A detailed derivation of a very simple program

We are going to derive a program "test all true", satisfying the following specification:

|[con N: int ; con f: int → bool ; var x: bool
; test all true {R: $x \equiv \langle\forall k : 0\le k \wedge k<N : f.k\rangle$}
]|

* * *

Our first step is to rewrite the right-hand side of R as a function (whose properties we'll subsequently exploit) of some argument(s). Since functions of integers are simpler than functions of functions, we consider the function f as a constant, but do consider the right-hand side of R a function of the two bounds of the range, that is, we rewrite the postcondition

R: $x \equiv H.0.N$

where H is given by

(0) $H.i.j \equiv \langle\forall k : i\le k \wedge k<j : f.k\rangle$.

* * *

Our next step is —unavoidably!— the investigation of the (hopefully exploitable) properties of the function H, just introduced. Standard application of predicate calculus (and *nothing* else but lemmata like

$$k<j \lor k=j \quad \equiv \quad k<j+1 \quad)$$

yields the following three properties of H

(1) $\quad i \geq j \quad \Rightarrow \quad H.i.j$

(2) $\quad i<j \quad \Rightarrow \quad (H.(i+1).j \land f.i \equiv H.i.j)$

(3) $\quad i<j \quad \Rightarrow \quad (H.i.(j-1) \land f.(j-1) \equiv H.i.j)$

In (1) we recognize "bases" — (1) holds for all i,j ! — in (2) and (3) "steps".

* * *

Heading for a repetion, our third step is –unavoidably!– the choice of an invariant that can be (i) initialized, (ii) maintained, and (iii) used to conclude R . Postponing our concerns regarding (ii), we observe that –thanks to Leibniz's Principle– we can initialize

$$H.i.j \equiv H.0.N$$

–viz. by $i,j := 0,N$ – but that the left-hand side has to be generalized so as to contain x in such a fashion that we can conclude R . We choose for invariant $P0 \land P1$ with

P0: $\quad H.i.j \land x \equiv H.0.N$

P1: $\quad 0 \leq i \land j \leq N \quad .$

Ad P0 we remark that the alternative left-hand sides $H.i.j \vee x$ and $H.i.j \equiv x$ would have given invariants as easily initialized as P0. But in order to conclude R, i.e.

$$x \equiv H.0.N$$

from

$$(H.i.j \ \underline{op}\ x) \equiv H.0.N \quad ,$$

the known value of $H.i.j$ should be the unit element of $\underline{op}$. Since (1) tells us that the known value of $H.i.j$ is true, this rules out the disjunction. In view of (2) and (3), the equivalence seems to lead to an invariant hard to maintain, and hence the conjunction in P0 .

* * *

Our fourth step is to determine under what additional condition the invariant implies R . To this end we observe

$$\begin{array}{ll}
 & R \\
= & \quad \{\text{definition of } R\} \\
 & x \equiv H.0.N \\
= & \quad \{P0\} \\
 & x \equiv x \wedge H.i.j \\
= & \quad \{\text{pred. calc.}\} \\
 & \neg x \vee H.i.j \\
\Leftarrow & \quad \{(1)\} \\
 & \neg x \vee i \geq j \qquad ,
\end{array}$$

after which observation we decide to use the negation of the last expression, i.e.

(4) $\quad x \wedge i<j$

as guard.

* * *

Our fifth step is to derive the command(s) guarded in the repetion. In order to use (2) we consider a guarded command that comprises $i:=i+1$ and leaves j constant. (Note that $i:=i+1$ and $j:=j-1$ work towards falsification of $i<j$, and that is nice for termination.) We consider for as yet unknown E the requirement that

$$x \wedge i<j \rightarrow i, x := i+1, E$$

leave Po invariant. We observe

$$\begin{array}{ll}
 & wp.(i,x := i+1, E).Po \\
= & \quad \{\text{def. } := \text{ ; def. } Po\} \\
 & H.(i+1).j \wedge E \equiv H.0.N \\
= & \quad \{Po\text{, ex hypothesi}\} \\
 & H.(i+1).j \wedge E \equiv H.i.j \wedge x \\
= & \quad \{x \text{ from the guard}\} \\
 & H.(i+1).j \wedge E \equiv H.i.j \\
= & \quad \{i<j \text{ from the guard, (2)}\} \\
 & H.(i+1).j \wedge E \equiv H.(i+1).j \wedge f.i \\
\Leftarrow & \quad \{\text{Leibniz}\} \\
 & E \equiv f.i
\end{array}$$

i.e. the guarded command

$$x \wedge i<j \;\rightarrow\; i,x := i+1, f.i$$

leaves P0 invariant. Similarly we derive from the requirement that P0 is maintained that the guarded command

$$x \wedge i<j \;\rightarrow\; j,x := j-1, f.(j-1)$$

meets the requirement. Hence a solution is

```
test all true:
  |[ var i,j : int; i,j := 0,N  {inv: P0 ∧ P1; bnd j-i}
  ; do x ∧ i<j → i,x := i+1, f.i
     [] x ∧ i<j → j,x := j-1, f.(j-1)
    od {P0 ∧ (¬x ∨ i≥j), hence}
  ]| {R}
```

Remark In the context of this note it is not interesting that in the above program the nondeterminacy can be reduced (by removal of one of the guarded commands). Nondeterminacy can always be reduced. (End of Remark.)

* * *

Cognoscenti recognize an Eindhoven influence, in particular of Anne Kaldewaij, who did much to establish the superiority of the tail invariant, and of Wim H.J. Feijen, who taught me (in WF155) how to derive the guard $x > i{<}j$ "from outside in", that is by confronting the postcondition with the invariant. These influences are gratefully acknowledged.

This note has been written for the sake of the fifth step, in which the repeatable statement is derived. (I don't remember ever having done it that way.)

The way in which nondeterminacy entered this program was a surprise; it was a pleasure to see that it almost dictated a tail invariant.

Austin, 18 October 1993

PS The weakest solution of the equation

$$X: [X \wedge Y \Rightarrow Z]$$

is $Y \Rightarrow Z$. It helps to know this. (End of PS.)

prof. dr. Edsger W. Dijkstra
Department of Computer Sciences
The University of Texas at Austin
Austin, TX 78712-1188
USA