G52DOA - Total Correctness

Venanzio Capretta

Total Correctness

So far we learned proof rules to show that a program is *partially correct*: the specification holds if the computation terminates. But nothing so far guarantees that the the program will at some point stop. It is possible that the computation will go on forever for some input, in which case our proof of correctness doesn't tell us anything. In fact, it is possible to write a silly program that satisfies any partial correctness specification by never stopping:

		$\{P\}$
$\{\mathsf{invariant} : \top\}$	while true do	$\{\top \wedge \mathsf{true}\}$
$\{\top\}$	skip	$\{\top\}$
		$\{Q\}$

Here, the extra implications that we need to prove are trivial:

$$P \to \top$$
$$\top \wedge \neg\mathsf{true} \to Q$$

The first one is trivially true because its conclusion, $\top$, is always true. The second one is trivially true because one of its premises, $\neg\mathsf{true}$, is always false.

It is therefore essential that we extend our proof rules to allow us to prove *total correctness*, that is, to prove termination of while loops. Let's use square brackets, in place of curly ones, to indicate that the specification requires termination. Therefore, the following Hoare triple:

$$[P]\ p\ [Q]$$

states that *"if we start the program p in a state satisfying P, then* ***the computation will terminate*** *in a state satisfying Q."*

For the skip, assignment, and conditional commands, there is no difference between proving partial or total correctness, because those instructions just perform a single operation. The only instruction that may cause non-termination is the while loop. To ensure that a loop terminates, we must show that its boolean condition must become false sooner or later. This is done by choosing an integer expression, called the *variant*, and showing that it decreases at every repetition of the loop body, and that it becomes zero or negative only if the boolean condition is false. Expressed as a derivation rule, this gives:

$$\frac{[I \land b \land \mathsf{exp} = e_0]\ p\ [I \land \mathsf{exp} < e_0] \qquad I \land b \rightarrow \mathsf{exp} > 0}{[I]\ \mathsf{while}\ b\ \mathsf{do}\ p\ [I \land \neg b]}$$

This rule extends the previous rule of partial correctness by introducing the variant expression exp. This is such that whenever the condition b is true, that is, whenever the loop body is executed, exp is positive. We use the ghost variable e_0 to denote the value of exp at the beginning of the execution of the loop body. The rule then requires that, after execution of the loop body, the value of exp has become smaller. Since exp is an integer expression, if it keeps decreasing, it must sooner or later become non-positive. At that point the condition b cannot be true anymore, because it implies that exp is positive, and the loop terminates.

The way to use this rule inside a tableau proof is to declare the variant just before the while loop, on the left, in the same position where we declare the invariant, and then proceed similarly as before:

$$\begin{array}{r|c|l}
 & & [P] \\
[\mathsf{invariant} : I; \mathsf{variant} : \mathsf{exp}] & \mathsf{while}\ b\ \mathsf{do} & [I \land b \land \mathsf{exp} = e_0] \\
[R] & p & [I \land \mathsf{exp} < e_0] \\
 & & [Q]
\end{array}$$

The extra implications that we have to prove are:

$$\begin{array}{l}
P \rightarrow I \\
I \land b \rightarrow \mathsf{exp} > 0 \\
I \land b \land \mathsf{exp} = e_0 \rightarrow R \\
I \land \neg b \rightarrow Q
\end{array}$$

Usually, we separate the proof of partial correctness from the proof of termination. This is in many cases straightforward and can be carried out by just examining the change of the variant during the loop body. In other words, quite often we don't need to know the invariant I to determine that the variant decreases. (But be careful, in some cases the knowledge of I is essential.)

As an example, let's look at the program whose partial correctness we proved in the last lecture:

$$\begin{array}{r|l|l}
 & & \{\mathsf{x} > 0 \land \mathsf{x} = x_0\} \\
\{\mathsf{x} > 0 \land \mathsf{x} = x_0 \land 0 = 0\} & \mathsf{y} := 0; & \{\mathsf{x} > 0 \land \mathsf{x} = x_0 \land \mathsf{y} = 0\} \\
\{\mathsf{invariant} : I\} & \mathsf{while}\ \mathsf{x} > 0\ \mathsf{do}\ (& \{I \land \mathsf{x} > 0\} \\
\{I[\mathsf{x} - 1/\mathsf{x}][\mathsf{x} + \mathsf{y}/\mathsf{y}]\} & \mathsf{y} := \mathsf{x} + \mathsf{y}; & \{I[\mathsf{x} - 1/\mathsf{x}]\} \\
\{I[\mathsf{x} - 1/\mathsf{x}]\} & \mathsf{x} := \mathsf{x} - 1 & \{I\} \\
 &) & \{I \land \neg \mathsf{x} > 0\} \\
 & & \{\mathsf{y} = x_0(x_0 + 1)/2\}
\end{array}$$

The proof of termination is in this case trivial and doesn't depend on the invariant:

$$\begin{array}{r|l|l}
 & \mathsf{y} := 0; & \\
[\mathsf{variant} : \mathsf{x}] & \mathsf{while}\ \mathsf{x} > 0\ \mathsf{do}\ (& [\mathsf{x} = x_0] \\
[\mathsf{x} - 1 < x_0] & \mathsf{y} := \mathsf{x} + \mathsf{y}; & [\mathsf{x} - 1 < x_0] \\
[\mathsf{x} - 1 < x_0] & \mathsf{x} := \mathsf{x} - 1 & [\mathsf{x} < x_0] \\
 &) &
\end{array}$$

With the extra, trivial, requirement that the following implications hold:

$$\mathsf{x} > 0 \rightarrow \mathsf{x} > 0$$
$$\mathsf{x} = x_0 \rightarrow \mathsf{x} - 1 < x_0.$$