Random Border Mirror Transform: A diversity based approach to an effective and efficient mirror adaptive random testing

Michael Omari, Jinfu Chen*, Patrick Kwaku Kudjo, Hilary Ackah-Arthur, Rubing Huang
(School of Computer Science and Communication Engineering, Jiangsu University, Zhenjiang, 212013, China)
*Corresponding author's email: jinfuchen@ujs.edu.cn

Abstract- **Mirror Adaptive random testing (MART) is an overhead reduction strategy for adaptive random testing methods. Theoretically speaking, MART's advantage over ordinary ARTs is determined by the mirroring scheme selected. Incidentally, an inherent problem with MART relates to the difficulty in the choice of scheme for any testing task. This is because a higher scheme (larger mirror domains) does not necessarily guarantee efficient utilization of testing resources due to lack of diversity of mirror generated test cases. The culprit has been identified as the mapping functions used as substitutes to complex ART methods. In this paper, we present a new method for generating diversified mirror test cases by randomly displacing the mirror partitions upon which the mapping functions of MART operates. The result of simulations and experiments conducted shows a remarkable improvement over MART's effectiveness and efficiency across MART schemes especially where program failures are unrelated to one or more input parameters.**

Keywords: ***Adaptive random testing, software testing, mirror adaptive random testing, test case diversity*.**

I. INTRODUCTION

Software testing is a vital and integral part of software engineering process that ensures quality of software product. It does this by actively detecting bugs before serious software failures actually take place in operation[1]. Its prime objective is to enhance reliability of systems managed, controlled or operated by a software artifact. Essentially, testing involves selecting samples from a set of all possible combination of inputs to a software(referred to as input domain), executing the selected samples one by one and determining whether the outputs from each sample match the software specification[2]. In practice, it is almost always impossible to test every element of the input domain (as in exhaustive testing). Therefore there is the need to device strategies for selecting inputs in a way that minimizes resources needed to reveal program fault(s). Test case selection is therefore a critical task in software testing. There have been a wide range of testing strategies which have been published in literature and implemented in software testing tools, each possessing some strengths and weaknesses[3]. Random testing (RT) has been the basic underlying strategy for most software testing strategies and have been successfully implemented in many real-world testing tools [4-6]. It selects test input randomly from the input domain. The advantages of random testing are many including; simplicity of test generation, low overhead cost and ease of automation associated with the technique[3] . In recent past, a new method for random testing has been proposed by Chen et al [7]. The strategy referred to as adaptive random testing (ART) aims at improving the effectiveness of RT by distributing randomly generated test cases more evenly throughout the input domain. The intuition behind this strategy is based on the distribution of failure causing inputs. The works of [8, 9] have established that many failure causing inputs of software tends to crystallize within the input domain. ART takes advantage of this information to improve the effectiveness of RT because evenly spreading test cases within the input domain increases the probability of finding program failures. Using twelve (12) real-world programs, the works of [7, 10], have revealed that ART can outperformed RT by 30% in some cases to as much as 80% in terms of the number of test cases used to detect the first program fault (F-measure). Since the introduction of ART, there has been numerous algorithms which implement the even distribution idea using a variety of notions including partitioning based restriction based, distance based etc. Achieving effectiveness through even distribution of test cases is just one side of the coin and balancing the effectiveness with efficiency is also another. There are issues which bother on the use of ART as a testing strategy. These issues ranges from practical concerns that would affect the use of present ART techniques, to more abstract questions about the theoretical justification of the approach [11]. Right from the onset, it was suggested that the future research directions on ART strategies should be targeted at lowering its overheads and broadening the method's applicability, rather than vainly trying to seek further improvements in effectiveness[11]. Practicality remains a major concern due to the computational burden of ART. This concern has been expressed by Acuri [12] and has prompted several researches into how best to achieve the even distribution objective with lower overhead cost. Consequently, most research focus on ART has shifted to bridging the inefficiency gap between ART and RT. Indeed several research on ART in the past [13-16] have had this objective in mind. Mirror adaptive random testing (MART) is one of such methods.

***Contribution of this paper*:**
We propose a novel mirror test case generation technique called random border mirror transform (RBMT) that enhances failure detection capability of MART and guarantee consistent performance (effectiveness and efficiency) irrespective of the choice of MART scheme; a major drawback to MART's success.

***The rest of the paper is summarized as follows*:**
Chapter II introduces the background to the problems with MART and highlights where there is the need for improvement. In chapter III we present an in-depth formalization of a novel solution to MARTs problems. Chapter IV presents the

978-1-7281-3927-2/19/$31.00 ©2019 IEEE
DOI 10.1109/QRS.2019.00020

research design, parameters and system environment followed by simulation and experimental studies comparing RBMT, MART, FSCS and RT in Chapter V. We conclude the paper with findings from results obtained in chapter VI.

II. BACKGROUND

A. *Mirror Adaptive Random Testing (MART) as a solution to the overhead cost of ART.*

One of the well-studied ART overhead reduction strategies is mirror adaptive random testing (MART). MART is based on a combination of divide and conquer and heuristic strategies [17]. It seeks to reduce ART overheads by concentrating test case generation effort of ART in a single partition (referred to as source domain) whiles using simple mapping functions to generate images from the source domain (SD) into other set of partitions (referred to as mirror domain(s)). MART is therefore not a standalone ART algorithm but requires a joint application with other ART method. To give a deeper insight into the overhead challenge of ART, let's take a classical ART algorithm; fixed sized candidates set (FSCS). To generate the nth test case $(1 < n \leq N)$, (where N is the size of test suit) firstly, k (a constant) number of candidate test cases must be randomly selected, then for each candidate, compute the Euclidean distance to all previously (n-1) executed test cases. The complexity of FSCS is thus $O(\sum_{n=2}^{N}(n-1).k) = O(k.n^2)$. The computational cost of FSCS progressively worsens with increasing number of failed attempt at finding program fault. The reduction in the computations using MART is directly proportional to number of mirror partitions implemented. For instance, when FSCS was used to in combination with MART [17], the order of computation was reduced from $O(n^2)$ to $O(n^2/m^2)$; m being the number of subdomains.

B. *Characteristics of MART*

An empirical studies [17, 18] shows that MART comprises of three major components including mirror partitioning scheme, mirror mapping functions and a mirror selection order. These characteristics combine to form MART mirroring scheme.

(1) Mirror partitioning scheme

One essential and perhaps most critical component of MART is the selection of partitioning scheme. In a d dimensional input domain, we define a MART partitioning scheme as $\omega = (\omega_1, \omega_2, \omega_3 \ldots, \omega_d)$ where ω_i is the number of partitions on dimension i; $i = 1,2,3, \ldots d$. In defining a partitioning scheme, there must be at least one dimension i where $\omega_i > 1$. Fig.1 provides some possible MART partitioning schemes in two dimension input domain. Schemes $\omega = (1,2)$, $\omega = (2,2)$ and $\omega = (2,1)$ are presented in (a), (b) and (c) respectively. By way of notation, we represent a MART scheme as MART $\omega_1 \times \omega_2 \times \omega_3 \ldots \times \omega_d$. Chen et al [17] recommended that ω_i should be as small as possible with $1 \leq \omega_i \leq 2$ giving a maximum of $\gamma = 2^d$ number of subdomains and $m = \gamma - 1$ mirror domains.

(2) MART mapping functions

A mapping function defines how a mirror test case is generated. Many mapping functions have been proposed but the most frequently used functions under MART include reflect and translate mapping functions [15]. In fig. 2, we provide an illustration of these two mapping functions.

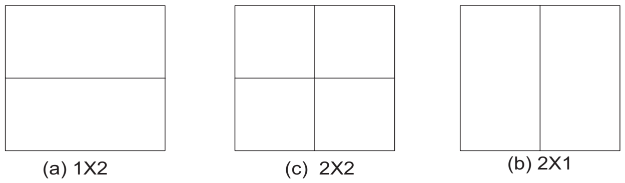


Fig. 1 simple 2D partitioning schemes

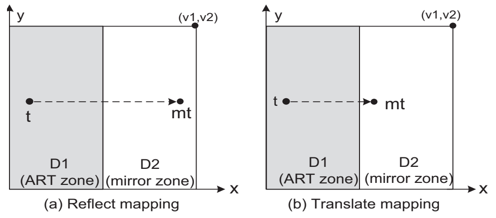


Fig. 2 Two simple mirror mapping functions

Given a test case $t = (x, y)$ (generated using ART) within the source domain, a reflect function produces a mirror test case $mt = (v_1 - x, y)$. The Translate function will map a test case $t = (x, y)$ in D1 into D2 as $mt = \left(x + \frac{v_1}{2}, y\right)$. Previous works [17, 18] on MART based on simulations revealed that the performance of MART under these two mapping functions is not significantly different.

(3) Mirror selection order

Mirror selection order defines the sequence in which the pool of the mirror generated test cases should be executed. Three selection approaches have been suggested random order, sequential order and adaptive random sequence order.

C. *Test case diversity-The problems with MART mapping functions*

Diversity constitutes the underpinning principle for most test case generation strategies [19]. Apart from fault-based techniques which assume certain types of faults as targets, diversity seems to be implicitly considered, if not explicitly, in the design of many test case selection strategies [19]. The concept of diversity has been implemented in diverse ways in the past as an intrinsic property of test case generation using a variety of notions. For instance, the different types of control coverage and dataflow coverage criteria yield test sets with different notions of diversity. Distribution and dispersion metrics have been defined to measure how effective ART methods distribute test cases. But as has been pointed out by Chen et al [20] distance alone cannot be a good criterion for measuring diversity. Consider a 2×2 partitioning scheme of MART demonstrated in fig. 3(a) and its test case distribution. The original test case selected in the source domain (SD) is $t(x_1, y_1)$. Mirror test cases mt_1, mt_2 and mt_3 are generated using MART. Fig. 3(b) also shows four set of test es t_1, t_2 t_3 and t_4. It can be observed that each test case have similar values along the line of mirror dimensions. This is an inherent problem with the mapping function of MART. Even though test cases in 3(a) seems to be relatively far apart based on dispersion metric of even distribution [21] test case distribution is more diversified in 3(b) than 3(a). Therefore test case distribution in (b) is more likely to detect failure than test case distribution (a).

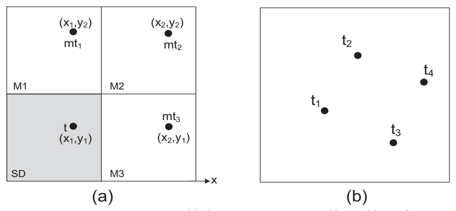


Fig. 3 Two possible test case distribution

D. Parameter unrelated failure regions and the reliability of MART -a diversity related problem of mirror functions

A failure unrelated parameter exists where a software failure is not related to values of a particular dimension of input domain. Fig.4 is a schematic illustration of failure unrelated parameter scenario. MART 2×1 expressed in fig 4(a) uses translate mapping function f to map a test case (t); $f(t) \rightarrow (x + 1/2\, x, y)$. In Fig.4 (b) MART 1X2 also using translate function f maps a test case (t) with$f(t) \rightarrow (x, y + 1/2\, y)$. It can be observed from both diagrams that if a test case does not find fault within the source domain, another test case generated that is solely based on variation of the horizontal parameter (similar to how a mapping function operates) does not influence the chances of selecting a point in the failure region. The failure region is therefore said to be unrelated to the horizontal dimension. Since MART 2×1 mirror test cases are based solely on variation in the horizontal parameter (which does not relate to the failure region), it is less effective compared to MART 1×2 (in b). Therefore given the same failure region MART 2×1 (a) and MART 1×2 (b) have significantly different level of effectiveness. In reality, the tester does not have a clue about existence of failure unrelated parameter and which dimension it relates to inform the choice of scheme to select. According to Kuo et.al [22] partitioning scheme should be designed to minimize the effect of dimension unrelated parameters by avoiding the partitioning of the "uncritical parameters" (such as appears in fig. 4a) which is difficult to determine without a critical analysis of source code.

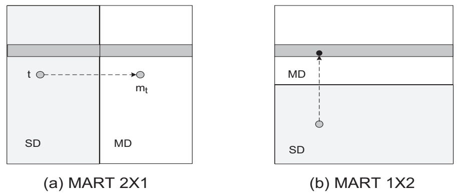


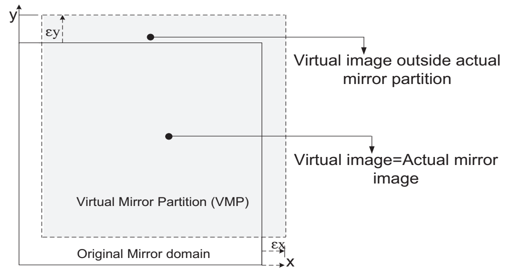
Fig. 4 Failure unrelated parameter crises of MART

III. PROPOSED METHOD: RANDOM BORDER MIRROR TRANSFORM (RBMT)

This chapter presents the proposed approach to address the problem associated with MART. In addressing the problem, we switch our focus from the mapping functions by randomly "shifting the data" on which the mapping functions operates. This provides a different perspective to finding a solution. In the preceding subsections, we explain in detail RBMT and how it resolves the outstanding problems of MART.

A. Creation of virtual mirror partition

Without loss of generality, consider a two (2) dimensional input domain presented in fig. 5 (showed in the bold lines). Let εx and εy are random displacements on x and y dimensions respectively. A new partition created by the displacement is called a virtual mirror partition. The virtual mirror partition (VMP) created is the shaded region and its position is defined by εx and εy.



y
εy
Virtual image outside actual mirror partition
Virtual image=Actual mirror image
Virtual Mirror Partition (VMP)
Original Mirror domain
εx
x

Fig.5 Creating virtual mirror partition

B. Random Border Displacement vector

Given a mirror partition M_j; $1 \leq j < \gamma$ with lower and upper limit on dimension i $(D_i), 1 \leq i \leq d$ given as u_i and v_i respectively, the range on (D_i) is given by $r_i = [u_i, v_i]$. Let α_i be a random value between (α_1, α_2) where $0 \leq \alpha_1 < \alpha_2 \leq 1$. A random border displacement vector on D_i is defined by $\varepsilon_i = \alpha_i \times r_i$. A virtual mirror partition for M_j defined on D_i (vmp_{ji}) is bounded by u_vmp_{ji} and v_vmp_{ji} where $u\_vmp_{ji} = u_i + \varepsilon_i$ and $v\_vmp_{ji} = v_i + \varepsilon_i$ i.e. $\left|v_{vmp_{ji}} - u_{vmp_{ji}}\right| = |r_i|$. By inference, when $\alpha_i = 0\ \forall i$, $\varepsilon_i = 0$ which indicates that vmp_j is equivalent to MART mirror partition M_j. Conversely when $\alpha_i = 1\ \forall i$, $\varepsilon_i = r_i$ is an inversion of the mirror partition M_j implying that the displacement vector inverts all the test cases generated by the translate function of MART. Thus, a necessary condition for unique mirror image in vmp_j for every test case generated in the source domain is $0 < \alpha_i < 1$.

We define a schematic diversity transform vector (SDTV) for a selected partitioning scheme ω in RBMT as $\partial_{\omega}^{\sigma} = \bigcup_{j=1}^{m} vmp_j$ where $\sigma = 1,2,3, \dots 2^d - 1$ and $m = 2^d - 1$ (the number of mirror domains). Consider a two dimensional input domain with minimum values and maximum values on each dimension given as (0, 0) and (1, 1) respectively. The selected scheme $\omega = (2,2), \sigma = 3, m = 3$; and $\varepsilon_x = (\varepsilon_{x1}, \varepsilon_{x2}$ and $\varepsilon_{x3})$ and $\varepsilon_y = (\varepsilon_{y1}, \varepsilon_{y2}$ and $\varepsilon_{y3})$ are displacements for virtual partitions vmp1, vmp2 and vmp3 in their respective components. These details are captured in Fig.6. A SDTV is given by;

$\partial_{(2,2)}^{3} = \left\{ \left(x + \varepsilon_{x1}, \frac{y}{2} + \varepsilon_{y1}\right), \left(\frac{x}{2} + \varepsilon_{x1}, 1 + \varepsilon_{y1}\right); \left(\frac{x}{2} + \varepsilon_{x2}, \frac{y}{2} + \varepsilon_{y2}\right), \left(1 + \varepsilon_{x2}, 1 + \varepsilon_{y2}\right);\ \left(\frac{x}{2} + \varepsilon_{x3}, y + \varepsilon_{y3}\right), \left(1 + \varepsilon_{x3}, \frac{y}{2} + \varepsilon_{y3}\right) \right\}$.

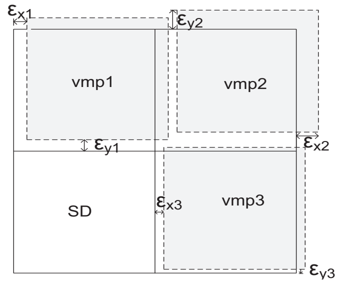


Fig. 6 virtual mirror partitions showing different displacement vector on each dimension

In general, given two virtual partitions $vmp_a$and $vmp_b \in \partial_\omega^\sigma$, if $\varepsilon_{ak} \neq \varepsilon_{bk}\ \forall k = 1,2,3,\ldots,d$ where k represents dimension for the displacement vectors then the value of random boarder displacements on each dimension is different for each member of ∂_ω^σ . Therefore the range on each dimension for each m is different.

C. *Virtual mirror transform and Transform discriminant* (T^d)

The virtual image of RBMT is the image of the original test case generated in the source domain within the VMP. A displacement vector is an all-dimension displacement parameter that is added to the translated test case using ordinary translate mapping function. Mathematically, $Virtual\ Image = Translate_{MART} + Displacement\ vector$.
The transform discriminant T^d is a parameter obtained from the position of the virtual image on dimension d.

D. *Mirror transforms*

There are therefore two possible ways for generating VMP under the following conditions;
(1) When Displacement vector<0 for all dimensions
(2) When Displacement vector>0 for all dimensions

(1) Edge Transform (RBMT-E): when $\varepsilon < 0$

Given $\varepsilon_i = \alpha_i \times r_i$ and a test case t_i generated in the source domain, the transform discriminant $T_i^d = u_i + \tau_i - \varepsilon_i$ where, τ_i is the displacement of t_i from u_i. If $T_i^d < u_i$ then the mirror test case is outside of the input range of D_i within the virtual partition. Hence, its corresponding partition pair within the actual mirror partition contains the mirror test case $mt_i = v_i - u_i - \tau_i$ otherwise $mt_i = T_i^d$ is in the region where the VMP coincides with the actual mirror partition.
Thus;

$$mt_i = \begin{cases} v_i - u_i - \tau_i & if\ T_i^d < u_i \\ u_i + \tau_i - \varepsilon_i & if\ T_i^d \geq u_i \end{cases}$$

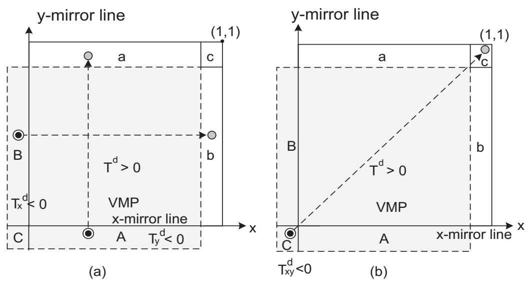


Fig. 7 Edge Transform showing (a) Partial and (b) complete transforms

The transform test case is reverted to the border when $T_i^d < u_i$. The edge transform (RBMT-E) will therefore maintain the border selection preference of most ART's. Assume a unit magnitude in a 2D dimension input domain (i.e. the range on each dimension is [0, 1]) presented in fig 7. The letters a and b represents the actual mirror partitions whiles the corresponding caps A and B represents the virtual partition pair. The selection of either pair depends on the value of T_i^d. Transformation is complete in RBMT-E if $\forall i = 1,2,3 \ldots d,\ T_i^d < u_i$. The image is transformed in an all-coordinate fashion as demonstrated in 7(b) otherwise the transform is partial and the image is transformed only along the dimensions where $T_i^d < u_i$ depicted in 7(a).

(2) Center Transform (RBMT-C): when $\varepsilon > 0$

By reverse arguments for $\varepsilon > 0$ for all dimensions, we obtain the mirror test cases for RBMT-C as;

$$mt_i = \begin{cases} u_i - \tau_i - v_i & if\ T^d > v_i \\ u_i + \tau_i + \varepsilon_i & if\ T^d \leq v_i \end{cases}$$

(3) Edge transform with translate (RBMT-ET)

Given a lower and upper limit of the partition formed by VMP and the actual mirror partition where $T_i^d \geq u_i$ as u_i and $v_i - \varepsilon_i$ respectively, with midpoint $M_i^d = mid\{u_i, (v_i - \varepsilon_i)\}$ on each dimension for this partition can be realigned to form a contiguous input range to minimize the fragmentations within the input space. There are two ways to determine the mirror test case mt_i depending on the value of T_i^d;

$$mt_i = \begin{cases} \dfrac{3u_i - v_i - \varepsilon_i}{2} + \tau_i & if\ T_i^d > M_i^d \\ \dfrac{u_i + v_i - 3\varepsilon_i}{2} + \tau_i & if\ T_i^d < M_i^d \end{cases}$$

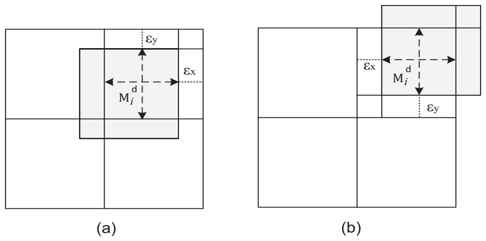


Fig.8 Edge (a) and Center (b) Transform with virtual translate in RBMT 2X2

The partition for RBMT-ET is represented in fig 8 (a).

(4) Center transform with translate (RBMT-CT)

Fig. 8(b) represents the partition for RBMT-CT. By inference from RBMT-ET, a mirror test case for RBMT-CT is given by;

$$mt_i = \begin{cases} \dfrac{3(u_i + \varepsilon_i) - v_i}{2} + \tau_i & if\ T^d > M_i^d \\ \dfrac{u_i - v_i - 3\varepsilon_i}{2} + \tau_i & if\ T_i^d < M_i^d \end{cases}$$

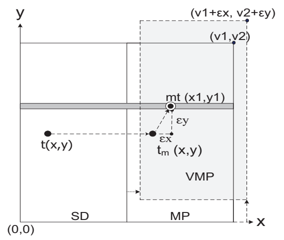


Fig. 9 Transform of mirror test case to detect failure region insensitive to input parameter.

E. *Sidestepping the failure unrelated parameter problem*

Let's consider a basic form of RBMT, RBMT-C and how it avoids the failure unrelated parameter problem. Supposed in a

2D (using a 2×1 partitioning scheme) depicted in fig.9, a mirror test case t_m does not reveal failure due to the relationship of the failure region and the horizontal dimension. With the displacement vector (diversity vector) ε_i added to the translated test case t_m , the sterile test case(t_m) is given a 'kick' to fall in the failure region.

Algorithm for RBMT-ART

```
1. Partition input domain according to selected MART scheme
2. Select a source domain and remain partitions as store in mp
3.  for each mirror partition (mp)
4.        for each dimension ($d$)
5.             Generate a random border vector ($\varepsilon_d$)
6.        end for//d
7.        Create a virtual mirror partition for mp using $\varepsilon_d$
8.  end for// creation of virtual mirror partitions
9.  Use ART to generate a test case //MART begins from here
10.  Execute the test case
11.  If failure is revealed goto step 21
12.  for each virtual mirror partition
13.        Generate mirror image ($m_t$) and add to testBuffer
14.  end for
15.  while testBuffer pointer <|testBuffer|
16.        Execute test case corresponding to testBuffer pointer
17.        If failure is found goto step 21
28.        Increment testBuffer pointer
19.  end while
20.  goto step 9
21.  stop
```

F. *Complexity analysis of RBMT-ART*

The complexity of RBMP-ART is similar to that of MART (from step 9). The creation of VMPs (algorithm steps 3-8) is O($d \times m$)and performed only once. The computation of mirror image (m_t) is also of constant order. Therefore the complexity of RBMT is dependent on the ART algorithm implemented in the source domain and m (the number of mirror domains). The complexity of RBMT-ART using FSCS (in this report) is thus O(n^2/m^2) for the nth test case.

IV. SIMULATIONS AND EXPERIMENTAL STUDIES

A. *Research questions*

We are guided by the following key research questions in the evaluation of the proposed method.

RQ1: How effective is RBMT in failure detection compared to MART and FSCS?
RQ2: How efficient is RBMT-ART in relation to MART?

B. *Evaluation metrics*

Historically, the effectiveness of ART has been measured by using F-measure (F) [7, 23]. The term F-ratio has also been of a particular interest as it measures ART effectiveness improvement upon RT. i.e. F-ratio=$F_{ART}/F_{RT.}$ We use these same metrics in assessing the effectiveness of RBMT due to their intuitive appeal. Given a faulty program with failure rate θ, F_{RT} =1/θ. Without any analytical model for estimating F_{ART}, it has been estimated by iterating each experiment (that return a single F_{ART} datum) until a sufficiently large (statistically reliable) data (N) is obtained and taking the average. As to how N is calculated, readers can refer to Chen et al [7].

C. *Parameter setting*

The parameters $\alpha_1$and α_2 ($\alpha_1 < \alpha < \alpha_2$) determines the proportion of the input range on each dimension that is used to create virtual partitions. Intuitively we do not expect different values of these parameters to significantly influence the effectiveness of RBMT so long as their setting induces diversity in the mirror test cases. Therefore α is arbitrarily placed in the range [0.02, 0.05].

D. *System environment*

The simulation and experiments were carried out on a system with the following characteristics; Processor: Del Inspiron 3847, Intel® Core™ i3-4170 CPU @3.7GHz (4CPUs) ~3.7GHz, Memory: 8GB, OS: Microsoft windows10 Pro 64-bit, Development environment: Dev C++ 4.9 -std= c++11.

V. RESULTS OF SIMULATIONS AND EXPERIMENTS

A. *Simulations*:

Answer to RQ1: In this simulation, we evaluate the effectiveness of RBMT using multiple dimensions with hyper cubic failure region for two different failure rates (θ) =0.001 and θ=0.005. This is because it has been established [15, 24] This is because it has been established [15, 24] that whiles RT's effectiveness is only dependent θ, the effectiveness of ART is influenced by a myriad of factors including dimension and failure rates of the program in question. The selected dimensions were D = 3, 7 and 10. Three (3) schemes of MART were deemed sufficient for our purpose for lack of time. For ease of illustration, d dimensions MART schemes comprising full length dimensions(x=1) and bisected length dimensions (y=2), is presented as $x^{d-1}y^k$ in place of MART$x_1 \times x_2 \times x_3 \ldots \times x_{d-k} \times y_1 \times y_2 \times y_3 \ldots y_d$, where k represents the number of bisected dimension components of MART scheme. For example, we use 1^4x2^3 to represent MART $1 \times 1 \times 1 \times 2 \times 2 \times 2$.

Table 1 gives a summary of the effectiveness (in terms of F-Ratios) of RBMT in its entire variant forms (RBMT-C, E, CT and ET) in comparison with FSCS and MART for the selected schemes. From table 1 we make the following observations:

- The effectiveness of RBMT is better in the translated versions in higher dimensions than the basic forms in both values of θ. RBMT's effectiveness is generally similar to MART and FSCS in smaller dimensions.
- In all selected dimensions under study, the improvement rate of FSCS, RBMT and MART over RT is better in smaller failure rate than higher failure rate.
- In both higher and smaller dimensions, RBMT in its basic forms have almost similar effectiveness compared to MART but far better than FSCS.
- As the number of bisected dimensions increases, the effectiveness in both RBMT and MART improves irrespective of the size of the failure rate (θ).
- The effectiveness gap between FSCS and the two mirroring approaches RBMT and MART increases with increasing dimensions of input domain irrespective of the scheme implemented.

Table 1 Simulation results showing F-ratios (F_{ART}/F_{RT}) for RBMT compared to MART and FSCS

METHOD	Failure rate (θ)=$\mathbf{0.001}$ MART schemes (ω)				Failure rate ($\boldsymbol{\theta}$) $= \mathbf{0.005}$ MART schemes (ω)			
	3D	1x1x2	1x2x2	2x2x2		1x1x2	1x2x2	2x2x2
FSCS	0.7692				0.7980			
MART		0.7535	0.7636	0.7843		0.8104	0.8484	0.8453
RBMT-C		0.7467	0.7480	0.7649		0.8041	0.8351	0.8295
RBMT-E		0.7612	0.7762	0.7803		0.8231	0.8364	0.8476
RBMT-CT		0.7410	0.7640	0.7731		0.8016	0.8061	0.8143
RBMT-ET		0.7643	0.7631	0.7662		0.8051	0.8107	0.8286
	7D	1^5x2^2	1^2 x2^5	2^7		1^5x2^2	1^2 x2^5	2^7
FSCS	1.6152				1.8591			
MART		1.4851	1.2972	1.1115		1.5607	1.0930	0.6724
RBMT-C		1.5120	1.2931	1.1615		1.6035	1.1537	0.7494
RBMT-E		1.5512	1.3566	1.1652		1.6303	1.1514	0.7299
RBMT-CT		0.9775	0.9393	0.8998		1.2099	0.7245	0.7198
RBMT-ET		1.0072	0.9458	0.9038		0.9234	0.8934	0.7342
	10D	1^7x2^3	1^5 x2^5	1^3x2^7		1^7x2^3	1^5 x2^5	1^3x2^7
FSCS	2.7453				3.6177			
MART		2.1017	1.7108	1.3017		3.9762	4.1365	4.0294
RBMT-C		2.1454	1.7234	1.4738		3.6111	3.9077	3.7732
RBMT-E		2.1834	1.7535.	1.4823		3.8292	3.8979	3.7724
RBMT-CT		0.9813	0.9523	0.9552		1.3284	1.3270	1.7742
RBMT-ET		1.0943	1.0645	0.9345		1.2913	1.3401	1.7776

Table II Experiment programs and their essential features

	D	**Input Domain**		**Seeded fault type**				**Total**	**Failure rate**
Program		FROM	TO	AOR	ROR	CR	SVR		
bessj	2	(-2.0,-1000)	(300.0,15000.0)	2	1		1	4	0.001298
cel	4	(0.001,0.001,0.001,0.001)	(1.0,300,10000,1000)	1	1		1	3	0.000332
el2	4	(0.0,0.0,0.0,0.0)	(250,250,250,250)	1	3	2	3	9	0.000332
gammq	2	(0.0, 0.0)	(1700.0, 40.0)		3		1	4	0.000574
golden	3	(−100.0,-100.0,100.0)	(60,60,60)		3	1	1	5	0.000830
plgndr	3	(10.0,0.0, 0.0)	(500.0, 11.0, 1.0)	1	2		2	5	0.000368
sncndn	2	(-5000.0,-5000.0)	(5000.0,5000.0)			4	1	5	0.001623

B. Empirical studies

In our empirical study we selected real-life programs with input domain not exceeding four dimensions because we intend to evaluate all possible schemes of MART and RMBT in order to assess how effectiveness can be influenced by the choice of scheme. This requires a comparison of 2^d -1 number of possible schemes which will be very tedious and time consuming to evaluate if input dimension is too high. Table II provides some essential features about the programs for our experiment. The seeded error types includes; Arithmetic Operator Replacement (AOR), Relational Operator Replacement (ROR), Constant Replacement (CR) and Scalar variable Replacement (SVR). These programs have been used extensively in the past [7, 10, 14, 15] to evaluate the effective ness/efficiency of most ART algorithms. Huang et al [15] have analyzed these programs and reported that three(3) contains failure unrelated parameters (FUP). This information has been captured in table III. In the analysis, we focus our attention on two key objectives (theoretical propositions) of MART; MART as an ART meant to enhance the effectiveness of RT and MART as an efficient alternative to ordinary ART meant to reduce computational time to find program fault. Essentially, these two set of objectives are logically complementary. We divided our analysis into two parts; an assessment based on programs with FUP and programs without FUP. For ease of presentation, in fig. 10 and 11 (b and c) we use M1, M2, M3, M4, M5, M6, M7, M8, M9, M10, M11, M12, M13, M14,M15 to represent MART permutations ($1^3 \times 2$, $1^2 \times 2 \times 1$, $1 \times 2 \times 1^2$, 2×1^3, $1^2 \times 2^2$,$1 \times 2 \times 1 \times 2$,$1 \times 2^2 \times 1$, $2 \times 1^2 \times 2$, $2 \times 1^2 \times 2$, $2 \times 1 \times 2 \times 1$, $2^2 \times 1^2$, 1×2^3, $2^2 \times 1 \times 2$, $1 \times 1 \times 2^2$, 2^4) respectively.

Table III Programs with failure unrelated parameters

Program	Number of FUP's	Dimensions not relating to failure region
plgndr	1	3
cel	3	1,3 and 4
el2	2	3 and 4

(1) **Answer to RQ1:** Effectiveness of RBMT compared to MART and FSCS.

- *Part I: Programs without FUP*

With the exception of golden where RBMT recorded an improvement in effectiveness over MART in majority of schemes, RBMT effectiveness is generally similar to MART and FSCS when programs are without FUP (Fig.10 d-g).

- *Part II: Programs with FUP*

The results of RBMT and MART in programs with FUP are presented in fig.10 (a-c). From the figure, the effectiveness of RBMT is very consistent across different schemes. Notice that as the number of FUP's increases, the effectiveness gap between the RBMT variants and MART also increases as well. We observe that the worst case scenario for RBMT is as good as the best case of MART and comparable to FSCS in the presence of a FUP. On the other hand, there are a greater percentage of MART schemes which are far less effective when compared to RT though ART is meant to improve the effectiveness of RT.

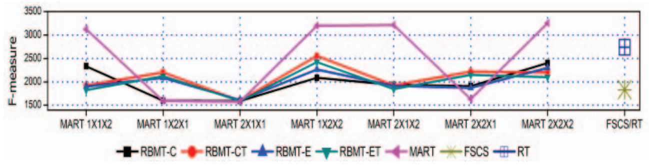
(2) **Answer to RQ2:** Efficiency comparison of RBMT to MART and FSCS.

- *Part I: Programs without FUP*

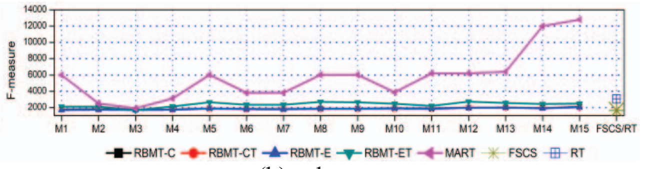
In programs without FUP (Fig.11 d-g), the average time taken for RBMT variants to find fault is similar to MART. Both methods did used shorter time to find program fault as compared to FSCS.

- *Part II: Programs with FUP*

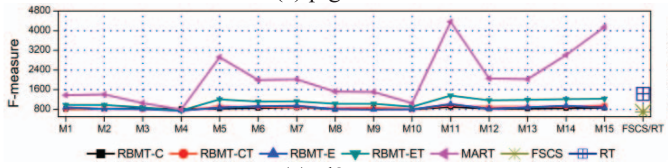
From fig. 11(a-c) we notice that in the case of MART, the Fm-time recorded is clearly dependent on two variables; the f-measure values and the dynamics of the mirroring scheme. RBMT used much shorter time to detect program fault in every scheme compared to FSCS, this is in sharp contrast to MART In fact FSCS recorded lower or similar Fm-time when compared to some of the schemes of MART particularly in the



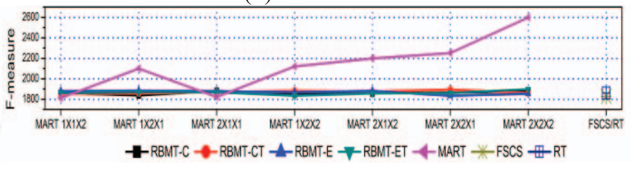
(a) plgndr



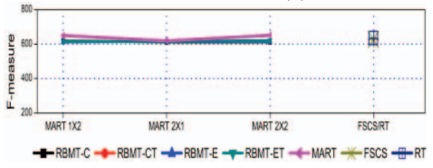
(b) cel



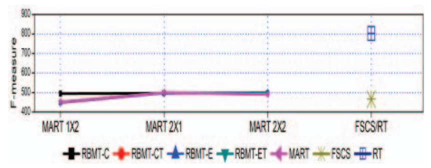
(c) el2



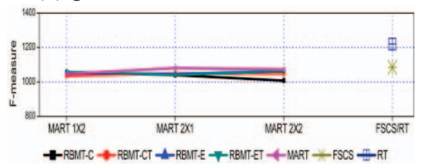
(d) golden



(e) bessj

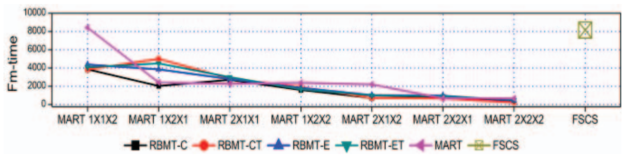


(f) sncndn

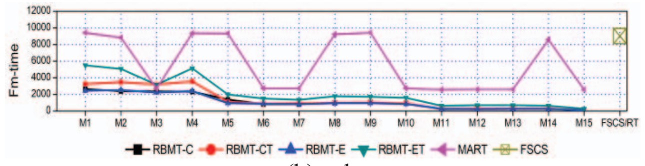


(g) gammq

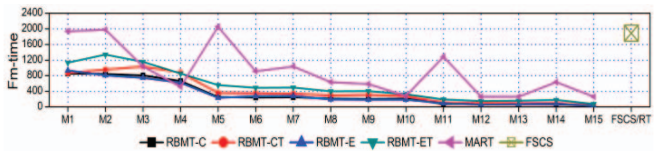
Fig. 10 Average number of test cases used to detect the first program failure.



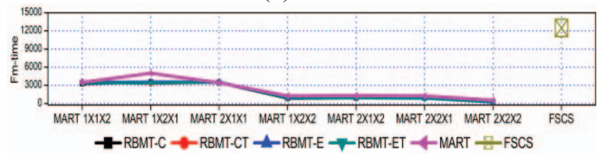
(a) plgndr



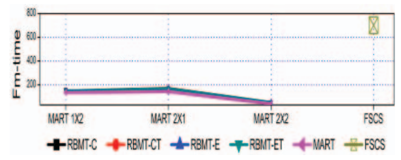
(b) cel



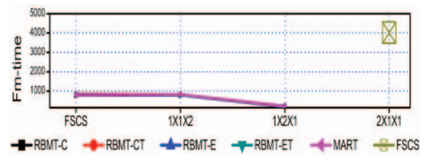
(c) el2



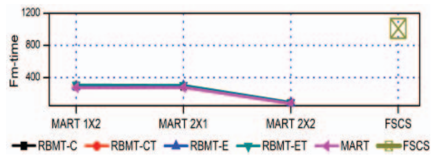
(d) golden



(e) bessj



(f) gammq



(g) sncndn

Fig. 11 Average Fm-time recorded for all schemes for all seven (7) selected programs

cel program. Compared to RBMT, the Fm-time recorded was primarily influenced by the selected mirroring scheme. Therefore RBMTs advantage over MART in the execution time was aided by the comparative advantage in effectiveness due to improved diversity of mirror generated test cases.

VI. CONCLUSIONS

The choice of MART as an alternative approach to ordinary ART is with the promise that program failure(s) can be detected within a shorter time. We have empirically verified that under certain circumstances, ordinary ART can outperform MART (in some selected schemes) in terms of the time taken to detect program fault. This is attributable to the lack of diversity in MART's mirror generated test cases; an inherent weakness in the mapping functions of MART. To solve this problem, we introduced a schematic diversity transform vector derived from random border displacements of the original mirror partitions. This leads to a more diversified mirror test case. In the simulated studies conducted, we observed that, RBMT is more effective than MART in very high dimensions in some variants of RBMT. We also conducted an experiment using some selected real-world programs. Our proposed method recorded a remarkable improvement over MART's effectiveness in programs with failure regions not relating to one or more dimensions of input parameters. The experiment also shows that using the maximum recommended partitioning scheme for RBMT preserves effectiveness across all possible schemes; a phenomenon which is unlikely to occur using the current implementation of MART. With that, the tester can select the highest possible scheme for the maximum possible efficiency. RBMT retains the flexibility of MART whiles maintaining a balance between efficiency and effectiveness. As a future direction to this paper, a full scale investigation will be conducted to assess RBMT under different program failure patterns. Also, an investigation will be conducted to determine an optimal setting of the displacement vectors of RBMT if it exists. Finally, the recommended maximum mirroring scheme of MART could be adjusted upward to see its impact RBMT'S effectiveness.

ACKNOWLEDGEMENT
This work is partly supported by National Natural Science Foundation of China (NSFC grant numbers: U1836116, 61502205 and 61872167), the project of Jiangsu provincial Six Talent Peaks (Grant number XYDXXJS-016) and the Postdoctoral Science Foundation of China (Grant number 2019T120399.

REFERENCES

[1] F.-C. Kuo, T. Y. Chen, H. Liu, and W. Chan, "Enhancing adaptive random testing in high dimensional input domains," in *Proceedings of the 2007 ACM Symposium on Applied Computing*, pp. 1467-1472.

[2] T. Y. Chen, F.-C. Kuo, R. G. Merkel, and T. Tse, "Adaptive random testing: The art of test case diversity," *Journal of Systems and Software,* vol. 83, pp. 60-66, 2010.

[3] A. F. Tappenden and J. Miller, "A novel evolutionary approach for adaptive random testing," *IEEE Transactions on Reliability,* vol. 58, pp. 619-633, 2009.

[4] D. L. Bird and C. U. Munoz, "Automatic generation of random self-checking test cases," *IBM Systems Journal,* vol. 22, pp. 229-245, 1983.

[5] R. H. Cobb and H. D. Mills, "Engineering software under statistical quality control," *IEEE Software,* vol. 7, pp. 45-54, 1990.

[6] T. Y. Chen, D. H. Huang, and Z. Q. Zhou, "Adaptive random testing through iterative partitioning," in *International Conference on Reliable Software Technologies*, 2006, pp. 155-166.

[7] T. Y. Chen, H. Leung, and I. Mak, "Adaptive random testing," in *Annual Asian Computing Science Conference*, 2004, pp. 320-329.

[8] P. E. Ammann and J. C. Knight, "Data diversity: An approach to software fault tolerance," *Ieee transactions on computers,* pp. 418-425, 1988.

[9] G. B. Finelli, "NASA software failure characterization experiments," *Reliability Engineering & System Safety,* vol. 32, pp. 155-169, 1991.

[10] K. P. Chan, T. Y. Chen, and D. Towey, "Restricted random testing: Adaptive random testing by exclusion," *International Journal of Software Engineering and Knowledge Engineering,* vol. 16, pp. 553-584, 2006.

[11] R. G. Merkel, "Analysis and enhancements of adaptive random testing," Swinburne University of Technology, 2005.

[12] A. Arcuri and L. Briand, "Adaptive random testing: An illusion of effectiveness?," in *Proceedings of the 2011 International Symposium on Software Testing and Analysis*, 2011, pp. 265-275.

[13] A. Shahbazi, A. F. Tappenden, and J. Miller, "Centroidal voronoi tessellations-a new approach to random testing," *IEEE Transactions on Software Engineering,* vol. 39, pp. 163-183, 2013.

[14] C. Mao, T. Y. Chen, and F.-C. Kuo, "Out of sight, out of mind: a distance-aware forgetting strategy for adaptive random testing," *Science China Information Sciences,* vol. 60, p. 092106, 2017.

[15] R. Huang, H. Liu, X. Xie, and J. Chen, "Enhancing mirror adaptive random testing through dynamic partitioning," *Information and Software Technology,* vol. 67, pp. 13-29, 2015.

[16] C. Chow, T. Y. Chen, and T. Tse, "The art of divide and conquer: an innovative approach to improving the efficiency of adaptive random testing," in *in Proceedings on 13th International Conference on Quality Software (QSIC),* 2013, pp. 268-275.

[17] T. Y. Chen, F.-C. Kuo, R. G. Merkel, and S. P. Ng, "Mirror adaptive random testing," *Information and Software Technology,* vol. 46, pp. 1001-1010, 2004.

[18] F.-C. Kuo, "An indepth study of mirror adaptive random testing," in *In Proceedings on the 9th International Conference on Quality Software, 2009. QSIC'09.* , 2009, pp. 51-58.

[19] T. Y. Chen, "Fundamentals of test case selection: Diversity, diversity, diversity," in *in Proceedings on the 2nd International Conference on Software Engineering and Data Mining (SEDM), 2010* 2010, pp. 723-724.

[20] F.-C. Kuo, T. Y. Chen, H. Liu, and W. K. Chan, "Enhancing adaptive random testing for programs with high dimensional input domains or failure-unrelated parameters," *Software Quality Journal,* vol. 16, pp. 303-327, 2008.

[21] T. Y. Chen, F.-C. Kuo, and H. Liu, "Adaptive random testing based on distribution metrics," *Journal of Systems and Software,* vol. 82, pp. 1419-1433, 2009.

[22] F.-C. Kuo, *On adaptive random testing*: Swinburne University of Technology, Faculty of Information & Communication …, 2006.

[23] K. P. Chan, T. Y. Chen, and D. Towey, "Restricted random testing," in *Software Quality—ECSQ 2002*, ed: Springer, 2002, pp. 321-330.

[24] T. Y. Chen, F.-C. Kuo, and Z. Q. Zhou, "On favourable conditions for adaptive random testing," *International Journal of Software Engineering and Knowledge Engineering,* vol. 17, pp. 805-825, 2007.