Proofs for first-order logic

Part 1 : tableaux

Derivation systems

- The purpose of derivation systems is to provide a **purely syntactic method of establishing entailment and validity**.
- Purely syntactic: we don’t have to mention the interpretation of any symbol to produce a proof.
- The simplest derivation systems for first-order logic were axiomatic.
    - A sequence of formulas counts as a derivation in such a system if each individual formula in it is either among a fixed set of “axioms” or follows from formulas coming before it in the sequence by one of a fixed number of “inference rules”.
- In Natural Deduction, there are only **derivation rules**.

Logical consequence

- Intuitively: something is satisfiable iff it can be true ; so a set of sentences is satisfiable iff there is a model in which all the sentences are true.

- A model M **satisfies a (possibly infinite) set of sentences** Γ iff every sentence in Γ is true in the model

    - Written: $M \vDash \Gamma$

- If there is such a model for a set of sentences, we say that **the set is satisfiable**.

- A sentence ψ is a **logical consequence** of a (possibly infinite) set of premises Γ iff there does not exist any model M such that M satisfies Γ, but M does not satisfy ψ.

    - Written: $\Gamma \vDash \psi$

- If there is a model M such that (i) M satisfies Γ but (ii) M does not satisfy ψ, it is called a **counter-model** for the argument form Γ, ψ.

Theorems and derivations

- If Γ is a set of formulas (the premises), ϕ is **derivable** from Γ, written $\Gamma \vdash \phi$, iff there is finite sequence of formulae from the premises to ϕ, where each step in the sequence is justified by a rule.
- ϕ is a **theorem**, written $\vdash \phi$, iff ϕ is derivable from ∅.
- In an ideal derivation system, we want:
  - **Soundness**: If $\vdash \phi$, $\vDash \phi$, and if $\Gamma \vdash \phi, \Gamma \vDash \phi$
  - **Completeness**: If $\vDash \phi$, $\vdash \phi$, and if $\Gamma \vDash \phi, \Gamma \vdash \phi$

Tableaux

- Tableaux operate with **signed formulas**: a signed formula is a pair consisting in a truth-value and a sentence.
  - Intuitively: $V : \exists x Px$ means that we assume $\exists x Px$ to be true, whereas $F : \neg p \vee q$ means that we assume $\neg p \vee q$ to be false.
- A tableau consists of **signed formulas arranged in a downward-branching tree**.
- In begins with a **set of assumptions**: we assume that all the premises of an argument form are true, and that its conclusion is false.
  - The idea is to **search for a counter-model**!
- We then use the tableaux rules to **analyse each of these assumptions**.
- Each rule allows us to add one or more signed formulas to the end of a branch (non-branching rules), or to add new branches with new signed formulas.
- A tableau is **closed** if every one of its branches contains a matching pair of signed formulas T A and F A.

Tableaux and counter-models

- Tableaux are very useful to search for counter-models.
- At the root of a tableau, we'll assume that in a (putative) counter-model all the premises are satisfied, but that the conclusion is falsified.
- If all the branches in the tableau close, we'll know that assuming the existence of such a counter-model leads to a contradiction ; if some branches are left open, we'll use those branches to construe counter-models.

Negation

$$\begin{array}{c} V : \neg\phi \\ | \\ F : \phi \end{array} \qquad\qquad \begin{array}{c} F : \neg\phi \\ | \\ V : \phi \end{array}$$

Conjunction

$V : \phi \land \psi$

$V : \phi$

$V : \psi$

$F : \phi \land \psi$

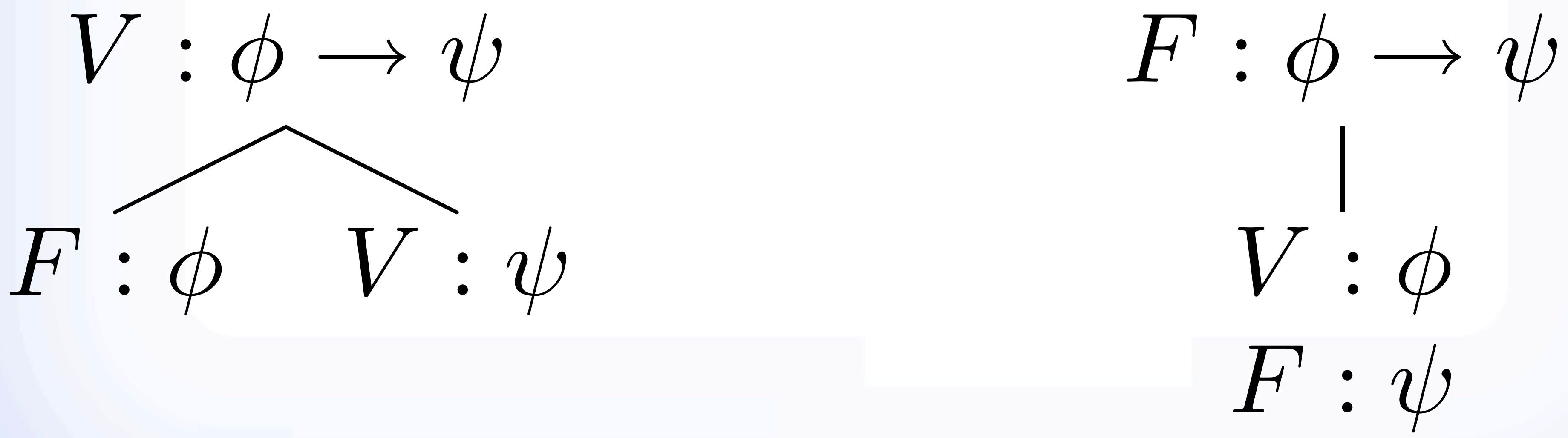
$F : \phi \qquad F : \psi$

Disjunction

$$\begin{array}{ccc} & V : \phi \vee \psi & \\ V : \phi & & V : \psi \end{array}$$

$$\begin{array}{c} F : \phi \vee \psi \\ | \\ F : \phi \\ F : \psi \end{array}$$

Conditional



$V : \phi \to \psi$

$F : \phi \quad V : \psi$

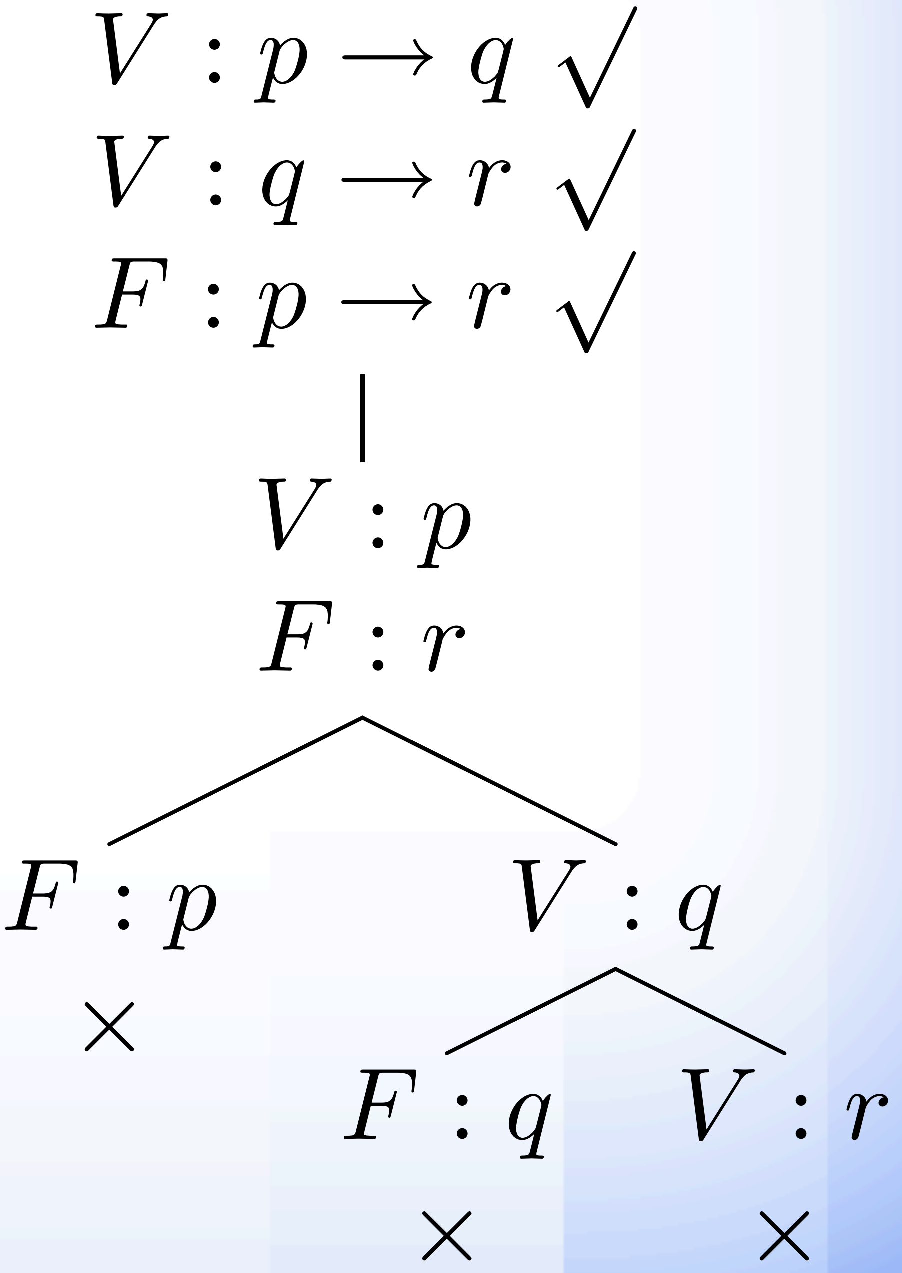
$F : \phi \to \psi$

$V : \phi$

$F : \psi$

Illustration

$$\begin{array}{c} p \longrightarrow q \\ q \longrightarrow r \\ \hline p \longrightarrow r \end{array}$$



$V : p \rightarrow q \; \surd$

$V : q \rightarrow r \; \surd$

$F : p \rightarrow r \; \surd$

$V : p$

$F : r$

$F : p$ $\quad$ $V : q$

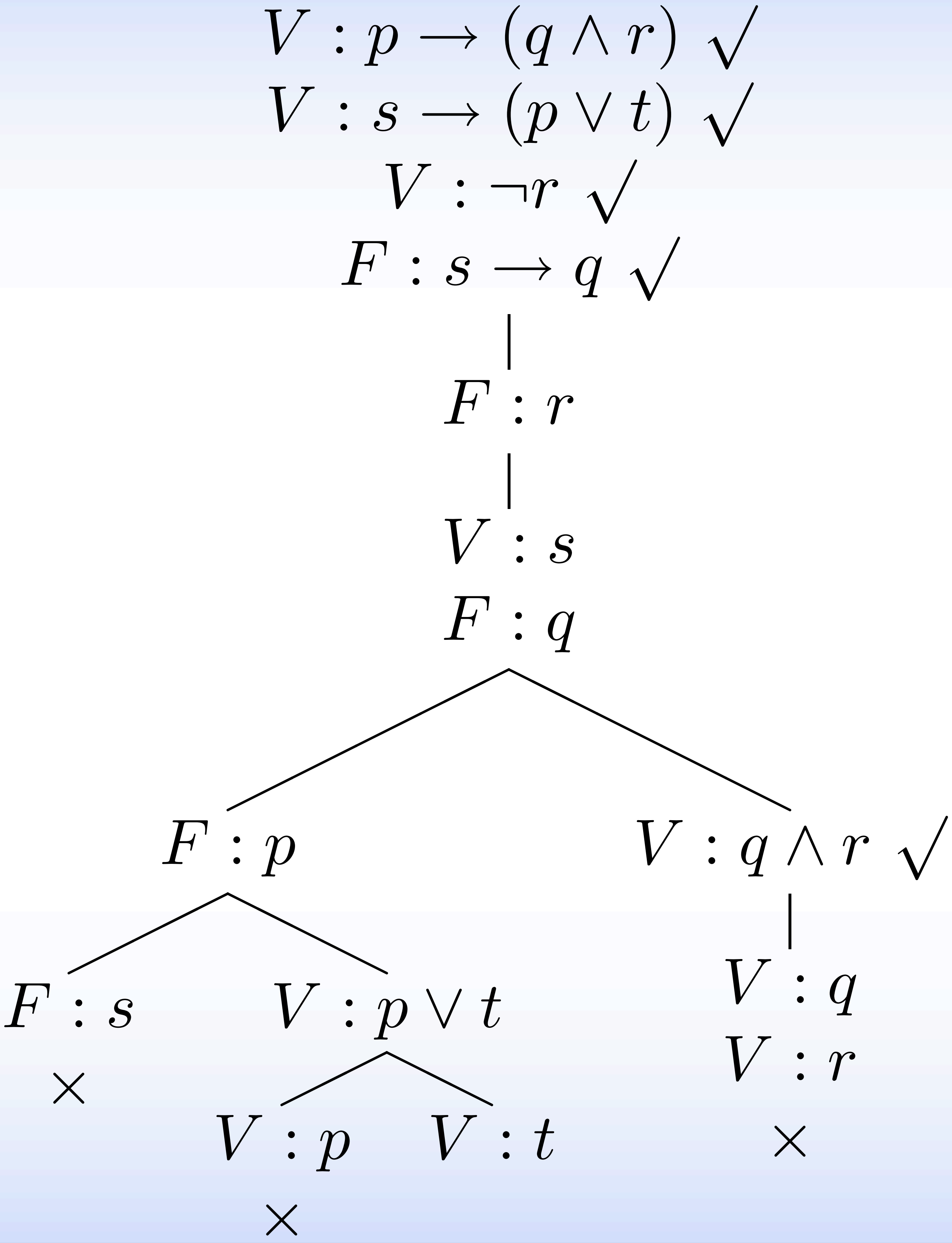
$\times$

$F : q$ $\quad$ $V : r$

$\times$ $\quad$ $\times$

Tableaux and counter-models

$$p \rightarrow (q \wedge r), s \rightarrow (p \vee t), \neg r \not\models s \rightarrow q$$



$V : p \rightarrow (q \wedge r)\ \surd$

$V : s \rightarrow (p \vee t)\ \surd$

$V : \neg r\ \surd$

$F : s \rightarrow q\ \surd$

$F : r$

$V : s$

$F : q$

$F : p$

$V : q \wedge r\ \surd$

$F : s$

$\times$

$V : p \vee t$

$V : p$

$\times$

$V : t$

$V : q$

$V : r$

$\times$

True existential formulas

- **Instance of a quantified sentence** :
  - If t is a closed term, we get an **instance** of $\exists x \phi x$ if we erase the quantifier, and replace all the bound occurrences of x by t ; we write the result ϕt
  - The instance used has to be **arbitrary**.
- The rule may be applied as many time as necessary

$T : \exists x \varphi x$

$|$

$T : \{\varphi t\}$

with t

<u>arbitrary</u>

- What does it mean to chose an arbitrary instance ?
    - The closed term used when instantiating the quantifier **should not occur on the branch connecting the root to the node where we copy the instance**.
- Let us consider the argument : Someone is thinking. Therefore, Macron is thinking. Intuitively it's not valid, so the form: $\exists x Px \nvDash Pa$ should not be valid.

T : $\exists x \; Px$

F : Pa

T : Pb

T : Pc

$\vdots$

The tableau does not close because $\exists x Px$ cannot be instantiated to Pa

False existential formulas

- The negation of an existential formula has a **universal import**: it means that ϕ is false for all individuals in the domain ;

- So here we can chose the instances, and we must instantiate the formula **to closed terms occurring on the branch connecting the root to the instance, if there are such terms.**
  - If there are no such terms: just chose any closed term for the instantiation.

$$F : \exists x \varphi x$$

$$|$$

$$F : \{\varphi(t)\}$$

Illustration

- The argument :
  - Nobody is walking. So Socrates is not walking.
  - Is valid!
- We show it by instantiating the formula to Pa, since « a » occurs on the branch.

$T : \neg \exists x\, Px$

$|$

$F : \exists x\, Px$

$F : \neg Pa$

$|$

$T : Pa$

$F : Pa$

$\times$

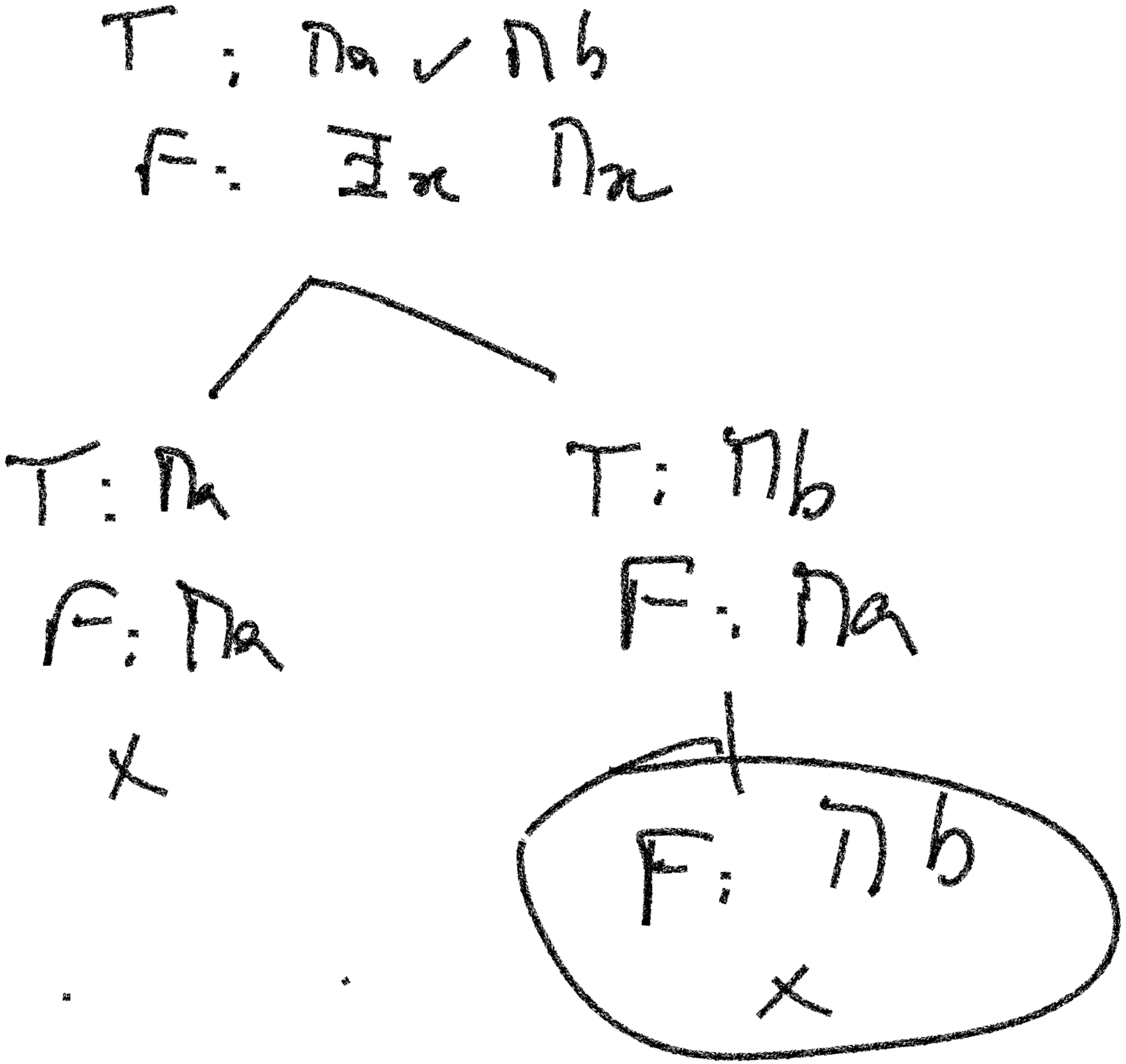
VALID

- Why is it crucial to instantiate the quantified formula to all closed terms occurring on a branch?

- Let us consider the argument : Socrates is walking, or Meno is walking. So someone is walking. The form is:

- $Ma \vee Mb \vDash \exists x Mx$

- Before looking at the tableau, let's think about the situation. In order to search for a counter-model, we suppose the conclusion is false, so we suppose that nobody is walking. But this contradicts the premise **only** if we assert **both** that Socrates is not walking **and** that Meno is not waling.


T : Πa ∨ Πb
F : ∃x Πx
T : Πa
F : Πa
×
T : Πb
F : Πa
F : Πb
×

True universal formulas

- Of course they have a **universal import**.
- So just like false existential, they should be instantiated to closed terms occurring on a branch, if there are such terms; if not: take any term for the instantiation.
- Illustration : $\forall x Mx \vDash Ma$

$$\begin{array}{c} T : \forall x \varphi x \\ | \\ T : \{\varphi(t)\} \end{array}$$

V: $\forall x\, Tx$

F: Ta

|

V: Ta

×

Valid!

False universal formulas

- Like true existential formulas, they have an existential import
- The term chosen for the instantiation should therefore be arbitrary: the instantiation term should not occur on the branch connection the root to the instantiation node.

$$F : \forall x\, \varphi x$$

$$F : \left[\, \varphi t \,\right]$$

- Let us consider : $\neg \forall x Px \nvDash \neg Pa$

- Let us first reason informally. Its is clearly not inconsistent to claim both that something is not P (premise), and that Pa (negation of the conclusion).

- Let us take $M = < D, I >$ with $D = \{\alpha, \beta\}, I(P) = \{\alpha\}$

- $M \vDash \neg \forall x Px$. $M \vDash \neg Pb$, since $\beta \notin I(P)$. But we also have $M \vDash Pa$. So we have found a counter-model to the argument form.

- Now let us search for such a counter-model by drawing the tableau:

$T : \neg \forall x \, Px$

$F : \neg Pa$

$|$

$T : Pa$

$F : \forall x \, Px$

$|$

$F : Pb$

$|$

$F : Pc$

$\vdots$

How to apply the rules?

1. First apply the rules for propositional logic ;
2. Then apply the rule that have an existential import, namely F : ∀ et V : ∃ ; introduce one new arbitrary term for each instantiation ;
3. Apply the rules that have a universal import, namely V : ∀ et F : ∃:

- If closed terms occur on the branch connecting the instantiation node to the root, instantiate to all those terms ;

- If there are no such terms, introduce any new closed term.

4. If some branch is still open, repeat until the tree loops to infinity (if it does).

Illustration 1

- $\exists x(Px \wedge Qx) \vDash \exists x Px$

$T : \exists x(Px \wedge Qx)$ ✓

$F : \exists x Px$ ✓

|

$T : Pa \wedge Qa$ ✓

|

$T : Pa$

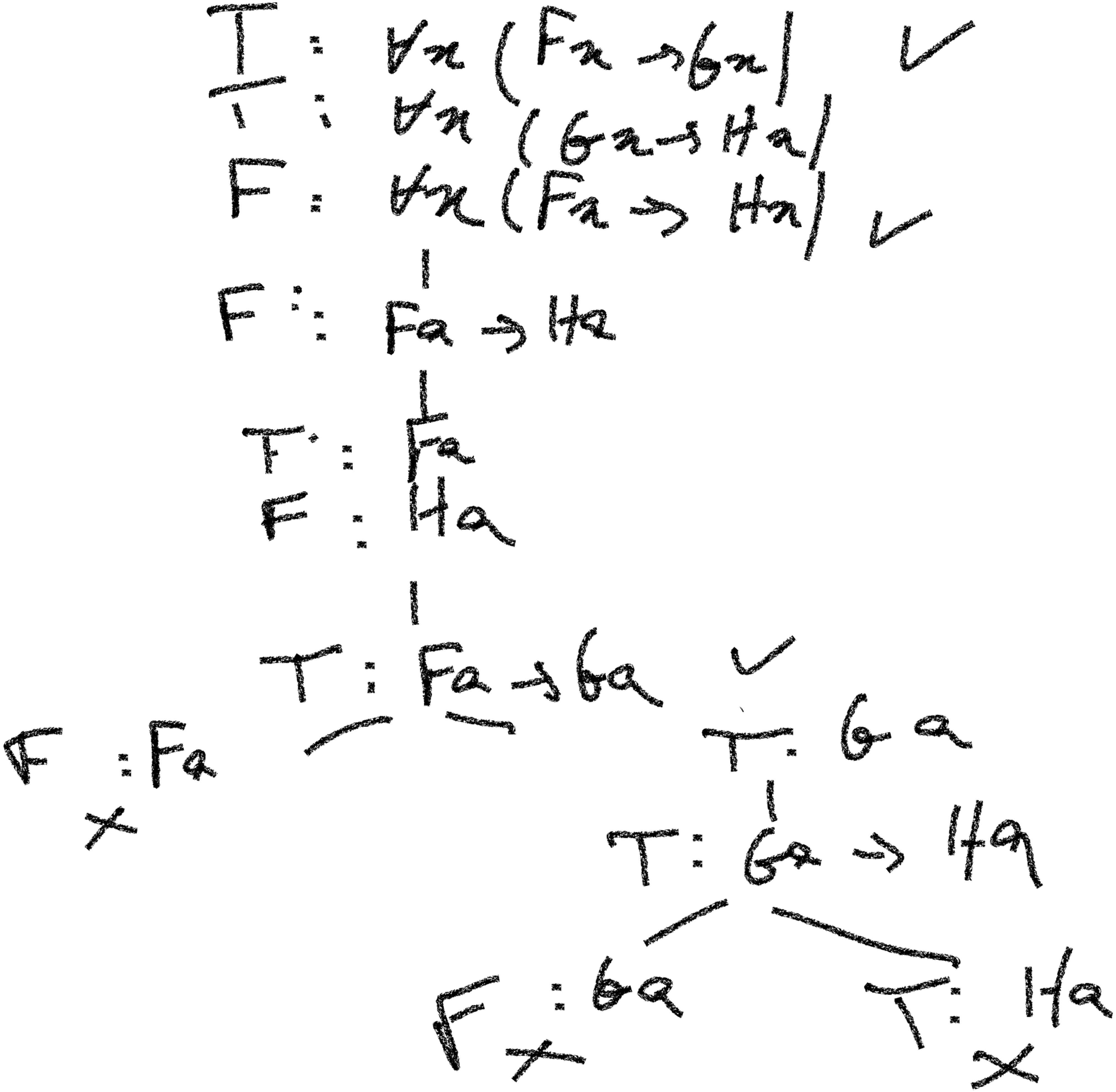
$T : Qa$

|

$F : Pa$

Illustration 2 :Barbara

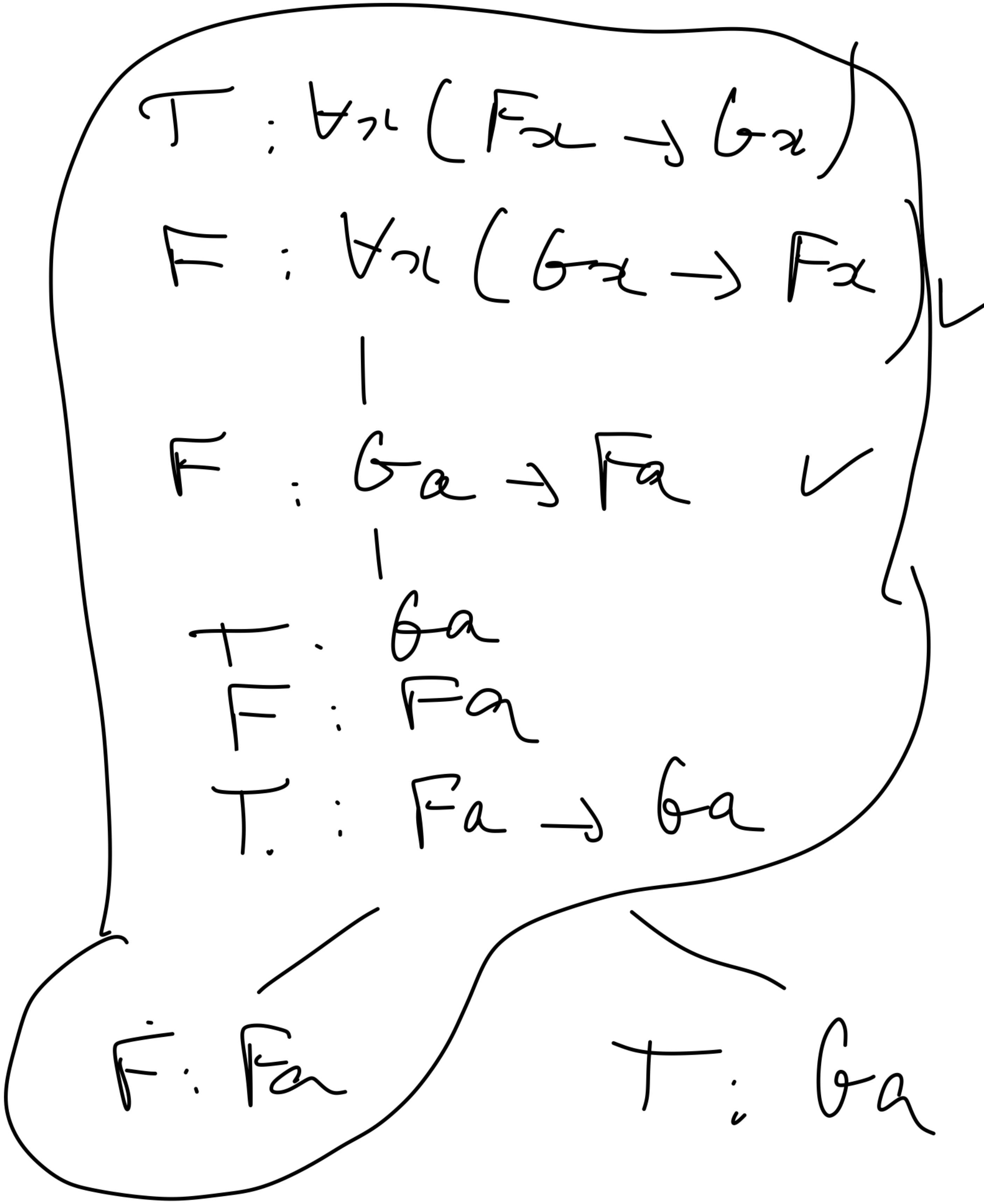
- $\forall x(Fx \to Gx), \forall x(Gx \to Hx) \vDash \forall x(Fx \to Hx)$


T : ∀x(Fx → Gx) ✓
T : ∀x(Gx → Hx)
F : ∀x(Fx → Hx) ✓
F : Fa → Ha
T : Fa
F : Ha
T : Fa → Ga ✓
F : Fa
×
T : Ga
T : Ga → Ha
F : Ga
×
T : Ha
×

Finding a counter-model

- Let us show that: $\forall x(Fx \rightarrow Gx) \nvDash \forall x(Gx \rightarrow Fx)$

- We need an object in the domain to interpret « a », let us say α.
- M = <D, I> tel que :
- $I(G) = \{\alpha\}, I(F) = \varnothing, D = \{\alpha\}$



Infinite trees

- In order to construct a completed tableau for an invalid argument, on has to check that **every necessary step has been applied**:
  - Each existential rule has been applied at least once;
  - Universal rules have been applied to instantiate the quantifiers to all the terms occurring in the relevant branches.
- For instance, let's suppose we want to show that $\nvdash \exists x \forall y Rxy$, and so $\nvDash \exists x \forall y Rxy$

F: $\exists x \forall y\, Rxy$

F: $\forall y\, Ray$

F: Rab

F: $\forall y\, Rby$

F: Rbc

F: $\forall y\, Rcy$

F: Rcd

$\vdots$

infinite tree!

- **Tableaux cannot serve as a decision procedure for FOL,** since a tree may cycle infinitely …

- … and then we cannot prove in a finite number of steps whether the argument form is valid.

- Note that we still can build a finite model, for instance with $D = \{\alpha\}, I$ sending every names to α, and $I(R) = \varnothing$

- If we rely on the tree and on the (usual) idea that each name occurring on the open branch should refer to a different object, the domain has to be infinite though, for instance: $D = \mathbb{N}, I(a) = 1, I(b) = 2, I(c) = 3...etc,$ and $I(R) = \varnothing$

Problem

- Let us consider identity statements between closed terms like $s = t$.
- Define rules for: $V : t = t$, $F : t = t$, and $V : s = t$; (for the last case, remember Leibniz's Law!)

Soundness with tableaux

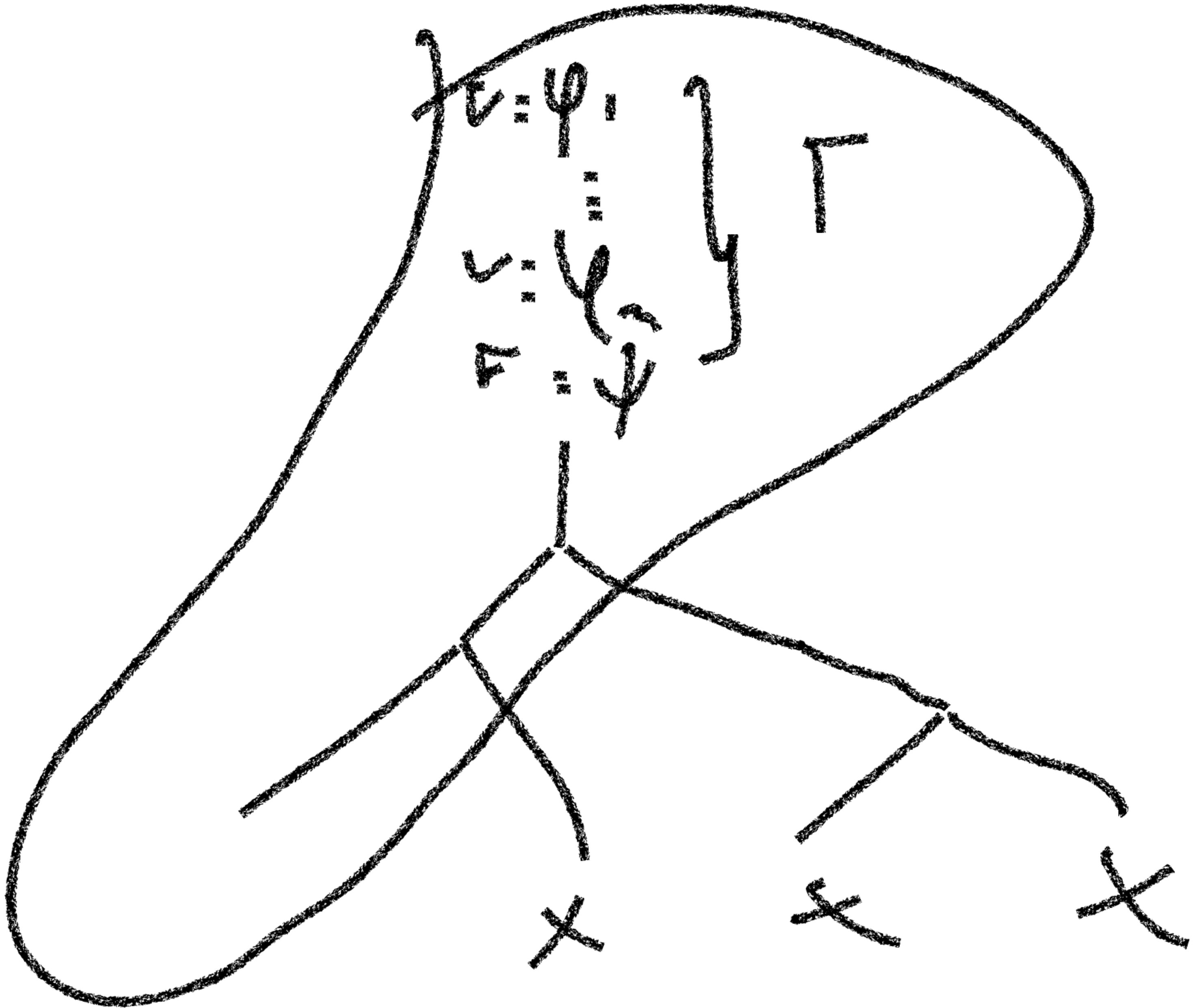
- **Soundness**: if $\Gamma \vdash \phi$, $\Gamma \vDash \phi$
- To find a proof, we could look at the contrapositive:
    - If $\Gamma \nvDash \phi$, then $\Gamma \nvdash \phi$
    - What does $\Gamma \nvDash \phi$ mean? It means that there is a counter-model to the argument form, so the set $\Gamma, \neg\phi$ is satisfiable.
    - What does $\Gamma \nvdash \phi$ in the tableaux system? It means that the tableau we build from $\Gamma, \neg\phi$ stays open when we apply the tableaux rules.
    - So if we prove that:
        - **If Γ is satisfiable, then the tree we build from Γ stays open when we apply the tableaux rules**
    - We will have proved the soundness theorem for tableaux

Proof

- Terminology: we'll say that **a tableau-branch is satisfiable** iff there is a model in which: all true formulas occurring on the branch are true, and all false formulas on the branch are false.

- We want to prove that: If Γ is satisfiable, then the tree we build from Γ stays open when we apply the tableaux rules

- This means that when we apply the rules an arbitrary number of times to a set of sentences, at least one branch in the tree remains open and so is satisfiable.
  - Indeed we know that: if a tableau closes, **no branch** is satisfiable; by taking the contrapositive: if **at least a branch** remains satisfiable, the tableau does not close



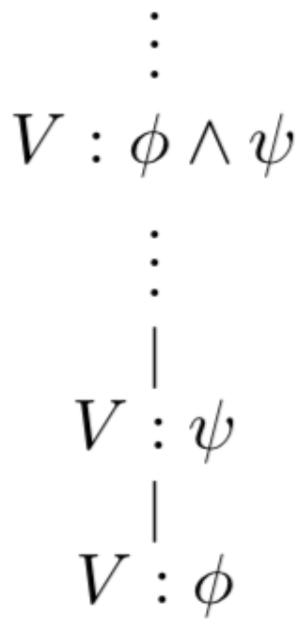
At least one branch has to be satisfiable!

What we should prove then:

- If a branch is satisfiable, and we apply a non-branching rule to it, the branch remains satisfiable.
- If a branch is satisfiable, and we apply a branching rule to it, at least one of the branches obtained after applying the rule is satisfiable.
- We should check that this is the case for all our rules.

Rule for true conjunctions

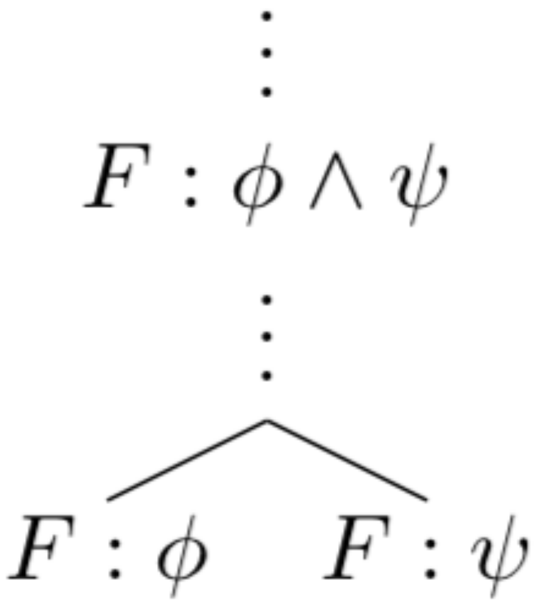
- Let us suppose Γ is satisfiable, that there is a true conjunction in Γ, and that we apply the rule for $V : \phi \wedge \psi$.
- We extend the tree thus:



- But now if Γ is satisfiable, since $V : \phi \wedge \psi$ occurs in Γ, there is a model M such that $M \vDash \phi \wedge \psi$, hence $M \vDash \phi$ and $M \vDash \psi$, which entails that the extended branch is satisfiable.

Rule for false conjunctions

- Now we suppose Γ is satisfiable, that there is a true conjunction in Γ, and that we apply the (branching) rule for $F : \phi \wedge \psi$.
- We extend the tree thus:



- Now if Γ is satisfiable, since $F : \phi \wedge \psi$ occurs in Γ, there is a model M such that $M \nvDash \phi \wedge \psi$, which entails that either $M \nvDash \phi$ or $M \nvDash \psi$.
  - First case: the branch on the left is satisfiable;
  - Second case: the branch on the right is satisfiable;
  - Whatever the case, at least one branch is satisfiable.

Problem

- Prove that when a tree is extended with $V : \phi \rightarrow \psi$ or with $F : \phi \rightarrow \psi$, it remains open.