Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Kevin Kappelmann, Lukas Stevens

WS 24/25
5.12.2024

Semantics of Programming Languages

Exercise Sheet 8

Note: The tutorial is cancelled this week due to the Dies Academicus.

Exercise 8.1 Knaster-Tarski Fixed Point Theorem

The Knaster-Tarski theorem tells us that for each set P of fixed points of a monotone function f we have a fixpoint of f which is a greatest lower bound of P. In this exercise, we want to prove the Knaster-Tarski theorem.

First we give a construction of the greatest lower bound of all fixed points P of the function f. This is the union of all sets u smaller than P and $f\ u$. Then the task is to show that this is a fixed point, and that it is the greatest lower bound of all sets in P.

Let us define *Inf_fixp*:

definition *Inf_fixp* :: "$('a\ set \Rightarrow 'a\ set) \Rightarrow 'a\ set\ set \Rightarrow 'a\ set$" **where**
"$Inf\_fixp\ f\ P = \bigcup\{u.\ u \subseteq \bigcap P \cap f\ u\ \}$"

To work directly with this definition is a little cumbersome, we propose to use the following two theorems:

lemma *Inf_fixp_upperbound*: "$X \subseteq \bigcap P \Longrightarrow X \subseteq f\ X \Longrightarrow X \subseteq Inf\_fixp\ f\ P$"
by (*auto simp*: *Inf_fixp_def*)

lemma *Inf_fixp_least*: "$(\bigwedge u.\ u \subseteq \bigcap P \Longrightarrow u \subseteq f\ u \Longrightarrow u \subseteq X) \Longrightarrow Inf\_fixp\ f\ P \subseteq X$"
by (*auto simp*: *Inf_fixp_def*)

Now prove, that *Inf_fixp* is acually a fixed point of *f*.

Hint: First prove $Inf\_fixp\ f\ P \subseteq f\ (Inf\_fixp\ f\ P)$, this will be used for the other direction. It may be helpful to first think about the structure of your proof using pen-and-paper and then translate it into Isar.

lemma *Inf_fixp*:
assumes *mono*: "*mono f*"
and *P*: "$\bigwedge p.\ p \in P \Longrightarrow f\ p = p$"
shows "$Inf\_fixp\ f\ P = f\ (Inf\_fixp\ f\ P)$"

Now we prove that it is a lower bound:

lemma *Inf_fixp_lower*: "$Inf\_fixp\ f\ P \subseteq \bigcap P$"

And that it is the greatest lower bound:

lemma *Inf_fixp_greatest*:
  assumes "$f\ q = q$"
    and "$q \subseteq \bigcap P$"
  shows "$q \subseteq$ *Inf_fixp* $f\ P$"

Homework 8.1 Idempotence of Dead Varibale Elimination

Submission until Wednesday, Dec 11, 23:59pm.

Dead variable elimination (*bury*) is not idempotent: multiple passes may reduce a command further and further. Give an example where *bury* (*bury* c X) $X \neq$ *bury* c X. Hint: a sequence of two assignments.

We define a textually identical function *bury* in the context of true liveness analysis (theory *HOL−IMP.Live_True*).

fun *bury* :: "*com* ⇒ *vname set* ⇒ *com*" **where**
"*bury SKIP X = SKIP*" |
"*bury* (x ::= a) X = (*if* $x \in X$ *then* x ::= a *else SKIP*)" |
"*bury* (c_1;; c_2) X = (*bury* c_1 (L c_2 X);; *bury* c_2 X)" |
"*bury* (*IF* b *THEN* c_1 *ELSE* c_2) X = *IF* b *THEN bury* c_1 X *ELSE bury* c_2 X" |
"*bury* (*WHILE* b *DO* c) X = *WHILE* b *DO bury* c (L (*WHILE* b *DO* c) X)"

The aim of this homework is to prove that this version of *Hw1.bury* is idempotent. This will involve reasoning about *lfp*. In particular we will need that *lfp* is the least pre-fixpoint. This is expressed by two lemmas from the library:

lfp_unfold: *mono* $f \Longrightarrow$ *lfp* $f = f$ (*lfp* f)
lfp_lowerbound: $f\ A \leq A \Longrightarrow$ *lfp* $f \leq A$

Prove the following lemma for showing that two fixpoints are the same, where *mono_def*: *mono* $f = (\forall x\ y.\ x \leq y \longrightarrow f\ x \leq f\ y)$.

theorem *lfp_eq*: "⟦ *mono* f; *mono* g; *lfp* $f \subseteq U$; *lfp* $g \subseteq U$;
  $\bigwedge X.\ X \subseteq U \Longrightarrow f\ X = g\ X$ ⟧ $\Longrightarrow$ *lfp* f = *lfp* g"

It says that if we have an upper bound U for the *lfp* of both f and g, and f and g behave the same below U, then they have the same *lfp*.

The following two tweaks improve proof automation:

lemmas [*simp*] = *L.simps*(5)
lemmas *L_mono2* = *L_mono*[*unfolded mono_def*]

To show that *Hw1.bury* is idempotent we need a lemma:

theorem *L_bury*[*simp*]: "$X \subseteq Y \Longrightarrow L$ (*bury* c Y) $X = L\ c\ X$"
proof(*induction c arbitrary*: X Y)

The proof is straightforward except for the case *WHILE b DO c*. The definition of L in this case means that we have to show an equality of two *lfp*s. Lemma *lfp_eq* comes to the rescue. We recommend the upper bound *lfp* $(\lambda Z.\ vars\ b \cup Y \cup L\ c\ Z)$. One of the two upper bound assumptions of lemma *lfp_eq* can be proved by showing that U is a pre-fixpoint of f or g (see lemma *lfp_lowerbound*).

Now we can prove idempotence of *Hw1.bury*, again by induction on c, but this time even the *While* case should be easy.

theorem *bury_bury*: "$X \subseteq Y \Longrightarrow bury\ (bury\ c\ Y)\ X = bury\ c\ X$"

Idempotence is a corollary:

corollary "$bury\ (bury\ c\ X)\ X = bury\ c\ X$"

Homework 8.2 True Liveness refines Liveness

Submission until Wednesday, Dec 11, 23:59pm.

In the lecture, we introduced two liveness analyses, namely Liveness *Live.L* and True Liveness *Live_True.L*. Prove that True Liveness refines the Liveness analysis, i.e. show the former is a subset of the latter.

theorem *True_L_subs_L*: "$Live\_True.L\ c\ X \subseteq Live.L\ c\ X$"