

基于 SPEA2+SDE 算法的测试用例自动生成技术研究

谭 鑫 彭耀鹏 杨 帅 郑 炜

(西北工业大学软件与微电子学院 西安 710072)

摘 要 软件测试是确保软件质量的重要手段。然而随着软件结构和功能的日益多样化,软件测试的复杂度和成本大为提高。测试用例自动生成技术可以降低手工测试的高额成本,同时提高测试结果的可信度。主要研究了基于进化算法的测试用例自动生成技术,通过比较不同算法对于若干经典程序的测试用例生成效率,提出了 SPEA2+SDE 算法,其可以很好地用于测试用例的自动生成。最后通过 Kruskal-Wallis 非参数检验,说明了上述结论的广泛性和可靠性。

关键词 SPEA2+SDE 算法,测试用例自动生成,进化算法,非参数检验

中图分类号 TP311.5 **文献标识码** A

Automated Test Case Generation Based on SPEA2+SDE

TAN Xin PENG Yao-peng YANG Shuai ZHENG Wei

(School of Software and Microelectronics, Northwestern Polytechnical University, Xi'an 710072, China)

Abstract Software testing is crucial to ensure software quality. However, the complexity and cost will increase a lot with the growing variety of software structures and functionality. Automated test case generation is aimed at reducing the high cost as well as improving the reliability of the test results. This paper mainly discussed the technology of automated test case generation based on evolutionary algorithm. By comparing the testing efficiency of different algorithms on several classic programs, SPEA2+SDE performs best among all the algorithms in generating the test case automatically. Finally, we used Kruskal-Willis test to analyze the test results, proving that the conclusion above is general and reliable.

Keywords SPEA2+SDE, Automated test case generation, Evolutionary algorithm, Non-parametric test

1 引言

软件故障定位是软件测试过程中最为耗时和费力的活动之一。有时,一些项目由于不正确的测试,直到软件正确运行数年之后才发现某些输入参数会导致程序出现严重故障,造成不可挽回的后果。因此,寻找一个高效的测试用例集,使得能够在短时间内尽量定位出被测程序的所有故障,至关重要。测试用例自动生成技术^[1]可以降低手工测试的高额成本,将测试人员从繁重的劳动中解脱出来,同时也能提高测试过程的可信度。进化算法是目前最为流行的用于测试用例生成的搜索算法^[2]。基于进化算法的测试用例自动生成技术通过适应度函数的引导,使得测试用例生成效率相对于传统技术明显提高。

本文提出的 SPEA2+SDE 算法^[3]可以很好地用于测试用例的自动生成。首先对 SPEA2+SDE 算法进行了介绍;其次,将 SPEA2+SDE 算法和其它两种进化算法:IBEA-HV、MOEA-II,以及随机算法(Random),应用于 4 个经典的被测程序:“CAS”、“ISBN”、“QUEEN”、“TRIANGLE”,以执行时间和程序覆盖率为双目标^[4]进行测试用例的生成;最后通过 Kruskal-Wallis 非参数检验^[5]对各组实验数据进行了比

较,说明了实验结果的可靠性,并得出 SPEA2+SDE 算法可以高效、广泛地用于测试用例自动生成。

2 SPEA2+SDE 算法

2.1 SPEA2 算法

SEPA2 是 Zitzler 和 Thiele 在 2001 年提出的对 SPEA 的改进版本^[6]。他们在适应度分配策略、个体分布性的评估方法以及非支配解集的更新 3 个方面进行了改进。在 SPEA2 中,个体的适应度函数为 $F(i) = R(i) + D(i)$,其中, $R(i)$ 同时考虑到个体 i 在外部种群和进化种群中的个体支配信息, $D(i)$ 是由个体 i 到它的第 k 个邻近个体的距离决定的拥挤度度量。在构造新群体时,首先进行环境选择,然后进行交配选择。在进行环境选择时,首先选择适应度小于 1 的个体进入外部种群,当这些个体数目小于外部种群的大小时,选择进化种群中适应度较低的个体;当这些个体数目大于外部种群的大小时,则运用环境选择进行删减。在交配选择中,运用锦标赛机制选择个体进入交配池。SPEA2 引入了基于近邻规则的环境选择,简化了 SPEA 中基于聚类的外部种群更新方法。虽然其计算复杂度仍为种群规模的立方,但是基于近邻规则的环境选择得出的解分布的均匀性是很多其他方法无法超越的。

2.2 SPEA2+SDE 算法

SPEA2 进化算法由 SPEA 改进而来,在精英保留机制的基础上,考虑了个体 i 在进化种群和外部种群中的支配信息。为了防止早熟收敛导致的解集的收敛性差、多样性降低的问题,SPEA-II 增加了个体的多样性维持机制。然而这种机制的增加虽然能够改善个体的收敛性,却会导致最终的 Paerto 解集远离 Paerto 前端。

解决这个问题的方法是为个体添加密度估计,因此引入 SDE(Shift-based Density Estimation)。该方法不是对进化过程中的选择标准的改进,而是引入另一种选择标准:密度估计。在 SDE 技术中,每次进化都会统计出每个个体的密度,在 SPEA2 中,密度的计算方法是该个体到第 k 近距离的个体的适应度值的差。SDE 同时考虑了个体的分布和收敛性信息。具体做法是:给定理想的密度估计值,将种群中的收敛性差的个体移动到拥挤度高的区域,这样就能给那些收敛性差的个体一个较高的密度值,使其在选择过程中更容易被淘汰掉。因此 SPEA-II+SDE 技术最终得到的解集既能满足收敛性要求,又能尽可能地接近 Pareto 前端。

2.3 编码方式

进化算法涉及的主要编码方式有:二进制编码、格雷码编码、实数编码和符号编码等。其中二进制编码最为常用。本文实验中采用二进制编码,其方式如下:

对于 $n(n \geq 1)$ 维连续函数 $f(x), x = (x_1, x_2 \dots x_n), x_i \in [u_i, v_i] (i = 1, 2, \dots, n)$ 采用定长二进制编码。 n 维变量的二进制边长为 l_i ,那么 x 的编码从左到右依次构成总长度为 $L = \sum l_i$ 的二进制编码串。则变量 x_i 可表示为:

$$a_{k_1}^i a_{k_2}^i \dots a_{k_{l_i}}^i$$

其中 $a \in \{0, 1\}, (a_{k_1}^i)^2 + (a_{k_2}^i)^2 + \dots + (a_{k_{l_i}}^i)^2 = x_i$ 。

则染色体结构为:

$$a_{k_1}^1 a_{k_2}^1 \dots a_{k_{l_1}}^1 a_{k_1}^2 a_{k_2}^2 \dots a_{k_{l_2}}^2 \dots a_{k_1}^n a_{k_2}^n \dots a_{k_{l_n}}^n$$

例如,对于参数(9, 2, 13),若用 3×4 位长的二进制编码来表示,则个体符号串(染色体)为 $X: 100100101101$ 。

2.4 适应度函数

进化论中的适应度,是表示某一个体对环境的适应能力,也表示该个体繁殖后代的能力。遗传算法的适应度函数也叫评价函数,是用来判断群体中个体的优劣程度的指标,它是根据所求问题的目标函数来进行评估的。

遗传算法在搜索进化过程中一般不需要其他外部信息,仅用适应度函数来评估个体或解的优劣,并作为以后遗传操作的依据。由于遗传算法中,适应度函数要比较排序并在此基础上计算选择概率,因此适应度函数的值要取正值。由此可见,在不少场合,将目标函数映射成求最大值形式且函数值非负的适应度函数是必要的。

适应度函数的设计主要满足以下条件:

- 1) 单值、连续、非负;
- 2) 合理、一致性;
- 3) 计算量小;
- 4) 通用性强。

本文以分支覆盖率和到达目标分支的执行时间为双目

标,所采用的适应度函数^[7]如下:

$$Fitness_1(test, object) = approach_level(test, object) + normalise(branch_distance(test, object))$$

$$Fitness_2(test, object) = time(test, object)$$

其中, $Fitness_1$ 是以分支覆盖率为目标的适应度函数,其值越小越好。方法水平 $approach_level$ 反映了目标路径与实际路径的层接近度,该值越小表示越接近目标路径,个体质量越好。分支距离^[8] $branch_distance$ 与路径上各节点有关。对于条件语句, $branch_distance$ 的计算与谓词表达式的类型紧密相关。不同谓词分支距离的计算方法如表 1 所列,其中, c_1, c_2 都为算术表达式, K 为一正常量。

表 1 不同谓词分支距离的计算方法

谓词表达式	branch_distance
$\llbracket c_1 > c_2 \rrbracket$	$\begin{cases} 0, & c_1 - c_2 > 0 \\ c_2 - c_1 + k, & \text{else} \end{cases}$
$c_1 < c_2$	$\begin{cases} 0, & c_1 - c_2 < 0 \\ c_1 - c_2 + k, & \text{else} \end{cases}$
$c_1 \geq c_2$	$\begin{cases} 0, & c_1 - c_2 \geq 0 \\ c_2 - c_1 + k, & \text{else} \end{cases}$
$c_1 \leq c_2$	$\begin{cases} 0, & c_1 - c_2 \leq 0 \\ c_1 - c_2 + k, & \text{else} \end{cases}$
$c_1 = c_2$	$\begin{cases} 0, & c_1 - c_2 = 0 \\ c_1 - c_2 + k, & \text{else} \end{cases}$
$c_1 > c_2$	$\begin{cases} 0, & c_1 - c_2 \neq 0 \\ k, & \text{else} \end{cases}$

由于最大分支距离通常未知,因此通过下面的公式将 $branch_distance(d)$ 量化到 $[0, 1)$ 。

$$normalise(d) = 1 - 1.001^{-d}$$

$Fitness_2$ 表示到达目标分支的执行时间,其值越小越好。

3 测试用例自动生成

本文将 IBEA-HV、MOEAD-II、SPEA2+SDE 这 3 种进化算法以及随机算法(Random)应用于 4 个被测程序:“CAS”、“ISBN”、“QUEEN”、“TRIANGLE”,以执行时间和程序覆盖率为双目标进行测试用例的自动生成。

3.1 3 种进化算法的异同

IBEA-HV、MOEAD-II、SPEA2+SDE 是 3 种经典的进化算法。初代种群产生之后,按照适者生存和优胜劣汰的原理,逐代演化产生出越来越好的近似解,在每一代,根据问题域中个体的适应度大小选择个体,并借助于自然遗传学的遗传算子进行选择、交叉和变异,产生出代表新的解集的种群。

3 种算法的不同之处在于其采用不同的遗传算子进行选择、交叉和变异。

3.2 被测程序的描述

选取了 4 个被测程序:“CAS”、“ISBN”、“QUEEN”、“TRIANGLE”。其中:

- 1) “CAS”,用于检测当前输入是否是合法的“化学物质登录号”。
- 2) “ISBN”,用于检测当前输入是否是合法的“国际标准书号”。
- 3) “QUEEN”,用于解决八皇后问题。

4)“TRIANGLE”,根据输入的3条边的长度,判断其形状。

3.3 以“TRIANGLE”为例生成测试用例

“TRIANGLE”程序的输入需要以3个整数分别作为三角形的3条边的长度,进而判断该三角形的形状。“TRIANGLE”程序如图1所示。

```
if (a+b>c && b+c>a && a+c>b){
    if (a==b || b==c)
        if (a==c){
            type="equilateral triangle";
            approach_level=0;
        }
    else{
        type="isosceles triangle";
        approach_level=1;
    }
}
else{
    type="ordinary triangle";
    approach_level=2;
}
else{
    type="not triangle"
    approach_level=3;
}
```

图1 “TRIANGLE”程序片段

1)编码:本实验中假定3条边的长度 l_i 是区间 $[0,1023]$ 的整数,所以每条边需要10位二进制表示,染色体长度为 3×10 ,且 a 大于或等于 b 和 c 。

2)目标分支:为判断该三角形为等边三角形。

3)fitness1:描述分支覆盖率的适应度函数,在此程序中相关参数的计算如图2所示。

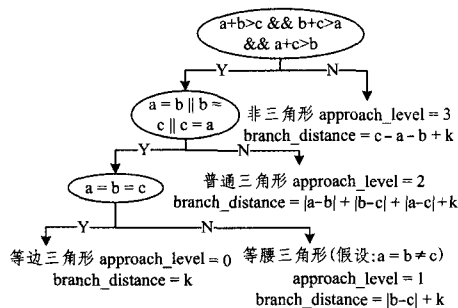


图2 “TRIANGLE”fitness1的执行过程

4)fitness2:描述到达目标分支 $a=b=c$ 所用的时间。

5)随机生成初始种群,代入3种进化算法:IBEA-HV、MOEAD-II、SPEA2+SDE以及一种随机算法,进行测试用例的生成。

3.4 “CAS”、“ISBN”、“QUEEN”生成测试用例

被测程序“CAS”、“ISBN”、“QUEEN”测试用例的生成方法与3.3节类似,在此不再赘述。

4 实验结果与评估

4.1 实验结果

本文SPEA2+SDE、IBEA-HV、MOEAD-II 3种进化算法以及随机算法(Random)应用于4个被测程序:“CAS”、“ISBN”、“QUEEN”、“TRIANGLE”,以执行时间和程序覆盖率为双目标进行测试用例的自动生成。每个被测程序对应于每种进化算法分别执行10次,统计每次到达目标分支的评估次数^[9],以评估次数为标准判断3种进化算法与随机算法在本实验中的优劣,评估次数越少,该算法性能越好。评估次数的平均值如表2所列。

表2 算法对不同测试用例的平均评估次数

待测程序 \ 算法	IBEA-HV	MOEAD-II	RANDOM	SPEA2+SDE
CAS	162.30	1484.20	25550.30	169.50
ISBN	4754.40	99999.00	99999.00	4169.40
QUEEN	13434.50	99999.00	99999.00	5191.60
TRIANGLE	9403.80	99999.00	99999.00	3997.20

注:评估99999.00次表示已达到允许评估次数的上限

经过比较,得出每个测试程序的算法优劣对比结果如表3所列,可以看出进化算法的执行效率明显优于随机算法。特别地,SPEA2+SDE进化算法总体表现最佳。

表3 各算法执行效果的对比

测试程序	算法比较(优→劣)
CAS	IBEA-HV=SPEA2+SDE>MOEAD-II>RANDOM
ISBN	SPEA2+SDE>IBEA-HV>MOEAD-II>RANDOM
QUEEN	SPEA2+SDE>IBEA-HV>MOEAD-II>RANDOM
TRIANGLE	SPEA2+SDE>IBEA-HV>MOEAD-II>RANDOM

4.2 实验结果评估

为了说明实验结果的可靠性以及广泛性,对实验结果进行了Kruskal-Willis^[16]检验。

Kruskal-Willis检验是关于 $N(N \geq 3)$ 组数据的非参数性测试^[7],它是用来检测总体函数分布的一致性原假设和其替代假设。Kruskal-Willis检验的优点在于,可以不对待检验的数据的分布进行假设。通过观察待检测数据的渐进显著性值来拒绝或接受原假设。Kruskal-Willis检验不会只针对图形上显示的数据的波动性进行处理,而且不会进行关于数据分布的假设,因此与参数检验相比,更具有一般性。遗传算法在每次进行遗传进化的过程中都具有一定的随机性,而且这种随机性难以预测。因此为了更一般地说明问题,采用Kruskal-Willis非参数检验的方法,对4种算法中的每种算法取10个值来保证结论的一般性,从而保证结果的可靠性。

在对本实验结果的非参数检验统计中,使用Kruskal-Willis检验不同算法待测程序的评估次数。

1)原假设为 H_0 :4种算法对不同待测程序的评估次数无显著性差异。

2)择假设为 H_1 :4种算法对不同待测程序的评估次数有显著性差异。

Kruskal-Willis检验结果如表4、表5所列。根据观察,4组待测程序的渐进显著性均极小,几乎接近0,因此拒绝原假设,并得出结论:4种算法对不同待测程序的评估次数是有显

著性差异的。所以,4.1节实验所得结论在95%的置信度是可靠的。

表4 Kruskal-Wallis 检验结果

	秩		
	group	N	秩均值
CAS	1	10	14.70
	2	10	17.20
	3	10	35.20
	4	10	14.90
	总数	40	
ISBN	1	10	10.90
	2	10	30.50
	3	10	30.50
	4	10	10.10
	总数	40	
QUEEN	1	10	10.10
	2	10	30.50
	3	10	30.50
	4	10	10.90
	总数	40	
TRIANGLE	1	10	13.60
	2	10	30.50
	3	10	30.50
	4	10	7.40
	总数	40	

注:算法1,2,3,4分别代表 IBEA-HA、MOEAD-II、RANDOM、SPEA2+SDE

表5 Kruskal-Wallis 检验统计量

程序	CAS	ISBN	QUEEN	TRIANGLE
卡方	21.376	33.467	33.471	35.047
df	3	3	3	3
渐近显著性	8.7928E-5	2.5666E-7	2.5621E-7	1.1905E-7

结束语 本文主要从进化算法在测试用例的自动生成方面的应用展开研究,以分支覆盖率和执行时间为双目标,研究了进化算法与随机算法以及不同进化算法之间在测试用例自动生成方面的执行效率,从而得出了在测试用例的生成应用中,进化算法的执行效率明显优于随机算法。特别地,SPEA2+SDE 进化算法的执行效率明显优于 IBEA-HV、MOEAD-II 两种进化算法。最后,为了说明实验结果的可靠

性和广泛性,对上述实验结果进行 Kruskal-Wallis 检验,得出:实验结果所得结论在95%的置信度是可靠的。

参 考 文 献

[1] Ferrer J, Chicano F, Alba E. Evolutionary Algorithms for the Multi-Objective Test Data Generation Problem [J]. Software: Practice and Experience, 2012, 42(11): 1331-1362

[2] McMinn P. search-based software test data generation; a survey [J]. Software Testing, Verification and Reliability, 2004, 14(2): 105-156

[3] Li Mi-qing, Yang Sheng-xiang, Liu Xiao-hui. Shift-Based Density Estimation for Pareto-Based Algorithms in Many-Objective Optimization [J]. IEEE Transactions on Evolutionary Computation, 2014, 18(3)

[4] 韩丽霞. 求解多目标优化问题的新遗传算法 [J]. 计算机科学, 2013, 40(6A): 64-66, 95

[5] 王静龙, 梁小筠. 非参数统计分析 [M]. 北京: 高等教育出版社, 2006: 78-156

[6] Zitzler E, Laumanns M, Thiele L. SPEA2: improving the strength Pareto evolutionary algorithm; CH-8092 [R]. Zurich, Switzerland, 2001

[7] Harman M, McMinn P. A Theoretical Empirical Study of Search-Based Testing: Local, Global, and Hybrid Search [J]. IEEE Transactions on Software Engineering, 2010, 36(2)

[8] Mark H, Kiran L, Phil M. A multi-objective approach to search-based test data generation [C] // Proceedings of Genetic and Evolutionary Computation (GECCO 2007). London, England, UK, 2007

[9] Mark H, Phil M, de Souza Jerffeson T, et al. Search Based Software Engineering: Techniques, Taxonomy, Tutorial [M] // Empirical Software Engineering and Verification: International Summer Schools, LASER 2008-2010, Elba Island, Italy, Revised Tutorial Lectures. 2012: 1-59

[10] 杜强, 贾丽艳. SPSS 统计分析从入门到精通 [M]. 北京: 人民邮电出版社, 2009: 118-138

[22] Naps T, Roßling G, Almström V, et al. Exploring the role of visualization and engagement in computer science education [J]. ACM SIGCSE Bull, 2003, (35): 131-152

[23] 樊文强. 基于关联主义的大规模网络开放课程 (MOOC) 及其学习支持 [J]. 远程教育杂志, 2012, 30(3): 31-36

[24] Wing J M. Computational thinking [J]. Communications of the ACM, 2006, 49(3): 33-35

[25] Byrne M D, Catrambone R, Stasko J T. Do Algorithm Animations Aid Learning?; GIT-GVU-96-18 [R]. Georgia Institute of Technology, 1996

[26] Byrne M, Catrambone R, Stasko J. Evaluating animations as student aids in learning computer algorithms [J]. Comput. & Educ., 1999, 33(4): 253-278

[27] Kehoe C, Stasko J, Taylor A. Rethinking the evaluation of algorithm animations as learning aids: An observational study [J]. Hum-Comput Studies, 2001, 54(2): 265-284

[28] Vector B. Inventing on principle [OL]. <https://vimeo.com/36579366>

(上接第 437 页)

[17] Karavirta V. XAAL-extensible algorithm animation language [D]. Finland: Helsinki University of Technology, 2005

[18] Roßling G, Gliesche F, Jahel T, et al. Enhanced expressiveness in scripting using Animal Script [C] // Proceedings of the 3rd Program Visualization Workshop. UK, 2004: 10-17

[19] Roßling G, Schüler M, Freisleben B. The ANIMAL algorithm animation tool [C] // Proceedings of the 5th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education (ITiCSE'00). Helsinki, Finland: ACM Press, 2000: 37-40

[20] Bäsken M, Näher S. GeoWin-A Generic Tool for Interactive Visualization of Geometric Algorithms [M] // Diehl S, ed. Software Visualization: International Seminar. Dagstuhl, Germany: Springer, 2001: 88-100

[21] Karavirta V, Shaffer C A. JSAV: The JavaScript Algorithm Visualization Library [C] // Proceedings of the 18th Annual Conference on Innovation and Technology in Computer Science Education (ITiCSE 2013). Canterbury, UK, 2013