Comparison analysis of two test case prioritization approaches with the core idea of adaptive

Jian Ding [1], Xiao-Yi Zhang[2]
1. AVIC CHENG DU AIRCRAFT INDUSTRIAL (GROUP) CO., LTD, Chengdu
E-mail: 21125559@qq.com
2. Department of Automatic Control, Beihang University, Beijing
E-mail: xiaoyizhang@buaa.edu.cn

Abstract: Test case prioritization problem (TCP) has been widely discussed. It aims to controlling the test case execution sequence to improve the effectiveness of software testing. The key issue of TCP is to identify which test cases can provide useful information for failure detection and fault localization. So far, many TCP approaches have been proposed. Among them, Adaptive Random Testing (ART) and Dynamic Random Testing (DRT) are two of the most popular approaches to solve TCP with a basic idea borrowed from Cybernetics: adaptive. Both ART and DRT has been widely explored and observed with good performances in experimental studies. Nevertheless, although they are proposed by two related research groups, they are developed independently and in parallel. In fact, their mechanisms have many similarities and differences and, for the completeness of the domains of Adaptive Testing and Software Cybernetics, many issues concerning the comparison between these two approaches should be further explored. In this paper, we specifically explores the relationship between these two adaptive TCP approaches. Their mechanisms are described respectively with explorations of their distinctions, similarities, and respective characteristics. Moreover, based on these explorations, we analyse their advantages from the aspects of failure detection and fault understanding. During the analysis, a symbolic-graphic combination method is applied. Finally simulation based on real-life programs is conducted to observe our analysis. Our comparison analysis can support the selection of a proper testing approach according to various practical environments with different targets. Furthermore, the clarification of the two easily confused concepts is also a complement for the framework of Adaptive Testing and Software Cybernetics.

Key Words: Software testing, Adaptive testing approaches, Dynamic, Software Cybernetics

1 INTRODUCTION

In software engineering, software testing is an activity involving the execution of the subject program to evaluate one or more properties of interest. The main targets of software testing are to detect program failures [1] and collect enough information to support debugging [2–4]. For software working in delicate environments, such as flight control software, a failure may cause disruptive catastrophes. On the other hand, because the architecture of current software tends to be more complex, the faults are more difficult to be discovered. Therefore, software testing is an essential, but resource-consuming activities in the cycle of software development and maintenance [5]. A hot research topic concerns the automation of the testing process to reduce its expense. One of the main streams aims to find effective ways of ordering the test case executions, called Test Case Prioritization problem (TCP) [6]. If the test cases useful to the testing target are executed first, the effectiveness of the whole testing process can be improved.

The most basic TCP approach is Random Testing (RT). During the testing process, RT selects test cases from the given test suite following a pure random strategy. Intuitively, RT does not use any searching heuristics, thus it can hardly find program failures and reveal the faulty components effectively. However, many experimental studies has proved that RT is not only basic but also effective in failure detection. This phenomenon can be explained by software testing psychology [7]. Specifically, testers usually tend to believe that the software under test (SUT) is correct. Thus, any subjective testing strategy will be affected by this subconsciousness and keep away from the failure revealing input. However, pure random is an unconscious strategy. It well avoids this psychological problem, ensuring an unbiased testing process and convergent failure detection results.

Although effective, RT can still be expected to be improved by designing rational heuristics. Chen et al. [8] have proved that we can save more than 50% overhead for the detection of certain failure patterns following a sophisticated arrangement the test selection process, without using any knowledge about the location of the failure domain. This motivates researchers to explore more delicate TCP approaches to improve test effectiveness. During the optimization of selection strategies we should also not introduce many subjective factors. Thus, applying adaptive mechanisms to adjust the prioritization criteria according to the temporary status during the testing process is a potential way.

To date, many approaches have been proposed to automatically arrange the process of testing. Among them there are

978-1-5090-4657-7/17/$31.00 ©2017 IEEE

Authorized licensed use limited to: Universita degli Studi di Salerno. Downloaded on June 11,2025 at 18:30:42 UTC from IEEE Xplore. Restrictions apply.

two well-known approaches with the core idea of "adaptive", the Adaptive Random Testing (ART) and the Dynamic Random Testing (DRT). ART aims to improve the degree of test case diversity while DRT aims to adapt the test profile to the property of failure domain. Both of them keep some extents of randomness to reserve the advantages of RT. These two approaches has also been observed with good performance [2, 3, 9]. ART and DRT are respectively proposed by Chen et al. [1,8,10,11] and Cai et al. [9,12,13]. This two teams have frequent communications and cooperations [2, 3, 14]. However, ART and DRT are respectively developed in parallel and, except for some literature reviews with seldom have intersections [14]. Their verifications are also based on different criteria.

Because both ART and DRT can be called "adaptive random approaches" and have many similarities. Students and testers are easy to get confused when trying to learn their mechanisms. In addition, researches concerning adaptive testing approaches may be imprecise in their usages and derivations. Furthermore, for testers who want to put them into practice, it may be difficult for them to make an appropriate choice. Thus it is meaningful to provide a discussion about the relationships between ART and DRT.

In this paper, we make a special analysis about the relationships between ART and DRT. Their mechanisms are described respectively and some critical and confusing questions are analysed, including which one is "more" adaptive, how randomness is integrated into these two approaches and how to explain their basic intuitions. According to the analyses, the distinctions and similarities between ART and DRT are clarified. Furthermore, we explore more intrinsic characteristics respectively from the aspects of failure detection and fault understanding which are respectively two critical steps of testing and debugging. Here we use the symbolic-graphic combination method to make our analyses explorations more directive and clear. Finally graphic simulations based on real-life programs are conducted to further verify our conclusions.

The rest of this paper is organized as follows. Section 2 describes ART and DRT, respectively. Section 3 further explores the questions related to their mechanism and some more intrinsic characteristics related to their ability of failure detection and fault understanding. In Section 4, we conduct simulations to verify the advantages of ART and DRT, respectively. Finally, conclusions are given in Section 5 followed with out future work.

2 General description of ART and DRT

2.1 Test Case Prioritization Problem

Denote PG as the subject program and the set T as the given test suites. Here, a specific test case in T is denoted by t. Usually, T could be the entire input domain whose size are quite large such that exhaustive execution of all test cases in T can be slow, or even impossible. Thus, in many situations, testers should rationally arrange the testing process such that they could complete the target with a limited cost. Consequently, the Test Case Prioritization problem (TCP) is proposed to improved the effectiveness of testing:

Given: The subject program PG and a test suite T.

Problem: Find a permutation of T, denoted by $\rho(T)$, such that executing test cases following the sequence of $\rho(T)$ can reach the intended testing target earlier.

Intuitively, if the test cases providing more useful information are executed first, the whole process of testing will be accelerated. Thus, the critical issue of devising a good TCP approach is to find a rational criterion to measure the usefulness of each test case t in T.

2.2 Adaptive Random Testing

When studying the patterns of program failures, researchers found that the failure causing inputs tend to cluster together, likely to form contiguous failure domains [15]. Thus, if the output of test case t is correct, the test cases which are close to t may also not be failure revealing. Oppositely, if the test cases are quite different with each other, failures will be exposed earlier. Therefore, a test case is useful if it is much distinct with the executed test cases. Motivated by this assumption, ART is developed with the key intuition of improve the degree of test case diversity.

ART identifies a test case as useful if it is much different from the current executed test suite. In addition, to relieve the computational overhead of test case selection and meanwhile keep the advantages of Random Testing (RT), some extents of randomness are also added.

The most basic algorithm is called the Fixed Size Candidate Set ART algorithm (FSC-ART) [8]. During the selection of test cases, suppose the entire test suite is T can be divided into $T_e \cup T_r$, where T_e denotes the set of already executed test cases and T_r denotes the set of remaining test cases. Now we should select a test case from T_r as the next execution. During the implementation, FSC-ART firstly forms a candidate test suite $T_c \subset T_r$ by randomly selecting k test cases from T_r. Then for each test $t \in T_c$, FSC-ART calculates the distance between t and each test case $t' \in T_e$. Denote the distance between test case t and t' as $Dis(t, t'$. Then the distance between each test case $t \in T_c$ and the test suite T_e, denoted by $Dis(t, T_e)$, are calculated as follows.

$$Dis(t, T_e) = \min_{t' \in T_e} Dis(t, t') \tag{1}$$

Finally, the test case t^* satisfying

$$t^* = \arg\max_{t \in T_c} Dis(t, T_e) \tag{2}$$

will be selected as the next execution.

2.3 Dynamic Random Testing

Another limitation of the pure random testing is that the test profile will not change even if the failure causing input has already been found. Here, the term test profile can be explained as a set of rules, principles, or limitations which controls the generation or selection of test cases [12, 16]. Suppose the program input domain can be divided into several sub-domains. That is to say, the entire test suite T can be divided into several subsets $T_1, T_2, T_3, \ldots, T_m$ satisfying:

1. $T_1, T_2, T_3, \ldots, T_m \subset T$
2. $T_1 \cup T_2 \cup T_3 \cup .... \cup T_m = T$

Authorized licensed use limited to: Universita degli Studi di Salerno. Downloaded on June 11,2025 at 18:30:42 UTC from IEEE Xplore. Restrictions apply.

Here $T_1, T_2, T_3, \ldots, T_m$ are called as the partitions of T, Commonly, they can be derived according to the specification of the subject software such as the software requirements. Many approaches such as the well known "DESSERT" [17] and "CHOCOLATE" [18] have been proposed to accomplish this work efficiently.

Originally, the main purpose of software testing is to detect whether the program is correct. However, when a failure is found, the process of debugging will start to remove the faults related to the detected failure [19]. In this situation, the target will be turned to fault understanding, that is, making engineers understand how the failures were generated. Intuitively, if the failure pattern, e.g., the shape and partitions of the failure domain, can be effectively identified, the mechanism of the detected failures will be more easily to be described.

Cai et al. [12] notice this transition of testing targets. They introduce a new TCP approach called Dynamic Random Testing based on the ideas of Software Cybernetics. The key intuition of DRT is to dynamically adjust the test profile during the process of testing, according to the direct feedback of the current testing results achieved from T_e. DRT arranges the process of test case prioritizatioin as follows.

Define a test profile as a vector $\mathbf{p} = [p_1\ p_2\ \ldots\ p_m]$, satisfying that

$$\sum_{i=1,2,\ldots,m} p_i = 1. \tag{3}$$

Here p_i represents the probability of selecting test cases from partition T_i. Originally, without any heuristic, RT is conducted and the test profile satisfies the equation of

$$p_1 : p_2 : \ldots : p_m = |T_1| : |T_2| : \ldots : |T_m|. \tag{4}$$

This means the sampling proportion of the sub-domains equal to the proportion of their size. Then DRT select a test case t according to $\mathbf{p}$. If t reveals a failure, DRT will pays more attention on the related sub-domain in later selections. Oppositely, if t is a pass test case, DRT will pay less attention on the related sub-domain. Specifically, suppose $t \in T_l$, if t reveals a failure, then the test profile $\mathbf{p}$ will be adjusted as follows.

$$p_j = \begin{cases} p_j - \varepsilon/(m-1) & p_j \geq \varepsilon/(m-1) \\ 0, & p_j < \varepsilon/(m-1) \end{cases} for j \neq l \tag{5}$$

$$p_l = 1 - \sum_{j \neq l} p_j \tag{6}$$

And if t is passed, the test profile $\mathbf{p}$ will be adjusted as follows.

$$p_l = \begin{cases} p_l - \delta & p_l \geq \delta \\ 0, & p_j < delta \end{cases} \tag{7}$$

$$p_j = \begin{cases} p_j + \delta/(m-1) & p_l \geq \delta \\ p_j + p_l/(m-1), & p_l < \delta \end{cases} for j \neq l \tag{8}$$

This selection will continue until the stop criterion (e.g. the quantity of test case executions) is satisfied.

3 Exploration of the mechanisms and characteristics

After the description of ART and DRT, we will explore some meaningful issues related to their mechanisms and characteristics from the prospect of their abilities for failure detection and fault understanding.

3.1 ART and DRT: extensions of RT

From the description, ART and DRT are based on their own heuristics. However, both of them are extensions from Random Testing (RT) and also keep some extents of randomness.

For ART, from Eq. (2), it only considers the next execution in the candidate set T_c. Because T_c is formed by randomly selecting a fixed number of test cases, the final selected test case can also be regarded as a random variable. It can be proved that the smaller size of T_c the higher degree of randomness that ART has. If we set $|T_c| = 1$, ART will be equivalent RT. Normally, we set $|T_c|$ value from 3 to 10 and quite high degree of randomness is also kept.

DRT also has high degree of randomness. Firstly, it selects the test case according to a test profile, which identifies the probability of choosing each sub-domains. Normally, at the beginning of testing, the probabilities of selecting most sub-domains will not be 0, which means most sub-domains will have probabilities to be selected. In addition, when a specific sub-domain is focused, we also need to conduct random selection to determine the final test case in this sub-domain. Thus, the test selection also possesses quite high degree of randomness.

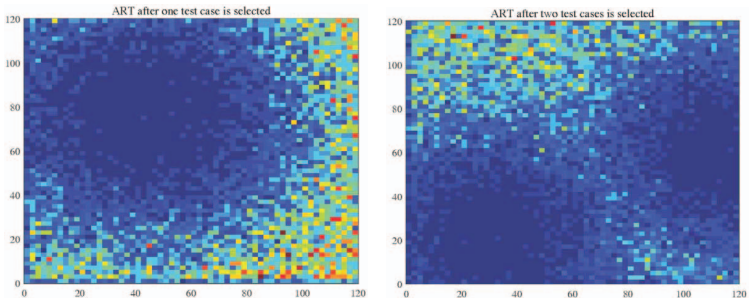
Fig. 1 shows the probability distribution of selecting the next test case using different TCP approaches, where the warm colors represent areas with quite higher probabilities while the cold colors represents those with quite lower probabilities. From Fig. 1, we found that for all the situations, the next test cases are random variables in certain areas in the input domain. Because of these randomness, the selection process keeps the basic principles of testing psychology, making them universally applicable.

3.2 The heuristics behind ART and DRT

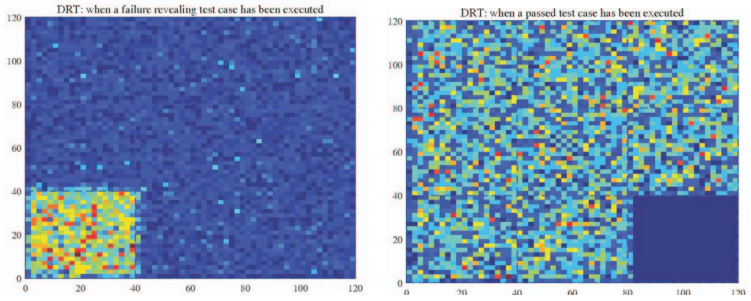
From Fig. 1, we can find more interesting phenomena. In Fig. 1(a), the executed test suite T_e contains only one test case whose input has the coordination value of $(41.76, 74.02)$. We can see that, in the input domain, the distribution spread from this point and the regions far from the point will have higher probabilities to be selected. Similarly, in Fig. 1(b), the distribution spread from the two selected test inputs $(31.0120.88)$ and $(115.3159.53)$. This observation visually illustrates the heuristic of ART: preferring the test cases far from the executed test suite T_e. This heuristic can also be reflected by Eq. 2.

On the other hand, In Fig. 1(c), in the previous executions, i.e. the test cases belonging to T_e, a failure has been detected in sub-domain T_1. Then for the next selection, all the test cases in T_1 will have higher probability to be selected. In Fig. 1(d), in the previous execution, a test case in sub-domain T_3 is selected and it is passed, Then for the next selection, test cases belonging to sub-domain T_3 will

Authorized licensed use limited to: Universita degli Studi di Salerno. Downloaded on June 11,2025 at 18:30:42 UTC from IEEE Xplore. Restrictions apply.



(a) ART: after one test case has been selected (b) ART: after two test cases has been selected



(c) DRT: after a failure revealing test case has been executed (d) DRT: after a passed test case has been executed

Figure 1: The probability distribution graphs for the next selection of test cases using ART and DRT, respectively.

have the lowest probability to be selected. Generally, for DRT, it prefers to examine the regions which are easy to detect failures.

Comparing the heuristics of ART and DRT, for the passed test cases, their "attitude" are quite similar: they all do not prefer to generate new test cases similar to these passed test cases. Intuitively, this preference is very helpful to failure detection.

However, for failure revealing test cases, they have converse attitudes. ART emphasizes more about the diversity of test cases and do not specially prefers the failure domain, while DRT prefers to generate test cases similar to them. Intuitively, checking more similar test cases around the failure revealing test cases, as what DRT does, is good for describing the pattern of this failure, while checking test cases far from the already selected test cases, as what ART does, can help to explore new failures caused by other faults. Thus, ART and DRT may have different abilities of of failure detection and fault description, which will be further explored in the rest of this paper.

3.3 Which is "more" adaptive?

Both ART and DRT are claimed as adaptive approaches. However, their mechanisms are quite different. When selecting the next test case execution, ART is based on the inputs of current test cases T_e while DRT is based on the current test profile and will adjust the profile according to the output of the next execution. Thus, it is an interesting issue (and also a debate) to discuss which of them is "more" adaptive.

Commonly, the concept of "adaptive" is mentioned in the domain of automatic control. Cai et al. [20] introduce the idea of Software Cybernetics, which describes the software

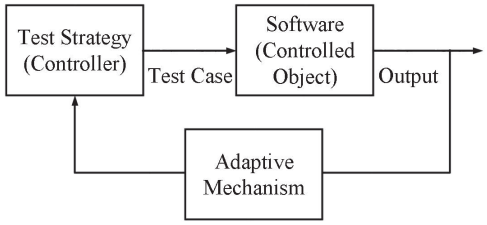


Figure 2: The framework of adaptive testing

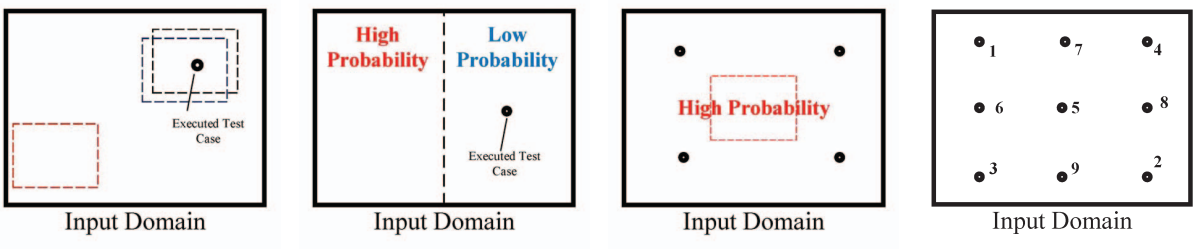
testing as Fig. 2.

From Fig. 2, the software is treated as the controlled object. It receives inputs and provides outputs according to its system mechanism. And the test strategy is treated as a controller. It can generate and provide test cases to the software. If a test approach is adaptive, according to the control theory, there should be a feedback from the software output to the controller. This means the controller should be able to adjust itself according to the current test results.

For DRT, the test profile is the controller and it will be adjusted after every execution of each test case. And the adjustment relies on whether the new execution is passed or failure revealing (See Eq. (5) $-$ Eq. (8)). In this meaning, DRT is a "real" adaptive (TCP) approach.

However, for ART, the next test case relies more on the inputs of test cases belonging to T_e and does not have much relationship with their outputs (See Eq. (2)). That is to say, the feedback is an internal mechanism of the controller and does not across the software (i.e. the controlled object). Therefore, ART is not a strict adaptive TCP approach, although many new algorithms for ART do apply the output

Authorized licensed use limited to: Universita degli Studi di Salerno. Downloaded on June 11,2025 at 18:30:42 UTC from IEEE Xplore. Restrictions apply.



(a) Rapidly expanding the input domain (b) A binary search style (c) Four test cases have been generated (d) A probable sequence of generating test cases

Figure 3: The characteristic of ART in searching the input domain

information.

Nevertheless, in a broad sense without the environment of control theory, if the future selection strategy is based on the current selected test cases, the approaches are adaptive. In this meaning, both ART and DRT are adaptive approaches.

3.4 ART: ability for searching failures

In this section, we analyse the characteristics of ART and then explain why it is good at failure detection. The problem of failure detection can be interpreted by searching for the failure domain from the entire input domain; and the ability of failure detection mainly depends on the rationality of searching strategy.

Fig. 3 illustrates the characteristics about the searching strategy of ART. The first characteristic is that ART can rapidly expand the entire input domain, which means the test suite selected by ART can efficiently examine every suspicious area. From Fig. 3(a), suppose the failure domain is a square and the failure rate equal to 1. If a test case t, represented by a point, has been examined, the square around this test case (See the black dashed box) will not contain failures. We say that t *covers* this square area. In this situation, still selecting test cases in this area is meaningless for failure detection. And it is better to check others areas which are not intersected with this area (such as the red dashed box). For example, assume 1) the size of T_c is 3, 2) the proportion between the size of failure domain T_f and the entire input domain T, i.e. $|T_f|/|T|$, is 0.1, and 3) the failure domain is not near the boundary of the input domain. For ART, according to Eq. 2, the probability of checking an intersection area, such as the area enclosed by the blue dashed box, is only 0.064 (i.e. 0.4^3), much lower than that for RT (which is 0.4). Therefore, the waste test cases are largely reduced and then the entire input domain can be covered more quickly.

The second characteristic is that the searching track is effective, similar to the algorithm of binary search. From Fig. 3(b), we evenly divide the input domain into two parts. Suppose test case t belonging to one part has been executed. Then we have high probability to select the next test case from the other part. Furthermore, if four test cases at the corners of the input domain have been executed, then we may have high probability to select the next test case at the center of the input domain (See Fig. 3(c)). And if nine test cases evenly covering the input domain have been generated, the generation process will probably following the sequence shown by Fig. 3(d)). From Fig. 3(b) $-$ Fig. 3(d), the new test cases will always have high probability to be selected in the most suspicious areas of the input domain. We find that characteristic is in accord with the idea of an effective searching algorithm, the binary search. Because ART can form a even spread of test suite no matter how the size of executed test suite is limited, it have an overall perspective for failure detection.

3.5 DRT: ability for fault understanding

In this section, we analyse the characteristics of DRT and then explain why it is good at fault understanding.

A critical issue associated to fault understanding is to describe the failure pattern. Failure pattern refers to the properties such as the shape and failure rate of a specific failure domain caused by a certain fault. If we want to describe a failure, it is better to check more test cases in the corresponding failure domain.

Having detecting a failure, ART will pay more attention on searching new failure areas and, as a result, we may not have enough test cases to describe this already detected failure. However, in current configurations of software development, if a failure have been found, the debugging process will start and engineers will need more failure description informations. Different with ART, DRT will focus more on the already detected failures.

According to Eq. (5) $-$ Eq. (8), the sampling probabilities of the high suspicious sub-domains will become higher and those of the less suspicious sub-domains will become lower. Fig. 4 shows an ideal situation. Suppose the failure probabilities of the sub-domains follow the proportion of $1:2:\ldots:9$, and the final test profile using DRT is convergent, that is, the sampling probability of the sub-domains also follow the proportion by $1:2:\ldots:9$. Under this ideal profile, DRT selects more test cases to describe the high suspicious regions while uses less test cases to describe the low suspicious regions. And, consequently, all the failure domains are well described.

Although the convergence of DRT relies on the selection of the ε and δ values and also influenced by the accuracy of partition, many well-designed algorithms for DRT have been developed to improve its convergence rate [9, 13].

Authorized licensed use limited to: Universita degli Studi di Salerno. Downloaded on June 11,2025 at 18:30:42 UTC from IEEE Xplore. Restrictions apply.

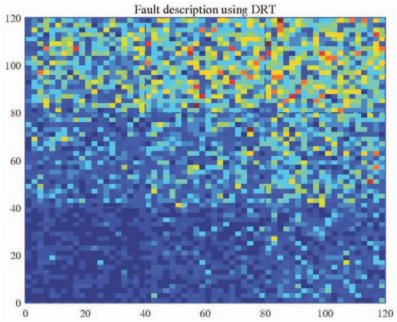


Figure 4: Using DRT to describe the failure pattern, in which the failure probabilities of the sub-domains following the proportion of $1:2:3:...:9$

Table 1: Failure detection times of each TCP approach

Approaches / Size of T_e	20	40	60
DRT	12	16	25
ART	16	20	24
RT	12	19	19

4 Simulation for the performance of ART and DRT

In this section, we will observe the performance of ART and DRT through simulation. The basic situation, including the input domain, the failure domain and the partitions, is shown in Fig. 5, where the area enclosed by the red dashed lines is the failure domain with the failure rate of 1, while the blue dashed lines partition the input domain (i.e. the entire figure) into sixteen sub-domains. This shape of failure domain and the failure rate is constructed according a real program *bessj* [21]

Firstly, we implement RT, ART and DRT respectively to detect the failure domain, and the size of the executed test suite T_e is limited to 20, 40, and 60, respectively. For each instance, we repeat 30 times and record the times of the implementations in which the failures are detected. The results are shown in Table 1.

From Table 1, we found that ART indeed has a good failure detection ability. When $|T_e| < 20$, for 16 out of 30 instances, ART detects the failures, while at only 12 instances that RT and DRT detects the failures. With the grow of $|T_e|$ the rate of failure detection for each approaches grows obviously. And ART always has a good performance, outperforming RT for all settings. In addition, DRT has good performance after executing 60 test cases, but it do not have good performance when the the size of $|T_e|$ is quite small. This indicates that DRT requires sufficient test cases to support an accurate adjustment of the test profile.

Fig. 6 illustrates the distribution of the generated test cases after 1000 executions, where the blue star points represent the passed test cases while the the red star points represent the failure revealing test cases. We can see that DRT (See Fig. 6(b)) examines more failure revealing test cases and provides a more explicitly outline of the shape of the failure domain than ART and RT (See Fig. 6(a) and Fig. 6(c) respectively), which is in accord with our analysis in Section 3.5.

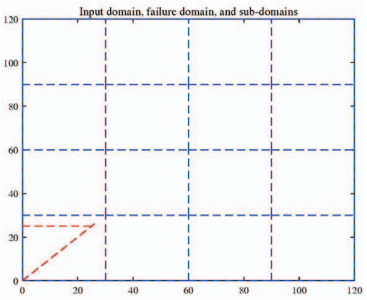


Figure 5: The basic situation

5 Conclusions and future works

This paper compares two adaptive Test Case Prioritization (TCP) approaches, the Adaptive Random Testing (ART) and the Dynamic Random Testing (DRT). We analyse their mechanism of generating test cases and arranging the test process and clarify their differences. Furthermore, we deeply explore their characteristics and advantages from the aspects of failure detection and fault understanding. According to our exploration, we derive the following conclusions:

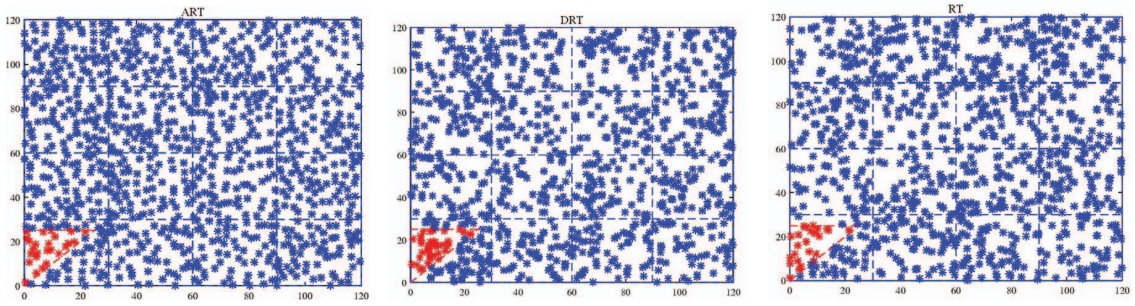
1. Both ART and DRT are extensions of RT.
2. ART and DRT have different heuristics. Especially, they have distinct attitudes and concerning the failure revealing test cases and quite similar strategies when tackling the passed test cases.
3. DRT relies on the test results of the current executed test cases; thus it is "more" adaptive, while ART is not.
4. ART is good at failure detection because of its characteristics that it can rapidly cover the entire input domain and its search strategy is similar to the algorithm of binary search.
5. DRT is good at fault understanding because it has a good ability to describe the pattern of the failure domain.

Both ART and DRT are popular TCP approaches and many details concerning their mechanism are easy to get confused. Thus, we believe our clarification can make contributions in combing the knowledge in the research domains of Adaptive Testing and Software Cybernetics.

In addition, according to the above list of our conclusions, engineers can select the proper approach according to their testing target. For example, if engineers focus on exploring the detected failures and removing the related bugs, they may choose DRT. And if engineers are making an overall testing for the software developed by the third-party company, ART is recommended.

Last but not the least, our work may facilitate the better study of TCP approaches. Having known their searching

Authorized licensed use limited to: Universita degli Studi di Salerno. Downloaded on June 11,2025 at 18:30:42 UTC from IEEE Xplore. Restrictions apply.



(a) ART (b) DRT (c) RT

Figure 6: The failure description of each TCP approach

styles, we can combine their advantages to develop better testing strategies. For example, we can search the non-failure domains using ART and setting a conversion probability to conduct DRT after finding the failure domains.

This paper is our first attempt to specifically distinguish ART and DRT. In our future work, more details should be further studied, such as different types of failure patterns, the influence of partition accuracy on DRT, the mechanisms of different algorithms for ART, etc. In addition, we will also make use of their respective advantages to propose new adaptive testing approaches, as discussed above.

REFERENCES

[1] 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, no. 1, pp. 60–66, 2010.

[2] X.-Y. Zhang, D. Towey, T. Y. Chen, Z. Zheng, and K.-Y. Cai, "Using partition information to prioritize test cases for fault localization," in *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, vol. 2. IEEE, 2015, pp. 121–126.

[3] Y. Guo, X. Y. Zhang, and Z. Zheng, "Exploring the instability of spectra based fault localization performance," in *Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*, vol. 1. IEEE, 2016, pp. 191–196.

[4] S. Yoo, M. Harman, and D. Clark, "Fault localization prioritization: Comparing information-theoretic and coverage-based approaches," *ACM Transactions on Software Engineering and Methodology*, vol. 22, no. 3, p. 19, 2013.

[5] X. Xie, T. Y. Chen, F.-C. Kuo, and B. Xu, "A theoretical analysis of the risk evaluation formulas for spectrum-based fault localization," *ACM Transactions on Software Engineering and Methodology*, vol. 22, no. 4, p. 31, 2013.

[6] G. Rothermel, R. H. Untch, C. Chu, and M. J. Harrold, "Prioritizing test cases for regression testing," *Software Engineering, IEEE Transactions on*, vol. 27, no. 10, pp. 929–948, 2001.

[7] G. J. Myers, T. Badgett, and C. Sandler, *The Psychology and Economics of Software Testing*. John Wiley & Sons, Inc., 2012, pp. 5–18. [Online]. Available: http://dx.doi.org/10.1002/9781119202486.ch2

[8] T. Y. Chen and R. Merkel, "An upper bound on software testing effectiveness," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 17, no. 3, p. 16, 2008.

[9] Y. Li, B.-B. Yin, J. Lv, and K.-Y. Cai, "Approach for test profile optimization in dynamic random testing," in *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, vol. 3. IEEE, 2015, pp. 466–471.

[10] H. Liu, X. Xie, J. Yang, Y. Lu, and T. Chen, "Adaptive random testing by exclusion through test profile," in *QSIC 2010*. IEEE, 2010, pp. 147–156.

[11] 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.

[12] K. Cai, H. Hu, C. Jiang, and F. Ye, "Random testing with dynamically updated test profile," in *Proceedings of the 20th International Symposium On Software Reliability Engineering (ISSRE 2009)*, 2009, pp. 1–2.

[13] J. Lv, H. Hu, and K.-Y. Cai, "A sufficient condition for parameters estimation in dynamic random testing," in *Computer Software and Applications Conference Workshops (COMPSACW), 2011 IEEE 35th Annual*. IEEE, 2011, pp. 19–24.

[14] J. Lv, H. Hu, K. Y. Cai, and T. Y. Chen, "Adaptive and random partition software testing," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 12, pp. 1649–1664, Dec 2014.

[15] P. G. Bishop, "The variation of software survival time for different operational input profiles (or why you can wait a long time for a big bug to fail)," pp. 98–107, June 1993.

[16] Y. Li, B.-B. Yin, J. Lv, and K.-Y. Cai, "Approach for test profile optimization in dynamic random testing," in *Computer Software and Applications Conference (COMPSAC), 2015 IEEE 39th Annual*, vol. 3. IEEE, 2015, pp. 466–471.

[17] T. Y. Chen, P. L. Poon, S. F. Tang, and T. H. Tse, "Dessert: a divide-and-conquer methodology for identifying categories, choices, and choice relations for test case generation," *IEEE Transactions on Software Engineering*, vol. 38, no. 4, pp. 794–809, July 2012.

[18] T. Y. Chen, P.-L. Poon, and T. Tse, "A choice relation framework for supporting category-partition test case generation," *IEEE transactions on software engineering*, vol. 29, no. 7, pp. 577–593, 2003.

[19] X. Y. Zhang, D. Towey, T. Y. Chen, Z. Zheng, and K. Y. Cai, "A random and coverage-based approach for fault localization prioritization," pp. 3354–3361, May 2016.

[20] K.-Y. Cai, T. Y. Chen, Y.-C. Li, W.-Y. Ning, and Y.-T. Yu, "Adaptive testing of software components," in *Proceedings of the 2005 ACM symposium on Applied computing*. ACM, 2005, pp. 1463–1469.

[21] R. Huang, H. Liu, X. Xie, and J. Chen, "Enhancing mirror adaptive random testing through dynamic partitioning," vol. 67. Elsevier, 2015, pp. 13–29.

Authorized licensed use limited to: Universita degli Studi di Salerno. Downloaded on June 11,2025 at 18:30:42 UTC from IEEE Xplore. Restrictions apply.