Global Journal of Computer Science and Technology: G
Interdisciplinary
Volume 20 Issue 2 Version 1.0 Year 2020
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

Spectrum-based Fault Localization Techniques Application on Multiple-Fault Programs: A Review

By Abubakar Zakari, Shamsu Abdullahi, NuraModi Shagari, Abubakar Bello Tambawal, Nuruddeen Musa Shanono, Jaafar Zubairu Maitama, Rasheed Abubakar Rasheed, Alhassan Adamu & Salish Mamman Abdulrahman

Kano University of Science and Technology

Abstract- Software fault localization is one of the most tedious and costly activities in program debugging in the endeavor to identify faults locations in a software program. In this paper, the studies that used spectrum-based fault localization (SBFL) techniques that makes use of different multiple fault localization debugging methods such as one-bug-at-a-time (OBA) debugging, parallel debugging, and simultaneous debugging in localizing multiple faults are classified and critically analyzed in order to extensively discuss the current research trends, issues, and challenges in this field of study. The outcome strongly shows that there is a high utilization of OBA debugging method, poor fault isolation accuracy, and dominant use of artificial faults that limit the existing techniques applicability in the software industry.

Keywords: software fault localization, fault interference, fault isolation, program debugging, multiple faults.

GJCST-G Classification: D.2.M

SPECTRUMBASEDFAULTLOCALIZATIONTECHNIQUESAPPLICATIONONMULTIPLEFAULTPROGRAMSAREVIEW

Spectrum-based Fault Localization Techniques Application on Multiple-Fault Programs: A Review

Abubakar Zakari [α], Shamsu Abdullahi [σ], NuraModi Shagari [ρ], Abubakar Bello Tambawal [Ѡ], Nuruddeen Musa Shanono[¥], Jaafar Zubairu Maitama[§], Rasheed Abubakar Rasheed [χ], Alhassan Adamu [ν] & Salish Mamman Abdulrahman [ϴ]

Abstract- Software fault localization is one of the most tedious and costly activities in program debugging in the endeavor to identify faults locations in a software program. In this paper, the studies that used spectrum-based fault localization (SBFL) techniques that makes use of different multiple fault localization debugging methods such as one-bug-at-a-time (OBA) debugging, parallel debugging, and simultaneous debugging in localizing multiple faults are classified and critically analyzed in order to extensively discuss the current research trends, issues, and challenges in this field of study. The outcome strongly shows that there is a high utilization of OBA debugging method, poor fault isolation accuracy, and dominant use of artificial faults that limit the existing techniques applicability in the software industry.

Keywords: software fault localization, fault interference, fault isolation, program debugging, multiple faults.

I. Introduction

In recent years, advances in software development have led to the increase in complexity of software programs, which adversely resulted in a rise in software failures [1]. The introduction of these failures in a software program due to increasing complexity has negative impacts on software quality, and this has been attributed to the lack of software conformance to it defined requirements [2]. Effective fault localization is important, as 50% to 80% of development and maintenance costs are spent in the debugging process that involves failure detection, fault localization, as well as fault repair [3, 4]. Furthermore, this process (fault localization) is also considered as one of the most tedious, time-consuming, and costly activities in the debugging process [3]. In the past few decades, fault localization has received much research attention, notably because the process tends to be difficult when conducted manually [5-7]. Manual fault localization techniques are costly especially when applied in large-scale software programs that have thousands or millions of lines of code [8].

In order to address the issues of manual fault localization techniques, researchers have proposed various automated fault localization techniques [9-24]. Some techniques exploited program execution behavior whilst others attempted to build models to explain program failure [12, 25]. Hence, most of these techniques are proven to be helpful in facilitating software development and maintenance process especially on single-fault programs [26]. Although empirical studies revealed that failure in programs can be caused by multiple faults [11, 27], most existing studies localize faults based on the assumption that a program has a single fault [28]. Consequently, this presumption adversely impacts the effectiveness of fault localization due to the possibility of having more than one fault in a faulty program [29, 30]. Principally, this is due to fault interference, a phenomenon which plays a major role in the reduction of fault localization techniques effectiveness in the context of multiple fault. Previous studies [27, 29, 31-33] have empirically investigated this phenomenon and its effects on fault localization inferencing.

As for related work, Parmar et al. [34] surveyed few automated fault localization techniques extensively where most of the techniques reviewed were statistical-based, which are focused on localizing single faults. Similarly, another study [35] surveyed some of the most important techniques and approaches in the domain of software fault localization, to give readers an overview of progress made in the field of research. However, the study also does not review works related to multiple fault localization. Moreover, Wong et al. [1, 36] conducted an extensive general survey on software fault localization and highlighted both traditional and advanced software fault localization techniques holistically. Furthermore, issues and challenges facing both single fault localization and multiple fault localization were highlighted in general. Likewise, a previous study [37] surveyed and categorized some of the most important techniques for automated fault localization and some

*Author α ¥ § ν ϴ: Department of Computer Science, Kano University of Science and Technology, Wudil, P.M.B 3244, Kano, Nigeria.
e-mail: abubakar.zakari@yahoo.com
Author σ ρ Ѡ χ: Faculty of Computer Science and Information Technology, University of Malaya, 50603, Kuala Lumpur, Malaysia.
Author χ: Department of Information Technology, Faculty of Computer Science and Information Technology, Bayero University Kano, 3011 BUK Kano, Nigeria.*

challenges in the field of study were also highlighted. The study also does not address multiple fault localization. Additionally, Perez et al. [38] investigated automated fault localization techniques in relation to single fault localization. Another study [39] conducted a survey on the state-of-the-art Spectrum-based fault localization techniques (SBFL) with respect to cost and quantity of faults. The study highlighted the recent advances and challenges on SBFL research. However, studies on multiple fault localization were not highlighted. Zakari et al. [40] conducted a survey on software fault localization techniques. The study highlights some issues and limitations in the field of study. Additionally, Zakari et al. [41] conducted a systematic mapping study on software fault localization to highlights the recent trends in the research domain. Overall, even though these studies investigated fault localization holistically, there has been limited or no studies conducted to review studies that use SBFL techniques for multiple fault localization.

In order to address this gap, we conducted a review to analyse, classify and critically investigate studies on multiple fault localization that are specifically based on SBFL techniques. Based on our methodology in Section 2, 30 studies are selected for this study. This survey is essential so that software engineers and testers will be able to deeply understand the field of study. Additionally, through this survey, researchers would be able to identify research issues and challenges to eventually propose effective solutions.

The remaining part of this paper is organized into different sections. Section 2 highlights the research methodology. Section 3 gives the discussion. Section 4 presents the issues and challenges. The study is concluded in Section 5.

II. Research Methodology

In this section, the methodology adopted for paper selection is presented to aid in selecting the most suitable papers in the research area. Papers on multiple fault localization that strictly used the SBFL technique were selected for this study. This is important so as to narrow down the review space to a more define problem space and to also select important papers. The systematic methodology was adopted following the guidelines of Kitchenham [42].

a) Search Criteria

In order to select the papers for this survey, a search was conducted on various digital library sources to not miss out on relevant papers. In this process, the following digital libraries were selected; IEEE Xplore, ACM, and Springer. Primarily, search criteria were conducted by composing a search query. This involved inclusion of important terms, keywords, and their synonyms based on the purpose of this paper. Hence, only peer-reviewed articles were targeted. The search string used in our former study is adopted [41].

The earliest selected study was published in 2011 and the last date was set to 2018 in order to confirm that all related relevant papers within this period are included. Based on Kitchenham [42] recommendation, only papers written in English were considered. Additionally, the search is narrowed down to only papers that address the issue of multiple fault localization and also utilized the SBFL technique for fault localization. Therefore, papers that do not utilized SBFL technique were excluded, also, survey/review papers were also not considered. Based on these criteria, 1160 potential papers were initially collected.

b) Paper Selection Strategy

Based on the above defined search criteria, a three-stage paper selection strategy shown in Figure 1 was conducted, as follows:

Stage 1: In this stage, 1160 potential papers were thoroughly checked to remove any duplicates. However, a large number of irrelevant papers were also observed due to conflict between topics. For instance, faults and localization terms are related to topics in electrical engineering research field or can be related to other fields in physics and telecommunications. Finally, after Stage 1, 350 papers were considered.

Stage 2: In this stage, the abstracts of the 350 selected papers were checked based on the purpose of this paper. In this process, papers were classified based on their application on multiple-fault programs and the basic techniques utilized. As a result, 120 papers were obtained.

Stage 3: In this stage, the research team read the full text of all the 120 papers. Out of these, 30 papers were found to directly relate to the purpose of this paper.

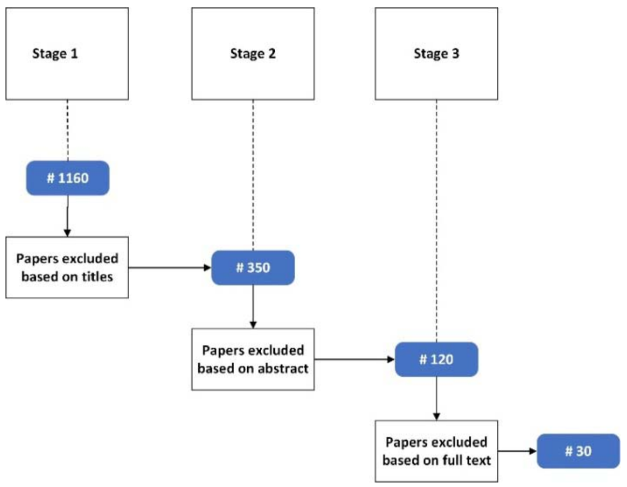
Stage 1
Stage 2
Stage 3
# 1160
Papers excluded based on titles
# 350
Papers excluded based on abstract
# 120
Papers excluded based on full text
# 30

Figure 1: Flowchart of research methodology

III. Discussion

In this section, the selected papers are critically analyzed. In addition, recent trends were identified in the field of study.

a) *Studies on Fault Interference Phenomenon*

Fault interference is a phenomenon that alters the behavioral normality of tests execution when more than one fault exists in a program under test. This phenomenon is inevitable in a multiple fault scenario. Existing studies [29, 31] showed the high occurrence of both "constructive interference" and "destructive interference" where the latter is the most prevalent. A study by [33] also has nearly the same results based on an experiment on object-oriented Java programs, whereby "destructive interference" has the most prevalence. Constructive interference occurs when a test case that passed in the presence of single fault fails in the presence of multiple faults. On the other hand, destructive interference takes place when a test case that failed in the presence of single fault passes in the presence of multiple faults. This also implies that the higher the number of faults, the higher the frequency of interferences. It was observed that test cases that failed on a multiple-fault program might not be enough to effectively localize faults. Hence, most existing studies observed the reduction in effectiveness especially when using the SBFL techniques for fault localization. However, most of the existing studies report that a single fault can be localized with relatively good effectiveness [29, 43].

Moreover, the existing studies found out that the more faults a program has, the more interference occurs. This means that faults will be tough to localize if a program has many faults. We have identified four studies from our selected papers that investigate the fault interference phenomenon. These studies showed the impact of multiple faults on localization inferencing.

Table 1: Investigative studies on fault interference

Source	Reference	Year	Fault interference	The most prevalent
IEEE (ICSM)	[32]	2011	√	Destructive interference
ACM (ISSTA)	[27]	2011	√	Destructive interference
IEEE (ISESEM)	[33]	2013	√	Destructive interference
Springer (ESE)	[29]	2015	√	Destructive interference

The studies highlighted in Table 1 are all the investigative studies done from 2011 to access the impact of fault interference on localization inferencing on programs with multiple faults, as well as to identify which type of interference is the most prevalent. Practically, the studies in Table 1 have all been conducted using the One-bug-at-a-time (OBA) debugging method. These studies do not only show that interference occurs but clearly shows the disadvantages of utilizing an OBA method for multiple fault localization. We observed that all the studies on fault interference phenomenon have concluded that destructive interference is the most prevalent.

b) *Classification of Debugging Methods Utilized in Localizing Multiple Faults across the Selected Studies*

In this section, debugging methods were identified that are used to localize multiple faults. Hence, the selected papers were classified based on the method utilized. We have identified three prominent debugging methods that are used in localizing multiple faults from the selected papers which are OBA debugging method, parallel debugging method, and simultaneous debugging method. Table 2 shows that 80% of the selected papers used the OBA method, followed by parallel debugging method with 16.7%, and simultaneous method with 3.3%.

Table 2: Distribution of multiple fault localization methods

Method	Papers	%
One-bug-at-a-time (OBA) debugging	[27, 29, 32, 33, 43-62]	80%
Parallel debugging	[17, 63-66]	16.7%
Simultaneous debugging	[67]	3.3%

This shows that the OBA method is the most utilized among the three identified debugging methods. This indicates that most of the studies utilizing the SBFL technique used the OBA method for multiple fault localization, with few studies adopting both parallel debugging method and simultaneous debugging method.

c) *Fault Types that are Utilized in All the Selected Studies*

Fault types play a vital role in software fault localization research, especially in localizing multiple faults. Fault types are the type of faults used in the selected papers to generate multiple-fault versions. There are two main categories of faults in software fault localization research domain, which are real faults and artificial faults. This section will highlight statistically the use of faults types by our selected papers.

Artificial faults are faults that are manually seeded or created using mutation-based fault injection techniques in order to create program versions with many faults. Moreover, real faults are real world faults that naturally resides in the program under test without human interference in adding it. A recent study [68] evaluated the effectiveness of existing fault localization formulae by using both artificial faults and real faults. The result of the study shows that the outcome of a fault localization technique used in programs with artificial faults is insignificant as compared to the same experiment on programs with real faults. This also implies that generalization of fault localization results based on programs with artificial faults is not realistic as compared to results on the same programs containing real faults. However, we observed that artificial faults are the most commonly used faults in the selected papers despite its disadvantages. Therefore, this undermines the generalization of the existing studies results in the software industry [1].

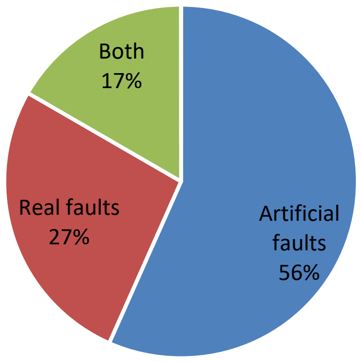

Figure 2: Fault utilization across all the selected studies

Figure 2 shows the distribution of fault utilization across all the selected papers. From the figure, we observed that 56% of the studies used artificial faults, 27% used real faults, while 17% used both real and artificial faults in their experiments, respectively.

d) *The Evaluation Metrics Utilized Across the Selected Studies*

Evaluation metrics are standard metrics used in assessing the effectiveness of a given software fault localization technique. Table 3 shows the list of identified evaluation metrics used from the selected papers. Four key evaluation metrics were identified, namely Exam score, Expense score, Wasted effort, and Precision & recall. Expense score is defined as the percentage of code that a programmer needs to examine so as to find only the first bug in a multiple-fault program [8]. On the other hand, Exam score is defined as a percentage of executable statements that needs to be examined to find a fault [43]. Also, Wasted effort was defined by Abreu et al. [69] as the percentage of non-faulty program statements that were checked before the faulty statement is found. And, Precision & recall refers to the number and ratio of lines of code that are identified to be faulty with respect to the overall program lines of code.

Out of the 30 selected papers, 33.3% of the studies use Expense score, 23.3% use Exam score, 13.3% use Wasted effort, and lastly 10% use Precision & recall. However, for 6 (20%) studies, the evaluation metric used was not clear. Therefore, the findings show that Expense score and Exam score are the most utilized by the selected papers, respectively.

Table 3: Evaluation metrics utilized across all selected studies

Evaluation metrics	Studies
Expense score	[17, 27, 29, 33, 52-57]
Exam score	[43, 49-51, 63, 64, 67]
Wasted effort	[46-48, 65]
Precision & recall	[44, 45, 66]

e) *Fault Isolation*

Fault isolation is the process of isolating faults caused by different failures into separate clusters for efficient and more effective multiple fault localization. Most of the selected papers that utilized method such as parallel debugging used various clustering algorithms to isolate faults.

From the selected papers, k-mean clustering has shown to be better than most of the existing clustering algorithms used for isolating faults [63]. Clustering algorithms have contributed immensely to fault isolation [70], which further aids in localizing multiple faults. From the selected studies, five works were found to utilize various clustering algorithms for fault isolation, thus representing 16.7% of the selected studies [17, 63-66]. This trend shows the importance of clustering algorithms in multiple fault localization.

IV. Issues & Challenges

While investigating the selected papers, different research issues and challenges were identified. Highlighting these issues and challenges is important and is expected to help researchers to further address them.

Firstly, fault interference is no doubt an inevitable factor as it occurs when more than one fault exists in a software program. This phenomenon reduces fault localization techniques inferencing due to fault-to-failure complexity [29]. With the utilization of method such as parallel debugging, this phenomenon has been subsided with the aid of clustering algorithms for fault isolation. However, various studies indicated the lack of accuracy of these algorithms in isolating faults [11, 63]. Perhaps, better clustering algorithms are needed to resolve this issue which will help reduce the impact of fault interference and improve localization effectiveness.

OBA debugging method has gained a lot of attention in recent years particularly among studies utilizing the SBFL technique, with 80% of the selected papers utilizing the method. Various studies have hinted on the shortcomings in using the method for localizing multiple faults, because more time needs to be spent in the localization process and additional faults might be introduced in the software program [20, 29, 33]. However, its increased usage is alarming, with methods design to solve the problem having less attention such as simultaneous method with 3.3% contribution, and parallelization method with 16.7% contribution. Hence, in order to improve localization effectiveness in multiple fault context, more studies in these two methods (parallel and simultaneous methods) is of eminent importance.

Furthermore, artificial faults are often used to replicate real faults behavior. These faults are manually seeded or inserted using mutation-based fault injection techniques in a software program. As depicted in Figure 2, most of the studies are using artificial faults (56%), but this trend can be associated with the high usage of standard subject programs such as Siemens suite programs. This trend is a concern because Siemens suite programs contain single faults by default [69]. Therefore, a researcher has to seed the faults (artificial fault) to create multiple-fault versions. This process is expected to cause bias in the whole fault localization process and raise many questions on the credibility of the fault localization technique in the software industries. Hence, more utilization of real programs with real faults can aid in generating credible fault localization results and further encourage the use of fault localization techniques in the software industry.

Moreover, on fault isolation, looking at the importance of clustering algorithms in the isolation of multiple faults and the lack of accuracy affecting the existing algorithms utilized in the literature, exploring machine learning algorithms might improve fault isolation accuracy and enhance both effectiveness and efficiency in the fault localization process in the research domain. Overall, multiple fault localization research area has gain reasonable attention in the last decade. However, for the research area to progress, the highlighted issues and challenges need to be resolved with novel or enhanced solutions.

V. Conclusion

Over the years, software has become larger and more complex with fault localization being even more difficult than ever before. Fault localization has become even more challenging when applied to software programs with multiple faults, particularly when using the one-bug-at-a-time (OBA) debugging method. Multiple faults reduce the efficacy of the existing fault localization techniques due to fault interference phenomenon. However, the utilization of OBA method will increase software time-to-delivery and also bring more faults to the program under test. Researchers have proposed various techniques and methods to tackle this problem and provide an environment for developers to localize multiple faults simultaneously. Methods such as simultaneous and parallelization have been used to help solve these issues. However, with all the research efforts, the localization effectiveness and fault isolation accuracy are still not optimal. In this study, 30 papers from 2011-2018 based on multiple fault localization using spectrum-based fault localization (SBFL) techniques were extensively reviewed. Additionally, trends, issues, and challenges were identified and discussed to further help researchers get a holistic understanding of the field of study.

Based on the obtained results, research on multiple fault localization using the SBFL technique has gained momentous attention in the last decade. Key findings relate to fault interference, multiple fault debugging methods, fault types, evaluation metrics utilized, and fault isolation were identified and explored. Furthermore, the use of artificial faults to access a fault localization technique effectiveness does not depict real industrial reality. Artificial faults are dominantly used by our selected in multiple fault localization research (See Figure 2) particularly due to the high utilization of the Siemens suite programs by the selected studies. Therefore, addressing these issues are crucial to the application of the existing multiple fault localization techniques in the software industry.

References Références Referencias

1. Wong, W.E., et al., A survey on software fault localization. IEEE Transactions on Software Engineering, 2016. 42(8): p. 707-740.
2. Zakari, A., A.A. Lawan, and G. Bekaroo. A Hybrid Three-Phased Approach in Requirement Elicitation. in International Conference on Emerging Trends in Electrical, Electronic and Communications Engineering. 2016. Springer.
3. Collofello, J.S. and S.N. Woodfield, Evaluating the effectiveness of reliability-assurance techniques. Journal of systems and software, 1989. 9(3): p. 191-195.
4. Zakari, A. and S.P. Lee. Simultaneous Isolation of Software Faults for Effective Fault Localization. in 2019 IEEE 15th International Colloquium on Signal Processing & Its Applications (CSPA). 2019. IEEE.
5. Agrawal, H., R.A. DeMillo, and E.H. Spafford, Efficient Debugging with Slicing and Backtracking.

6. Rosenblum, D.S., A practical approach to programming with assertions. IEEE transactions on Software Engineering, 1995. 21(1): p. 19-31.
7. Hennessy, J., Symbolic debugging of optimized code. ACM Transactions on Programming Languages and Systems (TOPLAS), 1982. 4(3): p. 323-344.
8. Yu, Y., J.A. Jones, and M.J. Harrold. An empirical study of the effects of test-suite reduction on fault localization. in Proceedings of the 30th international conference on Software engineering. 2008. ACM.
9. Jones, J.A., M.J. Harrold, and J. Stasko. Visualization of test information to assist fault localization. in Proceedings - International Conference on Software Engineering. 2002.
10. Zheng, A.X., et al. Statistical debugging of sampled programs. in Advances in Neural Information Processing Systems. 2003.
11. Jones, J.A., J.F. Bowring, and M.J. Harrold. Debugging in parallel. in Proceedings of the 2007 international symposium on Software testing and analysis. 2007. ACM.
12. Wong, E., et al. A crosstab-based statistical method for effective fault localization. in Software Testing, Verification, and Validation, 2008 1st International Conference on. 2008. IEEE.
13. Abreu, R., P. Zoeteweij, and A.J. van Gemund. Localizing software faults simultaneously. in 2009 Ninth International Conference on Quality Software. 2009. IEEE.
14. Jeffrey, D., N. Gupta, and R. Gupta. Effective and efficient localization of multiple faults using value replacement. in Software Maintenance, 2009. ICSM 2009. IEEE International Conference on. 2009. IEEE.
15. Wong, W.E. and Y. Qi, BP NEURAL NETWORK-BASED EFFECTIVE FAULT LOCALIZATION. International Journal of Software Engineering and Knowledge Engineering, 2009. 19(4): p. 573-597.
16. Lamraoui, S.-M. and S. Nakajima, A Formula-based Approach for Automatic Fault Localization of Multi-fault Programs. Journal of Information Processing, 2016. 24(1): p. 88-98.
17. Sun, X., et al., IPSETFUL: an iterative process of selecting test cases for effective fault localization by exploring concept lattice of program spectra. Frontiers of Computer Science, 2016. 10(5): p. 812-831.
18. Wang, X. and Y. Liu, Fault localization using disparities of dynamic invariants. Journal of Systems and Software, 2016. 122: p. 144-154.
19. Baudry, B., F. Fleurey, and Y. Le Traon. Improving test suites for efficient fault localization. in Proceedings - International Conference on Software Engineering. 2006.
20. Jones, J.A. and M.J. Harrold. Empirical evaluation of the tarantula automatic fault-localization technique. in 20th IEEE/ACM International Conference on Automated Software Engineering, ASE 2005. 2005.
21. Dickinson, W., D. Leon, and A. Podgurski. Finding failures by cluster analysis of execution profiles. in Proceedings of the 23rd international conference on Software engineering. 2001. IEEE Computer Society.
22. Renieres, M. and S.P. Reiss. Fault localization with nearest neighbor queries. in Automated Software Engineering, 2003. Proceedings. 18th IEEE International Conference on. 2003. IEEE.
23. Perez, A., R. Abreu, and A. van Deursen. A test-suite diagnosability metric for spectrum-based fault localization approaches. in Proceedings of the 39th International Conference on Software Engineering. 2017. IEEE Press.
24. Kang, D., J. Sohn, and S. Yoo. Empirical evaluation of conditional operators in GP based fault localization. in Proceedings of the Genetic and Evolutionary Computation Conference. 2017. ACM.
25. Naish, L., H.J. Lee, and K. Ramamohanarao, A model for spectra-based software diagnosis. ACM Transactions on software engineering and methodology (TOSEM), 2011. 20(3): p. 11.
26. Abreu, R. and P. Zoeteweij. An evaluation of similarity coefficients for software fault localization. in 2006 12th Pacific Rim International Symposium on Dependable Computing (PRDC'06). 2006. IEEE.
27. DiGiuseppe, N. and J.A. Jones. On the influence of multiple faults on coverage-based fault localization. in Proceedings of the 2011 international symposium on software testing and analysis. 2011. ACM.
28. Zheng, W., D.S. Hu, and J. Wang, Fault Localization Analysis Based on Deep Neural Network. Mathematical Problems in Engineering, 2016. 2016.
29. DiGiuseppe, N. and J.A. Jones, Fault density, fault types, and spectra-based fault localization. Empirical Software Engineering, 2015. 20(4): p. 928-967.
30. Zakari, A., S.P. Lee, and I.A.T. Hashem, A Community-Based Fault Isolation Approach for Effective Simultaneous Localization of Faults. IEEE Access, 2019. 7: p. 50012-50030.
31. Debroy, V. and W.E. Wong. Insights on fault interference for programs with multiple bugs. in 2009 20th International Symposium on Software Reliability Engineering. 2009. IEEE.
32. DiGiuseppe, N. and J.A. Jones. Fault interaction and its repercussions. in Software Maintenance (ICSM), 2011 27th IEEE International Conference on. 2011. IEEE.
33. Xue, X. and A.S. Namin. How Significant is the Effect of Fault Interactions on Coverage-Based Fault Localizations? in 2013 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. 2013. IEEE.

34. Parmar, P. and M. Patel, Software Fault Localization: A Survey. International Journal of Computer Applications, 2016. 154(9).
35. PAL, D. and R. MOHIUDDIN, Automated Bug Localization of Software Programs: A Survey Report1. 2013.
36. Wong, W.E. and V. Debroy, A survey of software fault localization. Department of Computer Science, University of Texas at Dallas, Tech. Rep. UTDCS-45, 2009. 9.
37. Alipour, M.A., Automated fault localization techniques: a survey. Oregon State University, 2012.
38. Perez, A., R. Abreu, and E. Wong, A Survey on Fault Localization Techniques. Cited on pages iv and, 2014. 88.
39. Souza, H.A., M.L. Chaim, and F. Kon, Spectrum-based Software Fault Localization: A Survey of Techniques, Advances, and Challenges. arXiv preprint arXiv:1607.04347, 2016.
40. ZakariA, A. and S.P. LeeA, Software Fault Localization: Issues and Limitations. 2017.
41. Zakari, A., et al., Software Fault Localization: A Systematic Mapping Study. IET Software, 2018.
42. Kitchenham, B., et al., Systematic literature reviews in software engineering–a systematic literature review. Information and software technology, 2009. 51(1): p. 7-15.
43. Wong, W.E., et al., The DStar Method for Effective Software Fault Localization. Ieee Transactions on Reliability, 2014. 63(1): p. 290-308.
44. Yu, Z.X., et al., Does the Failing Test Execute a Single or Multiple Faults? An Approach to Classifying Failing Tests. 2015 Ieee/Acm 37th Ieee International Conference on Software Engineering, Vol 1, 2015: p. 924-935.
45. Zhang, X.-Y., Z. Zheng, and K.-Y. Cai, Exploring the usefulness of unlabelled test cases in software fault localization. Journal of Systems and Software, 2018. 136: p. 278-290.
46. Zhang, M., et al. Boosting spectrum-based fault localization using PageRank. in Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis. 2017. ACM.
47. Laghari, G., A. Murgia, and S. Demeyer. Fine-tuning spectrum based fault localisation with frequent method item sets. in Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering. 2016. ACM.
48. Neelofar, N., et al., Improving spectral-based fault localization using static analysis. Software: Practice and Experience, 2017.
49. Zhang, L., et al., A theoretical analysis on cloning the failed test cases to improve spectrum-based fault localization. Journal of Systems and Software, 2017. 129: p. 35-57.
50. Ghandehari, L.S., et al., A Combinatorial Testing-Based Approach to Fault Localization. IEEE Transactions on Software Engineering, 2018.
51. Xu, J., et al., A general noise-reduction framework for fault localization of Java programs. Information and Software Technology, 2013. 55(5): p. 880-896.
52. Neelofar, N., L. Naish, and K. Ramamohanarao, Spectral-based fault localization using hyperbolic function. Software: Practice and Experience, 2018. 48(3): p. 641-664.
53. Lucia, L., et al., Extended comprehensive study of association measures for fault localization. Journal of software: Evolution and Process, 2014. 26(2): p. 172-219.
54. Perez, A., R. Abreu, and A. Riboira, A dynamic code coverage approach to maximize fault localization efficiency. Journal of Systems and Software, 2014. 90: p. 18-28.
55. Naish, L. and K. Ramamohanarao. Multiple Bug Spectral Fault Localization Using Genetic Programming. in Software Engineering Conference (ASWEC), 2015 24th Australasian. 2015. IEEE.
56. Fu, W., et al. Test case prioritization approach to improving the effectiveness of fault localization. in Software Analysis, Testing and Evolution (SATE), International Conference on. 2016. IEEE.
57. Lee, J., J. Kim, and E. Lee. Enhanced Fault Localization by Weighting Test Cases with Multiple Faults. in Proceedings of the International Conference on Software Engineering Research and Practice (SERP). 2016. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (World Comp).
58. Sun, S.-F. and A. Podgurski. Properties of effective metrics for coverage-based statistical fault localization. in Software Testing, Verification and Validation (ICST), 2016 IEEE International Conference on. 2016. IEEE.
59. Xia, X., et al., " Automated debugging considered harmful" considered harmful: A user study revisiting the usefulness of spectra-based fault localization techniques with professionals using real bugs from large systems. 2016.
60. Perez, A., R. Abreu, and M. D'Amorim. Prevalence of Single-Fault Fixes and Its Impact on Fault Localization. in Software Testing, Verification and Validation (ICST), 2017 IEEE International Conference on. 2017. IEEE.
61. Xiaobo, Y., L. Bin, and L. Jianxing. The Failure Behaviors of Multi-Faults Programs: An Empirical Study. in Software Quality, Reliability and Security Companion (QRS-C), 2017 IEEE International Conference on. 2017. IEEE.
62. Wang, Y., et al., Spectrum-Based Fault Localization via Enlarging Non-Fault Region to Improve Fault

Absolute Ranking. IEEE ACCESS, 2018. 6: p. 8925-8933.

63. Huang, Y., et al. An empirical study on clustering for isolating bugs in fault localization. in Software Reliability Engineering Workshops (ISSREW), 2013 IEEE International Symposium on. 2013. IEEE.
64. Steimann, F. and M. Frenkel. Improving coverage-based localization of multiple faults using algorithms from integer linear programming. in 2012 IEEE 23rd International Symposium on Software Reliability Engineering. 2012. IEEE.
65. Högerle, W., F. Steimann, and M. Frenkel. More Debugging in Parallel. in 2014 IEEE 25th International Symposium on Software Reliability Engineering. 2014. IEEE.
66. Wang, Y., et al., WAS: A weighted attribute-based strategy for cluster test selection. Journal of Systems and Software, 2014. 98: p. 44-58.
67. Zheng, Y., et al., Localizing multiple software faults based on evolution algorithm. Journal of Systems and Software, 2018. 139: p. 107-123.
68. Pearson, S., et al. Evaluating and improving fault localization. in Proceedings of the 39th International Conference on Software Engineering. 2017. IEEE Press.
69. Abreu, R., P. Zoeteweij, and A.J. Van Gemund, Simultaneous debugging of software faults. Journal of Systems and Software, 2011. 84(4): p. 573-586.
70. Zhang, D., J. Jiang, and L. Chen. Program behavior characterization and clustering: An empirical study for failure clustering. in Software Reliability Engineering Workshops (ISSREW), 2013 IEEE International Symposium on. 2013. IEEE.