Exploring means to facilitate software debugging

SOLTANI, M.S.

9 About the Author

Mozhan Soltani, born in 1989, received her B.Sc. degree in Software Engineering and Management from Gothenburg University, Gothenburg, Sweden, in 2014. She received her M.Sc. in Software Engineering from Gothenburg University, Gothenburg, Sweden, in 2016. In 2016, Mozhan Soltani moved to the Netherlands to do her PhD research on the use of automated test generation for software debugging at the Software Engineering Research Group (SERG), at Delft University of Technology. In 2019, she moved to Leiden University to complete the PhD research on exploring means to facilitate software debugging.

Bibliography

[1] https://github.com/netty/netty/issues/8285, accessed on 2019-06-20.

[2] https://github.com/axios/axios/issues/376, accessed on 2019-06-20.

[3] https://github.com/axios/axios/issues/246, accessed on 2019-06-20.

[4] https://github.com/apache/dubbo/issues/1500, accessed on 2019-06-20.

[5] https://github.com/apache/dubbo/issues/3236, accessed on 2019-06-20.

[6] https://github.com/activeadmin/activeadmin/issues/2623, accessed on 2019-06-20.

[7] https://github.com/activeadmin/activeadmin/issues/3185, accessed on 2019-06-20.

[8] https://github.com/activeadmin/activeadmin/issues/4650, accessed on 2019-06-20.

[9] https://developer.github.com/v3/issues/, accessed on 2015-05-01.

[10] https://github.com/angular/angular.js/issues/16697, accessed on 2019-06-20.

[11] https://github.com/ageitgey/face_recognition/issues/854, accessed on 2019-06-20.

[12] https://github.com/airbnb/lottie-android/issues/1202, accessed on 2019-06-20.

[13] https://github.com/barryvdh/laravel-debugbar/issues/237, accessed on 2019-06-20.

[14] https://github.com/activeadmin/activeadmin/issues/4250, accessed on 2019-06-20.

[15] The state of the octoverse. https://octoverse.github.com/, accessed on 2019-05-01.

[16] The state of the octoverse 2017. https://octoverse.github.com/2017/, accessed on 2019-05-01.

[17] The state of the octoverse: top programming languages of 2018. https://github.blog/2018-11-15-state-of-the-octoverse-top-programming-languages/, accessed on 2019-05-01.

[18] UI/Application Exerciser Monkey. `https://developer.android.com/studio/test/monkey.html`, 2017. [Online; accessed 24-July-2017].

[19] Aldanial, 2019. https://github.com/AlDanial/cloc, Accessed: 2019-08-20.

[20] Boost, 2019. https://www.boost.org/, accessed on 2019-07-01.

[21] Boost.contract example, 2019. https://www.boost.org/doc/libs/1_67_0/libs/contract/doc/html/boost/contract/public_functio_idp69202896.html, Accessed:
2019-08-20.

[22] Cloc, 2019. http://cloc.sourceforge.net/, Accessed: 2019-08-20.

[23] Cofoja. https://github.com/nhatminhle/cofoja, 2019. accessed on 2019-07-01.

[24] cofoja-api, 2019. https://github.com/wao/cofoja-api, Accessed: 2019-08-22.

[25] Cofoja example, 2019. http://blog.code-cop.org/2018/02/complete-cofoja-setup-example.html, Accessed: 2019-08-20.

[26] Github explore, 2019. https://github.com/explore, Accessed: 2019-07-15.

[27] icontract, 2019. https://pypi.org/project/icontract/, accessed on 2019-07-01.

[28] Inf3143_tp2, 2019. https://github.com/Parquery/mapry, Accessed: 2019-08-22.

[29] Inf3143_tp2, 2019. https://github.com/nic-lovin/INF3143_TP2, Accessed: 2019-08-22.

[30] Jml example, 2019. http://www.eecs.ucf.edu/ leavens/JML-release/org/jmlspecs/samples/stacks/, Accessed: 2019-08-20.

[31] Jml home page: The java modeling language (jml), 2019. http://www.eecs.ucf.edu/ leavens/JML/index.shtml, accessed on 2019-07-01.

[32] library-manager, 2019. https://github.com/openminded-oscar/library-manager, Accessed: 2019-08-22.

[33] Miniurl, 2019. https://github.com/gunbuster135/MiniUrl, Accessed: 2019-08-22.

[34] Pycontracts, 2019. https://andreacensi.github.io/contracts/, accessed on 2019-07-01.

[35] The state of the octoverse, 2019. https://octoverse.github.com/, accessed on 2019-05-01.

[36] streamline, 2019. https://github.com/denipotapov/streamline, Accessed: 2019-08-22.

[37] Valid4j, 2019. http://www.valid4j.org/, accessed on 2019-07-01.

[38] Valid4j example, 2019. http://www.valid4j.org/concepts.html, Accessed: 2019-08-20.

[39] S. Adee. Bad bugs: The worst disasters caused by software fails, 2019.

[40] S. Adolph, W. Hall, and P. Kruchten. Using grounded theory to study the experience of software development. *Empirical Software Engineering*, 16(4):487–513, 2011.

[41] S. Afshan, P. McMinn, and M. Stevenson. Evolving readable string test inputs using a natural language model to reduce human oracle cost. In *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation*, pages 352–361, March 2013.

[42] N. M. Albunian. Diversity in search-based unit test suite generation. In *Int'l Symposium on Search Based Software Engineering*, pages 183–189. Springer, 2017.

[43] N. Alshahwan, X. Gao, M. Harman, Y. Jia, K. Mao, A. Mols, T. Tei, and I. Zorin. Deploying Search Based Software Engineering with Sapienz at Face-

book. In *Search-Based Software Engineering. SSBSE 2018.*, volume 11036 of *LNCS*. Springer, 2018.

[44] A. Ang, A. Perez, A. van Deursen, and R. Abreu. *Revisiting the Practical Use of Automated Software Fault Localization Techniques*. IEEE, United States, 2017.

[45] J. Anvik, L. Hiew, and G. C. Murphy. Coping with an open bug repository. In *Proceedings of the 2005 OOPSLA workshop on Eclipse technology eXchange*, pages 35–39. ACM, 2005.

[46] Apache. Ant. `http://ant.apache.org/`, 2017. [Online; accessed 25-January-2018].

[47] Apache. Commons Collections. `https://commons.apache.org/proper/commons-collections/`, 2017. [Online; accessed 25-January-2018].

[48] Apache. Log4j. `https://logging.apache.org/log4j/2.x/`, 2017. [Online; accessed 25-January-2018].

[49] A. Arcuri. RESTful API Automated Test Case Generation. In *2017 IEEE International Conference on Software Quality, Reliability and Security (QRS)*, pages 9–20. IEEE, jul 2017.

[50] A. Arcuri and L. Briand. A hitchhiker's guide to statistical tests for assessing randomized algorithms in software engineering. *Software Testing, Verification and Reliability*, 24(3):219–250, 2014.

[51] A. Arcuri and G. Fraser. On Parameter Tuning in Search Based Software Engineering. In *Population English Edition*, pages 33–47. 2011.

[52] A. Arcuri and G. Fraser. Parameter tuning or default values? an empirical investigation in search-based software engineering. *Empirical Software Engineering*, 18(3):594–623, 2013.

[53] A. Arcuri and G. Fraser. Java enterprise edition support in search-based junit test generation. In *International Symposium on Search Based Software Engineering*, pages 3–17. Springer, 2016.

[54] A. Arcuri, G. Fraser, and J. P. Galeotti. Automated unit test generation for classes with environment dependencies. In *Proceedings of the 29th ACM/IEEE international conference on Automated software engineering - ASE '14*, pages 79–90, Vasteras, Sweden, 2014. ACM Press.

[55] A. Arcuri, G. Fraser, and J. P. Galeotti. Generating tcp/udp network data for automated unit test generation. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering*, pages 155–165. ACM, 2015.

[56] A. Arcuri, G. Fraser, and R. Just. Private api access and functional mocking in automated unit test generation. In *Software Testing, Verification and Validation (ICST), 2017 IEEE International Conference on*, pages 126–137, Tokyo, Japan, 2017. IEEE, IEEE Computer Society.

[57] S. Artzi, J. Dolby, F. Tip, and M. Pistoia. Directed test generation for effective fault localization. In *Proceedings of the 19th International Symposium on Software Testing and Analysis*, ISSTA '10, pages 49–60. ACM, 2010.

[58] S. Artzi, S. Kim, and M. D. Ernst. Recrash: Making software failures reproducible by preserving object states. In *Proceedings of the 22Nd European Conference on Object-Oriented Programming*, ECOOP '08, pages 542–565, Berlin, Heidelberg, 2008. Springer-Verlag.

[59] A. Baars, M. Harman, Y. Hassoun, K. Lakhotia, P. McMinn, P. Tonella, and T. Vos. Symbolic search-based testing. In *Proceedings of the 2011 26th IEEE/ACM International Conference on Automated Software Engineering*, pages 53–62. IEEE Computer Society, 2011.

[60] C. Baier, J.-P. Katoen, and K. G. Larsen. *Principles of model checking*. MIT press, 2008.

[61] A. Baresel, D. Binkley, M. Harman, and B. Korel. Evolutionary testing in the presence of loop-assigned flags: A testability transformation approach. In *ACM SIGSOFT Software Engineering Notes*, volume 29, pages 108–118, Boston, Massachusetts, USA, 2004. ACM, ACM.

[62] K. L. Barriball and A. While. Collecting data using a semi-structured interview: a discussion paper. *Journal of Advanced Nursing-Institutional Subscription*, 19(2):328–335, 1994.

[63] V. R. Basili, F. Shull, and F. Lanubile. Building knowledge through families of experiments. *IEEE Transactions on Software Engineering*, 25(4):456–473, 1999.

[64] J. Bell, N. Sarda, and G. Kaiser. Chronicler: Lightweight recording to reproduce field failures. In *Proceedings of the 2013 International Conference on Software Engineering*, ICSE '13, pages 362–371, Piscataway, NJ, USA, 2013. IEEE Press.

[65] F. A. Bianchi, M. Pezzè, and V. Terragni. Reproducing concurrency failures from crash stacks. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pages 705–716, Paderborn, Germany, 2017. ACM, ACM.

[66] C. Bird, A. Bachmann, E. Aune, J. Duffy, A. Bernstein, V. Filkov, and P. Devanbu. Fair and balanced?: bias in bug-fix datasets. In *Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, pages 121–130. ACM, 2009.

[67] T. F. Bissyandé, D. Lo, L. Jiang, L. Réveillere, J. Klein, and Y. Le Traon. Got issues? who cares about it? a large scale investigation of issue trackers from github. In *2013 IEEE 24th international symposium on software reliability engineering (ISSRE)*, pages 188–197. IEEE, 2013.

[68] H. Borges, M. T. Valente, A. Hora, and J. Coelho. On the popularity of github applications: A preliminary note. *arXiv preprint arXiv:1507.00604*, 2015.

[69] C. Boyapati, S. Khurshid, and D. Marinov. Korat: Automated testing based on java predicates. In *Proceedings of the 2002 ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA '02, pages 123–133, New York, NY, USA, 2002. ACM.

[70] P. Braione, G. Denaro, A. Mattavelli, and M. Pezzè. Combining symbolic execution and search-based testing for programs with complex heap inputs. In *Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA 2017, pages 90–101, New York, NY, USA, 2017. ACM.

[71] V. Braun and V. Clarke. Using thematic analysis in psychology. *Qualitative research in psychology*, 3(2):77–101, 2006.

[72] L. Briand, D. Bianculli, S. Nejati, F. Pastore, and M. Sabetzadeh. The Case for Context-Driven Software Engineering Research: Generalizability Is Overrated. *IEEE Software*, 34(5):72–75, 2017.

[73] O. Bühler and J. Wegener. Evolutionary functional testing. *Computers & Operations Research*, 35(10):3144–3160, 2008.

[74] R. P. Buse, C. Sadowski, and W. Weimer. Benefits and barriers of user evaluation in software engineering research. *ACM SIGPLAN Notices*, 46(10):643–656, 2011.

[75] B. Cabral and P. Marques. Exception Handling: A Field Study in Java and .NET. In *ECOOP 2007 âĂŞ Object-Oriented Programming*, volume 4609, pages 151–175. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[76] Y. Cao, H. Zhang, and S. Ding. Symcrash: Selective recording for reproducing crashes. In *Proceedings of the 29th ACM/IEEE International Conference on Automated Software Engineering*, ASE '14, pages 791–802. ACM, 2014.

[77] C. Casalnuovo, P. Devanbu, A. Oliveira, V. Filkov, and B. Ray. Assert use in github projects. In *Proceedings of the 37th International Conference on Software Engineering-Volume 1*, pages 755–766. IEEE Press, 2015.

[78] C. Casalnuovo, P. Devanbu, A. Oliveira, V. Filkov, and B. Ray. Assert use in github projects. In *Proceedings of the 37th International Conference on Software Engineering-Volume 1*, pages 755–766. IEEE Press, 2015.

[79] M. Ceccato, A. Marchetto, L. Mariani, C. D. Nguyen, and P. Tonella. An empirical study about the effectiveness of debugging when random test cases are used. In *Proceedings of the 34th International Conference on Software Engineering*, pages 452–462. IEEE Press, 2012.

[80] S. Chandra, E. Torlak, S. Barman, and R. Bodik. Angelic debugging. In *Proceedings of the 33rd International Conference on Software Engineering*, pages 121–130. ACM, 2011.

[81] N. Chen and S. Kim. Star: Stack trace based automatic crash reproduction via symbolic execution. *IEEE Tr. on Sw. Eng.*, 41(2):198–220, 2015.

[82] H. Cibulski and A. Yehudai. Regression test selection techniques for test-driven development. In *Software Testing, Verification and Validation Workshops (ICSTW), 2011 IEEE Fourth International Conference on*, pages 115–124, March 2011.

[83] J. Clause and A. Orso. A technique for enabling and supporting debugging of field failures. In *29th International Conference on Software Engineering (ICSE'07)*, pages 261–270. IEEE, 2007.

[84] R. Coelho, L. Almeida, G. Gousios, A. v. Deursen, and C. Treude. Exception handling bug hazards in android. *Empirical Software Engineering*, pages 1–41, 2016.

[85] R. Coelho, L. Almeida, G. Gousios, and A. van Deursen. Unveiling exception handling bug hazards in android based on github and google code issues. In *Proceedings of the 12th Working Conference on Mining Software Repositories*, MSR '15, pages 134–145. IEEE Press, 2015.

[86] R. Coelho, L. Almeida, G. Gousios, A. van Deursen, and C. Treude. Exception handling bug hazards in Android. *Empirical Software Engineering*, 22(3):1264–1304, jun 2017.

[87] C. A. Coello Coello. Constraint-handling techniques used with evolutionary algorithms. In *Proc. of the Genetic and Evolutionary Computation Conference Companion (GECCO Companion)*, pages 563–587. ACM, 2016.

[88] M. Črepinšek, S.-H. Liu, and M. Mernik. Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR)*, 45(3):35, 2013.

[89] J. W. Creswell and J. D. Creswell. *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications, 2017.

[90] Y. Dang, R. Wu, H. Zhang, D. Zhang, and P. Nobel. Rebucket: A method for clustering duplicate crash reports based on call stack similarity. In *Proceedings of the 34th International Conference on Software Engineering*, ICSE 2012, pages 1084–1093. IEEE Press, 2012.

[91] D. De Vaus and D. de Vaus. *Surveys in social research*. Routledge, 2013.

[92] K. Deb. Multi-objective optimization. In *Search Methodologies*, pages 403–449. Springer US, 2014.

[93] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *TEVC*, 6(2):182–197, 2002.

[94] J. Dietrich, D. J. Pearce, K. Jezek, and P. Brada. Contracts in the wild: A study of java programs. In *31st European Conference on Object-Oriented Programming (ECOOP 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2017.

[95] Dubbo. A high-performance, java based, open source RPC framework. `http://dubbo.io`, 2018. [Online; accessed 25-January-2018].

[96] F. Eichinger, K. Krogmann, R. Klug, and K. Böhm. Software-defect localisation by mining dataflow-enabled call graphs. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 425–441. Springer, 2010.

[97] Elastic. Elasticsearch: RESTful, Distributed Search and Analytics. `https://www.elastic.co/products/elasticsearch`, 2018. [Online; accessed 25-January-2018].

[98] H.-C. Estler, C. A. Furia, M. Nordio, M. Piccioni, and B. Meyer. Contracts in practice. In *International Symposium on Formal Methods*, pages 230–246. Springer, 2014.

[99] R. Feldt, R. Torkar, T. Gorschek, and W. Afzal. Searching for cognitively diverse tests: Towards universal test diversity metrics. In *Proc. Int'l Conf. Software*

Testing Verification and Validation Workshops (ICSTW), pages 178–186. IEEE, 2008.

[100] A. Fink. *The survey handbook*. Sage, 2003.

[101] G. Fraser and A. Arcuri. 1600 faults in 100 projects: Automatically finding faults while achieving high coverage with evosuite. *Empirical Software Engineering*, 20(3):611–639, 2013.

[102] G. Fraser and A. Arcuri. Evosuite: On the challenges of test case generation in the real world. In *2013 IEEE Sixth International Conference on Software Testing, Verification and Validation*, pages 362–369. IEEE, 2013.

[103] G. Fraser and A. Arcuri. Whole test suite generation. *IEEE Transactions on Software Engineering*, 39(2):276–291, Feb. 2013.

[104] G. Fraser and A. Arcuri. Automated test generation for java generics. In *International Conference on Software Quality*, pages 185–198, Vienna, Austria, 2014. Springer, Springer.

[105] G. Fraser and A. Arcuri. A large-scale evaluation of automated unit test generation using evosuite. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 24(2):8, 2014.

[106] G. Fraser, A. Arcuri, and P. McMinn. A memetic algorithm for whole test suite generation. *Journal of Systems and Software*, 103:311–327, 2015.

[107] G. Fraser, J. M. Rojas, J. Campos, and A. Arcuri. E vo s uite at the sbst 2017 tool competition. In *Proceedings of the 10th International Workshop on Search-Based Software Testing*, pages 39–41. IEEE Press, 2017.

[108] G. Fraser, M. Staats, P. McMinn, A. Arcuri, and F. Padberg. Does automated white-box test generation really help software testers? In *Proceedings of the 2013 International Symposium on Software Testing and Analysis*, pages 291–301. ACM, 2013.

[109] G. Fraser, M. Staats, P. McMinn, A. Arcuri, and F. Padberg. Does automated unit test generation really help software testers? A controlled empirical study. *ACM Trans. Softw. Eng. Methodol.*, 24(4):23:1–23:49, 2015.

[110] G. R. Gibbs. Thematic coding and categorizing. *Analyzing qualitative data. London: Sage*, pages 38–56, 2007.

[111] B. G. Glaser and J. Holton. Remodeling grounded theory. In *Forum Qualitative Sozialforschung/Forum: Qualitative Social Research*, volume 5, 2004.

[112] P. Godefroid, N. Klarlund, and K. Sen. Dart: Directed automated random testing. In *Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '05, pages 213–223, New York, NY, USA, 2005. ACM.

[113] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proc. Int'l Conf. on Genetic Algorithms and Their Application*, pages 41–49. L. Erlbaum Associates Inc., 1987.

[114] M. Gómez, R. Rouvoy, B. Adams, and L. Seinturier. Reproducing context-sensitive crashes of mobile apps using crowdsourced monitoring. In *Proceedings of the International Conference on Mobile Software Engineering and Systems*, MOBILESoft '16, pages 88–99, New York, NY, USA, 2016. ACM.

[115] F. Gordon and A. Arcuri. Evosuite at the sbst 2016 tool competition. In *The 9th International Workshop on SEARCH-BASED SOFTWARE TESTING (SBST)*, 2016.

[116] B. Hailpern and P. Santhanam. Software debugging, testing, and verification. *IBM Systems Journal*, 41(1):4–12, 2002.

[117] M. Harman. The current state and future of search based software engineering. In *2007 Future of Software Engineering*, pages 342–357. IEEE Computer Society, 2007.

[118] M. Harman, L. Hu, R. Hierons, A. Baresel, and H. Sthamer. Improving evolutionary testing by flag removal. In *Proceedings of the 4th Annual Conference on Genetic and Evolutionary Computation*, pages 1359–1366, New York, USA, 2002. Morgan Kaufmann Publishers Inc., Morgan Kaufmann.

[119] M. Harman, L. Hu, R. Hierons, J. Wegener, H. Sthamer, A. Baresel, and M. Roper. Testability transformation. *IEEE Transactions on Software Engineering*, 30(1):3–16, 2004.

[120] M. Harman, S. A. Mansouri, and Y. Zhang. Search-based software engineering: Trends, techniques and applications. *ACM Computing Surveys (CSUR)*, 45(1):11, 2012.

[121] M. Harman and P. McMinn. A theoretical & empirical analysis of evolutionary testing and hill climbing for structural test data generation. In *Proceedings of the 2007 international symposium on Software testing and analysis*, pages 73–83. ACM, 2007.

[122] M. Harman, P. McMinn, J. De Souza, and S. Yoo. Search based software engineering: Techniques, taxonomy, tutorial. In *Empirical software engineering and verification*, pages 1–59. Springer, 2012.

[123] P. Hooimeijer and W. Weimer. Modeling bug report quality. In *Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering*, pages 34–43. ACM, 2007.

[124] M. Höst, B. Regnell, and C. Wohlin. Using students as subjectsâĂŤa comparative study of students and professionals in lead-time impact assessment. *Empirical Software Engineering*, 5(3):201–214, 2000.

[125] S. E. Hove and B. Anda. Experiences from conducting semi-structured interviews in empirical software engineering research. In *11th IEEE International Software Metrics Symposium (METRICS'05)*, pages 10–pp. IEEE, 2005.

[126] S. A. Jacob and S. P. Furgerson. Writing interview protocols and conducting interviews: Tips for students new to the field of qualitative research. *The qualitative report*, 17(42):1–10, 2012.

[127] M. Jähne, X. Li, and J. Branke. Evolutionary algorithms and multiobjectivization for the travelling salesman problem. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 595–602. ACM, 2009.

[128] Java Design Patterns. Design patterns implemented in Java. `http://java-design-patterns.com`, 2018. [Online; accessed 25-January-2018].

[129] H. Jaygarl, S. Kim, T. Xie, and C. K. Chang. OCAT: Object Capture-based Automated Testing. In *Proceedings of the 19th International Symposium on Software Testing and Analysis*, ISSTA '10, pages 159–170, New York, NY, USA, 2010. ACM.

[130] JDK. Stack trace has invalid line numbers. `https://bugs.openjdk.java.net/browse/JDK-7024096`, 2016. [Online; accessed 25-January-2018].

[131] C. Jee and T. Macaulay. Top software failures in recent history, 2019.

[132] Y. Jia and M. Harman. Constructing subtle faults using higher order mutation testing. In *Source Code Analysis and Manipulation, 2008 Eighth IEEE International Working Conference on*, pages 249–258. IEEE, 2008.

[133] J. Jiang, D. Lo, J. He, X. Xia, P. S. Kochhar, and L. Zhang. Why and how developers fork what from whom in github. *Empirical Software Engineering*, 22(1):547–578, 2017.

[134] W. Jin and A. Orso. Bugredux: Reproducing field failures for in-house debugging. In *Proceedings of the 34th International Conference on Software Engineering*, ICSE '12, pages 474–484, Piscataway, NJ, USA, 2012. IEEE Press.

[135] R. Just, D. Jalali, and M. D. Ernst. Defects4J: a database of existing faults to enable controlled testing studies for Java programs. In *Proceedings of the 2014 International Symposium on Software Testing and Analysis - ISSTA 2014*, pages 437–440, San Jose, CA, USA, 2014. ACM Press.

[136] F. M. Kifetew, W. Jin, R. Tiella, A. Orso, and P. Tonella. Sbfr: A search based approach for reproducing failures of programs with grammar based input. In *Proceedings of the 28th IEEE/ACM International Conference on Automated Software Engineering*, ASE'13, pages 604–609, Piscataway, NJ, USA, 2013. IEEE Press.

[137] F. M. Kifetew, W. Jin, R. Tiella, A. Orso, and P. Tonella. Reproducing field failures for programs with complex grammar-based input. In *Proceedings of the 2014 IEEE International Conference on Software Testing, Verification, and Validation*, ICST '14, pages 163–172, Washington, DC, USA, 2014. IEEE Computer Society.

[138] F. M. Kifetew, A. Panichella, A. De Lucia, R. Oliveto, and P. Tonella. Orthogonal exploration of the search space in evolutionary test case generation. In *Proc. Int'l Symposium on Software Testing and Analysis (ISSTA)*, pages 257–267. ACM, 2013.

[139] B. A. Kitchenham and S. L. Pfleeger. Principles of survey research: part 3: constructing a survey instrument. *ACM SIGSOFT Software Engineering Notes*, 27(2):20–24, 2002.

[140] J. D. Knowles, R. A. Watson, and D. W. Corne. Reducing local optima in single-objective problems by multi-objectivization. In *International Conference on Evolutionary Multi-Criterion Optimization*, pages 269–283. Springer, 2001.

[141] P. S. Kochhar, T. F. Bissyandé, D. Lo, and L. Jiang. Adoption of software testing in open source projects–a preliminary study on 50,000 projects. In *2013 17th European Conference on Software Maintenance and Reengineering*, pages 353–356. IEEE, 2013.

[142] P. S. Kochhar, T. F. Bissyandé, D. Lo, and L. Jiang. An empirical study of adoption of software testing in open source projects. In *2013 13th International Conference on Quality Software*, pages 103–112. IEEE, 2013.

[143] P. S. Kochhar and D. Lo. Revisiting assert use in github projects. In *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*, pages 298–307. ACM, 2017.

[144] P. S. Kochhar and D. Lo. Revisiting assert use in github projects. In *Proceedings of the 21st International Conference on Evaluation and Assessment in Software Engineering*, pages 298–307. ACM, 2017.

[145] P. S. Kochhar, D. Wijedasa, and D. Lo. A large scale study of multiple programming languages and code quality. In *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, volume 1, pages 563–573. IEEE, 2016.

[146] R. Kramer. icontract-the java/sup tm/design by contract/sup tm/tool. In *Proceedings. Technology of Object-Oriented Languages. TOOLS 26 (Cat. No. 98EX176)*, pages 295–307. IEEE, 1998.

[147] G. Kudrjavets, N. Nagappan, and T. Ball. Assessing the relationship between software assertions and faults: An empirical investigation. In *2006 17th International Symposium on Software Reliability Engineering*, pages 204–212. IEEE, 2006.

[148] C. Le Goues, T. Nguyen, S. Forrest, and W. Weimer. Genprog: A generic method for automatic software repair. *Ieee transactions on software engineering*, 38(1):54–72, 2011.

[149] G. T. Leavens and Y. Cheon. Design by contract with jml, 2006.

[150] A. Leitner, I. Ciupa, M. Oriol, B. Meyer, and A. Fiva. Contract driven development = test driven development - writing test cases. In *Proceedings of the the 6th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on The Foundations of Software Engineering*, ESEC-FSE '07, pages 425–434, New York, NY, USA, 2007. ACM.

[151] A. Leitner, A. Pretschner, S. Mori, B. Meyer, and M. Oriol. On the effectiveness of test extraction without overhead. In *2009 International Conference on Software Testing Verification and Validation*, pages 416–425. IEEE, 2009.

[152] Y. Li and G. Fraser. Bytecode testability transformation. In *International Symposium on Search Based Software Engineering*, pages 237–251, Szeged, Hungary, 2011. Springer, Springer.

[153] R. Likert. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

[154] B. Liskov and J. Guttag. *Program development in JAVA: abstraction, specification, and object-oriented design*. Pearson Education, London, England, UK, 2000.

[155] Q. Luo, F. Hariri, L. Eloussi, and D. Marinov. An empirical analysis of flaky tests. In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering - FSE 2014*, pages 643–653, 2014.

[156] A. Machiry, R. Tahiliani, and M. Naik. Dynodroid: An input generation system for android apps. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, pages 224–234. ACM, 2013.

[157] A. Maiga, A. Hamou-Lhadj, M. Nayrolles, K. Koochekian Sabor, and A. Larsson. An empirical study on the handling of crash reports in a large software company: An experience report. In *2015 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, pages 342–351, Bremen, Germany, sep 2015. IEEE.

[158] J. Malburg and G. Fraser. Combining search-based and constraint-based testing. In *Automated Software Engineering (ASE), 2011 26th IEEE/ACM International Conference on*, pages 436–439, Lawrence, KS, USA, 2011. IEEE, IEEE Computer Society.

[159] K. Mao, M. Harman, and Y. Jia. Sapienz: Multi-objective automated testing for android applications. In *Proceedings of the 25th International Symposium on Software Testing and Analysis*, pages 94–105. ACM, 2016.

[160] P. McMinn. Search-based software test data generation: a survey. *Software Testing, Verification and Reliability*, 14(2):105–156, jun 2004.

[161] P. McMinn. Search-based software testing: Past, present and future. In *Software testing, verification and validation workshops (icstw), 2011 ieee fourth international conference on*, pages 153–163, Berlin, Germany, 2011. IEEE, IEEE Computer Society.

[162] B. Meyer. Applying'design by contract'. *Computer*, 25(10):40–51, 1992.

[163] B. Meyer. Building bug-free oo software: An introduction to design by contract. *Availabe at http://archive. eiffel. com/doc/manuals/technology/contract*, 1998.

[164] A. Moghaddam. Coding issues in grounded theory. *Issues in educational research*, 16(1):52–66, 2006.

[165] K. Moran, M. Linares-Vásquez, C. Bernal-Cárdenas, C. Vendome, and D. Poshyvanyk. Automatically discovering, reporting and reproducing android

application crashes. In *2016 IEEE International Conference on Software Testing, Verification and Validation (ICST)*, pages 33–44, April 2016.

[166] T. Mortensen, R. Fisher, and G. Wines. Students as surrogates for practicing accountants: Further evidence. In *Accounting Forum*, volume 36, pages 251–265. Elsevier, 2012.

[167] M. D. Myers and M. Newman. The qualitative interview in is research: Examining the craft. *Information and organization*, 17(1):2–26, 2007.

[168] L. Naish, H. J. Lee, and K. Ramamohanarao. A model for spectra-based software diagnosis. *ACM Transactions on software engineering and methodology (TOSEM)*, 20(3):11, 2011.

[169] S. Narayanasamy, G. Pokam, and B. Calder. Bugnet: Continuously recording program execution for deterministic replay debugging. In *Proceedings of the 32Nd Annual International Symposium on Computer Architecture*, ISCA '05, pages 284–295, Washington, DC, USA, 2005. IEEE Computer Society.

[170] M. Nayrolles and A. Hamou-Lhadj. BUMPER: A Tool for Coping with Natural Language Searches of Millions of Bugs and Fixes. In *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, pages 649–652, Suita, Osaka, Japan, mar 2016. IEEE.

[171] M. Nayrolles, A. Hamou-Lhadj, S. Tahar, and A. Larsson. Jcharming: A bug reproduction approach using crash traces and directed model checking. In *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, pages 101–110. IEEE, 2015.

[172] M. Nayrolles, A. Hamou-Lhadj, S. Tahar, and A. Larsson. A bug reproduction approach based on directed model checking and crash traces. *Journal of Software: Evolution and Process*, pages n/a–n/a, 2016. JSME-15-0137.R1.

[173] M. Nayrolles, A. Hamou-Lhadj, S. Tahar, and A. Larsson. A bug reproduction approach based on directed model checking and crash traces. *Journal of Software: Evolution and Process*, 29(3):e1789, mar 2017.

[174] G. News and Media Limited or its affiliated companies. Robot kills worker at Volkswagen plant in Germany, 2015.

[175] Oracle. What's New in JDK 8. `https://www.oracle.com/technetwork/java/javase/8-whats-new-2157071.html`, 2019. Accessed: 2019-05-14.

[176] A. Orso and G. Rothermel. Software testing: a research travelogue (2000–2014). In *Proceedings of the on Future of Software Engineering*, pages 117–132. ACM, 2014.

[177] C. Pacheco, S. K. Lahiri, M. D. Ernst, and T. Ball. Feedback-directed random test generation. In *Proceedings of the 29th International Conference on Software Engineering*, ICSE '07, pages 75–84, Washington, DC, USA, 2007. IEEE Computer Society.

[178] A. Panichella, F. M. Kifetew, and P. Tonella. Reformulating branch coverage as a many-objective optimization problem. In *2015 IEEE 8th International Conference on Software Testing, Verification and Validation (ICST)*, pages 1–10, April 2015.

[179] A. Panichella, R. Oliveto, M. Di Penta, and A. De Lucia. Improving multi-objective test case selection by injecting diversity in genetic algorithms. *IEEE Trans. Software Eng.*, 41(4):358–383, 2015.

[180] S. Panichella, A. Panichella, M. Beller, A. Zaidman, and H. C. Gall. The impact of test case summaries on bug fixing performance: an empirical investigation. In *Proceedings of the 38th International Conference on Software Engineering, ICSE 2016, Austin, TX, USA, May 14-22, 2016*, pages 547–558, 2016.

[181] C. Parnin and A. Orso. Are automated debugging techniques actually helping programmers? In *Proceedings of the 2011 international symposium on software testing and analysis*, pages 199–209. ACM, 2011.

[182] S. Parsa, M. Vahidi-Asl, S. Arabi, and B. Minaei-Bidgoli. Software fault localization using elastic net: A new statistical approach. In *International Conference on Advanced Software Engineering and Its Applications*, pages 127–134. Springer, 2009.

[183] S. L. Pfleeger and B. A. Kitchenham. Principles of survey research: part 1: turning lemons into lemonade. *ACM SIGSOFT Software Engineering Notes*, 26(6):16–18, 2001.

[184] I. S. W. B. Prasetya. *T3, a Combinator-Based Random Testing Tool for Java: Benchmarking*, pages 101–110. Springer International Publishing, Cham, 2014.

[185] I. W. B. Prasetya. T3, a combinator-based random testing tool for java: benchmarking. In *International Workshop on Future Internet Testing*, pages 101–110. Springer, 2013.

[186] W. Prasetya, T. Vos, and A. Baars. Trace-based reflexive testing of oo programs with t2. In *2008 1st International Conference on Software Testing, Verification, and Validation*, pages 151–160. IEEE, 2008.

[187] P. Puschner and R. Nossal. Testing the results of static worst-case execution-time analysis. In *Real-Time Systems Symposium, 1998. Proceedings. The 19th IEEE*, pages 134–143. IEEE, 1998.

[188] R. Ramler, D. Winkler, and M. Schmidt. Random test case generation and manual unit testing: Substitute or complement in retrofitting tests for legacy code? In *Software Engineering and Advanced Applications (SEAA), 2012 38th EUROMICRO Conference on*, pages 286–293. IEEE, 2012.

[189] J. Roche. Adopting DevOps practices in quality assurance. *Communications of the ACM*, 56(11):38–43, 2013.

[190] J. M. Rojas, J. Campos, M. Vivanti, G. Fraser, and A. Arcuri. Combining multiple coverage criteria in search-based unit test generation. In *International Symposium on Search Based Software Engineering*, pages 93–108. Springer, 2015.

[191] J. M. Rojas, G. Fraser, and A. Arcuri. Automated unit test generation during software development: A controlled experiment and think-aloud observations. In *Proceedings of the 2015 International Symposium on Software Testing and Analysis*, pages 338–349. ACM, 2015.

[192] J. M. Rojas, G. Fraser, and A. Arcuri. Seeding strategies in search-based unit test generation. *Software Testing, Verification and Reliability*, 26(5):366–401, aug 2016.

[193] J. M. Rojas, M. Vivanti, A. Arcuri, and G. Fraser. A detailed investigation of the effectiveness of whole test suite generation. *Empirical Software Engineering*, 22(2):852–893, 2017.

[194] J. Rößler, A. Zeller, G. Fraser, C. Zamfir, and G. Candea. Reconstructing core dumps. In *Proceedings - IEEE 6th International Conference on Software Testing, Verification and Validation, ICST 2013*, pages 114–123. IEEE, 2013.

[195] RxJava. Reactive Extensions for the JVM. `https://github.com/ReactiveX/RxJava`, 2018. [Online; accessed 25-January-2018].

[196] A. Sakti, G. Pesant, and Y.-G. Guéhéneuc. Instance generator and problem representation to improve object oriented code coverage. *IEEE Transactions on Software Engineering*, 41(3):294–313, 2014.

[197] A. Schroter, A. Schröter, N. Bettenburg, and R. Premraj. Do stack traces help developers fix bugs? In *2010 7th IEEE Working Conference on Mining Software Repositories (MSR 2010)*, pages 118–121. IEEE, 2010.

[198] S. S. Sharma, D. G. Kleinbaum, and L. L. Kupper. Applied regression analysis and other multivariate methods. 1978.

[199] S. E. Sim, S. Easterbrook, and R. C. Holt. Using Benchmarking to Advance Research: A Challenge to Software Engineering. In *Proceedings of the 25th International Conference on Software Engineering*, ICSE '03, pages 74–83, Portland, Oregon, USA, 2003. IEEE Computer Society.

[200] D. I. Sjøberg, J. E. Hannay, O. Hansen, V. B. Kampenes, A. Karahasanovic, N.-K. Liborg, and A. C. Rekdal. A survey of controlled experiments in software engineering. *IEEE transactions on software engineering*, 31(9):733–753, 2005.

[201] M. Soltani, A. Panichella, and A. van Deursen. Evolutionary testing for crash reproduction. In *Proceedings of the 9th International Workshop on Search-Based Software Testing - SBST '16*, pages 1–4, 2016.

[202] M. Soltani, A. Panichella, and A. van Deursen. A Guided Genetic Algorithm for Automated Crash Reproduction. In *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*, pages 209–220, Buenos Aires, Argentina, may 2017. IEEE.

[203] M. Soltani, A. Panichella, and A. van Deursen. A guided genetic algorithm for automated crash reproduction. In *Proceedings of the 39th International Conference on Software Engineering*, pages 209–220. IEEE Press, 2017.

[204] M. Soltani, A. Panichella, and A. van Deursen. Search-Based Crash Reproduction and Its Impact on Debugging. *Software Engineering, IEEE Transactions on*, 2018.

[205] J. Steven, P. Chandra, B. Fleck, and A. Podgurski. jrapture: A capture/replay tool for observation-based testing. In *Proceedings of the 2000 ACM SIGSOFT International Symposium on Software Testing and Analysis*, ISSTA '00, pages 158–167, New York, NY, USA, 2000. ACM.

[206] N. Tillmann and J. De Halleux. Pex: White box test generation for .net. In *Proceedings of the 2Nd International Conference on Tests and Proofs*, TAP'08, pages 134–153, Berlin, Heidelberg, 2008. Springer-Verlag.

[207] A. Vargha and H. D. Delaney. A critique and improvement of the cl common language effect size statistics of mcgraw and wong. *Journal of Educational and Behavioral Statistics*, 25(2):101–132, 2000.

[208] D. Weeratunge, X. Zhang, and S. Jagannathan. Analyzing multicore dumps to facilitate concurrency bug reproduction. *SIGARCH Comput. Archit. News*, 38(1):155–166, Mar. 2010.

[209] J. Wegener, A. Baresel, and H. Sthamer. Evolutionary test environment for automatic structural testing. *Information and Software Technology*, 43(14):841–854, 2001.

[210] Y. Wei, Y. Pei, C. A. Furia, L. S. Silva, S. Buchholz, B. Meyer, and A. Zeller. Automated fixing of programs with contracts. In *Proceedings of the 19th international symposium on Software testing and analysis*, pages 61–72. ACM, 2010.

[211] C. Weiss, R. Premraj, T. Zimmermann, and A. Zeller. How long will it take to fix this bug? In *Fourth International Workshop on Mining Software Repositories (MSR'07: ICSE Workshops 2007)*, pages 1–1. IEEE, 2007.

[212] R. Wettel, M. Lanza, and R. Robbes. Software systems as cities: A controlled experiment. In *Proceedings of the 33rd International Conference on Software Engineering*, pages 551–560. ACM, 2011.

[213] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén. *Experimentation in Software Engineering*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[214] X. Xiao, T. Xie, N. Tillmann, and J. De Halleux. Precise identification of problems for structural test generation. In *Software Engineering (ICSE), 2011 33rd International Conference on*, pages 611–620, Waikiki, Honolulu , HI, USA, 2011. IEEE, ACM.

[215] J. Xuan, X. Xie, and M. Monperrus. Crash reproduction via test case mutation: let existing test cases help. In *Proceedings of the 2015 10th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2015*, pages 910–913, New York, New York, USA, 2015. ACM Press.

[216] XWiki. The Advanced Open Source Enterprise and Application Wiki. `http://www.xwiki.org/`, 2018. [Online; accessed 25-January-2018].

[217] T. Yu, T. S. Zaman, and C. Wang. Descry: reproducing system-level concurrency failures. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pages 694–704. ACM, 2017.

[218] D. Yuan, Y. Luo, X. Zhuang, G. R. Rodrigues, X. Zhao, Y. Zhang, P. U. Jain, and M. Stumm. Simple testing can prevent most critical failures: An analysis of production failures in distributed data-intensive systems. In *Proceedings of*

the 11th USENIX Conference on Operating Systems Design and Implementation, OSDI'14, pages 249–265, Berkeley, CA, USA, 2014. USENIX Association.

[219] C. Zamfir and G. Candea. Execution synthesis: A technique for automated software debugging. In *Proceedings of the 5th European Conference on Computer Systems*, EuroSys '10, pages 321–334, New York, NY, USA, 2010. ACM.

[220] A. Zeller. Isolating cause-effect chains from computer programs. In *Proceedings of the 10th ACM SIGSOFT symposium on Foundations of software engineering*, pages 1–10. ACM, 2002.

[221] A. Zeller. *Why Programs Fail - A Guide to Systematic Debugging, 2nd Edition*. Academic Press, 2009.

[222] H. Zeng and D. Rine. Estimation of software defects fix effort using neural networks. In *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.*, volume 2, pages 20–21. IEEE, 2004.

[223] C. Zhang, J. Yang, D. Yan, S. Yang, and Y. Chen. Automated breakpoint generation for debugging. *JSW*, 8(3):603–616, 2013.

[224] T. Zimmermann, R. Premraj, N. Bettenburg, S. Just, A. Schroter, and C. Weiss. What makes a good bug report? *IEEE Transactions on Software Engineering*, 36(5):618–643, 2010.