控制与决策
Control and Decision

具有重组学习和混合变异的动态多种群粒子群优化算法

唐可心， 梁晓磊， 周文峰， 马千慧， 张煜

引用本文:
唐可心, 梁晓磊, 周文峰, 等. 具有重组学习和混合变异的动态多种群粒子群优化算法[J]. 控制与决策, 2021, 36(12): 2871–2880.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2020.0898

具有重组学习和混合变异的动态多种群粒子群优化算法

唐可心[1], 梁晓磊[1], 周文峰[1], 马千慧[1], 张 煜[2†]

(1. 武汉科技大学 汽车与交通工程学院，武汉 430065；2. 武汉理工大学 物流工程学院，武汉 430063)

摘 要: 为解决粒子群优化算法中种群多样性与收敛性间的矛盾,提出一种具有重组学习和混合变异的动态多种群粒子群优化算法. 该算法动态划分多种群并融入重构粒子作为引导因子,在增加种群多样性的同时保留优秀粒子的空间信息;在算法执行阶段对最优个体施加混合变异,基于时变概率实施反向学习策略或者邻域扰动操作,帮助粒子快速跳出局部困境,加强对附近区域内的精细搜索. 基于14个多类型标准测试函数,并与其他的改进粒子群算法进行对比,验证了几种改进措施的有效性和叠加影响. 为进一步探究概率性混合变异策略的敏感性,对变异方式及参数设置进行仿真实验,结果表明,所采用的极值扰动策略具有显著的优势,合理地控制学习强度可以充分发挥反向学习的作用,并给出影响参数的建议取值范围. 实验结果还表明,所提出的算法能够更好地平衡种群的开发与勘探能力,提高求解精度和收敛性能.

关键词: 粒子群算法；重构粒子；反向学习；极值扰动；概率选择

中图分类号: TP18 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2020.0898 **开放科学(资源服务)标识码(OSID):**

引用格式: 唐可心,梁晓磊,周文峰,等. 具有重组学习和混合变异的动态多种群粒子群优化算法[J]. 控制与决策, 2021, 36(12): 2871-2880.

Dynamic multi-population particle swarm optimization algorithm with recombined learning and hybrid mutation

TANG Ke-xin[1], *LIANG Xiao-lei*[1], *ZHOU Wen-feng*[1], *MA Qian-hui*[1], *ZHANG Yu*[2†]

(1. School of Automobile and Traffic Engineering，Wuhan University of Science and Technology，Wuhan 430065，China；2. School of Logistics Engineering，Wuhan University of Technology，Wuhan 430063，China)

Abstract: To solve the contradiction between population diversity and convergence in particle swarm optimization, an improved particle swarm optimization algorithm, called dynamic multi-population particle swarm optimization with recombined learning and hybrid mutation, is proposed. In the proposed algorithm, a population is divided dynamically and a new particle is reconstructed as a guiding factor. It retains the spatial information of the excellent particles while increasing population diversity. During the execution of the algorithm, a hybrid mutation strategy is applied to adjust the optimal solution. The opposition-based learning and neighborhood-disturbance operations are implemented based on a time-varying probability, which help the particles jump out of the local dilemma quickly, and strengthen good searching in the nearby areas. The effectiveness and superposition effects of several proposed improvement operations compared with several improved particle swarm algorithms based on a set of 14 multi-type benchmark functions are verified. In order to further explore the sensitivity of probability-based hybrid mutation strategy, a large number of simulation experiments are carried out to analyze the mutation mode and parameter settings. The results show that the disturbed extreme strategy has significant advantages. Controlling the learning intensity reasonably can make the opposition-based learning show better performances, furthermore, a suggested value range is given. Finally, experimental results show that the proposed algorithm can get a better balance between the exploitation and exploration for the swarm searching and improve the solution accuracy and convergence performance.

Keywords: particle swarm algorithm；reconstructed particles；opposition-based learning；disturbed extremum；probability options

收稿日期: 2020-07-05；**修回日期:** 2020-09-23.

基金项目: 国家自然科学基金项目(61603280, 71874132).

责任编委: 龙建成.

†通讯作者. E-mail: sanli@whut.edu.cn.

0 引 言

粒子群优化算法(particle swarm optimization, PSO)是由Kennedy等[1]受鸟群觅食行为启发而提出的一种基于群体智能的随机优化方法,其主要思想是通过粒子个体的学习行为和群体之间的合作交互实现问题求解. 由于PSO算法具有结构简单、并行性强、收敛速度快等优势,多被应用于解决多目标优化[2]、神经网络训练[3]、图像处理[4]等问题. 然而,PSO也存在早熟收敛、易陷入局部最优等现象. 针对这些问题,众多学者从多方面提出了改进方法,如参数调整、精英选择和算法混合等不同策略[5]. Shi等[6]较早提出了一种惯性权重线性递减的规则;文献[7]在此基础上通过动态调整策略提高算法的收敛性和收敛速度. 此外,也有学者对PSO的加速因子进行了类似的改进,提出时变自组织PSO算法(HPSO-TVAC)[8]. 为进一步提高粒子的寻优能力,Zhang等[9]将邻域最优解作用到速度更新公式上,通过迭代进程决定学习强度. 近年来,利用改变拓扑结构来选择学习榜样也得到了较好的结果. 如Chen等[10]利用双差分突变策略设计两层新型种群结构,采用两种不同控制参数的推理突变操作产生精英粒子,以保证种群多样性. 文献[11]提出了一种具有异构分簇的粒子群算法(APSO-C),采用K-均值聚类对种群进行动态分簇,通过Ring型结构对各簇进行信息交流,选择学习样本. 许多研究表明,动态多种群结构[12]和自适应学习框架[13]均能有效提高算法的全局搜索能力,在一定程度上可以克服PSO的早熟现象. 混合算法方面,一些学者也采用遗传算子[14]和模拟退火操作[15]加深局部寻优,结合差分进化算法[16]修正全局最优粒子. 而文献[17]还将粒子群算法与人工蜂群算法(ABC)进行杂交,提出一种重组算法.

粒子群优化算法的寻优能力主要取决于两个特征:开采性和勘探性[18]. 为了平衡两者能力,提高算法的综合性能,与已有研究不同,本文提出一种具有重组学习和混合变异的动态多种群粒子群优化算法(dynamic multi-population particle swarm optimization algorithm with recombined learning and hybrid mutation, DMPSORH). 借鉴小生境的思想将种群划分成规模相等的子种群以及局部PSO模型,再对其实施不同的速度更新策略. 同时,利用重构粒子引导速度方向来加强种群的局部开采能力,借助于历史最优位置的子空间信息指导粒子进行局部搜索. 并且,为了进一步扩大全局有效搜索范围,提出一种基于概率性度量概念的混合变异策略,在保持最优学习轨迹的同时进行自适应维度更新. DMPSORH在迭代过程中随机选择变异方式,采用反向学习策略(opposition-based learning, OBL)和极值扰动操作(disturbed extremum, DE),为最优粒子逃离局部困境提供更多步长增量,引导其探索更多新区域. 重构粒子与混合变异结合,可以在进化早期保持种群的多样性,并加快后期收敛速度,有效地兼顾全局勘探与局部开采能力.

1 具有重组学习和混合变异的动态多种群粒子群优化算法

1.1 标准粒子群优化算法原理

在一个$\boldsymbol{D}$维的搜索空间中,设有一个个体数为n的种群$\boldsymbol{X}=(\boldsymbol{X}_1,\boldsymbol{X}_2,\dots,\boldsymbol{X}_n)$,而向量$\boldsymbol{X}_i=[X_{i1},X_{i2},\dots,X_{iD}]^{\mathrm{T}}$表示第$i$个粒子的位置. 同样,可用向量$\boldsymbol{V}_i=[V_{i1},V_{i2},\dots,V_{iD}]^{\mathrm{T}}$,$\boldsymbol{P}_i=[P_{i1},P_{i2},\dots,P_{iD}]^{\mathrm{T}}$分别表示第$i$个粒子的速度及其极值(Pbest). 除此之外,向量$\boldsymbol{P}_g=[P_{g1},P_{g2},\dots,P_{gD}]^{\mathrm{T}}$代表种群的全局极值(Gbest),然后依据目标函数F计算出每个$\boldsymbol{X}_i$的适应度值. 第$k+1$代时,第i个粒子的第d维的速度和位置更新公式如下:

$$V_{id}^{k+1}=\omega V_{id}^{k}+c_1r_1(P_{id}^{k}-X_{id}^{k})+c_2r_2(P_{gd}^{k}-X_{id}^{k}),\tag{1}$$

$$X_{id}^{k+1}=X_{id}^{k}+V_{id}^{k+1}.\tag{2}$$

其中:ω是惯性权重;c_1、c_2是学习因子,为非负常数;r_1、r_2是[0,1]之间的随机数.

飞行速度$\boldsymbol{V}$可以表征粒子在整个搜索空间中进行遍历的能力[5],它将直接影响粒子位置更新的步长,而为了避免粒子越界,飞出搜索空间,通常将其限制在$[-V_{\max},V_{\max}]$的飞行区域中,$V_{\max}$一般取值为可行域的10 %～20 %.

1.2 动态多种群划分策略

多种群PSO算法其实也是一种基于特殊邻域拓扑结构的局部PSO算法[19],相比于固定划分多种群,动态随机重组可以避免过于限制粒子的自由,提升个体信息交流效率. 文献[20]基于分类的思想以适应度值的均值和标准差为测度,通过评价粒子的位置关系将种群分为多个等级. 但是在迭代后期,优秀种群会吸引更多个体加入,将导致粒子过于聚集,种群间信息交流困难. 所以,本文在此基础上以适应度值升序的中位值为界划分子种群,每一代都实行动态调整. 适应度值较小的为“顶层粒子”,较大的为“底层粒子”;再从两者当中分别抽出同等数量的粒子组成局部PSO模型,并确保3个种群规模均衡,分别为离

最优解较近的“亲邻域”、局部PSO模型的“重组域”以及“疏离域”.

“亲邻域”中的粒子适应度值较小,需要继承种群中最优解的位置信息,缩小步长进行更加细密的搜索;而“疏离域”中的粒子则应扩大“步伐”,在向全局极值靠近的同时探索周围新的优解来提升开采能力;作为局部模型的“重组域”将动态调整种群的多样性,既要吸取个体经验,也要共享全局信息.

1.3　重构粒子引导及速度更新机制

粒子群在每代的飞行中都将更新历史最优位置,而自身极值中的认知经验便会被忽略. 本文在执行速度更新时利用重构粒子保留Pbest的子空间信息,不仅是为了保持精英学习的轨迹,而且也从多维角度增加群体的多样性. 基于重构粒子引导的速度更新公式为

$$V_{id}^{k+1}=\omega V_{id}^{k}+(c_1-c_3)r_1(P_{id}^{k}-X_{id}^{k})+ c_2r_2(P_{gd}^{k}-X_{id}^{k})+c_3r_3(P_{\mathrm{mix}}^{k}-X_{id}^{k}),\quad (3)$$

其中P_{mix}^{k}为第k代的重构粒子,其每一维度都随机选自每个粒子当前历史最优位置Pbest所对应的某一维,且不重复选取同一个粒子的多个维度.

重构粒子生成过程如图1所示.

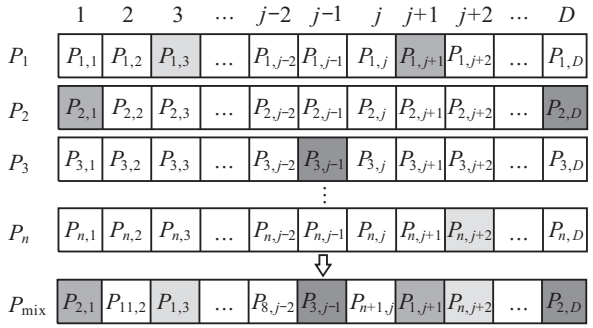


图 1　重构粒子生成过程

下面以二维的搜索空间覆盖范围为例,对引入重构粒子的策略进行特性分析. 为便于简化,引入记号如下:

$$\alpha=\omega V_i^k+c_2r_2(P_g^k-X_i^k),\ \beta=c_1(P_i^k-X_i^k),$$
$$\mu=c_3(P_{\mathrm{mix}}^k-X_i^k),\ \beta'=(c_1-c_3)(P_i^k-X_i^k).$$

同时,将式(1)改写成

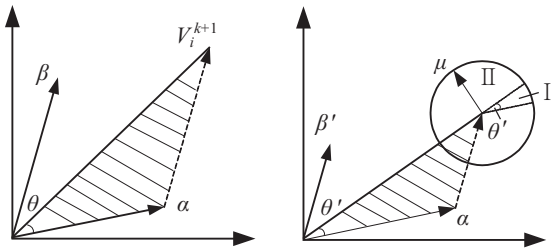
$$V_i^{k+1}=\alpha+r_1\beta,\quad (4)$$

式(3)改写成

$$V_i^{k+1}=\alpha+r_1\beta'+r_3\mu,\quad (5)$$

$$\beta'=\frac{(c_1-c_3)}{c_1}\beta.\quad (6)$$

在改写后的速度更新公式中,由于两式存在相同项α,暂且将它看作定值,只比较其他项的变化对更新结果的影响. 因为r_1、r_3是[0,1]之间的随机数,所以式(4)中V_i^{k+1}的更新方向可能是由两个向量构成的三角形范围,如图2(a)所示;而对于式(5),V_i^{k+1}的最终变化方向在一定程度上取决于μ,当前两项的向量式加权和将速度方向控制在一个范围内时,μ的引入可以引导粒子拥有更多的选择方向,如图2(b)所示. μ在I区域将会小幅度地改变既定的搜索方向;II区域的μ相对于图2(a),将沿合成向量方向进行伸缩并伴随一定的旋转角度. 多方向的随机扰动使得粒子因记忆历史最优位置信息而减缓了当前的学习强度,但在一定程度上反而增强了搜索过程中的多样性. 这样的寻优方式更加适合“疏离域”中的粒子在解空间中进行波动性搜索,加深局部学习. 同时,图2也可以解释某些通过新增个体来丰富多样性的策略得不到理想结果的原因,即改进措施只是单纯强调群体的多样性而忽略了对不同搜索阶段造成的影响程度. 粒子的搜索步长随着迭代进程变小,促使μ延续Pbest的经验逐渐与β'方向一致,继而实现“优中寻优”;相反,如果此时新增个体的精英学习操作较为薄弱,则它的引入将会具有“对立性”,使得个体缺乏正向进化.



(a)　标准粒子群　　(b)　重构粒子群

图 2　速度更新方式对比

本文所构建的3个子种群各自更新方式如下:“亲邻域”中的粒子采用基本PSO的更新公式,即式(1)和(2);“疏离域”利用式(3)和(2)进行更新;“重组域”介于两者之间,由于个体间的差异在算法前期较大,侧重于其认知部分可以实现多方交流,而在后期加强全局极值的引领力,能够促使粒子向最优解的周围聚集. 因此,学习因子引入余弦、正弦函数,分别呈现递减和递增的趋势. 粒子位置按照式(2)更新,有

$$V_{id}^{k+1}=\omega V_{id}^{k}+2\cos\left(\frac{\pi t}{2T}\right)r_1(P_{id}^{k}-X_{id}^{k})+ 2\sin\left(\frac{\pi t}{2T}\right)r_2(P_{gd}^{k}-X_{id}^{k}).\quad (7)$$

其中:t为当前迭代次数,T为最大迭代次数. 式(7)为“重组域”种群的速度更新公式.

1.4 概率性混合变异策略选择

粒子群算法在优化后期收敛速度变得缓慢的主要原因是其难以摆脱当前局部极值,导致精度下降. 很多学者为此提出了一些改进方法,如重新初始化[21]、自适应PSO[22]和杂交策略[23]等,并在不同程度上取得了良好的效果,提高了收敛速度和精度. 但是,这些改进措施缺乏对算法收敛于局部最优的根本原因以及其在不同阶段的动态效果的综合考虑,存在一定的局限性. 文献[24]和文献[25]从数学理论的角度,通过严格推导得到如下公式,从而解释了PSO在进化停滞状态下的“聚集”现象:

$$\lim_{t\to\infty} x(t) = P'_g = \frac{c_1P_i + c_2P_g}{c_1 + c_2} = (1-\lambda)P_i + \lambda P_g, \tag{8}$$

$$\lambda = \frac{c_2}{c_1 + c_2}. \tag{9}$$

由式(8)和(9)可知,在算法进程中,自身极值P_i和全局极值P_g将决定粒子聚集位置. 当所有粒子逐步靠近P'_g却没有找到比P_g更优的位置时,将处于停滞状态,从而向P'_g聚集陷入局部最优,所以给P_g施加变异策略将是摆脱局部极值的有效途径. 不过,变异方式及程度将决定找到更优解的几率. 较大的变异步长确实能帮助粒子跳出当前“困境”,但同时也有可能导致与最优值“失之交臂”,增加寻优的难度和复杂性. 因此,在变异过程中也需要伴随着邻域学习,加深高密度的搜索,这样既能给予粒子更多的搜寻机会,也能保证其不逃离次优解的区域.

本文基于上述分析,提出一种概率性混合变异策略(probability-based hybrid mutation, M-P),综合采用反向学习策略(OBL)和极值扰动操作(DE),以时变概率随机选择变异方式.

1) 最优粒子的反向学习变异.

它的主要思想是计算并评估Gbest与其反解,从两者中选择较优值作为下一代的全局引导,增加跳出局部最优的概率. 下面给出相关定义.

定义1 反向点. 设$X_i = (x_{i1}, x_{i2}, \dots, x_{iD})$为$D$维空间中的一个点,且$x_{ij} \in [a_j, b_j]$,则$X_i$对应的反向点$X'_i = (x'_{i1}, x'_{i2}, \dots, x'_{iD})$可定义为

$$x'_j = a_j + b_j - x_j. \tag{10}$$

定义2 最优粒子反向解. 设$X = (x_1, x_2, \dots, x_D)$是全局极值点,其反向解$X^* = (x_1^*, x_2^*, \dots, x_D^*)$可定义为

$$x_j^* = k(\mathrm{da} + \mathrm{db}) - x_j. \tag{11}$$

其中:$x_j \in [a, b], k \in U(0,1)$为一般化系数;$[\mathrm{da}, \mathrm{db}]$为搜索空间中的动态边界,按下式计算得到:

$$\mathrm{da} = \min(x),\ \mathrm{db} = \max(x). \tag{12}$$

利用动态边界代替搜索空间中的固定边界,既能保证生成的反向解位于逐步缩小的空间中[26],也能控制过度学习,提高有效变异率. 同时,本文考虑到全局极值的最优特性,采用Gbest的维度最值作为动态边界,不引入其他粒子携带的搜索信息影响它的“优越性”. 当然,在反向学习的过程中,反向解有飞越动态边界的可能,因此,对其采用下式的方法生成在$[\mathrm{da}, \mathrm{db}]$中的随机数:

$$x_j^* = \mathrm{da} + k(\mathrm{db} - \mathrm{da}),\ x_j^* < \mathrm{da} \text{ or } x_j^* > \mathrm{db}. \tag{13}$$

2) 最优粒子的邻域学习变异.

文献[27]提出了一种高斯变异策略(Gaussian mutation, GM),采用下式对Gbest进行扰动:

$$\mathrm{Gbest}'_d = \mathrm{Gbest}_d + r(X_{\max} - X_{\min})\mathrm{Gaussian}(\mu, \delta^2). \tag{14}$$

其中:Gbest'_d为最优粒子扰动后的d维分量,$[X_{\min}, X_{\max}]$表示d维空间的搜索范围,μ是高斯随机数的均值,δ^2是高斯随机数的方差. 文献[27]中$\delta^2 = |\mathrm{Gbest}_d|, r \in (0,1)$是服从均匀分布的随机数. 该策略在一些函数的测试中取得了较好的效果,如Rastrigin函数,但对于另外一些复杂函数却作用不大.

本文提出的全局极值扰动策略,主要是依据方差可调的正态随机分布对Gbest实施变异,得到新的最优粒子Gbest',即

$$\mathrm{Gbest}_d^{t\prime} = N(\mathrm{Gbest}_d^t, \delta). \tag{15}$$

其中δ表示相对于Gbest_d的不确定度,是关于循环变量t的递减函数,用以衡量扰动值的偏离程度. 本文的δ按下式变换:

$$\delta = 10^{\zeta_1 - (\zeta_1 - \zeta_2)\cdot t/T}. \tag{16}$$

其中ζ_1、ζ_2是控制正态扰动幅度的半径参数,这里分别取-1和-3,表示δ随算法的迭代进程从10^{-1}递减到10^{-3}. 在算法早期δ较大时,扰动策略使得全局最优解在附近较大的邻域进行搜索学习,并为其提供简单高效的多路径并行引导;而后期较小的δ,能够控制当前最优粒子几乎不再跳出较优区域,保证算法具有较好的收敛性.

一般地,在群体进化的初期,进行多维度学习可以帮助最优粒子加速跳出局部极值区域,为其提供更多的学习机会;而在末期,全局极值已经接近最优解,尽量保留各维度的优势信息有利于提高算法精度. 因此,本文采用下式进行变异维度的规模选择.

$$\dim = \lceil (1 - (t-1)/T) \cdot D \rceil. \tag{17}$$

其中: dim表示选择变异的维度数量, D表示维数, $\lceil\ \rceil$表示向上取整. 同时, 基于贪婪保留策略对Gbest采用逐维更新的方式, 即

$$\mathrm{Gbest}_d = \begin{cases} \mathrm{Gbest}'_d, \ \mathrm{fit}(\mathrm{Gbest}'_d) < \mathrm{fit}(\mathrm{Gbest}_d); \\ \mathrm{Gbest}_d, \ \mathrm{otherwise}. \end{cases} \tag{18}$$

下面给出M-P策略的伪代码用来描述其基本步骤. 其中: TR为时变概率, 是服从$U(0,1)$的随机数; OR是使用反向学习策略的概率.

算法　M-P策略.

randomly disorganize D;
update dim according to eq.(17);
for $d = 1$: length (dim)
　if TR $<$ OR then
　　generate the opposite Gbest by eq.(11);
　　if $\mathrm{Gbest}_d < \min(\mathrm{Gbest})$ or $\mathrm{Gbest}_d > \max(\mathrm{Gbest})$ then
　　generate the opposite Gbest by eq.(13) again;
　　end
　else
　　execute the DE strategy on Gbest by eqs.(15) and (16);
　end
　compare fit and select Gbest according to eq.(18);
end

1.5　算法流程及时间复杂性分析

综合算法的构造思想及以上具体策略, 现给出算法流程(见图3)和时间复杂度分析.

由图3可以看出, DMPSORH主要包括5个部分: 初始化, 划分种群, M-P策略, 引入重构粒子, 速度、位置与极值的更新. 算法迭代一次各部分所需的操作为: 1) 初始化种群的复杂度为$O(N\cdot D)$; 2) 进行划分动态子种群的操作是$O(1)$; 3) M-P策略中变异维度规模呈线性递减, 若取最大规模进行计算, 则复杂度为$O(D)$; 4) 生成重构粒子作为引导的基本运算是$O(D)$; 5) 对粒子的飞行速度、位置以及对极值的更新为$O(N{\cdot}D)$. 因此, DMPSORH算法迭代一次的时间复杂度是

$$O(N\cdot D)+O(1)+O(D)+O(D)+O(N\cdot D)=O(N\cdot D),$$

与基本PSO算法具有相同的复杂度.

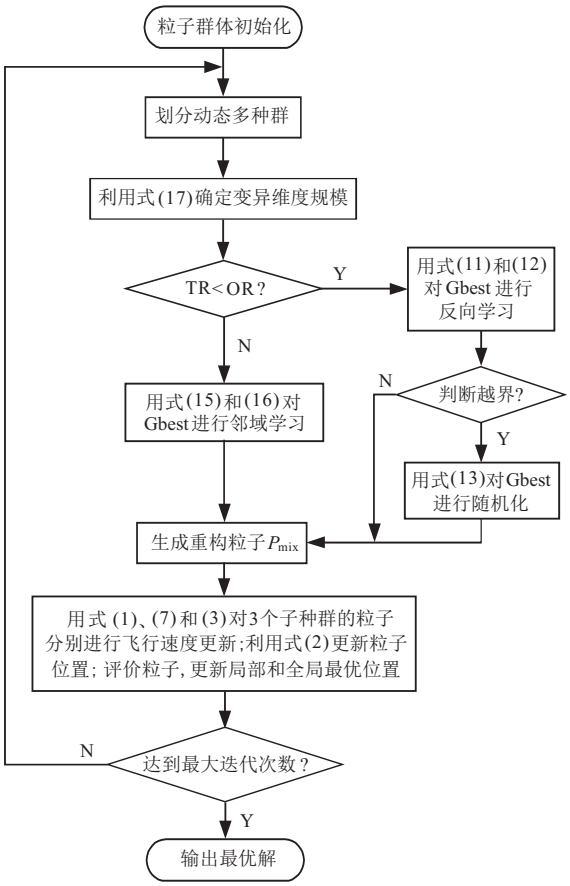


图3　算法流程

2　数值实验及分析

2.1　测试函数

为了测试算法的性能, 本文实验采用表1中的14个国际标准benchmark测试函数, 按照其属性分为单峰函数($f_1 \sim f_2$)、多峰函数($f_3 \sim f_9$)、带平移变换的函数($f_{10} \sim f_{11}$)和带旋转变换的多峰函数($f_{12} \sim f_{14}$). 维度$D = 30$, 定义域$X \in R^D$, F^*是理论极值. "threshold"为解的精度阈值$(F - F^*)$, 用来判断算法

表1　14个测试函数

funs.	name	X	F^*	threshold
f_1	Sphere	$[-100,100]^D$	0	1.00e-100
f_2	Quadric	$[-100,100]^D$	0	5.00e-06
f_3	Rosenbrock	$[-2.084,2.084]^D$	0	5.00e-02
f_4	Griewank	$[-600,600]^D$	0	0.00e+00
f_5	Ackley	$[-32,32]^D$	0	5.00e-14
f_6	Rastrigin	$[-5.12,5.12]^D$	0	0.00e+00
f_7	Schwefel	$[-500,500]^D$	0	5.00e-04
f_8	Weierstrass	$[-0.5,0.5]^D$	0	0.00e+00
f_9	Nocon_Rastrigin	$[-5.12,5.12]^D$	0	0.00e+00
f_{10}	Shifted_Sphere	$[-100,100]^D$	−450	1.00e-10
f_{11}	Shifted_Rosenbrock	$[-100,100]^D$	390	1.00e+01
f_{12}	Rotated_Rosenbrock	$[-100,100]^D$	−900	5.00e+01
f_{13}	Rotated_Griewank	$[-100,100]^D$	−500	5.00e-01
f_{14}	Rotated_Schwefel	$[-100,100]^D$	100	5.00e+03

是否运行成功.

2.2 算法构造策略的有效性和叠加影响分析

本文通过划分多种群、引入重构粒子以及使用M-P策略来提升PSO的综合性能. 本节将在基本PSO算法的基础上逐步增加相应措施, 比较几种策略在相同迭代次数下的作用. bPSO是基本PSO算法; DMPSO、PSOR、PSOH是在PSO框架中分别增加3种策略; DMPSOR是在划分动态多种群的基础上融入重构粒子作为引导, DMPSOH在划分动态多种群的基础上增加对最优解的变异策略, PSORH是在引入重构粒子的基础上实施混合变异; DMPSORH是结合3种策略的综合算法. 同时还与其他两种改进算法进行比较, 分别是HPSO-TS[28]和CLPSO[29].

本节实验所有算法均采取相同的迭代次数$T = 3\,000$. 对每个测试函数独立运行30次, 记录中位值和平均执行时间. 实验环境为: Intel i5 CPU 2.30 GHz. RAM 4.0 GB, Windows10操作系统, Matlab R2014a. 其他参数设置如下: 种群规模size = 30; 惯性权重$\omega = 0.8 \times \mathrm{e}^{(-0.5\times t/T)}$; 最大飞行速度$V_{\max}$为可行域的1/10; 学习因子$c_1 = 1, c_2 = 1.2, c_3 = 0.3$ (bPSO、DMPSO、PSOH、DMPSOH中$c_3 = 0$); OR = 0.2. HPSO-TS和CLPSO的参数设置与原文献相同. 实验结果见表2, 最优值用粗体显示, 次优值用斜体显示, 各算法运行时间见图4.

表 2 算法构造策略的测试实验

funcs.	bPSO	DMPSO	PSOR	PSOH	DMPSOR	DMPSOH	PSORH	DMPSORH	HPSO-TS	CLPSO
f_1	4.25e+02	3.13e+02	8.72e-06	1.61e-106	1.28e-11	1.25e-106	5.50e-107	*4.16e-115*	**0.00e+00**	7.60e-10
f_2	2.80e+03	2.47e+03	8.73e+00	5.37e-05	9.47e-02	3.63e-05	4.96e-06	*2.14e-07*	**0.00e+00**	2.78e+03
f_3	4.91e+01	5.37e+01	2.44e+01	6.76e+00	2.03e+01	*1.23e-01*	1.48e+01	**6.09e-03**	2.89e+01	2.27e+01
f_4	4.60e+00	5.50e+00	4.17e-02	**0.00e+00**	4.43e-02	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**	*2.82e-07*
f_5	7.12e+00	7.41e+00	5.49e+00	2.13e-14	5.66e+00	2.13e-14	2.13e-14	*1.42e-14*	**0.00e+00**	1.50e-05
f_6	6.71e+01	6.29e+00	2.04e+01	**0.00e+00**	2.34e+01	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**	*1.90e-04*
f_7	6.73e+03	6.51e+03	6.44e+03	3.82e-04	6.60e+03	**3.82e-04**	**3.82e-04**	**3.82e-04**	8.42e+03	*3.83e-04*
f_8	1.52e+01	1.55e+01	1.29e+01	**0.00e+00**	1.33e+01	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**	*1.79e-04*
f_9	7.66e+01	5.60e+01	3.60e+01	**0.00e+00**	3.30e+01	**0.00e+00**	**0.00e+00**	**0.00e+00**	**0.00e+00**	*3.50e-02*
f_{10}	2.40e+03	1.87e+03	1.16e+03	7.97e-10	5.38e+02	6.22e-10	*1.05e-10*	**2.43e-11**	9.18e+04	1.00e-09
f_{11}	1.11e+08	3.39e+07	4.29e+07	2.47e+02	1.39e+06	1.33e+02	*2.17e+01*	**8.59e-01**	1.58e+11	5.71e+01
f_{12}	2.28e+02	2.04e+02	8.82e+01	2.73e+01	8.55e+01	*2.07e+01*	2.22e+01	**1.41e+01**	2.51e+04	5.03e+01
f_{13}	4.60e+02	4.03e+02	1.31e+02	*7.51e-02*	1.08e+02	8.38e-02	8.25e-02	**4.68e-02**	1.51e+04	1.97e+01
f_{14}	5.46e+03	5.28e+03	4.30e+03	4.83e+03	**3.98e+03**	4.32e+03	4.04e+03	*4.00e+03*	1.11e+04	5.61e+03

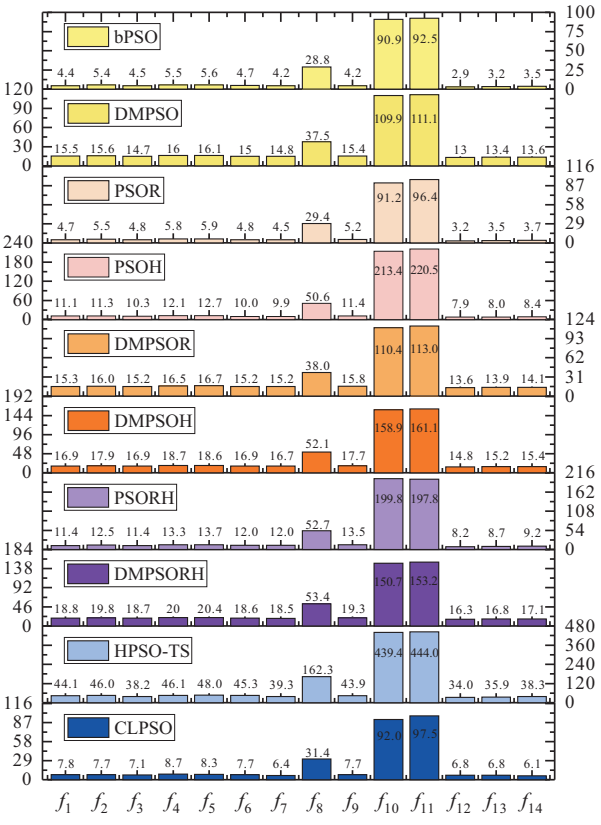


图 4 不同算例下的算法时间统计

表2的实验结果表明:

1) 在相同的迭代次数下, 本文所提出的算法DMPSORH在14个测试函数中取得10个最优结果和4个次优结果, HPSO-TS、CLPSO分别取得7个最优结果和5个次优结果.

2) 引入重构粒子的PSOR算法和实施变异操作的PSOH算法在所有测试函数上都不同程度地优于bPSO, 可见这两种策略对算法性能的提升有一定促进作用.

3) 叠加两种策略的算法DMPSOH、PSORH都取得了5个最优结果和2个次优结果, 并且在大部分函数上都比PSOH表现优越; 同时, DMPSOR算法在函数f_1、f_2、f_{10}、f_{11}上比PSOR更具有优势, 并在f_{14}上取得最优结果, 只有在$f_5 \sim f_8$上稍差于PSOR. 可以保守地认为, 在进行单峰及平移变换函数的优化时, 动态划分子种群可以在一定程度上改善PSOR算法的性能.

4) 与前面7个算法相比, 结合3种策略的DMPSORH算法除了在f_{14}上略差于DMPSOR外, 其

他计算结果均更优.

综上所述,本文提出的3种策略都具有积极作用,并且将其融合能够促进算法性能大幅度提升,从而取得更满意的计算结果.

图4中,HPSO-TS算法的执行时间最长,bPSO的时间最短,其次是PSOR、CLPSO. 对比前4种算法可以发现,划分种群和实施变异会导致算法时间具有明显的增长,而融入重构粒子的策略对运行时间影响较小. DMPSORH除了在f_{10}、f_{11}上的时间比PSOH、DMPSOH和PSORH少之外,其他测试时间均比前7个算法长. 总之,组合策略确实会在一定程度上增加算法的执行时间,但对于优化平移函数而言,DMPSORH所需时间并非处于劣势.

2.3　算法的搜索过程对比及分析

为了对比测试算法的搜索性能,本文选取不同叠加方式并与其他两种改进类算法进行对比实验. 图5给出了PSO、PSOR、PSORH、DMPSORH、HPSO-TS和CLPSO六种算法在30次独立运行中的迭代过程.

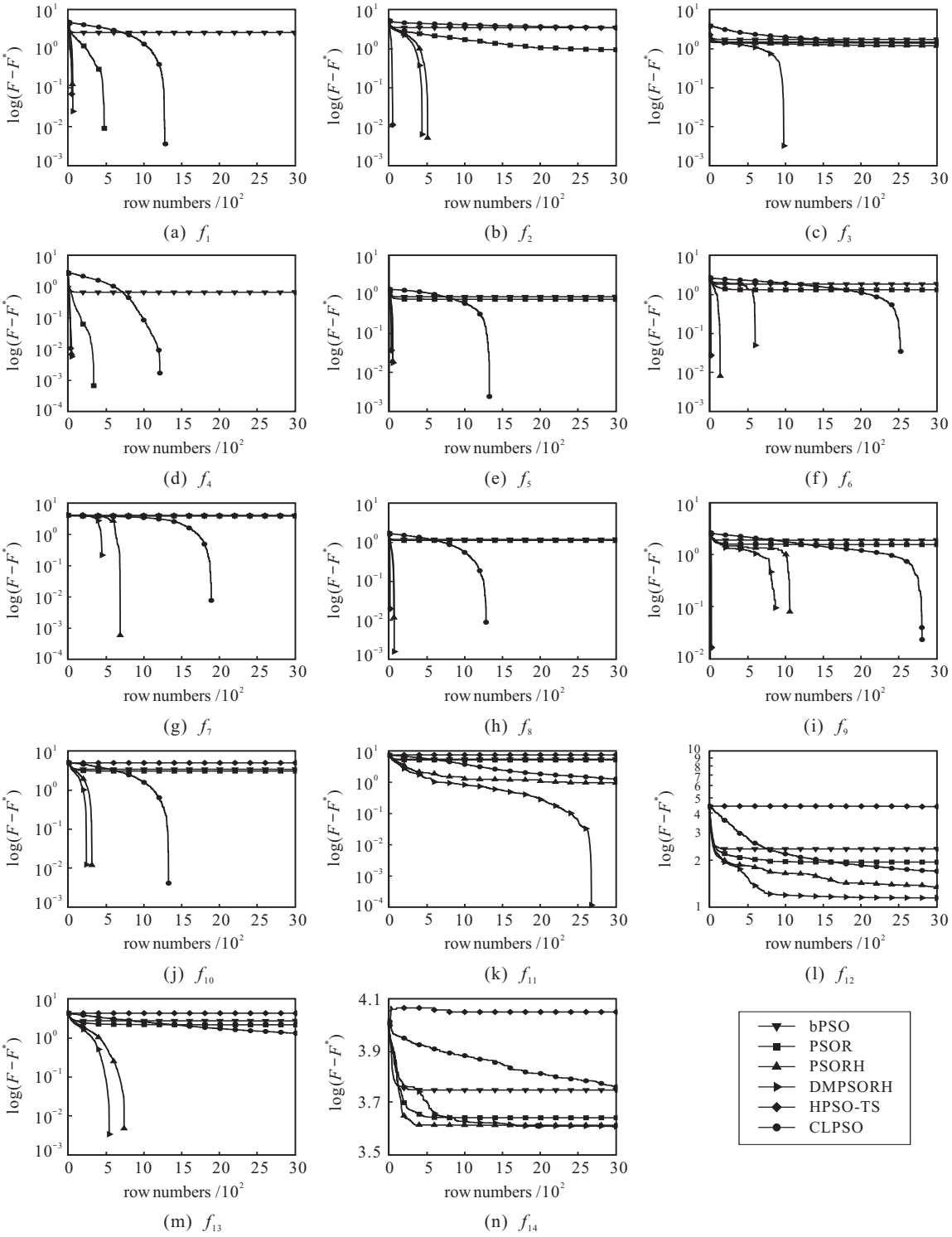


图5　不同算例下的算法平均收敛曲线

其中: x 是迭代次数, y 轴以对数形式表示每代最优适应度值在独立运行中的中位值.

从图5可以看出, 与数值实验结果相似, 6种算法的进化趋势互有差异. DMPSORH和PSORH的收敛精度在大部分函数上都明显优于bPSO、 PSOR和CLPSO. 在函数 f_1、 f_4、 f_5、 f_8 上, DMPSORH与PSORH的收敛曲线基本重合, 这说明在对这5个函数进行优化时, M-P策略起主导作用. 从图5(b)、(g)和(i)～(n)中也可以发现各策略的独立效果还是不如叠加组合的效果好. 此外, 对于优化复杂函数(f_{10} ～ f_{14}), DMPSORH算法的进化趋势表现最好, 收敛精度和速度更具优势, 而HPSO-TS的表现相对最差, 其次是基本粒子群算法bPSO.

2.4 M-P策略与参数敏感性分析

由上述实验结果和分析可知, M-P策略对提升算法性能有显著优势. 为了进一步探究M-P策略中变异方式的效用, 将不含极值扰动策略的DMPSORH (DMPSORH-Null) 和集成GM策略代替DE策略的DMPSORH (DMPSORH-GM) 作为对比算法. 每个函数迭代次数为3 000, 独立运行30次. 实验中记录算法运行30次的成功率 (success rate, SR), 即算法最终解达到表1中阈值次数占运行总次数的比例. 同时, 也统计算法中变异策略成功次数的中位值 (median number of successful mutation, MNSM), 若变异后的Gbest更优, 则表明变异策略成功. 表3给出了实验结果, 其中“median”代表中位值.

表 3 不同变异方式的比较实验

funs.	DMPSORH-Null			DMPSORH-GM			DMPSORH		
	median	SR / %	MNSM	median	SR / %	MNSM	median	SR / %	MNSM
f_1	2.98e-113	100	2.53e+03	3.08e-113	100	2.52e+03	4.16e-115	100	2.53e+03
f_2	5.53e-07	100	2.83e+03	4.83e-07	100	2.84e+03	2.14e-07	100	2.88e+03
f_3	8.51e-03	90	2.82e+03	8.42e+00	0	2.88e+03	6.09e-03	100	2.82e+03
f_4	0.00e+00	57	2.00e+02	0.00e+00	53	2.40e+02	0.00e+00	77	2.00e+02
f_5	2.13e-14	100	2.88e+02	1.60e-14	100	2.72e+02	1.42e-14	100	2.89e+02
f_6	0.00e+00	100	1.80e+02	0.00e+00	100	1.68e+02	0.00e+00	100	1.86e+02
f_7	3.82e-04	87	2.70e+02	1.51e+03	0	6.00e+00	3.82e-04	93	2.80e+02
f_8	0.00e+00	87	2.75e+02	0.00e+00	80	2.35e+02	0.00e+00	90	2.91e+02
f_9	0.00e+00	100	2.51e+02	3.62e-06	10	7.56e+02	0.00e+00	100	2.68e+02
f_{10}	1.00e-07	0	2.85e+03	2.27e-13	97	4.06e+02	2.43e-11	100	1.30e+03
f_{11}	8.11e+01	0	2.83e+03	4.89e+00	77	2.87e+03	8.59e-01	90	2.89e+03
f_{12}	5.86e+01	50	2.83e+03	1.72e+01	87	9.93e+02	1.41e+01	93	2.90e+03
f_{13}	7.61e-01	37	2.84e+03	5.78e-02	100	6.32e+02	4.68e-02	100	2.77e+03
f_{14}	4.05e+03	90	1.89e+03	4.22e+03	90	1.81e+03	4.00e+03	90	2.13e+03

从表3的仿真结果可以看出:

1) 除 f_{10} 外, DMPSORH在大部分测试函数上均比DMPSORH-Null、DMPSORH-GM的结果更优, 并且它的求解成功率也是最高. 此外, 在求解精度相当、成功率相同的情况下(f_2、 f_5、 f_6、 f_{14}), DMPSORH的MNSM也高于其他两种算法, 说明融入DE策略的算法更具优势.

2) 结合GE策略的DMPSORH-GM算法在函数 f_3、 f_7 上明显差于DMPSORH-Null和DMPSORH. 但在 f_3 的平移和旋转变换以及 f_4 的旋转变换, 即 f_{11}、 f_{12}、 f_{13} 上, DMPSORH-GM的结果优于DMPSORH-Null, 略差于DMPSORH. 这表明GE策略在搜索空间复杂化之后能够起到一定的积极作用, 但是DE的效果更佳, 算法稳定性更强.

M-P策略中的两种变异方式以OR为转换标志. 为了测试OR对DMPSORH算法性能的影响, 本次实验将0.1作为间隔, OR取[0.1, 0.9]之间的数值, 算法最大运行次数设置为3 000. 由于 f_1 (Sphere)、 f_4 (Griewank)、 f_6 (Rastrigin)、 f_8 (Weierstrass) 和 f_9 (Nocon_Rastrigin) 在测试中都收敛到了极小值, 实验中抽取 $T = 200$ 的数据进行比较. 每个测试函数分别运行30次, 记录中位值. 实验结果见表4和表5.

表4的实验结果表明, f_1、 f_4、 f_6、 f_8 和 f_9 受OR的影响比较大. 随着取值的增加, 第200代的收敛精度在逐渐减小且有较明显的波动. 但是, 这5个测试函数在达到最大迭代次数时都可以收敛到极小值, 说明在相同的迭代进程中, OR的取值将会影响算法的收敛速度.

表 4　OR敏感性测试实验($T = 200$)

funs.	OR								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
f_1	1.93e-07	3.95e-10	2.74e-11	1.49e-11	1.38e-12	1.22e-12	1.52e-13	1.65e-13	1.22e-13
f_4	9.34e-07	3.80e-09	1.18e-09	1.30e-12	3.99e-13	4.34e-12	3.62e-12	1.97e-12	8.82e-13
f_6	9.99e-01	5.37e-06	1.13e-07	2.18e-09	1.94e-11	5.58e-12	2.46e-12	8.25e-13	8.97e-14
f_8	5.26e-02	6.70e-03	1.53e-03	9.59e-04	6.45e-04	4.17e-04	2.27e-04	2.61e-04	1.81e-04
f_9	1.42e+01	1.48e+01	1.62e+01	1.12e-07	3.91e-09	8.91e-10	6.37e-12	8.84e-13	2.79e-13

表 5　OR敏感性测试实验($T = 3\,000$)

funs.	OR								
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
f_2	3.26e-07	2.14e-07	7.05e-07	7.12e-07	1.28e-06	2.03e-06	3.21e-06	1.31e-05	5.13e-05
f_3	4.41e-03	6.09e-03	1.47e-02	3.11e-02	5.32e-02	7.44e-01	3.24e+00	6.51e+00	8.86e+00
f_5	2.13e-14	1.42e-14	1.42e-14	1.24e-14	1.24e-14	1.42e-14	1.42e-14	7.11e-15	7.11e-15
f_7	3.82e-04	3.82e-04	3.82e-04	3.82e-04	3.82e-04	3.82e-04	3.82e-04	3.82e-04	3.82e-04
f_{10}	2.24e-11	2.43e-11	3.33e-11	1.98e-11	3.10e-11	4.05e-11	5.80e-11	9.99e-11	1.66e-10
f_{11}	8.00e-01	8.59e-01	1.18e+00	4.82e+00	1.49e+01	2.24e+01	1.98e+01	2.68e+01	2.42e+01
f_{12}	1.49e+01	1.41e+01	1.50e+01	1.53e+01	1.54e+01	1.60e+01	1.66e+01	2.24e+01	1.61e+01
f_{13}	5.79e-02	4.68e-02	3.69e-02	5.54e-02	6.89e-02	6.16e-02	6.41e-02	7.63e-02	9.48e-02
f_{14}	4.30e+03	4.00e+03	4.03e+03	3.94e+03	4.07e+03	4.24e+03	4.43e+03	4.44e+03	4.19e +03

从表5的测试结果可以发现,未收敛到极小值的9个函数对OR的敏感性较低,波动幅度一般在0～3个数量级. 但除了f_5的极值随OR取值增加而变优以外,其他函数都呈现出基本不变(f_7、f_{12}、f_{13}、f_{14})或者变差(f_2、f_3、f_{10}、f_{11})的趋势. 由此可见,OR的波动对这些函数的影响不大,却在一定程度上决定了算法的求解质量.

基于上述分析可以得出OR对不同测试函数的影响程度会有明显差异的结论,即对表4中的函数进行优化时可以采用较大的OR来加快收敛速度,而对表5中的函数则偏向用较小的OR提高算法的收敛精度. 不过,考虑到算法的通用性和整体性并综合实验数据,本文给出OR的取值范围在[0.2, 0.4]之间,更有利于加强算法的求解能力. 此外,表3的数据结果显示:对函数$f_1 \sim f_9$优化时,OBL表现出较强的促进作用,在M-P策略中占据重要地位;但是,从OR的敏感性测试中可以看出,频繁使用该策略反而会降低部分函数的求解精度. 所以,充分发挥OBL策略需要合理地控制学习强度以及应用范围.

3　结　论

本文在融合粒子重组学习和概率性混合变异策略的基础上,提出了一种动态多种群优化算法. 利用重构粒子引导调整速度更新方向,并与标准粒子群搜索探测进行一般性对比分析,在保持子种群精英学习的基础上考虑了群体的多样性和收敛性;算法中采用M-P策略对全局最优解进行逐维变异,扩大群体搜索范围的同时增强了邻域探索能力,从而较好地保持了算法的开发与勘探能力之间的平衡. 实验结果表明,本文提出的混合改进措施具有一定协作互补作用,DMPSORH的性能在叠加各策略之后比所选对比算法表现更好,具有较为明显的优势. 本文算法力求综合多方信息以优化种群搜索,但还存在一定不足,尤其面向多峰函数优化问题时种群间隐含信息的提取与利用还有待于进一步深入研究. 后续工作将结合算法对工业生产中的高维优化和多目标优化问题进行应用研究.

参考文献(References)

[1] Kennedy J, Eberhart R. Particle swarm optimization[C]. Proceedings of the IEEE International Conference on Neural Networks. Perth: IEEE Press, 1995: 1942-1948.

[2] Hu W, Yen G G, Zhang X. Multi-objective particle swarm optimization based on Pareto entropy[J]. Journal of Software, 2014, 25(5): 1025-1050.

[3] Han H G, Lu W, Hou Y, et al. An adaptive-PSO-based self-organizing RBF neural network[J]. IEEE Transactions on Neural Networks and Learning Systems, 2018, 29(1): 104-117.

[4] Teng Q Z, Tang T, Li Z J, et al. Three-dimensional reconstruction of sandstone section image based on particle swarm optimization[J]. Journal of Electronics & Information Technology, 2011, 33(8): 1871-1876.

[5] Xia X W, Liu J N, Gao K F, et al. Particle

swarm optimization algorithm with reverse-learning and local-learning behavior[J]. Chinese Journal of Computers, 2015, 38(7): 1397-1407.

[6] Shi Y H，Eberhart R C. A modified particle swarm optimizer[C]. Proceedings of IEEE ICEC Conference. Piscataway: IEEE,1998: 69-73.

[7] Yang X F, Liu S L. Dynamic adjustment strategies of inertia weight in particle swarm optimization algorithm[J]. International Journal of Control and Automation, 2014, 7(5): 353-364.

[8] Ratnaweera A, Halgamuge S K, Watson H C. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 240-255.

[9] Zhang S P, Zhong W B. Hybrid particle swarm optimization algorithm of new learning factors and constraint factor[J]. Application Research of Computers, 2015, 32(12): 3626-3628.

[10] Chen Y G, Li L X, Peng H P, et al. Particle swarm optimizer with two differential mutation[J]. Applied Soft Computing, 2017, 61: 314-330.

[11] Li W F, Liang X L, Zhang Y, et al. Research on PSO with clusters and heterogeneity[J]. Acta Electronica Sinica, 2012, 40(11): 2194-2199.

[12] Niu B, Huang H L, Tan L J, et al. Symbiosis-based alternative learning multi-swarm particle swarm optimization[J]. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 2017, 14(1): 4-14.

[13] Ni H M, Liu Y J, Li P C. Adaptive dynamic reconfiguration multi-objective particle swarm optimization algorithm[J]. Control and Decision, 2015, 30(8): 1417-1422.

[14] Ma C, Deng C, Xiong Y, et al. An intelligent optimization algorithm based on hybrid of GA and PSO[J]. Journal of Computer Research and Development, 2013, 50(11): 2278-2286.

[15] Feng H K, Bao J S, Jin Y. Hybrid particle swarm optimization algorithm for multiple vehicle dragging goods problem[J]. Computer Integrated Manufacturing Systems, 2010, 16(7): 1427-1436.

[16] Li F, Liu J C, Shi H T, et al. Multi-objective particle swarm optimization algorithm based on decomposition and differential evolution[J]. Control and Decision, 2017, 32(3): 403-410.

[17] Kiran M S, Gundüz M. A recombination-based hybridization of particle swarm optimization and artificial bee colony algorithm for continuous optimization problems[J]. Applied Soft Computing, 2013, 13(4): 2188-2203.

[18] Kadirkamanathan V, Selvarjah K, Fleming P J. Stability analysis of the particle dynamics in particle swarm optimizer[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(3): 245-255.

[19] Gao Y L, Yan P. Unified optimization based on multi-swarm PSO algorithm and cuckoo search algorithm[J]. Control and Decision, 2016, 31(4): 601-608.

[20] Tong Q J, Li M, Zhao Q. An improved particle swarm optimization algorithm based on classification[J]. Modern Electronics Technique, 2019, 42(19): 11-14.

[21] Sabat S L, Ali L, Udgata S K. Integrated learning particle swarm optimizer for global optimization[J]. Applied Soft Computing Journal, 2011, 11(1): 574-584.

[22] Guo Y N, Liu D D, Cheng J, et al. Adaptive cultural algorithm adopting mixed mutation[J]. Acta Electronica Sinica, 2011, 39(8): 1913-1918.

[23] Xin B, Chen J, Zhang J, et al. Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: A review and taxonomy[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part C, 2012, 42(5): 744-767.

[24] Clerc M, Kennedy J. The particle swarm: Explosion stability and convergence in a multi-dimensional complex space[J]. IEEE Transactions on Evolutionary Computation, 2002, 6(1): 58-73.

[25] Hu W, Li Z S. A simpler and more effective particle swarm optimization algorithm[J]. Journal of Software, 2007, 18(4): 861-868.

[26] Zhou X Y, Wu Z J, Wang H, et al. Elite opposition-based particle swarm optimization[J]. Acta Electronica Sinica, 2013, 41(8): 1647-1652.

[27] Li Y M, Zhang H F, Cheng L, et al. Particle swarm optimization based on multi-subgroup and subspace learning strategy[J]. Computer & Digital Engineering, 2018, 46(9): 1768-1772.

[28] Zhou W F, Liang X L, Tang K X, et al. Hybrid particle swarm optimization algorithm with topological time-varying and search disturbance[J]. Journal of Computer Applications, 2020, 40(7): 1913-1918.

[29] Liang J J, Qin A K, Suganthan P N, et al. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(3): 281-295.

作者简介

唐可心(1996−), 女, 硕士生, 从事智能优化算法、人工神经网络优化的研究, E-mail: 539974009@qq.com;

梁晓磊(1985−), 男, 副教授, 博士, 从事智能优化、复杂系统建模与仿真等研究, E-mail: liangxiaolei@wust.edu.cn;

周文峰(1995−), 男, 硕士生, 从事智能优化的研究, E-mail: 710158203@qq.com;

马千慧(1998−), 女, 硕士生, 从事优化算法、柔性车间调度优化的研究, E-mail: 2778380498@qq.com;

张煜(1974−), 男, 教授, 博士生导师, 从事交通运输、物流领域内的系统建模、优化算法、智能决策、仿真与分析等研究, E-mail: sanli@whut.edu.cn.

(责任编辑：李君玲)