Exercise 8.1  Definite Assignment Analysis

In the lecture, you have seen a definite assignment analysis that was based on the large-step semantics. 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 $AA :: \text{com} \Rightarrow \text{name set}$ (“assigned after”) which collects the names of all variables assigned by a command and a binary predicate $D :: \text{name set} \Rightarrow \text{com} \Rightarrow \text{bool}$ which checks that a command accesses only previously assigned variables. Conceptually, the ternary predicate from the lecture (call it $D_{\text{lec}}$) and the two-step approach should relate by the equivalence

$$ D_{\text{lec}} V c \iff D V c \land (V \cup AA c) $$

1. Download the theory ex08_template and 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 $D_{\text{incr}}$ and $D_{\text{mono}}$.

Homework 8  Read Variables

Submission until Wednesday, December 21, 2011, 12:00 (noon).

Instantiates the $\text{vars}$ typeclass for commands, such that $\text{vars } c$ is the set of variables read by the command.

Then show, that an execution does not depend on variables not read by the command, w.r.t. the small-step semantics. I.e., show the following lemma:

\[
\text{lemma } \forall (c,s) \rightarrow ((c',s'); s = t \text{ on } X; \ \text{vars } c \subseteq X) \\
\implies \exists t'. (c,t) \rightarrow ((c',t'); s' = t' \text{ on } X)
\]
Hint: You may want to show the lemma for a single small-step first, i.e.,

**Lemma eq_step**: "\[ \[(c,s) \rightarrow (c',s') \land s = t \text{ on } X; \ \text{vars } c \subseteq X\]\[
\implies \exists t'. \ (c,t) \rightarrow (c',t') \land s' = t' \text{ on } X\]"