Proactive Synthesis of Recursive Tree-to-String Functions from Examples (Artifact)*

Mikaël Mayer†1, Jad Hamza‡2, and Viktor Kuncak3

1 **EPFL IC IINFCOM LARA, INR 318, Station 14, CH-1015 Lausanne**
`mikael.mayer@epfl.ch`
2 **EPFL IC IINFCOM LARA, INR 318, Station 14, CH-1015 Lausanne**
`jad.hamza@epfl.ch`
3 **EPFL IC IINFCOM LARA, INR 318, Station 14, CH-1015 Lausanne**
`viktor.kuncak@epfl.ch`

Abstract

This artifact, named `Prosy`, is an interactive command-line tool for synthesizing recursive tree-to-string functions (e.g. pretty-printers) from examples. Specifically, Prosy takes as input a Scala file containing a hierarchy of abstract and case classes, and synthesizes the printing function after interacting with the user. Prosy first pro-actively generates a finite set of trees such that their string representations uniquely determine the function to synthesize. While asking the output for each example, Prosy prunes away questions when it can infer their answers from previous answers. In the companion paper, we prove that this pruning allows Prosy not to require that the user provides answers to the entire set of questions, which is of size $O(n^3)$ where n is the size of the input file, but only to a reasonably small subset of size $O(n)$. Furthermore, Prosy guides the interaction by providing suggestions whenever it can.

1998 ACM Subject Classification F.3.1 Specifying and Verifying and Reasoning about Programs – D.3.4 Processors

Keywords and phrases programming by example, active learning, program synthesis

Digital Object Identifier 10.4230/DARTS.3.2.16

Related Article Mikaël Mayer, Jad Hamza and Viktor Kuncak, "Proactive Synthesis of Recursive Tree-to-String Functions from Examples", in Proceedings of the 31st European Conference on Object-Oriented Programming (ECOOP 2017), LIPIcs, Vol. 74, pp. 19:1–19:30, 2017.
`http://dx.doi.org/10.4230/LIPIcs.ECOOP.2017.19`

Related Conference European Conference on Object-Oriented Programming (ECOOP 2017), June 18-23, 2017, Barcelona, Spain

1 Scope

We designed this artifact to support repeatability of all the experiments of the companion paper, allowing users to test the interaction on the benchmarks of the evaluation section in the companion paper. Users can verify the number of questions and their variety (suggestion, multiple-choice, raw) for each benchmark. In particular, users can verify that the number of questions Prosy asked ($O(n)$) is much smaller than the initial set of questions ($O(n^3)$), and that both the number of questions and their inputs vary depending on the answers. Furthermore, users can also test the interaction on more benchmarks and create their own benchmarks.

* This work was supported in part by European Research Council (ERC) Project Implicit Programming and an EPFL-Inria Post-Doctoral grant.
† Front-end and interaction.
‡ Algorithm implementation.

2 Content

The artifact package is a zip file including:

- Instructions on how to rebuild the artifact from scratch (README.md).
- Detailed instructions for using the artifact (WALKTHROUGH.md)
- The sources which can be recompiled from scratch using SBT.
- A JAR file which can be used directly as described in WALKTHROUGH.md

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/epfl-lara/prosy`.

4 Tested platforms

The artifact is known to work on any platform running Java 8, thus including Windows, Linux and Mac OS.

5 License

EPL-1.0 (`http://www.eclipse.org/legal/epl-v10.html`)

6 MD5 sum of the artifact

930966950860a59c2783101e1fcfc59e

7 Size of the artifact

5.9 MB