Scalable and Precise Static Analysis of JavaScript Applications via Loop-Sensitivity (Artifact)

Changhee Park and Sukyoung Ryu

Department of Computer Science, KAIST, Daejeon, Republic of Korea
{changhee.park,sryu.cs}@kaist.ac.kr

Abstract

This artifact is an implementation of the loop-sensitive analysis (LSA) technique that can improve analysis precision and scalability in analyzing JavaScript applications by distinguishing loop iterations automatically during analysis. It has been built on SAFE, the open-source JavaScript static analyzer framework, and its package contains all benchmarks that we used in the companion ECOOP paper for evaluation.

1998 ACM Subject Classification F.3.2 Semantics of Programming Languages

Keywords and phrases JavaScript, static analysis, loops

Digital Object Identifier 10.4230/DARTS.1.1.12

Related Article Changhee Park and Sukyoung Ryu, "Scalable and Precise Static Analysis of JavaScript Applications via Loop-Sensitivity", in Proceedings of the 29th European Conference on Object-Oriented Programming (ECOOP 2015), LIPIcs, Vol. 37, pp. 735–756, 2015.
http://dx.doi.org/10.4230/LIPIcs.ECOOP.2015.735

Related Conference 29th European Conference on Object-Oriented Programming (ECOOP 2015), July 5–10, 2015, Prague, Czech Republic

1 Scope

The artifact is designed to support repeatability of all the experiments of the companion paper. The artifact is an implementation of the loop-sensitive analysis (LSA) technique that can improve analysis precision and scalability in analyzing JavaScript applications by distinguishing loop iterations automatically during analysis. It has been build on SAFE [1, 2], the open-source JavaScript static analyzer framework, as an extension named SAFE_{lsa} and it provides an option to enable or disable our new technique making it simple to compare analysis results of SAFE and SAFE_{lsa}. The package contains all benchmarks we used in the experiments and the result from running the artifact with some target programs provides statistical data that we used as criteria for the evaluation of our new technique as well as analysis results.

2 Content

The artifact package includes:

- Scala and Java sources of SAFE_{lsa};
- benchmarks we used for all the experiments of the companion paper;
- an `index.html` file to include detailed instructions about how to build and use SAFE_{lsa}.

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS). The latest version of the artifact is available at `http://plrg.kaist.ac.kr/pch`.

© Changhee Park and Sukyoung Ryu;
licensed under Creative Commons Attribution 3.0 Germany (CC BY 3.0 DE)

4 Tested platforms

The artifact is known to work on platforms with operating systems with a Unix-style shell; for example, Linux, Mac OS X, or Cygwin on Windows. It requires J2SDK[1] 1.6 or 1.7, Ant[2] 1.6.5 or later, and Bash 2.5 or later. We performed all experiments in the companion paper on a Mac OS X x64 machine with 3.4GHz Intel Core i7 CPU and 16GB Memory.

5 License

BSD (`http://opensource.org/licenses/BSD-2-Clause`)

6 MD5 sum of the artifact

07EBBEF865C2DF6AD2A2487AC9CDD315

7 Size of the artifact

129.2 MB

Acknowledgements. This work is supported in part by Korea Ministry of Education, Science and Technology(MEST) / National Research Foundation of Korea(NRF) (Grants NRF-2014R1A2A2A01003235 and NRF-2008-0062609), Samsung Electronics, and Google.

References

1 KAIST PLRG. SAFE: Scalable Analysis Framework for ECMAScript. `http://safe.kaist.ac.kr`, 2014.

2 Hongki Lee, Sooncheol Won, Joonho Jin, Junhee Cho, and Sukyoung Ryu. SAFE: Formal specification and implementation of a scalable analysis framework for ECMAScript. In *FOOL'12: International Workshop on Foundations of Object Oriented Languages*, 2012.

[1] `http://java.sun.com/javase/downloads/index.jsp`
[2] `http://ant.apache.org/bindownload.cgi`