Summary of Search-based Crash Reproduction using Behavioral Model Seeding

Derakhshanfar, P.; Devroey, Xavier; Perrouin, Gilles; Zaidman, A.E.; van Deursen, A.

DOI
10.1109/ICST49551.2021.00039

Publication date
2021

Document Version
Final published version

Published in
2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST)

2021 14th IEEE Conference on Software Testing, Verification and Validation (ICST) | 978-1-7281-6836-4/20/$31.00 ©2021 IEEE | DOI: 10.1109/ICST49551.2021.00039

Summary of Search-based Crash Reproduction using Behavioral Model Seeding

Pouria Derakhshanfar*, Xavier Devroey*, Gilles Perrouin†, Andy Zaidman* Arie van Deursen*
* Delft University of Technology, Delft, Netherlands. Emails: p.derakhshanfar@tudelft.nl, x.d.m.devroey@tudelft.nl, a.e.zaidman@tudelft.nl, arie.vandeursen@tudelft.nl
† PReCISE, NADI, University of Namur, Namur, Belgium. Email: gilles.perrouin@unamur.be

***Abstract*—This is an extended abstract of the article: Pouria Derakhshanfar, Xavier Devroey, Gilles Perrouin, Andy Zaidman and Arie van Deursen. 2019. Search-based crash reproduction using behavioural model seeding. In: Software Testing, Verification and Reliability (May 2020). http://doi.org/10.1002/stvr.1733.**

***Index Terms*—model seeding, seed learning, crash reproduction, search-based software testing**

Search-based crash reproduction approaches assist developers during debugging by generating a test case, which reproduces a crash given its stack trace. One of the fundamental steps of this approach is creating objects needed to trigger the crash [1]. One way to overcome this limitation is seeding: using information about the application during the search process [2]. With seeding, existing classes usages participate in the search process to produce realistic sequences of method calls, which create the required objects.

In our study [3], we introduced behavioural model seeding: a new seeding method that learns class usages from both the system under test and existing test cases. We synthesized learned usages in a behavioural model (*i.e.,* a transition system) [4]. Then, this model serves to guide the evolutionary process.

To assess behavioural model seeding, we evaluated it against test seeding (the state-of-the-art technique for seeding realistic objects used in unit test generation) [2] and no seeding (without seeding any class usage). For our evaluation, we used a benchmark of 122 hard-to-reproduce crashes stemming from six open-source projects [1], [5].

Our results indicate that model seeding outperforms other seeding approaches in all aspects: crash reproduction effectiveness, efficiency, and search process initialization rate. Model seeding increases the number of reproduced crashes by 7% and 6% compared to no seeding and test seeding, respectively. We manually investigated the improvements and outline three factors: dissimilarity between call sequences when sampling them from behaviour models, learning behavioural models from multiple information sources, and prioritizing classes to use for seeding.

This research was partially funded by the EU Horizon 2020 ICT-10-2016-RIA "STAMP" project (No.731529), the EU Horizon 2020 H2020-ICT-2020-1-RIA "COSMOS" project (No.957254), Andy Zaidman's "TestShift" (VI.C.182.032) project from the Dutch Science Foundation NWO, and the Dutch 4TU project "Big Software on the Run" project. Gilles Perrouin is an FNRS Research associate.

In summary, we made the following contributions:

1) we provide an evaluation of test seeding techniques applied to search-based crash reproduction,
2) we design a novel behavioural model seeding strategy applied to search-based crash reproduction,
3) we offer an open-source implementation of test seeding and model seeding strategies in the Botsing framework [6], and
4) we further discuss our model-seeding improvements in our replication package [7].

Our article is available open access at http://doi.org/10.1002/stvr.1733. The latest version of our implementation of model seeding for crash reproduction is available at https://github.com/STAMP-project/botsing.

References

[1] M. Soltani, P. Derakhshanfar, X. Devroey, and A. van Deursen, "A benchmark-based evaluation of search-based crash reproduction," *Empirical Software Engineering*, vol. 25, no. 1, pp. 96–138, jan 2020.

[2] J. M. Rojas, G. Fraser, and A. Arcuri, "Seeding strategies in search-based unit test generation," *Software Testing, Verification and Reliability*, vol. 26, no. 5, pp. 366–401, Aug. 2016.

[3] P. Derakhshanfar, X. Devroey, G. Perrouin, A. Zaidman, and A. Deursen, "Search-based crash reproduction using behavioural model seeding," *Software Testing, Verification and Reliability*, vol. 30, no. 3, p. e1733, may 2020.

[4] X. Devroey, G. Perrouin, M. Cordy, H. Samih, A. Legay, P.-Y. Schobbens, and P. Heymans, "Statistical prioritization for software product line testing: an experience report," *Software & Systems Modeling*, vol. 16, no. 1, pp. 153–171, feb 2017.

[5] P. Derakhshanfar and X. Devroey, "Jcrashpack: A java crash reproduction benchmark," Apr. 2020. [Online]. Available: https://doi.org/10.5281/zenodo.3766689

[6] P. Derakhshanfar, X. Devroey, A. Panichella, A. Zaidman, and A. Van Deursen, "Botsing, a Search-based Crash Reproduction Framework for Java," in *35th IEEE/ACM International Conference on Automated Software Engineering (ASE '20), September 21–25, 2020, Virtual Event, Australia*. ACM/IEEE, aug 2020, pp. 1278–1282.

[7] P. Derakhshanfar, X. Devroey, G. Perrouin, A. Zaidman, and A. van Deursen, "Replication package of "Search-based Crash Reproduction using Behavioral Model Seeding"," Oct. 2019. [Online]. Available: https://doi.org/10.5281/zenodo.3673916

978-1-7281-6836-4/21/$31.00 ©2021 IEEE
DOI 10.1109/ICST49551.2021.00039