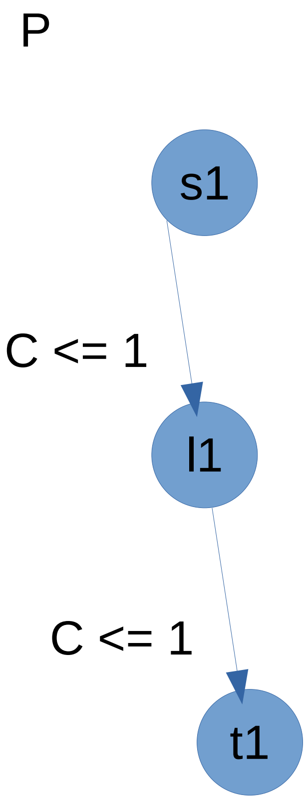
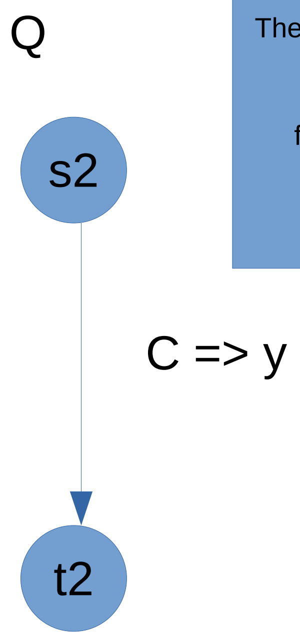
These slides illustrate some limitations of the compositional Method. Here we can prove that False is true in the final state, but that's ok: this program won't terminate anyway.



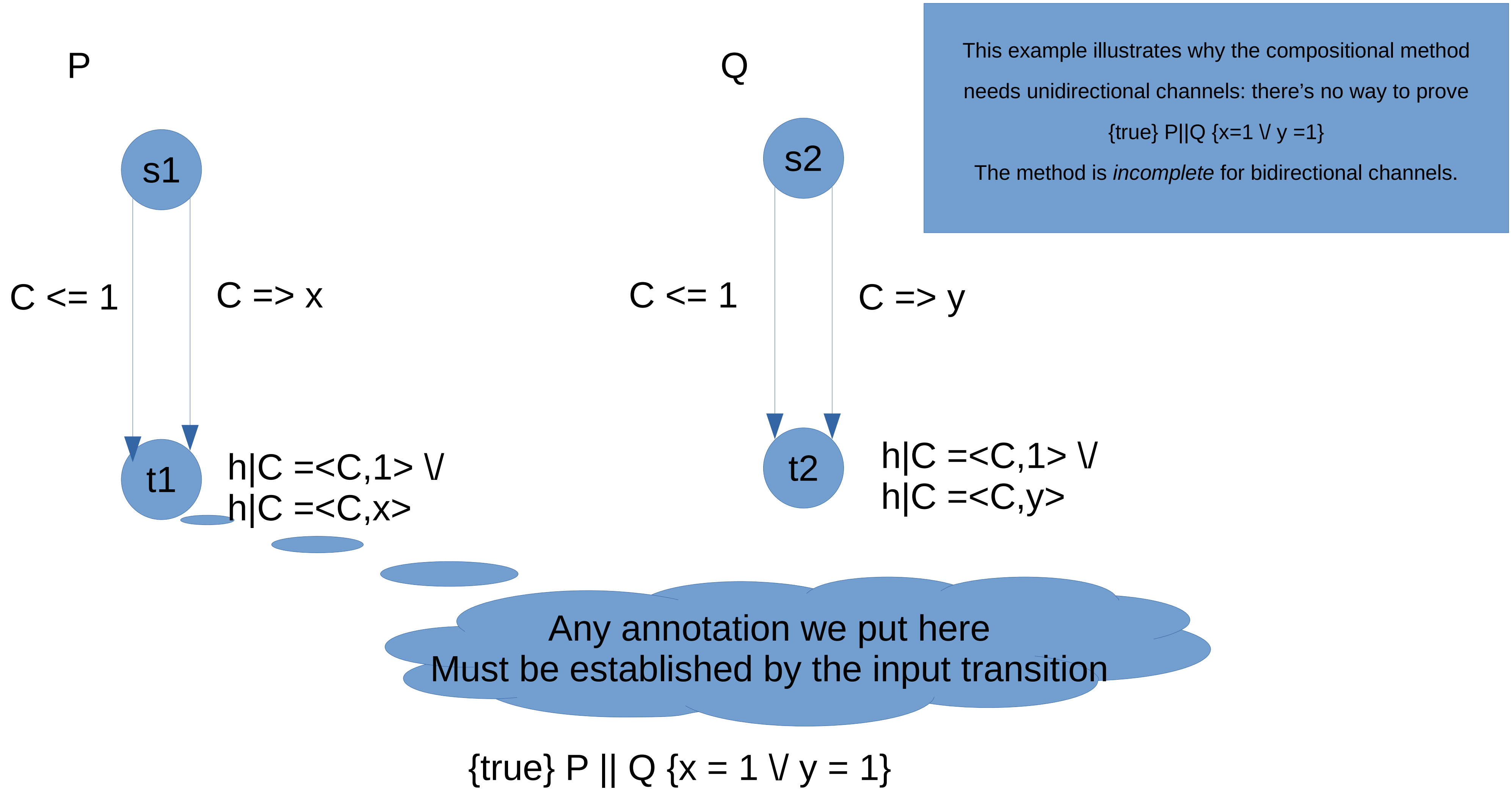


{h|C=[]} Q {h|C = <C,y>}

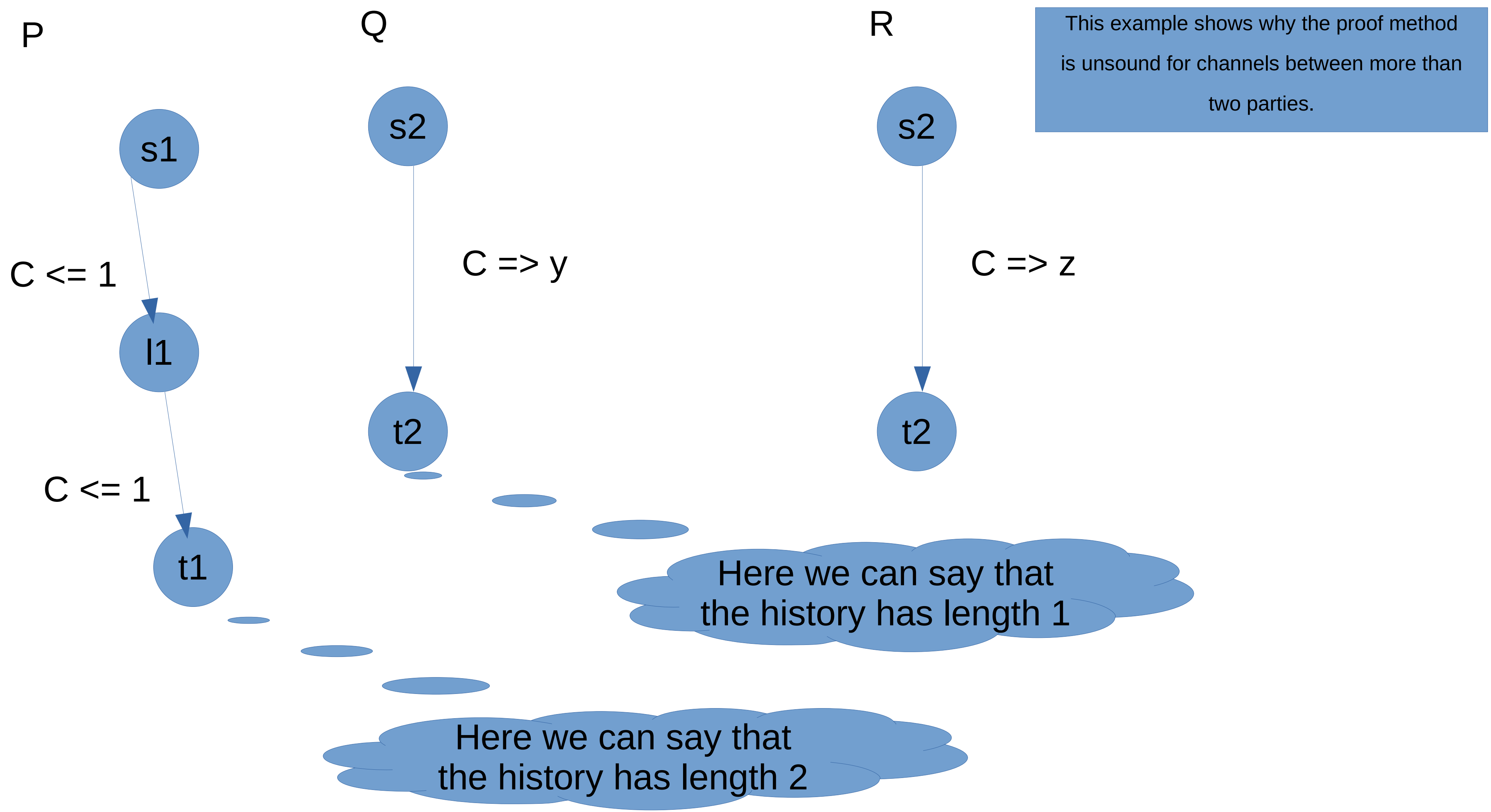
{h|C=[]} P {h|C = <C,1>,<C,1>}

{h|C=[]} P||Q {h|C = <C,1>,<C,1> /\ h|C = <C,y>}

{h|C=[]} P||Q {false}



P
s1
C <= 1
C => x
t1
h|C =<C,1> \/
h|C =<C,x>
Q
s2
C <= 1
C => y
t2
h|C =<C,1> \/
h|C =<C,y>
This example illustrates why the compositional method
needs unidirectional channels: there’s no way to prove
{true} P||Q {x=1 \/ y =1}
The method is incomplete for bidirectional channels.
Any annotation we put here
Must be established by the input transition
{true} P || Q {x = 1 \/ y = 1}



P
Q
R
This example shows why the proof method is unsound for channels between more than two parties.
s1
C <= 1
l1
C <= 1
t1
s2
C => y
t2
s2
C => z
t2
Here we can say that the history has length 1
Here we can say that the history has length 2