Research Directions in Model-Based Metamorphic Testing and Verification [*]

T. H. Tse
Department of Computer Science
The University of Hong Kong, Pokfulam, Hong Kong
thtse@cs.hku.hk

1. Introduction

Model-based software testing generally refers to test case selection and result evaluation based on a behavioral model of the target software. It may not, however, be possible to evaluate test results in some situations. For embedded software in a real-time environment, for instance, the results may only appear in a fraction of a second and cannot be observed by the tester. In this paper, we discuss the research opportunities for such situations despite the challenges.

2. Metamorphic Testing

We have proposed a metamorphic testing method [2, 3]. to address situations where there is no easy mechanism to evaluate test results. Intuitively, even if we may not have sufficient information to evaluate a test case, the latter may still have useful information. Follow-up test cases should be further constructed from the original test cases with reference to some necessary conditions of the problem to be implemented. Such necessary properties are known as *metamorphic relations*. This method has been applied to mathematical software such as the solutions of partial differential equations [2].

3. Research Directions

(a) Model-Based Metamorphic Testing: Metamorphic testing is not only useful to conventional mathematical software. It can be applied to real-time embedded systems specified by a dynamic behavioral model.

Consider context-sensitive software, for example. The streetlights in a smart streetlight system should adjust themselves according to the locations of the visitors in the proximity [4]. We use isotropic properties of the contexts as metamorphic relations for testing. More specifically, the software should provide equivalent levels of illumination to the same visitor at various locations within the streetlight zone.

[*] This work is supported in part by a grant of the Research Grants Council of Hong Kong and a grant of the University of Hong Kong.

Using the concept of observational equivalence in object-oriented algebraic specifications [1] as a special form of metamorphic relations, our method has also been successfully applied in a multi-million dollar industrial project for ASM, who is the world's largest supplier of semiconductor assembly and packaging equipment.

(b) Model-Based Metamorphic Verification: We also propose a method to integrate testing, proving, and debugging. Instead of testing the model-based metamorphic relations, we prove them using global symbolic evaluation [3]. If a relation cannot be satisfied, our method guarantees to identify failure-causing test cases. The diagnostic information accompanying these cases will support debugging.

(c) "Metamorphic Model"-Based Testing: Conventional model-based testing is treated an algorithm problem in which the model has already been defined and fixed. Instead, we propose to treat the testing process as a control problem in software cybernetics. Metamorphic relations are not just a means to by-pass testing problems. The relations are part of the model for the target software. The "metamorphic model"-based testing method serves as a controller and the integrated model and the target software serve as the controlled objects.

References

[1] H. Y. Chen, T. H. Tse, and T. Y. Chen. TACCLE: a methodology for object-oriented software testing at the class and cluster levels. *ACM Transactions on Software Engineering and Methodology*, 10 (1): 56–109, 2001.

[2] T. Y. Chen, J. Feng, and T. H. Tse. Metamorphic testing of programs on partial differential equations: a case study. In *Proceedings of the 26th Annual International Computer Software and Applications Conference* (*COMPSAC 2002*), pages 327–333. IEEE Computer Society Press, Los Alamitos, California, 2002.

[3] T. Y. Chen, T. H. Tse, and Z. Q. Zhou. Semi-proving: an integrated method based on global symbolic evaluation and metamorphic testing. In *Proceedings of the ACM SIGSOFT International Symposium on Software Testing and Analysis* (*ISSTA 2002*), pages 191–195. ACM Press, New York, 2002.

[4] T. H. Tse, S. S. Yau, W. K. Chan, H. Lu, and T. Y. Chen. Testing context-sensitive middleware-based software applications. In *Proceedings of the 28th Annual International Computer Software and Applications Conference* (*COMPSAC 2004*), volume 1, pages 458–465. IEEE Computer Society Press, Los Alamitos, California, 2004.