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

WS 2016/17
24. 01. 2017

Semantics of Programming Languages

Exercise Sheet 12

Exercise 12.1 Using the VCG

Use the VCG to prove correct a multiplication and a square root program:

```
definition MUL :: com
lemma "⊢
  {λs. 0 ≤ s ''y'' ∧ s=sorig}
  MUL
  {λs. s ''z'' = s ''x'' ∗ s ''y'' ∧ (∀ v. v∉{''z'',''c''} ⟶ s v = sorig v)}"

definition "SQRT ≡
  ''r'' ::= N 0;;
  ''s'' ::= N 1;;
  WHILE (Not (Less (V ''x'') (V ''s''))) DO (
    ''r'' ::= Plus (V ''r'') (N 1);;
    ''s'' ::= Plus (V ''s'') (V ''r'');;
    ''s'' ::= Plus (V ''s'') (V ''r'');;
    ''s'' ::= Plus (V ''s'') (N 1)
  )"

lemma "⊢
  {λs. s=sorig ∧ s ''x'' ≥ 0}
  SQRT
  {λs. (s ''r'')^2 ≤ s ''x'' ∧ s ''x'' < (s ''r''+1)^2 ∧ (∀ v. v∉{''s'',''r''} ⟶ s v = sorig v)}"
```

Exercise 12.2 Total Correctness

Prove total correctness of the multiplication and the square root program.

Prove the following syntax-directed conditional rule (for total correctness):

```
lemma IfT:
  assumes "⊢t {P1} c1 {Q}" and "⊢t {P2} c2 {Q}"
  shows "⊢t {λs. (bval b s ⟶ P1 s) ∧ (¬ bval b s ⟶ P2 s)} IF b THEN c1 ELSE c2 {Q}"

lemmas Seq_bwd = Hoare_Total.Seq[rotated]
lemmas hoareT_rule[intro?] = Seq_bwd Hoare_Total.Assign Hoare_Total.Assign' IfT
```

lemma "$\vdash_t$
  $\{\lambda s.\ 0 \le s\ ''y'' \wedge s{=}sorig\}$
  MUL
  $\{\lambda s.\ s\ ''z'' = s\ ''x'' * s\ ''y'' \wedge (\forall v.\ v{\notin}\{''z'',''c''\} \longrightarrow s\ v = sorig\ v)\}$"
lemma "$\vdash_t$
  $\{\lambda s.\ s{=}sorig \wedge s\ ''x'' \ge 0\}$
  SQRT
  $\{\lambda s.\ (s\ ''r'')\hat{\ }2 \le s\ ''x'' \wedge s\ ''x'' < (s\ ''r''{+}1)\hat{\ }2 \wedge (\forall v.\ v{\notin}\{''s'',''r''\} \longrightarrow s\ v = sorig\ v)\}$"

Homework 12.1 Program Verification

Submission until Tuesday, 31 January 2017, 10:00am.

Define an annotated command *Cdiff* that subtracts x from y and prove:

lemma "$\vdash \{\lambda s.\ s\ ''x'' = x \wedge s\ ''y'' = y \wedge 0 \le x\}\ strip\ (Cdiff\ x\ y)\ \{\lambda t.\ t\ ''y'' = y - x\}$"

Homework 12.2 Collecting Semantics

Submission until Tuesday, 31 January 2017, 10:00am.

Note: This is a typical exam exercise.

Show the iterative computation of the collecting semantics of the following program in a table like the one on page 228 of the book.

```
x := 2; y := 1 {A_0} ;
{A_1}
WHILE 0 < x
DO {A_2} ( y := y * x; x := x - 1 {A_3} )
{A_4}
```

Note that two annotations have been suppressed to make the task less tedious. You do not need to show steps where only the suppressed annotations change.

Because the program contains two variables, the state sets in the table should be represented as sets of pairs (x, y). In order to keep the table compact, you can also just write xy, e.g. 02 instead of $(0, 2)$ — the values of the variables do not exceed single digits.

	0									
A_0	{}									
A_1	{}									
A_2	{}									
A_3	{}									
A_4	{}									

Homework 12.3 A Hoare Calculus with Execution Times

Submission until Tuesday, 31 January 2017, 10:00am.

In this homework, we will consider a hoare calculus with execution times. We first give a modified big-step semantics to account for execution times. A judgement of the form $(c,\ s) \Rightarrow n \Downarrow t$ has the intended meaning that we can get from state s to state t by an terminating execution of program c that takes exactly n time steps.

inductive
  big_step_t :: "$com \times state \Rightarrow nat \Rightarrow state \Rightarrow bool$" ("$\_ \Rightarrow \_ \Downarrow \_$" 55)
where
Skip: "$(SKIP,s) \Rightarrow Suc\ 0 \Downarrow s$" |
Assign: "$(x ::= a,s) \Rightarrow Suc\ 0 \Downarrow s(x := aval\ a\ s)$" |
Seq: "$[\![\ (c1,s1) \Rightarrow x \Downarrow s2;\ \ (c2,s2) \Rightarrow y \Downarrow s3\ ;\ z{=}x{+}y\ ]\!] \Longrightarrow (c1;;c2,\ s1) \Rightarrow z \Downarrow s3$" |
IfTrue: "$[\![\ bval\ b\ s;\ \ (c1,s) \Rightarrow x \Downarrow t;\ y{=}x{+}1\ ]\!] \Longrightarrow (IF\ b\ THEN\ c1\ ELSE\ c2,\ s) \Rightarrow y \Downarrow t$" |
IfFalse: "$[\![\ \neg bval\ b\ s;\ \ (c2,s) \Rightarrow x \Downarrow t;\ y{=}x{+}1\ ]\!] \Longrightarrow (IF\ b\ THEN\ c1\ ELSE\ c2,\ s) \Rightarrow y \Downarrow t$" |
WhileFalse: "$[\![\ \neg bval\ b\ s\ ]\!] \Longrightarrow (WHILE\ b\ DO\ c,s) \Rightarrow Suc\ 0 \Downarrow s$" |
WhileTrue:
"$[\![\ bval\ b\ s1;\ \ (c,s1) \Rightarrow x \Downarrow s2;\ \ (WHILE\ b\ DO\ c,\ s2) \Rightarrow y \Downarrow s3;\ 1{+}x{+}y{=}z\ \ ]\!]$
$\Longrightarrow (WHILE\ b\ DO\ c,\ s1) \Rightarrow z \Downarrow s3$"

Next, we define a Hoare calculus that also accounts for execution times.

type_synonym $assn$ = "$state \Rightarrow bool$"
type_synonym tbd = "$state \Rightarrow nat$"

abbreviation $state\_subst$ :: "$state \Rightarrow aexp \Rightarrow vname \Rightarrow state$"
  where "$s[a/x] \equiv s(x := aval\ a\ s)$"

definition $hoare\_Tvalid$ :: "$assn \Rightarrow com \Rightarrow tbd \Rightarrow assn \Rightarrow bool$"
  "$\models_T \{P\}\ c\ \{q \Downarrow Q\}\ \ \longleftrightarrow\ \ (\forall s.\ P\ s \longrightarrow (\exists t\ p.\ ((c,s) \Rightarrow p \Downarrow t) \wedge p \le (q\ s) \wedge Q\ t))$"

inductive
  $hoareT$ :: "$assn \Rightarrow com \Rightarrow tbd \Rightarrow assn \Rightarrow bool$" **where**
Skip: "$\vdash_T \{P\}\ SKIP\ \{\lambda s.\ Suc\ 0 \Downarrow P\}$" |
Assign: "$\vdash_T \{\lambda s.\ P(s[a/x])\}\ x{::=}a\ \{\lambda s.\ Suc\ 0 \Downarrow P\}$" |
Seq: "$[\![\bigwedge u.\ \vdash_T \{\lambda\ s.\ P_1\ s \wedge e2'\ s = u\}\ c_1\ \{e1 \Downarrow \lambda\ s.\ P_2\ s \wedge e2\ s \le u\};$
  $\vdash_T \{P_2\}\ c_2\ \{\ e2 \Downarrow P_3\};$
  $\bigwedge s.\ P_1\ s \Longrightarrow e1\ s + e2'\ s \le e\ s]\!]$
  $\Longrightarrow\ \vdash_T \{P_1\}\ c_1;;c_2\ \{\ e \Downarrow P_3\}$" |
If: "$[\![\ \vdash_T \{\lambda s.\ P\ s \wedge bval\ b\ s\}\ c_1\ \{e1 \Downarrow Q\};\ \vdash_T \{\lambda s.\ P\ s \wedge \neg\ bval\ b\ s\}\ c_2\ \{e1 \Downarrow Q\}\ ]\!]$
  $\Longrightarrow\ \vdash_T \{P\}\ IF\ b\ THEN\ c_1\ ELSE\ c_2\ \{\lambda s.\ e1\ s + Suc\ 0 \Downarrow Q\}$" |
While:
  "$[\![(\bigwedge u\ z{::}nat.$
  $\vdash_T \{\lambda s.\ INV\ (z{+}1)\ s \wedge e'\ s = u\}\ c\ \{e'' \Downarrow \lambda s.\ INV\ z\ s \wedge e\ s \le u\});$
  $(\bigwedge s\ z.\ INV\ (z{+}1)\ s \Longrightarrow bval\ b\ s \wedge e\ s \ge 1 + e'\ s + e''\ s);$
  $(\bigwedge s.\ INV\ 0\ s \Longrightarrow \neg\ bval\ b\ s \wedge 1 \le e\ s)$
  $]\!]$

$\Longrightarrow \vdash_T \{\lambda s.\ (\exists z.\ \mathit{INV}\ z\ s)\}\ \mathit{WHILE}\ b\ \mathit{DO}\ c\ \{e \Downarrow \mathit{INV}\ 0\}$” |
conseq: “$[\![\forall s.\ P'\ s \longrightarrow (P\ s \wedge e'\ s \leq e\ s);\ \vdash_T \{P\}\ c\ \{e' \Downarrow Q\};\ \forall s.\ Q\ s \longrightarrow Q'\ s]\!] \Longrightarrow$
$\vdash_T \{P'\}\ c\ \{e \Downarrow Q'\}$”

Your task is to prove soundness of the calculus:

theorem *hoareT_sound*: “$\vdash_T \{P\}\ c\ \{e \Downarrow Q\} \Longrightarrow \models_T \{P\}\ c\ \{e \Downarrow Q\}$”

Download the file *Big_StepT.thy* from the website and use the provided template. You can follow the outline of the soundness proof for the Hoare calculus that you know from the lecture. As for previous exercise sheets, you can split your homework submission into two files.