

doi:10.3969/j.issn.1007-7162.2013.02.017

处理等式约束的新微分进化算法

雷 盛, 刘 伟

(广东工业大学 应用数学学院, 广东 广州 510006)

摘要: 针对有等式约束的优化问题, 提出一种新的微分进化算法. 该算法是通过解参数方程的方法处理等式约束, 从而降维求解的微分进化算法. 数值实验结果表明, 与现有的其他算法相比, 新算法具有较快的收敛速度和较高的求解精度, 是一种有效的智能算法.

关键词: 微分进化; 等式约束; 参数方程

中图分类号: O242

文献标志码: A

文章编号: 1007-7162(2013)02-0090-05

New Differential Evolution Algorithm for Equality Constrained Optimization

Lei Sheng, Liu Wei

(School of Applied Mathematics, Guangdong University of Technology, Guangzhou 510006, China)

Abstract: A new differential evolution algorithm is presented to solve the problem with equality constrained optimization. The parametric equation method was taken to keep particles satisfying equality constraints, and the penalty method was used to handle inequality constraints, thus reducing the dimensionality of the optimization problem. The experiment results demonstrate that the new differential evolution algorithm is superior to some other methods, and it is effective.

Key words: differential evolution; equality constraints; parametric equation

微分进化算法(differential evolution algorithm, DE 算法)是由 Rainer Storn 等^[1-3]于 1995 年提出的一种实数编码, 解决全局优化问题的启发式算法. 已被证明在求解过程中具有高效性、收敛性、鲁棒性等优点. 因此微分进化算法在计算机科学、管理科学、控制工程的科学领域得到了广泛的应用.

微分进化算法在求解约束优化问题时, 处理等式约束常用的一种方法是把一个等式约束转化为两个不等式约束, 这种方法会在算法求解过程中得到大量不可行解, 而且在初始化时也不能保证个体合法^[4-5]. 因此降低了算法的求解精度和收敛的速度, 并且不一定能得到满足等式约束的最优解. 另外一种方法是在等式约束的右边添加一个极小的数, 把等式约束条件减弱, 这种方法的缺陷是等式约束条件减弱以后, 得到解可能比真实的最优解要小, 约束条件减弱得越多, 即添加的数取值越大, 得到的解所对应的目标函数值可能要比真实的最优值更好, 但

是, 这样的解不是真实的可行解^[6-9]. 本文提出了一种新的微分进化算法求解有等式约束优化问题. 首先根据搜索空间的维数和等式约束条件确定参数, 并求解参数方程; 然后在种群初始化、交叉和变异操作过程中, 用参数方程的解更新个体以确保个体满足等式约束, 对于违反不等式约束条件的个体使用惩罚函数进行惩罚. 数值实验验证了新算法既有较快的收敛速度, 又能以较大概率搜索到约束优化问题的最优解.

1 基本微分进化算法

优化问题模型描述如下.

$$\begin{aligned} \min & f(X). \\ \text{s. t. } & \begin{cases} Q_i(X) \leq 0, i = 1, 2, \dots, N; \\ P_j(X) = 0, j = 1, 2, \dots, M. \end{cases} \end{aligned} \tag{1}$$

其中变量 $X \in R^D$, R 代表实数域, $X = (x_1, x_2, \dots, x_D)$, $x_i \in [a_i, b_i]$ ($i = 1, 2, \dots, D$). $Q_i(X)$ 为优化

问题的 N 个不等式约束条件, $P_i(X)$ 为 M 个等式约束条件.

DE 算法是求解有 n 个连续变量采用实数编码的全局优化的算法. 其基本思想是:对种群中的每一个个体 i , 从当前种群中随机选择 3 个不同且不等于个体 i 的个体. 以其中 1 个个体为基础, 另外 2 个个体为参照做一个随机的扰动. 将得到的新个体与个体 i 进行比较, 保留较优个体, 实现种群的进化.

假设 DE 算法的种群规模为 NP, 每个个体有 D 维变量. 则第 G 代第 i 个个体可表示为 $X_{i,G}$, 其中 $i = 1, 2, \dots, \text{NP}$. 算法的主要算子包括: 变异、交叉、选择. 算法流程如下.

步骤 1 初始化参数: 种群规模 NP, 变异因子 X_r , 交叉概率 C_r , 以及种群最大进化代数 Gen_max.

步骤 2 初始化种群: 随机产生 NP 个个体. 计算每个个体 $X_{i,G}$ 的适应值 $f(X_{i,G})$.

步骤 3 变异: 对个体 i 通过变异操作得到变异个体 $V_{i,G+1}$, 变异方式如下:

$$V_{i,G+1} = X_{a,G} + X_r(X_{b,G} - X_{c,G}), \quad (2)$$

其中 $a, b, c \in \{1, 2, \dots, \text{NP}\}$ 且 $a \neq b \neq c \neq i$.

步骤 4 交叉: 将变异得到的中间个体 $V_{i,G+1} = (v_{1i,G+1}, v_{2i,G+1}, \dots, v_{Di,G+1})$ 和个体 $X_{i,G}$ 进行交叉操作, 得到候选个体 $U_{i,G+1} = (u_{1i,G+1}, u_{2i,G+1}, \dots, u_{Di,G+1})$, 交叉操作的方法如下:

$$u_{ji,G+1} = \begin{cases} v_{ji,G+1}, & \text{如果 } \text{rand}() < C_r \text{ 或 } \text{mbr}(i) = j; \\ x_{ji,G}, & \text{其他.} \end{cases} \quad (3)$$

步骤 5 选择: 对候选个体 $U_{i,G+1}$ 进行适应度评价, 然后和个体 $X_{i,G}$ 比较, 将较优个体保留到下一代种群. 判断是否满足算法终止条件. 若满足则输出种群中最优个体, 否则转到步骤 3.

2 新微分进化算法

在文献[8]中处理等式约束的方法是: 把个体的维数为 D 的 M 个等式作为一个带有参数的方程组. 不失一般性选择其中的 $D - M$ 个变量作为求解的参数. 本文将这种方法引入到微分进化算法中. 通过参数方程组求解之后, 可以得到参数解的一般形式:

$$\begin{aligned} x_{D-M+1} &= q_1(x_1, x_2, \dots, x_{D-M}), \\ x_{D-M+2} &= q_2(x_1, x_2, \dots, x_{D-M}, x_{D-M+1}), \\ &\dots \\ x_{D-1} &= q_{M-1}(x_1, x_2, \dots, x_{D-3}, x_{D-2}), \\ x_D &= q_M(x_1, x_2, \dots, x_{D-2}, x_{D-1}). \end{aligned} \quad (4)$$

其中 $q_i, i = 1, 2, \dots, M$ 是参数的多项式. 然后将所得到的参数解代入原问题的不等式约束和目标函数中, 从而降低优化问题解的维数.

惩罚函数法是处理不等式约束优化问题最简单的一种方法. 从本质上讲, 这种方法是通过将不可行解添加惩罚来将约束问题转化成无约束问题^[8-11]. 所有违反约束的个体都要在适应值函数中添加惩罚项. 具体方式为:

$$\text{Fitness}(X) = \begin{cases} f(X), & \text{若 } X \text{ 是可行解;} \\ f(X) + NK, & \text{若 } X \text{ 是不可行解.} \end{cases} \quad (5)$$

式中: K 为个体 X 不满足约束条件的个数, N 是一个很大的正数.

新算法步骤如下.

步骤 1 参数初始化: 参数包括种群规模 NP, 变异因子 X_r , 交叉概率 C_r .

步骤 2 种群初始化: 随机产生初始个体 $X_{i,0}$, $i = 1, 2, \dots, \text{NP}$. 把参数解式(4)代入 $X_{i,0}$, 计算得到结果 $(X_{i,0})^*$, 然后判断 $(X_{i,0})^*$ 是否在约束边界内, 如果个体不满足约束条件则重新随机产生 $X_{i,0}$. 然后按式(5)计算适应值.

步骤 3 变异: 对个体 i 利用式(2)进行变异操作得到变异个体 $V_{i,G+1}$.

步骤 4 交叉: 将变异得到的中间个体 $V_{i,G+1} = (v_{1i,G+1}, v_{2i,G+1}, \dots, v_{Di,G+1})$ 和个体 $X_{i,G}$ 利用式(3)进行交叉操作, 得到候选个体 $U_{i,G+1} = (u_{1i,G+1}, u_{2i,G+1}, \dots, u_{Di,G+1})$.

步骤 5 选择: 对候选个体 $U_{i,G+1}$ 进行适应度评价, 然后和个体 $X_{i,G}$ 比较, 将较优个体保留到下一代种群. 判断是否满足算法终止条件. 若满足则输出种群中最优个体, 否则转到步骤 3.

3 数值实验

本文选择了 5 个具有代表性的带等式约束的优化问题.

$$f_1(x) = \frac{x_1 + x_2 + x_3}{x_1^2 + x_2^2 + x_3^2}.$$

$$\text{s. t. } \begin{cases} x_1 + x_2 + x_3 = 10, \\ x_1^2 + 2x_2^2 + 3x_3^2 \leq 90, \\ x_1, x_2, x_3 \geq 0. \end{cases}$$

$$\begin{aligned} f_2(x) &= x_1^2 - x_2^2 + x_3^2 - x_4^2 + 2x_1x_2 - x_1 - x_2 - 5x_3 - \\ &x_4 - 6x_5 + x_6 + x_7. \end{aligned}$$

$$\begin{aligned} 2x_1 + 2x_2 + x_3 - x_6 &\leq 4, \\ x_2 + 2x_3 + 2x_4 - x_7 &\leq 4, \\ x_3 + x_4 + x_5 &\leq 4, \\ \text{s. t} \quad x_1 + x_2 - x_6 &= 0, \\ x_3 + x_4 - x_7 &= 0, \\ x_1, x_2, \dots, x_7 &\geq 0. \end{aligned}$$
$$\begin{aligned} f_3(x) &= (x_1 - 10)^2 + 5(x_2 - 12)^2 + x_3^4 + 3(x_4 - 11)^2 + 10x_5^6 + 7x_6^2 + x_7^4 - 4x_6x_7 - x_1 - x_2 - x_3 - x_4 - x_5 - 10x_6 - 9x_7 + x_8 + x_9 + x_{10}. \\ -127 + 2x_1^2 + 3x_2^4 - x_3 + 4x_4^2 + 3x_5 + 2x_8 &\leq 0, \\ -282 + 7x_1 + 4x_2 + 10x_3^2 + 2x_4 - x_5 - x_9 &\leq 0, \\ -196 + 24x_1 + x_2^2 + 6x_6^2 - 7x_7 - x_{10} &\leq 0, \\ 4x_1^2 + x_2^2 - 3x_1x_2 + 2x_3^2 + 5x_6 - 11x_7 &\leq 0, \\ \text{s. t} \quad x_3 + x_5 - x_8 &= 0, \\ x_2 + x_4 - x_9 &= 0, \\ x_1 + x_7 - x_{10} &= 0, \\ -10 \leq x_i \leq 10, i = 1, 2, \dots, 10. \end{aligned}$$

$$\begin{aligned} f_4(x) &= 3x_1 + 4x_2 + 2x_3 + x_4 - 5 \sum_{i=1}^4 x_i^2 - \sum_{i=5}^{13} x_i - x_5 - x_{10} - x_{11} - x_{12} + x_{14} + x_{15} - x_{16} + x_{17}. \\ -2x_1 + x_2 - x_{10} + x_{11} + 2x_{14} - 10 &\leq 0, \\ 2x_1 - 4x_3 + x_{10} - x_{12} + 2x_{15} - 10 &\leq 0, \\ 3x_2 + 2x_3 + 2x_{11} + x_{12} + x_{16} - 10 &\leq 0, \\ -6x_1 + 2x_{10} - x_{14} &\leq 0, \\ -7x_2 + 2x_{11} + x_{16} &\leq 0, \\ -5x_3 + 2x_{12} - x_{15} &\leq 0, \\ -6x_4 - 2x_5 + x_{10} + x_{17} &\leq 0, \\ \text{s. t} \quad -2x_6 - x_7 + x_{11} &\leq 0, \\ -2x_8 - x_9 + x_{12} &\leq 0, \\ 2x_1 + x_{10} - x_{14} &= 0, \\ 3x_3 + x_{12} - x_{15} &= 0, \\ x_2 + x_{11} + x_{16} &= 0, \\ 4x_4 + x_5 - x_{17} &= 0, \\ 0 \leq x_{1,2,\dots,9,13} \leq 1, 0 \leq x_{10,11,12} \leq 100, \\ 0 \leq x_{14,15,17} \leq 110, -100 \leq x_{16} \leq 100. \end{aligned}$$

$$\begin{aligned} f_5(x) &= -x_1x_2 - 2x_2 + (x_3 - 10)^2 + 4(x_4 - 5)^2 + (x_5 - 3)^2 - x_5 + 2(x_6 - 1)^2 - x_6 + 5x_7^2 - x_7 + 7(x_8 - 11)^2 - x_8 + 2(x_9 - 10)^2 - x_9 + (x_{10} - 7)^2 - x_{10} + x_{11}^2 - 14x_{11} + x_{12} + x_{13} + x_{14} + 45. \end{aligned}$$

$$\begin{aligned} -105 + 3x_1 + 4x_2 - 3x_7 + 9x_8 + x_{11} &\leq 0, \\ 10x_1 - 8x_2 - 18x_7 + x_8 + x_{12} &\leq 0, \\ -8x_1 + 2x_2 + 4x_9 - 3x_{10} + x_{13} - 12 &\leq 0, \\ 3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4 - 120 &\leq 0, \\ 5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4 - 40 &\leq 0, \\ x_1^2 + 2(x_2 - 2)^2 - 2x_1x_2 + 13x_5 - 7x_6 + x_{14} &\leq 0, \\ \text{s. t} \quad 0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6 - 30 &\leq 0, \\ -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} &\leq 0, \\ x_1 + x_2 - x_{11} &= 0, \\ x_5 + x_6 - x_{14} &= 0, \\ x_7 + x_8 - x_{12} &= 0, \\ x_9 + x_{10} - x_{13} &= 0, \\ -10 \leq x_i \leq 10, i = 1, 2, \dots, 10, -20 \leq x_j \leq 20, \\ j = 11, 12, 13, 14. \end{aligned}$$

数值实验在 Matlab 中完成. 对于问题 $f_1(x)$, 算法参数中种群规模 $N = 30$, 进化代数 $T = 200$, $X_f = 0.5$, $C_r = 0.5$. 问题 $f_2(x)$, 算法参数 $N = 30$, $T = 1\,000$, $X_f = 0.5$, $C_r = 0.5$. 问题 $f_3(x)$, 算法参数 $N = 30$, $T = 500$, $X_f = 0.5$, $C_r = 0.5$. 问题 $f_4(x)$, 算法参数 $N = 30$, $T = 2\,000$, $X_f = 0.5$, $C_r = 0.5$. 问题 $f_5(x)$, 算法参数 $N = 30$, $T = 5\,000$, $X_f = 0.5$, $C_r = 0.5$. 为了比较方便, 新算法记为 LDE, 文献[3]中的算法记为 JDE, 文献[4]中的方法记为 CDE. 文献[3]把约束条件中的等式约束 $h_j(x) = 0$ 转化成不等式约束, 即 $h_j(x) = 0 \Leftrightarrow |h_j(x)| < \varepsilon$, 其中 $\varepsilon = 0.1$. 文献[4]中, 处理约束条件的方法是把个体违反每个约束条件的量累加起来, 选择违反程度小且适应值好的个体.

3 种算法对每一个测试函数都独立重复运行 10 次, 为了比较算法的效率, 本节比较了 10 次实验 3 种算法的平均 CPU 时间和求得的最优解及平均解 (见表 1). 并给出了一次实验的进化曲线图 (见图 1 ~ 图 5).

表 1 中“ - ”表示算法在绝大多数情况下不能收敛到最优解, 所以未作统计. 由表 1 可以看出在 5 个测试函数中, 新算法求解所需要的 CPU 时间都要比另外 2 个算法少, 得到的最好解和平均解的精度也要高于另外两种算法. 当测试函数 $f_4(x)$ 、 $f_5(x)$ 中等式约束条件较多且比较复杂的时候, 其他两种算法即使花费很多的时间和求解代数也很难求到最优解. 说明了新算法简洁、高效. 从进化曲线图 1 中可以看出对于较简单的测试函数 $f_1(x)$ 新算法求解到最优解的进化代数少于 20 代而其他两种算法需要 100 代; 从进化曲线图 2 中可以看出对于测试函数 $f_2(x)$

表 1 5 个测试函数收敛精度和运行时间的比较

Tab.1 Five testing functions convergence accuracy and running time comparison

测试函数	算法	最好解	平均解	平均 CPU 时间/s
$f_1(x)$	LDE	-0.054 9	-0.054 5	0.032 0
	JDE	-0.043 2	-0.028 4	0.096 1
	CDE	-0.053 4	-0.051 3	0.079 1
$f_2(x)$	LDE	-40.000 0	-40.000 0	0.206 8
	JDE	-39.074 8	-37.630 1	0.223 1
	CDE	-40.212 4	-39.624 5	0.570 1
$f_3(x)$	LDE	680.660 1	680.673 3	0.236 5
	JDE	680.786 2	688.203 1	0.261 2
	CDE	681.152 8	685.259 7	0.338 3
$f_4(x)$	LDE	-15	-14.998 0	0.688 3
	JDE	-19.034 3	-8.716 4	2.463 9
	CDE	-	-	-
$f_5(x)$	LDE	24.226 5	24.626 5	3.961 4
	JDE	-	-	-
	CDE	27.159 4	34.672 0	7.473 3

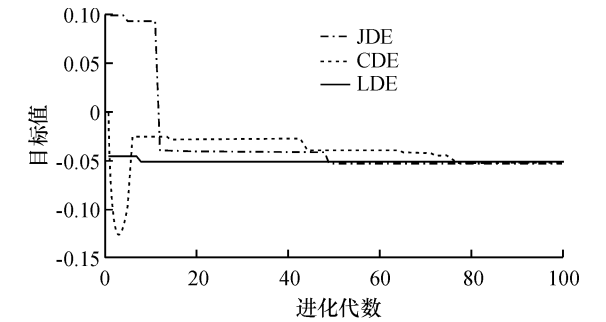


图 1 $f_1(x)$ 的进化曲线图

Fig.1 Convergence curves for function $f_1(x)$

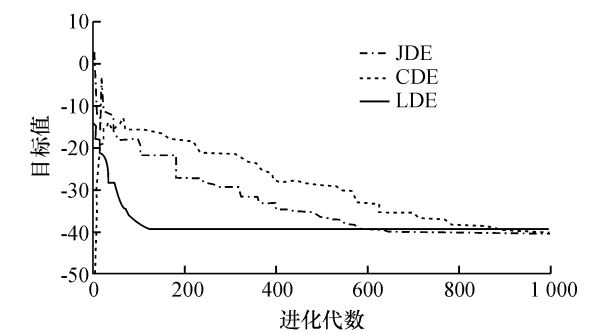


图 2 $f_2(x)$ 的进化曲线图

Fig.2 Convergence curves for function $f_2(x)$

新算法求解到最优解需要 100 代,而 CDE 需要进化 1 000 代,JDE 需要进化 600 代;从进化曲线图 3 中可以看出对于测试函数 $f_3(x)$ 新算法求解到最优解需要进化 100 代,而另外两种算法需要进化 500 代.

由图 4 和图 5 可以看出当测试函数中等式约束较多较复杂的时候,另外两种算法需要的进化代数远大于新算法,新算法求解到 $f_4(x)$ 的最优值需要进化 300 代,JDE 需要进化 2 000 代,而 CDE 在大多数情况下不能收敛到最优解;新算法求解到 $f_5(x)$ 的最优值需要进化 100 代,JDE 需要 1 000 代,CDE 需要 5 000 代.综上所述,新算法具有很快的收敛速度和很高的求解精度,是一种高效稳健的智能算法.

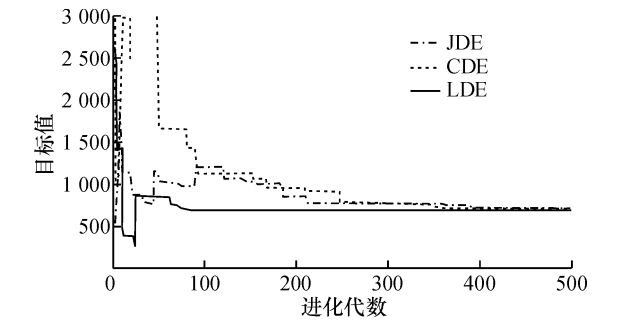


图 3 $f_3(x)$ 的进化曲线图

Fig.3 Convergence curves for function $f_3(x)$

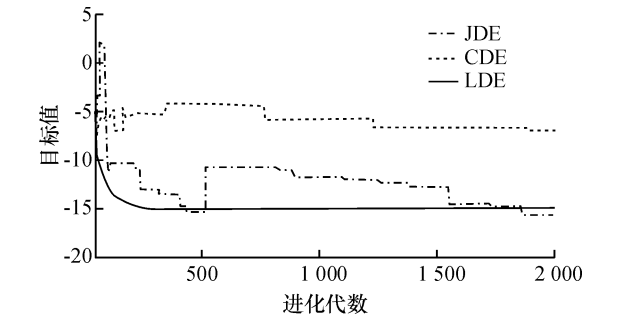


图 4 $f_4(x)$ 的进化曲线图

Fig.4 Convergence curves for function $f_4(x)$

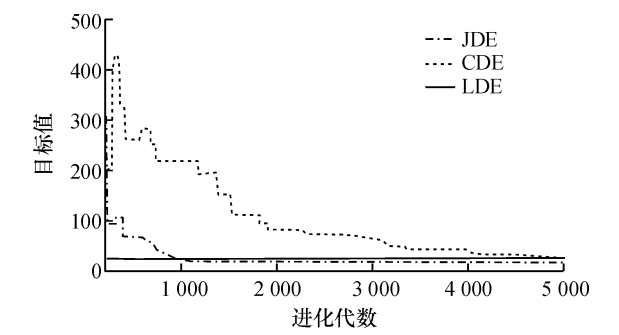


图 5 $f_5(x)$ 的进化曲线图

Fig.5 Convergence curves for function $f_5(x)$

4 结论

本文提出了一种用参数方程处理等式约束,降低优化问题维数,用惩罚函数处理不等式约束的微分进化算法.数值实验显示该算法提高了求解的速度和精度.新算法减法是一种便于实现、通用性强、

高效的方法.

参考文献:

- [1] Rainer Storn, Kenneth Price. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces [J]. Journal of Global Optimization, 1997, 11(4):341-359.
- [2] Rainer Storn. Designing nonstandard filters with differential evolution [J]. IEEE Signal processing Magazine, 2005, 22(1):103-106.
- [3] Sandra Paterlinia, Thimo Krinkb. Differential evolution and particle swarm optimization in partitional clustering [J]. Computational Statistics & Data Analysis, 2006, 50(5):1220-1247.
- [4] 李秀梅,刘华毅,徐景德. 一种新的遗传算法求解约束优化问题[J]. 计算技术与自动化,2003,22(1):17-20.
Li Xiu-mei, Liu Hua-yi, Xu Jing-de. A new genetic algorithm for constrained optimization[J]. Computing Technology and Automation,2003,22(1):17-20.
- [5] 闵涛,李辉,杨晓莉. 求解约束优化问题的一种新算法[J]. 计算机工程与应用,2010,46(15):51-52.
Ming Tao, Li Hui, Yang Xiao-li. New algorithm for constrained optimization problem [J]. Computer Engineering and Applications, 2010,46(15):51-52.
- [6] 刘伟,刘海林. 基于外点法的混合遗传算法求解约束优化问题[J]. 计算机应用,2007,27(1):238-240.
Liu Wei, Liu Hai-lin. Hybrid genetic algorithm based on external point method for constrained optimization[J]. Journal of Computer Applications,2007,27(1):238-240.
- [7] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: a comparative case and the strength Pareto approach[J]. IEEE Trans on Evolutionary Computation,1999,3(4):257-271
- [8] 刘伟,蔡前凤,王振友. 一种新的遗传算法求解有等式约束的优化问题[J]. 计算机工程与设计,2007(28):3184-3185.
Liu Wei, Cai Qian-feng, Wang Zheng-you. New genetic algorithm for equality constrained optimization problem [J]. Computer Engineering and Design,2007(28):3184-3185
- [9] Kusum Deep, Manoj Thakur. A new crossover operator for real coded genetic Algorithms [M]. India: Elsevier Inc, 2007.
- [10] 余新华,孙作龙. 带约束函数优化问题的新算法[J]. 武汉理工大报,2002,24(5):13-16.
Yu Xin-hua, Sun Zuo-long. A new algorithm to optimize functions with restraints[J]. Journal of Wuhan University of Technology,2002,24(5):13-16.
- [11] 江家宝,尤振燕,孙俊. 基于微分进化算法的多阶段投资组合优化[J]. 计算机工程与应用,2007,43(3):189-193.
Jiang Jia-bao, You Zhen-yan, Sun Jun. Multi-stage portfolio optimization using differentiation evolution algorithms [J]. Computer Engineering and Applications, 2007, 43(3):189-193.