Mirror Adaptive Random Testing by Static Partitioning

Kamelia Yousofi Barforoush, Farhad Ramezani, Homayun Motameni

Abstract— software's are being used extensively and having a huge impact on humans life. Random testing as one of the simplest testing methods is being used to test software systems. Based on the rational that distributing test cases more evenly will result in having batter chance to reveal non-point pattern failure regions, various Adaptive Random Testing (ART) methods have been proposed. In this paper we propose mirror adaptive random testing by static partitioning which while having same computational overhead cost will have better performance compared with that of mirror adaptive random testing.

Index Terms—random testing, adaptive random testing, mirror, static partitioning.

I. INTRODUCTION

Testing a program with all possible inputs referred to as exhaustive testing is not feasible [1]. To improve failure detection capability of test cases, researchers proposed various methods ([2, 3, 4, 5, and 6]). Random testing as a standard software testing method has been in use for many years [7, 8]. Random testing starts with selecting test cases randomly until the time that a failure is revealed or the tester is out of resources. Random testing has been used for calculating reliability estimates [9], mainly because of its simplicity. The main criticize that exist against random testing is that does not uses of any information from the program under the test [10]. Chan et al in [11] showed that in most programs failure patterns can be categorized into three fold. These categories are: 1) block pattern; 2) strip pattern; or 3) point pattern. The block pattern which is shown in Fig. 1. Covers the case in which failures are clustered in one or more continues areas, and is the most common case. The strip pattern which covers the case which involves a continuous area that is elongated in one dimension while quite narrow in the other is presented in Fig.2. The last pattern is point pattern in which covers the case that failure patterns are distributed in small groups in the input domain as in Fig.3. Considering that mostly failure patterns in software's can fit into one of the categories mentioned, with the intention of improving the effectiveness of random testing some methods are proposed in [12,13]. These methods take into account location of previously executed test cases and improve the Efficiency by creating test case considering those previously executed test cases locations. A general consensus exist that testing (ART) methods have been proposed. The intuition behind ART is that if newly generated test case are far more even distribution of test cases helps in revealing failures sooner. Based on this simple idea, numerous adaptive random enough from previous once the failure regions will be find sooner.

Fig. 1. Block pattern

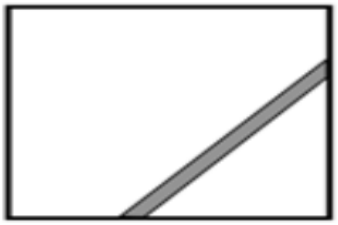

Fig. 2. Strip pattern

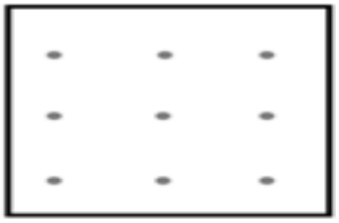

Fig. 3. Point pattern

In [14] two basic ART method is proposed. Distance-Based ART (D-ART) calculates distance of new generated test cases to all previous ones. Restricted Random Testing (RRT) uses similar approach creating restricted area around each test case. Both of this methods have high computational overhead.

To decrease this computational overhead Mirror Adaptive Random Testing (M-ART) has been proposed in [15]. M-ART method suggests to partition the input domain into M disjoint subdomains, and tries to decrease the computational over head by creating test cases only from one partition.

In this paper we propose a method that increases the performance of M-ART by choosing partitions in a systematic way.

This article is organized as follows: In section 2, various ART methods are presented. In section 3, the algorithm of the proposed method, named mirror adaptive random testing

Manuscript Received on May 04, 2015.

Mrs. Kamelia Yousofi Barforoush Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran

Mr. Farhad Ramezani Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran

Dr. Homayun Motameni Department of Computer Engineering, Sari Branch, Islamic Azad University, Sari, Iran

by static partitioning, is presented. In Section 4, we present our simulation results. Our conclusion is presented in Section 5.

II. RELATED WORKS

In D-ART at the start of the algorithm a test case is selected randomly then list of candidate test cases are generated. According to the rational of ART the test case which is farthest to the executed test cases should be chosen. So the distance of every test case In the candidate set to the previous executed test cases is measured and the one that is farthest is chosen [16]. If we consider the size of the candidate test cases as *e* and the number of test cases required for the first failure as *n*. because every time we have to calculate the distance between every created test case and all of the previous test cases, the computational overhead which is required for choosing the ith test case could be calculated as *e*(i-1)*. As a result, the order of time complexity of Distance-Based ART could be calculated by Eq. 1, [17].

$$\sum_{i=1}^{n} e \times (i-1) = e \times \sum_{i=1}^{n-1} i \in O(n^2) \quad (1)$$

In restricted random testing whenever a test case which does not reveals any failure is tested a restricted area around that test is created. The next test case will be selected randomly and if it is out of all the failure causing inputs is generated. If it is not out of all the failure causing inputs then it is discarded and another test case will be chosen randomly. In the i'th round for choosing i'th test case we require selection of a test case that is outside of all the i-1 previously restricted regions. If we assume *c* as the number of test cases we require to generate to find a test case out of all the restricted regions then the order of time complexity of RRT could be calculated by Eq. 2, [17]:

$$\sum_{i=1}^{n} c \times (i-1) = c \times \sum_{i=1}^{n-1} i \in O(n^2) \quad (2)$$

Having this mind that the computational overhead of both of the proposed approaches are quadratic. Various adaptive random testing has been proposed to decrease the computational overhead such as ART by random portioning and ART by bisection in [14]. Both methods use partitioning to decrease the overhead. Advantage of this algorithm is that they do not create a candidate list but they choose a subdomain within the main domain in which the next test case should be selected within that subdomain. While the rationale behind both is the same the difference between them is because of the partitioning scheme used. In ART by random partitioning every time that a test case is executed. It will partition the subdomain which itself is located in into 4 subdomain. Then among all the existing partitions the biggest is selected as the next test case generation area. The reason for choosing the biggest partition is that the probability for a test case being chosen from the biggest partition to be farther from all the previously executed partitions is high. In ART by bisection in each iteration the input domain is divided into two section and an area will be selected as the next area to select a test case from it only if it does not contain any previous executed test cases. The time complexity of both of the above mentioned approaches are $o(n \log n)$ with respect to the number of executed test cases.

To improve the failure detection capability of the algorithms localization has been proposed. Although using localization improved the performance it created a phenomenon called edge preference in ART. To eliminate this problem K.K.Sabor et al used an enlarged input domain method in [18, 19].

Despite the fact that the two mentioned methods decreases the computational overhead, they did not show a good performance compared to the D-ART or RRT. To improve the performance while lowering the computational overhead ART through iterative partitioning has been proposed. In this approach in each iteration the input domain is partitioned to $p \times p$ equal subdomain. Then up until the time that at least one subdomain not having any test case exist, test cases are selected from empty subdomains. When one test case is chosen and tested from each subdomain, the current partitioning scheme is discarded and a new $p+1 \times p+1$ is created. Every time that a new partitioning scheme is constructed, all the previous test cases from previous subdomain should be mapped to the new subdomains. In this case there will be new subdomains not containing any test cases. The algorithms continues till the time that a failure is detected or testing resources are exhausted. The main overhead that exist in this approach is the overhead of mapping previous test cases into the new subdomain. As Expressed in [17] if we call the number of dimensions of input domain *k* and the number of test cases required for revealing the first failure *n* then the time complexity of this algorithm will be $O(3^{k+1}.n^{1+1/k})$. if the value of *k* become large then 3^{k+1} will be a very large value [17].

To further decrease the computational overhead while maintaining performance. K.K.Sabor et al in [20] proposed adaptive random testing by static partitioning. They used coloring technique to color subdomain. In this approach the input domain is divided into $p \times p$ subdomains and each subdomain is colored to be white, green, yellow or red. Red cells are cells that contain at least 1 test case generated within them so they are not suitable to be chosen as the next test generation region. White cells are the cells that not only does not contain any tested test case but also are not adjacent to any previously tested test cases. Green cell are the one that does not contain any test case tested before but they have one and only one neighbor cell which contains a tested test case and is red. Yellow cells are the cells that does not contain any tested test case but have at least two neighbor cells containing tested test cases. In this approach up until the time that white cells exist, white cells are chosen afterwards if no white cells exist green and yellow cells are chosen till the time that all cells become red. If all the cells become red and no failure has been detected, all the cells will be colored white and the procedure will get started again. The approach will continue till the time that a failure causing input is found or resources get exhausted. Mirror adaptive random testing has been proposed in [15]. In Mirror adaptive random testing the input domain is partitioned. Afterwards any existing ART method can be applied to the source subdomain. After testing the test case if no failure is revealed the test case will be mirrored to each remained partition and tested. Thus if we partition the input domain into M disjoint subdomains the computational overhead will decrease by $\frac{1}{m^2}$.

In M-ART we have two kind of subdomains source subdomain and mirror subdomain. The test case is chosen from source subdomain

then with the help of a mirror function the test case in the source subdomain will be mapped to the mirror subdomains.

III. METHODOLOGY

In this section we propose our approach that will increase the performance of M-ART by more even distribution of test cases.

In the first step input domain is partitioned into $p \times p$ subdomains. Then first test case is chosen from the source subdomain if the test case is a failure causing input then testing will be finished and debugging will start but, if not the cell is colored red and it's adjacent are colored green if they are white and are colored yellow if they are green. In this approach to improve the performance of mirror adaptive random testing we try to choose partitions that are far from the previously executed test cases, so the priority is given to white cells then to green and after all to the yellow cells. If all the cells are red then none are capable of being selected for being the next test case generation region so all of them will be colored white and algorithm will start again this process will continue until a failure causing input is found or the resources are exhausted. Fig 4 shows the iterations of the proposed approach.

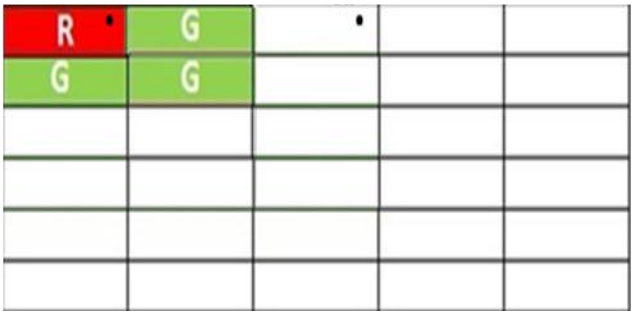

Fig.4 Mirror Adaptive Random Testing By Static Partitioning first iteration

In Fig 4. One test case is selected from the source subdomain and the test case did not reveal any failure. The cell is colored red and all adjacent cells are colored green. In the second iteration the priority is with the white cells so the test case is mirrored to a white cell shown in Fig 4. Assuming that this test case did not reveal a failure either. We colored the cell red and all the adjacent cells to green if they were white and yellow if they were green. The result will be as Fig.5

Fig 5. Mirror Adaptive Random Testing By Static Partitioning second iteration

In the third iteration next white cell will be found then the test case will be mirrored to it and will be generated within that region. If the test case reveals a failure then testing will end otherwise the adjacent cells and the cell itself will be colored and algorithm will continue.

A. *Mirror Adaptive Random Testing by static partitioning*

1. Partition input domain into $p \times p$ sepretaed subdomains
2. Color all the cells white.
3. While resources are not exhausted or failure causing input is not found
   - 3.1 choose one test case from the source subdomain
   - 3.2 color the source subdomain cell red.
   - 3.3 if the test cases reveals failure break the testing
   - 3.4 else while all the cells are not red
     - 3.4.1 If at least one white cell exist
       - 3.4.1.1 map the source test case to the white cell and test it
       - 3.4.1.2 if test reveals failure break test
       - 3.4.1.3 else color the cell red and adjacent cell to green if they are white and to yellow if they are green
     - 3.4.2 else if at least one green cell exist
       - 3.4.2.1 map the source test case to the green cell and test it
       - 3.4.2.2 if test reveals failure break test
       - 3.4.2.3 else color the cell red and adjacent cell to green if they are white and to yellow if they are green
     - 3.4.3 else if at least one yellow cell exist
       - 3.4.3.1 map the source test case to the yellow cell and test it
       - 3.4.3.2 if test reveals failure break test
       - 3.4.3.3 else color the cell red and adjacent cell to green if they are white and to yellow if they are green
   - 3.5 color all cells white

IV. EVALUATIONS

To evaluate the effectiveness of M-Art a series of evaluation has been conducted. In these empirical evaluations input space domain is considered to be 2-dimensional and square. For each failure rate in each run a failure causing region of specified size and pattern was assigned within the input domain. For block pattern a square of specified size has been used as failure causing region. For strip pattern two points on the adjacent borders of the input domain has been randomly selected and have been connected to each other to form a strip pattern with predefined size. For the point pattern circular regions with size *w*, which is chosen according to failure rate and number of regions, were randomly located in the input domain without overlapping each other. For each simulation, 5000 test runs were executed and the average F-measure is calculated. F-measure is defined as the number of test cases that are needed to be executed since the revealing of the first failure. On the ground that when a failure is detected testing will stop and debugging will start this measure reflects the real world practices. If we assume the probability of all test cases in input domain to be failure causing input as equal, then having failure rate = *c*, random testing F-measure will be $\frac{1}{c}$.

Since increasing number of mirrors in M-ART will degrade randomness of testing method it is suggested not to pass over a partitioning scheme of 6×6. Since we proposed an approach that choose partitions in a way that it intends to select partitions that are far away from those partitions containing previously executed test cases, we can divide the input space into a 50×50 cells still preserving randomness of our approach.

In the first part of evaluations we compared effectiveness of our approach on block pattern with random testing. We evaluated our approach considering different failure rates. The result is shown in Table 1.

Table 1. M-ART by static partitioning with block pattern

Failure Rate	Random F-Measure	F-Msr For Proposed Method	F(ART) / F(Random)
0.01	100	53	53
0.005	200	105	52.5
0.001	1000	509	50.9
0.0005	2000	1000	50

As it is shown in Table 1 by decreasing the failure rate, the effectiveness of our approach decreases as well. This is expected as when failure rate decreases the probability of an input to be failure causing decreases which causes degrade in method performance.

In the second part of experiments we evaluated our approach on the strip failure pattern. The result is shown in Table 2.

Table 2. M-ART by static partitioning with strip pattern

Failure Rate	Random F-Measure	F-Msr For Proposed Method	F(ART) / F(Random)
0.01	100	75	75
0.005	200	74	74
0.001	1000	724	72.4
0.0005	2000	1300	66

As it is shown in Table 2. The performance of the M-ART by static partitioning applied on strip pattern is less compared with the time that it is applied on block pattern. The reason is the essence of the ART, the most favorable pattern for ART methods is block pattern, and then second favorable is strip pattern while the worst is point pattern. Having a decreased performance in strip pattern our approach still outperforms random testing. The performance decreases by decreasing failure rate since when the probability of an input to be failure causing one decreases, the performance decrease will be Inevitable.

In the third part of experiments we studied the performance of our method in existence of a strip pattern. We evaluated our approach considering different failure rates. The result is shown in Table 3.

Table 3. M-ART by static partitioning with strip pattern

Failure Rate	Random F-Measure	F-Msr For Proposed Method	F(ART) / F(Random)
0.01	100	53	101
0.005	200	105	95
0.001	1000	509	94
0.0005	2000	1000	94

As it is shown in Table 3, applying our approach does not increase the performance. The reason is that in case of point pattern the failure causing inputs are not gathered in one part of input domain so more even distribution of test cases does not necessarily lead to better performance. This is consistent with our result as it is shown in Table 3 not only our approach did not increase but also decreased in some cases.

Summarizing, we proposed an approach which increases the performance of M-ART by more even distribution of test cases. We tested our approach in case of existence of all three possible failure patterns. In case of block pattern we increased the performance by 47%. In case of strip pattern our approached increased the performance by 25%. In case of point pattern since more even distribution of test cases does not lead to better performance we have partial increase in performance. We did not expect any better performance in point pattern and our result shown that though in some cases some partial increase in performance happens it is not generalizable.

V. CONCLUSION

In this article a new Adaptive random testing approach named Mirror adaptive random testing by static partitioning with the aim of increasing performance have been proposed. This approach increased the performance of mirror adaptive random testing by choosing mirror partitions in a special order that increase the even distribution of test case. In each iteration the partition that contains the executed test case is colored and all its adjacent are colored accordingly. Then in the next iteration the next partition that the test case in the source subdomain should be mapped to it will be chosen according to its color.

The performance of the proposed approach has been compared with that of random testing. The approach outperformed random testing when failure pattern was of block or strip type. In case of point pattern the performance of approach was near the random testing preforming partial better or worse in some cases.

Considering the performance increase we suggest testers that if adaptive random testing is suitable for their goals to use Mirror adaptive random testing by static partitioning. Pursuing this further this method can be used being combined with other approaches.

In our future work we want to test our approach with different number of partitions. We also want to combine our approach with different adaptive random testing approaches to increase the performance.

REFERENCES

1. B.Beizer, Software Testing Techniques, Second Edition, Van Nostrand Reinhold Company Limited,1990
2. J. W. Laski , B. Korel, "A data flow oriented program testing strategy", IEEE Transactions on Software Engineering, Vol. 9, No. 3, pp. 347-354, 1983.
3. J. Offutt and S. Liu, "Generating test data from SOFL specifications", The Journal of Systems and Software, Vol. 49, No. 1, pp. 49-62, 1999.
4. J. Offutt, S. Liu, A. Abdurazik and P. Ammann, "Generating test data from state-based specifications", Software Testing, Verification & Reliability, Vol. 13, No.1, 2003, pp. 25-53.
5. P. Stocks and D. Carrington, "A framework for specification-based testing", IEEE Transactions on Software Engineering, Vol. 22, No. 11, pp.777-793, 1996
6. L. J. White and E. I. Cohen, "A domain strategy for computer program testing", *IEEE* Transactions on Software Engineering, Vol. 6, No. 3, pp. 247-257, 1980.
7. R. Hamlet. "Random testing". In Encyclopedia of Software Engineering, pp 970–978. Wiley, New York, 1994.
8. P. S. Loo and W. K. Tsai. "Random testing revisited". Information and Software Technology, Vol.30. No 7, pp. 402-417, 1988.
9. E. J.Weyuker and B. Jeng. "Analyzing partition testing strategies". IEEE Transactions on Software Engineering, Vol. 17, No. 7, pp.703-711, July 1991.
10. G. Myers, The Art of Software Testing, NewYork: John Wiley & Sons, 1979
11. F. T. Chan, T. Y. Chen, I. K. Mak, and Y. T. Yu. "Proportional sampling strategy: guidelines for software testing practitioners". Information and Software Technology, Vol. 38, No 12, pp775– 782, 1996.
12. T. Y. Chen, T. H. Tse and Y. T. Yu, "Proportional sampling strategy: a compendium and some insights". The Journal of Systems and Software, 58, pp. 65-81, 2001.
13. I. K. Mak, on the effectiveness of random testing, Master's thesis, The University of Melbourne, 1997.
14. T. Y. Chen, G. Eddy, R. Merkel and P. K. Wong, "Adaptive random testing through dynamic partitioning", Proceedings of the 4th International Conference on Quality Software (QSIC2004), pp79 – 86, 2004.
15. T.Y.Chen, F.C.Kuo, R.G.Merkel, S.P.Ng , "Mirror adaptive random testing", Elsevier,2004
16. T. Y. Chen, H. Leung, and I. K. Mak, "Adaptive random testing" in Proceedings of the 9th Asian Computing Science Conference, LNCS 3321, pp. 320-329, 2004.
17. T. Y. Chen, D.H.HUANG AND Z.Q. ZHOU, "On Adaptive Random Testing Through Iterative Partitioning", journal of information science and engineering 27, pp-1449-1472, 2011.
18. K. K. Sabor and M. Mohsenzadeh "Adaptive random testing through dynamic partitioning by localization with restriction and enlarged input domain". In Proceeding Of the 2012 international conference on information technology and software engineering, Lecture Notes in Electrical Engineering Volume 212, pp 147-155, 2013.
19. K. K. Sabor and M. Mohsenzadeh. "Adaptive random testing through dynamic partitioning by localization with distance and enlarged input domain". International journal of innovative technology and exploring engineering. November 2012, pp 1-5.
20. K. K. Sabor and S. Thiel "Adaptive random testing through static partitioning". Accepted in th IEEE/ACM International Workshop on Automation of Software Test.

AUTHORS PROFILE

Mrs. Kamelia yousofi barforoush obtained her bachelor of science in software engineering on 2010 and her master of science in software engineering in 2014. Since 2013 she worked as network engineer in shuttle of Babol in Iran.