An Empirical Analysis and Comparison of Random Testing Techniques

Johannes Mayer
Ulm University
Dept. of Applied Information Processing
89069 Ulm, Germany
johannes.mayer@uni-ulm.de

Christoph Schneckenburger
Ulm University
Dept. of Applied Information Processing
89069 Ulm, Germany
christoph.schneckenburger@uni-ulm.de

ABSTRACT

Testing with randomly generated test inputs, namely Random Testing, is a strategy that has been applied succefully in a lot of cases. Recently, some new adaptive approaches to the random generation of test cases have been proposed. Whereas there are many comparisons of Random Testing with Partition Testing, a systematic comparison of random testing techniques is still missing. This paper presents an empirical analysis and comparison of all random testing techniques from the field of Adaptive Random Testing (ART). The ART algorithms are compared for effectiveness using the mean F-measure, obtained through simulation and mutation analysis, and the P-measure. An interesting connection between the testing effectiveness measures F-measure and P-measure is described. The spatial distribution of test cases is determined to explain the behavior of the methods and identify possible shortcomings. Besides this, both the theoretical asymptotic runtime and the empirical runtime for each method are given.

Categories and Subject Descriptors

D.2.5 [**Software Engineering**]: Testing and Debugging—*Testing tools*

General Terms

Verification

Keywords

Random Testing, Adaptive Random Testing, Testing effectiveness, F-measure, P-measure, Runtime

1. INTRODUCTION

Software testing [37], i. e. the systematic execution of the software under test with certain test cases, is an important part of software quality assurance. Random Testing (RT) [24] is a common strategy which randomly generates test inputs. This method is simple, unbiased (i. e. not influenced by wrong assumptions made by human testers), and efficient. It has successfully been applied e. g. to test data base systems [42] and Java Just-In-Time (JIT) compilers [44], as well as to assess the robustness of Windows NT applications [22]. However, it has been criticized that Random Testing does not take any information about the system under test (SUT) into account [37]. Chan et al. [3] pointed out that failure-causing inputs tend to be clustered within the input domain—they even roughly classified failure patterns (i. e. patterns of failure causing inputs within the input domain) into the categories block, strip, and point. Based on this observation, Chen et al. [14, 17, 29] introduced Adaptive Random Testing (ART) which is designed to evenly spread test cases, because two nearby test cases have a high probability of either detecting no failure or detecting the same failure (pattern). Since then, a lot of techniques have been used to implement the (informal) notion of evenly spread test cases [4, 5, 6, 7, 8, 9, 12, 13, 15, 16, 32, 30, 31]. Each of these approaches has its own strengths and weaknesses, regarding runtime and testing effectiveness. But, whereas there are a lot of comparisons of Partition Testing and Random Testing [3, 18, 19, 23, 26, 28, 43], only two papers comparing random testing methods have been published so far [10, 11]. These papers only compare Random Testing with two "standard" ART methods (D-ART and RRT), mainly using the density function of one testing effectiveness measure, the F-measure.[1] Furthermore, Chan et al. [3] found out, that the effectiveness of partition testing strategies does not only depend on the failure rate, but also on the geometric shape of the failure pattern. Studies on single ART methods have shown that this applies as well to ART methods, whereas the shape of the failure pattern has obviously no influence on the performance of Random Testing.

The present paper describes some empirical studies that have been conducted to analyze and compare all ART methods published so far regarding testing effectiveness and runtime. These results lead to new insights into the respective methods and help to compare the methods among each other. We investigate the mean F-measure using simulations and mutation analysis, the distribution of the F-measure, the P-measure for various test set sizes, the spatial distribution of individual test cases, and the runtime. Thereby, we can compare the individual random testing techniques regarding different aspects and give hints for the improvement of these methods. Our study is unique in its broad range,

[1]The latter paper also uses the so-called P-measure.

ISESE'06, September 21–22, 2006, Rio de Janeiro, Brazil.

since mostly comparison is based on the mean F-measure, obtained through simulation and partly through mutation analysis (but with only a couple of manually generated mutants). Furthermore, there is no comparison yet taking all or nearly all ART techniques into account.

Section 2 briefly introduces the necessary preliminaries, including testing effectiveness measures and presents an interesting relationship between both the F-measure and the P-measure. After a short description of the investigated ART methods in Section 3, the results of the empiricial studies are described in Section 4 followed by a discussion. Section 5 concludes the present paper and gives perspectives for future research.

2. PRELIMINARIES

2.1 Notation

An input is said to be *failure-causing* if the program under test fails for this input. The *failure rate* θ denotes the percentage of failure causing inputs within the input domain. The geometric shape of the failure-causing inputs within the input domain is said to be the *failure pattern*.

2.2 Testing Effectiveness Measures

The *P-measure* and the *E-measure* are most commonly used to measure testing effectiveness. Both measures relate to a fixed test set. The *P-measure* denotes the probability of a given test set to detect a failure. The *E-measure* denotes the average number of failures detected by the test set. These metrics have been used in a number of publications to compare Random Testing with Partition Testing [19, 23, 26, 28, 36, 43].

The P-measure and the E-measure, however, are not that appropriate to compare "dynamic" testing strategies, i. e. strategies that select test cases one by one until a stopping criterion is fulfilled. Therefore, Chen et al. [14] have introduced the *F-measure*. The *F-measure* denotes the number of test cases necessary to detect the first failure and, therefore, is more appropriate for dynamic testing techniques. Moreover, it is also quite intuitive, since testing is often stopped when the first failure has been revealed. Chen et al. [10, 11] determined the theoretical mean F-measure for Random Testing with replacement[2], and also the F-measure distribution, which is geometric. Thus, if the failure rate is θ, the theoretical mean F-measure of Random Testing is $1/\theta$ and the probability that the F-measure is exactly i is $\mathbb{P}(F = i) = (1-\theta)^{i-1}\theta$. Since the F-measure depends on the failure rate, the *relative F-measure* relates the F-measure to the theoretical mean F-measure of Random Testing. This relative metric is ideal to compare the effectiveness of a "dynamic" testing method with that of Random Testing—independent of the (in general unknown) failure rate.

There is an interesting relationship between the F-measure and the P-measure not discovered previously: The value of the cumulative distribution function (cdf) of the F-measure at position f is equal to the probability that the first failure is detected with at most f test cases. This probability is the P-measure for test sets of size f. Therefore, the cdf of the F-measure can be interpreted as the function of the P-measure, where the variable of this function denotes the varying size of the test set.

[2]i. e. test cases can be selected several times

3. ADAPTIVE RANDOM TESTING TECHNIQUES

In the following, the investigated Adaptive Random Testing (ART) methods will be described briefly and their theoretically asymptotic runtime will be given.[3] For each of the following methods we assume that it terminates as soon as a failure is detected (or resources are exhausted). For more details on the presented testing techniques please consult the respective cited publications.

The first ART algorithm was Distance-Based ART (D-ART) [14, 17, 29] with fixed size candidate set. This algorithm randomly selects the first test case. For each further test case, a set of k randomly selected candidates is generated and that candidate which maximizes the minimum (Euclidean) distance to all previously executed test cases is chosen as the next test case. Chen et al. recommended a candidate set size of $k = 10$. The runtime of this algorithm is obviously quadratic in the F-measure, since the distance between each candidate and all previously executed test cases has to be computed.

Restricted Random Testing (RRT) [5] is similar to D-ART. For the selection of a test case, circular exclusion zones are located around each previously executed test case. Candidates are randomly generated one-by-one and discarded, if they are contained within an exclusion zone. The first candidate not within any exclusion zone is selected as the next test case and executed. A coverage ratio $R = 1.5$ (i. e. the sum of the areas of all exclusion zones related to the area of the input domain—disregarding overlapping) has been recommended. Therefore, the size of an individual exclusion zone is shrinking for an increasing number of exclusion zones. Under the assumption of a logarithmic number of candidates (in the number of previously executed test cases), which will be investigated in Section 4.4, the runtime of this algorithm is in $\mathcal{O}(F^2 \log F)$, where F denotes the F-measure.

Due to the immense runtime of D-ART and RRT, Chen et al. [4, 12, 13] proposed to apply the principle of mirroring to D-ART resp. RRT. The respective methods are abbreviated M-D-ART and M-RRT. In case of Mirror Adaptive Random Testing (M-ART), the input domain is subdivided into a fixed number of subdomains with equal size and shape. One subdomain is (randomly) chosen as the source subdomain and the remaining ones are mirror subdomains. The D-ART resp. RRT method is applied in the source subdomain, and each test case is immediately mapped from the source subdomain into the mirror subdomains (always in the same random order). Chen et al. recommend a small number of partitions with only at most one subdivision in each dimension and the mirror function "Translate". Thus, we choose the partitioning scheme $X2Y2$ with exactly one bisection of each dimension. The runtime of M-D-ART resp. M-RRT is only reduced by a constant factor $1/m^2$ compared to that of D-ART resp. RRT, where m denotes the number of partitions—in case of $X2Y2$, the constant factor is $1/4^2 = 1/16$. Therefore, the asymptotic runtime remains the same, i. e. quadratic.

Because of the (at least) quadratic runtime of D-ART and RRT (even using the principle of mirroring), Chen et al. [7] introduced Adaptive Random Testing by Dynamic Partitioning, namely Adaptive Random Testing by Bisection

[3]Our description always presumes a two-dimensional input domain. The multi-dimensional procedure is similar.

(ART-B) and Adaptive Random Testing by Random Partitioning (ART-RP).

ART-B randomly selects the first test case. Thereafter the input domain is bisected along the longer side into two subdomains (of equal size and shape). One of them contains the first test case. The second test case is randomly chosen from the other ("empty") subdomain. Since each subdomain now contains a test case, each one is again bisected along the longer side and the test cases are assigned to the respective subdomains. Further test cases are selected from "empty" subdomains until all subdomains contain a test case. Then all subdomains are bisected again and so on. If this method is implemented such that previously executed test cases are associated with their containing subdomain, it has linear runtime (in the F-measure).

ART-RP randomly selects the next test case each time from the largest subdomain (initially the whole input domain). Each test case partitions the current subdomain into four new subdomains, such that the selected test case is a corner of all four subdomains. The subdomain from which a test case has been chosen is replaced by the induced four subdomains and the procedure is repeated. If the subdomains are stored in a priority queue [20] in the order of their area, the runtime of this algorithm is in $\mathcal{O}(F \log F)$, where F denotes the F-measure.

ART-B and ART-RP cannot avoid nearby test cases being selected one after the other. Therefore, it was proposed to use the "neighboring" previously executed test cases to constrain the selection of the test cases from within the subdomains (according to D-ART resp. RRT). The respective methods are called ART by Random Paritioning [8] resp. Bisection [32] with Localization (ART-RP$_{\text{D-ART}}$, ART-RP$_{\text{RRT}}$, ART-B$_{\text{D-ART}}$, ART-B$_{\text{RRT}}$). There are always at most 2 (ART-RP) or 8 (ART-B) neighboring previously executed test cases. Thus, the localization-based ART methods have (asymptotically) the same (nearly) linear runtime as their underlying algorithms (ART-B resp. ART-RP). It has been recommended to use the parameters $k = 3$ resp. $R = 0.4$ for the RP variants, and $k = 13$ resp. $R = 0.7$ for the Bisection variants.

Another strategy to prevent nearby successive test cases is restriction, which is used by ART-B with Restriction (ART-BR) [30]. Test cases are randomly selected from a restricted subdomain. A restriction of 30% on each side (i. e. the restricted subdomain has only 40% of the width and height of the original subdomain and is centered within the original subdomain) has been recommended in the above cited publication. The runtime is of course of the same order as that of ART-B, i. e. linear in the F-measure.

Lattice-Based ART (LART) selects the test cases based on a regular rectangular lattice [31]. At the beginning, the algorithm establishes a 1×1 lattice and refines it in each iteration to a $(2^n - 1) \times (2^n - 1)$ lattice ($n = 2, 3, 4, ...$). In each iteration, the lattice points not already selected in previous iterations are selected in random order and randomly translated by a small random vector. More precisely, the selection of the test cases in one iteration is split into two passes. First, those lattice points are chosen that are farer distant from lattice points also present in previous lattices. In the second pass, all other lattice points are selected. For the maximum size of the random translation vector, 10% of the lattice spacing has been recommended. The runtime of this method is obviously linear (in the F-measure).

ART through Iterative Partitioning (IP-ART) [9] starts with a 1×1 partitioning scheme. After the selection of a test case from an "empty" partition (i. e. one that does not contain a previously executed test case and is not marked as a neighbor of such a partition), its (at most) eight neighbors are marked. As soon as all partitions are marked, the method switches from a $n \times n$ to a $(n + 1) \times (n + 1)$ partitioning scheme (also marking both partitions containing test cases and their neighbors). It can be deduced that the asymptotic runtime of this method is in $\mathcal{O}(F^{1.5})$ with F denoting the F-measure [33].

4. EMPIRICAL STUDY

RT and the ART methods are analyzed and compared in the following sections regarding testing effectiveness and runtime.

4.1 Analysis and Comparison of the Mean Relative F-Measure

In this section, an analysis of the mean relative F-measure of all ART methods is presented obtained through a simulation study and mutation analysis.

4.1.1 Simulation Study

In our simulation study, the mean F-measure was determined for common failure patterns (block, strip, and point failure pattern)—as roughly categorized by Chan et al. [3]—and different failure rates ($\theta = 0.01$, $\theta = 0.005$, $\theta = 0.002$, $\theta = 0.001$, $\theta = 0.005$). The failure patterns were randomly generated as done by Chen et al. [10, 11]. That means, that for the block failure pattern, a random square with width $\sqrt{\theta A}$ totally within the input domain was generated, where A denots the area for the input domain. For each failure rate a failure pattern was randomly generated 50,000 times for each ART method. For all our studies (also those described in the following sections) the parameters of each ART method were chosen as recommended (cf. Section 3) and the input domain was a two-dimensional square of pairs of real numbers (except the mutation analysis with "gammq").

Table 1 shows the results for the block failure pattern, which—according to Chan et al. [3]—is the most common one. The minimum mean relative F-measure in each column is set in bold face. Besides the mean relative F-measure, its accuracy on confidence level 99% obtained through the Central Limit Theorem [2] is given. The mean relative F-measure for the block failure pattern ranges from about 0.5 (LART, $\theta = 0.005$) to about 0.8 (ART-RP, $\theta = 0.001$). This means that on average only between 50% and 80% of the test cases selected by RT are necessary to detect the first failure. One should keep in mind that no testing strategy (deterministic or random) can be better than 50% the number of test cases of RT if the location of the failure region is unkown (and random) and the size of the failure region in each dimension can be neglected compared to the size of the input domain in that dimension [35]. D-ART and RRT have a mean relative F-measure of roughly about 60%, which decreases for lower failure rates. The results obtained for the mirroring variants are mostly not significantly different from their counterparts. The only method which significantly outperforms RRT is LART. However, its mean relative F-measure seems to vary from failure rate to failure rate. The least effective method is ART-RP. Moreover, its

Table 1: The mean F-measure of the respective ART method related to the theoretical mean F-measure of Random Testing for the block failure pattern

	Failure Rate θ			
	0.010	0.005	0.002	0.001
D-ART	0.673 (±0.006)	0.659 (±0.006)	0.650 (±0.006)	0.639 (±0.006)
RRT	0.648 (±0.005)	0.633 (±0.005)	0.612 (±0.005)	0.603 (±0.005)
M-D-ART	0.686 (±0.006)	0.670 (±0.006)	0.650 (±0.006)	0.645 (±0.006)
M-RRT	0.667 (±0.006)	0.649 (±0.005)	0.624 (±0.005)	0.607 (±0.005)
ART-RP	0.768 (±0.008)	0.777 (±0.008)	0.791 (±0.008)	0.795 (±0.008)
ART-$RP_{D\text{-}ART}$	0.707 (±0.007)	0.713 (±0.007)	0.721 (±0.007)	0.725 (±0.007)
ART-RP_{RRT}	0.681 (±0.006)	0.686 (±0.006)	0.690 (±0.007)	0.697 (±0.007)
ART-B	0.735 (±0.007)	0.738 (±0.007)	0.734 (±0.007)	0.740 (±0.007)
ART-$B_{D\text{-}ART}$	0.663 (±0.005)	0.651 (±0.006)	0.636 (±0.005)	0.634 (±0.005)
ART-B_{RRT}	0.680 (±0.006)	0.673 (±0.006)	0.672 (±0.006)	0.668 (±0.006)
ART-BR	0.658 (±0.007)	0.663 (±0.006)	0.674 (±0.006)	0.679 (±0.007)
LART	**0.551** (±0.005)	**0.510** (±0.004)	**0.596** (±0.005)	**0.539** (±0.005)
IP-ART	0.633 (±0.005)	0.622 (±0.005)	0.605 (±0.005)	0.602 (±0.005)

failure finding effectiveness decreases for lower failure rates. The failure finding effectiveness of ART-B however seems to be the same for all failure rates. The principle of localization applied to the dynamic partitioning methods improves them significantly. ART-$B_{D\text{-}ART}$ has an F-measure comparable to that of D-ART. Finally, ART-BR is more effective than ART-B. However, its F-measure is decreasing for decreasing failure rate (in contrast to that of ART-B) and it cannot compete with ART-$B_{D\text{-}ART}$ for lower failure rates.

4.1.2 Mutation Analysis

In order to obtain the mean relative F-measure also for real programs we used mutation analysis which was proposed by DeMillo et al. [21] and Hamlet [25] as a means to measure the adequacy of a test set. Mutants are constructed by systematically inserting faults into the system under test. The number of (non-equivalent) mutants, producing a different output than the original for at least one test case, related to the total number of (non-equivalent) mutants is called the mutation score and used as a measure for test set adequacy. Offutt et al. [40] have investigated mutation analysis and developed systems automating mutation analysis for Fortran [27] and Java [39] programs. One of the key findings of Offutt [38] is the fact that test sets that are able to "kill" mutants with simple faults (i. e. obtained by applying exactly one mutation operator exactly once) are usually also able to detect more complex faults. Recently, it has been shown that mutation analysis is not only useful to measure the adequacy of a particular test set, but also to compare testing techniques [1].

Table 2: Mutant statistics for the Numerical Recipes' programs "gammq" and "sncndn"

Number of Mutants ...	gammq	sncndn
... that did not compile	151	94
... with timeout	76	69
... with failure rate greater than 0.05	72	221
... with failure rate at most 0.05	235	47
Total number of mutants	534	431

Table 3: Empirical mean relative F-measure of 235 **mutants of "gammq" with failure rate less than or equal to** 0.05

ART Method	Mean	Minimum	Maximum
D-ART	0.521	0.404	0.753
RRT	0.560	0.454	0.792
M-D-ART	0.477	0.406	0.873
M-RRT	0.669	0.535	1.082
ART-RP	1.049	0.957	1.155
ART-$RP_{D\text{-}ART}$	0.671	0.607	0.942
ART-RP_{RRT}	0.769	0.717	0.995
ART-B	0.673	0.602	0.909
ART-$B_{D\text{-}ART}$	0.510	0.330	0.770
ART-B_{RRT}	0.634	0.559	0.891
ART-BR	0.648	0.488	1.140
LART	9.270	4.392	23.161
IP-ART	0.654	0.630	0.712

Our study is based on the Numerical Recipes' [41] programs "gammq" and "sncndn". These were implemented in C and we converted them into Java for our study, which required only mariginal changes. MuJava [39] was used to automatically generate mutants for each of these programs—using only "traditional" mutation operators, since the programs were not object-oriented. Table 2 shows the numbers of mutants obtained. Some of the mutants were semantically incorrect and therefore did not compile. Others did not "survive" a 10 s timeout, which was used to filter non-terminating mutants. Finally, using the property that the theoretical mean F-measure of Random Testing is the inverse of the failure rate, we could obtain an empirical failure rate for each mutant (using sample size 10,000). We only retained those mutants with failure rate $\theta \leq 0.05$, since for higher failure rates there is no need for more effective random testing techiques than RT.

We executed each ART method 10,000 times for each mutant and recorded the empirical mean F-measure. Table 3 shows the empirical mean F-measure of each ART method related to the empirical mean F-measure of RT for the mutants of "gammq". The second column contains the mean of the empirical mean relative F-measure averaged over all mutants. The third resp. the fourth column contains the minimum resp. the maximum empirical mean relative F-measure (of at least one particular mutant). The respective results for the mutants of "sncndn" are given in Table 4. As expected, D-ART and RRT mostly yield the best re-

Table 4: Empirical mean relative F-measure of 47 mutants of "sncndn" with failure rate less than or equal to 0.05

ART Method	Mean	Minimum	Maximum
D-ART	0.469	0.456	0.496
RRT	0.398	0.387	0.477
M-D-ART	0.913	0.879	0.969
M-RRT	0.816	0.786	0.959
ART-RP	1.142	1.010	1.178
ART-$RP_{D\text{-}ART}$	0.934	0.826	0.961
ART-RP_{RRT}	1.023	0.904	1.066
ART-B	0.952	0.900	0.980
ART-$B_{D\text{-}ART}$	0.451	0.439	0.464
ART-B_{RRT}	0.782	0.755	0.806
ART-BR	10.883	2.729	11.297
LART	39.265	8.848	40.936
IP-ART	0.673	0.641	0.692

sults. However, unlike assumed by Chen et al. [12, 13, 14], M-ART (M-D-ART as well as M-RRT) is not as effective as the underlying method (D-ART resp. RRT). For "sncndn", M-D-ART resp. M-RRT perform significantly worse than D-ART resp. RRT. Furthermore, M-RRT performs not that good for "gammq". It turns out that ART-RP can perform worse than RT, which is not desired for an ART method. As expected, ART-B is significantly more effective and not worse than RT. The localization algorithms have a better mean relative F-measure than their underlying methods. ART-$B_{D\text{-}ART}$ is as effective as D-ART. ART-BR and LART perform (at least sometimes) much worse than RT. Finally, it turnes out that IP-ART is among the best methods regarding the mean F-measure. However, it cannot be confirmed that IP-ART performs as good as D-ART resp. RRT as our simulation results and also those of Chen et al. [9] seemed to suggest.

4.2 Analysis and Comparison based on the F-Measure Distribution and the P-Measure

The (probability) density function and the cumulative density function (cdf) of the F-measure, which is the function of the P-measure depending on the size of the test set, have been determined with a similar simulation study as the one described in Section 4.1.1. The simulations were performed with 5,000,000 samples, failure rate 0.01, and the block failure pattern as similarly done in [34] for D-ART and RRT. For the D-ART resp. RRT there are at least two important facts [34]. On the one hand, the failure finding probability of the first 15–20 test cases of D-ART and RRT is worse than that of pure Random Testing. On the other hand, there is a (local) maximum up to which the probability of detecting a failure is increasing and after which this probability is decreasing. This maximum is about the 30 th test case for D-ART and about the 40 th for RRT. Further investigation has clearly shown, that the position of the first intersection of the density functions of D-ART resp. RRT and RT does not depend on the failure rate. Since for smaller failure rates more test cases are necessary and thus the first test cases become less important, the fact that the mean F-measure of both D-ART and RRT decreases for lower failure rates can be explained. However, the position of the local maximum varies in the failure rate by the obvious scaling factor θ. The P-measure function reveals that D-ART and RRT are more effective than RT starting with about the 30 th test case—independent of the failure rate. Furthermore, RRT outperforms D-ART starting with about the 60 th test case—with the obvious scaling factor θ.

The F-measure density function of the mirroring variants of D-ART resp. RRT is quite similar to that of D-ART resp. RRT. A notable difference is that the drop-off at the beginning is delayed to the 4 th test case (in contrast to the second test case for D-ART and RRT). This can be explained by the fact that the selection of the second test case of D-ART resp. RRT is performed in the 5 th test case by the mirror variants. Furthermore, the increase after the drop-off is also delayed and not as abrupt as for D-ART resp. RRT. Finally, the F-measure density function does not increase that much after the drop-off at the beginning as for D-ART resp. RRT. The P-measure is quite similar to that of the underlying methods.

The empirical density function of the F-measure of ART-B clearly contains some steps, up to which the probability of detecting a failure is more or less constant and after which it decreases abruptly (similar to the density function of LART in Figure 1). These "steps" are determined by the partitioning scheme, i. e. the abrupt drop-offs occur at F-measure 2^n—obviously independent of the failure rate. There are two differences worth being mentioned between the restricted and the unrestricted variant of ART-B. On the one hand, the probability of detecting a failure with the first test cases is significantly larger for ART-BR. For lower failure rates this level reduces to the probabilty of 0.01. Thus, ART-BR is at least not worse than RT and ART-B for the first test cases. On the other hand, the density function of ART-B slightly decreases up to the 2^n th test case, whereas for the restricted variant, the density function is more or less constant up to the next drop-off for low values of the F-measure. The P-measure exhibits that ART-BR always performs better than standard ART-B and also than RT.

The only method which really shows a F-measure distribution resembling the geometric distribution is ART-RP with and without localization. Moreover, the P-measure indicates, that, although all ART-RP variants seem to be less effective than most of the other ART methods, ART-RP with and without localization is always more effective than RT.

Although both IP-ART and LART are lattice-based methods—at least up to a certain degree—their empirical density functions of the F-measure are significantly different (cf. Figure 1).[4] The empirical density function of the F-measure of LART resembles that of ART-Bisection with Restriction. The "steps" are even more articulate than for the Bisection methods and coincide with the number of test cases after which the lattice is refined. The probability of detecting a failure of the first 49 test cases—i. e. a lattice with 7×7 lattice points is used—is about the same. From the 50 th to the 113 th test case—the first pass of the next lattice-level—the probability of detecting a failure is again constant, and so on. One additional benefit of this method is, that after

[4]Error bars denote confidence intervals (on confidence level 99%) obtained through the Central Limit Theorem [2], regarding the relative frequency of each histogram class as the mean of 0/1-valued random variables.

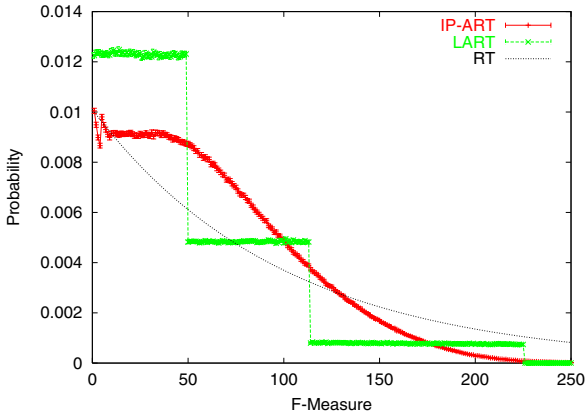

Figure 1: Empirical density function of the F-measure of IP-ART and LART in contrast to the theoretical density function of the F-measure of RT for the block failure pattern and failure rate 0.01

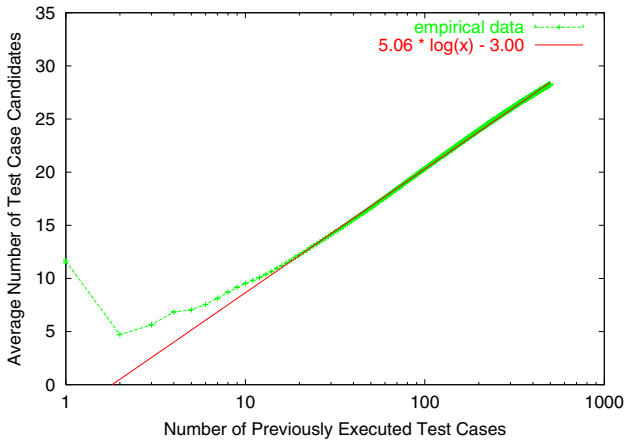

Figure 3: Average number of test case candidates required by RRT

a fixed number of test cases, it can be guaranteed, that—at least a block failure pattern—has been detected. The empirical density function of the F-measure of IP-ART, however, resembles that of D-ART resp. RRT. It has varying performance at the beginning, then it is constant up to the 40 th test case and decreases thereafter resembling the density of a geometric distribution. The P-measure indicates that IP-ART outperforms LART for more than about 140 test cases.

4.3 Spatial Distribution of the Test Cases

To determine possible shortcomings of the various ART methods, the spatial distribution of the i th test case has been sampled for all ART methods (as already done in [34] for D-ART and RRT). We each ART method was executed 10,000,000 times with a program that contains no fault and has a two-dimensional square input domain. The algorithms were stopped as soon as the size of the test set was 100. Thereafter, we computed the spatial distribution of the i th test case ($1 \leq i \leq 100$) for each method. Figure 2 shows the spatial density function of the i th test case for some ART methods and some (characteristic) values of i, where white denotes high and black denotes low frequency of occurrence.

For D-ART and RRT, whose spatial distribution of the test cases is very similar, the second test case strictly prefers the corners and partially the boundary. Later on, a uniform distribution is achieved partially, with a less frequent zone (more specifically, a frame) close to the boundary of the input domain [34]. The spatial distribution of the 5 th to 8 th test case of M-D-ART is the same as that of the second test case of D-ART mapped into the respective subdomain.

It is interesting that IP-ART exhibits a similar spatial distribution as D-ART resp. RRT, but the lattice structure is very dominating—at least for small values of i. However, the distribution of say the 100 th test case is very similar to that of D-ART resp. RRT. The frame with low frequency and also the more frequent regions adjacent to the boundary and especially to the corners can easily be seen.

One can figure out that each test case of ART-B has a spatial uniform distribution. This is also the result obtained by our experiment where no preference for any location can be seen (as for RT). This seems to be the reason, why ART-B behaves similar for all failure rates. ART-B and Localization on the one hand cleary shows a lattice-like preference of the i th test case, e. g. for $i = 100$. However, the D-ART variant is at least for low values of i quite similar (regarding the spatial distribution of test cases) to D-ART. Especially, the preference of the corners for $i = 2$ and also of the center for $i = 5$ can also be observed.

LART and ART by Bisection with Restriction exhibit a strong preference for some locations (determined through the lattice points surrounded by a small square or the restricted partitions).

4.4 Analysis of the Number of Candidates of RRT

Whereas D-ART always selects a fixed number (e. g. $k = 10$) of test case candidates, RRT generates candidates as long as one is not within any (circular) exclusion zone. So far, the average number of candidates selected by RRT has not yet been investigated. Previously, we have conjectured that the number of candidates is logarithmic in the number of previously executed test cases and used this assumption in Section 3 to determine the theoretical asymptotic runtime of RRT. In the following, we describe a study to investigate how many candidates are used on average.

We have executed RRT 1,000,000 times on a program containing no faults and stopped testing each time when the size of the test set was 500. For the generation of the i th test case (i. e. with $i-1$ previously executed test cases in the test set so far), the number of test case candidates where recorded and averaged over all 1,000,000 runs. Using the Central Limit Theorem [2], we obtained confidence intervals for the average on confidence level 99%. Furthermore, we did a regression using the least-squares method to determine the parameters a, b of the function $a \log(x) + b$ for the average number of candidates. Figure 3 depicts the empirical average number of candidates for 0 to 499 previously executed test cases (abscissae in log scale) with the errorbars indicating the 99% confidence interval and the function $5.06 \log(x) - 3.00$ obtained by the least-squares method, where x denotes the number of previously executed test cases.

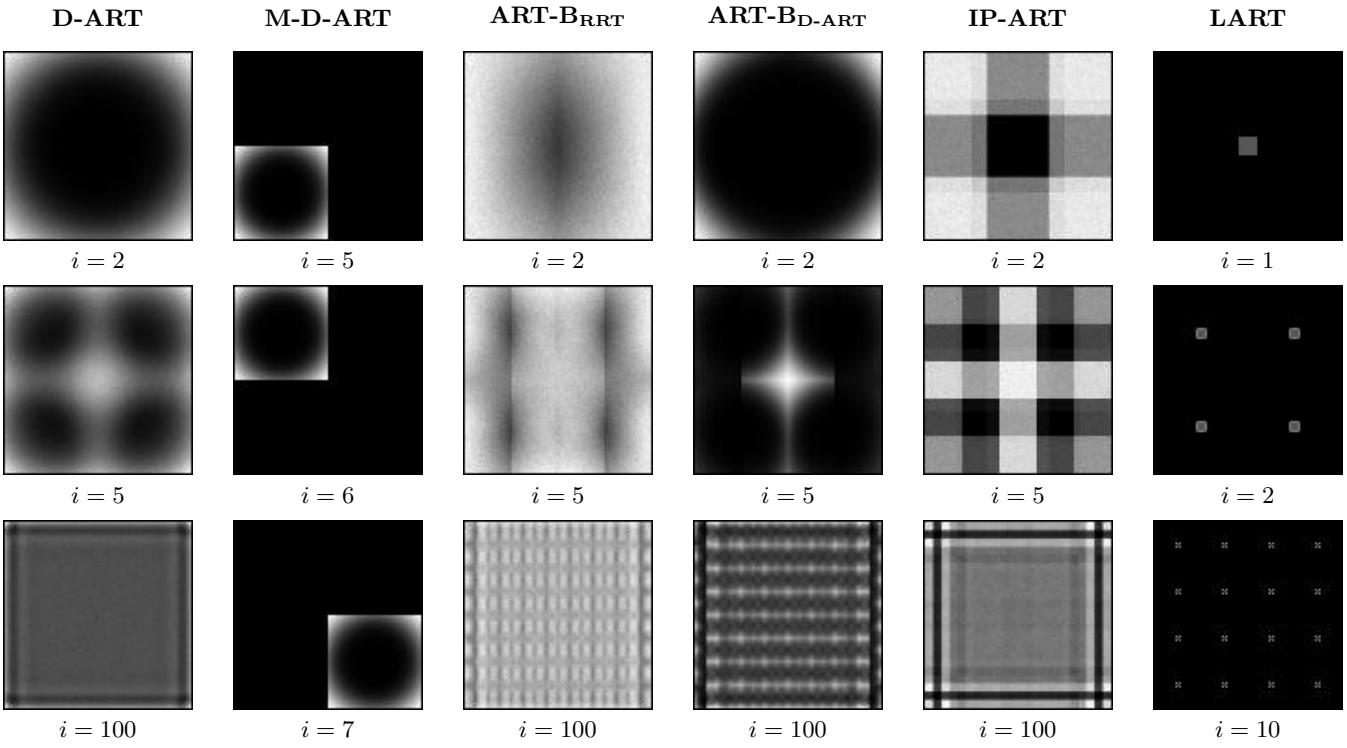

Figure 2: Spatial distribution of the i th test case of the respective random testing method (in columns)

The results clearly show that the average number of test case candidates is probably logarithmic in the number of previously executed test cases—starting with about 20 previously executed test cases. Furthermore, the average number of candidates is significantly higher than that of D-ART from at least the 20 th test case on. Most notably, for one previously executed test case, approximately 12 candidates are necessary—much more than for two and more previously executed test cases. This can be explained by the fact that the exclusion disc around the first test case will cover much more of the input domain than the exclusion discs around the first and second test case, which will significantly overlap and thereby reduce the total area of the exclusion zones.

4.5 Analysis and Comparison of the Runtime

Theoretical asymptotic runtimes are only one side of the coin. They do not show hidden constants that are essential for practical use. Therefore, we performed an empirical analysis of the runtime of all ART methods on a Sun Fire V40z server with four AMD Opteron 850 CPUs each with 2.4 GHz and 16 GB main memory. The ART methods were implemented in Java and executed with the Java HotSpot^{TM} 64-Bit Server VM (build 1.5.0_06-b05, mixed mode). We determined the average runtime of each ART method required for the computation of a test set of size 2000. For this purpose, a total of 50,000 samples were obtained for each method to estimate the average.

Whereas RT only requires about 0.0003 s to generate a test set of size 2000, D-ART needs about 0.95 s and RRT approximately 1.15 s. With about 0.055 s resp. 0.025 s for a test set of the same size, M-D-ART resp. IP-ART are much faster. The runtimes of ART-B (resp. ART-BR) and LART show some abrupt increases. These occur exactly at those times when a new lattice or partitioning scheme is employed. However, the average runtime of IP-ART seems to be quite smooth. The reason for that is that the time when a new partitioning scheme is employed varies from run to run for IP-ART. Therefore, the average runtime does not show such abrupt increases.

4.6 Discussion

Our empirical study confirmed that D-ART and RRT are indeed the best ART methods. Not only is the mean relative F-measure for simulated failure patterns with about 60% (for the block failure pattern) quite close to the optimal 50%, our results obtained by mutation analysis also confirmed that these methods perform best regarding the mean F-measure. D-ART resp. RRT outperform RT after only about 30 test cases according to the P-measure (for failure rate 0.01—and with the obvious scaling factor θ for other failure rates). The decreasing mean relative F-measure with decreasing failure rate can be explained by the spatial distribution of the individual test cases. Early test cases of D-ART resp. RRT prefer the corners and partially also the boundary of the input domain, since there are less neighboring previously executed test cases. However, from at least the 100 th test case on, test cases are nearly uniformly distributed within the input domain. Therefore, the influence of the early test cases vanishes for lower failure rate. This has also been observed with the density function of the F-measure for the block failure pattern through an abrupt drop-off for the second test case. We could thus refine the study by Chen et al. [10, 11] and show that the distribution is not nearly geometric (at least for the early test cases).

Regarding the runtime of RRT, it could be shown that the number of candidates necessary is approximately logarithmic in the number of previously executed test cases—as supposed. Therefore, the asymptotic runtime of $\mathcal{O}(F^2 \log F)$ for RRT was confirmed, where F denotes the F-measure. Our empirical runtime study has shown that D-ART and RRT have the highest runtime by far—at least two orders of magnitude more than all other ART methods (for up to 2000 test cases). It has thus been confirmed that D-ART and RRT are very slow.

It could be shown that, whereas the asymptotic runtime of the mirroring variants of D-ART and RRT is at least quadratic, the runtime for the generation of up to 2000 test cases is still quite good. However, the empirical runtime for 2000 test cases of M-D-ART is by a factor of two higher than that of IP-ART. Furthermore, the aymptotic runtime of IP-ART is much better than quadratic, which could also be observed in our experiments. Whereas the simulation results suggest, that M-ART is as good as D-ART resp. RRT, the results of our mutation analysis have shown, that this is not true (at least for some programs).

Regarding the mean F-measure, ART-B performs as mentioned in [7]. This was confirmed by our simulations as well as by the mutation analysis. The "steps" found in the density function of the F-measure for the block failure pattern have not been reported before. They are due to the fact that the partitioning scheme is always refined after the same number of test cases. However, the spatial distribution of the test cases generated by ART-B is very good, namely each test case is uniformly distributed within the input domain. Furthermore, the runtime of ART-B is with only about 0.002 s for 2000 test cases very good and asymptotically linear in the F-measure.

ART-RP also has a very low runtime, by a factor of about three worse than that of ART by Bisection, and asymptotically $\mathcal{O}(F \log F)$. Moreover, the spatial distribution is also quite close to a uniform distribution and the density function of the F-measure (for block pattern) resembles that of a geometric distribution. However, the mean F-measure can be worse than that of RT, as mutation analysis reveals. This is not tolerable. Therefore, ART-B should be preferred.

The localization variants of ART-B and ART-RP perform quite good in terms of the mean F-measure—confirmed through simulation and mutation analysis. Only ART-$\mathrm{RP}_{\mathrm{RRT}}$ may be problematic. Sometimes, such as for the mutants of "sncndn", it performs significantly worse than RT (as the maximum mean relative F-measure 1.066 reveals). On the other hand, especially the localization algorithms based on ART-B have a very good mean relative F-measure (in the simulation as well as in the mutation analysis). Especially, ART-$\mathrm{B}_{\text{D-ART}}$ has a mean relative F-measure comparable to D-ART and RRT both in the simulation and in the mutation analysis. The spatial distribution of the test cases is quite similar to that of D-ART and RRT, although the lattice structure is still noticeable. Having a look at the density function of the F-measure (for the block pattern) reveals that it roughly resembles that of D-ART resp. RRT, but the "steps" within the density function of the F-measure of ART-B can still be observed to a certain degree. The runtime of at most 0.02 s for 2000 test cases is also quite low for all localization algorithms. Therefore, ART-$\mathrm{B}_{\text{D-ART}}$ seems to be an effective and efficient alternative to D-ART and RRT.

Whereas the mean relative F-measure obtained by simulation suggests that ART-BR and LART are good ART methods, the mutation analysis reveals that these methods can be much less effective than RT. The spatial distributions of test cases are very similar for LART and ART-BR. They reveal why these methods are sometimes very ineffective: The test cases are distributed according to the lattice points (dilated by a little square). Failure patterns between the lattice points are detected very late using a much finer lattice. In case of the mutants of "sncndn", the failure region has mostly the form of a strip at the boundary of the input domain. The failure pattern of the mutants of "gammq" has in many cases the form of a triangular strip and is also close to the boundary and corner. Therefore, due to the nearly deterministic location of the test cases, and the form and location of the failure region of the mutants, both methods perform so poorly. Despite this, their runtime is very good—about 0.002 s at most for the generation of 2000 test cases.

The recently published method IP-ART has a mean relative F-measure comparable to D-ART and RRT for simulated failure patterns. Mutation analysis however shows that, despite the mean relative F-measure of IP-ART is about 65% and thus very good, IP-ART is not as effective as D-ART and RRT. At least for the block failure pattern, the distribution of the F-measure is however quite similar to that of D-ART resp. RRT. The spatial distribution of the test cases reveals on the one hand a strong similarity to D-ART and RRT, and exhibits on the other hand the lattice structure underlying the IP-ART method. Finally, the runtime of IP-ART is asymptotically much better than that of D-ART and RRT, and our experiments have shown, that IP-ART requires only about 0.025 s to generate 2000 test cases. IP-ART is thus a good alternative to D-ART and RRT.

We can conclude that D-ART and RRT are those ART methods that perform best, but the runtime of these methods is too high. ART-$\mathrm{B}_{\text{D-ART}}$ and IP-ART seem to be appropriate alternatives, which turned out to be nearly as effective as both D-ART and RRT, but have a significantly lower runtime.

The results obtained through comparison of the mean F-measure are mostly valid—as also Chen et al. [10, 11] pointed out. However, our results show that simulations with randomly generated failure patterns are not that suitable to determine the effectiveness. The crucial point is that failure regions in real programs are located deterministically. It seems as if even the failure regions of the mutants of one single program seem to resemble each other in shape and position. Therefore, simulations with randomly located failure regions seem to be unrealistic, since a method has to be effective (at least as effective as RT) for *every program*.

Although testing methods should be equally efficient independent of the location of the failure pattern, the spatial distribution of the test cases revealed that the test cases generated by nearly all ART algorithms clearly prefer certain regions of the input domain (such as the corners, the boundary, and the center). However, there are methods such as ART-B whose test cases are uniformly distributed.

As pointed out in [34], the mean F-measure of testing with unlimited resources has no meaning for testing with constrained resources. However, it has been explained in the present paper that the cdf of the F-measure is the P-

measure for various sizes of the test set. Thus, analysis of testing with unlimited resources can be used to gain valuable information for the resource-constrained case, namely the P-measure.

We have investigated the random testing techniques for a two-dimensional square input domain (except for "gammq") of pairs of real numbers. Higher dimensional input domains and more complex inputs are out of the scope of the present paper. However, Merkel [35] has considered more complex input domains for ART. In order to apply the ART methods to more complex input domains, metrics and partitioning schemes for the respective input domains have to be figured out.

5. CONCLUSION

We have investigated Adaptive Random Testing (ART) methods according to their effectiveness regarding the F-measure, the number of test cases necessary to detect the first failure, and the P-measure, the probability of detecting a failure with a given test set. Furthermore the spatial distribution of the test cases generated by the individual methods, revealed several shortcomings of the respective methods. Finally, theoretical asymptotic runtimes are given, the assumption of a logarithmic number of candidates (in the number of previously executed test cases) for RRT has been confirmed, and the real runtimes for the generation of test sets of fixed size have been determined.

It turned out, that D-ART and RRT are really the most effective ART methods. However, their runtime is quite high. Our studies showed that ART-$B_{\text{D-ART}}$ as well as IP-ART are good alternatives being nearly as effective as D-ART and RRT on the one hand, and being very efficient on the other hand.

The spatial distribution of the test cases revealed that the test cases generated by a lot of ART methods are not uniformly distributed within the input domain. Instead, certain locations within the input domain are preferred by the test cases generated. Therefore, improved methods should be developed that generate test cases such that no part of the input domain is preferred.

Finally, it has to be pointed out that there are some ART methods, namely D-ART, RRT, IP-ART, and ART-$B_{\text{D-ART}}$, that are much more effective than RT in general and at least no worse. These methods should be preferred to RT in cases where one single execution of the system under test takes a lot of time.

6. REFERENCES

[1] J. H. Andrews, L. C. Briand, and Y. Labiche. Is mutation an appropriate tool for testing experiments? In *Proceedings of the 27th International Conference on Software Engineering (ICSE 2005)*, pages 402–411. ACM, 2005.

[2] G. Casella and R. L. Berger. *Statistical Inference*. Wadsworth Group, Duxbury, CA, USA, 2002.

[3] 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*, 38:775–782, 1996.

[4] K. P. Chan, T. Y. Chen, F.-C. Kuo, and D. Towey. A revisit of adaptive random testing by restriction. In *Proceedings of the 28th International Computer Software and Applications Conference (COMPSAC 2004)*, pages 78–85. IEEE Computer Society, 2004.

[5] K. P. Chan, T. Y. Chen, and D. Towey. Restricted random testing. In J. Kontio and R. Conradi, editors, *Proceedings of the 7th European Conference on Software Quality (ECSQ 2002)*, volume 2349 of *Lecture Notes in Computer Science*, pages 321–330. Springer, 2002.

[6] K. P. Chan, T. Y. Chen, and D. Towey. Normalized restricted random testing. In *Proceedings of the 18th Ada-Europe International Conference on Reliable Software Technologies*, volume 2655 of *Lecture Notes in Computer Science*, pages 368–381. Springer, 2003.

[7] T. Y. Chen, G. Eddy, R. Merkel, and P. K. Wong. Adaptive random testing through dynamic partitioning. In *Proceedings of the 4th International Conference on Quality Software (QSIC 2004)*, pages 79–86. IEEE Computer Society, 2004.

[8] T. Y. Chen and D. Huang. Adaptive random testing by localization. In *Proceedings of the 11th Asia-Pacific Software Engineering Conference (APSEC 2004)*, pages 292–298. IEEE Computer Society, 2004.

[9] T. Y. Chen, D. H. Huang, and Z. Q. Zhou. Adaptive random testing through iterative partitioning. In *Proceedings of the 11th International Conference on Reliable Software Technologies*, volume 4006 of *Lecture Notes in Computer Science*, pages 155–166. Springer-Verlag, Berlin, 2006.

[10] T. Y. Chen, F.-C. Kuo, and R. Merkel. On the statistical properties of the F-measure. In *Proceedings of the 4th International Conference on Quality Software (QSIC 2004)*, pages 146–153. IEEE Computer Society, 2004.

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

[12] T. Y. Chen, F.-C. Kuo, R. G. Merkel, and S. P. Ng. Mirror adaptive random testing. In *Proceedings of the 3rd International Conference on Quality Software (QSIC 2003)*, pages 4–11. IEEE Computer Society, 2003.

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

[14] T. Y. Chen, H. Leung, and I. K. Mak. Adaptive random testing. In M. J. Maher, editor, *Proceedings of the 9th Asian Computing Science Conference (ASIAN 2004)*, volume 3321 of *Lecture Notes in Computer Science*, pages 320–329. Springer, 2004.

[15] T. Y. Chen and R. Merkel. Quasi-random testing. In *Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering (ASE 2005)*, pages 309–312. ACM, 2005.

[16] T. Y. Chen and R. Merkel. Efficient and effective random testing using the Voronoi diagram. In *Proceedings of the Australian Software Engineering Conference (ASWEC 2006)*, pages 300–308. IEEE Computer Society, 2006.

[17] T. Y. Chen, T. H. Tse, and Y. T. Yu. Proportional sampling strategy: A compendium and some insights. *The Journal of Systems and Software*, 58:65–81, 2001.

[18] T. Y. Chen and Y. T. Yu. On the relationship between partition and random testing. *IEEE Transactions on Software Engineering*, 20:977–980, 1994.

[19] 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.

[20] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, second edition, 2001.

[21] R. A. DeMillo, R. J. Lipton, and F. G. Sayward. Hints on test data selection: Help for the practicing programmer. *IEEE Computer*, 11:34–41, 1978.

[22] J. E. Forrester and B. P. Miller. An empirical study of the robustness of Windows NT applications using random testing. In *Proceedings of the 4th USENIX Windows Systems Symposium*, pages 59–68, 2000.

[23] W. J. Gutjahr. Partition testing vs. random testing: The influence of uncertainty. *IEEE Transactions on Software Engineering*, 25(5):661–674, 1999.

[24] R. Hamlet. Random testing. In *Encylopedia of Software Engineering*, pages 970–978. Wiley, 1994.

[25] R. G. Hamlet. Testing programs with the aid of a compiler. *IEEE Transactions on Software*, 3(4):279–290, 1977.

[26] R. G. Hamlet and R. Taylor. Partition testing does not inspire confidence. *IEEE Transactions on Software Engineering*, 16:1402–1411, 1990.

[27] K. N. King and A. J. Offutt. A fortran language system for mutation-based software testing. *Software Practice and Experience*, 21(7):685–718, 1991.

[28] P. S. Loo and W. K. Tsai. Random testing revisited. *Information and Software Technology*, 30:402–417, 1988.

[29] I. K. Mak. On the effectiveness of random testing. Master thesis, Department of Computer Science, University of Melbourne, Australia, 1997.

[30] J. Mayer. Adaptive random testing by bisection with restriction. In *Proceedings of the Seventh International Conference on Formal Engineering Methods (ICFEM 2005)*, volume 3785 of *Lecture Notes in Computer Science*, pages 251–263. Springer-Verlag, Berlin, 2005.

[31] J. Mayer. Lattice-based adaptive random testing. In *Proceedings of the 20th IEEE/ACM International Conference on Automated Software Engineering (ASE 2005)*, pages 333–336. ACM, 2005.

[32] J. Mayer. Adaptive random testing by bisection and localization. In *Proceedings of the Fifth International Workshop on Formal Approaches to Testing of Software (FATES 2005)*, volume 3997 of *Lecture Notes in Computer Science*, pages 72–86. Springer-Verlag, Berlin, 2006.

[33] J. Mayer. Efficient and effective random testing based on partitioning and neighborhood. In *Proceedings of the 18th International Conference on Software Engineering and Knowledge Engineering (SEKE 2006)*, pages 479–484. Knowledge Systems Institute Graduate School, Skokie, USA, 2006.

[34] J. Mayer and C. Schneckenburger. Statistical analysis and enhancement of random testing methods also under constrained resources. In *Proceedings of the International Conference on Software Engineering Research and Practice (SERP 2006)*, pages 16–23. CSREA Press, 2006.

[35] R. Merkel. *Analysis and Enhancements of Adaptive Random Testing*. PhD thesis, Swinburne University of Technology, Australia, 2005.

[36] S. Morasca and S. Serra-Capizzano. On the analytical comparison of testing techniques. In *Proceedings of the International Symposium on Software Testing and Analysis (ISSTA 2004)*, pages 154–164. ACM, 2004.

[37] G. J. Myers. *The Art of Software Testing*. Wiley, New York, 1979.

[38] A. J. Offutt. Investigations of the software testing coupling effect. *ACM Transactions on Software Engineering Methodology*, 1(1):5–20, 1992.

[39] J. Offutt, Y.-S. Ma, and Y.-R. Kwon. An experimental mutation system for Java. *ACM SIGSOFT Software Engineering Notes*, 29(5):1–4, 2004.

[40] J. Offutt and R. H. Untch. Mutation 2000: Uniting the orthogonal. In *Proceedings of Mutation 2000: Mutation Testing in the Twentieth and the Twenty First Centuries, San Jose, CA*, pages 45–55, 2000.

[41] W. H. Press, B. P. Flannery, S. A. Teulolsky, and W. T. Vetterling. *Numerical Recipes*. Cambridge University Press, 1986.

[42] D. Slutz. Massive stochastic testing of SQL. In *Proceedings of the 24th International Conference on Very Large Data Bases (VLDB 1998)*, pages 618–622, 1998.

[43] E. J. Weyuker and B. Jeng. Analysing partition testing strategies. *IEEE Transactions on Software Engineering*, 17:703–711, 1991.

[44] T. Yoshikawa, K. Shimura, and T. Ozawa. Random program generator for Java JIT compiler test system. In *Proceedings of the 3rd International Conference on Quality Software (QSIC 2003)*, pages 20–24. IEEE Computer Society, 2003.