文章编号:1001-0920 (2012) 10-1441-06

求解约束优化问题的粒子进化变异遗传算法

鲁延京, 陈英武, 杨志伟

(国防科学技术大学 信息系统与管理学院，长沙 410073)

摘 要: 设计一种求解约束优化问题的粒子进化变异遗传算法 (IGA_PSE). 首先, 分析候选解约束条件离差统计信息与约束违反函数之间的关系及其性质, 基于约束条件离差统计信息提出一种改进约束处理方法; 其次, 基于粒子进化策略提出 3 种新变异算子; 然后, 讨论该算法早熟收敛的 3 种情况, 并提出相应的种群多样化维持策略; 最后, 通过数值实验表明所提出的算法能够有效求解约束优化问题.

关键词: 约束优化问题；遗传算法；粒子进化变异算子；早熟收敛

中图分类号: TP18 **文献标志码:** A

Improved GA with particle swarm's evolutionary strategy for solving constrained optimization problems

LU Yan-jing, *CHEN Ying-wu*, *YANG Zhi-wei*

(College of Information Systems and Management，National University of Defense Technology，Changsha 410073，China．Correspondent：LU Yan-jing，E-mail：yanjinglu_nudt@yahoo.com)

Abstract: An improved genetic algorithm(GA) with particle swarm's evolutionary(IGA_PSE) strategy is proposed to solve constrained optimization problems(COP). Firstly, the relation and its characters between the statistics information of the degree of constraint deviation and the constraint violation functions of candidate solutions are analyzed, and an improved constraint handling method is proposed by using statistics information of the degree of constraint condition deviation. Secondly, three novel mutation operators with particle swarm's evolutionary strategy are applied to IGA_PSE. Then, three situations of premature convergence are argued, and the corresponding strategy of diversity maintenance is proposed. Finally, numerical experiments of standard test functions show that the proposed method can solve the constraint optimization problems effectively.

Key words: constrained optimization problems；genetic algorithm；particle swarm's evolutionary mutation operator; premature convergence

1 引 言

基于智能优化方法求解约束优化问题具有易工程化、求解效率高、鲁棒性好等特点, 目前在约束优化问题求解中已得到广泛应用, 如遗传算法 (GA)[1-2]、粒子群算法 (PSO)[3]、差分进化方法 (DE)[4] 和进化规划 (EP)[5]等. 利用智能优化算法求解约束优化问题的两个核心内容是“约束处理技术”和“智能算法设计”, 尤其前者对算法的执行效率有着重要的影响. 文献 [6-7] 对具有代表性的几种约束条件的处理方法进行了综述, 这些方法包括保留可行解方法、罚函数法、区分可行解与不可行解方法以及其他混合方法; 文献 [8] 从约束处理技术和进化算法两个基本方面对约束优化进化算法进行了综述, 指出如何求解约束优化问题是进化计算的一个重要研究领域, 具有十分重要的理论和实际意义. 为了提高智能优化算法在解决工程问题中的效果, 针对约束处理方法, 许多学者提出了一些独特的处理方式[6,9-10].

虽然, 智能优化方法在求解约束优化问题上具备明显优势, 但智能优化方法也存在着先天不足, 如: GA 收敛速度慢, 局部寻优能力差; PSO 算法收敛速度快, 但易陷入局部最优等. 为此, 许多学者提出了各种形式的改进策略来提高智能优化方法求解问题的效率, 如: 在 GA 中添加精英策略[11], 使用了智能适应度函数[12]和知识策略[13]; 文献 [14] 提出了一种基于群体适应度方差的自适应变异算子来改进 PSO 算法,

收稿日期: 2011-06-11；**修回日期:** 2011-09-26.

基金项目: 国家自然科学基金项目(70901074, 70971131, 71031007).

作者简介: 鲁延京(1982−), 男, 博士生, 从事系统管理与综合集成技术、体系结构设计与优化的研究；陈英武(1963−), 男, 教授, 博士生导师, 从事系统规划与管理决策技术、国防采办与项目管理等研究.

使PSO算法的求解过程不易陷入局部最优.

本文提出一种可用于求解约束优化问题的改进遗传算法——粒子进化变异遗传算法(IGA_PSE). 首先, 从约束处理、新的变异算子、算法多样性维持机制3个方面对所提出的算法进行设计; 然后, 给出了将该算法应用到约束优化问题求解中的流程; 最后, 通过对6个标准测试函数的数值实验表明了所提出的粒子进化变异算法可有效求解约束优化问题.

2 问题描述

约束优化问题是在满足等式或不等式约束的条件下最优化某个目标函数. 现实中很多大型复杂工程问题都可以抽象为此类问题, 比如复杂系统设计、卫星调度、生产流程安排和库存控制等. 约束函数优化问题一般转化为如下模型进行求解:

$$\begin{aligned}&\min\ \ f(X).\\&\text{s.t. }\ g_i(X)\leqslant 0,\ i=1,2,\cdots,q;\\&\qquad h_i(X)-\varepsilon\leqslant 0,\ i=q+1,q+2,\cdots,m;\\&\qquad x_j^L\leqslant x_j\leqslant x_j^U,\ j=1,2,\cdots,n.\end{aligned}\tag{1}$$

其中: X 为决策变量, $f(X)$ 为目标函数; $g_i(X)(i=1,2,\cdots,q)$ 为不等式约束, q 为不等式约束个数; $h_i(X)$ $(i=q+1,q+2,\cdots,m)$ 为等式约束, $(m-q)$ 为等式约束个数, ε 为一个很小的正数; x_j^L 和 x_j^U 分别为第 j 个决策变量 x_j 的下限和上限约束; n 表示决策变量的个数, 即优化问题的维度.

3 粒子进化变异遗传算法设计

3.1 约束处理改进

由于简单易行, “罚函数法”是处理约束条件的常用方法之一. 但是, 在实际应用中, 若选择的罚因子太小, 则很难产生可行解; 若罚因子太大, 则会过早陷入局部最优. 只有选择合适的罚因子, 才能使算法获得较好的搜索效果, 因此罚因子的选择面临着很大挑战. 区分可行解与不可行解, 将约束条件转化为候选解违反约束条件的程度函数, 是处理约束的另一种方法. 该方法虽然需要增加不少额外计算过程, 但是避免了由于罚因子选择带来的搜索效果不佳的问题. 针对个体是否为可行解和不可行解两种情况, 文献[15]提出了基于联赛选择机制的种群最小最大选择规则. 该规则简单可行, 但也存在诸如容易陷入局部最优解等缺陷. 为此, 针对该规则许多学者提出了改进策略[9-10, 16].

候选解 X' 违反每一个约束条件的程度函数定义如下:

$$s_i(X')=\begin{cases}\max\{0,g_i(X')\},\ i=1,2,\cdots,q;\\ |h_i(X')|,\ i=q+1,q+2,\cdots,m.\end{cases}\tag{2}$$

$$G(X')=\sum_{i=1}^{m}s_i(X').\tag{3}$$

其中 $G(X')$ 表示候选解 X' 违反约束条件的总程度. 显然, 如果 X' 为可行解, 则必有 $G(X')=0$; 若 X' 为不可行解, 则必有 $G(X')>0$; $G(X')$ 越大, 不可行解违反约束程度越大. 考虑对候选解 X' 下所有约束偏离程度进行统计分析, 定义其均值 $\bar{s}$, 标准差 σ, 变异系数cv分别如下:

$$\bar{s}=\Big(\sum_{i}^{m}s_i(X')\Big)\Big/m=G(X')/m,\tag{4}$$

$$\sigma=\sqrt{\frac{\sum_{i}^{m}(s_i(X')-\bar{s})^2}{m-1}},\tag{5}$$

$$\text{cv}=\sigma/\bar{s}.\tag{6}$$

其中: s_i, $\bar{s}$, σ 和cv具有如下性质:

性质 1 X' 为一不可行解 $\Leftrightarrow G(X')>0\Leftrightarrow\bar{s}>0$.

性质 2 X' 为一可行解 $\Leftrightarrow s_i=0\Leftrightarrow G(X')=0\Leftrightarrow\bar{s}=0\Rightarrow\sigma=0$.

性质 3 $\sigma>0\Leftrightarrow\text{cv}>0\Rightarrow X'$ 为一不可行解.

性质 4 $\bar{s}$ 越大, X' 违反约束程度越大; $\bar{s}$ 越小, X' 违反约束程度越小. 当 $\bar{s}$ 不变时, σ 越大, X' 违反约束程度越大; σ 越小, X' 违反约束程度越小.

若 X' 为不可行解, 则 $\exists k\in[1,m]$, $s_k\neq 0$. 因为 $s_i\geqslant 0$, 所以 $s_k>0$, 此时必有 $G(X')>0$, $\bar{s}>0$, 则性质1的充分性得到证明. 由 $\bar{s}>0$, 必有 $G(X')>0$. 由于 $s_i\geqslant 0$, 根据 $G(X')>0$, 必有 $\exists k\in[1,m]$, $s_k\neq 0$, 即存在一个违反约束条件的约束, 因此 X' 为不可行解. 于是, 性质1的必要性得到证明. 性质2的证明与性质1类似. 需要注意的是, $\sigma=0$ 描述了 $s_i=\bar{s}$, 并不能确定 s_i 是否为0.

由cv的定义可知, 性质3的充分必要条件是成立的. 若 $\sigma>0$, 则 $\exists k\in[1,m]$, 有 $s_k\neq\bar{s}>0$, 由性质1可知, $\sigma>0\Rightarrow\bar{s}>0\Rightarrow X'$ 为一可行解.

由式(2)和(3)可知, $G(X')$ 描述了 X' 违反约束条件的总程度, $\bar{s}$ 描述了 X' 违反约束的平均程度. σ 是对 X' 下约束条件偏离 $\bar{s}$ 的程度度量, σ 越小 s_i 越接近均值 $\bar{s}$. 由于 s_i 的整体趋势是趋近于0, 导致当 $\bar{s}$ 不变时, σ 越大 s_i 越偏离均值 $\bar{s}$, 造成 s_i 整体不均匀, X' 违反约束程度越大; σ 越小 s_i 越接近均值 $\bar{s}$, s_i 整体比较均匀, X' 违反约束程度越小.

下面依据上述性质, 对Jimenez & Verdegay规则进行改进.

1) 如果两个解 X_1, X_2 都是可行解, 则选择使目标函数值 $f(X)$ 最小的个体.

2) 如果一个为可行解 X_1, 另一为不可行解 X_2, 则选择可行解 X_1.

3) 如果 X_1, X_2 都为不可行解:

①$\bar{s}_{X_1} \leqslant \bar{s}_{X_2}$, $\sigma_{X_1} \leqslant \sigma_{X_2}$ 或 $\bar{s}_{X_1} \geqslant \bar{s}_{X_2}$, $\sigma_{X_1} \geqslant \sigma_{X_2}$, 则选择 $\bar{s}$ 和 σ 都较小的个体;

②$\bar{s}_{X_1} \leqslant \bar{s}_{X_2}$, $\sigma_{X_1} \geqslant \sigma_{X_2}$ 或 $\bar{s}_{X_1} \geqslant \bar{s}_{X_2}$, $\sigma_{X_1} \leqslant \sigma_{X_2}$, 则选择 cv 较小的个体.

3.2　3种新的粒子进化变异算子

PSO 算法是 Kennedy 和 Eberhart[17]发明的一种全局优化智能算法, 它源于对鸟类捕食行为的模拟. Shi 和 Eberhart[18]将惯性权重标引入粒子群算法, 形成了标准粒子群算法, 其粒子进化方程为

$$V(t+1) = \omega V(t) + C_1 \mathrm{rand}(p_{\mathrm{best}}(t) - x(t)) + C_2 \mathrm{rand}(g_{\mathrm{best}}(t) - x(t)), \tag{7}$$

$$x(t+1) = x(t) + V(t+1). \tag{8}$$

PSO 算法通过粒子追随自己找到的最好解 (个体极值 p_{best}) 和整个群体的最好解 (全局极值 g_{best}) 完成优化, 有全局寻优能力. PSO 算法在多维空间函数寻优、动态目标寻优等方面具有收敛速度快、解质量高、鲁棒性好等优点. 但 PSO 算法很容易陷入局部最优. 当粒子搜索到的当前最优解为一局部最优位置时, 一旦所有粒子收敛于该位置后, 这些粒子将很难跳出局部最优.

个体进化的一个基本目标是使目标值达到最优. 全局搜索获得的最优值与当前代搜索获得的最优值为个体进化提供了一种搜索方向. 如同粒子追寻最优个体在搜索空间前进一样, 粒子进化变异算子根据个体搜索得到全局最优解与局部最优解进行变异. 设某代 GA 中精英解个体为 X^g, 当前代中除精英个体外的当前局部最优解个体为 X^p, 则候选解 X 粒子进化变异结果为

$$\tilde{X} = w*X + r_1*C_1*(X^p - X) + r_2*C_2*(X^g - X). \tag{9}$$

其中: $\tilde{X}$ 为 X 的变异结果; r_1 和 r_2 为 [0,1] 之间的随机数; $(X^p - X)$ 体现了变异后的解从全局最优解中学习的方向; $(X^g - X)$ 体现了从当前代局部最优解中学习的方向; C_1, C_2 为 $\tilde{X}$ 从全局最优解 X^g 和局部最优解 X^p 中学习的程度, 且 $C_1 > 0$, $C_2 > 0$.

粒子进化变异算子具有全局搜索的性能. 事实上, 已知式 (9) 中, X, X^p, $X^g \in [l, u]$, r_1, $r_2 \in [0, 1]$, 假设 $w = C_1 = C_2 = 1$, 则有

$$\tilde{X} \in [l - (r_1 + r_2)(u - l), u + (r_1 + r_2)(u - l)]. \tag{10}$$

又由 $(r_1+r_2)(u-l) > 0$, 因此可能存在 $\tilde{X} > u$ 或 $\tilde{X} < l$ 的情况. 例如, 取

$$r_1 = r_2 = 1,\ X = 0.2u + 0.8l,$$

$$X^p = 0.8u + 0.2l,\ X^g = 0.7u + 0.3l,$$

则变异后的解为 $\tilde{X} = u + 0.3(u - l) > u$. 因此, 粒子进化变异算子的全局搜索性能也意味着在产生新的变异解时可能会发生解超界的意外情况. 根据这种特性, 本文提出 3 种处理超界的粒子进化变异算子, 以适应解在边界搜索和区域搜索的情况.

3.2.1　边界搜索粒子进化变异算子 (PSEMO_BS)

现实问题中约束函数的解空间往往是不确定的. 有些约束问题的最优解可能出现在边界上. 针对这一类问题, 设计了用于边界搜索的变异算子. 在不影响新个体解的合理性前提下, 对式 (9) 产生的新个体进行如下处理:

$$\tilde{X} = \begin{cases} u, & a \geqslant u; \\ a, & l < a < u; \\ l, & a \leqslant l. \end{cases} \tag{11}$$

其中

$$a = w*X + r_1*C_1*(X^p - X) + r_2*C_2*(X^g - X).$$

该算子将超界的解设置为边界值, 从而大大增加了对边界进行搜索的概率.

3.2.2　区域搜索粒子进化变异算子 (PSEMO_DS)

使变异算子产生的新解始终保持在 [l,u] 范围之内, 区域搜索粒子进化变异算子采用如下分步策略获得 r_1 和 r_2.

Step 1: 设 $\varphi(r_1) = w*X + r_1*C_1(X^p - X)$, 则 $\varphi(r_1)$ 满足

$$\begin{cases} l \leqslant \varphi(r_1) \leqslant u, \\ 0 \leqslant r_1 \leqslant 1. \end{cases} \tag{12}$$

通过求解式 (12) 可得 r_1 的取值范围为

$$\begin{cases} \max\left\{0, \dfrac{l - wX}{C_1(X^p - X)}\right\} \leqslant r_1 \leqslant \\ \min\left\{1, \dfrac{u - wX}{C_1(X^p - X)}\right\}, \text{ 当} X^p - X > 0; \\ r_1 = 0,\ X^p - X = 0; \\ \max\left\{0, \dfrac{u - wX}{C_1(X^p - X)}\right\} \leqslant r_1 \leqslant \\ \min\left\{1, \dfrac{l - wX}{C_1(X^p - X)}\right\},\ X^P - X < 0. \end{cases} \tag{13}$$

Step 2: 将 $\varphi(r_1)$ 带入式 (9), 那么要满足可行域条件 [l,u], r_2 必须满足

$$\begin{cases} l \leqslant \varphi(r_1) + r_2 * C_2 * (X^g - X) \leqslant u, \\ 0 \leqslant r_2 \leqslant 1. \end{cases} \tag{14}$$

通过求解式 (15) 可得 r_2 的取值范围为

$$\begin{cases} \max\left\{0, \dfrac{l-\phi(r_1)}{C_2(X^g-X)}\right\} \leqslant r_2 \leqslant \\ \min\left\{1, \dfrac{u-\phi(r_1)}{C_2(X^g-X)}\right\},\ X^g - X > 0; \\ r_2 = 0,\ \text{当} X^g - X = 0; \\ \max\left\{0, \dfrac{u-\phi(r_1)}{C_2(X^g-X)}\right\} \leqslant r_2 \leqslant \\ \min\left\{1, \dfrac{l-\phi(r_1)}{C_2(X^g-X)}\right\},\ X^g - X < 0. \end{cases} \tag{15}$$

3.2.3 全局搜索粒子进化变异算子 (PSEMO_GS)

上述两类变异算子分别适用于个体在约束条件边界和约束区域内进行搜索. 为了在兼顾边界和区域内搜索情况下获取更加优良的后代, 本文算法设计了一种粒子进化变化优选算子 (PSEMO_GS). 该算子首先对个体进行边界搜索和区域搜索; 然后利用 2.2.2 节中的个体比较函数, 对搜索到的两个后代进行优选, 选取优良的个体作为变异后的子代个体.

3.3 避免早熟收敛的机制设计

学者们指出, 遗传算法早熟收敛发生在算法还未搜索到全局最优个体时, 种群在遗传算子的作用下不再产生优于父代的个体[19]. 经验表明, 保持种群的多样性, 使其维持产生优良后代的能力是有效避免早熟收敛的主要策略之一[20]. 为了达到维持适当多样性的目的, 两项工作是必须的: 一是度量种群的多样化, 二是保持种群多样化的策略. 学者们提出了多种度量种群多样化的方法, 常用的一种方法是利用海明距离[21]度量. 在此基础上, 很多人提出了维持种群多样化的策略, 并提出了一些基于种群多样化的遗传算法, 如面向 GA 的多样性控制方法 (DCGA)[22]和面向 GA 的动态多样性控制方法 (DDCGA)[23]. 文献 [20] 证明了提高交叉和变异算子参数能在一定程度上增加种群个体. 当群体多样性处于临界状态时, 对种群进行再次初始化也能提高种群多样性.

设种群规模为 popsize, 其中可行解与不可行解规模分别为 psize 和 npsize, 因此有

$$\text{psize} + \text{npsize} = \text{popsize}. \tag{16}$$

设第 t 代种群中, 所有可行解集合的目标函数方差 σ^t_{FP} 定义如下:

$$\bar{f}^t_{\text{FP}} = \Big(\sum_i^{\text{psize}} f^t_i(X)\Big)\Big/\text{psize}, \tag{17}$$

$$\sigma^t_{\text{FP}} = \sqrt{\frac{\sum\limits_i^{\text{psize}} (f^t_i(X) - \bar{f}^t_{\text{FP}})^2}{\text{psize}-1}}. \tag{18}$$

其中: $X \in$ {可行解个体}, $(f^t_i(X),\ i \in [0, \text{psize}])$ 为第 t 代中第 i 个可行解的目标函数值. 当 $\sigma^t_{\text{FP}} \to 0$ 时, 第 t 代所有可行解个体目标函数值趋于一致. 此时, 如果获得的最优解不是全局最优解, 则根据早熟收敛的定义, 第 t 代种群的多样化会受到破坏, 局部最优解会发生早熟收敛. 因此, 第 t 代发生早熟收敛的判定依据之一为

$$\begin{cases} f^{t*} > f^*, \\ \sigma^t_{\text{FP}} = 0, \end{cases} \tag{19}$$

其中 f^{t*} 和 f^* 分别为第 t 代获得的局部最优解和全局最优解. 当发生早熟收敛时, 需要增加种群的多样性, 以消除早熟收敛对算法寻优的负面影响.

在算法中, 某一代的不可行解集合虽然对于寻找全局最优解没有直接的意义, 但是其对于保持种群的多样性却有至关重要的意义. 相关文献已经指出, 使群体中保持一定规模的非可行解被认为是一种比较好的策略[1, 20]. 设第 t 代种群中, 所有可行解集合的目标函数方差 σ^t_{NFP} 定义如下:

$$\bar{f}^t_{\text{NFP}} = \Big(\sum_i^{\text{npsize}} f^t_i(X)\Big)\Big/\text{npsize}, \tag{20}$$

$$\sigma^t_{\text{NFP}} = \sqrt{\frac{\sum\limits_i^{\text{npsize}} (f^t_i(X) - \bar{f}^t_{\text{NFP}})^2}{\text{npsize}-1}}. \tag{21}$$

其中: $X \in$ {不可行解个体}, $f^t_i(X)(i \in [0, \text{npsize}])$ 为第 t 代中第 i 个不可行解的目标函数值. 当 $\sigma^t_{\text{NFP}} \to 0$ 时, 第 t 代所有不可行解个体目标函数值趋于一致, 不可行解个体趋于同质. 此时, 虽然对算法第 t 代寻优不会造成直接的影响, 但这种情况会对个体子代的质量产生直接影响, 进而会影响到算法全局寻优的能力. 因此, 不可行解集合同质也是算法需要极力避免的情况之一. 为了保持种群的多样性, 并尽可能避免不可行解个体的同质, 本文算法设置不可行解规模下限值 (MNPS), 并认为第 t 代中, 当不可行解规模低于 MNPS 时, 种群的多样性也将遭到破坏. 显然, MNPS 数值越大, 种群中能够保持的不可行解比例越大. MNPS 值过高, 算法的收敛性会降低, MNPS 值过低, 算法易陷入局部最优.

此外, 若算法运行过程中经过若干代后仍然不能产生更优的全局可行解, 则虽然此时种群多样性能够满足, 但算法也易陷入早熟收敛 (事实上此时算法可能已陷入局部最优中). 针对这种情况, 设 RepeatCount 为某一全局最优解连续出现的次数, 若 RepeatCount 出现次数大于最大遗传代数的 20%, 则需要对种群进行重新初始化, 以寻找更优的种群个体.

综上所述, 第 t 代种群进行多样性维持的判断依据为

$$((f^{t*} > f^*) \text{ and } (\sigma^t_{\text{FP}} = 0)) \text{ or } (\text{npsize} < \text{MNPS}) \text{ or } (\text{RepeatCount} > 0.2 * \text{popsize}). \tag{22}$$

针对上述情况, 当不可行解规模低于MNPS时, 种群多样性维持策略为随机生成[MNPS*popsize]个新不可行解来替代原有解质量较低个体; 当可行解个体集合和不可行解个体集合发生同质时, 分别对其集合中的个体施以0.5概率的交叉操作产生新个体来代替原有个体; 当若干代后不能产生更优的全局最优解时, 将随机产生新的种群, 以替代原有种群.

3.4 算法设计

本文算法的适应度函数采用基于线性排序的适应度函数[24]. 该函数是将个体按目标函数值线性排序后分配适应度值, 个体得到的适应度值只与个体的排序目标值在种群的序位有关, 其计算公式为

$$\mathrm{Fitn}(\mathrm{pos}) = 2-\mathrm{sp}+2\times(\mathrm{sp}-1)\times\frac{\mathrm{pos}}{\mathrm{popsize}-1}. \quad (23)$$

基于线性排序的适应度避免了由于个体目标函数值的过大或过小造成的收敛过快或过慢的问题.

算法杂交操作采用算术杂交算子(ACO)[6], 变异操作采用具有全局搜索功能的PSEMO_GS算子. 综上所述, 应用本文算法求解约束优化问题的伪代码如下:

```
BEGIN
t = 0;
Initialize population P(t);
Evaluate population P(t);
WHILE termination condition not satisfied DO
    BEGIN
        t = t + 1;
        P(t) = SelectP(t − 1);
        Crossover P(t) with ACO;
        Mutate P(t) with PSEMO_GS;
        KeepDiversity;
        Evaluate population P(t);
    END;
END
```

4 数值实验分析

为了评价本文算法的性能, 从文献[16, 24]中选取6个标准测试函数作为数值实验的对象, 算法中的主要参数设置如表1所示.

依据表1的参数设置, 本文对每个问题独立运行30次, 得到问题的最好结果、中值、最差结果、均值和标准差如表2所示. 将计算结果与SMES方法[16]和RY方法[24]进行了比较. 结果表明, 本文算法可满足求解约束优化问题的需要, 设计的避免早熟收敛机制是合理且有效的, 但在最优解求解的稳定性方面有待加强. IGA_PSE虽然在稳定性上不及RY, 但求解的最优值要比SMES和RY好一些, 而且该算法更适合于求解约束数量比较多的函数, 例如测试函数g02和g10. IGA_PSE算法稳定性欠佳的原因是, 粒子进化变异算子使算法收敛更快, 而多样性维持机制多次起作用, 导致算法求解结果的稳定性下降.

表 1　实验主要参数及取值

算 子	取 值	说 明
error	1.0e-6	误差精度
N	200	种群规模
T	1 000	最大进化代数
P_c	0.8	交叉概率
P_m	0.2	变异概率
sp	2	选择操作压差
C_1	2	变异算子向局部最优解学习程度
C_2	10	变异算子向全局最优解学习程度
MNPS	0.25	不可行解规模下限值

表 2　IGA_PSE求解结果与SMES和RY的比较

测试函数/最优值		IGA_PSE	SMES[16]	RY[24]
g01/−15.000	best	−15.000	−15.000	−15.000
	median	−15.000	−15.000	−15.000
	worst	−15.000	−15.000	−15.000
	mean	−15.000	−15.000	−15.000
	st.dev	0.0e+00	0.0e+00	0.0e+00
g02/−0.803 619	best	−0.803 606	−0.801 158	−0.803 515
	median	−0.788 457	−0.787 125	−0.785 800
	worst	−0.665 325	−0.678 203	−0.726 88
	mean	−0.785 694	−0.777 458	−0.781 975
	st.dev	4.3e-02	2.4 e-02	2.0e-02
g04/ −30 665.539	best	−30 665.539	−30 665.539	−30 665.539
	median	−30 665.534	−30 665.536	−30 665.539
	worst	−30 665.392	−30 665.472	−30 665.539
	mean	−30 665.154	−30 665.531	−30 664.710
	st.dev	4.7e+00	1.3 e+00	2.0e-05
g05/5 126.498	best	5 126.501	5 126.498	5 126.497
	median	5 139.302	5 160.198	5 127.372
	worst	5 378.669	5 304.167	5 142.472
	mean	5 160.198	5 174.492	5 128.881
	st.dev	5.3e+02	5.0e+02	3.5e+00
g08/ −0.095 825	best	−0.095 825	−0.095 826	−0.095 825
	median	−0.095 825	−0.095 826	−0.095 825
	worst	−0.095 825	−0.095 826	−0.095 825
	mean	−0.095 825	−0.095 826	−0.095 825
	st.dev	1.4e-13	0.0e+00	2.6e-17
g10/7 049.331	best	7 053.732	7 076.724	7 054.316
	median	7 354.623	7 319.405	7 372.613
	worst	7 965.293	7 816.830	8 835.655
	mean	7 253.645	7 330.398	7 559.192
	st.dev	2.8e+02	1.5e+02	5.3e+02

5 结　论

本文算法充分利用了候选解违反约束条件的离差统计信息对联赛选择机制进行了改进, 提高了种群个体选择质量; 综合粒子进化策略提出了3种新变异算子适应于不同的边界搜索、区域搜索及全局搜索, 具有一定普适性. 对6个标准测试函数数值实验表明,

粒子进化变异算子运行可靠, 多样性维持机制设计合理有效, 可有效求解约束优化问题. 尤其是针对具有多约束的非线性约束优化问题, 本文算法虽然求解稳定性欠佳, 但求解精度总体优于其他算法.

参考文献(References)

[1] 林丹, 李敏强, 寇纪淞. 基于遗传算法求解约束优化问题的一种算法[J]. 软件学报, 2001, 12(4): 628-632.
(Lin D, Li M Q, Kou J S. A GA-based method for solving constrained optimization problem[J]. J of Software, 2001, 12(4): 628-632.)

[2] Wang Yong, Liu Hui, Cai Zixing, et al. An orthogonal design based constrained optimization evolutionary algorithm[J]. Engineering Optimization, 2007, 39(6): 715-736.

[3] 魏静萱, 王宇平. 一种解决约束优化问题的模糊粒子群算法[J]. 电子与信息学报, 2008, 30(5): 1218-1221.
(Wei J X, Wang Y P. Fuzzy particle swarm optimization for constrained optimization problems[J]. J of Electronics & Information Technology, 2008, 30(5): 1218-1221.

[4] Huang V L, Qin A K, Suganthan P N. Self-adaptive differential evolution algorithm for constrained real-parameter optimization[C]. IEEE Congress on Evolutionary Computation. Vancouver, 2006: 17-24.

[5] Kim J H, Myung H. An evolutionary programming techniques for constrained optimization problems[J]. IEEE Trans on Evolutionary Computation, 1997, 11(2): 129-140.

[6] Michalewicz Z, Schoenauer M. Evolutionary algorithm for constrained parameter optimization problems[J]. IEEE Trans on Evolutionary Computation, 1996, 4(1): 1-32.

[7] 胡一波. 求解约束优化问题的几种智能算法[D]. 西安: 西安电子科技大学, 理学院, 2009: 7-12.
(Hu Y B. Intllingent algorithms for constrained optimization problems[D]. Faculty of Science, Xidian University, 2009: 7-12.)

[8] 王勇, 蔡自兴, 周育人, 等. 约束优化进化算法[J]. 软件学报, 2009, 20(1): 11-29.
(Wang Y, Cai Z X, Zhou Y R, et al. Constrained optimization evolutionary algorithms[J]. J of Software, 2009, 20(1): 11-29.)

[9] Kalyanmoy Deb. An efficient constraint handling method for genetic algorithms[J] . Computer Methods in Applied Mechanics and Engineering, 2000, 186(2-4): 311-338.

[10] Coello C A C, Mezura Montes E. Handling constraints in genetic algorithms using dominance-based tournaments[J]. Proc of the Fifth Int Conf on Adaptive Computing Design and Manufacture. 2002, 5(4): 273-284.

[11] Wang Yong, Cai Zixing, Zhou Yuren, et al. An adaptive tradeoff model for constrained evolutionary optimization[J]. IEEE Trans on Evolutionary Computation, 2008, 12(1): 80-92.

[12] De Jong K A. An analysis of the behavior of a class of genetic adaptive systems[D]. Michigan: Department of Computer and Communication Science, University of Michigan, 1975.

[13] Jason Cooper, Chris Hinde. Improving genetic algorithm's efficiency using intelligent fitness function[C]. 16th Int Conf on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems. Loughborough, 2003: 636-643.

[14] 吕振肃, 侯志荣. 自适应变异的粒子群优化算法[J]. 电子学报, 2004, 32(3): 416-420.
(Lv Z S, Hou Z R. Particle swarm optimization with adaptive mutation[J]. Acta Electronica Sinica, 2004, 32(3): 416-420.)

[15] Fernando Jimenez, Jos'e L Verdegay. Evolutionary techniques for constrained optimization problems[C]. The 7th European Congress on Intelligent Techniques and Soft Computing. Berlin: Springer-Verlag, 1999.

[16] Mezura Montes E, Coello C A C. A simple multi-membered evolution strategy to solve constrained optimization problems[J]. IEEE Trans on Evolutionary Computation, 2005, 9(1): 1-17.

[17] Kennedy J, Eberhart R C. Particle swarm optimization[C]. IEEE Int Conf on Neural Networks. Perth: IEEE Press, 1995: 1942-1948.

[18] Shi Y, Eberhart R C. A modified swarm optimizer[C]. IEEE Int Conf of Evolutionary Computation. Alaska: IEEE Press, 1988: 69-73.

[19] Fogel D B. An introduction to simulated evolutionary optimization[J]. IEEE Trans Neural Network, 1994, 5(1): 3-14.

[20] Elena Simona Nicoar. Mechanisms to avoid the premature convergence of genetic algorithms[J]. Universitatea Petrol-Gaze din Ploieti, 2009, 61(1): 87-96.

[21] LIU Xinping, LIU Ying. Adaptive genetic algorithm based on population diversity[C]. 2009 Int Forum on Information Technology and Applications. Chengdu, 2009: 510-512.

[22] Hisashi Shimodaira. DCGA: A diversity control oriented genetic algorithm[C]. Genetic Algorithms in Engineering Systems: Innovations and Applications. Newport Beach, 1997: 444-449.

[23] Pei-Chann Chang, Wei-Hsiu Huang, Ching-Jung Ting. Dynamic diversity control in genetic algorithm for mining unsearched solution space in TSP problems[J]. Expert Systems with Appllication, 2010, 37(3): 1863-1878.

[24] Runarsson, Yao X. Stochastic ranking for constrained evolutionary optimization[J]. IEEE Trans on Evolutionary Computation, 2000, 4(3): 284-294.