多维实数编码遗传算法*

雷德明
(武汉交通科技大学自控系 430063)

摘 要 系统地分析了几种常见编码策略,在此基础上提出一种新的编码策略——多维实数编码。仿真结果验证了这种新编码方式的有效性与合理性。

关键词 编码策略,多维实数编码,遗传算法
分类号 TP 302.7

Multidimensional Real- coded Genetic Algorithm

Lei Deming
(Wuhan Transportation University)

Abstract First several ordinary coded methods are analyzed systematically, then a new coded strategy of genetic algorithm is given, which is called the multidimensional real- coded. The simulation results demonstrate its validity.

Key words coded strategy, multidimensional real- coded, genetic algorithm

1 引 言

遗传算法(GA)作为求解大规模复杂问题强有力的计算工具,已被人们广泛接受,并得到了广泛应用。通过应用,人们逐步认识到该算法自身存在计算效率低、过早收敛等缺陷。这些缺陷限制了算法的求解质量,使其不能适应更复杂、计算要求更高的场合,如利用 GA 在线获取与调节模糊规则等。

遗传算法有许多优异特性,其中之一是不直接对变量做迭代运算,而是利用变量的编码,通过对个体不断施加遗传操作(复制、交叉、变异)求得全局最优解。因此,适当的编码策略与编码长度,将直接影响 GA 的精度与效率。

Holland 在运用模式定理分析编码机制时,建议使用二进制编码。但二进制编码不能直接反映问题的固有结构,精度不高,个体长度大,占用计算机内存多;而且二进制编码 GA 的稳定性不如实值编码 GA[1]。

除二进制编码外,还有几种常见的实数编码策略,即:十进制编码、实值编码和指数编码。尽管实数编码精度高,适合于复杂大空间搜索,但这些实数编码策略并未得到广泛应用。为充分发挥实数编码在数值优化方面的巨大优势,本文提出一种全新的实数编码策略——多维实数编码,它把个体表示为一多维矩阵。仿真结果表明,多维实数编码 GA 具有收敛速度快、稳定性好等特点。

2 编码策略分析

2.1 几种常见的编码策略

优化问题的一般形式为

$$\min f(x), \quad x = (x_1, x_2, \cdots, x_n) \tag{1}$$

$$a_i \leqslant x_i \leqslant b, \quad i = 1, 2, \cdots, n$$

二进制编码是应用最早、最广泛的一种编码方式,它根据公式

$$x_i = a_i + \frac{\sum_{j=1}^{p} t_j 2^{j-1}}{2^p - 1}(b_i - a_i), \quad t_j \in \{0, 1\} \tag{2}$$

将变量 x_i 映射成长度为 p 的二进制位串,然后按顺序将每个 x_i 所对应的位串连接起来,即构成种群的基本单位(个体)。个体的长度为 $n \times p$, p 的大小根据实际要求确定,它决定了搜索空间的大小。

十进制编码与二进制编码相似,只是每个基因位有 10 种可能取值(0 ~ 9)。

* 1998- 11- 09 收稿,1999- 03- 07 修回

实值编码则直接把每个变量当作基因处理, 它是一种变形的十进制编码(每个基因位的可能取值远不止 10 种), 也是一种没有编码的编码方式。实值编码受到人们的重视[2], 但其过短的个体长度不利于充分发挥 GA 强大的搜索能力, 效率也相对低下。

指数编码将变量分成数字段与一位指数位进行编码, 其中数字段由变量所有的有效数字组成, 它特别适合于大范围搜索。

2.2 多维实数编码

完整的交叉过程包括: 根据交叉概率 p_c 选择交叉个体, 确定交叉位置, 互换相应位置间的基因位。一般的交叉操作无法将不同位置上的基因位互换, 导致个体通过交叉所获取的信息量少, 且易产生无效交叉。所谓无效交叉是指完全不能或基本不能产生新个体的交叉。无效交叉既浪费进化时间, 又不利于种群进化。

多维实数编码遗传算法能将不同位置上的基因互换, 使无效交叉发生的可能性大大降低。同时, 其合理的编码长度也有助于算法在短时间内获得高精度的全局最优解。

下面以二维实数编码为例加以说明。二维实数编码的个体为一个二维矩阵, 例如

$$x = (-\ a_0a_1a_2.a_3a_4a_5a_6,\ b_0b_1b_2.b_3b_4b_5b_6)$$

则对应的编码为

$$\begin{bmatrix} - & a_0 & a_1 & a_2 & a_3 & a_4 & b_0 & b_1 & b_2 & b_3 & b_4 \\ - & a_0 & a_1 & a_2 & a_5 & a_6 & b_0 & b_1 & b_2 & b_5 & b_6 \end{bmatrix}$$

若整数部分中的某些数字(如 a_1, a_2) 可能取值为 0 ~ 9, 则编码为

$$\begin{bmatrix} - & a_0 & a_1 & a_3 & a_4 & b_0 & b_1 & b_2 & b_3 & b_4 \\ - & a_0 & a_2 & a_5 & a_6 & b_0 & b_1 & b_2 & b_5 & b_6 \end{bmatrix}$$

这样做可节省存贮空间。对一些特殊情况, 编码时做如下处理: 若 $-1 < x_i < 0$, 则整数部分的最高位与该变量取值范围之外的某一负整数对应, 小数部分的每个数字与整数部分的其它数字均不带负号; 若 $x_i \leqslant -1$, 则负号仅加在整数部分的最高位前, 该变量的其它数字也不带负号。

上述编码尽管浪费了一些存贮空间, 但与二进制编码相比, 一般可节省存贮空间 60% 左右, 而且精度越高, 节省空间越多。

两点交叉: 根据 p_c 选择个体进入交配池, 然后任选交配池中两个体, 确定交叉行与交叉位置后, 互换相应位置的基因位。

$$\text{父代 1:}\begin{bmatrix} - & a_1 & a_2 & a_3 & b_1 & b_2 & b_3 \\ - & \underline{a_1} & \underline{a_4} & \underline{a_5} & b_1 & b_4 & b_5 \end{bmatrix}$$

$$\text{父代 2:}\begin{bmatrix} \underline{c_1} & \underline{c_2} & \underline{c_3} & d_1 & d_2 & d_3 \\ c_1 & c_4 & c_5 & d_1 & d_4 & d_5 \end{bmatrix}$$

$$\text{子代 1:}\begin{bmatrix} - & a_1 & a_2 & a_3 & b_1 & b_2 & b_3 \\ & c_1 & c_2 & c_3 & b_1 & b_4 & b_5 \end{bmatrix}$$

$$\text{子代 2:}\begin{bmatrix} - & a_1 & a_4 & a_5 & d_1 & d_2 & d_3 \\ & c_1 & c_4 & c_5 & d_1 & d_4 & d_5 \end{bmatrix}$$

变异: 首先确定参加变异的个体数 m, m 为 $N \times p_m$ 的整数部分(N 为种群规模, p_m 为变异概率), 然后分别在适应值最大的 m 个个体附近搜索以适应更大的个体, 若能则取代原个体, 否则原个体不变。由于快速的种群进化要求最优个体不断更迭, 不断变优, 故在最优个体附近搜索新个体的次数明显多于在其它个体附近的搜索次数。

变异的主要目的是不断产生适应值更大的最优个体, 引导整个种群进化。这一点完全不同于二进制编码。在二进制编码 GA 中, 变异是为了恢复某些丢失的重要信息, 防止因所有个体某个基因位具有相同的基因码, 使搜索限制在搜索空间的某个仿射子空间上。这种现象在多维实数编码 GA 中不可能发生。

许多研究表明, GA 善于产生平均适应值高的种群, 但缺乏产生最优个体强有力的手段。强调变异的目的是不断产生适应值更大的最优个体, 以弥补 GA 这方面的缺陷。另外, 最优个体始终不参与复制和交叉, 但参与变异。

二维实数编码 GA 的步骤如下:

Step1: 编码, 生成初始群体, 计算各个体的适应值;

Step2: 根据复制概率 $p_i = f_i \Big\backslash \sum_i f_i$ (f_i 为个体 i 的适应值) 对种群加以复制, 选择个体进入下一代;

Step3: 按上述两点交叉操作对个体进行交叉运算;

Step4: 解码, 计算各个体的适应值并排序;

Step5: 按上述变异操作, 在适应值最大的部分个体附近搜索新解, 若能则对新解编码后取代原个体, 否则原个体不变;

Step6: 重复 Step2 - Step5, 直到满足终止条件。

对一般优化问题, 二维实数编码是合适的。只有在精度要求特别高的场合, 才用到三维编码。三维编码是在上述二维编码的基础上加一行。例如

$$x = a_1.a_2a_3a_4a_5a_6a_7a_8a_9a_{10}$$

则编码为

$$\begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ a_1 & a_5 & a_6 & a_7 \\ a_8 & a_9 & a_{10} & < \end{bmatrix}$$

当列数比较多时, 第 3 行有些元素可为空。

三维实数编码 GA 的主要步骤与二维实数编码 GA 完全一样, 只是交叉时要单独处理第 3 行。只有两个第 3 行均为交叉行时, 才互换两个体第 3 行相应位置间的基因位, 即第 3 行元素不能与第 1、2 行互换。当搜索接近结束时, 可适当增加第 3 行交叉的概率。

3 仿　　真

本文用如下 5 个典型测试函数来检验二维实数编码 GA 的性能。检验标准为: 1) 反映收敛快慢的平均进化代数, 即算法收敛时种群进化代数的平均值; 2) 反映解优劣的最优解平均值; 3) 反映算法稳定性的最优解标准差。

5 个典型测试函数如下:

$F_1 = \sum_{i=1}^{3} x_i^2$, $|x_i| \leqslant 5.12$, 最大值为 78.643 2;

$F_2 = \sum_{i=1}^{3} \mathrm{int\,eger}(x_i)$, $|x_i| \leqslant 5.12$, 最大值为 25;

$$F_3 = \left\{ \sum_{i=1}^{5} i\cos[(i+1)x + 1] \right\} \times \left\{ \sum_{i=1}^{5} i\cos[(i+1)y + 1] \right\} + 0.5[(x + 1.425\,3)^2 + (y + 0.800\,32)^2]$$

其全局最小值为 - 185.229 3, 易陷入局部最小值 - 184.839;

$$F_4 = 0.5 - \frac{\sin^2 \sqrt{x^2 + y^2} - 0.5}{[1 + 0.001(x^2 + y^2)]^2}$$

其中, $-100 < x, y < 100$, 最大值为 1;

$$F_5 = (4 - 2.1x^2 + x^4/3)x^2 + xy + (-4 + 4y^2)y^2$$

其中, $-100 < x, y < 100$, 最小值为 - 1.031 628。

仿真时, 参数设置如下: $N = 20$, $P_m = 0.15$, $P_c = 0.6$。当 $|x - x^*| \leqslant 0.005$ 时, 算法收敛, x^* 为已知的全局最优解。随机运行程序 50 次, 仿真结果如表 1 所示。

表 1　二维实数编码 GA 的仿真结果

函数	平均进化代数	最优解平均值	最优解标准差
F_1	14.71	78.643 196	0
F_2	9.85	25	0
F_3	24	- 185.113	0.207
F_4	8	1	0
F_5	30	- 1.031 625	10^{-6}

其中 F_4 和 F_5 优化时, 采用了重新启动技术, 即种群进化一定代数后, 重新产生一个新的种群。但变量的取值范围缩小了。

上述仿真结果说明, 本文的多维实数编码遗传算法不仅能在短时间内搜索到高精度的全局最优解, 而且稳定性也非常好。

4 结　　论

与二进制编码相比, 实数编码在数值优化方面具有更高的精度和效率, 而目前常见的几种实数编码方式(如十进制编码、实值编码等) 并不具备上述优点。多维实数编码则不同, 它具有高精度与高效率。可以预见, 多维实数编码遗传算法将有良好的应用前景。

参 考 文 献

1 张晓缋, 方浩, 戴冠中. 遗传算法的编码机制研究. 信息与控制, 1997, 26(2): 134- 139

2 黄晓峰, 潘立登, 陈标华, 等. 实数编码遗传算法中交叉操作的效率分析. 控制与决策, 1998, 13(增刊): 496- 499

3 Holland J H. Adaptation in nature and artificial system. Ann Arbor: University of Michigan Press, 1975

4 Goldberg D. Genetic algorithm in search, optimization and machine learning. RA: Addison- Wesley, 1989

作 者 简 介

雷德明　男, 1968 年生。1996 年于西安交通大学获理学硕士学位, 现为武汉交通科技大学自控系讲师。研究方向为进化算法及其在控制中的应用。