A Theoretical Study: The Impact of Cloning Failed Test Cases on the Effectiveness of Fault Localization*

Yichao Gao
School of Automation Science and Electrical Engineering
Beihang University
Beijing, China
gaoyichao2@gmail.com

Zhenyu Zhang, Long Zhang
Institute of Software
Chinese Academy of Sciences
Beijing, China
zhangzy@ios.ac.cn, zlong@ios.ac.cn

Cheng Gong, Zheng Zheng +
School of Automation Science and Electrical Engineering
Beihang University
Beijing, China
zhengz@buaa.edu.cn

Abstract—**Statistical fault localization techniques analyze the dynamic program information provided by executing a large number of test cases to predict fault positions in faulty programs. Related studies show that the extent of imbalance between the number of passed test cases and that of failed test cases may reduce the effectiveness of such techniques, while failed test cases can frequently be less than passed test cases in practice. In this study, we propose a strategy to generate balanced test suite by cloning the failed test cases for suitable number of times to catch up with the number of passed test cases. We further give an analysis to show that by carrying out the cloning the effectiveness of two representative fault localization techniques can be improved under certain conditions and impaired at no time.**

Keywords- Software debugging; fault localization; class imbalance

I. INTRODUCTION

In the process of software development, it is very hard to thoroughly avoid the appearance of software defects in final releases. As a result, a large number of software developers have been working to improve the quality and reliability of software. Manual debugging is helpful in finding out faults in the early stage of software developing activities. However, with the rapid development and the spread of software, the scale and complexity of software have increased dramatically. At the same time, the problems caused by software faults grow even more quickly, and the efficacy and efficiency of manual debugging are no longer applicable.

In the realistic software testing process, to find faults in a program is usually a time-consuming and labor-intensive work. Software developers have been committed to the automated or semi-automated fault localization method. Many fault localization methods have been proposed, of which one class, having considerable accuracy and the ease to apply, the spectrum-based fault localization, has attracted extensive attentions [1-8]. This class of techniques obtains the execution information by executing a large number of test cases, and uses correlation algorithms to estimate the extent of suspiciousness of each executable program unit. All the suspicious program units are ranked into a list accordingly, in a manner to facilitate the locating of faults in a faulty program.

So far there is no spectra-based fault localization algorithm, which can perfectly satisfy real-life requirements in software development practice or has been extensively adopted by industry. One reason is that such fault localization algorithms rely on huge information of both passed and failed executions to provide sufficient data for statistical analysis and in practice, to obtain a small amount of the failed cases may usually require executing a large number of executions. Secondly, in the software debugging and repairing process, when faults are disclosed and fixed continuously, it becomes more and more difficult to locate the remaining faults effectively, efficiently, and accurately. Thirdly, the accurate positioning of the hidden faults usually tends to be a hard job, as hidden faults often cause few software failures. How to accurately locate software faults with few failed execution information becomes an urgent task in the research of automated software debugging.

Existing studies showed that many existing statistical fault localization techniques ignored the organizational structure of the test suites. In the literature [9], the authors referred the proportion of the failed and the passed sets of test cases as test case category ratio. For different such ratios, they conducted in-depth research with experiments to show that spectrum-based fault localization techniques tend to exhibit better localization capability with a more balanced test suite, that is, having identical number of passed and failed test cases, as input. However, how to construct a balanced test suite is unknown.

From a theoretical point of view, this paper proposes to clone the failed test cases to construct a balanced test suite, and analyzes the impact of such a method when applied to different fault localization techniques. The main contribution of this paper includes (1) to propose a strategy to construct a balanced test suite, and (2) to analyze its impact on different fault localization techniques.

The paper is organized as follows. Section 2 describes the related work. Section 3 gives the cloning strategy to construct a balanced test suite and analyzes its impact on representative fault localization techniques, starting from the theoretical point of view. Section 4 summarizes and gives the future work.

II. RELATED WORK

According to the IEEE 729-1983 standard, software defects are existing errors, fault, and other problems, in the software product development or maintenance process. A fault localization task is to locate the root cause of software defects in source code. At present, software fault location technologies include slicing-based fault localization, model-

978-0-7695-5039-8/13 $26.00 © 2013 IEEE
DOI 10.1109/QSIC.2013.23

based fault localization, spectrum-based fault localization, and so on. Among them, the spectrum-based fault localization forms an extremely big family [1][4][6][8].

Spectrum-based fault localization techniques are influenced by the characteristics of test suite, say the extent of class balance for passed and failed test cases, which is expressed by the ratio of the number of failed test cases and that of passed test cases in a test suite [9]. In recent years, the test suite reduction techniques, paid much extensive attention by relevant researchers, attempt in a sense of higher fault localization accuracy to reduce the size of test suite. Hao et al. [10] and related studies have suggested that it can effectively improve the efficiency of fault localization algorithms, by cutting similar test cases and reducing information redundancy. The literature [11] also comes to similar conclusions. However, there are related studies that reached contrary conclusions. In the literature [12], with ten test suite reduction methods, by comparing four fault localization techniques, Jones et al. concluded that current test suite reduction techniques can reduce the efficiency of fault localization techniques.

Gong et al. [9] empirically showed that a test suite containing approximately identical numbers of passed and failed test cases can mostly favor existing fault localization techniques. This paper targets at giving analytical study to consolidate this point.

III. Our Method and Analysis

In this section, we firstly outline the problems of class imbalance of test suite briefly, and after that give the cloning strategy, which construct a test suite with balanced class ratio by cloning the failed test cases. Finally, we select two representative fault localization techniques, Jaccard and Wong2, and give an analytical study on the impact of the cloning strategy on their fault localization effectiveness.

A. Problem Setings

Given a program *Q*, a software testing process starts with a given test suite *W*, the test cases in which are categorized into two groups, passed set and failed set, according to whether the execution of *Q* over that test case produces an expected output. Here, we use *P* to represent the size of passed test cases, and let *F* represent the size of failed test cases. Using the coverage information of *Q* with respect to each passed or failed test case as input, a fault localization technique is expected to predict the suspicious statement that relates to program faults.

It is known that failed test cases are often less than passed test cases (denoted as $F \ll P$ in this paper) in real-life scenario [9]. And previous studies showed that many fault localization techniques manifest low fault localization effectiveness with such imbalanced class ratio ($F \ll P$). A straightforward idea is to construct a test suite with balanced class ratio. To know the impact of such a balanced test suite on fault localization, we in this paper propose a simplest solution as follows.

[Cloning Strategy] *In the development process of software testing, software developers proposed a lot of cloning strategy. We chose the strategy—cloning entire failed test cases. Cloning the failed test cases for* $\left\lfloor \frac{P}{F} \right\rfloor - 1$ *times, so that the synthesized test suite contains P passed test cases and* F' ($\cong P$) *failed test cases, where* F' *represents the number of failed test cases after cloning*

Our research question in this paper is as follows.

[**Research Problem**] *Will a fault localization technique benefit from the above cloning strategy, in terms of fault localization effectiveness?*

In the next section, we will visit two representative fault localization techniques and give analytical investigation. The analysis in this paper is based on the following assumptions. (1) The program execution over each failed test case always exercises the faulty statement. (2) The target faulty program contains only one faulty statement.

B. Definitions

The rest of the paper uses the following symbols:

- The symbol i represents a statement.
- The symbol t represents the faulty statement.
- The symbol a_{ef}^{i} denotes the number of failed test cases that exercise the statement i.
- The symbol a_{nf}^{i} denotes the number of failed test cases that do not exercise the statement i.
- The symbol a_{ep}^{i} denotes the number of passed test cases that exercise the statement i.
- The symbol a_{np}^{i} denotes the number of passed test cases that do not exercise the statement i.
- The symbol r_i denotes the rank of the statement i in the generated ranked list, which is got by sorting all the statements in order of their calculated suspiciousness score. Here, $r_i > r_t$ denotes that the rank of the statement i is higher than that of the faulty statement t. $r_i = r_t$ denotes that the rank of the statement i is equal to that of the faulty statement t. $r_i < r_t$ denotes that the rank of the statement i is lower than that of the faulty statement *t*.

To ease the presentation, we define the following terms.

- S_B denotes the set of statements which suspiciousness scores are greater than that of the faulty statement *t*.
- S_F denotes the set of statements which suspiciousness scores are equal to that of the faulty statement *t*.
- S_A denotes the set of statements which suspiciousness scores are less than that of the faulty statement *t*.

C. Theoretical analysis

In this section, we select two representative and simplest fault localization techniques, Jaccard and Wong2, as examples to analyze the impact of applying our cloning strategy to improve fault localization effectiveness of fault localization techniques. In addition, some other techniques are also suitable for the strategy, such as Ochiai, Tarantula, Wong1, CBI etc.

The Jaccard algorithm

The suspiciousness formula of Jaccard is as follows:

$$Jaccard(i) = \frac{a_{ef}^{i}}{a_{ef}^{i} + a_{nf}^{i} + a_{ep}^{i}} = \frac{a_{ef}^{i}}{F + a_{np}^{i}}$$

We next construct the function *T* to compare the suspiciousness of a statement *i* and the faulty statement *t*.

$$T = Jaccard(i) - Jaccard(t) = \frac{a_{ef}^i}{F+a_{ep}^i} - \frac{a_{ef}^t}{F+a_{ep}^t} = \frac{a_{ef}^i}{F+a_{ep}^i} - \frac{F}{F+a_{ep}^t}$$

We let a_{ef}^i=$x\in[0, F]$, a_{ep}^i=$y\in[0, P]$, and a_{ep}^t=$z\in[0, P]$, where x and F are variables, and y and z are constants. So,

$$T = \frac{x}{F+y} - \frac{F}{F+z} = \frac{F(x-F)+xz-Fy}{(F+y)(F+z)}$$

We define the following term to simplify our discussion.

$$f(x, F) = F(x-F) + xz - Fy \qquad (3.1)$$

After applying the cloning strategy, the test suite is changed, and we can apply the formula *Jaccard*(i) to re-calculate T. This time, we use T' to stand for the new value of T after applying the cloning strategy. The value of y and z are not changed, while x and F are changed to $x' = cx, F' = cF$ in T'. We assume the size of failed test suite after cloning is $c \in (1, +\infty)$ times that of before and get,

$$T' = \frac{cx}{cF+y} + \frac{cF}{cF+z} = \frac{c[cF(x-F)+xz-Fy]}{(cF+y)(cF+z)}$$

We define the following term to simplify our discussion.

$$g(x, F) = cF(x-F)+xz-Fy \qquad (3.2)$$

To know which set of S_A, S_F or S_B statement i belongs to, we follow three rules. 1) If $f(x,F)<0 \Rightarrow r_i < r_t$, the statement i belongs S_A set. 2) If $f(x,F)=0 \Rightarrow r_i = r_t$, the statement i belongs S_F set. 3) if $f(x,F)>0$, $r_i > r_t$, the statement i belongs S_B set. So we have the following two judgments.

a) When $xz-Fy<0$, from formula (3.1), we know that $f(x,F)<0 \Rightarrow r_i < r_t$; according to formula(3.2), we know that $g(x,F)<0 \Rightarrow r_i < r_t$. So we can safely conclude that before cloning, the statement i belongs to set S_A, and after cloning, it still belongs to set S_A.

b) When $xz-Fy=0$, our analysis consists of two parts.

i) If $x=F$, from formula (3.1), we know that $f(x,F)=0 \Rightarrow r_i = r_t$; according to formula (3.2), we know that $g(x,F)=0 \Rightarrow r_i = r_t$. So we can safely conclude that before the cloning action, the statement i belongs to set S_F, and after that, it still belongs to set S_F.

ii) If $x<F$, from formula (3.1), we know that $f(x,F)<0 \Rightarrow r_i < r_t$; according to formula (3.2), we know that $g(x,F)<0 \Rightarrow r_i < r_t$. So we can safely conclude that before the cloning action, the statement i belongs to set S_A, and after the copy, it still belongs to set S_A.

c) When $xz-Fy>0$, our analysis consists of two parts.

i) If $x=F$, from formula (3.1), we know that $f(x,F)>0 \Rightarrow r_i > r_t$; according to formula (3.2), we know that $g(x,F)>0 \Rightarrow r_i > r_t$. So we can safely conclude that before the cloning action, the statement i belongs to set S_B, and after that, it still belongs to set S_B.

ii) If $x<F$, according to formula (3.1), we know that,

$$case1: \; if(x,F) \le 0 \;\Rightarrow\; r_i \le r_t$$

$$\Rightarrow \begin{cases} f(x,F) = F(x-F) + xz - Fy \le 0 \\ g(x,F) = cF(x-F) + xz - Fy \\ c > 1 \end{cases}$$

$$\Rightarrow \quad g(x,F) < 0$$

It means before cloning, the statement i belongs to set S_A or S_F, and after cloning belongs to set S_A

$$case\,2: \; if(x,F) > 0 \;\Rightarrow\; r_i > r_t$$

$$\Rightarrow \begin{cases} f(x,F) = F(x-F) + xz - Fy > 0 \\ g(x,F) = cF(x-F) + xz - Fy \\ c > 1 \end{cases}$$

$$\Rightarrow \begin{cases} if(1 < c < \frac{xz-Fy}{F(F-x)}) \; g(x,F) > 0 \Rightarrow r_i > r_t \\ if(c = \frac{xz-Fy}{F(F-x)}) \; g(x,F) = 0 \Rightarrow r_i = r_t \\ if(c > \frac{xz-Fy}{F(F-x)}) \; g(x,F) < 0 \Rightarrow r_i < r_t \end{cases}$$

It means, before cloning, the statement i belongs to set S_B, and after cloning belongs to set S_A, S_F or S_B.

In summary, under the following certain conditions, the effectiveness of Jaccard can be improved by applying the cloning strategy.

1) $\exists i(i \neq t)$, $\{i \mid a_{ef}^i a_{ep}^t - F a_{ep}^i > 0\} \cap \{i \mid a_{ef}^i < F\} \cap \{i \mid F(a_{ef}^i - F) + a_{ef}^i a_{ep}^t - F a_{ep}^i = 0\}$. This conditions means there exist certain such kind of statements that belong to set S_F before cloning and belong to set S_A after cloning.

2) $\exists i(i \neq t)$, $\{i \mid a_{ef}^i a_{ep}^t - F a_{ep}^i > 0\} \cap \{i \mid a_{ef}^i < F\} \cap \{i \mid c > \frac{a_{ef}^i a_{ep}^t - F a_{ep}^i}{F(F-a_{ef}^i)} > 1\}$. This conditions means there exist certain such kind of statements that belong to set S_B before cloning and belong to set S_A after cloning.

The Wong2 algorithm

The suspiciousness formula of Wong2 is as follows:

$$Wong2(i) = a_{ef}^i - a_{ep}^i$$

We next construct the function T to compare the suspiciousness of a statement i and the faulty statement t.

$$T = Wong2(i) - Wong2(t)$$

$$= a_{ef}^i - a_{ep}^i - (a_{ef}^t - a_{ep}^t) = a_{ef}^i - a_{ep}^i - (F - a_{ep}^t)$$

We let a_{ef}^i=$x\in[0, F]$, a_{ep}^i=$y\in[0, P]$ and a_{ep}^t=$z\in[0, P]$, where x and F are the variables, and y and z are constants

$$T = x - y - F + z \qquad (3.3)$$

After applying the cloning strategy, the test suite is changed, and we can apply the formula *Wong2*(i) to re-calculate T. This time, we use T' to stand for the new value of T after applying the cloning strategy. The value of y and z are not changed, while x and F are changed to $x' = cx, F' = cF$ in T'. We assume the size of failed test suite after cloning is $c \in (1, +\infty)$ times that of before and get

$$T' = cx - y - cF = c(x-F) - y + z \qquad (3.4)$$

To know which set of S_A, S_F or S_B statement i belongs to, we follow three rules. 1) If $T'<0 \Rightarrow r_i < r_t$, the statement i belongs S_A set. 2) If $T'=0 \Rightarrow r_i = r_t$, the statement i belongs S_F set. 3) If $T'>0$, $r_i > r_t$, the statement i belongs S_B set. So we have the following two judgments.

a) When $x=F$, from the formula (3.3) and the formula (3.4), we can get:

$$\begin{cases} if\ T > 0 \Rightarrow T' > 0 \\ if\ T = 0 \Rightarrow T' = 0 \\ if\ T < 0 \Rightarrow T' < 0 \end{cases}$$

It means the category the statement i belonging to before and after applying the cloning strategy is unchanged.

b) When $x<F$, our analysis consists of two parts.

i) If $T \le 0 \Rightarrow r_i \le r_t$, from the formula (3.3) and the formula (3.4), we know that

$$\Rightarrow \begin{cases} T \;\; = x - F + z - y \le 0 \\ T' = c(x-F) - y + z \\ c > 1 \end{cases}$$
$$\Rightarrow \;\; T' < 0$$

It means before cloning, the statement *i* belongs to set S_A or S_F, and after cloning belongs to set S_A.

ii) If $T>0 \Rightarrow r_i > r_t$, according to the formula (3.3) and the formula (3.4),we know that,

$$\begin{cases} T \;= x - F + z - y \le 0 \\ T' = c(x-F) - y + z \\ \quad = (c-1)(x-F) + x - F + z - y \\ c > 1 \end{cases}$$

$$\Rightarrow \begin{cases} if(1 < c < \frac{y-z}{x-F}) \;\; T' > 0 \; => \; r_i > r_t \\ if(c = \frac{y-z}{x-F}) \;\;\; T' = 0 \; => \; r_i = r_t \\ if(c > \frac{y-z}{x-F}) \;\;\; T' < 0 \; => \; r_i < r_t \end{cases}$$

It means before cloning, the statement *i* belongs to set S_B, and after cloning belongs to set S_A, S_F or S_B.

In summary, under the following certain conditions, the effectiveness of Wong2 can be improved by applying the cloning strategy.

1) $\exists i(i \neq t)$, $\{i | a_{ef}^i < F\} \cap \{i | \; a_{ef}^i - F + a_{ep}^t - a_{ep}^i = 0 \}$. This conditions means there exist certain such kind of statements that belong to set S_F before cloning and belong to set S_A after cloning.

2) $\exists i(i \neq t)$, $\{i | a_{ef}^i - F + a_{ep}^t - a_{ep}^i > 0\} \cap \{i | a_{ef}^i < F\} \cap \{i | \; c > \frac{a_{ep}^i - a_{ep}^t}{a_{ef}^i - F} \}$. This conditions means there exist certain such kind of statements that belong to set S_B before cloning and belong to set S_A after cloning.

D. Summary

From the theoretical analysis in section 3, we draw the following conclusions. (1) By applying the cloning strategy on the Jaccard and Wong2 algorithms, their fault localization effectiveness can be improved under certain conditions. (2) By applying the cloning strategy on the Jaccard and Wong2 algorithms, their effectiveness will never be impaired.

IV. CONCLUSION

The effectiveness of spectrum-based fault localization techniques can be affected by the number of passed and failed test cases. Previous studies shown that a balanced test suite, which means containing approximately identical number of passed test cases and failed test cases, can mostly increase the fault localization effectiveness of such techniques.

In this paper, we propose a cloning strategy to produce a balanced test suite and from a theoretical perspective, analyze the impact of such a cloning strategy on two representative techniques Jaccard and Wong2. We show that applying the cloning strategy on the two techniques will improve, or preserve at least, their fault localization effectiveness.

If a fault is in a non-executable statement (such as the case of a code omission fault), to reflect the effectiveness of a technique, we follow previous studies (such as [13]) to mark the directly infected statement or an adjacent executable statement as the fault position, and apply the above metrics [14]. For simple fault programs, we have already proved this conclusion in theory, so there is no need for experimental verification.

The work in this paper will be improved in two steps. (i) The similar process of analysis will be conducted on similar formulas by using other cloning strategy. (ii) The similar process of analysis will be conducted under the multi-fault assumption

ACKNOWLEDGMENT

This research is supported in part by a grant from the National Science and Technology Major Project of the Ministry of Science and Technology of China (Grant No. 2012ZX01039-004) and grants from the Natural Science Foundation of China (project no. 61003027).

REFERENCES

[1] J. Jones and M. Harrold, "Empirical Evaluation Of The Tarantula Automatic Fault-Localization Technique". Proc. International Conference on Automated Software Engineering (ASE 2005), St. Louis, Missouri, USA, 2005, pp. 273-282.

[2] R. Abreu, P. Zoeteweij and V. Gemund, "An Evaluation Of Similarity Coefficients For Software Fault Localization". Proc. Pacific Rim International Symposium on Dependable Computing, Riverside, USA, 2006, pp. 39-46.

[3] W. Eric, T. Wei, Y. Qi and L. Zhao, "A Crosstab-Based Statistical Method For Effective Fault Localization". Proc International Conference on Software Testing (ICST 2008), Lillehammer, Norway, 2008, pp. 42-51.

[4] B. Liblit, M. Naik, X. Zheng, A. Aiken and M. I. Jordan, "Scalable Statistical Bug Isolation". Proc. ACM SIGPLAN Conference on Programming Language Design and Implementation, Chicago, IL, USA, 2005, pp. 15-26.

[5] L. Naish, J. L. Hua and K. Ramamohanarao, "A Model For Spectra-Based Software Diagnosis". ACM Transactions on Software Engineering and Methodology, 2011, 20(3): 11.

[6] Z. Zhang, W. K. Chan, T. H. Tse, Y. T. Yu and P. Hu, "Non-Parametric Statistical Fault Localization". Journal of Systems and Software, 2011, 84(6), pp. 885-905.

[7] K. J. Baah, A. Podgurski and M. Harrold, "Causal Inference For Statistical Fault Localization". Proc. International Symposium on Software Testing and Analysis, Trento, Italy, 2010, pp. 73-83.

[8] V. Debroy and W. Eric, "Insights On Fault Interference For Programs With Multiple Bugs". Proc. International Symposium on Software Reliability Engineering, Mysuru, India, 2009, pp. 165-174.

[9] C. Gong, Z. Zheng, W. Li, P. Hao, "Effects of class imbalance in test suites: an empirical study of spectrum-based fault localization". Proc. International Computer Software and Applications Conference (COMSAC 2012), Izmir, Turkey, 2012, pp. 470-475.

[10] D. Hao, Y. Pan, L. Zhang, W. Zhao, H. Mei and J. Sun, "A Similarity-aware Approach to Testing Based Fault Localization".Proc. International Conference on Software Engineering (ICSE 2005). St. Louis, Missouri, USA. 2005, pp. 291-294.

[11] B. Benoit, F. Franck and Y. Traon, "Improving Test Suites for Efficient Fault Localization". Proc. International Conference on Software Engineering (ICSE 2006), Shanghai, China.2006, pp.82-92.

[12] Y. Yu, J. Jones and M. Harrold, "An Empirical Study of the Effects of Test-Suite Reduction on Fault Localization". Proc. International Conference on Software Engineering (ICSE 2008). Leipzig, Germany, 2008, pp. 201-211.

[13] D. Jeffrey, N. Gupta, and R. Gupta, "Fault localization using value replacement". Proc. International Symposium on Software Testing and Analysis (ISSTA 2008), New York, USA. 2008, pp. 167–178.

[14] Z. Zhang, W. K Chan, T. H. Tse, B. Jang and X. Wang, "Capturing Propagation of Infected Program States". Proc. European software engineering conference (ESEC/FSE'09), New York, USA, 2009, pp. 24–28.