非结构化四边形网格生成新算法

陈建军[1),2)] 郑 耀[1),2)] 陈立岗[1)] 梁 义[1)]

[1)](浙江大学工程与科学计算研究中心，杭州 310027) [2)](浙江大学航空航天学院，杭州 310027)

摘 要 改进了一类基于递归区域分解过程的四边形网格生成算法。引入一套健壮的网格模板，为子域的网格剖分提供统一的处理方案，不再限制最终子域为 4 节点、6 节点或 8 节点子域，提高了算法的时空效率。结合新的子域网格生成过程和自动区域分解算法，利用背景网格和网格源控制分解线上点的布置，得到一个全自动的非结构化四边形网格生成算法。最后通过网格及数值模拟实例验证了算法性能和实用性。

关键词 网格生成 非结构化网格 四边形单元 网格模板

中图法分类号：O242.21 **文献标识码**：A **文章编号**：1006-8961(2008)09-1796-08

A New Unstructured Quadrilateral Mesh Generation Algorithm

CHEN Jian-jun[1),2)], ZHENG Yao[1),2)], CHEN Li-gang[1)], LIANG Yi[1)]

[1)] (*Center for Engineering and Scientific Computation, Zhejiang University, Hangzhou* 310027)

[2)] (*School of Aeronautics and Astronautics, Zhejiang University, Hangzhou* 310027)

Abstract A quadrilateral mesh generation scheme using recursive domain bi-divisions is enhanced in this paper. Robust mesh templates are designed, to provide a unified way for sub-domain meshing. The templates pose no restrictions on the numbers of boundary nodes (e.g. six or eight) of sub-domains, and can mesh the 'big' regular sub-domains bounded linearly in many cases, thus to highly improve the timing and storage performance of the algorithm. Coupling the new sub-domain mesh generation procedure with an automatic domain decomposition procedure, we obtain a new automatic unstructured quadrilateral mesh generation algorithm. Finally, mesh examples and simulation results are presented to demonstrate the capabilities of the new algorithm.

Keywords mesh generation, unstructured mesh, quadrilateral elements, mesh templates

1 引 言

平面或曲面特定问题分析中，由于计算结果的精度高于三角形单元，四边形单元往往被优先采用，但其网格生成算法也更为复杂。现有的四边形单元网格生成方法主要可以分为两类：间接法和直接法[1]。间接法在区域内部预先生成三角形单元，随后通过单元分解或者合并生成全四边形单元网格或者三角形和四边形的混合单元网格。直接法包括映射法、基于栅格法和前沿推进法等，其中以前沿推进法(AFT)应用最为广泛，其边界适应能力好、网格质量较高，但也存在实现复杂、浮点运算量大等缺点。

基于分治法思想，Talbert 等人提出一个新的全四边形单元网格生成的思路[2]：通过递归分解将问题域划分成多个满足要求的子域；随即按照一定规则在子域中完成网格剖分；最后合并子域网格。Talbert 等人[2]要求最终子域需为 6 节点区域，并根据 6 节点子域的内角特性将其划分成 4 类 13 种原型分别处理。Chae 等人增加了 8 节点子域 5 类共

基金项目：国家自然科学基金重大研究计划(90405003)；国防基础科研项目(B1420061036)；浙江大学国防预研基金(128200-8112D1)

收稿日期：2007-07-23；**改回日期**：2008-01-16

第一作者简介：陈建军(1979～)，男。讲师，博士后。当前主要研究方向为几何网格生成及其并行化、并行自适应求解环境、高性能计算。E-mail：chenjj@zju.edu.cn

15 种原型以及层状原型[3]。Nowottny 则用简化后的 AFT 算法在三角形或四边形子域中生成网格[4],遗憾的是,该算法在子域分解到只剩下 6 个节点时,仍需要将子域分成 3 类 7 种原型以完成最后的网格封闭计算。Sarrate 等人发展了上述算法,利用背景网格控制网格的疏密,并将该算法应用于自适应分析中[5]。

上述算法存在两个缺陷:一是最终子域大都是 6 节点或 8 节点子域,过于狭小,致使子域分解层次过深,不仅降低了分解效率,且过多的附加约束会影响到最终网格质量的提高;二是基于子域内角等特性的子域分类规则过于繁琐,子域种类过多,剖分规则复杂,使得算法实现难度增大,且其中牵扯到大量的浮点运算,增加了算法的时间消耗。

注意到利用直线分解区域,在区域内部会形成一些较规则的几何形体,若能正确地识别这些区域为"四边形"或"三角形",随后利用健壮有效的网格模板完成子域网格生成[6],可以有效地缓解上面提及的两个问题。另一方面,在应用模板法的网格生成器[6~8]中引入自动的拓扑分解及几何分解算法后,可以消除原先的手工分解过程,得到一个全自动的四边形单元网格生成器。

2　网格生成流程

假定初始的几何区域由一条逆时针转向的外环和 n 条顺时针转向的内环所描述,在其中序列化生成网格需要 5 个主要环节,如图 1 所示。

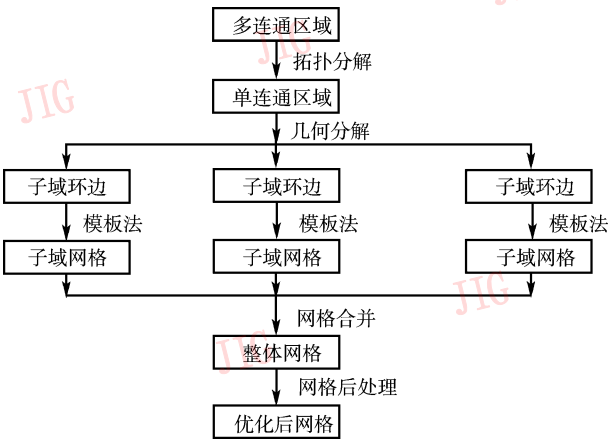


图 1　序列化网格生成流程图

Fig. 1　Flowchart of the serial mesh generation algorithm

(1)拓扑分解　选择"最佳"的 n 对不同环上的节点,连接多连通区域为单连通区域。拓扑分解可以类似几何分解[2~5,9]一样自动完成,也可手工完成,且一般手工工作量不大。

(2)几何分解　选择待分解区域环上"最佳"的两个可视点,分解其为两个子域。对子域重复几何分解过程,直至其满足利用网格模板直接生成网格的条件。本文借用了 Sarrate 等人给出的自动分解算法[5],但引入了网格源[9,10]作为局部网格密度控制工具,改进了在分解线上布置新点的策略。

(3)子域网格生成　利用网格模板生成子域网格。

(4)网格合并　剔除几何分解线和拓扑分解线上的重复网格信息。

(5)网格后处理　通过拓扑优化及光滑化,提高整体网格质量。

图 2 通过一个实例说明了上述 5 个环节,需要注意的是,整个网格生成中不需要任何人工干涉,包括拓扑分解环节也是自动完成的。

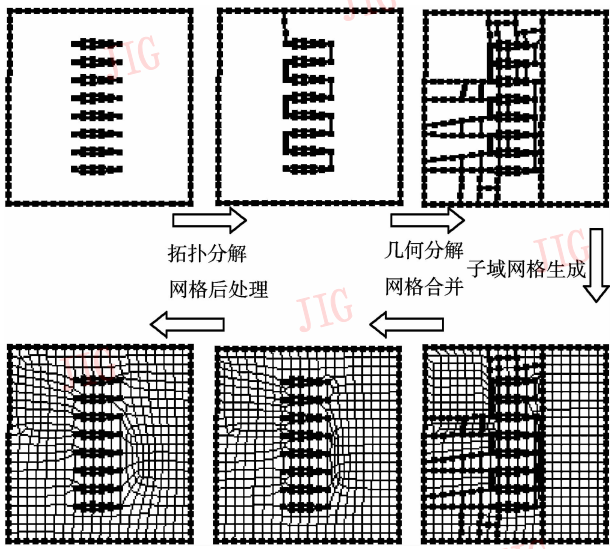


图 2　序列化网格生成的 5 个环节

Fig. 2　Schematic view of the five steps of the algorithm

3　分解线内新节点的生成

如何在分解线内产生点对最终网格的疏密控制至关重要。Talbert 等人基于单元尺寸场在分解线内线性变化的假设,根据分解线两个端点的单元尺寸值在分解线内布点[2],但当假设不成立时,该方法将给出不恰当的节点分布。

除此以外,递归中点布点的方法也经常使用[11,12]。但该算法会引入较大的节点分布误差。这种误差在分解一条分解线时也许可以容许,但递归分解过程会放大这种误差效应,导致最终网格的疏密效果和用户的需要差异较大。

Sarrate 等人以背景网格为密度控制方式,通过理论推导提出了一种节点分布策略[5],其主要思想是利用分解线和背景网格的交点将分解线分段,并假设每一段中单元尺寸呈线性过渡。遗憾的是,该算法很难推广到密度控制方式包含网格源的网格生成算法中。

以减少节点分布误差为目标,基于贪婪思想,本文提出了一种启发式算法,它包含两个环节:

(1)插入新节点 从单元尺寸值较小的分解线端点 a_0 出发,向另一个端点 b 推进,逐个插入点。定义第 i 次插入的点为 a_i,且有 $|a_{i-1}a_i| = h(a_{i-1})$。当最新插入的点越过端点 b,且子域的边界节点数目为偶数(以保证生成全四边形单元)时,停止插点。比较此时的节点分布误差绝对值和放弃最后两个新节点情形下的节点分布误差绝对值,选择误差绝对值较小的方案。

(2)调整新节点位置 记新插入的 n 个点为 $a_1,\cdots,a_n$,两个端点为 a_0 和 a_{n+1}($a_{n+1}=b$)。调整 $a_1,\cdots,a_n$ 的位置,使得:

$$|a_{i-1}a_i| = 0.5(h(a_{i-1})+h(a_i))\,l / l_{\text{ideal}} \quad (i=1,\cdots,n+1)$$

其中,l 为分解线实际长度,l_{ideal} 为分解线理想长度。

$$l_{\text{ideal}} = 0.5(h(a_0) + h(a_{n+1})) + h(a_1) + \cdots + h(a_n)$$

为证明新节点分布策略的合理性,图 3 给出了一个实例。以连接两个内环的拓扑分解线为例,用户通过定义背景网格,希望两端密、中间疏。利用本文算法得到的新节点分布情形,完全符合用户的期望(图 3)。而若利用 Talbert 等人的算法[2],新节点将沿分解线等间距分布,这显然是不正确的。

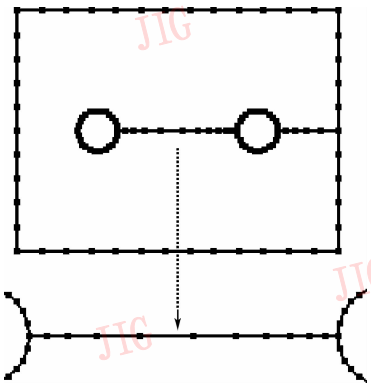


图 3 中间需粗化的分解线新节点分布示例

Fig. 3 Distribution of inner mesh nodes in a splitting line with the middle part coarsened

4 子域网格生成及网格后处理

4.1 最终子域的形状要求

最终子域指那些能直接利用网格模板进行网格生成的子区域,它的形状要求定义了递归几何分解过程的终止条件。根据所使用网格模板的不同,这种要求是变化的。本文算法中,称满足条件 1 或同时满足条件 2 ~ 5 的子域为最终子域:

条件 1 边界节点数为 4 或 6;

条件 2 内角为 120°的边界节点数目为 3 或 4,这些节点被称为角点;

条件 3 没有内角大于 $180+\mu_1$ 的边界节点;

条件 4 如果在两个相邻角点之间出现多个连续凹角,凹角和平角的差值和小于 μ_1;

条件 5 如果在两个相邻角点之间出现多个连续凸角,平角和凸角的差值和小于 μ_2。

拥有 3 个或 4 个角点的子域分别被称为三角形子域和四边形子域。数值实验表明参数 μ_1 和 μ_2 的值可设为 15。

4.2 网格模板

设 N_1、N_2、N_3 和 N_4 对应单位正方形 4 条边界的分段数。为保证生成全四边形单元,有:

$$(N_1 + N_2 + N_3 + N_4)\%2 = 0$$

旋转正方形,总可以使得:

$$(N_1 - N_4) \geqslant |N_2 - N_3|$$

图 4 和图 5 分别给出了 $N_4 \neq 1$ 和 $N_4 = 1$ 时单位正方形的网格模板[6~9]。式(1)和式(2)则分别给出了这两种情形下模型参数的计算公式,其中,N_{el} 表示网格中包含的单元总数。

$$\begin{cases} K_1 = (N_1 - N_2 + N_3 - N_4 + 2)/2 \\ K_2 = (N_1 + N_2 - N_3 - N_4 + 2)/2 \\ K_3 = N_1 - K_1 - K_2 = N_4 - 2 \\ K_4 = K_1 + K_5 = K_2 + K_6 \\ K_5 = N_2 - K_7 - 1 \\ K_6 = N_3 - K_7 - 1 \\ K_7 = \min((N_2 - 1) * (N_3 - 1), 1) \end{cases} \quad (1)$$

单位正方形内的网格布置完成后,可利用超限映射函数[13]将其映射到四边形区域内。三角形子域可分解成一个四边形子域和若干个四边形单元的组合[6,9],其网格模板无需特别设计。

$$\begin{cases} K_1 = \begin{cases} 0 & N_3 - N_2 = N_1 - 1 \\ N_2 - K_4 & \text{其他} \end{cases} \\ K_2 = \begin{cases} 0 & N_2 - N_3 = N_1 - 1 \\ N_3 - K_4 & \text{其他} \end{cases} \\ K_3 = N_1 - K_1 - K_2 \\ K_4 = \begin{cases} 0 & \max(N_2, N_3) - \\ & \min(N_2, N_3) = N_1 - 1 \\ \min(N_2, N_3) - 1 & \text{其他} \end{cases} \end{cases} \tag{2}$$

图 6 给出了使用这两类网格模板形成的网格示例,图 6(a)和 6(b)对应网格疏密在纵向作单向过渡时的情形,而图 6(c)对应网格疏密在横向和纵向均存在过渡时的情形。可以看出,本文给出的网格模板有效地解决了传统的映射法很难形成过渡网格的缺陷。

4.3　最终子域的形状要求

本文给出的最终子域形状要求不再限制子域的边界节点数目为 4、6 或者 8,使得区域中间那些由分解直线描述的“大”子域能直接利用网格模板生成网格,从而有效地降低递归几何分解的深度。但在曲线边界附近,仍会形成较多 4 节点、6 节点或 8 节点子域。

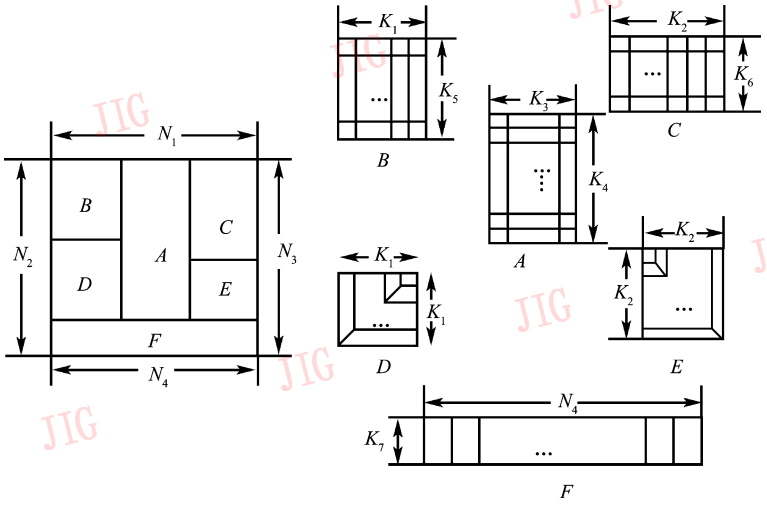


图 4　$N_4 \neq 1$ 时单位正方形的网格模板

Fig. 4　Mesh template for the unit square when $N_4 \neq 1$

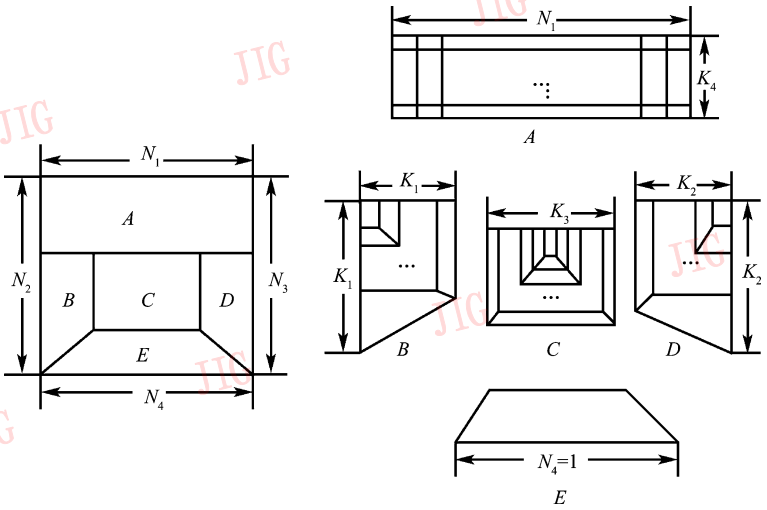


图 5　$N_4 = 1$ 时单位正方形的网格模板

Fig. 5　Mesh template for the unit square when $N_4 = 1$

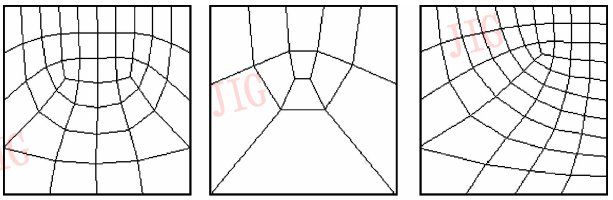

(a)单向过渡：$N_4 \neq 1$ (b)单向过渡：$N_4=1$ (c)双向过渡：$N_4 \neq 1$

图 6 利用四边形区域网格模板生成的过渡网格实例

Fig. 6 Graded mesh samples generated with quadrilateral mesh templates

对于 4 节点子域，可将其处理成 $N_1=N_2=N_3=N_4=1$ 的四边形。对于 6 节点子域，Talbert 等人讨论了 13 种子情形去处理，在这里我们利用一些简单的规则识别其为四边形或三角形。定义两个子域边界节点 V_1 和 V_2 的拓扑长度 $d_t(V_1, V_2)$ 为从 V_1 出发，逆时针遍历边界节点序列至 V_2 所经过的边界节点总数，且 $d_t(V_1, V_1)=0$。下面给出规则的定义：

R1 如果子域包含凹角，V_1 是内角最大的节点，V_2 满足 $d_t(V_1, V_2)=3$；定义除 V_1 和 V_2 以外的其他 4 个节点为角点，识别子域为四边形。

R2 如果子域中内角在 180°附近的节点数目为 3，定义其他 3 个节点为角点，识别子域为三角形。

R3 如果子域中内角在 180°附近的节点数目小于 3，定义 4 个内角最小的节点为角点，识别子域为四边形。

对于 8 节点子域，仍然利用 4.1 节给出的最终子域形状要求来识别它，但此时 μ_1 和 μ_2 的值被放大成一个很大的数，如 1.0×10^{10}，以完全消除条件 3～5的约束。

4.4 最终子域的大小

从图 6 给出的网格实例可以看出，本文给出的网格模板对于单元尺寸在区域内部呈线性过渡的规则几何体有很好的适用性。但若区域内部单元尺寸呈非线性过渡时（如区域内部定义了网格源），这些网格模板是不能直接使用的，而需要继续分解该区域。实践中，如何检测单元尺寸变化的性质是个技术难题，为简化处理，用户可通过设定子域边界数目的最大值 g_{max} 来规避上述问题。

图 7(a)给出了一个边界节点均匀分布的正方形，其中心定义了一个点源，以期加密中心附近的网格。显然，该正方形不能直接套用图 4 给出的网格模板，解决的办法是取 $g_{max}=30$。图 7(b)和图 7(c)分别给出了相应的子区域结构和最终网格，可以看出最终网格很好地满足了用户在区域中心附近加密网格的需求。

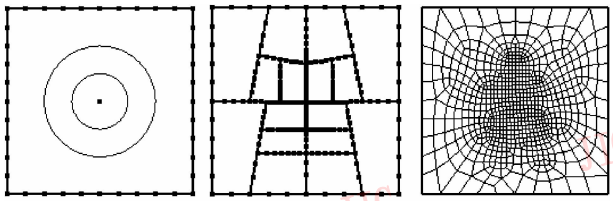
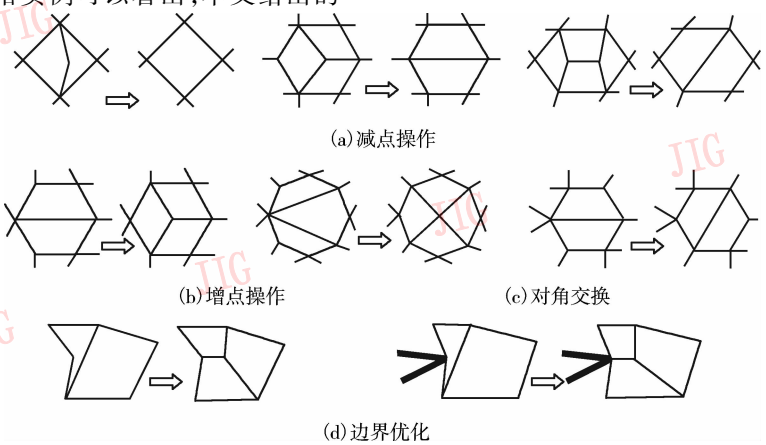

(a)边界离散及网格源设置 (b)子区域结构 (c)最终网格

图 7 中心区域需加密的正方形的网格生成方案

Fig. 7 A regularly bounded square with the center region refined

4.5 网格后处理

合并所有子域网格可形成一个全局网格，为提高其质量，需要采取一系列后处理措施。由于本文算法产生的全局网格可能包含很多度值（指包含节点的单元数）不理想的节点，需要在网格光滑化前，进行拓扑优化。图 8 给出了我们实现的 4 类拓扑优化操作[14]。


(a)减点操作

(b)增点操作 (c)对角交换

(d)边界优化

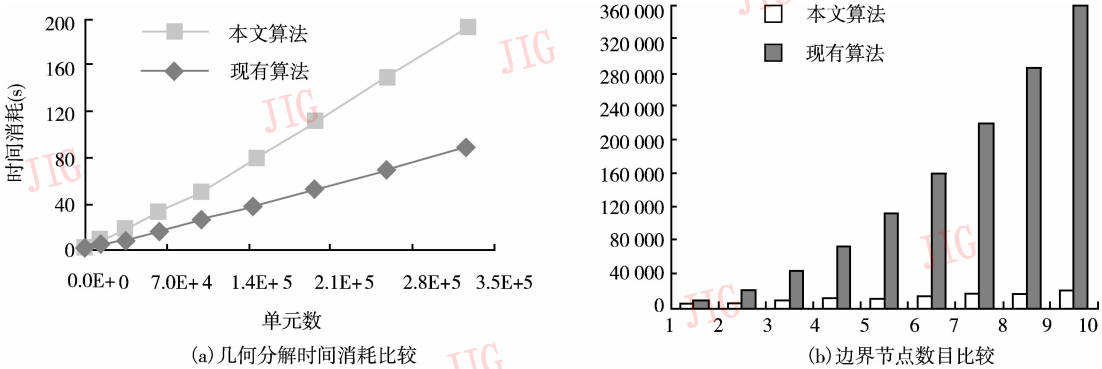
图 8 4 类拓扑优化操作

Fig. 8 Four categories of topological optimization operations

5　数值实验

前人算法[2,5]限定最终子域边界节点数目为 6 或者 8,本文则通过引入模板法来放松对最终子域的要求,降低几何分解过程的层次,且优化其时空效率。为量化这种改进,通过限定最终子域边界节点数目不超过 8 来模拟前人算法[2,5],并与本文算法作比较。9 个不同规模的算例是通过对同一个几何形体调整单元尺寸控制值大小而形成。

由于网格生成和优化过程消耗的时间和占用的内存主要和问题规模相关,对同一规模问题,两类算法的差别主要体现在几何分解过程上。图 9(a)给出了两类算法在几何分解过程的时间消耗曲线,可以看出两者性能有一倍左右的差距。几何分解过程的内存消耗主要和边界节点(包括初始边界节点和内部约束边界节点)的数目相关。边界节点定义包括基本定义和引用定义两部分,前者包含的信息有节点的坐标和单元尺寸值,采用双精度浮点数需消耗 24 个字节;后者主要指子域边界环对节点的引用,一次引用占用 4 个字节。图 9(b)比较了两类算法产生的边界节点数目,可以看出两者有数量级的差异。对于算例 9 而言,利用本文算法可少生成 342 244 个节点,减少节点引用 2 123 762 次,共计节省内存 15.9MB。若考虑定义在节点及其引用上的其他附加信息,实际节省的内存更为可观。



(a)几何分解时间消耗比较　　(b)边界节点数目比较

图 9　本文算法和前人算法的比较

Fig. 9　Comparison of the timing and storage performance between our algorithm and previous algorithms

图 10 给出了一个具有圆形外边界和莲花状孔洞的问题域及其网格,命名为 lotus。用户希望加密孔洞附近的网格,最终网格配置很好地满足了用户这一需求,且整体网格的疏密过渡控制得很合理。

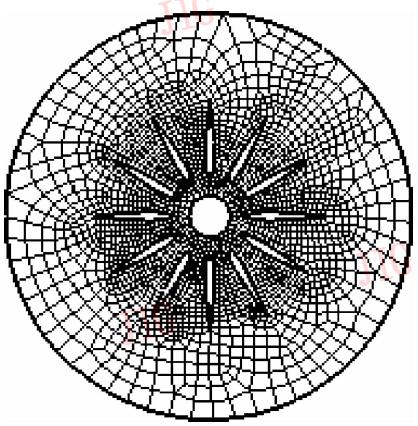


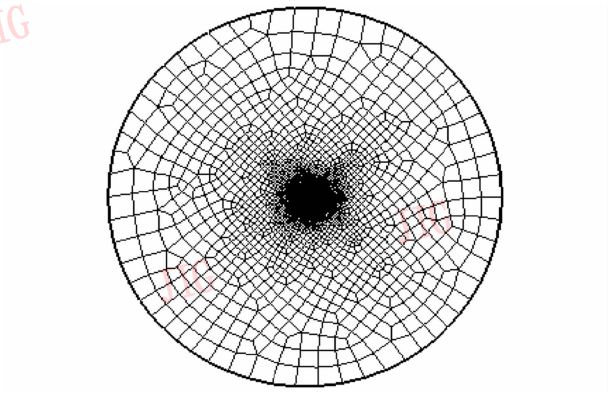
图 10　网格实例:lotus

Fig. 10　Mesh sample: lotus

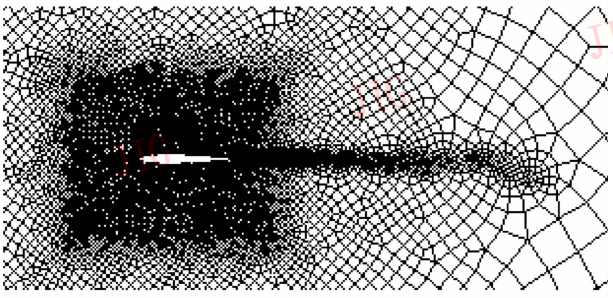
图 11 通过实例 naca 更好地证明了本文算法适应局部密度控制的能力。问题域具有圆形外边界,内部含一个翼型孔洞。紧挨孔洞定义了两个点源和两个线源,在更大的一个矩形区域中,背景网格的控制尺寸也非常小。孔洞附近和外边界附近的单元尺寸值相差约 200 倍。图 11 给出了全局网格视图以及翼型孔洞附近网格的放大图,可以看到,在网格源的影响下,孔洞附近的网格单元尺寸过渡很缓慢,且形成明显的尾迹区;其附近还有一个矩形加密区,这反映的是背景网格对单元尺寸的控制效果。

表 1 列出了 lotus 和 naca 的质量数据。为证明拓扑优化的重要性,表 1 还列出了没有经过拓扑优化处理时网格的质量数据。表中 N_{el} 表示单元总数,*MIA*、*MAA*、*MAL* 分别为所有单元最小角、最大角和最大长细比(长细比 = 最长边长/最短边长),*A*60、*A*120 分别为最小角小于 60°和最大角大于 120°的单元总数,*L*25 为长细比大于 2.5 的单元总数,括号

里的数据为相应值和 N_{el}的比值。从表 1 可以看出，拓扑优化不仅大幅度提升了网格质量，还能消除边界附近的无效单元。



(a)全局网格



(b)翼型孔洞附近的矩形加密区和尾迹区

图 11 网格实例：naca

Fig. 11 Mesh sample: naca

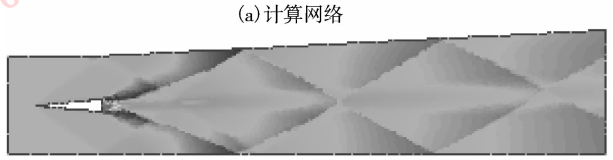
表 1 未经过及经过拓扑优化的网格对应的质量指标比较

Tab. 1 Comparison of quality merits with or without mesh topologies optimized

实例	优化	N_{el}	*MIA*	*MAA*	*MAL*	*A*60(%)	*A*120(%)	*L*25(%)
lotus	否	2 254	18.25	202.68	4.85	213 (9.4)	293 (13.0)	28 (1.2)
	是	2 195	35.42	148.47	3.41	58 (2.6)	69 (3.1)	9 (0.4)
naca	否	16 679	36.87	164.98	4.90	828 (5.0)	2 888 (17.3)	97 (0.6)
	是	16 439	53.13	146.86	2.79	78 (0.5)	657 (4.0)	4 (0.02)

利用本文算法生成的非结构网格系统，结合基于非结构网格系统的超声速燃烧软件包(PCP)，对超燃冲压发动机内的可压缩湍流流场进行了数值模拟，德国宇航中心（DLR）的化学推进实验室已对相同的发动机进行了实验研究。燃烧室的一侧壁面为倾斜的，最小截面尺寸为 45mm × 50mm，燃烧室后面为一马赫数为 2 的 Laval 喷管。计算网格以及计算得到的温度分布如图 12 所示。可以看出，当流场中激波处网格经过加密后，计算结果能够很准确地捕捉到楔形体前端的体前激波、楔形体尾部的膨胀波系以及流场其他位置的反射波。

(a)计算网络



(b)温度分布图

图 12 超燃冲压发动机内的可压缩湍流流场数值模拟

Fig. 12 Simulation of turbulent combustion in a Scramjet engine

6 结 论

本文改进了一类基于分而治之思想的四边形网格生成算法，主要贡献体现在：

(1)设计了一套健壮的网格模板，放松了对最终子域的形状要求；

(2)引入了背景网格和网格源，提高了算法对网格疏密的控制能力；

(3)引入了自动拓扑分解和几何分解过程，实现了算法的全自动性。

实验结果表明，该算法完全能够满足实际计算分析的需求。

参考文献(References)

1 Owen S J. A survey of unstructured mesh generation technology [A]. In: Proceedings of the 7th International Meshing Roundtable [C], Dearborn, MI, USA, 1998: 239 ~ 267.

2 Talbert J A, Parkinson A R. Development of an automatic two dimensional finite element mesh generator using quadrilateral elements and Bezier curve boundary definition [J]. International Journal for Numerical Methods in Engineering, 1990, **29**(7): 1551 ~ 1567.

3 Chae S W, Jeong J H. Unstructured surface meshing using operators [A]. In: Proceedings of the 6th International Meshing Roundtable [C], Park City, Utah, USA, 1997: 281 ~ 291.

4 Nowottny D. Quadrilateral mesh generation via geometrically optimized domain decomposition [A]. In: Proceedings of the 6th International Meshing Roundtable [C], Park City, Utah, USA, 1997: 309 ~ 320.

5 Sarrate J, Huerta A. Efficient unstructured quadrilateral mesh generation [J]. International Journal for Numerical Methods in

Engineering, 2000, **49**(10): 1327 ~ 1350.

6 Chen Jian-jun, Zheng Yao. A robust and quality guaranteed pattern module scheme for multi-subdomain methods in mesh generation[J]. Journal of Computer-aided Design and Computer Graphic, 2005, **17**(10): 2286 ~ 2292. [陈建军, 郑耀. 多子域网格生成方法中健壮保质的型模板[J]. 计算机辅助设计与图形学学报, 2005, **17**(10): 2286 ~ 2292.]

7 Li Hua, Cheng Geng-dong. New method for graded mesh generation of quadrilateral finite elements[J]. Computers and Structures, 1996, **59**(5): 823 ~ 829.

8 Li Hua, Cheng Geng-dong, Gu Yuan-xian. A new method for fasting mesh generation of quadrilateral finite elements: pattern module's method[J]. Journal of Computational Structural Mechanics and Applications, 1996, **13**(1): 25 ~ 39. [李华, 程耿东, 顾元宪. 一种新的全四边形网格快速生成方法——模板法[J]. 计算结构力学及其应用, 1996, **13**(1): 25 ~ 33.]

9 Chen Jian-jun. Unstructured mesh generation and its parallelization [D]. Hangzhou: Zhejiang University, 2006. [陈建军. 非结构化网格生成及其并行化的若干问题研究[D]. 杭州:浙江大学,2006.]

10 Zheng Y, Weatherill N P, Turner-Smith E T. An interactive geometry utility environment for multi-disciplinary computational engineering[J]. International Journal for Numerical Methods in Engineering, 2002, **53**(6): 1277 ~ 1299.

11 Said R, Weatherill N P, Morgan P, *et al*. Distributed parallel Delaunay mesh generation [J]. Computer Methods in Applied Mechanics and Engineering, 1999, **177**(1/2): 109 ~ 125.

12 Zheng Y, Lewis R W, Gethin D T. Three-dimensional unstructured mesh generation: Part 1. fundamental aspects of triangulation and point creation[J]. Computer Methods in Applied Mechanics and Engineering, 1996, **134**(3/4): 249 ~ 268.

13 Haber R, Shephard M S, Abel J F, *et al*. A general two-dimensional graphical finite element preprocessor utilizing discrete transfinite mappings [J]. International Journal for Numerical Methods in Engineering, 1981, **17**(7): 1015 ~ 1044.

14 Chen Li-gang, Zheng Yao, Chen Jian-jun. Topological improvement for quadrilateral finite element meshes[J]. Journal of Computer-aided Design and Computer Graphic, 2007, **19**(1): 78 ~ 83. [陈立岗, 郑耀, 陈建军. 全四边形有限元网格的拓扑优化策略[J]. 计算机辅助设计与图形学学报, 2007, **19**(1): 78 ~ 83.]