Automatic Software Repair: A Survey

Extended Abstract

Luca Gazzola
Università degli Studi di Milano-Bicocca
luca.gazzola@disco.unimib.it

Daniela Micucci
Università degli Studi di Milano-Bicocca
micucci@disco.unimib.it

Leonardo Mariani
Università degli Studi di Milano-Bicocca
mariani@disco.unimib.it

ACM Reference Format:
Luca Gazzola, Daniela Micucci, and Leonardo Mariani. 2018. Automatic Software Repair: A Survey: Extended Abstract. In *ICSE '18: 40th International Conference on Software Engineering , May 27-June 3, 2018, Gothenburg, Sweden.* ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3180155.3182526

Debugging software failures is still a painful, time consuming, and expensive process. For instance, recent studies showed that debugging activities often account for about 50% of the overall development cost of software products [3]. There are many factors contributing to the cost of debugging, but the most impacting one is the extensive manual effort that is still required to identify and remove faults. So far, the automation of debugging activities essentially resulted in the development of techniques that provide useful insights about the possible locations of faults, the inputs and states of the application responsible for the failures, as well as the anomalous operations executed during failures. However, developers must still put a relevant effort on the analysis of the failed executions to exactly identify the faults that must be fixed. In addition, these techniques do not help the developers with the synthesis of an appropriate fix.

Recently, researchers focused on a new class of approaches, namely program repair techniques [1, 4], whose key idea is to try to automatically repair software systems by producing an actual fix that can be validated by the testers before it is finally accepted, or that can be adapted to properly fit the system. The benefit of using these techniques is that the fix both explains the reason of the failure and provides a possible solution to the problem, thus alleviating the effort necessary to identify and correct faults. Since program repair techniques have the potential to dramatically reduce the debugging effort, they attracted the interest of many researchers who produced several approaches for repairing different classes of faults under different conditions and hypotheses. The proposed algorithms, techniques, and heuristics have been integrated, experimented, and studied, producing a heterogeneous and articulated research framework where automatic repair techniques are proliferating. Important results have been already achieved, but at the same time these results revealed the existence of relevant challenges that should be faced.

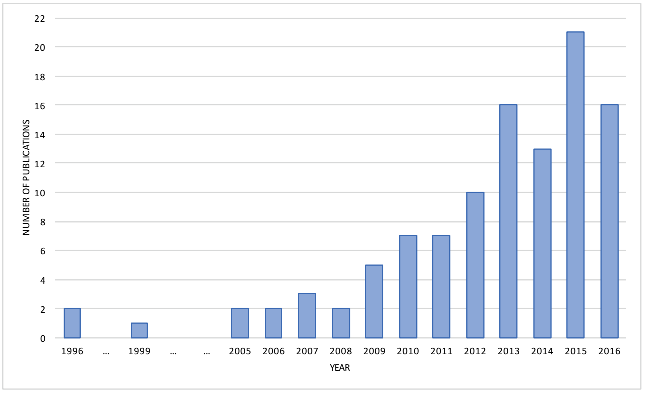

Figure 1: Publications per year from 1996 to 2016.

Interest in program repair techniques greatly increased in the last few years, as shown in Figure 1, our TSE paper [2] contributes to the research on such techniques by surveying a body of 108 papers about automatic software repair techniques, illustrating the algorithms and the approaches, comparing them on representative examples, critically analyzing their capabilities in repairing faulty programs, surveying the results achieved so far and summarizing them into a set of key facts that show the trade-offs and factors influencing the effectiveness of these approaches, and identifying the key open challenges that we believe are the most important and that can influence the future research in the area.

Keywords: Automatic Program Repair, Generate and Validate, Search-Based, Semantics-driven repair, Correct by Construction, Program Synthesis, Self-Repairing.

ACKNOWLEDGMENTS

This work has been partially supported by the EU H2020 "Learn" project, which has been funded under the ERC Consolidator Grant 2014 program (ERC Grant Agreement n. 646867) and the "GAUSS" national research project, which has been funded by the MIUR under the PRIN 2015 program (Contract 2015KWREMX).

REFERENCES

[1] A. Arcuri. 2008. On the automation of fixing software bugs. In *Companion of the 30th International Conference on Software Engineering.*
[2] L. Gazzola, D. Micucci, and L. Mariani. to appear. Automatic software repair: A survey. *IEEE Transactions on Software Engineering* (to appear).
[3] T. Britton L., Jeng, G. Carver, and P. Cheak. 2013. Reversible Debugging Software - quantify the time and cost saved using reversible debuggers. (2013).
[4] W. Weimer, T. Nguyen, C. Le Goues, and S. Forrest. 2009. Automatically finding patches using genetic programming. In *Proceedings of the 31st International Conference on Software Engineering.*