Technische Universität München
Fakultät für Informatik
Dr. Peter Lammich
Simon Wimmer

WS 2018/19
05. 02. 2019

Semantics of Programming Languages

Exercise Sheet 15

Exercise 15.1 Program Verification

(Pen & Paper)

Solution Aux lemma:

```
lemma lran_eq_replicate_conv: "lran a l h = replicate n x ⟷ (∀ i∈{l..<h}. a i = x) ∧ n=nat
(h−l)"
  apply (auto simp: list_eq_iff_nth_eq)
  using zle_iff_zadd by auto

program_spec check_anbn
  assumes "0≤h"
  ensures "(i>j) ⟷ (∃ n. lran a 0 h = replicate n 0 @ replicate n 1)"
  defines ‹
    i = 0;
    j = h − 1;
    while (i<j ∧ a[i] == 0 ∧ a[j] == 1)
      @variant ‹j›
      @invariant ‹0≤i ∧ j<h ∧ i = h−1 − j ∧ i − 1 ≤ j
        ∧ lran a 0 i = replicate (nat i) 0
        ∧ lran a (j+1) h = replicate (nat i) 1
        ›
    {
      i=i+1;
      j=j−1
    }
  ›
  supply [simp del] = lran_tail
  apply vcg_cs
  subgoal by (clarsimp simp: lran_eq_replicate_conv Ball_def) smt
  subgoal premises prems for a h j
  proof −
    let ?i="h − 1 − j"

    consider "?i = j" | "?i=j+1" | "?i<j" "a ?i ≠ 0" | "?i<j" "a j ≠ 1"
      using prems(2−4) by linarith
```

```
then show ?thesis proof cases
  case 1
  hence [simp]: "h = 2*j + 1" by auto
  have "∄ n. lran a 0 h = replicate n 0 @ replicate n 1"
  proof (rule_tac ccontr; clarsimp)
    fix n assume "lran a 0 (2 ∗ j + 1) = replicate n 0 @ replicate n 1"
    then have "nat (2 ∗ j + 1) = n + n"
      by − (drule arg_cong[where f=length], simp)
    then show False
      using ⟨?i = j⟩ ⟨j < h⟩ by (smt int_nat_eq of_nat_add)
  qed
  then show ?thesis by auto
next
  case 2
  let ?n = "nat ?i"
  from ⟨j < h⟩ ⟨h − 2 ≤ 2 ∗ j⟩ have "lran a 0 h = lran a 0 (j + 1) @ lran a (j + 1) h"
    by (simp add: lran_split)
  also have "... = replicate ?n 0 @ replicate ?n 1"
    using prems(3,4,5,6) 2 by (simp add: lran_split)
  finally show ?thesis
    using 2 by auto
next
  case 3
  have "?i ≥ 0"
    using ⟨j < h⟩ by simp
  have "∄n. lran a 0 h = replicate n 0 @ replicate n 1"
  proof (rule_tac ccontr; clarsimp)
    fix n assume A: "lran a 0 h = replicate n 0 @ replicate n 1"
    then have "nat h = n + n"
      by − (drule arg_cong[where f=length], simp)
```

Just stating that you use $?i \geq 0$, $?i < j$, and $j < h$ would be enough here.

```
    have B: "lran a 0 h = lran a 0 ?i @ a ?i # lran a (?i + 1) h"
      apply (subst lran_split[where p = ?i])
      subgoal
        using ⟨?i ≥ 0⟩ .
      subgoal
        using ⟨?i < j⟩ ⟨?i ≥ 0⟩ by simp
      by (smt lran_prepend1 ⟨j < h⟩ ⟨?i < j⟩)
    have "nat ?i ≥ n"
```

You do not need to provide the following justification in an exam

```
    proof (rule ccontr)
      assume "¬ n ≤ nat (h − 1 − j)"
      with ⟨?i ≥ 0⟩ have "n > nat ?i"
        by auto
      with A B show False
        by (clarsimp simp: list_eq_iff_nth_eq nth_append)
```

```
        (metis ‹a ?i ≠ 0› ‹0 ≤ ?i› ‹nat ?i < n› int_eq_iff nth_replicate trans_less_add1)
    qed
    moreover have "2 * ?i < h"
    proof -
      have "2 * ?i < h ⟷ 2 * ?i < ?i + j + 1"
        by simp
      also have "... ⟷ ?i < j + 1"
        by (simp add: algebra_simps)
      also have "... ⟷ True"
        using ‹?i < j› by simp
      finally show ?thesis
        by simp
    qed
    ultimately show False
      using ‹nat h = n + n› ‹j < h› by (auto simp add: algebra_simps)
  qed
  with 3 show ?thesis
    by auto
next
  case 4
```

An informal proof would look similar to case *3*.

```
  then show ?thesis using prems
    apply (clarsimp simp: list_eq_iff_nth_eq nth_append)
    apply (rule_tac exI[where x="nat j"])
    apply auto
    done
qed
qed
done
```

Exercise 15.2 Hoare-Logic

(Pen & Paper)

Solution:

1. No. Consider $t \neq t'$, and $R = UNIV$. Then $(c,s) \Rightarrow t$ and $(c,s) \Rightarrow t'$ for any s.
2. $wlp\ (REL\ R)\ Q\ s = (\forall t.\ (s,\ t) \in R \longrightarrow Q\ t)$
3. **Soundness** We first show $(REL\ R,\ s) \Rightarrow t \longleftrightarrow (s,\ t) \in R$. In the $\longrightarrow$-direction this follows by rule inversion on the big step, and in the $\longleftarrow$-direction we use

rule *Rel*. Now

$$\begin{aligned} & wlp\ (REL\ R)\ Q\ s \\ \longleftrightarrow\quad & (\forall\, t.\ (REL\ R,\ s) \Rightarrow t \longrightarrow Q\ t) \\ \longleftrightarrow\quad & (\forall\, t.\ (s,\ t) \in R \longrightarrow Q\ t) \end{aligned}$$

Completeness We need to show $HT\_partial\ (wlp\ c\ Q)\ (REL\ R)\ Q$.

$$\begin{aligned} & HT\_partial\ (wlp\ c\ Q)\ (REL\ R)\ Q \\ \longleftrightarrow\quad & (\forall\, s.\ wlp\ c\ Q\ s \longrightarrow wlp\ c\ Q\ s) \\ \longleftrightarrow\quad & True \end{aligned}$$