Technische Universität München
Fakultät für Informatik
Prof. Tobias Nipkow, Ph.D.
Fabian Huch

WS 2020/21
18.01.2021

Semantics of Programming Languages

Exercise Sheet 10

Exercise 10.1 Using the VCG

Use the VCG to prove correct a multiplication and a square root program:
*Note: With the template, use *unbundle COM/unbundle ACOM* before writing down a *com/acom* definition, otherwise parsing will be slow.

```
definition MUL :: com where
"MUL =
  ''z''::=N 0;;
  ''c''::=N 0;;
  WHILE (Less (V ''c'') (V ''y'')) DO (
    ''z''::=Plus (V ''z'') (V ''x'');;
    ''c''::=Plus (V ''c'') (N 1))"

theorem MUL_partially_correct:
"⊢ {λs. 0 ≤ s ''y'' ∧ s=sorig}
     MUL
   {λs. s ''z'' = s ''x'' * s ''y'' ∧ (∀ v. v∉{''z'',''c''} ⟶ s v = sorig v)}"

definition SQRT :: com where
"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)
  )"

theorem SQRT_partially_correct:
"⊢ {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 10.2 Total Correctness

Prove total correctness of the multiplication and square root program.

Rotated rule for sequential composition:

lemmas *Seq_bwd = Hoare_Total.Seq[rotated]*

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

lemma *IfT*:
assumes "$\vdash_t \{P1\}\ c_1\ \{Q\}$" **and** "$\vdash_t \{P2\}\ c_2\ \{Q\}$"
shows "$\vdash_t \{\lambda s.\ (bval\ b\ s \longrightarrow P1\ s) \wedge (\neg\ bval\ b\ s \longrightarrow P2\ s)\}\ IF\ b\ THEN\ c_1\ ELSE\ c_2\ \{Q\}$"

lemmas *hoareT_rule[intro?] = Seq_bwd Hoare_Total.Assign Hoare_Total.Assign' IfT*

theorem *MUL_totally_correct*:
"$\vdash_t \{\lambda s.\ 0 \le s\ ''y'' \wedge s{=}sorig\}$
MUL
$\{\lambda s.\ s\ ''z'' = s\ ''x'' * s\ ''y'' \wedge (\forall v.\ v \notin \{''z'', ''c''\} \longrightarrow s\ v = sorig\ v)\}$"

theorem *SQRT_totally_correct*:
"$\vdash_t \{\lambda s.\ s{=}sorig \wedge s\ ''x'' \ge 0\}$
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 10.1 Division

Submission until Sunday, Jan 24, 23:59. Write a simple IMP program that divides (positive) inputs p by q and returns the results n and r such that $q * n + r = p$ and $0 \le r < q$, i.e. n is the result and r the reminder.

One of the difficulties here is that there is no minus operation in IMP - so you'll likely have to use two loops.

definition *DIV* :: *com*

Prove correctness using the VCG.

The actual proof should be easy once you got the invariants right.

theorem *DIV_correct*:
"$\vdash \{\lambda s.\ s{=}s_0 \wedge s\ ''p'' > 0 \wedge s\ ''q'' > 0\}$
DIV
$\{\lambda s.\ s_0\ ''q'' * s\ ''n'' + s\ ''r'' = s_0\ ''p'' \wedge 0 \le s\ ''r'' \wedge s\ ''r'' < s_0\ ''q''\}$"

Now, prove total correctness.

theorem *DIV_t_correct*:
"$\vdash_t \{\lambda s.\ s{=}s_0 \wedge s\ ''p'' > 0 \wedge s\ ''q'' > 0\}$
DIV
$\{\lambda s.\ s_0\ ''q'' * s\ ''n'' + s\ ''r'' = s_0\ ''p'' \wedge s\ ''r'' < s_0\ ''q''\}$"

Homework 10.2 Collecting Semantics

Submission until Sunday, Jan 24, 23:59.

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 concrete semantics book (submit as text/comment in your submission file or via email).

```
x := 2; y := 1 {A0} ;
{A1}
WHILE 0 < x
DO {A2} ( y := y * x; x := x - 1 {A3} )
{A4}
```

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.