Perfect is the Enemy of Good: Best-Effort Program Synthesis (Artifact)

Hila Peleg
University of California San Diego, CA, USA
hpeleg@eng.ucsd.edu

Nadia Polikarpova
University of California San Diego, CA, USA
npolikarpova@eng.ucsd.edu

Abstract

Program synthesis promises to help software developers with everyday tasks by generating code snippets automatically from input-output examples and other high-level specifications. The conventional wisdom is that a synthesizer must always satisfy the specification exactly. We conjecture that this all-or-nothing paradigm stands in the way of adopting program synthesis as a developer tool: in practice, the user-written specification often contains errors or is simply too hard for the synthesizer to solve within a reasonable time; in these cases, the user is left with a single over-fitted result or, more often than not, no result at all. In this paper we propose a new program synthesis paradigm we call *best-effort program synthesis*, where the synthesizer returns a ranked list of partially-valid results, i.e., programs that satisfy some part of the specification.

To support this paradigm, we develop *best-effort enumeration*, a new synthesis algorithm that extends a popular program enumeration technique with the ability to accumulate and return multiple partially-valid results with minimal overhead. We implement this algorithm in a tool called BESTER, and evaluate it on 79 synthesis benchmarks from the literature. Contrary to the conventional wisdom, our evaluation shows that BESTER returns useful results even when the specification is flawed or too hard: *i)* for all benchmarks with an error in the specification, the top three BESTER results contain the correct solution, and *ii)* for most hard benchmarks, the top three results contain non-trivial *fragments* of the correct solution. We also performed an exploratory user study, which confirms our intuition that partially-valid results are useful: the study shows that programmers use the output of the synthesizer for comprehension and often incorporate it into their solutions.

2012 ACM Subject Classification Theory of computation → Program specifications; Software and its engineering → Automatic programming

Keywords and phrases Program Synthesis, Programming by Example

Digital Object Identifier 10.4230/DARTS.6.2.16

Related Article Hila Peleg and Nadia Polikarpova, “Perfect Is the Enemy of Good: Best-Effort Program Synthesis”, in 34th European Conference on Object-Oriented Programming (ECOOP 2020), LIPIcs, Vol. 166, pp. 2:1–2:30, 2020. `https://doi.org/10.4230/LIPIcs.ECOOP.2020.2`

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

1 Scope

The artifact is a VirtualBox VM that contains the BESTER synthesizer, the BESTER benchmark suite and the version of CVC4 against which they were compared, and the interactive CLI used in our user study. The artifact can be used to run all empirical benchmarks in the paper. The raw data on which our graphs are based is generated by the provided scripts.

2 Content

The artifact package includes:

- A Virtualbox VM containing:
  - A clone of the Bester repository, with the code already compiled and packaged into a single jar
  - Our benchmark suite, including the modified benchmarks with inserted errors
  - The scripts for running all experiments in the empirical evaluation in the paper, including the scripts that generate the subset data for the CVC4-subsets baseline
  - The CLI for our user study
- The PDF of the artifact guide
- The PDF of the paper *Perfect is the Enemy of Good: Best-Effort Program Synthesis*
- The specification of the SyGuS language used in our benchmarks

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, the artifact is also available at: `https://drive.google.com/file/d/1Zqy94Q7hjBF2Tu_jlri2Cuj6-jyWKRpC/view`.

Additionally, we recommend getting the code directly from `https://github.com/peleghila/bester` and building it with SBT.

4 Tested platforms

Tested on VirtualBox 6.1 and later. Requires at least 8GB of memory on the host machine.

5 License

The artifact is available under CRAPL license[1].

6 MD5 sum of the artifact

036d1165a99c9770b1f51ece8a8c39c3

7 Size of the artifact

8.55 GiB

[1] `http://matt.might.net/articles/crapl/`