文章编号：1006-3080(2020)06-0780-12 DOI：10.14135/j.cnki.1006-3080.20190917001

一种基于超体积指标的多目标进化算法

王学武， 魏建斌， 周 昕， 顾幸生
（华东理工大学化工过程先进控制和优化技术教育部重点实验室，上海 200237）

摘要：基于超体积指标的进化算法能够有效地解决多目标优化问题，可以获得收敛性和分布性均较好的解集，但计算复杂度高、程序运行效率低。针对二维和三维的多目标优化问题，提出了一种基于超体积指标的多目标进化算法(MOEA-HV)。利用精确计算种群中个体的独立贡献超体积来指导种群进化，在基于指标的进化算法(IBEA)前对所有种群个体进行非支配排序，提前删除被支配的个体，从而减少个体独立贡献超体积的计算量来提升运行效率，同时与NSGA-Ⅲ算法相结合来优化算法的分布性。实验结果表明，MOEA-HV算法的运行效率更高，且能够获得较好的收敛性和分布性。

关键词：多目标优化；超体积；非支配排序；IBEA
中图分类号：TP301 **文献标志码：**A

多目标优化问题通常是指具有多个目标同时具有多个最优解的一类问题，解决这类问题需要同时对相互作用、相互冲突的多个目标进行优化和综合考虑，因此多目标优化已经被广泛应用于电力调度[1]、网络优化[2]、化工过程[3]、结构设计[4]、数据挖掘[5]等领域。

以焊接机器人路径规划问题为例，衡量焊接路径好坏的指标往往不止一个，通常有路径长度、焊接用时、能量消耗、焊接变形量等，因此可将焊接机器人路径规划问题看作一个多目标优化问题。进化算法是一类模拟生物自然选择与自然进化的随机搜索算法[6]，通过种群迭代来协调局部与全局搜索能力，从而获得收敛性和分布性均较好的解集，能够有效地解决多目标优化问题。王学武等[7]通过将基于分解的多目标进化算法与自适应邻域相结合，以焊接路径长度和能量消耗为优化目标，较好地解决了弧焊机器人路径规划问题。Mac等[8]基于Pareto支配关系提出了改进的多目标粒子群优化算法，以焊接路径长度和路径平滑度为优化目标求解移动机器人路径规划问题，在不同种类的环境中均得到了较好的仿真结果。

多目标优化算法是解决多目标优化问题的核心，关键在于如何设计算法以获得较好的综合性能，而超体积(Hypervolume, HV)指标是唯一一个在Pareto支配关系上严格单调的度量指标[9]，超体积值越大，对应算法的综合性能越好，因此超体积可以作为归档策略、多样性机制或选择标准来指导算法搜索[10]，从而考虑将评价算法性能的超体积指标整合到算法中来解决多目标优化问题[11]。Bader等[9]提出的HypE算法通过蒙特卡洛模拟来近似精确的超体积，能够大大降低算法的时间复杂度、平衡估计的准确性和算法的计算复杂度。Igel等[12]提出的MO-CMA-ES算法将具有协方差矩阵自适应的精英进化策略与基于非支配排序的多目标选择相结合，实验结果表明基于超体积贡献的s-MO-CMA比基于拥挤距离的c-MO-CMA具有更好的性能。Beume等[13]提出的SMS-EMOA算法采用在每次迭代中只产生和淘汰一个个体的稳态进化策略，在二维和三维的测试问题上表现出较好的综合性能。

上述基于超体积指标的算法具有计算复杂度

收稿日期：2019-09-17
基金项目：国家自然科学基金(62076095, 61973120, 61673175)
作者简介：王学武(1972—)，男，陕西合阳人，副教授，博士，主要研究方向为智能优化算法、焊接机器人智能化技术、焊接自动化、系统建模、控制与优化。E-mail: wangxuew@ecust.edu.cn
引用本文：王学武, 魏建斌, 周 昕, 等. 一种基于超体积指标的多目标进化算法 [J]. 华东理工大学学报(自然科学版), 2020, 46(6): 780-791.
Citation: WANG Xuewu, WEI Jianbin, ZHOU Xin, *et al*. Hypervolume-Based Multi-Objective Evolutionary Algorithm[J]. Journal of East China University of Science and Technology, 2020, 46(6): 780-791.

高、程序运行效率低的缺点, 因此选择有效计算超体积的方法来提升算法的运行效率并且获得较好的综合性能对基于超体积指标的进化算法来说具有重要意义。

1　相关工作

计算超体积的方法主要分为两类: 精确计算和近似估计, 这两类方法的共同目的均是要平衡运行效率和计算精度。在近似估计超体积的方法中, Hernández 等[14] 提出了超体积牛顿方法(HNM), 将其与 SMS-EMOA 结合能够快速有效地解决连续双目标优化问题。Deng 等[15] 指出精确计算超体积是一个 NP-难问题, 提出通过极坐标变换降维, 与蒙特卡洛模拟方法相比, 该方法在对随机点进行排序时更加稳定。Shang 等[16] 提出了一种基于 R2 指标的 R2-HVC 估计方法, 通过使用不同的线段来近似超体积贡献值, 提高了计算效率和估计精度。在精确计算超体积的方法中, While 等[17] 提出了将目标进行切片的 HSO 算法, 能够有效计算超体积, 在此基础上改进的还有 SHSO*[10]、IIHSO[18] 和 FPL[19] 算法。While 等[20] 通过计算解集上每个点的独立超体积进行迭代, 生成许多的被支配点来加速计算, 降低了计算复杂度, 并通过将 IHSO 算法中的最佳优先级排队机制和 WFG 算法中生成许多被支配点来加速计算的方式进行结合, 提出了 IWFG[21] 算法。文献 [22-24] 将超体积计算问题转换为 Klee 的度量问题, 其最坏情况的时间复杂度为 $O\left(n\lg n+\frac{nd}{2}\lg n\right)$ 或 $O\left(n\lg n+\frac{nd}{2}\right)$ [25], 在四维的超体积计算中时间复杂度降低为 $O\left(\frac{nd}{2}\right)$ [26]。

基于分而治之的思想, FHV 算法[27] 能够并行计算超体积, 处理高维多目标优化问题时非常有效。QHV 算法[28] 能够降低时间和空间复杂度, 获得了较好的算法性能。Lacour 等[29] 通过将支配区域划分为超矩形来计算超体积指标, 提出了超体积盒子分解算法(HBDA), 并提出了一种非增量方法和一种增量方法, 其时间复杂度分别为 $O(n^{\left\lfloor\frac{p-1}{2}\right\rfloor+1})$ 和 $O(n^{\left\lfloor\frac{p}{2}\right\rfloor+1})$。Jiang 等[30] 通过删除不相关的解和转移超体积贡献值提出了 FV-MOEA 算法, 大大减少了精确计算超体积的计算量。

通过超体积指标来指导算法搜索, 其 HV 性能指标普遍较高, 但仍然存在运行效率低、分布性不好的缺点, 而基于参考点的 NSGA-Ⅲ搜索的解分布均匀, 因此本文提出了一种基于超体积指标的多目标进化算法(MOEA-HV), 针对二维和三维的多目标优化问题, 采用分治递归思想的切片法来精确计算超体积, 同时在基于指标的进化算法(IBEA)[11] 前对所有种群个体进行非支配排序提前删除被支配的个体, 从而减少个体独立贡献超体积的计算量, 提升运行效率, 通过与 NSGA-Ⅲ结合来优化算法的分布性。

2　MOEA-HV 算法

2.1　IBEA

IBEA 主要包括两个部分: 一是将超体积指标与适应度评价函数进行结合, 计算种群中每个个体的适应度值, 公式如下:

$$F(x^1)=\sum_{x^2\in P\setminus\{x^1\}}-\mathrm{e}^{-I(\{x^2\},\{x^1\})/k},x^1\in P \tag{1}$$

其中: $I(\{x^2\},\{x^1\})$ 表示计算个体的独立贡献超体积; k 是一个大于 0 的比例缩放因子[31], 用于淘汰适应度值小的个体, 即在原种群中删除对总的超体积的独立贡献最小的个体, 如图 1 和图 2 所示(以二维为例)。

在图 1、2 中, A 和 B 均表示一个个体, H 表示计算超体积的参考点, 阴影部分 A_{HV} 和 B_{HV} 表示个体 A 和 B 的独立支配区域面积, 其中图 1 表示个体

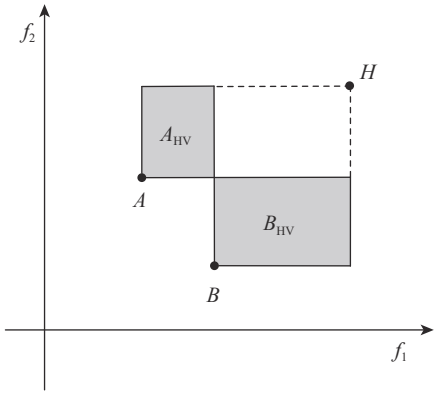

图 1　在非支配关系下 $B_{\mathrm{HV}}>A_{\mathrm{HV}}>0$

Fig. 1　$B_{\mathrm{HV}}>A_{\mathrm{HV}}>0$ in non-dominated relationship

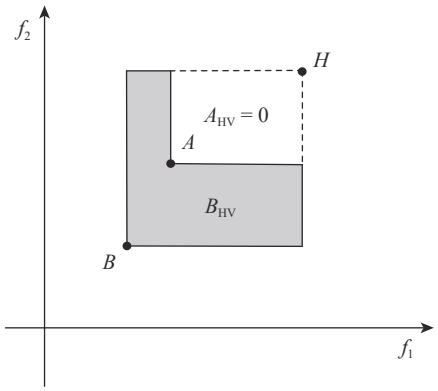

图 2　在支配关系下 $B_{\mathrm{HV}}>A_{\mathrm{HV}}=0$

Fig. 2　$B_{\mathrm{HV}}>A_{\mathrm{HV}}=0$ in dominated relationship

A 和 B 在互不支配时的独立支配区域面积，图 2 表示个体 B 支配个体 A 时的独立支配区域面积，均有 $B_{HV}>A_{HV}$，因此不论是根据个体的支配关系还是个体对总的超体积的独立贡献，更应该淘汰个体 A。事实上，超体积指标在 Pareto 支配关系上是严格单调的，因此能够作为指导种群进化的有效度量指标[9]。

重新计算种群中每个个体的适应度值，其更新公式为

$$F(x)=F(x)+\mathrm{e}^{-I(\{x*\},\{x\})/k},x\in P \tag{2}$$

通过二元锦标选择父代进行交配从而产生新的个体，与原种群一起构成新的种群，继续进行迭代。

IBEA 通过将超体积指标和适应度评价方法相结合来指导种群进化，与基于超体积指标的其他典型算法 HypE[9]、MO-CMA-ES[12] 和 SMS-EMOA[13] 相比，在保证综合性能最优或次优的前提下，其运行效率是最高的，这也是本文选择 IBEA 作为算法基础框架的主要原因。

2.2 NSGA-Ⅲ

NSGA-II[32] 是基于拥挤度距离建立偏序集来筛选个体指导种群进化，能够有效处理多目标优化问题。NSGA-Ⅲ[33] 在 NSGA-II 的基础上通过使用一组预定义的参考点在临界层选择个体来确保种群良好的分布性，该算法在最坏情况下的时间复杂度为 $O(N^2\lg^{M-2}N)$ 或 $O(N^2M)$。

基于参考点方法的 NSGA-Ⅲ能够维持种群个体分布的多样性和均匀性，因此将 NSGA-Ⅲ整合到 IBEA 框架中，获得的解能够同时具有良好的收敛性和分布性，使得程序的运行效率更高，同时解也能够均匀分布在整个 Pareto 前沿面。有两种方法可以将 NSGA-Ⅲ和 IBEA 结合：一是在 IBEA 运行完成的基础上再运行 NSGA-Ⅲ，也就是 IBEA 先迭代更新完成，获得收敛性较好的种群，然后将获得的种群作为 NSGA-Ⅲ的初始种群继续进行迭代优化，以期获得分布性也较好的解；二是先运行 NSGA-Ⅲ，然后将获得的分布均匀的种群作为 IBEA 的初始种群、利用超体积指标指导种群进化来继续对选择过程增压，以期获得综合性能均较好的解。

本文选择第 1 种结合方式。因为在基于超体积指标的算法中，IBEA 的收敛速度较快，分布性较差，因此重点在于改善 IBEA 的分布性，在 IBEA 运行的基础上再运行 NSGA-Ⅲ能够实现此目的。而对于第 2 种方式，先利用 NSGA-Ⅲ获得分布性较好的解，然后在此基础上通过 IBEA 继续进行个体的迭代选择，很可能会破坏之前已经均匀分布的解，反而降低了算法的综合性能。

2.3 非支配排序

IBEA 通过计算种群中每个个体的独立贡献超体积及其适应度值淘汰适应度值最小的个体，并依次进行迭代，使得算法最终能获得一个较好的解集，但计算复杂度高，主要原因在于 IBEA 计算了每个个体的独立贡献超体积。本文的改进之处是先通过 Pareto 支配关系对个体进行筛选，得到一组非支配解集，然后计算非支配解集中每个个体的独立贡献超体积，淘汰贡献值最小的个体并依次迭代。通过提前删除质量不好的个体而避免计算其独立贡献超体积，能够有效节约个体的超体积计算量，而且减小了计算复杂度，提升运行效率。以二维为例，图 3 示出了非支配排序后的解集分布情况，很容易计算出每个个体的独立贡献超体积(即面积)。图 4 示出了经过初始化后产生的种群个体分布情况，处于乱序状态，此时要淘汰独立贡献超体积最小的个体仍然需要计算每个个体的贡献值，浪费了部分计算资源，因此本文考虑在 IBEA 前引入非支配排序来提升算法的运行效率。

三维的非支配排序情况与二维类似，但个体独

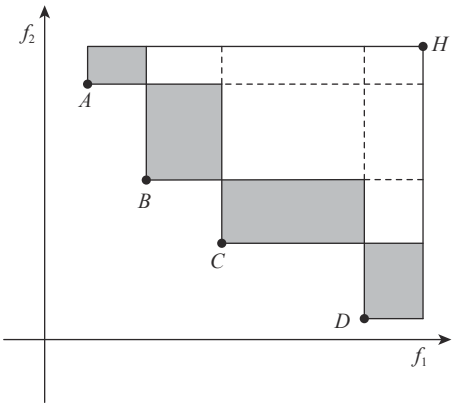

图 3 非支配排序后的解集

Fig. 3 Solution set after non-dominated sort

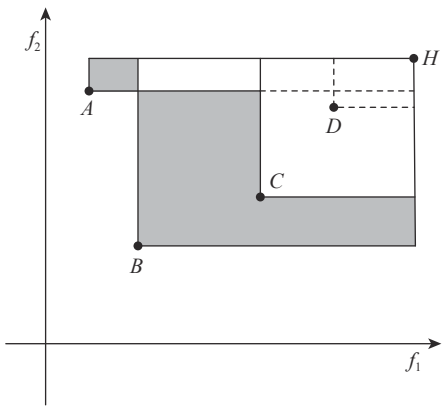

图 4 乱序下的解集

Fig. 4 Solution set in chaos

立贡献超体积的计算更加复杂。本文采用切片法来精确计算三维个体的独立贡献超体积, 基本思想是计算整个解集的超体积与去除该个体后的超体积之差。具体来说, 先利用某一维度(本文选择第一维度)对整个超体积进行切片以达到降维的目的, 通过计算切割面的面积与切片在第一维度上的深度的乘积得到整个解集的超体积, 然后用同样的方法计算去除该个体后的超体积, 两个超体积之差即为三维情况下的个体独立贡献超体积(即体积)。假设解集中有 4 个个体 P_1、P_2、P_3、P_4, 图 5(a)、(b)、(c)和(d)分别示出了整个解集的超体积、经过切片后的子超体积、个体 P_1 和 P_2 的独立贡献超体积。通过精确计算每个个体的独立贡献超体积, 淘汰独立贡献超体积最小的个体来指导种群进化。

2.4　MOEA-HV 算法步骤

通过精确计算种群中个体的独立贡献超体积来指导种群进化, 在 IBEA 前对所有种群个体进行非支配排序并与 NSGA-Ⅲ结合, 共同组成了 MOEA-HV 算法。算法步骤如下:

输入: 种群数量、组合迭代次数、参数 k、参考点

输出: Pareto 前沿面

(1)初始化。获得初始种群, 并生成参考点。

(2)非支配排序。对种群进行非支配排序, 提前删除被支配的个体, 减少超体积的计算量。

(3)计算非支配排序后种群中每个个体的适应度值, 并淘汰适应度值最小的个体。

(4)选择、交叉、变异, 产生新的种群。利用二元锦标法选择最优的两个个体到交配池进行交叉变异, 产生新的个体与原种群合并为新的种群。

(5)嵌入 NSGA-Ⅲ。将新得到的种群作为 NSGA-Ⅲ的初始种群继续进行迭代, 满足迭代要求则停止迭代并输出。

3　实验结果及分析

仿真实验使用 Lenovo-PC 计算机, 处理器为 Intel(R) Core(TM) i5-4200M CPU @ 2.50 GHz, 内存为 4.00 GB, 操作系统为基于 x64 处理器的 64 位操作系统 Windows 8.1 中文版, 软件版本为 MATLAB R2018a, 程序运行均在 PlatEMO[34] 平台实现。

为检验 MOEA-HV 的运行效率和综合性能, 与其他典型的基于超体积的算法 IBEA[11]、HypE[9]、MO-CMA-ES[12]、SMS-EMOA[13] 以及 NSGA-Ⅲ[33] 进行对比实验, 选择 2 个目标的 ZDT 和 3 个目标的 DTLZ 系列中具有代表性的典型测试函数 ZDT1~ZDT6 和 DTLZ1~DTLZ7 进行实验, 通过运行时间评价算法的运行效率, 通过 IGD[35] 和 HV[36] 指标评价算法的综合性能, 其中 IGD 值越小说明算法的综合性能越好, HV 值越大说明算法的综合性能越好, 这两个评价指标都能够综合评价算法的收敛性和分布性。设置种群大小为 100, 迭代次数为 500, 所有算法在相同条件下独立运行 10 次, k=0.03。表 1~3 分别列出了各算法的运行时间、IGD 均值(IGD_m)和 HV 均值(HV_m), 其中黑体数字为最优值。图 6 示出

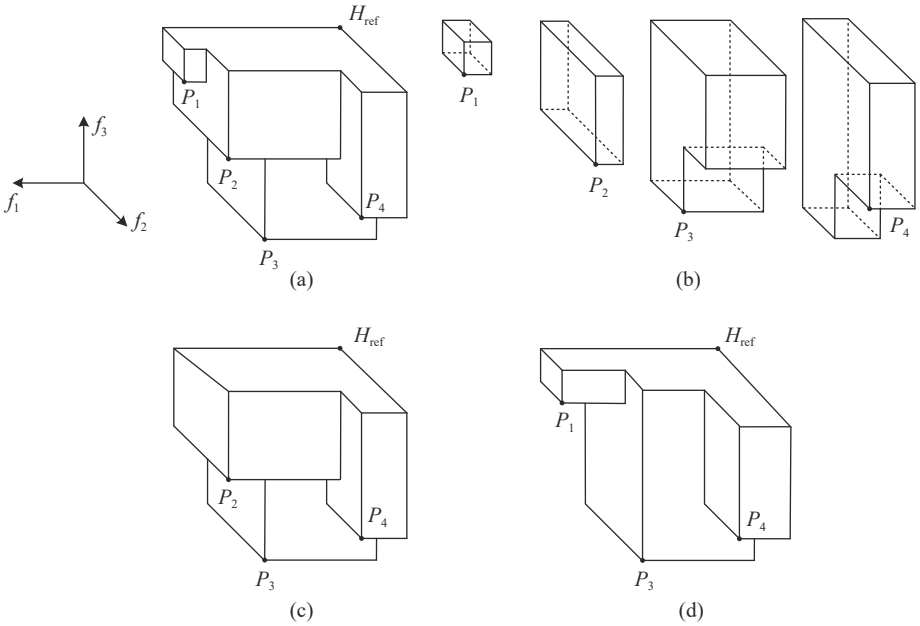

图 5　三维情况下个体独立贡献超体积的计算

Fig. 5　Calculation of the individuals' exclusive hypervolume contributions in three dimension

了 MOEA-HV 算法在测试函数 ZDT2、ZDT3、DTLZ2 和 DTLZ7 上的 Pareto 前沿面。

从表 1 中各算法的运行时间均值来看，MOEA-HV 的运行效率要远远高于其他基于超体积的典型算法，在 IBEA 的基础上提升近两倍，但比 NSGA-Ⅲ要慢，究其原因在于 NSGA-Ⅲ不需要计算超体积，而对于基于超体积的算法来说，不论采用何种策略、何种计算方法，超体积的时间计算成本仍然非常大。对于基于超体积的算法来说，计算复杂度高是制约算法性能最重要的因素，因此 MOEA-HV 最大

表 1 各算法运行时间均值

Table 1 Mean runtime of each algorithm

Function	Runtime/s					
	IBEA	HypE	MO-CMA-ES	SMS-EMOA	NSGA-Ⅲ	MOEA-HV
ZDT1	5.805 9	$2.382\ 7\times10^{2}$	$2.505\ 1\times10^{1}$	$1.037\ 1\times10^{2}$	**2.294 8**	3.565 1
ZDT2	5.677 5	$2.093\ 2\times10^{2}$	$2.030\ 5\times10^{1}$	$1.037\ 1\times10^{2}$	**2.562 5**	3.794 6
ZDT3	6.307 8	$2.180\ 1\times10^{2}$	$2.114\ 8\times10^{1}$	$1.037\ 1\times10^{2}$	**2.476 0**	3.664 3
ZDT4	5.921 3	$4.812\ 4\times10^{2}$	$5.670\ 2\times10^{1}$	$3.082\ 6\times10^{2}$	**2.109 9**	3.363 5
ZDT5	5.803 1	$1.219\ 2\times10^{2}$	$4.012\ 4\times10^{1}$	$9.799\ 1\times10^{1}$	**2.432 5**	3.764 6
ZDT6	6.341 6	$2.758\ 3\times10^{2}$	$2.182\ 9\times10^{1}$	$1.056\ 9\times10^{2}$	**2.132 2**	3.442 4
DTLZ1	6.750 3	$1.530\ 4\times10^{3}$	$2.123\ 8\times10^{1}$	$1.400\ 5\times10^{2}$	**2.254 0**	3.193 0
DTLZ2	6.557 4	$2.574\ 3\times10^{3}$	$2.925\ 0\times10^{1}$	$1.764\ 9\times10^{3}$	**2.366 4**	3.392 4
DTLZ3	6.540 3	$7.428\ 3\times10^{2}$	$2.852\ 4\times10^{1}$	$7.409\ 8\times10^{2}$	**2.315 8**	3.418 9
DTLZ4	6.984 5	$1.607\ 4\times10^{3}$	$2.712\ 1\times10^{1}$	$1.798\ 2\times10^{3}$	**3.156 7**	3.647 5
DTLZ5	8.936 0	$1.952\ 2\times10^{3}$	$3.120\ 9\times10^{1}$	$9.536\ 4\times10^{2}$	**3.634 6**	4.783 0
DTLZ6	7.703 3	$3.425\ 2\times10^{3}$	$2.469\ 5\times10^{1}$	$9.919\ 2\times10^{2}$	**3.865 8**	5.082 8
DTLZ7	6.612 4	$1.945\ 6\times10^{3}$	$2.041\ 4\times10^{1}$	$1.591\ 3\times10^{3}$	**2.695 8**	3.815 0

表 2 各算法 IGD 均值

Table 2 Mean IGD of each algorithm

Function	IGD_m					
	IBEA	HypE	MO-CMA-ES	SMS-EMOA	NSGA-Ⅲ	MOEA-HV
ZDT1	$4.587\ 6\times10^{-3}$	$2.353\ 4\times10^{-1}$	$6.216\ 7\times10^{-3}$	**$3.651\ 3\times10^{-3}$**	$3.887\ 9\times10^{-3}$	$3.887\ 9\times10^{-3}$
ZDT2	$9.304\ 7\times10^{-3}$	$2.217\ 0\times10^{-1}$	$6.057\ 1\times10^{-3}$	$4.320\ 3\times10^{-3}$	$3.807\ 7\times10^{-3}$	**$3.807\ 2\times10^{-3}$**
ZDT3	$6.669\ 7\times10^{-2}$	$1.988\ 6\times10^{-1}$	**$8.550\ 1\times10^{-3}$**	$2.675\ 5\times10^{-2}$	$1.192\ 7\times10^{-2}$	$2.914\ 5\times10^{-2}$
ZDT4	$1.923\ 0\times10^{-2}$	$4.601\ 2\times10^{-1}$	8.874 1	**$3.627\ 3\times10^{-3}$**	$4.112\ 8\times10^{-3}$	$4.192\ 9\times10^{-3}$
ZDT5	2.330 0	1.782 9	$3.374\ 9\times10^{-1}$	**$4.727\ 1\times10^{-1}$**	$4.873\ 8\times10^{-1}$	$5.749\ 2\times10^{-1}$
ZDT6	$5.027\ 5\times10^{-3}$	$3.070\ 2\times10^{-3}$	$4.025\ 2\times10^{-3}$	**$2.986\ 3\times10^{-3}$**	$3.002\ 4\times10^{-3}$	$3.001\ 5\times10^{-3}$
DTLZ1	$1.705\ 1\times10^{-1}$	$1.338\ 3\times10^{-1}$	8.173 3	$3.179\ 1\times10^{-2}$	$2.062\ 4\times10^{-2}$	**$2.061\ 9\times10^{-2}$**
DTLZ2	$8.041\ 8\times10^{-2}$	$1.337\ 4\times10^{-1}$	$1.122\ 5\times10^{-1}$	$7.753\ 1\times10^{-2}$	$5.447\ 0\times10^{-2}$	**$5.446\ 8\times10^{-2}$**
DTLZ3	$4.726\ 8\times10^{-1}$	$3.652\ 5\times10^{-1}$	$5.184\ 0\times10^{1}$	$8.768\ 3\times10^{-2}$	$5.782\ 4\times10^{-2}$	**$5.777\ 5\times10^{-2}$**
DTLZ4	$8.088\ 1\times10^{-2}$	$4.716\ 6\times10^{-1}$	$1.313\ 8\times10^{-1}$	$7.801\ 5\times10^{-2}$	$2.897\ 3\times10^{-1}$	**$5.449\ 7\times10^{-2}$**
DTLZ5	$1.707\ 6\times10^{-2}$	$1.513\ 4\times10^{-2}$	**$1.125\ 8\times10^{-2}$**	$1.609\ 3\times10^{-2}$	$1.306\ 4\times10^{-2}$	$1.289\ 6\times10^{-2}$
DTLZ6	$2.312\ 1\times10^{-2}$	$2.176\ 0\times10^{-1}$	$5.023\ 0\times10^{-2}$	**$1.824\ 3\times10^{-2}$**	$1.977\ 2\times10^{-2}$	$1.894\ 5\times10^{-2}$
DTLZ7	$1.487\ 3\times10^{-1}$	$6.986\ 2\times10^{-1}$	$1.244\ 8\times10^{-1}$	$7.642\ 8\times10^{-2}$	**$7.594\ 8\times10^{-2}$**	$7.685\ 3\times10^{-2}$

表 3　各算法 HV 均值

Table 3　Mean HV of each algorithm

Function	HV_m					
	IBEA	HypE	MO-CMA-ES	SMS-EMOA	NSGA-Ⅲ	MOEA-HV
ZDT1	$7.199\,3\times10^{-1}$	$5.832\,3\times10^{-1}$	$7.173\,8\times10^{-1}$	$\mathbf{7.207\,6\times10^{-1}}$	$7.202\,9\times10^{-1}$	$7.203\,0\times10^{-1}$
ZDT2	$4.434\,7\times10^{-1}$	$2.409\,5\times10^{-1}$	$4.423\,6\times10^{-1}$	$\mathbf{4.453\,3\times10^{-1}}$	$4.450\,2\times10^{-1}$	$4.450\,3\times10^{-1}$
ZDT3	$6.872\,6\times10^{-1}$	$\mathbf{7.219\,1\times10^{-1}}$	$5.819\,7\times10^{-1}$	$6.411\,1\times10^{-1}$	$6.015\,1\times10^{-1}$	$6.398\,9\times10^{-1}$
ZDT4	$7.049\,8\times10^{-1}$	$6.923\,3\times10^{-1}$	–	$\mathbf{7.207\,2\times10^{-1}}$	$7.197\,1\times10^{-1}$	$7.190\,0\times10^{-1}$
ZDT5	$8.170\,4\times10^{-1}$	$8.027\,8\times10^{-1}$	$9.711\,4\times10^{-1}$	$8.200\,6\times10^{-1}$	$\mathbf{8.312\,9\times10^{-1}}$	$8.265\,4\times10^{-1}$
ZDT6	$3.871\,2\times10^{-1}$	$3.889\,2\times10^{-1}$	$3.879\,8\times10^{-1}$	$\mathbf{3.890\,0\times10^{-1}}$	$3.889\,3\times10^{-1}$	$3.889\,5\times10^{-1}$
DTLZ1	$4.637\,1\times10^{-1}$	$6.104\,9\times10^{-1}$	–	$8.149\,5\times10^{-1}$	$8.408\,9\times10^{-1}$	$\mathbf{8.409\,6\times10^{-1}}$
DTLZ2	$5.580\,1\times10^{-1}$	$5.363\,0\times10^{-1}$	$4.208\,6\times10^{-1}$	$5.484\,5\times10^{-1}$	$5.595\,8\times10^{-1}$	$\mathbf{5.596\,0\times10^{-1}}$
DTLZ3	$2.441\,0\times10^{-1}$	$2.736\,3\times10^{-1}$	–	$5.193\,2\times10^{-1}$	$5.403\,4\times10^{-1}$	$\mathbf{5.416\,5\times10^{-1}}$
DTLZ4	$5.574\,5\times10^{-1}$	$3.532\,0\times10^{-1}$	$4.828\,1\times10^{-1}$	$5.481\,2\times10^{-1}$	$4.469\,6\times10^{-1}$	$\mathbf{5.594\,2\times10^{-1}}$
DTLZ5	$\mathbf{1.986\,6\times10^{-1}}$	$1.960\,3\times10^{-1}$	$1.943\,1\times10^{-1}$	$1.949\,8\times10^{-1}$	$1.932\,6\times10^{-1}$	$1.938\,4\times10^{-1}$
DTLZ6	$\mathbf{1.970\,0\times10^{-1}}$	$1.102\,8\times10^{-1}$	$1.776\,4\times10^{-1}$	$1.950\,7\times10^{-1}$	$1.899\,6\times10^{-1}$	$1.908\,9\times10^{-1}$
DTLZ7	$2.695\,3\times10^{-1}$	$2.056\,7\times10^{-1}$	$2.513\,7\times10^{-1}$	$\mathbf{2.725\,3\times10^{-1}}$	$2.694\,7\times10^{-1}$	$2.705\,0\times10^{-1}$

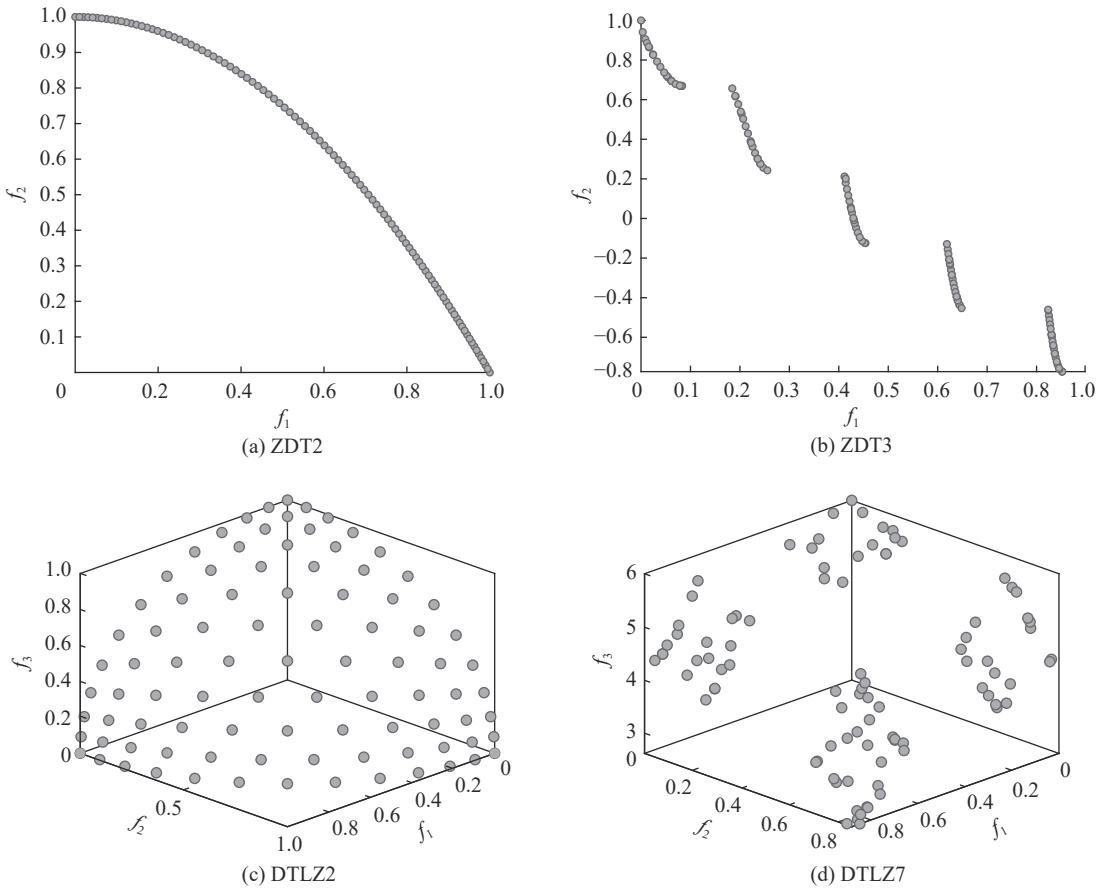

图 6　MOEA-HV 算法在部分测试函数上的 Pareto 前沿面

Fig. 6　Pareto fronts of MOEA-HV on some test functions

的突破就是在保持良好收敛性和分布性的同时能有效降低计算复杂度, 节约计算超体积的时间成本, 提高算法的运行效率。

从表 2 和表 3 中各算法的 IGD_m 和 HV_m 可以看

出，MOEA-HV 的收敛性和分布性要明显优于 IBEA，同时与 NSGA-Ⅲ结合后算法综合性能与 NSGA-Ⅲ相比也更优，与其他算法相比能够取得最优或次优的综合性能，且性能稳定。总体来看，MOEA-HV 在 DTLZ 系列测试函数上的性能要优于在 ZDT 系列测试函数上的性能，因为加入非支配排序和基于参考点的 NSGA-Ⅲ算法后，三维解集中的非支配个体要比二维的多，因此 MOEA-HV 的解集在 Pareto 前沿面上分布会更加均匀。具体来说，MOEA-HV 在测试函数 ZDT1、ZDT2、ZDT5、ZDT6 和 DTLZ1~DTLZ7 上的综合性能最优或次优，说明该算法能够较好地处理具有凹性、多模态、偏好的问题，但在测试函数 ZDT3、ZDT4 上性能表现不佳，算法在处理不连续以及多模态混合问题时尚且有待改进，同时测试函数 ZDT5 具有欺骗性，导致算法容易陷入局部最优，解的分布性较差，测试函数 DTLZ5 和 DTLZ6 具有退化性，比较难收敛，因此测试的所有算法在这两个测试函数上表现的收敛性均较差。与其他基于超体积的典型算法相比，MOEA-HV 的运行效率更高，且能够保持较好的收敛性和分布性。

3.1 参数 k 的影响

MOEA-HV 算法是基于 IBEA 的框架，其中，适应度值计算函数中的参数 k 对算法的综合性能有一定影响，因此有必要对 k 的选择进行独立的对比实验，选择使算法表现出最好性能的参数值。设置种群大小为 100，迭代次数为 500，IBEA 在相同条件下独立运行 10 次，参数 k 分别为 0.030、0.035、0.040、0.045、0.050、0.055、0.060，进行 7 组对比实验，测试函数为 ZDT1~ZDT6和 DTLZ1~DTLZ7，分别比较不同参数下算法 IBEA 的运行时间均值、IGD_m 和 HV_m，结果如图 7 所示。

相同条件下程序的运行时间越短说明算法的运行效率越高。从图 7(a)中可以看出参数 k 取 0.030 时，在 ZDT3、DTLZ1~DTLZ4、DTLZ6 上能够得到最小值，k 取 0.035 时运行时间均值也较短，说明参数设置得较小能够在一定程度上节约算法的时间成

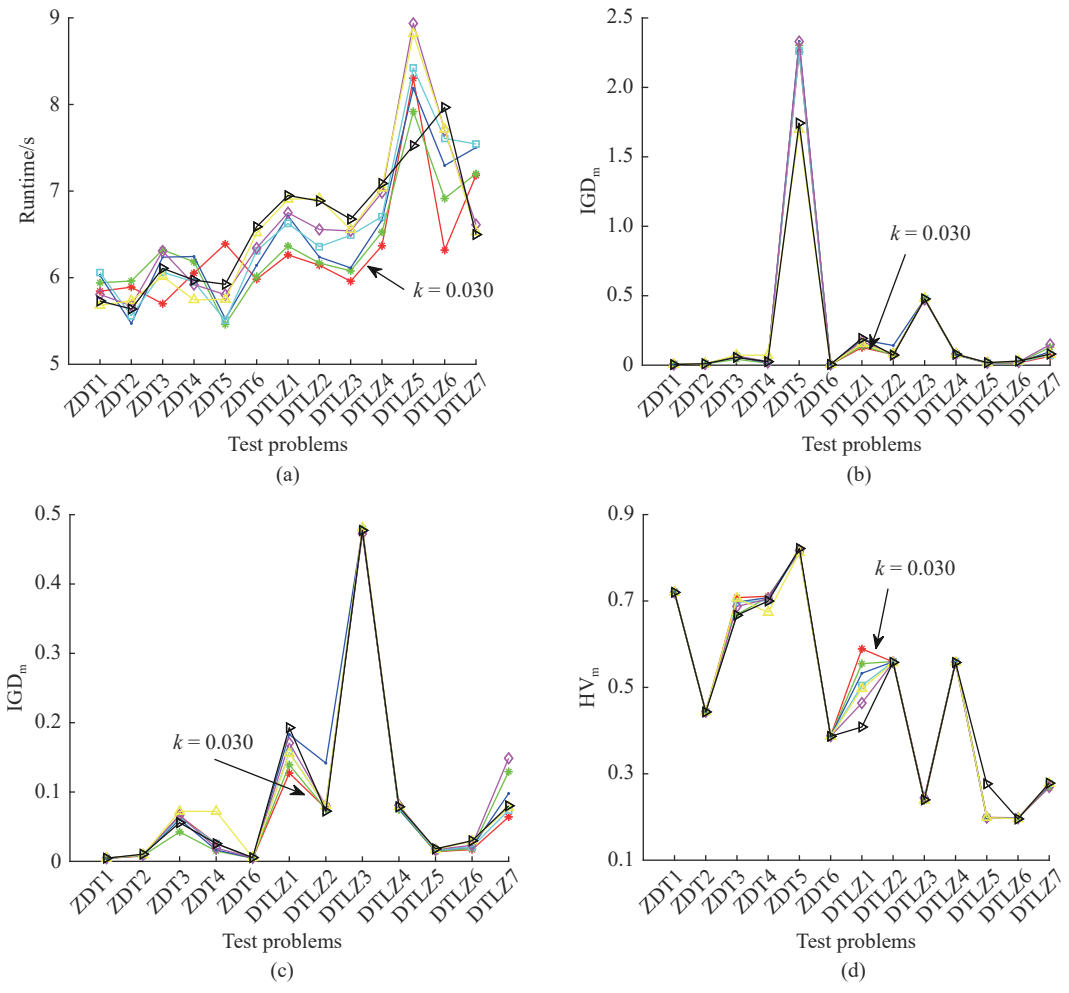

图 7 不同参数 k 对 IBEA 运行时间、IGD_m 值和 HV_m 值的影响

Fig. 7 Influence of different k on runtime, IGD_m and HV_m of IBEA

本, 且对于 ZDT 系列测试函数来说, 运行时间受参数设置影响较小, 而对于 DTLZ 系列测试函数来说, k 取 0.030 具有较大优势。从图 7(b)可以看出, 测试函数 ZDT5 的分布性较差, 不论参数取何值, 其 IGD_m 值都远远超过其他测试函数的 IGD_m 值, 因此考虑去掉 ZDT5 测试函数来比较不同参数对算法 IGD_m 值的影响。如图 7(c)所示, k 取 0.030 时在 DTLZ1、DTLZ7 上的 IGD_m 值明显小于其他参数的 IGD_m 值, 且比较稳定, 说明 k 取 0.030 时算法的解集在 Pareto 前沿面上分布更加均匀。从图 7(d)可以看出 k 取 0.030时, 算法在测试函数 ZDT3 和 DTLZ1 上的 HV_m 值最大, 在其他测试函数上也好于其他参数, 且性能稳定, 收敛性更好, 综上可认为在比较的 7 个参数中, k 取 0.030时 IBEA 的运行效率更高, 且具有良好的综合性能, 因此为了获得最好的算法性能, MOEA-HV 中的参数 k 也取为 0.030。

3.2　非支配排序的影响

IBEA 的计算复杂度高, 主要原因是需要计算每个个体的独立贡献超体积, 而通过 Pareto 支配关系对个体进行筛选能够提前删除质量不好的个体从而降低了整个解集中个体的超体积计算量, 能够有效提高程序的运行效率。设置种群大小为 100, 迭代次数为 500, 在相同条件下独立运行 10 次, 对在 IBEA 中是否加入非支配排序进行对比实验以探究非支配排序对算法的影响。实验的测试函数为 ZDT1~ZDT6 和 DTLZ1~DTLZ7, 分别比较 IBEA 中是否加入非支配排序的运行时间均值、IGD_m 值和 HV_m 值, 如图 8 所示, 其中 NDSort+IBEA(Mean) 和 NDSort+IBEA(Sd) 分别表示算法 IBEA 加入非支配排序后所获得的平均值以及标准差, 而 IBEA(Mean) 和 IBEA(Sd) 则表示没有加入非支配排序得到的平均值和标准差。

对于 2 个目标的 ZDT 系列测试函数来说, 计算的超体积为二维的(即面积), 比较简单, 使得加入非支配排序程序的算法运行时间与 IBEA 相近甚至更长。因为加入非支配排序后算法的运行效率更高, 但同时算法的程序也更长, 导致前者节约的时间与后者所消耗的时间刚好抵消, 因此在迭代之后两者的运行时间相近。从图 8(a)中可以看出, 对于 3 个目标的 DTLZ 系列测试函数来说, 加入非支配排序后算法的运行时间显著降低, 是因为三维的超体积更难计算, 经过非支配排序之后能有效删除不好的个体, 从而大大减少超体积的计算量。同时, 从图 8(b)、(c)中可以看出, 加入非支配排序后算法的性能更加稳定, 且能在一定程度上提高算法的综合性能。

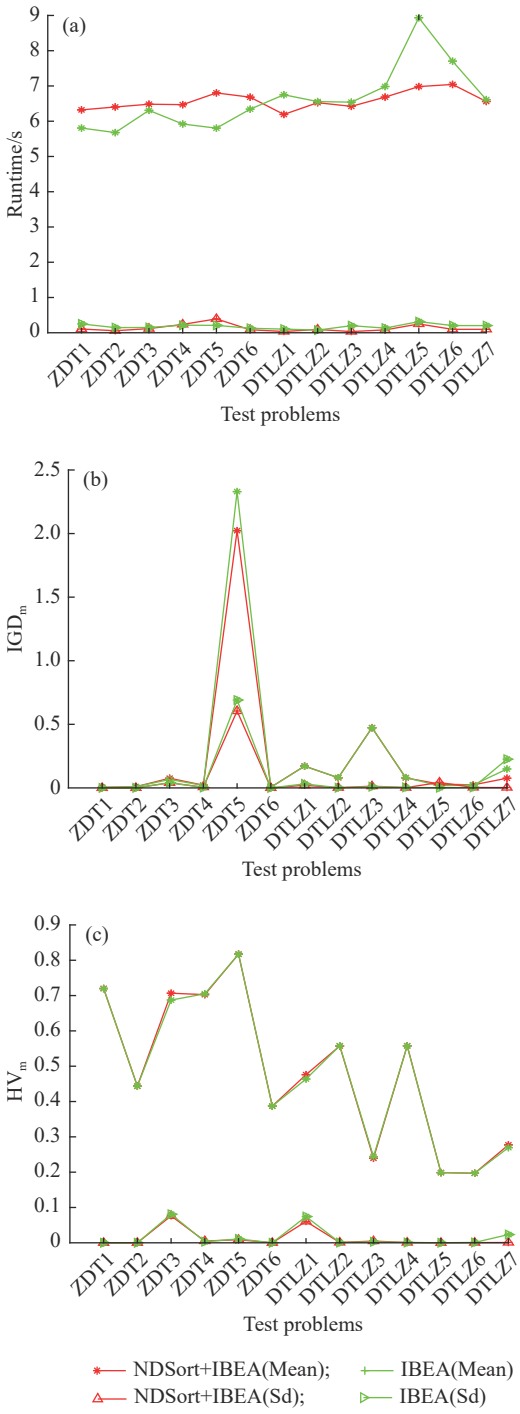

图 8　非支配排序对 IBEA 运行时间均值、IGD_m 值和 HV_m 值的影响

Fig. 8　Influence of non-dominated sort on runtime, IGD_m and HV_m of IBEA

3.3　NSGA-Ⅲ的影响

将 IBEA 与 NSGA-III 结合能明显改善算法的分布性, 但关键在于如何结合, 本文主要考虑两种结合方式: 一是 IBEA 先迭代更新完成获得收敛性较好的种群, 然后将获得的该种群作为 NSGA-III 的初始种群继续进行迭代优化, 以期获得分布性也较好的解(简称 IBEA+NSGA-III); 二是先运行 NSGA-III, 然后

将获得分布均匀的种群作为IBEA的初始种群，利用超体积指标进化继续增压选择，以期获得综合性能均较好的解(简称NSGA-III+IBEA)。设置种群大小为100，两种结合方式中IBEA迭代次数均为100，NSGA-III迭代次数均为500，在相同条件下独立运行10次，通过实验分析决定何种方式更有利于整合到MOEA-HV算法中。实验的测试函数为ZDT1~ZDT6和DTLZ1~DTLZ7，分别比较IBEA与两种不同结合方式IBEA+NSGA-III、NSGA-III+IBEA的运行时间均值、IGD_m值、HV_m值和标准差(Standard deviation, Sd)，如图9所示。

从图9(a)中可以看出，与NSGA-III结合后算法的运行时间大幅下降，且IBEA+NSGA-III比其他两者的综合运行时间更短，但从图9(b)的标准差数据来看，NSGA-III+IBEA比IBEA+NSGA-III的运行时间更加稳定，因为算法NSGA-III的运行速度快，且分布性好，所获得的初始种群较好，因此第2种结合方式(NSGA-Ⅲ+IBEA)的运行时间浮动更小、更稳定。

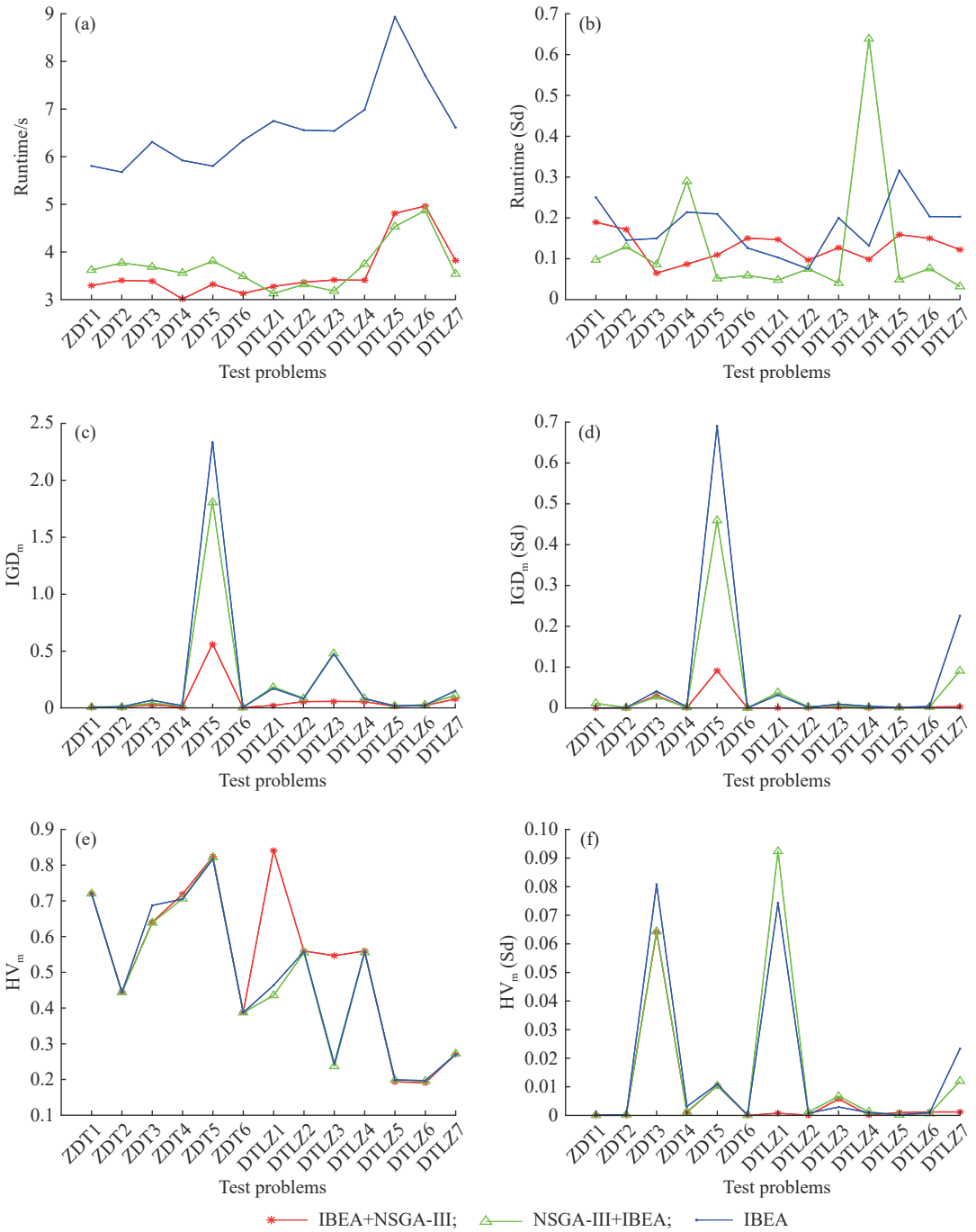

图9 NSGA-Ⅲ对IBEA运行时间、IGD_m值和HV_m值的影响

Fig. 9 Influence of NSGA-Ⅲ on runtime, IGD_m and HV_m of NSGA-Ⅲ

图 9(c)~(f)表明 IBEA+NSGA-III 在所有测试函数上的 IGD_m 值最小、HV_m 值最大, 性能最稳定, 且在 ZDT5、DTLZ1 和 DTLZ3 测试函数上要明显优于 NSGA-III+IBEA, 因此本文的组合算法选择 IBEA+NSGA-III 结合方式。

4　结　论

针对二维和三维的多目标优化问题, 本文提出了一种基于超体积指标的多目标进化算法 (MOEA-HV)。利用分治递归思想的切片法精确计算种群中个体的独立贡献超体积来指导种群进化, 通过在 IBEA 前对所有种群个体进行非支配排序提前删除被支配的个体, 从而减少个体独立贡献超体积的计算量来提升运行效率, 同时通过与 NSGA-Ⅲ结合来优化算法的分布性, 主要有 3 点改进之处: (1)设计对比实验选择更好的参数; (2)通过非支配排序提前删除不好的个体, 能有效节约计算超体积的时间成本; (3)与 NSGA-Ⅲ结合来优化算法的分布性。

在今后的研究中, 考虑将算法 MOEA-HV 应用于解决焊接机器人多目标路径规划问题和更高维的问题, 并会继续研究精确计算和近似估计超体积的方法, 更加有效地提升算法的运行效率, 同时结合其他的优化策略持续改善算法的综合性能。

参考文献:

[1] 朱永胜, 王杰, 瞿博阳, 等. 采用基于分解的多目标进化算法的电力环境经济调度[J]. 电网技术, 2014, 38(6): 1 577-1 584.

[2] MURUGESWARI R, RADHAKRISHNAN S, DEVARAJ D. A multi-objective evolutionary algorithm based QoS routing in wireless mesh networks[J]. Applied Soft Computing, 2016, 40: 517-525.

[3] 王珊珊, 杜文莉, 陈旭, 等. 基于约束骨干粒子群算法的化工过程动态多目标优化[J]. 华东理工大学学报 (自然科学版), 2014, 40(4): 449-457.

[4] RYU N, LIM S, MIN S, *et al*. Multi-objective optimization of magnetic actuator design using adaptive weight determination scheme[J]. IEEE Transactions on Magnetics, 2017, 53(6): 1-4.

[5] 陈志旺, 白锌, 杨七, 等. 区间多目标优化中决策空间约束、支配及同序解筛选策略[J]. 自动化学报, 2015, 41(12): 2 115-2 124.

[6] 郑金华, 李珂, 李密青, 等. 一种基于Hypervolume指标的自适应邻域多目标进化算法[J]. 计算机研究与发展, 2012, 49(2): 312-326.

[7] 王学武, 夏泽龙, 顾幸生. 基于DMOEA/D-ET算法的焊接机器人多目标路径规划[J]. 华南理工大学学报(自然科学版), 2019, 47(4): 99-106.

[8] MAC T T, COPOT C, TRAN D T, *et al*. A hierarchical global path planning approach for mobile robots based on multi-objective particle swarm optimization[J]. Applied Soft Computing, 2017, 59: 68-76.

[9] BADER J, ZITZLER E. HypE: An algorithm for fast hypervolume-based many-objective optimization[J]. Evolutionary Computation, 2011, 19(1): 45-76.

[10] ZHOU X, GUO P, CHEN C L P. An algorithm for calculating the hypervolume contribution of a set[C]// World Automation Congress 2012. Mexico: IEEE, 2012: 439-443.

[11] ZITZLER E, SIMON K. Indicator-based selection in multiobjective search[C]// International Conference on Parallel Problem Solving from Nature. Berlin, Heidelberg: Springer, 2004: 832-842.

[12] IGEL C, HANSEN N, ROTH S. Covariance matrix adaptation for multi-objective optimization[J]. Evolutionary Computation, 2007, 15(1): 1-28.

[13] BEUME N, NAUJOKS B, EMMERICH M. SMS-EMOA: Multiobjective selection based on dominated hypervolume [J]. European Journal of Operational Research, 2007, 181(3): 1 653-1 669.

[14] HERNÁNDEZ V A S, SCHUTZE O, WANG H, *et al*. The set-based hypervolume newton method for bi-objective optimization[J]. IEEE Transactions on Cybernetics, 2020, 50(5): 2 186-2 196.

[15] DENG J, ZHANG Q. Approximating hypervolume and hypervolume contributions using polar coordinate[J]. IEEE Transactions on Evolutionary Computation, 2019, 23(5): 913-918.

[16] SHANG K, ISHIBUCHI H, NI X. R2-based hypervolume contribution approximation[J]. IEEE Transactions on Evolutionary Computation, 2020, 24(1): 185-192.

[17] WHILE L, HINGSTON P, BARONE L, *et al*. A faster algorithm for calculating hypervolume[J]. IEEE Transactions on Evolutionary Computation, 2006, 10(1): 29-38.

[18] BRADSTREET L, WHILE L, BARONE L. A fast many-objective hypervolume algorithm using iterated incremental calculations[C]// 2010 IEEE Congress on Evolutionary Computation. Spain: IEEE, 2010: 1-8.

[19] FONSECA C M, PAQUETE L, LOPEZ-IBANEZ M. An improved dimension-sweep algorithm for the hypervolume indicator[C]// 2006 IEEE International Conference on Evolutionary Computation. Canada: IEEE, 2006: 1 157-1 163.

[20] WHILE L, BRADSTREET L, BARONE L. A fast way of calculating exact hypervolumes[J]. IEEE Transactions on Evolutionary Computation, 2012, 16(1): 86-95.

[21] WHILE L, BRADSTREET L. Applying the WFG

algorithm to calculate incremental hypervolumes[C]// 2012 IEEE Congress on Evolutionary Computation. Australia: IEEE, 2012: 1-8.

[22] OVERMARS M H, YAP C K. New upper bounds in Klee's measure problem[C]//Proceedings 1988 29th Annual Symposium on Foundations of Computer Science. USA: IEEE, 1988: 550-556.

[23] GAZIT H. New upper bounds in Klee 's measure problem[J]. SIAM Journal on Computing, 1991, 20(6): 1 034-1 045.

[24] BEUME N. S-metric calculation by considering dominated hypervolume as Klee 's measure problem[J]. Evolutionary Computation, 2009, 17(4): 477-492.

[25] BEUME N, RUDOLPH G. Faster S-metric calculation by considering dominated hypervolume as Klee 's measure problem[C]// 2nd IASTED International Conference on Computational Intelligence, CI 2006. USA: ACTA Press, 2006: 231-236.

[26] GUERREIRO A P, FONSECA C M, EMMERICH M T. A fast dimension sweep algorithm for the hypervolume indicator in four dimensions[C]// 24th Canadian Conference on Computational Geometry, CCCG 2012. Canada: Canadian Conference on Computational Geometry, 2012: 77-82.

[27] WATANABE T, TATSUKAWA T, OYAMA A. On the fast hypervolume calculation method[C]// 2015 IEEE Congress on Evolutionary Computation (CEC). Japan: IEEE, 2015: 965-969.

[28] RUSSO L M S, FRANCISCO A P. Quick hypervolume[J]. IEEE Transactions on Evolutionary Computation, 2012, 18(4): 481-502.

[29] LACOUR R, KLAMROTH K, FONSECA C M. A box decomposition algorithm to compute the hypervolume indicator[J]. Computers & Operations Research, 2017, 79: 347-360.

[30] JIANG S, ZHANG J, ONG Y S, *et al*. A simple and fast hypervolume indicator-based multiobjective evolutionary algorithm[J]. IEEE Transactions on Cybernetics, 2015, 45(10): 2 202-2 213.

[31] 郑金华, 邹娟. 多目标进化优化[M]. 北京:科学出版社, 2018:175-176.

[32] 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.

[33] DEB K, JAIN H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach: Part I. Solving problems with box constraints[J]. IEEE Transactions on Evolutionary Computation, 2014, 18(4): 577-601.

[34] YE T, RAN C, ZHANG X, *et al*. PlatEMO: A matlab platform for evolutionary multi-objective optimization[Educational Forum][J]. IEEE Computational Intelligence Magazine, 2017, 12(4): 73-87.

[35] BOSMAN P A N, THIERENS D. The balance between proximity and diversity in multiobjective evolutionary algorithms[J]. IEEE Transactions on Evolutionary Computation, 2003, 7(2): 174-188.

[36] 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.

Hypervolume-Based Multi-Objective Evolutionary Algorithm

WANG Xuewu, WEI Jianbin, ZHOU Xin, GU Xingsheng
（Key Laboratory of Advanced Control and Optimization for Chemical Processes, Ministry of Education,
East China University of Science and Technology, Shanghai 200237, China）

Abstract: Hypervolume-based evolutionary algorithms can effectively solve the multi-objective optimization problem and obtain promising solution sets with fast convergence and uniform distribution. However, this kind of algorithms have higher computational complexity and lower programming efficiency. Aiming at the two-dimensional and three-dimensional multi-objective optimization problems, this paper proposes a hypervolume-based multi-objective evolutionary algorithm (MOEA-HV) so that the individuals ' exclusive hypervolume contributions can be accurately calculated to guide the evolution of the whole population. Before the indicator-based evolutionary algorithm(IBEA) being utilized, the proposed algorithm employs non-dominated sorting among all individuals to delete dominated individuals so that the amount of calculation of the individuals ' exclusive hypervolume contributions can be reduced and the operational efficiency can be improved. Meanwhile, other strategies in NSGA-Ⅲ are utilized to optimize the distribution of the proposed algorithm. It is shown via the experiment results that the proposed MOEA-HV has higher efficiency while maintaining the trade-off between the fast convergence and the uniform distribution.

Key words: multi-objective optimization; hypervolume; non-dominated sorting; IBEA