Verifikation nebenläufiger Programme
Wintersemester 2004/05

Ulrich Hannemann Jan Bredereke

21 A Proof System for GCL$^+$ Programs

To describe partial correctness of a program S, we use correctness formulae of the form $\{p\}S\{q\}$, where p, q are assertions, p is called a *precondition* and q a *postcondition*. We often refer to a formula $\{p\}S\{q\}$ as a *Hoare triple* or *correctness formula*. Informally, such a triple expresses that if p holds in the initial state of S, i.e., for the values of the variables at the start of the execution of S, then q holds for any final state of S, that is, if a computation of S terminates then q holds for the values of the variables at termination; for instance, $\{x = 5\}\, x := x{+}1\, \{x = 6\}$ is valid according to the definition below. As observed previously, e.g., in Session 2, apart from program variables, one in general needs logical variables, collected in the set $Lvar$, in order to characterise programs by means of pre- and postconditions. Therefore, let $VAR \stackrel{\text{def}}{=} Lvar \cup Pvar$ and let $\Sigma \stackrel{\text{def}}{=} VAR \to VAL$ in the definition of $T[\![S]\!]$.

Definition 21.1 (Validity of a correctness formula) For a program S and assertions p and q, a correctness formula $\{p\}S\{q\}$ is *valid*, if $\models \{p\}T[\![S]\!]\{q\}$. This is expressed by $\models \{p\}S\{q\}$. □

Observe that such a formula expresses partial correctness, since validity of $\{p\}T[\![S]\!]\{q\}$ expresses partial correctness for the underlying transition system $T[\![S]\!]$ of S.

In the previous session we defined the semantics $\mathcal{O}[\![S]\!]$. One can also characterise correctness of Hoare formulae in terms of this simple relational semantics:

Lemma 21.2 $\models \{p\}S\{q\}$ iff for all σ, if $\sigma \models p$ and $(\sigma, \tau) \in \mathcal{O}[\![S]\!]$, then $\tau \models q$.

Proof
For arbitrary transition systems P, the definition of $\{p\}P\{q\}$, from Session 2, amounts to the following, when choosing $T[\![S]\!]$ for P:

$\models \{p\}T[\![S]\!]\{q\}$ iff for all σ, if $\sigma \models p$ and $(\sigma, \tau) \in \mathcal{O}[\![T[\![S]\!]]\!]$, then $\tau \models q$. Together with Definition 20.4, stating that $\mathcal{O}[\![S]\!] = \mathcal{O}[\![T[\![S]\!]]\!]$, the lemma follows immediately.

■

Next we formulate a proof system PS_{seq} in which all valid Hoare triples can be formally derived. In such a proof system we can derive formulae of the form $\{p\}\ S\ \{q\}$ by means of *axioms*, which allow the derivation of Hoare triples without any assumption, and *rules* of the form

$$\frac{\ldots,\ \{p_i\}\ S_i\ \{q_i\},\ \ldots,\ p_j \to q_k,\ \ldots}{\{p\}\ S\ \{q\}}\ ,$$

by which the formula below the line, called the *conclusion* of the rule, can be derived if we have derived all the formulae above the line, which constitute the *premises* of the rule.

The basic axioms are as follows.

Axiom 21.1 (Skip) $\{p\}$ **skip** $\{p\}$.

Axiom 21.2 (Assignment) $\{q[\bar{e}/\bar{x}]\}\ \bar{x} := \bar{e}\ \{q\}$.

Here $q[\bar{e}/\bar{x}]$ denotes the substitution of each free occurrence of variable x_i by expression e_i within q.

Example 21.3 Since $(x = 5 \wedge y = 3)[x + y/x] \equiv x + y = 5 \wedge y = 3$, the assignment axiom yields $\{x + y = 5 \wedge y = 3\}\ x := x + y\ \{x = 5 \wedge y = 3\}$. □

Axiom 21.3 (Guard) $\{b \to q\}\ b\ \{q\}$.

Axiom 21.4 (Guarded assignment) $\{b \to q[\bar{e}/\bar{x}]\}\ \langle b \to \bar{x} := \bar{e}\rangle\ \{q\}$.

Observe that the above set of axioms is not minimal: Axioms 21.1, 21.2, and 21.3 can all be derived from Axiom 21.4.

First we give a general rule which can be applied to any statement. With this rule the precondition of an already derived Hoare triple can be strengthened and the postcondition can be weakened.

Rule 21.5 (Consequence)

$$\frac{p \to p_0,\ \{p_0\}\ S\ \{q_0\}, q_0 \to q}{\{p\}\ S\ \{q\}}.$$

Example 21.4 From Example 21.3 above we have $\{x + y = 5 \wedge y = 3\}\ x := x + y\ \{x = 5 \wedge y = 3\}$. Assuming that we can derive the valid implication $(x = 2 \wedge y = 3) \to (x + y = 5 \wedge y = 3)$, the consequence rule leads to

$$\{x = 2 \wedge y = 3\}\ x := x + y\ \{x = 5 \wedge y = 3\}.$$ □

Rule 21.6 (Sequential composition)

$$\frac{\{p\}\ S_1\ \{r\},\ \{r\}\ S_2\ \{q\}}{\{p\}\ S_1; S_2\ \{q\}}.$$

Example 21.5 By the assignment axiom and the consequence rule one can derive $\{x = 2\}\ y := 3\ \{x = 2 \wedge y = 3\}$, and $\{x = 2 \wedge y = 3\}\ x := x + y\ \{x = 5 \wedge y = 3\}$. Hence, the sequential composition rule leads to

$$\{x = 2\}\ y := 3; x := x + y\ \{x = 5 \wedge y = 3\}.$$ □

Rule 21.7 (Choice)

$$\frac{\{p\}\ S_i\ \{q\},\ \text{for all } i \in \{1, \ldots, n\}}{\{p\}\textbf{if}\ \ []_{i=1}^{n} S_i\ \ \textbf{fi}\ \{q\}}.$$

Rule 21.8 (Guarded command)

$$\frac{\{p \wedge b_i\}\ S_i\ \{q\}, \text{ for all } i \in \{1, \ldots, n\}}{\{p\}\ \textbf{if}\quad [\!]_{i=1}^{n}\, b_i \to S_i\ \ \textbf{fi}\ \ \{q\}}.$$

Example 21.6 Since we can derive $\{y = 0 \wedge x = 0\}\ y := 1\ \{x = 0 \wedge y = 1\}$, the rule for guarded command leads to

$$\{y = 0\}\ \textbf{if}\quad x = 0 \to y := 1\ \ \textbf{fi}\ \ \{x = 0 \wedge y = 1\}. \qquad \square$$

For the iteration construct we have the following rules:

Rule 21.9 (Exit-loop)

$$\frac{\{p\}S_B\{p\},\ \{p\}S_E\{q\}}{\{p\}\textbf{do}\ S_B\ [\!]\ S_E\ ;\ \textbf{exit}\ \ \textbf{od}\{q\}}.$$

Rule 21.10 (Do-loop)

$$\frac{\{p \wedge b_i\}\ S_i\ \{p\}}{\{p\}\ \textbf{do}\ \ [\!]_{i=1}^{n}\, b_i \to S_i\ \ \textbf{od}\ \{p \wedge \neg b_G\}}.$$

Example 21.7 (Greatest common divisor) Consider

$$\textbf{do}\ \ x < y \to y := y - x [\!] x > y \to x := x - y\ \ \textbf{od}.$$

We would like to derive

$$\begin{gathered}\{x = n \wedge y = m\}\\ \textbf{do}\ \ x < y \to y := y - x [\!] x > y \to x := x - y\ \ \textbf{od}\\ \{x = y = gcd(n, m)\}.\end{gathered}$$

How this is done, using the above axiom system, is shown below.

By Axiom 21.2 one derives

$$\{gcd(x, y - x) = gcd(n, m)\}\ y := y - x\ \{gcd(x, y) = gcd(n, m)\},$$

and by the consequence rule

$$\{x < y \wedge gcd(x, y - x) = gcd(n, m)\}\ y := y - x\ \{gcd(x, y) = gcd(n, m)\}.$$

Since

$$\models (x < y \wedge gcd(x, y - x) = gcd(n, m)) \leftrightarrow (x < y \wedge gcd(x, y) = gcd(n, m)),$$

this yields, again by the consequence rule,

$$\{x < y \wedge gcd(x, y) = gcd(n, m)\}\ y := y - x\ \{gcd(x, y) = gcd(n, m)\}.$$

Similarly one derives

$$\{x > y \wedge gcd(x, y) = gcd(n, m)\}\ x := x - y\ \{gcd(x, y) = gcd(n, m)\}.$$

Then Rule 21.10 leads to

$$\begin{array}{c}\{gcd(x,y) = gcd(n,m)\}\\ \textbf{do}\ \ x < y \rightarrow y := y - x [] x > y \rightarrow x := x - y\ \ \textbf{od}\\ \{gcd(x,y) = gcd(n,m) \wedge x = y\}.\end{array}$$

By the consequence rule we derive

$$\begin{array}{c}\{x = n \wedge y = m\}\\ \textbf{do}\ \ x < y \rightarrow y := y - x [] x > y \rightarrow x := x - y\ \ \textbf{od}\\ \{x = y = gcd(n,m)\}.\end{array} \tag{1}$$

Example 21.8 (Integer division: proof system approach) One of the simplest algorithms for division of a non-negative integer x by a positive integer y is based on repeated subtraction, and is represented in GCL as follows:

$Div_1 \equiv$
$q, r := 0, x;$
do
$\quad y \leq r \rightarrow q, r := q + 1, r - y$
od.

The desired postcondition of this program is given by $x = q * y + r \wedge 0 \leq r < y$. We show how one can prove the above algorithm correct, using the proof system of this section. By Axiom 21.2 we derive

$$\{x = (q+1) * y + r - y \wedge y \leq r\} q, r := q + 1, r - y \{x = q * y + r \wedge 0 \leq r\}.$$

By the consequence rule and some properties of $I\!N$ we derive

$$\{(x = q * y + r \wedge 0 \leq r) \wedge y \leq r\} q, r := q + 1, r - y \{x = q * y + r \wedge 0 \leq r\}$$

and hence by the do-loop rule

$$\begin{array}{c}\{x = q * y + r \wedge 0 \leq r\}\\ \textbf{do}\ y \leq r \rightarrow q, r := q + 1, r - y\ \ \textbf{od}\\ \{(x = q * y + r \wedge 0 \leq r) \wedge r < y\}.\end{array}$$

Again we apply the assignment Axiom 21.2 to derive

$$\{x = 0 * y + x \wedge 0 \leq x\}\ q, r := 0, x \{x = q * y + r \wedge 0 \leq r\}.$$

We see that the above precondition is equivalent to $0 \leq x$, so after another application of the consequence rule 21.5 we finally derive by the sequential composition rule 21.6

$\{0 \leq x\}$
$q, r := 0, x;$
do
$\quad y \leq r \rightarrow q, r := q + 1, r - y$
od
$\{(x = q * y + r \wedge 0 \leq r) \wedge r < y\}.$

Observe that the derived postcondition is equivalent to the desired postcondition. Hence we obtain

$$\{0 \leq x\}\ Div_1\ \{x = q * y + r \wedge 0 \leq r < y\}. \qquad \square$$

Note that the rules in PS_{seq} only use the pre- and postconditions of the Hoare triples above the line. The structure of the programs in these triples is not relevant and hence in the rule for a compound construct the components can be considered as black boxes. With such a proof system we can verify the design steps during the process of top-down program development.

Usually, soundness of a proof system is demonstrated by proving that its axioms are valid and that its rules preserve validity, i.e., if the hypotheses of a rule are valid then so is its conclusion. An induction argument on the length of the derivation of a correctness formula then suffices to prove validity of any formula derived in that system.

In our set-up a slightly different strategy is followed because validity of a Hoare-style correctness formula for S is by definition equivalent to validity of the corresponding correctness formula for $T[\![S]\!]$, and for the latter a sound and complete characterisation has been already given. Hence, soundness and relative completeness of our Hoare logic will be proved as a corollary of the following key result of this chapter, which states that correctness of a program in Hoare logic corresponds with the existence of an inductive assertion network for its associated transition diagram, and vice versa.

In order to prove this result, we need to establish the structural properties of assertion networks for transition diagrams corresponding to GCL$^+$ statements, e.g., that an assertion network $\mathcal{Q}$ for $T[\![S_1; S_2]\!]$ can be obtained as a composition $\mathcal{Q}_1; \mathcal{Q}_2$ (defined below) of assertion networks for $T[\![S_i]\!]$, $i = 1, 2$. When examining the composition operations for transition systems one sees that essentially the component networks are "glued" together by unifying the appropriate entry and exit locations. The composition operations for assertion networks assume that the assertions for these unified locations are the same for all the component assertion networks. As a consequence, the composition operations for assertion networks are *partial* functions.

Notice that in the present syntactic set-up, an assertion network for $T[\![S]\!] \stackrel{\text{def}}{=} (L, T, s, t)$ is a function $\mathcal{Q}$ associating an *assertion* $\mathcal{Q}(l)$ with every location $l \in L$. This is justified as every predicate needed can be expressed as an assertion from first-order logic over $I\!N$ (see Chapter 5 of [dRdBH$^+$01] for an extensive proof of this).

Definition 21.9 (Operations on assertion networks)

- Let P_1 and P_2 be transition systems represented by (L_1, T_1, s, r) and (L_2, T_2, r, t), where $L_1 \cap L_2 = \{r\}$, and assume that $\mathcal{Q}_1$ and $\mathcal{Q}_2$ are assertion networks for P_1 and P_2. The assertion network $\mathcal{Q}_1; \mathcal{Q}_2$ for $P_1; P_2$ is defined as follows, provided that $\mathcal{Q}_1(r) \equiv \mathcal{Q}_2(r)$, i.e., $\mathcal{Q}_1(r)$ and $\mathcal{Q}_2(r)$ are syntactically equal:

$$\begin{array}{lll} (\mathcal{Q}_1; \mathcal{Q}_2)(l) & \stackrel{\text{def}}{=} \mathcal{Q}_1(l), & \text{for } l \in L_1, \\ & \stackrel{\text{def}}{=} \mathcal{Q}_2(l), & \text{for } l \in L_2. \end{array}$$

- Let P_i be a transition system represented by (L_i, T_i, s, t), for $i = 1, \ldots, n$, where $L_i \cap L_j = \{s, t\}$, for $1 \leq i < j \leq n$, and assume that $\mathcal{Q}_i$ is an assertion network for P_i, for $i = 1, \ldots, n$. The assertion network **if** $[]_{i=1}^{n} \mathcal{Q}_i$ **fi** is defined for **if** $[]_{i=1}^{n} P_i$ **fi** provided that $\mathcal{Q}_i(s) \equiv \mathcal{Q}_j(s)$ and $\mathcal{Q}_i(t) \equiv \mathcal{Q}_j(t)$ for $1 \leq i < j \leq n$:

$$\textbf{if } []_{i=1}^{n} \mathcal{Q}_i \textbf{ fi } (l) \stackrel{\text{def}}{=} \mathcal{Q}_i(l), \qquad \text{for } l \in L_i.$$

- Let P_B and P_E be transition systems represented by (L_B, T_B, s, r) and (L_E, T_E, s, t), where $L_B \cap L_E = \{s\}$. Assume that $\mathcal{Q}_B$ and $\mathcal{Q}_E$ are assertion networks for P_B and P_E. The assertion network **do** $\mathcal{Q}_B$ [] $\mathcal{Q}_E$; **exit od** for **do** P_B [] P_E ; **exit od** is defined provided that $\mathcal{Q}_B(s) \equiv \mathcal{Q}_B(r)$, and $\mathcal{Q}_B(s) \equiv \mathcal{Q}_E(s)$:

$$\textbf{do } \mathcal{Q}_B \text{ [] } \mathcal{Q}_E \text{ ; } \textbf{exit od}(l) \begin{array}{ll} \stackrel{\text{def}}{=} \mathcal{Q}_B(l), & \text{for } l \in L_B \backslash \{r\}, \\ \stackrel{\text{def}}{=} \mathcal{Q}_E(l), & \text{for } l \in L_E. \end{array} \tag{2}$$

The operations introduced above are *partial* operations. If for some operation and assertion networks $\mathcal{Q}_i$ the operation is defined, then we say that the $\mathcal{Q}_i$ networks are *compatible*.

Lemma 21.10 (Structural properties of (inductive) assertion networks)

1. $\mathcal{Q}$ is an assertion network for P_1 ; P_2 iff there exist compatible assertion networks $\mathcal{Q}_1$ and $\mathcal{Q}_2$ for P_1 and P_2 s.t. $\mathcal{Q} = \mathcal{Q}_1$; $\mathcal{Q}_2$. Moreover, $\mathcal{Q}$ is inductive iff $\mathcal{Q}_1$ and $\mathcal{Q}_2$ are inductive.
2. $\mathcal{Q}$ is an assertion network for **if** $[]_{i=1}^{n} P_i$ **fi** iff there exist compatible assertion networks $\mathcal{Q}_1, \ldots, \mathcal{Q}_n$ for $P_1, \ldots, P_n$, with $\mathcal{Q} =$ **if** $[]_{i=1}^{n} \mathcal{Q}_i$ **fi** . Moreover, $\mathcal{Q}$ is inductive iff $\mathcal{Q}_1, \ldots, \mathcal{Q}_n$ are inductive.
3. $\mathcal{Q}$ is an assertion network for **do** P_B [] P_E ; **exit od** iff there exist compatible assertion networks $\mathcal{Q}_B$ and $\mathcal{Q}_E$ for P_B and P_E, such that $\mathcal{Q} =$ **do** $\mathcal{Q}_B$ [] $\mathcal{Q}_E$; **exit od**. Moreover, $\mathcal{Q}$ is inductive iff $\mathcal{Q}_B$ and $\mathcal{Q}_E$ are inductive.

Proof
No proof

■

Before we state the relation between a Hoare-logic proof and the existence of inductive assertion networks we first introduce some notation. Let S be a GCL$^+$ program: let $\vdash \{p\}\mathcal{Q}(S)\{q\}$ abbreviate the claim that there exists an inductive assertion network $\mathcal{Q}(S)$ for $T[\![S]\!]$ such that for the entry location s one has $\mathcal{Q}(S)(s) = p$, and for the exit location t one has $\mathcal{Q}(S)(t) = q$. Let $\vdash \{p\}T[\![S]\!]\{q\}$ abbreviate the claim that there exists an inductive assertion network for $T[\![S]\!]$ which is correct w.r.t. p and q. One should notice the difference between $\vdash \{p\}\mathcal{Q}(S)\{q\}$ and $\vdash \{p\}T[\![S]\!]\{q\}$. The latter formula states that there exists an inductive assertion network $\mathcal{Q}$ for $T[\![S]\!]$ such that for the entry location s one has that $\models p \rightarrow \mathcal{Q}(s)$ holds, and for the exit location t that $\models \mathcal{Q}(t) \rightarrow q$ holds.

Theorem 21.11 (Equivalence between Hoare logic and Floyd's inductive assertion method)
Let S be a GCL$^+$ program.

- If S is guarded then

$$\vdash \{p\}S\{q\} \text{ iff } \vdash \{p\}\mathcal{Q}(S)\{q\}.$$

- If S is not guarded then

$$\vdash \{p\}S\{q\} \text{ iff there exists an } I \text{ such that } \models p \to I \text{ and } \vdash \{I\}\mathcal{Q}(S)\{q\}.$$

- $\vdash \{p\}S\{q\}$ iff $\vdash \{p\}T\,[\![S]\!]\,\{q\}$.

Proof
Observe that the last claim of the theorem follows directly from the first two. Next we simultaneously prove, by induction on the syntactic structure of S, the implication ($\Rightarrow$) for the first two claims.

Basic case: Guarded assignment.
Let $S \equiv \langle b \to \bar{x} := \bar{e}\rangle$ and assume $\vdash \{p\}S\{q\}$. This formula can only be derived from Axiom 21.4 by applying the rule of consequence several times. Hence there exist p' and q' with

$$\vdash \{p'\}S\{q'\} \text{ with } p' \equiv b \to q'[\bar{e}/\bar{x}],\ \models p \to p',\ \text{and}\ \models q' \to q.$$

From this we deduce $\models p \to (b \to q'[e/x])$, hence $\models p \to (b \to q[\bar{e}/\bar{x}])$, which is equivalent to $\models p \wedge b \to q[\bar{e}/\bar{x}]$. This is, by definition of inductive, exactly the verification condition for a syntactic assertion network for $T\,[\![\langle b \to \bar{x} := \bar{e}\rangle]\!]$ with assertion p for the entry location, and assertion q for the exit location. Hence

$$\vdash \{p\}\mathcal{Q}(\langle b \to \bar{x} := \bar{e}\rangle)\{q\}.$$

Induction step:

- Sequential composition.
  Let $S \equiv S_1\,;\,S_2$ and assume $\vdash \{p\}S\{q\}$. This formula can only be derived from the sequential composition rule, and by applying the rule of consequence several times afterwards. Hence there exist p', q', and r such that

$$\vdash \{p'\}S_1\{r\},\ \vdash \{r\}S_2\{q'\} \text{ and } \models p \to p',\ \models q' \to q.$$

  Hence by the rule of consequence $\vdash \{r\}S_2\{q\}$. Thus by the induction hypothesis, there exists an r' with $\models r \to r'$ and $\vdash \{r'\}\mathcal{Q}(S_2)\{q\}$, choosing $r' \stackrel{\text{def}}{=} r$ in case S_2 is guarded. Moreover, again by the rule of consequence applied to $\models p \to p'$, $\models r \to r'$ and $\vdash \{p'\}S_1\{r\}$ we deduce $\vdash \{p\}S_1\{r'\}$. By the induction hypothesis, there exists an I with $\models p \to I$, $\vdash \{I\}\mathcal{Q}(S_1)\{r'\}$, and $p \equiv I$ if S_1 is guarded. Notice that the premises for the sequential composition of $\mathcal{Q}(S_1)$ and $\mathcal{Q}(S_2)$ are met, cf. Definition 21.9, and by Lemma 21.10 this sequential composition is inductive. Hence $\vdash \{I\}\mathcal{Q}(S_1\,;\,S_2)\{q\}$. Moreover $p \equiv I$ if S_1 and thus S is guarded.

- Guarded choice.
  Let $S \equiv \textbf{if}\ \ [\!]_{i=1}^{n} S_i\ \ \textbf{fi}$ with all S_i guarded, and assume $\vdash \{p\}S\{q\}$. Then by the same arguments as above there exist p' and q' such that $\models p \to p'$, $\models q' \to q$ and $\vdash \{p'\}\textbf{if}\ \ [\!]_{i=1}^{n} S_i\ \ \textbf{fi}\ \{q'\}$ is deduced from the choice rule 21.7. Hence $\vdash \{p'\}S_i\{q'\}$ for all i. Now by the rule of consequence $\vdash \{p\}S_i\{q\}$ for all i. Moreover all S_i are guarded, hence by the induction hypothesis $\vdash \{p\}\mathcal{Q}(S_i)\{q\}$. Also the premises for the choice construct between assertion networks are met, and by Lemma 21.10 this choice is inductive. Hence $\vdash \{p\}\mathcal{Q}(\textbf{if}\ \ [\!]_{i=1}^{n} S_i\ \ \textbf{fi}\ )\{q\}$.

- Iteration.
  Let $S \equiv \textbf{do}\ S_B [\!] S_E\ ;\ \textbf{exit}\ \ \textbf{od}$ with S_B and S_E guarded, and assume $\vdash \{p\}\ S\ \{q\}$. By the same arguments as above we deduce the existence of I and q' such that $\models p \to I$, $\models q' \to q$, $\vdash \{I\}S_B\{I\}$ and $\vdash \{I\}S_E\{q'\}$, and therefore $\vdash \{I\}S_E\{q\}$. By the induction hypothesis we deduce $\vdash \{I\}\mathcal{Q}(S_B)\{I\}$ and $\vdash \{I\}\mathcal{Q}(S_E)\{q\}$ since S_B and S_E are guarded. This yields, using Lemma 21.10,

  $$\vdash \{I\}\mathcal{Q}(\textbf{do}\ S_B [\!] S_E\ ;\ \textbf{exit}\ \ \textbf{od})\{q\}.$$

  Since $\models p \to I$ the result follows.

This finishes the proof of the implication $\Rightarrow$ for the first two claims. For the converse implication we only have to prove that $\vdash \{p\}\mathcal{Q}(S)\{q\}$ implies $\vdash \{p\}S\{q\}$, since in case $\models p \to I$ and $\vdash \{I\}\ \mathcal{Q}(S)\ \{q\}$ this implies $\vdash \{I\}S\{q\}$, and therefore $\vdash \{p\}S\{q\}$. This is quite straightforward, so that we only prove the basic case and one induction step.

Basic case: Guarded assignment.
Assume $\vdash \{p\}\mathcal{Q}(\langle b \to \bar{x} := \bar{e}\rangle)\{q\}$, then by definition of inductiveness $\models p \wedge b \to q[\bar{e}/\bar{x}]$, which is equivalent to $\models p \to (b \to q[\bar{e}/\bar{x}])$. Hence by the consequence rule and the guarded assignment axiom: $\vdash \{p\}\langle b \to \bar{x} := \bar{e}\rangle\{q\}$.
Induction step:

- Sequential composition.
  Let $S \equiv S_1\ ;\ S_2$ and assume $\vdash\ \{p\}\mathcal{Q}(S_1\ ;\ S_2)\{q\}$. By the structural properties of inductive assertion networks, cf. Lemma 21.10, there exist inductive assertion networks $\mathcal{Q}_1$ for S_1, and $\mathcal{Q}_2$ for S_2, such that $\mathcal{Q}(S_1\ ;\ S_2) = \mathcal{Q}_1\ ;\ \mathcal{Q}_2$. By Definition 21.9 we have that $\mathcal{Q}_1(r) = \mathcal{Q}_2(r)$, where r is the exit location of S_1 and the entry location of S_2. Put $r' \stackrel{\text{def}}{=} \mathcal{Q}_1(r)$, then $\vdash \{p\}\mathcal{Q}(S_1)\{r'\}$ and $\vdash \{r'\}\mathcal{Q}(S_2)\{q\}$. Hence by induction hypothesis $\vdash \{p\}S_1\{r'\}$ and $\vdash \{r'\}S_2\{q\}$. Applying the sequential composition rule yields $\vdash \{p\}S_1\ ;\ S_2\{q\}$, the desired result.

- The other cases are left as an exercise to the reader.

■

In the above theorem, the existence of an I which satisfies $\models p \to I$ and also $\vdash \{I\}\mathcal{Q}(S)\{q\}$ is essential in case S is a loop construct, as can been seen from Example 21.7, see also the exercises.

Now soundness and relative completeness of our Hoare logic for GCL$^+$ can be established as a corollary of Theorem 21.11.

Theorem 21.12 (Soundness and relative completeness)
Proof system PS_{seq} for GCL$^+$ is sound and relatively complete.

■

References

[dRdBH+01] Willem-Paul de Roever, Frank S. de Boer, Ulrich Hannemann, Jozef Hooman, Yassine Lakhneche, Mannes Poel, and Job Zwiers. *Concurrency Verification.* Number 54 in Cambridge Tracts in Theoretical Computer Science. Cambridge University Press, Cambridge, UK, April 2001.