基于参考点的多模态多目标粒子群算法

李国森[1] 闫 李[1*] 郭倩倩[1] 周同驰[1] 王永林[1] 岳彩通[2,3]

[1](中原工学院电子信息学院 河南 郑州 450007)

[2](郑州大学电气工程学院 河南 郑州 450001)

[3](郑州大学产业技术研究院 河南 郑州 450001)

摘 要 针对多目标优化算法在求解多模态多目标问题时存在 Pareto 解集不完整、收敛性差等缺点,提出一种基于参考点的多模态多目标粒子群算法(RPMOPSO)。利用佳点集原理初始化参考点以保证参考点的均匀分布性;引入参考点策略促使种群形成稳定的小生境以增加种群多样性;采用扩散机制改变粒子的飞行轨迹以捕捉到更多 Pareto 解。在十二个常用测试问题中比较六种算法的性能,结果表明,RPMOPSO 在决策空间上具有良好的收敛能力,且能够找到更多的 Pareto 解。

关键词 进化算法 粒子群优化 多目标优化 多模态优化 Pareto 最优解集

中图分类号 TP183 **文献标志码** A DOI:10.3969/j.issn.1000-386x.2020.11.032

REFERENCE-POINT-BASED MULTIMODAL MULTI-OBJECTIVE PARTICLE SWARM OPTIMIZATION

Li Guosen[1] Yan Li[1*] Guo Qianqian[1] Zhou Tongchi[1] Wang Yonglin[1] Yue Caitong[2,3]

[1](*School of Electronic & Information Engineering, Zhongyuan University of Technology, Zhengzhou 450007, Henan, China*)

[2](*School of Electrical Engineering, Zhengzhou University, Zhengzhou 450001, Henan, China*)

[3](*Industrial Technology Research Institute, Zhengzhou University, Zhengzhou 450001, Henan, China*)

Abstract The multi-objective algorithm has disadvantage of incomplete Pareto set and poor convergence in solving multimodal multi-objective optimization problems. To solve this problem, a reference-point-based multimodal multi-objective particle swarm optimization (RPMOPSO) is proposed. The reference points were initialized by the good point set to ensure that the reference points were uniformly distributed in the decision space; the reference point strategy was introduced to induce stable niches to assure the diversity of the population; the diffusion mechanism was used to change the flight trajectory of the particle to capture more Pareto solutions. The performance of six algorithms was compared on twelve commonl test functions . The results show that RPMOPSO has good convergence ability in decision space and can find more Pareto solutions.

Keywords Evolutionary algorithm Particle swarm optimization Multi-objective optimization Multimodal optimization Pareto optimal set

0 引 言

在实际工程应用中,往往存在多个目标相互冲突的多目标优化问题[1]。由于目标之间的冲突性,这类问题不存在多个目标同时最优的唯一解,而是存在一组互不占优的折衷解,称为 Pareto 最优解集(Pareto-optimal Set,PS)[2]。近年来,许多解决多目标问题的优化算法被相继提出[3],如 NSGA-II[4]、PESA-II[5]、SPEA2[6]和 MOEA/D[7]等。这些算法得到的 Pareto 最

收稿日期:2019-07-15。国家自然科学基金项目(61673404,61876169);河南省高等学校重点科研项目(19A120014);河南省高等学校青年骨干教师培养计划项目(2016GGJS-094);中国纺织工业联合会科技指导性项目(2017054,2018104)。**李国森**,硕士生,主研领域:最优化理论,智能计算。**闫李**,博士。**郭倩倩**,助工。**周同驰**,博士。**王永林**,讲师。**岳彩通**,博士。

优解集在目标空间映射后得到的 Pareto 前沿(Pareto Front,PF)分布均匀且收敛[8-9]。

在多目标优化问题中,存在一类问题具有多模态的特性,如路径规划问题[10]、背包问题[11]、火箭发动机设计问题[12]和作业车间调度问题[13]。这些问题存在着多个不同的 Pareto 解集,对应着目标空间中的同一个 Pareto 前沿[14],被称为多模态多目标优化问题。如何设计多模态多目标算法使其能够获得多个不同的 Pareto 解集是解决这类问题的关键。多个 Pareto 解集有利于决策者根据自己的喜好选择合适的解决方案。

多模态多目标优化问题在提出之后,逐渐获得研究学者的关注。Deb 等[14]提出了 Omni-optimizer 算法,将决策空间中拥挤距离的概念引入到非支配排序方法中以提高解在决策空间中的多样性。Liang 等[16]提出了 DN-NSGAII 算法,采用基于决策空间的非支配排序算法以改善解在决策空间中的分布。Yue 等[10]采用基于环形拓扑的小生境方法,并将特殊拥挤距离嵌入到非支配排序方法中以获得更多的 Pareto 解。Liu 等[11]利用双档案机制和重组策略指导种群进化,提高种群的多样性和收敛性。Liang 等[17]采用自组织映射网络在决策空间中建立邻域关系,并采用精英学习策略避免算法收敛过快。

但是上述研究在解决多模态多目标问题时仍然存在 Pareto 解集不完整、收敛性较差等问题,本文以多目标粒子群算法(Multi-objective Particle Swarm Optimization, MOPSO)[18]为框架,提出了一种基于参考点的多模态多目标粒子群算法(RPMOPSO)。通过引入参考点策略以提高种群多样性,并采用扩散机制找到更多的 Pareto 解。利用十二个测试函数,将 RPMOPSO 与其他六种算法进行性能比较,结果表明 RPMOPSO 在寻优性能和收敛性上表现良好。

1　基本概念

1.1　多模态多目标优化问题

如果多目标问题在决策空间中存在多个 Pareto 解集,且这些解集映射到目标空间中同一 Pareto 前沿,则称为多模态多目标优化问题(multimodal multi-objective optimization problems,MMOPs)[10]。该优化问题如图 1 所示,决策空间中 Pareto 解集 PS_1、PS_2 对应着目标空间中相同的 PF。例如,A_1 和 A_2 在决策空间中的位置不同,但对应目标空间中相同的点 A'。找到 A_1 和 A_2 这两个解,可以给决策者不同的选择方案。

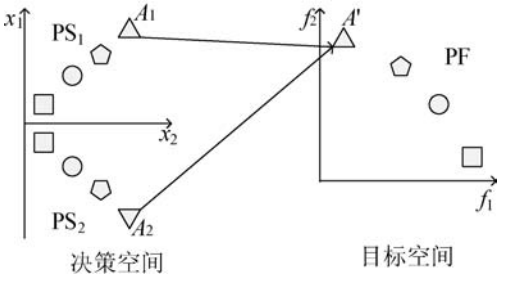


图 1　多模态多目标优化问题

1.2　粒子群算法描述

设种群规模为 N 的种群,在一个 D 维的决策空间中进行寻优。在 t 时刻,第 i 个粒子在其个体最优位置 $p_i(t)$ 和全局最优位置 $g_i(t)$ 的引导下从当前位置 $x_i(t)$ 以速度 $v_i(t)$ 飞行[19-20],则该粒子在 $(t+1)$ 时刻的速度 $v_i(t+1)$ 和位置 $x_i(t+1)$ 可表示为:

$$v_i(t+1) = wv_i(t) + c_1r_1(p_i(t) - x_i(t)) + c_2r_2(g_i(t) - x_i(t)) \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

式中:w 为惯性权重;c_1 和 c_2 为学习因子;r_1 和 r_2 为区间[0,1]上的随机数。

随后为了进一步控制粒子的飞行速度,将压缩因子 χ 引入粒子的速度公式[21],表示如下:

$$v_i(t+1) = \chi(v_i(t) + c_1r_1(p_i(t) - x_i(t)) + c_2r_2(g_i(t) - x_i(t))) \quad (3)$$

$$\chi = \frac{2}{\left|2 - \rho - \sqrt{\rho^2 - 4\rho}\right|} \quad \rho = c_1 + c_2, \rho > 4 \quad (4)$$

2　算法设计

2.1　参考点策略

在传统粒子群算法中,每个粒子采用共享的全局最优信息更新自身的位置,以促使种群快速收敛,但在一定程度上降低了解的多样性。

为了提高种群的多样性以定位到更多 Pareto 解,本文提出一种参考点策略。参考点策略的主要思想是每个参考点存储一个其所在邻域内的最优解(记为 o_*),而每个粒子仅与相邻的参考点直接通信。具体地,本文采用佳点集原理[22-23]设计参考点,佳点集原理所产生的点能够均匀分布在决策空间。在算法初始化时,每个参考点所存储的最优解 o_* 为其本身。在每次迭代开始时,每个参考点会首先计算种群中离自己最近的个体,并将该个体与其所保留的最优信息 o_* 比较,最后将非支配的一个个体保留为 o_*。在粒子进行更新时,每个粒子选择离自己最近的参考点,并学习该参考点所保留的最优信息 o_*。因此,每个粒子在参考

点形成的小生境内进化,使得每个粒子的搜索相对独立。图2为一个二维决策空间的参考点策略示意图,其中:$x_i \in [x_i^{\min}, x_i^{\max}], i = 1,2$。每个粒子仅和其最近的参考点进行交流,当存在多个全局最优解时,每个粒子能够在各自的小生境内独立寻优,使得算法能够找到更多的 Pareto 解集。

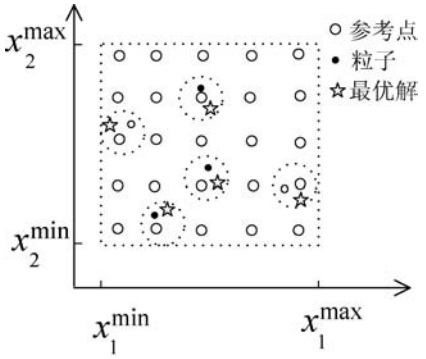


图2 参考点策略示意图

因此,在粒子进行速度更新时,式(3)重新定义为:

$$v_i(t+1) = \chi(v_i(t) + c_1 r_1(p_i(t) - x_i(t)) + c_2 r_2(o_* - x_i(t))) \quad (5)$$

2.2 扩散机制

本文采用扩散机制,以帮助粒子跳出局部最优。一方面,传统粒子群算法存在早熟停滞或者陷入局部最优的问题;另一方面,如果过多的粒子在某一次迭代中选择同一参考点,将导致这些粒子聚集在同一最优解附近,从而过度开发某一区域,而难以找到更多最优解。因此,扩散机制的基本思想是改变这些粒子的飞行轨迹,将其扩散到更大的区域,使其有机会选择新的参考点。

在 t 次迭代,设第 j 个($j = 1,2,\cdots,N$) 参考点被种群中所有粒子选择的次数为 φ_j。例如,种群中有 m 个粒子均选择了第 j 个参考点,则 $\varphi_j = m$。假设第 i 个粒子选择了第 j 个参考点。如果 $rand(0,1) < \frac{\varphi_j}{\varphi_{\text{mean}}}$ 成立,$rand(0,1)$ 表示取值为[0,1]的随机数,φ_{mean} 为所有参考点被选择次数的平均值,那么第 i 个粒子将进行扩散,即按如下公式进行位置的更新:

$$x_i(t+1) = x_i(t) + \frac{Levy(\lambda)}{\| p_i(t) - x_i(t) \|} \cdot (p_i(t) - x_i(t)) \quad (6)$$

式中:$Levy(\lambda)$ 是服从参数 $\lambda(1 < \lambda \leqslant 3)$ 的 Levy 分布的随机数;$\| p_i(t) - x_i(t) \|$ 表示 $p_i(t)$ 和 $x_i(t)$ 之间的距离。在式(6)中,一方面,Levy 分布具有长距离和短距离跳跃相间的特性[24-25]。其长跳跃的搜索方式有利于全局探索,且不容易陷入局部停滞;另一方面,粒子根据历史最优位置和当前位置的距离动态调整搜索范围,以提高算法的全局勘探能力。当 $p_i(t)$ 和 $x_i(t)$ 距离很近时,粒子的搜索范围将变得很大,以搜索到更多的解;反之,粒子不会立即收敛到最优解,避免陷入局部最优。

2.3 算法步骤

步骤1 设置算法的种群大小 N、学习因子(c_1、c_2)、最大迭代次数 $MaxIter$,根据佳点集原理初始化种群 P,外部档案 $EXA = P$,参考点 $RP = P$,参考点用于保留最优解的档案 $RPA = P$。

步骤2 计算第 j 个($j = 1,2,\cdots,N$) 参考点被所有粒子选择的次数为 φ_j。

步骤3 对于第 i 个($i = 1,2,\cdots,N$) 粒子 $P\{i\}$,计算该粒子离参考点最近的点(记为 $RP\{j\}$),则参考点 $RP\{j\}$ 所保留的最优信息 o_* 为 $RPA\{j\}$。根据式(5)计算粒子的速度,然后根据式(2)计算粒子的位置。

步骤4 如果满足 $rand(0,1) < \frac{\varphi_j}{\varphi_{\text{mean}}}$,则执行扩散机制,按照式(6)计算粒子的位置。

步骤5 对于第 j 个参考点 $RP\{j\}$,计算种群中离其最近的粒子,将该粒子与该参考点保留的最优解 $RPA\{j\}$ 进行比较,保留一个非支配解进入档案 $RPA\{j\}$。

步骤6 更新外部档案 $EXA = [P;EXA]$,对外部档案进行非支配排序,选择占优性强且稀疏的前 N 个解存入 EXA。

步骤7 若迭代次数达到最大迭代次数 $MaxIter$,输出外部档案;否则,返回步骤2。

3 实 验

3.1 测试函数

本文采用 12 个经典的多模态多目标测试函数[10],包括 MMF1-MMF8、SYM-PART simple、SYM-PART rotate、SYM-PART rot. + trans. 和 Omni-test 函数($D = 3$)。其中,Omni-test 函数在决策空间中的 Pareto解集个数最多,用来测试算法在目标空间的寻优能力。

3.2 实验设置

使用六种优化算法与 RPMOPSO 进行比较,包括三种多模态多目标算法(MO_Ring_PSO_SCD[10]、Omni-optimizer[9] 和 DN-NSGAII[16]) 和三种多目标算法(NSGA-II、PESA-II 和 MOEA/D)。其中,RPMOPSO 参数如下:$c_1 = c_2 = 2.05$。其他算法的参数值与相应文献

中的参数值相同。所有算法的种群大小 N 和最大迭代次数 $MaxIter$ 分别设置为 800 和 100[10],外部归档大小设置为 800,独立运行 25 次。采用 MATLAB R2014b 进行仿真,运行环境为 Intel ® Core™ I7 – 4790 处理器,内存为 16 GB。

3.3　评价指标

本文采用 Pareto 解集近似性(PSP)[10] 和超体积(Hv)[26] 指标衡量算法性能。PSP 用于度量所获得 PS 的收敛性和分布性。Hv 用于评价所获得 PF 的收敛性和分布性。PSP 值越大意味着在决策空间中获得的 PS 的收敛性和多样性越好。Hv 值越大意味着在目标空间中获得的 PF 越接近真实的 PF。

3.4　实验结果

3.4.1　策略性能对比

为了验证所提策略在解决多模态多目标问题时的有效性,将 RPMOPSO 算法和 MOPSO[10]、RPMOPSO-I(采用参考点策略的 MOPSO 算法)和 RPMOPSO-II(采用扩散机制的 MOPSO 算法)进行比较。表 1 所示为结合不同策略的算法的 PSP 的均值、标准差,以及 Wilcoxon 秩和检验的 h 值。

表 1　各策略 PSP 指标统计

测试函数	RPMOPSO 均值 ± 标准差	RPMOPSO-I 均值 ± 标准差(h)	RPMOPSO-II 均值 ± 标准差(h)	MOPSO 均值 ± 标准差(h)
MMF1	**82.95 ±3.78**	56.27 ±2.23(1)	2.92 ±0.40(1)	2.87 ±0.45(1)
MMF2	**157.20 ±20.59**	84.10 ±13.95(1)	8.74 ±1.57(1)	4.52 ±2.45(1)
MMF3	**185.11 ±22.18**	99.35 ±15.24(1)	7.14 ±1.70(1)	5.54 ±2.27(1)
MMF4	**138.80 ±3.75**	117.22 ±2.95(1)	3.35 ±0.72(1)	3.11 ±0.68(1)
MMF5	**38.33 ±1.06**	28.27 ±1.40(1)	2.16 ±0.33(1)	1.91 ±0.36(1)
MMF6	**41.44 ±1.30**	34.28 ±1.36(1)	2.47 ±0.38(1)	2.18 ±0.47(1)
MMF7	**141.01 ±5.11**	106.66 ±3.93(1)	3.67 ±0.31(1)	3.61 ±0.98(1)
MMF8	**70.27 ±1.86**	40.04 ±2.19(1)	1.12 ±0.24(1)	1.30 ±0.39(1)
SYM-PART simple	**50.62 ±1.68**	24.08 ±1.05(1)	0.83 ±0.54(1)	0.62 ±0.50(1)
SYM-PART rotated	**41.37 ±4.75**	23.26 ±1.34(1)	0.75 ±0.41(1)	0.67 ±0.55(1)
SYM-PART rot. + trans	**55.70 ±4.35**	22.91 ±1.54(1)	0.78 ±0.05(1)	0.40 ±0.12(1)
Omni-test	**16.99 ±1.12**	6.28 ±2.53(1)	1.08 ±0.38(1)	0.65 ±0.15(1)

可以看出,RPMOPSO-I 的性能优于 MOPSO,而 RPMOPSO-II 的性能略优于 MOPSO。因此,参考点策略和扩散机制能有效地改善基本 MOPSO 的性能。此外,秩和检验的结果表明,两种策略的结合能够使算法性能得到显著的提升。其原因在于参考点策略使 RPMOPSO 能够找到更多的 Pareto 最优解。参考点策略可以在决策空间中产生稳定的小生境,每个粒子在自己的小生境邻域内进化,且不会受其他邻域的影响,以提高种群的多样性。扩散机制能够使粒子向更大的范围内寻优,因此陷入局部最优的粒子不仅可以跳出当前的最优解,而且可以捕捉到更多的最优解。

3.4.2　算法性能对比

本节将 RPMOPSO 与三种多模态多目标算法和三种多目标算法进行比较。首先,比较不同算法在决策空间获得 PS 的能力,不同算法的 PSP 值如表 2 所示。结果表明 RPMOPSO 在所有测试函数中具有较好的表现,其性能优于其他六种算法。MO_Ring_PSO_SCD 排第二名,因其使用环形拓扑能够形成小生境,能够找到更多 Pareto 解。DN-NSGAII 和 Omni-optimizer 的表现紧随其后,两者采用决策空间的拥挤距离作为环境选择提高了算法的性能。NSGA-II、MOEA/D 和 PESA-II 表现较差,其原因在于这三种多目标算法没有考虑决策空间中解的分布。其次,评价算法在目标空间中获得的 PF 的性能,不同算法的 Hv 值如表 3 所示。可以看出 RPMOPSO 在 Hv 性能指标上表现不是最好的。实际上,所有算法在同一个测试函数上的 Hv 值是很接近的。原因是所有的算法都考虑了最优解在目标空间中的分布。

表 2 不同算法 PSP 指标统计

测试函数	RPMOPSO 均值 ± 标准差	MO_Ring_ PSO_SCD 均值 ± 标准差(*h*)	DN-NSGAII 均值 ± 标准差(*h*)	Omni-optimizer 均值 ± 标准差(*h*)	NSGA-II 均值 ± 标准差(*h*)	MOEA/D 均值 ± 标准差(*h*)	PESA-II 均值 ± 标准差(*h*)
MMF1	**82.95 ±3.78**	66.80 ±2.89(1)	44.61 ±2.62(1)	47.64 ±6.46(1)	23.67 ±3.77(1)	7.07 ±3.17(1)	27.39 ±4.41(1)
MMF2	**157.20 ±20.59**	98.13 ±12.91(1)	62.49 ±12.38(1)	77.93 ±31.42(1)	49.73 ±24.39(1)	4.74 ±3.78(1)	36.05 ±19.51(1)
MMF3	**185.11 ±22.18**	132.71 ±23.13(1)	74.82 ±20.14(1)	92.34 ±31.74(1)	65.36 ±18.99(1)	6.56 ±4.39(1)	59.32 ±18.54(1)
MMF4	**138.80 ±3.75**	115.30 ±4.33(1)	41.38 ±4.06(1)	39.82 ±8.85(1)	18.35 ±5.68(1)	2.54 ±1.94(1)	44.67 ±12.69(1)
MMF5	**38.33 ±1.06**	32.91 ±1.27(1)	15.16 ±1.54(1)	14.75 ±1.39(1)	10.96 ±1.13(1)	3.59 ±1.45(1)	17.17 ±1.85(1)
MMF6	**41.44 ±1.30**	36.01 ±1.26(1)	17.18 ±1.23(1)	17.77 ±1.73(1)	12.82 ±0.82(1)	5.00 ±2.36(1)	20.68 ±1.87(1)
MMF7	**141.01 ±5.11**	108.01 ±5.88(1)	94.87 ±5.03(1)	99.96 ±21.34(1)	36.79 ±5.79(1)	7.46 ±3.88(1)	41.93 ±15.95(1)
MMF8	**70.27 ±1.86**	47.21 ±1.42(1)	18.05 ±1.82(1)	17.63 ±6.75(1)	0.32 ±0.31(1)	0.16 ±0.12(1)	1.64 ±1.23(1)
SYM-PART simple	**50.62 ±1.68**	21.46 ±1.01(1)	0.35 ±1.82(1)	0.40 ±0.16(1)	0.01 ±0.00(1)	0.01 ±0.01(1)	0.01 ±0.02(1)
SYM-PART rotated	**41.37 ±4.75**	18.27 ±1.52(1)	3.53 ±1.08(1)	2.14 ±3.85(1)	0.06 ±0.07(1)	0.01 ±0.00(1)	0.13 ±0.11(1)
SYM-PART rot. + trans	**55.70 ±4.35**	10.59 ±4.34(1)	4.31 ±5.09(1)	1.97 ±10.93(1)	0.03 ±0.04(1)	0.01 ±0.01(1)	0.09 ±0.10(1)
Omni-test	**16.99 ±1.12**	11.45 ±0.58(1)	1.06 ±0.80(1)	0.94 ±0.17(1)	0.43 ±0.21(1)	0.04 ±0.01(1)	0.28 ±0.14(1)

表 3 不同算法 Hv 指标统计

测试函数	RPMOPSO 均值 ± 标准差	MO_Ring_ PSO_SCD 均值 ± 标准差	DN-NSGAII 均值 ± 标准差	Omni-optimizer 均值 ± 标准差	NSGA-II 均值 ± 标准差	MOEA/D 均值 ± 标准差	PESA-II 均值 ± 标准差
MMF1	3.66 ±7.74e -04	3.66 ±4.54e -04	3.66 ±1.12e -03	**3.67 ±2.43e -05**	3.67 ±1.26e -03	3.67 ±5.74e -04	3.66 ±2.81e -04
MMF2	3.65 ±8.17e -03	3.65 ±7.61e -03	3.66 ±9.08e -03	**3.67 ±4.69e -05**	3.66 ±4.82e -03	3.67 ±8.90e -04	3.66 ±2.14e -03
MMF3	3.66 ±4.50e -03	3.65 ±6.63e -03	3.67 ±5.25e -04	**3.67 ±2.52e -05**	3.66 ±1.14e -02	3.67 ±2.64e -04	3.66 ±1.46e -03
MMF4	3.33 ±1.17e -03	3.30 ±9.54e -04	3.18 ±3.31e -04	3.32 ±3.84e -05	3.32 ±2.18e -06	**3.33 ±1.68e -04**	3.33 ±9.64e -04
MMF5	3.66 ±7.79e -04	3.66 ±3.89e -04	3.67 ±3.22e -04	3.67 ±1.81e -05	**3.67 ±1.69e -05**	3.67 ±1.03e -03	3.66 ±3.96e -04
MMF6	3.66 ±4.04e -04	3.66 ±3.33e -04	3.66 ±1.41e -03	**3.67 ±2.19e -05**	3.67 ±8.45e -05	3.67 ±6.44e -04	3.66 ±4.03e -04
MMF7	3.67 ±2.58e -04	3.67 ±2.10e -04	3.66 ±1.24e -03	3.67 ±4.44e -05	**3.67 ±1.73e -05**	3.67 ±3.18e -04	3.66 ±9.29e -04
MMF8	3.21 ±1.01e -03	3.21 ±1.09e -03	3.21 ±1.22e -03	3.21 ±1.20e -04	**3.21 ±7.70e -05**	3.21 ±2.39e -04	3.21 ±5.71e -03
SYM-PART simple	1.32 ±1.66e -04	1.30 ±1.65e -03	1.32 ±1.94e -04	1.32 ±3.21e -04	1.32 ±2.32e -04	**1.32 ±1.07e -04**	1.32 ±7.04e -04
SYM-PART rotated	1.32 ±2.70e -04	1.29 ±3.55e -03	1.32 ±4.49e -04	1.32 ±2.55e -04	**1.32 ±2.41e -04**	1.32 ±5.98e -04	1.32 ±3.74e -03
SYM-PART rot. + trans	1.32 ±5.25e -03	1.29 ±2.64e -03	1.32 ±3.45e -4	1.32 ±2.84e -4	**1.32 ±2.77e -4**	1.32 ±7.50e -04	1.32 ±5.36e -03
Omni-test	62.03 ±2.58e -03	61.93 ±2.14e -01	62.06 ±3.95e -04	62.06 ±2.46e -04	**62.06 ±1.20e -04**	62.06 ±2.63e -03	61.99 ±3.20e -02

为了进一步比较算法的性能,将所有算法在 Omni-test 测试函数上获得的 PS 和 PF 分别在图 3、图 4 进行展示。Omni-test 测试函数有 27 个 PS。从图 3 可以看出,RPMOPSO 获得的 Pareto 最优解能够比较全面地覆盖真正的 PS。MO_Ring_PSO_SCD、DN-NSGAII 和 Omni-optimizer 获得的 PS 并不完整。NSGA-II、PESA-II 和 MOEA/D 搜索到的 PS 更少。其中,MOEA/D 仅获得一个 PS。这说明没有小生境技术很难维持多个 PS。从图 4 可以看出,所有算法在 Omni-test 测试函数上都能够较好的覆盖整个真实的 PF,且能够逼近真正的 PF。

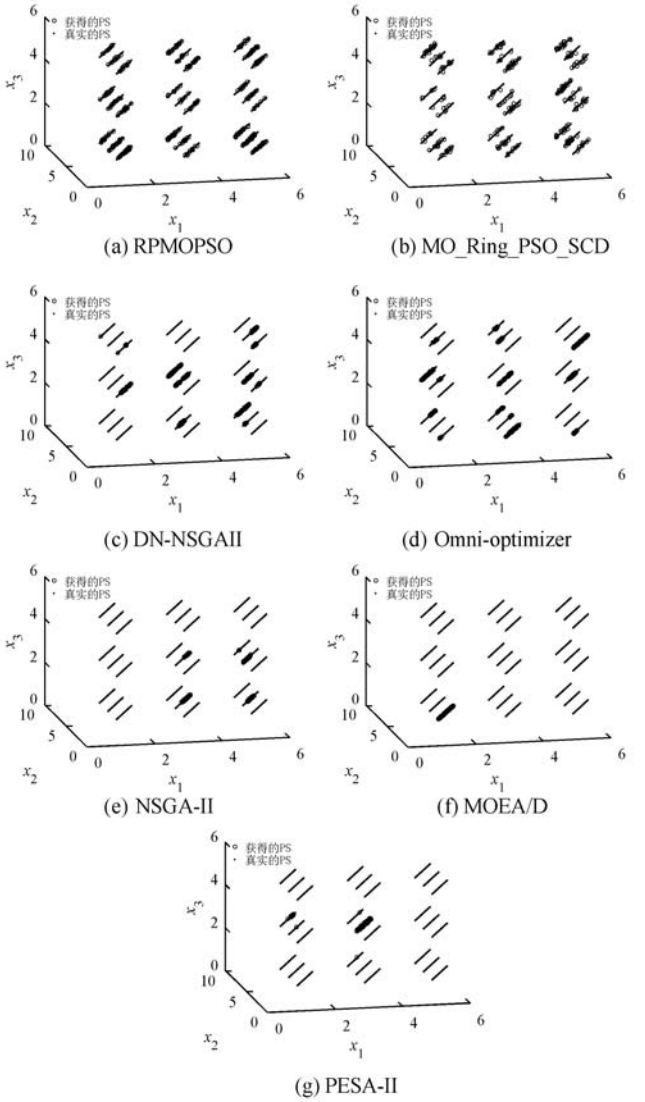
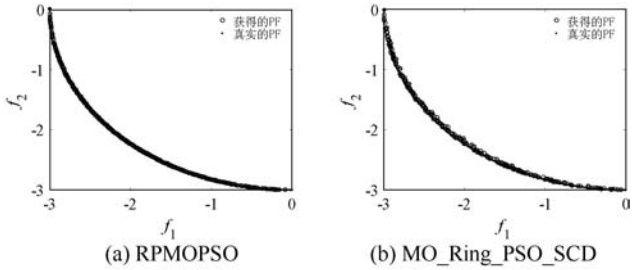


图 3　各算法所获得的 PS 对比



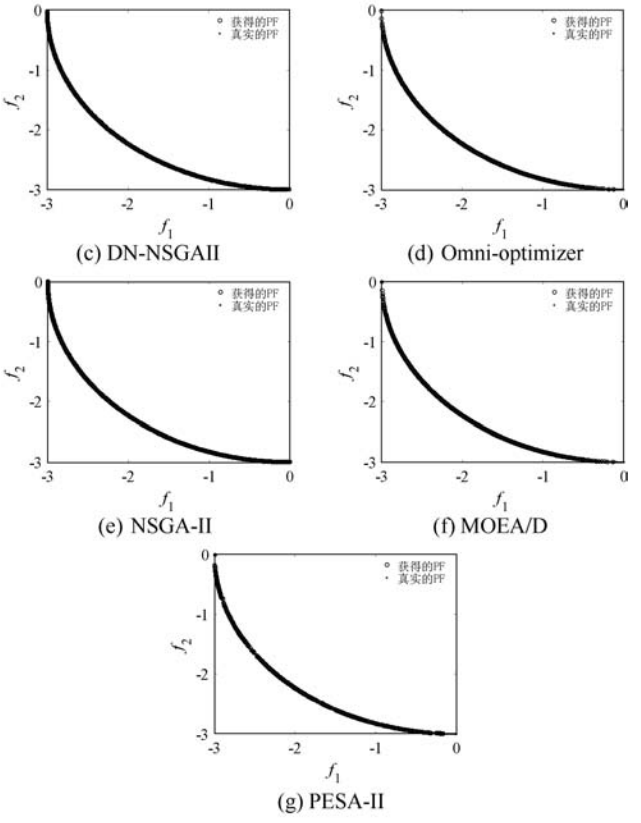


图 4　各算法所获得的 PF 对比

综上,在获得近似 PF 的前提下,RPMOPSO 在决策空间中获得的 PS 分布性良好。

3.4.3　算法收敛性比较

为了比较算法的收敛性[10],将四个多模态多目标算法(RPMOPSO,MO_Ring_PSO_SCD,Omni-optimizer 和 DN-NSGAII)在测试函数 MMF4 上进行比较。MMF4 在决策空间上有 4 个 PS,如图 5 所示,其 PS 的分布将决策空间划分为 4 个子区域,分别为区域 1、区域 2、区域 3 和区域 4。这样每个子区域面积相等,且每个子区域均有一个 PS。算法的收敛性可以定义为每一代种群分布在每个子区域中解的比例[10]。在理想的情况下,如果算法在测试函数 MMF4 上收敛性很好,那么每个子区域中解的比例应等于 25%。

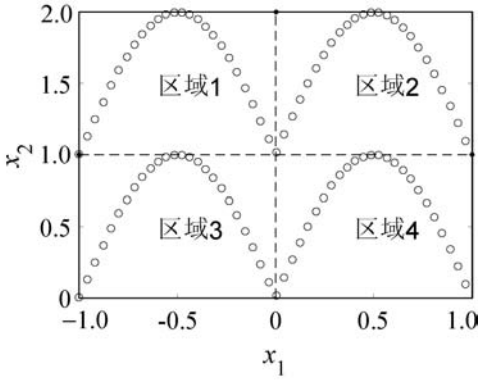


图 5　测试函数 MMF4 的 PS 分布

图 6 为算法迭代过程中,在每个子区域中解的比

例变化曲线。从图 6(a)中可以看出,本文算法中的区域 1、区域 2 所在的曲线随迭代次数增加而逐渐上升,而区域 3、区域 4 所在的曲线呈下降趋势。在迭代次数接近 50 时,四个区域的解的比例趋于 25%。在第 80 代至第 100 代时,每个区域的解的比例基本上接近 25%。这说明 RPMOPSO 具有良好的收敛性。对于 MO_Ring_PSO_SCD,图 6(b)中每个区域的解的比例最终能够收敛到 25% 左右,但是区域 1、区域 2 的解的比例略大于区域 3、区域 4 的解的比例,这说明其收敛性略差于 RPMOPSO。对于 DN-NSGAII,在第 30 代以后,区域 1、区域 4 的解的比例远低于区域 2、区域 3 的解的比例。对于 Omni-optimizer,在第 45 代以后,区域 2、区域 3 的解的比例低于区域 1、区域 4 的解的比例。在图 6(c)、(d)中,每个区域曲线的变化趋势为频繁震荡,并不稳定。这说明 DN-NSGAII 和 Omni-optimizer 算法表现出很差的收敛性。

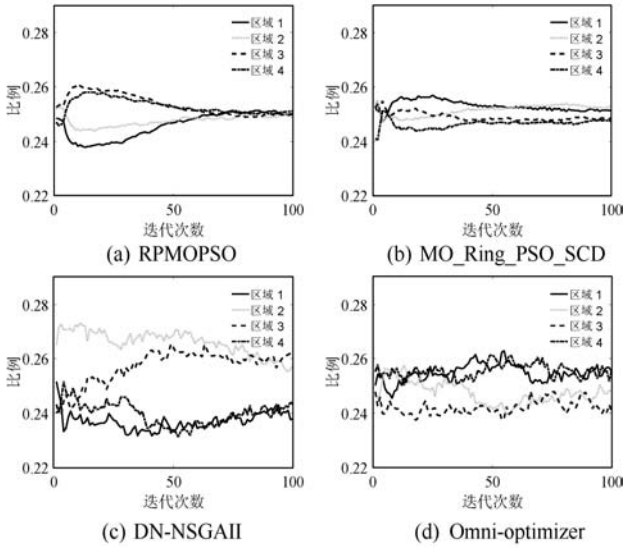


图 6 算法的收敛性对比

可以看出,RPMOPSO 在收敛性上具有良好的性能。

4 结 语

本文提出了一种基于参考点的多模态多目标粒子群算法(RPMOPSO)。采用了参考点策略和扩散机制,参考点策略的引入促使种群形成若干小生境,每个粒子在它的小生境邻域内独立进化,以提高种群的多样性。扩散机制能够改变粒子的飞行轨迹,扩散粒子到更大的搜索区域,极大地提高了粒子捕捉到更多 Pareto 解的可能性。通过和多模态多目标算法及多目标优化算法的比较,证明了 RPMOPSO 能够找到更多的 Pareto 解,且具有很好的收敛性。未来将尝试把算法应用于求解实际中的多模态多目标问题。

参 考 文 献

[1] Ali M, Siarry P, Pant M. An efficient differential evolution based algorithm for solving multi-objective optimization problems[J]. European Journal of Operational Research, 2018, 217(2): 404 – 416.

[1] 刘建昌,李飞,王洪海,等. 进化高维多目标优化算法研究综述[J]. 控制与决策,2018,33(5):879 – 887.

[3] Tian Y, Cheng R, Zhang X Y, et al. An indicator-based multi-objective evolutionary algorithm with reference point adaptation for better versatility[J]. IEEE Transactions on Evolutionary Computation, 2018, 22(4): 609 – 622.

[4] Elarbi M, Bechikh S, Gupta A, et al. A new decomposition-based NSGA-II for many-objective optimization[J]. IEEE Transactions on Systems Man and Cybernetics Systems, 2018, 48(7): 1191 – 1210.

[5] Shieh M D, Li Y, Yang C C. Comparison of multi-objective evolutionary algorithms in hybrid Kansei engineering system for product form design[J]. Advanced Engineering Informatics, 2018, 36(1): 31 – 42.

[6] Lwin K T, Rong Q, Maccarthy B L. Mean-VaR portfolio optimization: A nonparametric approach[J]. European Journal of Operational Research, 2017, 260(2): 751 – 766.

[7] Trivedi A, Srinivasan D, Sanyal K, et al. A survey of multiobjective evolutionary algorithms based on decomposition [J]. IEEE Transactions on Evolutionary Computation, 2017, 21(3): 440 – 462.

[8] 谢承旺,王志杰,魏波,等. 一种双链结构的多目标进化算法 DCMOEA[J]. 控制与决策,2015,30(4):577 – 584.

[9] Li M Q, Yang S X, Liu X H. Diversity comparison of Pareto front approximations in many-objective optimization [J]. IEEE Transactions on Cybernetics, 2017, 44 (12): 2568 – 2584.

[10] Yue C T, Qu B Y, Liang J. A multi-objective particle swarm optimizer using ring topology for solving multimodal multi-objective problems [J]. IEEE Transactions on Evolutionary Computation, 2018, 22(5): 805 – 817.

[11] Liu Y P, Yen G G, Gong D W. A multi-modal multi-objective evolutionary algorithm using two-archive and recombination strategies[J]. IEEE Transactions on Evolutionary Computation, 2019, 23(4): 660 – 674.

[12] Chiba K, Kanazaki M, Watanabe S. Structurisation and visualisation of design space for launch vehicle with hybrid rocket engine[J]. International Journal of Automation and Logistics, 2016, 2(1): 26 – 44.

[13] Han Y Y, Gong D W, Jin Y C, et al. Evolutionary multiobjective blocking lot-streaming flow shop scheduling with ma-

chine breakdowns[J]. IEEE Transactions on Cybernetics, 2019,49(1):184 – 197.

[14] Deb K, Abouhawwash M, Seada H. A computationally fast convergence measure and implementation for single-, multiple-, and many-objective optimization[J]. IEEE Transactions on Emerging Topics in Computational Intelligence, 2017, 1(4):280 – 293.

[15] Valdez S I, Botello-Aceves S, Becerra H M, et al. Comparison of a concurrent and a sequential optimization methodology for serial manipulators using metaheuristics[J]. IEEE Transactions on Industrial Informatics, 2018, 14(7):3155 – 3165.

[16] Liang J J, Yue C T, Qu B Y. Multimodal multi-objective optimization: A preliminary study[C]//2016 IEEE Congress on Evolutionary Computation(CEC). IEEE, 2016:2454 – 2461.

[17] Liang J J, Guo Q, Yue C. A self-organizing multi-objective particle swarm optimization algorithm for multimodal multi-objective problems[C]//Advances in Swarm Intelligence. Springer, 2018:550 – 560.

[18] Sengupta S, Basak S, Peters R. Particle swarm optimization: A survey of historical and recent developments with hybridization perspectives[J]. Machine Learning and Knowledge Extraction, 2018, 1(1):157 – 191.

[19] 李俊,罗阳坤,李波. 基于异维变异的差分混合粒子群算法[J]. 计算机科学,2018,45(5):208 – 214.

[20] Gong Y J, Li J J, Zhou Y C. Genetic learning particle swarm optimization[J]. IEEE Transactions on Cybernetics, 2017, 46(10):2277 – 2290.

[21] Huang G W, Cai Y G, Cai H. A time varying constrict factor PSO algorithm research[J]. Journal of Computational Methods in Sciences and Engineering Preprint, 2018, 18(3): 1 – 11.

[22] 张宇航,项铁铭,王建成. 基于维度变化的萤火虫优化算法[J]. 工业控制计算机,2017,30(3):20 – 21.

[23] Li Y P, Ni Z W, Jin F F, et al. Research on clustering method of improved glowworm algorithm based on good-point set[J]. Mathematical Problems in Engineering, 2018, 11(1):1 – 8.

[24] 费腾,张立毅,孙云山. 配送中心选址的自适应 Levy 分布混合变异鱼群算法[J]. 计算机应用与软件,2017,34(1):252 – 257.

[25] Kordestani J K, Firouzjaee H A, Meybodi M R. An adaptive bi-flight cuckoo search with variable nests for continuous dynamic optimization problems[J]. Applied Intelligence, 2018, 48(1):97 – 117.

[26] Jiang S W, Zhang J, Ong Y S. A simple and fast hypervolume indicator-based multi-objective evolutionary algorithm[J]. IEEE Transactions on Cybernetics, 2017, 45(10):2202 – 2213.

(上接第 191 页)

[5] Li B, Pei Y, Wu H, et al. Heuristics to allocate high-performance cloudlets for computation offloading in mobile ad hoc clouds[J]. Journal of Supercomputing, 2015, 71(8): 3009 – 3036.

[6] Zhou B, Dastjerdi A V, Calheiros R N, et al. A context sensitive offloading scheme for mobile cloud computing service[C]//2015 IEEE 8th International Conference on Cloud Computing, 2015.

[7] Zhao X, Hung W N N, Yang Y, et al. Optimizing communication in mobile ad hoc network clustering[J]. Computers in Industry, 2013, 64(7):849 – 853.

[8] Guan Z, Melodia T. The value of cooperation: minimizing user costs in multi-broker mobile cloud computing networks[J]. IEEE Transactions on Cloud Computing, 2017, 5(4): 780 – 791.

[9] Liu F, Shu P, Lui J C S. AppATP: an energy conserving adaptive Mobile-Cloud transmission protocol[J]. IEEE Transactions on Computers, 2015, 64(11):3051 – 3063.

[10] Shah-Mansouri H, Wong V W S, Schober R. Joint optimal pricing and task scheduling in mobile cloud computing systems[J]. IEEE Transactions on Wireless Communications, 2017, 16(8):5218 – 5232.

[11] Yaqoob I, Ahmed E, Gani A, et al. Heterogeneity-aware task allocation in mobile ad hoc cloud[J]. IEEE Access, 2017, 5:1779 – 1795.

[12] Zhang Y, Niyato D, Wang P. Offloading in mobile cloudlet systems with intermittent connectivity[J]. IEEE Transactions on Mobile Computing, 2015, 14(12):2516 – 2529.

[13] Guo X, Liu L, Chang Z, et al. Data offloading and task allocation for cloudlet-assisted ad hoc mobile clouds[J]. Wireless Networks, 2018, 24(1):1 – 10.

[14] Zhou B, Dastjerdi A V, Calheiros R N, et al. MCloud: a context-aware offloading framework for heterogeneous mobile cloud[J]. IEEE Transactions on Services Computing, 2017, 10(5):797 – 810.

[15] Shi T, Yang M, Li X, et al. An energy-efficient scheduling scheme for time-constrained tasks in local mobile clouds[J]. Pervasive & Mobile Computing, 2016, 27:90 – 105.

[16] Karaoglu B, Heinzelman W. Cooperative load balancing and dynamic channel allocation for cluster-based mobile ad hoc networks[J]. Mobile Computing IEEE Transactions on, 2015, 14(5):951 – 963.

[17] Liu Y G, Cui Q, Zhang M, et al. Cloud computing task scheduling strategy based on genetic algorithm[J]. Information Technology, 2017, 8:177 – 180.

[18] Yousafzai A, Chang V, Gani A, et al. Directory-based incentive management services for ad-hoc mobile clouds[J]. International Journal of Information Management, 2016, 36(6):900 – 906.