Garbage-Free Abstract Interpretation Through Abstract Reference Counting (Artifact)

Noah Van Es
Software Languages Lab, Vrije Universiteit Brussel, Belgium
noah.van.es@vub.be

Quentin Stiévenart
Software Languages Lab, Vrije Universiteit Brussel, Belgium
quentin.stievenart@vub.be

Coen De Roover
Software Languages Lab, Vrije Universiteit Brussel, Belgium
coen.de.roover@vub.be

Abstract

This artifact is a modified version of Scala-AM, an abstract interpretation framework implemented in Scala. Specifically, we extended Scala-AM with several implementations of machine abstractions that each employ a different approach to abstract garbage collection. These include traditional (tracing-based) approaches to abstract garbage collection, as well as our own novel approach using abstract reference counting. In particular, using the machine abstraction that employs abstract reference counting (with cycle detection) results in a garbage-free abstract interpreter can greatly improve both the precision and performance of the corresponding machine abstraction in the original version of the Scala-AM framework.

We have set up the framework in such a way that one can easily run a variety of experiments to use, evaluate and compare these approaches to abstract garbage collection. This artifact contains documentation on how these experiments can be configured, specifically to reproduce the results presented in the companion paper.

2012 ACM Subject Classification Theory of computation → Program analysis

Keywords and phrases static analysis, abstract interpretation, abstract garbage collection, reference counting

Digital Object Identifier 10.4230/DARTS.5.2.7

Funding *Noah Van Es*: Funded by a PhD Fellowship of the Research Foundation - Flanders (FWO)

Related Article Noah Van Es, Quentin Stiévenart, and Coen De Roover, "Garbage-Free Abstract Interpretation Through Abstract Reference Counting", in 33rd European Conference on Object-Oriented Programming (ECOOP 2019), LIPIcs, Vol. 134, pp. 10:1–10:33, 2019.
`https://dx.doi.org/10.4230/LIPIcs.ECOOP.2019.10`

Related Conference 33rd European Conference on Object-Oriented Programming (ECOOP 2019), July 15–19, 2019, London, United Kingdom

1 Scope

This artifact implements abstract reference counting, our novel approach to abstract garbage collection that is presented in the companion paper, as well as existing tracing-based approaches to abstract garbage collection as an extension to the Scala-AM framework [1, 2]. While the formalization in the paper uses a minimalistic language λ_{ANF}, this implementation can be used to analyze a larger subset of the Scheme programming language, and in addition supports multiple configurations for the abstract domain and context-sensitivity of the analysis.

We have set up the framework in such a way that the experiments and results that are reported in the companion paper can easily be reproduced. In particular, the framework can directly be used to compare the precision, performance and overhead of analyses that employ different approaches to abstract garbage collection.

2 Content

The artifact package includes:

- a manual (`ecoop2019arc-artifact-manual.pdf`) that briefly describes our implementation (i.e., a modified version of the Scala-AM framework) and provides detailed instructions on how it can be used, in particular to reproduce the experiments of the companion paper.
- the source code of our implementation (`scala-am-abstractgc.zip`), which can be run locally.
- a VM image (`ecoop2019arc-artifact-vm.ova`) that comes pre-loaded with our implementation and all the dependencies that are required to run it.

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 source code of the artifact is also available at: `https://github.com/noahvanes/scala-am-abstractgc`. Moreover, the detailed instructions for reproducing the experiments conducted in the companion paper are accessible at `https://soft.vub.ac.be/~noahves/ecoop2019arc/ecoop2019arc-artifact-manual.pdf`.

4 Tested platforms

The artifact can be installed on any platform running the Java Virtual Machine, version 8 or more recent. The provided VM image (.ova) requires around 7.5 GB of free space on disk, and we recommend using it with at least 4GB of RAM.

5 License

The artifact is available under the MIT license (`https://opensource.org/licenses/MIT`).

6 MD5 sum of the artifact

8d8ac158bb40ecd3bf4c94727787fa4b

7 Size of the artifact

3.76 GiB

References

1 Quentin Stiévenart, Jens Nicolay, Wolfgang De Meuter, and Coen De Roover. Building a modular static analysis framework in Scala (tool paper). In *Proceedings of the 2016 7th ACM SIGPLAN Symposium on Scala*, pages 105–109. ACM, 2016.

2 Quentin Stiévenart, Maarten Vandercammen, Wolfgang De Meuter, and Coen De Roover. Scala-AM: A modular static analysis framework. In *Source Code Analysis and Manipulation (SCAM), 2016 IEEE 16th International Working Conference on*, pages 85–90. IEEE, 2016.