Journal of System Simulation

Volume 27 | Issue 4 Article 9

8-20-2020

Multi-objective Cuckoo Search Algorithm

Xingshi He
1. School of Science, Xi'an Polytechnic University, Xi'an 710048, China;;

Li Na
1. School of Science, Xi'an Polytechnic University, Xi'an 710048, China;;

Xinshe Yang
1. School of Science, Xi'an Polytechnic University, Xi'an 710048, China;;2. School of Science & Technology, Middlesex University, London NW4 4BT, UK;;

Yu Bing
1. School of Science, Xi'an Polytechnic University, Xi'an 710048, China;;3. Shanghai Baosight Software Corporation, Shanghai 201300, China;

Multi-objective Cuckoo Search Algorithm

Abstract

Abstract: It is challenging to solve multi-objective optimization problems with getting high-quality Pareto fronts accurately. The multi-objective Cuckoo Search algorithm (MOCS) was designed by firstly applying the recently developed Cuckoo Search Algorithm (CS) in solving Multi-objective optimization problems, and the fitness function based Pareto definiteness was improved, and the Gradual archive reduction method based on niche technology was proposed to improve the Archive solutions quality. The simulation test results and related performance indicators of nine test problems show that, MOCS algorithm is obviously improved in the aspect of the convergence, the diversity and the uniformity compared with the classic NSGA-II algorithm.

Keywords

multi-objective algorithms, cuckoo search algorithm, gradual archive reduction method based on niche technology, multi-objective cuckoo search algorithm, pareto optimal solutions

Recommended Citation

He Xingshi, Li Na, Yang Xinshe, Yu Bing. Multi-objective Cuckoo Search Algorithm[J]. Journal of System Simulation, 2015, 27(4): 731-737.

第 27 卷第 4 期 系统仿真学报© Vol. 27 No. 4
2015 年 4 月 Journal of System Simulation Apr., 2015

多目标布谷鸟搜索算法

贺兴时[1]，李娜[1]，杨新社[1,2]，余兵[1,3]
(1. 西安工程大学理学院，西安 710048；2. 密德萨斯大学科学与技术学院，英国伦敦 NW4 4BT；
3. 上海宝信软件股份有限公司，上海 201300)

摘要：解决多目标优化问题，并得到精确的、高质量的Pareto前沿解是非常具有挑战性的。*将CS算法运用于多目标问题解的迭代更新过程，对传统的基于Pareto支配关系的适应度函数进行了改进，并提出基于小生境技术的逐步档案缩减法用于档案解的缩减与维护过程,设计出了多目标布谷鸟搜索算法(MOCS)*。通过仿真实验验证以及相关性能指标的测试结果得出，MOCS算法与经典的NSGAII算法相比，在所得解的收敛性、多样性和均匀性方面均有所改善。

关键词：多目标算法; 布谷鸟搜索算法; 多目标布谷鸟搜索算法; 基于小生境技术的逐步档案缩减法; Pareto最优解

中图分类号：TP18　　文献标识码：A　　文章编号：1004-731X (2015) 04-0731-07

Multi-objective Cuckoo Search Algorithm

He Xingshi[1], *Li Na*[1], *Yang Xinshe*[1,2], *Yu Bing*[1,3]
(1. School of Science, Xi'an Polytechnic University, Xi'an 710048, China; 2. School of Science & Technology, Middlesex University, London NW4 4BT, UK; 3. Shanghai Baosight Software Corporation, Shanghai 201300, China)

Abstract: It is challenging to solve multi-objective optimization problems with getting high-quality Pareto fronts accurately. *The multi-objective Cuckoo Search algorithm (MOCS) was designed by firstly applying the recently developed Cuckoo Search Algorithm (CS) in solving Multi-objective optimization problems, and the fitness function based Pareto definiteness was improved, and the Gradual archive reduction method based on niche technology was proposed to improve the Archive solutions quality*. The simulation test results and related performance indicators of nine test problems show that, MOCS algorithm is obviously improved in the aspect of the convergence, the diversity and the uniformity compared with the classic NSGA-II algorithm.

Keywords: multi-objective algorithms; cuckoo search algorithm; gradual archive reduction method based on niche technology; multi-objective cuckoo search algorithm; pareto optimal solutions

引言

多目标优化问题(MOP: Multi-objective Optimization Problem)一直以来都是一个备受关注的研究课题，其求解大致基于2种思路：一种是将各目标进行线性聚合，转换成单目标问题来求解；另一种是基于Pareto最优解的方法。第1种思路即对各目标按其相对重要性进行加权后组合成单目标问题来求解[1-3]，但多目标问题并不存在类似于单目标那样的绝对最优解，同一问题用不同优化方法所得结果可能大不相同。因此，多目标优化问题的求解更倾向于获得一组不存在优劣关系的最优解，再人为地进行选择，这样的解便是Pareto最优解集。基于Pareto最优解的多目标优化问题很早就得到重视，目前已经发展了较多方法[4-5]。在20世纪80年代中期，作为关键智能技术之一的进化算法理论在

收稿日期：2014-02-24　修回日期：2014-06-12;
基金项目：陕西省软科学基金项目(2012KRM58)；陕西省教育厅自然科学基金项目(11JK0188);
作者简介：贺兴时(1960-)，男，陕西，硕士，教授，研究方向为智能算法；李娜(1989-)，女，陕西，硕士，研究方向为智能算法。

该领域开始得到应用，并逐渐形成众多的多目标进化算法(MOEAs)[6]。进化多目标优化算法一般要达到以下3个目标：

(1) 进化解的非支配前沿与Pareto前沿距离最短(收敛性好)；

(2) 进化解的前沿分布性能好(一般是均匀分布，均匀性好)；

(3) 所得的非支配前沿范围大，即非支配解的值覆盖了每个目标尽可能广的范围(多样性好)。

纵观国内外该领域的研究情况，粒子群算法、蚁群算法、人工萤火虫算法等智能算法陆续被用于求解多目标优化问题，并取得了不错的效果。2009年，剑桥大学的Yang Xin-She等人模拟布谷鸟寻窝产蛋行为，提出一种新型的启发式算法——布谷鸟搜索算法(CS)，该算法具有所用参数少、搜索路径优、寻优能力强等特点，其收敛性也得到了证明，目前在工程优化等问题中已得到成功应用[7-9]。本文力求找到更为均匀且收敛更快的Pareto解集，提出1种基于CS的多目标布谷鸟搜索算法(MOCS)。

1 多目标优化算法基本概念

1.1 多目标优化问题的一般描述

多目标问题一般是指两个或者两个以上的目标，并且这些目标往往是相互冲突的。不失一般化，我们考虑极小化的多目标问题，其一般描述如下：

$$\begin{aligned} &min\ f=(f_1(x),f_2(x),\cdots,f_K(x)) \\ &s.t.\ g_i(x)\leqslant 0, i=1,2,\cdots,I \\ &\quad h_j(x)=0, j=1,2,\cdots,J \end{aligned} \tag{1}$$

上式中，$f_k(x)$ 为第 k $(k=1,2,\cdots,K)$ 个目标函数，目标总数为K。I和J分别为不等式约束、等式约束的个数。

1.2 Pareto支配关系

定义1：对于极小化多目标问题的两个解向量 X_1 和 X_2，若满足以下条件则称 X_1 支配 X_2，记为 $X_1 \prec X_2$：

(1) 对 $\forall k\in\{1,2,\cdots,\mathrm{K}\}$ 都有 $f_k(X_1)\leq f_k(X_2)$；

(2) 至少存在一个 $k\in\{1,2,\cdots,\mathrm{K}\}$ 使 $f_k(X_1)<f_k(X_2)$。

如果解空间中不存在支配X_1的解，则称解X_1是非支配的(或称为非劣的)，这样的解也叫做Pareto最优解。

1.3 Pareto最优解集与Pareto前沿

定义2：所有Pareto最优解构成的集合称为多目标优化问题的Pareto最优解集。

Pareto最优解集中的每个解之间互不支配，没有优劣关系。在目标空间，所有Pareto最优解的目标向量构成了多目标问题的Pareto前沿，又称为非支配前沿。

2 MOCS算法的设计

2.1 适应度函数

多目标优化算法的设计中，适应度的选择是尤为关键的一步。在MOCS算法中，对基于Pareto支配关系的适应度计算方法进行了改进。改进后的适应度函数计算主要分2部分进行：第1部分是基于Pareto非支配分层的适应度计算，第2部分是对第一部分得到的非支配解(即第1层的解)进行适应度的增强。

(1) 基于Pareto非支配分层的适应度[10]

Pareto非支配分层的具体实施过程如下：首先，确定解集 $X=[\mathrm{X}_1,\mathrm{X}_2,\cdots\mathrm{X}_n]$ 中所有解的支配关系，找出所有非支配的解并标记支配层数 $s=1$；再将这些非支配解从解集X中删除，确定出剩下解的支配关系，找出所有非支配的解并标记支配层数为 $s=2$；如此下去，直到解集X中的所有解都被标记了支配层数。这样，适应度的计算公式如下：

$$\mathrm{fitness}(X_i)=\frac{1}{s_i} \tag{2}$$

(2) 非支配解的适应度增强

为了提高算法所得解的均匀性与多样性，需要

控制解在目标空间的分布，使之不至于拥挤。因此，我们使用了基于共享机制的小生境技术，对非支配解中分布较稀疏的解的适应度进行了增强。所谓小生境技术是指当两个个体的距离小于预先给定的某个值时，对其中适应度值较小的个体进行惩罚。基于共享机制的小生境技术[11]则是通过共享函数，对群体中聚成小块的个体进行惩罚，再通过小生境数来衡量个体的指定范围内的拥挤程度。小生境数越大，解的周围越拥挤，相应的适应度就应越小。因此，增强后的适应度函数设计如下：

$$\text{fitness}(X_i)=\frac{1}{s_i}+I(s_i)\times\frac{1}{\text{NicheCount}_i} \tag{3}$$

其中$I(s)$为示性函数，用来表示只对支配层数为1的解，即只对非支配解进行适应度增强：

$$I(s)=\begin{cases}1, & s=1\\ 0, & s\neq 1\end{cases} \tag{4}$$

NicheCount_i为解Xi的小生境数：

$$\text{NicheCount}_i=\sum_{j=1}^{n}s(d_{ij}) \tag{5}$$

其中d_{ij}为第i个解与第j个解之间的距离，本文采用的是欧氏距离。s(d)为共享函数，σ_{share}为小生境半径(阈值)：

$$s(\text{d})=\begin{cases}1-\dfrac{d}{\sigma_{\text{share}}}, & d\leqslant\sigma_{\text{share}}\\ 0, & d>\sigma_{\text{share}}\end{cases} \tag{6}$$

2.2 档案的生成与维护

2.2.1 档案的生成与预处理

为了避免增加计算的复杂度，我们在算法的执行初期，将每次迭代所得解直接归档。为了进一步避免内存的占用，归档以后的解需要进行预处理。预处理过程分两步进行：

(1) 去除档案中重复的解，避免冗余；

(2) 找出档案中解的支配关系，去除被支配的解，只保留非支配的解。

因此，经预处理后的档案解是互不相同的，且互不支配的解。

2.2.2 基于小生境技术的逐步档案缩减法

如果档案中解的个数超过了档案的最大容量ArchiveMax，则需要对档案进行缩减，以删除密集的解，提高算法所得解的均匀性与多样性。档案的缩减有很多方法，例如小生境策略、拥挤距离策略、k-临近策略和网格策略等[12]。通过实验发现，以上方法所得解的均匀性仍不是很好。因此，本文在小生境策略的基础上，提出了基于小生境技术的逐步档案缩减法，并取得了满意的效果。

基于小生境技术的逐步档案缩减法是在小生境技术的基础上，逐步删除每个基点的小生境半径σ_{share}以内的所有点。该方法的前提是得到了充足的档案解，且这些档案解都是非支配的、充分收敛的解。对于这样的档案，我们就可以运用基于小生境技术的逐步档案缩减法对其进行逐步缩减。以双目标问题为例，档案缩减的步骤如下：

(1) 计算档案解Archive的目标函数值

$F=[f_1(\text{Archive}),f_2(\text{Archive})]^T$，并按目标空间某一维（如$f_1$或者$f_2$）值的从小到大顺序对$F$进行排序。这样一来，矩阵$F$的相邻两列便是目标空间上的相邻两点，以便于后面的判断。令$i=1,n=\text{size}(F,2)$，即n为F的列数，也是对应解的个数;

(2) 选定第i个点为基点；

(3) 分别计算i与$i+1,i+2$之间的距离，记为$d1,d2$(显然$d1<d2$)，并对$d1,d2$进行判断：

① 若$d1>\sigma_{\text{share}}$，则以$i+1$为新的基点，即令$i=i+1$，转(3)；

② 若$d2<\sigma_{\text{share}}$，则在Archive和F中删除$i+1,i+2$，更新Archive和F以及个数n，转(2)；

③ 若$d1<\sigma_{\text{share}}<d2$，则在Archive和$F$中删除$i+1$，以$i+2$为新的基点，更新Archive和$F$以及个数$n$，令$i=i+1$，转(3)。

直到所有的点都被判断。

这样，在缩减后的档案中，每个解的小生境半径以内除自身以外就不会有其他的解，并且随着迭代的进行，受支配关系的压迫，这些基点会逐步向

其前一个基点的小生境边缘与Pareto前沿的交点位置逼近，最终收敛到Pareto前沿并以 σ_{share} 为间隔均匀排列。通过调整 σ_{share} 的值，便可控制最终所得解的数量。

这种方法也可以扩展到更多目标的问题。对于 M 个目标的问题，只需要将档案中的解按目标空间的前 $M-1$ 维依次进行排序，再对每个基点的超球体逐步判断并删减即可。因此，基于小生境技术的逐步档案缩减法对高维问题也是可行的。

3 MOCS算法的步骤

(1) 初始化：给定算法所需的参数：解的维数 m，鸟窝的个数 n，被发现的概率 Pa，最大迭代次数 N，搜索域上下界 ub 和 lb，初始档案Archive为空集（$Archive=[\,]$），档案的最大容量ArchiveMax，小生境半径 σ_{share}，迭代次数 $t=0$. 随机生成 $m\times n$ 的初始鸟窝位置矩阵 $X0^{(t)}$；

(2) 若 $t<=N$，则转(3)，否则，停止计算，输出Archive；

(3) 位置更新：通过levy飞行原则实现鸟窝位置的更新，得到新的鸟窝位置矩阵 $X1^{(t)}$：

$$X1_i^{(t)}=X0_i^{(t)}+\alpha\oplus Levy(\lambda),i=1,2,...,n \quad (7)$$

其中 $\alpha>0$ 表示步长，$\oplus$ 表示点对点乘法；

(4) 适应度择优：合并 $X0^{(t)}$ 与 $X1^{(t)}$，得到 $m\times 2n$ 的矩阵 X'，按式(3)计算 X' 的适应度，并从大到小进行排列，选出前 n 个对应的解，组成新的解矩阵，记为 $X2^{(t)}$；

(5) 随机淘汰：对 $X2^{(t)}$ 中每个解 $X2_i^{(t)}$ 赋予一个随机的数 ε_i，并根据淘汰概率 Pa 进行解的随机淘汰得到 $X3^{(t)}$：

$$X3_i^{(t)}=\begin{cases}X2_i^{(t)} & ,\ \text{if } \varepsilon_i<Pa\\ \text{lb+(ub-lb)*rand(m,1)} & ,\ \text{else}\end{cases} \quad (8)$$

(6) 适应度择优：合并 $X2^{(t)}$ 与 $X3^{(t)}$，得到 $m\times 2n$ 的 X''，按式(3)计算 X'' 的适应度，并从大到小排列，选出前 n 个对应的解，组成新的解矩阵，记为 $X4^{(t)}$；

(7) 档案更新及预处理：将 $X4^{(t)}$ 直接并入档案Archive，即Archive=[Archive, $X4^{(t)}$]，对Archive进行去重复处理，并按Pareto支配关系，删除被支配的解，只保留非支配的解；

(8) 判断：若Archive中解的个数大于ArchiveMax (溢出)，转(9)，否则转(10)；

(9) 档案缩减：按照基于小生境技术的逐步档案缩减法对Archive进行缩减，直到档案不再溢出，再转(10)；

(10) 令 $X0^{(t)}=X4^{(t)}$，$t=t+1$，转(2)。

在位置更新过程中采用的是levy飞行机制，等式(7)对随机行走这类随机问题是必不可少的。通常，随机行走问题是一种Markov链，即物体下一时刻的状态或位置只与当前所处状态位置(上述等式的第一部分)和转移概率(第二部分)有关，与以前状态无关。编程过程中[13]，一般令

$$\beta=\frac{3}{2},\ \sigma=\left[\frac{\Gamma(1+\beta)\times sin(\frac{\beta\pi}{2})}{\Gamma(\frac{1+\beta}{2})\times\beta\times 2^{\frac{\beta-1}{2}}}\right]^{\frac{1}{\beta}} \quad (9)$$

其中伽马函数 $\Gamma(a)=\int_0^\infty e^{-t}t^{a-1}dt$。

再令 $u=rand(m,1)\times\sigma, v=rand(m,1)$，则步长 $\alpha=0.01\times(\frac{u}{|v|})^{\frac{1}{\beta}}$，因此，Levy飞行机制下的解的迭代格式即为：

$$X1_i^{(t)}=X0_i^{(t)}+\alpha\times rand(m,1) \quad (10)$$

可以看出，MOCS算法利用CS算法实现解的更新过程，levy飞行机制使得算法对搜索域的探索能力增强，提高了搜索效率。每次生成新解后会进行一次适应度择优过程，在基于非支配分层与适应度增强机制的适应度函数下，算法被迫向非支配的、更加稀疏多样的解收敛，这也就保证了算法的收敛性与多样性。而随机淘汰机制又使得算法可以跳出局部最优，向全局最优收敛。在档案的缩减过程中，采用了基于小生境技术的逐步档案缩减法，该方法易于实现，复杂度低且能得到更均匀的Pareto最优解集。因此，MOCS算法在收敛性以及所得解的均匀性和多样性方面表现都很突出。

4 仿真试验及算法性能评价

为了验证MOCS算法的有效性，我们对算法进行了仿真试验。采用Kalyanmoy Deb等人在文献[14]中给出的FON，POL，KUR，ZDT3和ZDT6共5个测试问题进行了数值实验，并采用文献[15]所提供的3种评价指标对MOCS算法进行评价，并与NSGAII算法做了对比。这3种指标描述如下：

(1) 收敛性指标 γ：算法搜索到的解逼近Pareto最优前沿的程度

确定方法：在真实Pareto前沿上均匀地取n个点，算出算法所得解与这些点的距离最小值，再求平均。

(2) 多样性指标 η：算法所得解的散布范围

确定方法：

$$\eta = \frac{\sum_{k=1}^{M}(f_k(z_k^1) - f_k(z_k^2))/(f_k^{\max} - f_k^{\min})}{M} \quad (11)$$

其中M为目标函数的个数，z_k^1和z_k^2分别为第k个目标在该算法下求得的目标值最大、最小的解，$f_k^{\max}$与$f_k^{\min}$分别为第k个目标在Pareto前沿的最大、最小值。

(3) 均匀性指标 σ_d：解在目标空间分布的均匀程度

确定方法：计算每个解与其他解的最小规范欧氏距离 D_{NE}，这些距离的方差就是 σ_d，其中，解a与b的规范欧氏距离 D_{NE} 为：

$$D_{NE}(a,b) = \sqrt{\sum_{k=1}^{M}\left(\frac{f_k(a) - f_k(b)}{f_k^{\max} - f_k^{\min}}\right)} \quad (12)$$

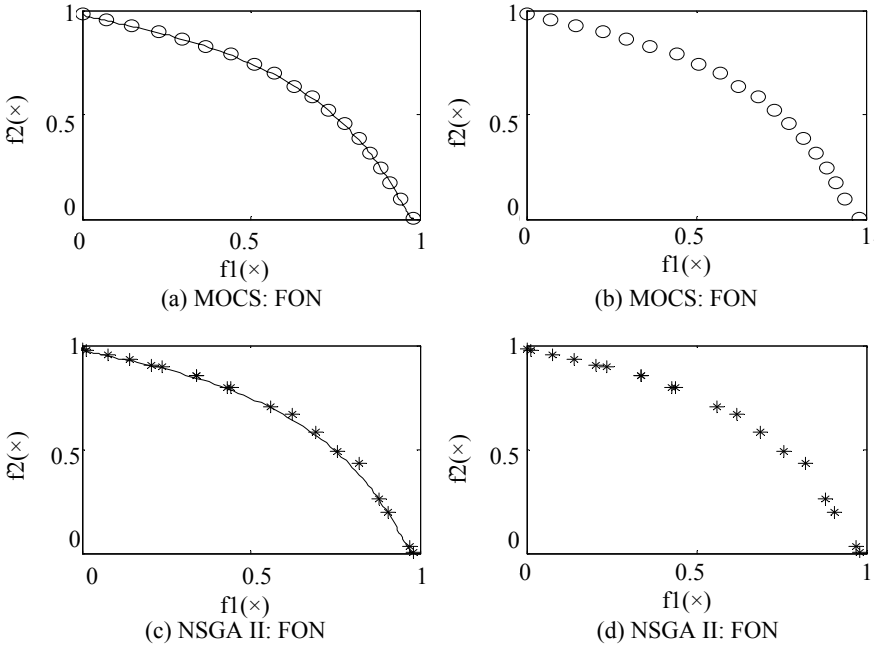
显然，γ 和 σ_d 越接近于0越好，而 η 越接近于1越好。

试验参数设置如表1所示。

表1 MOCS算法参数设置

	FON	POL	KUR	ZDT3	ZDT6
N	400	300	400	400	400
ArchiveMax	400	120	120	100	100
σ_{share}	0.075	0.5	0.5	0.1	0.05

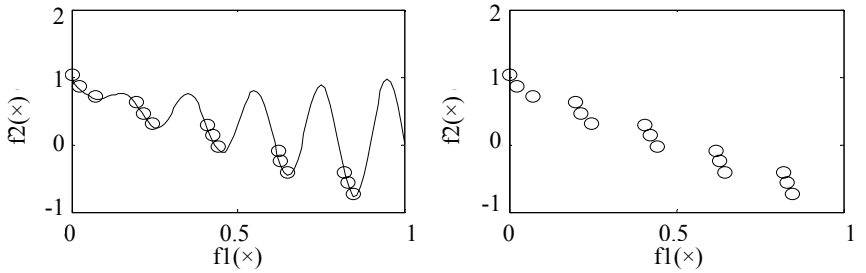
在表1所示的参数设置下，MOCS算法与NSGAII算法的运行结果如图1～5所示。



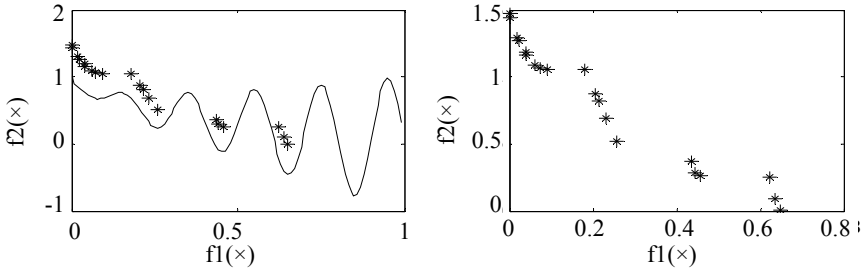
(a) MOCS: FON (b) MOCS: FON

(c) NSGA II: FON (d) NSGA II: FON

图1 MOCS算法与NSGAII算法对FON问题的结果对比图

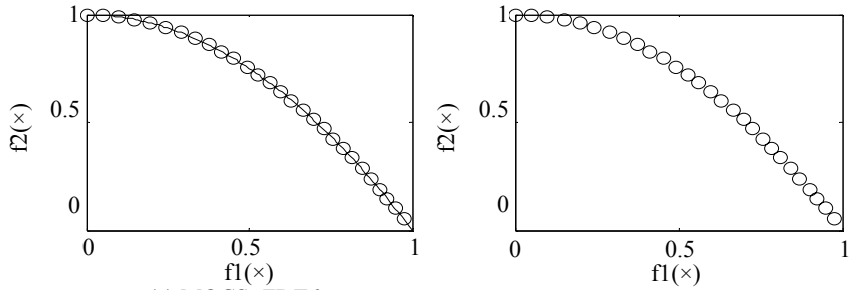


(a) MOCS: ZDT3 (b) MOCS: ZDT3

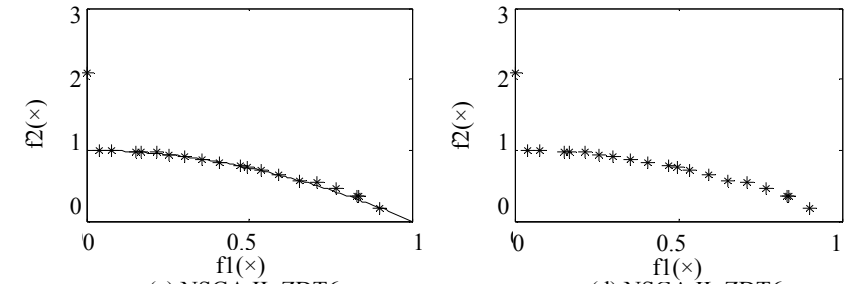


(c) NSGA II: ZDT3 (d) NSGA II: ZDT3

图2 MOCS算法与NSGAII算法对ZDT3问题的结果对比图

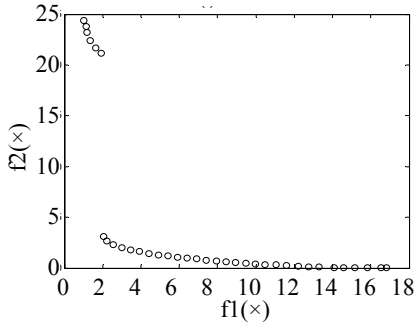


(a) MOCS: ZDT6 (b) MOCS: ZDT6



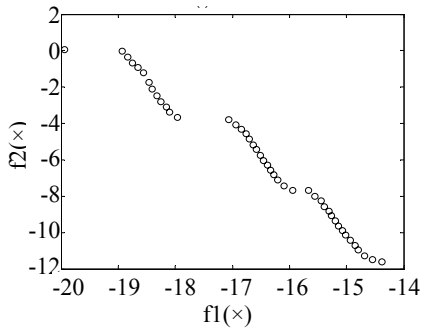
(c) NSGA II: ZDT6 (d) NSGA II: ZDT6

图3 MOCS算法与NSGAII算法对ZDT6问题的结果对比图



(a) MOCS: POL

图4 MOCS算法对POL问题的结果图



(b) MOCS: KUR

图5 MOCS算法对KUR问题的结果图

MOCS算法与NSGAII算法的性能指标计算结果如表2所示。

表2 性能评价指标

		FON	ZDT3	ZDT6
γ	NSGAII	0.007 4	0.050 0	0.063 0
(收敛性)	MOCS	0.006 3	0.023 6	0.003 9
η	NSGAII	0.999 6	0.845 8	1.403 4
(多样性)	MOCS	1.000 0	0.998 4	0.999 1
σ_d	NSGAII	0.079 1	0.057 7	0.202 1
(均匀性)	MOCS	0.012 0	0.018 8	0.011 9

图1~3中的图(a)(c)反应了算法搜得解与真实Pareto前沿的位置关系，即反映了算法的收敛性；图(b)(d)反映了算法搜得解的多样性与均匀性；图(a)(b)为MOCS算法的结果，图(c)(d)为NSGAII算法的结果。图4和图5也清晰反映出MOCS算法对POL和KUR问题也能找到均匀、收敛的Pareto前沿。通过图1~图5可以看出，MOCS算法在收敛性、多样性和均匀性方面都优于NSGAII算法，且均匀性的改善尤为突出。对于ZDT3这样的高维问题，MOCS算法也得到了不错的效果，说明MOCS算法对于处理高维问题仍然很好。通过性能指标的计算结果也能看出，MOCS算法在收敛性、多样性和均匀性方面较NSGAII算法都有所提高，尤其是均匀性方面的改善最为明显。

5 结论

本文提出了一种新的解决多目标问题的进化算法——MOCS算法。MOCS算法充分利用CS算法可并行计算、收敛性能好等优势，将CS算法运用于解的更新过程，为算法提供多样的解，并通过Pareto支配关系迫使算法迅速收敛到Pareto前沿。运用Pareto非支配分层关系设计适应度函数，并用小生境技术与共享函数原则对非支配层解的适应度进行增强，使搜索过程中得到的解足够分散。又利用外部档案存储优质的解，并用基于小生境技术的逐步档案缩减法，对档案进行缩减，最终得到均匀的Pareto前沿解。通过仿真试验以及相关性能评价指标测试，MOCS算法与经典的NSGAII算法相比，在收敛性、多样性和均匀性方面都有所提高，取得了不错的效果。

参考文献：

[1] 崔逊学, 林闯, 方廷健. 多目标进化算法的研究与进展 [J]. 模式识别与人工智能, 2003, 16(3): 306-314.

[2] 赵亮, 雎刚, 吕剑虹. 一种改进的遗传多目标优化算法及其应用 [J]. 中国电机工程学报, 2008, 28(2): 96-102.

[3] Fonseca C M, Fleming P J. An Overview of Evolutionary Algorithms in Multiobjective Optimization [J]. Evolutionary Computation (S1063-6560), 1995, 3(1): 1-16.

[4] K Deb. Multi-objective Optimization Using Evolutionary Algorithms [M]. Chichester, UK: Wiley, 2001.

[5] Tan K C, Khor E F, Lee T H. Multi-objective Evolutionary Algorithms and Applications [M]. London, UK: Springer, 2005.

[6] Coello C A C, Veldhuizen D A V, Lamont G B. Evolutionary Algorithms for Solving Multi-objective Problems [M]. New York, USA: Kluwer Academic Publishers, 2002.

[7] Yang X S, Deb S. Cuckoo search via Levy flights [C]// World Congress on Nature & Biologically Inspired Computing, India. USA: IEEE Publications, 2009: 210-214.

[8] 王凡, 贺兴时, 王燕, 杨松铭. 基于CS算法的Markov模型及收敛性分析 [J]. 计算机工程, 2012, 38(11): 180-182, 185.

[9] Yang X S, Deb S. Engineering optimization by cuckoo search [J]. Mathematical Modelling and Numerical Optimisation (S2040-3607), 2010, 1(4): 330-343.

[10] 曾劲涛, 李金忠, 唐卫东, 等. 多目标微粒群优化算法及其应用研究进展 [J]. 计算机应用研究, 2011, 28(4): 1225-1231.

[11] 王晓鹏. 多目标优化设计中的Pareto遗传算法 [J]. 系统工程与电子技术, 2003, 25(12): 1558-1561.

[12] 徐鹤鸣. 多目标粒子群优化算法的研究 [D]. 上海交通大学, 2013.

[13] Yang, X S, 2008. Nature-Inspired Meta-heuristic Algorithms [M]. UK: Luniver Press, 2008.

[14] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multi-objective genetic algorithm: NSGA-II [J]. IEEE Trans. on Evolutionary Computation (S1089-778X), 2002, 6(2): 182-197.

[15] 雷德明, 吴智铭. Pareto档案多目标粒子群优化 [J]. 模式识别与人工智能, 2006, 19(4): 475-480.