Automatic Test Generation for Haskell Programming Assignments

Vladimír Štill
Faculty of Informatics, Masaryk University
Czech Republic
xstill@mail.muni.cz

ABSTRACT

Automatic testing of programming assignments is highly desirable as it can provide fast feedback for the students and allows the teachers to teach efficiently even in courses with many students. However, writing tests for students' solutions can be tedious. In this work, we present a novel approach to test generation for small Haskell assignments. Such assignments usually consist of one function (with the possibility to use helper functions in its definition) that the students are supposed to program according to a teacher's specification. The teacher is not required to write tests for this function. Instead, we make use of an example solution, which the teacher should have to assess the difficulty of the assignment. Using the example solution, and (if needed) a specification of input values for the function, our tool can automatically generate randomized tests. If these tests fail, the student is presented with a counterexample which shows the input values, the expected output of the tested function and the output computed by their solution.

CCS CONCEPTS

• **Social and professional topics → Computing education**.

KEYWORDS

programming education, automatic testing, Haskell, QuickCheck

ACM Reference Format:
Vladimír Štill. 2020. Automatic Test Generation for Haskell Programming Assignments. In *Proceedings of the 2020 ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE '20), June 15–19, 2020, Trondheim, Norway*. ACM, New York, NY, USA, 1 page. https://doi.org/10.1145/3341525.3393972

1 TESTING CAPABILITIES

Our tool *hsExprTest* is based on QuickCheck [2], a Haskell tool for specification-based testing of Haskell programs. It uses QuickCheck for automatic generation of test inputs, and to shrink test inputs, which allows us to produce small counterexamples. It also uses QuickCheck's ability to generate printable and shrinkable functions [1], which is used when testing implementations of higher-order functions (i.e., functions which take functions as arguments, for example, a `map` function, which takes a function and applies it to each element of a list, producing a list of results).

ITiCSE '20, June 15–19, 2020, Trondheim, Norway

However, unlike QuickCheck, *hsExprTest* uses the teacher's solution as a specification and automatically derives a test expression which compares the teacher's and student's solutions. The expression generation is driven by the type of teacher's solution (and optionally also input specification). It can include specialization of polymorphic functions and wrapping of inputs for higher-order functions in such a way they can be displayed and shrunk by QuickCheck.

For the simplest exercise functions (i.e. total functions operating on basic Haskell data types like numbers, lists, tuples, and functions over these types), our tool can generate tests just from the solution. For more complex input data, it is necessary to either constrain inputs (for example to positive numbers, or to a specific range) or to provide means of generation of random inputs. Input data generators for custom data types can be written using the QuickCheck library. Finally, to accommodate more complex exercises within the same framework, it is also possible to write custom QuickCheck tests.

2 USAGE AND AVAILABILITY

Our tool is open source and available on GitHub, at https://github.com/vlstill/hsExprTest. The documentation for testing Haskell can be found at https://github.com/vlstill/hsExprTest/tree/master/doc/hs.md. The tool is used in the introductory course to functional programming at the Faculty of Informatics of Masaryk University since 2014 and is continually being improved. While the tool is integrated with the Information System of Masaryk University, which is used to submit assignments and present results to students, it is also possible to use it with different submission frontends.

3 GOALS OF THE POSTER

The poster aims to present our novel testing method which simplifies testing of small Haskell programming exercises. We believe that by making testing easier, we can promote the creation of more and better exercises which students can use to improve their new skills. The poster could also spark a broader conversation about methods used in programming courses.

REFERENCES

[1] Koen Claessen. 2012. Shrinking and Showing Functions: (Functional Pearl). In *Proceedings of the 2012 Haskell Symposium* (Copenhagen, Denmark) *(Haskell '12)*. Association for Computing Machinery, New York, NY, USA, 73–80. https://doi.org/10.1145/2364506.2364516

[2] Koen Claessen and John Hughes. 2000. QuickCheck: A Lightweight Tool for Random Testing of Haskell Programs. *Proceedings of the ACM SIGPLAN International Conference on Functional Programming, ICFP* 46 (01 2000). https://doi.org/10.1145/1988042.1988046