解约束优化问题的一种新的罚函数模型

胡一波[1,2] 王宇平[1]

(西安电子科技大学计算机学院 西安 710071)[1] (西安电子科技大学理学院 西安 710071)[2]

摘 要 罚函数法是进化算法中解决约束优化问题最常用的方法之一,它通过对不可行解进行惩罚使得搜索逐步进入可行域。罚函数常定义为目标函数与惩罚项之和,其缺陷一方面在于此模型的罚因子难以控制,另一方面当目标函数值与惩罚项的函数值的差值很大时,此模型不能有效地区分可行解与不可行解,从而不能有效处理约束。为了克服这些缺点,首先引入了目标满意度函数与约束满意度函数,前者是根据目标函数对解的满意度给出的一个度量,而后者是根据约束违反度对解的满意度给出的一个度量。然后将两者有机结合,定义了一种新的罚函数,给出了一种新的罚函数模型。并且设置了自适应动态罚因子,其随着当前种群质量和进化代数的改变而改变。因此它很易于控制。进一步设计了新的杂交和变异算子,在此基础上提出了解决约束优化问题的一种新的进化算法。通过对6个常用标准测试函数所作的数据仿真实验表明,提出的算法是十分有效的。

关键词 进化算法,约束优化,满意度函数,罚函数

中图法分类号 TP18 **文献标识码** A

New Penalty Model for Constrained Optimization Problems

HU Yi-bo[1,2] WANG Yu-ping[1]

(School of Computer Science and Technology, Xidian University, Xi'an 710071, China)[1]

(Department of Mathematics Science, Faculty of Science, Xidian University, Xi'an 710071, China)[2]

Abstract Penalty function method is one of the most widely used methods for constrained optimization problems in evolutionary algorithms. It makes the search approach the feasible region gradually by the way to punish the infeasible solutions. The penalty functions are usually defined as the sum of the objective function and the penalty terms. The methods will bring two main drawbacks. Firstly, it is difficult to control penalty parameters, secondly, when the difference between the objective function value and the constrained function value is great, the algorithm can not effectively distinguish feasible from infeasible solutions, and thus can not handle the constraints effectively. To overcome the defects, two satisfaction degree functions defined by the objective function and the constraints function were designed, respectively. A new penalty function was constructed by these two satisfaction degree functions. Moreover, we designed an adaptive penalty factor which is varying with the quality of the population and the number of generations. As a result, the penalty factor can be easily controlled. Thus a new penalty function optimization model was proposed. Furthermore, a new crossover operator and a new mutation operator were designed. Based on these, a new evolutionary algorithm for constrained optimization problems was proposed. The simulations are made on six widely used benchmark problems, and the results indicate the proposed algorithm is very effective.

Keywords Evolutionary algorithm, Constrained optimization, Penalty function

1 引言

进化算法在优化问题求解上取得了巨大的成功,与传统优化方法相比,进化算法只要求被优化的函数值是可计算的,不要求它具有连续性及可微性,所以它是一种可广泛应用于工程领域复杂优化问题的有效算法。同时它又是基于群体进化的算法,往往能收敛到全局最优解,所以它克服了一般的基于梯度优化方法局部收敛的一些缺点,在解决全局优化问题方面显示了极大的潜在优势。

实际遇到的数值优化问题绝大多数是有约束的,目前有多种方法求解约束优化问题,Michalewic 将现有的约束优化算法分成4类[2]:保存可行解方法、惩罚函数法、可行解优于不可行解的方法以及其他混合方法。

与其他优化方法一样,进化算法求解约束优化问题面临的主要问题是如何处理约束条件,罚函数法是最常用的方法。其基本思想是对任一违反了约束的个体,通过在其适应值函数上添加一个惩罚项的方法对它加以惩罚,从而使它的适应度降低。然后,通过选择算子把具有较好适应值的个体选择

到稿日期:2008-09-19 返修日期:2009-04-22 本文受国家自然科学基金(No. 60374063)资助。

胡一波(1974—),女,博士研究生,研究方向为进化算法及智能计算 CAD,E-mail:yibohu@126.com;王宇平(1961—),男,教授,博士生导师,研究方向为多目标优化、进化算法。

出来，然后进行杂交、变异来扩大搜索空间，生成下一代种群。从而将原来的约束问题变成了无约束问题来处理。罚函数法的主要优点是简单易行，主要的缺点是难以选择合适的罚因子，用以确定对不可行解的合适的惩罚力度。只有选择合适的罚因子，才能使进化算法获得好的搜索效果[1,2]。而罚因子的选取相当困难，若罚因子太小，则惩罚项在目标函数中所占比重较小，从而很难产生可行解；若罚因子太大，则会较早地收敛到某个局部最优点。而按目前罚函数的框架，如何确定罚因子还没有好的解决方法。本文提出了一种新的罚函数框架，在此框架内，罚因子很易被控制。

注意到一般的罚函数模型为适应度函数等于目标函数与惩罚项的和，这种模型的缺陷一方面在于此罚函数的罚因子难以确定，另一方面当目标函数值与惩罚函数值的差别很大时，此模型不能有效地区分可行解与不可行解。

为了克服这些缺点，本文提出了一种新的罚函数模型来处理约束优化问题。本文首先分别由目标函数及约束函数构造取值在[0,1]范围的目标满意度函数与约束满意度函数，用它们替代原目标函数及约束函数来评价个体，这样做的好处在于当目标函数值与惩罚函数值的差别很大时，此模型仍能有效地区分可行解与不可行解，然后我们将由这两个函数构造一个适应度函数。

另外，在适应度函数中，我们构造了动态的罚因子。构造罚因子时，考虑了当前种群的质量以及进化代数，从而克服了罚因子难以控制这一困难。当前种群的质量（种群的可行解所占比例）越高，罚因子越小；代数越大，罚因子越大。基于这些，本文提出了解决约束优化问题的一种新的进化算法，为了提高搜索效率，不仅设计了新的扩展算术杂交算子和自适应变异算子，而且在选择策略上，使群体中保持一定数量的非可行解，这也是目前认为是较好的一种策略。这样进化算法可从可行域内和可行域外两个方向进行搜索，使其更容易找到全局最优解。我们对5个常用的标准测试函数作了数据仿真实验，实验结果表明本文提出算法是有效的。

2 约束优化问题

考虑优化问题：

$$\min f(x)$$

$$s.t\ g_i(x)\leqslant 0 \quad i=1,2,\cdots,q$$

$$h_j(x)=0 \quad j=q+1,\cdots,m$$

$$x=(x_1,x_2,\cdots x_n)\in R^n$$

这里，$x\in F\subseteq S$，S 为目标函数的搜索空间，F 为可行域，$l(i)\leqslant x_i\leqslant u(i)$，$l(i)$，$u(i)$为常数 $i=1,2,\cdots,n$，$F=\{x\mid g_i(x)\leqslant 0, h_j(x)=0, \mid i=1,2\cdots,q, j=q+1,\cdots n\}$。

进化算法处理约束条件常见的方法为罚函数法，惩罚函数一般取个体 x 到可行域的距离的函数，通常用下面定义的 $f_j(x)$构造罚函数：

$$f_j(x)=\begin{cases}\max\{0,g_j(x)\}, & 1\leqslant j\leqslant q\\ |h_j(x)| & q+1\leqslant j\leqslant m\end{cases}$$

$f_j(x)$表示个体 x 在第 j 个约束的违反度。令罚函数 $p(x)=\sum_{j=1}^{m}f_j(x)$，它表示个体 x 违反约束的程度，也表示个体 x 到可行域的距离。

3 基于新罚函数模型的进化算法

3.1 目标满意度函数与约束满意度函数

当目标函数值与惩罚函数值的差别很大时，常用罚函数不能有效地区分可行解与不可行解，为了克服这一缺点，我们分别利用目标函数及惩罚函数定义如下的目标满意度函数与约束满意度函数：

$$f_1(x)=\frac{f_{\max}-f(x)}{f_{\max}-f_{\min}}$$

$$p_1(x)=\frac{p_{\max}-p(x)}{p_{\max}}$$

其中，$f_{\max}$与 $f_{\min}$为当前种群中的目标函数最大值与目标函数最小值，$p_{\max}$为当前种群中的惩罚函数最大值。显然，$f_1(x)$与 $p_1(x)$的值域均为[0,1]，且它们分别为 $f(x)$与 $p(x)$的减函数。也就是在同一代种群中，个体的函数值越小，其目标满意度越高；同样地，个体的约束违反度越小，其约束满意度越高。因此，我们可通过个体的目标满意度及约束满意度来评价其好坏。

3.2 新罚函数模型

一般的罚函数模型构造适应度函数为目标函数与惩罚函数的和，具体如下：

$$F(x)=f(x)+r_t\cdot p(x)$$

其中 $r_t>0$ 为罚函数的参数（罚因子），它体现了惩罚力度的大小。

使用此罚函数法模型的缺陷一方面在于很难选取恰到好处的罚因子。罚因子过大，虽然可以得到可行解，但解的质量可能很差（即目标函数值不够小），以至很难求得最优解；罚因子太小，说明对不可行解罚得不够，最终会得到一个不可行解；当然也不可能找到最优解。由于问题千差万别，无法确定通用的 r_t，也即用罚函数法解决约束优化问题时很难设置合适的参数因子。为此，利用当前种群的信息来设置动态罚因子，若种群中不可行解的数目较多，加大惩罚力度；反之，若种群中可行解的数目较多，则减少惩罚力度，加大对目标函数极小化的力度。这样，一方面有利于使搜索向可行解的区域移动，加快进化过程。另一方面可使罚因子易于控制。所以，我们在惩罚项中设置了当前种群中不可行解与可行解之比这一参数来控制惩罚的力度。

另一方面若利用目标函数与罚函数的和形式构造适应度函数，当目标函数值与罚函数值的差很大时，此模型不能有效地区分可行解与不可行解。此时，我们使用目标函数值满意度与约束满意度的乘积形式来替代目标函数值与惩罚项和形式构造适应度函数可解决这一问题。事实上，注意到对于控制个体优劣的两项指标：目标函数值满意度与约束满意度，用它们的乘积构造适应度函数比和构造适应度函数更能有效地使算法搜索到最优解。这可用如下的例子说明。如在进化过程中，当两个个体目标满意度与约束满意度的和相等时，保留函数值满意度与约束满意度更接近的个体可能更好。具体地，如目标函数值满意度与约束满意度均为0.5的个体 x^1 与目标函数满意度与约束满意度分别为1和0的个体 x^2，尽管两个个体的函数值满意度与约束满意度的和都是1。但显然，目标函数值满意度与约束满意度均为0.5的个体会更好。因此，我们用这两个函数的乘积来定义一种新的罚函数 $F(x)$ 如下：

$$F(x)=f_1(x)\cdot p_1(x)^{\frac{m}{n}\frac{2t-1}{t}}$$

其中，m 与 n 分别表示当前种群中不可行解与可行解的数目，

t 表示进化代数。我们注意到在以上适应度函数式中，我们加了一项函数 $g(t)=\frac{2t-1}{t}$，它是值域为[1,2]的单减函数。这样设计是基于以下事实。

(1)因为约束优化问题的最优解首先必须是可行解，所以在进化过程中，当种群中可行点与不可行点的数目相当时，设计适应度函数时约束满意度所占比重应高于函数值满意度所占比重，为此，设计 $g(t)$ 的值域为[1,2]。

(2)另外，在解决约束优化问题时，合理的进化过程应该是：进化初期应该保留较多的不可行解以增加种群的多样性，进化后期应保留较多的可行解以保证算法收敛。即进化过程中，惩罚不可行点的力度应随进化代数的增大而加强，为此，设计 $g(t)$ 为单减函数，此时由于 $p_1(x)\in[0,1]$，所以当 x 固定时，$p_1(x)^{\frac{m}{n}\frac{2t-1}{t}}$ 为 t 的增函数。

3.3 杂交算子

为了提高杂交算子搜索最优解的能力，在算术杂交算子中添加当前种群中最优个体的信息。

按适应度 $F(x)$ 的大小，用选择算子从种群 $P(t)$ 中选出 $N1$($N1$ 为偶数)个父母点(设父母点集合为 P)，对每对父母点 x^i 和 x^{i+1} 构造新的杂交算子，两个后代按下列方式产生。

$$\begin{cases} o^i=\frac{r_{11}}{r_{11}+r_{12}+r_{13}}\cdot x^i+\frac{r_{12}}{r_{11}+r_{12}+r_{13}}\cdot x^{i+1}+\frac{r_{13}}{r_{11}+r_{12}+r_{13}}\cdot x^g \\ o^{i+1}=\frac{r_{21}}{r_{21}+r_{22}+r_{23}}\cdot x^i+\frac{r_{22}}{r_{21}+r_{22}+r_{23}}\cdot x^{i+1}+\frac{r_{23}}{r_{21}+r_{22}+r_{23}}\cdot x^g \end{cases}$$

其中 r_{ij} $(i=1,2;j=1,2,3)$ 为[0,1]随机数，$x^i,x^{i+1}\in P(t)$，x^g 为当前种群中最好的个体，$o^i,o^{i+1}\in P1(t)$，$P1(t)$ 为杂交后代集合。

3.4 局部搜索算子与变异算子

局部搜索的步长应随着代数的增加而逐步缩小，在最优点的附近小范围内进行搜索，目的在于加大搜索效率，增强算法搜索最优解的性能。具体如下：

$\tilde{x}_j=x_j+(u_j-l_j)\cdot r_2$ if $r_1\leqslant 0.5$

$\tilde{x}_j=x_j-(u_j-l_j)\cdot r_2$ else

变异主要是为了使点跳出局部最优，保证可行域内的任何一点到其它点都是可达的。具体如下：

$$\hat{x}_j=x_j+\frac{(u_j-l_j)}{t}\cdot r_4 \text{ if } r_3\leqslant 0.5$$

$$\hat{x}_j=x_j-\frac{(u_j-l_j)}{t}\cdot r_4 \text{ else}$$

其中，$r_i(i=1,2,3,4)$ 是[0,1]间的随机数。x_j 是 $x(x\in P1(t))$ 的第 j 个坐标，l_j,u_j 分别是搜索空间中点的第 j 个坐标的下界与上界，$\tilde{x}_j,\hat{x}_j$ 分别是 $\tilde{x},\hat{x}$ 的第 j 个坐标，$P2(t),P3(t)$ 分别是局部搜索与变异后代集合。

3.5 选择算子

采用精英保留选择算子，具体操作如下：

从集合 $P(t)\cup P1(t)\cup P2(t)\cup P3(t)$ 中选择 M 个好的个体作为下一代种群 $P(t+1)$。

3.6 基于新罚函数模型的新的进化算法

step1 初始化，随机产生 M 个点的初始种群 $P(t)$，置 $t=0$

Step2 从 $P(t)$ 中按赌轮选择算子选出 $N1$ 个点作为父母点，设父母点集合为 P

Step3 对 P 执行上述杂交算子，得 $P1(t)$

Step4 对 $P1(t)$ 执行上述变异算子，得 $P2(t),P3(t)$

Step5 执行上述精英保留选择算子，得 $P(t+1)$

Step6 若满足终止条件，则停，否则令 $t=t+1$ 转 step2

3.7 数值实验与比较

从文献[3]中选择 6 个标准测试问题，分别记为 $P1-P6$，它们分别对应文献[4]中的 g03，g04，g06，g08，g09 和 g11，注意 P2，P4 是最大化问题，首先将目标函数两边乘以 -1 转化为最小化问题，再将得到的最小值乘以 -1，得到问题的最大值。对这些问题执行本文提出的算法，每个问题独立运行 10 次，记录了最优值、平均值、最差值、标准差以及最优点的违反度。这些结果分别记为：best，mean，worst，st-dev，and $p(x)$。把算法((记作 M1)的数值结果与 4 种最新算法的结果进行比较，这 4 种算法分别是：the homomorphous maps (记作：HM)[3]；SR[4]；ASCHEA[5]；SMES[6]。将 4 种算法分别记作 M2，M3，M4，M5。

表 1 列出了 6 个测试函数的最优值(optimal)，用我们的方法计算 4 个测试函数得到最优值、平均值、最差值、标准差。

表 1 M1 计算 P1—P6 所得的最优值、平均值、最差值、标准差

问题	optimal	best	mean	Worst	St. dev
P1	1.000	1.0000006	0.99999998	0.9999967	1.3562141e-7
P2	−30665.539	−30665.53926	−30665.5391	30665.53901	4.253218e−6
P3	−6961.8113	−6961.78924	−6961.53829	−6961.084285	6.3187e−2
P4	−0.095825	−0.09582504	−0.09582504	−0.09582504	0
P5	680.63	680.6315	680.634	680.6912	4.39485e−2
P6	0.750	0.74999918	0.74999929	0.74999960	1.35428e−7

从表 1 中可看出，除了问题 P3 和 P5，在已知的最优值的有效数位上，我们不但都达到了最优，而且我们给出的最优值保留了更多的有效数字。对于 P3，已知最优点代入验证带有违反度 $P^*=6.5616\times10^{-6}$，M1 得到最优值只比已知最优值大 0.05，不带任何约束违反度，对于问题 P4，从表 1 中可以看出本文提出的算法非常稳定地收敛到最优值，10 次独立运算都一致达到已知最优值 −0.09582504，而且比已知最优值有效数字多。对于问题 P1，P6 已知的最优解中含有无理数，所以可以断定由其它算法得到的最优值与我们的算法得到的最优值一样只是已知最优值的近似，对于相应的等式约束，一定存在违反度。该算法对于 P3，P4 得到的最优值分别为 1.0000006，0.74999918，比已知最优值 0.75，1 略好，这是因为存在一定的违反度，而违反度非常小，分别为：2.769319−7，5.7483e−7。其它算法也一定存在违反度，但未给出具体的违反度，无法比较。

另外，还可以从表 1 看出，M1 计算这 6 个函数不但最好值达到或接近已知最优值，平均值，甚至最差值都达到或接近已知最优值，这说明了本算法的稳定性。

表 2 列出了 6 个测试函数的最优值(optimal)及分别用 5 种算法计算它们得到的最优值、平均值、最差值。

表 2 M1—M5 计算 6 个函数得到的最优值、平均值、最差值、NA 表示未提供数据

问题		best	mean	worst
P1	M1	1.0000006	0.99999998	0.9999967
	M2	0.9997	0.9989	0.9978
	M3	1.000	1.000	1.000
	M4	1.0	0.99989	NA
	M5	1.000	1.000	1.000

	M1	−30665.53926	−30665.5391	−30665.53901
	M2	−30664.5	−30665.3	−30645.9
P2	M3	−30665.539	−30665.539	−30665.539
	M4	−30665.5	−30665.5	NA
	M5	−30665.539	−30665.539	−30665.539
	M1	−6961.78924	−6961.538290	−6961.084285
	M2	−6952.1	−6342.6	−5473.9
P3	M3	−6961.814	−6875.940	−6350.262
	M4	−6961.81	−6961.81	NA
	M5	−6961.814	−6961.284	−6952.482
	M1	−0.09582504	−0.09582504	−0.09582504
	M2	−0.0958250	−0.0891568	−0.0291438
P4	M3	−0.095825	−0.095825	−0.095825
	M4	−0.095825	−0.095825	NA
	M5	−0.095825	−0.095825	−0.095825
	M1	680.6315	680.634	680.6912
	M2	680.91	681.16	683.18
P5	M3	680.630	680.656	680.763
	M4	680.630	680.641	NA
	M5	680.632	680.643	680.719
	M1	0.74999918	0.74999929	0.74999960
	M2	0.75	0.95	0.75
P6	M3	0.750	0.750	0.750
	M4	0.75	0.75	NA
	M5	0.75	0.75	0.75

从表 2 可以看出，对于问题 P4，该算法在最优值方面优于 HM，平均值、最差值方面均优于其它算法。对于问题 P2，P3，P5，该算法在最优值方面与其它算法一样达到最优，在平均值、最差值方面明显优于 HM，与其它算法效果相同。对于问题 P1，P4，P6，该算法在最优值方面优于其它算法，在平均值、最差值方面略低于 SR，SMES，比其它的算法效果好。这些实验结果表明该算法是非常有效的。

结束语 提出了一种新的罚函数模型来处理约束优化问题。本文首先引入了函数值满意度函数与约束满意度函数，将这两个函数的乘积构造为适应度函数，并将反映种群质量信息参数的当前种群的可行解与不可行解的比值设置在适应度函数中，此模型不但能解决罚因子难以确定的问题，而且在进化过程中能有效地区分可行解与不可行解，从而使算法更加有效。

参 考 文 献

[1] Coello C A C. Constraint-handling using an evolutionary multiobjective optimization technique[J]. Civil Engineering and Environmental Systems, 2000, 17: 319-346

[2] Smith A E, Coit D W. Constraint handling techniques-Penalty functions[M]. Back T, Fogel D B, Michalewicz Z, eds. in Handbook of Evolutionary Computation, New York: Oxford Univ. Press and Inst. Physics, 1997

[3] Koziel S, Michalewicz Z. Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization[J]. Evolutionary Computation, 1999, 7: 19-44

[4] Runarsson T P, Yao X. Stochastic ranking for constrained evolutionary optimization[J]. IEEE Trans. on Evolutionary Computation, 2000, 4: 284-294

[5] Hamida S B, Schoenauer M. ASCHEA: New results using adaptive segegational constraint handling[C]// Proc. Congr. Evolutionary Computation. vol, 1. May 2002: 884-889

[6] Mezura-Montes E, Coello C A. A Simple Multimembered Evolution Strategy to Solve Constrained Optimization Problems[J]. IEEE Transsactions on Evolutionary Computation, 2005, 9(1): 1-17

(上接第 217 页)

执行得较好且能选择出最重要的特征。SVR-D1.1 优点不仅局限于预测结果的拟合度好，而且随着测试环境相关因子的改变，该方法可以动态选择最相关特征和允许预测模型的调整，表现出周期性更新模型的能力。这大大提高了它解决实际事务的预测问题的潜力。

参 考 文 献

[1] Gu Lichuan, Ni Zhiwei, Li shw. Forecasting Method on Pear Scab Based on Fusion Reasoning[C]// CIS2007 Proceedings. Harbin, China, IEEE Press, 2007: 401-404

[2] 高灵旺，陈继光，等. 农业病虫害预测预报专家系统平台的开发[J]. 农业工程学报，2006(10)：154-158

[3] 刘莉，李绍稳，等. 模糊聚类在砀山酥梨黑星病预测专家系统中的应用[J]. 农业网络信息，2004(2)：12-14

[4] Xiong J T, Xiong F L, Tu R S. Case-based reasoning in forecasting insect pests[A]// Xiong F L, Lee J K, Mizoguchi R, eds. Proc. of PACES'95 [C]. Beijing: Publishing House of Electronics Industry, 1995: 627-630

[5] Vapnik V N. The nature of statistical learning theory[M]. New York: Springer-Verlag, 1995

[6] Niu Dongxiao, Li Jinchao, Li Jinying, et al. Daily load forecasting using support vector machine and case-based reasoning[C]// Second IEEE Conference on Industrial Electronics and Applications, ICIEA2007. 2007: 1271-1274

[7] Cherkassky V, Ma Y. Practical selection of SVM parameters and noise estimation for SVM regression[J]. Neural Networks, 2004, 17(1): 113-126

[8] 刘向东，骆斌，陈兆乾. 支持向量机最优模型选择的研究[J]. 计算机研究与发展，2005，42(4)：576-581

[9] Muller K-R, Mika S, Ratsch G, et al. An introduction to kernel-based learning algorithms[J]. Neural Networks, IEEE Transactions, 2001, 12(2): 181-201

[10] Guyon I, Elisseeff A. An introduction to variable and feature selection[J]. Journal of Machine Learning Research, 2003(3): 1157-1182