ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.20, No.2, February 2009, pp.305−314
doi: 10.3724/SP.J.1001.2009.03183

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种基于正态分布交叉的ε-MOEA*

张 敏+, 罗文坚, 王煦法

(中国科学技术大学 计算机科学技术系,安徽 合肥 230027)

A Normal Distribution Crossover for ε-MOEA

ZHANG Min+, LUO Wen-Jian, WANG Xu-Fa

(Department of Computer Science and Technology, University of Science and Technology of China, Hefei 230027, China)

+ Corresponding author: E-mail: xfwang@ustc.edu.cn

Zhang M, Luo WJ, Wang XF. A normal distribution crossover for ε-MOEA. *Journal of Software*, 2009,20(2): 305−314. http://www.jos.org.cn/1000-9825/3183.htm

Abstract: The simulated binary crossover (SBX) has been extensively adopted in the real-coded multiobjective evolutionary algorithms (MOEAs). Through the comparisons and analyses of the SBX and the mutation operator in the evolution strategy (ES), this paper proposes a normal distribution crossover (NDX) with the introduction of discrete recombination operator in ES. The NDX and SBX operators are compared and analyzed through an example designed in the one dimensional search space, and then the NDX is applied to a steady-state multiobjective evolutionary algorithm named ε-MOEA (ε-dominance based multiobjective evolutionary algorithm) proposed by Deb, *et al*. The algorithm ε-MOEA with NDX (ε-MOEA/NDX) has been tested and compared on the 10 benchmark functions taken from the ZDT and DTLZ standard test suites. Experimental results demonstrate that algorithm ε-MOEA/NDX is distinctly superior to the ε-MOEA/SBX and NSGA-II algorithms, which are representatives of the state-of-the-art in the area.

Key words: evolutionary multiobjective optimization; ε-MOEA (ε-dominance based multiobjective evolutionary algorithm); normal distribution crossover (NDX); simulated binary crossover (SBX)

摘 要: 实数编码的多目标进化算法常使用模拟二进制交叉(simulated binary crossover,简称 SBX)算子.通过对SBX以及进化策略中变异算子进行对比分析,并引入进化策略中的离散重组算子,提出了一种正态分布交叉(normal distribution crossover,简称 NDX)算子.首先在一维搜索空间实例中对 NDX 与 SBX 算子进行比较和分析,然后将NDX 算子应用于 Deb 等人提出的稳态多目标进化算法ε-MOEA(ε-dominance based multiobjective evolutionary algorithm)中.采用 NDX 算子的ε-MOEA(记为ε-MOEA/NDX)算法在多目标优化标准测试集 ZDT 和 DTLZ 的 10 个函数上进行了实验比较.实验结果和分析表明,采用 NDX 的ε-MOEA 所求得的 Pareto 最优解集质量明显优于经典算法ε-MOEA/SBX 和 NSGA-II.

关键词: 进化多目标优化;ε-MOEA(ε-dominance based multiobjective evolutionary algorithm);正态分布交叉;模拟二进制交叉

* Supported by the Overseas Young Scholars Collaborative Research Grant of the National Natural Science Foundation of China under Grant No.60428202 (国家自然科学基金委海外青年学者合作研究基金)

Received 2007-12-26; Accepted 2008-09-30

中图法分类号: TP18 文献标识码: A

现实世界中的优化问题往往具有多个目标.由于多目标优化问题自身的复杂性,运筹学中的传统方法已难以独立解决.进化算法作为一种群体搜索方法,能够在一次求解中获得一组解集,非常适合求解多目标优化问题.早在 1967 年,Rosenberg 就在其博士论文中提出可利用遗传算法求解多目标优化问题[1].但直到 1985 年才出现了基于向量评估的 VEGA 算法[1].虽然 VEGA 存在不足,难以找到处于前沿中间位置的最优解,但自 Goldberg 于 1994 年引入运筹学中的 Pareto 支配关系[2]和 Zitzler 于 1999 年提出精英策略[3]以来,利用进化算法求解多目标优化问题、基于 Pareto 支配关系的多目标进化算法(MOEAs)已越来越受到关注.

进化多目标优化的目标是使群体快速逼近并均匀分布于真实 Pareto 前沿,即快速性、分布性和逼近性 3 个目标[4].根据 Coello Coello 的观点,目前已有进化多目标优化(EMO)的算法研究可分为两个阶段[5]:第 1 阶段的特点是利用 Pareto 支配关系设计简单的适应度函数,代表性算法有 VEGA[1],NSGA[6],NPGA[2]和 MOGA[7];第 2 阶段的特点是利用精英群体维持当前 Pareto 最优解的分布性,代表性算法有 SPEA[3],SPEA2[8],PAES[9]和 NSGA-II[10].但这些算法都难以保证既能找到分布好的 Pareto 前沿,又能拥有快速的计算速度[4].正如文献[5]所指出的,ε-支配概念[11]是当前多目标进化算法的主要研究趋势之一,而 Deb 等人利用该支配概念提出的稳态多目标进化算法ε-MOEA(ε-dominance based multiobjective evolutionary algorithm)[4]是目前能够较好地满足快速性和分布性这两个目标的算法.

多目标优化的第 3 个目标是解集对真实 Pareto 前沿的逼近性[4].如何达到该目标,不仅与算法采用的选择策略相关,也与其遗传操作算子相关.对此,本文通过对比分析模拟二进制交叉(simulated binary crossover,简称 SBX)算子[12]和进化策略[13]中的变异及重组算子,提出了一种基于正态分布的实数交叉(normal distribution crossover,简称 NDX)算子.文中首先在一维搜索空间中对算子 SBX 和 NDX 进行了比较和讨论,然后将 NDX 算子应用于ε-MOEA 中,与经典算法进行了实验比较和分析.

本文第 1 节给出多目标优化中的基本概念及ε-MOEA[4]中ε-Pareto 支配关系和 SBX 算子的定义.第 2 节提出正态分布交叉(NDX)算子.第 3 节将算子 NDX 应用于ε-MOEA[4]中,并给出算法ε-MOEA/NDX 的步骤和时间消耗.第 4 节在标准测试集 ZDT 和 DTLZ 中的 10 个函数上,与经典算法ε-MOEA/SBX[4]和 NSGA-II[10]进行实验比较和分析.第 5 节对本文进行总结.

1 预备知识

1.1 问题描述

定义 1(多目标优化问题). 一个多目标优化问题一般可以表示为(以最小化为例)

$$\min f(\boldsymbol{x}) = \{f_1(\boldsymbol{x}), f_2(\boldsymbol{x}),..., f_m(\boldsymbol{x})\} \tag{1}$$

其中, $\boldsymbol{x} = (x_1, x_2,..., x_n)$ 表示 n 维决策空间 $\mathbf{X}$ 中的一个解向量, $f_k(\boldsymbol{x})(1 \le k \le m)$ 为第 k 个目标函数.这里没有考虑约束条件,有关多目标约束处理参见文献[10,14,15].

定义 2(Pareto 支配关系). 对决策空间 $\mathbf{X}$ 中的任意两个决策向量 $\boldsymbol{a}$ 和 $\boldsymbol{b}$,$\boldsymbol{a}$ 支配 $\boldsymbol{b}$ 即 $\boldsymbol{a} \prec \boldsymbol{b}$ 当且仅当

$$\forall 1 \le i \le m, f(\boldsymbol{a}) \le f(\boldsymbol{b}), \exists 1 \le j \le m, f(\boldsymbol{a}) < f(\boldsymbol{b}) \tag{2}$$

定义 3(Pareto 最优). 对决策空间 $\mathbf{X}$ 中的任一向量 $\boldsymbol{a}$,$\boldsymbol{a}$ 为 Pareto 最优,当且仅当

$$\nexists \boldsymbol{a}' \in \mathbf{X}, \boldsymbol{a}' \prec \boldsymbol{a} \tag{3}$$

其中,$\boldsymbol{a}$ 也可称为在 $\mathbf{X}$ 中是非支配的(nondominated).$\mathbf{X}$ 中的非支配集合 P 称为 Pareto 最优解集(Pareto optimal set),相应的集合 $f(P)$称为 Pareto 最优前沿(Pareto optimal front).

1.2 Deb等人的ε-Pareto支配

Laumanns 等人在文献[11]中提出了ε-Pareto 支配的概念,并对其进行了全面的分析.Deb 等人利用该概念,提出了一种基于空间划分的ε-Pareto 支配关系[4],定义如下:

定义 4(Debs' ε-Pareto 支配"$\prec_\varepsilon$"). 对决策空间 **X** 中的任意两个决策变量 $\boldsymbol{a}$ 和 $\boldsymbol{b}$,定义如下:

$$\boldsymbol{a} \prec_\varepsilon \boldsymbol{b} \Leftrightarrow \forall 1 \le i \le m,\ B_i(\boldsymbol{a}) \le B_i(\boldsymbol{b}),\ \exists 1 \le j \le m,\ B_j(\boldsymbol{a}) < B_j(\boldsymbol{b}) \tag{4}$$

其中,向量 $B(\boldsymbol{a})$表示决策变量 $\boldsymbol{a}$ 在目标空间中的格子位置,定义为

$$B_j(\boldsymbol{a}) = \left\lfloor \left(f_j(\boldsymbol{a}) - f_j^{\min} \right) / \varepsilon_j \right\rfloor \tag{5}$$

其中,$f_j^{\min}$ 为第 j 个目标可能的最小值,ε_j 为第 j 个目标值可接受的偏差,都需要预先设定.可以看出,式(5)将目标空间划分为若干个格子,通过目标向量所处格子间的支配关系定义决策向量间的支配关系.该支配关系在可接受的偏差范围内降低了目标空间中非支配个体的数目,从而可以获得更好的分布性.

1.3 模拟二进制交叉(SBX)

目前在实数编码的多目标进化算法中,最常用的交叉算子为 Deb 等人提出的模拟二进制交叉(SBX)[12],其定义为:对两个父个体 x_1 与 x_2,按照以下方式生成两个子个体 c_1 和 c_2:

$$\begin{cases} c_{1,i} = \left[(1+\beta)x_{1,i} + (1-\beta)x_{2,i} \right] / 2 \\ c_{2,i} = \left[(1-\beta)x_{1,i} + (1+\beta)x_{2,i} \right] / 2 \end{cases},\ 1 \le i \le n \tag{6}$$

其中,β为随机变量,在每一维上都需要重新生成,方式如下:

$$\beta = \begin{cases} (2u)^{\frac{1}{\eta+1}}, & u \le 0.5 \\ \left(2(1-u)\right)^{-\frac{1}{\eta+1}}, & u > 0.5 \end{cases} \tag{7}$$

在式(7)中,u 是均匀分布于区间(0,1)上的随机数;η是交叉参数,为一常数.

2 正态分布交叉(NDX)算子

2.1 正态分布的引入

在给出正态分布交叉算子之前,首先观察进化策略[13]中的个体生产方式,如下式:

$$x' = x + \sigma \cdot N(0,1) \tag{8}$$

其中,x 表示当前个体,x'表示新生成的个体,σ表示搜索步长,$N(0,1)$表示正态分布随机变量.

再观察式(6),以第 i 维为例(不妨设 $x_{1,i} \ge x_{2,i}$),可以将其变换为如下形式:

$$c_{1/2,i} = (x_{1,i} + x_{2,i})/2 \pm \beta(x_{1,i} - x_{2,i})/2 \tag{9}$$

与式(8)相比,不难看出,$(x_{1,i} + x_{2,i})/2$ 表示两个父个体在第 i 维上的中点,若将 $(x_{1,i} - x_{2,i})/2$ 视为在该维上的搜索步长,则式(8)与式(9)十分相似.因此,可以考虑将正态分布引入到交叉操作 SBX 中,即将正态分布引入到式(9)中,如下式:

$$c_{1/2,i} = (x_{1,i} + x_{2,i})/2 \pm A \cdot (x_{1,i} - x_{2,i})/2 \cdot \left| N(0,1) \right| \tag{10}$$

其中,参数 A 表示搜索步长与 $(x_{1,i} - x_{2,i})/2$ 之间的比率系数.本质上,所得的式(10)是用 $A \cdot \left| N(0,1) \right|$ 代替了式(6)中的参数β.下面将对此进行分析.

Herrera 等人在文献[16]中对实数交叉算子进行了总结和分析,并对交叉算子的探索(exploration)和开发(exploitation)能力进行了分析,其定义如图 1 所示.这里采用该方法对式(6)和式(10)进行对比分析.考虑搜索空间是一维的情形.

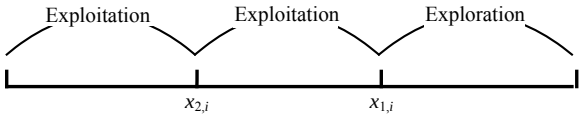

Fig.1 Definition of exploration and exploitation intervals[16]

图 1 探索(exploration)与开发(exploitation)区间的定义[16]

对 SBX 算子而言,由式(7)可得参数 $0\le\beta\le1$,$\beta>1$ 的概率均为 0.5.根据式(6),当 $0\le\beta\le1$ 时,生成的两个子个体位于两个父个体之间;而当 $\beta>1$ 时,两个子个体则在两个父个体之外.因此,算子 SBX 对一维搜索空间开发和探索的概率都是 0.5.对式(10)而言,根据图 1 中的定义,当 $A\cdot|N(0,1)|\le1$ 时,生成的子个体位于父个体的区间内,是对搜索空间的开发;否则,是对搜索空间的探索.设在式(10)中对空间开发和探索的概率分别为 P_1 和 P_2,则可得

$$P_1=\int_{-1/A}^{1/A}\frac{1}{\sqrt{2\pi}}\mathrm{e}^{-x^2/2}\mathrm{d}x,\ P_2=1-P_1 \tag{11}$$

可以看出,不同的 A 对应不同的开发和探索概率.

显然,对式(10)而言,可以设定其开发和探索的概率与 SBX 完全一致,即维持开发和探索的概率都是 0.5.此时,由式(11)中 P_1=0.5 可以计算出 A=1.481,代入式(10):

$$c_{1/2,i}=(x_{1,i}+x_{2,i})/2\pm1.481\cdot(x_{1,i}-x_{2,i})/2\cdot\left|N(0,1)\right| \tag{12}$$

为便于理解,下面在一维搜索空间中对 SBX 和式(12)进行比较.给定两个父个体在空间中的位置,对式(6)和式(12)独立计算 10 000 次,即分别产生 20 000 个子个体,统计其在搜索空间的分布情况,具体如图 2 和图 3 所示,其中 x_1=0.7,x_2=0.2,η=15,横轴 x 表示子个体在搜索空间中的取值范围,纵轴 frequency 表示在给定区间上的出现次数.从图中不难看出,式(12)搜索到的空间比 SBX 算子更为广阔,自然更容易跳出局部最优,对多目标优化而言,也更容易得到完整的 Pareto 解集.有必要指出,式(12)进行开发搜索的概率为 0.5,因此求解精度并不会受到影响,这些都将在实验中得到进一步的验证.

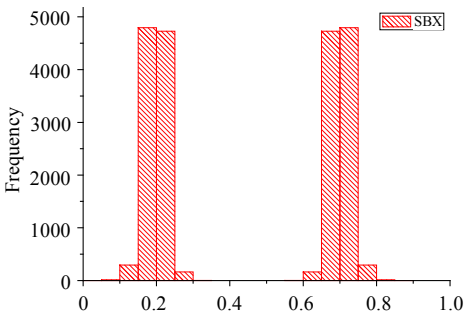

Fig.2 SBX: Space distribution in one dimension

图 2 SBX 算子:一维搜索空间分布

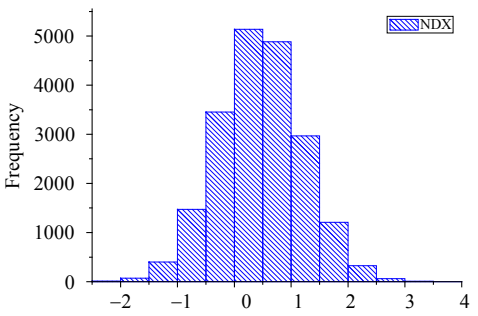

Fig.3 Equation (12): Space distribution in one dimension

图 3 式(12):一维搜索空间分布

2.2 重组操作的引入

对于一维以上的搜索空间,将进化策略[13]中的离散重组(discrete recombination)操作引入到式(12)中,可以进一步增强空间搜索能力,从而建立一种正态分布交叉(NDX)算子,描述如下:

$$\begin{cases}c_{1/2,i}=(x_{1,i}+x_{2,i})/2\pm1.481\cdot(x_{1,i}-x_{2,i})/2\cdot\left|N(0,1)\right|, & u\le0.5\\ c_{1/2,i}=(x_{1,i}+x_{2,i})/2\mp1.481\cdot(x_{1,i}-x_{2,i})/2\cdot\left|N(0,1)\right|, & u>0.5\end{cases} \tag{13}$$

其中,u 与式(7)中的相同,为区间(0,1)上均匀分布的随机数.

由式(13)中可以看出,采用离散重组操作后,每个子个体在每一维中都具有 2 个等概率的取值,则对 n 维的搜索空间而言,每个子个体的取值共有 2^n 种,且每种取值概率均为 $1/2^n$.若不采用该重组操作,则每个子个体的取值只有 1 种,且包含在前面的 2^n 种之中.因此,采用离散重组操作进一步增强了 NDX 算子的空间搜索能力.

至此,式(13)已完整地描述了本文所设计的正态分布交叉(NDX)算子.一方面,从交叉算子的探测与开发角度考虑,NDX 算子仍具有与 SBX 算子相当的开发能力;另一方面,从跳出局部最优的角度考虑,算法 ε-MOEA/NDX 由于采用正态分布交叉(NDX)算子,搜索到的空间更为广阔.

2.3 与多体交叉算子UNDX的比较

在实数编码的交叉算子中,虽然 UNDX[17]及 UNDX-m[17]算子也采用正态分布,但却是一种多体交叉算子,

至少需要 3 个父个体参与,在第 3 节的算法 1 中将看到这类算子难以直接应用于两群体同时进化的ε-MOEA[4]中,而本文建立的二元交叉算子 NDX 却可以直接应用.与此同时,在 UNDX 和 UNDX-m 算子中,其算子参数需要事先设定,则求解问题的通用性将受到限制,而 NDX 算子则不存在此问题,可以方便地应用于各种问题的求解中.此外,需要指出的是,由于算子设计出发点不同,从 UNDX 及 UNDX-m 算子无法直接得出 NDX 算子.

3 基于NDX的ε-MOEA(ε-MOEA/NDX)

Deb 等人提出的ε-MOEA[4]拥有进化群体和精英群体,且两个群体都参与进化,将第 2 节中建立的 NDX 算子应用于该算法中,记为ε-MOEA/NDX.为便于分析算法的时间消耗,给出ε-MOEA/NDX 的具体步骤,见算法 1.

算法 1. 基于 NDX 的ε-MOEA(ε-MOEA/NDX).

1. 随机生成初始群体 $P(0)$,并将其中的非支配个体复制到精英群体 $E(0)$中,置计数器 $t=0$
2. 利用二元联赛法从进化群体 $P(t)$中选择出一个父个体 p
3. 从精英群体 $E(t)$中随机选择一个个体 e 作为另一父个体
4. 利用父个体 p 和 e 生成子个体 c

4.1. 对 p 和 e 利用正态分布交叉(NDX)算子生成子个体 c

4.2. 对子个体 c 进行多项式变异(PM)

5. 更新进化群体 $P(t)$得到 $P(t+1)$:判断 c 与 $P(t)$中个体的支配关系

5.1. 若 c 被支配,则保持 $P(t)$不变

5.2. 若 c 支配 $P(t)$中个体,则随机替换其中一个

5.3. 否则,随机替换 $P(t)$中的一个个体

6. 更新精英群体 $E(t)$得到 $E(t+1)$

6.1. 利用式(5)计算个体 c 在目标空间中的格子位置

6.2. 根据格子位置 B 判断 c 与 $E(t)$中个体的支配关系

6.2.1. 若 c 被支配,则保持 $E(t)$不变

6.2.2. 若 c 支配其中个体,则从 $E(t)$中删除这些个体,并将 c 加入到 $E(t)$中

6.2.3. 若不存在支配关系且 $E(t)$中没有个体与 c 的 B 值相同,则将 c 加入到 $E(t)$中

6.2.4. 否则,即 $E(t)$中存在一个个体 a 与 c 的 B 值相同,先判断两者在目标函数 f 上的支配关系,若存在支配关系,将被支配的剔除;否则选择与格子最低点距离最近的个体

7. 若终止条件满足,则将精英群体输出;否则,$t=t+1$ 并转到步骤 2

分析算法ε-MOEA/NDX 在最坏情形下每一代的计算时间消耗,设进化群体 $P(t)$和精英群体 $E(t)$的规模分别为 N 和 E,目标函数维数为 M.

步骤 2 采用二元联赛选择,时间消耗为 $O(M+2)$;步骤 3 为随机选择,时间消耗为 $O(1)$;步骤 4 为个体的交叉和变异,所需时间为 $O(M+M)$;步骤 5 中最多需要与 N 个个体比较,时间消耗为 $O(MN)$;而步骤 6 与步骤 5 相似,最多有 E 次比较,所需时间为 $O(ME)$,则算法ε-MOEA/NDX 在最坏情形下每一代的时间消耗为 $O(MN+ME+3M+3)$.而在实际问题求解过程中,选择合适的参数ε可使 E 与 N 同阶,则上述的时间消耗可简化为 $O(2MN+3M+3)$.

再看算法 NSGA-II[10]在最坏情形下每一代的计算时间消耗,设群体规模和目标函数维数分别为 N 和 M.

NSGA-II 的主要步骤为:1) 非支配排序;2) 排挤距离计算;3) 群体选择;4) 交叉和变异.在最坏情形下,根据文献[10]的分析,步骤 1)~步骤 3)的时间消耗分别为 $O(2MN^2)$,$O(M(2N)\log(2N))$,$O(2N\log(2N))$;步骤 4)的时间消耗为 $O(N(M+2+M))$,则算法在最坏情形下每一代的时间消耗为 $O(2MN^2+N(2M+2)\log(2N)+2NM+2N)$.

需要指出的是,在算法的每一代运行中,ε-MOEA/NDX 产生 1 个个体,而 NSGA-II 产生了 N 个个体.因此,为公平比较,考虑在最坏情形下两种算法产生 N 个个体的时间消耗,则:

- 算法ε-MOEA/NDX 的时间消耗为 $O(2MN^2+3NM+3N)$.
- 算法 NSGA-II 的时间消耗为[10]$O(2MN^2+N(2M+2)\log(2N)+2NM+2N)$.

可以看到,由于最高项相同,两种算法产生 N 个个体的时间消耗是等量级的,虽然文献[18]指出算法 NSGA-II 步骤 1)的时间消耗可以降低至 $O(N\log^{M-1}N)$,而对算法ε-MOEA/NDX 的步骤 4 和步骤 5 也可采用类似的策略,则两者的时间消耗仍是等量级的;在其他项的比较中,ε-MOEA/NDX 小于 NSGA-II.因此,在最坏情形下,算法ε-MOEA/NDX 的时间消耗略低于 NSGA-II.

另外,算法ε-MOEA/NDX 与 Deb 等人提出的ε-MOEA[4]唯一的不同在于交叉算子,即前者采用 NDX 算子,后者采用 SBX 算子.因此,两种算法具有相同的时间消耗.

下面具体的实验结果也证明了以上结论.

4 实验结果与讨论

4.1 测试函数与评估指标

为了评估算法ε-MOEA/NDX 的性能,这里选择了二维 ZDT 测试集[10]和 DTLZ 测试集[19]中的 10 个函数,其中在 ZDT 的 5 个函数中 $f_2=g\cdot h$,在 DTLZ 的 5 个函数中 M=3,即目标数为 3,详细描述见表 1.

Table 1 Benchmark functions used in the experiment

表 1 实验中使用的测试函数

Function names	Function descriptions	Dimension	Parameter domains
ZDT1	$f_1=y_1$ $g=1+9\sum_{i=1}^{k}z_i/k$ $h=1-\sqrt{f_1/g}$	30	[0,1]
ZDT2	As ZDT1, except $h=1-(f_1/g)^2$	30	[0,1]
ZDT3	As ZDT1, except $h=1-\sqrt{f_1/h}-(f_1/h)\sin(10\pi f_1)$	30	[0,1]
ZDT4	As ZDT1, except $g=1+10k+\sum_{i=1}^{k}\left(z_i^2-10\cos(4\pi z_i)\right)$	10	$y_1\in[0,1]$ $z_1,\ldots,z_k\in[-5,5]$
ZDT6	$f_1=1-\exp(-4y_1)\sin^6(6\pi y_1)$ $g=1+9\left(\sum_{i=1}^{k}z_i/k\right)^{0.25}$ $h=1-(f_1/g)^2$	10	[0,1]
DTLZ2	$f_1=(1+g)\prod_{i=1}^{M-1}\cos(y_i\pi/2)$ $f_{m=2:M-1}=(1+g)\left(\prod_{i=1}^{M-m}\cos(y_i\pi/2)\right)\sin(y_{M-m+1}\pi/2)$ $f_M=(1+g)\sin(y_1\pi/2)$ $g=\sum_{i=1}^{k}(z_i-0.5)^2$	12	[0,1]
DTLZ4	As DTLZ2, except $y_i\leftarrow y_i^{100}$	12	[0,1]
DTLZ5	As DTLZ2, except $y_i\leftarrow(1+2gy_i)/(2+2g),\ i=2,\ldots,M-1$	12	[0,1]
DTLZ6	As DTLZ5, except $y_i\leftarrow y_i^{100}, g=1+9\sum_{i=1}^{k}z_i^{0.1}$	12	[0,1]
DTLZ7	$f_{m=1:M-1}=y_m$ $f_M=(1+g)\left(M-\sum_{i=1}^{M-1}\left[\frac{f_i}{1+g}(1+\sin(3\pi f_i))\right]\right)$ $g=1+9\sum_{i=1}^{k}z_i/k$	22	[0,1]

多目标优化的目标是快速找到逼近真实 Pareto 前沿且在其上分布均匀的解集,因此,本文将在算法的速度、解集的逼近性和分布性上进行评估,同时以图形方式进行比较说明.Zitzler 等人在文献[20]中对多目标优化评估指标进行了分类研究.在本文的实验中,我们以其中的一元距离指标 GD、标准化空间指标 H 和二元覆盖指标 C 及运行时间对算法进行评估.

距离指标 GD[20]衡量了算法求得的前沿与真实 Pareto 前沿的逼近程度,定义为

$$GD=\frac{1}{|A|}\sum_{i=1}^{|A|}\min\left(\|a_i-p\|,\ p\in P^*\right),\ a_i\in A \tag{14}$$

其中,集合 A 为算法求得的前沿,P^*为参考点集合,在文中是对真实 Pareto 前沿的均匀采样.显然,该值越小,表明求得的前沿越接近真实 Pareto 前沿,即具有良好的逼近性.

标准化空间指标 H[20]表示解集覆盖真实 Pareto 前沿与参考点 RP 所包围空间的比率,与解集的分布性和逼近性都相关,具体定义如下:

$$H=\bigcup_{i=1}^{|A|}hv_i\Big/hv_{p^*} \tag{15}$$

其中,右式中分子表示解集 A 与参考点 RP 包围的空间,分母为真实 Pareto 前沿 P^*与 RP 包围的空间.文中计算该值时,参考点 RP 为真实 Pareto 前沿 P^*的最高点.

覆盖指标 C[3]是一个二元指标,定义了两个解集 A 和 B 之间的覆盖情况,具体如下:

$$C(A,B)=\left|\{b\in B;\exists a\in A:a\prec b\}\right|\Big/|B| \tag{16}$$

一般情况下,$C(A,B)\neq 1-C(B,A)$,因此在实验分析中,这两个值都需要考虑.

4.2 实验结果及分析

我们对ε-MOEA/NDX 与 Deb 等人提出的ε-MOEA[4](记为ε-MOEA/SBX)和 NSGA-II[10]进行了比较.实验中,具体的参数设置与文献[4]中的相同,其中初始群体规模 N 和函数评估次数 FE 见表 2.

Table 2 Initial population size N and function evaluation number FE

表 2 算法的初始群体规模 N 与函数评估次数 FE

	Functions									
	ZDT1	ZDT2	ZDT3	ZDT4	ZDT6	DTLZ2	DTLZ4	DTLZ5	DTLZ6	DTLZ7
N	100	100	100	100	100	100	100	100	100	100
FE	20 000	20 000	20 000	40 000	20 000	30 000	30 000	20 000	30 000	100 000

所有算法均在 VC++ 2005 上编译执行,其中本文算法源码可从作者处获得,而算法ε-MOEA/SBX 和 NSGA-II 的源码来自 Deb 的 KanGAL(http://www.iitk.ac.in/kangal/codes.shtml).实验平台为:Windows XP SP2, Pentium(R)1.7GHZ,1.00 GB RAM.对每个测试函数,每种算法均独立运行 30 次,且对每个函数的每次运行,所有算法的初始群体均是相同的,以达到公平比较的目的.3 种算法的距离指标 GD、标准化空间指标 H 和运行时间 Time 见表 3,覆盖指标 C 见表 4.

在表 3 中,粗体部分表明是最优的均值,不难看出,对二维 ZDT 测试集的 5 个函数和三维 DTLZ4,DTLZ6, DTLZ7 函数,ε-MOEA/NDX 在距离指标 GD 和标准化空间指标 H 上都获得了最好的结果.对函数 DTLZ2,尽管在逼近性上弱于算法ε-MOEA/SBX 和 NSGA-II,但由于采用 NDX 算子,搜索到的空间更为广阔,从而能够获得更为完整的 Pareto 前沿,因此,在标准化空间指标 H 上优于另外两种算法.函数 DTLZ6 具有一个退化的 Pareto 前沿,在三维目标空间中是一条曲线.Deb 等人在文献[19]中指出 NSGA-II 和 SPEA2 都难以找到真实 Pareto 前沿,从指标 GD 和 H 中也能看到算法ε-MOEA/SBX 和 NSGA-II 均难以逼近真实 Pareto 前沿,而算法ε-MOEA/NDX 却可以快速找到完整的 Pareto 前沿.在算法的运行时间方面,算法ε-MOEA/NDX 与ε-MOEA/SBX 相当,但明显优于 NSGA-II,这与第 3 节中的分析相一致.

在表 4 中,粗体部分表明在自由度为 29、显著水平为 0.05 的 T 检验中,覆盖指标 $C(A,B)$优于 $C(B,A)$.从中不难看出,对 ZDT 测试集的 5 个函数,算法ε-MOEA/NDX 在覆盖指标 C 上均明显优于算法ε-MOEA/SBX 和 NSGA-II,特别是在 ZDT6 函数上;而在 DTLZ 的测试集中,在函数 DTLZ2,DTLZ6 和 DTLZ7 中,ε-MOEA/NDX 均明显优于ε-MOEA/SBX 和 NSGA-II,特别是在 DTLZ7 函数中;只是在函数 DTLZ5 上差于这两种算法.

图 4~图 6 对算法在函数 ZDT6 和 DTLZ6 中求得的最优解集进行了图形化比较.对每种算法而言,图中给出的最优解前沿均为 30 次独立运行中距离真实 Pareto 最优解最近的结果.其中,图 4 和图 6 显示的分别是函数 ZDT6 和 DTLZ6 的真实 Pareto 前沿和 3 种算法求得的最优解集,图 5 是对图 4 中一块区域的局部放大.从图中

可以看出,算法ε-MOEA/NDX 具有优秀的逼近能力.

因此,从以上的实验结果及其分析中可以看到,相对算法ε-MOEA/SBX 和 NSGA-II,ε-MOEA/NDX 在求得的 Pareto 最优解集的质量上得到明显的提高.

Table 3 Mean values and standard deviations of the *GD*, *H* and Time indicators

表 3 距离指标 *GD*、标准化空间指标 *H* 和运行时间 Time 的均值和方差

Algorithms		*GD*		*H*		Time	
		Mean values	Standard deviations	Mean values	Standard deviations	Mean values	Standard deviations
ZDT1	ε-MOEA/NDX	**9.3722E−04**	8.7173E−05	**9.9160E−01**	1.9500E−04	8.0470E−01	2.4200E−02
	ε-MOEA/SBX	1.7000E−03	1.9128E−04	9.8990E−01	4.2626E−04	**7.7290E−01**	1.2100E−02
	NSGA-II	2.3000E−03	3.5362E−04	9.8800E−01	7.7258E−04	1.1432E+00	2.3600E−02
ZDT2	ε-MOEA/NDX	**1.1000E−03**	1.2393E−04	**9.8310E−01**	6.1858E−04	7.6040E−01	1.5000E−02
	ε-MOEA/SBX	2.3000E−03	3.0315E−04	9.7650E−01	1.5000E−03	**7.0830E−01**	1.8000E−02
	NSGA-II	2.5000E−03	3.8279E−04	9.7450E−01	1.7000E−03	1.0875E+00	2.3800E−02
ZDT3	ε-MOEA/NDX	**6.9138E−04**	6.2500E−05	**9.9480E−01**	3.3899E−04	**7.2400E−01**	1.8000E−02
	ε-MOEA/SBX	9.7441E−04	1.3636E−04	9.9180E−01	2.9000E−03	7.7600E−01	1.1800E−02
	NSGA-II	1.3000E−03	1.4441E−04	9.9260E−01	1.4000E−03	1.1333E+00	2.5900E−02
ZDT4	ε-MOEA/NDX	**6.9669E−04**	2.6479E−04	**9.9380E−01**	6.6885E−04	**7.0890E−01**	3.6600E−02
	ε-MOEA/SBX	1.4000E−03	6.5614E−04	9.9220E−01	1.5000E−03	7.4640E−01	1.7300E−02
	NSGA-II	1.5000E−03	5.9805E−04	9.9000E−01	1.4000E−03	1.4375E+00	1.4200E−02
ZDT6	ε-MOEA/NDX	**2.9621E−04**	1.3446E−05	**9.8890E−01**	1.5593E−05	5.6300E−01	1.3300E−02
	ε-MOEA/SBX	7.1000E−03	9.1342E−04	9.5700E−01	4.1000E−03	**4.1720E−01**	1.2400E−02
	NSGA-II	1.4800E−02	2.0000E−03	9.2340E−01	8.4000E−03	7.4950E−01	1.0400E−02
DTLZ2	ε-MOEA/NDX	1.3900E−02	6.3354E−04	**8.8540E−01**	1.5000E−03	**1.1411E+00**	2.1500E−02
	ε-MOEA/SBX	**1.2900E−02**	5.2549E−04	8.8240E−01	2.7000E−03	1.1984E+00	2.3300E−02
	NSGA-II	1.3300E−02	9.1798E−04	8.1210E−01	1.2900E−02	1.8948E+00	2.2400E−02
DTLZ4	ε-MOEA/NDX	**1.1300E−02**	4.0280E−04	**8.9660E−01**	1.5000E−03	1.1865E+00	4.5300E−02
	ε-MOEA/SBX	1.1900E−02	7.9000E−03	6.7780E−01	2.5270E−01	**1.1677E+00**	1.4520E−01
	NSGA-II	1.4500E−02	5.2000E−03	7.9710E−01	9.5000E−02	1.9901E+00	7.5400E−02
DTLZ5	ε-MOEA/NDX	2.2000E−03	1.8808E−04	9.5770E−01	1.4000E−03	**7.6880E−01**	1.8500E−02
	ε-MOEA/SBX	**9.1098E−04**	5.9963E−05	**9.6730E−01**	6.6094E−04	8.1980E−01	2.1600E−02
	NSGA-II	1.3000E−03	1.6719E−04	9.6410E−01	2.1000E−03	1.1656E+00	1.1300E−02
DTLZ6	ε-MOEA/NDX	**3.6594E−04**	7.8103E−06	**9.7370E−01**	1.7311E−05	**1.0682E+00**	1.2000E−02
	ε-MOEA/SBX	8.6860E−01	5.7600E−02	0.0000E+00	0.0000E+00	1.5797E+00	3.9900E−02
	NSGA-II	5.3960E−01	7.0500E−02	0.0000E+00	0.0000E+00	1.6750E+00	1.3200E−02
DTLZ7	ε-MOEA/NDX	**7.8000E−03**	3.6669E−04	**8.9740E−01**	2.8000E−03	**3.6427E+00**	5.6100E−02
	ε-MOEA/SBX	7.9000E−03	3.1592E−04	8.9000E−01	2.9600E−02	4.1589E+00	1.3410E−01
	NSGA-II	1.8800E−02	2.7000E−03	8.6660E−01	2.3500E−02	6.8656E+00	7.9600E−02

Table 4 Mean values and standard deviations of the *C* indicator

表 4 覆盖指标 *C* 的均值和方差

Test functions	C (ε-MOEA/NDX, ε-MOEA/SBX)	*C* (ε-MOEA/SBX, ε-MOEA/NDX)	*C* (ε-MOEA/NDX, NSGA-II)	*C* (NSGA-II, ε-MOEA/NDX)
	Mean values (Standard deviations)	Mean values (Standard deviations)	Mean values (Standard deviations)	Mean values (Standard deviations)
ZDT1	**0.265 8** (0.046 4)	0.008 3 (0.007 0)	**0.209 7** (0.067 5)	0.003 7 (0.006 7)
ZDT2	**0.344 3** (0.048 2)	0.006 0 (0.010 0)	**0.243 0** (0.064 4)	0.003 3 (0.005 5)
ZDT3	**0.300 0** (0.107 4)	0.014 1 (0.018 8)	**0.200 0** (0.060 7)	0.003 7 (0.008 2)
ZDT4	**0.271 5** (0.124 0)	0.024 7 (0.056 0)	**0.146 3** (0.125 4)	0.009 7 (0.021 0)
ZDT6	**0.939 7** (0.055 8)	0 (0)	**0.994 3** (0.005 7)	0 (0)
DTLZ2	**0.082 5** (0.021 3)	0.033 8 (0.016 4)	**0.033 3** (0.017 3)	0.004 8 (0.006 5)
DTLZ4	0.028 3 (0.028 2)	0.029 4 (0.017 5)	**0.054 3** (0.024 5)	0.004 1 (0.010 7)
DTLZ5	0.011 0 (0.010 6)	**0.199 1** (0.039 7)	0.033 0 (0.015 3)	**0.138 5** (0.028 5)
DTLZ6	**1** (0)	0 (0)	**0.965 0** (0.024 9)	0 (0)
DTLZ7	**0.013 6** (0.010 7)	0.004 0 (0.006 2)	**0.095 7** (0.029 6)	0.001 1 (0.003 4)

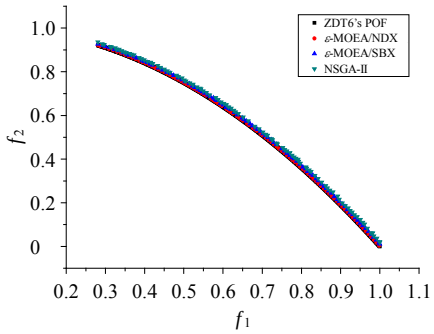

Fig.4 Pareto fronts obtained by the algorithms on function ZDT6

图 4 算法在函数 ZDT6 上的前沿

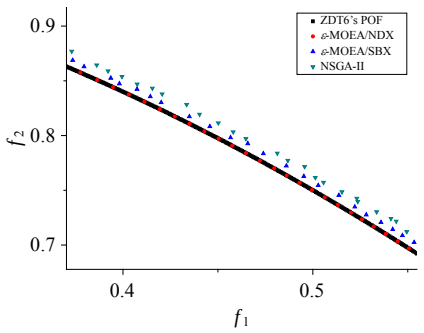

Fig.5 Enlargement of part in Fig.4

图 5 对图 4 局部的放大

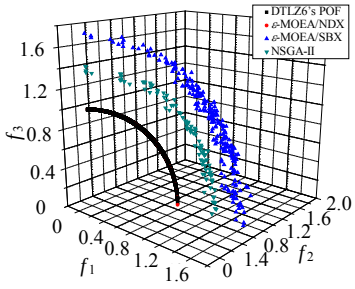

Fig.6 Pareto fronts obtained by the algorithms on function DTLZ6

图 6 算法在函数 DTLZ6 上的前沿

5 结 论

通过对模拟二进制交叉(SBX)算子和进化策略中变异算子的分析和研究,并引入进化策略的重组操作,提出了一种基于正态分布的实数交叉算子(NDX).在一维情形下分析了该算子对搜索空间开发和探测的概率,并以此确定了算子的搜索步长.将 NDX 算子应用于算法ε-MOEA[4]中,与经典算法 NSGA-II 和ε-MOEA/SBX 在 ZDT 和 DTLZ 测试集中的 10 个测试函数上进行了实验比较和分析,结果表明,由于算法ε-MOEA/NDX 采用正态分布交叉(NDX)算子,与 SBX 算子相比,能够搜索到更为广阔的空间,且仍具有与其相当的空间开发能力,因此,在求得的非支配解集的质量上得到明显的提高.在下一步的工作中,我们将利用进化策略的收敛性分析[21]来分析算法ε-MOEA/NDX 在连续空间上的收敛性问题.

References:

[1] Deb K. Multi-Objective Optimization Using Evolutionary Algorithms. Chicester: John Wiley & Sons, 2001.

[2] Horn J, Nafpliotis N, Goldberg DE. A niched Pareto genetic algorithm for multi-objective optimization. In: Michalewicz Z, ed. Proc. of the 1st IEEE Conf. on Evolutionary Computation. Piscataway: IEEE Press, 1994. 82−87.

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

[4] Deb K, Mohan M, Mishra S. Evaluating the ε-domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions. Evolutionary Computation, 2005,13(4):501−525.

[5] Coello Coello CA. Evolutionary multi-objective optimization: A historical view of the field. IEEE Computational Intelligence Magazine, 2006,1(1):28−36.

[6] Srinivas N, Deb K. Multi-Objective optimization using non-dominated sorting in genetic algorithms. Evolutionary Computation, 1994,2(3):221−248.

[7] Fonseca CM, Fleming PJ. Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. In: Forrest S, ed. Proc. of the 5th Int'l Conf. on Genetic Algorithms. University of Illinois at Urbana-Champaign, 1993. 416−423.

[8] Zitzler E, Laumanns M, Thiele L. SPEA2: Improving the strength Pareto evolutionary algorithm. In: Giannakoglou K, ed. Proc. of the EUROGEN 2001. Athens: Int'l Center for Numerical Methods in Engineering (CIMNE), 2002. 95−100.

[9] Knowles JD, Corne DW. Approximating the non-dominated front using the Pareto archived evolution strategy. Evolutionary Computation, 2000,8(2):149−172.

[10] Deb K, Pratap A, Agarwal S, Meyarivan T. A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Trans. on Evolutionary Computation, 2002,6(2):182−197.

[11] Laumanns M, Thiele L, Deb K, Zitzler E. Combining convergence and diversity in evolutionary multi-objective optimization. Evolutionary Computation, 2002,10(3):263−282.

[12] Deb K, Agrawal RB. Simulated binary crossover for continuous search space. Complex Systems, 1995,9(4):115−148.

[13] Bäck T, Schwefel HP. An overview of evolutionary algorithms for parameter optimization. Evolutionary Computation, 1993,1(1): 1−23.

[14] Zhang M, Geng HT, Luo WJ, Huang LF, Wang XF. A hybrid of differential evolution and genetic algorithm for constrained multiobjective optimization problems. In: Wang TD, ed. Proc. of the SEAL 2006. LNCS 4247, Berlin: Springer-Verlag, 2006. 318−327.

[15] Geng HT, Zhang M, Huang LF, Wang XF. Infeasible elitists and stochastic ranking selection in constrained evolutionary multi-objective optimization. In: Wang TD, ed. Proc. of the SEAL 2006. LNCS 4247, Berlin: Springer-Verlag, 2006. 336−343.

[16] Herrera F, Lozano M, Sánchez AM. A taxonomy for the crossover operator for real-coded genetic algorithms: An experimental study. Int'l Journal of Intelligent Systems, 2003,18(3):309−338.

[17] Kita H, Ono I, Kobayashi S. Multi-Parental extension of the unimodal normal distribution crossover for real-coded genetic algorithms. In: Angeline PJ, ed. Proc. of the CEC'99. Piscataway: IEEE Press, 1999. 1581−1587.

[18] Jensen MT. Reducing the run-time complexity of multiobjective EAs: The NSGA-II and Other Algorithms. IEEE Trans. on Evolutionary Computation, 2003,7(5):503−515.

[19] Deb K, Thiele L, Laumanns M, Zitzler E. Scalable multi-objective optimization test problems. In: Yao X, ed. Proc. of the 2002 Congress on Evolutionary Computation. Piscataway: IEEE Service Center, 2002. 825−830.

[20] Zitzler E, Thiele L, Laumanns M, Fonseca CM, Fonseca VG. Performance assessment of multiobjective optimizers: An analysis and review. IEEE Trans. on Evolutionary Computation, 2003,7(2):117−132.

[21] Guo CH, Tang HW. Global convergence properties of evolution strategies. Mathematica Numeric Sinica, 2001,23(1):105−110 (in Chinese with English abstract).

附中文参考文献:

[21] 郭崇慧,唐焕文.演化策略的全局收敛性.计算数学,2001,23(1):105−110.

张敏(1981－),男,安徽肥东人,博士,主要研究领域为进化算法,约束优化,多目标优化.

罗文坚(1974－),男,博士,副教授,主要研究领域为自然计算,硬件进化.

王煦法(1948－),男,教授,博士生导师,CCF高级会员,主要研究领域为智能信息处理.