混合整数非线性规划问题的改进差分进化算法 *

邓长寿[1,2], 任红卫[1], 彭　虎[1]
(1. 九江学院 信息科学与技术学院, 江西 九江 332005; 2. 合肥工业大学 计算机网络系统研究所, 合肥 230009)

摘　要: 提出一种改进差分进化算法求解混合整数非线性规划问题。该算法利用同态映射方法,解决差分进化算法无法直接处理整数决策变量问题;提出改进的自适应交替变异算子,提高算法的搜索性能;提出一种自适应保留不可行解的方法处理约束条件,并对差分进化算法的选择算子进行改进,提出一种直接处理约束条件的新选择算子。六个常用的混合整数非线性规划问题的实验结果表明了该方法的有效性和适用性。

关键词: 混合整数非线性规划问题; 同态映射; 自适应交替变异算子; 约束处理
中图分类号: TP18　　**文献标志码**: A　　**文章编号**: 1001-3695(2012)02-0445-04
doi:10.3969/j.issn.1001-3695.2012.02.010

Improved differential evolution algorithm for mixed integer nonlinear programming problem

DENG Chang-shou[1,2], REN Hong-wei[1], PENG Hu[1]
(1. *School of Information Science & Technology*, *Jiujiang University*, *Jiujiang Jiangxi* 332005, *China*; 2. *Institute of Computer Network*, *Hefei University of Technology*, *Hefei* 230009, *China*)

Abstract: This paper proposed an improved differential evolution algorithm for the mixed integer nonlinear programming problem. Used a homomorphous mapping in the algorithm to deal the integer decision variables, and used a self-adaptive alternating mutation operator to improve the search performance. In addition, it used a self-adaptively infeasible solution biased way to construct a new selection operator which could deal the constraints directly. The experiment results of six benchmarks show the new algorithm is efficient and practical for the mixed-integer nonlinear programming problem.

Key words: mixed integer nonlinear programming problem; homomorphous mapping; self-adaptive alternating mutation operator; constraint handling

0　引言

数值优化问题是科学研究、工程设计及管理科学中常见的问题,其中许多问题属于混合整数非线性规划问题(mixed integer nonlinear programming problem, MINP),其数学模型如下:

$$\min f(x,y)$$
$$\text{s.t.}\begin{cases} g_i(x,y)\leqslant 0,\ i=1,2,\cdots,m \\ h_i(x,y)\leqslant 0,\ i=m+1,m+2,\cdots,m+l \\ x^l\leqslant x\leqslant x^u \\ y^l\leqslant y\leqslant y^u \\ x\in R \\ y\in I \end{cases} \tag{1}$$

其中:x 为实型决策变量,y 为离散变量(整数或者二进制数);$g_i(x,y)(i=1,2,\cdots,m)$表示 m 个不等式约束;$h_i(x,y)(i=m+1,m+2,\cdots,m+l)$表示 l 个等式约束。

求解 MINP 的确定性方法有分支定界法、广义 Benders 分解法和近似方法等,这些基于梯度的传统方法只能得到与初始点有关的局部最优解,而不能保证搜索到全局最优点。MINP 通常是高度非线性、不可微、多峰及非凸的,对于这样的优化,确定性算法往往无能为力。近年来,许多研究人员利用进化算法求解 MINP 问题。文献[1]比较了基于模拟退火算法(M-SIMPSA)、遗传算法以及演化策略三种算法求解 MINP 的差异;文献[2,3]分别利用混合粒子群算法(PSO)和量子 PSO 求解 MINP;文献[4]利用混合进化算法求解 MINP;文献[5,6]分别利用差分进化算法(DE)求解 MINP。这些算法求解 MINP 时,对于整数决策变量采用实数取整的方法,即在实数域进行优化求解,然后对实数值进行取整,得到离散变量值。这种方法对部分测试用例获得了较好的结果,而对另外一类测试问题很难取得好的结果[7]。此外,对于约束条件的处理,这些方法通常选择使用罚函数方法,然而利用罚函数处理约束条件时,存在惩罚因子难以确定的问题。如果惩罚因子过大,则容易早熟收敛;反之,容易出现收敛速度慢,甚至无法收敛到可行解的情况。针对 MINP,本文提出一种改进的差分进化算法(IDE)。

1　差分进化算法

DE 算法首先在问题的可行解空间随机产生初始种群,然后对当前种群进行变异和交叉操作,产生一个过渡种群;接着

收稿日期: 2011-07-02; **修回日期**: 2011-08-15　**基金项目**: 国家自然科学基金资助项目(50705039);江西省教育厅科技计划资助项目(GJJ10616, GJJ11616)

作者简介: 邓长寿(1972-),男,安徽肥西人,教授,博士,CCF 会员,主要研究方向为计算智能、数据挖掘(dengtju@yahoo.com.cn);任红卫(1968-),男,讲师,硕士,主要研究方向为人工智能;彭虎(1981-),男,讲师,硕士,主要研究方向为群体智能算法.

利用基于贪婪的思想对当前种群和过渡种群中的对应个体进行一对一的选择，产生新一代群体。DE 算法根据变异和交叉操作的不同形成了不同的模式。常见的是 DE/rand/1/bin 和 DE/best/1/bin。其中 DE/rand/1/bin 的算法描述如下[8]。

设 DE 算法种群规模为 NP，每个个体是 D 维向量，则第 G 代中的个体可表示为 $X_{i,G}(i=1,2,\cdots,NP)$。DE 算法的主要操作包括产生初始种群、变异算子、交叉算子和选择算子。

产生初始种群即在 D 维空间中，随机产生均匀分布的满足上下界约束的 NP 个个体的种群 $X_{i,0}(i=1,2,\cdots,NP)$。

DE 算法的变异算子是利用两个随机个体间的偏差扰动产生新个体。变异操作后得到中间个体记为 $v_{i,G+1}$，即

$$v_{i,G+1}=X_{r1,G}+F\times(X_{r2,G}-X_{r3,G}) \tag{2}$$

交叉算子是将当前所得到的中间个体 $v_{i,G+1}$ 与个体 $X_{i,G}$ 进行杂交，如式(3)所示。交叉操作得到当前个体的候选个体 $u_{i,G+1}=(u_{1i,G+1},u_{2i,G+1},\cdots,u_{Di,G+1})$。

$$u_{ij,G+1}=\begin{cases} v_{ij,G+1} & \text{if}(\text{rand}<CR)\ \text{or}\ j=R(i) \\ X_{ij,G} & \text{otherwise} \end{cases} \tag{3}$$

其中：$i=1,2,\cdots,NP,j=1,2,\cdots,D$；$\text{rand}\in[0,1]$ 是一个均匀分布的随机数；$R(i)\in[1,D]$ 间的随机整数；$CR\in[0,1]$ 为交叉概率。采用这种交叉策略，可以确保下一代个体中至少有一个染色体来源于中间个体。

选择算子先评价候选个体的函数适应值，然后根据式(4)决定哪个个体进入下一代种群。

$$X_{i,G+1}=\begin{cases} u_{i,G+1} & \text{if}\ f(u_{i,G+1})\leqslant f(X_{i,G}) \\ X_{i,G} & \text{otherwise} \end{cases} \tag{4}$$

从上述 DE 算法的描述可以发现，DE 算法的变异算子仅在实数域有效。当其用于整数决策变量或者二进制决策变量时，变异算子无法保持封闭性。

2　求解 MINP 的改进差分进化算法

2.1　同态映射算子

为了使 DE 算法有效求解 MINP 中的整数决策变量(包括二进制数)问题，本文提出一种同态映射的方法实现利用实数编码的差分进化算法求解整数规划问题。

假设待求解的整数规划问题的变量 y 取值范围为 $[-Z,Z]$，$x\in[0,1]$ 是实数。将区间 $[0,1]$ 划分为 $(2Z+1)$ 个相等的子区间，将各个子区间的上下界记为 $SR=\left[0,\frac{1}{2Z+1},\frac{2}{2Z+1},\cdots,\frac{2Z+1}{2Z+1}\right]$。例如，当 $x\in\left[0,\frac{1}{2Z+1}\right)$ 时，$y=-Z$；$x\in\left[\frac{1}{2Z+1},\frac{2}{2Z+1}\right)$ 时，$y=-Z+1$；依此类推，可以在小区间与一个整数之间建立一对一映射关系。同态映射本质上是利用一个小的区间来表示一个整数。

2.2　自适应交替变异算子

DE 算法中，DE/rand/1/bin 模式的变异算子中基变量是一个随机量；而 DE/best/1/bin 模式的变异算子中基变量是当前种群中的最好向量。因此 DE/rand/1/bin 模式变异算子具有较好的全局搜索能力，而 DE/best/1/bin 模式的变异算子具有较强的局部搜索能力。为了充分发挥这两种变异算子的优势，算法在演化迭代求解过程中，自适应地交替使用这两种不同的算子。在算法迭代初期，以较大的概率选择 DE/rand/1/bin 模式的变异算子；在迭代的后期，以较大的概率选择 DE/best/1/bin 变异算子，以获得较强的局部搜索能力。交替变异算子定义如下：

$$v_{i,G+1}=\begin{cases} X_{r1,G}+F\times(X_{r2,G}-X_{r3,G}) & \text{if}(\lambda<\text{rand}) \\ X_{\text{best},G}+F\times(X_{r2,G}-X_{r3,G}) & \text{otherwise} \end{cases} \tag{5}$$

其中：变量 λ 控制两种不同变异模式的选择概率。λ 随着迭代次数自适应调整，如式(6)所示。

$$\lambda=2-\exp(\text{gen}/\text{GENMAX}\times\log(2)) \tag{6}$$

其中：gen 是当前的迭代次数，GENMAX 是最大迭代次数。变量 λ 随着迭代次数自适应地变化。以最大代数 500 为例(下文 problem 5)，变量 λ 随着迭代次数的变化如图 1 所示。

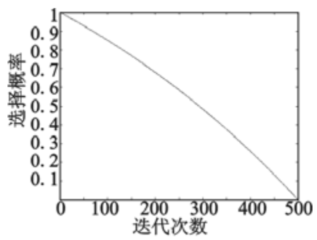

图1　选择概率自适应变化

2.3　边界约束处理算子

DE 算法中，通过变异算子与交叉算子所产生的新个体中的某分量可能违反了边界约束条件。本文采用重新生成新的分量方法解决此问题，即利用式(7)定义的边界约束处理算子，生成一个新的分量来代替因变异和交叉操作所产生的不良分量。

$$u_{ij}(t+1)=\begin{cases} x_j^l+\text{rand}(0,1)(x_j^l-x_j^u) & \text{if}\ u_{ij}(t+1)>x_j^u\ \text{or}\ u_{ij}(t+1)<x_j^l \\ u_{ij}(t+1) & \text{otherwise} \end{cases} \tag{7}$$

边界约束处理算子不仅可以解决边界约束问题，而且适当增加了个体的多样性。

2.4　约束处理

进化算法求解 MINP 时，约束条件的处理是其关键之一。本文在处理约束条件的 Deb 准则[9]的基础上，提出一种概率 Deb 准则；基于概率 Deb 准则，重新定义一种选择算子。该选择算子在目标函数适应值与违反约束条件的程度之间进行平衡，选择优秀个体进入下一代种群。

Deb 准则如下：a)当两个比较的个体中，一个个体为可行解，另外一个个体为不可行解时，选择可行解；b)当两个比较的个体均为可行解时，选择目标函数值小的个体；c)当两个比较的个体均为不可行解时，选择违反约束条件程度小的个体。

Deb 准则的主要缺陷是难以发挥不可行解的作用，特别是当群体中的绝大部分个体均为可行解或者是最优解处于搜索区域的边界时，不可行解将很难进入群体。针对此问题，本文对 Deb 准则的第一条进行修改，提出一种概率 Deb 准则。概率 Deb 准则第一条如下：

当在两个比较的个体中，一个个体为可行解，另外一个个体为不可行解时，以较大的概率选择可行解，以较小的概率选择不可行解，而且选择不可行解的概率逐渐变为 0。

概率 Deb 准则的第二条和第三条与 Deb 准则相同。

为了定义新选择算子，首先给出个体违反约束的程度定义。对于式(1)所示的 MINP，违反约束程度的函数为

$$\Phi(x)=\sum_{j=1}^{l}\max(0,g_i(x))+\sum_{j=m+1}^{m+l}|h_i(x)| \tag{8}$$

对于式(1)所示的最小约束优化问题,设 X 和 Y 为参与竞争的两个不同个体。定义函数 select(X,Y) 如下:

$$\text{select}(X,Y)=\begin{cases}1 & \text{if } ((\Phi(X)>0 \wedge \Phi(Y)=0)\wedge(\text{rand}<\varepsilon/\text{gen}))\\ 1 & \text{if } (\Phi(X)=0)\wedge(\Phi(Y)=0)\wedge(f(X)<f(Y))\\ 1 & \text{if } (\Phi(X)>0)\wedge(\Phi(Y)>0)\wedge(\Phi(X)<\Phi(Y))\\ 0 & \text{otherwise}\end{cases} \tag{9}$$

其中:ε 为较小的实数,下文的实验中 $\varepsilon=0.001$;gen 为当前的迭代次数。若 select$(X,Y)=1$,则个体 X 优于个体 Y;若 select$(X,Y)=0$,则个体 Y 优于个体 X。从式(9)的 select(X,Y) 函数可以看出,随着迭代次数的增加,不可行解进入下一代的概率逐渐趋小,最后渐趋于 0,保证了最后的最优解为可行解。

基于上述定义,新选择算子如式(10)所示。

$$X_{i,G+1}=\begin{cases}u_{i,G+1} & \text{if select}(u_{i,G+1},X_{i,G})\\ X_{i,G} & \text{otherwise}\end{cases} \tag{10}$$

上述选择算子包含了处理约束条件,避免了罚函数方法选择惩罚因子难以确定的问题。

2.5　改进 DE 算法步骤

将最好的个体向量记为 x_{best},改进差分进化算法的具体步骤如下:

a)初始化。包括设置最大迭代代数、种群中个体数目、变异概率、交叉概率和生成初始种群 X。

b)对于整数型决策变量,执行同态映射算子。计算每个个体的函数适应值和违反约束的程度,按照改进差分进化算法选择算子,求出 x_{best}。

c)不满足停止条件,则执行步骤 d)f);否则转步骤 g)。

d)按照式(2)所示的变异算子对种群实施变异操作,得到中间种群 V。

e)利用交叉算子对种群中的每个个体进行交叉操作,得到新的候选种群 U,对 U 中的每个个体实施利用边界约束处理算子进行操作。

f)对于整数型决策变量,执行同态映射算子。计算候选种群 U 每个个体的适应值和违反约束的程度,按式(9)的选择算子,从种群 X 和候选种群 U 中选择个体进入下一代种群;重新计算最好的个体向量 x_{best}。

g)输出最优结果,算法结束。

3　数值实验

为了检验本文算法的性能,利用常用的六个混合整数非线性规划问题进行实验[1]。六个问题分别如下,其中,problem 2 和 problem 4 分别用该文献中不含等式约束条件的问题。

1)problem 1

$$\begin{aligned}&\min f(x,y)=2x+y\\ &\text{s.t.}\quad 1.25-x^2-y\leqslant 0\\ &x+y\leqslant 1.6,0\leqslant x\leqslant 1.6,y\in\{0,1\}\end{aligned} \tag{11}$$

已知最优解为$(x,y;f)=(0.5,1;2)$。

2)problem 2

$$\begin{aligned}&\min f(x_1,y)=-y+2x_1-\ln(x_1/2)\\ &\text{s.t.}\quad -x_1-\ln(x_1/2)+y\leqslant 0\\ &0.5\leqslant x_1\leqslant 1.4,y\in\{0,1\}\end{aligned} \tag{12}$$

已知最优解为$(x_1,y;f)=(1.375,1;2.124)$。

3)problem 3

$$\begin{aligned}&\min f(x,y)=-0.7y+5(x_1-0.5)^2+0.8\\ &\text{s.t.}\quad -\exp(x_1-0.2)-x_2\leqslant 0\\ &x_2+1.1y\leqslant -1,x_1-1.2y-0.2\leqslant 0\\ &0.2\leqslant x_1\leqslant 1,-2.22554\leqslant x_2\leqslant -1,y\in\{0,1\}\end{aligned} \tag{13}$$

已知全局最优解是$(x_1,x_2,y;f)=(0.94194,-2.1,1;1.07654)$

4)problem 4

$$\begin{aligned}&\min f(y_1,v_1,v_2)=7.5y_1+5.5(1-y_1)+7v_1+6v_2+\\ &50\frac{y_1}{0.9[1-\exp(-0.5v_1)]}\\ &\text{s.t.}\quad 0.9[1-\exp(-0.5v_1)]-2y_1\leqslant 0\\ &0.8[1-\exp(-0.4v_2)]-2(1-y_1)\leqslant 0\\ &v_1\leqslant 10y_1,v_2\leqslant 10(1-y_1),v_1,v_2\geqslant 0,y_1\in\{0,1\}\end{aligned} \tag{14}$$

5)problem 5

$$\begin{aligned}&\min f(x_1,x_2,x_3,y_1,y_2,y_3,y_4)=(y_1-1)^2+(y_2-1)^2+\\ &(y_3-1)^2-\ln(y_4+1)+(x_1-1)^2+(x_2-2)^2+(x_3-3)^2\\ &\text{s.t.}\quad y_1+y_2+y_3+x_1+x_2+x_3\leqslant 5\\ &y_3^2+x_1^2+x_2^2+x_3^2\leqslant 5.5\\ &y_1+x_1\leqslant 1.2,y_2+x_2\leqslant 1.8,y_3+x_3\leqslant 2.5,y_4+x_1\leqslant 1.2\\ &y_2^2+x_2^2\leqslant 1.64,y_3^2+x_3^2\leqslant 4.25,y_2^2+x_3^2\leqslant 4.64\\ &x_1,x_2,x_3\geqslant 0,y_1,y_2,y_3,y_4\in\{0,1\}\end{aligned} \tag{15}$$

已知最优解是$(x_1,x_2,x_3,y_1,y_2,y_3,y_4;f)=(0.2,1.280624,1.954483,1,0,0,1;3.557463)$。

6)problem 6

$$\begin{aligned}&\max f(x_1,x_2,x_3,y_1,y_2)=-5.357854x_1^2-0.835689y_1x_3-\\ &37.29329y_1+40792.141\\ &\text{s.t.}\quad a_1+a_2y_2x_3+a_3y_1x_2-a_4x_1x_3\leqslant 92\\ &a_5+a_6y_2x_3+a_7y_1y_2+a_8x_1^2-90\leqslant 20\\ &a_9+a_{10}x_1x_3+a_{11}y_1x_1+a_{12}x_1x_2-20\leqslant 5\\ &27\leqslant x_1,x_2,x_3\leqslant 45,y_1\in\{78,\cdots,102\},y_2\in\{33,\cdots,45\}\end{aligned} \tag{16}$$

其中:a_1a_{12}如表 1 所示。已知最优解为 $x_2,y_2,(x_1,x_3,y_1;f)=(27,27,78;32217.4)$。

表 1　problem 6 中的数据

a_1	85.334 407	a_5	80.512 49	a_9	9.300 961
a_2	0.005 685 8	a_6	0.007 131 7	a_{10}	0.004 702 6
a_3	0.000 626 2	a_7	0.002 995 5	a_{11}	0.001 254 7
a_4	0.002 205 3	a_8	0.002 181 3	a_{12}	0.001 908 5

实验中各个问题求解时,改进差分进化算法的系数设置如表 2 所示。实验时,每个问题独立求解 10 次,本文算法的求解结果及与遗传算法、演化策略、模拟退火以及混合进化算法 HEA 求解结果的对比如表 3 和 4 所示。从表 3 的求解结果可以看出,对于 problem 1、problem 2 和 problem 3,本文算法均能够以最少的目标函数评价次数获得 100% 的成功率;对于 problem 4、problem 5 和 problem 6,本文算法均可以获得 100% 的成功率。从表 4 的求解结果可以看出,对于 problem 4 和 problem 5,本文算法的成功率要优于混合进化 HEA 算法。对于其他问题,两者的成功率相同。

表 3 和 4 的求解结果表明本文方法求解 MINP 是有效的。在目标函数评价次数和成功率两个指标方面,本文算法具有明

显的竞争优势。

表 2　求解六个问题的参数

问题	参数			
	F	CR	NP	最大迭代次数
problem 1	0.3 + 0.7 × rand	rand	10	200
problem 2	0.3 + 0.7 × rand	rand	40	50
problem 3	0.3 + 0.7 × rand	rand	50	300
problem 4	0.3 + 0.7 × rand	rand	30	500
problem 5	0.3 + 0.7 × rand	rand	70	500
problem 6	0.3 + 0.7 × rand	rand	50	700

表 3　本文方法与其他方法[2]求解结果对比

问题	IDE		$GA-R=10^3$		GA-Deb sch		$(\mu+\lambda-ES)$		M-SIMPSA		M-SIMPSA-pen	
	#F	C%	#F	C%	#F	C%	#F	C%	#F	C%	#F	C%
problem 1	468	100	6787	100	6191	100	1518	100	607	99	16282	100
problem 2	1876	100	13939	100	15298	100	2255	100	10582	83	14440	100
problem 3	3160	100	107046	90	110233	90	1749	100	b	0	38042	100
problem 4	15000	100	22489	100	23730	80	b	0	14738	100	42295	100
problem 5	14560	100	102778	60	34410	90	6710	100	22309	60a	63751	97a
problem 6	7230	100	37167	100	35255	100	2536	100	27410	87	33956	95

注：a. 收敛到了非最优解；b. 停止运行

表 4　本文方法与 HEA[4]求解结果对比

问题	IDE		HEA	
	#F	C%	#F	C%
problem 1	468	100	780	100
problem 2	1 876	100	1 612	100
problem 3	3 160	100	5 250	100
problem 4	15 000	100	8 175	90
problem 5	14 560	100	9 008	80
problem 6	7 230	100	2 580	100

表 5 给出了与改进 DE 算法（MDE）[5]实验结果比较（文献[5]中仅利用六个问题中的三个问题作为求解对象，MDE 的实验数据直接引自文献[5]）。从表 5 可以看出，对于 problem1，本文方法在最优值、平均值与 MDE 算法相同，而本文算法的标准差要比 MDE 算法稍逊。对于 problem5，本文算法在最优值、平均值和标准差三个方面均优于 MDE 算法。对于 problem6，本文算法的最优值、平均值与 MDE 算法相同，而标准差小于 MDE 算法。因此，对于上述三个问题，本文算法总体上优于 MDE 算法。

表 5　与文献[5]MDE 求解结果对比

问题	方法	最优值	平均值	标准差
problem1	IDE	2	2	2.96e−006
	MDE	2	2	0
problem5	IDE	3.557 461 6	3.557 461 9	1.1143e−007
	MDE	3.557 463	3.557 463	2.8282e−5
problem6	IDE	3.22174e+004	3.22174e+004	0.0102
	MDE	3.22174e+004	3.22174e+004	0.1999

此外，需要进一步指出的是，problem5 的已知最优解为 3.557 463，此时解变量的值为（0.2，1.280 624，1.954 483，1，0，0，1）。

对于此已知最优解，九个约束条件的值分别为：[−0.564893，0.00000126665，0，−0.51936，−0.545517，0，−0.000002170624，−0.429996202711，−0.819996202711]。从最优解的约束条件取值可以知道，该最优解违反了约束条件 b），因此，此解仅是一个近似可行解，而非真正的最优解。针对 problem 5，本文算法求出的最优解为 3.557 461 675 521 48，此时各个变量的值为（0.19999985760090，1.28062467412759，1.95448213382243，1，0，0，1）。此时九个约束条件的值分别为：[−0.5649，−0.0000，−0.000，−0.5194，−0.5455，−0.0000，−0.0000，−0.43，−0.82]。即对于 problem 5，本文算法发现了新的最优解。图 27 分别给出了 problem 16 的一次求解收敛图。

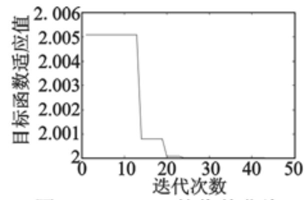

图2　problem 1的收敛曲线

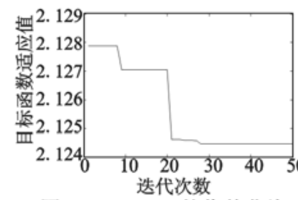

图3　problem 2的收敛曲线

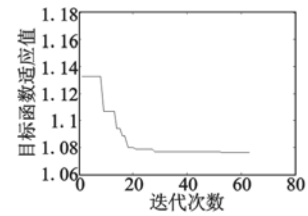

图4　problem 3的收敛曲线

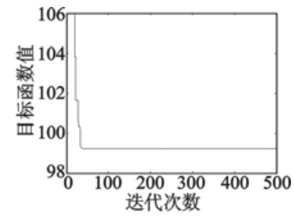

图5　problem 4的收敛曲线

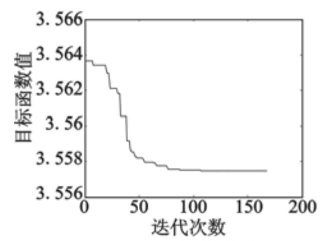

图6　problem 5的收敛曲线

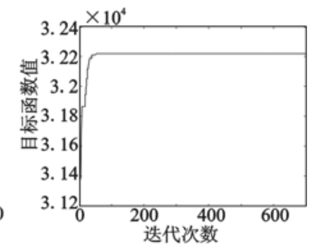

图7　problem 6的收敛曲线

4　结束语

针对 MINP 这类复杂的优化问题，本文提出一种改进的差分进化求解方法。定义一种同态映射算子，有效处理 MINP 中的整数决策变量；定义一种自适应变异算子，提高算法的整体性能；利用一种自适应的不可行解驱动搜索求解机制，自动处理约束条件。本文方法能够成功求解六个常用的标准 MINP 问题，对于其中的第 5 个算例，本文方法求出了一个新的最优解，表明了本文方法的有效性，适于求解工程实践中的 MINP。

参考文献：

[1] COSTA L，OLIVEIRA P. Evolutionary algorithms approach to the solution of mixed integer nonlinear programming problems[J]. Computers and Chemical Engineering，2001，25(1)：257-266.

[2] 贺毅君，陈德钊. 适于混合整数非线性规划的混合粒子群优化算法[J]. 浙江大学学报，2008，42(5)：747-751.

[3] 张兰，邢志栋. 基于量子粒子群求解混合整数非线性规划[J]. 计算机工程与应用，2010，46(9)：49-50，82.

[4] 李宏，焦永昌，张莉. 一种求解混合整数规划的混合进化算法[J]. 控制与决策，2008，23(10)：1098-1102.

[5] 吴亮红，王耀南，陈正龙. 求解混合整数非线性规划问题的改进差分进化算法[J]. 小型微型计算机系统，2007，28(4)：666-669.

[6] 刘俊梅，高岳林. 非线性混合整数规划问题的改进差分进化算法[J]. 工程数学学报，2010，27(6)：967-974.

[7] 谭瑛，高慧敏，曾建潮. 求解整数规划问题的微粒群算法[J]. 系统工程理论与实践，2005，5(1)：126-129.

[8] 邓长寿，梁昌勇. 求解函数优化的新型差异演化算法[J]. 计算机应用研究，2009，26(6)：2047-2049.

[9] DEB K. An efficient constraint handling method for genetic algorithms[J]. Computation Methods in Applied Mechanics and Engineering，2000，86(24)：311-338.