

# 用于约束多目标优化问题的双群体差分进化算法

孟红云<sup>1)</sup> 张小华<sup>2)</sup> 刘三阳<sup>1)</sup>

<sup>1)</sup>(西安电子科技大学应用数学系 西安 710071)

<sup>2)</sup>(西安电子科技大学智能信息处理研究所 西安 710071)

**摘 要** 首先给出一种改进的差分进化算法,然后提出一种基于双群体搜索机制的求解约束多目标优化问题的差分进化算法.该算法同时使用两个群体,其中一个用于保存搜索过程中找到的可行解,另一个用于记录在搜索过程中得到的部分具有某些优良特性的不可行解,避免了构造罚函数和直接删除不可行解.此外,文中算法、NSGA-II和SPEA的时间复杂度的比较表明,NSGA-II最优,文中算法与SPEA相当.对经典测试函数的仿真结果表明,与NSGA-II相比较,文中算法在均匀性及逼近性方面均具有一定的优势.

**关键词** 差分进化算法;约束优化问题;多目标优化问题

中图法分类号 TP18

## A Differential Evolution Based on Double Populations for Constrained Multi-Objective Optimization Problem

MENG Hong-Yun<sup>1)</sup> ZHANG Xiao-Hua<sup>2)</sup> LIU San-Yang<sup>1)</sup>

<sup>1)</sup>(Department of Applied Mathematics, Xidian University, Xi'an 710071)

<sup>2)</sup>(Institute of Intelligent Information Processing, Xidian University, Xi'an 710071)

**Abstract** An improved differential evolution approach is given first, and a new algorithm based on double populations for Constrained Multi-objective Optimization Problem (CMOP) is presented. In the proposed algorithm, two populations are adopted, one is for the feasible solutions found during the evolution, and the other is for infeasible solutions with better performance which are allowed to participate in the evolution with the advantage of avoiding difficulties such as constructing penalty function and deleting infeasible solutions directly. In addition, the time complexity of the proposed algorithm, NSGA-II and SPEA are compared, which show the best is NSGA-II, followed by SPEA and the proposed algorithm simultaneously. The experiments on benchmarks indicate that the proposed algorithm is superior to NSGA-II in the measure of GD and SP.

**Keywords** differential evolution; constrained optimization problem; multi-objective optimization problem

## 1 引 言

达尔文的自然选择机理和个体的学习能力推动了进化算法的出现和发展,用进化算法求解优化问

题已成为一个研究的热点<sup>[1-3]</sup>.但目前研究最多的却是无约束优化问题.然而,在科学研究和工程实践中,许多实际问题最终都归结为求解一个带有约束条件的函数优化问题,因此研究基于进化算法求解约束优化问题是非常有必要的.不失一般性,以最小

化问题为例,约束优化问题(Constrained Optimization Problem, COP)可定义如下:

$$\begin{aligned} \min_{\mathbf{x} \in R^n} F(\mathbf{x}) &= (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_k(\mathbf{x})) \\ (\text{COP}) \text{ s. t. } g_i(\mathbf{x}) &\leq 0, i = 1, 2, \dots, p \\ h_j(\mathbf{x}) &= 0, j = 1, 2, \dots, q \end{aligned} \quad (1)$$

其中,  $F(\mathbf{x})$  为目标函数,  $g_i(\mathbf{x}), h_j(\mathbf{x})$  称为约束条件,  $\mathbf{x} = (x_1, x_2, \dots, x_n) \in R^n$  称为  $n$  维决策向量. 将满足所有约束条件的解空间  $S$  称为式(1)的可行域. 特别的, 当  $k=1$  时, 式(1)为单目标优化问题; 当  $k>1$  时, 式(1)为多目标优化问题.  $g_i(\mathbf{x})$  为第  $i$  个不等式约束,  $h_j(\mathbf{x})$  是第  $j$  个等式约束. 另一方面, 对于等式约束  $h_j(\mathbf{x})=0$  可通过容许误差(也称容忍度)  $\delta>0$  将它转化为两个不等式约束:

$$\begin{cases} h_j(\mathbf{x}) - \delta \leq 0 \\ -h_j(\mathbf{x}) - \delta \leq 0 \end{cases} \quad (2)$$

故在以后讨论问题时,仅考虑带不等式约束的优化问题. 进一步, 如果  $\mathbf{x}$  使得不等式约束  $g_i(\mathbf{x})=0$ , 则称约束  $g_i(\mathbf{x})$  在  $\mathbf{x}$  处是积极的. 在搜索空间  $S$  中, 满足约束条件的决策变量  $\mathbf{x}$  称为可行解, 否则称为不可行解.

**定义 1(全局最优解).**  $\mathbf{x}^* = (x_1^*, x_2^*, \dots, x_n^*)$  是 COP 的全局最优解, 是指  $\mathbf{x}^* \in S$  且  $F(\mathbf{x}^*)$  不劣于可行域内任意解  $\mathbf{y}$  所对应的目标函数  $F(\mathbf{y})$ , 表示为  $F(\mathbf{x}^*) \leq F(\mathbf{y})$ . 对于单目标优化问题,  $F(\mathbf{x}^*) \leq F(\mathbf{y})$  等价于  $F(\mathbf{x}^*) \leq F(\mathbf{y})$ , 而对于多目标优化问题是指不存在  $\mathbf{y}$ , 使得  $F(\mathbf{y})$  Pareto 优于  $F(\mathbf{x}^*)$ .

目前, 进化算法用于无约束优化问题的文献居多, 与之比较, 对约束优化问题的研究相对较少<sup>[4-6]</sup>. 文献[7]对当前基于进化算法的各种约束处理方法进行了较为详细的综述. 对于约束优化问题的约束处理方法基本上分为两类: 基于罚函数的约束处理技术和基于多目标优化技术的约束处理技术. 由于罚函数法在使用中不需要约束函数和目标函数的解析性质, 因此经常被应用于约束优化问题, 但该类方法对罚因子有很强的依赖性, 需要根据具体问题平衡罚函数与目标函数. 为了避免复杂罚函数的构造, Verdegay 等<sup>[8]</sup>将进化算法中的竞争选择用于约束处理, 并在比较两个解的性能时提出了 3 个准则, 但他的第 3 个准则“可行解优于不可行解”这一准则合理性不强. 然而该文的这一准则却为进化算法求解约束优化问题提供了新思路, 获得了良好效果.

因为在现实中存在一大类约束优化问题, 其最优解位于约束边界上或附近, 对于这类问题, 在最优解附近的不可行解的适应值很可能优于位于可行域内部的大部分可行解的适应值, 因此无论从适应值本身还是从最优解的相对位置考虑, 这样的不可行解对找到最优解都是很有帮助的, 故如何有效利用搜索过程中的部分具有较好性质的不可行解是解决此类问题的难点之一. 基于以上考虑, 本文拟给出一种求解约束多目标优化问题的基于双群体机制的差分进化算法, 并对文中算法的时间复杂度与 NSGA-II<sup>[9]</sup> 和 SPEA<sup>[10]</sup> 进行比较, 最后用实验仿真说明文中算法的可行性及有效性.

## 2 用于约束优化的双群体差分进化算法

### 2.1 差分进化算法

差分进化算法是一类简单而有效的进化算法, 已被成功应用于求解无约束单目标和多目标优化问题<sup>[11-14]</sup>. 该算法在整个运行过程中保持群体的规模不变, 它也有类似于遗传算法的变异、交叉和选择等操作, 其中变异操作定义如下:

$$\mathbf{C} = \mathbf{P}_{r1} + F \cdot (\mathbf{P}_{r2} - \mathbf{P}_{r3}) \quad (3)$$

其中,  $\mathbf{P}_{r1}, \mathbf{P}_{r2}, \mathbf{P}_{r3}$  为从进化群体中随机选取的互不相同的 3 个个体,  $F$  为位于区间  $[0.5, 1]$  中的参数. 式(3)表示从种群中随机取出的两个个体  $\mathbf{P}_{r2}, \mathbf{P}_{r3}$  的差, 经参数  $F$  放大或缩小后被加到第 3 个个体  $\mathbf{P}_{r1}$  上, 以构成新的个体  $\mathbf{C} = (c_1, c_2, \dots, c_n)$ . 为了增加群体的多样性, 交叉操作被引入差分进化算法, 具体操作如下:

针对父代个体  $\mathbf{P}_r = (x_1, x_2, \dots, x_n)$  的每一分量  $x_i$ , 产生位于区间  $[0, 1]$  中的随机数  $p_i$ , 根据  $p_i$  与参数  $CR$  的大小关系确定是否用  $c_i$  替换  $x_i$ , 以得到新的个体  $\mathbf{P}'_r = (x'_1, x'_2, \dots, x'_n)$ , 其中  $x'_i = \begin{cases} c_i, & p_i < CR \\ x_i, & p_i \geq CR \end{cases}$ .

如果新个体  $\mathbf{P}'_r$  优于父代个体  $\mathbf{P}_r$ , 则用  $\mathbf{P}'_r$  来替换  $\mathbf{P}_r$ ; 否则保持不变. 在差分进化算法中, 选择操作采取的是贪婪策略, 即只有当产生的子代个体优于父代个体时才被保留, 否则, 父代个体被保留至下一代.

大量研究与实验发现差分进化算法在维护群体的多样性及搜索能力方面功能较强, 但收敛速度相对较慢, 因此本文拟给出一种改进的差分进化算法用于多目标优化问题, 仿真实验表明, 改进的差分进化算法在不破坏原有算法维护群体多样性的前提

下,可改善差分进化算法的收敛速度.

## 2.2 基于双群体的差分进化算法

### 2.2.1 基本概念

以下仅讨论带不等式约束的多目标优化问题

$$\min F(\mathbf{x}) = (f_1(\mathbf{x}), \dots, f_k(\mathbf{x}))$$

$$\text{s. t. } g_i(\mathbf{x}) < 0, i = 1, 2, \dots, p;$$

$$\mathbf{x} = (x_1, \dots, x_n), l_j \leq x_j \leq u_j, j = 1, 2, \dots, n \quad (4)$$

**定义 2.**  $\mathbf{x}$  称为式(4)的不可行解,是指至少存在一个  $1 \leq i \leq p$ , 满足  $g_i(\mathbf{x}) \geq 0$ .

**定义 3.**  $\mathbf{x}$  违反约束的强度,即约束违反度函数定义为  $P(\mathbf{x}) = \sum_{i=1}^p \max(0, g_i(\mathbf{x}))^\sigma$ , 本文取  $\sigma=2$ .

**定义 4.**  $\mathbf{x}$  违反约束的数目  $N(\mathbf{x}) = \sum_{i=1}^p \text{Num} \cdot (g_i(\mathbf{x}))$ , 其中  $\text{Num}(x) = \begin{cases} 0, & x < 0 \\ 1, & x \geq 0 \end{cases}$ .

**定义 5.** 不可行解  $\mathbf{x}$  优于不可行解  $\mathbf{y}$ , 是指  $\mathbf{x}$  的约束向量  $(P(\mathbf{x}), N(\mathbf{x}))$  Pareto 优于  $\mathbf{y}$  的约束向量  $(P(\mathbf{y}), N(\mathbf{y}))$ .

### 2.2.2 基本思想

由上一节分析可知,在搜索过程中遇到的不可行解不能简单丢掉.因此,在设计算法时不但要考虑算法的收敛速度,而且还必须保证群体中可行解的优势地位;另一方面,对于多目标优化问题,维持搜索群体的多样性与考虑群体的收敛速度是同等重要的.基于此考虑,本节采用基于双群体的差分进化算法求解约束多目标优化问题,其中群体  $PoP_f$  用来保存搜索过程中遇到的可行解,  $PoP_c$  用来保存搜索过程中遇到的占优不可行解,同时  $PoP_f$  具有较强的记忆功能,可记忆  $PoP_f$  中每一个体搜索到的最优可行解和整个群体  $PoP_f$  到目前为止搜索到的最优可行解,分别记为  $lbest$  和  $gbest$ , 其中  $lbest$  表示个体对自身的思考和认知,  $gbest$  表示个体间的信息交流,这一点和 PSO 算法类似.与此同时,我们还通过一种改进的差分进化算法产生新的群体,在产生新群体的过程中,群体  $PoP_c$  中的部分个体参与了个体再生,并通过新生成的个体更新  $PoP_f$ ,  $PoP_c$ ,  $lbest$  和  $gbest$ .

为了避免性能较优的不可行解被删除,本文拟采用双群体搜索机制,其中群体  $PoP_f = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{N_1}\}$  用于记录可行解,群体  $PoP_c = \{\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_{N_2}\}$  记录不可行解,  $N_1, N_2$  分别为群体  $PoP_f$  与  $PoP_c$  的规模,满足  $N_1 > N_2$ ,  $lbest = \{\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_{N_1}\}$  和

$gbest = \{\mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_{N_2}\}$  分别为群体  $PoP_f$  中每一个个体  $\mathbf{x}_i$  搜索到最优可行解  $\mathbf{z}_i$  和群体  $PoP_f$  迄今为止搜索到最优可行解.

### 2.2.3 改进的差分进化算法

为了维护群体  $PoP_f$  的多样性和收敛性,同时有效地利用已搜索到的不可行解的某些优良特性,下面给出一种改进的差分进化算法,并通过以下两种方式产生新的个体.

方法 1:  $\mathbf{C} = \mathbf{x}_{r_1} + F_1(\mathbf{z}_{r_2} - \mathbf{x}_{r_2}) + F_2(\mathbf{g}_{r_4} - \mathbf{x}_{r_3})$ , 其中  $\mathbf{x}_{r_1}, \mathbf{x}_{r_2}, \mathbf{x}_{r_3} \in PoP_f$ ,  $\mathbf{z}_{r_2} \in lbest$ ,  $\mathbf{g}_{r_4} \in gbest$ .

方法 2:  $\mathbf{C} = \mathbf{x}_{r_1} + F_1(\mathbf{z}_{r_2} - \mathbf{y}_{r_3}) + F_2(\mathbf{g}_{r_4} - \mathbf{y}_{r_5})$ , 其中  $\mathbf{x}_{r_1} \in PoP_f$ ,  $\mathbf{y}_{r_3}, \mathbf{y}_{r_5} \in PoP_c$ ,  $\mathbf{z}_{r_2} \in lbest$ ,  $\mathbf{g}_{r_4} \in gbest$ .

方法 1 的目的在于通过向最优个体学习,改善算法的收敛速度.方法 2 的主要目的在于和不可行个体进行信息交流,共享不可行解的一些优良特性,增加群体的多样性.在具体操作过程中,首先用改进的差分进化算法产生新的个体  $\mathbf{C} = (c_1, c_2, \dots, c_n)$ , 然后针对父代个体  $\mathbf{P}_r = (x_1, x_2, \dots, x_n)$  的每一个分量  $x_i$ , 产生位于区间  $[0, 1]$  中的随机数  $p_i$ , 根据  $p_i$  与参数  $CR$  的大小关系确定是否用  $c_i$  来替换  $x_i$ , 得到新的个体  $\mathbf{P}'_r = (x'_1, x'_2, \dots, x'_n)$ .

如果  $\mathbf{P}'_r$  是可行解,而且  $PoP_f$  的规模小于给定规模  $N_1$ , 则可直接将  $\mathbf{P}'_r$  插入  $PoP_f$ ; 如果插入后的群体的规模大于给定规模  $N_1$ , 首先两两比较  $PoP_f$  中的个体, 如果存在两个个体  $\mathbf{x}_i, \mathbf{x}_j \in PoP_f$ , 满足  $F(\mathbf{x}_i)$  Pareto 优于  $F(\mathbf{x}_j)$ , 则将个体  $\mathbf{x}_j$  删除, 如果不存在, 也就是说集合  $PoP_f$  中任意两个个体所对应的目标向量都不可比较, 则计算  $PoP_f$  中任意两个个体间的距离, 随机删除距离最小的两个个体中的一个.

如果  $\mathbf{P}'_r$  是不可行解, 而且  $PoP_c$  的规模小于给定规模  $N_2$ , 则可直接将  $\mathbf{P}'_r$  插入群体  $PoP_c$  中; 如果  $PoP_c$  等于给定规模阈值  $N_2$ , 计算插入  $\mathbf{P}'_r$  后的群体  $PoP_c$  中任意两个个体的约束向量, 如果存在两个个体  $\mathbf{y}_i, \mathbf{y}_j \in PoP_c$ , 满足约束向量  $(P(\mathbf{y}_i), N(\mathbf{y}_i))$  Pareto 优于约束向量  $(P(\mathbf{y}_j), N(\mathbf{y}_j))$ , 则删除  $\mathbf{y}_j$ ; 如果不存在, 则删除满足  $P(\mathbf{y}_i) = \max_{\mathbf{y} \in PoP_c} P(\mathbf{y})$  的个体  $\mathbf{y}_i$ .

经过以上操作,群体  $PoP_f$  和  $PoP_c$  的规模不会大于给定规模阈值.最后利用新生成的群体  $PoP_f$  更新最优个体集合  $lbest$  和  $gbest$ , 群体  $gbest$  的更新方法和 SPEA 算法中外部群体的更新方法相同, 而  $lbest$  的更新方法如下: 如果新生成的可行解

$P'_r$  Pareto 优于对应的局部最优解  $z_i$ , 则用  $P'_r$  替换  $z_i$ ; 否则不予替换。

### 2.3 算法的基本流程

综上所述, 基于双群体的差分进化算法的约束处理技术的流程可表示如下:

1. 随机生成  $N$  ( $N \geq N_1 + N_2$ ) 个个体, 判断每一个体的可行性, 然后根据个体可行性将其插入到对应的群体  $PoP_f$  或  $PoP_c$  中; 并初始化  $lbest$  和  $gbest$  及参数  $CR$  和  $P_d$ .
2. 判断搜索是否结束, 如果结束, 转向步 5, 否则转向步 3.
3. 生成随机数  $p$ , 如果  $p \geq P_d$ , 根据方法 1, 生成新的个体  $P'_r$ ; 否则, 根据方法 2 生成新的个体  $P'_r$ , 如果  $P'_r$  是可行解, 将  $P'_r$  插入到  $PoP_f$  中; 否则  $P'_r$  插入到  $PoP_c$  中, 反复执行直到生成  $N_1$  个可行解.
4. 根据新生成的群体  $PoP_f$  更新最优个体集  $lbest$  和  $gbest$ , 转向步 2.
5. 输出最优解集  $gbest$ .

## 3 算法分析

### 3.1 算法的性能衡量

约束优化问题的算法性能衡量可分为两部分, 一部分为最终获得的最优解的性能的衡量, 如通过 GD<sup>[15]</sup> 来度量最优群体的逼近性, SP<sup>[16]</sup> 来衡量最优解的分布均匀性, 或通过计算目标函数的次数衡量算法的复杂度和算法的收敛速度. 另一部分是针对约束优化问题来衡量群体的多样性, Koziel 和 Michalewicz<sup>[17]</sup> 给出一种多样性度量准则  $\rho$ , 其定义如下:

$$\rho = \frac{|F|}{|S|} \quad (5)$$

其中,  $|F|$  表示每一次搜索过程中生成的可行解的数目,  $|S|$  为所生成的所有个体的数目. 相应地, 为了衡量群体中的不可行解违反约束的强度, 可采用约束违反度函数的均值来度量:

$$P_{avg} = \sum_{y \in PoP_c} P(y) / |PoP_c| \quad (6)$$

其中,  $|PoP_c|$  表示集合  $PoP_c$  所包含元素的数目. 然而在实际问题中, 决策者往往只对某一范围的最优解感兴趣, 故下边只评价本文算法对标准测试函数最终获得的最优解集的逼近性与均匀性, 并与 NSGA-II 进行比较.

### 3.2 算法的时间复杂性分析

我们仅考虑种群规模对算法时间复杂度的影响, 设可行群体  $PoP_f$  的规模为  $N_1$ , 不可行群体  $PoP_c$  的规模为  $N_2$ , 群体  $lbest$  的规模为  $N_1$ , 群体

$gbest$  的最大规模为  $N_3$ , 则文中算法迭代一次的最坏时间复杂度可计算如下:

算法中重组和变异操作的时间复杂度为  $O(N_1)$ ; 判断进化群体中个体可行性所需时间复杂度为  $O(N_1)$ ; 更新群体  $PoP_f$ ,  $PoP_c$  和  $lbest$  的时间复杂度分别为  $O(N_1)$ ,  $O(N_2)$  和  $O(N_1)$ ; 计算群体  $PoP_f$  和  $gbest$  的适应度所需时间复杂度为  $O(N_1 + N_3)$ ; 用于更新最优群体  $gbest$  的最坏时间复杂度为  $O((N_1 + N_3)^2)$ ; 保持最优群体  $gbest$  和进化群体  $PoP_f$  多样性的最坏时间复杂度为  $O((N_1 + N_3)^3) + (N_1 + N_3) \log(N_1 + N_3)$ ; 则算法迭代一次所需的最坏时间复杂度为

$$O(N_1) + O(N_1) + O(N_1) + O(N_2) + O(N_1) + O(N_1 + N_3) + O((N_1 + N_3)^2) + O((N_1 + N_3)^3) + O((N_1 + N_3) \log(N_1 + N_3)) \quad (7)$$

上述复杂度可化简为

$$O((N_1 + N_3)^3) + O((N_1 + N_3)^2) + O((N_1 + N_3) \log(N_1 + N_3)) \quad (8)$$

设  $N$  为所有种群的规模, 令  $N = N_1 + N_2 + N_3$ , 则本文算法的最坏时间复杂度

$$O((N_1 + N_3)^3) + O((N_1 + N_3)^2) + O((N_1 + N_3) \log(N_1 + N_3)) < O(N^3) \quad (9)$$

NSGA-II<sup>[9]</sup> 和 SPEA<sup>[10]</sup> 是多目标进化算法中两个最具有代表性的优秀算法, 这两个算法的最坏时间复杂度分别为  $O(N^2)$  和  $O((N + \bar{N})^3)$ , 其中  $N, \bar{N}$  分别为进化种群规模和外部种群集的规模. 因而, SPEA 和本文算法的最坏时间复杂度为  $O(N^3)$ , 这比 NSGA-II 的时间复杂度稍高一些, 但接下来的实验结果告诉我们, 本文算法的均匀性及逼近性却明显优于 NSGA-II. 事实上, SPEA 和本文算法的时间复杂度主要用于环境选择 (environmental selection) 上, 如果文中对  $gbest$  采取 NSGA-II 中的多样性保持策略, 则本文算法的复杂度将降至  $O(N^2)$ .

## 4 实验结果与分析

### (1) 测试函数与参数设置

为了验证本文给出的算法的可行性, 我们采用 Deb<sup>[18]</sup> 建议的用来测试约束多目标优化算法性能的 4 个常见的测试函数来检验本文算法的性能. 可行解集合的规模  $N_1 = 100$ , 不可行解集合的规模  $N_2 = 10$ . 初始化时, 随机生成个体的数目  $N = 200$ , 参数  $CR = 0.25$ ,  $P_d = 0.5$ .  $F_1, F_2$  为位于区间  $[0, 5, 1, 0]$  中的一致随机数. Deb 给出的测试函数可用统一的

解析式表示,即

$$\begin{cases} \min F(x) = \min \begin{Bmatrix} f_1(\bar{x}) \\ f_2(\bar{x}) \end{Bmatrix} = \\ \min \begin{Bmatrix} x_1 \\ c(\bar{x}) \left[ 1 - \frac{f_1(\bar{x})}{c(\bar{x})} \right] \end{Bmatrix}, \\ g(\bar{x}) = \cos(\theta) [f_2(\bar{x}) - e] - \sin(\theta) f_1(\bar{x}) \geq \\ a |\sin\{b\pi[\sin(\theta) [f_2(\bar{x}) - e] + \cos(\theta) f_1(\bar{x})]^c\}|^d. \end{cases}$$

其中,  $c(x) = 41 + \sum_{i=2}^5 [x_i^2 - 10\cos(2\pi x_i)]$ ,  $0 \leq x_1 \leq 1, -5 \leq x_i \leq 5, i = 2, 3, 4, 5$ . 测试函数选取不同的参数  $\theta, a, b, c, d, e$  时, 所构造的测试函数性质不同, 可行解和不可行解的分布也不同, 最终导致全局 Pareto 最优解集的不同. 其中通过控制参数  $b$  的大小, 可以控制 Pareto 前端不连续的段数,  $b$  越大段数越多; 而较小参数  $d$  可以使得每一不连续 Pareto 前端仅包含一个 Pareto 点; 参数  $a$  调节连续可行域到 Pareto 前端的点的距离,  $a$  越大距离越远, 其作用在于调节问题求解的难度; 参数  $c$  的作用在于改变分段 Pareto 前端之间的分布特性, 当  $c = 1$  时, Pareto 前端为均匀分布; 当  $c > 1$  时, Pareto 前端向  $f_1$  较大的方向移动; 当  $c < 1$  时, 则 Pareto 前端向  $f_2$  较大的方向移动. 基于以上分析我们选取不同的参数构造 4 个常用的测试函数检验本节算法的性能, 这些测试函数的参数取值具体如下.

测试函数 CTP1:

$\theta = 0.1\pi, a = 40, b = 0.5, c = 1, d = 2, e = -2$ . 可行解、不可行解、全局 Pareto 前端分布如图 1 所示.

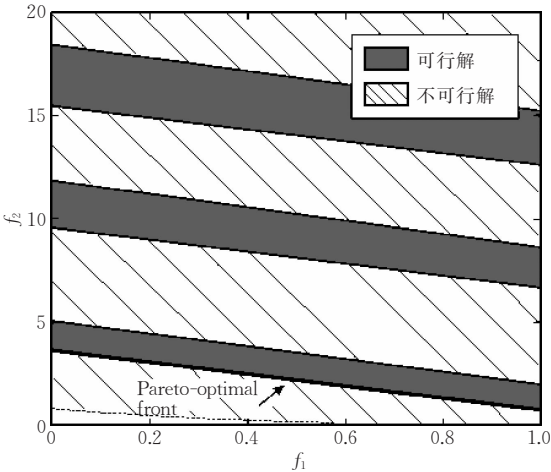


图 1 测试函数 CTP1 在目标空间的示意图

测试函数 CTP2:

$\theta = -0.05\pi, a = 40, b = 5, c = 1, d = 6, e = 0$ . 可行解、不可行解、全局 Pareto 前端分布如图 2 所示.

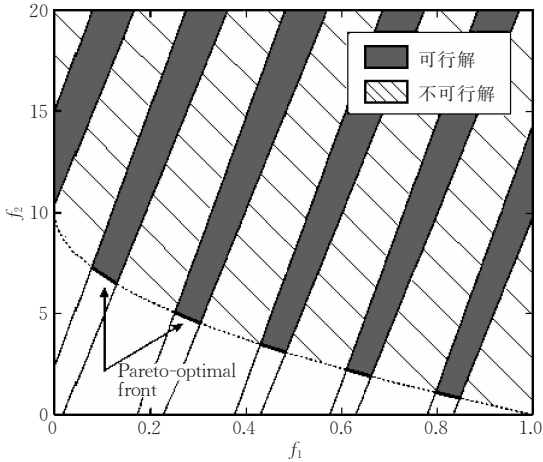


图 2 测试函数 CTP2 在目标空间的示意图

测试函数 CTP3:

$\theta = -0.2\pi, a = 0.2, b = 10, c = 1, d = 6, e = 1$ . 可行解、不可行解、全局 Pareto 前端分布如图 3 所示.

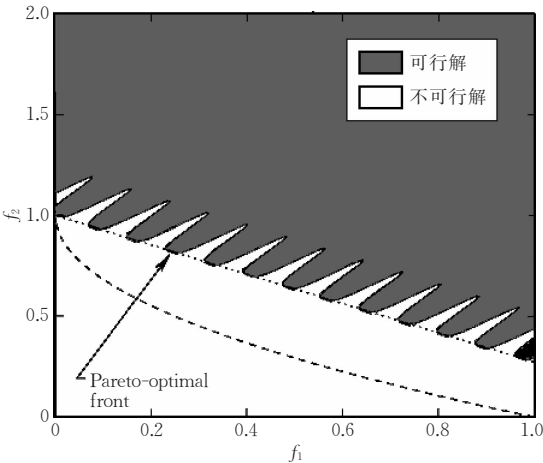


图 3 测试函数 CTP3 在目标空间的示意图

测试函数 CTP4:

$\theta = -0.2\pi, a = 0.1, b = 10, c = 1, d = 0.5, e = 1$ . 可行解、不可行解、全局 Pareto 前端分布如图 4 所示.

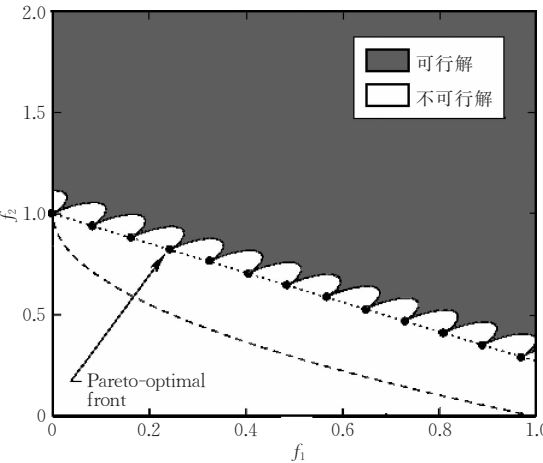


图 4 测试函数 CTP4 在目标空间的示意图

(2) 实验结果与分析

在相同的测试函数和目标函数计算次数下,将本文算法和经典的 NSGA-II 算法进行比较,并将各算法独立运行 30 次,然后统计两种算法所得 Pareto 最优解集的均匀性(Spacing,SP)与逼近性(Generational Distance,GD)的最好、最差、均值、方差和中间值,以此作为衡量算法性能的标准.由于真实 Pareto 最优集是未知的,故我们将两种算法所得的 60 个近似 Pareto 最优解集之并集的 Pareto 滤集作为真实 Pareto 最优解集的逼近,其中测试函数 CTP1,CTP2,CTP3 的函数值计算次数为 10200,而 CTP4 的函数值计算次数为 610014.这里,集合  $I$  的 Pareto 滤集  $\text{Pareto}(I)$  定义为  $\text{Pareto}(I)=\{x|x\in I, \nexists y\in I, F(y)\leq F(x)\}$ .图 5~图 8 为从 30 次运行中随机选择的一次运行结果,从实验曲线可以看到本文算法求出的 Pareto 前端在逼近性方面要优于 NSGA-II.

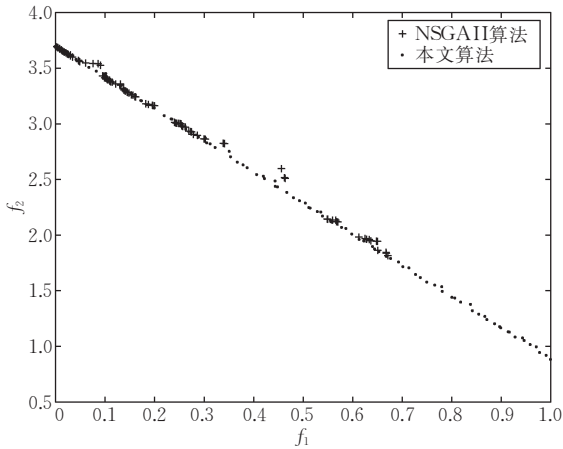


图 5 测试函数 CTP1 的 Pareto 前端

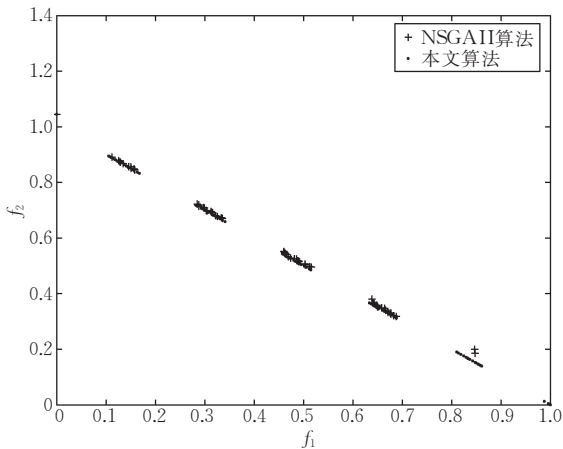


图 6 测试函数 CTP2 的 Pareto 前端

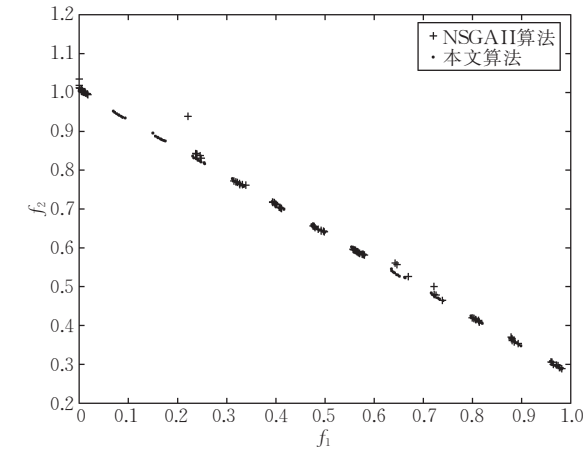


图 7 测试函数 CTP3 的 Pareto 前端

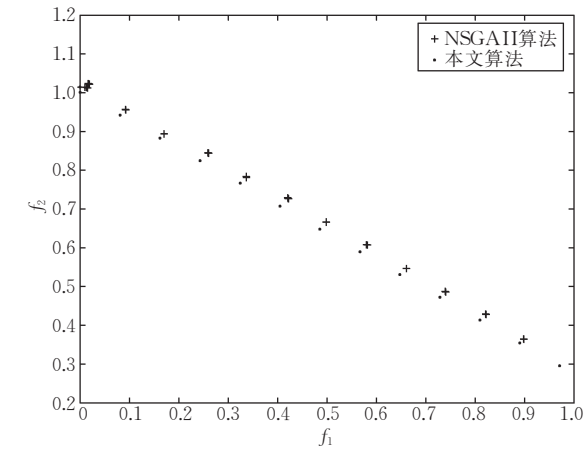


图 8 测试函数 CTP4 的 Pareto 前端

为了进一步定量地评价两种算法的逼近性与均匀性,表 1~表 4 给出了两种算法对上述 4 个测试函数的  $SP,GD$  的统计结果,从表中数据容易看出,在解集的逼近性和均匀性方面本文算法对 4 个测试函数的标准方差都明显小于经典的 NSGA-II 算法,这说明本文的算法性能更稳定.另一方面,上述定量的度量结果也表明在搜索过程中适当地运用性能较优的不可行解的信息不仅有助于保持群体的多样性,而且增强了算法的搜索功能,并在一定程度上起到了维持解集的均匀性的作用.

表 1 测试函数 CTP1 评价准则的统计结果

	SP		GD	
	NSGA-II 算法	本文算法	NSGA-II 算法	本文算法
最优值	0.4285	0.5187	0.0005	9.821e-05
最差值	0.7179	0.6138	0.0021	5.367e-04
平均值	0.5749	0.5705	0.0017	2.0625e-004
中间值	0.5694	0.5684	0.0015	2.636e-04
Std.Dev.	0.1842	0.1043	0.0013	0.0003

表 2 测试函数 CTP2 评价准则的统计结果

	SP		GD	
	NSGA-II 算法	本文算法	NSGA-II 算法	本文算法
最优值	0.3275	0.2689	0.0008	1.547e-05
最差值	0.4121	0.3351	0.0017	1.784e-04
平均值	0.3924	0.2965	0.001	2.306e-7
中间值	0.3732	0.2974	0.0013	8.033e-05
Std.Dev.	0.2157	0.0813	0.0011	0.0001

表 3 测试函数 CTP3 评价准则的统计结果

	SP		GD	
	NSGA-II 算法	本文算法	NSGA-II 算法	本文算法
最优值	0.5219	0.5187	0.0005	9.821e-05
最差值	0.7451	0.6138	0.0021	5.367e-04
平均值	0.3699	0.2965	0.0012	8.0325e-005
中间值	0.3791	0.2834	0.0013	2.636e-04
Std.Dev.	0.2312	0.1813	0.0013	0.0002

表 4 测试函数 CTP4 评价准则的统计结果

	SP		GD	
	NSGA-II 算法	本文算法	NSGA-II 算法	本文算法
最优值	0.2753	0.2245	0.0011	1.3232e-8
最差值	0.5634	0.5286	0.0036	3.371e-7
平均值	0.3490	0.3426	0.0023	4.6452e-8
中间值	0.3576	0.3381	0.0023	5.173e-8
Std.Dev.	0.1278	0.1138	0.0030	0.0001

5 结 论

本文借助粒子群算法的基本思想给出了一种改进的差分进化算法,在适当的利用部分优良不可行个体的基础上,提出了用于约束多目标优化问题的双群体差分进化算法.该算法中的两个群体分别用于记录进化过程中的可行解及部分性能较优的不可行解,其优点在于可以充分利用每次迭代产生的子代个体的信息.此外,还对文中算法的时间复杂度与 NSGA-II 和 SPEA 算法进行比较.经典测试函数的数值仿真结果表明,本文算法无论在解集的逼近性及均匀性方面都优于 NSGA-II 算法,这表明文中提出的基于双群体的差分进化算法在求解带约束的多目标优化问题方面有一定的优势.

正如“没有免费的午餐定理”<sup>[19]</sup>所指出的,任何一种算法不可能在所有的性能方面占尽优势,虽然本文算法在求解约束多目标优化问题方面具有一定的优势,但计算量要稍高于 NSGA-II 算法.接下来我们的研究将致力于如何降低算法的时间复杂度及本文算法的实际应用.

参 考 文 献

[1] Gen Mitsuo, Cheng Run-Wei. Genetic Algorithms & Engi-

neering Design. New York: John Wiley & Sons, Inc., 1997

[2] Glover Fred. Heuristics for Integer programming using surrogate constraints. Decision Sciences, 1977, 8(1): 156-166

[3] Goldberg D E. Genetic in Search, Optimization and Machine Learning. Reading, Massachusetts: Addison-Wesley, 1989

[4] Wang Yue-Xuan, Liu Lian-Chen et al. Constrained multiobjective optimization evolutionary algorithm. Journal of Tsinghua University (Sci & Tech), 2005, 45(1): 103-106(in Chinese)

(王跃宣,刘连臣等.处理带约束的多目标优化进化算法.清华大学学报(自然科学版),2005,45(1):103-106)

[5] Zhang Yong-De, Huang Sha-Bai. On ant colony algorithm for solving multiobjective optimization problems. Control and Decision, 2005, 20(2): 170-174(in Chinese)

(张勇德,黄莎白.多目标优化问题的蚁群算法研究.控制与决策,2005,20(2):170-174)

[6] Gao Yu-Gen, Cheng Feng, Wang Can, Wang Guo-Biao. A new improved genetic algorithms based on converting infeasible individuals into feasible ones and its property analysis. Acta Electronica Sinica, 2006, 34(3): 638-641(in Chinese)

(高玉根,程峰,王灿,王国彪.基于违约解转化法的遗传算法及其性能分析.电子学报,2006,34(3):638-641)

[7] Coello C A. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms; A survey of the state of the art, computer methods in applied mechanics and engineering, 2002, 191(11-12): 1245-1287

[8] Jiménez F, Verdegay J L. Evolutionary techniques for constrained optimization problems//Zimmermann Hans-Jürgen ed. Proceedings of the 7th European Congress on Intelligent Techniques and Soft Computing (EUFIT'99). Aachen, Germany, 1999;

[9] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multiobjective genetic algorithm; NSGA-II. IEEE Transactions on Evolutionary Computation, 2002, 6(2): 182-197

[10] Zitzler E, Thiele L. Multiobjective evolutionary algorithms; A comparative case study and the strength pareto approach. IEEE Transaction on Evolutionary Computation, 1999, 3(4): 257-271

[11] Storn R, Price K. Differential evolution; A simple and efficient adaptive scheme for global optimization over continuous spaces. Technical Report TR-95-012, International Computer Science Institute, Berkeley, 1995

[12] Lin Yung-Chien, Hwang Kao-Shing, Wang Feng-Sheng. Hybrid differential evolution with multiplier updating method for nonlinear constrained optimization problems//Proceedings of the CEC'2002. Piscataway, New Jersey, 2002, 1: 872-877

[13] Abbass H, Sarker R, Newton C. PDE: A Pareto-frontier differential evolution approach for multi-objective optimization problems//Proceedings of the Congress on EC 2001 (CEC'2001). IEEE Service Center, Piscataway, New Jersey, 2001, 2: 971-978

[14] Li Bing-Yu, Xiao Yun-Shi, Wu Qi-Di. Hybrid algorithm based on particle swarm optimization for solving constrained optimization problems. *Control and Decision*, 2004, 19(7): 804-807(in Chinese)  
(李炳宇, 萧蕴诗, 吴启迪. 一种基于粒子群算法求解约束优化问题的混合算法. *控制与决策*, 2004, 19(7): 804-807)

[15] Van Veldhuizen D A, lamont G B. Multiobjective evolutionary algorithm research: A history and analysis. Department Elec. Comput. Eng., Graduate School of Eng., Air Force Inst. Technol., Wright-Patterson AFB, OH. Technical Report TR-98-03, 1998

[16] Schott J. Fault tolerant design using single and multicriteria genetic algorithm optimization[Master dissertation]. Depart-

ment of Aeronautics and Astronautics, Massachusetts Institute of Technology, 1995

[17] Koziel S, Michalewicz. Evolutionary algorithms, homomorphous mapping, and constrained parameter optimization. *Evolutionary Computation*, 1996, 7(1): 19-44

[18] Deb K, Pratap A, Meyarivan T. Constrained test problems for multi-objective evolutionary optimization//*Proceedings of the 1st International Conference on Evolutionary Multi-Criterion Optimization*. Zurich, Switzerland, 2001: 284-298

[19] Wolpert D H, Macready W G. No free lunch theorems for search. Santa Fe, NM: Santa Fe Institute. Technical Report: SFI-TR-05-010, 1995



**MENG Hong-Yun**, born in 1975, Ph. D., associate professor. Her research interests include optimization theory and algorithm, natural computation, image processing.

**ZHANG Xiao-Hua**, born in 1974, Ph. D., associate professor. His research interests include natural computation, intelligent information processing, data mining, and digital watermark.

**LIU San-Yang**, born in 1959, Ph. D., professor, Ph. D. supervisor. His research interests include optimization theory and methods.

Background

Constrained optimization, both for nonlinear programming and multi-objective optimization, is a very important problem and has a variety of applications in engineering, management, mathematics and other fields. A common way to constrained optimization problem is to deal with constraints by penalty function, with the disadvantage and difficulty in choosing the penalty factors. In this paper, the authors propose a differential evolution based on double populations for constrained multi-objective optimization problem. By using the definitions and the mechanism in the paper, an

effective evolutionary algorithm for constrained multi-objective optimization problem is given and the time complexity is discussed and compared with the state of the art—NSGA- II and SPEA, which shows the time complexity of the proposed algorithm is as the same magnitude as SPEA and is higher than that of NSGA- II, while simulations illustrate that the proposed algorithm is superior to NSGA- II in the measure of GD and SP. However, it is worthy to note that the time complexity of our algorithm will decrease greatly if a strategy like NSGA- II is taken for updating optimal population gbest.