Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Sascha Böhme, Alexander Krauss

WS 2010/11
20. 12. 2010

Semantics of Programming Languages

Exercise Sheet 9

Exercise 9.1 Definite Assignment Analysis

Definite assignment 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 $D\ V\ c \longleftrightarrow D_{lec}\ V\ c\ (AA\ c)$

(a) Download the theory `Ex09_Template` and study the already defined small-step semantics for definite analysis.

(b) 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.

(c) 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*.