Matlab遗传算法优化工具箱(GAOT)的研究与应用*

周正武[1],丁同梅[1,2],田毅红[2,3],王晓峰[2,4]
(1.广东省技师学院,广东 博罗 516100; 2.天津大学 机械工程学院,天津 300072;
3.承德技师学院,河北 承德 067400; 4.郑州职业技术学院,河南 郑州 450121)

摘 要:介绍遗传算法的基本原理和 Matlab的遗传算法优化工具箱(GAOT),分析了优化工具函数。探讨 Matlab遗传算法工具箱在参数优化和非线性规划中的应用。通过优化实例,说明遗传算法是一种具有良好的全局寻优性能的优化方法。用 Matlab语言及 Matlab语言编制的优化工具箱进行优化设计具有语言简单、函数丰富、用法比较灵活、编程效率高等特点。

关键词:遗传算法;Matlab工具箱;优化
中图分类号:TH164 **文献标识码**:A **文章编号**:1007-4414(2006)06-0069-03

The study and application of genetic algorithm optimization toolbox in Matlab

Zhou Zheng-wu[1], Ding Tong-mei[1,2], Tian Yi-hong[2,3], Wang Xiao-feng[2,4]
(1. *College of high-grade technician, Boluo Guangdong* 516100, *China;*
2. *College of mechanical and engineering, Tianjin university, Tianjin* 300072, *China;*
3. *Chengde college of high-grade technician, Chengde Hebei* 067400, *China;*
4. *Zhengzhou vocational college of technology, Zhengzhou Henan* 450121, *China)*

Abstract: The paper introduces genetic algorithm (GA) and Matlab genetic algorithm optimization toolbox and analyses the optimization toolbox function. The function optimization problem of parameter optimization and nonlinear has been given to demonstrate that genetic algorithm is a better global optimization method. The optimization design in Matlab and Matlab optimization toolbox have simple language, abundant functions, flexible method and high programming efficiency.

Key words: genetic agolrithm; matlab toolbox; optimization

遗传算法(GA)是一类借鉴生物界自然选择和遗传原理的随机优化搜索算法。其主要特点是群体搜索策略和群体中个体间的信息交换、搜索不依赖于梯度信息。遗传算法几乎渗透到从工程到社会科学的诸多领域,广泛用于组合优化、机器学习、自适应控制、规划设计和人工生命等领域,是 21世纪有关智能计算中的关键技术之一。Matlab语言是一种高效率的用于科学工程计算的高级语言,它的语法规则简单、更贴近人的思维方式,通俗易懂。Matlab语言有着丰富的各种工具箱,Matlab的优化工具箱提供对各种优化问题的一个完整的解决方案。遗传算法优化工具箱就是其中之一。采用 Matlab遗传算法优化工具箱,不仅具有简单、易用、易于修改的特点,且为解决许多传统的优化方法难以解决的参数优化、非线性、多峰值之类的复杂问题提供有效的途径,为遗传算法的研究和应用提供很好的应用前景[1]。

1 遗传算法(GA)

1.1 遗传算法原理[2]

遗传算法(Genetic Agolrithm)是由美国 Michigan大学的 JohnHolland教授在 20世纪 60年代提出,它是模拟生物在自然环境中的遗传和进化过程而形成的一种自适应全局优化概率搜索算法。遗传算法是将问题的求解表示成"染色体",将其置于问题的"环境"中,根据适者生存的原则,从中选择出适应环境的"染色体"进行复制,即再生(reproduction, selection),通过交叉(crossover)、变异(mutation)两种基因操作产生出新一代更适合环境的"染色体"群,这样一代代不断改进,最后收敛到一个最适合环境的个体上(当然也有其它的收敛准则),求得问题的最佳解。由于最好的染色体不一定出现在最后一代,开始时保留最好的染色体,如果在新的种群又发现更好的染色体,则用它代替原来的染色体,进化完成后,这个染色体可看作是最优化的结果。

1.2 遗传算法的步骤[2]

遗传算法与传统搜索算法不同,它以适应度(fitness)函数(目标函数)为依据,通过对种群(popldAfon)中的所有个体实施遗传操作,实现群体内个体结构重组的迭代过程搜索法。选择(selection)、杂交(crossover)、变异(mutation)构成遗传算法的 3个主要遗传操作。参数编码、初始群体的设定、适应度函数的设计、遗传操作设计、控制参数设定等要素组成遗传算法的核心内容。其主要步骤有:

(1) 编码 由于 GA不能直接处理空间的数据。必须通过编码将其表示成遗传空间的基因型串结构数据。

(2) 初始种群的生成 随机产生 N 个初始串结构数据,每个串结构数据称为一个个体,也称为染色体(chromosome),N 个个体构成一个种群。

(3) 适应度评估检测 GA在搜索进化过程中一般不需要其它外部信息,仅用适应度来评估个体或解的优劣,并作为以后遗传操作的依据。对于不同的问题,适应度的定义方式也不同。

(4) 选择 选择或复制是为从当前个体中选出优良的个体,使其有机会作为父辈为下一代繁殖子孙。个体适应度越

* **收稿日期**:2006-10-08
作者简介:周正武(1968-),男,吉林人,讲师,主要从事机械专业的教学工作。

高,被选择的机会就越多。

(5) 杂交 杂交操作是遗传算法中的特色操作,将群体内的每个个体随机搭配成对,对每一对个体。按杂交概率交换它们之间的部分染色体。通过杂交可得到新一代个体,新个体组合了其父辈个体的特性。

(6) 变异 首先在群体中随机选择一个个体,对选中的个体以一定的概率随机地改变串结构数据中某个位的值。同生物界一样,GA中变异发生的概率很低,通常在 0.001～0.01之间取值。

1.3 遗传算法的流程图[4]

遗传算法的基本流程如图 1所示。

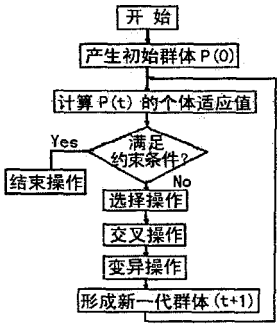

图 1 遗传算法操作流程图

1.4 系统参数的选取所遵循的原则[2]

(1) 种群数目 N 种群数目会影响 GA的有效性。N太小,GA会很差或根本找不出问题的解,因为太小的种群数目不能提供足够的采样点;N太大,会增加计算量,使收敛时间延长。一般种群数目在 20～160之间比较合适。

(2) 交叉概率 P_c 此参数控制着交叉操作的频率。P_c太大,会使高适应值的结构很快破坏掉;P_c太小,搜索会停滞不前。一般 P_c取 0.25～0.75。

(3) 变异概率 P_m 它是增大种群多样性的第二因素。P_m太小,不会产生新的基因块;P_m太大,会使 GA变成随机搜索。一般 P_m取 0.01～0.20。

2 Matlab遗传算法工具箱(GA Toolbox)

遗传算法在应用过程中必须要编制大量的程序进行优化计算,利用 Matlab遗传算法优化工具箱编程是最有效的方法和途径。

2.1 遗传算法函数[2]

遗传算法工具箱 (GAOT)包括许多实用的函数,这些函数按照功能可分为以下几类:主界面函数、选择函数、演化函数、其它的终止函数、二进制表示函数、演示程序等。

2.2 遗传算法函数的参数[2]

Matlab的遗传算法工具箱核心函数 GAOTV5其主程序 ga.m提供遗传算法工具箱与外部的接口。在 Matlab环境下,执行 ga.m并设定相应的参数,就可完成优化。其格式如下:

(1) function[pop] = initializega(num, bounds, eevalFN, eevalOps, options) -初始种群的生成函数; [输出参数]pop -生成的初始种群; [输入参数]num -种群中的个体数目; bounds -代表变量的上下界的矩阵; eevalFN -适应度函数; eevalOps -传递给适应度函数的参数; options -选择编码形式 (浮点编码或是二进制编码)[precision F-or-B],如 precision -变量进行二进制编码时指定的精度 F-or-B -为 1时选择浮点编码,否则为二进制编码,由 Precision指定精度。

(2) function [x, endPop, bPop, traceInfo] = ga(bounds, evalFN, evalOps, startPop, opts, ……termFN, termOps, selectFN, selectOps, xOverFNs, xOverOps, mutFNs, mutOps)。[输出参数] x、endPop、bPop、traceInfo; [输入参数]bounds、evalFN、evalOps、startPop、opts、termFN、termOps、selectFN、selectOps、xOverFNs、xOverOps、mutFNs、mutOps。

3 应用实例[3]

3.1 无约束优化问题

利用遗传算法计算函数 $f(x) = x + 10 * \sin(5x) + 7 * \cos(4x)$的最大值,其中 $x \in [0\ \ 9]$。[分析]选择二进制编码,种群中的个体数目为 10,二进制编码长度为 20,交叉概率为 0.95。变异概率为 0.08。采用 GOAT的程序编写如下:

(1) 编写目标函数文件 optm,文件存放在工作目录下。

```
function[sol, eval1] = opt(sol, options);
x = sol(1);
eval = x + 10 * sin(5 * x) + 7 * cos(4 * x);
```

(2) 生成初始种群,大小为 10。

```
initPop = initializega(10, [0 9], 'opt');
```

(3) 调用遗传算法主函数 ga.m程序。

```
[x endPop bPop trace] = ga([0 9], 'opt', [], initPop, [1e-6 1 1], 'maxGenTerm', 25, 'normGeomSelect', [0.08], ['arithXover'], [2], 'nonUnifMutation', [2 25 3]);
```

经过 25次遗传迭代,运算结果的最优解为:

$$X^* = 7.8566;\ f_{max} = 24.8554。$$

遗传算法一般用来取得近似最优解。另外,遗传算法的收敛性跟其初始值有关。随机取不同的初始群体,结果可能会稍有差异,但多次执行即可得到近似最优解。

3.2 有约束优化问题

考虑如下问题:

$$\mathrm{Min} f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$$

$$\text{s.t.}\quad g_1(x) = x_1 - 2x_2 + 11 \geq 0$$

$$g_2(x) = x_1^2/4 - x_2^2 + 1 \geq 0$$

[分析]本例中存在 2个不等式约束,需要将有约束问题转换成无约束问题来求解。对于多种用遗传算法满足约束的技术,工程中常用的策略是惩罚策略,通过惩罚不可行解,将约束问题转换为无约束问题。惩罚项的适值函数一般有加法和乘法两种构造方式,本例采用加法形式的适值函数,惩罚函数由两部分构成,可变乘法因子和违反约束乘法。种群中的个体数目为 100,实数编码,交叉概率为 0.95,变异概率为 0.08遗传算法求的是函数的极大值,因此,在求极小值问题时,需将极大值问题转换为极小值问题求解。采用 GOAT的程序编写如下:

编写目标函数文件 fitm,文件存放在工作目录下。

```
(1) function[sol, eval1] = fit(sol, options)
xl = sol(1);
x2 = sol(2);
rl = 0.1;
```

```
r2 = 0. 8;
%约束条件。
g1 = x1 - 2 *  x2 + 1;
g2 = x1. ^2/4 - x2. ^2 + 1:
%加惩罚项的适值
If(g1 > = 0) &(g2 > = 0)
eval = (x1 - 2). ^2 + (x2 - 1). ^2
else
eval = (x1 - 2). ^2 + (x2 - 1). ^2 + r1 * g1 + r2 * g2;
eval = - eval;
end
```

(2) 设置参数边界,本例边界为 2。

```
bounds = ones(2, 1) * [ - 1, 1];
```

(3) 调用遗传算法函数。

```
[X endPop bPop trace] = ga(bounds, 'Min');
```

(4) 性能跟踪。

```
plot(trace(:, 1), trace(:, 3), 'r- ');
hold on
plot(trace(:, 1), trace(:, 2), 'b* ');
xlabel('Generation'); ylabel('Fittness');
Legend('解的变化','种群平均值的变化');
```

经过 100次遗传迭代,运算结果为: $x = [1\ 1]$;此时极值 $eval(x) = 1$; $g_1(x) = 0$; $g_2(x) = 0.25$,显然最优解满足约束条件。

从以上两例寻优结果看出:遗传算法对非线形、多峰值函数完全可以避免陷入局部最优,而找到全局最优解,从解的变化可看出迭代次数不需太多就能达到最优解。

4 结 论

随着遗传算法作为一种全局优化算法在各个领域的应用越来越广泛,Matlab的遗传算法优化工具箱提供了一个标准、可扩展、简单的算法。利用 Matlab强大的矩阵运算能力,对传统优化算法难以实现全局最优解的非线性、多峰值函数最值问题进行优化。结果表明,对于非线形、多峰值函数的寻优问题,遗传算法不仅不会陷入局部最优点,而且有较强的搜索能力、收敛速度快、精度也较高,用法也比较灵活,完全避免一般优化算法的局部最优、收敛较慢等问题的不足。

参考文献:

[1] 刘万林,张新燕,晁 勤. Matlab环境下遗传算法优化工具箱的应用[J]. 新疆大学学报(自然科学版),2005,8(3):357 - 360.
[2] 飞思科技产品研发中心编著. Matlab 6. 5辅助优化计算与设计[M]. 北京:电子工业出版社,2003.
[3] 于 玲,贾春强. Matlab遗传算法工具箱函数及应用实例[J]. 机械工程师,2004,11(6):27 - 28.
[4] 谢勤岚,陈 红,陶秋生. Matlab遗传算法工具箱的设计[J]. 计算机与数字工程,2003(3):37 - 40.

(上接第 63页)

```
X[#5]    (X向工进至 D5处)
G91 G02 X[ - #21] Z[ - #8] R[#9]    (波纹面初始段圆弧)
M98 P1001 A - B 3/C - D -  ……  (调用子程序,参数传递)
G03 X[ - #21] Z[ - #8] R[#9] F30    (尾段波纹车削)
G01 X[ - #7]    (进刀至法兰中心)
G90 G00 Z5    (绝对值方式,退刀)
G28 X50 Z30    (返回参考点)
T0606    (换 6号刀精车波纹)
M03 S500    (调整转速)
G00 X[#3] Z5
G01 X [#4] F100
      Z[ - #20] F30
      X[#5]
G91 G02 X[ - #21] Z[ - #8 - 0. 1] R[#9]
M98 P1001 A - B - C - D - ……
G03 X[ - #21] Z[ - #8] R[#9]
G01 X[ - #7]
G90 G00 Z5
G28 X50 Z30
M09
M05
M30
```

子程序(Sub program)

```
%1001
WHILE #24 LE #11
G03 X[ - #10] Z[ - #8] R[#9] F30
G02 X[ - #10] Z[ - #8] R[#9]
#24 = #24 + 1
ENDW
M99
```

3 结束语

宏指令编程作为数控加工编程方法的一种补充,具有很强的实用性。尤其对于轮廓为函数化曲线的零件及尺寸系列化零件的数控加工程序编制而言,更体现出程序精炼、编辑修改方便的特点。虽然用 G代码编制子程序也解决同样的问题,但宏指令编程允许使用变量、算术和逻辑运算及条件转移,使编制相同加工操作的程序更方便、容易,将相同加工操作编为通用程序。基于此,在波纹法兰加工实践中,利用上述特点编制加工程序,缩短编程调试时间,提高加工效率。

参考文献:

[1] 金卫国,顾晓艳. 宏程序的应用[J]. 机械工人,2002(12):26 - 27.
[2] 余英良. 数控加工编程及操作[M]. 北京:高等教育出版社,2005.
[3] 吴 沁,包 磊,王景阳. TND360数控车床上加工变送器法兰的关键技术[J]. 机械研究与应用,2004,17(3):33 - 34.