控制与决策
Control and Decision

基于分解的多目标多因子进化算法

么双双， 董志明， 王显鹏

引用本文:
么双双, 董志明, 王显鹏. 基于分解的多目标多因子进化算法[J]. 控制与决策, 2021, 36(3): 637−644.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2019.0525

基于分解的多目标多因子进化算法

么双双[1], 董志明[2], 王显鹏[2,3†]

(1. 东北大学 信息科学与工程学院，沈阳 110004；2. 智能工业数据解析与优化教育部重点实验室，沈阳 110004；3. 辽宁省智能工业数据解析与优化工程实验室，沈阳 110004)

摘 要: 多目标多因子优化(MO-MFO)问题作为一类新的优化问题近年来受到了众多关注,其特点是需要利用单个种群来同时优化多个多目标优化任务. 针对该问题, 提出一个基于分解策略的多目标多因子进化算法(MFEA/D). 算法通过多组权重向量,将MO-MFO问题中的每个任务分解成一系列单目标优化子问题,并用单个种群同时优化. 在种群进化过程中提出不同任务之间的信息交流策略,以充分挖掘不同任务之间的有用信息,进而加快每个任务的收敛速度. 基于10个多目标多因子标准测试问题的实验结果表明,所提出的不同任务之间的信息交流策略能够加快问题的求解速度,使得MFEA/D算法显著优于当前的MO-MFEA算法.

关键词: 多目标优化；多因子优化；单个种群；分解；信息交流；进化算法

中图分类号: TP273 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2019.0525 **开放科学(资源服务)标识码(OSID):**

引用格式: 么双双,董志明,王显鹏. 基于分解的多目标多因子进化算法[J]. 控制与决策, 2021, 36(3): 637-644.

A multiobjective multifactorial evolutionary algorithm based on decomposition

YAO Shuang-shuang[1], *DONG Zhi-ming*[2], *WANG Xian-peng*[2,3†]

(1. College of Information Science and Engineering，Northeastern University，Shenyang 110004，China；2. Key Laboratory of Data Analytics and Optimization for Smart Industry，Ministry of Education，Shenyang 110004，China；3. Liaoning Engineering Laboratory of Operations Analytics and Optimization for Smart Industry，Shenyang 110004，China)

Abstract: As a new kind of optimization problems, multiobjective multifactorial optimization (MO-MFO) problems have attracted much attention in recent years. Its main feature is that only a single population can be used to optimize multiple multiobjective optimization tasks simultaneously. In this paper, a multifactorial evolutionary algorithm based on decomposition (MFEA/D) strategy is proposed. It decomposes a MO-MFO problem into a series of single objective optimization subproblems through multiple sets of weight vectors, and optimizes them simultaneously with a single population. In the process of population evolution, the information communication strategy between different tasks with a certain probability is proposed to mine useful information between different tasks, so as to accelerate the convergence rate of each task. The statistical analysis of the experiment results on 10 benchmark MO-MFO problems illustrates that the information communication strategy can help to accelerate the convergence rate and that the proposed MFEA/D strategy performs significantly better than the state-of-the-art MO-MFEAs in the literature.

Keywords: multiobjective optimization；multifactorial optimization；single population；decomposition；information communication；evolutionary algorithm

0 引 言

多因子优化(multifactorial optimization, MFO)通过单个种群,利用进化策略来同时优化多个任务[1]. 多因子优化背后的关键动机是,如果被优化的多个任务的最优解之间或者不同任务特征之间存在关联,则可以通过不同任务之间的信息迁移加快每个任务的搜索速度. 特别地,在实际工程问题求解中,知识的迁移和重用是很常规的[2]. 在多因子优化中,由于每一个任务都可以看作是影响种群进化过程的一个因子,K个不同任务优化问题也称作K因子问题.

多因子优化的范例自从被Gupta等[1]提出以来,就得到了广泛的关注,并应用到多个领域. Gupta等[3]

收稿日期: 2019-04-25；**修回日期:** 2019-11-05.

基金项目: 国家重点研发计划项目(2018YFB1700404)；国家自然科学基金项目(61573086, 71790614, 71621061)；教育部111创新引智基地项目(B16009).

†通讯作者. E-mail: wangxianpeng@ise.neu.edu.cn.

将一个双层优化问题转化成多任务优化问题,并用多因子算法进行求解;Chandra等[4]通过多因子优化的方法解决多步混沌时间序列预测问题.针对实际应用程序中多个稀疏重建任务需要被同时优化的问题,Li等[5]提出一种基于多目标进化算法的稀疏重构方法,Tang等[6]采用进化多任务的模块化训练方法对极限学习机的模块化拓扑结构进行优化.

多目标多任务问题指的是每个任务都是一个多目标优化问题.一个简单的求解多目标多任务问题的方式是将每一个任务割裂出来,分别优化.但是,在相似的环境中,不同任务之间往往存在隐式的关联.因此,发掘不同任务之间的关联,利用任务之间的潜在信息来加速各个任务的求解是有必要的.

基于分解的多目标进化算法在求解多目标优化问题时性能表现非常优越[7-13].然而,基于分解策略来求解MO-MFO问题目前还没有这方面的尝试.本文提出一个基于分解策略的多目标多因子算法.首先,将每个多目标优化任务通过分解策略分解成一系列单目标优化子问题.然后,通过一个种群去进化这些由多个任务分解成的所有单目标子问题.在种群进化过程中,为了加快问题的求解速度,提出一种不同任务之间的信息交流策略.最后,基于10个两任务标准测试问题对所提出的算法进行测试,结果表明本文提出的算法比MO-MFEA[2]更具有竞争性.

1 问题描述

1.1 多目标问题

一般而言,一个最小化所有目标的多目标问题可以表述成如下形式[14]:

$$\begin{aligned}&\min\ \boldsymbol{F}(\boldsymbol{x})=(f_1(\boldsymbol{x}),f_2(\boldsymbol{x}),\dots,f_M(\boldsymbol{x}));\\&\text{s.t. }\boldsymbol{x}\in\Omega.\end{aligned}\tag{1}$$

其中:决策空间$\Omega\subset\boldsymbol{R}^D$,$\boldsymbol{x}=(x_1,x_2,\dots,x_D)\in\Omega$表示一个$D$维的决策变量,$\boldsymbol{F}:\Omega\to\boldsymbol{R}^M$表示$M$个存在冲突关系的目标函数.

多目标优化的目的是找到一组具有代表性的Pareto前沿(Pareto front, PF)来帮助决策者了解目标之间的冲突关系,进而作出最优的权衡.

1.2 多目标多任务问题

不失一般性,一个多目标多任务优化问题可以表述成如下形式[2]:

$$\begin{aligned}&\{x_1,x_2,\dots,x_K\}=\\&\arg\min\{\boldsymbol{F}_1(\boldsymbol{x}),\boldsymbol{F}_2(\boldsymbol{x}),\dots,\boldsymbol{F}_K(\boldsymbol{x})\};\\&\text{s.t. }x_i\in\boldsymbol{X}_i,\ i\in\{1,2,\dots,K\}.\end{aligned}\tag{2}$$

其中:K表示任务个数;$\boldsymbol{x}_i$表示任务i的Pareto最优解集(Pareto set, PS);$\boldsymbol{X}_i\subset\boldsymbol{R}^{D_i}$表示任务$i$的决策空间,$D_i$表示第$i$个任务决策变量的长度;$\boldsymbol{F}_i:\boldsymbol{X}_i\to\boldsymbol{R}^{M_i}$表示任务$i$中$M_i$个存在冲突关系的目标函数,$i\in\{1,2,\dots,K\}$.

1.3 多因子优化

多因子优化的重要特点是通过单个种群去优化多个任务.因为每个任务都可以看作是影响整个种群进化过程的一个因子,所以这些问题组合在一起也称作K因子问题.对于种群内个体p_i,一些定义如下所示.

定义1 因子排名.因子排名r_j^i指的是个体p_i在任务T_j中按照优先级降序排列后的索引值.

定义2 技能因子.技能因子τ_i指的是被个体p_i匹配的任务索引.如果个体p_i被所有的任务评价,则$\tau_i=\arg\min\limits_{j\in\{1,2,\dots,K\}}\{r_j^i\}$.

定义3 标量适应性值.个体p_i的标量适应性值可以表示成$\varphi_i=1/r_{\tau_i}^i$.

与传统的基于种群的进化算法相比,多因子优化进化算法的单个种群产生方式有所不同.因为每个任务的变量维度、各个变量的约束范围可能不尽相同,所以需要一个“统一空间”$\boldsymbol{Y}$来编码整个种群[1].这里,$\boldsymbol{Y}$的维度$D^*=\max\{D_1,D_2,\dots,D_K\}$,每一维约束范围为[0,1].在统一空间内产生的任何个体都可以表示成$p_i=\{y_1,y_2,\dots,y_{D^*}\}$,$y_i$为在[0,1]之间随机产生的数值.但是,需要注意的是,对于一个确定的任务T_k,当其去评价个体p_i时,需要通过下面公式将p_i每一维的变量映射到原决策空间:

$$x_j=L_j+(U_j-L_j)\times y_j,\tag{3}$$

其中U_j和L_j表示任务T_k的第j个变量的上下界.

2 基于分解的多目标多因子进化算法

本文提出的基于分解的多目标多因子进化算法流程如下所示.

算法1 MFEA/D算法流程.

输入:多任务问题$T=\{T_1,T_2,\dots,T_K\}$,种群大小N,邻域大小n_e,替换个数n_r,邻域选择概率δ,随机匹配概率rmp,停止准则;

输出:每一个任务的近似Pareto最优解.

step 1:初始化.

step 1.1:使用Das & Dennis法[15]给每个任务T_i产生一系列权重向量Λ_i,权重向量个数为N/K;

step 1.2:根据欧氏距离,初始化每个任务内各个子问题的邻域B_i,子问题邻域大小为n_e;

step 1.3:在统一空间内初始化种群$P=\{p_1,p_2,\dots,p_N\}$,并且使得这些个体均匀地匹配给每个任务;

step 1.4:初始化每个任务的理想点$\boldsymbol{z}_i$.

step 2: 更新.

step 2.1: 令$i=1$.

step 2.2: 获取个体p_i的技能因子τ_i,并令$k=\tau_i$.

step 2.3: 在[0,1]内产生一个随机数r_1,根据下面条件选择候选亲本:

$$E=\begin{cases}\{p_j|p_j\in B_k(p_i)\},\ r_1<\delta;\\ \{p_j|p_j\in P,\ \forall\tau_j=k\},\ \text{otherwise.}\end{cases}$$

从E内随机选出3个不同个体x^{r_1}、x^{r_2}、x^{r_3},使用DE操作算子产生一个子代x',并使用多项式变异算子执行x',得到x^{new}.

step 2.4: 修复x^{new}的每一维变量,使其满足统一空间范围内的约束.

step 2.5: 在[0,1]内产生一个随机数r_2,如果$r_2<\text{rmp}$,则在$\{1,2,\dots,K\}$内随机产生一个整数k',且$k'\neq\tau_i$,令$k=k'$,$E=\{p_j|p_j\in P,\ \forall\tau_j=k\}$.

step 2.6: 设置子代x^{new}的技能因子为k.

step 2.7: 由任务T_k评价子代x^{new},并更新理想点$\boldsymbol{z}_k$.

step 2.8: 令$c=0$,按照以下步骤更新E:

1) 如果$c=n_r$或者$E=\varnothing$,转至step 2.9,否则从E中随机选出一个个体x_j;

2) 如果$g(x^{\text{new}}|\lambda_j,\boldsymbol{z}_k)\leqslant g(x_j|\lambda_j,\boldsymbol{z}_k)$,$x_j=x^{\text{new}}$,$c=c+1$;

3) 从E中删除x_j,返回1).

step 2.9: $i=i+1$,如果$i>N$,则转至step 3,否则返回step 2.2.

step 3: 停止准则. 如果终止条件满足,则输出种群P;否则,返回step 2.

这里需要注意的是,在step 2.3中,产生子代x'的方式与MOEA/D-DE中个体产生方式相同,也就是使用DE-rand/1/bin[16]操作算子产生子代x',再通过多项式变异算子操作得到x^{new}:

$$x'_j=\begin{cases}x_j^{r_1}+F\times(x_j^{r_2}-x_j^{r_3}),\ \text{rand}<\text{Cr};\\ x_j^{r_1},\ \text{otherwise.}\end{cases}\tag{4}$$

$$x_j^{\text{new}}=\begin{cases}x'_j+\sigma_j\times(b_j-a_j),\ \text{rand}<p_m;\\ x'_j,\ \text{otherwise.}\end{cases}\tag{5}$$

$$\sigma_j=\begin{cases}(2\times\text{rand})^{\frac{1}{\eta+1}}-1,\ \text{rand}<0.5;\\ 1-(2-2\times\text{rand})^{\frac{1}{\eta+1}},\ \text{otherwise.}\end{cases}\tag{6}$$

由于Tchebychef标量化函数对问题PF形状的凹凸性不敏感,在step 2.8,第2个判断节点中,评价函数采用的是Tchebycheff函数[14],定义如下所示:

$$g^{\text{tch}}(\boldsymbol{x}|\lambda,z^*)=\min\ \max\ w_i|f_i(x)-z_i^*|.\tag{7}$$

其中:$\lambda=\{w_1,w_2,\dots,w_M\}$,$w_i\geqslant 0$,$\sum_{i=1}^{M}w_i=1$.

不同于传统的基于分解的多目标进化算法[7-8],MFEA/D需要优化多个不同的多目标任务,所以在初始化(step 1)阶段需要初始化多个系列的权重向量,并且根据任务的数量,将种群均匀地关联给每一个任务. 在进化过程中,每次选取种群内的一个个体,与MOEA/D-DE类似,依据概率δ来决定父代解来源范围. 当父代解来源于该个体的邻域时,则进化更侧重于任务之间的“开发”(exploitation);当父代解来源范围为与该个体技能因子相同的种群时,则进化更侧重于单个任务的“探索”(exploration). 随机匹配概率rmp的存在使得不同任务之间能够进行信息交流. 如step 2.5所示,简单的表述就是由关联到任务A的个体作为亲本产生的子代被用来更新任务B,从而挖掘不同任务之间潜在联系,加速优化进程.

3　实验及结果分析

实验以MO-MFEA[2]算法为比较算法来分析本文提出算法的性能. 由于MFEA/D同样基于分解策略,本节以MOEA/D-DE作为比较算法作进一步比较实验来分析MFEA/D的性能(因为当只有一个任务时,MOEA/D-DE算法就是MFEA/D的确切形式). 本文提出的MFEA/D算法以及MO-MFEA、MOEA/D-DE均是由JAVA语言,在jMetal[17]框架下实现. 所有实验是在硬件配置为Intel(R) Core(TM) i7-7700K CPU @ 4.20 GHz, 24 GB RAM的个人电脑上完成.

3.1　测试函数与参数设置

10个两任务基准测试问题(CEC19复杂MO-MFO标准测试问题)被用来测试所提出的算法性能. 它们分别是CPLX1(F1+F2), CPLX2(F1+F7), CPLX3(F2 + F4), CPLX4(F2 + F9), CPLX5(F3 + F6), CPLX6(F3 + F9), CPLX7(F4 + F5), CPLX8(F5 + F7), CPLX9(F6 + F9)以及CPLX10(F7 + F8). 这些两任务基准测试问题每个任务都是由文献[10]中提出的具有复杂Pareto最优解集的多目标测试问题构成(LZ09测试问题). 其中,这些问题的目标个数、变量个数以及变量的约束范围如表1所示.

为了公平起见,所有问题的近似Pareto最优解个数最多为105,终止条件为目标函数评价次数EFs = 105 000(每个任务进化1 000代). 这里需要注意的是,一次目标函数评价是指对某一个任务的评价. 对于每个测试函数,每个算法均独立运行31次.

表 1 LZ09测试问题

problem	目标个数(M)	变量个数(D)	约束范围
F1	2	10	$[0,1]^D$
F2	2	30	$[0,1]^D$
F3	2	30	$[0,1]^D$
F4	2	30	$[0,1]^D$
F5	2	30	$[0,1]^D$
F6	3	10	$[0,1]^D$
F7	2	10	$[0,1]^D$
F8	2	10	$[0,1]^D$
F9	2	30	$[0,1]^D$

MO-MFEA算法[2]以及MOEA/D-DE算法[8]内部参数设置与原文献中设置相同. 在MFEA/D中,多项式变异算子中的参数设置与MO-MFEA中相同,对于每一个子问题,其邻域大小n_e为10;最大替换个数n_r为2;邻域选择概率δ为0.9;随机匹配概率rmp为0.1. 除此之外,DE操作被用来产生新个体,这里DE中的常量因子F设置为0.5,交叉常量Cr设置为0.9.

3.2 性能评价指标

多样性和收敛性是衡量一个算法获得的近似Pareto前沿是否具有竞争性的评价标准. 其中,超体积(hypervolume, HV)[18]和反向世代距离(inverted generational distance, IGD)[19]是两种常用的综合性能评价指标,它们都能标量化地反应算法的性能. 针对某一个测试问题,HV指标越大,表明该算法获得的近似Pareto前沿越能够代表该问题的真实Pareto前沿,而IGD指标反之. 超体积和反向时代距离的计算公式如下所示:

1) 超体积

$$\mathrm{HV}(\mathrm{PS}) = \mathrm{volume}(\bigcup_{s\in \mathrm{PS}} v_s),$$

$$v_s = [f_1(\boldsymbol{s}),\ z_1^{\mathrm{nad}}] \times \ldots \times [f_M(\boldsymbol{s}),\ z_M^{\mathrm{nad}}]. \tag{8}$$

PS为算法获取的近似Pareto最优解集,z_i^{nad}为真实Pareto前沿中第i个目标的最差值,$i \in \{1,2,\ldots,M\}$.

2) 反向世代距离

$$\mathrm{IGD}(\mathrm{PF},\mathrm{PF}^*) = \frac{1}{|\mathrm{PF}|}\sqrt{\sum_{i=1}^{|\mathrm{PF}|} d(p_i,\mathrm{PF}^*)^2},$$

$$d(p_i,\mathrm{PF}^*) = \min_{j=1}^{|\mathrm{PF}^*|} ||p_i - p_j^*||. \tag{9}$$

这里: PF为算法获得的近似Pareto前沿, PF*为真实Pareto前沿,p_i和p_j^*分别为PF和PF*中元素.

3.3 结果分析

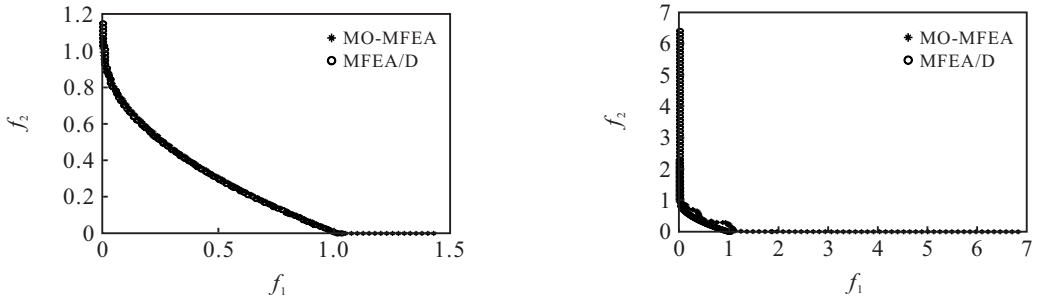
HV指标和IGD指标的中位数和四分位间距被用来统计分析算法性能. 对于每一个任务,T检验被

表 2 HV和IGD指标的中位数和四分位间距(MO-MFEA vs. MFEA/D)

problem	task No.	HV		IGD	
		MO-MFEA	MFEA/D	MO-MFEA	MFEA/D
CPLX1	T1	$6.511\text{e-}01_{9.5\text{e-}04}{}^{+}$	$\mathbf{6.607\text{e-}01_{1.7\text{e-}04}}$	$4.425\text{e-}04_{2.6\text{e-}05}{}^{+}$	$\mathbf{2.256\text{e-}04_{1.9\text{e-}05}}$
	T2	$5.506\text{e-}01_{1.5\text{e-}02}{}^{+}$	$\mathbf{6.449\text{e-}01_{4.4\text{e-}03}}$	$5.179\text{e-}03_{1.1\text{e-}03}{}^{+}$	$\mathbf{7.139\text{e-}04_{1.5\text{e-}04}}$
CPLX2	T1	$6.510\text{e-}01_{1.4\text{e-}03}{}^{+}$	$\mathbf{6.607\text{e-}01_{9.3\text{e-}05}}$	$4.450\text{e-}04_{2.7\text{e-}05}{}^{+}$	$\mathbf{2.227\text{e-}04_{9.5\text{e-}06}}$
	T2	$4.703\text{e-}01_{7.8\text{e-}02}{}^{+}$	$\mathbf{6.569\text{e-}01_{1.9\text{e-}03}}$	$8.637\text{e-}03_{3.0\text{e-}03}{}^{+}$	$\mathbf{3.928\text{e-}04_{1.5\text{e-}04}}$
CPLX3	T1	$5.610\text{e-}01_{2.6\text{e-}02}{}^{+}$	$\mathbf{6.450\text{e-}01_{4.3\text{e-}03}}$	$4.872\text{e-}03_{9.1\text{e-}04}{}^{+}$	$\mathbf{7.079\text{e-}04_{2.0\text{e-}04}}$
	T2	$6.345\text{e-}01_{7.1\text{e-}03}{}^{+}$	$\mathbf{6.498\text{e-}01_{3.5\text{e-}03}}$	$2.787\text{e-}03_{1.8\text{e-}03}{}^{+}$	$\mathbf{8.004\text{e-}04_{1.9\text{e-}04}}$
CPLX4	T1	$5.685\text{e-}01_{2.6\text{e-}02}{}^{+}$	$\mathbf{6.519\text{e-}01_{7.5\text{e-}04}}$	$3.958\text{e-}03_{9.7\text{e-}04}{}^{+}$	$\mathbf{4.120\text{e-}04_{2.3\text{e-}05}}$
	T2	$2.398\text{e-}01_{2.8\text{e-}02}{}^{+}$	$\mathbf{3.185\text{e-}01_{1.2\text{e-}03}}$	$4.428\text{e-}03_{9.2\text{e-}04}{}^{+}$	$\mathbf{3.906\text{e-}04_{3.4\text{e-}05}}$
CPLX5	T1	$6.257\text{e-}01_{1.1\text{e-}02}{}^{+}$	$\mathbf{6.492\text{e-}01_{3.6\text{e-}03}}$	$2.685\text{e-}03_{1.4\text{e-}03}{}^{+}$	$\mathbf{8.064\text{e-}04_{2.5\text{e-}04}}$
	T2	$2.370\text{e-}01_{4.4\text{e-}02}{}^{+}$	$\mathbf{3.054\text{e-}01_{2.7\text{e-}02}}$	$5.553\text{e-}03_{1.2\text{e-}03}{}^{+}$	$\mathbf{3.546\text{e-}03_{8.9\text{e-}04}}$
CPLX6	T1	$6.244\text{e-}01_{8.5\text{e-}03}{}^{+}$	$\mathbf{6.495\text{e-}01_{3.8\text{e-}03}}$	$2.694\text{e-}03_{1.6\text{e-}03}{}^{+}$	$\mathbf{7.863\text{e-}04_{2.3\text{e-}04}}$
	T2	$2.233\text{e-}01_{1.7\text{e-}02}{}^{+}$	$\mathbf{3.008\text{e-}01_{1.2\text{e-}02}}$	$5.293\text{e-}03_{9.6\text{e-}04}{}^{+}$	$\mathbf{1.143\text{e-}03_{7.2\text{e-}04}}$
CPLX7	T1	$6.371\text{e-}01_{4.6\text{e-}03}{}^{+}$	$\mathbf{6.503\text{e-}01_{4.6\text{e-}03}}$	$2.359\text{e-}03_{1.2\text{e-}03}{}^{+}$	$\mathbf{7.892\text{e-}04_{2.3\text{e-}04}}$
	T2	$6.320\text{e-}01_{1.3\text{e-}02}{}^{+}$	$\mathbf{6.453\text{e-}01_{6.2\text{e-}03}}$	$2.238\text{e-}03_{1.5\text{e-}03}{}^{+}$	$\mathbf{1.024\text{e-}03_{4.3\text{e-}04}}$
CPLX8	T1	$6.288\text{e-}01_{8.4\text{e-}03}{}^{+}$	$\mathbf{6.414\text{e-}01_{6.2\text{e-}03}}$	$2.399\text{e-}03_{1.3\text{e-}03}{}^{+}$	$\mathbf{1.193\text{e-}03_{4.0\text{e-}04}}$
	T2	$4.776\text{e-}01_{8.4\text{e-}02}{}^{+}$	$\mathbf{6.576\text{e-}01_{2.6\text{e-}03}}$	$1.211\text{e-}02_{9.0\text{e-}03}{}^{+}$	$\mathbf{3.649\text{e-}04_{1.2\text{e-}04}}$
CPLX9	T1	$2.376\text{e-}01_{2.7\text{e-}02}{}^{+}$	$\mathbf{2.943\text{e-}01_{1.7\text{e-}02}}$	$5.490\text{e-}03_{1.1\text{e-}03}{}^{+}$	$\mathbf{3.935\text{e-}03_{9.3\text{e-}04}}$
	T2	$2.250\text{e-}01_{5.9\text{e-}02}{}^{+}$	$\mathbf{3.010\text{e-}01_{5.9\text{e-}03}}$	$5.132\text{e-}03_{4.1\text{e-}03}{}^{+}$	$\mathbf{1.061\text{e-}03_{3.8\text{e-}04}}$
CPLX10	T1	$4.761\text{e-}01_{5.9\text{e-}02}{}^{+}$	$\mathbf{6.569\text{e-}01_{3.1\text{e-}03}}$	$9.849\text{e-}03_{6.8\text{e-}03}{}^{+}$	$\mathbf{4.694\text{e-}04_{2.4\text{e-}04}}$
	T2	$5.245\text{e-}01_{6.2\text{e-}02}{}^{+}$	$\mathbf{6.514\text{e-}01_{3.1\text{e-}03}}$	$6.636\text{e-}03_{4.8\text{e-}03}{}^{+}$	$\mathbf{5.535\text{e-}04_{2.6\text{e-}04}}$
$+/=/-$		20/0/0		20/0/0	

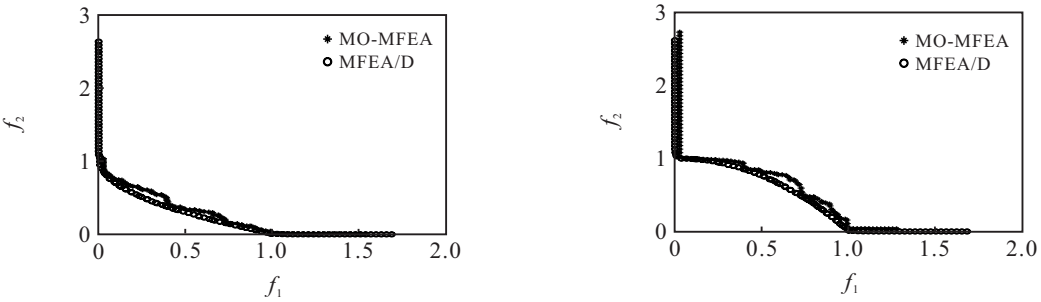
用来分析算法MFEA/D和MO-MFEA是否有显著性差异. 这里,显著性水平为0.05,符号“+”、“=”以及“−”分别表示MFEA/D显著优于、无显著差异、显著劣于MO-MFEA. 如表2所示,对于所有测试问题的每个任务,无论是HV指标还是IGD指标,MFEA/D算法取得的结果都要显著优于MO-MFEA(这里:四分位间距在中位数下角标位置表示,黑色字体表示该指标更好).

为了可视化地比较多次运行后MFEA/D与MO-MFEA之间的性能差异,中值达到平面(median attainment surface)[20]被用来作为比较的工具. 这里以测试问题CPLX2、CPLX4、CPLX6、CPLX10为例,算法MFEA/D和MO-MFEA获得的中值达到平面如图1～图4所示.



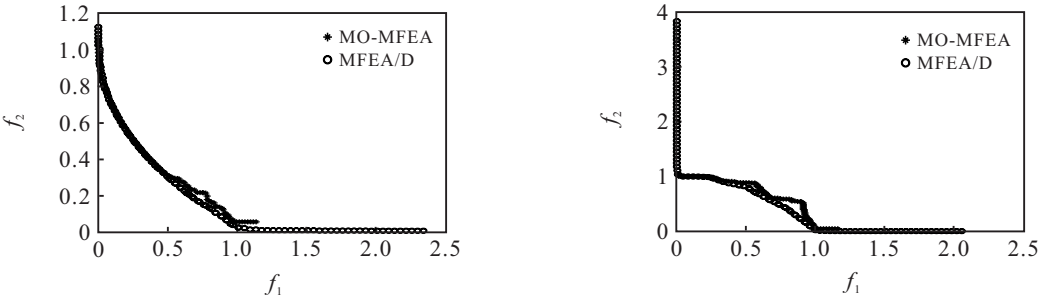
(a) CPLX2-T1　　(b) CPLX2-T2

图 1　CPLX2:中值达到平面



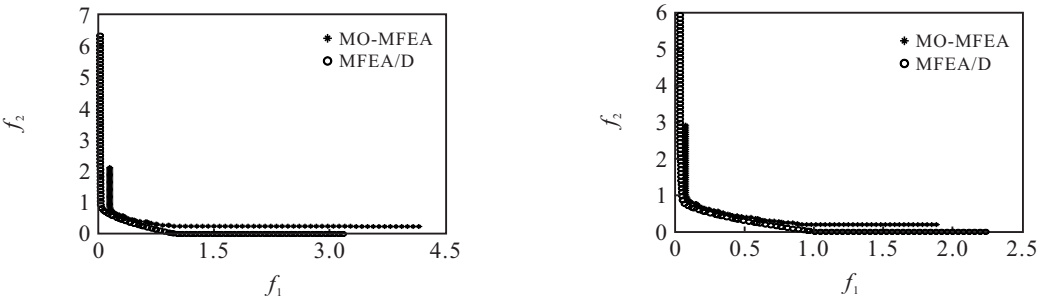
(a) CPLX4-T1　　(b) CPLX4-T2

图 2　CPLX4:中值达到平面



(a) CPLX6-T1　　(b) CPLX6-T2

图 3　CPLX6:中值达到平面



(a) CPLX10-T1　　(b) CPLX10-T2

图 4　CPLX10:中值达到平面

表3 运行时间比较 单位: s

problem	CPLX1	CPLX2	CPLX3	CPLX4	CPLX5	CPLX6	CPLX7	CPLX8	CPLX9	CPLX10
MO-MFEA	1.952	1.630	2.195	2.219	2.149	2.222	2.189	2.143	2.215	1.709
MFEA/D	1.324	0.963	1.499	1.508	1.409	1.522	1.540	1.411	1.436	0.968

表4 HV和IGD指标的中位数和四分位间距(MOEA/D-DE vs. MFEA/D)

problem	task No.	HV		IGD	
		MOEA/D-DE	MFEA/D	MOEA/D-DE	MFEA/D
CPLX1	T1	**6.608e-01**$_{\mathbf{1.4e-04}}^{-}$	6.607e-01$_{1.7e-04}$	**2.234e-04**$_{\mathbf{5.0e-06}}^{-}$	2.256e-04$_{1.9e-05}$
	T2	**6.472e-01**$_{\mathbf{9.6e-03}}^{=}$	6.449e-01$_{4.4e-03}$	**6.416e-04**$_{\mathbf{3.4e-04}}^{=}$	7.139e-04$_{1.5e-04}$
CPLX2	T1	**6.608e-01**$_{\mathbf{2.1e-04}}^{-}$	6.607e-01$_{9.3e-05}$	2.247e-04$_{5.9e-06}^{=}$	**2.227e-04**$_{\mathbf{9.5e-06}}$
	T2	6.524e-01$_{5.5e-03}^{+}$	**6.569e-01**$_{\mathbf{1.9e-03}}$	8.047e-04$_{3.8e-04}^{+}$	**3.928e-04**$_{\mathbf{1.5e-04}}$
CPLX3	T1	**6.467e-01**$_{\mathbf{5.8e-03}}^{=}$	6.450e-01$_{4.3e-03}$	**6.212e-04**$_{\mathbf{2.8e-04}}^{=}$	7.079e-04$_{2.0e-04}$
	T2	6.470e-01$_{2.9e-03}^{+}$	**6.498e-01**$_{\mathbf{3.5e-03}}$	1.007e-03$_{2.0e-04}^{+}$	**8.004e-04**$_{\mathbf{1.9e-04}}$
CPLX4	T1	6.465e-01$_{6.9e-03}^{+}$	**6.519e-01**$_{\mathbf{7.5e-04}}$	7.191e-04$_{3.3e-04}^{+}$	**4.120e-04**$_{\mathbf{2.3e-05}}$
	T2	3.091e-01$_{1.2e-02}^{+}$	**3.185e-01**$_{\mathbf{1.2e-03}}$	9.076e-04$_{4.2e-04}^{+}$	**3.906e-04**$_{\mathbf{3.4e-05}}$
CPLX5	T1	6.473e-01$_{2.1e-02}^{+}$	**6.492e-01**$_{\mathbf{3.6e-03}}$	9.484e-04$_{2.1e-03}^{+}$	**8.064e-04**$_{\mathbf{2.5e-04}}$
	T2	**3.184e-01**$_{\mathbf{3.8e-02}}^{-}$	3.054e-01$_{2.7e-02}$	**3.243e-03**$_{\mathbf{1.0e-03}}^{=}$	3.546e-03$_{8.9e-04}$
CPLX6	T1	6.418e-01$_{5.4e-02}^{+}$	**6.495e-01**$_{\mathbf{3.8e-03}}$	1.632e-03$_{5.5e-03}^{+}$	**7.863e-04**$_{\mathbf{2.3e-04}}$
	T2	**3.054e-01**$_{\mathbf{1.0e-02}}^{=}$	3.008e-01$_{1.2e-02}$	**9.693e-04**$_{\mathbf{4.1e-04}}^{=}$	1.143e-03$_{7.2e-04}$
CPLX7	T1	6.446e-01$_{3.8e-03}^{+}$	**6.503e-01**$_{\mathbf{4.6e-03}}$	1.115e-03$_{2.3e-04}^{+}$	**7.892e-04**$_{\mathbf{2.3e-04}}$
	T2	6.380e-01$_{7.9e-03}^{+}$	**6.453e-01**$_{\mathbf{6.2e-03}}$	1.440e-03$_{3.4e-04}^{+}$	**1.024e-03**$_{\mathbf{4.3e-04}}$
CPLX8	T1	6.374e-01$_{7.3e-03}^{+}$	**6.414e-01**$_{\mathbf{6.2e-03}}$	1.489e-03$_{3.4e-04}^{+}$	**1.193e-03**$_{\mathbf{4.0e-04}}$
	T2	6.518e-01$_{1.0e-02}^{+}$	**6.576e-01**$_{\mathbf{2.6e-03}}$	7.607e-04$_{4.1e-04}^{+}$	**3.649e-04**$_{\mathbf{1.2e-04}}$
CPLX9	T1	**3.221e-01**$_{\mathbf{4.3e-02}}^{-}$	2.943e-01$_{1.7e-02}$	**3.584e-03**$_{\mathbf{1.7e-03}}^{=}$	3.935e-03$_{9.3e-04}$
	T2	**3.044e-01**$_{\mathbf{1.1e-02}}^{=}$	3.010e-01$_{5.9e-03}$	**1.051e-03**$_{\mathbf{4.5e-04}}^{=}$	1.061e-03$_{3.8e-04}$
CPLX10	T1	6.540e-01$_{5.2e-03}^{=}$	**6.569e-01**$_{\mathbf{3.1e-03}}$	6.298e-04$_{3.6e-04}^{=}$	**4.694e-04**$_{\mathbf{2.4e-04}}$
	T2	4.951e-01$_{7.7e-02}^{+}$	**6.514e-01**$_{\mathbf{3.1e-03}}$	5.955e-03$_{5.2e-03}^{+}$	**5.535e-04**$_{\mathbf{2.6e-04}}$
+/=/−		11/5/4		11/8/1	

从图中可以看出,对于每一个任务,MFEA/D获得的结果更靠前一些,也就是说,MO-MFEA的中值达到平面被MFEA/D的中值达到平面包围,进而说明,针对这些标准测试问题,MFEA/D更具有竞争性. 同时,对于这10组测试问题,算法MFEA/D和MO-MFEA每个测试问题平均求解时间如表3所示. 从表3可以看出,MFEA/D算法所用求解时间更少.

为了进一步分析本文所提出算法不同任务之间交流对算法性能的影响,实验以MOEA/D-DE为比较算法. HV指标和IGD指标的中位数和四分位间距统计分析结果如表4所示. 从表4可以看出: 对于HV指标,MOEA/D-DE只有4个任务(共20个任务)的结果要显著优于MFEA/D; 而IGD指标中,MOEA/D-DE只有1个任务结果要显著优于MFEA/D,其他任务统计结果显示MFEA/D要显著优于MOEA/D-DE或者与MOEA/D-DE无显著差异.

综合以上实验结果可以看出,基于分解的多目标多因子进化算法性能优于MO-MFEA算法,并且不同任务之间的信息交流能够加快不同任务的求解. 主要原因可以总结成以下3个方面: 1) MFEA/D中每个任务预先定义的权重向量在一定程度上确定了关联到该任务上所有个体的多样性,进而保证种群有更多的精力去关注收敛; 2) 邻域以及邻域选择概率是权衡种群对单个任务“开发”和“探索”的关键,进而保证了算法求解每个任务的性能; 3) 不同任务之间潜在的有用知识的利用进一步加快了不同任务的求解速度.

3.4 参数敏感性分析

参数rmp对两个任务之间的信息交流频率有着很大的影响. 直观上而言,rmp越大,不同任务之间信

息交流的概率越大,rmp越小,每个任务更着重自己的优化进程. 邻域大小参数n_e决定了子问题之间的合作范围,n_e值越大,每个子问题可能更新的范围越大,越注重单个任务的“探索”能力;n_e值越小,越注重子问题本身的“挖掘”能力. 最大替换个数参数n_r是为了防止单个优秀个体将整个邻域个体或者种群替换掉,进而损失种群的多样性. 邻域选择概率参数δ是权衡新个体对单个任务“探索”和“挖掘”的关键参数. δ值越大,越注重邻域内的进化,也就是“挖掘”;反之,越注重种群进化的“探索”. 但是,只有确定了邻域大小n_e,邻域选择概率δ才有意义.

为了定量地分析rmp、n_e、n_r、δ参数对MFEA/D的影响,本实验通过控制变量法,分别对rmp给定为$\{0, 0.1, \ldots, 1.0\}$,n_e给定为$\{10, 20, \ldots, 100\}$,n_r给定为$\{1, 2, \ldots, 10\}$以及δ给定为$\{0, 0.1, \ldots, 1.0\}$的情况下进行参数敏感性分析实验.

对于上述每个测试问题,每组参数对应的算法均独立运行31次后作Friedman检验统计分析. 图5为不同rmp值情况下的HV/IGD指标平均排名,从中可以看出,随着rmp的增加,无论是HV中值指标平均排名还是IGD中值指标平均排名都是先减小后增大,并且在rmp为0.1～0.2时算法更具有竞争性. 同时,通过本实验可以看出,不同任务之间的交流能够提升问题的求解效果,但频繁的交流反而使得算法性能降低. 图6为不同n_e值情况下的HV/IGD指标平均排名,从中可以看出,MFEA/D算法随着参数n_e的增加性能变弱,这里给出推荐值为10. 不同n_r值情况下的HV/IGD指标平均排名如图7所示,从中可以看出,MFEA/D算法对n_r参数敏感程度不大. 但是,为了防止比较好的解将大部分个体替换,进而使得种群丧失多样性,这里给出推荐值为2～3. 图8表明,MFEA/D算法性能随着δ的增加先增大后降低,并且取值在0.6～0.9时变化不大,使得算法比较有竞争性.

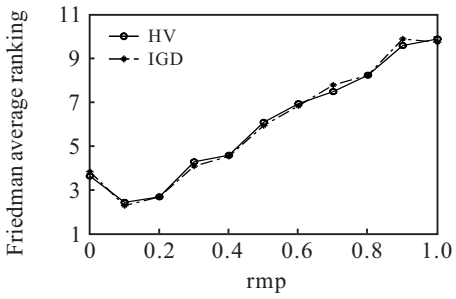


图5　MFEA/D算法在不同rmp值下的HV/IGD指标平均排名

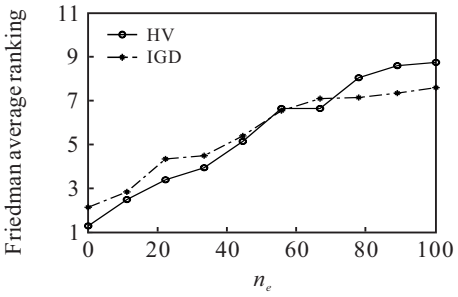


图6　MFEA/D算法在不同n_e值下的HV/IGD指标平均排名

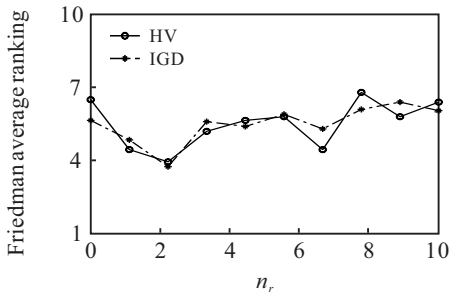


图7　MFEA/D算法在不同n_r值下的HV/IGD指标平均排名

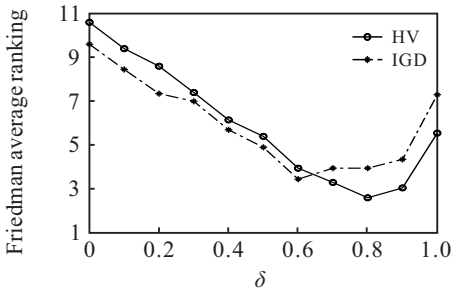


图8　MFEA/D算法在不同δ值下的HV/IGD指标平均排名

4　结　论

针对MO-MFO问题,本文提出了一个基于分解的多目标多因子进化算法(MFEA/D). 该算法通过分解策略将每个任务分解成一系列单目标优化问题,再通过一个种群去优化所有的单目标子问题. 在种群进化过程中,不同任务之间以一定的概率进行信息交流,从而发掘潜在有用信息,加快任务求解速度. 针对10个复杂MO-MFO标准测试问题的实验结果表明,MFEA/D算法显著优于MO-MFEA. 进一步的实验表明,不同任务之间的信息交流有助于加快问题的求解速度,但过频繁的信息交流会损害算法的性能. 未来的研究将去探索每一个子问题对其他任务的影响,进而给出一个合适的随机匹配概率,最大化地提高或者限制不同任务之间的信息交流频率,充分发挥不同任务之间潜在的有用信息,加快问题求解.

参考文献(References)

[1]　Gupta A, Ong Y S, Feng L. Multifactorial evolution: toward evolutionary multitasking[J]. IEEE Transactions

on Evolutionary Computation, 2016, 20(3): 343-357.

[2] Gupta A, Ong Y S, Feng L, et al. Multiobjective multifactorial optimization in evolutionary multitasking[J]. IEEE Transactions on Cybernetics, 2017, 47(7): 1652-1665.

[3] Gupta A, Madziuk J, Ong Y S. Evolutionary multitasking in bi-level optimization[J]. Complex & Intelligent Systems, 2015, 1(1): 83-95.

[4] Chandra R, Ong Y S, Goh C K. Co-evolutionary multi-task learning with predictive recurrence for multi-step chaotic time series prediction[J]. Neurocomputing, 2017, 243: 21-34.

[5] Li H, Ong Y S, Gong M, et al. Evolutionary multitasking sparse reconstruction: Framework and case study[J]. IEEE Transactions on Evolutionary Computation, 2019, 23(5): 733-747.

[6] Tang Z, Gong M, Zhang M. Evolutionary multi-task learning for modular extremal learning machine[C]. 2017 IEEE Congress on Evolutionary Computation (CEC). Piscataway: IEEE, 2017: 474-479.

[7] Zhang Q, Li H. MOEA / D: A multiobjective evolutionary algorithm based on decomposition[J]. IEEE Transactions on Evolutionary Computation, 2007, 11(6): 712-731.

[8] Li H, Zhang Q. Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II[J]. IEEE Transactions on Evolutionary Computation, 2009, 13(2): 284-302.

[9] 周攀, 张冬梅, 龚文引, 等. 基于正交设计的自适应ε占优MOEA/D算法研究[J]. 计算机应用与软件, 2013, 30(2): 58-64.
(Zhou P, Zhang D M, Gong W Y, et al. Research on adaptive epsilon-domination based orthogonal MOEA/D for multi-objective optimization[J]. Computer Engineering and Design, 2013, 30(2): 58-64.)

[10] 刘璐, 郑力明. 基于邻域和变异算子组合优化的MOEA/D算法[J]. 计算机工程, 2017, 43(3): 232-240.
(Liu L, Zheng L M. MOEA / D algorithm based on combinational optimization of neighborhood and mutation operator[J]. Computer Engineering, 2017, 43(3): 232-240.)

[11] 耿焕同, 周山胜, 韩伟民, 等. 基于自适应进化策略的MOEA/D算法[J]. 计算机工程与设计, 2019, 40(4): 1106-1113.
(Geng H T, Zhou S S, Han W M, et al. MOEA/D algorithm based on adaptive evolutionary strategy[J]. Computer Engineering and Design, 2019, 40(4): 1106-1113.)

[12] 田红军, 汪镭, 吴启迪. 一种求解多目标优化问题的进化算法混合框架[J]. 控制与决策, 2017, 32(10): 1729-1738.
(Tian H J, Wang L, Wu Q D. A hybrid framework of evolutionary algorithm for solving multi-objective optimization problems[J]. Control and Decision, 2017, 32(10): 1729-1738.)

[13] 李二超, 陈瑞婷. 基于变异算子和邻域值自适应的MOEA/D算法[J]. 计算机工程与应用, 2019, 55(9): 49-55.
(Li E C, Chen R T. Improved MOEA / D algorithm based on adaptive mutation operator and neighborhood size[J]. Computer Engineering and Applications, 2019, 55(9): 49-55.)

[14] Miettinen K. Nonlinear multiobjective optimization[M]. New York: Springer Science & Business Media, 1999: 5-6.

[15] Das I, Dennis J E. Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems[J]. SIAM Journal on Optimization, 1998, 8(3): 631-657.

[16] Storn R, Price K. Differential evolution-a simple and efficient heuristic for global optimization over continuous spaces[J]. Journal of Global Optimization, 1997, 11(4): 341-359.

[17] Nebro A J, Durillo J J, Vergne M. Redesigning the jMetal multi-objective optimization framework[C]. Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation. New York: ACM, 2015: 1093-1100.

[18] Emmerich M, Beume N, Naujoks B. An EMO algorithm using the hypervolume measure as selection criterion[C]. International Conference on Evolutionary Multi-Criterion Optimization. Berlin: Springer, 2005: 62-76.

[19] Schuetze O, Equivel X, Lara A, et al. Some comments on GD and IGD and relations to the Hausdorff distance[C]. Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation. New York: ACM, 2010: 1971-1974.

[20] Fonseca C M, Fleming P J. On the performance assessment and comparison of stochastic multiobjective optimizers[C]. International Conference on Parallel Problem Solving from Nature. Berlin: Springer, 1996: 584-593.

作者简介

么双双(1990−), 女, 博士生, 从事多因子优化、机器学习的研究, E-mail: yss_neu@163.com;

董志明(1991−), 男, 博士生, 从事多目标优化、机器学习及其应用的研究, E-mail: dongzm@stumail.neu.edu.cn;

王显鹏(1980−), 男, 教授, 博士, 从事基于机器学习的工业生产过程建模、操作优化、进化算法等研究, E-mail: wangxianpeng@ise.neu.edu.cn.

(责任编辑：齐 霁)