Since the evaluation of quantified formulas usually requires the evaluation of the formula for all possible elements of the universe, truth tables are unsuited for proving first-order formulas correct. Universes are usually infinite and even in a finite universe, the search space would quickly explode.

The extension of the tableaux method to first-order logic, on the other hand, is quite straightforward. Let us consider an example

$$\begin{tikzcd}
F((\forall x)(Px\supset Qx)\supset((\forall x)Px\supset(\forall x)Qx)) \arrow[d, dash] \\
T((\forall x)(Px\supset Qx)) \arrow[d, dash] \\
F((\forall x)Px\supset(\forall x)Qx) \arrow[d, dash] \\
T((\forall x)Px) \arrow[d, dash] \\
F((\forall x)Qx)
\end{tikzcd}$$

Up to this point we could proceed as in propositional logic. Now we have to begin decomposing quantifiers. The formula $(\forall x)Qx$ is false if Qx can be made false for at least one element k of the universe. Since the elements of the universe do not belong to the syntax of the formulas, we have to substitute x by a parameter a instead.

In the following step we decompose $T((\forall x)Px)$. We know that $(\forall x)Px$ is true, if Px is true for all elements of the universe. This means we can substitute every parameter for x and we choose a again, since this is necessary to complete the proof.

The rest of the proof is now straightforward and we get

$$\begin{tikzcd}
 & F((\forall x)(Px\supset Qx)\supset((\forall x)Px\supset(\forall x)Qx)) \arrow[d, dash] & \\
 & T((\forall x)(Px\supset Qx)) \arrow[d, dash] & \\
 & F((\forall x)Px\supset(\forall x)Qx) \arrow[d, dash] & \\
 & T((\forall x)Px) \arrow[d, dash] & \\
 & F((\forall x)Qx) \arrow[d, dash] & \\
 & F(Qa) \arrow[d, dash] & \\
 & T(Pa) \arrow[d, dash] & \\
 & T(Pa\supset Qa) \arrow[dl, dash] \arrow[dr, dash] & \\
F(Pa) \arrow[d, dash] & & T(Qa) \arrow[d, dash] \\
\times & & \times
\end{tikzcd}$$

Why did we decompose $F((\forall\mathtt{x})\mathtt{Qx})$ before $T((\forall\mathtt{x})\mathtt{Px})$ in the proof?

The parameter $\mathtt{a}$ that we substituted for $\mathtt{x}$ was supposed to indicate that $\mathtt{Qx}$ can be made false by some yet unknown element of the universe. Since we do not know this element, $\mathtt{a}$ should be a *new parameter* – this way we make sure that we don't make any further assumptions about $\mathtt{a}$ by accidentally linking it to a parameter that was introduced earlier in the proof.

Now if we decompose $T((\forall\mathtt{x})\mathtt{Px})$ before $F((\forall\mathtt{x})\mathtt{Qx})$ then we cannot use $\mathtt{a}$ as parameter for $\mathtt{Q}$, since it has already been used for $\mathtt{P}$ and is not unknown anymore. If we decompose $F((\forall\mathtt{x})\mathtt{Qx})$ first, then $\mathtt{a}$ is still new at this stage. Choosing the same $\mathtt{a}$ for $\mathtt{P}$ is a decision we make afterwards.

In informal mathematics, quantifiers are handled in exactly the same way. When proving $(\forall x)(Px \wedge Qx) \supset (\forall x)Qx$ we assume $(\forall x)(Px \wedge Qx)$ and then try to show $(\forall x)Qx$. For this purpose we assume a to be arbitrary, but fixed, and try to prove Qa. Since we know $(\forall x)(Px \wedge Qx)$, we also know that $Pa \wedge Qa$ holds for the arbitrary a that we just chose and conclude that Qa is in fact the case. Note that it was crucial to have the a before instantiating $(\forall x)(Px \wedge Qx)$.

16.1 Extension of the Unified Notation

The above example shows that there are two different ways to handle quantifiers in tableaux proofs.

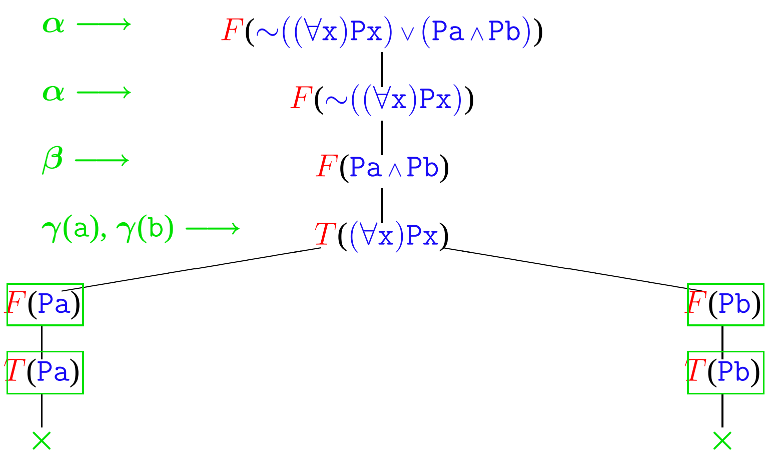
In the one case, we have formulas of the form $T((\forall x)A)$ and, by duality, $F((\exists x)A)$, which we call formulas of type $\boldsymbol{\gamma}$ of of *universal type*. γ-formulas are decomposed into $T(A[a/x])$ (and $F(A[a/x])$, respectively), where a is an arbitrary parameter. These formulas are often denoted by $\gamma(a)$.

In the other case, we have formulas of the form $F((\forall x)A)$ and, by duality, $T((\exists x)A)$, which we call formulas of type $\boldsymbol{\delta}$ of of *existential type*. δ-formulas are decomposed into $F(A[a/x])$ (and $T(A[a/x])$, respectively), where a is a new parameter. These formulas are often denoted by $\delta(a)$ and the requirement that a must be new is usually called the *proviso* of the rule.

Altogether we have now four types of inference rules.

$$\boxed{\dfrac{\boldsymbol{\alpha}}{\begin{array}{c}\boldsymbol{\alpha_1}\\ \boldsymbol{\alpha_2}\end{array}}} \qquad \boxed{\dfrac{\boldsymbol{\beta}}{\boldsymbol{\beta_1} \mid \boldsymbol{\beta_2}}} \qquad \boxed{\dfrac{\boldsymbol{\gamma}}{\begin{array}{c}\boldsymbol{\gamma(a)}\\ \boldsymbol{a}\text{ arbitrary parameter}\end{array}}} \qquad \boxed{\dfrac{\boldsymbol{\delta}}{\begin{array}{c}\boldsymbol{\delta(a)}\\ \boldsymbol{a}\text{ new parameter}\end{array}}}$$

Here is another example proof



Note that in this proof, the γ-formula $T((\forall x)Px)$ had to be instantiated twice to complete the proof. In general, formulas of universal type may be used arbitrarily often in a proof and therefore validity in first-order logic is not decidable.[1]

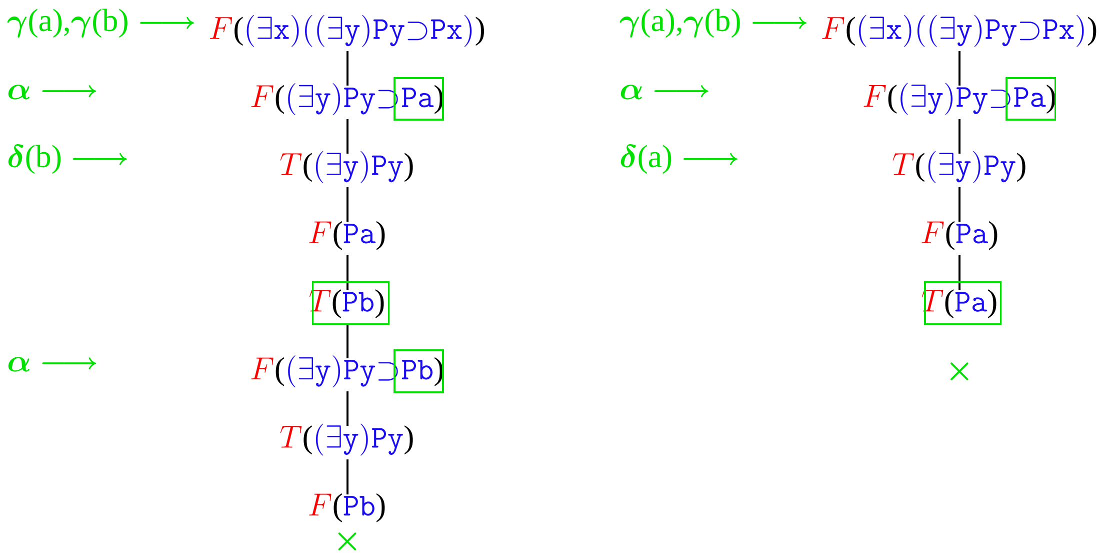
16.2 A liberalized δ rule

The proviso of the δ rule, which requires a to be a new parameter is quite restrictive and makes formal proofs more complicated than they have to be. It is, however, possible to liberalize the δ rule by replacing the proviso by the following requirement:

provided a is a new parameter
or a was not previously introduced on the same path by a δ rule, does not occur in δ,
and no parameter in δ was previously generated by a δ rule

In other words, if a does already occur in the proof then it was generated by some γ rule and that rule could have been applied later and still used the same parameter.

The following example shows the advantages of using a liberalized δ rule.



[1]This argument only appeals to the intuition. The actual proof of the undecidability of first-order logic is more complex, since one has to show that there is no other way to decide that a formula is not valid.