基于分支插桩的改进型评价模型及其应用

高峰青,王晓军
(南京邮电大学 计算机学院,江苏 南京 210003)

摘 要:软件测试是软件开发过程中极其重要的一环,提高软件测试的自动化程度对于确保软件开发质量、降低软件开发成本非常重要,而提高生成测试用例的自动化程度又是提高测试自动化程度的关键。当今用遗传算法生成测试数据是一种行之有效的方法,Korel 所提出的"分支函数"插装法在一定程度上优化了算法的执行效率。文中在此基础上,结合节点覆盖的思想,设计出一个能更好指导算法执行过程的模型。实验证明该适应度模型比单纯的插桩方式的遗传算法生成测试用例更加高效。

关键词:遗传算法;分支插桩;节点覆盖;适应度函数;测试用例
中图分类号:TP31 **文献标识码**:A **文章编号**:1673-629X(2012)10-0098-03

A Promoted Mode Based on Branch-stubbing & Applications

GAO Feng-qing, WANG Xiao-jun
(School of Computer, Nanjing University of Posts & Telecommunications, Nanjing 210003, China)

Abstract: Software test is an important step during software development. Improving the automation of software testing can increase the robustness of software and decrease the cost of development. The key of improving the automation ability of testing is improving the automatic test data generation. Now the genetic algorithm (GA) is a efficient way for producing test cases. Branch-stubbing proposed by Korel, is aidant for GA to a certain extent. Considering the node-covering, show a promoted mode. From the experimentation, see that it takes a good effect, and is prior to branch-stubbing method.

Key words: genetic algorithm; branch-stubbing; node-covering; fitness function; test case

0 引 言

遗传算法(GA, Genetic Algorithm),也称进化算法。遗传算法是受达尔文的进化论的启发,借鉴生物进化过程而提出的一种启发式搜索算法。借鉴生物进化论,遗传算法将要解决的问题模拟成一个生物进化的过程,通过复制、交叉、突变等操作产生下一代的解,并逐步淘汰适应度函数值低的解,增加适应度函数值高的解。这样进化 N 代之后,就可能会进化出适应度函数值很高的个体。遗传算法具有很好的全局搜索能力,可以不受搜索空间的限制,有效地处理多变量和复杂函数关系的优化问题,很适合用于解决服务组合这样的 NP 难问题,并且其特有的遗传算子(选择、交叉、变异)增加了算法搜索过程的灵活性。Korel[1] 首次依据分支函数最小化的思想将遗传算法引入到软件测试用例的生成领域。到目前为止,人们对遗传算法自动生成测试数据已进行了广泛而深刻的研究,应用到了多个领域[2]。

文献[3]将遗传算法运用到了黑箱测试中。

文献[4]提出了一种自适应的评价函数生成方法,一定条件下改进了遗传算法的收敛效率。

文献[5~7]则提出了利用控制依赖图路径生成测试数据的方法。

文献[8]构造了一种以树结构的适应度函数的改进算法,同时使用了自适应的交叉和变异算子生成下一代。

文献[9]则提出了多条路径的测试数据综合生成方式。

文献[10]详细阐述了编码的几种常用方式、提出了"分支函数叠加法"的评价函数构造方法。这种方法有一定的局限性,虽然能准确反应两个不同个体执行了同一路径的差异性,但对执行了两个不同路径的差异性评价没有一个合理的反映。

1 文中的主要工作

文中采用面向路径覆盖的测试用例生成方法,在

收稿日期:2012-02-09;**修回日期**:2012-05-14
基金项目:国家科技支撑计划项目(2007BAH17B04)
作者简介:高峰青(1986-),男,硕士研究生,主要研究方向为分布计算技术与应用;王晓军,副教授,硕士研究生导师,主要研究领域为分布计算技术与应用。

研究各种遗传算法的改进方法上,进一步设计出一个更加合理、高效的适应度计算模型,通过对参数的编码、适应函数的构建完整地实现出一个测试用例的自动生成模型。并且通过结果性能比较说明该方法在各方面的优越性。

2　改进的遗传算法生成测试数据

2.1　参数编码

在运用遗传算法求解问题时要求把问题的解空间表示为某种编码形式,主要有二进制编码和浮点数编码[8]。对实际的被测试单元一般有一个或多个参数,因此需要将多个参数进行位编码。常用的有多参数级联定点映射和多参数交叉映射[11],针对实际问题,文中采用参数级联定点映射编码形式。

2.2　适应度函数构建

适应度函数是联系遗传算法和实际求解问题的唯一桥梁。适应度函数的优劣将直接影响遗传算法的执行效率。本节将设计一个能更好地评价测试数据优劣的适应度函数。

Korel 所提出的"分支函数[1]"插装的做法,即在程序单元内部指定的逻辑路径所经过的每个分支点前插入一个实值函数 $f_i(x_1, x_2, \cdots, x_n)$,其中 $x_1, x_2, \cdots, x_n$ 为被测单元的形参变量。当一组测试数据驱动被测单元执行时,这些实值函数即被计算出来,这些 f_i 的取值将反映在当前这组测试数据下,被测单元的实际执行路径与指定逻辑路径的偏离程度,这样就可将评价函数定义为这些 f_i 的联合表达形式:

$$F = F[f_1(x_1, x_2, \cdots, x_n), f_2(x_1, x_2, \cdots, x_n), \cdots, f_m(x_1, x_2, \cdots, x_n)]$$

F 的值便成为评价形参变量的实际值(生成的测试数据)的优劣的一个很好的尺度。关于 F 应采取的具体形式,文献[2]选用了较为简单的连加形式:$F = \Sigma f_i$,对算法的执行效率有一定的提高。

但是,简单的叠加不能真正全面地反映出测试数据的优劣。一般对于测试数据基本需要解决两个问题:

1. 两个不同的个体(测试数据)执行了相同的路径,如何评价这两个数据的优劣;

2. 两个不同的个体执行了不同的测试路径,如何评价这两个数据的优劣。

分支函数叠加可以很好地解决第一个问题,但对于第二个问题,函数值的实际情况反映的不够精确。文中结合一个具体示例予以说明:

如图 1 所示代码,可以画出相应的分支结构图如图 2 所示(图中节点编号为代码行号)。对于这个示例,不难看出,该示例程序中共包含有 5 条路径。假定

```
boolean isRightTriangle (int x ; int y ; int z)
{
1:if (x > 0 & & x < 100)
2:if (y > 0 & & y < 100)
3:if (z > 0 & & z < 100)
4:      if (x * x + y * y == z * z){
5:printf ("Right triangle!") ;
6:            return true;}
7: return false;
}
```

图 1　例程 1

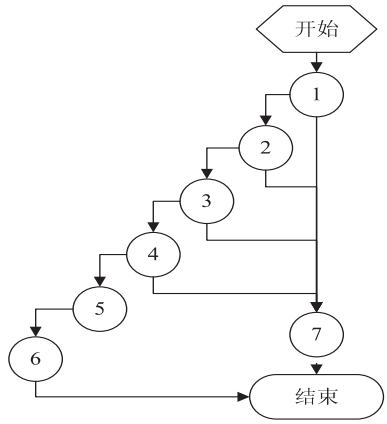

图 2　例程 1 的分支结构

当前的目标是找到能覆盖程序中所有真分支的测试数据(目标路径 s 为 1-2-3-4-5-6,图中粗体表示)。

经分支函数和评价函数插装后的程序如下:

```
boolean isRightTriangle (int x ; int y ; int z)
{ float f1 ,f2 ,f3 ,f4 ,F ;
f1 = max (1 - x , x - 99) ; / * 分支函数插桩 * /
1:if (x > 0 & & x < 100)
f2 = max (1 - y , y - 99) ; / * 分支函数插桩 * /
2:if (y > 0 & & y < 100)
f3 = max (1 - z ,z - 99) ; / * 分支函数插桩 * /
3:if (z > 0 & & z < 100)
f4 = abs (x * x+y * y-z * z) ; / * 分支函数插桩 * /
4:if (x * x + y * y == z * z){
5:printf ("Right triangle!") ;
if (f1 < 0) f1 = 0 ;
if (f2 < 0) f2 = 0 ;
if (f3 < 0) f3 = 0 ;
F = f1 + f2 + f3 + f4 ; / * 评价函数插桩,F 返回给遗传算法包 * /
6:return true;}
if (f1 < 0) f1 = 0 ;
if (f2 < 0) f2 = 0 ;
if (f3 < 0) f3 = 0 ;
F = f1 + f2 + f3 + f4 ; / * 评价函数插桩,F 返回给遗传算法包 * /
7: return false; }
```

假设有两个测试数据 D1 和 D2,D1 执行的路径 L1:1-2-3-7,D2 执行的路径 L2:1-2-3-4-7。如果这时 D1 的插装函数 $f3$ 的值小于 D2 的插装函数 $f4$,根据“分支函数叠加法”的原则认为 D1 比 D2 更“好”,但是,显然可以看出,L2 比 L1 更接近目标路径,因为 L2 覆盖了目标路径的 4 个节点,而 L1 只覆盖了 3 个节点。

为了更好地解决路径覆盖的合理性评价,引入测试数据执行路径的节点覆盖率定义:如果一次执行路径经历了 n 个节点块,其中有 m 个节点块属于目标路径,则 (m/n) 定义为测试数据执行路径的覆盖率 ρ(节点块是指程序中连续可执行的语句合并后的节点)。

此时,还需要将叠加的插装函数 F 在当前群中的所有个体中进行规格化 F',使其值属于[0,1]范围中。则令 $P=\eta * F/F_{max}+(1-\eta)*(1-\rho)$(其中 η 为系统调节参数,试验中取 0.45;F_{max} 为 F 可能的最大值,目的在于将 F 归一化),则 $P=0$ 时路径即为目标路径,P 值越大,测试路径与目标路径越远。用 P 函数则可以更好地评价测试数据的优劣。

2.3 个体选择

传统的遗传算法中,采用“轮盘赌”的方法选择个体,适应度高的个体获得高的选择概率。这里为了提高个体被选中的概率,采用“加速选择法”[2],同时将最大适应度的个体直接复制到下一代,直到有新的、更好的个体替换它为止。

2.4 交叉算子

最简单的交叉算子是单点交叉。改进的方式是增加交叉点,即多点交叉,但由于交叉点是随机产生确定,当交叉点分布不均匀时,交叉操作对位串结构的改变将限于局部串。为解决这个问题,可以采用均分交叉的方式,即先将串分为几段,然后在每段上随机产生交叉点,这样就保证了交叉点在整个位串上的均匀分布了。

2.5 变异算子

在遗传算法中,变异可防止群体进化停滞不前或冻结,若无变异,则新群体中的测试数据值可能局限于初试化数据的特征区域。对于二进制编码来说,变异的基本过程为按一定的突变概率 P_m 将位串的某一位或某几位的 1 变为 0 ,将 0 变为 1。

3 实验及结果分析

为了验证上述模型效率,结合例程 1(整型参数)和下面的例程 2(实型参数,目标路径 1-3-5-7-9-10,例程 2 的分支结构见图 4),对不同参数、不同路径复杂度的程序测试数据生成做了实验,并就传统的分支函数叠加评价的遗传算法、改进后的评价遗传算法的搜索效率进行了比较。对每种方法做 15 次试验,参数设置如下:例程 1(种群大小:60;个体长度:10;变异概率:0.008;$\eta=0.45$;总迭代次数:5000;随机范围[0,10000]),例程 2(种群大小:60;个体长度:20;变异概率:0.008;$\eta=0.45$;总迭代次数:200;随机范围[-10,10],误差范围:0.0001(由于 $\prod$ 为无理数,编码解码过程存在误差)),然后取其平均值,以找到解的次数和运行时间为主要参考指标,实验结果如表 1 所示。

```
voidwhichQuadrant (double arc){
1:   if(sin(arc)>0&&cos(arc)>0)
2:printf("第一象限");
3:   else if(sin(arc)>0&&cos(arc)< 0)
4:printf("第二象限");
5:   else if(sin(arc)< 0&&cos(arc)< 0)
6:printf("第三象限");
7:   else if(sin(arc)< 0&&cos(arc)>0)
8:printf("第四象限");
9:  else
10:printf("坐标轴");
}
```

图 3 例程 2

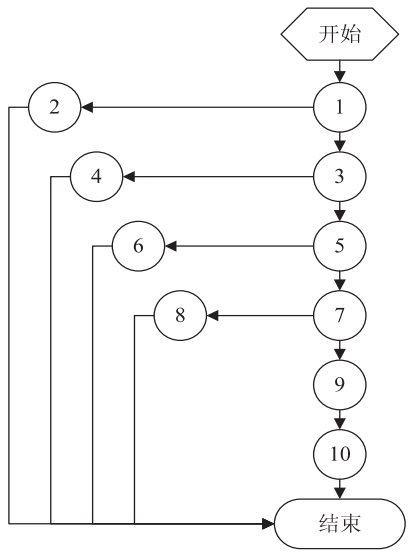

图 4 例程 2 的分支结构

表 1 搜索效率比较

	例程 1			例程 2		
实验指标	平均运算代数/代	平均运行时间/s	找到解的次数/次	平均运算代数/代	平均运行时间/s	找到解的次数/次
改进的评价模型	1081	21	7	7	19	12
分支函数叠加模型	2262	20	2	18	22	3

上述结果恰好验证了评价函数改进的优越性,使遗传算法的执行效率得到更进一步的提高。改进后的模型可以更有效地指导遗传算法向着最优化的方向快速进化。

(下转第 104 页)

选用更大的数据集 Libras Movement 进行测试,该数据集共有 360 个样本点,每个样本点是 90 维浮点型数据,样本点共分为 15 类。为了让实验结果更加明显,给出程序运行 100 次的结果:原算法花费 228031 毫秒时间,改进的算法花费 206200 毫秒时间。聚类过程之前的去除孤立点和初始聚类中心确定操作使得初始聚类中心更加符合实际的情况,这样才会使得聚类过程的迭代次数减少,在总的运行时间上才会少于原算法的运行时间。该实验结果表明,改进的算法在孤立点和初始聚类中心的处理时间花费是值得的,尤其对于大数据集来说。

由此可以得出,改进的算法在准确率、稳定性和收敛速度方面都有很大的提高。

4 结束语

K-Means 算法作为经典的聚类算法,对数据集是紧凑的并且簇与簇之间明显分离的情况聚类效果较好,但是它受孤立点和初始中心影响较大。文中首先运用统计方法对孤立点进行检测,然后提出一种新的基于分布的初始聚类中心确定策略。实验结果表明,改进的算法在准确率、稳定性和收敛速度方面都有很大的提高。

参考文献:

[1] 周卫星,廖 欢. 基于 K 均值聚类和概率松弛法的图像区域分割[J]. 计算机技术与发展,2010,20(2):68-70.

[2] Han Jiawei, Kamber M. Data Mining Concepts and Techniques [M]. Beijing: China Machine Press, 2007.

[3] Wu Xindong, Kumar V, Quinlan J R, et al. Top 10 algorithms in data mining[J]. Knowl. Info. Syst., 2008(14): 1-37.

[4] MacQ J. Some methods for classification and analysis of multivariate observations[C]//Proc of the 5th Berkeley Symposium on Mathematical Statistics and Probability. Berkeley, USA: [s. n.], 1967: 281-297.

[5] Tou J. Pattern Recognition Principles[M]. USA: Addison Wesley, 1974.

[6] Linde Y, Buzo A, Gary R. An Algorithm for Vector Quantizer Design[J]. IEEE Trans on Communication, 1980, 28(1): 84-95.

[7] Chomicki J, Godfrey P, Gryz J, et al. Skyline with Presorting Theory and Optimization[C]//Proc of the International Conference on Intelligent Information Systems. Wroclaw, Poland: [s. n.], 2005: 216-225.

[8] 黄震华,向 阳,张 波,等. 一种进行 K-Means 聚类的有效方法[J]. 模式识别与人工智能,2010,23(4):517-521.

[9] 贾俊平. 统计学[M]. 北京:中国人民大学出版社,2006.

[10] 朱颢东,钟 勇,赵向辉. 一种优化初始中心点的 K-Means 文本聚类算法[J]. 郑州大学学报(理学版),2009,41(2):30-30.

[11] 汪 中,刘贵全,陈恩红. 一种优化初始中心点的 k-means 算法[J]. 模式识别与人工智能,2009,22(2):299-304.

[12] 袁 方,周志勇,宋 鑫. 初始聚类中心优化的 k-means 算法[J]. 计算机工程,2007,33(3):66-66.

(上接第 100 页)

4 结束语

上述试验结果显示,文中提出的适应度函数能够使遗传算法更高效的收敛,比单纯的“分支函数叠加法”[12]有更好的效率。遗传算法是一个随机算法,初始种群的适应度值将会影响算法的收敛性和解的适应性。在初始化种群的时候,文中给出了两种算法来生成初始群体,提高种群的适应度。鉴于文中存在的一些不足,如参数的类型、范围等问题,还有待进一步思考,对于参数范围如何自适应处理,多路径覆盖如何高效产生等相关工作,简化了服务组合的工作量,尚需要进一步完善。这是下一步研究的重点。

参考文献:

[1] Korel B. Automated Software Test Data Generation[J]. IEEE Trans. on Software Eng., 1990, 16(8): 870-879.

[2] 荚 伟. 基于遗传算法的软件结构测试数据生成技术研究[J]. 北京航空航天大学学报,1997,23(1):36-40.

[3] 荚 伟. 遗传算法在软件测试数据生成中的应用[J]. 北京航空航天大学学报,1998,24(4):434-436.

[4] 潘祖烈. 基于遗传算法的黑箱测试用例自动生成模型[J]. 计算机工程,2008,34(9):205-210.

[5] 赵 明. 基于遗传算法的测试用例生成工具研究[J]. 计算机工程,2005,31(13):151-153.

[6] 伦立军. 基于遗传算法的测试数据生成研究[J]. 计算机工程,2005,31(23):82-84.

[7] 金 虎. 基于面向路径的遗传算法的测试用例自动生成[J]. 计算机工程,2007,33(3):21-23.

[8] 高海昌. 改进的遗传算法在测试数据自动生成中的应用[J]. 系统工程与电子技术,2006,28(7):1077-1081.

[9] 王元珍. 产生多条路径上测试用例的改进遗传算法[J]. 计算机工程,2006,32(13):196-205.

[10] Chaiyarataiia N. Recent Developments in Evolutionary and Genetic Algorithms: Theory and Applications[C]// Second International Conference on (Conf. Publ. No. 446) Genetic Algorithms in Engineering Systems: Innovations and Applications. [s. l.]: [s. n.], 1997: 270-277.

[11] Zeng L Z, Boualem B, Ngu A H H, et al. QoS-aware middleware for web services composition[J]. Software Engineering, 2004, 30(5): 311-327.

[12] 陈 亮,孙 敏. 基于免疫遗传算法的 Web 服务组合方法[J]. 计算机工程,2010,36(10):226-230.

基于分支插桩的改进型评价模型及其应用

万方数据 WANFANG DATA 文献链接

作者：高峰青， 王晓军
作者单位：南京邮电大学 计算机学院,江苏 南京 210003
刊名：计算机技术与发展
英文刊名：Computer Technology and Development
年，卷(期)：2012(10)

本文链接：http://d.g.wanfangdata.com.cn/Periodical_wjfz201210027.aspx