参数配对及 n-way 组合覆盖算法研究

黄 陇[1)] 杨宇航[1)] 李 虎[2)]

[1)](总参陆航研究所 北京 101121)

[2)](中国软件评测中心 北京 100044)

摘 要 组合测试是软件测试数据生成研究的一个重要领域,其中参数配对组合测试的应用最为广泛.对常用的参数配对组合覆盖方法进行了综述分析.目前主流的配对覆盖算法 AETG 和 IPO 所存在的主要问题是在确定水平取值时,具有盲目性和随机性,从而难以控制测试用例的规模.为此提出了改进的 AETG 算法和 IPO 算法,改进算法通过对参数进行预处理以及综合考虑各因素的水平组合等手段,对测试用例的规模进行控制,采用更加完备的方法尽早确定水平取值.为验证新算法的有效性,进行了仿真实验和实际测试,结果表明,改进算法所生成的测试用例数量要少于原算法,测试用例约减效果更为明显;测试用例数与配对数之间、测试用例数与因素水平数之间存在着某些规律性的联系,分析得出了一系列相关的结论.在配对覆盖的基础上,提出了遗传算法与 AETG 算法相结合的 n-way 组合覆盖算法,证明了其时间复杂性较已有算法得到了改善.

关键词 组合测试;配对测试;n-way 覆盖;算法

中图法分类号 TP311 **DOI 号**: 10.3724/SP.J.1016.2012.00257

Research on Algorithm of Parameter Pairwise and n-way Combinatorial Coverage

HUANG Long[1)] YANG Yu-Hang[1)] LI Hu[2)]

[1)](*Institute of Army Aviation*, *Beijing* 101121)

[2)](*China Software Testing Center*, *Beijing* 100044)

Abstract Combinatorial testing is an important part of software test data generation research domain. The parameter pairwise testing is the most widely-used technique. Various parameter pairwise coverage methods for test data generation are surveyed in the paper. The main problem of the existing and dominating pairwise coverage algorithm AETG and IPO is their blindness and randomicity when deciding level value, therefore the size of test cases is difficult to control. Based upon this, improved AETG and IPO algorithms are proposed. In the proposed methods, the size of test cases is effectively controlled by the pretreatment of parameters and the consideration of the level combination of all factors. The level value is decided as early as possible by more consummate way. To validate the effectiveness of new algorithms, emulation experiments and practical testing are carried out. The experiment results show that the number of test cases generated by improved algorithms is smaller than the one by original algorithms, and the reduction effect is more evident; disciplinarian relation exists between test case amount and pairwise amount, also between test case amount and factor level amount. A series of conclusions are drawn by analyzing the results. On the basis of pairwise coverage, n-way combinatorial coverage algorithm which combined with genetic and AETG algorithm is proposed. The fact that time complexity of new algorithm is lower than existing method is also proved.

Keywords combinatorial testing; pairwise testing; n-way coverage; algorithm

收稿日期:2009-09-01;最终修改稿收到日期:2011-10-17.本课题得到国家"八六三"高技术研究发展计划项目基金(2004AA119030)、河北省科技支撑计划项目(10213593)资助.**黄 陇**,男,1975年生,博士,高级工程师,主要研究方向为软件质量与可靠性、软件自动化测试.E-mail: huanglong789@sina.com. **杨宇航**,男,1961年生,博士,高级工程师,主要研究方向为可靠性工程. **李 虎**,男,1974年生,博士,主要研究方向为软件测试、计算机信息系统集成.

1 引 言

问题 Q:在软件功能测试中,针对测试场景中的某个被测方法,当该方法中各个参数的初始取值确定后,如何进行各参数取值的组合以生成测试用例,从而驱动该方法能够执行,并达到最佳的测试效果.解决 Q 的核心在于选取方法参数的哪些取值组合进行实际测试,亦即如何从大量的参数数据组合中(完全组合在实际测试中通常不可行)挑选适量的具有代表性、典型性的数据组合,使得用较少的测试用例即可对被测方法进行较为全面和客观的测试.这种通过检查系统参数的取值组合来进行充分的测试就是组合测试,组合测试已被软件业公认为是一种行之有效的测试方法[1].根据观察,对于很多应用程序来说,很多错误都是由少数几个参数的相互作用导致的.例如 Kuhn 和 Reilly 分析了 Mozilla 浏览器的错误报告记录,发现超过 70%的错误是由某两个参数的相互作用触发的[2].因此在 Q 中应用最为广泛的是配对组合测试,即保证任意 2 个参数的所有可能取值的组合至少被一个测试用例覆盖.目前,学术界从不同角度提出了一些解决该问题的方法,下面先对这些方法进行评述,然后讨论改进的 AETG、IPO 算法,在此基础上,进一步提出一种 n-way 组合覆盖算法.

2 常用的参数配对组合覆盖方法综述

从算法设计的基本原理上看,目前常用的参数配对组合方法可以分为试验设计方法、直接搜索方法和代数方法 3 大类.

2.1 试验设计方法

在试验设计(DOE)中,将方法调用中的各个参数称为因素,而将各个参数的所有可能的取值称为水平.DOE 方法最早是针对硬件系统的测试而提出的[3],但同样可以将其思想应用于软件测试.

(1)全面试验

全面试验即对所有因素的所有水平之间的各种组合情形均进行相应的试验.对于有 k 个因素,每个因素有 $l_i(1\leq i\leq k)$ 个取值的情况,全面试验需要 $\prod_{i=1}^{k} l_i$ 个测试用例.全面试验的优点是各参数的所有取值组合情况都能得到测试,可以发现任何和参数组合相关的错误.但是当 k 和 l_i 都较大时,所需要的测试用例的数量将会非常庞大,这对于资源有限的软件测试来说,通常是不可行的.

(2)单因素轮换试验

为了减少全面试验的次数,出现了单因素轮换试验.它只考虑单个因素的不同取值的变化情况,其它因素则固定不变.此方法虽然产生的测试用例数量较少,但它存在着以下局限性:首先变化因素的选择具有随机性,且取决于人的主观判断;其次当因素个数比较多时,不能有效地覆盖那些固定因素之间的取值组合情况,而这些组合情况却是数量众多的.此方法通常适用于某一因素在执行时起决定作用的情形.

(3)正交试验设计

正交试验设计是一种基于正交表的试验方法[4].正交表是一个二维阵列,其行表示测试用例,每一列表示一个因素,行列交叉点表示因素的某个取值.通常用 $L_n(q^s)$ 表示一个等水平的正交表,其中 L 表示正交设计,n 为测试用例数,即行数;q 表示水平数;s 为因素个数,即列数.例如 $L_9(3^4)$ 表示有 4 个因素,每个因素有 3 个水平,则需要安排 9 次试验即可.

正交试验设计的优点是满足配对组合覆盖的要求,有大量已知的表可直接使用;其局限性是等概率覆盖使得测试用例的数量增多;此方法适用于水平数不多且能够构造出相应的正交表的情形.

(4)均匀试验设计

均匀设计是由中国科学院应用数学研究所方开泰教授和中国科学院王元院士共同提出的一种试验设计方法[5].均匀设计的基本思想就是在确定试验布点时,只考虑试验点的"均匀分散"性,而不考虑其"整齐可比"性,从而大大减少了试验次数.

均匀设计的试验方案是通过均匀设计表来实现的,通常表示为 $U_n(q^s)$,其中 U 表示均匀设计,n 为测试用例数,即行数;q 表示水平数;s 为因素个数,即列数.在实施均匀设计时,首先根据因素数和水平数选择合适的均匀设计表,然后填写该表,以生成相对应的参数数据组合表.当各个因素的水平数不完全相同时,称之为混合水平的均匀设计.对于这种情形,可以采用两种方法加以解决:一是直接选用合适的混合水平均匀设计表;二是采用拟水平技术.此方法的优点是测试用例数量比较少,而其局限性是不能保证满足配对组合覆盖,并且所需要的均匀设计表可能不存在.

2.2 直接搜索方法

(1)AETG 方法

AETG(Automatic Efficient Test Generator)[6]

是美国贝尔实验室的 Cohen 和 Dalal 等人提出的一种启发式两两组合覆盖(也可扩展至 n 维组合覆盖)测试数据生成工具.其基本思想是每次生成一个测试用例,以覆盖最多的未被覆盖的配对组合,直到覆盖所有的配对组合.因此该方法实际上是一种贪心算法.

(2) IPO 方法

IPO(In-Parameter-Order)方法[7],是由美国北卡罗来纳大学计算机系的 Lei 和 Tai 提出的一种启发式配对组合覆盖生成算法,也是一种贪心算法.其基本思想是首先生成任意两个参数的所有配对,然后依次进行水平扩展和垂直扩展.水平扩展即每次新加入一个参数,并确定其取值;在水平扩展之后,通过垂直扩展来补充尚未被覆盖的配对组合.它与 AETG 的根本区别是它不是同时考虑所有参数,而是每次渐进考虑一个参数.

(3) 解空间树方法

文献[8]提出了一种新的启发式配对组合覆盖测试数据生成方法,称之为基于解空间树模型算法.解空间树的第一层树根结点有 b_1 个分支,分别表示参数 p_1 的 b_1 个取值,第 2 层的每个树根结点有 b_2 个分支,分别表示第 2 个参数 p_2 有 b_2 个取值,依次类推,第 n 层的每个树根结点有 b_n 个分支,分别表示第 n 个参数 p_n 有 b_n 个取值.这样从树根开始到底层叶结点的所有路径即构成了配对覆盖测试集 TS_n.

(4) 人工智能方法

采用遗传算法(GA)、模拟退火算法(SAA)等人工智能方法来求解配对组合覆盖测试数据生成问题,也是近年来学术界的一个研究方向.

该类方法的典型代表有:文献[9]使用模拟退火算法产生交互测试用例,其过程是首先任意选择一个初始可行的覆盖矩阵 $\boldsymbol{S}_{N\times K}$($N$ 为初始选择的测试用例数,K 为因素个数);然后改变 $\boldsymbol{S}$ 中某个元素的值,生成 $\boldsymbol{S}'$;若 $cost(\boldsymbol{S}')<cost(\boldsymbol{S})$,则接受 $\boldsymbol{S}'$,否则按 $e^{-(cost(\boldsymbol{S}')-cost(\boldsymbol{S}))/T}$ 的概率接受 $\boldsymbol{S}'$(其中,$cost()$ 为未覆盖的 n-way 组合数,T 为控制温度,其值缓慢变化 $T:=\alpha T(0<\alpha<1)$);当 $cost(\boldsymbol{S}')=0$ 时,算法停止.该方法综合考虑了运算效率和运算结果之间的关系,但只对规模较小的组合覆盖问题可行,当 3-way 覆盖时效果即欠佳;同时初始覆盖矩阵构造困难,并且较为费时.

Shiba 等人[10] 提出将遗传算法和蚁群算法(Ant Colony Algorithm)应用于组合覆盖测试数据生成.对于遗传算法,该文描述了编码、适应性函数、遗传操作等在组合覆盖问题中的可应用性,但并未进行详细的讨论,也没有给出具体的实现算法;对于蚁群算法,实际上是一种图论遍历方法,即每次选择启发值和生物信息值高的线路作为测试用例.

2.3　代数方法

(1) 正交矩阵方法

正交矩阵方法的理论基础是拉丁方和正交拉丁方[11].对于 r 个 n 阶的正交拉丁方,可以构成一个复合矩阵(即正交矩阵),矩阵中的元素是由 r 个 n 阶方阵的对应元素构成的数组,则根据该矩阵,可以得到有 $r+2$ 个参数、每个参数有 n 个取值的配对覆盖组合测试用例.

通常将正交矩阵表示为 $\boldsymbol{OA}_\lambda(N;t,k,v)$[9],该矩阵是 $N\times k$ 矩阵,任何 $N\times t$ 的子阵中 t 列的所有组合都出现 λ 次,对于软件测试而言,只需 $\lambda=1$ 即可.也就是说,N 表示组合测试用例的个数,t 表示 t-way 覆盖,k 表示参数的个数,v 表示各参数的取值个数.Bose[12] 最早提出了 $\boldsymbol{OA}(n^2;2,n+1,n)$ 的构造方法,在此基础上,Bush[13] 又进一步构造出了正交矩阵 $\boldsymbol{OA}(n^d;d,n+1,n)$.

正交矩阵方法有以下局限性:(1) 并不是对于所有的因素和水平组合情形,都存在正交矩阵,特别是对于存在的 n 阶(参数的取值个数)的正交拉丁方(最多有 $n-1$ 个),通过构造正交矩阵,最多可以解决 $n+1(n-1+2)$ 个参数的配对组合覆盖测试数据生成问题;(2) 任两个因素的各水平组合等概率出现,这对于试验设计是必须的,但对于软件测试并不是必须的,造成了测试用例数量的增长;(3) 正交矩阵的构造比较复杂,工作量大;(4) 要求各参数的取值个数相同,使其不具有应用的广泛性.此方法适用于阶数(取值个数)为素数幂的情形.

(2) 整数线性规划方法

根据运筹学的基本原理,可以采用{0,1}整数规划方法来求解配对组合覆盖问题[14-15].其基本思想是:设有 k 个因素,每个因素有 n 个取值,其测试用例总数为 $T=n^k$,分别用 $x_i(i=1,\cdots,T)$ 表示;配对组合数目为 $T'=\binom{k}{2}n^2$,构造整数线性规划表,该表共有 $\binom{k}{2}n^2+1$ 行,n^k+1 列.其中,令

$$\begin{cases} x_j=1,\ j=0,1,\cdots,T-1, & \text{测试用例 } j \text{ 被选中} \\ x_j=0,\ j=0,1,\cdots,T-1, & \text{测试用例 } j \text{ 未被选中} \\ a_{ij}=1,\ i=0,1,\cdots,T'-1, & \text{第 } i \text{ 个成对组合被测试用例 } j \text{ 所覆盖} \\ a_{ij}=0,\ i=0,1,\cdots,T'-1, & \text{第 } i \text{ 个成对组合未被测试用例 } j \text{ 所覆盖} \end{cases},$$

则已选择的所有测试用例为$\sum_{j=0}^{T-1}x_j$，第 i 个成对组合被覆盖的次数为$\sum_{j=0}^{T-1}a_{ij}x_j$. 为了满足配对组合覆盖目标，$\sum_{j=0}^{T-1}a_{ij}x_j$必须大于或等于 1，即每个成对组合至少被覆盖 1 次. 因此，就将成对组合覆盖转化为整数线性规划求解问题. 其约束条件为$\sum_{j=0}^{T-1}a_{ij}x_j\geq 1, i=0,1,\cdots,T'-1$，目标函数为 $\text{Min}\sum_{j=0}^{T-1}x_j$.

该方法虽然构思较为巧妙，但实际求解时，例如有 5 个参数，每个参数取 3 个值，则求解时间就超过 6.5 个小时[14]，因此该方法不实用.

(3) 基于正交表的方法

日本大阪大学的 Kobayashi 和 Tsuchiya 等人[16]提出了一种基于正交表的代数方法. 该方法的基本思想是通过正交表、具有块结构的表和通过近似方法构成出来的混合覆盖表等基本成分之间的交叉重叠算法来构造新的配对组合覆盖表[8]. 由于正交表和块结构的表都是通过有限域来构造的，因此该方法主要考虑的是当因素的取值个数 d 为素数幂的情形，对于不是素数幂的情况，则选择大于 d 且和 d 距离最小的素数幂做近似处理. 但某些情况下，近似处理的误差是比较大的.

(4) TConfig 方法[13-14]

加拿大渥太华大学的 Williams 提出了用覆盖矩阵(覆盖矩阵 ***CA*** 与正交矩阵 ***OA*** 的区别在于前者只要求各配对组合出现 1 次，但不要求等概率出现，而后者则需要等概率出现)方法构造满足成对覆盖要求测试用例的算法. 该算法以正交矩阵为基础，采用递归构造方法，通过不断扩大因素的值来构造覆盖矩阵.

TConfig 方法具有代数推导的精确性，并且已有了相应的自动化工具的支撑. 但该方法的局限性在于由于它使用的是添加构造块的模式，而构造块是基于正交矩阵的，因此它适用于各个因素的水平数相同的情形，对于其它情况则采取比较粗糙的近似方法.

2.4 小 结

由于配对组合覆盖测试数据生成问题是 NP 问题，在某些情况下只能得到近似最优解，因此需要根据实际情况灵活选择不同的方法. 试验设计方法的技术基础是已经存在的正交设计表和均匀设计表，但对于很多情况仍然没有构造出相应的试验设计表，因此该方法实际应用的局限性较大. 而直接搜索方法与代数方法相比较，具有以下优点：(1) 灵活性强. 对于增加或减少因素以及因素取值个数的变化等情况，直接搜索方法可以通过在原来求解结果的基础上进行适当调整与附加运算即可得到新解，而代数方法一般则需要依据新的参数信息重新进行计算；(2) 运算简便. 直接搜索方法大多是通过基础的组合数学方法即可得到解，而代数方法通常需要复杂的矩阵运算方能完成，并且计算量较大；(3) 适应性广. 直接搜索方法可以适用于各种因素与水平相组合的情形，但代数方法由于受到正交矩阵的限制，不具有通用性，只对某些特定类型的因素和水平情形才可解，目前对于水平为素数幂的正交矩阵有着比较好的构造方法，虽然可以通过它来生成新的覆盖矩阵，但与最小覆盖矩阵之间通常有较大的误差；(4) 从实现原理上来看，直接搜索方法比代数方法更贴近于软件测试的基本思想，即用尽可能少的测试用例覆盖尽可能多的配对组合.

在直接搜索方法中，AETG 和 IPO 是两种最为经典、实际应用最为广泛的配对组合覆盖方法. 相对于其他搜索方法，贪心算法的处理速度是最快的，在工业界中的实用性是最强的，并且很多贪心算法都能支持部分非经典的组合测试. 那么在继承良好执行性能的前提下，如何尽可能地减少测试用例的数量，从而减少测试成本，是一个值得深入研究的问题. 而 AETG 和 IPO 算法在生成测试用例时，均具有盲目性和随机性，未从整体上充分地考虑所有因素的各水平之间的组合情形，造成了对测试用例数量的控制效果不佳，并且难以得出规律性、指导性的结论. 为此，本文在 AETG 和 IPO 方法的基础上，分别提出了改进的配对组合覆盖测试数据生成算法.

3 改进的 AETG 算法

3.1 算法的基本思想

改进的 AETG 算法仍然属于贪心算法，但其生成参数值的策略发生了变化. 原算法在依据配对组合生成参数值的过程中所存在的最大问题是具有一定的盲目性和随机性，进而影响到了测试用例的约减效果及算法的效率. 每一组测试用例中首参数的选择对于该组测试用例的生成具有重要的影响，遵循贪心算法中最大覆盖的原则，应当将所有参数的各个取值进行综合考虑，得到全局最优解作为首参数，而不是随机选择，这样就可以尽早地覆盖更多的配对组合，将个别的未覆盖配对进行后续处理，能够更快地实现全配对覆盖；对于每一组测试用例而言，某个参数取值的确定，也应当将其各个水平在未覆

盖配对中出现的次数进行综合考查，即不仅要考虑其与已确定取值的参数的水平的组合情况，也要考虑其与未确定取值的参数的水平的组合情况，这样才能保证每一组测试用例真正实现对最多配对的覆盖.

3.2　改进点

新 AETG 算法同原算法相比，主要作了 3 个方面的改进，如表 1 所示.

表 1　新 AETG 算法的改进点

改进点	原算法	改进算法
1	初始时，各参数未排序，即随机顺序	依各参数取值个数进行降序排序
2	每次随机选择参数作为首参数	选择在未被覆盖的配对组合中出现次数最多的参数作为首参数
3	确定新参数值时，只执行单向匹配(向前匹配)	双向匹配(向前匹配和向后匹配)

3.3　算法描述

输入：k 个参数，每个参数有 $l_i(1\leq i\leq k)$ 个取值，未被覆盖的配对组合集 UP

输出：配对组合覆盖测试用例集 PC

算法：对 k 个参数依其 $l_i(1\leq i\leq k)$ 值的大小进行降序排列，当取值个数相同时，按字典序排列

//将取值个数多的参数排在前面，使其能够较快地收敛到下界 $\max(l_i\times l_j)(1\leq i.\ j\leq k)$

$s=1$;

While $UP\neq\varnothing$ do {

If $s==1$ then do{

选择序列中第一个参数 p_1 及其一个水平值 $v_j(1\leq j\leq l_1)$，使得 v_j 在 UP 中出现的次数最多，若 v_j 均未在 UP 中出现，则按字典序选择第一个水平值 v_1 作为 v_j;}

Else {

选择在 UP 中出现次数最多的水平所属的参数 p_m 作为 p_1，选择其一个水平值 $v_j(1\leq j\leq l_1)$，使得 v_j 在 UP 中出现的次数最多;}

}Endif

If $s<>1$ then do{

生成一个新的参数的逻辑序列，该序列中第一个元素为 p_m，依次考察 p_m 与 $p_1,p_2,\cdots,p_{m-1},p_{m+1},\cdots,p_k$ 的配对组合情况，确定 $p_1,p_2,\cdots,p_{m-1},p_{m+1},\cdots,p_k$ 的取值;}

}Endif

假设参数 $p_1,\cdots,p_j$ 的值已经确定，对于 $1\leq i\leq j$，p_i 的取值为 v_i，则按照下面的方法确定 p_{j+1} 的取值 v_{j+1}：首先执行向前匹配；

//向前匹配

对于 p_{j+1} 的每个取值 v，计算配对组合$\{p_{j+1}=v,p_i=v_i\}(1\leq i\leq j)$ 在 UP 中出现的次数，选择出现次数最多的 v 赋值给 p_{j+1}；若所有 p_{j+1} 的取值 v 与 p_i 的组合均未出现在 UP 中，则进一步执行向后匹配；

//向后匹配

对于 p_{j+1} 的每个取值 v，计算配对组合$\{f_{j+1}=v,p_s=v_s\}(j+1\leq s\leq k)$ 在 UP 中出现的次数，选择出现次数最多的 v 赋值给 p_{j+1}；

将 $p_1,p_2,\cdots,p_k$ 的一组取值作为一个元素加入 PC 中；

$UP=UP-$｛加入 p_{j+1} 所新覆盖的配对组合集｝；

$s++$;

}Endwhile

Return PC.

3.4　实验方案与结果

针对等水平和混合水平的不同情形，分别采用原 AETG 算法和改进的 AETG 算法进行了 10 组实验，生成了相应的测试用例，见表 2.

表 2　改进 AETG 算法与原算法的实验比较

序号	实验方案	配对总数	测试用例数量		约减比率/%
			原 AETG 算法	改进算法	
1	$k=4,\ l_1=l_2=l_3=l_4=3$	54	11	10	9
2	$k=5,\ l_i=3(1\leq i\leq 5)$	90	14	13	7
3	$k=5,\ l_i=6(1\leq i\leq 5))$	360	61	52	15
4	$k=10,\ l_i=2(1\leq i\leq 10)$	40	12	8	33
5	$k=13,\ l_i=3(1\leq i\leq 13)$	702	31	19	39
6	$k=3,\ l_1=l_2=2,\ l_3=3$	16	6	6	0
7	$k=6,\ l_1=2;\ l_2=l_5=3;$ $l_3=6;\ l_4=4;\ l_6=5$	215	35	31	11
8	$k=8,\ l_1=l_5=3;\ l_2=l_7=5;$ $l_3=l_5=l_8=4;\ l_4=2$	390	37	27	37
9	$k=11,\ l_3=l_5=2;\ l_6=5;$ $l_1=l_2=l_4=l_7=l_8=l_9=l_{10}=l_{11}=3$	492	34	20	41
10	$k=21,\ l_1=l_2=l_8=l_{13}=l_{17}=2;$ $l_3=l_7=l_{11}=l_{15}=4;\ l_9=5;$ $l_4=l_5=l_6=l_{10}=l_{12}=l_{14}=3;$ $l_{16}=l_{18}=l_{19}=l_{20}=l_{21}=3$	1944	53	28	47

注：k 为因素个数，$l_i(1\leq i\leq k)$ 为各因素的水平数.

3.5 算法分析

未被覆盖的配对组合集 $Uncover$ 的初始大小为 $|Uncover|=C_k^2 d^2$，其中 $d=\max\{l_i \mid 1\leq i\leq k\}$. 下面考虑最内层的循环操作，$\forall P_i(1\leq i\leq k)$($P$ 表示参数)，确定其每个取值 $v_j(v_j\in P_i)$ 所覆盖的 $Uncover$ 中的元素的个数这一操作的时间复杂性为 $O(|Uncover|)$，因此对于一个参数 P_i 的取值的确定需时间 $O(|Uncover|\times d)$，故在一个测试用例中，确定所有 k 个参数的取值的时间复杂性为 $O(|Uncover|\times d\times k)$，由文献[17]可知，配对组合覆盖的测试用例数的上界是 $\lfloor -\log(k(k-1)d^2/2)/\log(1-1/d^2)+1\rfloor$，因此，确定参数取值操作阶段的时间复杂性为 $O(|Uncover|\times d\times k\times\lfloor -\log(k(k-1)d^2/2)/\log(1-1/d^2)+1\rfloor)$. 算法在执行确定参数取值操作之前，首先进行了 k 个参数的排序，该操作耗时 $O(k\log k)$，因此改进 AETG 算法的总时间复杂性为

$O(|Uncover|\times d\times k\times\lfloor -\log(k(k-1)d^2/2)/\log(1-1/d^2)+1\rfloor+k\log k)$，即 $O(C_k^2 d^3 k\times\lfloor -\log(k(k-1)d^2/2)/\log(1-1/d^2)+1\rfloor+k\log k)$.

另一方面，该算法生成配对组合覆盖测试用例集所依赖的是参数个数及各参数的取值个数，在实际情况中这些值都是有限的，即未被覆盖的配对组合集 UP 总能执行到空集，因此该算法是能终止的.

4 改进的 IPO 算法

4.1 算法的基本思想

改进 IPO 算法仍然采用逐个考虑各个参数的递增策略. 设有 k 个参数，每个参数有 $l_i(1\leq i\leq k)$ 个取值，则满足配对覆盖的测试用例的下界为 $\max(l_i\times l_j)(1\leq i.j\leq k)$，因此按照水平数将各参数进行排序，可以有效提高算法的执行效率；根据水平数最多的两个因素 p_1、p_2 建立起的配对组合测试用例集 PC 的基数要大于 $p_i(i>2)$ 的水平数，若每次加入新因素后都进行垂直扩展，则会引入大量的无关值，不利于后续因素的加入，因此应当先水平扩展至所有参数，再进行一次性垂直扩展，最大限度地减少无关值；在参数 $p_i(1\leq i\leq k)$ 的水平扩展阶段，对于 p_i 之前的无关值，考查其与 p_i 的配对覆盖情况，用未覆盖的配对值取代无关值；在参数 $p_i(1\leq i\leq k)$ 的垂直扩展阶段，可以将其看作是 AETG 算法实施的初始环境，采用改进的 AETG 算法可以进一步约减测试用例.

4.2 改进点

新 IPO 算法同原算法相比，主要作了 3 个方面的改进，如表 3 所示.

表 3 新 IPO 算法的改进点

改进点	原算法	改进算法
1	初始时，各参数未排序，即随机顺序	依各参数取值个数进行降序排序
2	针对每个参数，都进行水平扩展与垂直扩展	先水平扩展至所有参数，再进行一次性垂直扩展
3	在垂直扩展阶段处理无关值	在水平扩展阶段处理无关值，在垂直扩展阶段结合改进 AETG

4.3 算法描述

输入：$p_1\cdots p_n$ 共 n 个参数，每个参数的取值集合 $T_1\cdots T_n$

输出：配对组合覆盖测试用例集 PC

算法：对 n 个参数依其 $|T_i|$ $(1\leq i\leq n)$ 值的大小进行降序排列，当取值个数相同时，按字典序排列；

```
对于前两个参数 p1，p2，建立配对组合测试用例集 PC；
PC={(v1,v2)|v1∈T1，v2∈T2分别是参数 p1，p2的
      水平值}；
If n==2 then Return PC；
/*依次加入新因素，只考虑水平扩展，不考虑垂直扩展*/
Else for i=3 to n do {
  (PC,UP)=call IPO_H(PC,pi)；
            //针对参数 pi做水平扩展，UP 是执行
            //函数调用后未被覆盖的配对组合集
  }Endfor
Endif
//当所有因素当加入后，进行一次性垂直扩展
PC=call IPO_V(PC,UP)；//垂直扩展

Function IPO_H(PC,pi) {
  UP={pi和 p1，p2…pi-1的各参数取值所构成的配
      对组合}；
  s=min(|Ti|,|PC|)；
  for j=1 to s do {
    将参数 pi的第 j 个水平值添加到 PC 中第 j 个测
    试用例的第 i 个位置上；
    UP=UP-{扩展后所新覆盖的配对组合}；
  }Endfor
  If s==|PC| then Return；
  Else for j=s+1 to |PC| do {
    UP'=∅；v'=0；
    For k=1 to |Ti| do {
      UP''={参数 pi的第 k 个水平值 vik添加到 PC
            中第 j 个测试用例的第 i 个位置上所覆
            盖的配对组合}；
      a=|UP''|；b=|UP'|；
      If a>=b then do{
        UP'=UP''；
        v'=vik；
```

}Endif
}Endfor
}Endfor
将参数 p_i 的水平值 v' 添加到 PC 中第 j 个测试用例的第 i 个位置上；
$UP=UP-$｛扩展后所新覆盖的配对组合｝；
//当新加入参数 p_i 的各水平值都已追加到 $|PC|$ 个测
//试用例后，对于 p_i 前面的元素 $p_j(1\leq j\leq i)$ 中的无
//关值做处理
If p_i 是最后一个参数，then 用 p_j 值域中的随机值代替无关值；
Else if $(p_j.w,p_i.u)$ 出现在 UP 中 then
用 w 取代 p_j 的无关值；
Else 用 p_j 值域中的随机值代替无关值；
Endif
Endif
Return(PC,UP)；
Endif
Endfunction.

Function $IPO\_V(PC,UP)$｛
$PC'=\varnothing$；
While $UP\neq\varnothing$ do｛
选择在 UP 中出现次数最多的水平所属的参数 p_m 作为 p_1，选择其一个水平值 $v_j(1\leq j\leq l_1)$，使得 v_j 在 UP 中出现的次数最多；
生成一个新的参数的逻辑序列，该序列中第一个元素为 p_m；
假设参数 $p_1\cdots p_j$ 的值已经确定，对于 $1\leq i\leq j$，p_i 的取值为 v_i，则按照下面的方法确定 p_{j+1} 的取值 v_{j+1}：
首先执行向前匹配
//向前匹配
对于 p_{j+1} 的每个取值 v，计算配对组合｛$p_{j+1}=v,p_i=v_i$｝$(1\leq i\leq j)$ 在 UP 中出现的次数，选择出现次数最多的 v 赋值给 p_{j+1}；
若所有 p_{j+1} 的取值 v 与 p_i 的组合均未出现在 UP 中，则进一步执行向后匹配；
//向后匹配
对于 p_{j+1} 的每个取值 v，计算配对组合｛$f_{j+1}=v,p_s=v_s$｝$(j+1\leq s\leq n)$ 在 UP 中出现的次数，选择出现次数最多的 v 赋值给 p_{j+1}；
将 $p_1,p_2\cdots p_n$ 的一组取值作为一个元素加入 PC' 中；
$UP=UP-$｛新生成的测试用例所新覆盖的配对组合集｝；
$PC=PC\cup PC'$；
Endwhile
Return PC；
Endfunction.

4.4　实验方案与结果

针对等水平和混合水平的不同实例，分别采用原 IPO 算法和改进的 IPO 算法，进行了 10 组实验，生成了相应的测试用例，见表 4.

表 4　改进 IPO 算法与原算法的实验比较

序号	实验方案	配对总数	测试用例数量		约减比率/%
			原 IPO 算法	改进 IPO 算法	
1	$n=4$，$\|T_i\|=7(1\leq i\leq 4)$	294	62	62	0
2	$n=7$，$\|T_i\|=4(1\leq i\leq 7)$	336	33	28	15
3	$n=7$，$\|T_i\|=2(1\leq i\leq 7)$	84	12	7	42
4	$n=8$，$\|T_i\|=2(1\leq i\leq 8)$	112	14	8	43
5	$n=10$，$\|T_i\|=3(1\leq i\leq 10)$	405	28	19	32
6	$n=3$，$\|T_1\|=\|T_2\|=2$，$\|T_3\|=3$	16	6	6 （无需做垂直扩展）	0
7	$n=6$，$\|T_1\|=\|T_6\|=3$，$\|T_2\|=2$；$\|T_3\|=5$； $l_4=4$；$l_5=8$	249	41	40 （无需做垂直扩展）	2
8	$n=6$，$\|T_1\|=\|T_6\|=3$，$\|T_2\|=\|T_3\|=4$； $\|T_4\|=\|T_5\|=5$	238	27	25 （无需做垂直扩展）	7
9	$n=7$，$\|T_1\|=\|T_6\|=5$，$\|T_2\|=\|T_5\|=3$； $\|T_3\|=\|T_7\|=7$；$\|T_4\|=8$	507	80	61	24
10	$n=8$，$\|T_1\|=2$，$\|T_2\|=4$；$\|T_3\|=\|T_4\|=3$； $\|T_5\|=5$；$\|T_6\|=\|T_7\|=2$；$\|T_8\|=6$	167	33	30 （无需做垂直扩展）	9

4.5　算法分析

未被覆盖的配对组合集 $Uncover$ 的初始大小为 $|Uncover|=C_n^2d^2$，其中 $d=\max\{|T_i||1\leq i\leq n\}$. 在水平扩展阶段，产生的测试用例数为 d^2 个，$\forall P_i(3\leq i\leq n)$（$P$ 表示参数），确定 P_i 取值这一操作最多可耗时 $O(|Uncover|\times(d^2-1))$（最多需进行 d^2-1 次不同取值能覆盖 $Uncover$ 中元素个数的比较）. 因此对于 k 个参数而言，其时间复杂性为 $O(|Uncover|\times(d^2-1)\times(n-2))$，在垂直扩展阶段，需要新生成的测试用例数量最多为 $\lfloor-\log(n(n-1)d^2/2)/\log(1-1/d^2)+1\rfloor-d^2$ 个，因此确定参数取值操作阶段的时间复杂性为 $O(|Uncover|\times d\times$

$n\times(\lfloor-\log(n(n-1)d^2/2)/\log(1-1/d^2)+1\rfloor-d^2))$. 算法在执行确定参数取值操作之前，首先进行了 n 个参数的排序，该操作耗时 $O(n\log n)$，因此改进的 IPO 算法的总时间复杂性为

$O(|Uncover|\times(d^2-1)\times(n-2)+|Uncover|\times d\times n\times(\lfloor-\log(n(n-1)d^2/2)/\log(1-1/d^2)+1\rfloor-d^2)+n\log n)$，即

$O(C_n^2d^2(d^2n-2d^2-k+2+dn)\times(\lfloor-\log(n(n-1)d^2/2)/\log(1-1/d^2)+1\rfloor-d^2)+n\log n)$.

另一方面，该算法生成配对组合覆盖测试用例集所依赖的是参数个数及各参数的取值个数，在实际情况中这些值都是有限的，即在水平扩展和垂直扩展阶段所执行的循环都是可以终止的，因此该算法是能终止的.

5 实验结果分析及结论

通过以上实验可以直观地看出，改进的 AETG 和 IPO 算法所生成的测试用例的数量在总体上要优于原算法，虽然在某些情形下约减的测试用例数量不是很多，但对于软件测试而言，即使约简数量很少的测试用例，对测试成本的节约和效率的提高也具有不可忽视的作用[18]. 同时，由于改进算法仍然属于贪心算法，其实际执行性能的优越性得到了保证. 即改进算法在保持原算法良好执行性能的基础上，提高了计算结果的精确度. 另一方面，改进 AETG 算法直到未被覆盖配对组合集为空才停止执行，而改进 IPO 算法在最终的垂直扩展阶段也直到未被覆盖配对组合集为空才停止执行，因此两种算法都保证了所生成的测试用例实现了对所有配对组合的覆盖，即改进算法的测试覆盖充分性与原算法是等价的，因此其揭示错误的能力也保持等价.

5.1 测试用例数量约减效果

根据上述实验结果，得到两个改进算法与原算法对于测试用例数量的约减情况，如图 1 和图 2 所示.

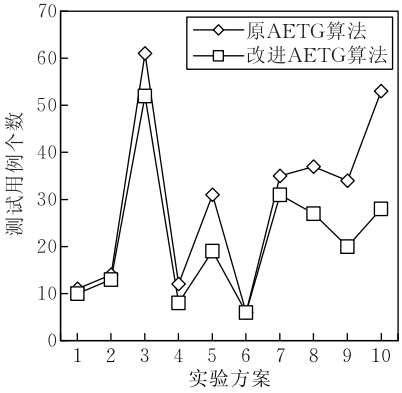


图 1 改进 AETG 算法与原算法测试用例数量比较

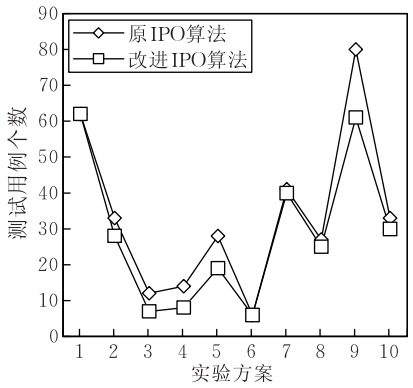


图 2 改进 IPO 算法与原算法测试用例数量比较

改进 AETG 算法在等水平和非等水平两种情形对测试用例的约减效果见图 3 和图 4.

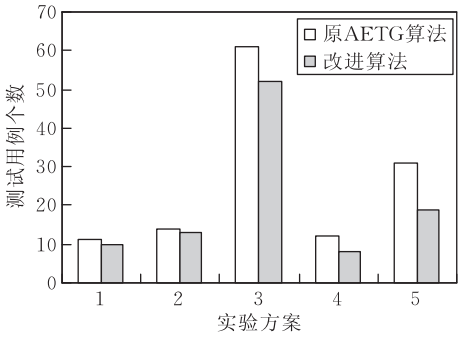


图 3 等水平时改进 AETG 算法的约减效果

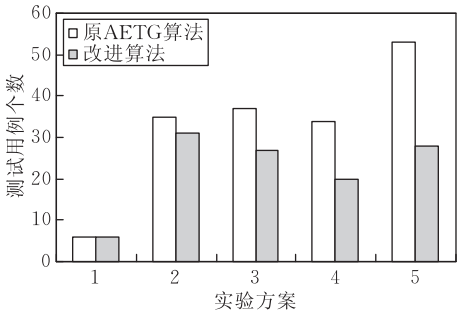


图 4 非等水平时改进 AETG 算法的约减效果

改进 IPO 算法在等水平和非等水平两种情形对测试用例的约减效果见图 5 和图 6.

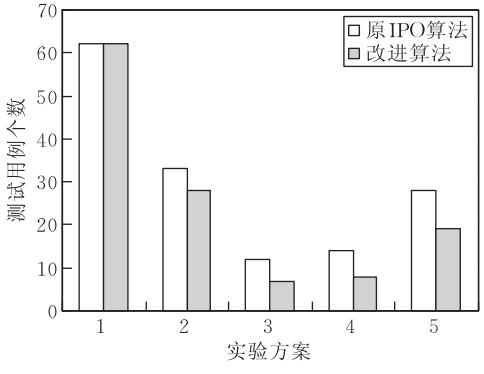


图 5 等水平时改进 IPO 算法的约减效果

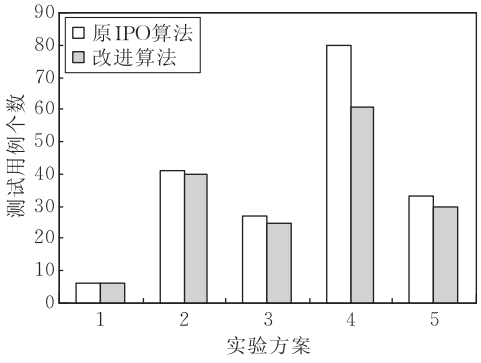


图 6　非等水平时改进 IPO 算法的约减效果

通过分析，可以得出下述结论：

结论 1. 对于 AETG 算法，当满足等水平情形时，水平数越多，则改进算法与原算法相比对测试用例数量的约减效果越明显，因素数越多，约减效果越明显；当满足非等水平时，因素数越多，则约减效果越明显.

结论 2. 对于 IPO 算法，当满足等水平情形时，因素数越多，约减效果越明显，水平数越少，则约减效果越明显；当满足非等水平时，各因素水平相差越小，或各因素的水平总数越多，则约减效果越明显，因素数越多，约减效果越明显.

5.2　测试用例数与配对总数的关系

通过对图 7～10 进行分析，可以得出如下结论：

结论 3. 对于 AETG 算法，当满足等水平情形时，在一般情况下，配对总数增长，则测试用例的数量也增长，二者成正比关系；当满足非等水平时，测试用例数与配对总数无固定的正反比关系，测试用例的数量关键取决于因素水平的个数.

结论 4. 对于 IPO 算法，当满足等水平情形时，测试用例数与配对总数无固定的正反比关系，测试用例的数量关键取决于因素水平的个数；当满足非等水平时，在一般情况下，配对总数增长，则测试用例的数量也增长，二者成正比关系.

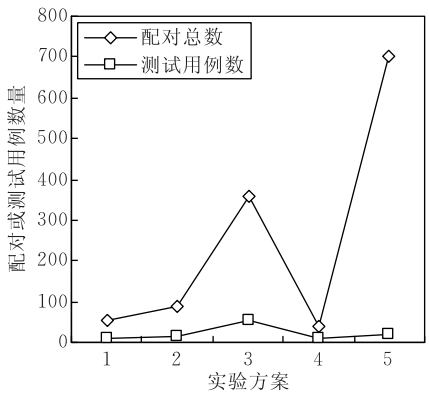


图 7　改进 AETG 算法中配对总数与测试用例数的关系(等水平)

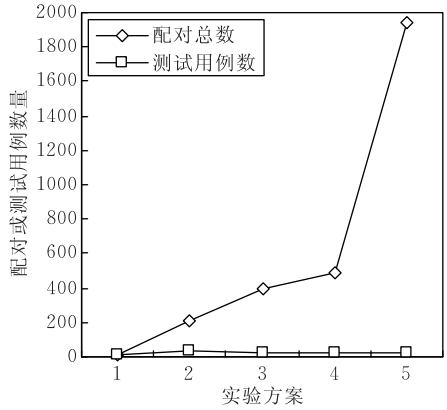


图 8　改进 AETG 算法中配对总数与测试用例数的关系(非等水平)

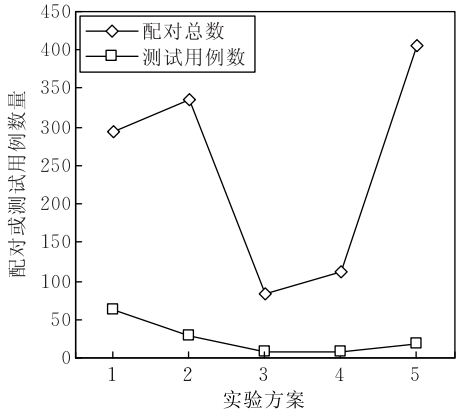


图 9　改进 IPO 算法中配对总数与测试用例数的关系(等水平)

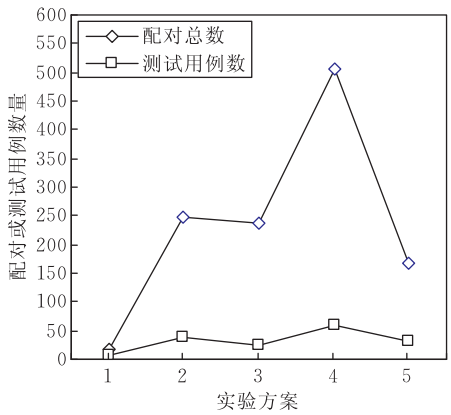


图 10　改进 IPO 算法中配对总数与测试用例数的关系(非等水平)

5.3　测试用例数与因素数水平数的关系

依据图 1～图 6，从另一个角度进行分析，即可得到改进算法生成的测试用例数与不同因素数水平数组合之间的关系，有以下结论：

结论 5. 对于 AETG 算法，当满足等水平情形时，首先考虑水平数，水平数越少，则生成的测试用例越少，当水平数相同时，因素数越少，则测试用例

越少;当满足非等水平时,不同水平的个数越少(即越接近于等水平),测试用例越少,当不同水平的个数相同时,因素数越多,则测试用例越多.

结论 6. 对于 IPO 算法,当满足等水平情形时,水平数增多,则生成的测试用例明显增多,当因素数相同时,水平数微小变化,则测试用例将大幅增长,而当水平数相同时,因素数小幅增加,则测试用例也小幅增加;当满足非等水平时,在一般情形下,各因素的水平总数越多,则测试用例也越多.

6 实例验证

在对某遥控软件[19]的测试过程中,分别应用原 AETG、IPO 算法以及改进的 AETG 和 IPO 算法生成了测试用例. 该软件的定时开功能的执行需要 3 个参数:时钟类型、受控设备类型以及开启时间. 其中各参数的类型和取值如下:

时钟类型:字符型,取值为系统时钟和自定义时钟;

受控设备类型:字符型,取值为电器设备、工控设备、包装设备、检测设备和照明设备;

开启时间:数值型,取值为 10 s, 99 s, 100 s, 101 s, 600 s.

上述 3 个参数的配对总数为 45,使用原 AETG 方法生成了 28 个测试用例,而应用改进的 AETG 算法则只需要 25 个测试用例;虽然使用原 IPO 方法和改进的 IPO 算法均生成了 25 个测试用例,但由于后者无需进行垂直扩展,因此生成测试用例的速度要优于前者. 使用改进算法生成的测试用例进行实际测试,取得了较好的效果.

值得指出的是,改进的 AETG 和 IPO 算法对于参数配对组合覆盖测试而言,各有其不同的适用场合. 改进的 AETG 算法适用于参数个数比较稳定的情形,而对于参数个数经常变化的情形,则使用改进的 IPO 算法效果较好.

7 *N*-way 组合覆盖算法

按照 Nair 等人[20]的观点,对于规模较小的问题,配对覆盖就能够发现大多数错误,并且 AETG 的测试实践也给予了证明[21]. Dunietz 等人①[22]认为 3-way 覆盖可以有效地进行代码块覆盖,而更高阶的覆盖可以实现代码路径的覆盖. 因此,在前面分析了配对组合覆盖的基础上,有必要进一步讨论 n-way 组合覆盖测试数据生成问题. 目前针对 n-way 组合覆盖算法的研究大多是以现有的配对组合覆盖算法为基础,将其进行扩展,以推广到 n-way 的情形. 但随着参数个数的增多,其算法的时间复杂性也急剧增加.

Williams[14]在其博士论文中提出可采用正交阵列 $\boldsymbol{OA}(\rho^d, n+1, n, d)$ 作为初始构造块,通过水平和垂直方向的矩阵连接运算来建立 d-way 的覆盖测试用例矩阵. 但当使用 $\boldsymbol{OA}(8,3,2,3)$ 时,实验结果表明丢失了 72 个 3-way 覆盖测试用例,造成误差较大. 因此该方法需要进一步完善.

文献[8]基于 AETG 和 IPO 方法,分别将其推广到 n-way 组合覆盖的情形,提出了两种 n-way 组合覆盖测试数据生成算法,并进行了实验分析. 但是随着维数 n 的增长,该文所提出的两种算法生成测试用例所耗费的时间也迅速的增长,因此需要进一步研究时间复杂性更低的新算法.

Ahmed[22]在其博士论文中系统地讨论了 DOE 方法在软件测试中的应用,但该文未解决 n-way 参数交互及混合水平参数组合的问题.

通过分析可以看出,目前 n-way 组合覆盖算法存在的主要问题是时间复杂性过高,因此在保证 n-way 组合覆盖目标的前提下,需要降低算法的时间复杂性. 对于 n-way 覆盖问题,从另一个角度,我们可以将 k 个参数的不同取值组合看作是对 k 个数值的一个编码排序问题,因此可以借鉴遗传算法的基本思想来解决该问题. 由于 n-way 覆盖问题是一个 NPC 问题[23],因此在使用遗传算法时,为了提高搜索速度和精度,还需要采用启发式方法加以补充. 为此,本文提出了一种将遗传算法和 AETG 方法相结合的算法,较好地解决了 n-way 覆盖测试数据生成问题,并通过与文献[8]中的方法进行比较,说明新方法的时间复杂性得到了改善.

7.1 算法描述

输入:设有 k 个参数,$P_1, P_2, \cdots, P_i (1 \leq i \leq k)$,每个参数的取值集合为 D_i,为了讨论方便,不妨设 $|D_1| = |D_2| = \cdots = |D_i| = d$

所有未被覆盖的 n-way 组合集 *Uncover*

输出:n-way 组合覆盖测试用例集 T

算法:确定初始种群规模 $N = d$,形式为

$N_S = SS\cdots S (1 \leq S \leq d)$, $Pi = S (1 \leq i \leq k)$;

① Dunietz I S, Ehrlich W K, Szablak B D, Mallows C L, Iannino A. Applying systematict factor covering experimental designs to software testing. Submitted for Publication, 1997

将 d 个染色体放入 T 中；
初始化染色体组规模 $g=d$；
GA：While $g<d^n$ do { //d^n是使用确定性遗传算法生
　　//成的测试用例的下限
从已有染色体组中随机选择两个父代染色体做杂交运算，杂交率 $P_c=2/g$，杂交算子为单点互换杂交，杂交点为 $n-1$ 基因座；
对于产生的两个杂交子代，分别计算各自所覆盖的 *Uncover* 中元素的个数 R；
执行选择算子，即选择个数 R 大者作为适宜子代加以保留，余者淘汰；
将适宜子代加入 T 中；
$g++$；
Uncover=*Uncover*−{被新染色体所覆盖的 n-way 组合}；
If $g=d^n$ then goto AETG
Else
从已有染色体组中随机选择两个父代进行变异，变异率 $P_m=d/g$；
/* 开始时染色体的全部基因都被变异，随着种群规模的扩大，变异基因的数量逐渐减少 */
变异算子为均匀变异，新的等位基因值为 $INT(1+r(d-1))$，r 是[0,1]内符合均匀分布的随机数；
对产生的两个变异子代，分别计算各自所覆盖的 *Uncover* 中元素的个数；
执行选择算子，即选择个数大者作为适宜子代加以保留，余者淘汰；
将适宜子代加入 T 中；
$g++$；
Uncover=*Uncover*−{被新染色体所覆盖的 n-way 组合}；
Endif
Endwhile
AETG：While *Uncover*<>∅ do{
选择参数 $P_1\,P_2\cdots P_{n-1}$ 的一个 $n-1$ 维取值组合 $(v_1v_2,\cdots,v_{n-1})$，使得该组合在 *Uncover* 中出现的次数最多；
/* 将 $n-1$ 个参数作为一个整体，即看作一个参数，则可类比为配对组合问题 */
令 $P_1=v_1$，$P_2=v_2$，…，$P_{n-1}=v_{n-1}$，确定余下的 $k-n+1$ 个参数的取值；
For $i=n$ to k do{
For $j=1$ to d do{
考察 P_j 与 $(v_1v_2,\cdots,v_{i-1})$ 中的 $n-1$ 个元素组成的 n-way 组合，共有 C_{i-1}^{n-1} 个；
计算各个组合所能覆盖的 *Uncover* 中的个数 t_j；
Endfor
选择 $\max\{t_j\}$；
$v_i=P_j$；
Endfor
生成新的测试用例 $(v_1v_2,\cdots,v_k)$，将其加入 T 中；
Uncover=*Uncover*−{被新测试用例所覆盖的 n-way 组合}；
Endwhile

7.2　时间复杂性分析

文献[8]中提出的 n-way 覆盖算法，是将 AETG 算法由配对覆盖扩展而成的. 下面将本文所提 n-way 覆盖算法(GA-AETG)的时间复杂性(记为 $O(G)$)与文献[8]中 n-way 覆盖算法的时间复杂性(记为 $O(A)$)进行分析比较.

Uncover$=C_k^n\times d^n$，由文献[8]知 $O(A)=C_k^{n2}\times d^{2n+1}\times k$.

在 GA-AETG 中，通过先执行 GA 算法，使 *Uncover* 得到减少：

$$Uncover'=Uncover-C_k^n\times d.$$

继续执行 AETG 过程后，$O(G)$为 $O(G)=(Uncover')^2\times k\times d=[C_k^n\times(d^n-d)]^2\times k\times d$，则时间复杂性减少量为

$$\begin{aligned}O(A)-O(G)&=(C_k^n)^2d^{2n+1}k-C_k^{n2}kd(d^n-d)^2\\&=(C_k^n)^2k(d^{2n+1}-d^{2n+1}+2d^{n+2}-d^3)\\&=(C_k^n)^2k(2d^{n+2}-d^3).\end{aligned}$$

另一方面，由于在 GA-AETG 中首先执行 GA 操作，造成新增加的时间复杂性为

$O(G')=C_k^n\times d^n\times 2\times d^n=2C_k^nd^{2n}$(因为杂交、变异操作均要进行两个染色体适应值的比较).

因为 $n\geq 2$，所以

$O(A)-O(G)\geq(C_k^n)^2\times k\times d^4=O(G'')$，因此只要证明 $O(G'')>O(G')$即可.

下面采用数学归纳法加以证明.

当 $n=2$ 时：

$O(G'')=(C_k^2)^2kd^4$，$O(G')=2C_k^2d^4$，

因为 $k\geq 2$，显然有 $O(G'')>O(G')$.

设当 $n=p$ 时，有 $O(G'')>O(G')$，

则当 $n=p+1$ 时，由于 $(C_k^p)^2kd^4>2C_k^pd^{2p}$，

所以，$C_k^pkd^4>2d^{2p}$，即

$\dfrac{(p+1)C_k^{p+1}kd^4}{k-p}>2d^{2p}$，即

$\dfrac{(p+1)C_k^{p+1}kd^4}{(k-p)2d^{2p}}>1$，因此

$\dfrac{C_k^{p+1}kd}{2d^{2p}}\times\dfrac{(p+1)d^3}{k-p}>1.$

由于要保证对任意随机变化的 d,k,p 均成立，

因此必有 $\dfrac{C_k^{p+1}\times k\times d}{2d^{2p}}>1$

即 $C_k^{p+1}\times k\times d^4>2d^{2(p+1)}$

有 $(C_k^{p+1})^2\times k\times d^4>2C_k^{p+1}\times d^{2(p+1)}$

即 $O(G'')>O(G')$ 得证.

通过以上分析可以看出,本文所提 n-way 覆盖算法(GA-AETG)在满足参数 n-way 组合覆盖的基础上,其时间复杂性较文献[8]中算法的时间复杂性得到了优化与改善.

8 结束语

本文首先对常用的参数配对组合覆盖方法进行了归纳与整理,分析比较了各种方法的优缺点及使用时机,用于指导实际的组合测试工作;对 AETG 算法和 IPO 算法分别进行了改进,实验结果表明新算法所生成的测试用例的数量要优于原有算法,即在保持原算法良好执行性能的基础上,提高了计算结果的精确度;基于实验结果,分析了测试用例数与配对组合总数、测试用例数与因素水平数之间存在的联系;在配对组合覆盖的基础上,进一步提出了基于遗传算法和 AETG 的 n-way 覆盖算法,证明了其时间复杂性较已有算法得到了改善.需要进一步研究的问题包括:考虑参数之间具有相互作用的情形,如何生成组合测试用例;开发相应的自动化组合测试工具等.

参 考 文 献

[1] Yan Jun, Zhang Jian. Combinatorial testing: Principles and methods. Journal of Software, 2009, 20(6): 1393-1405(in Chinese)
(严俊, 张健. 组合测试:原理与方法. 软件学报, 2009, 20(6): 1393-1405)

[2] Kuhn D R, Reilly M J. An investigation of the applicability of design of experiments to software testing//Caulfield M ed. Proceedings of the Annual NASA/IEEE Software Engineering Workshop (SEW). Los Alamitos: IEEE Press, 2002: 91-95

[3] Box G E P, Hunter J S. The Design of Experiments. London: Oliver and Boyd, 1965

[4] Fang Kai-Tai, Ma Chang-Xing. Orthogonal and Uniform Experiment Design. Beijing: Science Press, 2001(in Chinese)
(方开泰, 马长兴. 正交与均匀试验设计. 北京: 科学出版社, 2001)

[5] Fang Kai-Tai. Uniform Design and Uniform Design Table. Beijing: Science Press, 1994(in Chinese)
(方开泰. 均匀设计与均匀设计表. 北京: 科学出版社, 1994)

[6] Cohen David M, Dalal Siddhartha R, Fredman Michael L, Patton Gardner C. The AETG system: An approach to testing based on combinatorial design. IEEE Transactions on Software Engineering, 1997, 23(7): 437-444

[7] Lei Y, Tai K C. In_parameter_oder: A test gene-ration strategy for pairwise testing. Department of Computer Science, North Carolina State University, Raleigh, North Carolina: Technical Report TR-2001-03, 2001

[8] Nie Chang-Hai. Research on test case set reduction technique [Ph. D. dissertation]. Southeastern University, Nanjing, 2003(in Chinese)
(聂长海. 测试用例集约简技术研究[博士学位论文]. 东南大学, 南京, 2003)

[9] Cohen Myra B, Colbourn Charles J, Ling Alan C H. Augmenting simulated annealing to build interaction test suites// Proceedings of the 14th International Symposium on Software Reliability Engineering (ISSRE'03). 2003

[10] Shiba Toshiaki, Tsuchiya Tatsuhiro, Kikuno Tohru. Using artificial life techniques to generate test cases for combinatorial testing. Proceedings of the 28th Annual International Computer Software and Applications Conference (COMPSAC'04). 2004

[11] Lu Kai-Cheng, Lu Hua-Ming. Combinatorics. 3rd Edition. Beijing: Tsinghua University Press, 2002(in Chinese)
(卢开澄, 卢华明. 组合数学(第 3 版). 北京: 清华大学出版社, 2002)

[12] Bose R C. On the application of the properties of Galois fields to the construction of hyper Graeco-Latin squares. Sankhya, 1938, 3: 323-338

[13] Bush K A. Orthogonal arrays of index unity. Annals of Mathematical Statistics, 1952, 23: 426-434

[14] Williams Alan Webber. Software component interaction testing: Coverage measurement and generation of configurations [Ph. D. dissertation]. University of Ottawa, 2002

[15] Yu Xiu-Shan, Yu Hong-Min. The New Technique and Practice of Software Testing. Beijing: Publishing House of Electronics Industry, 2006(in Chinese)
(于秀山, 于洪敏. 软件测试新技术与实践. 北京: 电子工业出版社, 2006)

[16] Kobayashi Noritaka, Tsuchiya Tatssuhio, Kikuno Tohru. A new method for constructing pair-wise covering designs for software testing. Information Processing Letters, 2002, 81(2): 85-91

[17] Zhu Xiao-Jun. Research and implementation of software testing method based on parameter pairwise combination[M. S. dissertation]. Shanghai Normal University, Shanghai, 2004 (in Chinese)
(朱小骏. 参数配对组合的软件测试方法研究与实现[硕士学位论文]. 上海师范大学, 上海, 2004)

[18] Wang Zi-Yuan, Nie Chang-Hai, Xu Bao-Wen et al. Optimization generation method of combination testing case set for neighboring factors. Chinese Journal of Computers, 2007, 30(2): 200-211(in Chinese)

(王子元，聂长海，徐宝文等. 相邻因素组合测试用例集的最优生成方法. 计算机学报，2007，30(2)：200-211)

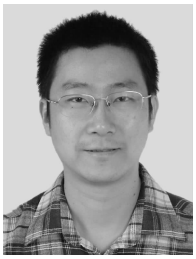
[19] Huang Long. Design and implemtation of banausic infrared remote control multifunctional switch. Journal of Wireless Engineering，2003，33(2)：23-25(in Chinese)
(黄陇. 实用型红外遥控多功能开关的设计与实现. 无线电工程，2003，33(2)：23-25)

[20] Nair V N，James D A，Ehrlich W K，Zevallos J. A statistical assessment of some software testing strategies and application of experimental design techniques. Statistica Sinica，1998，8：165-184

[21] Cohen D M，Dalal S R，Kajla A，Patton G C. The automatic efficient test Generator (AETG) system//Proceedings of the 5th International Symposium Software Reliability Engineering，Los Alamitos，CA，1994

[22] Ahmed Mohamed Salem. A software testing model：Using design of experiments (DOE) and logistic regression[Ph. D. dissertation]. Florida Institute of Techinology，2001

[23] Lei Y，Tai K C. In-parameter-order：A test generation strategy for pairwise testing//Proceedings of the 3rd IEEE International Symposium on High Assurance Systems Engineering. 1998：254-261



HUANG Long，born in 1975，Ph. D.，senior engineer. His research interests include software quality and reliability，software automation testing.

YANG Yu-Hang，born in 1961，Ph. D.，senior engineer. His research interest reliability engineering.

LI Hu，born in 1974，Ph. D.. His research interests include software testing，computer information system integration.

Background

This paper is supported by the National High-Tech Research Subjects and Development Plan of China under Grant Nos. 2004AA119030. The background of this paper is research of model driven Web Service test data generation. This project mainly research on the follows fields：WSDL document parsing，model adapter generation，the transformation from design model to testing model and test data generation based on U2TP. This paper focus on the research of combinatorial testing，that is how to generate the least test cases which coverage all the parameter pairwise coverage. On the basis of pairwise coverage，this paper also presents an n-way combinatorial coverage algorithm.