Scopes Describe Frames: A Uniform Model for Memory Layout in Dynamic Semantics (Artifact) *

Casper Bach Poulsen[1], Pierre Néron[2], Andrew Tolmach[3], and Eelco Visser[4]

1 **Delft University of Technology**
`c.b.poulsen@tudelft.nl`

2 **French Network and Information Security Agency (ANSSI)**
`pierre.neron@ssi.gouv.fr`

3 **Portland State University**
`tolmach@pdx.edu`

4 **Delft University of Technology**
`visser@acm.org`

Abstract

Our paper introduces a systematic approach to the alignment of names in the static structure of a program, and memory layout and access during its execution. We develop a uniform memory model consisting of frames that instantiate the scopes in the scope graph of a program. This provides a language-independent correspondence between static scopes and run-time memory layout, and between static resolution paths and run-time memory access paths. The approach scales to a range of binding features, supports straightforward type soundness proofs, and provides the basis for a language-independent specification of sound reachability-based garbage collection.

This Coq artifact showcases how our uniform model for memory layout in dynamic semantics provides structure to type soundness proofs. The artifact contains type soundness proofs mechanized in Coq for (supersets of) all languages in the paper. The type soundness proofs rely on a language-independent framework formalizing scope graphs and frame heaps.

1998 ACM Subject Classification F.3.1 Specifying and Verifying and Reasoning about Programs

Keywords and phrases Dynamic semantics, scope graphs, memory layout, type soundness, operational semantics

Digital Object Identifier 10.4230/LIPIcs...

1 Scope

The artifact is designed to document and support repeatability of the type soundness proofs in the companion paper [2], using the Coq proof assistant.[1] In particular, the artifact provides a language-independent framework formalizing scope graphs and frame heaps. The scopes-as-frames correspondence is formalized based on this framework, and gives rise to a suite of helper lemmas that are useful for proving the soundness of languages and their type systems, as well as garbage collection (GC) strategies.

* This artifact is a companion of the paper: Casper Bach Poulsen, Pierre Néron, Andrew Tolmach, and Eelco Visser, "Scopes Describe Frames: A Uniform Model for Memory Layout in Dynamic Semantics", Proceedings of the 30th European Conference on Object-Oriented Programming (ECOOP 2016), Rome, Italy, July 2016. This work was partially funded by the NWO VICI *Language Designer's Workbench* project (639.023.206). Andrew Tolmach was partly supported by a Digiteo Chair at Laboratoire de Recherche en Informatique, Université Paris-Sud.

[1] https://coq.inria.fr/

2 Content

The artifact package includes:

- Coq libraries formalizing scope graphs, frames, and the scopes-describe-frames correspondence;
- Coq formalizations and proofs of soundness for three example languages: `L1`; a superset of `L2` (differences summarized below); and a superset of the `L3` language with class-based inheritance and sub-typing that is briefly described in the companion paper, and covered in more detail in the companion technical report [3];
- instructions for using the artifact and for rebuilding it from scratch, provided as a `README` file; and
- pretty-printed versions of the Coq proof scripts.

Differences from paper. The languages described in our paper are simplified versions of the languages that we used to experiment with the scopes-describe-frames correspondence and its application to type and GC soundness. There are numerous small differences of naming and terminology. In additon, the languages in this artifact differ from the paper in the following ways:

- `L1` follows the semantics in the paper.
- L2 and L3 differ from the paper as follows:
  - Functions and function types are n-ary, and argument values are stored in call-frames immediately after they are computed (as opposed to in the big-step derivation tree).
  - The language provides three variants of n-ary let-binding: sequential lets, parallel lets, and recursive lets (following the static semantics given for these in [1, 4]). Recursive lets are restricted to bind values of function type only.
  - The language has boolean expressions and simple if-then-else branching.

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 our code is available on Github: https://github.com/metaborg/scopes-describe-frames.

4 Tested platforms

The artifact is known to type check using Coq 8.5.

5 License

Apache 2.0 (http://www.apache.org/licenses/LICENSE-2.0)

Acknowledgments. We thank the anonymous reviewers for their feedback on a previous version of the artifact. This research was partially funded by the NWO VICI *Language Designer's Workbench* project (639.023.206). Andrew Tolmach was partly supported by a Digiteo Chair at Laboratoire de Recherche en Informatique, Université Paris-Sud.

References

1 Pierre Néron, Andrew P. Tolmach, Eelco Visser, and Guido Wachsmuth. A theory of name resolution. In Jan Vitek, editor, *ESOP'15*, volume 9032 of *LNCS*, pages 205–231. Springer, 2015.

2 Casper Bach Poulsen, Pierre Néron, Andrew P. Tolmach, and Eelco Visser. Scopes describe frames: A uniform model for memory layout in dynamic semantics. In Shriram Krishnamurthi, editor, *30th European Conference on Object-Oriented Programming (ECOOP 2016)*, Leibniz International Proceedings in Informatics (LIPIcs), Dagstuhl, Germany, 2016. Schloss Dagstuhl–Leibniz-Zentrum für Informatik.

3 Casper Bach Poulsen, Pierre Néron, Andrew P. Tolmach, and Eelco Visser. Scopes describe frames: A uniform model for memory layout in dynamic semantics. Technical Report TUD-SERG-2016-010, Delft University of Technology, Programming Languages Research Group, Delft, The Netherlands, 2016.

4 Hendrik van Antwerpen, Pierre Néron, Andrew P. Tolmach, Eelco Visser, and Guido Wachsmuth. A constraint language for static semantic analysis based on scope graphs. In Martin Erwig and Tiark Rompf, editors, *PEPM'16*, pages 49–60. ACM, 2016.