1. A critical review on the evaluation of automated program repair systems

Kui Liu (1); Li, L. (2); Koyuncu, A. (3); Kim, D. (4); Zhe Liu (1); Klein, J. (3); Bissyandé, T.F. (3)

Source: *Journal of Systems and Software*, v 171, p 110817 (13 pp.), Jan. 2021; **ISSN:** 0164-1212; **DOI:** 10.1016/j.jss.2020.110817; **Publisher:** Elsevier B.V., Netherlands

Author affiliation: (1) Nanjing University of Aeronautics and Astronautics, College of Computer Science and Technology, China (2) Monash University, Faculty of Information Technology, Clayton, VIC, Australia (3) University of Luxembourg, Interdisciplinary Centre for Security, Luxembourg (4) Kyungpook National University, School of Computer Science and Engineering, Korea, Republic of

Abstract: Automated Program Repair (APR) has attracted significant attention from software engineering research and practice communities in the last decade. Several teams have recorded promising performance in fixing real bugs and there is a race in the literature to fix as many bugs as possible from established benchmarks. Gradually, repair performance of APR tools in the literature has gone from being evaluated with a metric on the number of generated plausible patches to the number of correct patches. This evolution is necessary after a study highlighting the overfitting issue in test suite-based automatic patch generation. Simultaneously, some researchers are also insisting on providing time cost in the repair scenario as a metric for comparing state-of-the-art systems.In this paper, we discuss how the latest evaluation metrics of APR systems could be biased. Since design decisions (both in approach and evaluation setup) are not always fully disclosed, the impact on repair performance is unknown and computed metrics are often misleading. To reduce notable biases of design decisions in program repair approaches, we conduct a critical review on the evaluation of patch generation systems and propose eight evaluation metrics for fairly assessing the performance of APR tools. Eventually, we show with experimental data on 11 baseline program repair systems that the proposed metrics allow to highlight some caveats in the literature. We expect wide adoption of these metrics in the community to contribute to boosting the development of practical, and reliably performable program repair tools. [All rights reserved Elsevier]. (0 refs)

Inspec controlled terms: program debugging - program testing - program verification - software maintenance - software reliability

Uncontrolled terms: time cost - critical review - baseline program repair systems - bugs - practice communities - software engineering research - automated program repair systems - reliably performable program repair tools - design decisions - APR systems - latest evaluation metrics - test suite-based automatic patch generation - generated plausible patches - APR tools

Classification Code: C6150G Diagnostic, testing, debugging and evaluating systems - C6110B Software engineering techniques

IPC Code: G06F9/44 - G06F11/36

Treatment: Bibliography (BIB) - Practical (PRA)

Database: Inspec

Data Provider: Engineering Village