2015

A revisit of three studies related to random testing

Tsong Yueh Chen
Swinburne University of Technology, tchen@ict.swin.edu.au

Fei-Ching Kuo
Swinburne University of Technology, dkuo@uow.edu.au

Dave Towey
United International College

Zhiquan Zhou
University of Wollongong, zhiquan@uow.edu.au

Publication Details

Chen, T., Kuo, F., Towey, D. & Zhou, Z. (2015). A revisit of three studies related to random testing. Science China Information Sciences, 58 (5), 052104:1-052104:9.

A revisit of three studies related to random testing

Abstract
© 2015 Science China Press and Springer-Verlag Berlin Heidelberg Software testing is an approach that ensures the quality of software through execution, with a goal being to reveal failures and other problems as quickly as possible. Test case selection is a fundamental issue in software testing, and has generated a large body of research, especially with regards to the effectiveness of random testing (RT), where test cases are randomly selected from the software's input domain. In this paper, we revisit three of our previous studies. The first study investigated a sufficient condition for partition testing (PT) to outperform RT, and was motivated by various controversial and conflicting results suggesting that sometimes PT performed better than RT, and sometimes the opposite. The second study aimed at enhancing RT itself, and was motivated by the fact that RT continues to be a fundamental and popular testing technique. This second study enhanced RT fault detection effectiveness by making use of the common observation that failure-causing inputs tend to cluster together, and resulted in a new family of RT techniques: adaptive random testing (ART), which is random testing with an even spread of test cases across the input domain. Following the successful use of failure-causing region contiguity insights to develop ART, we conducted a third study on how to make use of other characteristics of failure-causing inputs to develop more effective test case selection strategies. This third study revealed how best to approach testing strategies when certain characteristics of the failure-causing inputs are known, and produced some interesting and important results. In revisiting these three previous studies, we explore their unexpected commonalities, and identify diversity as a key concept underlying their effectiveness. This observation further prompted us to examine whether or not such a concept plays a role in other areas of software testing, and our conclusion is that, yes, diversity appears to be one of the most important concepts in the field of software testing.

Keywords
random, related, studies, testing, three, revisit

Disciplines
Engineering | Science and Technology Studies

Publication Details
Chen, T., Kuo, F., Towey, D. & Zhou, Z. (2015). A revisit of three studies related to random testing. Science China Information Sciences, 58 (5), 052104:1-052104:9.

SCIENCE CHINA
Information Sciences Manuscript

• **RESEARCH PAPER** •

2015, Vol. xxxxxx:1–xxxxxx:9
doi: xxxxxxxxxxxxxx

A Revisit of Three Studies Related to Random Testing

Tsong Yueh Chen[1], Fei-Ching Kuo[1], Dave Towey[2*] & Zhi Quan Zhou[3]

[1]*Department of Computer Science and Software Engineering, Swinburne University of Technology, Victoria 3122, Australia;*
[2]*School of Computer Science, The University of Nottingham Ningbo China, Ningbo 315100, China;*
[3]*School of Computer Science and Software Engineering, University of Wollongong, Wollongong, NSW 2522, Australia.*

Received ; accepted

Abstract Software testing is an approach that ensures the quality of software through execution, with a goal being to reveal failures and other problems as quickly as possible. Test case selection is a fundamental issue in software testing, and has generated a large body of research, especially with regards to the effectiveness of random testing (RT), where test cases are randomly selected from the software's input domain. In this paper, we revisit three of our previous studies. The first study investigated a sufficient condition for partition testing (PT) to outperform RT, and was motivated by various controversial and conflicting results suggesting that sometimes PT performed better than RT, and sometimes the opposite. The second study aimed at enhancing RT itself, and was motivated by the fact that RT continues to be a fundamental and popular testing technique. This second study enhanced RT fault detection effectiveness by making use of the common observation that failure-causing inputs tend to cluster together, and resulted in a new family of RT techniques: adaptive random testing (ART), which is random testing with an even spread of test cases across the input domain. Following the successful use of failure-causing region contiguity insights to develop ART, we conducted a third study on how to make use of other characteristics of failure-causing inputs to develop more effective test case selection strategies. This third study revealed how best to approach testing strategies when certain characteristics of the failure-causing inputs are known, and produced some interesting and important results. In revisiting these three previous studies, we explore their unexpected commonalities, and identify diversity as a key concept underlying their effectiveness. This observation further prompted us to examine whether or not such a concept plays a role in other areas of software testing, and our conclusion is that, yes, diversity appears to be one of the most important concepts in the field of software testing.

Keywords Adaptive random testing, diversity, metamorphic testing, proportional sampling strategy, random testing, software testing.

Citation T.Y. Chen, F.-C. Kuo, D. Towey, and Z.Q. Zhou. A Revisit of Three Studies Related to Random Testing. Sci China Inf Sci, 2015, : xxxxxx(9), doi: xxxxxxxxxxxxxx

1 Software Testing

Software testing involves execution of software with the intention of finding problems before releasing it for use: it is a form of quality assurance. It has been argued that software testing is among the most

* Corresponding author (email: Dave.Towey@nottingham.edu.cn)

mature areas of software development [3], and, given the current pervasion of software (and the desire for high quality in this software), the need for good testing has never been clearer.

When testing, the combination of input parameters required by the software for a single execution is referred to as a test case, and a test case that uncovers a problem is referred to as failure-causing. The complete set of possible test cases is the input domain for the software being tested, with each individual test case being a single element of this input domain. This set is normally huge, and usually only a very small proportion is ever tested. The ratio of failure-causing inputs to all possible inputs of the input domain is referred to as the failure rate. We are particularly interested in the effectiveness of the testing strategies for situations where the failure rate is small, because a small number of test cases, irrespective of how they are selected, should be able to reveal failure for a program with large failure rates.

Since the checking of all possible inputs is most often prohibitively difficult or expensive, it is essential for testers to make best use of their limited testing resources. To do this, various test case selection methods have evolved, based on different intuitions. For example, the intuition behind coverage testing is that if a program entity has not been executed, then there is no way to detect any fault associated with that entity. Thus, the desire to ensure execution of each program statement, branch, or path has led to metrics that inform the extent of statement, branch, and path coverage. With these metrics in place, test cases resulting in execution of the relevant code can be selected. Another example is partition testing — a family of testing methods which involve division of the input domain into multiple, disjoint partitions, and then selection of test cases from each partition. The rationale behind partition testing is to group inputs that are related to the same function or feature into the same partition, which then allows the testing of that particular function or feature to be achieved by selecting some representative test cases from the relevant partition. The ideal situation is to have all partitions homogeneous, that is, elements of the same partition are either all failure-causing, or all non-failure-causing inputs. In such a scenario, there is no need to test a partition with more than one of its elements. Unfortunately, no algorithm exists for constructing homogeneous partitions — other than the trivial, but practically useless, set-up of all single-input partitions.

As explained above, the intuitions behind coverage testing and partition testing are quite different. In fact, there is a great deal of variety in the motivations based on which various test case selection methods have been developed. An interesting question therefore is: is there any more fundamental, underlying intuition or attribute common to all testing methods? This paper revisits three of our previous software testing studies, examining some interesting and unexpected similarities among all three. Through reflection on these studies, we believe we have identified a key characteristic of the successful testing strategies: Diversity.

The next section presents the three software testing studies that are all related to random testing, our reflections on them, and our interpretation of the role of diversity in the successful testing. Section 3 expands the diversity discussion to another successful area of software testing research, Metamorphic Testing [7]. The power of diversity in software testing is discussed in Section 4, where some recent, surprising Metamorphic Testing results, and random testing's effectiveness, are also examined further. Section 5 concludes the paper.

2 A Revisit of Three Separate Studies

A fundamental, yet important and effective, testing strategy is random testing (RT), which requires that test cases (inputs) be selected randomly and independently from the input domain. A uniform test profile is normally assumed, in which every element of the input domain has the same probability of being selected as a test case. RT can be used alone or applied with other testing methods, and due to its conceptual simplicity and efficiency for test case generation, it has been a commonly-used testing method. It has been successfully applied in testing Java and .Net libraries [30], the OpenSSL library [19], mission-critical flight software [21], database systems [4], real-time embedded systems [2], and interrupt-driven embedded systems [32], to name a few.

The three separate studies on which we based our reflections are all related to RT, and include the

development of a sufficient condition for partition testing to perform equally or better than random testing [5,14–17]; an investigation into how to enhance the effectiveness of random testing [10,11]; and an examination of the upper bound of testing effectiveness, using random testing as a reference point [12].

2.1 A Sufficient Condition for Partition Testing to Outperform Random Testing

Random testing (RT) has often been compared to partition testing (PT), which involves division of the input domain into disjoint subsets, namely partitions, and selection of test cases from each partition. Since division of the input domain inevitably incurs overheads, a natural question is whether or not these overhead costs are justified in terms of bringing increased failure-finding effectiveness. This question has been investigated and debated extensively, with some apparently contradictory empirical results obtained.

In view of the conflicting empirical results, it appeared that no conclusive answer would be found through empirical study, and therefore theoretical analysis was identified as a possible alternative. Theoretical analysis, though often involving certain assumptions (sometimes unrealistic ones), has yielded some definite conclusions. One of the first such analyses was conducted by Weyuker and Jeng [36], who defined a partition testing strategy in which the input domain was divided into equally-sized partitions, with the same number of test cases being randomly selected, with replacement, from each of these partitions. This strategy is referred to as the Equal-Size-Equal-Sampling Strategy (ESESS), and it has been proven, under the assumption that every input is equally likely to be failure-causing, that if the same total number of test cases are used for both RT and the ESESS, then the probability of detecting at least one failure (referred to as the P-measure) for ESENS is not less than that of RT.

This was the first theoretically proven result for PT to perform at least as well as RT, and although its applicability was restricted to situations with equally-sized partitions, which may not be practically feasible, this result did motivate Chen and Yu to seek a more general approach, which they defined as the Proportional Sampling Strategy (PSS) [15]. The PSS states that all partitions should have the same ratio of the number of randomly selected test cases, with replacement, from a partition to the size of the relevant partitions — in other words, the PSS recommends that test cases should be randomly drawn from different partitions in proportion to the partition sizes. This strategy has been proven to perform at least as well as RT in that, for the same number of test cases, PSS is guaranteed to have an equal or higher P-measure — for any possible partitioning of the input domain, and for any program. In fact, PSS and RT have the same P-measure only if all partitions have equal failure rates: PSS has a higher P-measure than RT as long as some partitions have different failure rates. Because partitions are very unlikely to have equal failure rates, PSS more frequently performs better, rather than equally to, RT. Obviously, ESESS is a special case of PSS. It was also subsequently proven that PSS is not only sufficient, but is actually necessary, to ensure an equal or higher P-measure than that of RT [17].

A problem when applying ESESS in practice is that equally-sized partitions may not always be obtainable. Although PSS does not have this constraint of equal sizes for all partitions, it faces another problem — that it may not be practically feasible to have strictly proportional sampling, because resource limitations may restrict the number of test cases that could be executed. Therefore, some guidelines haven been proposed to address the situation where PSS cannot be strictly applied [14]. A new concept of General Proportional Sampling Strategy (GPSS) has also been proposed to address the problem that it may not be feasible to apply PSS on some occasions [14]. Suppose that there are k partitions (with sizes denoted by d_i, where $1 \leqslant i \leqslant k$), from which a total of n test cases are to be selected, with n_i test cases from the i^{th} partition. The test case distribution $(n_1, n_2, \ldots, n_k)$ is said to satisfy GPSS, if for any i and j such that $n_i/d_i < n_j/d_j$, then we have $(n_i + 1)/d_i > (n_j - 1)/d_j$. Obviously, PSS is a special case of GPSS. Furthermore, GPSS is always feasible, but this is not the case with PSS. More importantly, if the partitions do not have the same failure rates, then for any n, there always exists a test case distribution $(n_1, n_2, \ldots, n_k)$ satisfying GPSS and yielding a higher P-measure than RT.

In summary, in the absence of any details of the program under test, or information about the partitions (other than their sizes), it is recommended that PSS be applied if possible. As a reminder, the only assumption for PSS is that every input has the same chance of being failure-causing; there are no other constraints on its applicability.

2.2 Adaptive Random Testing

The second revisited study examined an enhancement to random testing which was inspired by the empirical observation that failure-causing inputs tend to form contiguous regions, and consequently, non-failure-causing inputs also form contiguous regions. An obvious inference, therefore, was that for any program in which no failure is detected by particular test cases, then a *good* next test case should be one further away from the previously executed ones. Based on this intuition, the approach of Adaptive Random Testing (ART) [11] was developed, which aims at enhancing random testing through the use of a more even- and well-spread distribution of random test cases throughout the input domain.

Several different principles have been used to achieve an even spread of random test cases across the input domain, including:

1. Selection: Instead of using the next randomly generated input as the next test case, a set of random inputs is generated as the candidate test cases, with the best candidate (against a selection criterion) then chosen as the next test case.

2. Exclusion: An exclusion region can be defined around each already executed test case, and when any randomly generated input falls into such a region, this input is discarded and another input randomly generated. If the input is outside the exclusion regions, then it is used as the next test case.

3. Dynamic adjustment of test profiles: Initially, the random test case generation is according to the uniform test profile. After each randomly generated test case is executed, the test profile is then adjusted with the aim of achieving an even spread of test cases across the input domain.

4. Partitioning: Already executed test cases are used to divide the input domain into partitions from which a specific partition is selected (according to a criterion), and the next test case is randomly selected from this designated partition instead of from the entire input domain.

Due to the variety of even-spreading principles, various ART algorithms have been developed [6, 9, 11, 13, 18, 25, 28, 33–35]. Although they are based on different principles to achieve an even spreading of test cases across the input domain, all of these algorithms have been empirically demonstrated to have a smaller F-measure than RT — the effectiveness metric F-measure is defined as the expected number of test cases required to detect the first failure. Furthermore, these ART algorithms also outperform RT with respect to the P-measure for the same number of test cases used [8, 33].

Obviously, compared with RT, ART algorithms incur additional computational overheads due to the even spreading of the random test cases across the input domain, and different ART algorithms have various orders of complexity for the test case generation, ranging from linear to quadratic order. Intuitively speaking, a higher order algorithm should achieve a more even spread of test cases, and hence a better failure detection effectiveness; but in practice, this is sometimes not the case. Similar to sorting, where different sorting algorithms (based on different intuitions) have various favourable and unfavourable conditions, different ART algorithms (also based on different principles) also have various favourable and unfavourable conditions for their application. For example, consider the first, and most commonly referred to, ART algorithm, the Fixed-Size-Candidate-Set ART (FSCS-ART), which uses the maxi-min criterion for candidate selection. In each round of test case generation (except for the first test case), a constant number of inputs are randomly generated as candidates for the next test case. The candidate with the largest distance from the nearest element amongst all already executed test cases is then selected as the next test case. Because of an initial concentration of test cases around the input domain boundary, rather than an even spread across the entire input domain, FSCS-ART performs poorly with higher failure rates. However, as more test cases are generated, an even spread is gradually and steadily achieved. In other words, FSCS-ART has a better F-measure than RT for smaller failure rates. However, ART's computational overheads (due to even spreading) grow quadratically, and so for smaller failure rates, FSCS-ART may be less cost-effective than RT. Therefore, a challenging question is: given a program,

how can a tester choose an appropriate ART algorithm, and best conduct ART in a cost-effective way. Some approaches towards solutions for this problem are proposed and discussed in [1].

The investigations of ART have not only delivered enhancements to RT, but also given rise to some new concepts, including adaptive random sequences and failure-based testing. Interested readers may refer to a survey and synthesis of ART research [10].

2.3 Theoretical Analysis of an Optimal Test Case Selection Strategy

The third study was an investigation of the upper bound of testing effectiveness: a theoretical analysis of possible improvements over the failure-finding effectiveness of random testing [12]. In terms of the number of test cases required to find a failure in a program being tested, the study revealed that, short of location details, any extra information about the failure-causing regions (e.g. shape, orientation, and size) can only reduce the number of test cases by a maximum factor of two: no testing strategy will use less than half the number of test cases required by random testing to find the first failure! The proof of this theoretical bound is based on the design of an optimal test case selection strategy that essentially defines a grid according to the patterns formed by the failure-causing inputs in the input domain, and then uses this grid to guide test case selection so as to guarantee that at least one test case will be drawn from the failure-causing regions, regardless of where in the input domain these regions are. In other words, at least one failure-causing input is guaranteed to be selected as a test case. As an analogy, imagine that all fish in the ocean are of the same size and shape: with the actual details of the fish size and shape, we could design an *optimal* net (in the sense of using less materials) guaranteed to catch them. The optimal test case selection strategy is designed along the same logic as the design of this net.

This study was the first investigation to analyse the effectiveness of test case selection strategies by simply assuming some prior information about the failure-causing inputs, without requiring any knowledge about the details of selection strategy methods or procedures. A significant benefit of such an approach is that we can evaluate the effectiveness of a class of test case selection strategies that make use of the same kind of information, even if their exact methods or procedures are not yet known.

Since ART uses much less information than the optimal test case selection strategy to generate test cases, intuitively speaking, it should have a higher F-measure. However, because of the experimental data showing that ART has, on average, about half the F-measure of RT (i.e., reduces the number of test cases by a factor of about two), we can conclude that the theoretical maximum improvement of two is a very tight bound, and that the performance of ART is in fact very close to that of the optimal test case selection strategy. As a consequence, this study implies that more effort should be invested in improving ART efficiency (e.g., reduction of the computational overheads related to the even spreading of test cases) rather than effectiveness (by improvement in the degree of even spreading). Another key implication is that the use of information regarding the location of failure-causing inputs is a very promising direction for future research. Obviously, exact information about the location of failure-causing inputs is not going to be available, but nevertheless, we can make use of some partial information, such as which parts of the input domain are more likely to have failure-causing inputs. In fact, this result further supports what has already been proposed by Hamlet and Taylor [23], and Morasca and Serra-Capizzano [29].

2.4 The Underlying Commonality: Diversity

The three studies described above were differently motivated, have different assumptions, and each produced different test case selection strategies. However, in spite of the fundamental and significant differences among them, surprisingly, there is a commonality: their selected test cases all turn out to be evenly spread across the input domain — that is, they all exhibit a kind of *spatial diversity* across the input domain! This striking commonality emphasises the intuition that there must be a more fundamental concept underlying all three of these testing techniques — PSS, ART and the optimal test case selection strategy. Motivated by this, a further investigation and analysis of these three test techniques has been conducted revealing some interesting observations. In PSS, the numbers of randomly selected test cases belonging to a partition are proportional to the relevant partition size. Thus, PSS effectively imposes

an even spread of random test cases across the entire input domain, and hence can be regarded as ART through the partitioning of the input domain. Obviously, the more partitions there are, the more even the spreading of test cases is. Also observed is that ART can be regarded as imitating the optimal strategy, which effectively defines a grid according to the patterns formed by failure-causing inputs to guide selection of test cases — in the absence of information about the failure-causing regions, an even spread of random test cases across the input domain is an intuitively appealing and straightforward approach to implementing such a grid. The various and independent empirical evaluations of the F-measure for ART, as well as the theoretically derived F-measure for the optimal test case selection strategy, collectively show the closeness of their performance, which means that ART and the optimal test case selection strategy are intrinsically the same — at least from the perspective of their failure detection effectiveness.

An important question naturally emerges: is the above commonality amongst these three studies just a coincidence or an isolated case? A further reflection shows that the answer is "no." In fact, in most test case selection approaches, regardless of the stated or implicitly implied intention, the actual process involves selection of a somehow *diverse* set of test cases! Obviously, the meaning of diverse varies from approach to approach: in partition testing, as explained in Section 1, partitions are normally designed in such a way that inputs from the same partition are related to the same function or feature. Partition testing aims at selecting test cases from as many different partitions as possible, rather than test cases from the same partition. Thus, partition testing incorporates diversity in the sense that more distinct partitions — and therefore more distinct functions or features — are tested, thus achieving diversity across functions or features. In coverage testing, the basic intuition is that if a program entity is not tested, then its associated fault (if it exists) has no chance of being revealed: coverage testing gives preference to testing program entities not yet executed — in other words, coverage testing also implicitly incorporates diversity, but does so across the program entity.

In summary, diversity appears to be an underlying concept in successful test case selection methods.

3 Diverse Diversity

The studies mentioned in the previous section show the importance of diversity, which has been either explicitly included or implicitly implied within a test case selection strategy. In this section, we would like to highlight another aspect of diversity in other areas of software testing.

A test oracle is a mechanism that can verify the correctness of the output for any input to a program. The oracle problem occurs when either the test oracle does not exist, or it is not practically feasible to use it. This is a difficult but frequently encountered problem in software testing. Metamorphic Testing (MT) is one of the many approaches that have been proposed to alleviate the oracle problem [24, 27]. Basically speaking, MT involves multiple program executions, with the inputs selected such that they and their outputs satisfy certain relations — the relationships between the inputs and their outputs are known as metamorphic relations, some necessary properties of the algorithm being implemented. The main idea behind MT is that although we may not be able to verify the correctness of an output, it may be possible to know the relationships between some inputs and their related outputs. For example, for the algorithm sine, we know that a possible metamorphic relation is $sin(x) = sin(x + 360)$, where x is in the unit of degrees. So, if $x = 29$ is somehow selected as a test case, then a new test case 389 (referred to as a follow-up test case) can be constructed from the metamorphic relation with reference to the original test case of 29. If the outputs do not comply with the expected relations, we can conclude that the program has some fault or mistake — however, due to an intrinsic limitation of testing, compliance cannot be interpreted as meaning that the program is correct: testing can show the presence of bugs, not their absence [20, p.16].

In two independent studies involving the application of MT [31,37], faults were found in three programs of the popular Siemens suite, namely, *print_token*, *schedule* and *schedule_2*. The Siemens suite has long been used by the testing community for benchmarking testing strategies, and hence programs in this suite have been extensively tested by various test case selection methods. Therefore, it is quite surprising, at first glance, that MT is able to detect some previously unknown faults. We want to emphasise that this

is not to say that MT is necessarily better than existing test case selection techniques, but rather that testing should be performed from *diverse* perspectives — MT was successful in revealing further faults because it is based on a perspective not previously used by the other testing techniques (testing some necessary properties involving multiple inputs for the program under test). The detection of these faults in the extensively tested Siemens suite is strong evidence that diversity is an underlying concept, not only for individual test case selection strategies, but also for a comprehensive and thorough testing. It is well known that a single test case selection strategy is not sufficiently powerful to reveal all faults — otherwise there would be one, and only one, strategy. As far as we know, no guidelines currently exist for how to choose an appropriate combination of strategies for thorough testing, but in light of the faults recently detected in the Siemens suite by MT, we believe that diversity should be a key factor in determining such a group of strategies. As a reminder, in addition to the above mentioned real-life faults, MT also revealed real-life faults for other extensively tested and often used open software [27].

4 Discussion

In this paper, we have argued the importance of the presence of diversity in test case selection and in determining a group of test case selection methods to support a comprehensive and thorough testing. We would like to share the result of a recent study [27] which illustrates how important a role diversity can also play in other areas of software testing.

As discussed in Section 3, MT has been proposed to alleviate the oracle problem. A study [27] was conducted recently to determine how effectively metamorphic relations (MRs) might actually replace an oracle in testing. One of the key findings in this study was that a small number of diverse MRs could have a similar fault-detection capability to a test oracle, and could therefore be used as an effective substitute. Furthermore, this oracle-like fault-detection capability, realized through the diversity of the MRs used, could be achieved by groups of relatively inexperienced testers working in an ad-hoc manner — resembling the Best Buy case [22], where vice-president Jeff Severts found that a diverse group of several hundred employees were ten times better able to predict sales than the company's expert forecasters [26]. Metamorphic testing involves more than just the generation of new (follow-up) test cases — such as the follow-up test case of 389 with respect to the original test case of 29 and the metamorphic relation $sin(x) = sin(x + 360)$, as explained in Section 3 — it also includes examination of properties of the algorithm under test, and enables groups of testers, of varying degrees of experience and expertise to collaborate in the testing. Thus, it is interesting to see how diversity also plays a key role in guiding the selection of metamorphic relations to enhance MT's cost-effectiveness, and to serve as a *virtual* oracle in the absence of an actual one.

Since RT does not make use of any information to guide test case selection, there has long been debate in the literature, with many studies conducted, to determine whether or not RT is a cost-effective method. These studies have somehow yielded controversial results, with some reporting RT to be cost-effective, but others not. As proven in [12], unless some kind of the information about the location of failure-causing inputs is available and applied, there is no possible way to use less than half the number of test cases required by RT to detect the first failure. In other words, there is only a difference factor of two between the effectiveness performance for RT and the optimal test case selection strategy. It should also be noted that the optimal strategy is assumed to make use of information of the failure-causing inputs other than their locations, such as, their sizes, shapes, orientations, and distributions. However, such information is normally not available prior to testing, and hence the optimal strategy is simply a theoretical ideal: actual, implemented strategies should be less effective. In other words, taking into account its efficient generation of test cases, from a theoretical perspective, RT is in fact a cost-effective test case selection strategy. It is further worthwhile noting that RT also exhibits some kind of diversity through its randomness. Thus, we believe that RT is not too far from the optimal strategy because of its inherent diversity.

So, in summary, all these investigations strongly identify diversity as a fundamental, underlying intuition in successful software testing. A possible reason for this may be related to the diversity possible in

human error, which may also help explain why the optimal strategy is only twice as efficient as RT at finding the first failure — because it needs to take account of all possible diversity of errors.

5 Conclusion

Some people may regard the role of diversity as overstated, or as little more than common sense — expected, taken for granted, and not really worth talking about or investigating seriously. In contrast, this paper has presented strong objective evidence to support its importance. The strengths gained from diversity — as seen in so many fields and disciplines — should mean that having diversity as a guiding principle in software testing is very philosophically appealing, in view of the fact that programmers can make many kinds of errors. Such a guiding principle may enable a shift in how some software testing research has been viewed, encouraging new insights and perspectives, potentially leading to better and stronger approaches. An example of this is the inherent ability of RT to generate a relatively diverse set of test cases due to randomness, but at a low cost. This reasonably easily obtained diversity is simple, yet effective (as proven by theoretical analysis [12]), and may serve to ensure that random testing will never be obsolete. Indeed, enhanced versions of random testing, such as adaptive random testing, should draw increased research attention and merit further development. Furthermore, ART involves a simple form of diversity — arguably *the* simplest form — and we suggest that other forms of diversity should be considered in the design of new testing methods. In fact, although ART's *spatial* diversity was inspired by the common occurrence of failure-causing input contiguity — and this spatial diversity remains an integral, defining characteristic of ART — ART itself has inspired other kinds of diversity-based testing approaches and research.

Given the diversity of human error, the future of successful software testing, as with so much else, may well lie in increased recognition of the importance of diversity. Recently, there has been a promotion for the development of a general theory for software engineering. We fully agree that a general theory for software engineering will enhance its significance and impact. If a theory of software testing is ever to be developed, then diversity shall certainly form a part of its foundation, and may well play a role similar in importance to that of gravity in physics!

Acknowledgment

The authors would like to acknowledge the support of a linkage grant from the Australian Research Council (project no. ARC LP100200208).

References

1 S. Anand, E. Burke, T. Y. Chen, J. Clark, M. B. Cohen, W. Grieskamp, M. Harman, M. J. Harrold, and P. McMinn. An orchestrated survey on automated software test case generation. *Journal of Systems and Software*, 86(8):1978–2001, Aug. 2013.

2 A. Arcuri, M. Z. Iqbal, and L. Briand. Black-box system testing of real-time embedded systems using random and search-based testing. In *Proceedings of the 22nd IFIP WG 6.1 International Conference on Testing Software and Systems*, ICTSS'10, pages 95–110, Berlin, Heidelberg, 2010. Springer-Verlag.

3 P. G. Armour. Not-defect, the mature discipline of testing. *Communications of the ACM*, 47(10):15, Oct 2004.

4 H. Bati, L. Giakoumakis, S. Herbert, and A. Surna. A genetic approach for random testing of database systems. In C. Koch, J. Gehrke, M. N. Garofalakis, D. Srivastava, K. Aberer, A. Deshpande, D. Florescu, C. Y. Chan, V. Ganti, C. Kanne, W. Klas, and E. J. Neuhold, editors, *Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, September 23-27, 2007*, pages 1243–1251. ACM, 2007.

5 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*, 28(12):775–782, 1996.

6 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*, 16(4):553–584, 2006.

7 T. Y. Chen, S. C. Cheung, and S. M. Yiu. Metamorphic testing: A new approach for generating next test cases. Technical report, hkust-cs98-01, Hong Kong University of Science and Technology, 1998.

8 T. Y. Chen, F.-C. Kuo, and R. G. Merkel. On the statistical properties of testing effectiveness measures. *Journal of Systems and Software*, 79(5):591–601, 2006.

9 T. Y. Chen, F.-C. Kuo, R. G. Merkel, and S. P. Ng. Mirror adaptive random testing. *Information and Software Technology*, 46(15):1001–1010, 2004.

10 T. Y. Chen, F.-C. Kuo, R. G. Merkel, and T. H. Tse. Adaptive random testing: The ART of test case diversity. *Journal of Systems and Software*, 83(1):60–66, Jan 2010.

11 T. Y. Chen, H. Leung, and I. K. Mak. Adaptive random testing. In M. J. Maher, editor, *Advances in Computer Science – ASIAN 2004. Higher-Level Decision Making*, volume 3321 of *Lecture Notes in Computer Science*, pages 320–329. Springer Berlin Heidelberg, 2004.

12 T. Y. Chen and R. Merkel. An upper bound on software testing effectiveness. *ACM Transactions on Software Engineering and Methodology*, 17(3):1–27, Jun 2008.

13 T. Y. Chen, R. Merkel, G. Eddy, and P. K. Wong. Adaptive random testing through dynamic partitioning. In *Proceedings of the Quality Software, Fourth International Conference*, QSIC '04, pages 79–86, Washington, DC, USA, 2004. IEEE Computer Society.

14 T. Y. Chen, T. H. Tse, and Y. T. Yu. Proportional sampling strategy: A compendium and some insights. *Journal of Systems and Software*, 58(1):65–81, 2001.

15 T. Y. Chen and Y. T. Yu. On the relationship between partition and random testing. *IEEE Transactions on Software Engineering*, 20(12):977–980, 1994.

16 T. Y. Chen and Y. T. Yu. On the expected number of failures detected by subdomain testing and random testing. *IEEE Transactions on Software Engineering*, 22(2):109–119, 1996.

17 T. Y. Chen and Y. T. Yu. The universal safeness of test allocation strategies for partition testing. *Information Sciences*, 129(1-4):105–118, 2000.

18 I. Ciupa, A. Leitner, M. Oriol, and B. Meyer. ARTOO: Adaptive random testing for object-oriented software. In *Proceedings of the 30th International Conference on Software Engineering (ICSE'08)*, pages 71–80, 2008.

19 Codenomicon. The heartbleed bug, 2014. Available from `http://heartbleed.com/` (Accessed 11 August 2014).

20 E. W. Dijkstra. In *Software Engineering Techniques, Report on a conference sponsored by the NATO Science Committee, Rome, Italy, 27th to 31st October 1969*, 1969.

21 A. Groce, G. J. Holzmann, and R. Joshi. Randomized differential testing as a prelude to formal verification. In *ICSE*, pages 621–631. IEEE Computer Society, 2007.

22 G. Hamel. *The Future of Management*. Harvard Business School Press, 2007.

23 D. Hamlet and R. Taylor. Partition testing does not inspire confidence. *IEEE Transactions on Software Engineering*, 16(12):1402–1411, 1990.

24 M. Harman, P. McMinn, M. Shahbaz, and S. Yoo. A comprehensive survey of trends in oracles for software testing. Technical report, cs-13-01, University of Sheffield, 2013.

25 H. Hemmati, A. Arcuri, and L. C. Briand. Achieving scalable model-based testing through test case diversity. *ACM Transactions on Software Engineering Methodology*, 22(1):6, 2013.

26 A. Kingl. Diversity helps to discern: why cross-company dialogue delivers (part 2). *South China Morning Post: Education Post (March 6th)*, 2013.

27 H. Liu, F.-C. Kuo, D. Towey, and T. Y. Chen. How effectively does metamorphic testing alleviate the oracle problem? *IEEE Transactions on Software Engineering*, 40(1):4–22, 2014.

28 J. Mayer. Lattice-based adaptive random testing. In *Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering*, ASE '05, pages 333–336, New York, NY, USA, 2005. ACM.

29 S. Morasca and S. Serra-Capizzano. On the analytical comparison of testing techniques. In *Proceedings of the 2004 ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA '04, pages 154–164, New York, NY, USA, 2004. ACM.

30 C. Pacheco, S. K. Lahiri, M. D. Ernst, and T. Ball. Feedback-directed random test generation. In *Proceedings of the 29th International Conference on Software Engineering*, ICSE '07, pages 75–84, Washington, DC, USA, 2007. IEEE Computer Society.

31 P. Rao, Z. Zheng, T. Y. Chen, N. Wang, and K. Cai. Impacts of test suite's class imbalance on spectrum-based fault localization techniques. In *The Symposium on Engineering Test Harness (TSETH '13) co-located with the 13th International Conference on Quality Software (QSIC '13)*, pages 260–267, 2013.

32 J. Regehr. Random testing of interrupt-driven software. In W. Wolf, editor, *EMSOFT 2005, September 18-22, 2005, Jersey City, NJ, USA, 5th ACM International Conference On Embedded Software, Proceedings*, pages 290–298. ACM, 2005.

33 A. Shahbazi, A. F. Tappenden, and J. Miller. Centroidal voronoi tessellations - a new approach to random testing. *IEEE Transactions on Software Engineering*, 39(2):163–183, 2013.

34 A. F. Tappenden and J. Miller. A novel evolutionary approach for adaptive random testing. *IEEE Transactions on Reliability*, 58(4):619–633, 2009.

35 A. F. Tappenden and J. Miller. Automated cookie collection testing. *ACM Transactions on Software Engineering Methodology*, 23(1):3, 2014.

36 E. J. Weyuker and B. Jeng. Analyzing partition testing strategies. *IEEE Transactions on Software Engineering*, 17(7):703–711, 1991.

37 X. Xie, W. E. Wong, T. Y. Chen, and B. Xu. Metamorphic slice: An application in spectrum-based fault localization. *Information and Software Technology*, 55(5):866–879, May 2013.