G52DOA - Loops and Invariants

Venanzio Capretta

Weakest Precondition for Conditionals

Given the postcondition for a conditional command, how do we compute a precondition that is as weak as possible (so it is easiest to satisfy)? That is, given a postcondition Q, how do we compute the weakest P such that the following is a correct Hoare Logic derivation:

$$
\begin{array}{r|l|l}
\{P\} & \mathsf{if}\ b\ \mathsf{then} & \{P \wedge b\} \\
\{R_1\} & \quad p_1 & \{Q\} \\
 & \mathsf{else} & \{P \wedge \neg b\} \\
\{R_2\} & \quad p_2 & \{Q\} \\
 & & \{Q\}
\end{array}
$$

where R_1 and R_2 are the weakest preconditions computed from the postcondition Q for the subprograms p_1 and p_2, respectively.

According to the derivation rule for conditionals, we must be able to prove the following two implications:

$$
\begin{array}{l}
P \wedge b \rightarrow R_1 \\
P \wedge \neg b \rightarrow R_2.
\end{array}
$$

The weakest proposition P for which these are provable is the one we are looking for:

$$P = (b \rightarrow R_1) \wedge (\neg b \rightarrow R_2).$$

Let us illustrate this kind of weakest precondition computation to prove that the following (needlessly complicated) program computes the minimum of two numbers.

$$
\begin{array}{r|l|l}
\{P[\mathsf{x}+\mathsf{y}/\mathsf{u}][\mathsf{x}-\mathsf{y}/\mathsf{z}]\} & \mathsf{z} := \mathsf{x}-\mathsf{y}; & \{P[\mathsf{x}+\mathsf{y}/\mathsf{u}]\} \\
\{P[\mathsf{x}+\mathsf{y}/\mathsf{u}]\} & \mathsf{u} := \mathsf{x}+\mathsf{y}; & \{P\} \\
\{P\} & \mathsf{if}\ \mathsf{z} < 0\ \mathsf{then} & \{P \wedge b\} \\
\{(\mathsf{u}+\mathsf{z})/2 = \min(\mathsf{x},\mathsf{y})\} & \quad \mathsf{u} := \mathsf{u}+\mathsf{z} & \{\mathsf{u}/2 = \min(\mathsf{x},\mathsf{y})\} \\
 & \mathsf{else} & \{P \wedge \neg b\} \\
\{(\mathsf{u}-\mathsf{z})/2 = \min(\mathsf{x},\mathsf{y})\} & \quad \mathsf{u} := \mathsf{u}-\mathsf{z} & \{\mathsf{u}/2 = \min(\mathsf{x},\mathsf{y})\} \\
 & \ ; & \{\mathsf{u}/2 = \min(\mathsf{x},\mathsf{y})\} \\
\{\mathsf{u}/2 = \min(\mathsf{x},\mathsf{y}\} & \mathsf{z} := \mathsf{u}/2 & \{\mathsf{z} = \min(\mathsf{x},\mathsf{y})\}
\end{array}
$$

with

$$\begin{aligned}
P = \quad & (\mathsf{z} < 0 \to (\mathsf{u}+\mathsf{z})/2 = \min(\mathsf{x},\mathsf{y})) \wedge \\
& (\neg\mathsf{z} < 0 \to (\mathsf{u}-\mathsf{z})/2 = \min(\mathsf{x},\mathsf{y})) \\
P[\mathsf{x}+\mathsf{y}/\mathsf{u}] = \quad & (\mathsf{z} < 0 \to (\mathsf{x}+\mathsf{y}+\mathsf{z})/2 = \min(\mathsf{x},\mathsf{y})) \wedge \\
& (\neg\mathsf{z} < 0 \to (\mathsf{x}+\mathsf{y}-\mathsf{z})/2 = \min(\mathsf{x},\mathsf{y})) \\
P[\mathsf{x}+\mathsf{y}/\mathsf{u}][\mathsf{x}-\mathsf{y}/\mathsf{z}] = \quad & (\mathsf{x}-\mathsf{y} < 0 \to (\mathsf{x}+\mathsf{y}+\mathsf{x}-\mathsf{y})/2 = \min(\mathsf{x},\mathsf{y})) \wedge \\
& (\neg\mathsf{x}-\mathsf{y} < 0 \to (\mathsf{x}+\mathsf{y}-(\mathsf{x}-\mathsf{y}))/2 = \min(\mathsf{x},\mathsf{y}))
\end{aligned}$$

To complete the proof of correctness we have to show that this last proposition follows from the the precondition of the program; since that precondition is just $\top$, then we must just prove $P[\mathsf{x}+\mathsf{y}/\mathsf{u}][\mathsf{x}-\mathsf{y}/\mathsf{z}]$, which simplifies to:

$$(\mathsf{x} < \mathsf{y} \to \mathsf{x} = \min(\mathsf{x},\mathsf{y})) \wedge (\neg\mathsf{x} < \mathsf{y} \to \mathsf{y} = \min(\mathsf{x},\mathsf{y})).$$

Proof Tableaux for Loops

Remember the Hoare Logic rule for while loops:

$$\frac{\{I \wedge b = \mathsf{true}\}\ p\ \{I\}}{\{I\}\ \mathsf{while}\ b\ \mathsf{do}\ p\ \{I \wedge b = \mathsf{false}\}}\ \text{Loop}$$

The explanation for this rules is this:

- The *invariant* I remains true at each execution of the loop;
- I must be true at the beginning, before the execution of the while instruction;
- It the body of the loop is executed, this means that, a part from I, also b must be true;
- If we execute the body of the loop, p, starting in a state satisfying $I \wedge b =$ true, then, after the execution of p, the invariant I must again be true;
- When we exit the loop, we are guaranteed that I is still true and the loop test b has become false.

When we construct a proof tableau for a while loop, We must first of all declare I as an invariant before the loop on the left:

$\{\text{invariant} : I\}$	while b do p	

Then we know that I and b will be true just inside the loop and I and $\neg b$ will be true after the termination of the loop:

$\{\text{invariant} : I\}$	while b do p	$\{I \wedge b\}$ $\{I \wedge \neg b\}$

To prove that I is actually an invariant we must show that it is still true after the execution of the loop body p from a state satisfying $I \wedge b$:

$$\begin{array}{r|l|l} \{\text{invariant} : I\} & \textsf{while } b \textsf{ do} & \{I \wedge b\} \\ & \quad p & \{I\} \\ & & \{I \wedge \neg b\} \end{array}$$

We must then propagate this proposition backwards through p to find the weaker precondition for p that guarantees I to be true at the end of p:

$$\begin{array}{r|l|l} \{\text{invariant} : I\} & \textsf{while } b \textsf{ do} & \{I \wedge b\} \\ \{R\} & \quad p & \{I\} \\ & & \{I \wedge \neg b\} \end{array}$$

For this to be valid, we must show, exploiting the implication rule, that R follows from the assertion just inside the loop:

$$I \wedge b \to R.$$

Finally, if we had started by trying to prove that the loop satisfies a given Hoare triple:

$$\{P\} \textsf{ while } b \textsf{ do } p \ \{Q\}$$

Then the complete proof looks like this:

$$\begin{array}{r|l|l} & & \{P\} \\ \{\text{invariant} : I\} & \textsf{while } b \textsf{ do} & \{I \wedge b\} \\ \{R\} & \quad p & \{I\} \\ & & \{I \wedge \neg b\} \\ & & \{Q\} \end{array}$$

where we still need to prove the implications:

$$\begin{array}{l} P \to I \\ I \wedge b \to R \\ I \wedge \neg b \to Q. \end{array}$$

As an example, here is the complete partial correctness proof for a program that computes the addition of all the positive integers up to x. We want to prove the Hoare triple:

$\{\mathsf{x} > 0 \wedge \mathsf{x} = x_0\}\ \mathsf{y} := 0; \textsf{while } \mathsf{x} > 0 \textsf{ do } (\mathsf{y} := \mathsf{x} + \mathsf{y}; \mathsf{x} := \mathsf{x} - 1)\ \{\mathsf{y} = x_0(x_0+1)/2\}.$

We start by putting it in the tableaux form:

$$\begin{array}{r|l|l} & & \{\mathsf{x} > 0 \wedge \mathsf{x} = x_0\} \\ & \mathsf{y} := 0; & \\ \{\text{invariant} : ?\} & \textsf{while } \mathsf{x} > 0 \textsf{ do } (& \\ & \mathsf{y} := \mathsf{x} + \mathsf{y}; & \\ & \mathsf{x} := \mathsf{x} - 1 & \\ &) & \\ & & \{\mathsf{y} = x_0(x_0+1)/2\} \end{array}$$

What should the invariant be? If we think for a moment about what the program does, we realize that it keeps adding the value of x to y and then decreasing x. Therefore, the value of y should be, at any point in the computation:

$$\mathsf{y} = x_0 + (x_0 - 1) + \cdots + (\mathsf{x} + 2) + (\mathsf{x} + 1) = \sum_{i=\mathsf{x}+1}^{i=x_0} i = \frac{(x_0 + \mathsf{x} + 1)(x_0 - \mathsf{x})}{2}.$$

This is just a hypothesis, we don't yet know whether it is true or false. We must also keep track of the fact that x never becomes negative:

$$I = \left(\mathsf{y} = \frac{(x_0 + \mathsf{x} + 1)(x_0 - \mathsf{x})}{2} \wedge \mathsf{x} \geq 0\right).$$

Let's try to choose this assertion as our invariant and see if we can prove it:

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

Now, we must first of all prove that the invariant is true at the beginning; let's fill in the first part of the tableaux:

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

Therefore, we need to prove that $\mathsf{x} > 0 \wedge \mathsf{x} = x_0 \wedge \mathsf{y} = 0 \to I$. This follows from the fact that a summation over an empty range is defined to be 0 by convention:

$$\sum_{i=x_0+1}^{i=x_0} i = 0$$

because the starting index of the summation, $x_0 + 1$ is already larger than the last index x_0.

Now we have to propagate I backward inside the body of the loop, using the weakest precondition calculation:

		$\{\mathsf{x} > 0 \wedge \mathsf{x} = x_0\}$
$\{\mathsf{x} > 0 \wedge \mathsf{x} = x_0 \wedge 0 = 0\}$	$\mathsf{y} := 0;$	$\{\mathsf{x} > 0 \wedge \mathsf{x} = x_0 \wedge \mathsf{y} = 0\}$
$\{\mathsf{invariant} : I\}$	$\mathsf{while}\ \mathsf{x} > 0\ \mathsf{do}\ ($	$\{I \wedge \mathsf{x} > 0\}$
$\{I[\mathsf{x}-1/\mathsf{x}][\mathsf{x}+\mathsf{y}/\mathsf{y}]\}$	$\mathsf{y} := \mathsf{x} + \mathsf{y};$	$\{I[\mathsf{x}-1/\mathsf{x}]\}$
$\{I[\mathsf{x}-1/\mathsf{x}]\}$	$\mathsf{x} := \mathsf{x} - 1$	$\{I\}$
	$)$	$\{I \wedge \neg\mathsf{x} > 0\}$
		$\{\mathsf{y} = x_0(x_0+1)/2\}$

where:

$$\begin{aligned} I[\mathsf{x}-1/\mathsf{x}] &= \left(\mathsf{y} = \frac{(x_0 + (\mathsf{x}-1)+1)(x_0 - (\mathsf{x}-1))}{2} \wedge (\mathsf{x}-1) \geq 0\right) \\ &= \left(\mathsf{y} = \frac{(x_0+\mathsf{x})(x_0-\mathsf{x}+1)}{2} \wedge \mathsf{x} > 0\right) \\ I[\mathsf{x}-1/\mathsf{x}][\mathsf{x}+\mathsf{y}/\mathsf{y}] &= \left(\mathsf{x}+\mathsf{y} = \frac{(x_0+\mathsf{x})(x_0-\mathsf{x}+1)}{2} \wedge \mathsf{x} > 0\right). \end{aligned}$$

We need to prove the implication:

$$I \wedge \mathsf{x} > 0 \to I[\mathsf{x}-1/\mathsf{x}][\mathsf{x}+\mathsf{y}/\mathsf{y}]$$

which follows from:

$$\begin{aligned} \mathsf{x} + \frac{(x_0+\mathsf{x}+1)(x_0-\mathsf{x})}{2} &= \frac{2\mathsf{x} + (x_0+\mathsf{x})(x_0-\mathsf{x}) + x_0 - \mathsf{x}}{2} \\ &= \frac{(x_0+\mathsf{x})(x_0-\mathsf{x}) + x_0 + \mathsf{x}}{2} \\ &= \frac{(x_0+\mathsf{x})(x_0-\mathsf{x}+1)}{2}. \end{aligned}$$

Finally, we must prove that the exit assertion of the loop implies the postcondition:

$$I \wedge \neg\mathsf{x} > 0 \to \mathsf{y} = x_0(x_0+1)/2$$

which is easily verified, since from I and $\neg\mathsf{x} > 0$ it follows that $\mathsf{x} = 0$.

Algebraic Manipulation of Proposition

We say that two propositions are equivalent if each implies the other:

$$A \equiv B = (A \to B) \wedge (B \to A).$$

The following rules of equivalence are provable in natural deduction:

Distributivity:	$A \wedge (B \vee C) \equiv (A \wedge B) \vee (A \wedge C)$
	$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C)$
de Morgan laws:	$\neg(A \wedge B) \equiv (\neg A) \vee (\neg B)$
	$\neg(A \vee B) \equiv (\neg A) \wedge (\neg B)$
Negation of Quantifiers:	$\neg\forall x, A \equiv \exists x, \neg A$
	$\neg\exists x, A \equiv \forall x, \neg A$
Material Implication:	$A \to B \equiv \neg A \vee B.$

Furthermore, propositions that do not depend on a variable x can be moved freely in and out of quantifiers:

$$\left.\begin{array}{l} A \wedge \forall x, B \equiv \forall x, A \wedge B \\ A \vee \forall x, B \equiv \forall x, A \vee B \\ A \to \forall x, B \equiv \forall x, A \to B \end{array}\right\} \text{if } x \text{ does not occur free in } A.$$

For example, using these algebraic laws, we can show that the proposition:

$$\forall x, \forall y, (\mathsf{E}(x,y) \to \forall z, (\mathsf{E}(y,z) \to \forall u, (\neg \mathsf{E}(z,u) \vee \neg \exists v, (\mathsf{E}(u,v) \wedge \mathsf{E}(v,x)))))$$

is equivalent to:

$$\neg \exists x, \exists y, \exists z, \exists u, \exists v, (\mathsf{E}(x,y) \wedge \mathsf{E}(y,z) \wedge \mathsf{E}(z,u) \wedge \mathsf{E}(u,v) \wedge \mathsf{E}(v,x))$$

which reveals that the meaning of the proposition is *"there is no pentagon of edges"*.