文章编号:1671-4598(2019)06-0190-04 DOI:10.16526/j.cnki.11—4762/tp.2019.06.041 中图分类号:TP311 文献标识码:A

基于IAG—ABC算法的路径覆盖测试用例生成技术

张 娜[1],张 唯[1],徐 璐[1],吴 彪[2],包晓安[1]
(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)

摘要:针对遗传算法(genetic algorithm,GA)存在搜索初期收敛速度过快、易陷入局部最优解、未能充分结合搜索过程中的反馈信息,同时人工蜂群(artificial bee colony,ABC)算法存在初期寻优速度缓慢、局部搜索具有很大随机性等问题,对遗传算法和人工蜂群算法分别进行了改进,并将改进后的两种算法进行融合,实现两者的优势互补,提出了一种自适应遗传—人工蜂群(improved adaptive genetic—artificial bee colony,IAG—ABC)算法;采用路径覆盖信息设计引导算法搜索方向的适应度函数,并用IAG—ABC算法实现路径覆盖的测试用例生成,实验结果表明,相对于标准遗传算法和已有的自适应遗传算法,IAG—ABC算法在测试用例生成效率和路径覆盖率上均有一定的优势。

关键词:遗传算法;人工蜂群算法;路径覆盖;测试用例生成

Path Coverage Test Case Generation Technology Based on IAG—ABC Algorithm

Zhang Na[1], Zhang Wei[1], Xu Lu[1], Wu Biao[2], Bao Xiaoan[1]
(1. Department of Information & Electronics, Zhejiang Sci—Tech University, Hangzhou 310018, China;
2. Graduate School of East Asian Studies, Yamaguchi University, Yamaguchi—shi 753—8514, Japan)

Abstract: The genetic algorithm (GA) has the issue of premature convergence, failing to make full use of feedback information and easy to fall into local optimum. At the same time, the artificial bee colony (ABC) algorithm has slow initial optimization speed and randomness local searching during the running time. This paper improves the genetic algorithm and artificial bee colony algorithm respectively. And the two improved algorithm are combined to propose an improved adaptive genetic—artificial bee colony (IAG—ABC) algorithm in order to realize the complementary advantages between the two algorithms. According to the approach level and branch distance to design fitness function and using the IAG—ABC algorithm to solve the test cases generation problem based on path coverage. The experimental results show that the IAG—ABC algorithm has advantages about test case generation speed and path coverage rate when compare with GA and IAGA algorithm.

Keywords: genetic algorithm; artificial bee colony algorithm; path coverage; test case generation

0 引言

随着软件功能的不断强大,软件复杂度日益提高,软件测试过程中所需的测试用例数量也不断地增大。生成量少且覆盖程度高的测试用例成为了提高软件测试的效率的关键。遗传算法因其简单而具有良好的搜索性能在解决测试用例生成问题上得到了广泛的应用[1]。

目前,在已有的基于遗传算法的测试用例生成研究中,张岩、巩敦卫[2]提出了一种通过改进适应度函数来增加稀有数据的适应度值的方法提高了遗传算法生成测试用例的效率。夏春艳[3]等人从穿过节点的难易程度出发设计适应度函数,采用遗传算法实现路径覆盖测试用例生成。丁蕊[4]等人基于遗传算法提出关键点路径表示法改进算法适应度函数,以快速生成路径覆盖测试数据。Xiao an Bao[5]等根据海明距离衡量个体相似度,量化计算种群多样性并设计自适应的交叉和变异算子,增强遗传算法的搜索能力。高雪笛[6]等设计自适应交叉、变异算子并引入模拟退火的个体保留机制,以加速数据优化过程。

但是,已有的遗传算法存在易陷入局部最优解、未充分利用系统的反馈信息的缺陷。而人工蜂群算法能够充分考虑系统的反馈信息从而更易搜索到全局最优解,但存在初始搜索缓慢、盲目搜索的缺陷[7-10]。

综上所述,本文提出一种自适应遗传—人工蜂群(IAG—ABC)算法。首先,改进遗传算法的遗传策略,以提高遗传算法后期种群多样性,避免陷入局部最优解;其次,

收稿日期:2018-11-18; **修回日期**:2018-12-06。
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:张 娜(1977-),女,浙江宁波人,硕士,副教授,主要从事为软件工程、软件测试方向的研究。
包晓安(1973-),男,浙江东阳人,硕士,教授,主要从事自适应软件、软件测试与智能信息方向的研究。

设计一种新的局部搜索策略，以提高蜂群算法的局部搜索效率；最后，在初期通过自适应遗传算法得到解的分布，在后期通过改进的人工蜂群算法寻找最优解。针对路径覆盖的测试需求设计适应度函数。从而将问题转化成通过适应度函数引导算法搜寻满足适应度函数的最优解。

1　IAG－ABC 算法

IAG－ABC 算法整体分为改进遗传算阶段和改进人工蜂群算法阶段。

1.1　IAG－ABC 中的遗传算法阶段

IAG－ABC 中的遗传算法部分采用实数编码的形式，为了降低优良基因被破坏的风险，将轮盘赌和优质个体保留法相结合的方式进行选择。在交叉和变异操作中，交叉率 P_c 和变异率 P_m 是能够影响算法的收敛和寻优能力的重要参数。标准的遗传算法中，无法根据当前种群的多样性选择灵活地选择交叉还是变异。本文引入参数 α 来度量进化过程中的种群的多样性，计算方法如公式（1）所示：

$$\alpha = \arcsin(f_{avg}/f_{\max}) \tag{1}$$

其中：f_{avg} 表示当前种群中个体适应度值的平均值，$f_{\max}$ 表示当前种群中最优个体的适应度值。同时，用反正弦函数能非线性地度量当前种群的适应度值的分散程度，用于直观地描述种群多样性。

本文设计的 P_c 和 P_m 计算公式如下式（2）和式（3）所示。其中，系数 P_{c1}，P_{c2}，P_{m1}，P_{m2} 的取值区间为（0，1），可以在优化过程中调整。

$$p_c = \begin{cases} p_{c1} - \dfrac{2(p_{c1}-p_{c2})}{\pi}\alpha, \alpha < \dfrac{\pi}{6} \\ p_{c1}\left(1-\dfrac{2\alpha}{\pi}\right), \alpha \geqslant \dfrac{\pi}{6} \end{cases} \tag{2}$$

$$p_m = \begin{cases} p_{m1}\left(1-\dfrac{2\alpha}{\pi}\right), \alpha < \dfrac{\pi}{6} \\ p_{m1} - \dfrac{2(p_{m1}-p_{m2})}{\pi}\alpha, \alpha \geqslant \dfrac{\pi}{6} \end{cases} \tag{3}$$

当 $\alpha < \dfrac{\pi}{6}$ 的时候，α 的值越小，种群的多样性越好，采用先执行交叉操作，后执行变异操作的策略。选择增大交叉概率，以提高进化出优质的新个体的概率；减小变异概率的值以加快算法收敛，当 $\alpha \geqslant \dfrac{\pi}{6}$，$\alpha$ 的值越大，种群多样性越差。此时采纳先执行变异操作，后执行交叉操作的策略。此时，交叉操作对于算法跳出局部最优解的作用不是很大，应增加变异概率的值，提高算法跳出局部极值的概率。

1.2　IAG－ABC 中的人工蜂群算法阶段

在标准的 ABC 算法中，雇佣蜂的开采（局部搜索）方式如式（4）所示，其中，其中，i，$k \in \{i=1, 2, \cdots N\}$ 且 $i \neq k$，$j \in \{i=1, 2, \cdots, D\}$，$R$ 为［－1，1］中的随机数[11]。

$$x'_{ij} = x_{ij} + R \times (x_{ij} - x_{kj}) \tag{4}$$

因此，局部搜索过程中存在一定的随机性，这是算法的初期搜索效率不高的主要原因。已有的研究表明，鉴档案精英学习[12]的方法中的精英个体引导策略能够有效提高算法的搜索效率。本文借鉴其思想，设计了基于当前最优解引导的自适应局部搜索策略，如式（5）所示：

$$x'_{ij} = x_{ij} + ED_{ij}R \times (x_{ij} - x_{kj}) \tag{5}$$

$$ED_{ij} = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2} \tag{6}$$

其中：ED_{ij} 表示局部搜索步长，计算方法如式（6）所示。本文用当前个体与目前最优个体之间的空间欧式距离表示 ED_{ij}。若 ED_{ij} 的值越大，则两个个体之间的差异越大，则需要自适应地增大搜索步长，提高搜索效率。反之，则需要自适应地缩小搜索范围。若新蜜源的适应度值大于旧蜜源，则对旧蜜源进行替换，否则继续进行局部搜索。

观察蜂根据公式（7）选择待开采蜜源，其中，F_i 表示的是第 i 个蜜源对应的适应度值。

$$p_i = F_i / \sum_{j=1}^{N} F_j \tag{7}$$

当一个蜜源被开采后殆尽（达到最大搜索次数 limit）时进入侦查蜂阶段，采用如下公式（8）寻找新蜜源。其中，x_{ij} 为新蜜源的第 j 维分量，$j \in \{i=1, 2, \cdots, D\}$，$R$ 是范围在（0，1）内的一个随机数，x_{maxj} 和 x_{minj} 分别为蜜源的第 j 维分量的上下界。

$$x_{ij} = x_{\min j} + R \times (x_{\max j} - x_{\min j}) \tag{8}$$

2　基于 IAG－ABC 算法的测试用例生成

IAG－ABC 算法并于路径覆盖的测试用例生成。如图（1）所示，是基于 IAG－ABC 算法的测试用例自动生成的模型图。

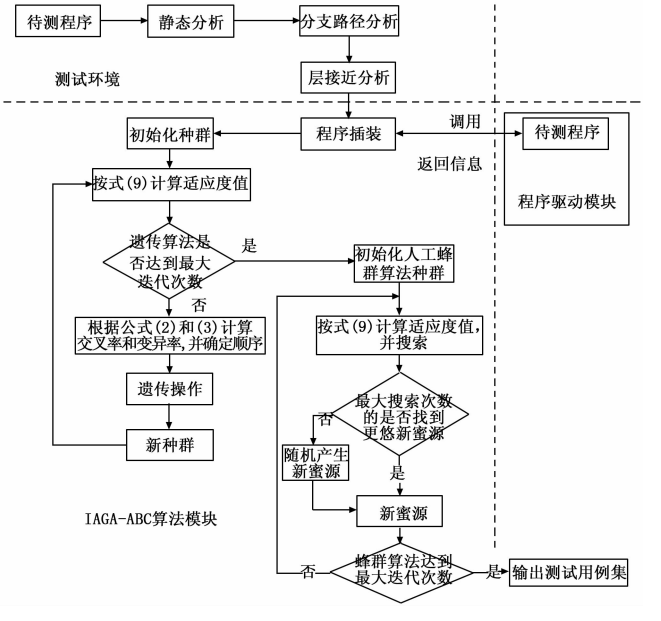

图 1　基于 IAG－ABC 算法的测试用例生成模型

2.1 适应度函数的设计

一般地，评判某个测试用例数据对程序路径的覆盖程度的标准有分支距离或者层接近度。本文按照 Mcdermid[13] 所提的插桩法对程序进行插桩，然后执行测试用例中的输入数据，以获取测试用例的分支距离和层接近度的信息。

具体步骤如下：假设被测程序中有 n 个分支谓词和 m 个关键节点，该被测程序有 N 个输入参数，则测试用例有 D 维，$X=(x_1, x_2, \cdots, x_D)$，则需要在第 i 个分支谓词前插入分支距离函数 $bd_i(x_1, x_2, \cdots, x_D)$，$i\in[1, n]$，将测试用例 X 的分支距离函数值记为 $BD(X)$，计算方法如式（9）所示，$BD(X)$ 的值越小表示，该用例的分支覆盖强度越高；在程序的每个关键节点前插入计数语句，用于统计测试用例 X 所经过的节点个数，最后与完全覆盖目标路径所需经过的节点个数相减，取差值的绝对值记为层接近度函数 $AL(X)$，$AL(X)$ 的值越小，表示该测试用例穿越目标路径的节点个数越多，路径覆盖强度越强。最后，适应度值 F 按公式（10）进行计算，F 的值越小，表明测试用例的覆盖目标路径的程度越高。

$$BD(X)=\sum_{i=0}^{n} bd(X) \tag{9}$$

$$F=AL(X)+BD(X) \tag{10}$$

2.2 算法流程

输入：初始化种群 TN，IAG－ABC 算法中遗传算法的最大迭代次数 N、系数 P_{c1}，P_{c2}，P_{m1}，P_{m2}的初始值、种群规模，搜索维度 D（测试用例输入参数的个数）；IAG－ABC 算法中人工蜂群算法的蜜蜂总数、最大搜索次数 limit 和最大迭代次数 M；

输出：所搜过程中的最优解 T。

Begin

1）按照公式（10）计算初始种群中个体的 F 值；

2）采用轮盘赌和最优个体保留法进行选择；

3）按照式（1）计算 α 的值；

4）根据 α 的值，按照式（2）和（3）计算 P_c 和 P_m，并确定交叉和变异的先后顺序；

5）进行交叉和变异操作产生新种群并计算个体的适应度值；

6）记录种群最优解，判断是否达到最大迭代次数 N，若是则执行步骤 7)；否则，返回步骤 2)；

7）按照公式（5）进行局部搜索，寻找适应度值更高的蜜源代替原蜜源；

8）按照公式（7）进行选择蜜源，在其附近根据公式（5）开采；

9）某蜜源开采殆尽时，根据公式（8）进行蜜源位置的重置；

10）记录优质蜜源，判断人工蜂群算法的迭代次数是否大于等于 M，是则输出 T，算法终止，否则返回步骤 6)。

End

3 实验及结果分析

3.1 实验对象

本文对标准遗传算法（GA）、文献［6］中的自适应遗传（IAGA）算法和本文的 IAG－ABC 算法进行性能对比，实验均在 MATLAB R2016b 上实现。采用 5 个常用测试基准函数，其信息如表 1 所示。

表 1 测试基准函数信息表

函数	名称	表达式	搜索范围	理论最优
F_1	Sphere	$f_1(x)=\sum_{i=1}^{n} x_i^2$	[－100，100]	0
F_2	Step	$f_2(x)=\sum_{i=1}^{n}([x_i+0.5])^2$	[－100，100]	0
F_3	Rastrigin	$f_3(x)=\sum_{i=1}^{n}[x_i^2-10\cos 2\pi x_i+10]$	[－5.12，5.12]	0
F_4	Ackley	$f_4(x)=-20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n}x_i^2}\right)-\exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i\right)+20+e$	[－32，32]	0
F_5	Rosenbrock	$f_5(x)=\sum_{i=1}^{n-1}[100(x_{i+1}-x_i^2)^2+(x_i-1)^2]$	[－30，30]	0

其中，F_1 和 F_2 是单峰函数，用于检验算法的寻优精度；F_3 和 F_4 是多峰函数，用于检测算法的全局寻优能力；F_5 则是一个复杂的单峰函数，用于评价算法是否易于陷入局部最优解。由于遗传算法的变异率和交叉率没有恒定的参数，本文借鉴 Schaffer 等[15] 人的研究，在以上两个实验中设置 GA 算法的交叉率设为 0.9，变异率设为 0.2；IAGA 和 IAG－ABC 中的交叉率计算公式中的 $Pc_1=0.8$，$Pc_2=0.6$，变异率计算公式中的 $Pm_1=0.05$，$Pm_2=0.005$；三种算法的搜索维度均为 30。

为了验证本文所提的 IAG－ABC 算法在测试用例生成上的有效性，本文选取了软件测试中常用的 5 个工业程序[14]并采用 GA、IAGA 和 IAG－ABC 来生成测试用例，被测程序的基本信息如表 2 所示。

表 2 被测程序信息表

编号	程序名	代码行数	路径节点数
PG1	Space	90	18
PG2	Tot_info	406	53
PG3	Replace	564	85
PG4	Sed	8063	382
PG5	Flex	11783	543

3.2 实验结果分析

在三种算法求解函数最小值优化问题的实验中，为了避免实验过程中存在的随机性，分别对每个函数进行了 50 次实验，然后记录实验过程中的均值、方差以及达到规定

精度所需的平均时间的平均值，见表 3。

表 3　三种算法在测试基准函数上的实验结果

函数	算法	均值	方差	规定精度	平均时间
F_1	GA	1.5842e－04	1.1192e－04	1.01e－04	12.3658
	IAGA	1.3694e－11	2.0183e－11		8.5618
	IAG－ABC	3.4516e－16	7.5683e－17		3.5697
F_2	GA	0	0	0	9.8546
	IAGA	0	0		3.5698
	IAG－ABC	0	0		1.2456
F_3	GA	5.6397	1.0569	1.2e－1	13.2589
	IAGA	3.8948e－09	2.5789e－07		5.6983
	IAG－ABC	0	0		0.8954
F_4	GA	0.2546	0.1269	1.32e－04	22.654
	IAGA	8.5675e－05	1.0369e－04		8.6951
	IAG－ABC	1.245e－14	3.5894e－15		2.1548
F_5	GA	73.2549	25.3241	1.09e＋02	12.1546
	IAGA	9.5124	12.5879		2.6589
	IAG－ABC	5.1236e－04	8.6591e－04		0.9325

从表 3 中的数据可以看出，相对于 GA 和 IAGA 算法，IAG－ABC 算法所寻得的平均最优解更接近理论最优、方差更小、能在更少的时间内达到规定的精度，表明算法的性能更优且更加稳定，证明了本文所提改进策略的有效性。

表 4 所示是三种算法在生成路径覆盖的测试用例过程中所需的平均迭代次数和所生成的测试用例的路径覆盖率的结果对比，图 2 是记录三种算法运行 50 次生成路径覆盖的测试用例所需时间分布的箱形图。

表 4　三种算法的测试用例生成性能对比

被测程序	GA		IAGA		IAG－ABC	
	迭代次数	覆盖率/%	迭代次数	覆盖率/%	迭代次数	覆盖率/%
Space	236.2	95.4	158.6	100	93.5	100
Tot_info	268.5	85.2	165.8	98.4	103.8	100
Replace	380.1	87.3	230.4	96.7	153.9	100
Sed	752.3	83.51	462.5	93.4	256.8	100
Flex	1580.6	85.4	1230.8	92.5	985.4	98.4

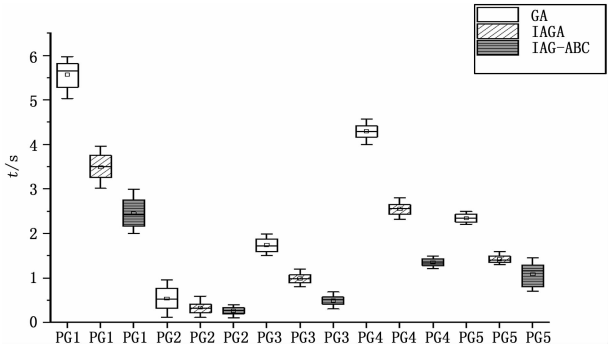

图 2　算法在生成测试用例中所需时间对比

由表 4 和图 2 可知，IAG－ABC 算法生成测试用例所需的迭代次数更少且生成的测试用例的覆盖率更高，且所需要的时间更短。

4　结束语

本文分别对标准的遗传算法和人工蜂群算法进行了改进，在改进之后将两者进行优势互补，将遗传算法的运行结果作为蜂群算法的初始种群，使用改进的人工蜂群算法进行最优解的搜索。通过实验证明了本文所提的 IAGA－ABC 算法相对于已有的 IAGA 算法在收敛速度和全局搜索能力上的优势，有助于提高路径覆盖的测试用例生成效率和路径覆盖率。目前，已有许多研究提出了关于遗传算法的各种改进，而混合算法还存在一定的研究空间。在未来的研究中，将会考虑融合其他智能搜索算法，以提高测试用例生成的效率。

参考文献：

[1] 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.

[2] 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.

[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] 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.

[5] Bao X，Xiong Z，Zhang N，et al. Path－oriented test cases generation based adaptive genetic algorithm [J]. Plos One，2017，12 (11)：e0187471.

[6] 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.

[7] 李龙燕. 基于人工蜂群算法的软件测试用例自动生成研究与实现 [D]. 河北工程大学，2017.

[8] 蒋雪婷，葛文萍，李玉涛. 基于改进量子蜂群算法的多中继选择 [J]. 计算机工程与设计，2017，38 (3)：571－575.

[9] 倪志伟，李蓉蓉，方清华，等. 基于离散人工蜂群算法的云任务调度优化 [J]. 计算机应用，2016，36 (1)：107－112，121.

[10] 李彦苍，彭　扬. 基于信息熵的改进人工蜂群算法 [J]. 控制与决策，2015，30 (6)：1121－1125.

[11] 葛　宇，梁　静. 一种多目标人工蜂群算法 [J]. 计算机科学，2015 (9)：257－262，281.

[12] 谢承旺，王志杰，夏学文. 应用档案精英学习和反向学习的多目标进化算法 [J]. 计算机学报，2017，40 (3)：757－772.

[13] Mcdermid J A. An automated framework for structural test－data generation [A]. IEEE International Conference on Automated Software Engineering [C]. IEEE，1998.

[14] Harman M. The Current State and Future of Search Based Software Engineering [A]. Future of Software Engineering [C]. IEEE Computer Society，2007.

[15] Schaffer J D，Caruana R A，Eshelman L J，et al. A study of control parameters affecting online performance of genetic algorithms for function optimization [A]. International Conference on Genetic Algorithms [C]. Morgan Kaufmann Publishers Inc. 1989：51－60.