Proposal for Master Thesis: Automated Proving of Haskell Program Properties

Dan Rosén

November 1, 2011

Proposal

This project is about proving properties of Haskell functions and programs by translating Haskell to First Order Logic (FOL) and using Automated Theorem Provers (ATPs). To do this, a tool needs to be produced that can do this translation, try different proving techniques and integrate this with an automated theorem prover. These parts are described in detail below:

Translate Haskell to FOL

Haskell is a big language and the aim is to produce a translator for a sufficiently big subset of the Haskell98 standard to prove interesting properties. Special care needs to be taken to care for the undefined bottom value that exists in Haskell but not directly in First Order Logic.

Investigate different proving methods for programs

There are several developed methods for proving properties for functional languages, including normal induction, fix-point induction, coinduction techniques like bisimilation, the generic approximation lemma, reasoning with ana- and catamorphisms and Hoare induction. The developed tool should be able to try to prove a given property with different proving techniques, on different parts of the program, i.e. try induction on different combinations of variables.

Integrate this with ATPs

The intended output of the translation and the proof methods is the format TPTP (Thousand Problems for Theorem Provers). This is a standardised format most theorem provers can read. Extra focus should be taken on one selected theorem prover and its output should be made readible for the end user, if the property is true, and could possibly be used to produce a counterexample if the property fails. Eprover is an example of suitable theorem prover.

Contributions

The contributions of this project is work towards an automated way to prove prove properties of Haskell programs. Futhermore, the Haskell to FOL translation that could be used in other projects and research.