文章编号: 1001-0920(2017)08-1386-09 DOI: 10.13195/j.kzyjc.2016.0631

一种基于群体分布特征的自适应多目标粒子群优化算法

耿焕同†, 陈 哲, 陈正鹏, 薛 羽

(南京信息工程大学 计算机与软件学院，南京 210044)

摘 要: 针对多目标优化问题求解,提出基于群体分布特征的多目标自适应粒子群优化算法(pdMOPSO). 首先借助统计方法分析归档集在决策空间的分布特征,以此划分进化状态,指导全局引导粒子的选择;然后设计粒子重排策略,动态调控种群的分布;最后依据进化状态设计不同的归档集维护策略,实现归档集中分布性和收敛性的均衡. 以ZDT、DTLZ和CEC09为测试集,与7种多目标优化算法对比,指标IGD、Spread和ER结果表明,所提出的算法在收敛性和分布性上均有显著优势.

关键词: 粒子群；多目标优化；群体分布特征；多样性保持；自适应

中图分类号: TP18 **文献标志码:** A

A self-adaptive multi-objective particle swarm optimization algorithm based on swarm distribution characteristic

CENG Huan-tong†, CHEN Zhe, CHEN Zheng-peng, XUE Yu

(School of Computer and Software，Nanjing University of Information Science & Technology，Nanjing 210044，China)

Abstract: A self-adaptive multi-objective particle swarm optimization algorithm based on the swarm distribution characteristic is proposed. Firstly, the interquartile range(IQR) is applied to analyze the distribution features of archive in the decision space and classify the state of evolution, for guiding global leader selecting. Then, a swarm variation operator is proposed to rearrange particles dynamically. Finally, according to the evolution state, an archive maintaining strategy is designed to achieve a balance between distribution and convergence. The proposed algorithm is compared with 7 state-of-the-art multi-objective optimization algorithms on ZDT, DTLZ and CEC09 Benchmarks. The indicators of IGD, Spread and ER show that the proposed algorithm has certain advantages over other algorithms in terms of convergence and distribution.

Keywords: particle swarm；multi-objective optimization；swarm distribution information；maintaining diversity；self-adaptive

0 引 言

在现实世界中,许多复杂优化问题需同时优化彼此冲突的多个目标,这类优化统称为多目标优化问题(MOPs),其求解方法一直是学术界和工程界关注的焦点, 基于种群的进化算法因对Pareto最优解集的凹凸性、可微性和连续性不敏感,现已成为求解MOPs的有效方法之一, 粒子群优化算法[1]是由Kennedy等在1995年提出的仿生算法,是具有形式简洁、收敛快速和参数少等优点的进化算法,已成功应用于单目标优化问题的求解中, 自2002年Coello等[2]首次提出多目标粒子群优化算法(MOPSO)以来,MOPSO求解方法正受到国内外学者的广泛关注,并已被证明是一种极具潜力的MOPs求解方法.

国内外对于MOPSO已取得许多标志性研究成果:在归档集维护策略方面,如平行网格策略[3]和R2策略[4],通过不同方法评估归档集中粒子优劣程度,从而使Pareto最优前沿具有更好的分布性能[5];在全局引导粒子选择策略方面,基于动态加权法[6]、基于偏好信息[7]等策略的MOPSO算法相继提出,通过选择合适的全局最优粒子,提高进化种群的多样性或收敛性;在平衡算法探索与探究策略方面,多种群[8]、信息交互[9]等策略被先后引入MOPSO算法,虽然角度

收稿日期: 2016-05-20；**修回日期:** 2016-10-26.

基金项目: 国家自然科学基金项目(61403206)；江苏省自然科学基金项目(BK20151458)；"青蓝工程"资助项目(2016).

作者简介: 耿焕同(1973−),男,教授,博士生导师,从事计算智能与约束多目标优化、气象资料预处理与资料同化等研究；陈哲(1991−),男,硕士生,从事多目标决策、动态多目标优化的研究.

†通讯作者. E-mail: htgeng@nuist.edu.cn

不同,但都尽可能地兼顾到探索和探究过程.

尽管MOPSO算法研究与应用已取得可喜进展,但鉴于MOPs的归档集特征和MOPSO算法的快速收敛特征,该类算法仍面临诸多挑战[3]: 首先,当前MOPSO算法缺少对种群进化状态的动态监测机制,易造成进化过程中种群多样性与局部收敛性的不平衡;其次,MOPSO快速收敛的特征导致种群易陷入局部最优是算法无法避免的问题;最后,算法收敛后归档集规模迅速扩大,如何控制归档集规模成为亟待解决的问题之一. 因此,为更好地运用MOPSO求解MOPs,本文针对第1类挑战,利用四分位差统计量能客观反映数据在空间中离散程度的原理,对归档集在决策空间上的四分位差及中位数统计量的变化情况进行分析,以估算当前优化所处的进化状态,进而确定选择侧重收敛性还是多样性的归档集维护策略和全局最优粒子选取策略;针对第2类挑战,设计基于四分位差的粒子重排策略,通过计算归档集在决策空间不同维度上的四分位差确定各个维度上的粒子重排的参数,实现粒子群在决策空间上的自适应位置调整;针对第3类挑战,利用自适应网格,分别设计基于邻域个体数的拥挤程度计算、基于角偏移量的拥挤程度计算和基于排序的收敛程度计算方法,并综合3种方法形成多关键字排序策略用以维护归档集. 综合以上分析,将设计并实现一种基于群体分布特征的自适应多目标粒子群优化算法.

1　问题定义与相关分析

1.1　多目标优化问题定义

不失一般性,以最小化为例,多目标优化问题定义如下:

$$\begin{aligned}&\min F(\boldsymbol{x})=\{f_1(\boldsymbol{x}),f_2(\boldsymbol{x}),\cdots,f_M(\boldsymbol{x})\}.\\&\text{s.t. } g_j(\boldsymbol{x})\leqslant 0,\ 0\leqslant j\leqslant p;\\&\quad h_k(\boldsymbol{x})=0,0\leqslant k\leqslant q.\end{aligned}$$

其中: $\boldsymbol{x}$为n维实数空间的向量; $F(\boldsymbol{x})$由M个相互冲突的优化目标组成,当$M\geqslant 2$时,称为多目标优化; $g_j(\boldsymbol{x})$为P个不等式约束; $h_k(\boldsymbol{x})$为q个等式约束.

1.2　粒子群优化算法

在粒子群优化算法中,粒子由位置向量$\boldsymbol{X}_i$和速度向量$\boldsymbol{V}_i$二元组表示,其中

$$\begin{aligned}&\boldsymbol{X}_i=[x_{i,1},x_{i,2},\cdots,x_{i,D}]^{\mathrm{T}}\in R^D,\\&\boldsymbol{V}_i=[v_{i,1},v_{i,2},\cdots,v_{i,D}],\ i=1,2,\cdots,N,\end{aligned}$$

N为粒子个数, D为决策变量个数. 在进化过程中,粒子根据下式更新速度和位置:

$$\begin{aligned}v_i^{t+1}=&\omega\times v_i^t+\theta_1\times r_1\times(\mathrm{pBest}_i^t-x_i^t)+\\&\theta_2\times r_2\times(\mathrm{gBest}_i^t-x_i^t),\end{aligned}\tag{1}$$

$$x_i^{t+1}=x_i^t+v_i^{t+1}.\tag{2}$$

其中: t为当前进化代数, ω为惯性权重系数, θ_1、θ_2为加速度系数, r_1、r_2为0∼1间的随机数, pBest_i^t为第i个粒子的历史最优粒子, gBest_i^t为全局引导粒子.

pBest_i^t和gBest_i^t作为独立项,在粒子速度更新时分别将粒子引向局部最优位置和全局最优位置,因此其对粒子的更新意义重大,而多数算法在选择时单一地依赖于分布性指标或收敛性指标,忽略了对不同进化状态下粒子运动特性的利用,造成了资源浪费.

1.3　种群分布特征的定量分析

在PSO中,粒子的速度位置更新发生在决策空间,映射于目标空间;反之,目标空间上的支配关系和拥挤变化影响粒子在决策空间的运动,因此决策空间是分析粒子特性的根本,尤其是非线性问题,往往决策空间极小的波动都会对目标空间带来巨大影响. 目前多数算法仅从目标空间分析种群,并设计对应策略来提高优化效果. 为更加全面、客观地评价进化种群,本文试图通过四分位差定量测算粒子群在决策空间的分布特征,分析出粒子在各个决策维度上不同的运动趋势,并以此自适应地调整粒子群在决策空间上的分布,从而提高种群在决策空间上的探索广度和目标空间上的分布多样性.

在统计学上,四分位差(iqr)定义为上四分位数x_{Q3}(位于75%)与下四分位数x_{Q1}(位于25%)的差,反映了中间50%数据的离散程度,去除了极值干扰,可有效避免因个别粒子分散导致的上下限区间过大的非聚集假象,其数值大小与离散程度成正比. 通过下式计算四分位差在空间总范围(ub−lb)占比,从而实现数值的标准化:

$$\mathrm{inf}=\mathrm{iqr}/(\mathrm{ub}-\mathrm{lb}).\tag{3}$$

对于由4个点组成的二维数组{(0.48, 0.1), (0.51, 0.2), (0.46, 0.3), (0.52, 0.4)}, x轴值组成的数组从小到大排序获得{0.46, 0.48, 0.51, 0.52},根据定义上四分位数$x_{Q3}=0.51$,下四分位数$x_{Q1}=0.48$,则二维数组在x轴上的四分位差$\mathrm{iqr}_x=0.03$. 同理,在y轴上,上下四分位数x_{Q3}、x_{Q1}分别为0.3、0.2,其四分位差$\mathrm{iqr}_y=0.1$. 如果定义的x、y轴上边界ub和下边界lb分别为1.0和0,则inf_x、inf_y分别为0.03和0.1, inf_x、inf_y的巨大差别间接反映了点在x与y轴分布性的巨大差别. 为了弥补四分位差对数列整体平移不敏感的缺陷,引入中位数差统计量进行辅助判别群体分布

的变化情况,中位数差为

$$\Delta \mathrm{mid}^t = |\mathrm{mid}^t - \mathrm{mid}^{t-1}|, \tag{4}$$

表示数列x在前后两代$t-1$和t时的中位数差值.

2 基于群体分布特征的自适应多目标粒子群优化算法

通过引入粒子群在决策空间的四分位差分布特征信息改进现有MOPSO,提出一种基于种群分布特征的自适应多目标粒子群优化算法(pdMOPSO). 下面介绍其中3种策略,分别为进化状态分类策略、粒子重排策略、归档集维护策略.

2.1 进化状态分类

在优化过程中,更为有效的进化算法应依据种群运动趋势识别不同的进化状态,并选择对应的搜索行为,但目前多数算法未做考虑,因此难以有效平衡算法的收敛性和多样性. 鉴于此,首先在分布特征inf_i基础上,借鉴信息熵理论,采用熵值法[3]转化归档集A的inf_i获得熵值e^t,进而间接刻画归档集A的多样性程度.有

$$e^t = -\sum_{i=1}^{n} \mathrm{inf}_i \times \log \mathrm{inf}_i - \sum_{i=1}^{n} \Delta \mathrm{mid}_i^t \times \log \Delta \mathrm{mid}_i^t. \tag{5}$$

计算相邻两代熵差

$$\Delta e^t = e^t - e^{t-1}. \tag{6}$$

不难发现,若A发生了更新,则e^t将会放大A在决策空间的变化,Δe^t也同样发生变化,Δe^t越大意味着A中越多粒子被替换,可以判定种群正在积极地探索未知潜在解空间;反之,Δe^t越趋于0,A越稳定,此时算法趋向于收敛状态,因此可以利用Δe^t对算法进化状态S进行分类. 其中:t为当前代数,n为决策变量个数,N为种群规模,μ为阈值,$\Delta \mathrm{mid}^t$为熵值化后的中位数差.具体为:

1) 若$|\Delta e^i| \geqslant \mu$,则表明算法处于“探索”状态,此时应加速收敛定位到潜在解空间.

2) 若$|\Delta e^i| < \mu$且$|A| < N$,则表明算法进入“探究”状态,此时应扩大归档集规模,搜索更多的潜在解空间.

3) 若$|\Delta e^i| < \mu$且$|A| = N$,则表明算法趋于“收敛”状态,此时应考虑通过维护归档集来增强多样性,否则易陷入局部最优导致无法发现真实最优前沿.

由上可知,进化状态S的划分主要依赖阈值μ,因此设置μ是关键. 其设计依据为:对于任一维有限区间U,理想最优分布是种群均匀分布在区间中,此时标准化距离为$1/N$.

$$\overline{\mathrm{inf}} = \frac{\frac{3}{4}U - \frac{1}{4}U}{U} = \frac{1}{2}. \tag{7}$$

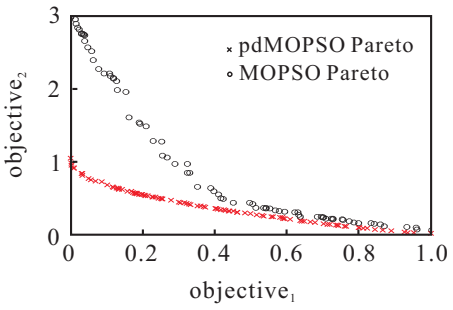
由式(3)和(7)计算个体均匀分布时的标准化四分位差$\overline{\mathrm{inf}}$,选取$\overline{\mathrm{inf}}$的变化小于$1/N$且种群未发生偏移$\Delta \mathrm{mid} \approx 0$时,根据式(6)得到$|\Delta e|$作为阈值$\mu$,并作为参照. 进一步考虑阈值$\mu$与决策空间维数$n$存在着线性关系,最终设计如下式所示的综合决策空间复杂度n、种群规模N和种群分布特征等信息:

$$\mu = -n \times \overline{\mathrm{inf}} \times \log \overline{\mathrm{inf}} + -n \times \left(\overline{\mathrm{inf}} + \frac{1}{N}\right) \times \log\left(\overline{\mathrm{inf}} + \frac{1}{N}\right), \tag{8}$$

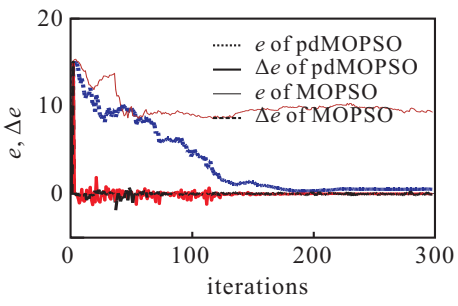
为了更好地阐释熵e和熵差Δe对进化状态的敏感性,以标准测试函数ZDT1为例,分别采用本文算法与agMOPSO[2]进行优化对比,ZDT1的决策空间维数为30,因此有

$$\mu = 30 \times |-0.5 \times \log(0.5) + 0.51 \times \log(0.51)| \approx 0.03.$$

图1为两种算法在ZDT1函数上的e及Δe变化趋势. 由图1可见,在进化初期,由于算法探索导致归档集频繁更新,e和Δe不稳定;在进化后期,算法趋于收敛,导致e和Δe趋向于稳定状态. 反之,每当Δe出现剧烈波动时,反映归档集出现了剧烈的变化,间接说明种群发现新的潜在解区域并进行了探索. 两种算法对比可直观地发现,本文算法在进化前期和后期分别有更好的全局探索能力和局部搜索能力.



(a) ZDT1 test problem



(b) e, Δe change in evolution

图1 两种算法在ZDT1函数上的e及Δe变化趋势

2.2 基于群体分布特征的粒子重排策略

当粒子群处于“收敛”状态时,一般采用扰动方式避免早熟收敛,目前多数算法在扰动时仅考虑进化代数与边界约束,忽略了群体的分布特征和进化状态,既浪费计算资源也不利于种群进行有效探索. 因此,本文将保留非支配解集中的边界解来提高分布广度、剔除拥挤解来提高分布均匀性的归档集维护思想应用于决策空间,结合群体分布特征设计粒子重排策略,更有效地防止陷入局部最优.

对进化种群在决策空间每一维采取如下步骤.

Step 1: 粒子集排序. 对所有粒子的第i维进行排序,获得粒子集的有序序列.

Step 2: 粒子集运动幅度计算. 由$\varepsilon = \mathrm{iqr}_i/\mathrm{iqr}'_i$计算当前粒子重排后的四分位差$\mathrm{iqr}_i$与上一代的四分位差$\mathrm{iqr}'_i$之比,并作为评估进化种群在第$i$维上的运动趋势,用于扰动幅度的修正.

Step 3: 边界解扰动. 对分布在上、下四分位x_{Q1}、x_{Q3}以外的边界解,分别进行$N = (x_{Q_1}, (\varepsilon \times \mathrm{iqr}_i)^2)$和$N = (x_{Q_3}, (\varepsilon \times \mathrm{iqr}_i)^2)$的正态扰动操作.

Step 4: 拥挤解扰动. 对分布在上、下四分位以内的拥挤解,将以$\mathrm{step}_i = (4 \times \mathrm{iqr}_i)/N$为步长进行均匀分布操作.

Step 5: 粒子扰动与归档集更新. 对处于不同分布特征的粒子实施有针对性和差别化的扰动策略,获得扰动后的临时粒子群,并与现有归档集进行合并更新操作,对当前进化种群不作调整.

2.3 归档集维护及全局引导粒子选择策略

2.3.1 归档集维护策略

归档集用于保存进化过程中发现的所有非支配解,受其规模的限制,需设计合理的粒子优劣比较规则,用于粒子排序和归档集的维护. 常见策略[4-5]更多地侧重于多样性保持,忽略了不同进化阶段对收敛性的考虑. 因此,本文设计一种基于不同进化状态的均衡多样性与收敛性的归档集维护策略,即当处于“探索”阶段应保留收敛程度强的粒子;当处于“收敛”状态应删除拥挤程度高的粒子;当处于“探究”状态,由第2.1节可知归档集规模小于预设上限,因此无需进行归档集维护. 归档集维护策略的详细步骤如下.

Step 1: 构建自适应网格. 确定粒子在网格中坐标向量id,有

$$\mathrm{id}_i = \left\lfloor \frac{f_i(x) - \mathrm{lb}_i}{\mathrm{div}_i} \right\rfloor, \tag{9}$$

其中lb_i与div_i分别为网格在第i个目标上的下边界和格距.

Step 2: 计算相邻性. 计算粒子x、y间的格差

$$\mathrm{diff}(x, y) = \sum_{i=1}^{M} (\mathrm{id}_i^x - \mathrm{id}_i^y). \tag{10}$$

当$\mathrm{diff}\,(x, y) < M$时为相邻,$M$为目标个数,有助于灵活调整选择压力.

Step 3: 计算粒子拥挤程度. 计算每个粒子x在其邻域网格内的拥挤程度DI_x,有

$$\mathrm{DI}_x = \sum_{y \in \mathrm{diff}\,(x,y) < M} (M - \mathrm{diff}\,(x, y)). \tag{11}$$

Step 4: 计算粒子收敛程度. 对每个粒子x在网格中的坐标求和,作为收敛程度CI_x,有

$$\mathrm{CI}_x = \sum_{i=1}^{M} \mathrm{id}_i^x. \tag{12}$$

若存在粒子DI、CI与粒子x的CI_x、DI_x相同,则需依据下式计算粒子x所作网格中的偏移值$\mathrm{CI2}_x$,作为辅助的收敛程度判断:

$$\mathrm{CI2}_x = \sqrt{\sum_{i=1}^{M} \left(\frac{f_i(x) - (\mathrm{lb}_i + \mathrm{id}_i) \times \mathrm{div}_i}{\mathrm{div}_i} \right)^2}. \tag{13}$$

Step 5: 归档集排序. 当进化状态为“探索”阶段时,依据各粒子的CI_x、DI_x和$\mathrm{CI2}_x$指标次序对归档集进行多关键字升序排序;当进化状态为“收敛”阶段时,依据各粒子的DI_x、CI_x和$\mathrm{CI2}_x$指标次序对归档集进行多关键字升序排序.

Step 6: 归档集维护. 依次删除排序靠后的粒子,直至满足归档集规模大小.

注意到,在Step1中,格距大小不仅关系到算法的计算效率,更关系到优化效果. 当然,实际计算时仅计算分配有粒子的网格,时间复杂度为线性,对算法的计算效率影响不大;优化效果对格距大小更为敏感,过小易出现每个粒子均有独立网格,过大易出现许多粒子拥挤在同一网格,两者均会导致粒子间的拥挤度难以区分,势必降低选择压力,最终影响到优化效果. 因此,本文在借鉴文献[2]进行格距设定的基础上,又通过$\mathrm{CI2}_x$偏移值进一步降低格距对算法性能的影响.

2.3.2 全局引导粒子选择策略

依据算法所处的进化状态和维护更新后归档集中各粒子的CI_x、DI_x值,确定全局引导粒子的选择策略. 具体如下:当算法处于“探索”和“探究”状态时,从CI_x值最小且DI_x值最小的粒子集中随机挑选粒子作为全局引导粒子;当算法处于“收敛”状态时,随机从DI_x值最小且CI_x值最小的粒子集中挑选. 通过为不同进化状态设定不同的全局引导粒子选择策略,较好地实现了算法搜索过程中的多样性与收敛性

的均衡,且算法实现时采用随机选择方式又进一步提高了算法的鲁棒性.

2.4 pdMOPSO算法流程

Step 1: 初始化种群和归档集规模N,最大迭代次数$G_{\max}$,权重系数ω,θ_1,θ_2,约束边界vmax, vmin,网格数$G_{\rm num}$,阈值μ(由第2.1节根据问题计算获得). 初始化粒子群Pop,归档集Archive,迭代次数器$t=0$.

Step 2: 由第1.4节判断进化状态S^t.

Step 3: 由第2.3.2节选择与进化状态相对应的全局引导粒子gBest_i^t.

Step 4: 由式(1)和(2)更新速度、位置,评估目标值,依据支配关系更新个体最优pBest_i^t.

Step 5: 当粒子群处于“收敛”状态时,对粒子群Pop采取重排获得Pop'.

Step 6: 根据支配关系,以Pop和Pop'更新归档集,并由第2.3.1节对归档集进行维护.

Step 7: 判断是否达到最大评价次数,未达到则转至Step 2,达到则结束,输出归档集.

2.5 算法时间复杂度分析

在pdMOPSO中,设置种群与归档集规模为N,决策空间维度为D,目标数为M,分析时间复杂度. 对于Step 2,计算e^t和Δe^t的总时间复杂度为$O(\text{DNlog}N)$,Step 4更新速度位置为线性操作. 当进化状态符合判定条件时,采取粒子重排策略(Step 5),需对每维进行排序,其时间复杂度为$O(\text{DNlog}N)$. 进入Step 6,归档集合并为新归档集,计算复杂度为$O(\text{MN}^2)$. 当新归档集规模超过上限N时,执行归档集维护策略,构建自适应网格,时间复杂度为$O(\text{MNlog}N)$,分别计算个体CI_x、DI_x和CI2_x指标,需要$O(N^2)$,并对所有指标进行多指标排序,理想计算复杂度为$O(\text{Nlog}N)$. 综上所述,pdMOPSO的每一代循环所需时间复杂度最坏为$O(\text{MN}^2)$,与非支配排序所需时间相同.

3 实验结果与分析

3.1 测试问题与对比算法的选择

为验证所提出基于分布信息的自适应多目标粒子群优化算法的有效性,选取ZDT[10]系列、CEC09测试集[11]中的UF2、UF7函数以及3目标DTLZ测试集[12]中的DTLZ1、DTLZ2函数. 对比算法选择3种多目标粒子群优化算法agMOPSO[2]、SMPSO[13]、OMPSO[14]以及进化算法NSGA-II[15]、MOMBI2[4]、SPEA2[16]、MOEA/D[17]. 为保证公平性,对于所有2目标测试问题,种群和归档集规模均为100, 最高评价次数30 000次;对于3目标测试问题,种群和归档集规模为150,最高评价次数为50 000. 进化算法交叉率为0.9,变异率为决策变量维数的倒数,其中agMOPSO、SPEA2和pdMOPSO的网格栅数为30,MOEA/D的邻域规模为20.

3.2 性能指标

为了评估算法获得解集的收敛性和多样性,采用广泛用于当前多目标进化算法评估的多样性Spread[15]、反转世代距离IGD[17]和错误率ER[15]作为性能评估指标.

1) 多样性Spread性能指标. 通过计算最优解集中每个点与其相邻点距离d_i、平均距离$\overline{d}$和边缘点与真实前沿边缘距离d_f,d_l,衡量解集的分布情况,有

$$\text{Spread}=\frac{d_f+d_l+\sum\limits_{i=1}^{N-1}\left|d_i-\overline{d}\right|}{d_f+d_l+(N-1)\overline{d}},\tag{14}$$

其中n为PF的规模. Spread值越小表明算法获得解的分布性越好.

2) IGD性能指标. 使用一组真实Pareto最优前沿PF*,计算其中每个点到算法获得的解集PF之间的最近距离的平均值. 有

$$\text{IGD}=\Big(\sum_{i=1}^{N^*}d_i\Big)\Big/N^*.\tag{15}$$

其中:N^*为PF*的规模,d_i为第i个真实前沿点到PF的最近欧氏距离. 该指标值越低,表明算法获得的近似Pareto前沿的收敛性和多样性越好.

3) ER性能指标,用以衡量最优解集在真实Pareto前沿上的比例. 有

$$\text{ER}=\sum_{i=1}^{N}e_i/N,\tag{16}$$

ER越小表示最优解集的质量越高,其中$e_i=0$表示最优解集中第i个解在真实Pareto前沿上,反之则$e_i=1$.

3.3 实验结果及分析比较

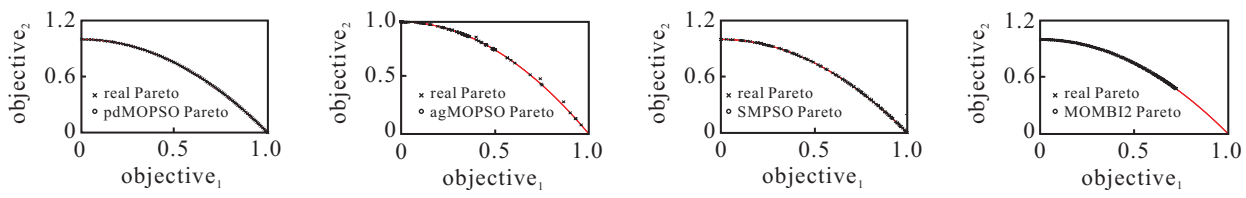
表1为8种算法在9个多目标测试问题上分别独立运行30次后的性能指标平均值. 为表明实验结果的有效性和具体的优劣性,本文使用Wilcoxon's Sign Rank Test检验方法对统计结果进行显著性分析. 图2和图3分别为部分对比算法在ZDT2、ZDT4测试问题上的仿真曲线,图4为算法性能指标统计盒图,图5为算法性能指标随评价次数变化曲线图.

由表1和图4可知,本文算法的Spread指标在7个2目标问题上均有显著优势,对于3目标问题,DTLZ1和DTLZ2则与MOMBI2持平,总体表明算法的分布性更好. 对于IGD指标,可以看出本文算法在所有2目标问题上均取得了最优均值,表明本文提出

的算法在求解2目标优化问题时具有更强的收敛性和分布性优势. 由表1中3目标问题DTLZ1和DTLZ2可以发现,本文算法优于3种对比多目标粒子群优化算法,尤其对于DTLZ1,大部分对比算法无法收敛,表明本文算法相比其他粒子群优化算法,更适合求解高维目标问题. 这是因为本文引入进化状态分类策略更好地利用了粒子群不同进化时期的不同优势,并利用粒子重排策略进行快速局部搜索,因此本文算法可以更好地逼近真实前沿面. 对于ER指标,本文算法全面领先对比算法,这表明本文算法能够更好地覆盖整个真实前沿面. 分析表1和图2可见,在ZDT4多模态测试函数上,部分算法无法做到很好的收敛,这是因为ZDT4测试函数有大约2^{10}个局部最优前沿,全局搜索能力差的算法很难跳出局部最优.

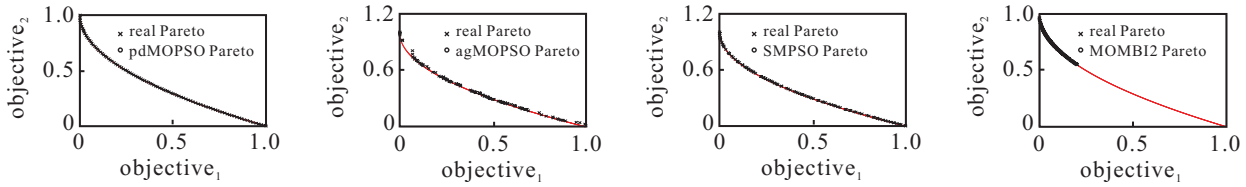
表1　性能指标平均值统计结果

性能指标	测试函数	算法							
		pdMOPSO	agMOPSO	SMPSO	OMPSO	MOMBI2	NSGA-II	SPEA2	MOEA/D
Spread	ZDT1	**8.209e-2**	7.866e-1	2.993e-1	4.355e-1	6.564e-1	3.468e-1	3.328e-1	2.826e-1
	ZDT2	**7.336e-2**	8.819e-1	1.680e-1	4.962e-1	6.999e-1	3.565e-1	3.255e-1	4.247e-1
	ZDT3	**5.137e-1**	7.174e-1	8.814e-1	9.181e-0	7.825e-1	5.481e-1	5.331e-1	8.759e-1
	ZDT4	**9.808e-2**	8.457e-1	6.084e-1	9.138e-1	6.872e-1	3.495e-1	7.005e-1	2.837e-1
	ZDT6	**1.136e-1**	1.458e+0	6.701e-1	5.580e-1	6.945e-1	3.498e-1	7.118e-1	1.351e-1
	UF2	**5.915e-1**	8.028e-1	7.945e-1	7.321e-1	8.444e-1	6.425e-1	6.415e-1	6.858e-1
	UF7	**5.586e-1**	7.482e-1	1.174e+0	8.245e-1	1.166e+0	1.152e+0	1.142e+0	9.000e-1
	DTLZ1	6.853e-1	9.806e-1	8.566e-1	8.874e-1	**6.132e-1**	8.876e-1	7.046e-1	6.746e-1
	DTLZ2	**5.625e-1**	6.111e-1	8.719e-1	6.081e-1	5.954e-1	6.991e-1	8.691e-1	6.321e-1
IGD	ZDT1	**7.139e-2**	7.776e-1	2.976e-1	4.335e-1	6.804e-1	5.518e-1	3.348e-1	2.823e-1
	ZDT2	**6.966e-2**	8.799e-1	1.587e-1	4.962e-1	6.629e-1	3.575e-1	3.285e-1	1.360e-1
	ZDT3	**4.827e-1**	7.204e-1	8.799e-1	9.131e-0	7.895e-1	5.381e-1	5.331e-1	8.754e-1
	ZDT4	**9.948e-2**	8.537e-1	5.407e-1	9.228e-1	6.902e-1	3.445e-1	3.865e-1	2.834e-1
	ZDT6	**9.656e-2**	1.478e+0	1.341e-1	4.970e-1	6.595e-1	3.478e-1	5.548e-1	6.811e-1
	UF2	**5.735e-1**	7.938e-1	7.938e-1	7.241e-1	8.344e-1	6.405e-1	6.165e-1	6.695e-1
	UF7	**5.006e-1**	7.522e-1	7.522e+0	8.195e-1	1.176e+0	1.172e+0	1.202e+0	8.814e-1
	DTLZ1	6.234e-1	9.536e-1	9.536e-1	6.853e-1	**6.042e-1**	8.276e-1	6.306e-1	6.689e-1
	DTLZ2	**5.055e-1**	6.061e-1	6.061e-1	6.081e-1	5.994e-1	7.041e-1	8.671e-1	6.399e-1
ER	ZDT1	**1.430e-2**	3.555e-1	1.261e-1	5.559e-1	7.631e-2	2.171e-1	3.369e-1	9.990e-1
	ZDT2	**1.055e-1**	3.617e-1	2.458e-1	5.829e-1	1.489e-1	3.493e-1	4.712e-1	8.132e-1
	ZDT3	**3.093e-1**	9.263e-1	5.183e-1	6.473e-1	3.161e-1	3.952e-1	4.853e-1	9.417e-1
	ZDT4	**3.776e-2**	9.785e-1	2.166e-1	1.000e+0	3.772e-1	8.900e-1	9.446e-1	9.913e-1
	ZDT6	**3.336e-4**	2.182e-2	1.403e-2	4.375e-2	1.000e+0	1.000e+0	1.000e+0	1.000e+0
	UF2	**9.832e-1**	1.000e+0	9.923e-1	1.000e+0	9.983e-1	9.968e-1	9.989e-1	8.642e-1
	UF7	**5.121e-1**	1.000e+0	1.000e+0	1.000e+0	6.820e-1	5.463e-1	9.994e-1	8.553e-1
	DTLZ1	**7.357e-1**	8.982e-1	9.825e-1	1.000e+0	9.294e-1	9.052e-1	9.423e-1	8.770e-1
	DTLZ2	**8.718e-1**	1.000e+0	9.988e-1	1.000e+0	9.775e-1	9.984e-1	1.000e+0	9.774e-1



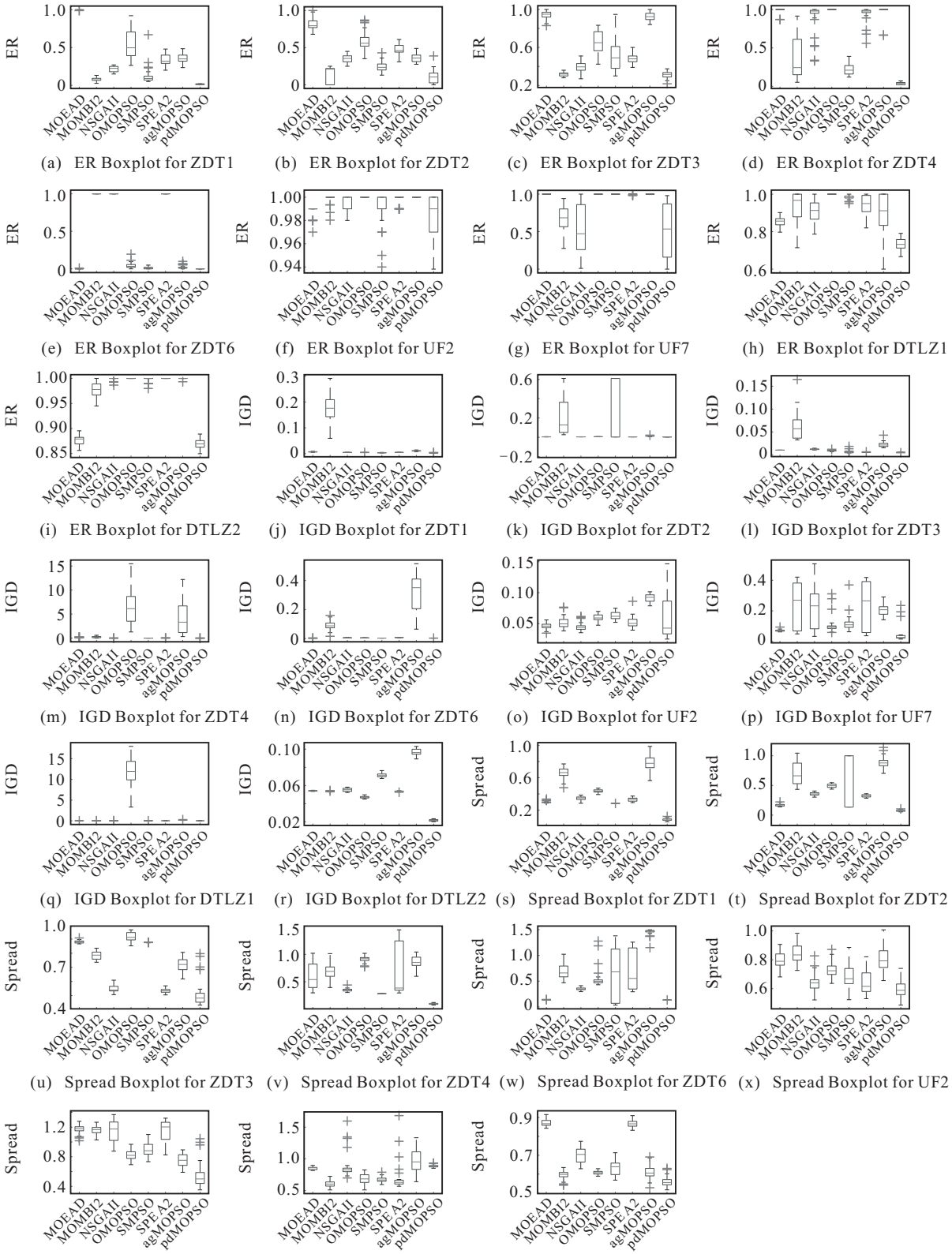
(a) pdMOPSO Pareto　(b) agMOPSO Pareto　(c) SMPSO Pareto　(d) MOMBI2 Pareto

图2　4种算法在ZDT2测试问题中获得Pareto



(a) pdMOPSO Pareto (b) agMOPSO Pareto (c) SMPSO Pareto (d) MOMBI2 Pareto

图 3 4种算法在ZDT4测试问题中获得Pareto



(a) ER Boxplot for ZDT1 (b) ER Boxplot for ZDT2 (c) ER Boxplot for ZDT3 (d) ER Boxplot for ZDT4

(e) ER Boxplot for ZDT6 (f) ER Boxplot for UF2 (g) ER Boxplot for UF7 (h) ER Boxplot for DTLZ1

(i) ER Boxplot for DTLZ2 (j) IGD Boxplot for ZDT1 (k) IGD Boxplot for ZDT2 (l) IGD Boxplot for ZDT3

(m) IGD Boxplot for ZDT4 (n) IGD Boxplot for ZDT6 (o) IGD Boxplot for UF2 (p) IGD Boxplot for UF7

(q) IGD Boxplot for DTLZ1 (r) IGD Boxplot for DTLZ2 (s) Spread Boxplot for ZDT1 (t) Spread Boxplot for ZDT2

(u) Spread Boxplot for ZDT3 (v) Spread Boxplot for ZDT4 (w) Spread Boxplot for ZDT6 (x) Spread Boxplot for UF2

(y) Spread Boxplot for UF7 (z) Spread Boxplot for DTLZ1 (z') Spread Boxplot for DTLZ2

图 4 性能指标盒图

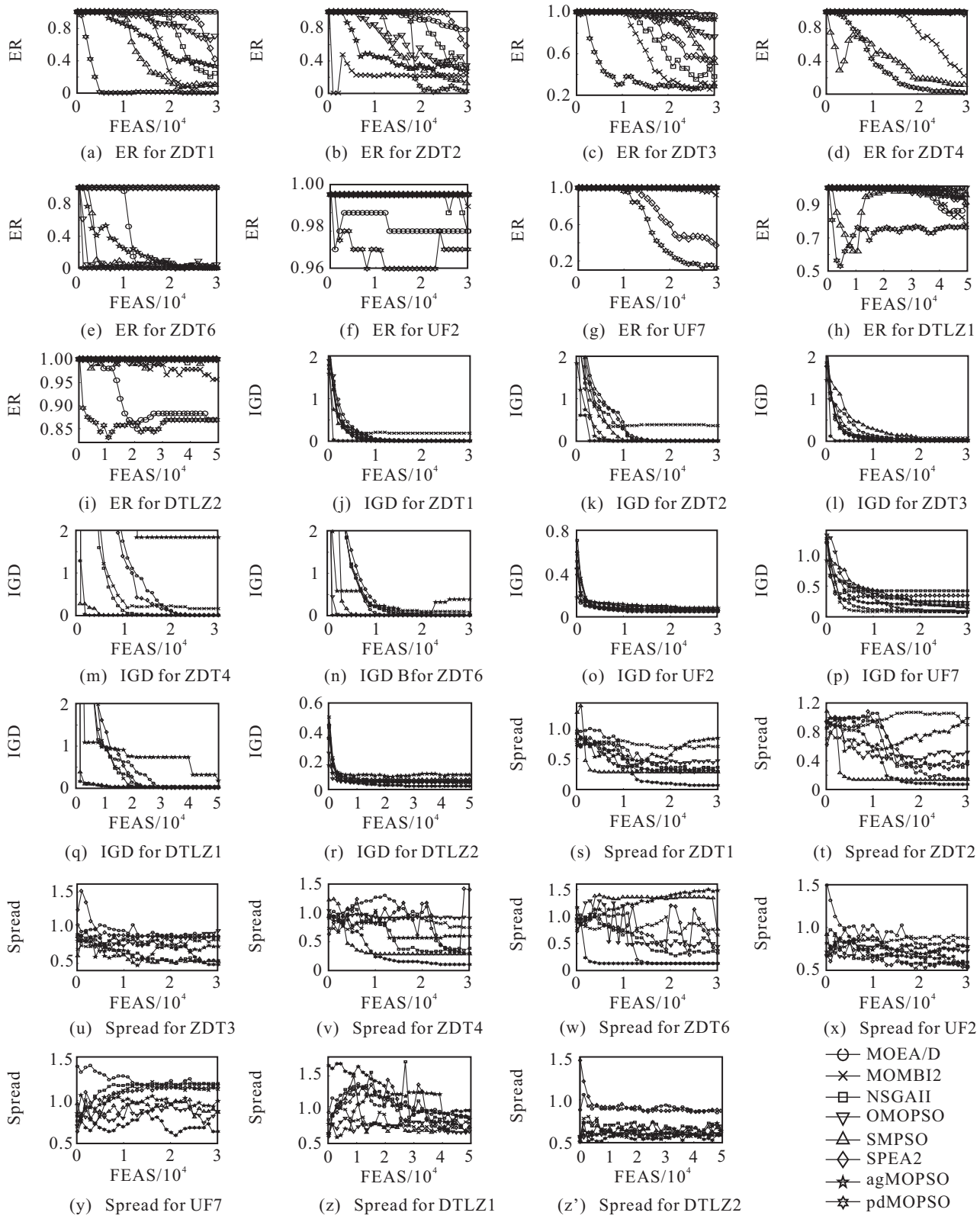


图 5　性能指标变化曲线

由图5可见,不同算法在进化后期均出现一定程度的进化停滞,表现为性能指标无变化或变化幅度过小. 此外,3种对比多目标粒子群进化算法出现了在进化后期无法避免的摆动问题,主要表现为性能指标值上下浮动或衰退. 本文算法得益于重新设计的归档集维护策略以及位置重置策略,可以在进化后期更容易避免发生同样问题.

为了直观地展示对比算法的收敛性和分布性,图2和图3给出了本文算法与部分对比算法在部分测试函数上的最优解集. 可以明显地发现,本文算法获得解已经非常接近真实前沿,3个对比算法则都有不同程度的前沿面缺失. 对于基于参考点的MOMBI2算

法,虽然近年来类似基于参考点的策略已广泛应用于解决高维问题中,但仍然无法避免该类算法十分依赖于参考点的生成和选择,同时该类算法在解决多目标问题上也暴露出了寻优能力不足的现象,这主要是因为空间中参考点不足. 综合考虑,本文算法在本次实验中,相比于其余7种对比算法,有着明显的优势.

4 结 论

为解决现有粒子群优化算法在求解多目标优化问题上存在的不足,通过统计方法,分析进化种群在决策空间的分布变化情况,从而获得种群在决策空间的分布性和迁移变化情况. 评估种群的多样性和进化状态,并以此为反馈信息对粒子群在决策空间进行自适应的位置调整,作为种群多样性保持策略,在提高算法探索广度的同时减少算法陷入局部收敛的可能. 同时,分别从收敛性和多样性角度设计两种归档集维护和引导粒子选择策略,通过判断进化状态从而实现策略的动态切换, 最终形成了基于群体分布信息的自适应多目标粒子群优化算法. 实验结果表明,所提出算法能够获得分布性更均匀、覆盖更广泛、收敛性更好的高质量解集. 下一步工作是考虑将算法扩展到高维目标优化问题的求解中.

参考文献(References)

[1] Kennedy J, Eberhart R. Particle swarm optimization[C]. Proc of the IEEE Int Conf on Neural Networks. Piscataway: IEEE, 1995: 1942-1948.

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

[3] 胡旺, Yen G G, 张鑫. 基于Pareto熵的多目标粒子群优化算法[J]. 软件学报, 2014, 25(5):1025-1050.
(Hu W, Yen G G, Zhang X. Multi-objective particle swarm optimization based on pareto entropy[J]. J of Software, 2014, 25(5): 1025-1050.)

[4] Hernández G Raquel, Coello C A C. Improved metaheuristic based on the r2 indicator for many-objective optimization[C]. Proc of the 2015 on Genetic and Evolutionary Computation Conf. ACM, 2015: 679-686.

[5] Al M N, Andrei P, John M D. MOPSO: MOPSO based on decomposition and dominance with archiving using crowding distance in objective and solution spaces[J]. IEEE Trans on Evolutionary Computation, 2014, 22(1): 47-77.

[6] 杨景明, 马明明, 车海军,等. 多目标自适应混沌粒子群优化算法[J]. 控制与决策, 2015, 30(12): 2168-2174.
(Yang J M, Ma M M, Che H J, et al. Mutli-objective adaptive chaotic particle swarm optimization algorithm[J]. Control and Decision, 2015, 30(12): 2168-2174.)

[7] Lee K B, Kim J H. Multiobjective particle swarm optimization with preference-based sort and its application to path following footstep optimization for humanoid robots[J]. IEEE Trans on Evolutionary Computation, 2013, 17(17): 755-766.

[8] Zhan Z H, Li J, Cao J, et al. Multiple populations for multiple objectives: A coevolutionary technique for solving multiobjective optimization problems[J]. IEEE Trans on Cybernetics, 2013, 43(2): 445-463.

[9] 杨宁, 霍炬, 杨明. 基于多层次信息交互的多目标粒子群优化算法[J]. 控制与决策, 2016, 31(5): 907-912.
(Yang N, Huo J, Yang M. Multi-objective particle swarm optimization algorithm based on interaction of multi-level information[J]. Control and Decision, 2016, 31(5): 907-912.)

[10] Zitzler E, Deb K, Thiele L. Comparison of multi-objective evolutionary algorithms: Empirical results[J]. IEEE Trans on Evolutionary Computation, 2000, 8(2): 173-195.

[11] Zhang Q, Zhou A, Zhao S, et al. Multiobjective optimization test instances for the CEC 2009 special session and competition[C]. Special Session on Performance Assessment of Multiobjective Optimization Algorithms. 2008: 1-30.

[12] Deb K, Thiele L, Laumanns M, et al. Scalable test problems for evolutionary multiobjective optimization[C]. Evolutionary Multiobjective Optimization. London: Springer, 2005: 105-145.

[13] Nebro AJ, Durillo J J, Garcia-Nieto J, et al. SMPSO: A new pso-based metaheuristic for multi-objective optimization[C]. Computational Intelligence in Miulti-criteria Decision-making. Piscataway: IEEE, 2009: 66-73.

[14] Sierra M R, Coello C A C. Improving PSO-based multi-objective optimization using crowding, mutation and ε-dominance[C]. Evolutionary Multi-Criterion Optimization. Beilin: Springer Heidelberg, 2005: 505-519.

[15] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE Trans on Evolutionary Computation, 2002, 6(2): 182-197.

[16] Zizler E, Laumanns M, Thiele L. SPEA2: Improving the strength Pareto evolutionary algorithm[R]. Lanusanne: Swiss Federal Institute of Technology Computer Engineering and Networks Laboratory, 2001.

[17] Zhang Q F, Hui L. MOEA/D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Trans on Evolutionary Computation, 2007, 11(6): 712-731.

（责任编辑：郑晓蕾）