Flow-Sensitive Type-Based Heap Cloning (Artifact)

Mohamad Barbar
University of Technology Sydney, Australia
CSIRO's Data61, Sydney, Australia

Yulei Sui
University of Technology Sydney, Australia

Shiping Chen
CSIRO's Data61, Sydney, Australia

Abstract

This artifact contains our implementation of a new flow-sensitive type-based points-to analysis, described in "Flow-Sensitive Type-Based Heap Cloning" by Mohamad Barbar, Yulei Sui, and Shiping Chen (ECOOP 2020). This analysis performs heap cloning based on C and C++ types rather than calling contexts. Packaged as a Docker image, the artifact allows users to reproduce the claims made in the "Evaluation" section of the associated paper (Section 5.2) and to build and analyse arbitrary software.

2012 ACM Subject Classification Software and its engineering → Automated static analysis

Keywords and phrases Heap cloning, type-based analysis, flow-sensitivity

Digital Object Identifier 10.4230/DARTS.6.2.1

Funding This research is supported by Australian Research Grant DP200101328.
Mohamad Barbar: supported by a PhD scholarship funded by CSIRO's Data61.

Related Article Mohamad Barbar, Yulei Sui, and Shiping Chen, "Flow-Sensitive Type-Based Heap Cloning", in 34th European Conference on Object-Oriented Programming (ECOOP 2020), LIPIcs, Vol. 166, pp. 24:1–24:26, 2020. `https://doi.org/10.4230/LIPIcs.ECOOP.2020.24`

Related Conference 34th European Conference on Object-Oriented Programming (ECOOP 2020), November 15–17, 2020, Berlin, Germany (Virtual Conference)

1 Scope

With this artifact, users can:

- Reproduce the performance and precision experiments of TypeClone and Sparse.
- Build software with a `ctir`-modified Clang which can subsequently be analysed by TypeClone.
- Write and run small unit tests for TypeClone.

The claims which can be deduced through the artifact are:

- That the slowdown exhibited by TypeClone against Sparse is around 11× and 25× without and with reuse, respectively, on average (geometric mean).
- That on average (geometric mean), TypeClone can answer over 16% and over 15% more alias queries with a "no-alias" response than Sparse (when not considering, and when considering, reuse, respectively).

2 Content

The artifact, when `typeclone.zip` is unzipped, contains:

- `README.md`: detailed instructions for running and testing the artifact.
- `README.html`: `README.md` rendered as HTML.

Dagstuhl Artifacts Series, Vol. 6, Issue 2, Artifact No. 1, pp. 1:1–1:2
DAGSTUHL ARTIFACTS SERIES Dagstuhl Artifacts Series
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

- `typeclone.tar`: a Docker image with Debian as the base OS containing:
  - `svf/`: SVF source and binary implementing TypeClone and Sparse.
  - `benchmarks/`: benchmarks used in our evaluation with scripts to generate data.
  - `llvm9/`: LLVM/Clang 9 source and binaries modified to produce `ctir` annotations.
  - `ptaben/`: unit tests for TypeClone.
  - `coreutils/`: GNU Coreutils 8.31 source and LLVM bitcode with `ctir` annotations.

Running `cd benchmarks; ./benchmark.sh` inside the Docker image reproduces Table 2, Table 3, and Table 4 from the paper. Further details are contained in `README.md`/`README.html`.

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). In addition, information about the latest versions of TypeClone and the `ctir`-modified Clang (outside the artifact environment) is available at: `https://github.com/SVF-tools/SVF/wiki/TypeClone`.

4 Tested platforms

The artifact has been successfully run on a Void Linux machine with 8GB of memory and Docker 19.03.8.

5 License

The artifact is available under the GNU Lesser General Public License v3.0 or later.

6 MD5 sum of the artifact

f7dddff83dbea3469c6da06dfbfb8269

7 Size of the artifact

1.3 GiB (4.8 GiB uncompressed)