文章编号: 1001-0920(2019)12-2611-08 DOI: 10.13195/j.kzyjc.2018.0325

求解约束优化问题的改进自适应μ约束处理技术

徐玉琴, 姚 然†, 李 鹏

(华北电力大学 电气与电子工程学院，河北 保定 071003)

摘 要: 针对当前的约束处理技术存在易陷入局部最优解、难以满足等式约束和多控制参数的问题,在μ约束处理技术的基础上,以梯度下降法和多目标拥挤距离为理论依据,设计反映种群约束违反度分布信息的ω参数,它可以自适应地调节约束违反度阈值μ的松弛进而有效地解决约束问题. 此外,改进了μ阈值比较准则以提高种群的多样性. 经对CEC2017的标准约束优化问题(Constraint optimization problems, COP)进行求解,并与其他先进算法相比较,结果表明,改进的μ约束处理技术能够高效地处理含等式约束的COP.

关键词: μ约束处理技术；自适应差分进化；约束优化问题；罚函数

中图分类号: TP301.6 **文献标志码:** A

Improved adaptive μ-constraint handling technique for solving constrained optimization problems

XU Yu-qin, YAO Ran†, LI Peng

(School of Electrical and Electronic Engineering，North China Electric Power University，Baoding 071003，China)

Abstract: To solve the problems existing in the current constraint handling techniques, where it's easy to fall into the local optimal solutions, difficult to satisfy the equality constraints and there are also multiple control parameters, the parameter ω is designed, based on the gradient descent method and multi-objective crowding-distance theory. In this paper, ω is a reflection of population constraint violation degree distribution information, which can adaptively adjust the relaxation of the constraint violation threshold μ to solve the constrained problem accurately. In addition, the μ threshold comparison criteria have been improved to increase population diversity. After the solutions of the standard constrained optimization problems (COP) of CEC2017, we compare the results with other advanced algorithms. And the comparison shows that the improved μ-constraint handling techniques can deal with complex equation constraints efficiently.

Keywords: μ-constraint handling technique；adaptive differential evolution；constrained optimization problem；penalty function

0 引 言

含约束的优化算法在医疗、能源、金融经济、交通运输等各个领域都有广泛的应用. 经典的基于梯度的非线性规划(Nonlinear programming, NLP)只能解得目标函数和约束都可微的局部最优解[1]. 然而,针对非凸、非光滑、多模态、多目标等复杂的大规模约束优化问题[2], NLP等确定性算法则显得束手无策. 对此,近20多年来,诸如遗传算法(Genetic algorithm, GA)、粒子群算法(Particle swarm optimization, PSO)、差分进化算法(Differential evolution algorithm, DE)等基于概率的全局优化群搜索算法已蔚然成风,日臻成熟. 其中DE在历届国际进化计算会议(Congress on evolutionary computation, CEC)中的表现最令人瞩目[3],其成果也是其他单一算法(如PSO、GA)难以相比的. 对于DE算法的研究,目前,人们把焦点放在高效的搜索机制上[4-5],而关于约束处理技术的研究并不是非常成熟,尤其是如何有效地处理含等式约束的约束优化问题(Constraint optimization problems, COP)至今仍然是一大挑战[6].

Mallipeddi等[7]在CEC2010上将目前进化算法的约束处理技术概括为三类: 罚函数(Penalty function method, PFM)、可行性法则(Superiority of feasible solutions, SF)和ε约束处理技术(ε-constraint, EC).

当前广泛应用的PFM是l_1精确罚函数[8]. 该方法通过惩罚系数将目标函数和约束条件相加以构造适应度函数,目的是将COP转化为较为简单的无

收稿日期: 2018-03-19；**修回日期:** 2018-09-11.

基金项目: 国家自然科学基金项目(51577068)；中央高校基本科研业务费专项资金项目(2018ZD01).

责任编委: 张化光.

†通讯作者. E-mail: imyaoran@163.com.

约束优化问题. 惩罚系数承担着调节目标函数与约束条件之间平衡的作用,最合适的惩罚系数需要大量的微调来整定. 繁琐的控制参数调试严重制约了PFM的应用范围,因此,近些年来研究人员提出了大量无控制参数整定的自适应罚函数(Self-adaptive penalty, SP). SP能够从种群进化过程中获取信息以反馈调节惩罚系数[1],例如根据当前种群可行比例[9]和违反约束度信息[10]自动调节惩罚系数. 尽管如此,PFM仍然存在着局限性. 当最优解在可行域边界附近或可行域不连通时,PFM很难适应[11].

针对PFM的缺陷,Deb[12]提出了SF,为自适应约束处理技术提供了新思路. 然而,该法则过于苛刻地区分可行解与不可行解,导致其过快地收敛于局部最优解而没有充分利用不可行解的有利信息. 因此,单一的SF鲜有使用[13],通常将SF与PFM配合使用[14].

为了充分利用SF的自适应潜力,降低SF苛刻的区分机制带来的不利影响,Takahama等[15]提出了EC. 该方法的本质是设置一个约束违反度阈值ε来区分可行解和不可行解,通过ε阈值比较置换SF的约束违反度比较来评价个体的优劣,并调节ε的松弛以有效地利用不可行解的信息获得高质量解. Li等[16]提出的有偏支配b也具有相似作用. 然而,EC并没有利用当前种群约束违反度的分布信息,而是固定初始值ε_0,并根据迭代次数以及其他人工设定的控制参数来修正ε值. Zhang等[6]在EC的基础上提出μ约束处理技术(μ-constraint, MC),MC利用当前种群中相对可行解的比例λ来修正μ值,不需要额外反复调优的控制参数. 该方法的本质是利用某些约束违反度分布特征来延缓μ值的收敛,使目标函数能从不可行区域获得更多有益信息. 一定程度上MC突破了EC的限制,但是,MC对于种群约束违反度分布信息的利用浅尝辄止,经大规模验证该约束处理方法在解决高维搜索空间的COP时存在严重不足.

针对MC不能有效处理高维搜索空间的问题,在MC的基础上,根据约束违反度的分布特点,结合梯度下降法和多目标拥挤距离理论,本文提出一种改进自适应μ约束处理技术(Improved μ-constraint, proMC). proMC可以有效解决约束处理技术的收敛性与种群多样性之间的冲突,进而在无控制参数调节的情况下求解复杂的具有高维搜索空间的COP.

1 约束优化问题的定义

1.1 通用非线性规划模型

实际问题中,大量的优化问题都有一定的约束条件. 通常求解最小值的约束优化问题可描述如下:

$$\begin{cases}\min f(X).\\ \text{s.t. } h_i(X)=0,\ i=1,2,\cdots,m;\\ \quad g_j(X)\leqslant 0,\ j=m+1,m+2,\cdots,p.\end{cases}\tag{1}$$

其中: $f(X)$是目标函数; $h_i(X)$是等式约束; $g_j(X)$是不等式约束; $X=\{x_1,x_2,\cdots,x_n\}\in \boldsymbol{R}^n, X_{\min}\leqslant X\leqslant X_{\max}$.

1.2 约束违反度

在含有等式约束的COP中,由于难以精确地满足等式约束,通常将等式约束转化为满足一定精度的不等式约束,如下式所示:

$$H_i(X)=\begin{cases}|h_i(X)|,\ |h_i(X)|-\delta>0;\\ 0,\ \text{otherwise.}\end{cases}\tag{2}$$

$$G_j(X)=\begin{cases}g_j(X),\ g_j(X)>0;\\ 0,\ \text{otherwise.}\end{cases}\tag{3}$$

其中: δ是等式约束的容忍参数(一般取值$\delta=10^{-4}$).

约束违反度是反映违反约束大小的度量,即

$$\phi=\Big(\sum_{i=1}^{m}H_i(X)+\sum_{j=m+1}^{p}G_j(X)\Big)^{\text{cp}}.\tag{4}$$

其中: cp是分布指数参数,本文cp = 2; ϕ是个体的约束违反度; 下文的$\phi_{\min}$、ϕ_{med}、$\phi_{\max}$分别是当前种群约束违反度的最小值、中值、最大值.

2 改进自适应μ约束处理技术

近年来,Zhang等[6]提出的μ约束处理技术具有无控制参数调节、满足等式约束、快速求得全局最优解等优点,已被广泛采用. 然而,由于高维搜索空间的约束违反度分布更加多样性,造成在高维搜索空间中,约束违反度阈值μ仅通过种群的可行个体比例λ参数将难以自适应调节. 因此,需要对约束违反度分布描述更加精细的参数,从而使μ更好地适应种群的进化.

2.1 改进自适应μ约束处理技术的原理

2.1.1 改进自适应μ阈值比较准则

相对于文献[6]的μ阈值比较准则,改进的μ阈值比较准则如下所示.

准则1 当一个个体为相对可行解且约束违反度大于零,另外一个个体为相对不可行解时,相对可行解优于相对不可行解.

准则2 当两个个体均为相对可行解时,目标函数值小的个体占优.

准则3 当两个个体均为相对不可行解时,个体约束违反度小的占优.

μ阈值比较是对多目标集合$(f(X),\phi(X))$的次

序关系的一种定义,这种定义使$\phi(X)$的优先级高于$f(X)$,因为一个可行解总是比一个最优解更重要.

定义 1　$f_1(f_2),\phi_1(\phi_2)$分别为在点$X_1(X_2)$处的目标函数值和约束违反度. 对于任意$\mu \geqslant 0$, $(f_1(X),\phi_1(X))$与$(f_2(X),\phi_2(X))$之间的μ阈值比较$<_\mu$定义如下:

$$(f_1,\phi_1)<_\mu(f_2,\phi_2)\Leftrightarrow\begin{cases}\phi_1\leqslant\mu<\phi_2\bigcap\mu>0;\\ f_1<f_2,\ \text{if}\ \phi_1,\phi_2\leqslant\mu;\\ \mu<\phi_1\leqslant\phi_2.\end{cases}\tag{5}$$

2.1.2　经典的μ约束处理技术

经典的μ约束处理技术通过种群的可行个体比例λ参数自适应调整约束违反度阈值μ,然后根据μ阈值比较准则决定种群的进化方向,即

$$\text{fp}_i=\begin{cases}0,\ \phi_i>\mu_G;\\ 1,\ \text{otherwise}.\end{cases}\tag{6}$$

其中: fp_i表示第i个个体是否为相对可行解,μ_G是第G代种群的约束违反度阈值. 种群的可行个体的比例为

$$\lambda=\frac{1}{N}\sum_{i=1}^{N}\text{fp}_i,\tag{7}$$

其中N是种群规模.

鉴于式(4),并考虑到文献[6]的ϕ、μ_G为$\text{cp}=1$时的表达式,广义的修正公式为

$$\mu_{G+1}^{\text{cp}}=\mu_G^{\text{cp}}\left(1-\frac{\lambda}{N}\right).\tag{8}$$

2.1.3　改进自适应μ约束处理技术修正公式

proMC通过对种群描述更为精细的自适应ω参数,可以更有效地自适应控制μ的收敛速度,进而对两个个体通过μ阈值比较准则来决定种群的进化方向.

由一阶泰勒展开式可得

$$f(X_{k+1})=f(X_k)+f'(X_k)(X_{k+1}-X_k).\tag{9}$$

其中: $f(X_k)$是函数$f(X)$在决策变量为X_k时的值,$f'(X_k)$是函数$f(X)$在X_k处的导数.

由式(9)知,在搜索步长非常小的情况下,为了保证$f(X_{k+1})-f(X_k)<0$成立,需要满足

$$X_{k+1}=X_k-\alpha f'(X_k)=X_k(1-\alpha X_k^{-1}f'(X_k)).\tag{10}$$

其中: $\alpha\in(0,1)$是搜索步长,X_k^{-1}是X_k的广义逆.

式(10)即为梯度下降法的修正公式. 为了满足无控制参数自适应调节,proMC算法需要引入对种群约束违反度动态分布描述更为精确的ω参数,令$\omega=X_k^{-1}f'(X_k)$,$\alpha=1/N$. 根据式(8)和(10),当$\text{cp}=2$时, proMC的修正公式为

$$\mu_{G+1}^2=\mu_G^2\left(1-\frac{1}{N}\cdot\omega\right)\ \text{或}\ \mu_{G+1}=\mu_G\sqrt{1-\frac{1}{N}\cdot\omega}.\tag{11}$$

其中: ω是种群约束违反度分布参数; u_G是第G代种群的约束违反度阈值,初始约束违反度阈值μ_0取值为初始种群约束违反度的中值.

由于$0<\alpha<1$,为了保证$\mu_{G+1}<\mu_G$,需要满足

$$0<\frac{1}{N}<1,\ 0<\omega<1.\tag{12}$$

已知$N\gg1$,可知满足条件$0<\dfrac{1}{N}<1$.

进化过程中μ_{G+1}的取值根据种群的分布可以用图1(a)、图1(b)、图1(c)三种情况进行概括. 第1种情况,当$\mu_G>\phi_{\max}$时,如图1(b)所示,此时μ_G的收敛速度滞后于种群的进化速度,则令$\mu_{G+1}=\phi_{\max}$. 第2种情况,当$\mu_G<\phi_{\min}$时,如图1(c)所示,此时μ_G的收敛速度超前于种群的进化速度,则令$\mu_{G+1}=\phi_{\min}$. 第3种情况,当$\phi_{\min}\leqslant\mu_G\leqslant\phi_{\max}$时,如图1(a)所示,参照式(11),需要模拟伪梯度下降方向. 根据多目标区间拥挤距离[16-17]以及大量实验表明,μ_G到ϕ_{med}和$\phi_{\min}$距离的均值点P的方向为最快下降方向,这样,既能加快收敛速度也能保证种群多样性.

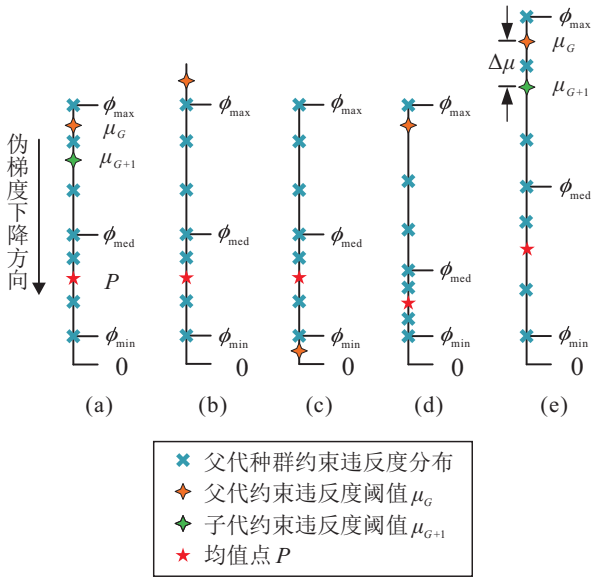


图 1　ω参数原理示意

定义 2　μ_G到ϕ_{med}以及$\phi_{\min}$距离之和的均值与$\phi_{\max}-\phi_{\min}$的比值为伪梯度下降方向γ,定义如下:

$$\gamma=\frac{|\mu_G-\phi_{\text{med}}|+|\mu_G-\phi_{\min}|}{2\cdot(\phi_{\max}-\phi_{\min})}.\tag{13}$$

根据实际,相较于图1(a),图1(d)的种群约束违反度在$\phi_{\min}$到ϕ_{med}之间的分布密度更大,于是图1(a)的种群约束违反度分布状态在时间上超前于图1(d). 图1(a)应以更慢的速度达到图1(d)的状态,避免

状态跳跃而忽略大量有利信息. 再分析式(13),根据图1(a)和图1(d)中μ_G、$\phi_{\min}$、ϕ_{med}、$\phi_{\max}$之间的关系,式(13)符合实际情况.

定义3 种群约束违反度的最大值与最小值之比为种群约束违反度分散度,即

$$\theta = \frac{\phi_{\max}}{\phi_{\min}}, \tag{14}$$

其中θ是种群约束违反度分散度,$\theta > 1$.

相较于图1(a),图1(e)的θ更大,于是,图1(e)的种群约束违反度分布状态在时间上超前于图1(a). 因此,图1(e)需要更加缓慢的下降速度以使种群约束违反度的分布在$\phi_{\min}$到ϕ_{med}之间更加集中.

根据式(13)和(14),由γ、θ特性知,$\omega \propto \gamma, \omega \propto 1/\theta$. 因此,$\omega$的表达式为

$$\omega = \gamma/\theta. \tag{15}$$

已知$0 < \gamma < 1, \theta > 1$,可知$0 < \omega < 1$,满足式(12)的要求.

由式(13)~(15)可知,当$\phi_{\mathrm{med}} \leqslant \mu_G \leqslant \phi_{\max}$时,$\mu_G$越接近$\phi_{\mathrm{med}}$,$\omega$越小,$\Delta\mu$越小,$\mu_G$的收敛速度越慢;$\mu_G$越接近$\phi_{\max}$,$\omega$越大,$\Delta\mu$越大,$\mu_G$的收敛速度越快. 当$\phi_{\min} \leqslant \mu_G < \phi_{\mathrm{med}}$时,为了平衡种群的多样性和收敛速度,令$\Delta\mu$处于不敏感范围,$\Delta\mu$的大小与$\mu_G$的大小无关. 由以上分析可知,$\omega$可以根据当前约束违反度的分布状态与$\mu_G$的关系作出自适应调整以控制种群的进化方向,如图1(e)所示.

2.2 改进自适应μ约束处理技术数值分析

以一个简单的含有两个机组的电力系统的安全约束经济调度优化问题为例,对proMC进行说明. 其数学模型如下式所示:

$$\begin{aligned}
\min\ & 0.1 \times (0.004x_1^2 + 12x_1 + 913) + \\
& |190\sin(0.085 \times (100 - x_1))| + \\
& 0.1 \times (0.003x_2^2 + 8x_2 + 649) + \\
& |150\sin(0.085 \times (100 - x_2))|; \\
\text{s.t. } & 0.000\,5x_1^2 + 0.8x_1 + 0.000\,4x_1x_2 + \\
& 0.6x_2 + 0.000\,6x_2^2 - 200 = 0, \\
& 200 \leqslant 0.1(x_1 - 107)^2 + 2x_2 \leqslant 335, \\
& 83 \leqslant 0.3x_1 + 0.4x_2 \leqslant 111.5, \\
& 90 \leqslant x_1 \leqslant 145, 80 \leqslant x_2 \leqslant 190.
\end{aligned} \tag{16}$$

该COP的初始代、第10代、第60代和第300代种群分布情况如图2所示. 图2中:实线表示约束,其中等式约束和不等式约束已在图中标记;虚线表示约束的松弛,松弛范围即为种群允许分布的范围;符号“×”表示种群的分布,共有20个个体;符号“⋆”表示最优点位置,全局最优点和局部最优点也在图中标记;灰色阴影区域是不等式可行域(图中省略两个无效不等式);背景等高线是目标函数$f(x)$在不同点的值组成的线. 由图2(a)~图2(d)四幅图可以明显地观察到,随着约束松弛逐渐减小,种群慢慢向等式约束上的最优点靠近. 约束松弛的范围严格限制了种群的活动范围,从而保证了算法的收敛性.

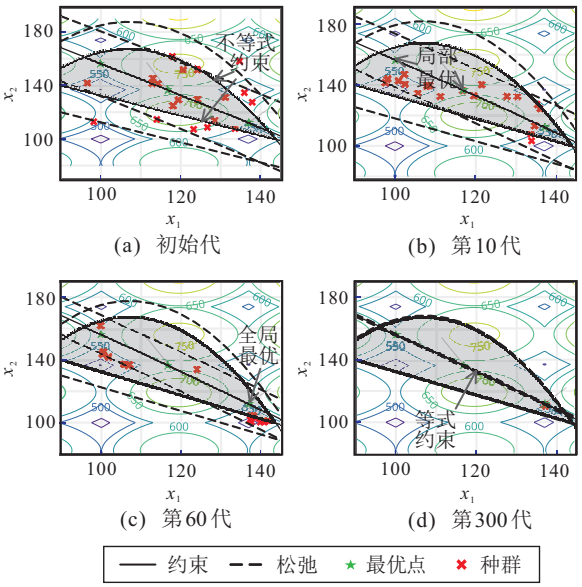


图2 每一代种群的分布以及约束逐渐松弛状况

式(11)中μ_G的收敛特性如图3(a)所示. 一般在迭代开始时,由于种群的多样性,个体间会快速交叉选择,μ_G迅速收敛. 迭代后期,由于靠近最优点,种群多样性降低,μ_G的收敛速度放缓. 此外,μ_G在种群约束违反度分布中的位置大多在ϕ_{med}与$\phi_{\max}$之间,符合2.1.3小节中算法设计的目的. 从多目标优化的视角来分析,proMC的特性如图3(b)所示,Pareto前沿的分布较为陡峭,说明μ_G约束处理方法迭代的后期收敛速度会异常缓慢,更多的迭代是从不等式可行域内目标函数的最低点向等式约束逐渐靠拢.

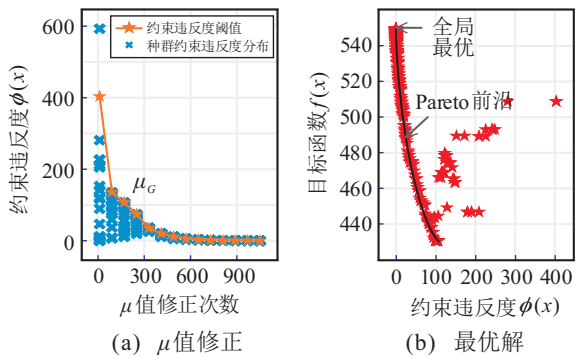


图3 μ值的修正情况以及最优解示意

2.3 μ约束处理技术算法描述

DE主要包括4步:1) 数据初始化;2) 选取目标向量进行变异;3)对变异的向量执行交叉操作;4)用贪

婪选择策略选择交叉后的向量.

考虑到DE的优化性能过于依赖控制参数,本文利用文献[4]提出的具有更健壮鲁棒性的jDEpro改进策略,并在jDEpro的基础上进行约束处理的修改. 此外,为了降低内存、加快计算速度,本文还采用更加贪婪的单数组循环[18]的编程思想.

为了充分测试proMC广泛适应的特性,本文还测试了proMC与SHADE[5]相结合的算法. 在SHADE原文代码的基础上,将16～24行代码用如下算法1替换.

算法1　改进 μ 约束处理算法.

1 // Improved μ constraint handling technique
2 // $u_{i,G}$: 实验向量; A: 外部档案(淘汰向量的集合)
3 // $\mathrm{CR}_{i,G}$: 第 G 代交叉因子; $F_{i,G}$: 第 G 代缩放因子
4 // S_{CR}: 成功的 $\mathrm{CR}_{i,G}$ 的集合; S_F: 成功的 $F_{i,G}$ 的集合
5 if $\phi(u_{i,G}) > \mu_G$ then
6　　if $\phi(u_{i,G}) \leqslant \phi(x_{i,G})$ then
7　　$x_{i,G+1} \leftarrow u_{i,G}$;
8　　$\phi_i \leftarrow \phi(u_{i,G})$;
9　　$A \leftarrow x_{i,G}, S_{\mathrm{CR}} \leftarrow \mathrm{CR}_{i,G}, S_F \leftarrow F_{i,G}$;
10　　end
11 else if $\phi(x_{i,G}) > \mu_G \bigcap \mu_G > 0$ then
12　　$x_{i,G+1} \leftarrow u_{i,G}, f_i \leftarrow f(u_{i,G})$;
13　　$\phi_i \leftarrow \phi(u_{i,G})$;
14　　$A \leftarrow x_{i,G}, S_{\mathrm{CR}} \leftarrow \mathrm{CR}_{i,G}, S_F \leftarrow F_{i,G}$;
15 else
16　　if $f(u_{i,G}) < f(x_{i,G})$ then
17　　$x_{i,G+1} \leftarrow u_{i,G}, f_i \leftarrow f(u_{i,G})$;
18　　$\phi_i \leftarrow \phi(u_{i,G})$;
19　　$A \leftarrow x_{i,G}, S_{\mathrm{CR}} \leftarrow \mathrm{CR}_{i,G}, S_F \leftarrow F_{i,G}$;
20　　if $\mu_G < \delta^2$ || $\phi_{\mathrm{med}} < \delta^2$ then
21　　$\mu_{G+1} \leftarrow 0$;
22　　else if $\mu_G > \phi_{\max}$ then
23　　$\mu_{G+1} \leftarrow \phi_{\max}$;
24　　else if $\mu_G < \phi_{\min}$ then
25　　$\mu_{G+1} \leftarrow \phi_{\min}$;
26　　else
27　　$\mu_{G+1} \leftarrow \mu_G \times \mathrm{sqrt}(1-\omega/N)$;
28　　end
29　　end
30 end

算法1中第5、6、11、16行即为改进 μ 阈值比较准则的框架. 相比于文献[6]的 μ 阈值比较准则,本文将文献[6]的比较准则1替换为2.1.1小节的准则1,如算法1的第11行. 这样,可以避免在约束违反度已经满足的情况下进行不必要的进化而导致种群多样性降低. 此外,将文献[6]比较准则3替换为2.1.1小节的准则3,如算法1的第6行. 这样,当完全可行解不存在时,对目标函数值更小的种群能够起到一个累积的作用,同时对搜索到更小的约束违反度也非常有利.

另外, μ 约束处理技术[6]和罚函数的更新机制[19]指出,只有在准则3发生时才更新 μ,此时 μ 的修正方法如2.1.3,在算法1中主要表现在第16～29行.

需要说明的是,本文使用的停止迭代的标准是Conceicao等[18]给出的方法. 虽然在大量的实验中,该方法会出现误判而导致程序提前中止的情况,但是这种情况并不多见.

3　实验结果与分析

3.1　实验数据与参数设置

通过20个CEC2017[11]标准约束测试COP对proMC进行测试,并与其他方法进行对比,以验证proMC的有效性.

按照CEC2017会议关于参数的设置要求进行参数设置.

jDEpro参数设置: 种群规模 N 设置为2D～20D (D表示决策变量的维度); 最大迭代次数iteration设置为20 000D; 收敛容差tol设置为 10^{-15}; 交叉概率CR设置为0.6; 缩放因子 F 设置为0.1; 抖动系数jitter设置为0.01; either-or差分策略pF设置为0.1.

SHADE参数设置: 种群规模 N 设置为2D～20D; 最大迭代次数iteration设置为20 000D; 收敛容差tol设置为 10^{-15}; 记忆容量 H 设置为6; 最优个数 p_{best} 设置为0.1D,即决策变量维度的10 %.

值得提及的是,关于参数的设置除了种群规模 N 以外,其他参数都是固定不变的. 因此,proMC-jDEpro或者proMC-SHADE算法只有一个控制参数,即种群规模 N. 大量实验表明,种群规模 N 一般设置为3D～10D即可,少数情况需要更大的种群,例如CEC2017问题中的C06和C15.

3.2　结果分析讨论

目前,较为先进的自适应DE搜索机制基本都是基于SHADE改进的[13,20],本文将proMC-jDEpro与proMC-SHADE的优化结果进行比较发现,二者的优化结果并没有显著差别. 鉴于proMC-jDEpro在计算速度上更具有优势,本文仅列出proMC-jDEpro对

CEC2017中COP进行求解的结果. 表1是CEC2017约束优化问题中C01～C20共20个优化问题、决策变量的维度为100维的优化结果. 表1中Best、Median和Mean分别表示25次实验中目标函数值排序后的最优解、中值和均值; SR表示25次实验中满足精度要求的成功率; Viol表示25次实验的平均约束违反度值. 由表1数据可以看出,对于复杂的等式约束,如C03、0C6、C07、C11、C16、C18, proMC都可以100%地求得最优解. 此外,由Best、Median和Mean的大小可知,对于多数优化问题,由proMC-jDEpro求得的解具有相当的稳定性,某种程度上,本文提出的算法可以降低结果的随机性.

表1 决策变量为100维度的测试结果(25次,最大迭代次数为20 000D)

function	C01	C02	C03	C04	C05	C06	C07	C08	C09	C10
Best	0	0	74.824 3	13.572 8	0	0	−4 822.549 3	0	0	0
Median	0	0	135.023 5	13.572 8	0	0	−4 799.352 7	0	0	0
Mean	0	0	141.214 6	13.572 8	0	0	−4 756.672 5	0	0	0
SR/%	100	100	100	100	100	100	100	100	100	100
Viol	0	0	0	0	0	0	0	0	0	0
function	C11	C12	C13	C14	C15	C16	C17	C18	C19	C20
Best	−89.440 9	3.980 7	33.204 0	0.396	5.497 7	0	1.028 7	36.381 0	0	3.175 8
Median	−89.432 9	3.981 6	33.204 1	0.396	8.639 3	0	1.044 2	36.384 1	0	3.526 9
Mean	−89.426 1	3.983 2	33.363 4	0.396	7.759 7	0	1.045 4	36.385 2	0	3.532 6
SR/%	100	100	100	100	100	100	0	100	0	100
Viol	0	0	0	0	0	0	50.5	0	72 969.513 8	0

表2为proMC-jDE与其他主流先进算法的对比. 为了突出优化算法间求解结果的主要差异,表2中仅给出部分问题的比较. 从表2中不难发现,对于含有复杂等式约束的COP,本文提出的proMC-jDE具有突出的优势.

表2 改进的μ约束处理与其他算法比较(100维)

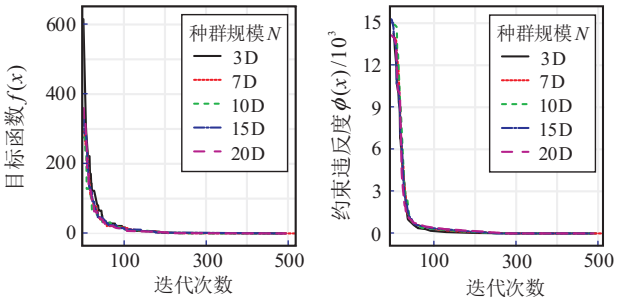
函数	性能指标	proMC-jDE	LSHADE44-IDE	LSHADE44	UDE
C03	Best	74.824 3	9.393e+07	1.34e+06	12 971
	Median	135.023 5	2.265e+08	2.47e+06	13 863
	Viol	0	0	0	0.002 1
C06	Best	0	15 439.912 4	17 164.3	4 603
	Median	0	15 594.732 9	15 803.2	2 782.1
	Viol	0	0.011 9	0.009 8	0.611 65
C07	Best	−4 822.55	−530.117 8	−683.178	−340.88
	Median	−4 799.352 7	−324.987 2	−301.161	−27.751
	Viol	0	0	0	2 289.2
C11	Best	−89.440 9	−4.485 7	−5.724 67	−161.19
	Median	−89.432 9	−2.641 5	−3.498 77	98.865
	Viol	0	0	0.000 0	40.068
C16	Best	0	452.389 3	477.522	50.265
	Median	0	527.787 5	534.071	81.681
	Viol	0	0	0	0.000 04
C18	Best	36.381 0	1 191.207 7	4 197.8	57 777
	Median	36.384 1	1 825.971 1	2 284.07	89 528
	Viol	0	2.738e+07	3.34e+06	9.422e+09

在其他3个算法中,LSHADE44&IDE[20]算法搜索的过程分为两个阶段. 第1阶段,以种群约束违反度均值最小化为目标进行优化求解. 在第2阶段,若第1阶段搜索到较多的可行解,则视这些可行解为第2阶段的初始值种群,以目标函数最小化为优化目标; 否则以约束违反度最小的解为初始种群进行优化求解. LSHADE44[13]算法类似于可行性法则,主要有两条法则: 1) 当两个个体比较时,选择违反约束程度小的个体,并更新目标函数值; 2) 如果两个个体的约束违反度都为零,则选择目标函数值小的个体. UDE[21]采用ε约束处理技术. 从表2中得知,相比于其他算法,proMC-jDE明显更优异,尤其是在处理复杂的等式约束时.

3.3 灵敏度分析

尽管proMC是一种无控制参数调节的约束处理技术, 但是作为约束优化算法基础的jDEpro或SHADE却拥有控制参数. jDEpro或SHADE的控制参数种群规模N会对目标函数值$f(x)$和约束违反度$\phi(x)$产生一定影响. 以可行域不连通的含多等式约束优化问题C06为例, $f(x)$和$\phi(x)$对种群规模N的灵敏度分析如图4所示. 从图4中可以看出,随着种群规模N的增大, $f(x)$和$\phi(x)$的收敛曲线变得更加光滑. 相对而言,种群规模N越大, $f(x)$的收敛速度越快, $\phi(x)$的收敛速度越慢. 因此,在满足种群多样性的条件下,种群规模N应尽量小. 此外,对照几条曲线可以发现,

种群规模N对$f(x)$和$\phi(x)$收敛性的影响并不是非常灵敏. 所以,种群规模N一般取值3D～10D.

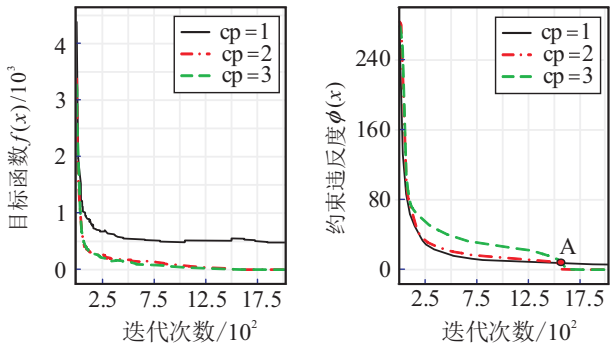


(a) 目标函数　(b) 约束违反度

图 4　proMC对种群规模N的灵敏度分析

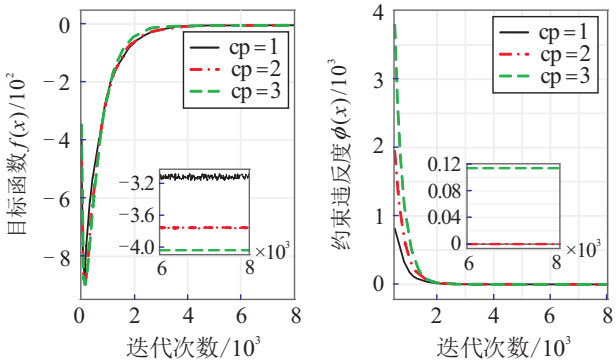
3.4　分布指数参数取值分析

分布指数主要通过控制约束的松弛来影响$f(x)$和$\phi(x)$的收敛特性. 由实际情况知,ω/N为相当小的值. 结合式(8)和(11)知,cp的值越大,种群约束违反度分布越分散,μ_G的收敛速度越慢,$f(x)$就可以从当前的种群中获得充分的有效信息. 因此cp越大,$f(x)$的收敛速度越快,$\phi(x)$的收敛速度越慢,如图5和图6所示,图6中嵌套小图为大图的尾部特性的放大.



(a) 目标函数　(b) 约束违反度

图 5　不同cp取值的C06收敛特性曲线



(a) 目标函数　(b) 约束违反度

图 6　不同cp取值的C11收敛特性曲线

对于可行域不连通的COP,如C06,其收敛特性如图5所示. 图5中:当cp = 2, 3时,对应的曲线$\phi(x)$收敛至0并获得最优解;而cp = 1时,对应的曲线$\phi(x)$并没有收敛至0. 在原始约束违反度分布相同的条件下,迭代初期cp越小,ω越大,μ_G收敛速度越快. 但是,此时极易发生状态跳跃,迭代初期的状态跳跃会失去大量有利信息,引起种群的进化方向发生偏移. 当这种偏移越来越严重时,种群的多样性严重下降,种群约束违反度间的差异也越来越小,将很难搜索到更优异的解,从而导致大量迭代循环中μ_G的值并不进行修正,而这将会进一步降低种群的多样性,形成雪崩式循环. 如图5所示,在A点前,cp = 1对应的曲线$\phi(x)$的收敛速度快于cp = 2. 但是,此种以牺牲种群多样性为代价的快速收敛,在A点之后其缺点慢慢浮现,最终cp = 1对应的曲线$\phi(x)$并没有满足约束违反度的要求.

然而,对于可行域连通且拥有唯一可行解的COP,如C11,cp越大,进化过程中发生遗传漂变的概率越高. 如图6所示:当cp = 1, 2时,$\phi(x)$曲线收敛至0并获得优异的解;当cp = 3时,$\phi(x)$曲线收敛至1.2×10^{-4},不满足$\delta = 10^{-4}$的要求. 因此,由以上两种极端情况可知,cp = 2的收敛特性更好.

4　结　论

本文首先对现有的约束处理技术进行综述,鉴于当前还没有一种有效的求解高维度COP的约束处理技术,在梯度下降法和多目标拥挤距离思想的基础上,提出了一种能够有效反映当前约束违反度分布信息的ω参数,并在MC的框架下,通过ω参数自适应控制μ的特点改进MC. 在此基础上,对CEC2017的COP进行实验,得出以下结论:

1) 在相同约束处理技术条件下,对于低维COP,不同的搜索机制求得的优化结果并无显著差异.

2) 对于复杂的含有等式约束的COP,适当地延迟迭代初期μ_G的收敛速度,有助于加快求得最优解.

3) 对于不同的约束处理技术,寻找有效的约束违反度分布信息对于自适应约束处理技术是非常有效的.

当然,对于一类特定的COP,最有效的做法是:适当地简化现有复杂的约束处理技术和搜索方法,以获得最满意的结果.

参考文献(References)

[1] 李智勇, 黄滔, 陈少淼, 等. 约束优化进化算法综述[J]. 软件学报, 2017, 28(6): 1529-1546.
(Li Z Y, Huang T, Chen S M, et al. Overview of constrained optimization evolutionary algorithms[J]. J of Software, 2017, 28(6): 1529-1546.)

[2] Neto J X V, Reynoso-Meza G, Ruppei T H, et al.

Solving non-smooth economic dispatch by a new combination of continuous GRASP algorithm and differential evolution[J]. Int J of Electrical Power & Energy Systems, 2017, 100(84): 13-24.

[3] Das S, Suganthan P N. Differential evolution: A survey of the state-of-the-art[J]. IEEE Trans on Evolutionary Computation, 2011, 15(1): 4-31.

[4] Brest J, Zamuda A, Fister I, et al. Some improvements of the self-adaptive jDE algorithm[C]. 2014 IEEE Symposium on Differential Evolution (SDE). Florida: IEEE, 2014: 1-8.

[5] Tanabe R, Fukunaga A. Evaluating the performance of SHADE on CEC 2013 benchmark problems[C]. 2013 IEEE Congress on Evolutionary Computation (CEC). Cancun: IEEE, 2013: 1952-1959.

[6] Zhang H, Rangaiah G P. An efficient constraint handling method with integrated differential evolution for numerical and engineering optimization[J]. Computers & Chemical Engineering, 2012, 37(4): 74-88.

[7] Mallipeddi R, Suganthan P N. Ensemble of constraint handling techniques[J]. IEEE Trans on Evolutionary Computation, 2010, 14(4): 561-579.

[8] Han S P, Mangasarian O L. Exact penalty functions in nonlinear Programming[J]. Mathematical Programming, 1979, 17(1): 251-269.

[9] Saha C, Das S, Pal K, et al. A fuzzy rule-based penalty function approach for constrained evolutionary optimization[J]. IEEE Trans on Cyberneics, 2014, 46(12): 2953-2965.

[10] Lemonge A C C, Barbosa H J C, Bernardino H S. Variants of an adaptive penalty scheme for steady-state genetic algorithms in engineering optimization[J]. Engineering Computations, 2015, 32(8): 2182-2215.

[11] Wu G, Mallipeddi R, Suganthan P N. Problem definitions and evaluation criteria for the CEC 2017 competition on constrained real-parameter optimization[R]. San Sebastian: IEEE (National University of Defense Technology, China, and Kyungpook National University, South Korea, and Nanyang Technological University, Singapore), 2017: 1-16.

[12] Deb K. An efficient constraint handling method for genetic algorithms[J]. Computer Methods in Applied Mechanics & Engineering, 1998, 186(2): 311-338.

[13] Poláková R. L-SHADE with competing strategies applied to constrained optimization[C]. 2017 IEEE Congress on Evolutionary Computation (CEC). San Sebastian: IEEE, 2017: 1683-1689.

[14] Mallipeddi R, Wu G, Lee M, et al. Gaussian adaptation based parameter adaptation for differential evolution[C]. IEEE Congress on Evolutionary Computation (CEC). Beijing: IEEE, 2014: 1760-1767.

[15] Takahama T, Sakai S. Constrained optimization by the constrained differential evolution with gradient-based mutation and feasible elites[C]. IEEE Int Conf on Evolutionary Computation. Vancouver: IEEE, 2006: 1-8.

[16] Li X, Zhang G. Biased multi-objective optimization for constrained single-objective evolutionary optimization[C]. Proc of the 11th World Congress on Intelligent Control and Automation. Shenyang: IEEE, 2014: 891-896.

[17] Sharma S, Rangaiah G P. An improved multi-objective differential evolution with a termination criterion for optimizing chemical processes[J]. Computers & Chemical Engineering, 2013, 56(1): 55-73.

[18] Conceicao E L T, Maechler M, Conceicao M E L T. DEoptimR: Differential evolution optimization in pure R[EB/OL]. (2016-11-19). http://CRAN.R-project.org/package = DEoptimR.

[19] Ali M, Kajee-Bagdadiz Z. A local exploration-based differential evolution algorithm for constrained global optimization[J]. Applied Mathematics and Computation, 2009, 208(1): 31-48.

[20] Tvrdik J, Polakova R. A simple framework for constrained problems with application of L-SHADE44 and IDE[C]. 2017 IEEE Congress on Evolutionary Computation (CEC). San Sebastian: IEEE, 2017: 1436-1443.

[21] Trivedi A, Sanyal K, Verma P, et al. A unified differential evolution algorithm for constrained optimization problems[C]. 2017 IEEE Congress on Evolutionary Computation (CEC). San Sebastian: IEEE, 2017: 1231-1238.

作者简介

徐玉琴(1964−), 女, 教授, 从事电力系统继电保护与安全防御等研究, E-mail: xyq@ncepu.edu.cn;

姚然(1993−), 男, 硕士, 从事电力系统分析、优化算法的研究, E-mail: imyaoran@163.com;

李鹏(1965−), 男, 教授, 博士生导师, 从事新能源并网发电与微网技术、电能质量分析与控制、电力电子技术等研究, E-mail: ncepulp@ncepu.edu.cn.

(责任编辑：李君玲)