Software Fault Localization: A Survey

Priya Parmar
Department Of Information Technology
G H Patel Collage of Engineering And Technology
Gujarat, India

Miral Patel
Department Of Information Technology
G H Patel Collage of Engineering
And Technology
Gujarat, India

ABSTRACT

Now a day's Software is required in all professional disciplines. Hence, it is required to check software for fault localization to maintain the software quality. Software fault localization is an activity of finding the locations of fault in a program. Considering the increasing complexity of software, manual fault localization is not feasible, there is a firm necessity for techniques which can lead software developers to the location of faults with essential interference. Scientists and analysts have designed many different methods for locating software faults in the past few years, which aims to make it more adequate by ambushing the problem in a unique way. This paper gives a comprehensive review of various methods and techniques for locating faults that have been proposed in such valuable published resource.

General Terms

Suspicious code, Survey.

Keywords

Fault Localization, Debugging, Software Testing.

1. INTRODUCTION

The Software is very fundamental in our lives. It is used everywhere like in the Aerospace, Medical, Telecommunication, Government Systems, Industries etc. A 2006 report from NIST (National Institute of Standards and Technology) is that software errors are costing the U.S. economy $59.5 billion annually; the cost has grown since then. Without proper Testing of software the faults can't be resolved. An unresolved fault leads to system failure with huge losses. Softwares like safety critical system failure's loss is limited not only to financial loss but, also to loss of life. Despite great advances in software development and testing, software is still far from bug-free. A development of 100% bug-free software is not possible. Automated localization of bugs is one of the essential issues in debugging. It has been observed that Debugging is more expensive and time consuming. Software Testing cost more than its development. It is required to maintain the quality of software. First it was done manually, but to reduce the time and expense researchers have proposed various methods which are automated. Still, they are not fully automated. Software fault localization is used to make the debugging process easier because it takes more efforts and resources.

There are different categories of software testing methods to test software, in which one class can be characterized as Static analysis and Dynamic analysis. Static analysis can detect program defects through checking real code. On other hand dynamic analysis executes the program code. There are many techniques for fault localization. One common way of finding fault is insert print statement around the suspicious code. This is subjective and not an ideal technique to find faults. The main aim is to automate the process of finding faults to relieve programmers from tedious debugging work. This paper includes many advanced fault localization techniques. Classification of techniques includes slice based technique,

Program spectrum based techniques, statistics-based techniques, machine learning-based techniques, program stat-based techniques, model based techniques, data mining based techniques, similarity-based techniques, artificial intelligence-based techniques. This study has been published in various academic journals and conferences, including 'IEEE', 'ACM', 'Springer', 'Journal of Automated Software Testing', 'IEEE/ACM International Conference on Automated Software Engineering'.

This paper is arranged as follows. First, describe the scope and motivation, then the Analysis and Discussion, acknowledgement and conclusion.

2. SCOPE AND MOTIVATION

Software fault localization is one of the most time consuming and expensive activity for debugging the program. Therefor, there is a high demand for automated fault localization techniques that can reduce human efforts and time. This area is very much popular among researchers these days. This paper provides a survey on software fault localization that have been published in various journals and conferences from 2003 to 2016 July. This survey can help researchers to study various techniques and tools that are used and the results that they get in order to do more advance research in the future.

3. ANALYSIS AND DISCUSSIO

Evaluation of the papers are based on methods used for fault localization, comparison with other techniques, different tools used and the dataset used.

Table 1. Presents programs used to study the effectiveness of different fault localization. Several programs have frequently been used by researchers for fault localization experiments. Among them is Siemens suite, gcc, space, grep, gzep, make, unix suit and NanoXML. Other programs were used only once. They are of different languages like C, C++,

Java, PHP and in various sizes (Small, medium, Large). Another important point is that most of the bug used in the experiments are artificially injected bugs. Some of them also have used real bugs. There are tools like Aollo, Daikon, BARINEL etc. are required to support automatic or semi-automatic suspiciousness computation. The programs and tools used for the study are open source or openly accessible, but the source code is not available.

All the papers are sorted by year and the result is displayed in Figure 1. As shown in figure number of papers published over the years grew after 2003. It indicates that researchers are more and more interested in the area of software fault localization over 10 years.

Observe from the literature review is that to increase the effectiveness of fault localization researchers have been used combination of multiple techniques at the same time. They have used best of multiple techniques and achieve more accuracy.

Table 1. Shows that number of papers in each category differ from each other. It indicates that researchers interest shift from one category from another as time changes. For example, Slicing based techniques were popular before 2007, whereas Information Retrieval based and Program Spectra based techniques are popular since then. These techniques in each category have their own advantages and disadvantages.

Figure 2. Show papers published in IEEE, ACM, IEEE/ACM on software fault localization from 2003 to July 2016.

Figure 3. Show different size of programs used in implementation of method for software fault localization from 2003 to 2016 July.

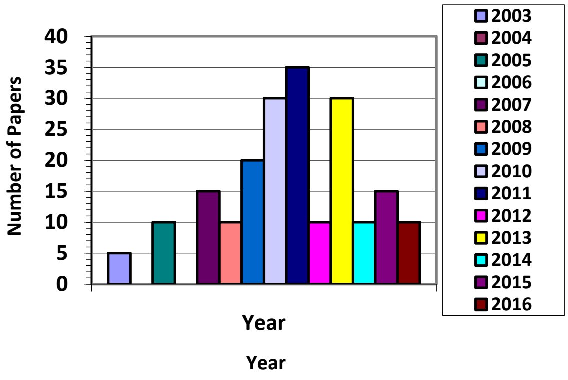

Fig 1: Papers on software fault localization from 2003 to July 2016.

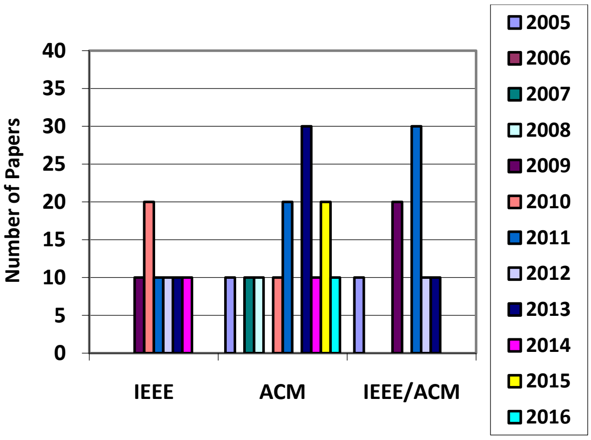

Fig 2: Papers published in IEEE, ACM, IEEE/ACM on software fault localization

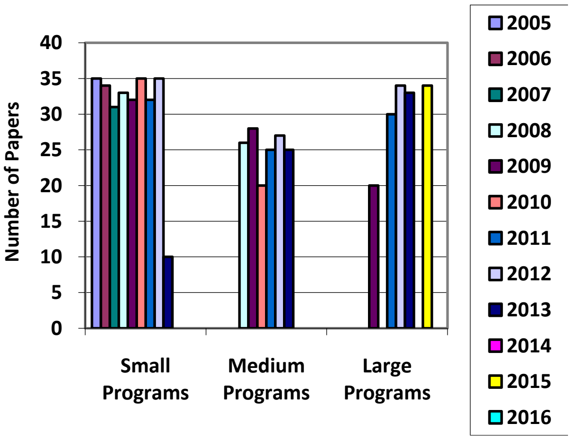

Fig 3: Programs used in implementation for software fault localization.

Table 1. Literature Review

Paper	Title	Journal Name	Year	Techniques & Tools Used for Fault Localization	Dataset	Size	Language
[1]	Automated Fault Localization Using Potential Invariants	Proceedings of ICSE	2003	Developed a program Carrot, that implements the debugging technique, The potential invariants & used Daikon for implementation.	tcas, print_tokens	Small Programs	C
[2]	Empirical evaluation of the tarantula automatic fault-localization technique	ASE	2005	Compare the Tarantula technique with the four techniques, Set union, Set intersection, Nearest Neighbor, Cause Transitions. Tarantula outperforms each technique.	print_tokens, print_tokens2, schedule, schedule2, replace, tcas, tot_info	Small Programs	C
[3]	SOBER: Statistical Model-based Bug Localization	ACM SIGSOFT	2005	Proposed a statistical approach to fault localization. Also, compared with CT and Liblit05, SOBER overcomes the limitation of both methods.	print tokens, print tokens2, replace, schedule, schedule2, tcas, and tot info	Small Programs	C
[4]	Effective Fault Localization Using Code Coverage	COMPSAC	2007	Used code coverage based Fault localization method, χDebug Tool is used for implementation. It presents three heuristics for prioritize source code in terms of its likelihood of containing a program bug. It also compares all three heuristics with Tarantula.	print_tokens, print_tokens2, schedule, schedule2, replace, tcas, tot_info	Small Programs	C
[5]	Debugging in Parallel.	Proceedings of ISSTA	2007	Two Parallel-Debugging Techniques that create specialized sets of test cases that can be assigned to different developers for simultaneous debugging.	Space	Small Programs	C
[6]	Fault Localization using Value Replacement	Proceedings of ISSTA	2008	Used Value Profile based approach (Interesting Value Mapping Pair IVMP) and Valgrind Infrastructure for Fault Localization. Here IVMP approach is compared with the Tarantula, and overall performs much better than the Tarantula approach.	tcas, totinfo, sched, sched2, ptok, ptok2, replace, space, grep-2.5, sed-4.1.5, flex-2.5.1, gzip-1.3	Small and Medium Programs	C
[7]	An empirical study of the effects of test-suite reduction on fault localization	Proceedings of ICSE	2008	Vector based & Statement based Reduction strategies that help to leverage the trade-offs between reduction and localization effectiveness	print tokens, print tokens2, replace, schedule, schedule2, space, tcas, tot info	Small Programs	C
[8]	Adaptive Random Test Case Prioritization	ASE	2009	Used coverage-based ART techniques for Fault Localization. They proposed nine ART techniques. Results show	print_tokens, print_tokens2, replace, schedule, schedule2, tcas,	Small and Medium Programs	C

				that these techniques are significantly more effective than random ordering.	tot_info, flex, grep, gzip, sed.		
[9]	Spectrum-Based Multiple Fault Localization	ASE	2009	Used BERINEL Bayesian reasoning approach with spectrum-based multiple fault localization. Also, compared the	print_tokens, print_tokens2, replace, schedule, schedule2, tcas, tot_info, space, gzip-1.3, sed-4.1.5, sed.	Small and Medium Programs	C
[10]	How Well Do Test Case Prioritization Techniques Support Statistical Fault Localization	IEEE International Computer Software and Applications Conference	2009	Presents impact of test case prioritization on the effectiveness of fault localization. Coverage-based techniques and random ordering can be more effective than distribution-based techniques in supporting statistical fault localization.	tcas, schedule, schedule2, tot_info, print_tokens, print_tokens2, replace	Small Programs	C
[11]	A family of code coverage-based heuristics for effective Fault localization	Elsevier	2010	Used code coverage-based heuristics, for implementation used χDebug tool.	tcas,schedule, schedule2, tot_info,print_tokens, print_tokens2, replace, Unix suite, space	Small Programs	C
[12]	Prioritizing Tests for Software Fault Localization	IEEE International Conference on Quality Software	2010	Test Prioritization approach is used for Maximize the improvement of diagnostic information.	print_tokens, print_tokens2, replace, schedule, schedule2, tcas, tot_info	Small Programs	C
[13]	On the Integration of Test Adequacy, Test Case Prioritization, and Statistical Fault Localization	IEEE International Conference on Quality Software	2010	16 test case prioritization techniques, 4 fault localization techniques are used for test case prioritization and statistical fault localization.	print_tokens, print_tokens2, replace, schedule, schedule2, tcas, tot_info.	Small Programs	C
[14]	Directed test generation for effective fault localization	Proceedings of ISSTA	2010	Directed test generation approach is used for Test case Generation with improved fault-localization effectiveness, Apollo tool is used for Implementation.	faqforge, webchess, schoolmate, phpsysinfo.	Small and Medium Programs	PHP, JS, C
[15]	Test input reduction for result inspection to facilitate fault localization	Springer	2010	Testing Reduction approach is used with three different strategies. Reduce the test inputs in an existing test collection to inspection effectiveness of fault localization	print_tokens, print_tokens2, replace, schedule, schedule2, tcas, tot_info, DC, TCC	Small and Medium Programs	C, Java
[16]	Are automated debugging techniques actually helping	Proceedings of ISSTA	2011	Used program slicing, Tarantula technique to Investigate how developers use and benefit from automated debugging tools	Tetris, NanoXML	Medium Programs	Java

	programmers?			through a set of human studies.			
[17]	Proximity based weighting of test cases to improve spectrum based fault localization	ASE	2011	Improve the effectiveness of spectrum based fault localization, this approach outperforms Qchiai.	print_tokens, print_tokens2, replace, schedule, schedule2, tcas, tot_info	Small Programs	C
[18]	Fault-localization using dynamic slicing and change impact analysis	ASE	2011	Dynamic slicing can be effective to improve performance of spectrum-based fault-localization	print_tokens, print_tokens2, schedule, schedule2, tcas, tot_info	Small Programs	C
[19]	Prioritizing tests for fault localization through ambiguity group reduction	ASE	2011	RAPTOR (Greedy diagnostic Prioritization by ambiguity group Reduction) Algorithm is used for Prioritizing Test.	print_tokens, print_tokens2, replace, schedule, schedule2, tcas, tot_info, space, grep, sed, flex, gzip	Small, Medium and Large Programs	C
[20]	On Practical Adequate Test Suites for Integrated Test Case Prioritization and Fault Localization	IEEE International Conference on Quality Software	2011	16 test case prioritization techniques, 4 statistical fault localization techniques are used.	tcas, totinfo, sched, sched2, ptok, ptok2, replace, grep(2.22.4.2), sed(1.183.02), flex(2.4.72.5.4), gzip(1.1.2–1.3).	Small, Medium and Large Programs	C
[21]	Mutation-based Statistical Test Inputs Generation for Automatic Fault Localization	IEEE International Conference on Software Security and Reliability	2012	Used new approach based on constraint solving and mutation-based statistical testing method called µTIL to improve the fault Localization Accuracy.	tcas	Small Programs	C
[22]	Diversity maximization speedup for fault localization	Proceedings of ASE	2012	Diversity Maximization Speedup (Dms) approach is used to maximize effectiveness	print_tokens, print_tokens2, replace, schedule, schedule2, tcas, tot_info, space, grep, sed, flex, gzip	Small, Medium and Large Programs	C
[23]	Using likely invariants for automated software fault localization	Proceedings of ASPLOS	2013	Propose an automatic diagnostic technique for isolating the root cause of software failures	Squid HTTP proxy server, MySQL database server, HTTP web server	Very large Servers	C, C++, XML
[24]	Using automated program repair for evaluating the effectiveness of fault localization	Proceedings of ISSTA	2013	Propose to develop AFL techniques from the viewpoint of fully automated debugging, and present the NCP score as the evaluation, measurement to assess and compare the effectiveness	libtiff, python, php, wireshark	Very large programs	C

	techniques			of various techniques.			
[25]	Where Should We Fix This Bug? A Two-Phase Recommendation Model	IEEE Transactions on Software Engineering	2013	(Machine learning-based) two-phase prediction model is used for debugging.	Firefox, Core	Large Programs	C and C++
[26]	Improving bug localization using structured information retrieval	ASE	2013	Present BLUiR, which embodies this insight, requires only the source code and bug reports, and takes advantage of bug similarity data if available.	SWT 3.1, Eclipse 3.1, AspectJ, Zxing		Java
[27]	Defects4J: a database of existing faults to enable controlled testing studies for Java programs	Proceedings of ISSTA	2014	Presents Defects4J, a database and extensible Framework to enable controlled testing studies for Java pro- grams.	JFreeChart, Closure Compiler, Commons Math, Joda-Time, Commons Lang	Large Programs	Java
[28]	Learning to Combine Multiple Ranking Metrics for Fault Localization	IEEE Conference on Software Maintenance and Evolution	2014	MULTRIC, learning-based approach is used to combining multiple ranking metrics. MULTRIC consists of two major phases, namely learning and ranking.	Daikon, Eventbus, Jaxen, Jester, Jexel, Jparsec, AcCodec, AcLang, Draw2d		
[29]	Evaluating the usefulness of IR-based fault localization techniques	Proceedings of ISSTA	2015	IR (Information Retrieval) based Technique is used.	Aspectj, SWT, Zxing, Jodatime		Java
[30]	Information retrieval and spectrum based bug localization: better together	Proceedings of ESEC/FSE	2015	IR based Technique, Spectrum Based Technique both are used.	AspectJ , Ant, Lucene , Rhino, iBugs		Java
[31]	A learning-to-rank based fault localization approach using likely invariants	Proceedings of ISSTA	2016	Savant-Rank based approach is used, Daikon tool is used for implementation.	JFreeChart, Closure Compiler, Commons Math, Joda-Time, Commons Lang		Java

4. ACKNOWLEDGMENTS

Thanks to Prof. Miral Patel from Department of IT, G H Patel Collage of Engineering and Technology for her discussions and suggestions during the preparation of this paper.

5. CONCLUSION

Effectiveness of techniques is not the only attribute to be consider but other factors like overhead, time, space, human efforts are also should be considered. Early fault localization is necessary as it leads to reduce time, efforts, cost and resources. A great number of fault localization techniques have been proposed in the last decades, with that software complexity is also increased. So these many techniques are far from perfect. Thus, there is a significant amount of research still to be done.

The survey concludes that the most significant method is coverage-based method, it should be used in future also. In many papers, authors have defined their own techniques (ART-Adaptive Random Testing), (DMS-Diversity maximization Speedup) etc. For the future work large datasets should be used to implement the method to increase accuracy. Survey shows that researchers have used different types of datasets, which are public archives, open source datasets or commercial repositories. Most widely used dataset are Siemens suites and UNIX programs. Such datasets are distributed freely and available to everyone.

6. REFERENCES

[1] Brock Pytlik, Manos Renieris, Shriram Krishnamurthy and Steven P. Reiss, “Automated Fault Localization Using Potential Invariants”, Proceedings of the 24th International Conference on Software Engineering - ICSE. September-2003.

[2] James A. Jones and Mary Jean Harrold, “Empirical evaluation of the tarantula automatic fault-localization technique”, Proceedings of the 20th IEEE/ACM international Conference on Automated software engineering - ASE '05.

[3] Chao Liu, Xifeng Yan, Long Fei, Jiawei Han and Samuel P. Midkiff, “SOBER: Statistical Model-based Bug Localization”, ACM SIGSOFT Software Engineering Notes. Vol. 30, ESEC-FSE’05, September 5–9, 2005.

[4] W. Eric Wong, Yu Qi, Lei Zhao, and Kai-Yuan Cai, “Effective Fault Localization Using Code Coverage”, 31st Annual International Computer Software and Applications Conference IEEE - Vol. 1- (COMPSAC 2007).

[5] James A. Jones, James F. Bowring and Mary Jean Harrold, “Debugging in Parallel”, Proceedings of the 2007 International Symposium on Software Testing and Analysis - ISSTA '07.

[6] Dennis Jeffrey, Neelam Gupta and Rajiv Gupta, “Fault Localization using Value Replacement”, Proceedings of the 2008 International Symposium on Software Testing and Analysis - ISSTA '08.

[7] Yanbing Yu, James A. Jones and Mary Jean Harrold, “An empirical study of the effects of test-suite reduction on fault localization”, Proceedings of the 13th international conference on Software Engineering - ICSE '08

[8] Bo Jiang, Zhenyu Zhang, W. K. Chan, T. H. Tse, “Adaptive Random Test Case Prioritization”, 2009 IEEE/ACM International Conference on Automated Software Engineering.

[9] Rui Abreu, Peter Zoeteweij, Arjan J.C. van Gemund, “Spectrum-Based Multiple Fault Localization”, 2009 IEEE/ACM International Conference on Automated Software Engineering.

[10] Bo Jiang, Zhenyu Zhang, T. H. Tse, T. Y. Chen, “How Well Do Test Case Prioritization Techniques Support Statistical Fault Localization”, 2009 33rd Annual IEEE International Computer Software and Applications Conference

[11] W. Eric Wonga,, Vidroha Debroy and Byoungju Choi, “A family of code coverage-based heuristics for effective Fault localization”, Journal of Systems and Software(Elsevier- 2010), Vol. 83, pp, 188–208.

[12] Alberto Gonzalez-Sanchez, Eric Piel, Hans-Gerhard Gross, Arjan J.C. van Gemund, “Prioritizing Tests for Software Fault Localization”, 2010 10th International Conference on Quality Software.

[13] Bo Jiang and W.K. Chan, “On the Integration of Test Adequacy, Test Case Prioritization, and Statistical Fault Localization”, 2010 10th International Conference on Quality Software.

[14] Shay Artzi, Julian Dolby, Frank Tip and Marco Pistoia, “Directed test generation for effective fault localization”, Proceedings of the 19th international symposium on Software testing and analysis - ISSTA '10.

[15] Dan Hao, Tao Xie, Lu Zhang, Xiaoyin Wang, Jiasu Sun, Hong Mei “Test input reduction for result inspection to facilitate fault localization”, Automated Software Engineering(Springer-2010).

[16] Chris Parnin and Alessandro Orso, “Are automated debugging techniques actually helping programmers?”, Proceedings of the 2011 International Symposium on Software Testing and Analysis - ISSTA '11.

[17] Aritra Bandyopadhyay and Sudipto Ghosh, “Proximity Based Weighting of Test cases to Improve Spectrum Based Fault Localization”, 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011).

[18] Elton Alves, Milos Gligoric, Vilas Jagannath and Marcelo D'Amorim, “Fault-localization using dynamic slicing and change impact analysis”, 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011).

[19] Gonzalez-Sanchez, Alberto Abreu, Rui Gross, Hans-Gerhard van Gemund, Arjan J.C, “Prioritizing tests for fault localization through ambiguity group reduction”, 26th IEEE/ACM International Conference on Automated Software Engineering (ASE 2011).

[20] Bo Jiang, W.K. Chan, T.H. Tse, ‘On Practical Adequate Test Suites for Integrated Test Case Prioritization and Fault Localization”, IEEE 2011 11th International Conference on Quality Software.

[21] Mickael Delahaye, Lionel C. Briand, Arnaud Gotlieb, Matthieu Petit, “Mutation-based Statistical Test Inputs Generation for Automatic Fault Localization”, 2012 IEEE Sixth International Conference on Software Security and Reliability.

[22] Jian Zhou, Hongyu Zhang, David Lo, “Where should the bugs be fixed? More accurate information retrieval-based bug localization based on bug reports”, 2012 34th International Conference on Software Engineering (ICSE).

[23] Liang Gong, David Lo, Lingxiao Jiang, Hongyu Zhang, “Diversity maximization speedup for fault localization”, Proceedings of the 27th IEEE/ACM International Conference on Automated Software Engineering - ASE 2012.

[24] Swarup Kumar Sahoo, John Criswell, Chase Geigle, Vikram Adve, “Using likely invariants for automated software fault localization”, “Proceedings of the eighteenth international conference on Architectural support for programming languages and operating systems - ASPLOS '13”.

[25] Yuhua Qi, Xiaoguang Mao , Yan Lei, and ChengsongWang, “Using automated program repair for evaluating the effectiveness of fault localization techniques”, Proceedings of the 2013 International Symposium on Software Testing and Analysis - ACM ISSTA 2013.

[26] Dongsun Kim, Yida Tao, Sunghun Kim and Andreas Zeller, “Where Should We Fix This Bug? A Two-Phase

Recommendation Model", IEEE Transactions on Software Engineering.

[27] Dunwei Gong, Yan Zhang, "Generating test data for both path coverage and fault detection using genetic algorithms", 2013-Frontiers of Computer Science.

[28] Jifeng Xuan, Martin Monperrus, "Learning to Combine Multiple Ranking Metrics for Fault Localization", 2014 IEEE International Conference on Software Maintenance and Evolution.

[29] Qianqian Wang, Chris Parnin, Alessandro Orso, "Evaluating the usefulness of IR-based fault localization techniques", Proceedings of the International Symposium on Software Testing and Analysis - ACM ISSTA 2015.

[30] Tien-Duy B. Le, Richard J. Oentaryo, David Lo, "Information retrieval and spectrum based bug localization: better together", Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering - ESEC/FSE.

[31] Tien-Duy B. Le, David Lo, Claire Le Goues, Lars Grunske, "A learning-to-rank based fault localization approach using likely invariants", Proceedings of the 25th International Symposium on Software Testing and Analysis - ACM ISSTA 2016.