文章编号: 1001- 2486(2008) 02- 0128- 07

基于学科间差异信息的协同优化改进算法*

冯向军, 戴金海
(国防科技大学 航天与材料工程学院, 湖南 长沙 410073)

摘 要: 分析了MDO(Multidisciplinary Design Optimization) 中广泛应用的协同优化算法的特点和存在的问题, 提出了一种基于学科间差异信息的协同优化改进算法 COMI(Collaborative Optimization based on Multidisciplinary Inconsistency) 。利用学科间差异信息构造了系统级松弛约束和系统级罚函数, 利用遗传算法作为系统级优化算法, 并采用标准算例比较了标准 CO(Collaborative Optimization) 算法、松弛 CO 算法与 COMI 算法的性能。结果表明 COMI 算法在设计结果可行性和最优值上平衡性较好。

关键词: 协同优化; 学科间差异; 多学科设计优化
中图分类号:TP301. 6 文献标识码: A

An Improved Collaborative Optimization Based on Multidisciplinary Inconsistency

FENG Xiang-jun, DAI Jin-hai
(College of Aerospace and Material Engineering, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: Reasons that cause computational difficulties in collaborative optimization are analyzed and the COMI(Collaborative Optimization Based on Multidisciplinary Inconsistency) algorithm is presented. Relax constraints and system level penalty function are constructed based on inconsistency information between disciplinaries, and a GA optimizer is used to solve a typical MDO question. Simulation results show that the feasibility and the optimal value of COMI is more balanced than the standard CO algorithm and the relax CO algorithm.

Key words: collaborative optimization; multidisciplinary inconsistency; multidisciplinary design optimization

多学科设计优化 MDO(Multidisciplinary Design Optimization) 作为一种解决复杂系统优化设计的方法, 近 20 年在航空航天领域得到高度重视和广泛应用[1]。

协同优化 CO(Collaborative Optimization) 算法是由 Braun 提出的目前应用较为广泛的 MDO 算法。CO 的思想是采用两级优化结构描述 MDO 问题, 由系统级优化器负责优化系统级目标, 由学科级优化器负责满足学科级约束, 由系统级优化器和学科级优化器联合求解以满足学科间一致性约束。在迭代优化过程中, 先由系统级优化器提出系统级设计目标值, 然后由各学科级优化器在负责满足本学科设计约束的同时, 尽量缩小本学科设计值与系统级目标值的差异。各学科设计值之间的不一致由系统级优化器利用学科间一致性约束在优化过程中加以协调。系统级优化器在协调学科间一致性的同时进行系统级目标的优化。

1 协同优化算法的计算结构及其局限性

1.1 标准 CO 算法

以一个典型的 MDO 问题[2] 为例介绍 CO 算法的数学模型。该问题是一个非线性优化问题, 其数学模型描述如下:

* 收稿日期: 2007- 09- 19
作者简介: 冯向军(1971—) , 男, 副教授, 在职博士生。

$$\begin{cases} \min & f(x) = x_1^2 + x_2^2 \\ \text{s.t} & x_1 + \beta x_2 < 4 \\ & \beta x_1 + x_2 > 2 \end{cases} \tag{1}$$

该问题的标准 CO 算法的数学模型描述为:

(1) 系统级优化模型

$$\begin{cases} \min & f(z) = z_1^2 + z_2^2 \\ \text{s.t} & J_1^* = (z_1 - x_1^{(1)})^2 + (z_2 - x_2^{(1)})^2 = 0 \\ & J_2^* = (z_1 - x_1^{(2)})^2 + (z_2 - x_2^{(2)})^2 = 0 \end{cases} \tag{2}$$

其中 J_1^* 和 J_2^* 是系统一致性约束, z_1 和 z_2 是系统级设计变量, $x_i^{(k)}$ 是第 k 个学科优化计算出的第 i 个系统级变量的设计值。

(2) 学科 1 优化模型

$$\begin{cases} \min & f(x) = (z_1 - x_1^{(1)})^2 + (z_2 - x_2^{(1)})^2 \\ s.t & x_1^{(1)} + \beta x_2^{(1)} < 4 \end{cases} \tag{3}$$

(3) 学科 2 优化模型

$$\begin{cases} \min & f(x) = (z_1 - x_1^{(2)})^2 + (z_2 - x_2^{(2)})^2 \\ \text{s.t} & \beta x_1^{(2)} + x_2^{(2)} > 2 \end{cases} \tag{4}$$

虽然 CO 在工程领域已经得到广泛应用, 但是在工程实践中, 协同优化算法表现出了计算方面的困难性[3-5]。造成 CO 计算困难性的原因包括[6]:

㈠ 系统级的学科一致性约束常常导致系统级优化问题不满足 Kraush-Kuhn-Tucker 条件, 从而拉格朗日算子不存在, 使得系统级优化问题不能用常见的基于导数信息的优化算法求解。

㈡由于 CO 算法采用两级优化结构描述 MDO 问题, 常常使得原本是线性优化的 MDO 原问题, 转化为非线性的优化问题, 增加了 CO 计算的难度。

㈣系统级的学科一致性约束是等式约束, 且约束个数往往大于系统级优化模型的设计变量个数, 极大地限制了系统级优化的自由度, 导致传统的优化算法对 CO 问题无解。

1.2 松弛 CO 算法

为了降低 CO 计算的难度, 可以通过将 CO 的学科一致性等式约束转化为不等式约束, 即将式(2)中的 $J_i^* = 0$ 变换为 $J_i^* < \varepsilon$, 当 ε 足够小时, 不等式约束趋近于等式约束。Alexzandrov 等分析了松弛因子 ε 对 CO 算法的影响, 指出松弛因子 ε 的取值对系统级松弛约束的性能有重要影响[6], 针对不同的 MDO 问题没有普遍适用的 ε。李响等提出了一种基于学科间不一致信息的动态松弛算法[7], 但是只利用了各学科设计点间的差异性。

2 COMII 算法

2.1 学科间不一致信息

从 CO 算法的流程以及图 1 的 CO 算法几何解释可知, 在迭代过程中, 针对同一的系统级设计期望点 P, 各学科给出的学科设计点 X_i 是不同的。所谓学科间不一致信息包含两个方面的内容:

(1) 各学科设计点 X_i 之间的差异性

假设 CO 算法在第 k 次迭代后, 各学科给出了各自的学科设计点 X_i, 定义范数:

$$D_{ij}^{DD(k)} = \| X_i - X_j \| \tag{5}$$

则 $D_{ij}^{DD(k)}$ 描述了第 k 次迭代后各学科设计点之间的差异性。上标 DD 表示学科到学科(Discipline to Discipline)。如果一个 MDO 问题涉及 n 个学科, 则 $D_{ij}^{DD(k)}$ 范数一共有 $n(n-1)/2$ 个。

(2) 学科设计点 X_i 与系统级设计期望点 P 之间的差异性

假设CO算法在第 k 次迭代后,针对系统级设计期望点 P,各学科给出了各自的学科设计点 X_i,定义范数:

$$D_i^{DS(k)} = \| X_i - P \| \tag{6}$$

则 $D_i^{DS(k)}$ 描述了各学科设计点与系统级设计期望点之间的差异性。上标DS表示学科级到系统级(Discipline to System)。如果一个MDO问题涉及 n 个学科,则 $D_i^{DS(k)}$ 范数一共有 n 个。

2.2 基于学科间不一致信息的松弛约束

从几何意义上来说,一组松弛约束实际上描述了设计空间中一组超球的交集。如图1的三学科CO所示,当 $J_i^* = 0$ 时,CO系统级优化寻找的是设计空间中与 x_1、x_2、x_3 三个学科设计点距离和最小的系统级设计点。而 $J_i^* < \varepsilon$ 时,系统级优化寻找的是以 x_1、x_2、x_3 为圆心,ε 为半径的三个圆的交集中目标函数值最小的点。显然,满足松弛约束的设计点比满足等式约束的设计点更容易优化。在图1中,设计空间是二维平面,所以松弛约束描述的是圆的交集。当设计空间是高维时,松弛约束描述的就是超球的交集[8]。

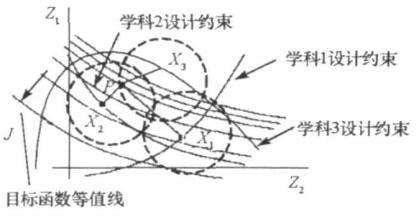

图1 松弛约束的几何解释
Fig. 1 The Geometrical Description of Relax Constrains

松弛约束 ε 如何构造是十分重要的问题,ε 太小则各超球没有交集,意味着系统级优化没有可行域,ε 太大又失去了系统级学科一致性约束的意义。而且在系统级优化迭代过程中,固定的 ε 不一定适合每一步迭代。针对以上分析,本文提出以下松弛约束算法:

在一个涉及 n 个学科的CO问题中,定义:

$$D_{\max}^{DD(k)} = \max_{i,j=1,\cdots,n \atop i \neq j} \left\{ D_{ij}^{DD(k)} \right\} \tag{7}$$

其中 $D_{ij}^{DD(k)}$ 范数的定义见2.1节。$D_{\max}^{DD(k)}$ 的几何含义是各学科给出的设计点之间的最大差异。第 k 次迭代后系统级一致性约束中各学科的松弛约束 $\varepsilon_i^{(k)}$ 定义为

$$\varepsilon_i^{(k)} = \lambda D_{\max}^{DD(k)}, \quad 0.5 < \lambda < 1 \tag{8}$$

根据式(8)可知

$$\varepsilon_i^{(k)} + \varepsilon_j^{(k)} = 2\lambda D_{\max}^{DD(k)} > D_{\max}^{DD(k)} \tag{9}$$

由式(9)可知,式(8)确定的 ε 可以确保各学科设计点所确定的超球必有交集。随着CO迭代的运行,各学科的设计点逐步趋向同一,则 ε 所确定的松弛约束也逐步逼近标准算法中等式形式的系统级学科一致性约束。

2.3 基于学科间不一致信息的系统级目标函数

在2.2节对松弛约束的分析中,需要注意的是,只有松弛约束并不能保证 $D_{\max}^{DD(k)}$ 随着CO迭代运行逐步减小。为了促进 $D_{\max}^{DD(k)}$ 的减小,还要利用 $D_i^{DS(k)}$ 范数对系统级目标函数进行修正。假设在CO迭代进行到第 k 步时,定义:

$$D_{sum}^{DS(k)} = \sum_{i=1}^{n} D_i^{DS(k)} \tag{10}$$

其中,$D_i^{DS(k)}$ 表示在第 k 步时,学科 i 的设计点与系统级设计期望点的差异。$D_{sum}^{DS(k)}$ 表示在第 k 步时所有学科设计点与系统级期望点的差异之和。在CO标准算法的目标函数中增加以下罚函数:

$$\Phi_k(x) = \begin{cases} 0, & D_{sum}^{DS(k)} < D_{sum}^{DS(k-1)} \\ r^{(k)} D_{sum}^{DS(k)}, & D_{sum}^{DS(k)} \geqslant D_{sum}^{DS(k-1)} \end{cases} \tag{11}$$

其中,$\Phi_k(x)$ 表示迭代在第 k 步时的动态罚函数,$r^{(k)}$ 为惩罚因子,$r^{(k)}$ 的取值使得 $\Phi_k(x) \gg f(x)$。则原问题的目标函数 $f(x)$ 变为 $f(x) + \Phi_k(x)$。增加罚函数的目的是限制CO在每次迭代过程中优先考虑减少各学科设计点与系统级设计期望点的差异。由图1,根据三角形的两边之和大于第三边可知,

$$D_i^{DS} + D_j^{DS} > D_{ij}^{DD} \tag{12}$$

则有

$$2D_{sum}^{DS(k)} = 2\sum_{i=1}^{n} D_i^{DS(k)} > \sum_{\substack{i,j=1 \\ i\neq 1}}^{n} D_{ij}^{DD} \tag{13}$$

随着 CO 迭代的进行, 由于罚函数的存在, $D_{sum}^{DS(k)}$ 会逐步减小, $D_{\max}^{DD(k)}$ 随之减小。注意, 虽然每一步迭代中 $D_{sum}^{DS(k)}$ 都会减小, 但 $D_{\max}^{DD(k)}$ 可能是振荡减小。在罚函数中采用 $D_{sum}^{DS(k)}$ 而不采用 $D_{\max}^{DD(k)}$ 是因为最终的设计目标是使系统级设计期望点与各学科设计点一致, 而不仅仅是各学科设计点间相互一致。

2.4 COMI 算法流程

该算法的基本过程是:

(1) 系统级优化器给出设计期望点 P;

(2) 学科级优化器根据学科级目标函数计算各学科设计点, 并根据优化结果按式(6) 计算 $D_i^{DS(k)}$ 范数;

(3) 系统级优化器根据各学科设计点的结果按式(5) 计算 $D_{ij}^{DD(k)}$ 范数, 并按式(8) 构造系统级学科一致性松弛约束 $\varepsilon_i^{(k)}$, 按式(11) 构造系统级罚函数 $\phi_k(x)$;

(4) 系统级优化器根据系统级学科一致性松弛约束 $\varepsilon_i^{(k)}$ 和系统级罚函数 $\phi_k(x)$ 优化求解新的设计期望点 P, 并返回步骤(1), 直到满足收敛条件。

3 仿真算例

为了比较 COMI 算法与其他 CO 算法的性能, 本文采用了 COMI 算法、标准 CO 算法以及文献[7] 提出的动态松弛算法三种方法求解一个减速器设计问题。该优化问题是 NASA 评估多学科设计优化方法性能的十个标准算例[11] 之一。

减速器优化的数学模型为:

$$\begin{aligned}
&\min\quad f(x) = C_{f1}x_1x_2^2(C_{f2}x_3^2 + C_{f3}x_3 - C_{f4}) - C_{f5}x_1(x_6^2 + x_7^2) + C_{f6}(x_6^3 + x_7^3) + C_{f7}(x_4x_6^2 + x_5x_7^2)\\
&s.t\\
&\qquad g_1 = C_{g1}/(x_1x_2^2x_3) \leqslant 1.0\\
&\qquad g_2 = C_{g2}/(x_1x_2^2x_3^2) \leqslant 1.0\\
&\qquad g_3 = C_{g3}x_4^3/(x_2x_3x_6^4) \leqslant 1.0\\
&\qquad g_4 = C_{g4}x_5^3/(x_2x_3x_7^4) \leqslant 1.0\\
&\qquad g_5 = \frac{\sqrt{\left(\frac{C_{A12}x_4}{x_2x_3}\right)^2 + C_{A1}}}{C_{g5}C_Bx_6^3} \leqslant 1.0\\
&\qquad g_6 = \frac{\sqrt{\left(\frac{C_{A12}x_5}{x_2x_3}\right)^2 + C_{A2}}}{C_{g6}C_Bx_7^3} \leqslant 1.0\\
&\qquad g_7 = x_2x_3/C_{g7} \leqslant 1.0\\
&\qquad g_8 = C_{g8}x_2/x_1 \leqslant 1.0\\
&\qquad g_9 = x_1/(C_{g9}x_2) \leqslant 1.0\\
&\qquad g_{10} = (C_{g10}x_6 + C_{g10(5)})/x_4 \leqslant 1.0\\
&\qquad g_{11} = (C_{g11}x_7 + C_{g10(5)})/x_5 \leqslant 1.0\\
&\qquad 2.6 \leqslant x_1 \leqslant 3.6\\
&\qquad 0.7 \leqslant x_2 \leqslant 0.8\\
&\qquad 17 \leqslant x_3 \leqslant 28\\
&\qquad 7.3 \leqslant x_4 \leqslant 8.3\\
&\qquad 7.3 \leqslant x_5 \leqslant 8.3\\
&\qquad 2.9 \leqslant x_6 \leqslant 3.9\\
&\qquad 5.0 \leqslant x_7 \leqslant 5.5
\end{aligned} \tag{14}$$

减速器多学科设计优化的目标是在满足减速器中转轴和齿轮约束的同时,使得减速器体积最小(质量最轻)。该优化问题有七个设计变量,x_1 为齿面宽度,x_2 为齿轮模数,x_3 为小齿轮齿数,x_4 和 x_5 为轴承间距,x_6 和 x_7 为大小齿轮轴的直径。g_1 为轮齿的最大弯曲应力,g_2 为轮齿最大接触应力,g_3 和 g_4 为轴的横向最大挠度,g_5 和 g_6 为轴内最大应力,g_7、g_8 和 g_9 为尺寸和空间限制,g_{10} 和 g_{11} 为轴尺寸计算的经验公式。C_{f1} 等参数的取值参见文献[8]。

采用 CO 算法求解时,将该优化问题分解为三个学科级优化和一个系统级优化,系统级设计变量分别为 z_1、z_2 和 z_3,学科 1 的设计变量分别为 x_1、x_2 和 x_3,学科 2 的设计变量分别为 x_1、x_2、x_3、x_4 和 x_6,学科 3 的设计变量分别为 x_1、x_2、x_3、x_5 和 x_7。

标准 CO 算法的系统级优化模型为:

$$\begin{cases} \min & f_1 + f_2 + f_3 \\ s.t & \left(\sum_{k=1}^{3}\sum_{i=1}^{3}\left(x_i^{(k)} - z_i\right)^2\right) = 0 \end{cases} \tag{15}$$

松弛 CO 算法的系统级优化模型为:

$$\begin{cases} \min & f_1 + f_2 + f_3 \\ s.t & \left(\sum_{k=1}^{3}\sum_{i=1}^{3}\left(x_i^{(k)} - z_i\right)^2\right) \leqslant \varepsilon \end{cases} \tag{16}$$

COMI 算法的系统级优化模型为:

$$\begin{aligned} \min \quad & f_1 + f_2 + f_3 + \Phi(x) \\ s.t \quad & \left(\sum_{k=1}^{3}\sum_{i=1}^{3}\left(x_i^{(k)} - z_i\right)^2\right) \leqslant \varepsilon \end{aligned} \tag{17}$$

式(16)和(17)中的 ε 和 Φ 分别按照式(8)和式(11)确定。

学科 1 的数学模型为:

$$\begin{cases} \min & J = \sum_{i=1}^{3}\left(x_i - z_i\right)^2 \\ s.t & f_1 = C_{f1}x_1x_2^2\left(C_{f2}x_3^2 + C_{f3}x_3 - C_{f4}\right) \\ & g_j \leqslant 1.0, j = 1, 2, 7, 8, 9 \\ & LB \leqslant x_k \leqslant UB, k = 1 \cdots 7 \end{cases} \tag{18}$$

学科 2 的数学模型为:

$$\begin{cases} \min & J = \sum_{i=1}^{3}\left(x_i - z_i\right)^2 \\ s.t & f_1 = -C_{f5}x_1x_7^2 + C_{f6}x_7^3 + C_{f1}x_5x_7^2 \\ & g_j \leqslant 1.0, j = 1, 2, 4, 6, 7, 8, 9, 11 \\ & LB \leqslant x_k \leqslant UB, k = 1 \cdots 7 \end{cases} \tag{19}$$

学科 3 的数学模型为:

$$\begin{cases} \min & J = \sum_{i=1}^{3}\left(x_i - z_i\right)^2 \\ s.t & f_1 = -C_{f5}x_1x_6^2 + C_{f6}x_6^3 + C_{f1}x_4x_6^2 \\ & g_j \leqslant 1.0, j = 1, 2, 3, 5, 7, 8, 9, 10 \\ & LB \leqslant x_k \leqslant UB, k = 1 \cdots 7 \end{cases} \tag{20}$$

式(18)~(20)中的 LB 和 UB 是式(14)中规定的设计变量 $x_k(k = 1 \cdots 7)$ 取值的上下限。考虑到前述 CO 算法固有的非线性和设计空间可能不连续等缺陷,优化求解器采用了遗传算法优化器。为了有统一

的标准对比各种 CO 算法性能, 各种 CO 算法中的优化求解器统一采用了 Matlab 平台的 GA Toolbox[9]。采用遗传算法作为优化求解器的另外一个优点是可以避免优化起始点带来的影响[10]。仿真结果如表 1 所示。

表 1 三种 CO 算法的结果对比表

Tab. 1 The result of the CO, the relax CO and the COMI

指标 \ 算法	标准 CO	松弛 CO	COMI
最优值	3100.7039	2997.6358	2978.8421
最优点	(3.5999, 0.7127, 17.0048, 7.7342, 8.2916, 3.8882, 5.0045)	(3.5146, 0.6959, 16.9978, 7.75554, 8.2908, 3.8999, 5.0148)	(3.4829, 0.6945, 16.9936, 7.7496, 8.2994, 3.8964, 5.0168)
学科 1 与系统期望值差	0.0001	0.0012	0.0003
学科 2 与系统期望值差	0.0001	0.0034	0.0002
学科 3 与系统期望值差	0.0003	0.0012	0.0002

根据文献[2] 可知, 减速器优化问题的解是(3.501, 0.7, 17.02, 7.3, 7.714, 3.346, 5.286), 最优值是 2986.921, 从表 1 的结果可以看出, 三种 CO 算法的计算结果与文献[2] 的结果十分相近, 说明三种方法都是可行的。从图 2 至图 4 各算法的收敛过程可知, 标准 CO 算法能够较早收敛到最优值附近, 而松弛 CO 和 COMI 的收敛过程较慢。但是从表 1 可知, 标准 CO 的最优值明显低于松弛 CO 算法和 COMI 算法。松弛 CO 算法和 COMI 算法能找到较好的最优值是因为它们都在学科一致性约束中采用了松弛条件, 而标准 CO 算法采用了严格的等式条件, 因此在优化过程中限制了寻优的搜索范围。严格的约束在获得较快收敛速度的同时, 牺牲了最优目标函数。

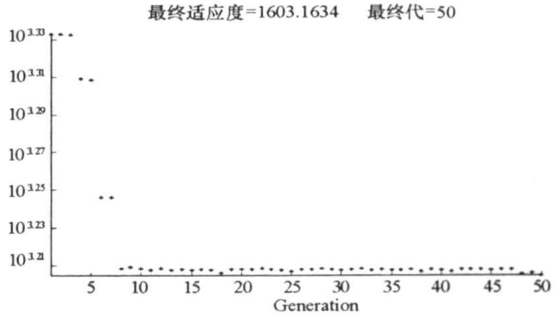

图 2 标准 CO 算法的迭代收敛过程

Fig. 2 The convergence process of the CO

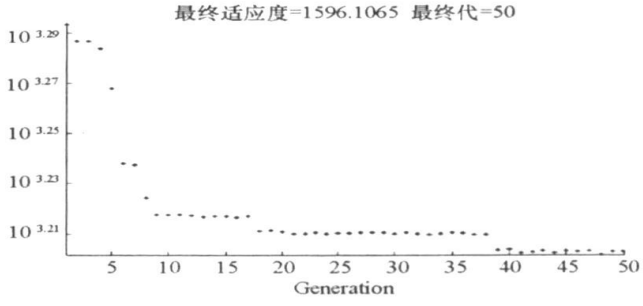

图 3 松弛 CO 算法的迭代收敛过程

Fig. 3 The convergence process of the relax CO

学科设计点和系统级期望点的近似程度越低, 表明设计结果的可行性越好。从表 1 计算结果可以看出, 松弛 CO 算法和 COMI 算法的计算效率相近, 但是从结果的质量上可以看出, COMI 算法明显强于

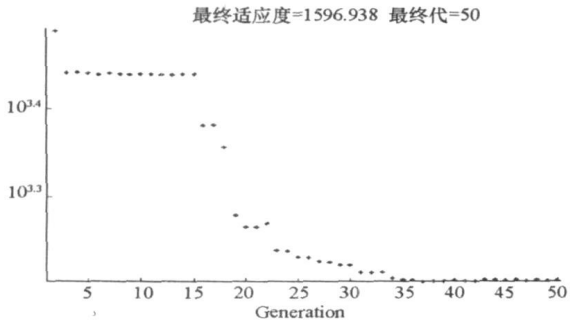

图 4 COMI 算法的迭代收敛过程

Fig. 4 The convergence process of the COMI

松弛 CO 算法。标准 CO 算法的设计可行性最好, 是因为标准 CO 算法采用了等式约束。松弛 CO 算法采用的松弛约束只考虑了学科间的不一致信息, COMI 算法采用的松弛约束和系统级目标函数综合考虑了学科间不一致信息以及学科级和系统级的不一致信息, 相对于松弛 CO 算法更接近于标准 CO 算法的等式约束, 因此具有更好的设计可行性。

4 结 论

本文分析了 CO 算法计算困难的原因, 并利用学科间不一致信息提出了 COMI 算法。COMI 算法利用学科间不一致信息构造了 $D_{\tilde{i}}^{DD(k)}$ 范数和 $D_i^{DS(k)}$ 范数, 并利用这两种范数改进了标准 CO 的学科一致性约束条件和系统级目标函数。仿真算例结果表明, COMI 算法相比标准 CO 算法和松弛 CO 算法, 在最优目标函数值和设计可行性之间取得了较好的平衡。

标准 CO 算法、松弛 CO 算法和 COMI 算法的本质区别在设计思想上。标准 CO 算法的设计思想要求最为苛刻, 设计结果的可行性也最好。但苛刻的代价是牺牲了最优目标函数。松弛 CO 算法的设计思想较适合工程要求, 因为它将标准 CO 算法的学科一致性等式约束放宽为不等式形式的松弛约束, 可以搜索到更优的设计值, 但是以降低设计可行性为代价。而 COMI 算法与标准 CO 算法相比, 由于采用了更加符合工程实际的松弛条件, 可以搜索到更优的设计值, 与松弛 CO 算法相比, 更多地利用了学科间不一致信息, 保证了设计可行性。

参 考 文 献:

[1] Perez R E, Liu H, Behdinan K. Evaluation of Multidisciplinary Optimization Approaches for Aircraft Conceptual Design[J]. AIAA 2004- 4537, 2004.

[2] 韩明红, 邓家 . 协同优化算法的改进[J]. 机械工程学报, 2006, 42(11).

[3] Kroo I, Manning V. Collaborative Optimization: Status and Directions[J]. AIAA- 2000- 4721, 2000.

[4] Alexandrov N M, Lewisy R M. Comparative Properties of Collaborative Optimization and other Approaches to MDO[R]. NASA/CR- 1999- 209354, 1999.

[5] Alexzandrov N, Kodiyanam S. Initial Result of an MDO Method Evaluation Study[J]. AIAA 98- 4884, 1998.

[6] Alexzandrov N M, Lewis R M. Analytical and Computational Aspects of Collaborative Optimization[R]. NASA/TM 2000 210104, 2000.

[7] 李响, 李为吉. 利用协同优化方法实现复杂系统分解并行设计优化[J]. 宇航学报, 2004, 25(3).

[8] 李响. 多学科设计优化方法及其在飞行器设计中的应用[D]. 西安: 西北工业大学, 2003.

[9] Genetic Algorithm and Direct Search Toolbox User's Guide[Z]. The MathWorks, Inc., 2006.

[10] 周明, 孙树栋. 遗传算法原理及应用[M]. 北京: 国防工业出版社, 1999.

[11] Srinivas K. Evaluation of Methods for Multidisciplinary Design Optimization (MDO), Phase II[R]. NASA/CR- 1998- 208716, 1998.