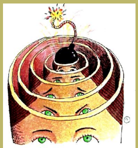
# Lecture 18:
# Floyd-Hoare Logic for Conditionals

Aims:

- To look at the inference rules for one- and two-armed conditionals.





## 18.1. Recap

- Let's start with an exercise that allows us to revise what we learned in the previous lecture.

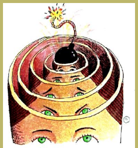
  Prove that $\vdash_{\text{par}} (\!|\, \textbf{True} \,|\!)\ \mathit{ProgA}\ (\!|\, u = x + y \,|\!)$ where *ProgA* is as follows:

  $z := x;$

  $z := z + y;$

  $u := z;$

- You can see that, because of the way we tackle proofs, i.e. backwards and using the rule of consequence, we never actually make explicit use of the Sequence rule. It *is* being used but only implicitly — in the way we lay out our proofs.



Recap
Conditionals
Module Home Page
Title Page
Page 3 of 8
Back
Full Screen
Close
Quit

## 18.2. Conditionals

- Two-armed-conditional

$$\frac{(\!|\, B \wedge P \,|\!)\, C_1\, (\!|\, Q \,|\!), \quad (\!|\, \neg B \wedge P \,|\!)\, C_2\, (\!|\, Q \,|\!)}{(\!|\, P \,|\!)\ \textbf{if}\ B\ C_1 \textbf{else}\ C_2\, (\!|\, Q \,|\!)}$$

  If $B$ is **true**, $C_1$ is executed; if $B$ is **false**, $C_2$ is executed. If we have proved that $C_1$ takes us from states satisfying $B \wedge P$ to states satisfying $Q$ and $C_2$ takes us from states satisfying $\neg B \wedge P$ to states satisfying $Q$, then we can conclude that the conditional command as a whole takes us from states satisfying $P$ to states satisfying $Q$.

- How do we push a condition $Q$ 'backwards' up through a two-armed conditional?

  1. Push $Q$ up through $C_1$. Call the result $P_1$.
  2. Push $Q$ up through $C_2$. Call the result $P_2$.
  3. Then the precondition of the conditional $P$ is $(B \Rightarrow P_1) \wedge (\neg B \Rightarrow P_2)$

- Prove that $\vdash_{\text{par}} (\!|\, \mathbf{True} \,|\!)\ \mathit{ProgB}\ (\!|\, y = x + 1 \,|\!)$ where *ProgB* is

$a := x + 1;$

$\mathbf{if}\ (a - 1) = 0$
$\{$
$\quad y := 1;$

$\}$
$\mathbf{else}$
$\{$
$\quad y := a;$

$\}$

- Here's the finished result. Make sure you make enough notes during the lecture so that you know how I arrived at this result.

$$
\begin{array}{l}
\llparenthesis\, \mathbf{True} \,\rrparenthesis \\
\llparenthesis\, ((x+1-1)=0 \Rightarrow 1 = x+1) \wedge \\
\qquad\qquad\qquad ((x+1-1) \neq 0 \Rightarrow x+1 = x+1) \,\rrparenthesis \text{Consequence (proof ①)} \\
a := x+1; \\
\llparenthesis\, ((a-1)=0 \Rightarrow 1 = x+1) \wedge ((a-1) \neq 0 \Rightarrow a = x+1) \,\rrparenthesis \text{Assignment} \\
\mathbf{if}\ (a-1)=0 \\
\{ \quad \llparenthesis\, 1 = x+1 \,\rrparenthesis \\
\qquad y := 1; \\
\qquad \llparenthesis\, y = x+1 \,\rrparenthesis \text{Assignment} \\
\} \\
\mathbf{else} \\
\{ \quad \llparenthesis\, a = x+1 \,\rrparenthesis \\
\qquad y := a; \\
\qquad \llparenthesis\, y = x+1 \,\rrparenthesis \text{Assignment} \\
\} \\
\llparenthesis\, y = x+1 \,\rrparenthesis \text{Two-armed-conditional}
\end{array}
$$

Proof ①: To show $\mathbf{True} \Rightarrow [((x+1-1)=0 \Rightarrow 1 = x+1) \wedge ((x+1-1) \neq 0 \Rightarrow x+1 = x+1)]$.

By arithmetic, $((x+1-1)=0 \Rightarrow 1 = x+1) \wedge ((x+1-1) \neq 0 \Rightarrow x+1 = x+1)$ simplifies to $(x=0 \Rightarrow x=0) \wedge (x \neq 0 \Rightarrow x+1 = x+1)$. $x=0 \Rightarrow x=0 \equiv \mathbf{True}$. $x+1 = x+1 \equiv \mathbf{True}$ by arithmetic. So we have $x \neq 0 \Rightarrow \mathbf{True} \equiv \mathbf{True}$. So this gives us $\mathbf{True} \wedge \mathbf{True} \equiv \mathbf{True}$.

Recap
Conditionals
Module Home Page
Title Page
Page 6 of 8
Back
Full Screen
Close
Quit

- Prove that $\vdash_{\text{par}} (\!|\, \mathbf{True} \,|\!)\ \mathit{ProgC}\ (\!|\, z = \min(x, y) \,|\!)$ where $\mathit{ProgC}$ is

**if** $x \geq y$
$\{$
    $z := y;$
$\}$
**else**
$\{$
    $z := x;$
$\}$

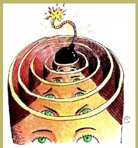
When you use the rule of consequence, remember you are stepping outside Floyd-Hoare logic. In this example, we will use the following fact of arithmetic:

$$a = \min(b, c) \equiv (a = b \vee a = c) \wedge a \leq b \wedge a \leq c$$

- One-armed-conditional

$$\frac{(\!|\, B \wedge P \,|\!)\ C\ (\!|\, Q \,|\!), \quad (\neg B \wedge P) \Rightarrow Q}{(\!|\, P \,|\!)\ \mathbf{if}\ B\ C\ (\!|\, Q \,|\!)}$$

If $B$ is **true**, $C$ is executed; if $B$ is **false**, the conditional does not execute any additional command. If we have proved that $C$ takes us from states satisfying $B \wedge P$





to states satisfying $Q$, and if we know that $Q$ follows directly in the other case, i.e. $(\neg B \wedge P) \Rightarrow Q$, then we can conclude that the conditional command as a whole takes us from states satisfying $P$ to states satisfying $Q$.

- How do we push a condition $Q$ 'backwards' up through a one-armed conditional?
  1. Push $Q$ up through $C$. Call the result $P'$.
  2. Then the precondition of the conditional $P$ is $(B \Rightarrow P') \wedge (\neg B \Rightarrow Q)$
- Here's an example. Below is *ProgD* and a proof that $\vdash_{\text{par}} \llparenthesis\, \mathbf{True} \,\rrparenthesis\ \mathit{ProgD}\ \llparenthesis\, x \geq 0 \,\rrparenthesis$

$$
\begin{array}{l}
\llparenthesis\, \mathbf{True} \,\rrparenthesis \\
\llparenthesis\, (x < 0 \Rightarrow -x \geq 0) \wedge (x \not< 0 \Rightarrow x \geq 0) \,\rrparenthesis \text{Consequence (proof ①)} \\
\mathbf{if}\ x < 0 \\
\{ \quad \llparenthesis\, -x \geq 0 \,\rrparenthesis \\
\qquad x := -x; \\
\qquad \llparenthesis\, x \geq 0 \,\rrparenthesis \text{Assignment} \\
\} \\
\llparenthesis\, x \geq 0 \,\rrparenthesis \text{One-armed conditional}
\end{array}
$$

Proof ①: To show that $\mathbf{True} \Rightarrow ((x < 0 \Rightarrow -x \geq 0) \wedge (x \not< 0 \Rightarrow x \geq 0))$.

From definitions of $<$ and $\geq$, $x < 0 \Rightarrow -x \geq 0 \equiv \mathbf{True}$. And $x \not< 0 \Rightarrow x \geq 0 \equiv \mathbf{True}$. So $(x < 0 \Rightarrow -x \geq 0) \wedge (x \not< 0 \Rightarrow x \geq 0) \equiv \mathbf{True}$. This leaves us with $\mathbf{True} \Rightarrow \mathbf{True} \equiv \mathbf{True}$.


## Acknowledgements

I continue to base material on that in Chapter 4 of [HR00].

Clip Art (of head with bomb) licensed from the Clip Art Gallery on DiscoverySchool.com.

Recap
Conditionals
Module Home Page
Title Page
Page 8 of 8
Back
Full Screen
Close
Quit

# References

[HR00] M. Huth and M. Ryan. *Logic in Computer Science: Modelling and Reasoning about Systems*. Cambridge University Press, 2000.