协同优化算法的改进*

韩明红　邓家褆
(北京航空航天大学机械工程及自动化学院　北京　100083)

摘要：针对协同优化算法是多学科设计优化方法中应用最广、效果最好的 MDO 算法，但是在应用中存在计算困难的现状，分析了引起协同优化算法计算困难的原因，提出一种改良的协同优化算法——ICO 多学科设计优化方法。改进学科一致性约束，增加系统级罚函数，采用快速启动方法，较好地克服了协同优化算法存在计算困难的缺点。标准算例试验结果表明，ICO 多学科设计方法能够有效提高算法的稳定性、可靠性和计算效率。

关键词：优化　计算机辅助设计　改进　协同优化算法　多学科设计优化
中图分类号：TP301.6

0　前言

多学科设计优化(Multidisciplinary design optimization, MDO)能够有效解决大规模复杂工程系统的设计问题，受到了世界各国的高度重视，但是作为一门只有 20 多年发展历程的新兴学科，在很多方面还不够完善。

协同优化算法(Collaborative optimization, CO)是 BRAUN 提出的一种按学科对 MDO 问题进行分解、协调、综合优化的方法，其每个学科的计算都有很好的自治性，不必考虑其他学科的影响[1]。CO 算法具有典型的两级优化结构，不同学科之间的一致性由附加到系统级优化中的等式约束来保证。一致性等式约束的值通过学科级优化得到，学科级优化的目标是在满足学科设计约束条件的同时，使得学科间的不一致性最小。但是在实际应用中，协同优化算法存在计算方面的困难[2-3]，对于很多优化问题无法得到收敛解或只能得到局部最优解。需要在协同优化算法计算困难的原因分析基础上，对协同优化算法进行改进研究。

1　协同优化算法计算困难的原因

引起协同优化算法计算困难，造成 BALLING[2] 和 ALEXANDROV 等[3]试验问题优化结果不好的主要原因是协同优化算法的数学表达造成的。CO 算法在系统级优化中加入了二次学科一致性等式约束，这就使得 CO 算法的系统级优化具有比原问题严重得多的非线性特性。另外，由于 Kuhn-Tucker 条件中的拉格朗日乘子不存在，或在最优解处学科一致性约束的 Jacobian 矩阵不连续，CO 算法的系统级优化问题无法满足 Kuhn-Tucker 条件。除此之外，还有可能出现不平滑的设计空间，系统级约束过多，学科级求解的不可行性等问题[4]。协同优化算法的上述缺点导致其使用传统的非线性优化算法求解时，必定面临严重的计算困难。

2　协同优化算法的改进

在已知的基于分解的求解非凸 MDO 问题的方法中，CO 算法最大限度地保持了学科的自治性和模块化，并且成功地解决了很多实际问题[5]，因此很有必要对 CO 算法进行深入的研究。

目前，已有学者在协同优化算法的改进方面进行了研究，提出了一些改进措施。BRAUN 等[6]提出了子系统罚函数法和松弛因子法的改进措施。ALEXANDROV 等[7]提出了另一种松弛因子法和分布式分析优化法(DAO)的改进措施。SOBIESKI 等[8]提出了使用响应面技术改进协同优化算法的措施。虽然这些措施基本上都是从 CO 算法的表达方式上进行改进的，但是并没有完全消除 CO 算法的计算困难，有些措施还带来了新的不稳定因素。基于上述分析并结合已有的改进方案，给出了一种改良的协同优化算法—ICO(Improved collaborative optimization)多学科设计优化方法。

ICO 多学科设计优化方法保持了 CO 算法模块化和学科自治性的优点，以新的表达方式来克服 CO 算法的计算困难。首先，使用 L_1 范数改进学科一致性约束。其次，增加使用 L_1 范数的系统级罚函数。最后，为了提高 ICO 算法的计算速度，给出了

* 国家自然科学基金(50275005)和国防基础科研(K1203010804)资助项目。20060119 收到初稿，20060519 收到修改稿

快速启动方法。

(1) 改进学科一致性约束。L_1 范数学科一致性约束定义为矢量 $(x_i - z)$ 的 1-范数，即

$$J_i(x) = \|x_i - z\|_1 = \sum_{j=1}^{n} |x_{ij} - z_j| \tag{1}$$

除此之外，为了确保优化子系统解空间的连续性，采用 ALEXANDROV 和 LEWIS 的松弛因子法对子系统约束进行改进，通过引入松弛变量，使用不等式约束取代子系统的等式约束。

改进后的子系统级优化模型为

$$\begin{aligned} \min \quad & J_i(x) = \sum_{j=1}^{n} |x_{ij} - z_j| \\ \text{s.t.} \quad & g_u(x) \le 0 \quad u = 1,2,\cdots,p \\ & h_v(x) \le \varepsilon_v \quad v = 1,2,\cdots,l \\ & \varepsilon_v \le 0 \quad j = 1,2,\cdots,l \end{aligned} \tag{2}$$

已经从数学理论上证明改进后的子系统级目标函数和标准 CO 算法中子系统级目标函数具有同样的收敛特性，且极限相同[4]。

(2) 增加系统级罚函数。增加系统级罚函数后的系统级优化模型为

$$\begin{aligned} \min \quad & F(z) + \gamma^{(k)} \sum_{i=1}^{N} (J_i^*(z))^2 \\ \text{s.t.} \quad & z_L \le z \le z_u \end{aligned} \tag{3}$$

式中 $\gamma^{(k)}$ 为惩罚因子，在优化过程中可随迭代次数 k 的增大不断进行调整；N 为学科数目；$J_i^*(z)$ 为第 i 个子系统最优解的目标函数值。

(3) 快速启动方法。通过对目前已有的几种 CO 算法改进方案进行分析，可以发现它们在子系统级优化过程中，并没有随系统级更新子系统级的初始值，而是取程序最初给定的初值。系统级优化器每次求得的最优解并没有在子系统级得到充分合理地利用。因此给出了快速启动方法，即在系统级优化迭代过程中，子系统级优化取前次系统级得到的最优解作为初值。采用快速启动方法后，程序的收敛速度加快，所需迭代次数也将明显减少。

3　算例试验

选取一个典型的函数优化问题和一个NASA的MDO 算法 Benchmark 测试问题来验证 ICO 算法的性能。CO 算法和 ICO 算法都在系统设计优化平台 SDOF 中实施[9]。

3.1　典型函数优化问题

BRAUN 等[6]使用该优化问题对标准 CO 算法进行了试验研究。它是一个约束非线性优化问题，其数学模型为

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

该优化问题标准 CO 算法的优化模型如下。

(1) 系统级优化模型

$$\begin{aligned} \min \quad & f(\boldsymbol{z}) = z_1^2 + z_2^2 \\ \text{s.t.} \quad & J_1^* = (z_1 - x_1)^2 + (z_2 - x_2)^2 = 0 \\ & J_2^* = (z_1 - x_1)^2 + (z_2 - x_2)^2 = 0 \end{aligned} \tag{5}$$

式中，z_1 和 z_2 为系统级设计变量，J_1^* 和 J_2^* 为系统级一致性等式约束。

(2) 第一个学科级优化模型

$$\begin{aligned} \min \quad & J_1 = (z_1 - x_1)^2 + (z_2 - x_2)^2 \\ \text{s.t.} \quad & x_1 + 0.1x_2 < 4 \end{aligned} \tag{6}$$

(3) 第二个学科级优化模型

$$\begin{aligned} \min \quad & J_2 = (z_1 - x_1)^2 + (z_2 - x_2)^2 \\ \text{s.t.} \quad & 0.1x_1 + x_2 > 2 \end{aligned} \tag{7}$$

该优化问题 ICO 算法的优化模型如下。

(1) 系统级优化模型

$$\min \quad f(\boldsymbol{z}) = z_1^2 + z_2^2 + J_1^{*2} + J_2^{*2} \tag{8}$$

式中，取惩罚因子 $\gamma = 1$

(2) 第一个学科级优化模型

$$\begin{aligned} \min \quad & J_1 = |z_1 - x_1| + |z_2 - x_2| \\ \text{s.t.} \quad & x_1 + 0.1x_2 < 4 \end{aligned} \tag{9}$$

(3) 第二个学科级优化模型

$$\begin{aligned} \min \quad & J_2 = |z_1 - x_1| + |z_2 - x_2| \\ \text{s.t.} \quad & 0.1x_1 + x_2 > 2 \end{aligned} \tag{10}$$

由文献[6]可知，当 $\beta = 0.1$ 时，最优解 x^* 为 (0.198, 1.980)，目标函数 f^* 为 3.998。两种方法系统级和子系统级都采用序列二次规划法(NLPQL)。选取 4 个不同的初始点进行了优化计算，优化结果如表 1 所示。

表 1　β=0.1 时的优化结果

起始点	CO 算法			ICO 算法		
	最优解 x^*	目标函数 f^*	迭代次数 n	最优解 x^*	目标函数 f^*	迭代次数 n
(1,1)	(0.162, 1.976)	3.930	52	(0.160, 1.983)	3.958	43
(4,–1)	(0.214, 1.972)	3.933	38	(0.192, 1.985)	3.978	22
(1,–1)	(0.179, 1.975)	3.932	48	(0.180, 1.991)	3.997	27
(10,3)	(0.638, 1.935)	4.151	86	(0.199, 1.983)	3.973	21

从表 1 的优化结果可以看到，对于不同的初始

点，标准 CO 算法和 ICO 算法基本上都收敛到了最优解附近，而且与最优目标函数值基本一致。但是对于所有起始点，ICO 算法迭代次数都大幅度减少，运行效率大大提高。同样本文在取 $\beta=0$，$\beta=0.3$，$\beta=0.5$，$\beta=1$时也得到了同样结论。

3.2 减速器设计优化算例

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之一[10]。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大量约束的同时，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，如图 1 所示。x_1为齿面宽度，x_2为齿轮模数，x_3为小齿轮齿数，x_4、x_5为轴承间距，x_6、x_7为大小齿轮轴的直径。

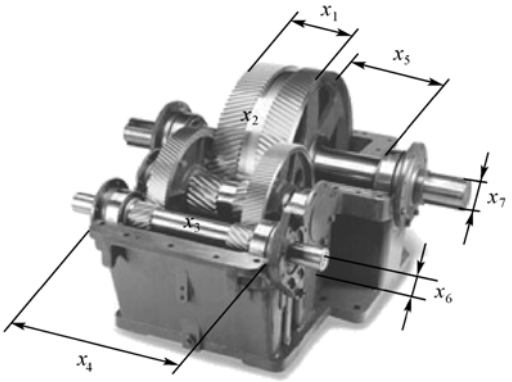

图 1 减速器结构图

减速器优化问题的数学模型为

$$\begin{aligned}\min f(\boldsymbol{x}) &= C_{f1}x_1x_2^2(C_{f2}x_3^2+C_{f3}x_3-C_{f4})-\\ &\quad C_{f5}x_1(x_6^2+x_7^2)+C_{f6}(x_6^3+x_7^3)+\\ &\quad C_{f1}(x_4x_6^2+x_5x_7^2) \qquad (11)\end{aligned}$$

$$\text{s.t.}\quad g_1=C_{g1}/(x_1x_2^2x_3)\le 1.0$$

$$g_2=C_{g2}/(x_1x_2^2x_3^2)\le 1.0$$

$$g_3=C_{g3}x_4^3/(x_2x_3x_6^4)\le 1.0$$

$$g_4=C_{g4}x_5^3/(x_2x_3x_7^4)\le 1.0$$

$$g_5=\frac{\sqrt{\left(\frac{C_{A12}x_4}{x_2x_3}\right)^2+C_{A1}}}{C_{g5}C_Bx_6^3}\le 1.0$$

$$g_6=\frac{\sqrt{\left(\frac{C_{A12}x_5}{x_2x_3}\right)^2+C_{A2}}}{C_{g6}C_Bx_7^3}\le 1.0$$

$$g_7=x_2x_3/C_{g7}\le 1.0$$

$$g_8=C_{g8}x_2/x_1\le 1.0$$

$$g_9=x_1/(C_{g9}x_2)\le 1.0$$

$$g_{10}=(C_{g10}x_6+C_{g105})/x_4\le 1.0$$

$$g_{11}=(C_{g11}x_7+C_{g105})/x_5\le 1.0$$

$$2.6\le x_1\le 3.6\quad 0.7\le x_2\le 0.8\quad 17\le x_3\le 28$$

$$7.3\le x_4\le 8.3\quad 7.3\le x_5\le 8.3$$

$$2.9\le x_6\le 3.9\quad 5.0\le x_7\le 5.5$$

式中 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。

采用标准 CO 算法和 ICO 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 算法的优化模型如下。

(1) 系统级优化模型

$$\begin{aligned}&\min \quad f=f_1+f_2+f_3 \qquad (12)\\ &\text{s.t.}\quad J_1^*=(x_1-z_1)^2+(x_2-z_2)^2+(x_3-z_3)^2=0\\ &\qquad J_2^*=(x_1-z_1)^2+(x_2-z_2)^2+(x_3-z_3)^2=0\\ &\qquad J_3^*=(x_1-z_1)^2+(x_2-z_2)^2+(x_3-z_3)^2=0\end{aligned}$$

(2) 学科 1 的优化模型

$$\begin{aligned}&\min \quad J_1=(x_1-z_1)^2+(x_2-z_2)^2+(x_3-z_3)^2 \qquad (13)\\ &\text{s.t.}\quad f_1=C_{f1}x_1x_2^2(C_{f2}x_3^2+C_{f3}x_3-C_{f4})\\ &\qquad g_j-1.0\le 0\quad j=1,2,7,8,9\end{aligned}$$

(3) 学科 2 的优化模型

$$\begin{aligned}&\min \quad J_2=(x_1-z_1)^2+(x_2-z_2)^2+(x_3-z_3)^2 \qquad (14)\\ &\text{s.t.}\quad f_2=C_{f6}x_7^3+C_{f1}x_5x_7^2-C_{f5}x_7^2x_1\\ &\qquad g_j-1.0\le 0\quad j=1,2,4,6,7,8,9,11\end{aligned}$$

(4) 学科 3 的优化模型

$$\begin{aligned}&\min \quad J_3=(x_1-z_1)^2+(x_2-z_2)^2+(x_3-z_3)^2 \qquad (15)\\ &\text{s.t.}\quad f_3=C_{f6}x_6^3+C_{f1}x_4x_6^2-C_{f5}x_6^2x_1\\ &\qquad g_j-1.0\le 0\quad j=1,2,3,5,7,8,9,10\end{aligned}$$

该优化问题 ICO 算法的优化模型如下。

(1) 系统级优化模型

$$\min \quad F=f_1+f_2+f_3+(J_1^{*2}+J_2^{*2}+J_3^{*2}) \qquad (16)$$

式中，取惩罚因子$\gamma=1$

(2) 学科 1 的优化模型

$$\begin{aligned}&\min \quad J_1=|x_1-z_1|+|x_2-z_2|+|x_3-z_3| \qquad (17)\\ &\text{s.t.}\quad f_1=C_{f1}x_1x_2^2(C_{f2}x_3^2+C_{f3}x_3-C_{f4})\\ &\qquad g_j-1.0\le 0\quad j=1,2,7,8,9\end{aligned}$$

(3) 学科 2 的优化模型

$$\min \quad J_2 = |x_1 - z_1| + |x_2 - z_2| + |x_3 - z_3| \tag{18}$$

$$\text{s.t.} \quad f_2 = C_{f6}x_7^3 + C_{f1}x_5x_7^2 - C_{f5}x_7^2x_1$$

$$g_j - 1.0 \le 0 \quad j = 1,2,4,6,7,8,9,11$$

(4) 学科 3 的优化模型

$$\min \quad J_3 = |x_1 - z_1| + |x_2 - z_2| + |x_3 - z_3| \tag{19}$$

$$\text{s.t.} \quad f_3 = C_{f6}x_6^3 + C_{f1}x_4x_6^2 - C_{f5}x_6^2x_1$$

$$g_j - 1.0 \le 0 \quad j = 1,2,3,5,7,8,9,10$$

为了便于标准 CO 算法快速的收敛，在标准 CO 算法系统级使用了松弛因子 $\varepsilon = 0.000\ 1$，系统级等式约束变为：$J_1^* \le \varepsilon$，$J_2^* \le \varepsilon$，$J_3^* \le \varepsilon$。两种方法系统级采用可行方向法(CONMIN)，子系统级采用序列二次规划法(NLPQL)：

表 2 列出了 NASA 给出的两种方法优化时的 4 个不同起始设计点，其中 X_1、X_2 和 X_3 为可行域内的设计点，X_4 为可行域外的设计点。

表 2　起始设计点

起始点	齿面宽度 x_1/cm	齿轮模数 x_2/cm	小齿轮齿数 x_3/cm	轴承间距 x_4/cm	轴承间距 x_5/cm	小齿轮轴直径 x_6/cm	大齿轮轴直径 x_7/cm
X_1	2.80	0.71	25.0	7.90	7.599	3.0	5.099
X_2	2.65	0.63	18.0	6.80	6.400	3.0	5.099
X_3	3.5	0.7	17.0	7.30	7.715	3.35	5.287
X_4	2.0	0.4	9.0	4.0	4.5	1.5	3.0

表 3 和表 4 为 4 个起始设计点分别对应标准 CO 算法和 ICO 算法的优化结果。

表 3　CO 算法优化结果

起始点	起始目标函数 f/cm^3	CO 算法					
		最优目标函数 f^*/cm^3	学科 1 约束 J_1^*/cm^3	学科 2 约束 J_2^*/cm^3	学科 3 约束 J_3^*/cm^3	迭代次数 n	最优可行解 x^*
X_1	3 932.589	2 986.984	64.451 68	0.039 932	0.040 147	53	无
X_2	2 311.211	2 987.951	0.004 875	0.001 225	1.238 817	24	(3.6, 0.7, 18.331 3, 7.3, 7.715 1, 3.348 8, 5.286 4)
X_3	2 994.341	2 988.286	0.0	0.006 139	0.012 086	51	(3.535, 0.7, 17.0, 7.3, 7.715 2, 3.349 77, 5.286 6)
X_4	3 228.671	2 988.199	0.025 197	0.001 675	0.006 357	28	(3.579 9, 0.7, 17.324, 7.3, 7.715 2, 3.349 5, 5.286 5)

表 4　ICO 算法优化结果

起始点	起始目标函数 f/cm^3	ICO 算法					
		最优目标函数 f^*/cm^3	学科 1 约束 J_1^*/cm^3	学科 2 约束 J_2^*/cm^3	学科 3 约束 J_3^*/cm^3	迭代次数 n	最优可行解 x^*
X_1	3 932.589	2 986.921	0.021 849	0.095 459	0.091 005	32	(3.501, 0.7, 17.02, 7.3, 7.714, 3.346, 5.286)
X_2	2 311.211	2 987.942	0.025 604	0.020 170	0.040 799	39	(3.495, 0.7, 17.02, 7.3, 7.715 1, 3.348 8, 5.286)
X_3	2 994.341	2 988.306	0.0	0.085	0.085	15	(3.535, 0.7, 17.0, 7.3, 7.715 2, 3.349 8, 5.286)
X_4	3 228.671	2 988.268	0.019 418	0.027 173	0.028 8	22	(3.522, 0.7, 17.17, 7.3, 7.715, 3.349 7, 5.286)

从中可以看到，起始点为 X_1 时，标准 CO 算法没有可行解，学科 1 的一致性约束严重冲突，系统级协调失败。而 ICO 算法取得了理想的优化结果。

起始点为 X_2、X_3 和 X_4 时，虽然两种方法均取得了较理想的优化结果，但是在起始点为 X_2 时，标准 CO 算法学科 3 的一致性约束较大，而 ICO 算法三个学科的一致性约束最大只有 0.040 799，优化协调取得了满意的结果。另外，ICO 算法除了起始点 X_2 时系统级迭代次数比标准 CO 算法多外，其他情况下迭代次数要少的多，尤其是起始点为 X_3 时，运算速度大幅提高。因此，与标准 CO 算法相比，ICO 算法更加稳定可靠，而且优化速度快。

起始点为 X_1 时标准 CO 算法和 ICO 算法的系统级目标函数的优化迭代历史曲线图如图 2、图 3 所示。

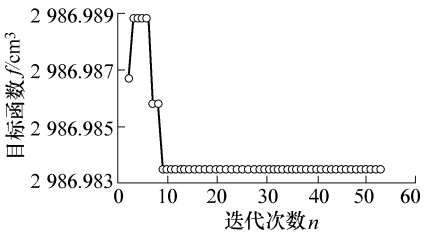

图 2　标准 CO 算法目标函数迭代历史

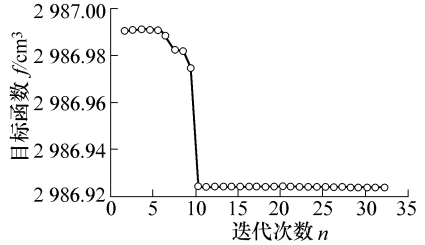

图 3　ICO 算法目标函数迭代历史

4 结论

分析了协同优化算法计算困难的原因，在此基础上提出了一种改良的协同优化算法—ICO 多学科设计优化方法。与标准 CO 算法相比，ICO 算法的优点如下。

(1) 使用 L_1 范数的学科一致性约束解决了标准 CO 算法学科一致性等式约束的引入导致的系统级优化问题的非线性特性增强，不平滑的设计空间和系统级对敏感性信息需求的问题。

(2) 使用 L_1 范数的系统级罚函数取代原有等式约束，避免了系统级约束的 Jacobian 矩阵为零对 Kuhn-Tucker 条件的破坏，保证子系统的拉格朗日乘子不为零，同时解决约束的超定问题。

(3) 在 ICO 算法中采用了快速启动方法，收敛速度快。

算例试验结果表明，ICO 算法比标准 CO 算法更加稳定、可靠，计算效率明显提高。但是对于不同的优化问题，系统级罚函数的惩罚因子 γ 取值不同对优化过程会有不同的影响，通常 γ 取值越大优化过程波动越大，建议在[0.5，2]区间内进行选择，如何方便快捷地选取最合适的惩罚因子值还有待进一步研究。松弛因子 ε 取值越小，优化结果越好，但优化迭代次数越多，可根据优化问题对时间和结果精度的不同要求进行选取。另外，ICO 算法还需要在大型复杂工程系统设计优化中进行应用、检验及进一步完善。

参 考 文 献

[1] HAN M H, DENG J T. Multidisciplinary design optimization methods for complex engineering systems[C]// Proceedings of the International Conference on Agile Manufacturing, ICAM2003, Beijing, 2003：347-354.

[2] BALLING R J. Execution of multidisciplinary design optimization approaches on common test problems[J]. AlAA Journal，1997，35(1)：178-186.

[3] ALEXANDROV N M, KODIYALAM S. Initial results of an MDO method evaluation study[R]. AIAA-98-4884，1998.

[4] 韩明红. 复杂工程系统多学科优化方法与技术研究[D]. 北京：北京航空航天大学，2004.

[5] MANNING V M. Large-scale design of supersonic aircraft via collaborative optimization[D]. San Francisco：Stanford University，1999.

[6] BRAUN R D, GAGE P, KROO I. Implementation and performance issues in collaborative optimization[R]. AIAA-96-4017，1996.

[7] ALEXANDROV N M, LEWIS R M. Comparative properties of collaborative optimization and other approaches to MDO[R]. NASA/CR-1999-209354，1999.

[8] SOBIESKI L P, KROO I. Collaborative optimization using response surface estimation[R]. AIAA-98-0915，1998.

[9] 韩明红，邓家褆. 复杂工程系统多学科设计优化集成环境研究[J]. 机械工程学报，2004，40(9)：100-105.

[10] KODIYALAM S. Evaluation of methods for multidisciplinary design optimization (MDO), Phase I[R]. NASA/CR-1998-208716, 1998.

IMPROVEMENT OF COLLABORATIVE OPTIMIZATION

HAN Minghong DENG Jiati

(*School of Mechanical Engineering and Automation, Beijing University of Aeronautics and Astronautics, Beijing 100083*)

Abstract：Collaborative optimization (CO) is the most widely used MDO method, which has good effect in application, however some serious computational difficulties are found in its application. Reasons that cause computational difficulties in collaborative optimization are analyzed and a new improved collaborative optimization method (ICO) is presented. Improved subsystem consistency constraint, system-level penalty function, and a quick-start strategy are used in ICO. These measures overcome the computational difficulties of CO better. Experimental results show that the robustness, reliability and computing efficiency of ICO are higher than CO.

Key words：Optimization Computer aided design Improvement Collaborative optimization Multidisciplinary design optimization

作者简介：韩明红，男，1974 年出生，博士后。主要研究方向为多学科设计优化技术、优化算法、产品设计理论，设计自动化集成环境。

E-mail：hanminghong@163.com