计算机科学 2002Vol. 29№. 9

求解非线性规划问题的一种新演化算法*)

A New Evolutionary Algorithm for Solving Non-linear Planning Problem

高汉平[1,2] 康立山[1] 陈毓屏[1]

(武汉大学软件工程国家重点实验室 武汉 430072)[1] (黄冈师范学院计算机系 黄冈 438000)[2]

Abstract Based on the evolutionary algorithm in the literature, this paper puts forward some strategies for improving this algoritm by extending crossover and adding more mutations. By the strategies, the community scales can be contracted, the speed of convergence speeded up, and the calculating abilities increased. Through the calculation of five kind different type examples for non-linear planning, the paper confirms that this calculation is quite adaptable and effective in scanning the overall solution and getting better results than MATLAB software computing.

Keywords Genetic algoritm, Evolutionary algorithm, Optimization

1. 引言

演化算法是建立在生物进化论基础上的算法，生物体可以通过遗传和变异来适应外界环境，物生其类，传种接代，这是生物的独特本领。各种生物所生的子代基本上像父代，这就是遗传；而所生的子代又不完全像父代，这就是变异。世代相传，使得生物体不断进化。演化算法就是利用了生物进化的思想而发展起来的一种通用的问题求解方法。由于其所具有的本质并行性以及自组织、自适应和自学习等智能特征，目前已广泛用于优化问题[2~5]，它能克服传统优化方法的不足，寻找全局最优解。该方法是按照生物遗传进化世代繁衍的优生劣汰、演化出优良品种的特点，形成寻优的计算方法。特别是演化算法已成功地应用到那些难以用传统方法进行求解的复杂问题之中，从而成为一个引人注目的研究方向。

文[1]根据遗传算法存在群体爬山能力差的弱点，提出了一种求解不等式约束下函数优化问题的新算法。该算法群体爬山能力强，搜索效率高，结构简单，编程容易，易于实现。同时该算法也存在一些不足：由于该算法只杂交不变异，对不同优化问题，群体规模视问题而定。当群体规模太大时，每代杂交只替换一个最差个体，势必影响收敛速度。另外，由于只杂交不变异，完全依靠扩大种群规模来保持多样性，显然存在不足之处，而且只杂交不变异会使群体的多样性逐渐减少，造成有效个体丢失。当个别情况下有效个体不在初始化种群之中时，该算法搜索效率会降低。本文针对文[1]算法不足之处，引入了变异操作和提高杂交率，并增加淘汰压力，且每次淘汰多个较差个体等，使得改进后的算法更趋完善。5 种不同类型的非线性规划问题的数值结果表明，改进算法适应性强，收敛速度快，能快速获得全局最优解。

2. 不同的算法描述

为了说明新算法的特点，我们只介绍文[1] 中的演化算法。

2.1 文[1]中的演化算法

通常演化算法[6]中，人们普遍认为遗传算子包括选择、杂交、变异算子以及其它高级操作。选择是依据每个个体的适应值来选择新个体的过程，适应值越大，被选择的可能性越大，其子孙在下一代的个数越多。选择算子模拟了生物圈的"适者生存"的自然选择现象，它使群体的多样性逐渐减少。杂交算子是对选择的个体进行杂交，其主要作用是保持群体中有价值的信息，尽量让适应值高的个体保持下去，并使群体朝着适应环境的有利方向演变。变异算子则是为保持群体多样性设置的，它能降低群体多样性减少的速度，增加了遗传算法搜索全局最优解的能力。

文[1]认为遗传算法不仅是一种群体随机搜索算法，而且是一种空间搜索算法，并针对遗传算法存在群体爬山能力差和运行速度慢等弱点，提出了如下求解不等式约束下函数优化问题的演化算法：

问题：求 $x^* \in S$ 使得 $f(x^*) \leqslant f(x), \forall x \in S, S=\{x \in D \mid g_i(x) \leqslant 0, i=1,2,\cdots,l\}$，$D=\{X \in R^n \mid l_i \leqslant x_i \leqslant u_i, I=1,2,\cdots,n\}$，用逻辑函数 better($x_1$,$x_2$) 表示测试点 x_1 优于测试点 x_2。记

$$h_i(x)=\begin{cases}0, g_i(x) \leqslant 0\\ g_i(x), g_i(x)>0\end{cases} \qquad H(X)=\sum_{i=1}^{m} h_i(x)$$

定义：better(x_1,x_2)：

$$\text{better}(x_1,x_2)=\begin{cases}H(x_1)<H(x_2), true\\ H(x_1)>H(x_2), false\\ H(x_1)=H(x_2) \cap f(x_1) \leqslant f(x_2), true\\ H(x_1)=H(x_2) \cap f(x_1)>f(x_2), false\end{cases}$$

记 M 个测试点(即 M 个 n 维向量) $x_i=(x_{i1},x_{i2},\cdots,x_{in})^T$，$i=1,2,\cdots,M$。以 M 个测试点(向量)所张成的子空间 $V=\{x \mid X \in D, x=\sum_{i=1}^{M} a_i x_i\}$ 作为搜索子空间，其中 a_i 满足条件 $\sum_{i=1}^{M} a_i=1$，$-0.5 \leqslant a_i \leqslant 1.5$。

算法描述如下：

1)随机产生 N 个测试点，$P=\{x_1,x_2,\cdots,x_N\}$，$x_i \in D$；
2)求 Xbest 与 Xworst，使得 $\forall x \in p$ better(Xbest,x)和 better(x,Xworst)；
3)如果 better(Xworst,Xbest)，则转到 5
4)从 p 中随机选取 M 个 $x_1', x_2', \cdots, x_M'$，组成搜索空间 V；

*)本课题得到国家自然科学基金资助(编号：60133010,60073043,70671042)。高汉平 副教授，主要从事计算机软件和计算机科学理论、演化计算的研究。康立山 教授，博士生导师，主要研究方向为计算机科学理论、并行计算、演化计算。陈毓屏 副教授，主要研究方向为计算机软件和计算机科学理论、并行计算、演化计算。

5)在 V 中随机取一点 $\bar{x}=\sum_{i=1}^{M}a_i x_i'$;
6)如果 better($\bar{x}$,Xworst),则 Xworst=$\bar{x}$;
7)转到 2;
8)输出 Xbest;
9)结束。

其中:Xbest 和 Xworst 分别表示群体中最好和最差个体,$\bar{x}$ 表示杂交后的个体。

该算法的特点是:把 N 个测试点(个体)组成的集合看成是群体,把计算新的测试点的过程看成是 M 个父体杂交的过程。该算法群体爬山能力强,结构简单,编程容易,效率高,在科学研究和应用研究中具有广泛的应用前景。但该算法要求搜索空间中目标具有较好的局域性,这时只要较小 N 和 M 就能搜索到全局最优。否则,要求全局最优,必须将 N 和 M 取得很大才行,这样计算量就大,效率低。这是该算法的缺点。

2.2 改进算法

文[1]的演化算法的缺点,主要是只杂交不变异造成的,它使得群体的多样性逐渐减少。为了弥补上述缺陷,本文对该算法作了两点改进:①扩大杂交率,增加淘汰压力,一次淘汰多个较差个体;②增加变异算子,保持群体多样性。改进算法如下:

新算法:

{ t=0;
随机初始化种群 P(t)=(x_1,x_2,…,x_n),$x_i\in S$;
计算 P(t)中个体的适应值;
Xbest=select($\min_{1\leqslant i\leqslant n}f(x_i)$);
Xworst=select($\max_{1\leqslant i\leqslant n}f(x_i)$)
While(不满足终止准则)do
{从 P(t)中选取 M 个个体:X_{best},x_2',…,x_m 点决定搜索子空间 V(M 个中包括最好个体 Xbest);
计算 $\bar{x}=\sum_{i=1}^{m}a_i x_i'$,$\bar{x}\in S$// 多父体杂交
//其中,$\sum_{i=1}^{m}a_i=1$,$-0.5\leqslant a_i\leqslant 1.5$;
如果 better($\bar{x}$,Xworst)),则 Xworst=$\bar{x}$;
从 p(t)中随机选出 q 个点,再在 q 个中找出最差点 $\approx\!\!\!\!\!\!\sim\ x$;
如果 better($\bar{x}$,$\approx\!\!\!\!\!\!\sim\ x$),则 $\approx\!\!\!\!\!\!\sim\ x=\bar{x}$
计算适应值并排序;
将适应值最差的 h 个个体按式(1)或(2)进行变异操作($x_i\in S$,j=1,2,…,h),组成新一代;
Xbest=select($\min_{1\leqslant i\leqslant n}f(x_i)$),
Xworst=select($\max_{1\leqslant i\leqslant n}f(x_i)$);
t=t+1;
} //end while
}

其中:Xbest 和 Xworst 分别表示从 n 个个体适应值中选取最好和最差个体,而 Xworst=$\bar{x}$ 表示杂交后个体 $\bar{x}$ 替代最差个体。上述算法中:变异策略是由随机数 r=random(0,1),做如下运算:

$$x_{ij}'=\begin{cases}x_{ij}-(\bar{x}_j-x_{ij})r, & r\leqslant 0.5\text{ 时}\\ x_{ij}+(x_{ij}-\hat{x}_j)r, & 0.5<r\text{ 时}\end{cases}\qquad(1)$$

其中:$\bar{x}$ 和 $\hat{x}$ 分别表示第 j 个变量的上、下界,x_{ij}为变异后的变量值,即 $r\leqslant 0.5$ 和 $0.5<r$ 时分别让变量 x_{ij}跳跃式减少和增大,其跳跃值由随机数 r 控制。如果变量没有上、下界约束同时又无法确定其取值范围时,式(1)可改为:

$$x_{ij}'=\begin{cases}(1-r)x_{ij}, & r\leqslant 0.5\text{ 时}\\ (1+r)x_{ij}, & 0.5<r\text{ 时}\end{cases}\qquad(2)$$

3. 算例验证

用本文改进算法进行了仿真实验,并与文[2,6~9]中提供的结果进行比较,本文改进算法计算结果达到最优解或好于文献原最优结果。

3.1 目标函数

为了说明改进算法的性能,本文对下列求最大化(最小化)进行测试。

算例 1[4,7]:

$\max f_1(x_1,x_2)=21.5+x_1\sin(4\pi x_1)+x_2\sin(20\pi x_2)$

约束条件:$\begin{cases}-3.0\leqslant x_1\leqslant 12.1\\ 4.1\leqslant x_2\leqslant 5.8\end{cases}$

当点 $x^*=(11.631407,5.724824)$时,最优解:$f_1(x_1,x_2)=38.818208$

算例 2[8]:用 MATLAB 求解无约束非线性规划最优化问题

$\min f_2(x)=e^{-x}+x^2$

当点 $x^*=0.351700$ 时,最优解:$f_2(x)=0.827200$。

算例 3[9,10]:

$$\max f_3(x_1,x_2,x_3)=6x_1+4x_2+2x_3-3x_1^2-2x_2^2-\frac{1}{3}x_3^2$$

约束条件:$\begin{cases}x_1+2x_2+x_3\leqslant 4\\ x_i\geqslant 0,(i=1,2,3)\end{cases}$

算例 3 是二次线性规划问题包括由线性和二次项组成的目标函数与一组线性约束条件组成。该问题在无约束情况下有唯一最优解:$x^*=(1,1,3)$,$f_3(x_1,x_2,x_3)=8$。在有约束时则成为不可行解。目标函数的等值面为三维空间的椭圆面,其约束条件为 $x_1+2x_2+x_3=4$ 平面与椭球面的切点,即:当切点 $x^*=(0.875,0.625,1.875)$时,$f_3(x_1,x_2,x_3)=7.25$。

算例 4[8]:求解资金使用过程中最优方案。

$$\max f_4(x_1,x_2,x_3,x_4)=\sqrt{x_1}+\sqrt{x_2}+\sqrt{x_3}+\sqrt{x_4}$$

约束条件:

$$\begin{cases}x_1\leqslant 400\\ 1.1x_1+x_2\leqslant 440\\ 1.21x_1+1.1x_2+x_3\leqslant 480\\ 1.31x_1+1.21x_2+1.1x_3+x_4\leqslant 532.4\\ x_1,x_2,x_3,x_4\geqslant 0\end{cases}$$

当点 $x^*=(86.2,104.2,126.2,152.8)$时,最优解:$f_4(x_1,x_2,x_3,x_4)=43.1$

算例 5[8]:用 MATLAB 求解约束非线性规划最优化问题

$\min f_5(x_1,x_2)=(4x_1^2+2x_2^2+4x_1x_2+2x_2+1)e^{x_1}$

约束条件:$\begin{cases}1.5+x_{12}-x_1-x_2\leqslant 0\\ -x_1x_2-10\leqslant 0\end{cases}$

当点 $x^*=(-9.5474,1.0474)$时,最优解:$f_5(x_1,x_2)=0.0236$。

3.2 实验结果

为了说明改进算法的性能,本文对 5 个测试问题各进行了 10 次计算。计算中,算法中的 M 均取 10,N 的值随问题不同而不同。对于每个问题的 10 次计算结果都十分接近,这表明该算法具有很强的稳定性。而且计算结果或达到最优解或好于文献原最优结果。其中:算例 1 是选自文[2,6]中的例子,文[2]中是用二进制编码实现的,最优解是:当点 $x^*=(9.623693,4.427881)$时,最优解:$f_1(x^*)=35.477938$;而文[6]中是用实数编码实现的,最优解是:当点 $x^*=(11.631407,5.724824)$时,最优解:$f_1(x^*)=38.818208$;而本文均用实数编码实现的,最优解是:当点 $x^*=(11.625545,5.7250440)$时,最优解:$f_1(x^*)=38.850294$。此例说明用实数编码实现的结果,均优于用二进制编码实现的结果,而且本文

结果最优。算例3是选自文[8,9]中的例子，文[8]中最优解是：当点 $x^*=(0.875,0.625,1.875)$ 时，最优解：$f_3(x^*)=7.25$；文[9]中提供最优解为：$x^*=(0.874976,0.627653,1.869710)$，最优解：$f_3(x^*)=7.249980$；本文最优解是：当点 $x^*=(0.875019,0.624989,1.874998)$ 时，最优解：$f_3(x^*)=7.25$，结果显然优于文[9]并且达到真实的最优解，其它算例均优于有关文献提供最优解。特别值得一提是算例2和算例5是由当前世界上公认最好数值计算软件MATLAB软件计算的，原求解 $f_2(x)$ 最优解为：当点 $x^*=0.351700$ 时，最优解：$f_2(x^*)=0.827200$；而本文获得最优解为：当点 $x^*=0.351734$ 时，最优解：$f_2(x^*)=0.827184$；原求解 $f_5(x_1,x_2)$ 最优解为：当点 $x^*=(-9.5474,1.0474)$ 时，最优解：$f_5(x_1,x_2)=0.0236$；而本文获得最优解为：当点 $x^*=(-9.547368,1.047406)$ 时，最优解：$f_5(x_1,x_2)=0.023551$；此例说明本算法计算结果优于MATLAB软件计算结果。计算结果见表1。

表1 改进算法对5个函数测试结果

函数	原已知最优解 $x^*,f(x^*)$	本文算法最好解 $x^*,f(x^*)$
Max $f_1(x_1,x_2)$	$X^*=(11.631407,5.724824)$ $f_1(x^*)=38.818208$	$X^*=(11.625545,5.7250440)$ $f_1(x_1)=38.850294$
Min $f_2(x)$	$x^*=0.531700$ $f_2(x^*)=0.827200$	$x^*=0.351734$ $f_2(x^*)=0.827184$
Max $f_3(x_1,x_2,x_3)$	$x^*=(0.875,0.625,1.875)$ $f_3(x^*)=7.25$	$x^*=(0.875019,0.624989,1.874998)$ $f_3(x^*)=7.25$
Max $f_4(x_1,x_2,x_3,x_4)$	$X^*=(86.2,104.2,126.2,152.8)$ $F_4(x^*)=43.1$	$X^*=(88.04,105.72,123.4,153.1)$ $F_4(x^*)=43.1469$
Min $F_5(x_1,x_2)$	$X^*=(-9.5474,1.0474)$ $F_5(x^*)=0.0236$	$X^*=(-9.547368,1.047406)$ $F_5=0.023551$

结束语 在考察了遗传算法和文[1]中的演化算法后，本文提出了对该算法进行扩大杂交点和添加变异等改进策略，通过测试结果表明，本算法具有通用，精确，稳健，高效等特点，是求解优化问题较好方法。并对演化算法的整体性能的提高，同样具有重要作用。

参考文献

1 郭涛，康立山，等．一种求解不等式约束下函数优化问题的新算法．武汉大学学报，1999，45(5)：771～778

2 Michalewicz Z，Schoenauer M. Evolutionary algorithms for constrained parameter optimization problems[J]. Evolutionary computation，1996，4(1)：1～32

3 Michalewicz Z．Genetic Algorithm + Data Structures = Evolutionary Programms[M]. Berlin：Springer-Verlag，1992

4 [日]玄光男，程伟，著．汪定伟，唐加福，黄敏译．遗传算法与程序设计．北京：科学出版社，2000.5～11，100～200

5 刘勇，康立山，陈毓屏，著．非数值并行算法——遗传算法．北京：科学出版社，1995

6 潘正君，康立山，陈毓屏，著．演化算法，清华大学出版社，1998

7 [美]Z. 米凯利维茨著．周家驹，何险峰译．演化程序——遗传算法和数据编码．北京：科学出版社．2000.24～33

8 施光燕，董加礼．最优化方法，高等教育出版社，1999.3～9，27～31，90～93

9 瓦格纳著．邓三瑞，王元超，秋同译．运筹学原理与应用．北京：国防工业出版社，1992.412～420

10 黄豪，沈成武，雷建平．一种连续变量的遗传算法．武汉科技大学学报，1999，23(2)：123～125

（上接第12页）

17 Spertus E. ParaSite：Mining Structural Information on the Web. The Sixth Intl. WWW Conf.（WWW 97）. Santa Clara，USA，April 1997

18 Weiss R，et al. HyPursuit：A Hierarchical Network Search Engine that Exploits Content-Link Hypertext Clustering. In：Proc. the 7th ACM Conf. on Hypertext. New York，1996

19 Kleinberg J. Authoritative Sources in a Hyperlinked Environment. In：Proc. ACM-SIAM Symposium on Discrete Algorithms，1998

20 Page L，et al. The PageRank Citation Ranking：Bringing Order to the Web

21 Singhal A，Buckley C，Mitra M. Pivoted Document Length Normalization. In：Proc. of the 19th International ACM-SIGIR Conference on Research and Development in Information Retrieval（SIGIR96）

22 Brin S，Page L. The Anatomy of a Large-Scale Hypertextual Web Search Engine

23 Direct Hit. http：//www.directhit.com

24 Ginsberg A. A Unified Approach to Automatic Indexing and Information Retrieval. IEEE Computer，1993，8(5)：46～56

25 Efthimiadis E N. User Choices：A New Yardstick for the Evaluation of Ranking Algorithms for Interactive Query Expansion. Information Processing and Management，31(4)

26 Balabanovic M，Shoham Y，Yun Y. An Adaptive Agent for Automated Web Browsing. Journal of Image Representation and Visual Communication，1995，6(4)

27 Knoblock A，Arens Y，Hsu C. Cooperating Agents for Information Retrieval. In：Proc. of the Second Intl. Conf. on Cooperative Information Systems，Toronto，Canada，1994

28 Dreilinger D，Howe A E. Experiences with Selecting Search Engines using Meta-Search

29 Voorhees E M，Gupta N K，Johnson-Laird B. The Collection Fusion Problem. In：The Third Text REtrieval Conf.（TREC-3），Gaithersburg，MD 1994.95～104

30 Kantor P，et al. Combining the Evidence of Multiple Query Representations for Information Retrieval. Information Processing & Management，1995，31(3)：431～448

31 Callan J，Zhihong L，Croft W B. Searching Distributed Collections With Inference Networks. 18th Annual Intl. ACM SIGIR Conference on Research and Development in Information Retrieval，Seattle，WA 1995.21～28

32 Selberg E，Etzioni O. The MetaCrawler Architecture for Resource Aggregation on the Web

33 Gauch S，Wang Guijun，Gomez M. ProFusion：Intelligent Fusion from Multiple，Distributed Search Engines

34 Galescu L，Ringger K. Augmenting Words with Linguistic Information for N-gram Language Models

35 Smeaton A F，Crimmins F. Using A Data Fusion Agent for Searching the WWW