控制与决策
Control and Decision

数据驱动选择策略的多目标差分进化算法

侯莹, 吴毅琳, 白星, 韩红桂

引用本文:
侯莹,吴毅琳,白星,韩红桂. 数据驱动选择策略的多目标差分进化算法[J]. 控制与决策, 2023, 38(7): 1816−1824.

在线阅读 View online: https://doi.org/10.13195/j.kzyjc.2021.1957

数据驱动选择策略的多目标差分进化算法

侯 莹[1,2†], 吴毅琳[1,2], 白 星[1,3], 韩红桂[1,2,3]

(1. 北京工业大学 信息学部，北京 100124；2. 北京工业大学 教育部数字社区工程研究中心，北京 100124；3. 北京工业大学 计算智能与智能系统北京市重点实验室，北京 100124)

摘 要: 针对多目标差分进化算法求解复杂多目标优化问题时,最优解选择策略中非支配排序计算复杂度高的问题,提出一种数据驱动选择策略的多目标差分进化(MODE-DDSS)算法. 首先,设计多目标差分进化算法的优化解排序等级评估准则,建立基于评估准则的优化解排序等级评估库;其次,设计基于优化解双向搜索机制和无重复比较机制的数据驱动选择策略,实现优化解的高效搜索和快速排序;最后,构建数据驱动选择策略的多目标差分进化算法,降低算法在最优解选择操作中的时间复杂度,提高算法的寻优效率. 实验结果表明,所提出的MODE-DDSS算法能够有效减少最优解在选择过程中的比较次数,提升多目标差分进化算法解决复杂多目标优化问题的寻优效率.

关键词: 数据驱动；选择策略；非支配排序；多目标优化；差分进化算法；寻优效率

中图分类号: TP273 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2021.1957 **开放科学(资源服务)标识码(OSID):**

引用格式: 侯莹,吴毅琳,白星,等. 数据驱动选择策略的多目标差分进化算法[J]. 控制与决策, 2023, 38(7): 1816-1824.

Multi-objective differential evolution algorithm with data-driven selection strategy

HOU Ying[1,2†], *WU Yi-lin*[1,2], *BAI Xing*[1,3], *HAN Hong-gui*[1,2,3]

(1. Faculty of Information Technology，Beijing University of Technology，Beijing 100124，China；2. Engineering Research Center of Digital Community of Ministry of Education，Beijing University of Technology，Beijing 100124，China；3. Beijing Key Laboratory of Computational Intelligence and Intelligent System，Beijing University of Technology，Beijing 100124，China)

Abstract: The multi-objective differential evolution (MODE) algorithm has high computational complexity of the selection strategy in solving complex multi-objective optimization problems. To address this issue, a multi-objective differential evolution with data-driven selection strategy (MODE-DDSS) is proposed. First, the ranking evaluation criteria of optimization solutions is designed, and the ranking evaluation database of optimization solutions based on evaluation criteria is established. Then, a data-driven selection strategy, based on a two-way search mechanism and a non-repeated comparison mechanism, is designed to search and compare the optimal solutions efficiently, and select the optimal solutions. Finally, a multi-objective differential evolution algorithm with the data-driven selection strategy is constructed, which reduces the complexity of optimal solution selection operation and improves the optimization efficiency of the algorithm. Experimental results show that the proposed MODE-DDSS algorithm can effectively reduce the number of comparison operations in the selection strategy, and improve the efficiency of the multi-objective differential evolution algorithm in solving complex multi-objective optimization problems.

Keywords: data-driven; selection strategy; non-dominated sorting; multi-objective optimization; differential evolution algorithm; optimization efficiency

0 引 言

多目标优化问题是指需要同时优化两个及两个以上目标的优化问题,当目标个数超过3个时称为高维多目标优化问题,广泛存在于工程实践和科学研

收稿日期: 2021-11-11；**录用日期:** 2022-03-15.

基金项目: 国家自然科学基金青年项目(61903010)；国家自然科学基金杰出青年基金项目(62125301)；国家重点研发计划项目(2018YFC1900800)；北京高校卓越青年科学家项目(BJJWZYJH01201910005020)；国家自然科学基金重大项目(61890931)；国家自然科学基金创新研究群体项目(62021003).

责任编委: 侯忠生.

†通讯作者. E-mail: houying17@sina.com.

*本文附带电子附录文件,可登录本刊官网该文“资源附件”区自行下载阅览

究中[1-4]. 多目标差分进化(multi-objective differential evolution, MODE)算法是一种元启发式优化算法,能够有效求解多目标优化问题[5-6]. 但是,当目标复杂度增加,即目标维度和解集规模增大时,优化解的性能不易被评估,选择最优解所需的比较操作次数激增,MODE算法的寻优效率下降. 因此,提升MODE算法寻优效率的关键,不仅要改进最优解的产生策略,更要关注最优解的选择策略,有效选择并保留高质量的最优解.

在求解复杂多目标优化问题时,针对如何提升MODE算法最优解选择效率的难题,学者们对选择策略的设计开展了大量研究工作[7-12]. 文献[13]采用快速非支配排序方法和拥挤距离完成种群的更新,利用前沿等级信息选出参与差分变异的父代个体,基于相关性排序选出部分目标维度,完成优化解的比较,提升了MODE算法对最优解的选择效率. 然而,该方法在构建目标子集时,需要设置参数以计算目标相关性. 文献[14]设计了用于存储进化过程信息的优化解档案集,引入档案集截断机制,以增加寻优过程的选择压力,保证在求解复杂优化问题时仍然能够获取一组收敛性和多样性较好的最优解. 此方法利用进化过程信息设计最优解选择策略,加快了种群的收敛速度,提高了算法的搜索性能. 但是,复杂多目标优化问题不易被定量描述,如何设定档案集的规模数和截断代数成为难题. 文献[15]提出一种基于反世代距离的进化算子,实现了变异算子的自适应调整,并利用快速非支配排序和拥挤距离选出最优解集,提高了MODE算法在求解高维和复杂多目标优化问题时最优解的选择效率[15]. 以上改进策略有效地提高了算法的寻优效率,但是如何设置额外的参数值是一个难题,影响了算法的实用性[16-17].

为了减少额外参数的引入,进一步提升算法的寻优效率,学者们充分利用算法自身蕴含的信息,改进选择策略[18-20]. 刘明凯等[21]提出一种加强非支配解占优策略,利用优化解函数值之间的几何夹角信息,描述非支配占优关系和拥挤距离,设计了基于夹角的选择操作机制来加快种群的收敛速度,并成功应用于电-气互联系统. 基于优化目标的函数值对优化解集进行预排序,利用历史比较信息忽略被支配的优化解,有效降低了非支配排序过程的比较次数,加快了选择操作的速度[22-23]. 文献[24]提出了一种高效非支配排序顺序搜索策略(efficient non-dominated sorting-sequential search, ENS-SS). 该策略以第1个目标对优化解进行预排序,利用优化解在非支配排序过程中的历史比较信息,仅将待排序优化解与已排序优化解进行比较,完成所有前沿的划分,显著提升了排序效率. 然而,由于每一个待分配到当前前沿的优化解,都需要和已分配到当前前沿的优化解逐一进行比较,增加了非支配排序的时间. 以上算法在一定程度上提高了MODE算法在求解复杂多目标优化问题时的寻优速度,但是非支配排序过程的数据没有被充分调用,仍然存在不同前沿优化解和同一前沿优化解间的非必要比较,目标维度和解集规模增加时,算法选择效率明显降低,寻优速度变慢[25-26]. 因此,充分利用选择阶段产生的优化解性能评估数据,提取数据中蕴含的特征,研究符合选择操作规律的最优解选择策略,设计适用于复杂多目标优化问题的优化算法,仍是一个值得研究的问题[27-28].

针对MODE算法在复杂多目标优化中选择策略寻优效率较低的问题,本文提出一种数据驱动选择策略的多目标差分进化(multi-objective differential evolution with data-driven selection strategy, MODE-DDSS)算法,建立基于优化解排序等级评估准则的排序等级评估库,对优化解进行高效搜索和无重复比较,实现优化解前沿的有效划分,降低多目标差分进化算法最优解选择操作的复杂度. 实验结果表明,所提出算法能够有效提升MODE算法在复杂多目标优化问题中的寻优速度,减少算法的寻优时间.

1　相关概念

1.1　Pareto占优理论

多目标优化问题的解,通常是由一组最优解组成的Pareto最优解集,相关概念如下.

1) 支配与非支配关系.

两个优化解$\boldsymbol{p}$和$\boldsymbol{q}$满足

$$\begin{cases} \forall f_m(\boldsymbol{p}) \leqslant f_m(\boldsymbol{q}), \\ \exists f_m(\boldsymbol{p}) < f_m(\boldsymbol{q}), \end{cases} m = 1, 2, \ldots, M \tag{1}$$

时,认为$\boldsymbol{p}$支配$\boldsymbol{q}$,记为$\boldsymbol{p} \prec \boldsymbol{q}$,否则$\boldsymbol{p}$和$\boldsymbol{q}$构成非支配关系. 其中: $f_m(\boldsymbol{p})$和$f_m(\boldsymbol{q})$分别是优化解$\boldsymbol{p}$和$\boldsymbol{q}$在第m个目标上的函数值, M是目标数量.

2) Pareto前沿.

满足式(2)的优化解$\boldsymbol{x}^*$称为最优解,所有$\boldsymbol{x}^*$构成Pareto最优解集,即

$$\boldsymbol{x}^* \prec \boldsymbol{x}. \tag{2}$$

使用优化算法获得的解集通常为近似最优解集,在目标空间的映射称为近似Pareto前沿.

3) 非支配排序.

通过确定优化解间的非支配关系,将优化解划分

成多级前沿的过程称为非支配排序. 非支配排序得到的前沿具有以下特点:

1) 同一前沿任意两个解$\boldsymbol{p}$和$\boldsymbol{q}$构成非支配关系.

2) 后一级前沿中不存在支配其前一级前沿的优化解,即对于某一级前沿的任一优化解$\boldsymbol{s}$,前一级前沿至少存在一个支配它的解$\boldsymbol{t}$,如下所示:

$$\forall \boldsymbol{s} \in F_{k+1}, \exists \boldsymbol{t} \in F_k \to \boldsymbol{t} \prec \boldsymbol{s},\ k = 1, 2, \ldots, K. \quad (3)$$

其中: F_k是第k个前沿, K是前沿数量.

求解复杂多目标优化问题时,需要设置较大的解集规模N或目标维度M,此时优化解之间的比较次数变多,非支配排序的复杂度增加. 因此,当解集规模和目标维度增加时,非支配排序中的比较次数增多,选择操作运行时间加长.

1.2 多目标差分进化算法

多目标差分进化算法是一种元启发式多目标进化优化算法,基本流程包括: 种群初始化,变异操作,交叉操作和选择操作. 首先,设置种群规模N,变异操作过程中的变异率F和交叉操作过程中的交叉率C_r,并随机生成初始种群. 然后,对第t代种群$\boldsymbol{X}_t$进行变异和交叉操作,生成新种群$\boldsymbol{W}_t$,其中常用的差分进化算子DE/rand/1可表示为

$$\boldsymbol{v}_{i,t} = \boldsymbol{x}_{i,t} + F(\boldsymbol{x}_{r_1,t} - \boldsymbol{x}_{r_2,t}). \quad (4)$$

$$w_{ji,t} = \begin{cases} v_{ji,t},\ \mathrm{rand}_{ij}[0,1] \leqslant C_r; \\ x_{j,t},\ \mathrm{otherwise}. \end{cases} \quad (5)$$

其中: $i = 1, 2, \ldots, N, j = 1, 2, \ldots, D, r_1$和$r_2$是两个异于$i$且互不相同的属于$[0, N]$的整数, $\boldsymbol{v}_{i,t}$为变异个体, $w_{ji,t}$为试验个体$\boldsymbol{w}_{i,t}$的第j维分量, t是当前进化代数. 最后,采用选择策略,从父代种群和新种群组成的候选解集中选出NP个个体构成下一代种群$\boldsymbol{X}_{t+1}$. 如此反复,直到满足寻优停止条件,输出最优解集$\boldsymbol{X}$.

2 数据驱动的最优解选择策略

最优解选择策略的设计,既要满足选择阶段的择优需求,又要避免繁琐的计算和过多的参数设置,从而实现快速寻找并保留最优解. 在最优解选择阶段,非支配排序操作会产生大量的优化解排序等级数据,这些数据直接反应了优化解的优化性能. 因此,对优化解的排序等级数据进行提取和利用,是提高最优解选择效率的关键.

2.1 数据的描述与存储

1) 数据的描述.

对优化解排序等级数据进行描述,是有效存储和利用数据的基础. 以求解最小化问题为例,设计优化解排序等级评估准则如下: 根据目标函数值对优化解集进行升序排序后,目标函数值小的优化解不易被支配,则优化解排序等级数值小; 目标函数值大的优化解,其被支配的概率较高,则优化解排序等级数值大; 目标函数值相等的优化解具有同一排序等级,确定其中任一优化解的排序等级数值,则能够确定其余优化解的排序等级数值.

2) 数据的存储.

对数据进行有序存储,是高效搜索数据和更新数据信息的重要环节. 以求解最小化问题为例,设计基于优化解排序等级评估准则的评估库. 首先,依据第1个目标函数值的升序情况,对优化解集进行排序,同时合并在所有目标上函数值完全相等的优化解,目标函数值越小的优化解,排序等级越小,依次存入所有优化解,并标记索引$r, 1 \leqslant r \leqslant N'(N' \leqslant N)$,此时优化解集记为$\boldsymbol{X}'$,由$N'$个解构成. 然后,依次根据第$j(j = 2, 3, \ldots, M)$个目标对$\boldsymbol{X}'$进行全排序,若优化解在第$j$个目标上的函数值相等,则索引小的优化解排在前面,最终得到大小为$N' \times M$的优化解排序等级表,完成优化解排序等级评估库的建立.

为了提高优化解排序等级评估库中信息的提取效率,采用堆排序方式,获取评估库中一行或者一列的数据. 定义相关概念如下:

1) 全排序行. 排序等级相同的优化解组成的集合称为全排序行,记为$\boldsymbol{R}_i, i = 1, 2, \ldots, N', |\boldsymbol{R}_i| \leqslant M$.

2) 全排序列. 按照目标j得到的优化解集合称为全排序列,记为$\boldsymbol{C}_j, j = 1, 2, \ldots, M$. 在全排序列$\boldsymbol{C}_j$中,任意两个优化解$\boldsymbol{p}$和$\boldsymbol{q}$满足$\boldsymbol{p}$在$\boldsymbol{q}$前面时存在两种情况,一是$\boldsymbol{p}$支配$\boldsymbol{q}$,二是$\boldsymbol{p}$和$\boldsymbol{q}$构成非支配关系且至少在一个异于$\boldsymbol{C}_j$的全排序列中满足$\boldsymbol{q}$在$\boldsymbol{p}$前面.

3) 最好全排序等级. 优化解在不同全排序列中最小的排序等级.

2.2 数据的搜索与比较

1) 双向搜索机制.

优化解排序等级评估库中存储了大量表示优化解性能的数据,对其进行有效搜索,是提高数据利用率的关键. 基于优化解排序等级评估库中的数据,设计先行后列,自下而上的双向搜索机制. 首先,根据待排序优化解的最好全排序等级,确定其在评估库中所处的位置,作为开始搜索的行; 然后,基于全排序列的存储特性,在待排序优化解所在列中,从待排序优化解的位置开始,自下而上地搜索,将待排序优化解与

属于当前前沿的优化解进行比较.

2) 无重复比较机制.

在优化解的比较过程中,为了避免已经确定前沿的优化解和被支配的待排序优化解重复参与比较,引入优化解被支配信息. 假设当前待排序优化解的索引为r,定义r的被支配信息为$\mathrm{DI}(r)$. 当r的前沿等级已确定时,用无穷数表示r的被支配信息,即$\mathrm{DI}(r)=\infty$;当r的前沿等级未确定,并且r在列搜索过程中被属于第k级前沿$\boldsymbol{F}_k$的任一优化解支配时,用k表示r的被支配信息,即$\mathrm{DI}(r)=k$.

在行搜索过程中,调用被支配信息,若$\mathrm{DI}(r)=\infty$则不参与比较,以避免对已排序优化解进行重复比较;当r的被支配信息小于k时,说明r满足参与第k级前沿比较的条件,应开始对其进行列搜索,以实现优化解数据的无重复比较.

2.3 数据驱动的最优解选择策略

1) 选择策略相关参数的初始化.

对优化解前沿向量Front,前沿数量K和被支配信息矩阵DI进行初始化,如下式所示.

$$\begin{cases}\mathrm{Front}=[0\quad\ldots\quad 0]\in\Omega^{1\times N'},\\ K=1,\\ \mathrm{DI}=[0\quad\ldots\quad 0]\in\Omega^{1\times N'}.\end{cases}\tag{6}$$

其中: Ω是Front和DI所在空间,大小为$1\times N'$.

2) 优化解前沿的划分.

在确定第k级前沿$\boldsymbol{F}_k$时,采用先行后列的双向搜索机制,从优化解排序等级评估库第k行开始搜索. 假设当前待排序优化解的索引值为r,在优化解排序等级评估库中,r首次被访问的位置记为(i,j),从位置$(i-1,j)$开始在列中向上搜索,所有属于$\boldsymbol{F}_k$且在目标j上全排序等级小于i的优化解组成比较集$\boldsymbol{S}$.

基于无重复比较机制,根据当前待排序优化解与$\boldsymbol{S}$中优化解间的支配关系划分前沿,即:

① $\boldsymbol{S}=\varnothing$. $\boldsymbol{C}_j$中不存在比当前待排序优化解好并且属于$\boldsymbol{F}_k$的优化解,当前待排序优化解直接分配至$\boldsymbol{F}_k,\mathrm{DI}(r)=\infty$.

② $\boldsymbol{S}\neq\varnothing$且当前待排序优化解被支配. $\boldsymbol{C}_j$中存在比当前待排序优化解好且属于$\boldsymbol{F}_k$的优化解,$\mathrm{DI}(r)=k$.

③ $\boldsymbol{S}\neq\varnothing$且当前待排序优化解不被支配. 在$k$列自下而上搜索到位置$(k,j)$,且当前待排序优化解仍满足这一情况时,当前待排序优化解分配至$\boldsymbol{F}_k,\mathrm{DI}(r)=\infty$.

3) 最优解的选择.

将所有优化解划分到对应前沿后,从第1级前沿开始,结合最后1级前沿的拥挤距离,选择所需数量的优化解组成Pareto最优解集.

2.4 数据驱动选择策略的复杂度分析

1) 时间复杂度.

基于以上分析,数据驱动的最优解选择策略(data-driven selection strategy, DDSS)的主要过程包括优化解排序等级评估库的构建以及优化解前沿的划分.

采用堆排序建立排序等级评估库,对应的时间复杂度为$O(MN\log N)$.

在优化解前沿的划分阶段,当每个优化解单独构成一级前沿时,DDSS具有最好时间复杂度. 此时的比较次数为MN,对应的时间复杂度为$O(MN)$. 因此,DDSS的最好时间复杂度为$O(MN\log N)$.

当所有优化解在同一前沿上时,DDSS具有最坏时间复杂度. 第1全排序行的优化解自动被分配到第1前沿,剩余优化解的比较次数记为T^C_{worst}. 由于N一般要比M大得多,可根据N和M的倍数关系,讨论DDSS的最坏时间复杂度.

当$N\%M=0$时,T^C_{worst}的计算方式如下所示(DDSS的时间复杂度为$O(N^2)$):

$$\begin{aligned}&T^C_{\mathrm{worst}}=\\&M^2\frac{N-M}{M}\Big(\frac{N-M}{M}-1\Big)=\\&N^2-3MN+2M^2.\end{aligned}\tag{7}$$

当$N\%M\neq 0$时,令$a=N\%M(0<a<M)$,T^C_{worst}的计算方式如下所示(此时DDSS的时间复杂度仍为O(N^2)):

$$\begin{aligned}&T^C_{\mathrm{worst}}=\\&M^2\frac{N-M-a}{M}\Big(\frac{N-M-a}{M}-1\Big)+aM\frac{N-a}{M}=\\&N^2-3MN+2M^2-aN+3aM+a^2.\end{aligned}\tag{8}$$

因此,DDSS的最坏时间复杂度为$O(N^2)$.

2) 空间复杂度.

优化解排序等级评估阶段的空间消耗在排序表上,空间复杂度为$O(MN)$;前沿划分阶段不需要额外的辅助空间,空间复杂度为$O(1)$. 因此,DDSS的空间复杂度为$O(MN)$.

3) 复杂度对比分析.

为了进一步说明DDSS的复杂度,将其与快速非支配排序策略(fast non-dominated sorting, FNDS)[9]、ENS-SS策略[24]、基于树结构的高效非支配排序策略

(efficient tree-based nondominated sorting, T-ENS)[29]和BOS策略[30]进行复杂度对比.

由表1可知,在最坏情况下,数据驱动选择策略的时间复杂度均好于其余算法. 在优化过程中,N与M通常满足N远远大于M,因此,当N具有较大值时,前沿数量增加,数据驱动选择策略的优势将更加明显. 当M较大时,种群存在大量非支配解,T-ENS策略的时间复杂度为$O(M^2N\ln N/\ln M)$,具有更短的寻优时间,但T-ENS策略需要更大的存储空间,算法的空间复杂度较大.

表1 不同非支配排序策略的复杂度

非支配排序策略	时间复杂度		空间复杂度
	最好情况	最坏情况	
FNDS[9]	$O(MN^2)$	$O(MN^2)$	$O(N^2)$
ENS-SS[24]	$O(MN\sqrt{N})$	$O(MN^2)$	$O(N)$
T-ENS[29]	$O(M^2N\ln N/\ln M)$	$O(MN^2)$	$O(N^2)$
BOS[30]	$O(MN\log N)$	$O(MN^2)$	$O(MN)$
DDSS	$O(MN\log N)$	$O(N^2)$	$O(MN)$

3 MODE-DDSS算法

MODE-DDSS算法的实现流程如图1所示. 首先,设置最大进化代数T,令$t=0$,根据N、M、F和C_r生成初始种群;其次,根据式(4)和(5),生成实验个体$\boldsymbol{w}_{i,t}$;然后,采用数据驱动选择策略,从$\boldsymbol{X}_t$和$\boldsymbol{W}_t$构成的候选解集中选出下一代的种群.

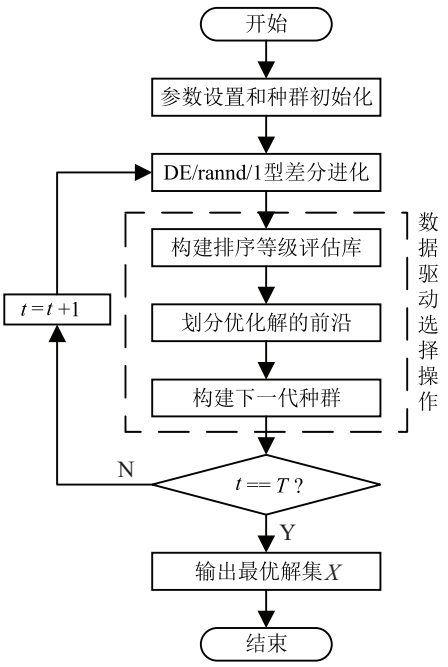


图1 MODE-DDSS算法的实现流程

1) 构建优化解排序等级评估库.

首先,按照第1个目标函数的升序值,对优化解集进行排序,合并函数值完全相等的优化解,并完成索引标记,此时解集大小为N'. 然后,按照第2到第M个目标对优化解集进行全排序,生成大小为$N'\times M$的优化解排序等级评估库.

2) 划分优化解的前沿.

对选择策略的相关参数进行初始化,设置方式如式(6)所示. 在确定第k个前沿时,采用先行后列的双向搜索机制,遍历优化解排序等级评估库,并采用无重复比较机制,将优化解划分到相应的前沿.

3) 构建下一代种群.

当已排序的优化解数目大于等于NP时,停止非支配排序. 计算最后一级前沿中优化解的拥挤距离,选择前沿等级小于最后一级前沿的优化解,以及最后一级前沿中拥挤距离较大的优化解,共NP个优化解作为下一代的种群,组成第$t+1$代种群$\boldsymbol{X}_{t+1}$.

4 实验设计及结果分析

4.1 实验设计

以优化目标维度和解集规模为变量,从算法的寻优时间和非支配排序过程的比较次数两个角度对算法性能进行测试. 为了验证MODE-DDSS算法的有效性,选取具有不同前沿特征的测试函数进行实验,各测试函数的特点如表2所示.

表2 测试函数的特点

测试函数	所属系列	目标数目	前沿特点
DTLZ1	DTLZ	$M\geqslant 2$	连续,线性
DTLZ7	DTLZ	$M\geqslant 2$	不连续
MW3	MW	$M=2$	不连续
MaF1	MaF	$M\geqslant 2$	连续,线性
MaF2	MaF	$M\geqslant 2$	连续,凹的

所有实验均在PlatEMO V2.9[31]测试平台上进行,计算机的处理器型号为3.00 GHz Intel Core i5-9500 CPU,运行环境为Matlab R2018b. 根据不同测试函数的前沿特点和算法运行的经验值,设置DTLZ、MW和MaF系列测试函数对应的进化代数T分别为100、500和250,F和C_r采用PlatEMO平台的默认值,分别设置为0.5和1.0. 选用以FNDS、ENS-SS、T-ENS和BOS为选择策略的多目标差分进化算法(分别记为MODE算法、MODE-ENS算法、MODE-TENS算法和MODE-BOS算法)进行对比实验. 为保证结果的公平性,采用不同策略的算法在每个算例中的参数设置保持一致.

4.2 目标维度变化时的实验结果与分析

为了验证MODE-DDSS算法求解高维多目标优化问题的有效性,选取DTLZ7和MaF1为测试函数. 由于决策变量维度D不影响选择效率,为了便于进行对比实验,D使用平台默认值,DTLZ7和MaF1分别设置为$D=M+19$和$D=M+9$. 设置M的

初始值为5, 以步长5递增至15, N分别为500和1 000, 独立运行10次, 结果如表3和表4所示, 数据以“均值(标准差)”的形式表示, 最好的结果已标粗显示, 表格最后一行是采用Wilcoxon符号秩检验的统计结果, 以“+/−/≈”的形式表示对应的算法好于/差于/接近MODE-DDSS算法.

表 3　目标维数变化时不同算法的寻优时间

函数	N/个	M/个	寻优时间/s				
			MODE-DDSS	MODE	MODE-ENS	MODE-TENS	MODE-BOS
DTLZ7	500	5	**6.66(0.27)**	9.92(0.27)−	6.86(0.24)≈	6.75(0.22)≈	7.2(0.25)−
		10	**6.79(0.20)**	13.14(0.40)−	7.31(0.18)−	7.09(0.28)−	7.40(0.20)−
		15	**6.70(0.16)**	17.02(0.32)−	7.25(0.14)−	7.29(0.17)−	7.38(0.13)−
	1 000	5	**16.04(0.32)**	30.07(0.33)−	16.75(0.48)−	16.40(0.39)−	17.22(0.35)−
		10	**16.90(0.46)**	37.66(0.70)−	18.92(0.39)−	17.60(0.44)−	18.09(0.56)−
		15	**16.56(0.25)**	51.09(0.90)−	18.96(0.57)−	18.37(0.41)−	18.00(0.50)−
MaF1	500	5	**15.53(0.28)**	24.70(0.24)−	16.12(0.15)−	15.90(0.24)−	16.50(0.19)−
		10	**15.56(0.25)**	31.04(0.68)−	16.90(0.15)−	15.90(0.24)−	16.50(0.19)−
		15	**15.63(0.26)**	36.34(0.87)−	17.40(0.28)−	16.94(0.30)−	18.63(0.27)−
	1 000	5	**39.71(1.25)**	76.20(1.69)−	42.54(1.08)−	40.91(1.49)−	41.96(0.75)−
		10	**39.56(1.08)**	103.36(4.76)−	44.81(1.09)−	44.17(1.18)−	43.25(0.36)−
		15	**39.47(0.41)**	120.49(2.04)−	46.67(0.45)−	46.49(0.88)−	46.06(0.35)−
+/−/≈			—	0/12/0	0/11/1	0/11/1	0/11/1

表 4　目标维数变化时不同算法的排序过程比较次数

函数	N/个	M/个	比较次数/次				
			MODE-DDSS	MODE	MODE-ENS	MODE-TENS	MODE-BOS
DTLZ7	500	5	$\mathbf{1.46\times10^7(1.19\times10^6)}$	$4.95\times10^8(0.00\times10^0)-$	$5.04\times10^7(3.60\times10^6)-$	$1.58\times10^7(9.53\times10^5)-$	$2.12\times10^7(8.96\times10^5)-$
		10	$\mathbf{2.59\times10^7(1.05\times10^6)}$	$9.90\times10^8(0.00\times10^0)-$	$1.19\times10^8(6.28\times10^6)-$	$2.64\times10^7(2.08\times10^6)-$	$2.98\times10^7(4.90\times10^5)-$
		15	$\mathbf{1.81\times10^7(9.07\times10^5)}$	$1.49\times10^9(0.00\times10^0)-$	$1.05\times10^8(3.68\times10^6)-$	$3.24\times10^7(2.93\times10^6)-$	$3.02\times10^7(5.27\times10^5)-$
	1 000	5	$\mathbf{4.34\times10^7(7.62\times10^6)}$	$1.98\times10^9(0.00\times10^0)-$	$1.41\times10^8(1.43\times10^7)-$	$5.44\times10^7(4.26\times10^6)-$	$6.40\times10^7(4.53\times10^6)-$
		10	$1.05\times10^8(2.84\times10^6)$	$3.96\times10^9(0.00\times10^0)-$	$4.69\times10^8(1.40\times10^7)-$	$\mathbf{9.51\times10^7(5.70\times10^6)}+$	$1.03\times10^8(1.72\times10^6)+$
		15	$\mathbf{7.01\times10^7(1.92\times10^6)}$	$5.94\times10^9(0.00\times10^0)-$	$4.21\times10^8(1.57\times10^7)-$	$1.17\times10^8(8.28\times10^6)-$	$9.40\times10^7(1.86\times10^6)-$
MaF1	500	5	$3.66\times10^7(1.84\times10^5)$	$1.25\times10^9(0.00\times10^0)-$	$1.16\times10^8(6.74\times10^5)-$	$\mathbf{3.34\times10^7(6.72\times10^5)}+$	$4.35\times10^7(1.05\times10^5)-$
		10	$\mathbf{2.93\times10^7(6.39\times10^6)}$	$2.49\times10^9(0.00\times10^0)-$	$2.46\times10^8(6.23\times10^6)-$	$8.07\times10^7(3.09\times10^6)-$	$5.53\times10^7(4.54\times10^5)-$
		15	$\mathbf{2.97\times10^7(1.21\times10^6)}$	$3.74\times10^9(0.00\times10^0)-$	$4.16\times10^8(1.57\times10^8)-$	$1.27\times10^7(7.59\times10^6)-$	$7.37\times10^7(1.64\times10^6)-$
	1 000	5	$1.41\times10^8(5.31\times10^5)$	$4.98\times10^9(0.00\times10^0)-$	$4.45\times10^8(2.67\times10^6)-$	$\mathbf{1.11\times10^8(1.96\times10^6)}\approx$	$1.48\times10^8(3.40\times10^5)-$
		10	$\mathbf{1.15\times10^8(1.40\times10^6)}$	$9.96\times10^9(0.00\times10^0)-$	$9.58\times10^8(1.63\times10^7)-$	$2.71\times10^8(7.30\times10^6)-$	$1.72\times10^8(1.20\times10^6)-$
		15	$\mathbf{1.17\times10^8(3.24\times10^6)}$	$1.49\times10^10(0.00\times10^0)-$	$1.64\times10^9(5.50\times10^7)-$	$4.60\times10^8(1.80\times10^7)-$	$2.18\times10^8(3.09\times10^6)-$
+/−/≈			—	0/12/0	0/12/0	2/9/1	1/11/0

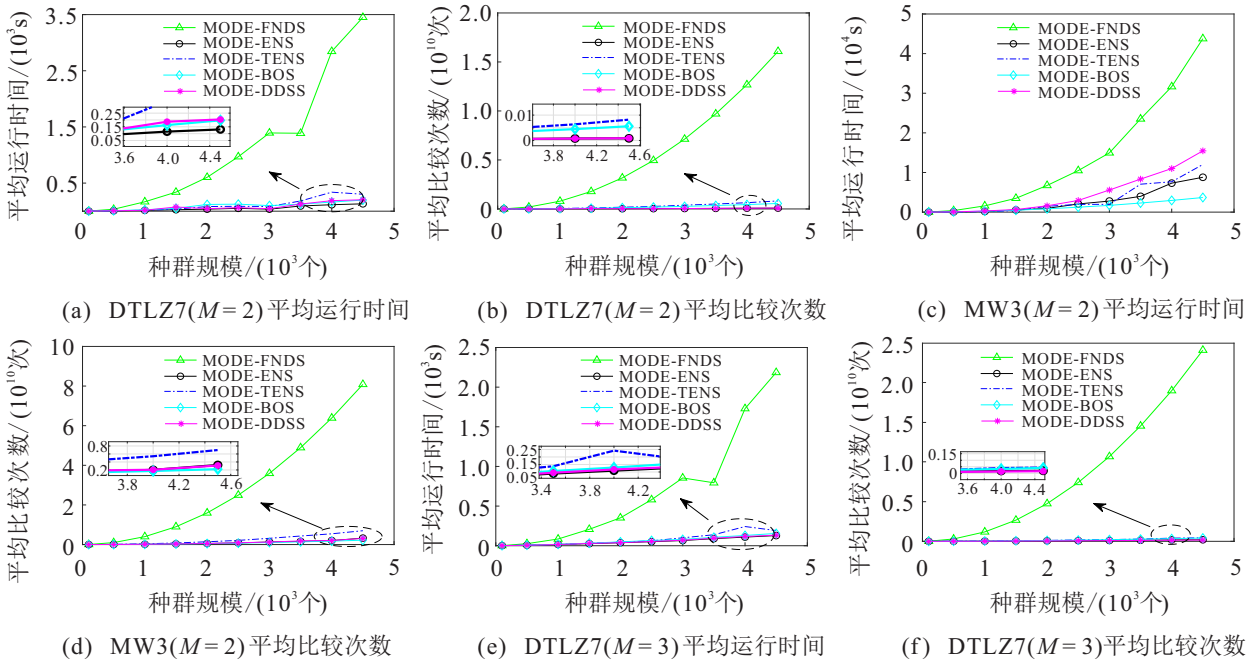
由表3的统计结果可知, 针对算法寻优时间指标, MODE-DDSS算法的结果明显优于其他算法. 种群规模固定的条件下, 算法寻优时间不随目标维数的增加而剧烈变化, 寻优效率稳定. 例如, 当MaF1为测试函数, 种群规模为500时, MODE-DDSS算法对5个、10个和15个目标函数的寻优时间均值分别为15.53 s、15.555 s和15.63 s, 增幅仅为0.17%和0.46%.

由表4的结果可知, 针对排序过程比较次数指标, MODE-DDSS算法在12个算例中取得9个优胜, MODE-TENS算法紧随其后, 取得3个优胜. 在求解15个目标函数时, MODE-DDSS算法的结果最优. 在求解5个目标函数时, MODE-DDSS算法的排序过程比较次数仅高于MODE-TENS算法. 例如, 当MaF1为测试函数, 种群规模为500时, MODE-DDSS算法(均值为3.66×10^7, 标准差为1.84×10^5), 仅高于MODE-TENS算法(均值为3.34×10^7, 标准差为6.72×10^5), 而优于MODE-FNDS算法(均值为1.25×10^9, 标准差为0)、 MODE-ENS算法(均值为1.16×10^8, 标准差为6.74×10^5)和MODE-BOS算法(均值为4.35×10^7, 标准差为1.05×10^5).

4.3　解集规模变化时的实验结果与分析

为了验证MODE-DDSS算法求解复杂多目标优化问题的有效性, 选取DTLZ1、DTLZ7、MW3、MaF1和MaF2为测试函数. 根据前沿的特点, 将以上测试函数分为两组, 一组是具有不连续前沿的函数DTLZ7和MW3; 另一组是具有连续前沿的函数DTLZ1、MaF1和MaF2. 决策变量维度D使用平台默认值, DTLZ1和DTLZ7分别设置为$D = M + 4$和

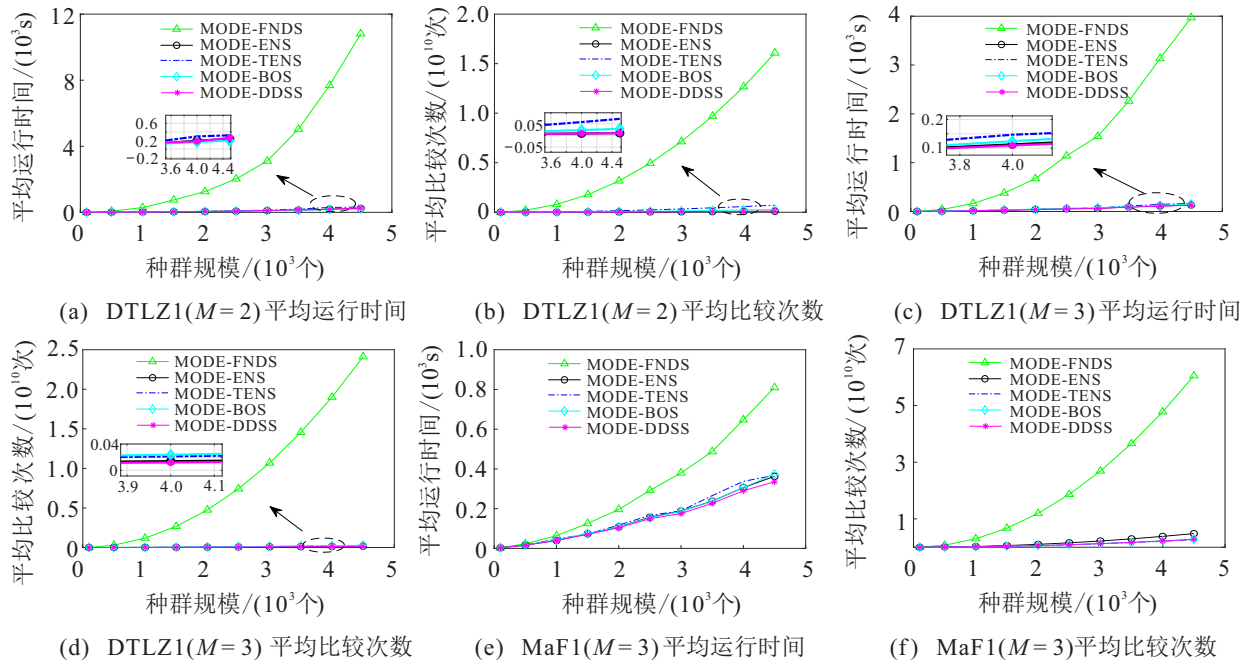
$D = M + 19$, MW3设置为$D = 15$, MaF1和MaF2设置为$D = M + 9$. 在每个算例中, 种群规模从500个, 以步长500递增至4 500个, 独立运行10次, 根据记录的指标均值, 绘制以种群规模为变量, 不同算法的寻优时间和排序过程平均比较次数的变化曲线, 如图2和图3所示.



(a) DTLZ7($M=2$)平均运行时间 (b) DTLZ7($M=2$)平均比较次数 (c) MW3($M=2$)平均运行时间

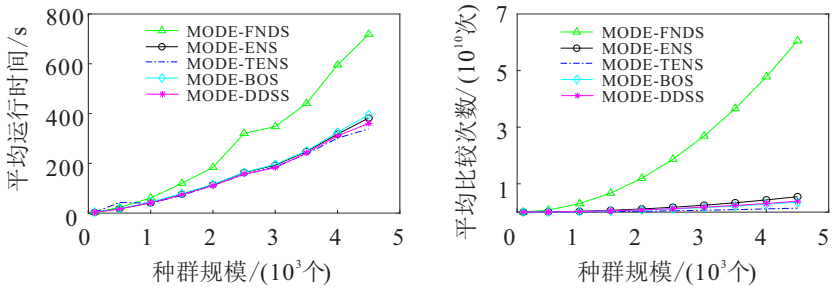
(d) MW3($M=2$)平均比较次数 (e) DTLZ7($M=3$)平均运行时间 (f) DTLZ7($M=3$)平均比较次数

图 2 不同算法平均寻优时间和排序过程平均比较次数的对比(具有不连续前沿的目标函数)



(a) DTLZ1($M=2$)平均运行时间 (b) DTLZ1($M=2$)平均比较次数 (c) DTLZ1($M=3$)平均运行时间

(d) DTLZ1($M=3$) 平均比较次数 (e) MaF1($M=3$)平均运行时间 (f) MaF1($M=3$)平均比较次数



(g) MaF2($M=3$)平均运行时间 (h) MaF2($M=3$)平均比较次数

图 3 不同算法平均寻优时间和排序过程平均比较次数的对比(具有连续前沿的目标函数)

对于具有不连续前沿的目标函数,当$M = 2$时,不同算法对DTLZ7和MW3的寻优时间和排序过程比较次数如图2(a)～图2(d)所示. 由图2(a)和图2(b)可见,MODE-DDSS算法与MODE-ENS算法的比较次数相当,均优于其他算法;MODE-DDSS算法的寻优时间略慢于MODE-ENS算法和MODE-BOS算法,但显著优于其他算法. 至于MW3,由图2(c)和图2(d)可见,MODE-DDSS算法的比较次数多于MODE-BOS算法,但优于其他算法;MODE-DDSS算法的寻优时间多于MODE-ENS算法、MODE-TENS算法和MODE-BOS算法,仅优于MODE-FNDS算法. 进一步分析MW3的特点可知,由于MW3的各个目标间存在约束关系,MODE-DDSS算法需要进行较多的判断操作,此时算法运行时间较长. 不同算法对DTLZ7($M = 3$)的寻优时间和比较次数如图2(e)和图2(f)所示,MODE-DDSS算法的比较次数和寻优时间与MODE-ENS算法相当,均优于其他算法.

对于具有连续前沿的目标函数,不同算法对DTLZ1($M = 2$)的寻优时间和比较次数如图3(a)～图3(b)所示, MODE-DDSS算法的寻优时间和比较次数均与MODE-ENS算法相当,优于其他算法. 当$M = 3$时,不同算法对DTLZ1、MaF1和MaF2的寻优时间和比较次数如图3(c)～图3(h)所示. MODE-DDSS算法对DTLZ1和MaF1的寻优时间和比较次数均优于其他所有算法,对于MaF2,MODE-DDSS算法的寻优时间略慢于MODE-TENS算法,但优于其他算法,比较次数仅次于MODE-TENS算法,与MODE-BOS算法相当. 可见,在求解具有连续前沿的多目标优化问题时,MODE-DDSS算法的表现更好.

5 结　论

围绕多目标差分进化算法在复杂多目标优化中寻优效率低的难题,本文提出了一种数据驱动选择策略的多目标差分进化算法. 该算法建立了优化解排序等级评估库,设计了基于优化解双向搜索机制和无重复比较机制的数据驱动选择策略,有效降低了选择操作的复杂度,提高了算法的寻优效率. 针对基准测试函数的实验结果表明,在求解高维优化问题和具有连续前沿的优化问题时,MODE-DDSS算法的寻优效果好于其他对比算法. 但是,对于目标函数之间存在约束的不连续前沿多目标优化问题,MODE-DDSS算法的寻优效率有待提高. 本文提出的数据驱动选择策略,原则上适用于所有基于Pareto占优的多目标优化算法. 未来将针对种群进化后期,个体全部为非支配关系的情况,兼顾最优解选择策略和差分进化算子,设计更加灵活的选择策略,进一步提高算法求解复杂多目标优化问题的寻优效率,并将算法应用于实际多目标优化问题的求解过程.

参考文献(References)

[1] Wang J H, Ren W B, Zhang Z Z, et al. A hybrid multiobjective memetic algorithm for multiobjective periodic vehicle routing problem with time windows[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2020, 50(11): 4732-4745.

[2] Han H G, Liu Z, Hou Y, et al. Data-driven multiobjective predictive control for wastewater treatment process[J]. IEEE Transactions on Industrial Informatics, 2020, 16(4): 2767-2775.

[3] 封文清, 巩敦卫. 基于在线感知Pareto前沿划分目标空间的多目标进化优化[J]. 自动化学报, 2020, 46(8): 1628-1643.
(Feng W Q, Gong D W. Multi-ob jective evolutionary optimization with ob jective space partition based on online perception of Pareto front[J]. Acta Automatica Sinica, 2020, 46(8): 1628-1643.)

[4] Wang Y, Wang L, Chen G C, et al. An improved ant colony optimization algorithm to the periodic vehicle routing problem with time window and service choice[J]. Swarm and Evolutionary Computation, 2020, 55: 100675.

[5] 侯莹, 韩红桂, 乔俊飞. 基于参数动态调整的多目标差分进化算法[J]. 控制与决策, 2017, 32(11): 1985-1990.
(Hou Y, Han H G, Qiao J F. Adaptive multi-objective differential evolution algorithm based on the dynamic parameters adjustment[J]. Control and Decision, 2017, 32(11): 1985-1990.)

[6] Hou Y, Wu Y L, Liu Z, et al. Dynamic multi-objective differential evolution algorithm based on the information of evolution progress[J]. Science China Technological Sciences, 2021, 64(8): 1676-1689.

[7] Chen X, Du W L, Qian F. Multi-objective differential evolution with ranking-based mutation operator and its application in chemical process optimization[J]. Chemometrics and Intelligent Laboratory Systems, 2014, 136(1): 85-96.

[8] 郑金华, 刘磊, 李密青, 等. 差分选择策略在复杂多目标优化问题中的研究[J]. 计算机研究与发展, 2015, 52(9): 2123-2134.
(Zheng J H, Liu L, Li M Q, et al. Difference selection strategy for solving complex multi-objective problems[J]. Journal of Computer Research and Development, 2015, 52(9): 2123-2134.)

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

[10] Wang H D, Yao X. Corner sort for Pareto-based many-objective optimization[J]. IEEE Transactions on Cybernetics, 2014, 44(1): 92-102.

[11] Wang J H, Zhang W W, Zhang J. Cooperative differential evolution with multiple populations for multiobjective

optimization[J]. IEEE Transactions on Cybernetics, 2016, 46(12): 2848-2861.

[12] Jamali A, Mallipeddi R, Salehpour M, et al. Multi-objective differential evolution algorithm with fuzzy inference-based adaptive mutation factor for Pareto optimum design of suspension system[J]. Swarm and Evolutionary Computation, 2020, 54: 100666.

[13] Bandyopadhyay S, Mukherjee A. An algorithm for many-objective optimization with reduced objective computations: A study in differential evolution[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(3): 400-413.

[14] Wang X P, Dong Z M, Tang L X. Multiobjective differential evolution with personal archive and biased self-adaptive mutation selection[J]. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 2020, 50(12): 5338-5350.

[15] Fan Q Q, Wang W L, Yan X F. Multi-objective differential evolution with performance-metric-based self-adaptive mutation operator for chemical and biochemical dynamic optimization problems[J]. Applied Soft Computing, 2017, 59(1): 33-44.

[16] 丁进良, 陈佳鑫, 马欣然. 基于自适应差分进化的常压塔轻质油产量多目标优化[J]. 控制与决策, 2020, 35(3): 604-612.
(Ding J L, Chen J X, Ma X R. Multi-objective optimization of light oil production in atmospheric distillation column based on self-adaptive differential evolution[J]. Control and Decision, 2020, 35(3): 604-612.)

[17] Tang L X, Wang X P, Dong Z M. Adaptive multiobjective differential evolution with reference axis vicinity mechanism[J]. IEEE Transactions on Cybernetics, 2019, 49(9): 3571-3585.

[18] Qiao J F, Hou Y, Han H G. Optimal control for wastewater treatment process based on an adaptive multi-objective differential evolution algorithm[J]. Neural Computing and Applications, 2019, 31(7): 2537-2550.

[19] Li K, Deb K, Zhang Q F, et al. Efficient nondomination level update method for steady-state evolutionary multiobjective optimization[J]. IEEE Transactions on Cybernetics, 2017, 47(9): 2838-2849.

[20] Trivedi V, Ramteke M. Using following heroes operation in multi-objective differential evolution for fast convergence[J]. Applied Soft Computing, 2021, 104: 107225.

[21] 刘明凯, 王占山, 邢彦丽. 基于强化多目标差分进化算法的电-气互联系统最优潮流计算[J]. 电工技术学报, 2021, 36(11): 2220-2232.
(Liu M K, Wang Z S, Xing Y L. Enhanced multi-objective differential evolutionary algorithm based optimal power flow calculation for integrated electricity and gas systems[J]. Transactions of China Electrotechnical Society, 2021, 36(11): 2220-2232.)

[22] Bechikh S, Chaabani A, Ben Said L. An efficient chemical reaction optimization algorithm for multiobjective optimization[J]. IEEE Transactions on Cybernetics, 2015, 45(10): 2051-2064.

[23] Bao C T, Xu L H, Goodman E D, et al. A novel non-dominated sorting algorithm for evolutionary multi-objective optimization[J]. Journal of Computational Science, 2017, 23(1): 31-43.

[24] Zhang X Y, Tian Y, Cheng R, et al. An efficient approach to nondominated sorting for evolutionary multiobjective optimization[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(2): 201-213.

[25] 余伟伟, 谢承旺, 闭应洲, 等. 一种基于自适应模糊支配的高维多目标粒子群算法[J]. 自动化学报, 2018, 44(12): 2278-2289.
(Yu W W, Xie C W, Bi Y Z, et al. Many-objective particle swarm optimization based on adaptive fuzzy dominance[J]. Acta Automatica Sinica, 2018, 44(12): 2278-2289.)

[26] Zhou Y R, Chen Z F, Zhang J. Ranking vectors by means of the dominance degree matrix[J]. IEEE Transactions on Evolutionary Computation, 2017, 21(1): 34-51.

[27] 王凌, 潘子肖. 基于深度强化学习与迭代贪婪的流水车间调度优化[J]. 控制与决策, 2021, 36(11): 2609-2617.
(Wang L, Pan Z X. Scheduling optimization for flow-shop based on deep reinforcement learning and iterative greedy method[J]. Control and Decision, 2021, 36(11): 2609-2617.)

[28] 马永杰, 陈敏, 龚影, 等. 动态多目标优化进化算法研究进展[J]. 自动化学报, 2020, 46(11): 2302-2318.
(Ma Y J, Chen M, Gong Y, et al. Research progress of dynamic multi-objective optimization evolutionary algorithm[J]. Acta Automatica Sinica, 2020, 46(11): 2302-2318.)

[29] Zhang X Y, Tian Y, Cheng R, et al. A decision variable clustering-based evolutionary algorithm for large-scale many-objective optimization[J]. IEEE Transactions on Evolutionary Computation, 2018, 22(1): 97-112.

[30] Roy P C, Islam M M, Deb K. Best order sort: A new algorithm to non-dominated sorting for evolutionary multi-objective optimization[C]. Proceedings of the 2016 on Genetic and Evolutionary Computation Conference Companion. Denver, 2016: 1113-1120.

[31] Tian Y, Cheng R, Zhang X Y, et al. PlatEMO: A MATLAB platform for evolutionary multi-objective optimization[educational forum[J]. IEEE Computational Intelligence Magazine, 2017, 12(4): 73-87.

作者简介

侯莹 (1982−), 女, 助理研究员, 博士, 从事多目标智能优化方法、协同优化的研究, E-mail: houying17@sina.com;

吴毅琳 (1998−), 女, 硕士生, 从事差分进化算法、智能优化的研究, E-mail: wuyl@emails.bjut.edu.cn;

白星 (1996−), 女, 博士生, 从事粒子群优化算法, 多任务优化的研究, E-mail: 838797151@qq.com;

韩红桂 (1983−), 男, 教授, 博士生导师, 从事复杂系统智能优化控制、神经网络控制器设计等研究, E-mail: rechardhan@bjut.edu.cn.