Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Simon Wimmer

WS 2016/17
20. 12. 2016

Semantics of Programming Languages

Exercise Sheet 9

Exercise 9.1 Security type system: bottom-up with subsumption

Use the template file `ex09_tmpl.thy`.

Recall security type systems for information flow control from the lecture. Such a type systems can either be defined in a top-down or in a bottom-up manner. Independently of this choice, the type system may or may not contain a subsumption rule (also called anti-monotonicity in the lecture). The lecture discussed already all but one combination: a bottom-up type system with subsumption.

1. Define a bottom-up security type system for information flow control with subsumption rule.
2. Prove the equivalence of the newly introduced bottom-up type system with the bottom-up type system without subsumption rule from the lecture.

Exercise 9.2 Definite Initialization Analysis

Use the template file `ex09_tmpl.thy`.

In the lecture, you have seen a definite initialization analysis that was based on the big-step semantics. Definite initialization analysis can also be based on a small-step semantics. Furthermore, the ternary predicate D from the lecture can be split into two parts: a function $AA :: com \Rightarrow name\ set$ ("assigned after") which collects the names of all variables assigned by a command and a binary predicate $D :: name\ set \Rightarrow com \Rightarrow bool$ which checks that a command accesses only previously assigned variables. Conceptually, the ternary predicate from the lecture (call it D_{lec}) and the two-step approach should relate by the equivalence $D\ V\ c \longleftrightarrow D_{lec}\ V\ c\ (V \cup AA\ c)$

1. Study the already defined small-step semantics for definite analysis.
2. Define the function AA which computes the set of variables assigned after execution of a command. Furthermore, define the predicate D which checks if a command accesses only assigned variables, assuming the variables in the argument set are already assigned.
3. Prove progress and preservation of D with respect to the small-step semantics, and conclude soundness of D. You may use (and then need to prove) the lemmas *D_incr* and *D_mono*.

Homework 9 Be Original!

Submission until Tuesday, 10 January 2016, 10:00am. (20 regular points, plus bonus points for nice submissions)

Think up a nice formalization yourself, for example

- Prove some interesting result about graph/automata/formal language theory
- Formalize some results from mathematics
- Find interesting modifications of IMP material and prove interesting properties about them
- ...

You should set yourself a time limit before starting your project. Also incomplete/unfinished formalizations are welcome and will be graded!

Please comment your formalization well, such that we can see what it does/is intended to do.

You are welcome to discuss your plans with the tutor before starting your project.

Merry Christmas!