一种用于多目标组合优化的三阶段混合蛙跳框架

申晓宁[1,2,3†], 陈庆洲[1], 潘红丽[1], 游 璇[1], 黄 遥[1]

(1. 南京信息工程大学 自动化学院，南京 210044；2. 南京信息工程大学 江苏省大气环境与装备技术协同创新中心，南京 210044；3. 南京信息工程大学 江苏省大数据分析技术重点实验室，南京 210044)

摘 要: 提出一种用于求解多目标组合优化问题的3阶段混合蛙跳框架. 该框架采用阶段化、模块化的设计思想,将种群的进化过程分为快速收敛、探索扩展、极值挖掘3个阶段. 在快速收敛阶段,迅速定位Pareto前沿,使整个群体快速地聚集在前沿附近;在探索扩展阶段,进一步提升解的精度并让种群均匀地分布在前沿上;在极值挖掘阶段,搜寻各目标上的边界极值,增强分布性能. 对于不同阶段的不同模块,采用不同的策略以提升框架的求解性能. 所提出框架对于多目标组合优化问题具有良好的通用性,在解决不同类型的问题时仅需设计相应的编码方式、个体生成算子和约束处理机制. 采用经典的多目标背包问题作为测试问题,与五种已有算法进行对比,结果表明,所提出框架具有良好性能,基于该框架设计的混合蛙跳算法具有更好的收敛性和分布性.

关键词: 多目标组合优化；混合蛙跳框架；快速收敛；探索扩展；极值挖掘；离散跳跃规则

中图分类号: TP301.6 **文献标志码:** A

DOI: 10.13195/j.kzyjc.2020.1616 **开放科学(资源服务)标识码(OSID):**

引用格式: 申晓宁,陈庆洲,潘红丽,等. 一种用于多目标组合优化的三阶段混合蛙跳框架[J]. 控制与决策, 2022, 37(4): 973-981.

A three phases shuffled frog leaping framework for multi objective combinatorial optimization

SHEN Xiao-ning[1,2,3†], *CHEN Qing-zhou*[1], *PAN Hong-li*[1], *YOU Xuan*[1], *HUANG Yao*[1]

(1. School of Automation，Nanjing University of Information Science and Technology，Nanjing 210044，China；2. Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology，Nanjing University of Information Science and Technology，Nanjing 210044，China；3. Jiangsu Key Laboratory of Big Data Analysis Technology，Nanjing University of Information Science and Technology，Nanjing 210044，China)

Abstract: A three phases shuffled frog leaping framework for multi objective combinatorial optimization is proposed. The framework adopts the idea of phasing and modularization, and divides the evolutionary process of population into three phases: rapid convergence, exploration and expansion, exploit extremum. In the rapid convergence phase, the Pareto front is quickly positioned so that the whole population quickly gathers near the front. In the exploration and expansion phase, the accuracy of the solution is further improved and the population is evenly distributed on the front. In the exploit extremum phase, the boundary extremum on each objective is searched to enhance the distribution performance. For different modules in different stages, different strategies are adopted to improve the solution performance of the framework. The proposed framework has good generality for multi-objective combinatorial optimization problems. When solving different types of problems, only the corresponding coding mode, individual generating operator and constraint processing mechanism need to be designed. The classical multi-objective knapsack problem is used as the test problem, and compared with five existing algorithms, the results show that the proposed framework has good performance, and the hybrid frog hop algorithm based on the framework has better convergence and distribution.

Keywords: multi objective combinatorial optimization；shuffled frog leaping framework；rapid convergence；exploration and expansion；exploit extremum；discrete leaping rule

0 引 言

目前,求解组合优化问题的算法种类繁多,已有大量的多目标智能优化算法应用在移动群智感知[1]、车间调度[2-3]、锚杆支护网[4]等组合优化问题中. 其中,大多数算法是根据某个具体问题的特点设计改进策略,进而提高算法的收敛性和多样性. 例如,文献

收稿日期: 2020-11-22；**修回日期:** 2021-02-10.

基金项目: 国家自然科学基金项目(61502239)；江苏省自然科学基金项目(BK20150924).

责任编委: 巩敦卫.

†通讯作者. E-mail: sxnystsyt@sina.com.

[5]基于人工鱼群算法求解冷链物流问题,采用由车辆等待时间改变客户服务时间的右移策略,提高了算法的收敛速度;文献[6]在求解带时间窗的车辆路径问题的混合多目标进化算法中,设计了一种基于启发信息简单插入搜索算子,尽可能地减少车辆数量,提高算法的收敛性能. 然而,该类算法只能解决特定的问题,缺乏一定的通用性. 当问题的类型发生变化时,算法便不再适用. 为了提高算法的通用性,一些学者着手研究多目标优化框架,对算法全局收敛性的提高和多样性的维护提出了统一机制. 田红军等[7]将基于种群的全局搜索与局部搜索相结合,设计了启发式的多目标进化算法混合框架; Zille等[8]基于问题变换,为大规模多目标优化问题建立一种适用于单种群进化算法的加权优化框架; Tian等[9]提出了一种协同进化框架用于求解多目标约束优化问题. 然而,上述文献均针对以遗传算法为代表的进化算法设计框架,且大多用于求解连续函数优化问题. 群智能算法是智能优化算法中除进化算法以外的另一类主流方法. 它模拟大自然生物的群体协作、信息交流和涌现现象,实现智能搜索行为. 目前,尚缺乏群智能算法在求解多目标组合优化问题中的框架设计和相关研究.

混合蛙跳算法是一种群智能算法[10],具有概念简单、参数较少、全局寻优能力强等优势. 它通过分组、局部搜索、全局混洗等过程产生较优后代,实现种群之间的信息交流[11]. 已有多目标混合蛙跳算法的研究大多针对青蛙个体之间的信息交互、局部搜索等提出改进措施. Fang等[12]使用差分进化算法的随机搜索技术代替蛙跳算法的局部搜索,改善蛙跳算法易早熟收敛的问题; Yang等[13]设计了领导青蛙、跟随青蛙和变异青蛙3个子种群、启发式青蛙激活机制和基于正态云模型的青蛙变异策略,求解水电站短期经济运行问题. 然而,它们未能充分考虑到多目标优化对收敛性提高和多样性维护的要求,产生的非支配解集往往不能收敛到问题真实的Pareto前沿或生成的解聚集在某一块或多块区域内,分布不均匀且宽广度不足. 此外,已有蛙跳算法针对问题的特定信息设计策略,此类方法能在某种问题上提升算法的性能,但当问题类型改变时,先前设计的策略可能会失效,针对新问题需要重新设计相应的收敛性或多样性维护机制,费时费力. 因此,亟需研究一种通用且模块化的混合蛙跳框架,以高效地求解各类多目标组合优化问题.

为增强算法的通用性,本文依据对多目标搜索空间的寻优机理,提出一种用于多目标组合优化的三阶段混合蛙跳框架. 首先,快速收敛阶段使群体向真实Pareto前沿快速靠拢;其次,探索扩展阶段在Pareto前沿附近进一步开发新的非支配解;最后,极值挖掘阶段深度探寻Pareto前沿的边界极值. 以多目标0/1背包问题[14]作为所提出框架的应用实例,与5种具有代表性的多目标智能优化算法的对比结果表明,混合蛙跳算法具有更好的收敛性和多样性,从而验证了所提出三阶段混合蛙跳框架是可行而有效的,它能够提高多目标组合优化问题的求解精度、增强Pareto非支配解集分布的均匀性和宽广度.

1 三阶段混合蛙跳框架

针对多目标组合优化问题,提出一种三阶段混合蛙跳框架(TP-SFLAF). 基于所提出框架给出了多目标背包问题的三阶段混合蛙跳算法.

1.1 TP-SFLAF的整体流程

通过对智能优化算法在多目标搜索空间中寻优机理的深入分析,将种群在目标空间中的搜索过程划分为如图1所示的3个阶段(以二目标空间为例),此时,可以提高多目标智能优化算法的求解性能. 在进化前期,种群应具备良好的全局收敛性,以尽快向Pareto前沿靠拢;在进化中期,应强调种群的延展性,在Pareto前沿上探索到更多且分布更均匀的解;在进化末期,种群应具有突出的挖掘能力,探寻到Pareto前沿的极端解.

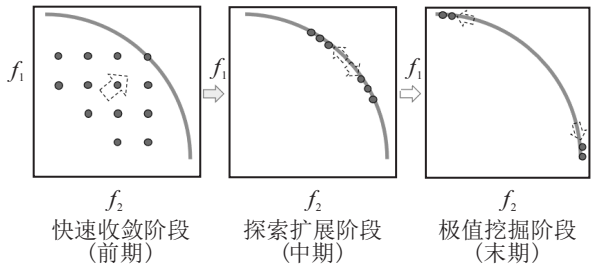

图1 多目标智能优化算法不同搜索阶段解析

基于上述分析,结合混合蛙跳算法的特征,设计了一种三阶段混合蛙跳框架TP-SFLAF,流程如图2所示. 所提出框架在群体初始化之后主要分为3个阶段: 1)快速收敛阶段; 2)探索扩展阶段; 3)极值挖掘阶段. 每个阶段包含3个主要模块,分别是分组排序、局部搜索、全局混洗. 值得注意的是,这3个模块在不同阶段分别具有不同的特征和实现方法. 3个阶段的具体实现方法分别由1.1.1～1.1.3节给出. 所提出框架适用于求解各类多目标组合优化问题,针对具体问题仅需设计并替换相应的个体编码方式、局部搜索策略(即个体更新和约束处理)以及各阶段的终止条件. 需要说明的是,所提出框架目前使用的是Pareto

支配关系,该关系在目标个数大于3的超多目标组合优化问题中易导致维数灾难,使得进化选择退化为随机选择而失去优胜劣汰的作用,所以对于该类问题,需在所提出框架中引入新的超多目标处理机制.

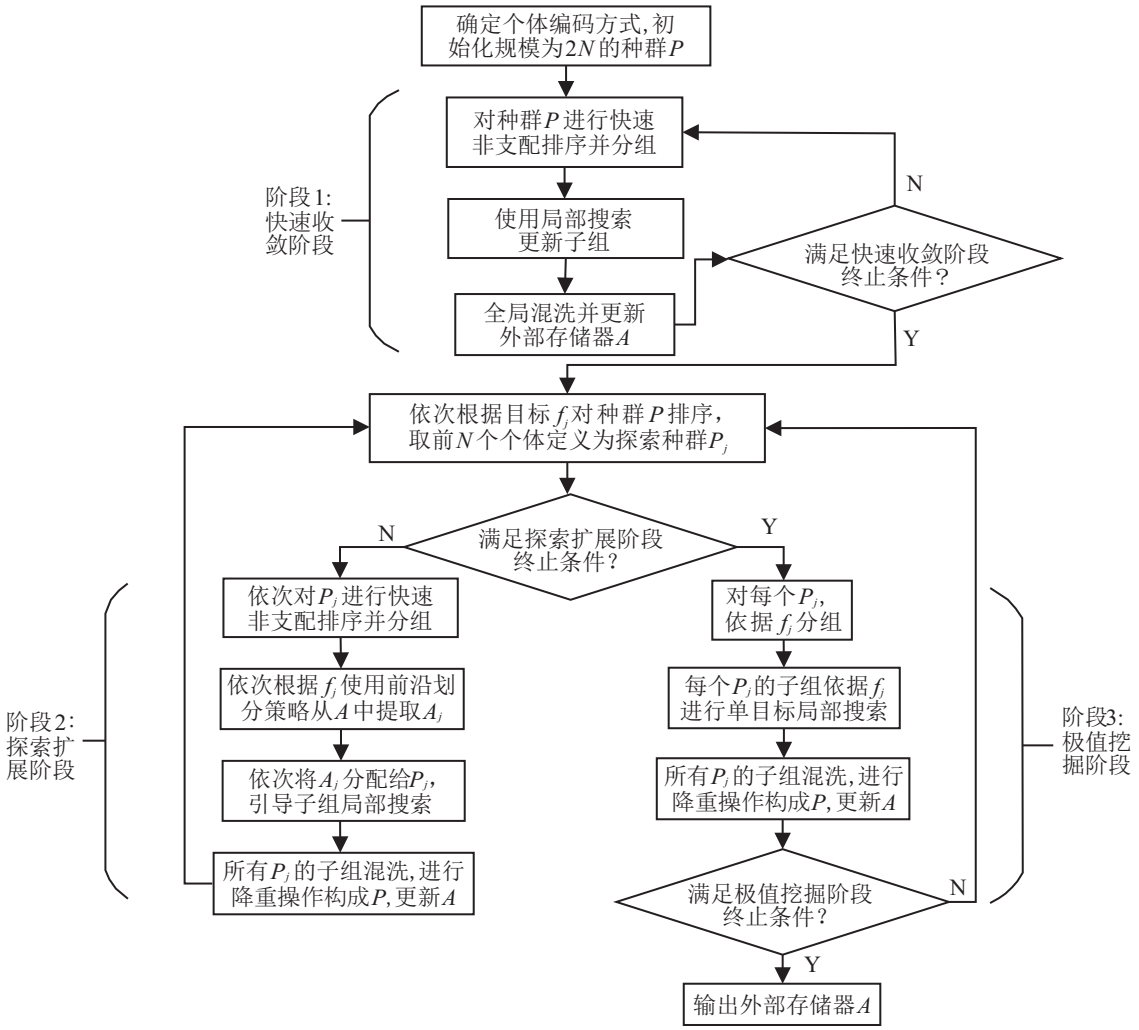

图 2　TP-SFLAF框架的流程

1.1.1　快速收敛阶段

多目标智能优化算法的初始化阶段是随机生成均匀分布在决策空间中的初始群体,而多目标组合优化问题的真实Pareto最优解集往往只占据决策空间的一小部分. 为了避免算法在搜索空间中盲目探索,从而造成计算资源的浪费,所提出框架TP-SFLAF在快速收敛阶段侧重于发挥混合蛙跳算法全局寻优能力强的特点,以快速定位到Pareto最优前沿所处区域. 同时,在该阶段只使用一个外部存储器保留群体中的非支配解,不过多考虑对种群多样性的维护,从而将前期的大部分计算资源聚焦于算法收敛性能的提高,使整个群体尽快地收敛到Pareto最优前沿附近.

快速收敛阶段的实现方法如下: 首先,基于非支配关系和排挤距离对种群进行快速非支配排序并分组. 基本混合蛙跳算法按排名顺序依次分组,该方法存在均匀性不足、个体信息不能充分交互的缺点. 为提高分组的均匀性,本节采用S型的分组方式. 将规模为$2N$且排序后的种群P分成w个子组,每组$2b$个个体,满足$2N = w \times 2b$. 将第$1 \sim$第w个个体依次放入第$1 \sim$第w组,第$w+1 \sim$第$2w$个个体倒序放入第w组$\sim$第1组,第$2w+1 \sim$第$3w$个个体依次放入第$1 \sim$第w组,以此类推. 其后,使用局部搜索更新各个子组,各子组的全局最优解从外部存储器A中随机选取,而局部最优解和局部最劣解则根据快速非支配排序号确定. 最后,将所有子组混洗并基于非支配概念及排挤距离更新外部存储器A. 当A中解的个数超过其最大容量时,将排挤距离较小的解从A中移除.

1.1.2　探索扩展阶段

快速收敛阶段使种群收敛到真实Pareto前沿附近. 此时,整个种群密集地聚集在前沿的某一块或若干区域,分布性较差. 为了扩展种群的分布范围并使其均匀地散布在前沿上,TP-SFLAF的探索扩展阶段

侧重于多样性的维护. 采用并行的思想,根据m个不同的优化目标,产生m个探索子种群和m个引导解集,各子种群跟踪相应的引导解集,分工搜索不同的Pareto前沿区域. 该阶段有助于算法向不同方向并行搜索,使得非支配解集由聚集状态转向探索Pareto前沿的不同区域,从而提升Pareto前沿的搜索精度和分布性能.

探索扩展阶段的实现方法如下: 首先,依次根据目标$f_j(j = 1,2,\ldots,m)$对阶段1求得的规模为$2N$的种群P排序,取前N个个体定义为第j个探索子种群P_j. 其次,依次对得到的m个探索子种群P_j进行快速非支配排序,并分别使用S型方式分组. 再次,采用前沿划分策略将外部存储器A(即当前搜索到的Pareto前沿)划分为m个引导集A_j,并使用A_j指导P_j的子组进行局部搜索. 具体而言,P_j各子组跟踪的全局最优解分别从A_j中随机选取,而各子组的局部最优解和局部最劣解则是依据快速非支配排序号确定. 前沿划分策略的伪代码见算法1,依次根据目标f_j对外部存储器A排序,选择前$\alpha\%$个在f_j上表现最好的解作为引导集A_j. 最后,将所有P_j的子组混洗,并对目标向量重复的解进行降重操作: 对某一解,将背包中已有的物品根据利润与重量的比值排序,从背包拿出一个比值最差的物品以构造出新解. 经降重后的混洗群体构成新种群P,并更新A.

算法1 前沿划分策略.

输入: A(外部存储器), $\alpha\%$(分割百分比), m(目标数);

输出: $A_j, j = 1,2,\ldots,m$(划分成m个引导集)

step 1: $\mathrm{num} = \mathrm{ceil}(\alpha\% \times \mathrm{size}(A))$;

step 2: for j = 1 to m do

step 3: $TA_j \leftarrow \mathrm{sort}(A, f_j, \downarrow)$; //假设为最大化问题

step 4: 选择TA_j前num个个体放入A_j;

step 5: end for

1.1.3 极值挖掘阶段

通过前两个阶段的执行,TP-SFLAF获得了收敛性与分布性都较好的Pareto前沿,但是对于前沿的边界极值搜索困难. 为了获取前沿的各边界点,极值挖掘阶段基于分解的思想,将群体的搜索方向依据目标个数分成m个,采用单目标优化的方式,在满足约束条件的前提下,搜索各单个目标的极值. 极值挖掘阶段进一步增强了TP-SFLAF的扩展性能,并大大提高了它对边界极值的探索能力.

极值挖掘阶段的实现方法如下: 首先,依次根据目标f_j对阶段2产生的种群P排序,取前N个个体定义为探索种群P_j; 其次,为每个P_j分配目标f_j作为搜索方向,将个体在单目标f_j上的排序值作为分组依据,使用S型方式将P_j分组; 再次,每个P_j的各子组分别依据f_j进行单目标局部搜索; 最后,混合所有P_j的子组,对每个目标向量重复解进行降重操作,生成新的种群P并更新外部存储器A.

1.2 基于多目标背包问题的改进策略

基于1.1节的框架TP-SFLAF设计适用于多目标背包问题的三阶段混合蛙跳算法TP-SFLA-KP. 针对多目标背包问题的特点,提出融合离散跳跃规则和贪婪生成策略的个体生成算子; 利用离散跳跃规则让局部最劣个体X_w跟踪局部最优或全局最优个体; 若产生的新解均差于X_w,则使用贪婪生成策略获取一个启发解替代X_w. 此外,设计了放松约束修复策略.

1.2.1 离散跳跃规则

基本蛙跳算法的局部搜索跳跃规则针对连续优化问题设计,将其直接应用于组合优化问题时会产生不可行解. 已有研究大多通过概率映射法将连续空间转换到离散空间[15],由于该方法未将搜索操作直接作用于离散个体,易导致个体有用信息发生丢失. 为解决该问题,同时增强个体之间信息的交互,本节针对多目标背包问题设计了对青蛙个体进行局部搜索的离散跳跃规则,以替代基本蛙跳算法中的更新公式. 采用二进制编码,编码串长度为每组的物品数量n,若第i个基因取值为1,则各组的第i个物品均放入相应背包,0则相反. 比较局部最劣个体和局部最优个体的编码串. 对于取值不同的基因位,让劣解跟踪优解,以便在决策空间中较好的方向上进行探索; 对于取值相同的基因位,使用基本位变异,以便挖掘个体附近更优秀的解. 离散跳跃规则通过在离散空间中直接对个体信息进行学习,提高了对个体优质信息的利用率. 离散跳跃规则的实现方法如算法2所示.

算法2 离散跳跃规则.

输入: X_b(局部最优个体), X_w(局部最劣个体), n(个体长度), $\beta\%$(信息接收百分比), p_m(变异概率);

输出: X_{wn}(新个体).

step 1: dif$\leftarrow\phi$, same$\leftarrow\phi$, index$\leftarrow\phi$, dis$\leftarrow$0, $X_{wn} \leftarrow X_{\mathrm{w}}$;

step 2: for i = 1 to n do

step 3: if $X_b(i) \neq X_w(i)$ then

step 4: dif $\leftarrow$ dif$\bigcup i$; //dif表示取值不同的基因位索引集合

step 5:　dis $\leftarrow$ dis + 1; //dis表示汉明距离

step 6:　else

step 7:　same $\leftarrow$ same$\bigcup i$; //same表示取值相同的基因位索引集合

step 8:　end if

step 9: end for

step 10: $n_1 =$ ceil$(\beta\% \times dis)$;

step 11: index $\leftarrow$ 从dif中随机选取n_1个基因位;

step 12: for j = 1 to n_1 do //劣解跟踪优解

step 13:　$X_{wn}(\text{index}(j)) \leftarrow X_b(\text{index}(j))$;

step 14 end for

step 15: for k = 1 to n-dis do

step 16:　if rand $< p_m$ then

step 17:　$X_{wn}(\text{same}(k)) \leftarrow \text{flip}\,(X_w(\text{same}(k)))$; //基本位变异

step 18:　end if

step 19: end for

1.2.2　贪婪生成策略

在基本混合蛙跳算法中,若局部最劣个体X_w在跟踪局部最优个体X_b或全局最优个体X_g后产生的新解仍然差于X_w,则在解空间内随机产生新解替换X_w. 该操作旨在增加算法对决策空间探索的多样性. 然而,该随机方式可能会以很大的概率生成比劣解质量更差的新解. 针对这一问题,本节引入多目标背包问题的启发信息,提出贪婪生成策略: 若某物品利润与重量的比值越大,则它被选中的概率越高. 所提出策略在降低随机性的同时也减少了混合蛙跳框架对无效决策空间的搜索,利用问题特定知识提高了生成解的质量,节省了计算资源.

贪婪生成策略的实现方法如下所述: 首先,为了增加解在有效决策空间内的生成范围,采用轮盘赌方法选择一个背包的启发信息进行利用; 其次,根据利润和质量的比值,使用sigmod函数计算各物品的选择概率,利润与质量之比越高表明该物品在此背包中被选取的可能性越大,提高了优解的产生几率; 最后,依据选择概率依次确定各物品是否放入背包,生成新解.

1.2.3　放松约束修复策略

传统的约束修复按利润重量比递减的顺序依次放入物品,以满足容量限制,同时尽可能增加整体利润. 然而,该方法过于贪婪,易使算法陷入早熟收敛. 针对该问题,本文提出一种放松约束修复策略. 计算各物品在各背包中的利润重量比矩阵,并由此依次计算第i个物品在各背包中的利润重量比均值. 对于违反约束的解,首先,对背包内的物品,将最小的一个物品移出各背包; 然后,对背包外的物品,在取值最大的k个物品中随机选取一个放入各背包(若背包外的物品数大于5,则令$k = 5$,否则,令k为背包外的物品数). 此时,若解依然违反约束,则继续循环使用删除和添加步骤,直至修复得到满足约束条件的新解. 所提出改进策略缓解了约束处理机制对问题启发信息的过度依赖,使群体保留更多的有效信息,增强了算法的探索能力.

2　实验仿真与结果分析

多目标背包问题(NOKP)[14]是一类典型的多目标组合优化问题. 给定m个背包和m组物品,每个背包对应一组物品,且每组物品的数量均为n,每组的每个物品都有唯一的利润和重量两种属性. 各组同一序号的物品同时放入或不放入相应背包. 例如,假设背包1的序号为1的物品放入背包1中,则背包j的序号为1的物品也要放入背包$j(j = 1, 2, \ldots, m)$中,同理,物品拿出也是如此. 在满足各背包容量约束的前提下,分别最大化各背包中物品的利润之和.

在多目标背包问题中验证所提出框架TP-SFLAF的性能,采用Matlab R2019b软件进行仿真实验,计算机处理器参数为AMD Ryzen 5 4600U with Radeon Graphics 2.10 GHz, 16 G运行内存. 本文使用8个二目标和三目标测试算例,其中算例2_100$(m = 2, n = 100)$取自文献[14],其余7个根据文献[14]的方法随机生成,每个背包对应物品的利润和质量均为区间[10, 100]内随机产生的整数. 设计两组实验: 1)验证个体生成算子的有效性; 2)将基于所提出框架设计的用于多目标背包问题的三阶段混合蛙跳算法TP-SFLA-KP分别与基本多目标蛙跳算法以及文献中5种具有代表性的多目标优化算法进行对比,验证所提出框架的有效性. 每组实验中各算法分别独立运行30次. 各组实验的测度均采用收敛性指标IGD(反世代距离)[16]和HV(超体积)[17],以及分布性能指标Spread[18]. IGD越小,HV越大,说明算法的收敛性越好; Spread越小,说明搜索到的解集分布越宽广、越均匀.

2.1　个体生成算子的有效性验证

2.1.1　离散跳跃规则的有效性验证

为检验1.2.1节设计的新型离散跳跃规则的有效性,将TP-SFLA-KP中离散跳跃规则替换成均匀交叉算子[19],将由此得到的算法与TP-SFLA-KP进行对比. 表1统计了两种策略在8个测试算例上的IGD、HV和Spread值. 为显著对比策略的优劣,引入显著

水平为0.05的Wilcoxon秩和检验[20]对8组实验结果进行统计测试,其中“+”表示离散跳跃规则显著优于均匀交叉算子,“=”表示两者无明显差异,“−”表示离散跳跃规则显著劣于均匀交叉算子. 由表1可以看出,在8个测试算例中,离散跳跃规则的IGD、HV和Spread值分别在8个、7个和8个上显著优于均匀交叉算子. 组合优化问题随着规模的增大,搜索空间成倍地扩大,可行解呈指数增长,搜索难度增加,算法的收敛速度和求解精度随之降低. 均匀交叉算子随机性较高,随着搜索空间规模的增大,算法的搜索较为盲目,效率低下. 相比之下,在离散跳跃规则中,劣解接收优解的部分有效信息,沿着优解的方向移动,同时在移动过程中对解的周围区域进行挖掘,从而实现对搜索空间的高效探索,提高了算法的收敛速度.

表1 离散跳跃规则和均匀交叉算子的结果对比

测试算例	IGD mean(std)		HV mean(std)		Spread mean(std)	
	离散跳跃规则	均匀交叉	离散跳跃规则	均匀交叉	离散跳跃规则	均匀交叉
2-75	**1.10e-2(3.10e-3)**	6.60e-2(1.56e-2)+	**8.23e-1(1.10e-2)**	7.81e-1(1.59e-2)+	**6.35e-1(5.16e-2)**	7.42e-1(5.72e-2)+
2-100	**1.35e-2(1.78e-3)**	6.98e-2(1.66e-2)+	**8.28e-1(1.44e-2)**	7.80e-1(2.02e-2)+	**5.78e-1(6.15e-2)**	7.74e-1(7.12e-2)+
2-125	**2.17e-2(3.92e-3)**	9.13e-2(1.73e-2)+	**8.37e-1(1.32e-2)**	7.71e-1(1.62e-2)+	**5.75e-1(3.99e-2)**	8.11e-1(4.52e-2)+
2-150	**1.20e-2(1.49e-3)**	7.60e-2(1.22e-2)+	**8.05e-1**(1.27e-2)	7.99e-1(**1.04e-2**)=	**5.90e-1**(4.74e-2)	8.05e-1(**3.39e-2**)+
2-175	**1.89e-2(3.17e-3)**	1.51e-1(1.75e-2)+	**8.23e-1(9.74e-3)**	7.27e-1(1.02e-2)+	**5.70e-1**(6.27e-2)	8.75e-1(**3.56e-2**)+
2-200	**1.69e-2(3.09e-3)**	1.85e-1(2.98e-2)+	**8.32e-1(1.19e-2)**	7.44e-1(1.56e-2)+	**5.93e-1**(5.05e-2)	8.90e-1(**2.99e-2**)+
3-100	**5.49e-2(3.61e-3)**	1.42e-1(1.00e-2)+	**8.53e-1(4.66e-2)**	6.62e-1(5.40e-2)+	**4.32e-1(2.54e-2)**	5.69e-1(2.93e-2)+
3-200	**5.92e-2(4.64e-3)**	1.89e-1(1.45e-2)+	**8.01e-1(4.29e-2)**	6.33e-1(4.30e-2)+	**4.49e-1(2.16e-2)**	6.56e-1(4.94e-2)+
总计+/=/−		8/0/0		7/1/0		8/0/0

2.1.2 贪婪生成策略有效性验证

为检验1.2.2节设计的贪婪生成策略的有效性,将TP-SFLA-KP中贪婪生成策略替换成随机生成算子,将由此得到的算法与TP-SFLA-KP进行对比. 表2统计了两种策略在8个测试算例上的IGD、HV和Spread值.

表2 贪婪生成策略和随机生成算子的结果对比

测试算例	IGD mean(std)		HV mean(std)		Spread mean(std)	
	贪婪生成策略	随机生成算子	贪婪生成策略	随机生成算子	贪婪生成策略	随机生成算子
2-75	**1.10e-2**(3.10e-3)	1.13e-2(**2.82e-3**)+	**8.23e-1(1.10e-2)**	8.22e-1(1.29e-2)=	**6.35e-1(5.16e-2)**	6.51e-1(5.51e-2)+
2-100	**1.35e-2**(1.78e-3)	1.38e-2(**1.70e-3**)+	**8.28e-1**(1.44e-2)	8.26e-1(**1.41e-2**)=	5.78e-1(6.15e-2)	**5.53e-1(5.73e-2)**−
2-125	**2.17e-2**(3.92e-3)	2.28e-2(**3.72e-3**)+	**8.37e-1**(1.32e-2)	8.36e-1(**1.02e-2**)=	**5.75e-1(3.99e-2)**	**5.75e-1**(5.12e-2)=
2-150	**1.20e-2(1.49e-3)**	1.27e-2(1.50e-3)+	**8.05e-1(1.27e-2)**	8.00e-1(1.29e-2)+	5.90e-1(**4.74e-2**)	**5.88e-1**(5.20e-2)=
2-175	**1.89e-2(3.17e-3)**	1.93e-2(2.79e-3)+	**8.23e-1(9.74e-3)**	8.22e-1(1.36e-2)=	5.70e-1(**6.27e-2**)	**5.58e-1**(6.94e-2)−
2-200	**1.69e-2(3.09e-3)**	1.81e-2(4.28e-3)+	8.32e-1(**1.19e-2**)	**8.33e-1**(1.22e-2)=	5.93e-1(5.05e-2)	**5.92e-1(4.76e-2)**=
3-100	**5.49e-2**(3.61e-3)	5.68e-2(**3.23e-3**)+	8.53e-1(4.66e-2)	**8.64e-1(1.58e-2)**−	4.32e-1(2.54e-2)	**4.16e-1(1.09e-2)**−
3-200	**5.92e-2**(4.64e-3)	6.03e-2(**2.73e-3**)+	**8.01e-1**(4.29e-2)	**7.99e-1**(2.13e-2)=	**4.49e-1**(2.16e-2)	4.62e-1(**1.58e-2**)+
总计+/=/−		8/0/0		1/6/1		2/3/3

由表2可知,贪婪生成策略的IGD值在8个测试算例中显著优于随机生成算子,说明贪婪生成策略提高了算法的收敛性能. 究其原因,它引入问题的启发信息减少了劣解的生成概率,提高了生成解的质量. 贪婪生成策略的Spread值在3个算例上劣于随机生成算子,原因是随机生成的解能够更加均匀地分布在搜索空间中,而贪婪生成策略则易于使生成的解集中在精英个体附近,导致解集的分布性能下降. 此外,与随机生成算子相比,贪婪生成策略的HV值在1个算例上显著优于,在1个算例中显著劣于,而在其余6个算例中两者无显著差异,原因是HV值同时取决于算法的收敛性和分布性,贪婪生成策略在收敛性能上优于随机生成算子,而在分布性能上却劣于随机生成算子.

2.2 与其他算法的对比

为验证TP-SFLAF框架的有效性,将基于该框架设计的适用于多目标背包问题的TP-SFLA-KP算法,与基本多目标蛙跳算法(MOSFLA)、以及5种具有代

表性的多目标智能优化算法(NSGA-II[21]、SPEA2[22]、MOEA/D[23]、Two-Arch2[24]、MOFPA[25])进行对比. 对比算法均使用二进制编码. 其中,MOSFLA仅利用TP-SFLAF框架的快速收敛阶段进行多目标搜索,并采用与TP-SFLA-KP相同的个体生成算子和约束处理机制;NSGA-II、SPEA2使用均匀交叉[19]和基本位变异算子;MOEA/D、Two-Arch2使用单点交叉和基本位变异算子. 由于除2_100之外的7组测试算例均为随机生成,Pareto真实前沿未知,将上述7种算法分别运行30次产生的所有非支配解集合并,从中确定

表3　TP-SFLA-KP算法与其他算法的对比结果

算例	2_75 mean(std)			2_100 mean(std)		
	IGD	HV	Spread	IGD	HV	Spread
TP-SFLA-KP	**1.10e-2(3.10e-3)**	8.23e-1(**1.10e-2**)	6.35e-1(5.16e-2)	**1.35e-2(1.78e-3)**	8.28e-1(1.44e-2)	5.78e-1(6.15e-2)
MOSFLA	2.43e-2(7.83e-3)+	8.08e-1(1.74e-2)+	6.85e-1(5.14e-2)+	1.93e-2(3.96e-3)+	8.11e-1(1.61e-2)+	6.14e-1(5.53e-2)+
NSGA-II	9.89e-2(2.39e-2)+	7.91e-1(1.32e-2)+	8.43e-1(5.65e-2)+	8.43e-2(2.08e-2)+	7.80e-1(1.69e-2)+	8.37e-1(**3.24e-2**)+
SPEA2	3.45e-2(1.06e-2)+	8.03e-1(1.60e-2)+	7.19e-1(7.80e-2)+	5.12e-2(9.24e-3)+	7.95e-1(1.46e-2)+	7.58e-1(5.92e-2)+
MOEA/D	3.18e-2(4.65e-3)+	**8.30e-1**(1.72e-2)−	**6.04e-1**(7.09e-2)−	3.11e-2(4.24e-3)+	**8.34e-1**(1.81e-2)−	**5.62e-1**(6.28e-2)−
Two-Arch2	1.55e-1(1.62e-2)+	7.66e-1(1.11e-2)+	9.29e-1(**3.43e-2**)+	1.55e-1(1.68e-2)+	7.43e-1(**1.17e-2**)+	9.41e-1(3.34e-2)+
MOFPA	7.67e-2(1.45e-2)+	8.03e-1(1.19e-2)+	8.23e-1(3.66e-2)+	9.24e-2(1.89e-2)+	7.73e-1(1.52e-2)+	8.32e-1(4.11e-2)+
总计+/=/−	6/0/0	5/0/1	5/0/1	6/0/0	5/0/1	5/0/1

算例	2_125 mean(std)			2_150 mean(std)		
	IGD	HV	Spread	IGD	HV	Spread
TP-SFLA-KP	**2.17e-2**(3.92e-3)	**8.37e-1**(1.32e-2)	**5.75e-1**(3.99e-2)	**1.20e-2(1.49e-3)**	**8.05e-1**(1.27e-2)	**5.90e-1**(4.74e-2)
MOSFLA	3.22e-2(**3.85e-3**)+	8.17e-1(1.27e-2)+	6.66e-1(5.33e-2)+	2.37e-2(4.83e-3)+	8.00e-1(1.26e-2)+	6.67e-1(5.05e-2)+
NSGA-II	7.75e-2(1.08e-2)+	7.74e-1(1.17e-2)+	8.10e-1(3.50e-2)+	1.20e-1(1.83e-2)+	7.64e-1(9.32e-3)+	8.84e-1(2.40e-2)+
SPEA2	6.25e-2(7.54e-3)+	7.81e-1(1.15e-2)+	8.01e-1(4.23e-2)+	5.38e-2(6.94e-3)+	7.83e-1(1.10e-2)+	7.94e-1(4.45e-2)+
MOEA/D	4.70e-2(5.23e-3)+	8.19e-1(1.57e-2)+	6.03e-1(5.78e-2)+	2.93e-2(3.73e-3)+	8.00e-1(1.35e-2)+	6.50e-1(5.92e-2)+
Two-Arch2	1.91e-1(1.27e-2)+	7.25e-1(1.25e-2)+	9.53e-1(2.90e-2)+	2.01e-1(1.18e-2)+	7.45e-1(1.01e-2)+	9.50e-1(**1.90e-2**)+
MOFPA	1.17e-1(1.42e-2)+	7.65e-1(**7.99e-3**)+	8.63e-1(**2.75e-2**)+	1.24e-1(1.94e-2)+	7.71e-1(**9.75e-3**)+	8.78e-1(2.71e-2)+
总计+/=/−	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0

算例	2_175 mean(std)			2_200 mean(std)		
	IGD	HV	Spread	IGD	HV	Spread
TP-SFLA-KP	**1.89e-2(3.17e-3)**	**8.23e-1(9.74e-3)**	**5.70e-1**(6.27e-2)	**1.69e-2(3.09e-3)**	**8.32e-1**(1.19e-2)	**5.93e-1**(5.05e-2)
MOSFLA	4.11e-2(6.85e-3)+	8.00e-1(1.42e-2)+	7.06e-1(4.01e-2)+	4.98e-2(9.81e-3)+	8.08e-1(1.36e-2)+	7.29e-1(3.77e-2)+
NSGA-II	1.40e-1(1.51e-2)+	7.50e-1(1.21e-2)+	8.93e-1(2.70e-2)+	1.52e-1(1.92e-2)+	7.60e-1(1.45e-2)+	8.95e-1(2.55e-2)+
SPEA2	7.53e-2(7.87e-3)+	7.76e-1(9.92e-3)+	8.22e-1(4.79e-2)+	1.03e-1(1.19e-2)+	7.69e-1(1.23e-2)+	8.51e-1(3.78e-2)+
MOEA/D	4.98e-2(3.96e-3)+	3.96e-3(7.97e-1)+	6.44e-1(6.24e-2)+	5.20e-2(5.54e-3)+	8.13e-1(1.19e-2)+	6.79e-1(5.08e-2)+
Two-Arch2	2.30e-1(1.36e-2)+	7.27e-1(1.13e-2)+	9.55e-1(**2.15e-2**)+	2.53e-1(1.41e-2)+	7.31e-1(**6.64e-3**)+	9.64e-1(**1.61e-2**)+
MOFPA	1.59e-1(1.44e-2)+	7.57e-1(9.75e-3)+	8.99e-1(3.00e-2)+	2.08e-1(1.78e-2)+	7.47e-1(7.63e-3)+	9.27e-1(1.68e-2)+
总计+/=/−	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0

算例	3_100 mean(std)			3_200 mean(std)		
	IGD	HV	Spread	IGD	HV	Spread
TP-SFLA-KP	**5.49e-2(3.61e-3)**	**8.53e-1**(4.66e-2)	**4.32e-1**(2.54e-2)	**5.92e-2(4.64e-3)**	**8.01e-1**(4.29e-2)	**4.49e-1**(2.16e-2)
MOSFLA	9.87e-2(6.14e-3)+	7.83e-1(4.51e-2)+	4.73e-1(**2.00e-2**)+	1.03e-1(7.02e-3)+	7.09e-1(4.62e-2)+	5.06e-1(**1.86e-2**)+
NSGA-II	1.52e-1(1.07e-2)+	6.52e-1(3.09e-2)+	5.70e-1(3.53e-2)+	1.51e-1(7.56e-3)+	6.21e-1(2.50e-2)+	6.19e-1(2.79e-2)+
SPEA2	1.26e-1(7.73e-3)+	7.11e-1(4.93e-2)+	5.36e-1(3.43e-2)+	1.30e-1(9.25e-3)+	6.69e-1(2.92e-2)+	5.53e-1(2.20e-2)+
MOEA/D	7.71e-2(5.73e-3)+	8.48e-1(4.07e-2)+	4.34e-1(2.36e-2)+	9.94e-2(6.68e-3)+	7.42e-1(6.04e-2)+	4.83e-1(2.41e-2)+
Two-Arch2	2.27e-1(1.16e-2)+	5.98e-1(**2.06e-2**)+	7.97e-1(2.99e-2)+	2.76e-1(9.66e-3)+	5.68e-1(**2.14e-2**)+	8.46e-1(2.54e-2)+
MOFPA	1.50e-1(1.40e-2)+	6.57e-1(2.70e-2)+	6.02e-1(2.69e-2)+	2.19e-1(1.25e-2)+	6.00e-1(2.23e-2)+	6.90e-1(2.28e-2)+
总计+/=/−	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0	6/0/0

出非支配解作为相应算例的近似Pareto真实前沿. 通过参数分析实验, TP-SFLA-KP中分割百分比α%和信息接受百分比β%的取值分别是50%和80%, 3个阶段的终止条件分别设定为目标评价次数达到20 000、60 000和100 000; 7种对比算法总的目标评价次数均取100 000, 种群规模N均取100, 其余参数设置均取自原文献, 实验结果如表3所示.

由表3可知, 在6个二目标和2个三目标且规模不同的测试算例中, 算法TP-SFLA-KP在IGD、HV、Spread测度上均显著优于MOSFLA、NSGA-II、Two-Arch2、SPEA2、MOFPA. 与MOEA/D相比, TP-SFLA-KP在8个算例中的IGD测度上均显著占优; 对于HV和Spread测度, 当二目标的问题规模小于等于100时, TP-SFLA-KP显著劣于MOEA/D(MOEA/D基于不同的权重向量在目标空间的各个方向进行搜索, 能够求得分布较广的非支配解集, 因此在小规模问题上的HV和Spread值较好), 而对于规模大于等于125的二目标问题以及规模为100和200的三目标问题, TP-SFLA-KP则显著优于MOEA/D(TP-SFLA-KP展现出更佳的快速收敛、探索扩展和极值挖掘能力). 上述结果表明, 三阶段混合蛙跳框架TP-SFLAF在求解多目标组合优化问题时是可行而有效的, 与具有代表性的多目标优化算法相比, 它在收敛性方面具有显著优势, 同时在中大规模问题中, 它也表现出了更加优异的分布性和扩展性. 以2_200测试问题为例, 图3中给出了7种算法搜索到的Pareto最优前沿. 由图3可见, TP-SFLA-KP在收敛精度、分布的均匀性和宽广度方面均具有最佳表现, 特别是它搜索到了两个优化目标上边界极值点附近的区域; 而MOSFLA、NSGA-II、SPEA2、Two-Arch2、MOFPA求得的非支配解均聚集在小范围内, 无法获取Pareto前沿上的所有区域, 说明它们的扩展性能有所欠缺; MOEA/D搜索到的解虽然分布相对宽广, 但依然未求得边界附近的解. 综上, 基于TP-SFLAF框架设计的TP-SFLA-KP算法对于不同规模的MOKP都有着较好的求解性能.

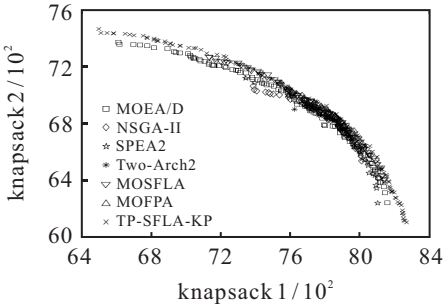

图3 在2_200测试问题上7种算法搜索到的Pareto前沿曲线对比

3 结 论

本文针对求解多目标组合优化问题的混合蛙跳算法通用性差的现状, 提出了一种模块化的三阶段混合蛙跳框架TP-SFLAF. 将算法在多目标空间的搜索过程划分为快速收敛、探索扩展、极值挖掘3个阶段. 针对各阶段的任务特点, 设计不同的种群寻优方式, 提高了框架的通用性与求解性能. TP-SFLAF框架在求解各类多目标组合优化问题时, 只需设计并替换适应问题的编码方式、个体更新和约束处理策略以及各阶段的终止条件即可. 以多目标背包问题作为TP-SFLAF框架的应用实例, 给出了融合离散跳跃规则和贪婪生成策略的个体生成算子及放松约束修复策略, 提出了TP-SFLA-KP算法. 在8个测试算例上将所提出算法与基本多目标蛙跳算法以及5种具有代表性的多目标智能优化算法进行对比, 验证所提出框架的有效性. 大量实验结果表明, 基于所提出框架设计的算法在收敛性和分布性方面的综合性能均优于对比算法, 它能在绝大多数不同规模的背包问题中搜索到收敛精度高、分布均匀且宽广的Pareto最优前沿.

参考文献(References)

[1] Ji J J, Guo Y N, Gong D W, et al. MOEA/D-based participant selection method for crowdsensing with social awareness[J]. Applied Soft Computing, 2020, 87: 105981.

[2] Han Y Y, Gong D W, Jin Y C, et al. Evolutionary multi-objective blocking lot-streaming flow shop scheduling with interval processing time[J]. Applied Soft Computing, 2016, 42: 229-245.

[3] Han Y Y, Li J Q, Sang H Y, et al. Discrete evolutionary multi-objective optimization for energy-efficient blocking flow shop scheduling with setup time[J]. Applied Soft Computing, 2020, 93: 106343.

[4] Guo Y N, Zhang X, Gong D W, et al. Novel interactive preference-based multiobjective evolutionary optimization for bolt supporting networks[J]. IEEE Transactions on Evolutionary Computation, 2020, 24(4): 750-764.

[5] Song M X, Li J Q, Han Y Q, et al. Metaheuristics for solving the vehicle routing problem with the time windows and energy consumption in cold chain logistics[J]. Applied Soft Computing, 2020, 95: 106561.

[6] Zhang W Q, Yang D J, Zhang G H, et al. Hybrid multiobjective evolutionary algorithm with fast sampling strategy-based global search and route sequence difference-based local search for VRPTW[J]. Expert Systems With Applications, 2020, 145: 113151.

[7] 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.)

[8] Zille H, Ishibuchi H, Mostaghim S, et al. A framework for large-scale multiobjective optimization based on problem transformation[J]. IEEE Transactions on Evolutionary Computation, 2018, 22(2): 260-275.

[9] Tian Y, Zhang T, Xiao J H, et al. A coevolutionary framework for constrained multiobjective optimization problems[J]. IEEE Transactions on Evolutionary Computation, 2021, 25(1): 102-116.

[10] Eusuff M M, Lansey K E. Optimization of water distribution network design using the shuffled frog leaping algorithm[J]. Journal of Water Resources Planning and Management, 2003, 129(3): 210-225.

[11] 申晓宁, 黄遥, 游璇, 等. 基于解空间反向跳跃和信息交互强化的新型混合蛙跳算法[J]. 控制与决策, 2021, 36(1): 105-114.
(Shen X N, Huang Y, You X, et al. A new shuffled frog leaping algorithm based on reverse leaping in solution space and information interaction enhancement[J]. Control and Decision, 2021, 36(1): 105-114.)

[12] Fang G H, Guo Y X, Wen X, et al. Multi-objective differential evolution-chaos shuffled frog leaping algorithm for water resources system optimization[J]. Water Resources Management, 2018, 32(12): 3835-3852.

[13] Yang Z, Yang K, Su L, et al. The shortꝉterm economical operation problem for hydropower station using chaotic normal cloud model based discrete shuffled frog leaping algorithm[J]. Water Resources Management, 2020, 34(3): 905-927.

[14] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach[J]. IEEE Transactions on Evolutionary Computation, 1999, 3(4): 257-271.

[15] Bhattacharjee K K, Sarmah S P. Shuffled frog leaping algorithm and its application to 0/1 knapsack problem[J]. Applied Soft Computing, 2014, 19: 252-263.

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

[17] van Veldhuizen D A, Lamont G B. Multiobjective evolutionary algorithm test suites[C]. Proceedings of the 1999 ACM symposium on Applied computing -SAC '99. San Antonio, 1999: 351-357.

[18] Wang Y N, Wu L H, Yuan X F. Multiꝉobjective selfꝉadaptive differential evolution with elitist archive and crowding entropyꝉbased diversity measure[J]. Soft Computing, 2010, 14(3): 193.

[19] Tanigaki Y, Narukawa K, Nojima Y, et al. Preference-based NSGA-II for many-objective knapsack problems[C]. 2014 Joint 7th International Conference on Soft Computing and Intelligent Systems (SCIS) and 15th International Symposium on Advanced Intelligent Systems (ISIS). Kitakyushu, 2014: 637-642.

[20] Wilcoxon F. Individual comparisons by ranking methods[J]. Biometrics Bulletin, 1945, 1(6): 80-83.

[21] 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.

[22] Zitzler E, Laumanns M, Thiele L. SPEA2: Improving the strength Pareto evolutionary algorithm[R]. Zurich: Swiss Federal Institute of Technology, 2001.

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

[24] Wang H D, Jiao L C, Yao X. Two_Arch2: An improved two-archive algorithm for many-objective optimization[J]. IEEE Transactions on Evolutionary Computation, 2015, 19(4): 524-541.

[25] Zouache D, Moussaoui A, Ben Abdelaziz F. A cooperative swarm intelligence algorithm for multi-objective discrete optimization with application to the knapsack problem[J]. European Journal of Operational Research, 2018, 264(1): 74-88.

作者简介

申晓宁 (1981−), 女, 教授, 博士, 从事计算智能、多目标优化等研究, E-mail: sxnystsyt@sina.com;

陈庆洲 (1996−), 男, 硕士生, 从事群智能计算、多目标优化的研究, E-mail: cqznljy@foxmail.com;

潘红丽 (1996−), 女, 硕士生, 从事群智能计算、调度技术的研究, E-mail: 1546498146@qq.com;

游璇 (1996−), 女，硕士生, 从事智能计算、多目标优化的研究, E-mail: 563752923@qq.com;

黄遥 (1993−), 男, 硕士, 从事群智能计算、调度技术的研究, E-mail: 873721417@qq.com.

(责任编辑：孙艺红)