Bachelor/Master Thesis Topic

Automated Validation of Patch Correctness and Maintainability with Symbolic Execution

Motivation and Background

Automated patch validation for real-world software has been drawing growing attention in the field. Many tools have been developed to examine the correctness of patched programs. For example, the-state-of-art KATCH [1] aims to automatically generate high coverage tests for patches to discover regression bugs, while other tools leverage test/execution similarity between original buggy and patched programs [2] or investigate in better oracle for automatically generated patches [3, 4]. In these tools, techniques such as dynamic symbolic execution [5, 6] are usually involved. As for patch maintainability, despite of previous contribution to the maintainability problem of patches [7, 8] or programs [9], the evaluation of the maintainability of patches still lack standard. The challenge of this thesis is to identify both the correctness and the maintainability of (manually or automatically generated) patches.

Goals

The student uses techniques such as symbolic execution and dynamic program analysis to validate patches (e.g. discovering regression bug/crashes). After that, the student should further evaluate the patches' maintainability from different aspects (e.g. syntax, semantic and AST [10]), which may be concluded in some metrics. The validation of the resulting tool can be conducted on real-world patches or patches generated by automated program repair tools.

Description of the Task

A detailed description of the task and the underlying techniques will be given personally on interest.

Research Type

Theoretical Aspects:	****☆
Industrial Relevance:	****☆
Implementation	****☆

Prerequisite

The student should be enrolled in the bachelor/master of software engineering or bachelor/master of computer science program, and has completed the required course modules to start a bachelor/master thesis.

Skills required

Programming skills in (preferably) Java and C/C++, Understanding of, or willingness to learn, the architectural and statistical foundations needed for the project.

References

[1] Paul Dan Marinescu and Cristian Cadar. 2013. KATCH: high-coverage testing of software patches. In Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2013). ACM, New York, NY, USA, 235-245. DOI: https://doi.org/10.1145/2491411.2491438

[2] Yingfei Xiong, Xinyuan Liu, Muhan Zeng, Lu Zhang, and Gang Huang. 2018. Identifying patch correctness in test-based program repair. In Proceedings of the 40th International Conference on Software Engineering (ICSE '18). ACM, New York, NY, USA, 789-799. DOI: https://doi.org/10.1145/3180155.3180182

[3] Jinqiu Yang, Alexey Zhikhartsev, Yuefei Liu, and Lin Tan. 2017. Better test cases for better automated program repair. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2017). ACM, New York, NY, USA, 831-841. DOI: https://doi.org/10.1145/3106237.3106274

[4] Qi Xin and Steven P. Reiss. 2017. Identifying test-suite-overfitted patches through test case generation. In Proceedings of the 26th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2017). ACM, New York, NY, USA, 226-236. DOI: https://doi.org/10.1145/3092703.3092718

[5] James C. King. 1976. Symbolic execution and program testing. Commun. ACM 19, 7 (July 1976), 385-394. DOI=http://dx.doi.org/10.1145/360248.360252

[6] Chen, T., Zhang, X., Guo, S., Li, H., & Wu, Y. (2013). State of the art: Dynamic symbolic execution for automated test generation. *Future Generation Comp. Syst., 29*, 1758-1773.

[7] Zachary P. Fry, Bryan Landau, and Westley Weimer. 2012. A human study of patch maintainability. In Proceedings of the 2012 International Symposium on Software Testing and Analysis (ISSTA 2012). ACM, New York, NY, USA, 177-187. DOI=http://dx.doi.org/10.1145/2338965.2336775

[8] Qi, Zichao, Fan Long, Sara Achour and Martin C. Rinard. "An Analysis of Patch Plausibility and Correctness for Generate-And-Validate Patch Generation Systems (Supplementary Material)." *ISSTA* (2015).

[9] Ilja Heitlager, Tobias Kuipers, and Joost Visser. 2007. A Practical Model for Measuring Maintainability. In Proceedings of the 6th International Conference on Quality of Information and Communications Technology (QUATIC '07). IEEE Computer Society, Washington, DC, USA, 30-39. DOI: https://doi.org/10.1109/QUATIC.2007.7

[10] Xuan-Bach D. Le, Duc-Hiep Chu, David Lo, Claire Le Goues, and Willem Visser. 2017. S3: syntax- and semantic-guided repair synthesis via programming by examples. In Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2017). ACM, New York, NY, USA, 593-604. DOI: https://doi.org/10.1145/3106237.3106309

Contacts

Lars Grunske/Minxing Tang, Humboldt-Universität zu Berlin, Institut für Informatik, Lehrstuhl Software Engineering, Unter den Linden 6, 10099 Berlin, Germany

Application

Please contact during office hours or write an email with the title: "AV-PCM-SE" to se-career@informatik.hu-berlin.de