Semantic Tableaux

A Complete Introduction to the Tableaux Method
for Propositional and First-Order Logic

Pascal Ludwig

Abstract

This document provides a comprehensive introduction to the semantic tableaux method (also known as the method of analytic tableaux or truth trees). We cover the method for propositional logic, extend it to first-order logic with quantifiers, and provide a complete proof of the soundness theorem.

Contents

I Propositional Logic **4**

1 Introduction **4**

2 Semantic Tableaux **4**
2.1 General Principle 4
2.2 Constructing the Tableau 4
2.3 Branches and Closure 4
2.4 Testing Validity 5

3 Rules for Propositional Logic **5**
3.1 Rules for Negation 5
3.2 Rules for Conjunction 5
3.3 Rules for Disjunction 6
3.4 Rules for the Conditional 6
3.5 Summary of Propositional Rules 6

4 Illustrations for Propositional Logic **6**
4.1 Example 1: Proof of Validity 6
4.2 Example 2: Construction of a Countermodel 7

5 Summary of the Method for Propositional Logic **9**

II First-Order Logic **10**

6 Introduction to First-Order Tableaux **10**

7 Rules for Quantifiers **10**
7.1 True Existential Formulas (T∃) . . . 10
7.1.1 Illustration: An Invalid Argument . . . 10
7.2 False Existential Formulas (F∃) . . . 11
7.2.1 Illustration: A Valid Argument . . . 11
7.2.2 Why Instantiate to All Closed Terms on a Branch? . . . 12
7.3 True Universal Formulas (T∀) . . . 12
7.3.1 Illustration: A Valid Argument . . . 12
7.4 False Universal Formulas (F∀) . . . 13
7.4.1 Illustration: An Invalid Argument . . . 13
7.5 Summary of Quantifier Rules . . . 14

8 Order of Rule Application **14**

9 Worked Examples for First-Order Logic **15**
9.1 Example 1: $\exists x(Px \wedge Qx) \models \exists x\, Px$ . . . 15
9.2 Example 2: The Barbara Syllogism . . . 16

10 Using Tableaux to Find Countermodels **17**
10.1 Example: $\forall x(Fx \rightarrow Gx) \not\models \forall x(Gx \rightarrow Fx)$ . . . 17

11 Infinite Tableaux **18**
11.1 Conditions for a Completed Tableau . . . 18
11.2 Example: $\nvdash \exists x \forall y\, Rxy$ . . . 18
11.3 Interpreting Infinite Tableaux . . . 19
11.4 Tableaux and Decidability . . . 19

12 Summary of First-Order Tableaux **19**

III Soundness of the Tableaux Method **21**

13 What Does Soundness Mean? **21**
13.1 The Soundness Theorem . . . 21
13.2 The Contrapositive Approach . . . 21
13.3 The Proof Strategy . . . 21

14 The Proof of Soundness **22**
14.1 Satisfiability of a Branch . . . 22
14.2 The Core Insight . . . 22
14.3 What We Need to Prove . . . 22

15 Verification for Propositional Rules **23**
15.1 Rule for True Negation (T¬) . . . 23
15.2 Rule for False Negation (F¬) . . . 23
15.3 Rule for True Conjunction (T∧) . . . 23
15.4 Rule for False Conjunction (F∧) . . . 24
15.5 Rule for True Disjunction (T∨) . . . 24
15.6 Rule for False Disjunction (F∨) . . . 24
15.7 Rule for True Conditional (T →) . . . 25
15.8 Rule for False Conditional (F →) . . . 25

16 Verification for Quantifier Rules **25**
16.1 Rule for True Existential (T∃) 25
16.2 Rule for False Existential (F∃) 26
16.3 Rule for True Universal (T∀) 26
16.4 Rule for False Universal (F∀) 27

17 Conclusion of the Proof **27**
17.1 Completing the Argument 27
17.2 Summary of the Soundness Proof 28

Part I
Propositional Logic

1 Introduction

We have shown how one can prove the validity of an argument form by proving that it is not possible to find a countermodel of that form. We will now present a formal proof method based on a systematization of this approach to validity: the **semantic tableaux method**, also known as the *semantic trees method*.

2 Semantic Tableaux

2.1 General Principle

The method consists in associating a semantic tableau (or tree) with any set of formulas written in a logical language. The formulas are, rigorously speaking, **signed formulas** (or equations) of the form:

$$\mathsf{T} : p \wedge q \tag{1}$$
$$\mathsf{F} : (p \vee q) \rightarrow r \tag{2}$$

Definition 2.1 (Satisfaction of a set of signed formulas)**.** We say that a **model** satisfies a set of such signed formulas if:

- all formulas preceded by "T" are true in the model,
- all formulas preceded by "F" are false in the model.

Thus, the signed formula (1) reads: "the formula $p \wedge q$ is true"; the signed formula (2) reads: "the formula $(p \vee q) \rightarrow r$ is false".

2.2 Constructing the Tableau

The tableaux we construct are **inverted trees**. We start from a set of signed formulas at the top of the tree (the *root*), and we decompose each formula by applying the rules.

The application of a rule to a given node **percolates** down the tree, meaning that we must go down all branches of the tree and write the result of applying the rule at the end of each branch.

Each rule can only be applied once to each formula in the tree. To keep track of this, it is helpful to mark formulas that have already been analyzed. Thus, the mark "✓" (or " | ") indicates that the formula has been analyzed:

$$\mathsf{F} : (p \vee q) \rightarrow r \ \checkmark$$

We must continue to extend the tree downward by analyzing the formulas obtained through rule application, until only unmarked atomic formulas remain.

2.3 Branches and Closure

Definition 2.2 (Branch)**.** A **branch** in a tableau is any sequence of nodes such that:

(i) it allows one to go from the root to a leaf (a "tip" of the tree, at the bottom),

(ii) without passing through the same node twice.

Once a tableau is fully constructed, we **close** all branches that contain a contradiction. A branch is closed using a cross: "$\times$".

Remark 2.1 (Strategy). A good time-saving strategy is to close branches containing a contradiction as soon as you detect it: there is no point in continuing to explore a closed branch, and therefore no point in extending it downward.

- If **all branches** of the tableau are closed, there is no model that satisfies the set of signed formulas placed at the root of the tree.
- If there remain **open branches**, these branches can be used to construct models that satisfy the signed formulas.

2.4 Testing Validity

Method for checking whether an argument form is valid
To check whether an argument form $\varphi_1, \ldots, \varphi_n \therefore \psi$ is valid:

1. Place at the root of a semantic tableau the following signed formulas:
   - all premises preceded by "T",
   - the conclusion preceded by "F".
2. Construct the complete semantic tableau for this set of signed formulas.
3. If there is no open branch in the tableau, this shows that no model can satisfy the signed formulas, hence that there is no model of the premises that is not a model of the conclusion. This suffices to prove the validity of the form.

3 Rules for Propositional Logic

There are two rules for each type of formula, depending on whether the formula is preceded by the value T or the value F. These rules correspond to the truth tables of the various connectives.

3.1 Rules for Negation

Rule $\mathsf{T}\neg$

$$\frac{\mathsf{T} : \neg\varphi}{\mathsf{F} : \varphi}$$

Rule $\mathsf{F}\neg$

$$\frac{\mathsf{F} : \neg\varphi}{\mathsf{T} : \varphi}$$

3.2 Rules for Conjunction

Rule $\mathsf{T}\wedge$

$$\frac{\mathsf{T} : \varphi \wedge \psi}{\begin{array}{c}\mathsf{T} : \varphi \\ \mathsf{T} : \psi\end{array}}$$

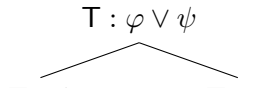
Rule $\mathsf{F}\wedge$

$$\begin{array}{c}\mathsf{F} : \varphi \wedge \psi \\ \hline \mathsf{F} : \psi \qquad\quad \mathsf{F} : \varphi\end{array}$$

The rule $\mathsf{T}\wedge$ is **non-branching**: both subformulas are true.
The rule $\mathsf{F}\wedge$ is **branching**: at least one of the two subformulas is false.

3.3 Rules for Disjunction

Rule T∨ **Rule F∨**

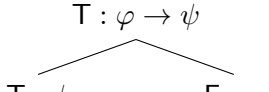


$$\frac{\mathsf{F} : \varphi \vee \psi}{\begin{array}{c}\mathsf{F} : \varphi \\ \mathsf{F} : \psi\end{array}}$$

The rule T∨ is **branching**: at least one of the two subformulas is true.
The rule F∨ is **non-branching**: both subformulas are false.

3.4 Rules for the Conditional

Rule T → **Rule F →**



$$\frac{\mathsf{F} : \varphi \to \psi}{\begin{array}{c}\mathsf{T} : \varphi \\ \mathsf{F} : \psi\end{array}}$$

The rule T → is **branching**: either the antecedent is false, or the consequent is true.
The rule F → is **non-branching**: the antecedent is true and the consequent is false.

3.5 Summary of Propositional Rules

Connective	Rule T	Rule F
¬	non-branching	non-branching
∧	non-branching	branching
∨	branching	non-branching
→	branching	non-branching

Remark 3.1 (Heuristic). It is good practice to always begin by applying rules that do not introduce branching, when possible: the resulting tableaux are generally smaller. However, the order of rule application does not matter for the final result in propositional logic.

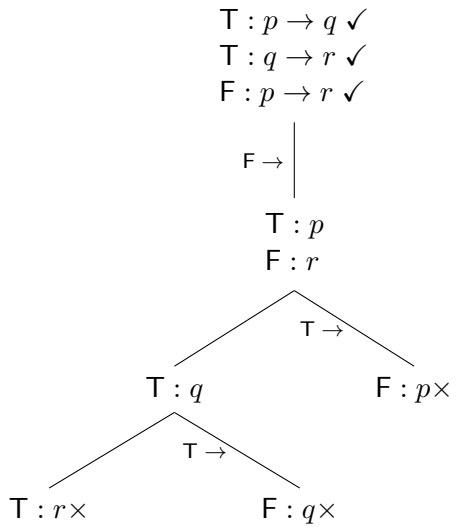
4 Illustrations for Propositional Logic

4.1 Example 1: Proof of Validity

Let us try to prove the validity of the following argument form (hypothetical syllogism):

$$\frac{p \to q \qquad q \to r}{p \to r}$$

To do this, we start with a set of signed formulas corresponding to the assertion of each premise and the negation of the conclusion:



A contradiction is found on each branch:

- Left branch: $\mathsf{T} : p$ and $\mathsf{F} : p$ (contradiction)
- Middle branch: $\mathsf{T} : q$ and $\mathsf{F} : q$ (contradiction)
- Right branch: $\mathsf{F} : r$ and $\mathsf{T} : r$ (contradiction)

All branches are closed, therefore the form is **valid**.

Remark 4.1. The first rule applied is $\mathsf{F} \to$ (applied to the conclusion): this is indeed a rule that does not introduce branching.

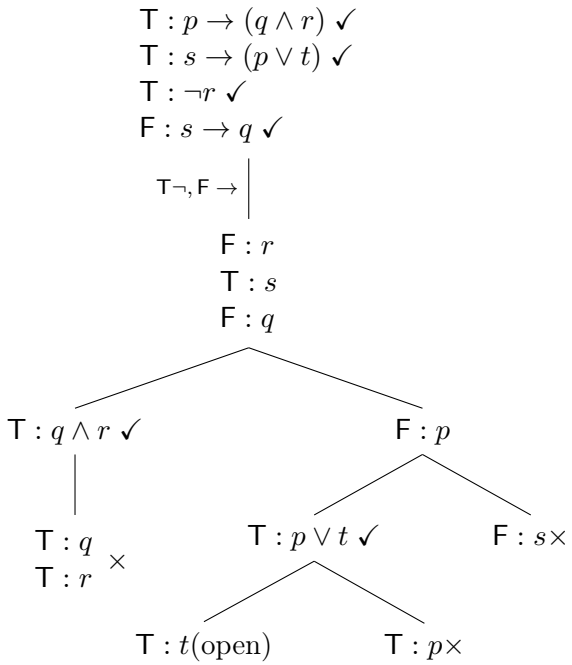
4.2 Example 2: Construction of a Countermodel

Let us now show how one can use the tableau of an argument form to find a countermodel, and thereby show that the form is not valid.

Consider the following argument form:

$$\frac{p \to (q \wedge r) \qquad s \to (p \vee t) \qquad \neg r}{s \to q}$$

Let us construct the tableau:



One branch remains **open** in this tableau. By tracing it back up, we can determine a countermodel for this argument form.

Reading the Countermodel

Following the open branch from the leaf to the root, we obtain the following signed formulas:

- $\mathsf{T} : t$
- $\mathsf{F} : p$ (from the left branch)
- $\mathsf{F} : q$
- $\mathsf{T} : s$
- $\mathsf{F} : r$

We can therefore define a truth assignment I such that:

$$I(p) = \mathsf{F}$$
$$I(q) = \mathsf{F}$$
$$I(r) = \mathsf{F}$$
$$I(s) = \mathsf{T}$$
$$I(t) = \mathsf{T}$$

Verification

Let us verify that, under this assignment, all premises are true and the conclusion is false:

- **Premise 1**: $p \to (q \wedge r)$
  $I(p) = \mathsf{F}$, so $p \to (q \wedge r)$ is **true** (a conditional with a false antecedent is true).
- **Premise 2**: $s \to (p \vee t)$
  $I(s) = \mathsf{T}$ and $I(p \vee t) = \mathsf{T}$ (since $I(t) = \mathsf{T}$), so $s \to (p \vee t)$ is **true**.

- **Premise 3**: $\neg r$

  $I(r) = \mathsf{F}$, so $\neg r$ is **true**.

- **Conclusion**: $s \to q$

  $I(s) = \mathsf{T}$ and $I(q) = \mathsf{F}$, so $s \to q$ is **false**.

The argument form is therefore **not valid**.

5 Summary of the Method for Propositional Logic

1. **Initialization**: Write at the root of the tableau:
   - $\mathsf{T} : \varphi_i$ for each premise φ_i
   - $\mathsf{F} : \psi$ for the conclusion ψ
2. **Development**: Apply the decomposition rules to non-atomic formulas, giving priority to non-branching rules.
3. **Closure**: Close ($\times$) any branch containing a contradiction, i.e., a pair $\mathsf{T} : \alpha$ and $\mathsf{F} : \alpha$ for the same atomic formula α.
4. **Conclusion**:
   - If all branches are closed $\Rightarrow$ the form is **valid**.
   - If at least one branch remains open $\Rightarrow$ the form is **invalid**, and the open branch provides a **countermodel**.

Part II
First-Order Logic

6 Introduction to First-Order Tableaux

The semantic tableaux method can be extended from propositional logic to first-order logic. The key addition is the treatment of **quantifiers**: the universal quantifier $\forall$ and the existential quantifier $\exists$.

When extending the method, we must be careful about how we **instantiate** quantified formulas. A quantified formula $\forall x\varphi(x)$ or $\exists x\varphi(x)$ can be instantiated by replacing the bound variable x with a **closed term** t (typically a constant like a, b, c, etc.), yielding the instance $\varphi(t)$.

The central insight is that quantifiers have different "imports":

- **Existential import**: The formula commits us to the existence of *at least one* witness. The instantiation term must be **arbitrary** (new, not previously used on the branch).
- **Universal import**: The formula applies to *all* individuals. We must instantiate to **all closed terms** occurring on the branch.

7 Rules for Quantifiers

7.1 True Existential Formulas (T$\exists$)

If we accept an existentially quantified formula as true, we are committed to recognizing that *at least one* instance of the formula is true.

Rule T$\exists$

$$\frac{\mathsf{T} : \exists x\, \varphi(x)}{\mathsf{T} : \varphi(t)}$$

where t is an **arbitrary** closed term that does **not** occur on the branch from the root to the instantiation node.

Remark 7.1 (Existential Import). True existential formulas have an **existential import**: they assert the existence of at least one witness. Because we don't know *which* individual satisfies the formula, we must choose an arbitrary (fresh) term for the instantiation. Using a term already present on the branch would be illegitimate, as it would amount to assuming something about a specific individual that we have no right to assume.

7.1.1 Illustration: An Invalid Argument

Consider the following argument:

> "Someone is thinking. Therefore, Macron is thinking."

Let T be the predicate "is thinking" and m the constant denoting Macron. The argument is formalized as:

$$\exists x\, Tx \models Tm \quad ?$$

Let us construct the tableau:

$$\begin{array}{l} \mathsf{T} : \exists x\, Tx \;\checkmark \\ \mathsf{F} : Tm \\ \quad\vert \\ \mathsf{T} : Ta \end{array}$$

The branch remains **open**. Why? Because of the restriction on rule $\mathsf{T}\exists$: we cannot instantiate to m, since m already occurs on the branch (in the conclusion). We must use an arbitrary fresh constant a.

Since $\mathsf{T} : Ta$ and $\mathsf{F} : Tm$ do not contradict each other, the tableau does not close.

Conclusion: The argument is **not valid**.

Remark 7.2. This corresponds to our intuition: just because *someone* is thinking, we cannot conclude that *Macron specifically* is thinking.

7.2 False Existential Formulas ($\mathsf{F}\exists$)

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

Rule $\mathsf{F}\exists$

$$\frac{\mathsf{F} : \exists x\, \varphi(x)}{\mathsf{F} : \varphi(t_1),\ \mathsf{F} : \varphi(t_2),\ \ldots}$$

where $t_1, t_2, \ldots$ are **all closed terms occurring on the branch** from the root to the instantiation node. If no such terms exist, introduce any fresh closed term.
This rule may be applied **multiple times** as new terms appear on the branch.

7.2.1 Illustration: A Valid Argument

Consider the argument:

> "Nobody is walking. Therefore, Socrates is not walking."

Let W be the predicate "is walking" and s the constant denoting Socrates. The argument is formalized as:

$$\neg\exists x\, Wx \models \neg Ws$$

Equivalently, using the duality $\neg\exists x\, Wx \equiv \forall x\, \neg Wx$:

$$\forall x\, \neg Wx \models \neg Ws$$

Let us construct the tableau:

$$\begin{array}{l} \mathsf{T} : \neg\exists x\, Wx\ \checkmark \\ \mathsf{F} : \neg Ws\ \checkmark \\ \quad\big| \\ \mathsf{F} : \exists x\, Wx\ \checkmark \\ \mathsf{T} : Ws \\ \quad\big| \\ \mathsf{F} : Ws\times \end{array}$$

We have $\mathsf{T} : Ws$ and $\mathsf{F} : Ws$ on the same branch: contradiction!

Conclusion: The argument is **valid**.

7.2.2 Why Instantiate to All Closed Terms on a Branch?

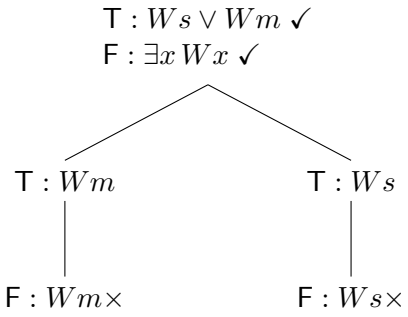
Consider another argument:

"Socrates is walking, or Meno is walking. Therefore, someone is walking."

Let W be the predicate "is walking", s denote Socrates, and m denote Meno. The argument is:

$$Ws \lor Wm \models \exists x\, Wx$$

Let us construct the tableau:



Both branches close! But notice: for the left branch, we instantiated $\mathsf{F} : \exists x\, Wx$ to s; for the right branch, we instantiated it to m. We needed to instantiate to **both** constants appearing on the respective branches.

If we had only instantiated to one constant (say s), the right branch would have remained open with $\mathsf{T} : Wm$ and $\mathsf{F} : Ws$ (no contradiction). This would incorrectly suggest the argument is invalid.

Conclusion: The argument is **valid**, but only if we instantiate universal-import rules to *all* relevant closed terms.

7.3 True Universal Formulas (T∀)

True universal formulas have a **universal import**: they assert that φ holds for *all* individuals.

Rule T∀

$$\frac{\mathsf{T} : \forall x\, \varphi(x)}{\mathsf{T} : \varphi(t_1),\ \mathsf{T} : \varphi(t_2),\ \ldots}$$

where $t_1, t_2, \ldots$ are **all closed terms occurring on the branch**. If no closed terms occur on the branch, introduce any fresh closed term.
This rule may be applied **multiple times** as new terms appear.

7.3.1 Illustration: A Valid Argument

Consider:

"Everybody thinks. Therefore, Socrates thinks."

Let T be the predicate "thinks" and s denote Socrates:

$$\forall x\, Tx \models Ts$$

$$\begin{array}{l} \mathsf{T} : \forall x\, Tx \;\checkmark \\ \mathsf{F} : Ts \\ \quad\quad\vert \\ \mathsf{T} : Ts\times \end{array}$$

The constant s occurs in the conclusion, so we instantiate the universal formula to s. We get $\mathsf{T} : Ts$ and $\mathsf{F} : Ts$: contradiction!

Conclusion: The argument is **valid**.

7.4 False Universal Formulas (F∀)

Like true existential formulas, false universal formulas have an **existential import**: denying that φ holds for all individuals means asserting that there exists at least one individual for which φ fails.

Rule F∀

$$\frac{\mathsf{F} : \forall x\, \varphi(x)}{\mathsf{F} : \varphi(t)}$$

where t is an **arbitrary** closed term that does **not** occur on the branch from the root to the instantiation node.

7.4.1 Illustration: An Invalid Argument

Consider:

$$\neg\forall x\, Px \models \neg Pa \quad ?$$

In words: "Not everything is P. Therefore, a is not P."

Model-Theoretic Argument. This argument is clearly **not valid**. Consider a model $\mathcal{M} = \langle D, I\rangle$ with:

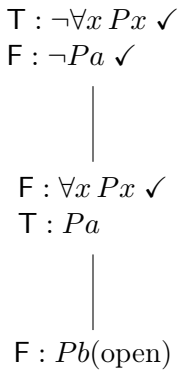
- $D = \{\alpha, \beta\}$
- $I(P) = \{\alpha\}$
- $I(a) = \alpha$

In this model:

- $\mathcal{M} \models \neg\forall x\, Px$ because $\mathcal{M} \models \neg P\beta$ (i.e., $\beta \notin I(P)$).
- $\mathcal{M} \models Pa$ because $I(a) = \alpha \in I(P)$.

So the premise is true but the conclusion $\neg Pa$ is false. We have found a **countermodel**.

Tableau Construction.



The rule F∀ requires an **arbitrary** term. We cannot use a (it occurs on the branch), so we use a fresh constant b. The branch remains open with $\mathsf{T} : Pa$ and $\mathsf{F} : Pb$—no contradiction.

Conclusion: The argument is **not valid**.

7.5 Summary of Quantifier Rules

Rule	Import	Instantiation Term	Branching
T∃	Existential	Arbitrary (fresh)	No
F∃	Universal	All terms on branch	No
T∀	Universal	All terms on branch	No
F∀	Existential	Arbitrary (fresh)	No

Remark 7.3. Rules with **existential import** (T∃, F∀) can only be applied **once** per formula. Rules with **universal import** (F∃, T∀) may need to be applied **multiple times** as new terms appear on a branch.

8 Order of Rule Application

In first-order logic, unlike propositional logic, the **order** in which rules are applied matters for efficiency and termination. Here is the recommended procedure:

Procedure for Constructing First-Order Tableaux

1. **Apply propositional rules first.**

   Apply all applicable rules for propositional connectives ($\neg$, $\wedge$, $\vee$, $\rightarrow$) until no more can be applied.

2. **Apply rules with existential import.**

   Apply the rules $\mathsf{F}\forall$ and $\mathsf{T}\exists$. For each application, introduce **one new arbitrary term** (a fresh constant not occurring on the branch). Mark the formula so you don't apply the rule to it again.

3. **Apply rules with universal import.**

   Apply the rules $\mathsf{T}\forall$ and $\mathsf{F}\exists$:

   - If closed terms occur on the branch connecting the instantiation node to the root, instantiate to **all** those terms.
   - If no such terms exist, introduce any new closed term.

   Do **not** mark these formulas, as they may need to be applied again.

4. **Repeat if necessary.**

   If some branches are still open, return to step 1 and repeat until:

   - All branches close (the argument is valid), or
   - The tableau loops infinitely (the argument is invalid), or
   - The tableau stabilizes with open branches (the argument is invalid).

9 Worked Examples for First-Order Logic

9.1 Example 1: $\exists x(Px \wedge Qx) \models \exists x\, Px$

This argument says: "Something is both P and Q. Therefore, something is P."

$$\begin{array}{c} \mathsf{T} : \exists x(Px \wedge Qx) \checkmark \\ \mathsf{F} : \exists x\, Px \checkmark \\ | \\ \mathsf{T} : Pa \wedge Qa \checkmark \\ | \\ \mathsf{T} : Pa \\ \mathsf{T} : Qa \\ | \\ \mathsf{F} : Pa \times \end{array}$$

Explanation:

1. Apply $\mathsf{T}\exists$ to the premise: introduce fresh constant a, yielding $\mathsf{T} : Pa \wedge Qa$.

2. Apply $\mathsf{T}\wedge$: get $\mathsf{T} : Pa$ and $\mathsf{T} : Qa$.

3. Apply $\mathsf{F}\exists$ to the conclusion: instantiate to a (the only term on the branch), yielding $\mathsf{F} : Pa$.

4. Contradiction: $\mathsf{T} : Pa$ and $\mathsf{F} : Pa$.

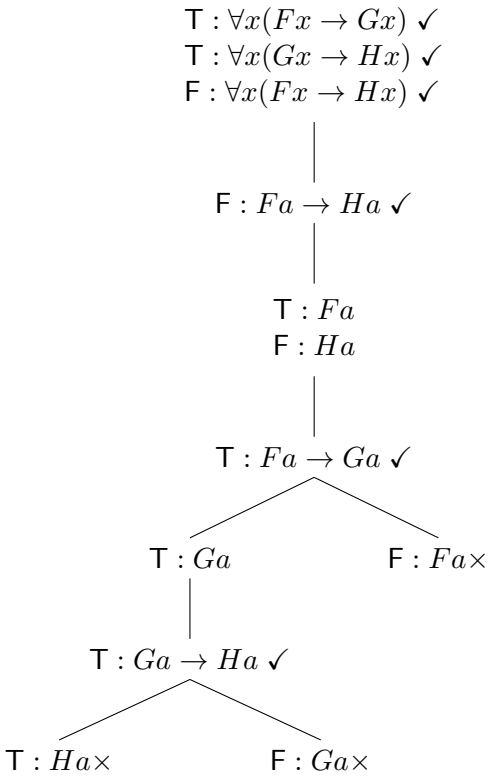
Conclusion: The argument is **valid**.

9.2 Example 2: The Barbara Syllogism

The Barbara syllogism is one of the most famous valid argument forms:

$$\frac{\forall x(Fx \to Gx) \qquad \forall x(Gx \to Hx)}{\forall x(Fx \to Hx)}$$

In words: "All Fs are Gs. All Gs are Hs. Therefore, all Fs are Hs."



Explanation:

1. Apply $\mathsf{F}\forall$ to the conclusion: introduce fresh a, get $\mathsf{F} : Fa \to Ha$.

2. Apply $\mathsf{F} \to$: get $\mathsf{T} : Fa$ and $\mathsf{F} : Ha$.

3. Apply $\mathsf{T}\forall$ to first premise with term a: get $\mathsf{T} : Fa \to Ga$.

4. Apply $\mathsf{T} \to$: branches into $\mathsf{F} : Fa$ (closes) and $\mathsf{T} : Ga$.

5. Apply $\mathsf{T}\forall$ to second premise with term a: get $\mathsf{T} : Ga \to Ha$.

6. Apply $\mathsf{T} \to$: branches into $\mathsf{F} : Ga$ (closes) and $\mathsf{T} : Ha$ (closes with $\mathsf{F} : Ha$).

All branches close. **Conclusion**: The Barbara syllogism is **valid**.

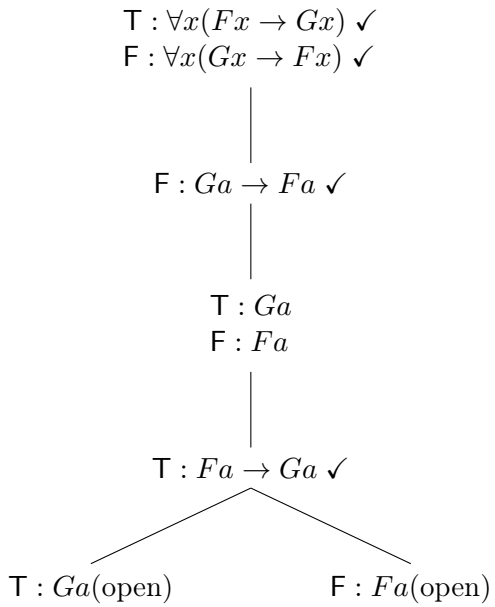
10 Using Tableaux to Find Countermodels

When a tableau has open branches, we can read off a **countermodel** from the open branch. The procedure is:

1. Identify all constants appearing on the open branch.
2. The domain D consists of one element for each constant (or we may identify some constants if needed).
3. For each predicate P:
   - If $\mathsf{T} : Pa$ appears, then $I(a) \in I(P)$.
   - If $\mathsf{F} : Pa$ appears, then $I(a) \notin I(P)$.

10.1 Example: $\forall x(Fx \to Gx) \not\models \forall x(Gx \to Fx)$

This argument claims: "All Fs are Gs. Therefore, all Gs are Fs." Intuitively, this is invalid (e.g., all dogs are mammals, but not all mammals are dogs).



Both sub-branches remain open. Reading from either branch, we have:

- $\mathsf{T} : Ga$
- $\mathsf{F} : Fa$

Countermodel:

- $D = \{\alpha\}$
- $I(a) = \alpha$
- $I(F) = \emptyset$ (nothing is F)
- $I(G) = \{\alpha\}$ (a is G)

Verification:

- Premise $\forall x(Fx \to Gx)$: True, because there are no Fs (the antecedent is always false).
- Conclusion $\forall x(Gx \to Fx)$: False, because Ga is true but Fa is false.

Conclusion: The argument is **not valid**.

11 Infinite Tableaux

Unlike propositional logic, tableaux for first-order logic may **not terminate**. A tableau may grow infinitely if universal-import rules keep introducing new instances without ever closing all branches.

11.1 Conditions for a Completed Tableau

In order to construct a completed tableau for an (invalid) argument, we must ensure that every necessary step has been applied:

1. Each rule with **existential import** ($\mathsf{T}\exists$, $\mathsf{F}\forall$) has been applied **at least once**.
2. Rules with **universal import** ($\mathsf{F}\exists$, $\mathsf{T}\forall$) have been applied to instantiate the quantifiers to **all terms occurring on the relevant branches**.

If these conditions are met and the tableau still has open branches, the argument is invalid. However, sometimes the process never terminates.

11.2 Example: $\nvdash \exists x \, \forall y \, Rxy$

Let us show that $\exists x \, \forall y \, Rxy$ is **not a logical truth** (not provable from the empty set of premises), i.e.:

$$\nvDash \exists x \, \forall y \, Rxy$$

We construct the tableau starting with $\mathsf{F} : \exists x \, \forall y \, Rxy$:

$\mathsf{F} : \exists x \, \forall y \, Rxy$

$\mathsf{F} : \forall y \, Ray$

$\mathsf{F} : Rab$

$\mathsf{F} : \forall y \, Rby$

$\mathsf{F} : Rbc$

$\mathsf{F} : \forall y \, Rcy$

$\vdots$(continues infinitely)

Explanation:

1. Apply $\mathsf{F}\exists$: Since no term exists yet, introduce a. Get $\mathsf{F} : \forall y \, Ray$.
2. Apply $\mathsf{F}\forall$: Must use arbitrary term, so introduce fresh b. Get $\mathsf{F} : Rab$.
3. Apply $\mathsf{F}\exists$ again (must instantiate to all terms): introduce term b. Get $\mathsf{F} : \forall y \, Rby$.

4. Apply F∀: introduce fresh c. Get $\mathsf{F} : Rbc$.

5. And so on...

The tableau **never closes** and grows **infinitely**. Each application of the existential-import rule introduces a new constant, which then requires new instantiations of the universal-import rule.

11.3 Interpreting Infinite Tableaux

An infinite open tableau demonstrates that the argument is **invalid**, but we cannot represent the full countermodel finitely using the standard reading.

Finite countermodel: We can still construct a finite countermodel by "collapsing" the domain:

- $D = \{\alpha\}$
- $I(a) = I(b) = I(c) = \cdots = \alpha$
- $I(R) = \emptyset$

In this model, $\exists x \, \forall y \, Rxy$ is false because for every x, there exists some y such that $\neg Rxy$ (in fact, Rxy is false for all pairs).

Infinite countermodel: If we insist that each constant names a distinct individual:

- $D = \mathbb{N} = \{0, 1, 2, 3, \ldots\}$
- $I(a) = 0$, $I(b) = 1$, $I(c) = 2$, etc.
- $I(R) = \emptyset$

11.4 Tableaux and Decidability

This example illustrates an important fact: **tableaux do not provide a decision procedure for first-order logic**. A tableau may cycle infinitely, and in general, we cannot determine in advance whether it will eventually close.

This is related to the **undecidability of first-order logic**: there is no algorithm that, given an arbitrary first-order formula, always terminates and correctly determines whether the formula is valid.

However:

- If the tableau **closes**, we know the argument is valid (tableaux are **sound**).
- If the tableau remains **open with a finite completed branch**, we know the argument is invalid.
- If the tableau **grows infinitely**, we may need other methods to determine validity.

12 Summary of First-Order Tableaux

1. **Quantifier Rules**:
   - T∃ and F∀ have **existential import**: use arbitrary (fresh) terms.
   - F∃ and T∀ have **universal import**: instantiate to all terms on the branch.
2. **Order of Application**:

(a) Propositional rules first.

(b) Existential-import rules (once each).

(c) Universal-import rules (to all terms).

(d) Repeat until closure or stabilization.

3. **Countermodels**: Read from open branches by assigning truth values to predicates based on signed atomic formulas.

4. **Infinite Tableaux**: Some invalid arguments produce infinite tableaux. This reflects the undecidability of first-order logic.

Part III
Soundness of the Tableaux Method

13 What Does Soundness Mean?

13.1 The Soundness Theorem

The **soundness theorem** for the tableaux method states that if we can derive a conclusion from premises using tableaux, then the conclusion is indeed a logical consequence of those premises:

Theorem 13.1 (Soundness)**.** *If* $\Gamma \vdash \varphi$*, then* $\Gamma \models \varphi$*.*

In words: if the tableau for the argument Γ/φ closes (meaning we can "prove" φ from Γ using tableaux), then φ is a genuine logical consequence of Γ (there is no countermodel).

13.2 The Contrapositive Approach

To prove soundness, it is more convenient to work with the **contrapositive**:

If $\Gamma \vdash \varphi$, then $\Gamma \models \varphi$.
Contrapositive: If $\Gamma \not\models \varphi$, then $\Gamma \nvdash \varphi$.

Let us unpack what each side of the contrapositive means:

- **What does $\Gamma \not\models \varphi$ mean?**

  It means that there exists a **countermodel** to the argument form: a model $\mathcal{M}$ in which all the premises in Γ are true, but the conclusion φ is false.

  Equivalently, the set $\Gamma \cup \{\neg\varphi\}$ is **satisfiable** (there is a model that makes all formulas in this set true).

- **What does $\Gamma \nvdash \varphi$ mean in the tableaux system?**

  It means that the tableau we build from the initial configuration—with all premises marked T and the conclusion marked F—**stays open** (does not close) when we apply the tableaux rules.

13.3 The Proof Strategy

So, to prove soundness, we need to show:

Key Claim: If $\Gamma \cup \{\neg\varphi\}$ is satisfiable, then the tableau we build from
$$\mathsf{T} : \gamma_1, \ldots, \mathsf{T} : \gamma_n, \mathsf{F} : \varphi$$
(where $\Gamma = \{\gamma_1, \ldots, \gamma_n\}$) stays open when we apply the tableaux rules.

If we prove this claim, we will have proved the soundness theorem for tableaux.

14 The Proof of Soundness

14.1 Satisfiability of a Branch

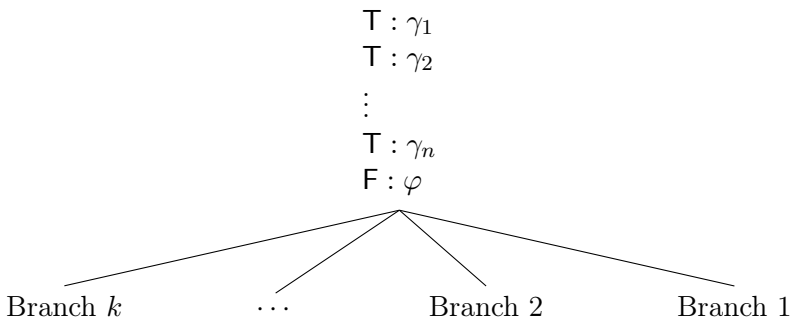
Definition 14.1 (Satisfiable Branch)**.** A tableau branch is **satisfiable** if and only if there exists a model $\mathcal{M}$ such that:

- All formulas marked T on the branch are true in $\mathcal{M}$.
- All formulas marked F on the branch are false in $\mathcal{M}$.

Remark 14.1. A branch that contains both $\mathsf{T} : \alpha$ and $\mathsf{F} : \alpha$ for some formula α is **not satisfiable**: no model can make α both true and false. Such a branch is **closed**.

14.2 The Core Insight

Consider what happens when we start building a tableau:



Key observation: If the initial set $\{\gamma_1, \ldots, \gamma_n, \neg\varphi\}$ is satisfiable, then at least one branch must remain satisfiable throughout the construction.

We know that:

- If a tableau **closes**, then **no branch is satisfiable** (every branch contains a contradiction).
- **Contrapositive**: If **at least one branch remains satisfiable**, then the tableau **does not close**.

14.3 What We Need to Prove

To complete the proof, we must show that applying any tableaux rule to a satisfiable branch preserves satisfiability in the appropriate sense:

1. **Non-branching rules**: If a branch is satisfiable and we apply a non-branching rule, the extended branch remains satisfiable.
2. **Branching rules**: If a branch is satisfiable and we apply a branching rule, **at least one** of the resulting branches is satisfiable.

We now verify this for each rule.

15 Verification for Propositional Rules

15.1 Rule for True Negation (T¬)

Rule: From $\mathsf{T} : \neg\varphi$, derive $\mathsf{F} : \varphi$.

This is a non-branching rule.

$$\begin{array}{c} \Gamma, \text{ including } \mathsf{T} : \neg\varphi \\ \Big| \\ \mathsf{F} : \varphi \end{array}$$

Proof that satisfiability is preserved:

Suppose Γ is satisfiable, so there exists a model $\mathcal{M}$ satisfying all signed formulas in Γ. Since $\mathsf{T} : \neg\varphi \in \Gamma$, we have $\mathcal{M} \models \neg\varphi$, which means $\mathcal{M} \not\models \varphi$.

Therefore, $\mathcal{M}$ also satisfies $\mathsf{F} : \varphi$. The extended branch is satisfiable. □

15.2 Rule for False Negation (F¬)

Rule: From $\mathsf{F} : \neg\varphi$, derive $\mathsf{T} : \varphi$.

This is a non-branching rule.

$$\begin{array}{c} \Gamma, \text{ including } \mathsf{F} : \neg\varphi \\ \Big| \\ \mathsf{T} : \varphi \end{array}$$

Proof:

Suppose Γ is satisfiable via model $\mathcal{M}$. Since $\mathsf{F} : \neg\varphi \in \Gamma$, we have $\mathcal{M} \not\models \neg\varphi$, which means $\mathcal{M} \models \varphi$.

Therefore, $\mathcal{M}$ satisfies $\mathsf{T} : \varphi$. The extended branch is satisfiable. □

15.3 Rule for True Conjunction (T∧)

Rule: From $\mathsf{T} : \varphi \wedge \psi$, derive $\mathsf{T} : \varphi$ and $\mathsf{T} : \psi$.

This is a non-branching rule.

$$\begin{array}{c} \Gamma, \text{ including } \mathsf{T} : \varphi \wedge \psi \\ \Big| \\ \mathsf{T} : \varphi \\ \mathsf{T} : \psi \end{array}$$

Proof:

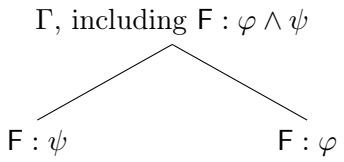
Suppose Γ is satisfiable via model $\mathcal{M}$. Since $\mathsf{T} : \varphi \wedge \psi \in \Gamma$, we have $\mathcal{M} \models \varphi \wedge \psi$.

By the semantics of conjunction, this means $\mathcal{M} \models \varphi$ **and** $\mathcal{M} \models \psi$.

Therefore, $\mathcal{M}$ satisfies both $\mathsf{T} : \varphi$ and $\mathsf{T} : \psi$. The extended branch is satisfiable. □

15.4 Rule for False Conjunction (F$\wedge$)

Rule: From $\mathsf{F} : \varphi \wedge \psi$, branch into $\mathsf{F} : \varphi$ (left) and $\mathsf{F} : \psi$ (right).
This is a branching rule.



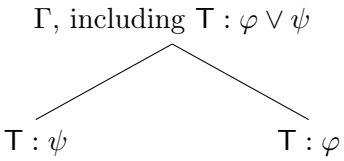
Proof:
Suppose Γ is satisfiable via model $\mathcal{M}$. Since $\mathsf{F} : \varphi \wedge \psi \in \Gamma$, we have $\mathcal{M} \not\models \varphi \wedge \psi$.
By the semantics of conjunction, this means $\mathcal{M} \not\models \varphi$ **or** $\mathcal{M} \not\models \psi$ (or both).

- **Case 1**: $\mathcal{M} \not\models \varphi$. Then $\mathcal{M}$ satisfies $\mathsf{F} : \varphi$, so the **left branch** is satisfiable.
- **Case 2**: $\mathcal{M} \not\models \psi$. Then $\mathcal{M}$ satisfies $\mathsf{F} : \psi$, so the **right branch** is satisfiable.

In either case, **at least one branch** is satisfiable. □

15.5 Rule for True Disjunction (T$\vee$)

Rule: From $\mathsf{T} : \varphi \vee \psi$, branch into $\mathsf{T} : \varphi$ (left) and $\mathsf{T} : \psi$ (right).
This is a branching rule.



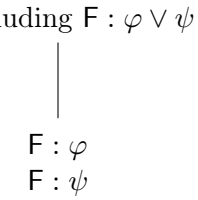
Proof:
Suppose Γ is satisfiable via model $\mathcal{M}$. Since $\mathsf{T} : \varphi \vee \psi \in \Gamma$, we have $\mathcal{M} \models \varphi \vee \psi$.
By the semantics of disjunction, this means $\mathcal{M} \models \varphi$ **or** $\mathcal{M} \models \psi$ (or both).

- **Case 1**: $\mathcal{M} \models \varphi$. Then the **left branch** is satisfiable.
- **Case 2**: $\mathcal{M} \models \psi$. Then the **right branch** is satisfiable.

In either case, **at least one branch** is satisfiable. □

15.6 Rule for False Disjunction (F$\vee$)

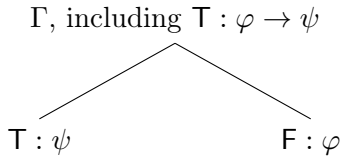
Rule: From $\mathsf{F} : \varphi \vee \psi$, derive $\mathsf{F} : \varphi$ and $\mathsf{F} : \psi$.
This is a non-branching rule.



Proof:
Suppose Γ is satisfiable via model $\mathcal{M}$. Since $\mathsf{F} : \varphi \vee \psi \in \Gamma$, we have $\mathcal{M} \not\models \varphi \vee \psi$.
By the semantics of disjunction, this means $\mathcal{M} \not\models \varphi$ **and** $\mathcal{M} \not\models \psi$.
Therefore, $\mathcal{M}$ satisfies both $\mathsf{F} : \varphi$ and $\mathsf{F} : \psi$. The extended branch is satisfiable. □

15.7 Rule for True Conditional ($\mathsf{T}\,\rightarrow$)

Rule: From $\mathsf{T} : \varphi \rightarrow \psi$, branch into $\mathsf{F} : \varphi$ (left) and $\mathsf{T} : \psi$ (right).
This is a branching rule.



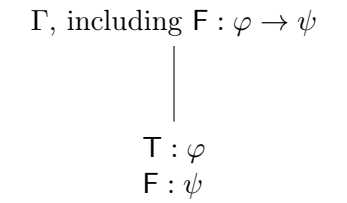
Proof:
Suppose Γ is satisfiable via model $\mathcal{M}$. Since $\mathsf{T} : \varphi \rightarrow \psi \in \Gamma$, we have $\mathcal{M} \models \varphi \rightarrow \psi$.
By the semantics of the conditional, this means $\mathcal{M} \not\models \varphi$ **or** $\mathcal{M} \models \psi$ (or both).

- **Case 1**: $\mathcal{M} \not\models \varphi$. Then the **left branch** is satisfiable.
- **Case 2**: $\mathcal{M} \models \psi$. Then the **right branch** is satisfiable.

In either case, **at least one branch** is satisfiable. □

15.8 Rule for False Conditional ($\mathsf{F}\,\rightarrow$)

Rule: From $\mathsf{F} : \varphi \rightarrow \psi$, derive $\mathsf{T} : \varphi$ and $\mathsf{F} : \psi$.
This is a non-branching rule.



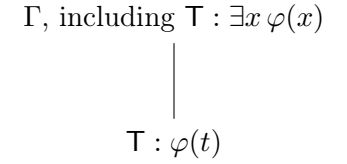
Proof:
Suppose Γ is satisfiable via model $\mathcal{M}$. Since $\mathsf{F} : \varphi \rightarrow \psi \in \Gamma$, we have $\mathcal{M} \not\models \varphi \rightarrow \psi$.
By the semantics of the conditional, this means $\mathcal{M} \models \varphi$ **and** $\mathcal{M} \not\models \psi$.
Therefore, $\mathcal{M}$ satisfies both $\mathsf{T} : \varphi$ and $\mathsf{F} : \psi$. The extended branch is satisfiable. □

16 Verification for Quantifier Rules

We now extend the proof to first-order logic by verifying that the quantifier rules also preserve satisfiability.

16.1 Rule for True Existential ($\mathsf{T}\exists$)

Rule: From $\mathsf{T} : \exists x\, \varphi(x)$, derive $\mathsf{T} : \varphi(t)$, where t is an arbitrary (fresh) closed term.
This is a non-branching rule.



Proof:

Suppose Γ is satisfiable via model $\mathcal{M} = \langle D, I \rangle$. Since $\mathsf{T} : \exists x\, \varphi(x) \in \Gamma$, we have $\mathcal{M} \models \exists x\, \varphi(x)$.

By the semantics of the existential quantifier, there exists some individual $d \in D$ such that φ is true of d in $\mathcal{M}$.

Now, t is a **fresh** constant not occurring elsewhere on the branch. We can **extend** the interpretation I by setting $I(t) = d$ (the witness for the existential). With this extended interpretation I', we have $\mathcal{M}' = \langle D, I' \rangle$ satisfying $\varphi(t)$.

Moreover, since t did not occur in Γ, the model $\mathcal{M}'$ still satisfies all formulas in Γ (the interpretation of other terms is unchanged).

Therefore, the extended branch is satisfiable. □

Remark 16.1. The freshness condition on t is **essential**. If t already occurred on the branch with a fixed interpretation, we could not freely assign it to be the witness for the existential, and the argument would fail.

16.2 Rule for False Existential (F∃)

Rule: From $\mathsf{F} : \exists x\, \varphi(x)$, derive $\mathsf{F} : \varphi(t)$ for all closed terms t occurring on the branch (or a fresh term if none exist).

This is a non-branching rule (though it may produce multiple formulas).

$$\begin{array}{c} \Gamma, \text{ including } \mathsf{F} : \exists x\, \varphi(x) \\ \Big| \\ \mathsf{F} : \varphi(t_1), \mathsf{F} : \varphi(t_2), \ldots \end{array}$$

Proof:

Suppose Γ is satisfiable via model $\mathcal{M} = \langle D, I \rangle$. Since $\mathsf{F} : \exists x\, \varphi(x) \in \Gamma$, we have $\mathcal{M} \not\models \exists x\, \varphi(x)$.

By the semantics of the existential quantifier (and its negation), this means $\mathcal{M} \models \forall x\, \neg\varphi(x)$: for **every** individual $d \in D$, φ is false of d.

In particular, for any closed term t occurring on the branch, $I(t) \in D$, so $\varphi(t)$ is false in $\mathcal{M}$.

Therefore, $\mathcal{M}$ satisfies $\mathsf{F} : \varphi(t)$ for every such t. The extended branch is satisfiable. □

16.3 Rule for True Universal (T∀)

Rule: From $\mathsf{T} : \forall x\, \varphi(x)$, derive $\mathsf{T} : \varphi(t)$ for all closed terms t occurring on the branch (or a fresh term if none exist).

This is a non-branching rule.

$$\begin{array}{c} \Gamma, \text{ including } \mathsf{T} : \forall x\, \varphi(x) \\ \Big| \\ \mathsf{T} : \varphi(t_1), \mathsf{T} : \varphi(t_2), \ldots \end{array}$$

Proof:

Suppose Γ is satisfiable via model $\mathcal{M} = \langle D, I \rangle$. Since $\mathsf{T} : \forall x\, \varphi(x) \in \Gamma$, we have $\mathcal{M} \models \forall x\, \varphi(x)$.

By the semantics of the universal quantifier, for **every** individual $d \in D$, φ is true of d.

In particular, for any closed term t occurring on the branch, $I(t) \in D$, so $\varphi(t)$ is true in $\mathcal{M}$.

Therefore, $\mathcal{M}$ satisfies $\mathsf{T} : \varphi(t)$ for every such t. The extended branch is satisfiable. □

16.4 Rule for False Universal (F∀)

Rule: From $\mathsf{F} : \forall x\, \varphi(x)$, derive $\mathsf{F} : \varphi(t)$, where t is an arbitrary (fresh) closed term.

This is a non-branching rule.

$$\begin{array}{c} \Gamma, \text{ including } \mathsf{F} : \forall x\, \varphi(x) \\ \Big| \\ \mathsf{F} : \varphi(t) \end{array}$$

Proof:

Suppose Γ is satisfiable via model $\mathcal{M} = \langle D, I \rangle$. Since $\mathsf{F} : \forall x\, \varphi(x) \in \Gamma$, we have $\mathcal{M} \not\models \forall x\, \varphi(x)$.

By the semantics of the universal quantifier (and its negation), this means $\mathcal{M} \models \exists x\, \neg\varphi(x)$: there exists some individual $d \in D$ such that φ is false of d.

Now, t is a **fresh** constant not occurring elsewhere on the branch. We can **extend** the interpretation I by setting $I(t) = d$ (the counterexample to the universal). With this extended interpretation I', we have $\mathcal{M}' = \langle D, I' \rangle$ satisfying $\neg\varphi(t)$, i.e., $\mathcal{M}' \not\models \varphi(t)$.

Moreover, since t did not occur in Γ, the model $\mathcal{M}'$ still satisfies all formulas in Γ.

Therefore, $\mathcal{M}'$ satisfies $\mathsf{F} : \varphi(t)$, and the extended branch is satisfiable. □

Remark 16.2. Again, the freshness condition is essential. If t were already interpreted as some specific individual, we could not reassign it to be the counterexample.

17 Conclusion of the Proof

We have verified that **every tableaux rule preserves satisfiability** in the appropriate sense:

- **Non-branching rules**: If the branch is satisfiable before applying the rule, it remains satisfiable after.
- **Branching rules**: If the branch is satisfiable before applying the rule, at least one of the resulting branches is satisfiable after.

17.1 Completing the Argument

Proof of the Soundness Theorem. We prove the contrapositive: if $\Gamma \not\models \varphi$, then $\Gamma \nvdash \varphi$.

Suppose $\Gamma \not\models \varphi$. This means there exists a countermodel: a model $\mathcal{M}$ such that $\mathcal{M} \models \gamma$ for all $\gamma \in \Gamma$, and $\mathcal{M} \not\models \varphi$.

Equivalently, the initial branch of the tableau

$$\mathsf{T} : \gamma_1, \ldots, \mathsf{T} : \gamma_n, \mathsf{F} : \varphi$$

is satisfiable (by $\mathcal{M}$).

Now, as we apply tableaux rules:

- If we apply a non-branching rule, the branch remains satisfiable (as we proved for each rule).
- If we apply a branching rule, at least one branch remains satisfiable.

By induction on the number of rule applications, at every stage of the tableau construction, **at least one branch is satisfiable**.

A satisfiable branch cannot be closed (it cannot contain both $\mathsf{T} : \alpha$ and $\mathsf{F} : \alpha$ for any formula α).

Therefore, the tableau never completely closes: $\Gamma \nvdash \varphi$. □

17.2 Summary of the Soundness Proof

Soundness Theorem: If $\Gamma \vdash \varphi$ (the tableau closes), then $\Gamma \models \varphi$ (there is no countermodel).

Proof Idea:

1. We proved the contrapositive: if there is a countermodel (the initial configuration is satisfiable), the tableau stays open.
2. We showed that each rule preserves satisfiability: if a branch is satisfiable, then after applying any rule, at least one branch remains satisfiable.
3. By induction, at least one branch remains satisfiable throughout, so the tableau never closes.

This completes the proof of soundness for the tableaux method in first-order logic.