Empirical Studies of Software Testing Techniques: Challenges, Practical Strategies, and Future Research

Lionel Briand, Yvan Labiche
Software Quality Engineering Laboratory
Systems and Computer Engineering Department - Carleton University
Ottawa, ON K1S5B6, Canada
{briand,labiche}@sce.carleton.ca

Abstract

This position paper aims at discussing a number of issues that typically arise when performing empirical studies with software testing techniques. Though some problems are general to all empirical disciplines, software testing studies face a number of specific challenges. Some of the main ones are discussed in sequence below.

1. Fault Seeding

One of the main issues one faces when assessing and comparing testing techniques is to quantify their fault detection effectiveness. Ideally, faults should be representative of real life faults [5] so that effectiveness results are themselves representative. But there are a number of problems with this strategy. First, many real-world components, subsystems or even systems do not have a large enough set of recorded defects to be suitable for quantitative analysis. Recall that, though we can expect testing techniques to be associated with fault detection rates, this relationship is statistical in nature. The detection of defects will always be, to a certain degree, a stochastic process. So, very often, when comparing test techniques, we are comparing fault detection distributions and resorting to statistical inference testing to determine whether the results could have been obtained by chance. To do so, we need to work with large enough fault samples. Such samples are rarely available in real-world systems (fortunately!).

Another issue is that when referring to real-world faults, we very likely refer to a statistical population that does not exist. Every company and system is likely to be associated with different fault populations. In other words, if we target “real fault”, our target is likely to be elusive. This is not to say that industrial studies are not useful. We will get back to that issue below.

Because of the issues discussed above, many researchers have resorted to fault seeding to perform empirical studies [2, 3, 6, 7, 12]. The problems associated with fault seeding are well-known. How to be impartial and unbiased when seeding fault for the purpose of assessing a technique? How do we ensure the results we obtain will be somewhat generalizable to real fault populations? There is no perfect answer. However, one technique has been to use mutation operators to seed faults [5, 11]. We cannot guarantee that the faults seeded are representative of a particular population, but we ensure that a wide variety of faults are systematically inserted, in a somewhat impartial, random manner (e.g., [4, 9, 10]). And this is probably the best we can do in an artificial setting.

We see such studies as a first attempt to obtain ball park figures on fault detection effectiveness [2], cost, and also a better insight into what type of faults are difficult to detect by a given technique [1]. Such studies have then to be complemented by field studies, as discussed next.

Another issue related to fault seeding is that if many of the faults seeded are too easy to detect, then we may not differentiate the fault detection effectiveness of alternative techniques. Therefore, we found it useful to always check the percentage of test cases that generate failures on a faulty program. Very easy faults as well as faults that cannot possibly have any impact on the system behavior should probably be ignored when assessing effectiveness.

2. Academic versus industrial settings

There have been a lot of discussions about the value of experiments in academic settings [8]. There have been concerns about the fact that artifacts may not be representative, in terms of scale and complexity, of their industrial counterparts (ref). When involving students, then it is also natural to question whether results can be generalized to professionals, who often exhibit substantially more experience.
And of course, as discussed above, there is the set of faults we have to work with, which then may not be representative of faults detected in the field. All the above are valid concerns. But what experience has shown us, in the more general field of empirical software engineering, is that there are no perfect empirical studies. Field studies, in industrial settings, also involve a number of problems.

- First, as mentioned above, the number of detected faults may not be large enough to allow for quantitative, statistical analysis.
- Second, when working with actual professionals, we often have very little control over their training. In situations where we want to assess new or advanced techniques, professionals may not be adequate subjects. Especially so if we want to assess the upper-bound, potential benefits of a technique when it is properly applied.
- Third, studies in academic settings are often easier to control. One of the important challenges of experimentation is to ensure that there are no confounding effects [13] between the factors you are studying (test technique) and other factors (extraneous, human). This is usually very hard to ensure in an industry context.

So, from the above discussion, we can see that both academic and industry settings exhibit strengths and weaknesses. Studies in academia are often strong in terms of internal validity (i.e., our capability to draw proper conclusions from the data) and weak as far as external validity is concerned (i.e., it is hard to know the extent to which you can generalize your results to industrial contexts). Field studies have exactly the opposite strengths and weaknesses. Though one might also argue that results in a given project and organization might not be generalizable to other organizations and projects.

We conclude that both academic and field studies are necessary. The former are good to obtain ballpark figures on the effectiveness of techniques and better understand their strengths and weaknesses. Field studies are more suited to assess to difficulties to apply techniques in practice and to confirm the results obtained on real sets of faults. They are also very useful to attempt cost-effectiveness analyses as cost data can be collected. No study is ever going to be perfect, or even close to that, but over time, the cumulative knowledge obtained by subsequent studies can allow us to build a body of knowledge. Last, studies in an academic setting are often a first step before studies in industrial settings [8].

3. Replication

The discussion above implies that studies be replicated across different settings. This implies that a single study is not likely to have a significant impact. Studies need to be replicated for the results to be credible. This has been known in many other disciplines for a long time and there are techniques, called meta-analyses, to draw conclusions from a body of experiments [13]. However, replication does not necessarily mean the exact repetition of a study. Often, artifacts may be different, the training and background of participants (if any) may vary substantially. Even the design of the study may change to address a particular threat to validity. Or simply, the number of observations may be larger or smaller, thus affecting our capability to obtain visible effects. All such types of information should be reported when writing about a replication, so that in the long run differences among studies can be explained, possibly through meta-analyses. It would probably be useful for the testing research community to define a template of information to report when performing and documenting testing experiments.

4. Involving Human Subjects

In software testing, one can classify studies in two categories. A first category consists of studies that involve human subjects applying testing techniques [1]. A second category characterizes what we would call simulations, where techniques are applied by simulating the construction of test sets and their execution on faulty version of programs [2]. Both types of studies have drawbacks and advantages. The former allows us to assess not only cost effectiveness but also the applicability of techniques by trained engineers, thus accounting for human factors in our conclusions. After all, humans cannot yet be factored out of software testing .... One important issue though is related to the abovementioned statistical nature of fault detection rates. To really compare test techniques, in a rigorous and quantitative manner, we need to generate large number of test sets for each of the techniques. This is usually infeasible when humans are involved and this is where simulation kicks in. It allows us to generate large number of test sets, run them on large numbers of faulty programs, and is therefore amenable to rigorous statistical analysis.

The main problem though is to guarantee that the simulation process does not introduce any bias in the results, that it is representative somehow of test sets that would be generated by humans. We know this is not the case as for one thing humans commit mistakes when using test techniques. Some techniques may be more complex and lead to more mistakes, especially if it is not well supported by tools.

So, again, we conclude that both types of studies are required as they are complementary.
5. Future Research

There are a number of practices that can help us alleviate the challenges that are discussed above. Because so much experimentation is required and preparing experiments is so effort intensive, it is necessary for researchers to share experimental material. Furthermore, experimenting on a set of common systems (benchmark) will make comparisons of techniques much easier. A set of systems can then be used as benchmarks for the initial validation of techniques. One difficulty is that different test techniques usually require different pieces of information about the system, e.g., specifications in different forms, or may target different types of systems. So choosing appropriate benchmarks may not be so easy. Experiment should lead to an experimentation package that would allow other researchers to easily replicate experiments. Ideally, it should contain all the necessary material to perform the experiment and should be publicly available, under certain conditions. This would allow the research community to converge much faster towards credible results.

Each time a data analysis is performed in an experiment, it would be nice if the data could be made available, even if it implies to somehow sanitizing them if confidentiality is an issue. This would allow other researcher to analyze the data and possibly draw different conclusions.

Last but not least, since we cannot expect experiments to be perfect with respect to all threats to validity, we need to establish standards regarding how to perform and report test experiments. Some of these can be inspired by other experimental fields but there will definitely be a lot of tailoring to be done. This will facilitate the reviewing of experimental test papers and help the publication of more results.

6. References


