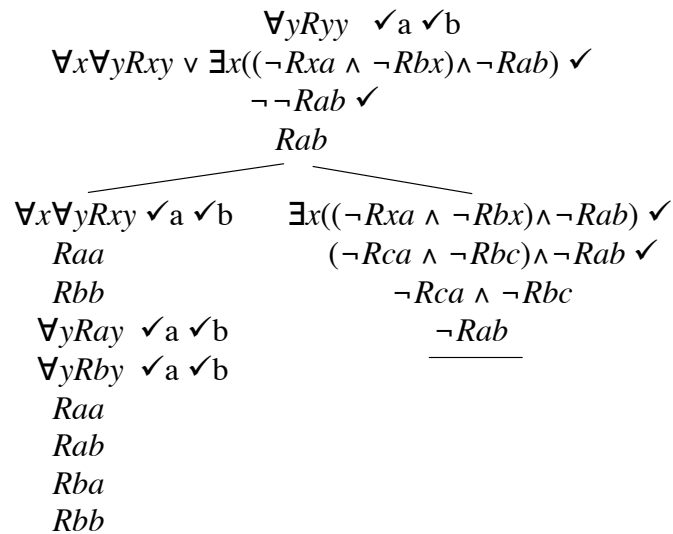




**Recap: Countermodels for Many-Place Predicates**

Is the following argument valid? If not, provide a counterexample.

$\forall yRyy, \forall x\forall yRxy \vee \exists x((\neg Rxa \wedge \neg Rbx) \wedge \neg Rab) \vdash \neg Rab$



A branch stays open. The argument is invalid.

Counterexample:

$$D = \{a, b\}$$

$$I(R) = \{ \langle a, a \rangle, \langle a, b \rangle, \langle b, a \rangle, \langle b, b \rangle \}$$

**Recap: “Pulling Out” Quantifiers**

Some boy loves every girl

$$\exists x(Bx \wedge \forall y(Gy \rightarrow Lxy))$$

or:  $\exists x\forall y(Bx \wedge (Gy \rightarrow Lxy))$  ?

Every boy loves some girl

$$\forall x(Bx \rightarrow \exists y(Gy \wedge Lxy))$$

or:  $\forall x\exists y(Bx \rightarrow (Gy \wedge Lxy))$  ?

Actually, both are fine in both cases. Convince yourself by “translating” back into English, and by testing with a tree whether the two formulae are equivalent in both cases.

**But careful!**

“Pulling out” quantifiers does not always work.

For example:

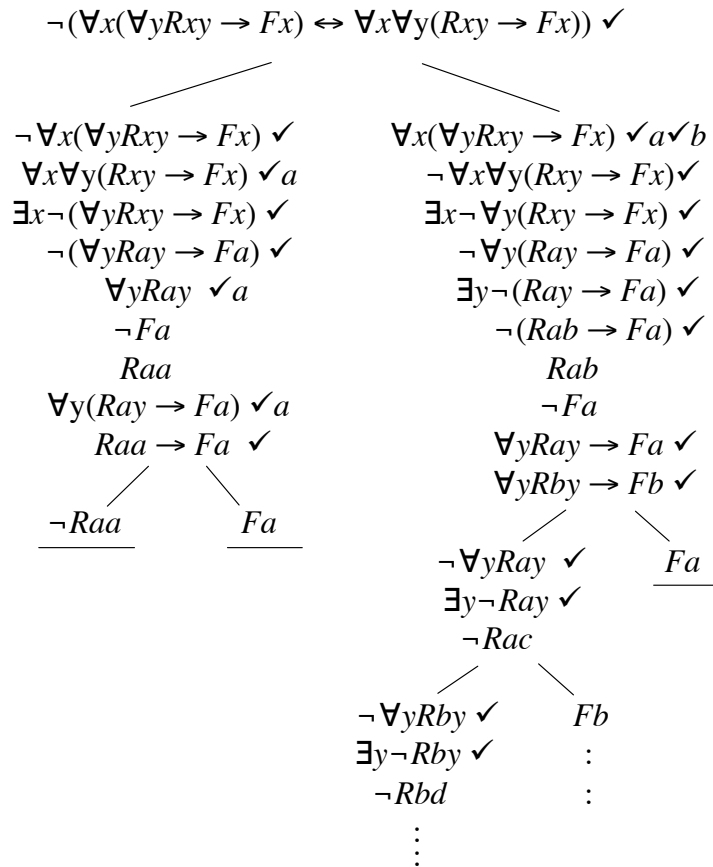
$$\forall x(\forall yRxy \rightarrow Fx)$$

is not equivalent with:

$$\forall x\forall y(Rxy \rightarrow Fx)$$

A tree shows that the equivalence

' $\forall x(\forall yRxy \rightarrow Fx) \leftrightarrow \forall x\forall y(Rxy \rightarrow Fx)$ ' does not hold:



The infinite branch will remain open.

### Countermodels for Infinite Trees

If a tree remains open and the counterexample can only be read off an infinite branch we face a problem: just as we cannot write down the infinite path, we also cannot write down an infinite model. We can help ourselves to the same “solution” that we used for the trees, however, and employ *the dots*, ‘...’, to indicate infinity.

The countermodel for the tree above, hence, becomes:

$$D = \{a, b, c, d, e, \dots\}$$

$$I(F) = \emptyset$$

$$I(R) = \{<a, b>\}$$

The countermodel is read off the leftmost branch, constructed as it would continue – no more not negated sentence will occur on it.

But also the interpretation of a predicate can be infinite:

*Example:*

Is the following inference valid? If not, provide a counterexample:

$$\forall x\exists yRxy \vdash Rab$$

$$\begin{array}{l} \forall x \exists y Rxy \checkmark a \checkmark b \checkmark c \checkmark d \\ \neg Rab \\ \exists y Ray \checkmark \\ \exists y Rby \checkmark \\ \quad Rac \\ \quad Rbd \\ \exists y Rcy \checkmark \\ \exists y Rdy \checkmark \\ \quad Rce \\ \quad Rdf \\ \quad \vdots \\ \quad \vdots \end{array}$$

The branch does not close. The inference is invalid.

Counterexample:

$$D = \{a, b, c, d, e, f, g, h, \dots\}$$

$$I(R) = \{\langle a, c \rangle, \langle b, d \rangle, \langle c, e \rangle, \langle d, f \rangle, \langle e, g \rangle, \langle f, h \rangle, \dots\}$$

*Exercise:*

Is the following inference valid? If not, provide a counterexample:

$$\forall x Fx \vdash \exists y \forall x (Fx \wedge Ryb)$$

## Identity

There is another class of apparently valid arguments that we cannot capture yet. We would like to be able to represent arguments like the following:

Cicero is a Roman orator.

Cicero is identical to Tully.

---

Therefore, Tully is a Roman orator.

We thus introduce the identity sign '=' as a special two-place predicate which will also get its own tree rules. 'Cicero is identical to Tully', or short, 'Cicero is Tully', is formalised (with 'c' for 'Cicero' and 't' for 'Tully') as ' $c = t$ '. We do not need to include '=' in our translation keys, since it will *always* be interpreted as identity; it is treated as a *logical constant*, like the quantifiers and sentential operators.

The argument above is thus formalised:

Key:  $c$  : Cicero

$T$  : Tully

$Rx$  :  $x$  is a Roman orator

$$Rc, c = t \vdash Rt$$

**Note:** It is important to keep the is of identity and the is of predication apart! Only use '=' if the sentence will also make sense when you replace the 'is' by 'is identical to' in the sentence. The identity sign can *only* have names on each side of it, never predicates.

'Lewis Carroll is Charles Dodgson' is an identity statement: it says that Lewis Carroll is identical to Charles Dodgson. "They" are the same person. We formalise it as ' $l = c$ ', with ' $l$ ' for Lewis Carroll and ' $c$ ' for C. Dodgson.

'Lewis Carroll is a philosopher', however, is *not* an identity statement. It is properly formalised as ' $Pl$ ', with ' $Px$ ' for ' $x$  is a philosopher'.

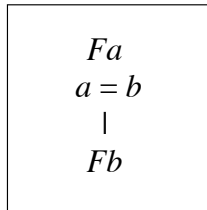
### Tree Rules for Identity

There are two intuitive rules for identity. The first one is motivated by examples like the Cicero case:  $Rc, c = t \vdash Rt$ .

#### First rule for '=':

For any constants  $a$  and  $b$ , if a sentence of the form ' $a = b$ ' appears on a branch of a tree, we may substitute ' $a$ ' for ' $b$ ' (and *vice versa*) in any sentence that occurs *on that branch* (not on any other branch).

The identity statement is *never* ticked off.

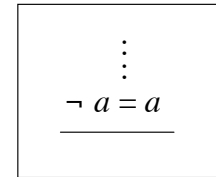


*Note:* You can also substitute into identity statements.

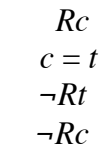
Obviously, everything is identical to itself, so a statement of the form ' $a = a$ ' is always true (for *any* constant, not just ' $a$ ', of course). This motivates the second rule for identity:

#### Second rule for '=':

For any constant ' $a$ ', if a statement of the form ' $\neg a = a$ ' occurs, close that branch.



*Example:* Tree for ' $Rc, c = t \vdash Rt$ '



*Examples:* Is the following inference valid?

$$a = b, b = c \vdash a = c$$

$$\begin{array}{l} a = b \\ b = c \\ \neg a = c \\ \neg b = c \\ \hline \end{array}$$

Does the following statement hold?

$$\vdash \forall x \forall y (x = y \rightarrow y = x)$$

$$\neg \forall x \forall y (x = y \rightarrow y = x) \checkmark$$

$$\exists x \neg \forall y (x = y \rightarrow y = x) \checkmark$$

$$\neg \forall y (a = y \rightarrow y = a) \checkmark$$

$$\exists y \neg (a = y \rightarrow y = a) \checkmark$$

$$\neg (a = b \rightarrow b = a) \checkmark$$

$$a = b$$

$$\neg b = a$$

$$\neg b = b \\ \hline$$

*Note:* We will *not* introduce how counterexamples are created for arguments involving identity. That does not mean, however, that any argument with identity in the exam will be valid! It only means that if it is invalid, you will not be asked to provide a counterexample.