题越来越复杂，为得到可信的计算结果，计算网格非常大，动辄上千万。然而，当前串行程序的计算效率受到计算机硬件水平的制约。例如，对于上千万网格单元的全机气动性能计算，串行程序通常需要几十甚至上百天时间，并且未来飞行器性能的确定，将依赖于在“虚拟风洞（即 CFD）”数据基础上产生的“虚拟飞行”[1]。因此，要解决这类千万量级网格工程计算问题，几乎唯一的可行手段是发展大规模并行计算。

本文发展的并行程序以已有的基于多块结构网格的串行程序为基础[2]，采用物理区域分割并行方法。该并行程序采用 MPI 实现消息传递，可以在任何配备 MPI 环境的大型机、工作站及集群系统中运行，具有很好的可移植性。在编程上采用 SPMD（Single Program Multiple Data）策略，即多个 CPU 同时执行相同的代码，每个 CPU 只处理自己相关的数据，CPU 间通过消息传递同步数据。具体过程为：首先划分整个物理区域为合理的多块结构网格，把各网格块尽量均衡地分配到各 CPU 上独立运算，每次全场子迭代结束后各 CPU 通过消息传递各网格

收稿日期：2010-06-01； 修回日期：2010-11-09

块边界流场数据实现整个流场信息沟通,然后主进程收集全场数据判断收敛情况并按需要读写各种中间文件。对于并行程序,数据发送接收过程中非常容易阻塞,造成死锁,因此数据传输方法对并行程序稳定性有重要影响。同时,负载平衡是并行效率的重要保证。针对多块结构网格的特点,本文提出了通用的数据传输方法,并通过设计特殊染色体利用遗传优化算法实现多块结构网格负载平衡优化。

通过对多个复杂外形跨音速算例的数值验证,计算结果与实验符合良好。大量实例计算也表明,本文采用的数据传输方法和负载平衡方法具有很好的稳定性和并行效率。

1　基于多块结构网格 NS 方程求解器

一般曲线坐标系下,无量纲形式的三维 NS 方程为:

$$\frac{\partial \boldsymbol{Q}}{\partial \tau}+\frac{\partial \boldsymbol{E}}{\partial \xi}+\frac{\partial \boldsymbol{F}}{\partial \eta}+\frac{\partial \boldsymbol{G}}{\partial \zeta}=\frac{\partial \boldsymbol{E}_v}{\partial \xi}+\frac{\partial \boldsymbol{F}_v}{\partial \eta}+\frac{\partial \boldsymbol{G}_v}{\partial \zeta} \tag{1}$$

其中: $\boldsymbol{Q}=\frac{1}{\boldsymbol{J}}[\rho,\rho u,\rho v,\rho w,\rho e,\rho k,\rho\omega]^{\mathrm{T}}$; $\boldsymbol{E},\boldsymbol{F},\boldsymbol{G}$ 和 $\boldsymbol{E}_v,\boldsymbol{F}_v,\boldsymbol{G}_v$ 分别为对流项和扩散项; $\boldsymbol{J}$ 为坐标变换 Jacobian 行列式 $\boldsymbol{J}=\left|\frac{\partial(\xi,\eta,,\tau)}{\partial(x,y,z,t)}\right|$。

通过在方程(1)中引入伪时间导数项,借助伪时间方向的"子迭代"技术,得到二阶时间精度的 LU-SGS 格式。

$$\begin{aligned}&(\boldsymbol{L}+\boldsymbol{D})\boldsymbol{D}^{-1}(\boldsymbol{D}+\boldsymbol{U})\Delta\boldsymbol{Q}^m\\&=-\frac{3\boldsymbol{Q}^m-4\boldsymbol{Q}^n+\boldsymbol{Q}^{n-1}}{2}-\Delta t\boldsymbol{R}^m\end{aligned} \tag{2}$$

其中:

$$\begin{aligned}\boldsymbol{L}&=-\alpha(\boldsymbol{A}^+_{i-1,j,k}+\boldsymbol{B}^+_{i,j-1,k}+\boldsymbol{C}^+_{i,j,k-1})\\\boldsymbol{D}&=\frac{3}{2}\boldsymbol{I}+\alpha\chi(\sigma_A+\sigma_B+\sigma_C)\boldsymbol{I}+2\alpha\nu\boldsymbol{I}\\\boldsymbol{U}&=\alpha(\boldsymbol{A}^-_{i+1,j,k}+\boldsymbol{B}^-_{i,j+1,k}+\boldsymbol{C}^-_{i,j,k+1})\\\alpha&=\frac{\Delta t}{Vol_{i,j,k}},\chi=1.01\end{aligned} \tag{3}$$

对流项采用 Roe, HLLE, AUSM 等格式离散,通过 MUSCL 插值达到二阶精度; 粘性项采用二阶中心格式; 湍流模型采用 $k-\omega$ 两方程模型。

为便于编程,边界条件的处理统一采用虚拟网格技术。图 1 为二维虚拟网格示意图,实线代表计算网格,虚线代表虚拟网格。每块网格迭代计算之前,要先处理边界条件,即根据不同的边界类型把相应数值赋入虚拟网格。这样,在计算无粘通量、粘性通量时可以不考虑边界的存在,边界和场内采用统一的离散格式,从而避免复杂的边界特殊处理。

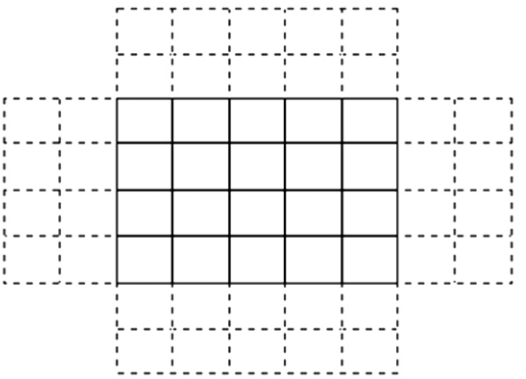

图 1　二维虚拟网格示意图

Fig. 1　Two dimensional virtual grid

2　数据传输方法

多块结构网格并行计算过程中,数据传递的主体为几何相邻的各网格块之间的边界信息交换。首先把各网格块分配到各进程,为保证负载平衡,应使每个进程分配的总网格单元数尽量相同,且几何相邻的网格块尽量分配到同一进程以减少数据传输量。具体负载平衡方法将在下一节讨论。图 2 显示了任意进程 i 和进程 j 的网格块分布情况,其中阴影部分代表该进程所分配网格块。类似于串行程序,并行程序边界条件处理也是对相应网格块的虚拟网格进行赋值,但后者由于几何相邻的两网格块可能被分配于不同进程,因此,并行程序的边界条件处理一般要通过进程间数据传输完成。数据传输方法就是负责建立各进程所属网格块间简单、通用的映射关系,并保证数据在此映射关系下安全、高效传输。本文针对多块结构网格,提出了一种通用数据传输方法,该方法稍作修改也可推广到非结构网格情况。具体实现过程为:

Step 1. 以进程为单位,统计需要发送到其它进程和从其它进程接收的数据总量。比如,进程 i 发送到进程 j 的数据量放在 exch_recv(j).num 中,进程 i 接收进程 j 的数据量放在 exch_send(j).num 中。

Step 2. 建立映射关系。对单一进程中所有网格边界循环,若标记为内边界,则记录其对应相邻网

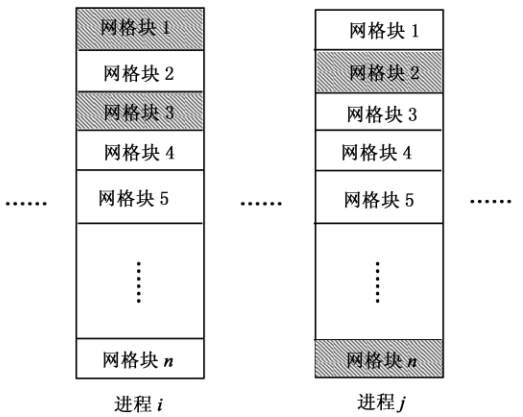

图 2 进程及网格块分布示意图

Fig. 2 Grid block distribution of different process

格的拓扑信息(块号 iblk 及 i, j, k 索引)。比如,进程 i 发送到进程 j 的拓扑信息存放在 exch_recv(j).posi(4, exch_recv(j).num) 中。搜索完毕,各进程传递此拓扑关系数组,从而建立固定的映射关系。

Step 3. 数据发送与接收。按 Step2 中的循环顺序,对需要发送的变量打包。比如,进程 i 发送到进程 j 的变量存放在 exch_recv(j).values(7, exch_recv(j).num) 中。然后调用 MPI 发送与接收函数实现进程间数据交换。数据交换完毕,每个进程按照 Step2 建立的映射关系完成虚拟网格边界赋值。由于所处理的网格量一般比较大,故采用缓冲方式,即先把数据发送到事先开辟的缓冲区中,具体发送接受过程在后台自动完成。

该数据传输方法可以编写成相应的数据传输模块,用比较小的工作量就可将串行程序改为并行程序。经测试,当计算网格量约 2000 万、进程数为 32 时,该并行程序在深腾 1800,Dell M600 及曙光 4000A 上都可以稳定运行。

3 基于遗传算法的负载平衡方法

并行计算的负载平衡问题在众多文献中均有讨论[3-8]。影响负载平衡的主要因素有两个:每个进程的计算量和进程间传输数据的通信时间。随着现代集群网络速度提升,传输数据量大小在一定范围内对通信时间影响不大,而且进程间通信时间占整个计算时间比例一般很小。因此本文分两步:先抓主要矛盾,即考虑每个进程计算量的平衡,也就是保证每个进程分配的网格单元数相当。满足该条件的解不止一个,将其都归入指定集合中。然后对该集合中所有解考虑网格块间几何相邻关系,从中选出最优解。

3.1 进程计算量平衡

对于多块结构网格,由于每个网格块网格单元数不同,有时甚至相差很大,因此要保证每个进程分配的网格单元数相当,必须找到一个合理的网格块组合形式。这是一个组合优化问题,该组合问题的搜索空间随网格块数和进程数增加而急剧增加,对于常规的优化方法而言,存在诸多计算困难。近几十年发展起来的遗传算法是一种模拟自然界优胜劣汰的进化算法,具有高效的全局搜索能力。因此本文采用遗传算法对网格块进行组合优化,从而达到进程计算量平衡目的。

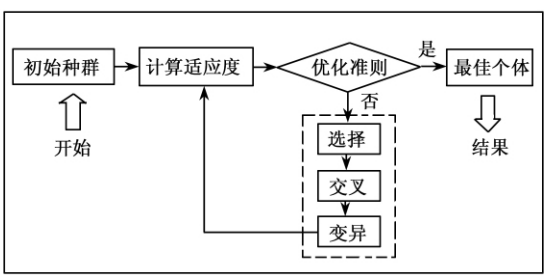

图 3 遗传算法流程

Fig. 3 Flow chart of genetic algorithm

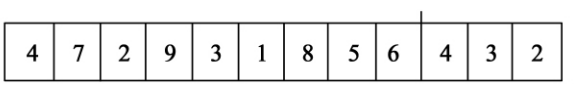

图 4 两段式染色体

Fig. 4 Two-part chromosome

常规遗传算法[9]基本流程如图 3 所示。图 3 中方框内的选择、交叉、变异为遗传算法的核心。然而,常规遗传算法的染色体编码一般采用二进制或实数编码,对于本文的组合优化问题是不适用的;其次,还要考虑不同进程对应的网格块序号和总网格块数也不一样。因此,参考文献[10],本文采用如图 4 所示的两段式染色体。染色体的前段是所有网格块序号的一个排列,后段是分配于各个进程的网格块总数。图 4 所示的染色体可以表示 9 块网格分配于 3 个进程的情况。其中,进程 1 分配 4 块网格,网格序号为 4,7,2,9;进程 2 分配 3 块网格,网格序号为 3,1,8;进程 3 分配 2 块网格,网格序号为 5,6。采用此两段式染色体,前段染色体编码的一个约束条件是:编码中不允许有重复的基因码,即不允许同

一块网格被重复分配。为满足此要求,本文的交叉算子采用 Goldberg 提出的部分匹配交叉[9](PMX)。例如,对于下面两个父个体,随机选择两个交叉点"|"。

父代 1　(4 7 2 | 9 3 1 | 8 5 6)

父代 2　(2 5 9 | 7 8 4 | 6 3 1)

首先,交换交叉点间的中间段,得到:

子代 1　(x x x | 7 8 4 | x x x)

子代 2　(x x x | 9 3 1 | x x x)

得到中间段的映射关系,有:

$$7\leftrightarrow 9,\ 8\leftrightarrow 3,\ 4\leftrightarrow 1$$

然后,对子代 1,子代 2 中的 x 部分,分别保留从其父个体中继承未选定的网格块序号 2,5,6 得到:

子代 1　(x x 2 | 7 8 4 | x 5 6)

子代 2　(2 5 x | 9 3 1 | 6 x x)

最后,根据中间段的映射关系,对于上面子代 1 中第一个 x,使用父代码 1 的 4,由 $4\leftrightarrow 1$,交换得到第一个 x 为 1,其余 x 类似。最终的子个体为:

子代 1　(1 9 2 | 7 8 4 | 3 5 6)

子代 2　(2 5 7 | 9 3 1 | 6 8 4)

对于变异操作,同样应满足编码中不允许有重复基因码的要求。因此本文采用逆位遗传算子[9]或交换遗传算子[9]。例如,对下面的父个体,随机选择两个变异点"|",两点间的编码逆序排列或两点交换编码。

父代　(4 7 2 | 9 3 1 8 | 5 6)

逆位算子　(4 7 2 | 5 8 1 3 | 9 6)

交换算子　(4 7 2 | 5 3 1 8 | 9 6)

对于后段染色体,只在变异操作时发生改变。唯一的约束条件是: 所有基因位的总和等于网格总块数。

适应度函数取为:

$$F_{fit} = \sum_{i=1}^{N_p} \mathrm{abs}\left(X_{\mathrm{avg}} - \sum_{j=1}^{N_i} X_{ij}\right) \tag{4}$$

其中,N_p 为总进程数,N_i 为第 i 个进程所分配的网格块总数,X_{avg} 平均每个进程的网格单元数,X_{ij} 为进程 i 中第 j 块网格单元数。

此外,为保证该算法的稳定性,本文引入子种群概念,并在每个子种群中引入精英保留策略。即将整个种群划分为多个小的子种群,每个子种群内执行选择、交叉、变异遗传操作,在迭代过程中保留每个子种群中的最佳个体,并传递到下一代。每次迭代搜索完成后,遍历种群中适应度靠前个体,若该个体满足平衡准则,则将其记录于一解集中,并保证最后得到的解集个体不重合。平衡准则为:

$$F_{\mathrm{bal}} = \max\left(\mathrm{abs}\left(X_{\mathrm{avg}} - \sum_{j=1}^{N_i} X_{ij}\right)/X_{\mathrm{avg}},\ i \in [1, N_p]\right) \tag{5}$$

其中各变量意义与(4)式相同。本文取 $F_{\mathrm{bal}} \leqslant 2.0\%$。

值得一提的是,在组合优化开始前,如果某网格块网格单元数大于平均网格单元数,需先将此网格块分割为若干小网格块,且尽量保持分割面最小。该过程只涉及新数组的开辟及相关赋值操作,对于复杂和简单网格都可以通过程序自动实现。

3.2　网格块间几何相邻关系评价

对于多块结构网格,每网格块有六个边界面,如果我们定义每个边界面所含网格单元数为其"面积",那么网格块间几何相邻关系可用如下面积函数进行综合评价:

$$A = A_t - \sum_{i=1}^{N_p} A_i \tag{6}$$

其中,A_t 为所有网格块内边界面积,A_i 为每个进程中由于网格块几何相邻而重合的面积。计算 3.1 节得到的解集中所有个体面积函数,函数值最小者即为所求最佳个体,即该个体每个进程网格块间几何相邻程度综合评价最高。

3.3　平衡算法有效性验证

为考察该算法的有效性,取一实际算例对其进行了验证。该算例中,网格块总数为 344,网格单元总数为 6410068,每块网格单元数随网格块序号分布情况见图 5。由图 5 可见,每块网格的网格单元数分布极不均匀,此类组合问题的计算难度随进程数增加而显著增加,因此对于 16 进程 F_{bal} 取0.5%,而 32 进程 F_{bal} 取 2%。表 1 和表 2 分别给出了进程数为 16 和 32 的组合计算结果。表 1 是从满足 $F_{\mathrm{bal}} \leqslant 0.5\%$ 平衡准则的 60 个解中按面积 A 由大到小选取的四个 Case,面积 A 最小的 16-Case4 为最优结果。表 2 是从满足 $F_{bal} \leqslant 2.0\%$ 平衡准则的 40 个解中选取的四个 Case,32-Case4 为最优结果。所有情况 F_{bal} 取值均不大,对于并行计算负载平衡而言,可以认为该结果已达到最优组合。4.2 节将详细给出利用此两组优化组合的实际计算效率比较。

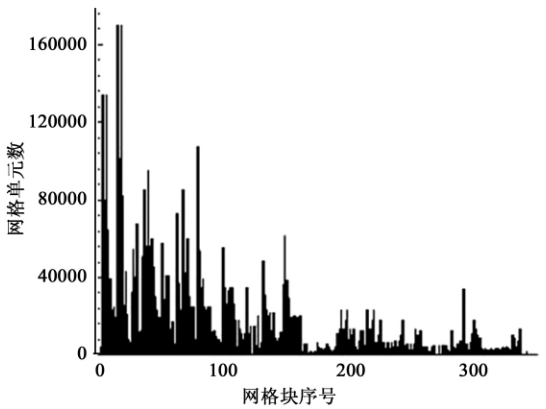

图 5 网格单元数随网格块序号分布情况

Fig. 5 Grid cell number VS grid block number

表 1 16 进程组合结果

Table 1 Grid combination results of 16 processes

	16 – Case1	16 – Case2	16 – Case3	16 – Case4
F_{bal}	0.0052%	0.0082%	0.0387%	0.0172%
面积 A	1098948	1093474	1083210	1077852

表 2 32 进程组合结果

Table 2 Grid combination results of 32 processes

	32 – Case1	32 – Case2	32 – Case3	32 – Case4
F_{bal}	0.991%	0.706%	0.165%	1.110%
面积 A	1134940	1129102	1118152	1106734

4 算例验证

本文以 AIAA DPW(Drag Prediction Workshop)标准计算模型对开发的并行程序进行了准确性及并行效率验证。

4.1 DLR-F6 翼身组合体

计算条件:来流马赫数 0.75,雷诺数 3.0×10^6,采用 $k-\omega$ 两方程模型。用 32 个 CPU 对攻角分别为 $-0.304°$,$0.49°$和 $1.23°$三种工况进行了计算,并比较了攻角对升力系数的影响。

图 6 给出了三种工况下升力系数随攻角变化曲线。可以看出,计算结果比实验结果略偏高,但这与该 Workshop 官方网站公布的 STAR – CCM + 及 CFL3D 等代码的计算结果相近。出现上述偏差的原因现在还没有定论,但本文并行程序计算结果与原串行程序一致,说明偏差不是并行引起。

本文还利用该并行程序作了三种工况下网格收敛性验证。所用稀网格为 340 万,中等网格为 570 万,密网格为 1000 万。表 3 列出了各种情况下的阻力系数和升力系数。随网格加密,阻力和升力系数有减小的趋势,但减小的幅度并不大。

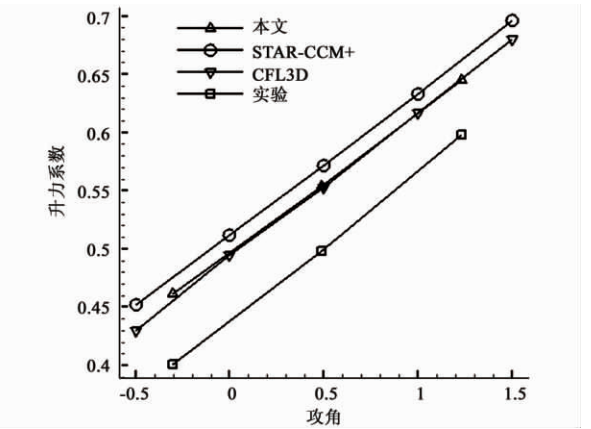

图 6 升力系数随攻角的变化

Fig. 6 Lift coefficient at different angle of attack

表 3 不同攻角下阻力系数与升力系数随网格稀疏的变化

Table 3 Drag and lift coefficient for different angle of attack and different grid cell number

攻角		稀网格	中网格	密网格
$-0.304°$	C_d	0.0307	0.0305	0.0303
	C_l	0.4626	0.4621	0.4611
$0.49°$	C_d	0.0348	0.0346	0.0344
	C_l	0.5553	0.5546	0.5534
$1.23°$	C_d	0.0404	0.0401	0.0401
	C_l	0.6441	0.6450	0.6447

4.2 DLR-F6 翼身组合体加发动机舱

本算例几何外形在 4.1 算例基础上加了发动机舱,图 7 显示了发动机舱附近的表面网格。计算工况为来流马赫数 0.75,雷诺数 3.0×10^6,攻角 $0.19°$,同样采用 $k-\omega$ 两方程模型。图 8 为表面压力分布情况,图 9 给出了沿翼展方向取 6 个不同截面的压力系数与实验对比。图中实线为计算结果,方块为实验数据。其中 $Y/B=0.331$ 截面处上下表面压力系数出现了交错,说明此处发动机对机翼的影响比较强烈。

为验证并行程序效率,针对该算例测试了加速比和并行效率。测试平台为深腾 1800,计算网格为 3.3 节实验网格,测试进程数分别取 1,2,4,8,16 和 32。表 4 列出了程序定常迭代 500 步的测试结果。加速比 S 和并行效率 E 的定义公式为:

$$S=\frac{T_1}{T_n},\quad E=\frac{S}{n}\times100\% \tag{7}$$

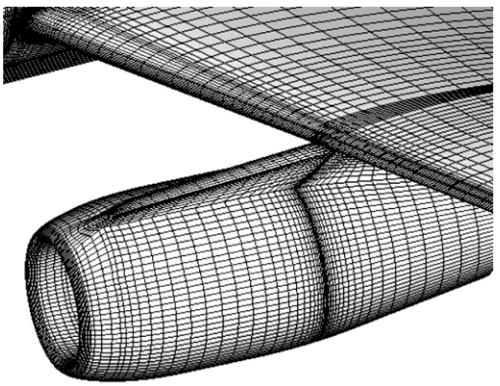

图 7　发动机舱附近表面网格

Fig. 7　Surface grid near the nacelle

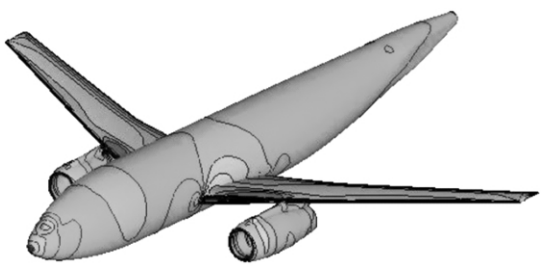

图 8　表面压力等值线

Fig. 8　Surface pressure contours

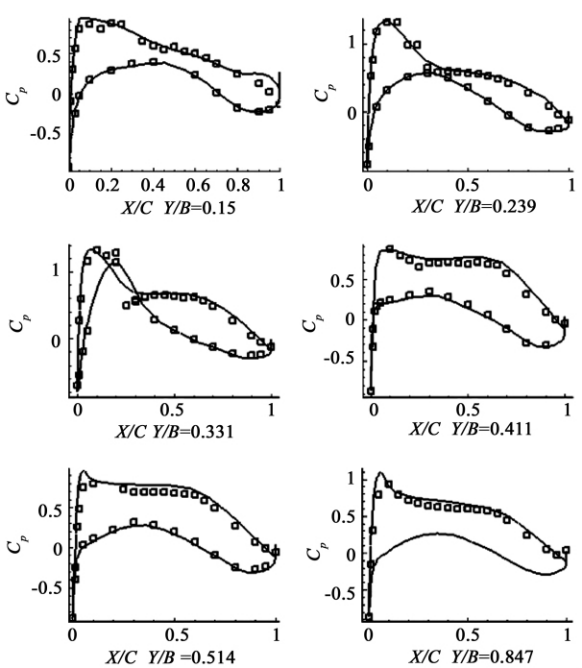

图 9　截面压力系数

Fig. 9　Pressure coefficient of different sections

其中，T_1 为一个进程的计算时间，T_n 为 n 个进程的计算时间。

表 4　并行加速比及并行效率

Table 4　Speed up and efficiency

	T_t/s	T_s/s	T_r/s	R	S	E
1			28532			
2	312	0.62	14707	0.02	1.94	97.0%
4	235	0.47	7399	0.03	3.86	96.4%
8	212	0.42	3991	0.05	7.15	89.4%
16	172	0.34	2225	0.08	12.82	80.1%
32	115	0.23	1266	0.09	22.54	70.4%

表 4 还统计了数据传输总时间 T_t，平均每迭代步数据传输时间 T_s，程序总运行时间 T_r 及数据传输时间占程序总运行时间比例 R。可见，随进程数目增加，单个进程所分配网格单元数线性减小，计算时间也基本线性减小，但数据传输时间减小并不明显，于是数据传输时间在整个计算中所占比例显著增大，故并行效率随进程数增加而明显降低。以 32 进程为例，单个进程分配的网格单元大约 20 万，其计算时间与数据传输时间可以比拟，并行效率只有 70% 左右。若 CPU 速度更快，并行效率也将降低，因此，并行效率与网格规模及 CPU 速度也有关系。网格规模及 CPU 速度确定后，存在一个可保证并行效率的最佳进程数以合理利用计算资源。

同时，为考察网格块几何相邻关系(即面积 A) 对并行效率的影响，利用表 1 和表 2 提供的组合数据对 16 和 32 进程各进行了测试计算，详细统计结果列于表 5。由表 5 可见，数据传输时间随面积 A 减小而下降，但下降幅度并不大，可以认为基本不变。因此，网格块几何相邻关系对并行效率影响很小。

表 5　16 和 32 进程详细测试结果

Table 5　Detailed test results of 16 and 32 processes

	T_t/s	T_s/s	T_r/s	E
16 – Case1	190	0.380	2191	81.4%
16 – Case2	186	0.372	2180	81.8%
16 – Case3	182	0.364	2189	81.5%
16 – Case4	172	0.344	2225	80.1%
32 – Case1	126	0.252	1270	70.2%
32 – Case2	123	0.246	1286	69.3%
32 – Case3	122	0.244	1263	70.6%
32 – Case4	115	0.230	1266	70.4%

5　结　论

本文提出了针对多块结构网格的并行计算数据传输及负载平衡方法，并发展了相应并行计算程序。

该负载平衡方法可以考虑进程网格量的平衡及网格块间几何相邻关系两个因素。数值实验表明,进程网格量平衡对并行效率影响敏感,而网格块几何相邻关系对其影响比较小。

参 考 文 献

[1] 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, 2003: 10 - 11.

[2] Yang G, Kondo M, Obayashi S. Multiblock navier-stokes solver for wing/fuselage transport aircraft [J]. JSME International Journal, 2002, 45(1): 85 - 90.

[3] Wang B, Zha G C. A general sub-domain boundary mapping procedure for structured grid cfd parallel computation [C]. 25th AIAA Applied Aerodynamics Conference, Miami, USA, June 25 - 28, 2007.

[4] 司海青, 王同光. 多块并行计算中负载平衡策略及时间成本估算方法[J]. 航空学报, 2007, 28(z1): 57 - 61. [Si Hai-qing, Wang Tong-guang. Load balancing strategy for parallel calculation and time cost estimation [J]. Acta Aeronautica et Astronautica Sinica, 2007, 28(z1): 57 - 61.]

[5] 许正, 李津, 朱自强. 网络连接机群上CFD计算的一种负载平衡方法[J]. 航空学报, 2005, 26(2): 129 - 134. [Xu Zheng, Li Jin, Zhu Zi-qiang. Load balancing strategy for parallel cfd calculation on cluster [J]. Acta Aeronautica et Astronautica Sinica, 2005, 26(2): 129 - 134.]

[6] 李桦, 王承尧, 王正华. PVM环境下提高并行计算加速比的数值实验研究[J]. 宇航学报, 1999, 20(1): 88 - 91. [Li Hua, Wang Cheng-yao, Wang Zheng-hua. Numerical research of raising speedup of parallel calculation on PVM [J]. Journal of Astronautics, 1999, 20(1): 88 - 91.]

[7] Rantakokko J. Partitioning strategies for structured multiblock grids [J]. Parallel Computing, 2000, 26(12): 1661 - 1680.

[8] Thune M. Partitioning strategies for composite grids [J]. Parallel Algorithms and Applications, 1997, 11: 325 - 348.

[9] 王小平, 曹立明. 遗传算法—理论、应用与软件实现[M]. 西安: 西安交通大学出版社, 2002.

[10] 欧阳杰平. 使用遗传算法解决MTSP问题的一种新的染色体设计[J]. 舰船电子工程, 2006, 26(3): 107 - 109. [Ouyang Jie-ping. A new approach to solving the multiple traveling salesperson problem using genetic algorithms [J]. Ship Electronic Engineering, 2006, 26(3): 107 - 109.]

作者简介: 李桂波(1982 -),男,博士研究生,研究方向为计算流体力学、计算气动弹性。

通信地址: 北京市海淀区北四环西路15号中国科学院力学研究所高温气体动力学重点实验室(100190)

电话: (010) 82544010

E-mail: liguiboo@hotmail.com

(编辑: 沃云峰)