一种遗传算法适应度函数的改进方法

张思才 张方晓
(中国工程物理研究院结构力学研究所 四川 绵阳 621900)

摘 要 针对简单遗传算法中线性适应度函数随进化过程恒定不变的缺点,提出一种可随进化代数动态调整的非线性适应度函数。以典型的遗传算法测试函数为算例,分别以 Goldberg提出的线性拉伸方法[1]与文中提出的改进遗传算法进行计算。计算结果表明文中提出的动态适应度函数对简单遗传算法的改进有较明显的效果。

关键词 遗传算法 适应度函数 优化计算

A MODIFIED METHOD TO FITNESS FUNCTION OF GENETIC ALGORITHMS

Zhang Sicai Zhang Fangxiao
(*Institute of Structural Mechanics, China Academy of Engineering physics, Mianyang, Sichuan, 621900, China*)

Abstract Aiming at the shortcoming of the simple genetic algorithm (GA) with the linear fitness function unfit for evolutionary process, a modified genetic algorithm with the nonlinear fitness function which can adapt to evolutionary process of algorithm is presented in this dissertation. GA with linear scaling method proposed by Goldberg[1] and modified GA in this paper respectively calculates the genetic algorithm's testing functions. The comparison between the results obtained by above-mentioned algorithms indicates that the nonlinear adaptive fitness function is effective for improving the simple genetic algorithm's performance.

Keywords Genetic algorithm Fitness function Optimization computation

0 引 言

遗传算法(genetic algorithms 简称 GA)是模拟生物进化过程中,“适者生存,优胜劣汰”规律而无需函数梯度信息的自适应全局搜索算法[1]。遗传算法倍受各行设计者青睐之原因在于:这类随机算法能够同时处理 N 个设计变量,有利于实现并行操作,提高多变量优化问题的计算效率;能够跳出局部最优解而做到全局搜索。遗传算法依靠选择操作模拟自然界中的“适者生存,优胜劣汰”这一过程,即选择操作来引导算法的搜索方向,而选择操作是以个体的适应度作为确定性指标,从当前群体中选择适应值高的个体以生成交配池。如此必然造成群体中基因信息的丢失,使群体中个体平均相似度增加,最终造成遗传算法早熟。文献[2]研究表明优化参数配置不当,遗传算法可能会出现不收敛的情况。为使遗传算法运用于工程结构优化领域,诸多学者对遗传算法做出不少改进[3~5]。大多数改进思想都是对遗传算法中的适应度函数、交叉概率和变异概率等方面进行改进,尤其以文献[5]中提出的自适应遗传算法为代表,其思想就是建立在个体适应度基础之上。此外,遗传算法本身是以激励机制——适应度函数为基础,故对适应度函数的改进应该是最基本的、最有效的改进方式。

Goldberg在简单遗传算法的基础上提出线性拉伸方法以扩大个体之间的差异,但是涉及到常数的选取问题。本文将对常数的选取进行讨论,为避免人为因素对算法的影响,本文提出一种非线性的动态适应度函数,以典型的遗传算法测试函数为考题验证本文提出的适应度函数的有效性与可行性。

1 遗传算法

遗传算法寻优的本质是以群体中各个体的适应度为依据,通过选择、交叉等操作反复迭代,不断寻求出适应度较好的个体,最终得到问题的最优解。适应度函数是评价群体中个体好坏的标准,是模拟自然选择的唯一依据。从而适应度函数选取的优劣直接影响遗传算法的收敛速度及能否找到最优解。

2 适应度函数的选取

首先是遗传算法运行初期阶段,群体中或许会出现少数适应度极好的个体,最终这些个体可能会充斥整个群体,使用于产生新个体作用较大的交叉操作失去作用,从而使得群体多样性降低,遗传算法提前收敛到某个局部最优解。因此,适应度函数的选取应尽量地避免早熟现象,即降低适应度较高的个体与其它个体适应度之间的差异,限制其复制数量以维护群体多样性。

其次是运行后期阶段,群体越来越集中,个体之间的差异减小,相互之间的竞争力也随即减弱。这必然造成个体被选择到下一代中的概率接近,使进化过程失去竞争力,退化为随机选择过程。因此,适应度函数的选取也应该克服这种退化现象,使算法在运行后期阶段能够扩大最佳个体适应度与其它个体适应度之间的差异,提高个体之间的竞争性。

收稿日期:2004-02-05。张思才,硕士,主研领域:结构设计及优化算法研究。

Goldberg的线性拉伸适应度函数为[1]：

$$F^*(X)=\frac{(c-1)F_{avg}}{F_{max}-F_{avg}}F(X)+\frac{F_{max}-cF_{avg}}{F_{max}-F_{avg}}F_{avg} \quad (1)$$

式中，$F^*(X)$为经过线性拉伸得到的适应度函数；$F(X)$为目标函数通过线性变换得到的适应度函数，且与常数的选取密切相关[1]；F_{avg}为当前群体个体平均适应度；F_{max}为当前群体中最佳个体的适应度；c为常数，建议取 1～2。显然，适应度函数的好坏直接取决于常数 c的选取。

对于求极小值问题，针对上述问题，本文构造具有随进化代数动态调整的非线性适应度函数为：

$$F^*(X)=\frac{[\sqrt[m]{n}]}{G(X)} \quad (2)$$

式中，$F^*(X)$为非线性适应度函数；$G(X)$是经过无约束处理后得到的目标函数；$[Q]$ 表示取值不大于 Q 的整数值；$m=1+\ln N$，N为根据问题复杂程度设定的最大进化代数；n为当前的进化代数。

文献[6]建议进化代数根据问题取 100～500代。由(2)式可知，此进化代数范围内的适应度函数都可以动态调整个体适应度。另外，文献[7]研究表明进化代数与个体位串长度有关。考虑个体位串长度及算法运行耗费，最大进化代数 N 设定为 200。

3 算　例

下面以典型的遗传算法测试函数验证文中构造的适应度函数之有效性、可行性，比较采用不同适应度函数的遗传算法得到的寻优结果。

3.1 Schaffer函数 *F6*

Schaffer函数 $F6$的具体形式为(3)式，其局部最优点很多，最优点是 $f6(0,0)=0$。

$$\min f6(x_1,x_2)=0.5+\frac{\sin^2\sqrt{x_1^2+x_2^2}-0.5}{[1+0.001\times(x_1^2+x_2^2)]^2}$$
$$x_i\in[-100,100] \quad (i=1,2) \quad (3)$$

设计变量以 15位二进制数表示，分别以不同 c值的线性拉伸遗传算法(LGA)与改进遗传算法(MGA)计算，其收敛过程分别如图 1所示，得到的近似最优值均为 1.9×10^{-5}。图 2给出了简单遗传算法(SGA)、LGA及MGA的进化情况，从图 2中可以看出采用非线性适应度函数的遗传算法收敛，且运行效率明显提高。表 1给出了不同 c值的 LGA收敛到函数最优或近似最优点对应的起始收敛代数。

表 1　线性拉伸适应度

序号	c	$F2$		$F6$	
		最优值	起始收敛代数	最优值	起始收敛代数
1	1.05	0.000003	151	0.000019	165
2	1.2	0.000002	104	0.000019	186
3	1.4	0.000001	187	0.000019	146
4	1.55	0.000001	126	0.000019	171
5	1.7	0.000003	88	0.000019	163
6	2.0	0.000000	123	0.000019	192
本文		0.000001	13	0.000019	129

3.2 De Jong函数 *F2*

De Jong函数 $F2$是一个二维函数，其函数具体形式为(4)式，唯一的全局最优点为(1,1)。

$$\min f2(x_1,x_2)=100(x_1^2-x_2)^2+(1-x_1)^2$$
$$x_i\in[-2.048,2.048] \quad (i=1,2) \quad (4)$$

设计变量以 10位二进制数表示，分别以不同 c值的 LGA与MGA计算，其代数情况分别如图 3所示。MGA优化得到的近似最优值为 10^{-6}，表 1给出了不同 c值的 LGA得到的函数最

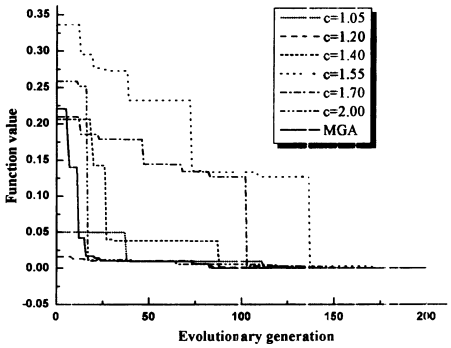

图 1　不同常数 c对应的优化结果

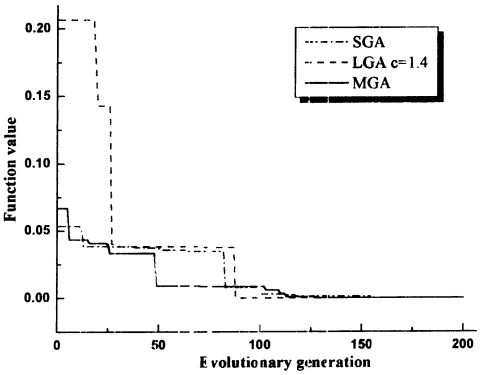

图 2　非线性适应度函数对应的收敛情况

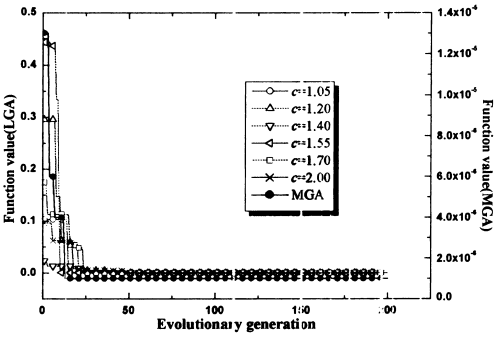

图 3　不同常数 c对应的优化结果

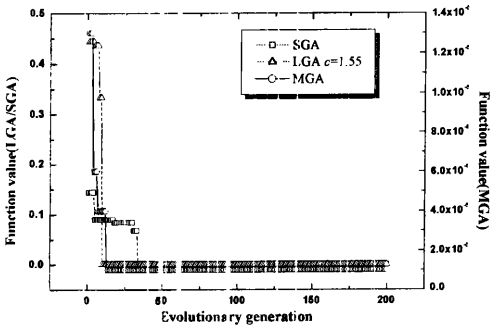

图 4　三种不同适应度函数对应的优化情况

优或近似最优点及对应的起始收敛代数。比较两种适应度函数改进方法得到的结果，表明 MGA 在最接近函数最优点的情况下可以明显提高运行效率。图 4给出了采用不同适应度函数的遗传算法的进化情况，从图 4中可以看出采用非线性适应度函数的遗传算法收敛，且运行效率明显提高。

3.3 讨 论

图 1~图 4表明文中提出的改进适应度函数方法使得遗传算法较 SGA、LGA在运算效率上有明显的提高，同时图 1与图 3及表 1表明 LGA 的收敛情况与常数 c密切相关。从表 1可以看出 $F2$函数以 LGA(c=2.0)经过 123代开始收敛到函数最优点，表明函数最优值的获得直接与设计者的决策密切相关。

4 结 论

针对简单遗传算法中的线性适应度的不足，本文提出了采用非线性适应度的改进遗传算法。以典型的遗传算法测试函数作为考题，计算结果表明改进遗传算法是有效可行的，同时也表明非线性适应度的遗传操作更适应进化过程。文中提出的非线性适应度函数可供改善遗传算法寻优性能参考。

参 考 文 献

[1] Goldberg D. E. Genetic algorithms in search, optimization and machine learning. Addison Wesley Publishing Reading Mass, 1989.

[2] 弥丽娜、陈治飞、孙昌志，“一种随机并行算法——Alopex算法的改进”，《沈阳工业大学学报》，2000, 22(4): 296~299.

[3] 马钧水、刘贵忠、贾玉兰，“改进遗传算法搜索性能的大变异操作”，《控制理论与应用》，1998, 15(3): 404~408.

[4] 李大卫、王梦光，“一种改进的混合遗传算法”，《信息与控制》，1997, 26(6): 449~454.

[5] Srinivas M., Patnaik L. M., Adaptive Probabilities of Crossover and Mutation in Genetic Algorithms. IEEE Transaction System, Man and Cybernetics, 1994, 24(4): 656~667.

[6] 王小平、曹立明，遗传算法——理论、应用与软件实现[M]，西安：西安交通大学出版社，2002.

[7] 张思才、张方晓，“遗传算法在离散变量结构优化设计中的应用”，《西南交通大学学报》，2003(2): 146~150.

(上接第 9页)

柄 HANDLE_B。当协作方 A要求继续对这条直线进行修正时，协作方 B是无法根据 HANDLE_A 找到 HANDLE_B并对其进行修改的。(一种替代的方法是协作方 B根据消息中的其它参数在整个数据库中查找匹配的对象，这样存在着如下问题：(1)搜索费时，减低了实时性。(2)可能有不止一个对象参数相同)。因此，我们需要为整个协同网络设计一个消息映射网，以防止这种情况的发生。

消息映射网算法描述：

```
ReceiveMessage( )
提取 Operation Type, Handle
if(Operation Type=Create){ /*创建 * /
        CreateLine( )
        LocaHandle=GeHandle( )
        CreateDictionary(Handle, LocaHandle) /*双向映射表 * /
}
else{       /*修改或删除 * /
        LocaHandle=SearchDic(Handle)
        If(! LocaHandle) LocaHandle=Handle
        Modify(LocaHandle) or Erase(LocaHandle)
}
```

以 A、B、C三方协同为例来说明整个协同过程：

(1) 协作方 A创建一条直线后，通过 ARX 获得直线句柄 HANDLE_A，按照通信协议组织成消息发送给协作方 B和 C。

(2) 协作方 B、C接收到服务器转发过来的消息，在本地创建完一条与协作方 A一样的直线后，获得这条直线在本地数据库的句柄，分别是 HANDLE_B和 HANDLE_C。然后为本方建立一个双向映射表。B、C方建立的映射表如图 3所示。

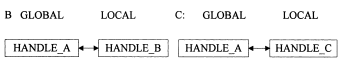

图 3 映射表

(3) 当协作方 B修改这条直线时，可通过 ARX 获得这条直线的句柄 HANDLE_B，然后搜索本地的映射表(LOCAL -> GLOBAL)，通过 HANDLE_B找到 HANDLE_A，并把 HANDLE_A 填入消息格式中的 HANDLE字段，由服务器转发给协作方 A和 C，A和 C接收到消息后，取出 HANDLE字段，然后分别搜索本地的映射表(GLOBAL -> LOCAL)，根据上面的映射算法，找到本地的句柄 HANDLE_A和 HANDLE_C，然后修改句柄相对应的直线。

4 结 论

按照操作语义构建消息的协同办法，为在异构 CAD系统中进行协同提供了重要思路。在我们的设计中，就成功实现了 AutoCADR14与 AutoCAD2000之间的基本协同。凭借这种协同方法，可以把更多的设计细节屏蔽在动态库中，从而极大地减轻了协同设计客户端与服务器的工作。

但是这种协同方法需要对 CAD的每个操作细节比较熟悉，以保证在各个协同方都能够完整无误的把操作重现出来。而且建立一个所有操作的状态转换表也是一个比较繁琐的过程。此外，由于动态链接库是利用 CAD的二次开发工具来实现的，因而采用这种方法进行协同的 CAD必须具备较完善的二次开发接口。

参 考 文 献

[1] Kvan T. Collaborative design: what is it? [J] AUTOMATION IN CONSTRACTION, 2000, Vol 9 No 4, 409~415.

[2] 史美林、向勇、杨光信，计算机支持的协同工作理论与应用[M]，北京：电子工业出版社，2000.

[3] 胡毓宁、吴旭光、周登文，“基于 STEP的 CAD协同工作[J]”，《计算机工程与应用》，2002, 03, 120~122

[4] 彭维、莫蓉、范晓坤、张铁昌，“基于消息通信的同步协同设计技术[J]”，June 2001, Vol 22, No 6

[5] 宋延杭、王川、李永宣，ObjectARX 实用指南—AutoCAD 二次开发[M]，北京：人民邮电出版社，1999.

[6] 邵俊昌、李旭东，AutoCAD ObjectARX2000开发技术指南[M]，北京：电子工业出版社，2000.

[7] LislieMonplaisir An integrated CSCW architecture for integrated product/process design and development[J]. Roboticsand computer Integrated manufacturing, 1999, 15, 145~153.

[8] RezayatM. The Enterprise2Web portal for life2cycle support[J]. Computer aided Design, 2000, 32(2): 85~96.

[9] Ping YiChao Yu-chouWang A data exchange framework for networked [J]. CAD/CAM Computers in Industry, 2001, 44, 131~140.