Verifikation nebenläufiger Programme
Wintersemester 2004/05

Ulrich Hannemann Jan Bredereke

9 The Proof Method of Owicki & Gries

9.1 Formulating a complete version

The solution of Owicki and Gries to the particular form of incompleteness signalled in Example 8.2 is the introduction of *auxiliary* variables that do not occur in the original transitions of a program but are added to their assignments in order to be able to *express assumptions about the other components*. These variables are not allowed in conditions inside transitions. Furthermore, auxiliary variables should not occur in the original assignments of the program – they only occur in assignments to auxiliary variables themselves, and thus the values of the program variables are also not affected by adding auxiliary variables. Within our semantic set up this is expressed by requiring that conditions c in our original program do not depend on these auxiliary variables, in the sense defined in Session 2. Hence *auxiliary variables do not influence control flow*, since the enabledness of transitions does not change by adding auxiliary variables.

Summarising, we have the following formal definition of auxiliary variables:

Definition 9.1 (Auxiliary variables) A set of program variables $\bar{z} = z_1, \ldots, z_n$ is a set of auxiliary variables of a program P if

- for any boolean condition c of P, $\bar{z} \cap var(c) = \emptyset$,
- for any state transformation f of P there exist state transformations g and h such that $f = g \circ h$, $\bar{z} \cap var(g) = \emptyset$, and the write variables of h are among $\bar{z}$.

(For the definition of $var(f)$ and the write variables of a function f we refer to Session 2.) □

Observe that the above second condition expresses that every state transformation f of P can be decomposed in a state transformation g which does not involve the auxiliary variables and a state transformation h which changes only the auxiliary variables. Note also that logical variables trivially satisfy this definition.

Now, the test for interference freedom allows one to check the consistency of the introduced assumptions when these are expressed in terms of the program variables and the auxiliary variables.

Example 9.2 (Continuation of Example 8.2) In our example we can use two auxiliary variables z_1 and z_2 to encode the location which the control flow of a process has reached: $z_i = 0$ iff P_i is at location s_i, and $z_i = 1$ iff P_i is at location t_i. Therefore we augment the program with assignments to these auxiliary variables, resulting in $P' \equiv P_1' \parallel P_2'$ as in Figure 1.

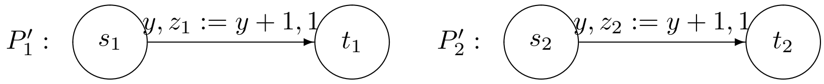
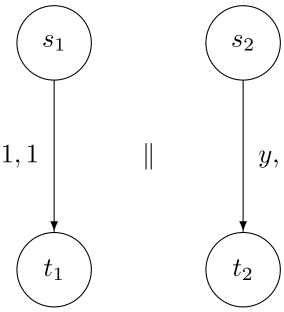


Figure 1: Adding auxiliary variables z_1 and z_2 to the program from Figure 4 of Session 8.

In the predicates defined in Figure 2 these auxiliary variables are used to express the relation between the values of y and the locations of the other process.

$$\mathcal{Q}_{s_1} \stackrel{\text{def}}{=} z_1 = 0 \wedge (z_2 = 0 \rightarrow y = 0) \wedge (z_2 = 1 \rightarrow y = 1)$$

$$\mathcal{Q}_{s_2} \stackrel{\text{def}}{=} z_2 = 0 \wedge (z_1 = 0 \rightarrow y = 0) \wedge (z_1 = 1 \rightarrow y = 1)$$



$$\mathcal{Q}_{t_1} \stackrel{\text{def}}{=} z_1 = 1 \wedge (z_2 = 0 \rightarrow y = 1) \wedge (z_2 = 1 \rightarrow y = 2)$$

$$\mathcal{Q}_{t_2} \stackrel{\text{def}}{=} z_2 = 1 \wedge (z_1 = 0 \rightarrow y = 1) \wedge (z_1 = 1 \rightarrow y = 2)$$

Figure 2: The use of auxiliary variables in predicates allows for the expression of interference free assertion networks.

We prove that this modified program P' is partially correct with respect to the specification $< y = 0 \wedge z_1 = 0 \wedge z_2 = 0, y = 2 >$:

1. Local correctness of P'_1 and P'_2 is straightforward.

2. Interference freedom:

   - Assume $\mathcal{Q}_{s_1} \wedge \mathcal{Q}_{s_2}$ holds, that is, $z_1 = 0 \wedge z_2 = 0 \wedge y = 0$ holds. Then after executing $y, z_2 := y + 1, 1$ we have $z_1 = 0 \wedge z_2 = 1 \wedge y = 1$, and thus $\mathcal{Q}_{s_1}$ holds.
   - Assume $\mathcal{Q}_{t_1} \wedge \mathcal{Q}_{s_2}$ holds, that is, $z_1 = 1 \wedge z_2 = 0 \wedge y = 1$ holds. Then after executing $y, z_2 := y + 1, 1$ we have $z_1 = 1 \wedge z_2 = 1 \wedge y = 2$, and thus $\mathcal{Q}_{t_1}$ holds.
   - Symmetrically, $\mathcal{Q}_{s_2}$ and $\mathcal{Q}_{t_2}$ are invariant under $y, z_1 := y + 1, 1$.

3. Clearly, $\models y = 0 \wedge z_1 = 0 \wedge z_2 = 0 \rightarrow \mathcal{Q}_{s_1} \wedge \mathcal{Q}_{s_2}$ and $\models \mathcal{Q}_{t_1} \wedge \mathcal{Q}_{t_2} \rightarrow y = 2$.

Hence P' is partially correct w.r.t. specification $< y = 0 \wedge z_1 = 0 \wedge z_2 = 0, y = 2 >$. $\square$

However, we started out wishing to prove P to be partially correct w.r.t. $< y = 0, y = 2 >$! So, *how does one argue that the former, a statement about P' involving z_1, z_2 and y, implies the latter, a statement involving P and only y?*

P''s partial correctness w.r.t. $< y = 0 \wedge z_1 = 0 \wedge z_2 = 0, y = 2 >$ means that every terminating $(y = 0 \wedge z_1 = 0 \wedge z_2 = 0)$-computation terminates in a state satisfying $y = 2$. Then also every terminating $(y = 0)$-computation terminates in a state satisfying $y = 2$, since (1) z_1 and z_2 do not occur in tests, and hence do not have any influence on the flow of control during program execution, and (2) neither z_1 nor z_2 occur in postcondition $y = 2$. That is, whatever the values of z_1 and z_2 are at the beginning of the computation, the same sequence of instructions from P_1 is executed as for $z_1 = 0 \wedge z_2 = 0$ at the beginning of that sequence, while the postcondition remains valid. Moreover, they do not affect assignments to y. That is, not only is the sequence of instructions executed for initial state $y = 0$ independent of the values of z_1 and z_2, but also the state transformation of y between the beginning and end of P' is independent of these values. Hence P' is partially correct w.r.t. specification $< y = 0, y = 2 >$.

This argument summarises soundness of the following *initialisation rule*, because we can initialise the auxiliary variables z_1 and z_2 both to 0 so that the old precondition $y = 0 \wedge z_1 = 0 \wedge z_2 = 0$ results in a new precondition $y = 0$ for P', while preserving partial correctness of P'.

Rule 9.1 (Initialisation rule)

$$\frac{\{\varphi\}\ P\ \{\psi\}}{\{\varphi \circ f\}\ P\ \{\psi\}},$$

where f is a function such that its write variables constitute a set of auxiliary variables for P which do not occur in ψ.

Here the format

$$\frac{\{\varphi_1\}\ P_1\ \{\psi_1\}}{\{\varphi_2\}\ P_2\ \{\psi_2\}}$$

is used to express the rule that $\models \{\varphi_1\}\ P_1\ \{\psi_1\}$ *implies* $\models \{\varphi_2\}\ P_2\ \{\psi_2\}$. If the latter is the case, the rule is called *sound*.

Example 9.3 (Continuation of Example 8.2) In more detail, with $\mathcal{Q}_{s_i}$ as in Figure 2 above, the following equivalences hold:

$$\begin{array}{ll} & \mathcal{Q}_{s_1} \wedge \mathcal{Q}_{s_2} \\ \leftrightarrow & z_1 = 0 \wedge (z_2 = 0 \rightarrow y = 0)\ \wedge (z_2 = 1 \rightarrow y = 1) \wedge \\ & z_2 = 0 \wedge (z_1 = 0 \rightarrow y = 0) \wedge (z_1 = 1 \rightarrow y = 1) \\ \leftrightarrow & \text{(by propositional logic)}\ \ z_1 = 0 \wedge z_2 = 0 \wedge y = 0. \end{array}$$

Choosing $(z_1, z_2) := (0, 0)$ for f, one has

$$\models (z_1 = 0 \wedge z_2 = 0 \wedge y = 0) \circ f \leftrightarrow y = 0.$$

Now, using these two results and given that

$$\{z_1 = 0 \wedge z_2 = 0 \wedge y = 0\}\ P'\ \{y = 2\}$$

holds for P' as above, the initialisation rule states:

$$\frac{\{z_1 = 0 \wedge z_2 = 0 \wedge y = 0\}\ P'\ \{y = 2\}}{\{y = 0\}\ P'\ \{y = 2\}}$$

and therefore (soundness of this rule) leads to

$$\models \{y = 0\}\ P'\ \{y = 2\}.$$

Please, observe that $\models y = 0 \rightarrow z_1 = 0 \wedge z_2 = 0 \wedge y = 0$ does not hold. Hence, one needs an *extra* rule to justify the step from $\models \{z_1 = 0 \wedge z_2 = 0 \wedge y = 0\}\ P'\ \{y = 2\}$ to $\models \{y = 0\}\ P'\ \{y = 2\}$. This justifies the initialisation rule, applied above. □

This raises as the next question how to get rid of P' in $\models \{y = 0\}\ P'\ \{y = 2\}$, for it is our intention to prove $\models \{y = 0\}\ P\ \{y = 2\}$!

Since every $(y = 0)$-computation in P has a *corresponding* $(y = 0)$-computation in P' which assigns the same values to y, we also obtain that P is partially correct w.r.t. $< y = 0, y = 2 >$.

This second argument summarises application of Owicki & Gries' so-called *auxiliary variables rule*, stating that a correctness statement about P' in the postcondition of which no auxiliary variables occur implies the similar statement about P, where P is obtained from P' by removing auxiliary variables.

Rule 9.2 (Owicki & Gries' auxiliary variables rule) Let $\bar{z}$ be a set of auxiliary variables of P'. Then

$$\frac{\{\varphi\}\ P'\ \{\psi\}}{\{\varphi\}\ P\ \{\psi\}}$$

provided $\bar{z} \cap var(\psi) = \emptyset$ and P is obtained from P' by restricting the state transformations of P' to all the variables excluding the auxiliary variable set $\bar{z}$. More precisely, let f be a state transformation of P' such that $f = g \circ h$, where g does not involve $\bar{z}$ and the write variables of h are among $\bar{z}$, then g is the corresponding state transformation of P.

Example 9.4 (Continuation of Example 8.2) In the case of our example, application of the auxiliary variables rule amounts to

$$\frac{\{y = 0\}\ (y, z_1) := (y + 1, 1) \parallel (y, z_2) := (y + 1, 1)\ \{y = 2\}}{\{y = 0\}\ y := y + 1 \parallel y := y + 1\ \{y = 2\}},$$

where the above assignments stand for the corresponding transition diagrams from Figures 3 of Session 8 and 1. Consequently its soundness gives that from

$$\models \{y = 0\}\ P'\ \{y = 2\}$$

one derives

$$\models \{y = 0\}\ P\ \{y = 2\},$$

with P as defined in Example 8.2. □

The general formulation of the proof method of Owicki & Gries [OG76] is given below.

Definition 9.5 (The proof method of Owicki & Gries) Consider $P \equiv P_1 \parallel \ldots \parallel P_n$. To prove $\{\varphi\}P\{\psi\}$ we introduce the *proof method of Owicki & Gries*:

1. Augment P_i by introducing auxiliary variables; every action $b \to f$ can be extended as follows: $b \to f \circ g$, where g is a state transformation such that its write variables are among the auxiliary variables $\bar{z}$ where $\bar{z} \cap var(\varphi, P, \psi) = \emptyset$. This leads to an augmented transition diagram $P' \equiv P'_1 \parallel \ldots \parallel P'_n$.

2. Associate a predicate $\mathcal{Q}_l$ with every location l of P'_i.

3. Prove *local correctness* of every P'_i: For every transition $l \xrightarrow{a} l'$ of P'_i, assuming $a \equiv b \to f$, we prove

$$\models \mathcal{Q}_l \wedge b \to \mathcal{Q}_{l'} \circ f.$$

4. Prove *interference freedom*, that is, for every transition $l \xrightarrow{a} l'$ of P'_i, and for every predicate $\mathcal{Q}_{l''}$ associated to a location l'' of P'_j, with $j \neq i$, assuming $a \equiv b \to f$,[1] we prove

$$\models \mathcal{Q}_l \wedge \mathcal{Q}_{l''} \wedge b \to \mathcal{Q}_{l''} \circ f.$$

5. Prove
   - $\models \varphi \to (\bigwedge_{i=1}^{n} \mathcal{Q}_{s_i}) \circ h$, for some state transformation h whose write variables $write(h)$ belong to the set of auxiliary variables $\bar{z}$, and where s_i denotes the initial location of P'_i, and
   - $\models (\bigwedge_{i=1}^{n} \mathcal{Q}_{t_i}) \to \psi$, where t_i denotes the final location of P'_i. $\square$

Let us trace how the proof method of Owicki & Gries has been applied in case of our example. Step 1 corresponds with the transformation of P in Figure 3 of Session 8 to P' as in Figure 1. Step 2 is given by Figure 2, step 3 is straightforward, and step 4 has been checked above. The first part of step 5 is trivial, since $\varphi \equiv y = 0$ and

$$\models \bigwedge_i \mathcal{Q}_{s_i} \circ h \leftrightarrow \bigwedge_i \mathcal{Q}_{s_i} \circ (z_1, z_2) := (0,0) \leftrightarrow 0 = 0 \wedge 0 = 0 \wedge y = 0 \leftrightarrow\ y = 0,$$

choosing $(z_1, z_2) := (0,0)$ for h; the second part of step 5 amounts to proving the validity of

$$\models z_1 = 1 \wedge z_2 = 1 \wedge y = 2 \to y = 2,$$

which is trivial.

Observe that when $n = 1$ this proof method still makes sense.

References

[OG76] S. Owicki and D. Gries. An axiomatic proof technique for parallel programs. *Acta Informatica*, 6:319–340, 1976.

[1] The intention is here that $b \to f$ identifies a label occurring in P', i.e., it is of the form $b \to f' \circ g$ with $b \to f'$ occurring in P.