浙江理工大学学报，2019，41(1)：65-71
Journal of Zhejiang Sci-Tech University
DOI:10.3969/j.issn.1673-3851(n).2019.01.010

基于相似路径的动态引导测试数据生成策略

包晓安[1]，徐海霞[1]，张娜[1]，吴彪[2]，钱俊彦[3]
(1.浙江理工大学信息学院，杭州 310018；2.山口大学东亚研究科，日本山口 753-8514；
3.桂林电子科技大学计算机科学与工程学院，广西桂林 541004)

摘　要：为了有效提高满足路径覆盖的测试数据质量，提出一种利用相似路径间启发信息的引导测试数据生成策略。首先，对初始测试数据与路径节点间的覆盖情况进行分析，区分出难易覆盖路径；其次，设计了一种路径相似度的计算方法，分析得出路径相关启发信息，并将该启发信息用于遗传算法寻优过程中；然后，构造带有权重影响因子的适应度评价函数，结合保留精英个体思想，设计自适应遗传算子并定向引导个体交叉变异；最后，将该策略应用于多个基准程序和工业程序，并与 Ahmed 方法、多路径覆盖方法和 EGA 方法比较。仿真实验结果表明，该策略在运行时间、路径覆盖率和已有测试数据的利用率上均有优势。

关键词：启发信息；遗传算法；动态引导；测试数据生成
中图分类号：TP311.5　　**文献标志码**：A　　**文章编号**：1673-3851 (2019) 01-0065-07

Dynamic guided test data generation strategy based on similar path

BAO Xiaoan[1], *XU Haixia*[1], *ZHANG Na*[1], *WU Biao*[2], *QIAN Junyan*[3]
(1.School of Information Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China; 2.Graduate School of East Asian Studies, Yamaguchi University, Yamaguchi 753-8514, Japan; 3.School of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: In order to effectively improve the quality of the test data that satisfies the path coverage, a guided test data generation strategy based on heuristic information between similar paths is proposed. Firstly, the difficult coverage paths were distinguished by analyzing the coverage between the initial test data and the path nodes. Secondly, a path similarity calculation method was designed to get path-related heuristic information, and the heuristic information was used for genetic algorithm optimization. Thirdly, a fitness evaluation function with the weight impact factor was constructed. In combination with the individual idea of retaining elites, the adaptive genetic operator was design, and the individual cross-variation was guided directionally. Finally, this strategy was applied in multiple benchmarks and industrial processes, and compared with Ahmed method, multipath coverage method and EGA method. The simulation results showed that the strategy had great advantages in running time, path coverage and utilization of existing test data.

Key words: heuristic information; genetic algorithm; dynamic guidance; test data generation

收稿日期：2018－08－21　　网络出版日期：2018－11－01
基金项目：国家自然科学基金项目(61502430,61562015)；广西自然科学重点基金项目(2015GXNSFDA139038)；浙江理工大学 521 人才培养计划项目
作者简介：包晓安(1973－)，男，浙江东阳人，教授，硕士，主要从事软件测试、智能信息处理方面的研究。
通信作者：张娜，E-mail：zhangna@zstu.edu.cn

0 引 言

在软件测试领域,测试数据的自动生成技术一直是研究热点[1-2]。已有研究表明,基于语句、分支等许多软件测试数据的生成问题,均可以转化为基于路径覆盖的测试数据生成问题[2]。因而,研究关于路径覆盖测试数据的生成问题很有必要。近年来,遗传算法普遍应用于基于路径覆盖的测试数据自动生成领域[3]并取得了一些成果。Ahmed 等[4]通过转化测试数据生成问题为结合遗传算法求解多个目标的优化问题,发现覆盖多条目标路径的测试数据只要运行一次遗传算法即可生成。张岩等[5]在面向目标路径覆盖的测试数据生成框架的基础上,将每代种群中目标路径经过的节点个体总数作为计算个体贡献度的依据,并且将个体贡献度作为权重来提升稀有数据的适应值。Pachauri 等[6]保留种群中精英个体以及上一代种群的执行记忆,使用覆盖给定路径中的兄弟路径的个体替换最差个体,以避免种群向低劣的方向退化。Gong 等[7]通过对路径间的相关性分析,进而对多路径覆盖的测试数据生成问题进行建模,将复杂问题简化为多个目标的优化问题,以此提高问题求解效率。丁蕊等[8]通过转换控制流图为关键点图,提出了关键点路径表示法,并且给出了软件测试数据快速生成的完整模型。

尽管对基于标准遗传算法[9]的路径覆盖测试数据生成方法的研究已经得到了一些有效的解决方法,但依旧存在一些缺陷:a)在搜索最优解的过程中,难以确保算法收敛至全局最优解[10];b)没有考虑路径间的相互关系,已覆盖路径的测试数据没有得到充分利用[11];c)缺乏有效的适应度评价函数来合理选择测试数据;d)随机的遗传算子的设定影响了遗传算法后期进化的效率。

针对标准遗传算法在路径覆盖应用上的缺陷,本文提出一种基于相似路径的动态引导进化生成测试数据策略。该策略首先区分出难易覆盖路径,将难覆盖路径作为目标路径进行求解;然后设计了一种路径相似度的计算公式,分析出难易覆盖路径间的启发信息并用于替代遗传算法的部分初始种群;最后考虑了分支权重对种群适应度的影响,分别为各影响因子赋予权重,构造带有权重影响因子的适应度评价函数,并以此设计自适应遗传概率,定向引导个体交叉变异,以快速得到满足路径覆盖的高质量测试数据。

1 路径相关性

本文首先对被测程序静态分析,给出相应的控制流图。对控制流图中的判定节点使用 Z 路径覆盖[8]方法处理成两个分支的选择结构,将其中多层循环嵌套结构转化成多个选择的并列结构,从而得到满足路径覆盖的全部理论路径。为了更好地表述路径之间、路径与节点间的关系,本文对文中涉及的路径的基本符号进行定义:

路径集 P:记待测程序为 G,由程序 G 中控制流图转化而来的全部理论路径集合为 $P=\{P_1,P_2,\cdots,P_s\}$,其中 s 表示路径集 P 中的路径总个数。

节点集 N:$N=\{N_1,N_2,\cdots,N_n\}$ 表示为某条路径上的节点集合,其中:n 表示路径 P_i 经过的节点总个数,集合 N 中的节点序列为程序 G 在运行时控制流经过的节点序列。那么第 i 条路径 P_i 上的第 j 个节点表示为 Pi_{Nj}。

测试数据集 C:$C=\{C_1,C_2,\cdots,C_m\}$ 表示为随机生成的初始测试数据的集合。其中 m 为测试数据的个数。

1.1 难易覆盖路径区分

结合随机生成的初始测试数据和程序 G 的路径信息进行分析,记录测试数据对全部理论路径的覆盖情况,若测试数据集 C 覆盖了分支路径 P_i 上的节点 N_j,则将该节点的覆盖矩阵值记为 1,若该节点没有被覆盖,则记为 0。那么,测试数据对路径节点的覆盖关系如式(1)所示:

$$\varphi(C_q,Pi_{Nj})=\begin{cases}1, C_q \text{ 覆盖了 } Pi_{Nj}\\ 0, C_q \text{ 未覆盖 } Pi_{Nj}\end{cases} \tag{1}$$

其中:q 表示第 q 个测试数据。

本文根据二进制矩阵(1)中初始测试数据对路径节点的覆盖情况,对难易覆盖路径作出如下划分:

a)易覆盖路径:路径上的每个节点均被初始测试数据覆盖,即其覆盖矩阵中的每个节点的覆盖值均为 1(例如 6 个节点的路径,其节点覆盖值表示为 111111),则将该路径划分为易覆盖路径。

b)难覆盖路径:路径上存在节点没有被初始测试数据覆盖,即其覆盖矩阵中的节点覆盖值存在 0(节点覆盖值为 0 的个数大于等于 1),则将该路径划分为难覆盖路径。

保留每条易覆盖路径的初始测试数据,作为被测程序最终的部分测试数据,将难覆盖路径作为目标路径,其测试数据运用优化后的遗传算法寻优求解,不断地迭代进化直至找出最佳路径覆盖的测试

数据。

1.2　路径相似度分析

目前,研究路径间启发式信息的方法较多,有从覆盖顺序路径节点角度,通过考虑上下节点的测试数据覆盖情况从而找到中间节点的测试数据,或者从覆盖组合路径的角度,考虑兄弟分支节点间的关系得到相应的启发信息,但这些方法比较模糊,前期需要大量的人工研究,没有具体的定量的计算方法。

本文为了便于寻找难覆盖路径的启发信息,设计一种路径相似度的计算方式。对于程序 G 中易覆盖路径 P_i 与难覆盖路径 P_j,将这两条等长路径中经过相同的节点数与路径总长度的比值,作为路径 P_i 与路径 P_j 的相似度值,记为 $S(P_i,P_j)$,那么,难易覆盖路径间的相似度可以表示为:

$$S(P_i,P_j)=\frac{1}{n}\sum_{l=1}^{n}\alpha_l(P_i,P_j) \tag{2}$$

其中:

$$\alpha_i(P_i,P_j)=\begin{cases}1, Pi_{Nk}=Pj_{Nk}\\ 0,\text{其他}\end{cases} \tag{3}$$

n 表示等长路径的节点总数;l 表示相同的分支节点数。式(3)取值表示两条路径 P_i,P_j 经过的分支节点是否相同,若相同则记 1,不同则记 0。例如:存在两条路径 P_1:$a\to b\to c\to d$,P_2:$a\to b'\to c\to d$,总节点数均为 4,共用 3 个相同的节点 a、c、d,只有 b 与 b'这处节点不同,则这两条路径相似度为 $S(P_1,P_2)=0.75$。

由式(2)可知,相似度 $S(P_i,P_j)$的取值范围为[0,1],且其值越大,表明两条路径穿越相同的节点数越多,利用易覆盖路径的测试数据来生成难覆盖路径测试数据的可能性也越大。

从路径相似度角度考虑,难覆盖路径的启发信息的具体获取方法为:计算难易覆盖路径间的相似度值,当相似度值大于某个阈值时,将该易覆盖路径的测试数据视为启发信息,并且将该启发信息用于替换求解难覆盖路径算法的部分初始种群,其余初始种群为随机生成的,提高初始种群适应度的同时不破坏种群的多样性。依据路径相似度值对待覆盖的难覆盖目标路径降序排序,优先求解相似度值较高的难覆盖路径,同时,优先被找到的难覆盖路径的测试数据将作为已有测试数据,一起参与启发信息的选取,减少剩余难覆盖目标路径的寻优时间。

2　自适应引导进化遗传算法

2.1　适应度函数设计

遗传算法的关键部分之一就是适应度函数的构造[12],也就是设计一个标准去判定一个可行解的优劣。本文借鉴寻找多目标路径覆盖[7]遗传算法中的适应度函数,其公式由层接近度 $A_{Pk}(x)$和分支距离 $D_{Pk}(x)$组成,具体如式(4)所示:

$$f_k(x)=A_{Pk}(x)+D_{Pk}(x) \tag{4}$$

其中:

$$A_{Pk}(x)=\frac{b(P_k,p(x))}{|P_k|} \tag{5}$$

$$D_{Pk}(x)=\begin{cases}0,\text{满足分支条件}\\ |c_2-c_1|+t,\text{不满足分支条件}\end{cases} \tag{6}$$

式(5)中:$p(x)$表示当前个体 x 覆盖的分支路径;P_k 为待覆盖的第 k 条路径;$b(P_k,p(x))$表示路径 $p(x)$与 P_k 的层接近度,其值为从第一个分支节点开始路径 P_k 与 $p(x)$不同节点的数目;$|P_k|$表示路径 P_k 中包含的分支节点数目。式(6)中:$|c_1-c_2|$是分支节点的表达式,为了区分表达 $c_1\neq c_2$ 的情况,加上一个常量 t(本文中取值范围为[0,1])[7-8]。

在寻找多个目标的遗传算法基础之上,本文考虑了分支权重因子对适应度的影响。若分支节点的嵌套深度越深,也就是该节点在路径上的序列越靠后,则生成覆盖该节点的测试数据的可能性就越低,其分支权重就会越大,对适应度值的影响也就越大。并且本文考虑到不同程序中循环嵌套结构的复杂度不尽相同,导致层接近度、分支距离、分支权重对各程序适应度值的影响也不相同,因此分别给予三个影响因子不同的权重系数[13]。根据实际运行的程序对影响因子赋予不同权重系数以权衡比例,新的个体适应度函数公式如式(7):

$$Fit(x)=\alpha A_{Pk}(x)+\beta D_{Pk}(x)+\gamma W_{Pk}(x) \tag{7}$$

其中:

$$W_{Pk}(x)=\frac{nd_i}{\sum_{i=1}^{l}nd_i} \tag{8}$$

式(7)中,$W_{Pk}(x)$是分支路径 P_k 上第 i 个节点的分支权重。其中,α、β、γ 为各影响因子的权重系数,$0\leqslant\alpha\leqslant1$,$0\leqslant\beta\leqslant1$,$0\leqslant\gamma\leqslant1$,且权重系数之和等于 1,各影响因子均不为零。本文的权重系数根据不同程序的特征,依靠以往的人工经验以及大量的仿真实验数据给出实验的具体取值。式(8)中,nd_i 为程序静态分析过程中所获得的节点的嵌套深度,分支权重值为当前节点 i 的嵌套深度与所有节点嵌套深

度总和的比值。

本文中算法评价条件包含两个层次:底层是测试数据对目标路径中每一个分支节点的覆盖程度,该条路径上所有节点均被覆盖,则该路径的测试数据寻优完成;顶层是多个难覆盖目标路径逐一被覆盖,直到所有难覆盖目标路径均被覆盖,则算法寻优结束。

2.2 遗传算子设计

本文在运用遗传算法求解时,结合保留精英个体思想,按适应度值降序排列当代种群,选取前若干组为精英种群,将余下的有序种群等分为若干个小种群并行遗传算法操作。在传统遗传算法的基础上,对传统随机的交叉算子以及变异算子进行调整,促使种群向优异的方向进化。

a)自适应交叉策略

为避免精英种群的优秀基因被破坏,将种群个体与当代精英个体的复制体进行自适应交叉,当种群个体与精英个体间适应度值相差较大时,使用较大的交叉率,相应地,当适应度值相差较小时,减小交叉率。本文假设为 5 个等分,则子种群为 S_1、S_2、S_3、S_4、S_5,且各子种群与精英个体间适应度值的差异度依次从小到大,即 $S_1<S_2<S_3<S_4<S_5$。本文中设置 S_4 与 S_5 两个子种群的交叉率为 0.9,S_1~S_3 三个子种群的交叉率设为 0.7。

b)自适应引导变异策略

由于精英个体已被保留,为使种群个体基因发生理想变异,可直接将“关键基因位”和“优秀基因值”[14]定位到精英个体上,引导种群中适应度较差的个体向精英个体的优秀基因变异。同时,根据种群个体与精英个体间适应度值的差异,自适应调整变异率。同上文交叉率的选取原则,本文中设置 S_4 与 S_5 两个子种群的变异率为 0.3,S_1~S_3 三个子种群的变异率设为 0.1。

3 策略框架与步骤

基于遗传算法的测试数据自动生成模型是近年来较为常用且简单高效的一种模型[15-16],如图 1 所示。该模型由三个部分组成,第一部分为测试环境部分,主要包括程序静态分析以及程序插装,是整个模型的基础;第二部分为遗传算法搜索部分,主要包括对种群个体评价以及遗传算子操作;第三部分为测试运行部分,运行插装的被测程序。

本文策略以上述模型为基础,给出动态引导测试数据进化生成策略的具体步骤,其描述如下:

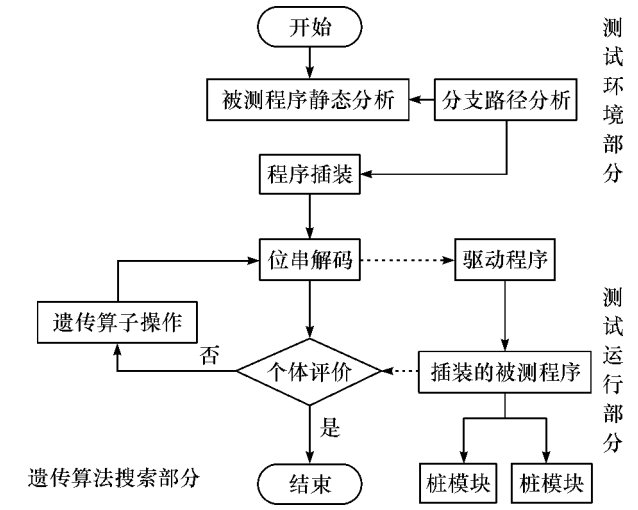

图 1 基于遗传算法的测试数据生成模型

a)静态分析被测程序,根据本文方法区分出难易覆盖路径。

b)给算法参数赋值,按式(2)计算路径相似度并初始化种群,对待覆盖目标路径进行排序。

c)根据式(7)计算初始种群适应度值并筛选出精英种群。

d)采用本文设计的遗传算子对划分的子种群并行遗传算法迭代寻优操作。

e)计算下一代种群的适应度值,与精英种群竞争并更新精英库。

f)判断当前目标路径的所有节点是否均已被覆盖,若是,将该路径的测试数据用于启发信息的查找并更新启发信息,然后从难覆盖路径集中删除该路径,选取下一条难覆盖路径作为目标路径,否则,跳转至步骤 b)。

g)判断待覆盖目标路径是否已完全被覆盖或者种群达到最大迭代次数,若满足,则迭代停止,输出测试数据集;反之,跳转到步骤 f)。

4 仿真实验及结果分析

本文选择三角形分类和冒泡排序两个基准程序,以及 Huffcode 程序、Sed 工业程序、Flex 词法分析器的部分程序作为被测程序,将本文方法与 Ahmed 方法[4]、多路径覆盖法[7]、使用已有测试数据方法(EGA 方法)进行比较实验,对比各方法在运行时间、路径覆盖率以及已覆盖路径测试数据的利用率三个方面的效果。实验程序采用 C++编写,并在 Visual Studio 2015 中运行,微机配置为 Intel(R) Core(TM) i5-3210M CPU @ 2.50GHz 2.49GHz,2.00GBRAM,64 位操作系统。遗传算法相关参数配置见表 1。

表 1　遗传算法相关参数设置

算法策略	参数取值
选择策略	精英选择,随机选择
交叉策略	单点交叉,定向交叉
变异策略	单点变异,定向变异
最大迭代次数	1000
种群大小	50

对于不同的程序,本文实验中的进化个体采用不同的编码方式[17]。各程序的基本信息见表 2。

经过 80 次仿真实验数据汇总以及以往的人工经验,最终本仿真实验中三角形分类程序的适应度函数权重系数取值设置为 $\alpha=0.8,\beta=0.1,\gamma=0.1$,路径相似度阈值设置为 0.7;冒泡排序程序和 Huffcode 程序的适应度函数权重系数取值设置为 $\alpha=0.7,\beta=0.1,\gamma=0.2$,路径相似度阈值设置为 0.75;Sed 工业程序以及 Flex 部分程序的适应度函数权重系数取值设置为 $\alpha=0.75,\beta=0.15,\gamma=0.10$,路径相似度阈值设置为 0.6。仿真实验数据表明,当各程序的权重系数和阈值为以上取值时本文策略可以取得较好的实验结果。

为验证本文测试数据生成效率,按照第 3 节所示策略步骤进行实验,对每组实验均运行 50 次后统计实验结果,不同方法在各程序中所需要的平均运行时间以及目标路径覆盖率见表 3。

表 2　被测程序基本信息

被测程序	代码行数	函数数目	输入范围	编码方式	待覆盖路径数目	易覆盖路径数目
三角形分类	26	1	[0,2048]	二进制	12	3
冒泡排序	24	1	[0,128]	二进制	24	12
Huffcode	163	2	[0,10000]	实数	64	21
Sed	108	6	[0,128]	字符	32	8
Flex	516	21	[0,128]	字符	252	15

表 3　不同方法的运行时间以及路径覆盖率

测试函数	Ahmed 方法		多路径覆盖方法		EGA 方法		本文方法	
	时间/s	覆盖率/%	时间/s	覆盖率/%	时间/s	覆盖率/%	时间/s	覆盖率/%
三角形分类	5.294	67.32	1.097	78.21	0.087	84.26	0.052	94.71
冒泡排序	4.321	58.46	2.245	81.73	0.236	91.54	0.136	96.62
Huffcode	1.313	62.89	0.873	76.14	0.528	98.96	0.332	99.71
Sed	—	—	3.675	40.88	0.312	63.51	0.172	82.45
Flex	—	—	15.865	73.58	7.961	79.53	3.894	87.89

对比表 2 和表 3 数据可以分析得出,采用本文策略,三角形分类程序的平均覆盖率达到 94.71%,算法运行所需时间为 0.052 s;冒泡排序程序和 Huffcode 程序由于目标路径间的相似度较大,可利用的含有启发式信息的数据较三角形分类程序多,所以其结果显著好于三角形分类程序,其运行时间略高于三角形分类程序,但平均覆盖率分别达到了 96.62%和 99.71%,体现了较高的路径覆盖率。由表 3 中各数据的对比容易看出,与 Ahmed 方法、多路径覆盖法以及 EGA 方法相比,对于三角形分类、冒泡排序、Huffcode、Sed 以及 Flex 程序,本文策略均可以在较短时间内取得较高的路径覆盖率。与 Ahmed 方法相比,本文策略在三角形分类程序中的目标路径覆盖率平均提高了 27.39%,消耗的时间平均降低了 5.242 s。以工业程序 Flex 为例,由于工业程序 Flex 中的选择循环嵌套结构较多,输入变量也更多,所以它的耗时相较于其他程序长,覆盖率也较难达到 100%;与运行时间最少的 EGA 相比,本文的平均耗时减少了 4.067 s,路径覆盖率也提升了 8.36%;与多路径覆盖方法相比,本文策略具有更明显的优势。

为了更直观地体现本文对易覆盖路径测试数据的利用效果,与另一种同样使用已有测试数据的方法 EGA 法相比(EGA 方法通过计算测试数据覆盖的路径与目标路径的差异度来选出可利用的测试数据,并将该测试数据用于替换遗传算法当前种群的部分个体,其中,遗传算法未考虑分支权重因素),给出在两种不同的方法下,各程序目标路径覆盖率随着已有数据利用率变动效果图,如图 2 所示。

从图 2 中各程序的实验对比曲线可以看出:a)当未使用已有测试数据时,本文方法比 EGA 方法的目标路径覆盖率高,这是因为自适应引导遗传算子操作促使算法快速找到最优解,带有权重影响因子的适应度评价函数也能根据程序特征准确筛选出符合条件的测试数据,这说明本文对遗传算子和适应度函数的优化是有效的;b)随着越来越多的已有测试数据的利用,本文策略趋近于 100%目标路径覆盖率的速度比 EGA 方法更快,在已有的易覆盖路径数据利用率达到 40%时,各程序目标路径覆盖率均达到了 80%,而 EGA 方法则需要利用更多的已有数据才能达到相同的目标路径覆盖率,由此体现了本文策略更高效的测试数据利用。

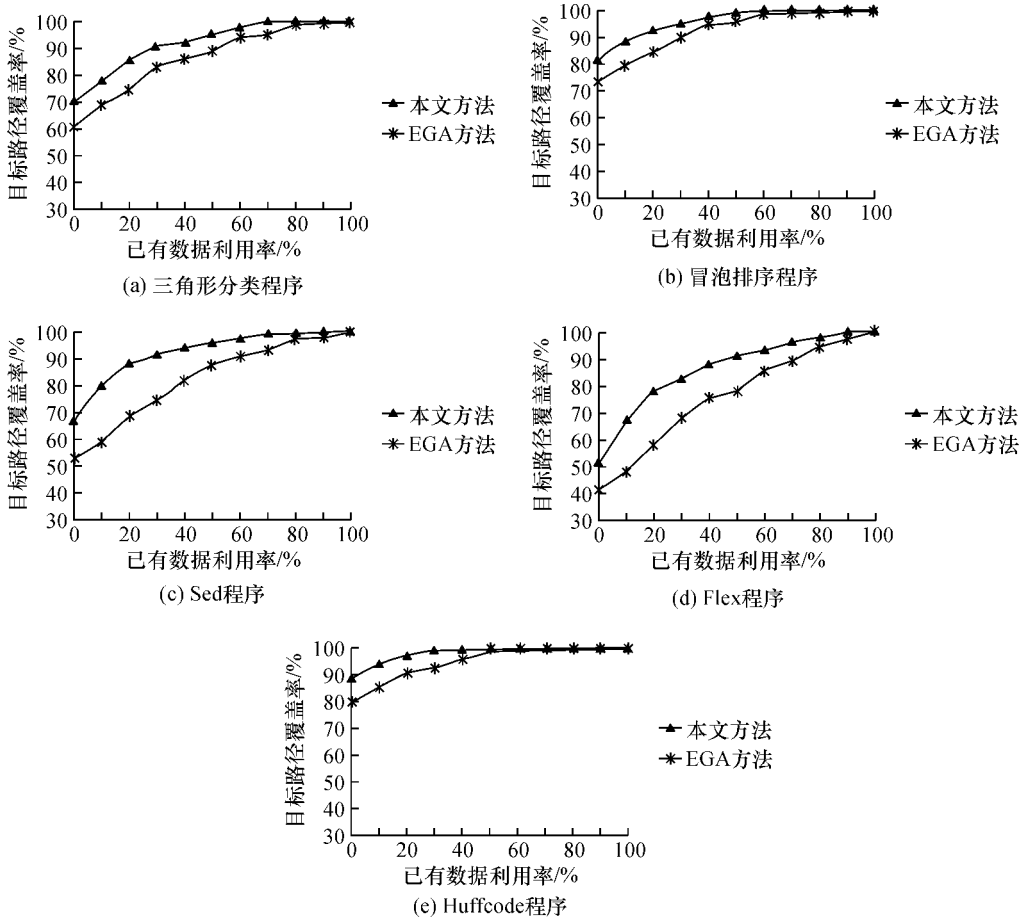

图2 各程序目标路径覆盖率曲线

综合以上实验结果,本文策略生成的测试数据在运行时间上有优势,但不明显,而在路径覆盖率以及已有数据利用率上则明显优于其他方法;同时,由所选程序静态分析可以得知,本文策略对于包含循环嵌套结构的程序以及复杂工业程序也是适用且高效的。

5 结 论

本文提出的基于相似路径的动态引导测试数据进化生成策略的主要内容如下:区分出了难易覆盖路径,将难覆盖路径作为目标路径进行排序,减少算法寻优的工作量;通过计算难易路径间的相似度,分析得出路径相关启发信息,并将该启发信息用于替换求解难覆盖路径的部分初始种群,提高算法初始种群的适应度值,实现了已有的易覆盖路径测试数据的高效利用;然后,增加对分支权重影响因素的考虑,设计了含有权重影响因子的适应度评价函数,使得评价结果更精确,并且设计了自适应遗传算子并定向引导个体交叉变异,最终达到快速生成满足理论路径覆盖的测试数据的目的。

另外,在仿真实验中发现,遗传算法参数阈值的设定在不同的被测程序中会产生不同的变化,根据不同程序的特征,如何利用较成熟的方法合理地选择算法参数(权重系数和阈值),如何更好地利用已有测试数据的启发信息将是接下来进一步研究的工作。

参考文献:

[1] 巩敦卫,张岩.一种新的多路径覆盖测试数据进化生成方法[J].电子学报,2010,38(6):1299-1304.

[2] 单锦辉,姜瑛,孙萍.软件测试研究进展[J].北京大学学报(自然科学版),2005,41(1):134-145.

[3] 夏春艳,张岩,宋丽.基于节点概率的路径覆盖测试数据进化生成[J].软件学报,2016,27(4):802-813.

[4] Ahmed M A, Hermadi I. GA-Based multiple paths test data generator[J]. Computer & Operations Research, 2008,35(10):3107-3127.

[5] 张岩,巩敦卫.基于稀有数据扑捉的路径覆盖测试数据进化生成方法[J].计算机学报,2013,36(12):2429-2440.

[6] Pachauri A, Srivastava G. Automated test data generation for branch testing using genetic algorithm: An improved approach using branch ordering, memory and elitism[J]. Journal of Systems & Software, 2013,

86(5):1191-1208.

[7] Gong D W, Zhang W Q, Zhang Y. Evolutionary generation of test data for multiple paths coverage[J]. Chinese Journal of Electronics,2011,19(2):233? 237.

[8] 丁蕊,董红斌,张岩,等.基于关键点路径的快速测试用例自动生成方法[J].软件学报,2016,27(4):814-827.

[9] 高雪笛,周丽娟,张树东,等.基于改进遗传算法的测试数据自动生成的研究[J].计算机科学,2017,44(3):209-214.

[10] 张娜,胡国亨,包晓安,等.基于多种群进化算法的测试用例优先级排序研究[J].浙江理工大学学报,2018,39(2):218-223.

[11] 吴川,巩敦卫,姚香娟.基于分支覆盖的回归测试路径选择[J].软件学报,2016,27(4):839-854.

[12] 史娇娇,姜淑娟.基于遗传算法的动态可变参数的测试数据自动生成工具[J].计算机科学,2012,39(5):124-127.

[13] 张娜,姚澜,包晓安,等.多目标优化的测试用例优先级在线调整策略[J].软件学报,2015,26(10):2451-2464.

[14] 曹凯,陈国虎,江桦,等.自适应引导进化遗传算法[J].电子与信息学报,2014,36(8):1884-1890.

[15] 谢晓园,徐宝文,史亮,等.面向路径覆盖的演化测试用例生成技术[J].软件学报,2009,20(12):3117-3136.

[16] Alshraideh M A, Mahafzah B A, Salman H S E, et al. Using genetic algorithm as test date generator for stored PL/SQL program units[J]. Journal of Software Engineering & Applications,2015,6(2):65-73

[17] 吴川,巩敦卫.基于路径相关性的回归测试数据进化生成[J].计算机学报,2015,38(11):2247-2261.

(责任编辑:康　锋)