第 36 卷 第 9 期 计 算 机 工 程 2010 年 5 月
Vol.36 No.9 Computer Engineering May 2010
·软件技术与数据库· 文章编号：1000—3428(2010)09—0053—03 文献标识码：A 中图分类号：TP301.6

3^n 镜像适应性随机测试算法

聂剑平，钱越英，陈润强
(北京应用气象研究所，北京 100029)

摘 要：针对适应性随机测试中的边缘效应问题，借鉴镜像适应性随机测试(MART)的基本思想，在引入镜像距离的基础上，提出 3^n MART 算法，通过改变候选用例与成功用例之间的距离判断策略，使其产生的用例在输入域中更均匀地分布。仿真实验结果证明，该算法可以较好地解决边缘效应问题，相比基于距离的适应性随机测试算法与 MART 算法，失效发现效率更高。

关键词：随机测试；适应性随机测试；测试用例

3^n Mirror Adaptive Random Testing Algorithm

NIE Jian-ping, QIAN Yue-ying, CHEN Run-qiang
(Beijing Institute of Applied Meteorology, Beijing 100029)

【Abstract】 Aiming at the problem of boundary effects in Adaptive Random Testing(ART), based on the basic thought of Mirror ART(MART), this paper proposes a new algorithm named 3^n MART by introducing mirror distance. Testing cases generated by the algorithm are more evenly spread in the input domain by changing the distance judgment strategy between the candidate testing cases and the successful testing cases. Simulation experimental result verifies that 3^n MART algorithm can solve the problem of boundary effects, and its efficiency of finding failure is higher than that of DART algorithm and MART algorithm.

【Key words】 Random Testing(RT); Adaptive Random Testing(ART); testing case

1 概述

在众多的测试用例选择策略中，随机测试(Random Testing, RT)是一种相对简单但十分有用的方法[1]。它避免了复杂的软件需求或软件结构的分析过程，仅仅在输入域中随机地选择测试用例，因此，用例产生过程简单，且易于自动化。尽管 RT 已被成功地应用于许多实际项目中，但有些研究者认为，它在选择测试用例的过程中很少或几乎没有用到软件需求或代码的任何信息[2]。以此为出发点，文献[3-4]利用程序失效的某些特性，提出了适应性随机测试(Adaptive Random Testing, ART)方法，提高了 RT 的效率。

根据 ART 的基本原理，其改进的算法被相继提出，如基于距离的适应性随机测试(DART)、有限随机测试(RRT)、局部适应性随机测试(RPRT)和二分随机测试(BPRT)。它们可以被粗略地分为 3 大类：最优法，排除法和局部法[2]。其中，DART 是一种较为典型且应用较广泛的适应性随机测试算法。文献[2]针对适应性随机测试算法普遍存在计算量较大的现象，提出了镜像适应性随机测试(Mirror ART, MART)算法。MART 在保证几乎相等的失效发现效率的基础上，极大地减少了 ART 算法的计算量，但它并没有克服 ART 算法的另一个缺点——边缘效应，即偏好产生靠近输入域边缘的测试用例。本文以此为出发点，基于镜像适应性随机测试算法的基本原理，以基于距离的适应性随机测试为例，提出 3^n 镜像适应性随机测试(3^n MART)算法。

2 适应性随机测试

引起失效的输入通常集中在一个或多个区域[5]。ART 就是以此为基础提出来的。通常，失效域形状可以分为 3 类：块状，条状和点状分布(失效域形状参见文献[5])。ART 通过利用成功用例(未发现失效的已执行用例)的空间分布确保测试用例在输入域尽可能均匀分布，且彼此间距离较远，从而提高了发现失效的效率，即减少了发现第 1 个失效所需要的测试用例数 *F-measure*。

2.1 基于距离的适应性随机测试

基于距离的适应性随机测试是第 1 个 ART 算法。其使用了 2 个用例集：成功用例集 $S=\{s_1,s_2,\cdots,s_l\}$ 和候选用例集 $C=\{c_1,c_2,\cdots,c_k\}$。S 中包含所有已经执行但没有发现失效的测试用例；C 中包含 k 个从输入域中随机产生的候选用例，下一个测试用例将从 C 中产生。通常 k 为定值，若无特别说明，本文中 $k=10$。首先计算每个候选用例 c_i 与成功用例集 S 之间的距离 d_i，d_i 为 c_i 与每个成功用例 s_j 之间距离的最小值。然后选择 d_i 最大的候选用例 c_q 作为下一个测试用例。

2.2 镜像适应性随机测试

镜像适应性随机测试的基本思想如下：

(1)按照一定的划分方法，将输入域分为 h 个同等大小和相同形状的子域，使用 ART 算法在其中一个子域(称为源子域)中产生一个测试用例 c_q。

(2)使用一个简单的映射函数，在剩余的子域中分别产生一个 c_q 的映像 I_i，$i=1,2,\cdots,h-1$。

(3)将 c_q 和各个子域产生的 I_i 作为一个测试用例集，按照一定顺序分别执行。如无特别说明，本文采用的执行顺序是

作者简介：聂剑平(1978－)，男，博士后，主研方向：软件测试，软件质量管理；钱越英，研究员、博士后合作导师；陈润强，工程师
收稿日期：2009-11-20 **E-mail：**nicknjp@126.com

先执行源子域中的用例，然后从上到下、由左至右顺序执行其他子域中的用例。

使用映射方法产生测试用例比 ART 算法简单得多。实验证明，在输入域划分较少的情况下，MART 的 *F-measure* 近似等于 ART 的 *F-measure*[2]。

2.3 边缘效应

由于 ART 的基本原理是利用成功用例的空间分布来确保测试用例在输入域均匀分布，且彼此间距离较远，而用例选择必须在输入域中进行，因此距离输入域中心位置较远的用例被选择的可能性明显大于距离输入域中心位置较近的用例[2]。即失效域位于输入域的边缘位置时，*F-measure* 较小；当失效域位于输入域的中央位置时，*F-measure* 较大。这就是 ART 的边缘效应。MART 虽然计算量比 ART 算法小，但由于其在源子域中应用了 ART 算法，而其他子域中产生的用例又是由源子域中的用例映射而来，因此 MART 算法在各个子域中都存在边缘效应。

3 适应性随机测试算法的改进

3.1 改进的基本思想

如 2.3 节所述，边缘效应产生的原因是 ART 产生的测试用例必须保证彼此间距离较远，而测试用例又必须存在于输入域中。借鉴 MART 的基本思想，本文提出镜像距离判断的概念。以二维输入域为例，如图 1 所示。

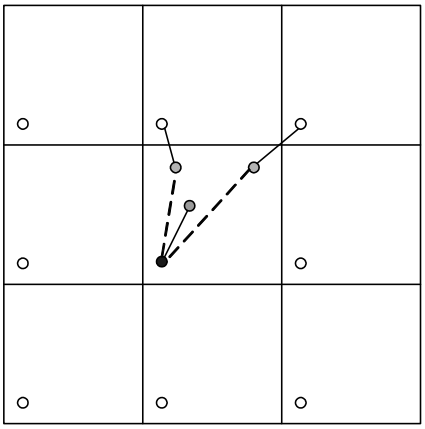


图 1　二维输入域中改进算法示意图

首先，将输入域划分为大小、形状相同的 9 个子域 D_i，$i=1,2,\cdots,9$，在中心子域 D_5 中应用 DART 算法，只在判断候选用例与成功用例之间的距离时稍做改变。黑色圆点 *s* 代表成功用例；白色圆点代表利用直接线性映射方法在各个子域中产生的成功用例的镜像点；灰色圆点代表候选用例。应用 MART 算法从候选用例中选择下一个用例时，判断候选用例与成功用例之间的距离，即图中的虚线长度。改进的算法则进行镜像距离计算，也就是计算候选用例与成功用例及其镜像点集中距离最近的点(有效镜像点)之间的距离，即图中的实线长度。候选用例 c_1 与成功用例 *s* 在 D_2 中的镜像点 s_2' 距离最近，其有效镜像点为 s_2'；候选用例 c_2 与成功用例 *s* 距离最近，其有效镜像点为成功用例 *s* 本身；候选用例 c_3 与成功用例 *s* 在 D_3 中的镜像点 s_3' 距离最近，其有效镜像点为 s_3'。因为不同的成功用例在其各个子域的镜像点位置不同，而镜像距离判断既考虑成功用例的位置又考虑其各个子域中镜像点的位置，所以这种方法可以使产生的用例在输入域更均匀地分布，从而有效地避免边缘效应。

3.2 改进的适应性随机测试算法

对于一个 *n* 维输入域，改进的 ART 算法首先将其划分为同等大小、相同形状的 3^n 个子域，按照均匀分布的原则，在中心子域(即源子域)中随机产生一个测试用例 s_1，并且利用一个简单的映射函数，在剩余的 3^n-1 个子域中各产生一个镜像用例 s_i'，$i=1,2,\cdots,3^n-1$，将 s_1 和 s_i' 按照一定的顺序组成一个测试用例集，分别执行，如果发现失效，则退出算法；如果未发现失效，则将 s_1 放入成功用例集 *S*。然后，在源子域中随机产生 *k* 个候选用例 c_j，$j=1,2,\cdots,k$，计算每个候选用例 c_j 与成功用例集 *S* 的镜像距离 d_j，d_j 为 c_j 与每个成功用例 s_i 之间镜像距离的最小值。再选择 d_j 最大的候选用例 c_q 及其在其他子域的镜像用例作为下一个测试用例集。本文的用例执行顺序是先执行 c_q，再从上到下、由左至右顺序执行其他子域的镜像用例。

需要注意的是，在计算镜像距离时，需先确认有效镜像点。确认有效镜像点时，无需进行距离的计算，根据候选用例和成功用例的位置即可计算出有效镜像点的坐标。例如，在一个 *n* 维输入域中，成功用例为 $s=\{s_1,s_2,\cdots,s_n\}$，候选用例为 $c=\{c_1,c_2,\cdots,c_n\}$，有效镜像点为 $s'=\{s_1',s_2',\cdots,s_n'\}$，令 *n* 维输入域各维边长为 l_i，$i=1,2,\cdots,n$，s_i' 只可能等于 s_i，$s_i+l_i/3$ 或 $s_i-l_i/3$，有效镜像点 $s'=\{s_1',s_2',\cdots,s_n'\}$ 各维的坐标为

$$s_i'=\begin{cases}s_i & \text{if } |s_i-c_i|\leqslant l_i/6\\ s_i+l_i/3 & \text{if } |s_i-c_i|>l_i/6 \text{ and } s_i<c\\ s_i-l_i/3 & \text{if } |s_i-c_i|>l_i/6 \text{ and } s_i>c_i\end{cases}$$

因此，镜像距离计算的公式为

$$dist(s,c)=\sqrt{\sum_{i=1}^{n}\left(s_i'-c_i\right)^2}$$

4 仿真实验与分析

4.1 仿真实验

仿真实验首先在二维条件下进行，令二维坐标轴分别为 *x* 轴和 *y* 轴，输入域为 $\{D|0\leqslant x\leqslant 120,\ 0\leqslant y\leqslant 120\}$，失效率 $\theta=0.01$，失效域以块状为例，形状为正方形，候选用例集大小为 10。需要说明的是，在本文的实验中，MART 算法也将输入域划分为同等大小、相同形状的 9 个子域，以中心子域为源子域。

实验 1 首先将失效域设置为 $\{F|0\leqslant x\leqslant 12,\ 0\leqslant y\leqslant 12\}$，分别运用 DART, MART 和 3^n MART 进行测试，计算各种算法的平均 *F-measure*；然后将失效域沿 $x=y$ 移动，*x* 轴和 *y* 轴移动的步长都为 6，即失效域为 $\{F|6\leqslant x\leqslant 18,\ 6\leqslant y\leqslant 18\}$，进行相同的测试，得到各种算法的平均 *F-measure*。再将失效域继续沿 $x=y$ 移动，步长相同，运用各种算法分别进行测试。这样将失效域移动 9 次。最后，失效域位于输入域的中心位置，即 $\{F|54\leqslant x\leqslant 66,\ 54\leqslant y\leqslant 66\}$，总共进行 10 次仿真实验。

实验 2 将失效域分别置于各个子域的相同位置，即当失效域位于 D_1 内时，失效域为 $\{F|12\leqslant x\leqslant 24,\ 100\leqslant y\leqslant 112\}$；当位于 D_2 内时，失效域为 $\{F|52\leqslant x\leqslant 64,\ 100\leqslant y\leqslant 112\}$。以此类推，得到失效域分布的 9 种情况。在每种情况下，使用 3^n MART 进行测试，重复 5 000 次。最后，得到 9 种情况下 3^n MART 的平均 *F-measure*。

实验 3 在三维条件下进行仿真实验，失效率 $\theta=0.001$，

失效域以块状为例，候选用例集大小为 10。分别计算失效域位置中心区域和边缘区域时 DART, MART 和 3^n MART 的平均 *F-measure*。

实验 4 在四维条件下进行仿真实验，其他条件不变，分别计算失效域在中心区域和边缘区域时各种算法的平均 *F-measure*。

4.2 仿真结果及其分析

实验 1 的仿真结果如表 1 所示。

表 1 实验 1 中 3 种算法的平均 *F-measure*

序号	失效域位置	DART	MART	3^n MART
1	$\{F\|0\leqslant x\leqslant 12,0\leqslant y\leqslant 12\}$	37.030	43.974	64.074
2	$\{F\|6\leqslant x\leqslant 18,6\leqslant y\leqslant 18\}$	72.715	88.079	64.602
3	$\{F\|12\leqslant x\leqslant 24,12\leqslant y\leqslant 24\}$	69.235	77.380	63.477
4	$\{F\|18\leqslant x\leqslant 30,18\leqslant y\leqslant 30\}$	68.433	81.475	63.587
5	$\{F\|24\leqslant x\leqslant 36,24\leqslant y\leqslant 36\}$	68.210	87.203	63.965
6	$\{F\|30\leqslant x\leqslant 42,30\leqslant y\leqslant 42\}$	67.477	42.611	62.988
7	$\{F\|36\leqslant x\leqslant 48,36\leqslant y\leqslant 48\}$	66.733	39.700	60.908
8	$\{F\|42\leqslant x\leqslant 54,42\leqslant y\leqslant 54\}$	66.554	70.122	58.645
9	$\{F\|48\leqslant x\leqslant 60,48\leqslant y\leqslant 60\}$	67.663	78.987	58.186
10	$\{F\|54\leqslant x\leqslant 66,54\leqslant y\leqslant 66\}$	67.808	70.908	58.425

由表 1 可知，DART 除了有边缘效应之外，还存在失效域位于次边缘区域时 *F-measure* 大于失效域位于其他所有区域的 *F-measure* 的现象。MART 的平均 *F-measure* 存在 2 个明显的低谷，即失效域位于 $\{F|0\leqslant x\leqslant 12,\ 0\leqslant y\leqslant 12\}$，$\{F|30\leqslant x\leqslant 42,\ 30\leqslant y\leqslant 42\}$ 和 $\{F|36\leqslant x\leqslant 48,\ 36\leqslant y\leqslant 48\}$ 时，显然，这些区域都位于子域的边缘位置。对于 3^n MART 算法，失效域位于中央区域时的 *F-measure* 明显小于失效域位于边缘区域时的 *F-measure*。但除了失效域为 $\{F|30\leqslant x\leqslant 42,\ 30\leqslant y\leqslant 42\}$ 和 $\{F|36\leqslant x\leqslant 48,\ 36\leqslant y\leqslant 48\}$ (这 2 个失效域是跨子域的)外，表 1 第 5 列的前 5 行近似等于 64，后 3 行近似等于 58。对应图 1 可知，失效域 1~失效域 5 都位于子域 D_7 中，失效域 8~失效域 10 都位于子域 D_5 中，而本文采用的测试用例执行顺序是先执行源子域中的用例，再从上到下、由左至右顺序执行其他子域的用例，这就导致了每次执行用例时 D_7 中的用例总是比 D_5 中的用例晚 6 个执行。因此，当失效域位于 D_7 中时，3^n MART 得到的 *F-measure* 总是比失效域位于 D_5 中得到的 *F-measure* 大 6。实际上，采用 3^n MART 算法在每个子域中产生的用例分布是比较均匀的。为进一步证明这个结论，进行仿真实验 2。

实验 2 的结果如表 2 所示。

表 2 失效域位于各子域时 3^n MART 的平均 *F-measure*

失效域位置	D_5	D_1	D_2
F-measure	57.716	58.867	59.911
失效域位置	D_3	D_4	D_6
F-measure	61.152	62.409	63.469
失效域位置	D_7	D_8	D_9
F-measure	64.517	65.398	66.298

从表 2 第 2 行的第 2 列开始，后一列的 *F-measure* 减前一列的 *F-measure* 都近似等于 1，从而验证了 *F-measure* 的差异与执行顺序有关，而不是因为用例在输入域分布不均匀，并证明了 3^n MART 产生的用例在输入域分布是比较均匀的。

表 3 和表 4 分别为输入域为三维和四维时得到的仿真结果，其中，失效率 $\theta = 0.001$。

表 3 三维输入域时 3 种算法的平均 *F-measure*

失效域位置	DART	MART	3^n MART
输入域中央位置	753.600	912.601	659.000
输入域边缘位置	561.455	669.684	688.082

表 4 四维输入域时 3 种算法的平均 *F-measure*

失效域位置	DART	MART	3^n MART
输入域中央位置	904.505	1530.4	693.064
输入域边缘位置	555.871	932.918	737.918

对比表 3 和表 4 可知，在失效率相同的情况下，随着输入域维度的增大，DART 和 MART 算法的效率急剧恶化，而 3^n MART 算法的效率比较稳定，衰减不大。

5 结束语

仿真实验结果证明，3^n MART 的优点在于：

(1)发现失效的效率总体上优于 DART 和 MART 算法。

(2)受输入域维度变化的影响较小，在高维情况下表现稳定。

(3)计算量较小，大约是 DART 算法计算量的 $1/3^n$。

另外，新算法在以下两方面有待进一步的研究和完善：

(1)本算法在对源子域中选出的用例进行映射时，使用的是直接线性映射函数，它虽然简单、易实现，但当源子域与其他子域中存在相对位置相同的失效域时，可能会降低算法的失效发现效率。

(2)对于可能对本算法产生影响的因素还需要进一步的实验研究，如失效率、失效域形状和候选用例集大小。

参考文献

[1] Hamlet R. Random Testing[M]//Marciniak J J. Encyclopedia of Software Engineering. New York, USA: Wiley, 2002.

[2] Kuo Fei-Ching. On Adaptive Random Testing[D]. Melbourne, Australia: Faculty of Information and Communication Technologies, Swinburne University of Technology, 2006.

[3] Chen T Y, Leung H, Mak I K. Adaptive Random Testing[C]// Proceedings of the 9th Asian Computing Science Conference. [S. l.]: ACM Press, 2004.

[4] Chen T Y, Poon P L, Tang S F, et al. On the Identification of Categories and Choices for Specification-based Test Case Generation[J]. Information and Software Technology, 2004, 46(13): 887-898.

[5] Chen T Y, Tse T H, Yu Y T. Proportional Sampling Strategy: A Compendium and Some Insights[J]. Journal of Systems and Software, 2001, 58(1): 65-81.

编辑 张 帆