

476 Let $b: bin$ be the user's variable, and let $n: nat$ be the implementer's variable, and let the operations be

$step = \mathbf{if\ } n > 0 \mathbf{\ then\ } n := n - 1 \mathbf{\ else\ } ok \mathbf{\ fi}$

$done = b := n = 0$

Show that there is no transformer to get rid of n so that

$step$ is transformed to ok

$done$ is transformed to $b := \perp$

even though the user cannot detect the difference.

After trying the question, scroll down to the solution.

§ Suppose a user is imagining that there is a variable n with some natural value, and $step$ decreases it until it is 0 , and $done$ says whether n is 0 . But actually there is no variable n , and $step$ does nothing, and $done$ always says “ n is not yet 0 ”. The user will never know that there is no variable n . That's why the question says “the user cannot detect the difference”.

The transformer would have to be a binary expression D in variables b and n such that

- (0) $\exists n. D$
- (1) $\forall n. D \Rightarrow \exists n'. D' \wedge \mathbf{if } n>0 \mathbf{ then } n:=n-1 \mathbf{ else } ok \mathbf{ fi} \Leftarrow ok$
- (2) $\forall n. D \Rightarrow \exists n'. D' \wedge (b:=n=0) \Leftarrow b:=\perp$

Instead of a binary expression, let D be a relation between a binary and a natural. Then

- (0) $\exists n. D b n$
- (1) $\forall n. D b n \Rightarrow \exists n'. D b' n' \wedge \mathbf{if } n>0 \mathbf{ then } n:=n-1 \mathbf{ else } ok \mathbf{ fi} \Leftarrow ok$
- (2) $\forall n. D b n \Rightarrow \exists n'. D b' n' \wedge (b:=n=0) \Leftarrow b:=\perp$

Now calculate, starting with the left side of 1.

$$\begin{aligned}
& \forall n. D b n \Rightarrow \exists n'. D b' n' \wedge \mathbf{if } n>0 \mathbf{ then } n:=n-1 \mathbf{ else } ok \mathbf{ fi} \text{ replace } \mathbf{if} \text{ and } := \text{ and } ok \\
= & \forall n. D b n \Rightarrow \exists n'. D b' n' \wedge (n>0 \wedge n'=n-1 \wedge b'=b) \vee (n=0 \wedge n'=n \wedge b'=b) \\
& \text{distribute } D b' n' \\
= & \forall n. D b n \Rightarrow \exists n'. (D b' n' \wedge n>0 \wedge n'=n-1 \wedge b'=b) \vee (D b' n' \wedge n=0 \wedge n'=n \wedge b'=b) \\
& \text{splitting} \\
= & \forall n. D b n \Rightarrow (\exists n'. D b' n' \wedge n>0 \wedge n'=n-1 \wedge b'=b) \vee (\exists n'. D b' n' \wedge n=0 \wedge n'=n \wedge b'=b) \\
& \text{one-point} \\
= & \forall n. D b n \Rightarrow (D b'(n-1) \wedge n>0 \wedge b'=b) \vee (D b'n \wedge n=0 \wedge b'=b) \\
& \text{context} \\
= & \forall n. D b n \Rightarrow (D b(n-1) \wedge n>0 \wedge b'=b) \vee (D b n \wedge n=0 \wedge b'=b) \text{ factor out } b'=b \\
= & b'=b \wedge \forall n. D b n \Rightarrow (D b(n-1) \wedge n>0) \vee (D b n \wedge n=0) \text{ context} \\
= & b'=b \wedge \forall n. D b n \Rightarrow (D b(n-1) \wedge n>0) \vee (\top \wedge n=0) \text{ identity} \\
= & b'=b \wedge \forall n. D b n \Rightarrow (D b(n-1) \wedge n>0) \vee n=0 \text{ basic } \forall \text{ law} \\
= & b'=b \wedge (\forall n. 0 \cdot D b n \Rightarrow (D b(n-1) \wedge n>0) \vee n=0) \text{ The } 0 \text{ case is } \top. \\
& \wedge (\forall n. \text{nat}+1 \cdot D b n \Rightarrow (D b(n-1) \wedge n>0) \vee n=0) \\
= & b'=b \wedge \forall n. \text{nat}+1 \cdot D b n \Rightarrow D b(n-1) \text{ change of variable} \\
= & b'=b \wedge \forall n. \text{nat} \cdot D b(n+1) \Rightarrow D b n \text{ mirror} \\
= & b'=b \wedge \forall n. D b n \Leftarrow D b(n+1)
\end{aligned}$$

According to (1), this is refined by ok (in a context without n). So

$$\begin{aligned}
& b'=b \wedge (\forall n. D b n \Leftarrow D b(n+1)) \Leftarrow b'=b \text{ context} \\
= & \top \wedge (\forall n. D b n \Leftarrow D b(n+1)) \Leftarrow b'=b \text{ identity and mirror} \\
= & b'=b \Rightarrow \forall n. D b n \Leftarrow D b(n+1) \text{ one-point } b' \\
(3) \quad = & \forall n. D b n \Leftarrow D b(n+1) \\
= & D b n \text{ is antimonotonic in } n; \text{ in other words, an initial (possibly empty,} \\
& \text{possibly infinite) segment of } D b 0; D b 1; \dots \text{ is } \top, \text{ and after that it's } \perp.
\end{aligned}$$

Now calculate, starting with the left side of 2.

$$\begin{aligned}
& \forall n. D b n \Rightarrow \exists n'. D b' n' \wedge (b:=n=0) \text{ replace } := \\
= & \forall n. D b n \Rightarrow \exists n'. D b' n' \wedge b'=(n=0) \wedge n'=n \text{ factor out } b'=(n=0) \\
= & \forall n. D b n \Rightarrow b'=(n=0) \wedge \exists n'. D b' n' \wedge n'=n \text{ one-point } n' \\
= & \forall n. D b n \Rightarrow b'=(n=0) \wedge D b' n \text{ splitting} \\
= & (\forall n. D b n \Rightarrow b'=(n=0)) \wedge (\forall n. D b n \Rightarrow D b' n)
\end{aligned}$$

According to (2), this is refined by $b:=\perp$ (in a context without n). So, explicitly quantifying over b and b' ,

$$\begin{aligned}
& \forall b, b'. (\forall n. D b n \Rightarrow b'=(n=0)) \wedge (\forall n. D b n \Rightarrow D b' n) \Leftarrow (b:=\perp) \text{ expand } := \\
= & \forall b, b'. (\forall n. D b n \Rightarrow b'=(n=0)) \wedge (\forall n. D b n \Rightarrow D b' n) \Leftarrow b'=\perp \text{ one-point } b' \\
= & \forall b. (\forall n. D b n \Rightarrow n>0) \wedge (\forall n. D b n \Rightarrow D \perp n) \text{ contrapositive}
\end{aligned}$$

$$\begin{aligned}
&= \forall b. (\forall n. n=0 \Rightarrow \neg D b n) \wedge (\forall n. D b n \Rightarrow D \perp n) && \text{one-point } n \\
(4) \quad &= \forall b. \neg D b 0 \wedge (\forall n. D b n \Rightarrow D \perp n)
\end{aligned}$$

From (4) we have $\neg D b 0$. So from (3) we have $\forall n. \neg D b n$. And this contradicts (0). Hence there is no data transformer to do the job.

If the user is willing to test *done* to see if $n=0$ before each use of *step*, then we can weaken *step* to $n>0 \Rightarrow (n:=n-1)$, and perhaps find a transformer to transform *step* to *ok* and transform *done* to $b:=\perp$.