

一种自适应多目标离散差分进化算法

张明明, 赵曙光, 王旭

ZHANG Ming-ming, ZHAO Shu-guang, WANG Xu

东华大学 信息科学与技术学院, 上海 201620

College of Information Science and Technology, Donghua University, Shanghai 201620, China

E-mail: mmzhang@mail.dhu.edu.cn

ZHANG Ming-ming, ZHAO Shu-guang, WANG Xu. Adaptive multi-objective discrete differential evolution. *Computer Engineering and Applications*, 2009, 45(26): 16-20.

Abstract: An adaptive Multi-objective Discrete Differential Evolution (MDDE) is proposed. By introducing differential evolution to multi-objective optimization, MDDE adopts a new adaptive discrete differential evolution strategy to enhance the ability of global exploration so as to achieve better Pareto approximate solutions. Moreover, for keeping good diversity, MDDE integrates fast Pareto sorting strategy and truncating operation based on crowding density and rank. The experimental results show that the proposed approach is able to effectively solve the multi-objective problems with discrete or continuous solution spaces and has better performance on convergence, diversity and stability.

Key words: multi-objective optimization; discrete differential evolution; Pareto optimal; adaptation

摘要: 提出一种自适应多目标离散差分进化算法。该算法将差分进化引入多目标优化领域, 采用一种新的自适应离散差分进化策略增强算法的全局搜索能力, 以获得更优的 Pareto 近似解, 并结合 Pareto 快速分层排序策略和基于聚集密度的按层修剪操作对种群进行更新维护, 使解集保持良好的多样性。实例测试和算法比较表明, 该算法能有效求解离散域和连续域上不同类型的多目标优化问题, 且在收敛性、分布性、稳定性方面均表现较好。

关键词: 多目标优化; 离散差分进化; Pareto 最优; 自适应

DOI: 10.3778/j.issn.1002-8331.2009.26.005 文章编号: 1002-8331(2009)26-0016-05 文献标识码: A 中图分类号: TP18

1 引言

现实世界中的许多决策或控制问题通常是多目标优化问题(MOP), 而大多数 MOP 的各目标之间存在冲突, 即所有目标无法同时取得最优值, 只能找到一组折衷后的 Pareto 非劣解, 这使得传统优化方法在求解 MOP 时存在较大困难。由于进化算法(EAs)天生的隐并行性, 可同时处理大规模的搜索空间, 每轮迭代可找到多个 Pareto 近似最优解, 使得 EAs 特别适合求解 MOP。近年来, 国内外学者提出了多种各具特色的多目标进化算法(MOEAs)^[1-3]。

差分进化算法(Differential Evolution, DE)^[4-5]是新近由 Storn 和 Price 提出的一种基于群体智能的并行全局搜索优化方法。相比于遗传算法和粒子群优化等算法, DE 采用基于差分的简单变异操作和一对一的竞争选择策略, 参数设置较少, 实现简单, 且多数情况下收敛速度更快, 具有较强的全局收敛能力和鲁棒性, 引起了国内外学者的广泛关注, 并取得了可喜的研究成果^[6-8]。但这些成果大多是对各类多维连续空间优化问题研究所取得的, 而对解决离散优化问题的 DE 研究还相对较少^[5,9]。另外, 将差分进化的思想应用于求解 MOP, 在近期才逐渐被研

究者所关注, 相关研究亦主要集中于利用 DE 求解连续域上的 MOP^[10-12]。由于离散多目标优化问题具有广阔的工程应用背景, 例如: 物流管理、网络优化等, 并鉴于 DE 在连续域实值优化上取得的成功, 因此对基于离散差分进化的多目标优化算法进行广泛、深入的研究具有重要的学术意义和工程价值。基于此, 提出了一种自适应多目标离散差分进化算法(MDDE)。通过与 SPEA^[1]和 NSGA-II^[2]进行实验比较, 证明了该算法不但可行且性能更优, 所获得的多个 Pareto 有效解能为决策者的最终决策提供有力保障。

2 多目标优化问题

以最小化问题为例, MOP 可表述如下:

$$\min Y=f(X)=(f_1(X), f_2(X), \dots, f_m(X))$$

$$\text{s.t. } g(X)=(g_1(X), g_2(X), \dots, g_k(X)) \leq 0 \quad (1)$$

其中, 决策向量 $X=(x_1, x_2, \dots, x_n) \in \Omega$, 目标向量 $Y \in A$, $g_i(X)$ 为约束条件, Ω 称为决策空间, A 称为目标空间。对于一个给定的多目标优化问题 $\min f(X)$, 相关定义如下:

基金项目: 国家自然科学基金(the National Natural Science Foundation of China under Grant No.60672026)。

作者简介: 张明明(1981-), 男, 博士研究生, 主要研究方向为多目标优化、可逆逻辑自动综合等; 赵曙光(1965-), 男, 博士, 教授, 博士生导师, 主要研究方向为人工智能、电路进化设计等; 王旭(1983-), 女, 博士研究生, 主要研究方向为进化算法、数据挖掘等。

收稿日期: 2009-06-03

修回日期: 2009-07-09

定义 1 (Pareto 支配) 对任意的决策向量 $X_1, X_2 \in \Omega$, 当且仅当 $f_i(X_1) < f_i(X_2)$, 称 X_1 Pareto 支配 X_2 , 记为 $X_1 > X_2$; 当且仅当 $f_i(X_1) \leq f_i(X_2)$, 称 X_1 Pareto 弱支配 X_2 , 记为 $X_1 \geq X_2$ ($i=1, 2, \dots, m$)。

定义 2 (Pareto 最优解) 对一个决策向量 $X^* \in \Omega$, 若不存在 $X \in \Omega$, 使得 $f_i(X^*) \geq f_i(X)$ ($i=1, 2, \dots, m$), 且其中至少一个不等式严格成立, 则称 X^* 为 Pareto 最优解, 又称为 Pareto 非劣解。

定义 3 (Pareto 最优边界) 给定一个 Pareto 最优解集 $\{X^*\}$, 则称 $PF_{\text{true}} = \{f(X) = (f_1(X), f_2(X), \dots, f_m(X)) | X \in \{X^*\}\}$ 为 Pareto 最优边界。

3 自适应多目标离散差分进化算法

3.1 算法结构

MDDE 利用提出的自适应离散差分进化策略对规模为 NP 的当前种群执行进化操作, 增强算法的全局搜索能力, 生成优良的实验种群, 并采用 Pareto 快速非劣分层排序策略和基于聚集密度的按层修剪操作对种群进行更新维护, 得到分布均匀的新种群。当达到最大进化代数 G_{max} 时, 迭代停止, 输出 Pareto 非劣解集。MDDE 的基本流程如下:

- (1) 初始化: 产生初始种群 P_c , $G=0$;
- (2) 目标评估: 解码 P_c 中个体, 计算子目标函数值;
- (3) 进化操作: 对 P_c 中个体进行自适应离散差分进化操作, 生成实验种群 P_c' ;
- (4) 目标评估: 解码 P_c' 中个体, 计算子目标函数值;
- (5) 分层排序: 对 $P_c \cup P_c'$ 进行 Pareto 快速分层排序;
- (6) 更新选择: 对种群 $P_c \cup P_c'$ 执行基于聚集密度的按层修剪操作, 产生新种群 P_{c+1} ;
- (7) 结束判断: 若 $G > G_{\text{max}}$, 则将 P_{c+1} 中的全部 Pareto 非劣解作为输出, 结束; 否则, $G=G+1$, 转(3)。

3.2 自适应离散差分进化策略

提出的离散 DE 策略采用二进制编码方案, 可对离散域、连续域以及混合优化等多种 MOP 进行灵活有效地编码。同时, 为了保证算法中差分运算的封闭性, 将个体的差定义为两个体归一化的 Hamming 距离, 即:

$$x_1 - x_2 \triangleq \frac{D_h(x_1, x_2)}{D} \quad (2)$$

其中, $D_h(x_1, x_2)$ 为二进制个体 x_1, x_2 的 Hamming 距离, D 为解个体的编码长度。

离散 DE 的执行步骤为: 首先, 对于当前进化群体 P_c 中每一目标个体 $x_{i,c}$ ($i=1, 2, \dots, NP$), 利用种群内个体间的合作, 对目标个体 x_i 的父代差分个体 $x_{i,2}$ 和 $x_{i,3}$ 进行二进制差分操作, 从而得到其父代基个体 $x_{i,1}$ 的变异概率 P_m^i , 即:

$$P_m^i \triangleq F \cdot (x_{i,2} - x_{i,3}) \quad (3)$$

其中, $x_{i,1}, x_{i,2}$ 和 $x_{i,3}$ 是在 P_c 中第 i 次随机选择的个体, 且个体序号 r_1, r_2, r_3, i 互不相同; F 为缩放比例因子, G 为当前进化代数。

其次, 利用 $x_{i,2}$ 和 $x_{i,3}$ 的差分信息对父代基个体 $x_{i,1}$ 进行随机

扰动, 即对 $x_{i,1}$ 执行变异概率为 P_m^i 的均匀变异操作, 产生变异个体 $v_{i,G+1}$ 。

然后, 利用交叉操作, 实现 $x_{i,G}$ 和 $v_{i,G+1}$ 的混合, 生成实验个体 $u_{i,G+1}$, 从而产生实验种群 P_c' , 即:

$$u_{i,G+1}^j = \begin{cases} v_{i,G+1}^j, & \text{if } (\text{rand}(j) \leq CR) \text{ or } j = \text{rnbr}(i) \\ x_{i,G+1}^j, & \text{otherwise} \end{cases} \quad (4)$$

其中, $\text{rand}(j)$ 为 $[0, 1]$ 之间随机数发生器产生的第 j 个估计值; CR 为范围在 $[0, 1]$ 之间的交叉算子; $\text{rnbr}(i)$ 为一个在 $[1, 2, \dots, D]$ 中取值的随机选择序列, 以确保实验个体 $u_{i,G+1}$ 至少有一个分量由变异个体 $v_{i,G+1}$ 的相应分量贡献。

另外, 相关研究表明, 取值固定的缩放比例因子和交叉算子容易造成早熟收敛, 以及鲁棒性较差等问题^[6-8]。因此, 采用自适应调整策略, 令 F 和 CR 跟随进化过程缓慢而单调地衰减。在迭代初期, F 和 CR 取较大值以便能够造成足够的扰动, 增强 DE 的搜索能力; 而在迭代后期, F 和 CR 取较小值以避免破坏优良个体, 从而加快收敛速度, 即:

$$F = F_0 \cdot e^{-a_1 \cdot (\frac{G}{G_{\text{max}}})} \quad (5)$$

$$CR = CR_0 \cdot e^{-a_2 \cdot (\frac{G}{G_{\text{max}}})} \quad (6)$$

其中, F_0 为变异算子的初值, CR_0 为交叉算子初值, G 为当前进化代数, G_{max} 为最大进化代数, a_1, a_2 为正的常数。

3.3 Pareto 快速分层排序

MDDE 采用类似 NSGA-II^[9] 的快速非劣分层排序策略, 根据个体间的 Pareto 支配关系对 $P_c \cup P_c'$ 中的个体进行分层排序, 从而构造互不相交的分层边界集 PF_r , 其中 r 为非支配层次序号, 算法的伪代码如下:

Pareto 快速分层排序算法:

- 1: Begin
- 2: $P = P_c \cup P_c'$; /* 合并当前种群 P_c 与实验种群 P_c' */
- 3: $r=1$; /* 边界集层号初始化 */
- 4: $PF_r = \Phi$; /* 当前边界集清空 */
- 5: while ($P \neq \Phi$) do /* 对 P 划分 */
- 6: for ($x \in P$) /* 将 P 中的个体依次加入 PF_r */
- 7: $PF_r = PF_r \cup \{x\}$;
- 8: for ($y \in PF_r$ and $y \neq x$) /* PF_r 中个体分别与 x 进行比较, 并从 PF_r 中删除被支配者 */
- 9: if ($x > y$), $PF_r = PF_r \setminus \{y\}$;
- 10: else if ($y > x$), $PF_r = PF_r \setminus \{x\}$;
- 11: end for
- 12: end for
- 13: $P = P \setminus PF_r$; /* 从 P 中删除 PF_r 包含的个体 */
- 14: $r=r+1$; /* 构造下一层边界集 */
- 15: end while
- 16: End

3.4 种群的更新维护

为了保持解群体的多样性, 使 Pareto 候选解集 P_{c+1} 中的个体能分布均匀地代表整个 Pareto 最优边界, MDDE 对 P_{c+1} 进行如下的更新与维护:

- (1) 初始化新一代种群 $P_{c+1} = \Phi$;
- (2) 依次将 PF_r ($r=1, 2, \dots, t$) 保存到 P_{c+1} 中, 直到满足

$$\left| \bigcup_{r=1}^t PF_r \right| < NP \text{ 且 } \left| \bigcup_{r=1}^{t+1} PF_r \right| \geq NP;$$

(3) 对第 $t+1$ 层 PF_{t+1} 执行基于个体聚集密度的修剪操作,

$$\text{使其满足 } |PF_{t+1}| = NP - \left| \bigcup_{r=1}^t PF_r \right|;$$

(4) 将修剪后的 PF_{t+1} 并入 P_{G+1} 中, 更新完毕。

修剪操作按层进行, 仅当需要时才激活, 从而提高了算法的效率。在修剪时, 首先计算个体 $x \in PF_{t+1}$ 的聚集密度 $\delta(x) = 1/(d_x^1 + d_x^2)$, 其中 d_x^1 和 d_x^2 分别是个体 x 与距离其最近的两个个体的欧式距离。然后, 判断 δ 最大的个体是否为边界点, 若不是边界点, 则修剪掉该个体; 否则, 判断 δ 次大的个体是否为边界点, 直至修剪掉一个个体。当集合中所剩个体都为边界点时, 则删除 δ 最大的边界点。最后, 重新计算当前集合中个体的聚集密度, 以备再次修剪。反复执行此过程, 直至满足群体规模的约定值。

4 实例仿真与性能比较

4.1 测试问题

连续域测试实例选自文献[3], 它们都是一些难以求解的问题, 具体形式由表 1 给出。

离散域测试实例选用多目标 0/1 背包问题(MOKP)。MOKP 是典型的 NP 难问题, 可表述为: 已知 m 种物品和 n 个容量分别为 $c_i (i=1, 2, \dots, n)$ 的背包, 物品 j 相对于背包 i 的重量为 w_{ij} , 效益值为 p_{ij} , 确定一种最优的物品装入方案, 使装入物品的总效益值最大化, 且装入包中的物品总重量不超过包的容量限制。其数学模型为:

$$\max f(X) = (f_1(X), f_2(X), \dots, f_n(X))$$

$$\text{s.t. } \sum_{j=1}^m w_{ij} \cdot x_j \leq c_i$$

其中: $f_i(X) = \sum_{j=1}^m p_{ij} \cdot x_j, i \in \{1, 2, \dots, n\}, X = (x_1, x_2, \dots, x_m) \in \{0, 1\}^m$ 。

考虑背包数 n 分别为 2, 3 和 4, 物品数 m 分别为 500 和

750 的 6 个 MOKP, 记为 2-500, 2-750 和 3-500 等。

4.2 性能评估准则

采用如下三个性能评估准则^[4,3]:

$$(1) GD = \frac{1}{n} \sqrt{\sum_{i=1}^N D_i^2}$$

其中, D_i 为在目标空间上第 i 个解到 Pareto 最优边界的最短欧式距离, N 为算法输出的非支配解的个数, n 是优化问题的子目标数。GD 定义了所得最终解集与 Pareto 最优边界的趋近程度, 指标越小, 说明趋近程度越高。

$$(2) SP = \sqrt{\frac{1}{N-1} \cdot \sum_{i=1}^N (\bar{d}_i - d_i)^2}$$

其中, $d_i = \min_{j=1 \wedge j \neq i}^N |f_k^i(x) - f_k^j(x)|, \bar{d}_i$ 是 d_i 的平均值。SP 定义了所得最终解集在目标空间上的均匀分布程度, 指标越小, 说明分布越均匀。

$$(3) C(A, B) = \frac{| \{b \in B; \exists a \in A: a \geq b\} |}{|B|}$$

定义了最终解集之间的相对 Pareto 支配关系。指标越大, 说明解集 A 支配 B 中解的数量越多, A 的质量越好。一般, $C(A, B) \neq C(B, A)$, 故要分别考虑。

4.3 实验环境与参数设置

SPEA、NSGA-II 和 MDDE 均采用二进制编码方式, 其中 SPEA 的数据采用文献[1]和文献[3]的实验结果, 选自 Web^[13]; NSGA-II 的源代码选自 Deb 的主页^[14]; MDDE 采用 Microsoft Visual C++ 6.0 环境下编程实现。在连续域测试实例中, NSGA-II 的参数设置同文献[2]; MDDE 的进化参数按式(2), (3), (5)和(6)计算, 令初值 $F_0 = CR_0 = 0.4, a_1 = a_2 = 2$ 。在离散域测试实例中, MOKP 的系数 w_{ij}, p_{ij} 初始值和各背包的容量 c_i 同文献[1], 取自 Web^[13]。各算法对不可行解的约束处理方法同文献[1], 即采用贪心修复机制。另外, 根据文献[1], NSGA-II 的交叉概率为 0.8, 变异概率为 0.01; MDDE 的进化参数设置同连续域测试实例。各算法的其他参数设置, 如表 2 所示。

表 1 连续域测试实例

实例	变量维数 m	变量取值范围	目标函数	Pareto 最优解	最优边界特性
F_1	30	$x_i \in [0, 1], i=1, 2, \dots, m$	$\min f_1(x) = x_1$	$x_1 \in [0, 1]$ $x_i = 0, i=2, \dots, m$	凸
			$\min f_2(x) = g(x)(1 - \sqrt{x_1/g(x)})$		
			$g(x) = 1 + 9 \cdot (\sum_{i=2}^m x_i) / (m-1)$		
F_2	30	$x_i \in [0, 1], i=1, 2, \dots, m$	$\min f_1(x) = x_1$	$x_1 \in [0, 1]$ $x_i = 0, i=2, \dots, m$	非凸
			$\min f_2(x) = g(x)(1 - (x_1/g(x))^2)$		
			$g(x) = 1 + 9 \cdot (\sum_{i=2}^m x_i) / (m-1)$		
F_3	30	$x_i \in [0, 1], i=1, 2, \dots, m$	$\min f_1(x) = x_1$	$x_1 \in [0, 1]$ $x_i = 0, i=2, \dots, m$	凸, 不连续
			$\min f_2(x) = g(x)(1 - \sqrt{x_1/g(x)} - (x_1/g(x)) \cdot \sin(10\pi x_1))$		
			$g(x) = 1 + 9 \cdot (\sum_{i=2}^m x_i) / (m-1)$		
F_4	10	$x_1 \in [0, 1]$ $x_i \in [-5, 5], i=2, \dots, m$	$\min f_1(x) = x_1$	$x_1 \in [0, 1]$ $x_i = 0, i=2, \dots, m$	非凸, 局优多
			$\min f_2(x) = g(x)(1 - \sqrt{x_1/g(x)})$		
			$g(x) = 1 + 10(m-1) + \sum_{i=2}^m (x_i^2 - 10\cos(4\pi x_i))$		

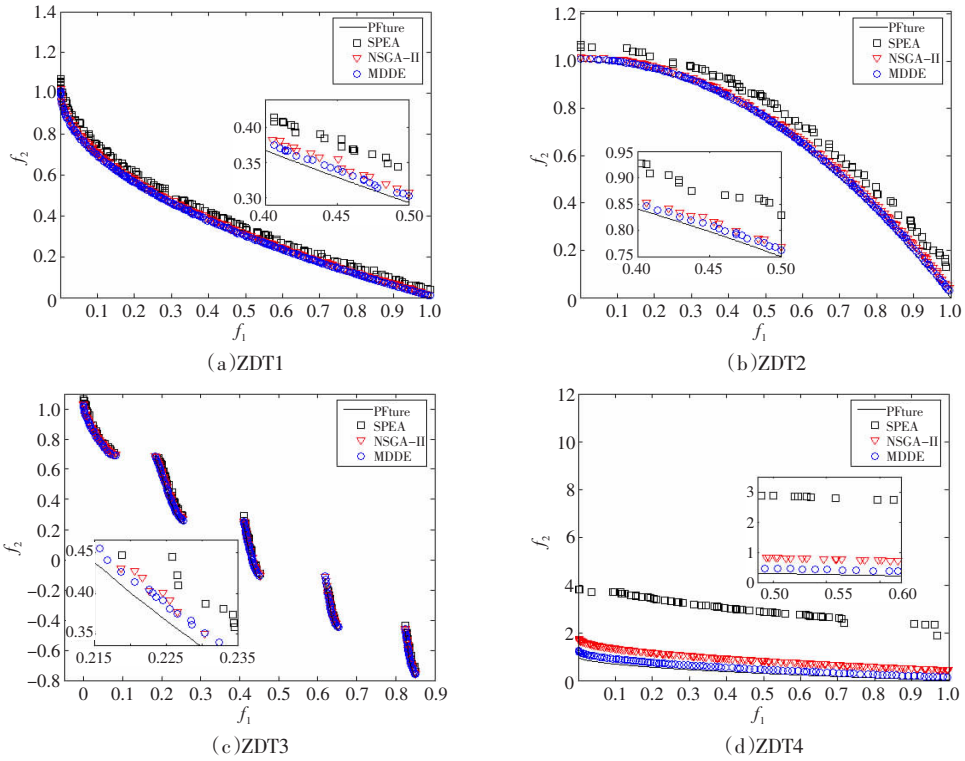


图1 三种算法求解 $F_1 \sim F_4$ 所得非劣解集的比较

表2 三种算法的实验参数设置

实例	目标数/ 背包数 n	变量维数/ 物品数量 m	编码 长度	种群 规模	最大进 化代数
F_1	2	30	900	100	250
F_2	2	30	900	100	250
F_3	2	30	900	100	250
F_4	2	10	300	100	250
MOKP	2	500	500	200	500
	2	750	750	250	500
	3	500	500	250	500
	3	750	750	300	500
	4	500	500	300	500
	4	750	750	350	500

4.4 结果分析

为了减少实验的随机性,各算法对每一测试实例均分别独立运行 30 次。

在求解连续域测试实例时,每种算法分别针对 $F_1 \sim F_4$ 计算各最终解集的 GD 和 SP , 求得 30 次运行的平均值和标准方差,结果见表 3,其中各实例后的第一行为平均值,第二行为标准方差。由表 3 可见,在各测试实例中,MDDE 的 GD 和 SP 一

表3 三种算法求解连续域优化问题的性能比较

实例	GD			SP		
	SPEA	NSGA-II	MDDE	SPEA	NSGA-II	MDDE
F_1	9.01E-03	1.37E-03	8.51E-04	1.33E-02	8.59E-03	4.49E-03
	8.37E-03	3.40E-04	1.16E-04	1.89E-03	8.73E-04	6.93E-04
F_2	1.08E-02	2.00E-03	9.99E-04	2.54E-02	9.02E-03	4.10E-03
	3.19E-03	4.34E-04	1.70E-04	6.59E-03	1.54E-03	5.41E-04
F_3	3.07E-03	9.91E-04	7.69E-04	1.83E-02	1.20E-02	6.29E-03
	1.41E-03	7.19E-04	1.79E-04	1.00E-02	7.11E-03	1.26E-03
F_4	5.70E-01	6.59E-02	4.84E-02	4.81E-02	1.53E-02	4.19E-03
	3.22E-01	1.84E-02	2.16E-02	8.00E-02	1.41E-02	8.79E-04

致优于其他 2 种算法,充分表明了MDDE 所得的最终解集更加逼近真实的 Pareto 最优边界,且在目标空间分布更加均匀,表现出了良好的收敛性和分布性。

另外,三种算法求解 $F_1 \sim F_4$ 的前 5 次运行后所得非劣解集在目标空间的分布情况如图 1 所示。可见,MDDE 所得的 Pareto 边界位于其他方法的左下方,更加趋近真实的 Pareto 最优边界,且解的数量更多,分布更均匀。

对于每一 MOKP 测试实例,针对各算法所得的 30 个最终解集,分别计算对应解集的相对性能指标 C ,并用数据统计 box plot 图对结果进行可视化描述,如图 2 所示。由图 2(a),(c)可见,各测试实例的全部 C (SPEA,MDDE)和 C (NSGA-II,MDDE)均为 0,表明 SPEA 和 NSGA-II 所得最终解集均不存在 Pareto

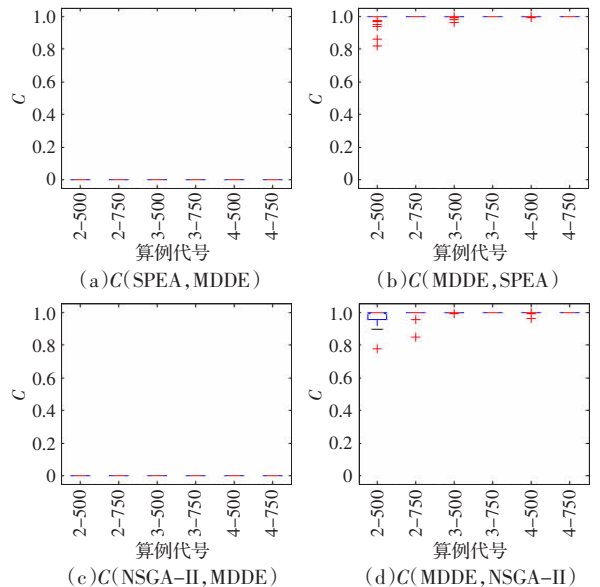


图2 三种算法求解 MOKP 的 C 指标比较

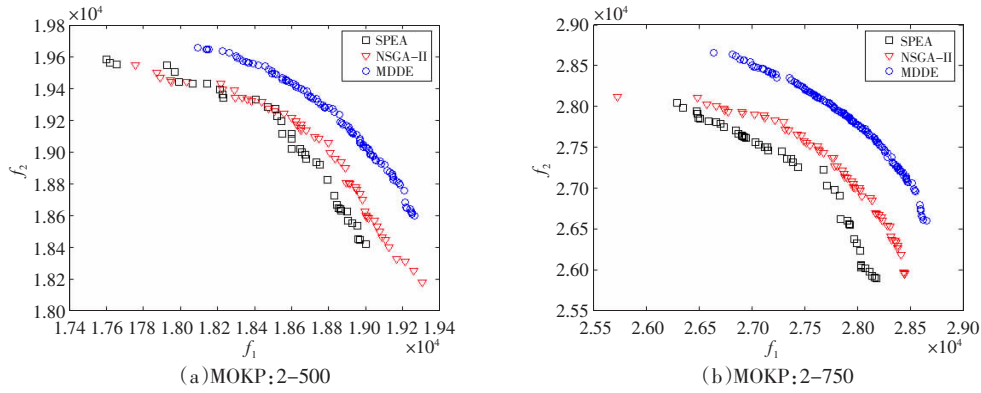


图3 三种算法求解2背包MOKP所得非劣解集的比较

优于 MDDE 结果的解。同时,由图 2(b), (d)可见,各测试实例 30 次运行结果数据统计的中位线均接近 1,四分位距除 2-500 外均约为 0,说明 MDDE 搜索到的最终解集均绝对支配于 SPEA 和 NSGA-II 的相应结果,表明 MDDE 能够搜索到更加逼近 Pareto 最优边界的非劣面,性能显著优于 SPEA 和 NSGA-II。另外,各算例独立运行的各次计算结果保持一致性,也体现了 MDDE 极佳的稳定性。

同时,表 4 比较了各算法独立运行 30 次求解各 MOKP 所得 SP 的平均值和标准方差。由表 4 可见,MDDE 的性能指标均较好,说明了其搜索到的最终解集分布性更好。另外,图 3 画出了三种算法分别求解实例 2-500 和 2-750 的前 5 次运行后所得非劣解集在目标空间的分布情况。可见,MDDE 所得的 Pareto 非劣解集更加均匀地逼近真实的 Pareto 最优边界,体现了 MDDE 良好的收敛性。

表4 三种算法求解MOKP的SP性能比较

MOKP	SPEA	NSGA-II	MDDE
2-500	3.56E+01	2.70E+01	1.29E+01
	1.24E+01	7.42E+00	3.14E+00
2-750	5.17E+01	4.08E+01	1.34E+01
	2.02E+01	7.34E+00	2.83E+00
3-500	5.18E+01	4.11E+01	3.86E+01
	6.24E+00	5.82E+00	3.22E+00
3-750	6.48E+01	5.66E+01	3.96E+01
	9.32E+00	8.03E+00	3.20E+00
4-500	8.63E+01	4.79E+01	4.43E+01
	4.53E+00	4.12E+00	4.35E+00
4-750	1.07E+02	6.15E+01	6.08E+01
	8.53E+00	7.17E+00	4.68E+00

5 结语

基于标准差分进化算法,提出了一种新的自适应离散差分进化策略,并与 Pareto 快速分层排序策略和基于聚集密度的按层修剪操作相结合,进而提出了一种自适应多目标离散差分进化算法。连续域和离散域上的实验结果表明,该算法能搜索到更趋近于 Pareto 最优边界且分布更均匀的近似集,其表现稳

定、性能较好。

参考文献:

- [1] Zitzler E, Thiele L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach[J]. IEEE Trans Evol Comput, 1999, 3(4): 257-271.
- [2] Deb K, Pratap A, Agarwal S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II[J]. IEEE Trans Evol Comput, 2002, 6(2): 182-197.
- [3] Zitzler E, Deb K, Thiele L. Comparison of multiobjective evolutionary algorithms: Empirical results[J]. Evol Comput, 2000, 8(2): 173-195.
- [4] Storn R, Price K. Differential evolution: A simple and efficient heuristic for global optimization over continuous spaces[J]. J Global Optim, 1997, 11(4): 341-359.
- [5] Price K, Storn R, Lampinen J. Differential evolution: A practical approach to global optimization[M]. London: Springer-Verlag, 2005.
- [6] Noman N, Iba H. Accelerating differential evolution using an adaptive local search[J]. IEEE Trans Evol Comput, 2008, 12(1): 107-125.
- [7] 谢晓锋, 张文俊, 张国瑞, 等. 差异演化的实验研究[J]. 控制与决策, 2004, 19(1): 49-52.
- [8] Rahnamayan S, Tizhoosh H R, Salama M M A. Opposition-based differential evolution[J]. IEEE Trans Evol Comput, 2008, 12(1): 64-79.
- [9] 刘波, 王凌, 金以慧. 差分进化算法研究进展[J]. 控制与决策, 2007, 22(7): 721-729.
- [10] Abbass H A, Sarker R. The Pareto differential evolution algorithm[J]. Int J Artif Intel Tools, 2002, 11(4): 531-552.
- [11] Zamuda A, Brest J, Boskovic B, et al. Differential evolution for multiobjective optimization with self adaptation[C]// Proc 2007 IEEE Congr Evol Comput. Singapore: IEEE Press, 2007: 3617-3624.
- [12] Alatas B, Akin E, Karci A, Modenar. Multi-objective differential evolution algorithm for mining numeric association rules[J]. Appl Soft Comput, 2008, 8(1): 646-656.
- [13] Zitzler E. SPEA[EB/OL]. (2008-08-01). <http://www.tik.ee.ethz.ch/~zitzler/testdata.html>.
- [14] Deb K. NSGA-II [EB/OL]. (2008-08-01). <http://www.iitk.ac.in/kangal/deb.shtml>.