求解多目标优化问题的自适应混沌混合蛙跳算法

田 祎
(商洛学院 陕西 商洛 726000)

摘 要 针对多目标优化问题提出一种自适应混沌混合蛙跳算法 MACSFLA(Adaptive chaos shuffled frog leaping algorithm for multiobjective optimization)。使用动态权重因子策略以提高混合蛙跳算法 SFLA(Shuffled Frog Leaping Algorithm)收敛效率,引入基于 Pareto 支配能力的 SFLA 子族群划分策略,使得 SFLA 能够应用于多目标优化问题。在此基础上,MACSFLA 首先利用 SFLA 快速寻优能力接近理论 Pareto 最优解,然后采用自适应网格密度机制动态维护外部存储器 Pareto 最优解规模,并使用自适应混沌优化技术改善 Pareto 最优解集样本多样性,最后利用 Pareto 最优解选择策略为青蛙种群选择最优更新粒子。多目标函数测试实验结果表明,与 MOPSO 和 NSGA-Ⅱ相比,MACSFLA 在 Pareto 最优解集均匀性和多样性上有明显优势。

关键词 多目标优化 混合蛙跳算法 Pareto 前端 混沌优化

中图分类号 TP18 **文献标识码** A DOI:10.3969/j.issn.1000-386x.2015.06.062

ADAPTIVE CHAOS SHUFFLED FROG LEAPING ALGORITHM FOR MULTI-OBJECTIVE OPTIMISATION SOLUTION

Tian Yi
(*Shangluo University*, *Shangluo 726000*, *Shaanxi*, *China*)

Abstract We propose an adaptive chaos shuffled frog leaping algorithm (MACSFLA) for multi-objective optimisation problem. It uses dynamic weighting factor strategy to improve the convergence efficiency of shuffled frog leaping algorithm (SFLA), and introduces Pareto control capability-based SFLA sub-ethnic partition strategy to make SFLA be able to apply to multi-objective optimisation. On this basis, MACSFLA first employs fast search ability of SFLA to approach the optimal solutions of theoretical Pareto, and then uses adaptive grid density mechanism to dynamically maintain the scale of optimal Pareto solution of external memoriser, and uses adaptive chaos optimisation technology to improve the sample diversity of optimal Pareto solution. Finally, it uses optimal Pareto solution selection strategy to select more update particles for frog populations. Results of multi-objective function test experiment show that, compared with MOPSO and NSGA-Ⅱ, MACSFLA has evident advantages in uniformity and diversity of optimal Pareto solution set.

Keywords Multi-objective optimisation Shuffled frog leaping algorithm Pareto-optimal front Chaos optimisation

0 引 言

多目标优化问题 MOP(multiobjective optimization problem)是指需要同时优化两个或多个互相冲突目标的优化问题[1]。MOP 往往不存在一个解使得所有目标同时取得最优,只能找出一组解使得目标函数尽可能达到 Pareto 前端[2],因此,寻找能够通过一次计算就可以得到均匀分布在 Pareto 前端最优解集的多目标优化算法是学者们研究的热点之一[3]。

传统的多目标优化算法将多目标优化问题通过加权方式转化为单目标问题,在每次求解过程中,只能得到一个解,算法效率较低,而且对权重值和目标给定次序比较敏感[4]。智能优化算法通过种群代与代之间维持潜在解组成的集合实现多目标优化,近十几年来,相继提出了不同的多目标优化算法,如早期的矢量评价遗传算法(VEGA)、非劣排序遗传算法(NSGA),随后出现的基于精英保留机制的 PAES、SPEA2 及经典的 NSGA-Ⅱ[5]等算法。将新的机制和策略引入多目标优化算法并对其改进是当前多目标优化的前沿领域,文献[6]利用多目标优化问题 PS 分布规则,提出了一种分布多目标优化问题估计算法 RM-MEDA;文献[7]针对多目标粒子群算法(MOPSO)[8]在优化 Pareto 前段分段不连续函数时存在收敛效率低、多样性差的缺陷,提出了一种基于杂草克隆的多目标粒子群算法;文献[9]通过引入克隆选择机制,提出了一种基于非支配邻域的改进多目标优化算法 NNIA-Ⅱ,实验结果证明了 NNIA-Ⅱ具有优秀的求解性能。

混合蛙跳算法 SFLA[10]是一种新型的群智能启发式计算技术,由于 SFLA 具有实现容易,收敛速度快等特点,被广泛使用[11,12],然而将 SFLA 应用于多目标优化问题的研究很少。SFLA 较强的寻优能力使其能够快速接近理论 Pareto 最优解,但是,对于多目标优化问题,SFLA 存在易于陷入局部最优及需要根据 Pareto 支配进行多目标函数适应度赋值等问题,针对这些

收稿日期:2013-11-26。陕西省教育厅科研项目(2013JK1160);商洛学院科研项目(12SKY007);商洛学院教改项目(12JYJX209)。田祎,讲师,主研领域:智能优化算法,数据挖掘。

问题,本文提出了一种自适应混沌混合蛙跳算法求解多目标优化问题 MACSFLA(Adaptive Chaos Shuffled Frog Leaping Algorithm for multiobjective optimization),并做了如下改进:

1)针对 SFLA 容易陷入局部极值的问题,使用动态权重因子策略以增强算法跳出局部最优能力和提高算法收敛效率;

2)引入基于 Pareto 支配能力的 SFLA 子族群划分策略,使得 SFLA 能够应用于多目标优化问题;

3)为提高 MACSFLA 算法的收敛效率,采用自适应网格密度机制动态维护外部存储器 Pareto 最优解规模,并使用自适应混沌优化技术改善 Pareto 最优解集样本多样性;

4)利用 Pareto 最优解选择策略为青蛙种群选择最优更新粒子。

通过对多目标函数实验测试,结果表明,与 MOPSO 和 NSGA-Ⅱ相比,MACSFLA 在 Pareto 最优解集均匀性和多样性上有明显优势。

1　多目标优化问题和混合蛙跳算法

1.1　多目标优化问题数学描述和基本概念

多目标优化是对多个目标函数同时进行优化,是优化问题主要的研究领域之一,不失一般性,一个决策变量维数为 n,目标变量数目为 m 的多目标优化问题可以表述为:

$$\begin{cases} \min y = f(x) = [f_1(x), f_2(x), \cdots, f_M(x)] \\ \text{s.t.} \quad g_i(x) \leqslant 0 \quad i = 1,2,\cdots,p \\ \qquad h_i(x) = 0 \quad i = 1,2,\cdots,q \end{cases} \tag{1}$$

其中,$x = (x_1, x_2, \cdots, x_n)$ 为 n 维决策变量,$y = (y_1, y_2, \cdots, y_m)$ 为目标函数,$g_i(x)$、$h_i(x)$ 分别为不等式约束和等式约束。

定义 1　Pareto 支配　设有 $f: R^n \to R^m$,对于给定决策变量 x_1 和 x_2($x_1, x_2 \in R^n$),则称与 x_2 相比,x_1 支配 x_2,当且仅当:

$$\forall i \in \{1,2,\cdots,m\} \quad f_i(x_1) \leqslant f_i(x_2) \wedge \exists \forall i \in \{1,2,\cdots,m\} \quad f_i(x_1) < f_i(x_2) \tag{2}$$

记为 $x_1 \succ x_2$。

定义 2　Pareto 最优解与 Pareto 最优解集　决策变量 x^* 称为 Pareto 最优解,当且仅当满足条件:

$$\neg \exists x: x \succ x^* \tag{3}$$

所有 Pareto 最优解的集合就是 Pareto 最优解集,并记为:

$$P^* \triangleq \{x^* \mid \neg \exists x: x \succ x^*\} \tag{4}$$

定义 3　Pareto 前端　Pareto 最优解集 P^* 中的解对应的 $y = (y_1, y_2, \cdots, y_m)$ 组成的集合成为 Pareto 前端,并记为:

$$PF^* \triangleq \{F(x^*) = (f_1(x^*), \cdots, f_m(x^*)) \mid x^* \in P^*\} \tag{5}$$

1.2　混合蛙跳算法(SFLA)

基本 SFLA 工作原理可以描述为:青蛙种群 F 包含 N 只青蛙,每只青蛙代表 n 维优化问题的一个解,设第 i 只青蛙的位置为 $X_i = \{x_{i1}, x_{i2}, \cdots, x_{in}\}$。青蛙种群将青蛙按照适应度值[13] $f(X_k)$ 降序排列,并平均分配到 M 个子族群 E_l($1 \leqslant l \leqslant M$)中,子族群青蛙数为 Q,其中,E_l 按照式(6)进行划分。

$$E_l = \{X_{l+M(k-1)} \in F \mid 1 \leqslant k \leqslant N\} \quad 1 \leqslant l \leqslant M \tag{6}$$

SFLA 对子族群 E_l 中适应度最差的个体 $X_{l,w}$ 进行更新:

$$X_{l,w}(t+1) = X_{l,w}(t) + r \cdot [X_{l,b}(t) - X_{l,w}(t)] \tag{7}$$

其中,$r \in (0,1)$ 为随机数,$X_{l,b}$ 为 E_l 适应度最好的个体。如果 $f[X_{l,w}(t+1)]$ 优于 $f[X_{l,w}(t)]$,则用新个体替代原个体,否则用种群内适应度最好的解 X_g 代替 $X_{l,b}$ 重新执行更新策略式(7),如果仍不能优于原个体,则随机生成青蛙取代原个体。所有子族群执行一定次数的局部迭代搜索后,青蛙种群重新混合,然后划分新的子族群,并重新进行局部搜索,算法如此往复,直到满足终止条件。

研究表明,SFLA 存在易于陷入局部极值的缺陷,特别是在算法迭代后期,当某个青蛙十分接近局部极值时,算法需要迭代多次才有可能跳出。为了避免 SFLA 早熟,对算法更新策略进行改进,引入动态权重因子策略:

$$r' = \xi_1 \times r_1 + \xi_2 \times rand(0,1) \tag{8}$$

$$r_1 = r_{\min} \times (r_{\max}/r_{\min})^{1/(1+c_1 \times t/T_{\max})} \tag{9}$$

其中,ξ_1、ξ_2、c_1 为比例系数,$r_{\min}$、$r_{\max}$ 为权重因子最小值与最大值,$T_{\max}$ 为算法最大迭代次数。从式(8)和式(9)可以看出,动态权重因子策略实质为在非线性指数函数递减权重因子的基础上增加了随机扰动,采用该权重因子策略使得算法在运算初期,r' 取值较大,青蛙能在比较大的空间进行搜索;算法运算后期,随机扰动让算法能够进一步深度搜索,提高了算法收敛精度。

对于多目标优化问题,SFLA 无法直接根据适应度值进行子族群划分,因此,本文提出基于 Pareto 支配能力的 SFLA 子族群划分策略:青蛙种群完成一次迭代后,将所有青蛙进行 Pareto 支配关系比较,对于青蛙 X_i,由其支配的青蛙组成的集合为 Z_i,即:

$$Z_i = \{X_j \mid X_i \succ X_j, X_j \in F\} \tag{10}$$

设 $|Z_i|$ 为 Z_i 内青蛙个数,种群根据 $|Z_i|$ 将青蛙降序排序,如果两只青蛙具有相同的 $|Z_i|$,则随机进行排列。当所有青蛙完成排序后,根据式(6)进行子族群划分。在子族群 E_l 内,对具有最小 $|Z_i|$ 的青蛙按照式(7)进行更新,如果子族群多只青蛙具有相同的 $|Z_i|$,则随机选择一个个体进行更新。如果更新后的个体支配原个体,则用新个体替代原个体,否则随机产生新的个体进行替代。

2　自适应混沌混合蛙跳算法

2.1　自适应网格密度机制

本文将 Pareto 最优解存储器记为 Archive 集,MACSFLA 每次进化会产生新的 Pareto 最优解粒子,称这些粒子为 Archive 集候选解,当候选解满足如下条件之一时就可以进入 Archive 集。

1. Archive 集为空集。

2. Archive 集为非空集,且候选解不受 Archive 集中任一个粒子支配。

3. 候选解支配 Archive 集所有粒子。

4. 候选解不受 Archive 集中至少一个粒子支配,且候选解所在网格的密度值比该非支配解所在网格的要小。

随着算法迭代不断增加,Archive 集规模不断增加,其计算复杂度为 $O(t \cdot m \cdot N^2)$,如果不控制 Archive 集规模,将很大程度地增加计算复杂度,因此,本文规定 Archive 集规模为 N,并采用自适应网格密度机制动态维护 Archive 集规模。自适应网格密度机制工作原理可以描述为:

Step1　自适应网格划分。将 m 维目标空间划分为 $K_1 \times K_2 \times \cdots \times K_m$ 相同的网格,网格 i 维宽度 d_i 为:

$$d_i = (f_i^{\max} - f_i^{\min})/K_i \tag{11}$$

$$f_i^{\max} = \max_{i=1}^{m} f_i(x) \quad f_i^{\min} = \min_{i=1}^{m} f_i(x) \tag{12}$$

Step2 Archive 集粒子网格定位。设第 t 代 Archive 集集合为 $A_t = \{a_1, \cdots, a_j, \cdots, a_h\}$ $(h \leqslant N)$，对于 a_j，其第 i 维网格编号 w_i^j 为：

$$w_i^j = \text{int}[(f_i(a_j) - f_i^{\min})/d_i] + 1 \tag{13}$$

由式(13)可以确定 a_j 在目标空间内的位置 $a_j(w_1^j, \cdots, w_m^j)$。

Step3 Archive 集网格粒子密度计算。设网格 G_k 的粒子密度为 D_k，并定义 D_k 为网格内 Pareto 最优解的个数，如果网格内没有粒子，其 D_k 为 0。

Step4 Archive 集粒子淘汰。当候选解加入 Archive 集后，可能会使得 Archive 集规模大于 N，因此需要进行 Archive 集粒子淘汰：对于网格 G_k，如果其 $D_k > 1$，则根据式(14)对其中粒子进行删除。

$$G_d = \text{int}(D_k \cdot (|A_{t+1}| - N)/|A_{t+1}| + 0.5) \tag{14}$$

其中，G_d 为网格 G_k 需要删除的粒子数，$|A_{t+1}|$ 为 Archive 集加入候选解后的规模。确定 G_d 后，根据式(15)计算 G_k 中粒子与理论 Pareto 前端距离，并依次删除远离 Pareto 前端的粒子。

$$D_{G_k,i} = \| P_{G_k,i} - P_{T,j} \| \tag{15}$$

其中，$D_{G_k,i}$ 为 G_k 内粒子 i 与理论 Pareto 前端最短距离。

2.2 自适应混沌优化技术

Archive 集中的个体决定了青蛙种群的进化方向，自适应网格密度机制在一定程度上保证了 Archive 集中粒子的均匀分布，但是，如果某些理论 Pareto 最优解没有存储在 Archive 集中，那么之后会很难再找到该 Pareto 最优解[4]。为了改善 Archive 集样本多样性，引入自适应混沌优化技术，其思路可以描述为：随机选择 Archive 集部分粒子进行混沌优化(限于篇幅，混沌优化过程不再赘述)，设粒子 i 在混沌优化第 $k-1$ 代、k 代和 $k+1$ 代的位置分别为 $C_{k-1,i}$、$C_{k,i}$ 和 $C_{k+1,i}$，$P_{k-1,i}$、$P_{k,i}$ 和 $P_{k+1,i}$ 分别为 $C_{k-1,i}$、$C_{k,i}$ 和 $C_{k+1,i}$ 的 Pareto 前端，则第 $k-1$ 代、k 代和 $k+1$ 代之间 Pareto 前端距离 $D_{k,k-1}$、$D_{k,k+1}$ 分别为：

$$D_{k,k-1} = \| P_{k,i} - P_{k-1,i} \| \quad D_{k,k+1} = \| P_{k+1,i} - P_{k,i} \| \tag{16}$$

定义混沌迭代控制系数 $diff_D$ 为 $D_{k,k-1}$、$D_{k,k+1}$ 的差值，即：

$$diff_D = D_{k,k+1} - D_{k,k-1} \tag{17}$$

当 $diff_D > 0$ 表示算法处于进化状态，可以减少混沌优化迭代次数，以提高算法速度，当 $diff_D < 0$ 表示算法进化能力降低，需要增加混沌优化迭代次数，以提高算法精度，混沌优化迭代次数 $K_{\max}^C$ 自适应策略为：

$$K_{\max}^C(k+1) = \begin{cases} K_{\max}^C(k)/2 & diff_D > 0 \\ K_{\max}^C(k) & diff_D = 0 \\ 2 \times K_{\max}^C(k) & diff_D < 0 \end{cases} \tag{18}$$

自适应混沌优化技术在每次混沌优化后，自适应调整 $K_{\max}^C$，直到新解优于原解或达到设定的最大迭代次数为止。

2.3 Pareto 最优解选择策略

MACSFLA 根据网格粒子密度值，在 Archive 集中为每只青蛙选择一个 Pareto 最优解进行更新。对于网格 G_k，如果 $D_k > 0$，则其网格适应度值 $f(G_k)$ 可以定义为 D_k 的倒数，即：

$$f(G_k) = 1/D_k \tag{19}$$

MACSFLA 根据网格 $f(G_k)$，采用轮盘赌的方式为每只青蛙选择一个网格，并从网格中随机选择一个个体作为最优解进行更新。该 Pareto 最优解选择策略使得 Archive 集中 D_k 越小的网格被选择的概率越大，确保了 Pareto 最优解多样性。

2.4 自适应混沌混合蛙跳算法实现流程

MACSFLA 算法流程可以描述为：

//MACSFLA 初始化

1. 初始化 MACSFLA 相关参数，设置全局最大进化代数 $T_{\max}$。

2. 对青蛙个体进行初始化，计算多目标函数值 $\{f_1(X_j), \cdots, f_m(X_j)\}$，$t \leftarrow 0$。

//Archive 集初始化

3. 根据 Pareto 支配关系对 Archive 集进行初始化。

While ($t \leqslant T_{\max}$) do

{

4. 将目标空间进行网格划分，确定 Archive 集粒子网格位置，并计算每个网格粒子密度。

// Pareto 最优解选择

5. 根据 Pareto 最优解选择策略为青蛙种群每只青蛙选择一个全局最优解。

//子族群划分

6. 根据多目标 SFLA 子族群划分策略对种群进行子族群划分。

//局部搜索

While ($k \leqslant K_{\max}$) do

{

For $l = 1:Q$

7. 对子族群 E_l 内最小 $|Z_i|$ 个体进行更新。

End

8. $k \leftarrow k + 1$

}

// Pareto 最优解集更新

9. 所有青蛙重新混合，并根据 Pareto 支配关系更新种群 Pareto 最优解集，$t \leftarrow t + 1$

//Archive 集规模调整

10. 根据 Archive 集加入条件确定能够加入 Archive 集的 Pareto 最优解。

While ($|A_t| > N$) do

{

11. 确定网格需要淘汰的粒子数。

12. 从粒子密度最大的网格开始，根据式(15)进行淘汰。

13. $|A_t| \leftarrow |A_t| - 1$

14. }

//自适应混沌优化

15. 采用自适应混沌优化技术随机选择 Archive 集部分粒子进行混沌优化。

}

16. 程序结束，Archive 集即为所求的 Pareto 最优解并输出。

3 性能验证

3.1 评价指标和测试函数

本文采用文献[14]中的世代距离指标 GD 和多样性指标 Δ 进行算法性能评价，并分别采用文献[1]和文献[4]中的 KUR、ZDT1、ZDT2、ZDT3 和 ZDT6 多目标函数进行实验仿真，其中，KUR 函数的 $n = 3$，决策变量取值范围为 $(-5, 5)$，其他 4 个函数的 $n = 30$，决策变量取值范围为 $[0, 1]$。

MACSFLA 参数设置如下：$N = 200$，$Q = 20$，$K_{\max} = 10$，$T_{\max} = 500$，$\xi_1 = 0.2$，$\xi_2 = 0.8$，$c_1 = 3$，$r_{\min} = 0.35$，$r_{\max} = 0.95$。

3.2　实验结果及其分析

分别采用 MACSFLA 算法、NSGA-Ⅱ算法和 MOPSO 算法对测试函数进行实验,每种算法独立测试 30 次,表 1 给出了世代距离指标 *GD* 对比结果,表 2 给出了多样性指标 Δ 对比结果(分别取最大值 *M* 和平均值 *AV*),图 1 给出了通过 MACSFLA 优化后测试函数 Pareto 前端和理论 Pareto 前端仿真图。

表 1　世代距离指标 *GD* 对比

算法		KUR	ZDT1	ZDT2	ZDT3	ZDT6
MACSFLA	*M*	0.0021	1.0045e-4	1.1146e-3	0.0522	0.0003
	AV	0.0011	2.3472e-5	1.0078e-4	0.0442	0.0002
NSGA-Ⅱ	*M*	0.0217	0.0238	0.0143	0.0913	0.0414
	AV	0.0118	0.0176	0.0098	0.0551	0.0311
MOPSO	*M*	0.0613	0.1322	0.1215	0.1713	0.1512
	AV	0.0426	0.0887	0.0990	0.07886	0.0987

表 2　多样性指标 Δ 对比

算法		KUR	ZDT1	ZDT2	ZDT3	ZDT6
MACSFLA	*M*	0.4227	0.4081	0.4113	0.7742	0.5326
	AV	0.2278	0.3357	0.3971	0.6654	0.5021
NSGA-Ⅱ	*M*	0.81115	0.9231	0.7523	1.1072	0.7019
	AV	0.7710	0.8123	0.6741	1.0034	0.6217
MOPSO	*M*	0.9211	0.8861	0.8235	1.1819	0.8347
	AV	0.9017	0.8071	0.7719	1.0021	0.7789

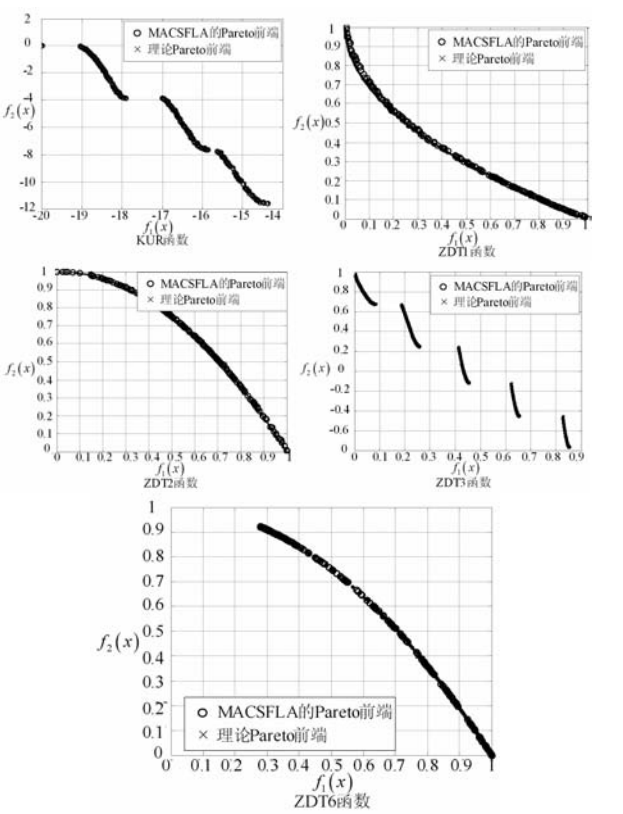

图 1　MACSFLA 算法获得的 Pareto 最优解

从表 1,表 2 和图 1 可以看出,MACSFLA 能够有效地获得 Pareto 前端, MACSFLA 得到的 Pareto 前端更加均匀,世代距离指标 *GD* 和多样性指标 Δ 要优于其他两种算法,而且对于 Pareto 前端两端端点的逼近程度很高,特别是对于不连续多目标函数 ZDT3,MACSFLA 得到的 Pareto 前端粒子均匀分布在理论 Pareto 前端上。

5　结　语

提出了一种新的基于自适应混沌混合蛙跳算法的多目标函数优化算法。对混合蛙跳算法权重因子进行了改进,设计了基于 Pareto 支配能力的 SFLA 子族群划分策略,提出了自适应网格密度机制动和自适应混沌优化技术,最终实现了多目标优化问题求解。仿真结果表明,与 MOPSO 和 NSGA-Ⅱ相比,MACSFLA 在 Pareto 最优解集均匀性和多样性上有明显优势。

参　考　文　献

[1] 戚玉涛,刘芳,常伟远,等. 求解多目标问题的 Memetic 免疫优化算法[J]. 软件学报,2013,24(7):1529-1544.

[2] 聂瑞,章卫国,李广文,等. 一种自适应混合多目标粒子群优化算法[J]. 西北工业大学学报,2011,29(5):695-701.

[3] 陈民铀,张聪誉,罗辞勇. 自适应进化多目标粒子群优化算法[J]. 控制与决策,2009,24(12):1851-1855,1864.

[4] 公茂果,焦李成,杨咚咚,等. 进化多目标优化算法研究[J]. 软件学报,2009,20(2):271-289.

[5] Deb K. Multi-Objective Optimization Using Evolutionary Algorithms [M]. New York:Wiley,2001.

[6] Zhang Q F,Zhou A M,Jin Y C. RM-MEDA:A regularity model-based multiobjective estimation of distribution algorithm [J]. IEEE Trans. on Evolutionary Computation,2008,12(1):41-63.

[7] 鲁鹏,章卫国,李广文,等. 一种基于杂草克隆的多目标粒子群算法[J]. 西北工业大学学报,2012,30(2):286-290.

[8] Coello C C A,Pulido G T,Lechuga M S. Handing multiple objectives with particle swarm optimization [J]. IEEE Trans. On Evolutionary Computations,2004,8(3):256-279.

[9] Yang D D,Jiao L C,Gong M G,et al. Adaptive ranks and K-nearest neighbor list based multiobjective immune algorithm [J]. Computational Intelligence,2010,26(4):359-385.

[10] Eusuff M M,Lansey K E. Optimization of water distribution network design using the shuffled frog leaping algorithm [J]. Water Resources Planning and Management,2003,129(3):210-225.

[11] 骆剑平,李霞,陈泯融. 混合蛙跳算法的 Markov 模型及其收敛性分析[J]. 电子学报,2010,38(12):2875-2880.

[12] 罗雪晖,杨烨,李霞. 改进混合蛙跳算法求解旅行商问题[J]. 通信学报,2009,20(7):130-135.

[13] 葛宇,王学平,梁静. 自适应混沌变异蛙跳算法[J]. 计算机应用研究,2011,28(3):945-947.

[14] 江铭炎,袁东风. 人工鱼群算法及其应用[M]. 北京:科学出版社,2012.