2015 IEEE/ACM 10th International Workshop on Automation of Software Test

Metamorphic Testing: A Simple Method for Alleviating the Test Oracle Problem

Tsong Yueh Chen
Department of Computer Science and Software Engineering
Swinburne University of Technology
Hawthorn, VIC 3122, Australia
Email: tychen@swin.edu.au

***Abstract*—The test oracle problem is regarded as one of the most challenging problems in software testing. Metamorphic testing has been developed to alleviate this problem, which is done using the relations involving relevant inputs and their outputs. This keynote speech will provide a summary of the state-of-the-art of metamorphic testing.**

Index terms*—*Metamorphic Testing; Test Oracles

I. Introduction

A test oracle is a mechanism against which the correctness of program computed outputs can be verified. The oracle problem is said to occur (or the program is said to be non-testable) if the test oracle does not exist or it is practically infeasible to apply it to verify the correctness of the computed outputs. The test oracle problem is one of the most challenging problems in software testing, and occurs quite frequently. Metamorphic Testing (MT) was proposed to alleviate this problem [1]. MT is based on the simple intuition that although we may not be able to know the correctness of the computed output for any particular input, we may know the relation between relevant inputs and their outputs. Such a relation is a property of the algorithm to be implemented, and is referred to as a Metamorphic Relation (MR) of the relevant algorithm. For details of MT, readers may refer to [2] and [3].

Since its inception [1], research related to MT can be classified into three categories: use of MT in various non-testable application domains; integration of MT with other testing and debugging methods; and fundamentals and foundations of MT. We will cover each of these categories in individual sections as follows.

II. Application of MT to Various Application Domains

MT has been used to test various application domains including:

- Compilers [4] [5]
- Health and Medical Systems [6] [7] [8]
- Embedded applications [9] [10]
- Bioinformatics software [11]
- Feature Modelling software [12] [13]
- Search Engines [14]
- Numerical Analysis [15] [16] [17]
- Service Computing [18] [19]
- Simulation software [20]
- Image Processing Systems [21]
- Decision Support Systems [22]
- Machine Learning software [23] [24]
- Optimization software [25]

The most important task in all this research has been to identify the most effective MRs, i.e. those MRs which are more likely to reveal the faults in the implementation under test. Knowledge and experience obtained in these studies have helped improve testing of other non-testable programs of a similar nature.

III. Integration of MT with other Methods

Many testing, analysis and debugging methods assume the existence of a test oracle. For example, in the context of debugging, the basic idea of slicing is to focus the attention only on those statements that were executed by inputs which yielded a failure, rather than the whole program because the related faulty statement must be amongst the executed statements. In other words, the application of the technique of slicing in debugging has assumed that the program being debugged is a not a non-testable program. An obvious question is how to extend the applicability of slicing to debugging beyond the domain of testable programs. MT can help to solve this problem.

MT has been integrated with fault-based testing techniques [26], symbolic execution [2], and slicing and spectrum-based fault localization [27]. Furthermore, we are aware of ongoing research integrating MT with other techniques.

IV. Foundations and Fundamentals of MT

There has been some research helping to build the theoretical foundations for MT.

It has been found that MT can reveal faults in some open source software that had been extensively tested in the past. Such software includes the programs print_token, schedule and schedule_2 of the popular Siemens suite which has been extensively used as a benchmark by the software testing community [27] [28]. These results imply that MT can be regarded as a test case selection strategy complementary to existing test case selection methods. It will be of great interest and importance to investigate further along this direction.

978-1-4673-7022-6/15 $31.00 © 2015 IEEE
DOI 10.1109/AST.2015.18

With the exception of MR identification, MT can be easily automated, and investigations have already been conducted into how to automatically generate MRs [29] [30] [31] [32].

Another interesting direction for future investigation is the issue of how to select good MRs, because the results of such research will have an impact on other research areas of MT, such as the prioritization of MRs [33]. It will be of great interest to know the key factors that determine the testing effectiveness of MRs.

Some research has also been devoted to how to prove MRs, such as the work by Gotlieb and Botella [34], and Chen et. al [2].

MRs can be used to substitute the test oracle, and a recent study has investigated to what extent this is possible [3]. This investigation has provided new insights and identified new research directions.

REFERENCES

[1] T. Y. Chen, S. C. Cheung, S. M. Yiu, "Metamorphic testing: a new approach for generating next test cases," Department of Computer Science, Hong Kong University of Science and Technology, Tech. Rep. HKUST-CS98-01, 1998.

[2] T. Y. Chen, T. H. Tse, Z. Q. Zhou, "Semi-proving: An integrated method for program proving, testing, and debugging," *IEEE Transactions on Software Engineering*, vol. 37, no. 1, pp. 109–125, 2011.

[3] H. Liu, F.-C. Kuo, D. Towey, T. Y. Chen, "How effectively does metamorphic testing alleviate the oracle problem?" *IEEE Transactions on Software Engineering*, vol. 40, no. 1, pp. 4–22, 2014.

[4] Q. Tao, W. Wu, C. Zhao, W. Shen, "An automatic testing approach for compiler based on metamorphic testing technique," in *Proceedings of the 17th Asia Pacific Software Engineering Conference (APSEC 2010)*, 2010, pp. 270–279.

[5] V. Le, M. Afshari, Z. Su, "Compiler validation via equivalence modulo inputs," in *Proceedings of the 35th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI'14)*, 2014, pp. 216–226.

[6] C. Murphy, M. S. Raunak, A. King, S. Chen, G. Kaiser, I. Lee, O. Sokolsky, L. Clarke, L. Osterweil, "On effective testing of health care simulation software," in *Proceedings of the 3rd International Workshop on Software Engineering in Health Case (SEHC 2011)*, 2011, pp. 40–47.

[7] L. L. Pullum, O. Ozmen, "Early results from metamorphic testing of epidemiological models," in *Proceedings of the 2012 ASE/IEEE International Conference on BioMedical Computing*, 2012, pp. 62–67.

[8] A. Ramanathan, C. A. Steed, L. L. Pullum, "Verification of compartmental epidemiological models using metamorphic testing, model checking and visual analytics," in *Proceedings of the 2012 ASE/IEEE International Conference on BioMedical Computing*, 2012, pp. 68–73.

[9] W. K. Chan, T. Y. Chen, H. Lu, T. H. Tse, S. S. Yau, "Integration testing of context-sensitive middleware-based applications: A metamorphic approach," *International Journal of Software Engineering and Knowledge Engineering*, vol. 16, no. 5, pp. 677–703, 2006.

[10] F.-C. Kuo, T. Y. Chen, W. K. Tam, "Testing embedded software by metamorphic testing: A wireless metering system case study," in *Proceedings of the IEEE International Conference on Local Computer Networks*, 2011, pp. 291–294.

[11] T. Y. Chen, J. W. K. Ho, H. Liu, X. Xie, "An innovative approach for testing bioinformatics programs using metamorphic testing," *BMC Bioinformatics*, vol. 10, pp. 1–12, 2009.

[12] S. Segura, R. M. Hierons, D. Benavides, A. Ruiz-Corteés, "Automated test data generation on the analyses of feature models: a metamorphic testing approach," in *Proceedings of International Conference on Software Testing, Verification and Validation*, 2010, pp. 35–44.

[13] S. Segura, R. M. Hierons, D. Benavides, A. Ruiz-Cortés, "Automated metamorphic testing on the analyses of feature models," *Information and Software Technology*, vol. 53, no. 3, pp. 245–258, 2011.

[14] Z. Q. Zhou, S. J. Zhang, M. Hagenbuchner, T. H. Tse, F.-C. Kuo, T. Y. Chen, "Automated functional testing of online search services," *Software Testing, Verification and Reliability*, vol. 22, no. 4, pp. 221–243, 2012.

[15] F. T. Chan, T. Y. Chen, S. C. Cheung, M. F. Lau, S. M. Yiu, "Application of metamorphic testing in numerical analysis," in *Proceedings of the IASTED International Conference on Software Engineering (SE'98)*, 1998, pp. 191–197.

[16] T. Y. Chen, J. Feng, T. H. Tse, "Metamorphic testing of programs on partial differential equations: a case study," in *Proceedings of the Annual International Computer Software and Applications Conference*, 2002, pp. 327–333.

[17] Z. Hui, S. Huang, Z. Ren, Y. Yao, "Metamorphic testing integer overflow faults of mission critical program: A case study," *Mathematical Problems in Engineering*, vol. 2013, pp. 1–6, 2013.

[18] W. K. Chan, S. C. Cheung, K. R. P. Leung, "A metamorphic testing approach for online testing of service-oriented software applications," *International Journal of Web Services Research*, vol. 4, no. 2, pp. 61–81, 2007.

[19] C. Castro-Cabrera, I. Medina-Bulo, "Application of metamorphic testing to a case study in web services compositions," *E-Business and Telecommunications Communications in Computer and Information Science*, vol. 314, pp. 168–181, 2012.

[20] T. Y. Chen, F.-C. Kuo, R. Merkel, W. K. Tam, "Testing an open source suite for open queuing network modelling using metamorphic testing technique," in *Proceedings of the IEEE International Conference on Engineering of Complex Computer Systems*, 2009, pp. 23–29.

[21] J. Ding, T. Wu, J. Q. Lu, X.-H. Hu, "Self-checked metamorphic testing of an image processing program," in *Proceedings of the International Conference on Secure Software Integration and Reliability Improvement*, 2010, pp. 190–197.

[22] F.-C. Kuo, Z. Q. Zhou, J. Ma, G. Zhang, "Metamorphic testing of decision support systems: a case study," *Software IET*, vol. 4, no. 4, pp. 294–301, 2010.

[23] X. Xie, J. W. K. Ho, C. Murphy, G. Kaiser, B. Xu, T. Y. Chen, "Testing and validating machine learning classifiers by metamorphic testing," *Journal of Systems and Software*, vol. 84, no. 4, pp. 544–558, 2011.

[24] C. Murphy, G. Kaiser, L. Hu, L. Wu, "Properties of machine learning applications for use in metamorphic testing," in *Proceedings of the International Conference on Software Engineering and Knowledge Engineering*, 2008, pp. 867–872.

[25] S. Yoo, "Metamorphic testing of stochastic optimisation," in *Proceedings of the International Conference on Software Testing, Verification, and Validation Workshops*, 2010, pp. 192–201.

[26] T. Y. Chen, T. H. Tse, Z. Q. Zhou, "Fault-based testing without the need of oracles," *Information Software and Technology*, vol. 45, no. 1, pp. 1–9, 2003.

[27] X. Xie, W. E. Wong, T. Y. Chen, B. Xu, "Metamorphic slice: an application in spectrum-based fault localization," *Information and Software Technology*, vol. 55, no. 5, pp. 866–879, 2013.

[28] P. Rao, Z. Zheng, T. Y. Chen, N. Wang, K. Cai, "Impacts of test suite's class imbalance on spectrum-based fault localization techniques," in *Proceedings of the International Conference on Quality Software*, 2013, pp. 260–267.

[29] H. Liu, X. Liu, T. Y. Chen, "A new method for constructing metamorphic relations," in *Proceedings of the International Conference on Quality Software*, 2012, pp. 59–68.

[30] U. Kanewala, J. M. Bieman, "Using machine learning techniques to detect metamorphic relations for programs without test oracles," in *Proceedings of IEEE 24th International Symposium on Software Reliability Engineering*, 2013, pp. 1–10.

[31] U. Kanewala, "Techniques for automatic detection of metamorphic relations," in *Proceedings of IEEE Seventh International Conference on Software Testing, Verification and Validation Workshops*, 2014, pp. 237–238.

[32] J. Zhang, J. Chen, D. Hao, Y. Xiong, B. Xie, L. Zhang, H. Mei, "Search-based inference of polynomial metamorphic relations," in *Proceedings of the 29th IEEE/ACM International Conference on Automated Software Engineering (ASE 2014)*, 2014, pp. 701–712.

[33] J. Mayer, R. Guderlei, "An empirical study on the selection of good metamorphic relations," in *Proceedings of the 30th Annual International Computer Software and Applications Conference (COMPSAC 2006)*, 2006, pp. 475–484.

[34] A. Gotlieb, B. Botella, "Automated metamorphic testing," in *Proceedings of the 27th Annual International Computer Software and Applications Conference (COMPSAC '2003)*, 2003, pp. 34–40.