一种基于新约束处理方法的遗传算法

苏勇彦[1], 王 攀[1], 范 衡[2]
SU Yong-yan[1], WANG Pan[1], FAN Zhun[2]

1.武汉理工大学 自动化学院, 武汉 430070
2.丹麦技术大学 机械系, 哥本哈根
1.School of Automation, Wuhan University of Technology, Wuhan 430070, China
2.Department of Mechanical Engineering, Technical University of Denmark, Copenhagen, Denmark

SU Yong-yan, WANG Pan, FAN Zhun.Genetic algorithm based on novel constraints addressing method.Computer Engineering and Applications, 2007, 43(14): 71-72.

Abstract: A new method to handle constrained optimization is proposed in this paper, to overcome certain disadvantages of the current methods.This method searches the solution space of the problem through the admixture crossover of feasible and infeasible solutions, and performs the selection operation on feasible and infeasible populations respectively.It avoids the difficulty of selecting the penalty factor in penalty strategy and makes the handling constraint simplify.Numerical results show that it is an effective method.
Key words: genetic algorithm; constraint handling; feasible solution; infeasible solution

摘 要: 针对目前的约束处理方法中存在的问题, 提出一种新的约束处理方法。该方法通过可行解和不可行解混合交叉的方法对问题的解空间进行搜索, 对可行种群和不可行种群分别进行选择操作。避免了惩罚策略中选取惩罚因子的困难, 使得约束处理问题简单化。实例测试结果表明, 该约束处理方法的有效性。
关键词: 遗传算法; 约束处理; 可行解; 不可行解
文章编号: 1002-8331(2007)14-0071-02 **文献标识码**: A **中图分类号**: TP301.6

1 引言

科学研究和生产生活中许多问题都可以转化为求解一个带约束条件的函数优化问题[1]。遗传算法(Genetic Algorithm)与许多基于梯度的优化算法比较, 具有不需要目标函数和约束条件可微, 且能收敛到多个全局最优解的优点[2], 因此, 它成为一种约束优化问题求解的有力工具。目前, 基于 GA 的约束处理方法有拒绝策略、修复策略、改进遗传算子策略以及惩罚函数策略等。但是这些方法都存在一些问题[3]: 修复策略对问题本身的依赖性, 对于每个问题必须设计专门的修复程序。改进遗传算子策略则需要设计针对问题的表达方式以及专门的遗传算子来维持解的可行性。惩罚策略解的质量严重依赖于惩罚因子的选取, 当惩罚因子不适当时, 算法可能收敛于不可行解。

本文针对目前的约束处理方法中存在的问题, 初步提出一种新的约束处理方法。该方法通过可行解和不可行解混合交叉的方法对问题的解空间进行搜索, 对可行种群和不可行种群分别进行选择操作。避免了惩罚策略中选取惩罚因子的困难, 使得约束处理问题简单化。实例测试结果表明, 该约束处理方法在一些问题上具有有效性。

2 约束处理方法描述

2.1 单目标有约束优化问题一般形式

$$\max f(x)$$
$$\text{s.t. } g_i(x)\leq 0;\ i=1,2,\cdots,m_1$$
$$h_i(x)=0;\ i=m_1+1,\cdots,m(=m_1+m_2)$$
$$x\in X$$

这里 $f, g_1, g_2, \cdots, g_{m_1}; h_{m_1+1}, h_{m_1+2}, \cdots, h_m$ 都是定义在 E^n 上的实值函数。X 是 E^n 上的子集, x 是 n 维实向量, 其分量为 $x_1, x_2, \cdots, x_n$。上述问题要求在变量 $x_1, x_2, \cdots, x_n$ 满足约束的同时极大化函数 f。函数 f 通常为目标函数。约束 $g_i(x)\leq 0$ 称为不等式约束; 约束 $h_i(x)=0$; 称为等式约束。集合 X 通常为变量的上下界限定的区域。向量 $x\in X$ 且满足所有约束, 则称之为问题的可行解。所有可行解构成可行域。否则, 为问题的不可行解, 所有不可行解构成不可行域。问题的目标是找到一个可行解 $\bar{x}$ 使得 $f(x)\leq f(\bar{x})$ 对于所有可行解 x 成立。那么, $\bar{x}$ 为最优解[4]。

2.2 算法描述

目前, 最常采用的约束处理方法为惩罚函数法。但优化搜索的效率对惩罚因子的选择有明显的依赖性。同时, 惩罚因子没有统一的选择标准, 使得惩罚因子选择非常困难。

基于惩罚函数的方法, 选择是对可行解与不可行解的混合

作者简介: 苏勇彦(1981-), 硕士研究生, 主要研究领域为进化计算、多目标优化; 王攀, 博士, 副教授, 主要研究领域为决策分析、智能控制、生物医学智能化系统; 范衡, 博士, 助理教授, 主要研究领域为演化算法、微机电系统设计。

种群进行的。若种群中只有可行解,则不会出现惩罚因子的问题。但若采用拒绝策略,完全拒绝不可行解则大大减少搜索范围,很难收敛到最优解。‘既然目标是找到可行的最优解,一定要用对不可行解进行惩罚的方法吗?’[5]正是根据这一思路,本文提出一种既利用不可行解扩大搜索范围,又不引入惩罚因子的约束优化处理方法。

该方法引入两个初始种群:可行种群和不可行种群。同时采用实数编码法,对两个种群进行的交叉操作为:分别从可行种群 popf 和不可行种群 popfinf 中随机选择一个个体,p_1,p_2 为父代个体进行混合的算术交叉,生成两个新个体 c_1,c_2。采用可行种群与不可行种群个体进行交叉的目的是扩大搜索空间。对交叉、变异后生成的新个体进行判断,将其分为可行种群和不可行种群,分别对可行种群和不可行种群进行选择操作。这样操作有利于将不可行种群吸引至可行区域,可望减少‘无功功率’。同时,在现实中存在一大类约束优化问题,其最优解位于约束边界上或附近,及最优点处不等式约束全部或大部分取为等号。因此,采用如下称为凸交叉的算术交叉,可方便地使解穿越边界,并在运用一些特殊方法后将解定位于约束边界附近,算术交叉操作及参数选择如下[4]:

$c_1=\lambda_1 p_1+\lambda_2 p_2$

$c_2=\lambda_1 p_2+\lambda_2 p_1$

$\lambda_1+\lambda_2=1$

$\lambda_1>0,\lambda_2>0$

如图 1 所示,可行区域的点 A 与不可行区域点 C 交叉时,生成的个体在 A、C 之间的连线上。由于可行种群来自于可行区域,不可行种群来自于不可行区域,对可行种群和不可行种群混合交叉产生的个体必位于可行区域与不可行区域之间,即约束边界附近的点。因此,该交叉方法可以搜索到约束边界附近的点。

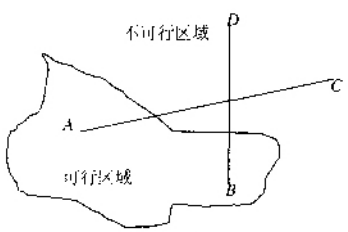

图 1 图示混合算术交叉

如何从交叉和变异中产生的新种群以及交叉变异之前的旧种群中找到最优解?这是选择操作要解决的问题。本文的方法是将原始种群分为可行种群和不可行种群,对两种群分别进行选择操作(具体操作见算法流程)。选择操作实现对个体适应值的评估,通过优胜劣态的进化原理最终收敛到最优解。这种方法就避免了确定惩罚因子带来的困难。

2.3 算法流程

步骤 1 初始化,设置变异概率 p_m、可行种群规模 N_1 和不可行种群规模 N_2,根据约束条件对初始种群中的个体进行判断,将初始种群分为可行种群 popf 和不可行种群 popfinf;

步骤 2 对可行种群 popf与不可行种群 popinf 进行算术交叉、变异操作;

步骤 3 根据约束条件对交叉、变异后生成的新种群中个体进行判断,将种群分为可行种群 popf和不可行种群 popinf,种群规模分别为 M_1,M_2(M_1,M_2 值在每代可变);

步骤 4 选择操作:从旧可行种群和新可行种群中根据个体适应值的大小选择较优的 N_1 个个体形成新的可行种群;同理,得到有 N_2 个个体的新的不可行种群;

步骤 5 判断算法终止条件是否满足?不满足转到步骤 2;满足,终止程序。

3 问题测试及实验结果

数值测试例子

问题 1[6]

$$\max\ f(x)=-2x_1^2+2x_1x_2-2x_2^2+4x_1+6x_2$$

$$\text{s.t. } x_1+x_2\leq 2$$

$$x_1+5x_2\leq 5$$

$$x_1\geq 0,\ x_2\geq 0$$

问题 2[4,7]

$$\min\ f(x)=5.357\,854\,7x_3^2+0.835\,689\,1x_1x_5+37.293\,239x_1-40\,792.141$$

$$\text{s.t. } 0\leq 85.334\,407+0.005\,686\,8x_2x_5+0.000\,26x_1x_4-0.002\,205\,3x_3x_5\leq 92$$

$$90\leq 80.512\,49+0.007\,131\,7x_2x_5+0.002\,995\,5x_1x_2+0.002\,181\,3x_3^2\leq 110$$

$$20\leq 9.300\,961+0.004\,702\,6x_3x_5+0.001\,254\,7x_1x_3+0.001\,908\,5x_3x_4\leq 25$$

$$78\leq x_1\leq 102$$

$$33\leq x_2\leq 45$$

$$27\leq x_3\leq 45$$

$$27\leq x_4\leq 45$$

$$27\leq x_5\leq 45$$

根据上文提出的约束处理方法,编写 matlab 程序进行实验,对问题 1,2 求解结果为:

问题 1 初始种群 popsize 大小为 400,可行种群与不可行种群分别为初始种群大小的一半且保持每代中由选择产生新的可行种群和不可行种群的规模保持不变。迭代次数 gen 为 150,变异概率 p_m 为 0.05,程序运行 10 次得到最好解 f(x)=7.161 1,对应的变量值为(1.128 7,0.774 2)。唐加福等采用混合遗传算法(HGA)得到的最优解 7.160 85[6]。

问题 2 初始种群大小 popsize 为 400,可行种群与不可行种群分别为初始种群大小的一半,迭代次数 gen 为 100,变异概率 p_m 为 0.05,程序运行 10 次得到最好解 f(x)=-3.091 8e+004,对应的变量值为(78.300 2,33.860 6,27.873 5,44.803 6,42.665 3)。与 Homaifar,Qi 和 Lai 用基于惩罚策略的遗传算法得到解[8]以及广义简约梯度法(GRG)得到的结果相比有很大的提高,如表 1 所示。同时,李敏强、寇纪淞等基于遗传算法的直接比较-比例方法[1]得到结果最好解为-30 665.539[9]。通过实验比较可知,该方法是一种很有效的约束处理方法。

表 1 4 种算法计算结果比较

项目	参考解	遗传算法基于全局参考的解	遗传算法基于局部参考的解	GRG 的解	本文的方法得到的解
f(x)	-30 665.5	-30 175.804	-30 182.269	-30 373.950	-30 918.00
x_1	78.00	80.61	81.49	78.62	78.300 2
x_2	33.00	34.21	34.09	33.44	33.860 6
x_3	29.995	31.34	31.24	31.07	27.873 5
x_4	45.00	42.05	42.20	44.18	44.803 6
x_5	36.776	34.85	34.37	35.22	42.665 3

(下转 86 页)

同样的方法编写和编译。

4.3 不同平台下的实验结果

笔者在不同平台下对编写的跨平台十字线叠加程序进行了测试，它们都能够不加任何修改地在4个不同CPU和4个不同操作系统下编译和运行，如表2所示。

表2 测试过的不同平台

处理器	操作系统	编译器	运行结果
AMD Athlon	Linux 2.6.17-2-k7	GCC 4.1.2	全部通过
Intel Xeon	FreeBSD 6.0	GCC 3.4.4	全部通过
SUNW Ultra-250	SunOS 5.8	GCC 3.2	全部通过
Intel EM64T	Windows XP x64	MinGW/GCC 3.4.2	全部通过

另外，在Windows下，程序在Cygwin和Visual C++ 6.0下也都编译通过，并且得到正确的图像处理结果。

图3是原始彩色图像，图4是中值滤波后图像，图5是灰度化和二值化后的图像，图6是形心计算、并在目标中心叠加十字线的图像。

图3 原始图像

图4 中值滤波后的图像

图5 灰度化和二值化后的图像

图6 处理后叠加十字线的图像

在不同平台下的可执行程序都得到了完全相同的图像处理结果，满足了跨平台的要求。

5 结论

本文在跨平台编程思想的基础上，对跨平台的BMP图像处理方法进行了实践。在多个不同的CPU和操作系统下对笔者编写的BMP图像处理程序进行了实验，得到了很好的跨平台结果。对比分析显示，跨平台编程方法可以大大提高图像处理程序的通用性和兼容性，为图像处理编程技术拓宽了思路。

（收稿日期：2007年1月）

参考文献：

[1] 李莹，张嵩，章坚武.嵌入式Linux操作系统上的BMP文件的处理[J].杭州电子工业学院学报，2004，24(4)：35-38.

[2] The.bmp file format[EB/OL].http://www.fortunecity.com/skyscraper/windows/364/bmpffrmt.html.

[3] Cross-platform[EB/OL].http://en.wikipedia.org/wiki/Cross-platform

[4] EasyBMP Cross-Platform Windows BMP Libray[EB/OL].http://easybmp.sourceforge.net.

[5] 谷口庆治.数字图像处理[M].朱虹，译.北京：科学出版社，2002：80-85.

[6] 闫伟，金元郁.基于Visual C++的运动目标形心捕获[J].微计算机信息，2005，21(2)：180-182.

（上接59页）

[12] Zhang J，Modestino W，Langan D A.Maximum-likelihood parameter estimation for unsupervised model-based image segmentation[J].IEEE Trans Image Processing，1994，3：404-420.

[13] Marroquin J L，Vemuri B C，Botello S，et al.An accurate and efficient Bayesian method for automatic segmentation of brain MRI[J].IEEE Trans Med Imag，2002，21：934-945.

[14] 詹劲峰，戚飞虎，王海龙.基于时空马尔可夫随机场的运动目标分割技术[J].通信学报，2000，7(5)：434-439.

[15] Marroquin J L，Botello S，Calderon F，et al.MPM-MAP for image segmentation[C]//Proc ICPR 2000，Barcelona，Spain，2000，4：300-310.

[16] Zhang Yong-yue，Brady M，Smith S.Segmentation of brain MR images through a hidden Markov random field model and the expection-maximization algorithm[J].IEEE Transactions on Medical Imaging，2001，20(1)，45-57.

（上接72页）

4 结论

利用遗传算法解决约束优化问题时，出现的问题是在进行交叉、变异操作时，会出现不可行解。可以按照处理不可行解的方法对约束优化问题处理方法进行分类。完全拒绝不可行解的方法就是拒绝策略，惩罚策略对不可行解的适应度值进行处理。本文的方法是引入可行种群和不可行种群两个种群进行混合交叉、变异操作来扩大搜索范围，增加种群的多样性，产生新的解；然后两个种群分别进行选择操作以提高种群的平均适应度值。通过几个常用的测试问题的求解以及与其它几种约束处理方法的比较表明：本文提出的约束处理方法具有很好的性能，且处理方法简单。

同时，必须指出，本研究在理论分析和实践应用上都尚不充分，进一步的研究问题还包括：所提方法的有效性分析；各种约束处理方法的比较研究，特别是各种处理方法的效率的比较，遗传算法中种群大小、交叉概率、变异概率、迭代次数等参数的选取对算法的性能的影响的研究。同时，任何算法都有其适用范围，因此进一步研究本文算法对哪些问题效果好也十分重要和必要。（收稿日期：2006年9月）

参考文献：

[1] 林丹，李敏强，寇纪淞.基于遗传算法求解约束优化问题的一种算法[J].软件学报，2001，12(4)：628-632.

[2] Goldberg D E.Genetic algorithms in search，optimization & machine learning[M].[S.l.]：Addison-Wesley publishing company，1989.

[3] Michalewicz Z，Dasgupta D，Le Riche R G，et al.Evolutionary algorithms for constrained engineering problems[J].Computers & Industrial Engineering Journal，1996，30(2)：851-870.

[4] 玄光男，程润伟.遗传算法与工程设计[M].北京：科学出版社，2000.

[5] Michalewicz Z.Evolutionary algorithms for constrained optimization[C]//IWEC'2000 International Workshop on Evolutionary Computation，Wuhan，2000：1-11.

[6] Tang Jia-fu，Wang Ding-wei，Ip A，et al.A hybrid genetic algorithm for a type of non-linear programming problems[J].Computers and Mathematics with Applications，1998，36(5)：11-21.

[7] Himmelblau M.Applied nonlinear programming[M].New York：McGraw-Hill，1972.

[8] Homaifar A，Qi C，Lai S.Constrained optimization via genetic algorithms[J].Simulation，1994，62(4)：242-254.

[9] 李敏强，寇纪淞，林丹，等.遗传算法的基本理论与应用[M].北京：科学出版社，2002.