Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Dr. Peter Lammich

WS 2015/16
19. 1. 2016

Semantics of Programming Languages

Exercise Sheet 13

Exercise 13.1 AI for Conditionals

Our current constant analysis does not regard conditionals. For example, it cannot figure out, that after executing the program

x:=*2*; *IF x<2 THEN x*:=*2 ELSE x*:=*1*

x will be constant.

In this exercise we extend our abstract interpreter with a simple analysis of boolean expressions. To this end, modify locale *Val_semilattice* in theory *Abs_Int0.thy* as follows:

- Introduce an abstract domain *'bv* for boolean values, add, analogously to *num'* and *plus'* also functions for the boolean operations and for *less*.
- Modify *Abs_Int0* to accomodate for your changes. Do not modify the locales ending in *_fun*, they are not needed for executable analysis, and can simply be commented out.
- Define a function *bval'* in *Abs_Int1*, and modify the *step'* function to take into account boolean values guaranteed to be false.
- Finally, adapt all theories necessary to get a more precise constant analysis.

Hint: Start with a fresh copy of the IMP/ folder.

Homework 13.1 Prefix Analysis

Submission until Tuesday, 26 January 2016, 10:00am. In this homework, you shall modify IMP to work on strings rather than integers, and define an abstract interpretation based analysis to return all possible one-element prefixes of the values of a variable.

The string operations shall be

Constant A string constant.

Concatenation Concatenate two strings.

Last Return the last character of the operand as a string of length one. Empty string if operand is empty.

Butlast Return the operand with the last character removed. Empty string if operand is empty.

Equal Compare two strings for equality.

otherwise keep the arithmetic and Boolean operations, i.e. Variables, And, Not.

The provided template file contains all necessary theory up to abstract interpretation in one file, with irrelevant parts stripped away. Use this template file as a basis for your modifications.

Finally, implement an abstract interpretation with the abstract domain *char option set*, which describes the set of possible one-element prefixes of a string, where *None* represents the empty string. Try to define the abstract operations as precise as possible! As starting point for your development, you can use the parity analysis, which is also included in the template file. You are required to interpret the locales *Val_Semilattice*, *Abs_int*, *Abs_Int_mono*, and *Abs_Int_measure*.