Formal Logic

Discrete Math, Fall 2025

Konstantin Chukharev

Set Theory

- Operations & laws
- Power sets
- Russell's paradox
- ZFC axioms
- Cartesian products
- Cardinality

Binary Relations

- Relation properties
- Equivalence relations
- Functions
- Partial orders
- Lattices
- Well-orders

Boolean Algebra

- Truth tables & laws
- Logic circuits
- Normal forms
- Karnaugh maps
- Binary Decision Diagrams (BDDs)

Formal Logic

- Syntax & semantics
- Natural deduction
- Soundness & completeness
- Categorical logic
- First-order logic

Formal Logic

"Logic is the anatomy of thought."

— John Locke

Gottfried Wilhelm Leibniz

Gottlob Frege

Kurt Gödel

Alfred Tarski

Gerhard Gentzen

Jacques Herbrand

Per Martin-Löf

Why Formal Logic?

From Boolean Algebra to Formal Reasoning

In Boolean Algebra, we studied how to *compute* with truth values — evaluating expressions like $(P \wedge Q) \vee \neg P$ given specific values for P and Q.

Now we ask a deeper question: which formulas are *always* true, regardless of the values we plug in? And how do we *prove* that an argument is valid?

Logic answers:

- What makes an argument *valid*?
- Can we *mechanically check* proofs?
- What can (and cannot) be proven?

Applications of Formal Logic

Computer Science:

- Program verification (“this code never crashes”)
- Type systems (Curry–Howard correspondence)
- Database queries (SQL WHERE clauses)
- Hardware design (circuit correctness)
- AI reasoning (knowledge bases, planners)

Mathematics:

- Foundations of mathematics
- Automated theorem proving
- Proof assistants (Lean, Coq, Isabelle)
- Decidability questions
- Model theory

Example: *Modus ponens* — the basic inference rule:

$$\frac{P \qquad P \to Q}{\therefore Q}$$

If “it rains” (P) and “if it rains, the ground is wet” ($P \to Q$), then ($\therefore$) “the ground is wet” (Q).

What is Propositional Logic?

Definition 1: *Logic* is the study of valid reasoning — distinguishing correct arguments from fallacies.

Definition 2: *Formal logic* studies reasoning using precise symbolic notation, enabling mechanical verification of arguments.

Definition 3: *Propositional logic* is the simplest formal logic, dealing with whole statements (*propositions*) that are either `true` or `false`.

Also known as *sentential logic*, *statement logic*, or *zeroth-order logic*.

Propositional logic is the *foundation* for more expressive logics (first-order, modal, temporal) that we'll explore later.

Syntax: The Language of Logic

Before we can ask whether a formula is *true*, we need to specify what counts as a valid formula. This is the job of **syntax**: defining the grammar of our logical language.

Syntax is purely about structure — it says nothing about meaning.

Definition 4: A *propositional language* $\mathcal{L}$ consists of:

- *Propositional variables* (atoms): $P, Q, R, \ldots$ or $p_1, p_2, p_3, \ldots$
- *Logical connectives*: $\neg$ (negation), $\wedge$ (conjunction), $\vee$ (disjunction), $\rightarrow$ (implication), $\iff$ (biconditional)
- *Punctuation*: parentheses (and) for grouping
- *Constants* (optional): $\top$ (true), $\bot$ (false)

Syntax: The Language of Logic [2]

Definition 5: A *well-formed formula* (WFF) is defined inductively:

1. Every propositional variable $P, Q, R, \ldots$ is a WFF.
2. The constants $\top$ and $\bot$ are WFFs.
3. If φ is a WFF, then $(\neg\varphi)$ is a WFF.
4. If φ and ψ are WFFs, then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \to \psi)$, $(\varphi \iff \psi)$ are WFFs.
5. Nothing else is a WFF.

Note: Outer parentheses can be omitted for readability, *e.g.*, $P \wedge Q$ instead of $(P \wedge Q)$.

Example:

- P, Q, $(P \wedge Q)$, $((P \wedge Q) \to R)$ are WFFs.
- $P \wedge, \to QR$, $PQ \wedge$ are *not* WFFs (malformed).

Operator Precedence

To reduce parentheses, we adopt *precedence conventions* (highest to lowest):

Priority	Connective	Name	Associativity	Example
1 (highest)	$\neg$	Negation	Right	$\neg\neg P \wedge Q \equiv (\neg(\neg P)) \wedge Q$
2	$\wedge$	Conjunction	Left	$P \wedge Q \wedge R \equiv (P \wedge Q) \wedge R$
3	$\vee$	Disjunction	Left	$P \vee Q \vee R \equiv (P \vee Q) \vee R$
4	$\rightarrow$	Implication	Right	$P \rightarrow Q \rightarrow R \equiv P \rightarrow (Q \rightarrow R)$
5 (lowest)	$\iff$	Biconditional	Left	$P \iff Q \iff R \equiv (P \iff Q) \iff R$

Example: Using precedence rules:

- $\neg P \wedge Q$ means $(\neg P) \wedge Q$
- $P \vee Q \wedge R$ means $P \vee (Q \wedge R)$
- $P \rightarrow Q \vee R$ means $P \rightarrow (Q \vee R)$
- $P \iff Q \rightarrow R \vee S$ means $P \iff (Q \rightarrow (R \vee S))$ (**Note:** not in all real systems! See NuSMV)

The Logical Connectives

Negation ($\neg$): "not"

- $\neg P$ is true iff P is false
- Flips the truth value

Disjunction ($\vee$): "or" (inclusive)

- $P \vee Q$ is true iff *at least one* is true
- Like bounded addition: $\max(P, Q)$

Conjunction ($\wedge$): "and"

- $P \wedge Q$ is true iff *both* are true
- Like multiplication: $1 \cdot 1 = 1$, else 0

Biconditional ($\iff$): "if and only if"

- $P \iff Q$ is true iff P and Q have *same* value
- Equivalence test: $(P \to Q) \wedge (Q \to P)$

Understanding Implication

Material implication $P \to Q$ ("if P then Q") often trips people up.

$P \to Q$ is false *only* when P is true and Q is false.

Definition 6: *Material implication* is defined as: $P \to Q \equiv \neg P \lor Q$

Read: "either P is false, or Q is true (or both)."

Example (Why is "false implies anything" true?): Consider: "If pigs fly, then I'm the Queen of England."

- Pigs don't fly (P is false)
- The statement is *vacuously true* — it makes no false claims!
- There's no counterexample: we never have P true and Q false.

Vacuous Truth

When P is false, $P \to Q$ is true *regardless* of Q.

Think of implication as a *promise*: "If P, then Q."

- P true, Q true: promise kept
- P true, Q false: promise broken
- P false: promise untested — not broken!

Example:

- "All unicorns are purple" — `true` (no unicorns)
- "Every element of $\emptyset$ is a dragon" — `true`
- "If $2 + 2 = 5$, then I can fly" — `true`

Warning: "All bugs in this code are fixed" is true if the code has no bugs!

Semantics: The Meaning of Logic

So far we've discussed *syntax* — the grammar of logical formulas.

Semantics is about *meaning*: when is a formula true? The answer depends on what truth values we assign to the variables.

Definition 7: An *interpretation* (also called *truth assignment* or *valuation*) is a function

$$\nu : V \to \mathbb{B}$$

that assigns a truth value to each propositional variable, where V is the set of variables and $\mathbb{B} = \{0, 1\} = \{\texttt{false}, \texttt{true}\}$.

Example: For variables $V = \{P, Q, R\}$, one interpretation is:

$$\nu(P) = \texttt{true}, \quad \nu(Q) = \texttt{false}, \quad \nu(R) = \texttt{true}$$

Semantics: The Meaning of Logic [2]

Definition 8: The *evaluation* (or *truth value*) of a formula φ under interpretation ν, written $[\![\varphi]\!]_\nu$, is defined recursively:

$$
\begin{aligned}
[\![P]\!]_\nu &= \nu(P) \quad \text{for propositional variable } P \\
[\![\top]\!]_\nu &= \texttt{true} \\
[\![\bot]\!]_\nu &= \texttt{false} \\
[\![\neg\alpha]\!]_\nu &= \texttt{true} \iff [\![\alpha]\!]_\nu = \texttt{false} \\
[\![\alpha \wedge \beta]\!]_\nu &= \texttt{true} \iff [\![\alpha]\!]_\nu = \texttt{true} \text{ and } [\![\beta]\!]_\nu = \texttt{true} \\
[\![\alpha \vee \beta]\!]_\nu &= \texttt{true} \iff [\![\alpha]\!]_\nu = \texttt{true} \text{ or } [\![\beta]\!]_\nu = \texttt{true} \\
[\![\alpha \to \beta]\!]_\nu &= \texttt{false} \iff [\![\alpha]\!]_\nu = \texttt{true} \text{ and } [\![\beta]\!]_\nu = \texttt{false} \\
[\![\alpha \iff \beta]\!]_\nu &= \texttt{true} \iff [\![\alpha]\!]_\nu = [\![\beta]\!]_\nu
\end{aligned}
$$

Note: The evaluation function extends the interpretation from atoms to all formulas, using the *compositional* (truth-functional) nature of propositional logic.

Truth Tables

Definition 9: A *truth table* lists all possible interpretations and the resulting truth values of formulas.

Constructing a Truth Table:

1. List variables $P_1, \ldots, P_n$
2. Create 2^n rows for all truth value combinations
3. Compute each subformula column-by-column
4. The final column gives the formula's value for each interpretation

Truth Tables for Basic Connectives

P	$\neg P$
true	false
false	true

P	Q	$P \wedge Q$	$P \vee Q$
true	true	true	true
true	false	false	true
false	true	false	true
false	false	false	false

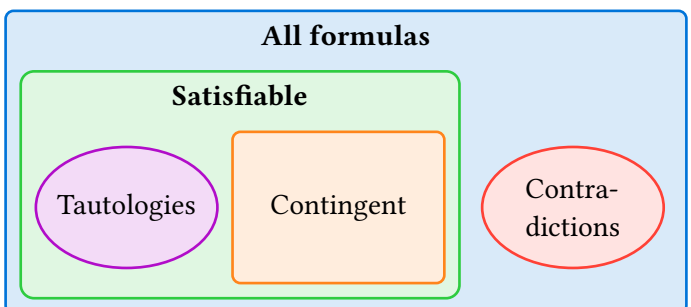
P	Q	$P \to Q$	$P \iff Q$
true	true	true	true
true	false	false	false
false	true	true	false
false	false	true	true

Note: Truth tables connect directly to Boolean algebra: $\wedge$ is multiplication, $\vee$ is (bounded) addition, $\neg$ is complement. See the Boolean Algebra lecture for optimization techniques like Karnaugh maps.

Semantic Classification of Formulas

Definition 10: Formulas are classified by their truth behavior across *all* interpretations:

- **Tautology** (valid): True under *every* interpretation. Notation: $\vDash \varphi$
- **Contradiction**: False under *every* interpretation.
- **Contingent**: True under *some* interpretations, false under others.
- **Satisfiable**: True under *at least one* interpretation (includes tautologies and contingent formulas).



Examples of Classifications

Formula	Classification	Reason
$P \vee \neg P$	Tautology	Law of Excluded Middle
$P \wedge \neg P$	Contradiction	Law of Non-Contradiction
$P \to P$	Tautology	Reflexivity of implication
$P \vee Q$	Contingent	Depends on values of P and Q
$P \wedge Q \to P$	Tautology	Simplification
$(P \to Q) \wedge (Q \to P)$	Contingent	Same as $P \iff Q$

Logical Equivalence

Definition 11: Two formulas φ and ψ are *logically equivalent*, written $\varphi \equiv \psi$, if they have the same truth value under *every* interpretation:

$$\varphi \equiv \psi \quad \text{iff} \quad \forall \nu.\, [\![\varphi]\!]_\nu = [\![\psi]\!]_\nu \quad \text{iff} \quad \vDash \varphi \iff \psi$$

Theorem 1: $\varphi \equiv \psi$ if and only if $\varphi \iff \psi$ is a tautology.

Proof: By definition, $\varphi \equiv \psi$ means $[\![\varphi]\!]_\nu = [\![\psi]\!]_\nu$ for all interpretations ν.

The biconditional $\varphi \iff \psi$ is true exactly when $[\![\varphi]\!]_\nu = [\![\psi]\!]_\nu$.

Therefore, $\varphi \iff \psi$ is true under all interpretations (a tautology) if and only if $\varphi \equiv \psi$. □

Fundamental Equivalence Laws

Just as Boolean algebra has laws for simplifying expressions, propositional logic has equivalence laws. In fact, they're the same laws! Here's a reference:

Identity:
- $P \wedge \top \equiv P$
- $P \vee \bot \equiv P$

Complement:
- $P \wedge \neg P \equiv \bot$
- $P \vee \neg P \equiv \top$

Double Negation:
- $\neg\neg P \equiv P$

De Morgan's Laws:
- $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$
- $\neg(P \vee Q) \equiv \neg P \wedge \neg Q$

Domination:
- $P \vee \top \equiv \top$
- $P \wedge \bot \equiv \bot$

Idempotence:
- $P \wedge P \equiv P$
- $P \vee P \equiv P$

Absorption:
- $P \wedge (P \vee Q) \equiv P$
- $P \vee (P \wedge Q) \equiv P$

Commutativity:
- $P \wedge Q \equiv Q \wedge P$
- $P \vee Q \equiv Q \vee P$

Associativity:
- $(P \wedge Q) \wedge R \equiv P \wedge (Q \wedge R)$
- $(P \vee Q) \vee R \equiv P \vee (Q \vee R)$

Distributivity:
- $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$
- $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$

Implication and Biconditional Laws

Implication Elimination:

- $P \to Q \equiv \neg P \lor Q$
- $\neg(P \to Q) \equiv P \land \neg Q$

Biconditional:

- $P \iff Q \equiv (P \to Q) \land (Q \to P)$
- $P \iff Q \equiv (P \land Q) \lor (\neg P \land \neg Q)$

Contrapositive:

- $P \to Q \equiv \neg Q \to \neg P$

Exportation:

- $(P \land Q) \to R \equiv P \to (Q \to R)$

Note: These equivalences can be verified by truth tables or used to *simplify* formulas algebraically — exactly like simplifying Boolean expressions for circuit optimization.

Semantic Entailment

Definition 12: A set of formulas Γ *semantically entails* (or *logically implies*) a formula φ, written $\Gamma \vDash \varphi$, if every interpretation that satisfies all formulas in Γ also satisfies φ:

$$\Gamma \vDash \varphi \quad \text{iff} \quad \forall \nu.\, (\forall \psi \in \Gamma.\, [\![\psi]\!]_\nu = \texttt{true}) \to [\![\varphi]\!]_\nu = \texttt{true}$$

Example: To check if $\{P, P \to Q\} \vDash Q$ (modus ponens):

1. Find all interpretations that satisfy all formulas in $\Gamma = \{P, P \to Q\}$:
   - $[\![P]\!]_\nu = \texttt{true}$ and $[\![P \to Q]\!]_\nu = \texttt{true}$
   - There is only one such interpretation: $\nu = \{P \mapsto \texttt{true}, Q \mapsto \texttt{true}\}$
2. These interpretations must have $[\![Q]\!]_\nu = \texttt{true}$ for $\Gamma \vDash Q$ to hold.
   - Indeed, $[\![Q]\!]_\nu = \texttt{true}$ under this interpretation.

Example: $\{P \to Q, Q \to R\} \vDash P \to R$ (hypothetical syllogism)

Example: $\{P \lor Q, \neg P\} \vDash Q$ (disjunctive syllogism)

Semantic Deduction Theorem

Theorem 2: $\Gamma \cup \{\varphi\} \vDash \psi$ if and only if $\Gamma \vDash \varphi \to \psi$

Special case: $\{\varphi\} \vDash \psi$ iff $\vDash \varphi \to \psi$

Note: The deduction theorem connects *entailment* with *implication*: to show that premises entail a conclusion, we can equivalently show that the conjunction of premises implies the conclusion.

Proof ($\Rightarrow$): Assume $\Gamma \cup \{\varphi\} \vDash \psi$.

- For any interpretation ν, if $[\![\chi]\!]_\nu = \texttt{true}$ for all $\chi \in \Gamma$ and $[\![\varphi]\!]_\nu = \texttt{true}$, then $[\![\psi]\!]_\nu = \texttt{true}$.
- Therefore, if $[\![\chi]\!]_\nu = \texttt{true}$ for all $\chi \in \Gamma$, then whenever $[\![\varphi]\!]_\nu = \texttt{true}$, we have $[\![\psi]\!]_\nu = \texttt{true}$.
- This means $[\![\varphi \to \psi]\!]_\nu = \texttt{true}$ under all interpretations satisfying Γ, so $\Gamma \vDash \varphi \to \psi$. □

Proof ($\Leftarrow$): Assume $\Gamma \vDash \varphi \to \psi$.

- For any interpretation ν, if $[\![\chi]\!]_\nu = \texttt{true}$ for all $\chi \in \Gamma$, then $[\![\varphi \to \psi]\!]_\nu = \texttt{true}$.
- If also $[\![\varphi]\!]_\nu = \texttt{true}$, then by the definition of implication, $[\![\psi]\!]_\nu = \texttt{true}$.
- Thus, whenever all formulas in $\Gamma \cup \{\varphi\}$ are true, so is ψ, hence $\Gamma \cup \{\varphi\} \vDash \psi$. □

Normal Forms

Literals, Clauses, Cubes, and Normal Forms

Definition 13: A *literal* is a propositional variable or its negation:

- *Positive literal*: P
- *Negative literal*: $\neg P$

Definition 14: A *clause* is a disjunction of literals: $(L_1 \vee L_2 \vee ... \vee L_k)$

Definition 15: A formula is in *conjunctive normal form* (CNF) if it is a conjunction of clauses:

$$\underbrace{\left(L_{1,1} \vee ... \vee L_{1,k_1}\right)}_{\text{clause 1}} \wedge ... \wedge \underbrace{\left(L_{n,1} \vee ... \vee L_{n,k_n}\right)}_{\text{clause n}}$$

CNF Conversion Algorithm

Algorithm to convert any formula to CNF:

1. **Eliminate biconditionals:** $\varphi \iff \psi \rightsquigarrow (\varphi \to \psi) \land (\psi \to \varphi)$
2. **Eliminate implications:** $\varphi \to \psi \rightsquigarrow \neg\varphi \lor \psi$
3. **Push negations inward** (De Morgan + double negation):
   - $\neg(\varphi \land \psi) \rightsquigarrow \neg\varphi \lor \neg\psi$
   - $\neg(\varphi \lor \psi) \rightsquigarrow \neg\varphi \land \neg\psi$
   - $\neg\neg\varphi \rightsquigarrow \varphi$
4. **Distribute** $\lor$ over $\land$:
   - $\varphi \lor (\psi \land \chi) \rightsquigarrow (\varphi \lor \psi) \land (\varphi \lor \chi)$

CNF Conversion Algorithm [2]

Example (Converting to CNF): Convert $(P \to Q) \to R$ to CNF:

1. Eliminate outer implication: $\neg(P \to Q) \lor R$
2. Eliminate inner implication: $\neg(\neg P \lor Q) \lor R$
3. Push negation inward: $(\neg\neg P \land \neg Q) \lor R$
4. Double negation: $(P \land \neg Q) \lor R$
5. Distribute $\lor$ over $\land$: $(P \lor R) \land (\neg Q \lor R)$

Final CNF: $(P \lor R) \land (\neg Q \lor R)$

Theorem 3 (Normal Form Existence): Every propositional formula is logically equivalent to a formula in CNF and to a formula in DNF.

Note:

- **CNF** is preferred for SAT solvers (clause-based reasoning)
- **DNF** is useful for model enumeration and some verification tasks
- Conversion may cause *exponential blowup* in formula size

Connection to SAT

Definition 16: The *satisfiability problem* (SAT): given a propositional formula φ (usually in CNF), determine whether φ is satisfiable.

Theorem 4 (Cook–Levin): SAT is **NP-complete** — the canonical NP-complete problem.

Why SAT matters:

- Foundation of computational complexity theory
- Practical SAT solvers handle formulas with *millions* of variables
- Applications: verification, planning, cryptanalysis, scheduling

See the dedicated lecture on SAT for DPLL algorithm, CDCL, and applications.

Logic Puzzles

"What's the sense of a question without an answer?" asked Alice.
"Ah, that's the kind that makes you think!" he replied.
"Think about what?" asked Alice.
"About what the answer could be," he replied.

— Raymond Smullyan

Why Logic Puzzles?

Logic puzzles are a playground for formal reasoning. They look like entertainment, but solving them requires exactly the same techniques we use for serious applications: formalizing constraints, applying inference rules, and checking for consistency.

What makes them valuable is that they reveal how easily informal intuition can go wrong.

In this section:

- Formalizing puzzle constraints as formulas
- Systematic solving techniques
- Connection to SAT solvers

Knights and Knaves: The Setup

Definition 17: On the *Island of Knights and Knaves*:

- **Knights** always tell the truth
- **Knaves** always lie
- Every inhabitant is either a knight or a knave

Your task: determine who is what from their statements.

Formalization: For each person X, let K_X mean "X is a knight."

A statement S made by X translates to: $K_X \iff S$

Why? If X is a knight, S is true. If X is a knave, S is false.

Puzzle 1: A Simple Start

Example: **Person A says:** "I am a knave."

Is A a knight or a knave?

Solution:

Let K_A = "A is a knight". A's statement is $\neg K_A$ ("I am a knave").

The constraint: $K_A \iff \neg K_A$

- If K_A is true: then $\neg K_A$ must be true (contradiction!)
- If K_A is false: then $\neg K_A$ must be false, so K_A is true (contradiction!)

This is a contradiction! No consistent assignment exists.

$\therefore$ This situation *cannot occur* on the island — A cannot make this statement.

Puzzle 2: Two Inhabitants

Example: **Person A says:** "We are both knaves."

Determine the types of A and B.

Formalization:

- K_A = "A is a knight", K_B = "B is a knight"
- A's statement: $\neg K_A \wedge \neg K_B$
- Constraint: $K_A \iff (\neg K_A \wedge \neg K_B)$

Analysis:

- If K_A (A is a knight): Then $\neg K_A \wedge \neg K_B$ is true, so $\neg K_A$ is true. Contradiction!
- So $\neg K_A$ (A is a knave): Then $\neg K_A \wedge \neg K_B$ is false. Since $\neg K_A$ is true, we need $\neg K_B$ to be false, so K_B is true.

Answer: A is a knave, B is a knight.

Puzzle 3: Mutual Accusations

A says: "B is a knave."

B says: "Me and A are of different types."

What are A and B?

Formalization:

- A's statement: $\neg K_B$ Constraint: $K_A \iff \neg K_B$
- B's statement: $K_A \not\Leftrightarrow K_B$ (different types) Constraint: $K_B \iff (K_A \not\Leftrightarrow K_B)$

From constraint 1: K_A and K_B have opposite values.

Substituting into constraint 2: $K_B \iff \top$ (since $K_A \not\Leftrightarrow K_B$ is always true given constraint 1)

So K_B is true (B is a knight), hence K_A is false (A is a knave).

Answer: A is a knave, B is a knight.

Check: A (knave) lies about B being a knave. B (knight) truthfully says they're different. ✓

Puzzle 4: The Fork in the Road

You reach a fork. One path leads to treasure, the other to doom.

A single inhabitant stands there. You may ask **one yes/no question**.

What question finds the treasure?

The key question: "If I asked you 'Does the left path lead to treasure?', would you say yes?"

Let L = "left leads to treasure".

- **Knight:** Answers the inner question truthfully. Says "yes" iff L.
- **Knave:** Would lie about the inner question. But asked *whether* he'd say yes, he lies *again* — double negation! Says "yes" iff L.

Both types answer "yes" exactly when left leads to treasure.

The double-negation trick: Asking a liar what they *would* say inverts the lie, yielding truth.

From Puzzles to SAT

Knights and Knaves puzzles are SAT problems in disguise!

The translation is direct:

- **Variables:** $K_A, K_B, \ldots$ (one per person: true = knight)
- **Constraints:** For each statement S by person X: add clause $K_X \iff S$
- **Solution:** Find a satisfying assignment

Modern SAT solvers handle puzzles with *thousands* of inhabitants.

Example (Puzzle 2 as CNF): Constraint: $K_A \iff (\neg K_A \wedge \neg K_B)$. Converting to CNF:

$$\begin{aligned}
&(\neg K_A \vee \neg K_A) \wedge (\neg K_A \vee \neg K_B) \wedge (K_A \vee K_A \vee K_B) \\
&\quad \equiv \neg K_A \wedge (\neg K_A \vee \neg K_B) \wedge (K_A \vee K_B) \\
&\quad \equiv \neg K_A \wedge K_B
\end{aligned}$$

SAT solver immediately finds a solution: $K_A = 0, K_B = 1$

The Hardest Logic Puzzle Ever

Boolos's puzzle (1996): Three gods — True, False, and Random — stand before you.

- True always speaks truly; False always lies
- Random answers randomly
- They respond with "da" or "ja" (you don't know which means yes/no)

Challenge: Determine each god's identity using exactly three yes/no questions.

This puzzle combines:

- Knights and Knaves logic
- Unknown language mapping
- Non-deterministic behavior

It was proven that three questions are both *necessary* and *sufficient*!

Reference: *G. Boolos, "The Hardest Logic Puzzle Ever", 1996.*

Self-Reference Puzzles

Example (The Liar's Paradox): Consider the statement: "This statement is false."

- If it's true, then what it says holds, so it's false. Contradiction!
- If it's false, then what it says doesn't hold, so it's not false, *i.e.*, true. Contradiction!

This is not a puzzle to solve — it's a *paradox*!

Unlike Knights and Knaves, there's no consistent truth assignment.

Paradoxes like this motivated:

- Tarski's hierarchy of truth predicates
- Gödel's incompleteness theorems
- Careful treatment of self-reference in formal systems

Takeaway: Formal logic helps us distinguish *solvable puzzles* from *genuine paradoxes*.

Logic Puzzle Strategies

Systematic approach:

1. **Formalize:** Assign a variable to each unknown
2. **Translate:** Convert each statement to a formula
3. **Constrain:** Add $(statement) \iff (speaker\ is\ truthful)$
4. **Simplify:** Apply logical equivalences
5. **Solve:** Find satisfying assignments (or prove none exist)
6. **Verify:** Check against all constraints

Example: **Always verify!** In Puzzle 2:

- A (knave) says "We are both knaves" — false since B is a knight. ✓
- $K_A = 0, K_B = 1$ satisfies all constraints. ✓

Proof Systems

*"Mathematics is not about numbers, equations, or algorithms:
it is about understanding."*

— William Paul Thurston

From Semantics to Syntax

So far we've studied *semantics* — what formulas *mean* in terms of truth values.

Now we turn back to *syntax* — how to *prove* formulas using purely symbolic manipulation.

Semantic approach:

- Assign truth values to variables
- Evaluate formula under each interpretation
- Check all 2^n rows of truth table
- **Answers:** "Is φ true everywhere?"

Syntactic approach:

- Start from axioms or premises
- Apply inference rules step-by-step
- Build a derivation (proof)
- **Answers:** "Can we derive φ?"

Key insight: A proof system derives formulas *without mentioning truth*.

It manipulates symbols according to rules — a purely mechanical process.

What is a Proof System?

A *proof system* is a precise, formal game with strict rules.

Think of it as a machine: you feed in some formulas (premises), turn the crank (apply rules), and out comes a conclusion. The machine doesn't "understand" the formulas — it just manipulates symbols according to patterns.

Definition 18: A **proof system** for a logic consists of:

1. A set of **axioms** — formulas accepted without justification
2. A set of **inference rules** — patterns for deriving new formulas from existing ones

Note: Every step in a proof is *mechanically checkable*. A computer can verify any proof by pattern-matching alone, without understanding what the formulas "mean."

Axioms: Starting Points

Definition 19: An *axiom* is a formula we accept as true without proof — a starting point for all derivations.

Different proof systems make different choices about axioms:

- **Many axioms:** Hilbert systems have infinitely many axiom *schemas*
- **Few axioms:** Some systems minimize axioms for elegance
- **No axioms:** Natural deduction needs *no axioms at all*!

The problem with axioms: We must accept them without justification. Axiom-heavy systems require us to "believe" certain formulas are valid before we can prove anything else.

Inference Rules: The Heart of Proof

An *inference rule* is a template for deriving new formulas. It specifies:

- **Premises** (input formulas we already have)
- **Conclusion** (output formula we can *derive*)

Rules are notated with premises above a horizontal line and conclusion below.

Definition 20: An inference rule has the general form:

$$\frac{\varphi_1 \quad \varphi_2 \quad \cdots \quad \varphi_n}{\psi}$$

Read: "If we have *premises* $\varphi_1, \varphi_2, \ldots, \varphi_n$, then we may derive the *conclusion* ψ."

Modus Ponens

Example: The most famous inference rule — *modus ponens*:

$$\frac{\varphi \qquad \varphi \to \psi}{\psi}$$

Interpretation: "From φ and the conditional $\varphi \to \psi$, we derive ψ."

Note: Rules are *syntactic* — they match patterns in formulas. The rule doesn't care what φ and ψ *mean*, only that they have the right form.

Proofs and Derivability

Definition 21: A *proof* (or *derivation*) of φ from premises Γ is a finite sequence of formulas where:

- The final formula is φ
- Each formula is justified as an axiom, a premise from Γ, or follows from earlier formulas by an inference rule

Definition 22: We write $\Gamma \vdash \varphi$ (read: "Γ proves φ") when such a derivation exists.

When $\Gamma = \emptyset$, we write $\vdash \varphi$ and call φ a **theorem**.

Example: A proof showing $P, P \to Q \vdash Q$:

1. P Premise
2. $P \to Q$ Premise
3. Q Modus ponens on 1, 2

Semantic vs. Syntactic Entailment

We now have **two turnstile symbols** with very different meanings:

- $\Gamma \models \varphi$ is *about meaning*: every interpretation satisfying Γ also satisfies φ.
- $\Gamma \vdash \varphi$ is *about derivation*: there exists a proof of φ from Γ using rules.

Semantic ($\models$):

- About *truth* and *meaning*
- Requires checking interpretations
- "What *is* true?"
- Exponential in worst case (2^n rows)
- External: refers to the world

Syntactic ($\vdash$):

- About *derivation* and *symbols*
- Requires applying rules
- "What can we *derive*?"
- Proof size varies (can be short!)
- Internal: stays within the system

The central question: Do $\models$ and $\vdash$ coincide?

If $\Gamma \models \varphi$ implies $\Gamma \vdash \varphi$ (and vice versa), we can use whichever method is more convenient.

Types of Proof Systems

Many proof systems exist for propositional logic.

They all prove the *same theorems*, but differ in their design:

Hilbert-style systems:
- Many axiom *schemas*
- Few rules (often just modus ponens)
- Proofs are linear sequences
- Historically important, but tedious

Natural deduction:
- **No axioms at all!**
- Many rules (intro/elim per connective)
- Tree-structured or Fitch-style proofs
- Mirrors how mathematicians reason

Note: Other systems: *sequent calculus* (Gentzen), *tableaux* (semantic trees), *resolution* (SAT solvers).

Hilbert Systems: Axiom-Heavy Approach

Hilbert systems have many axiom *schemas* (patterns generating infinitely many axioms) but typically only modus ponens as inference rule.

Axiom schemas (for any formulas φ, ψ, χ):

1. $\varphi \to (\psi \to \varphi)$
2. $(\varphi \to (\psi \to \chi)) \to ((\varphi \to \psi) \to (\varphi \to \chi))$
3. $(\neg\varphi \to \neg\psi) \to (\psi \to \varphi)$

Inference rule: Modus ponens only.

Hilbert proofs are long and unintuitive — even proving $P \to P$ requires several non-obvious steps.

And we must accept those axiom schemas as valid before we can prove anything.

Natural Deduction: The Axiom-Free Alternative

Natural deduction (Gentzen, 1934) takes a radical approach: **no axioms**.

Instead, we have *rules* for each logical connective — ways to *introduce* and *eliminate* it.

Why no axioms?

Each rule is self-evident: it just says what a connective *means*.

- To prove $\varphi \wedge \psi$, prove both φ and ψ
- To use $\varphi \wedge \psi$, extract φ or ψ

The rules *define* the connectives through their behavior.

Natural Deduction: Introduction and Elimination

Definition 23: An *introduction rule* shows how to **prove** a formula with a given connective as its main operator.

Definition 24: An *elimination rule* shows how to **use** a formula with a given connective to derive something new.

Example: **Conjunction ($\wedge$):**

- *Introduction:* From φ and ψ, conclude $\varphi \wedge \psi$
- *Elimination:* From $\varphi \wedge \psi$, conclude φ (or ψ)

Example: **Implication ($\rightarrow$):**

- *Introduction:* Assume φ, derive ψ, conclude $\varphi \rightarrow \psi$
- *Elimination:* From $\varphi \rightarrow \psi$ and φ, conclude ψ (modus ponens)

Note: Each connective gets exactly two rules: one to *build* formulas, one to *use* them.

Hypothetical Reasoning

One of the most powerful features of natural deduction is *hypothetical reasoning*:

- We can temporarily **assume** a formula
- Derive consequences from that assumption
- Then **discharge** the assumption to get a conditional result

This mirrors how mathematicians actually argue: "Suppose P holds. Then... Therefore, if P then Q."

Example (Proving $P \to P$):

1.	P	*Assumption*
2.	P	From 1
3.	$P \to P$	$\to$I 1

This is how mathematicians reason:

"Suppose n is even. Then... Therefore, *if* n is even, *then* n^2 is even."

Fitch Notation

Fitch notation is a structured format for natural deduction proofs:

- **Horizontal lines** separate premises from derived formulas
- **Indentation** (in full Fitch style) shows subproof scope
- Each line is **numbered** and **justified** by a rule

Example: Simple proof using *Modus Ponens* ($\rightarrow$E rule):

1	$P \rightarrow Q$	*Premise*
2	P	*Premise*
3	Q	$\rightarrow$*E 1, 2*

From $P \rightarrow Q$ and P, we derive Q.

Inference Rules: Overview

For each connective, we have:

- **Introduction rule** (I): How to *prove* a formula with that connective
- **Elimination rule** (E): How to *use* a formula with that connective

Connective	Introduction	Elimination	Intuition
$\wedge$	Combine two proofs	Extract component	"Both"
$\vee$	Provide one proof	Case analysis	"Either"
$\rightarrow$	Assume, derive	Modus ponens	"If...then"
$\neg$	Assume, derive $\bot$	Derive $\bot$	"Not"
$\bot$	—	Derive anything	"Absurdity"

Inference Rules for Conjunction

Conjunction Introduction (∧I):

$$\frac{\alpha \quad \beta}{\alpha \wedge \beta}$$

If we have both α and β, we can conclude $\alpha \wedge \beta$.

Conjunction Elimination (∧E):

$$\frac{\alpha \wedge \beta}{\alpha} \qquad \frac{\alpha \wedge \beta}{\beta}$$

From $\alpha \wedge \beta$, we can conclude either α or β.

Inference Rules for Disjunction

Disjunction Introduction ($\vee$I):

$$\frac{\alpha}{\alpha \vee \beta} \qquad \frac{\beta}{\alpha \vee \beta}$$

From either α or β, we can conclude $\alpha \vee \beta$.

Disjunction Elimination ($\vee$E):

$$\frac{\alpha \vee \beta \quad [\alpha]\ldots\gamma \quad [\beta]\ldots\gamma}{\gamma}$$

To use $\alpha \vee \beta$, assume each disjunct and show that both lead to the same conclusion γ.

Inference Rules for Implication

Implication Introduction (→I):

$$\frac{[\alpha]...\beta}{\alpha \to \beta}$$

To prove $\alpha \to \beta$, assume α and derive β.

This *discharges* the assumption α.

Implication Elimination (→E):

$$\frac{\alpha \to \beta \quad \alpha}{\beta}$$

This is *modus ponens*.

Inference Rules for Negation

Negation Introduction (¬I):

$$\frac{[\varphi]\ldots\bot}{\neg\varphi}$$

To prove $\neg\varphi$, assume φ and derive a contradiction $\bot$.

Negation Elimination (¬E):

$$\frac{\varphi \qquad \neg\varphi}{\bot}$$

From φ and $\neg\varphi$, derive contradiction.

Definition 25: *Contradiction* ($\bot$) represents logical inconsistency — a situation that cannot occur.

Inference Rules for Negation [2]

Ex Falso Quodlibet ($\bot$E):

$$\frac{\bot}{\varphi}$$

From contradiction, *anything* follows. (Latin: "from falsehood, anything")

Double Negation Elimination (DNE):

$$\frac{\neg\neg\varphi}{\varphi}$$

This rule is *classical* (not valid in intuitionistic logic).

Classical vs Intuitionistic: Double negation elimination distinguishes *classical* from *intuitionistic* logic. In constructive mathematics, proving $\neg\neg\varphi$ doesn't automatically give us φ — we need a *witness*.

Proof Strategies

Common proof patterns in natural deduction:

Direct Proof:
- Start from premises
- Apply rules step-by-step
- Derive conclusion directly

Proof by Contradiction:
- Assume negation of goal
- Derive contradiction ($\bot$)
- Conclude original goal

Conditional Proof:
- To prove $\varphi \to \psi$
- Assume φ
- Derive ψ, discharge assumption

Proof by Cases:
- Given $\varphi \vee \psi$
- Show goal follows from φ
- Show goal follows from ψ

Worked Example: Contrapositive

Example: Proving $(P \to Q) \to (\neg Q \to \neg P)$:

1	$P \to Q$	*Assumption*
2	$\neg Q$	*Assumption*
3	P	*Assumption*
4	Q	$\to$*E 1, 3*
5	$\bot$	$\neg$*E 2, 4*
6	$\neg P$	$\neg$*I 3-5*
7	$\neg Q \to \neg P$	$\to$*I 2-6*
8	$(P \to Q) \to (\neg Q \to \neg P)$	$\to$*I 1-7*

Worked Example: Proof by Contradiction (RAA)

Example (Law of Excluded Middle): Proving $P \vee \neg P$ using *reductio ad absurdum* (proof by contradiction):

1	$\neg(P \vee \neg P)$	*Assumption (for RAA)*
2	P	*Assumption*
3	$P \vee \neg P$	$\vee I$ *2*
4	$\bot$	$\neg E$ *1, 3*
5	$\neg P$	$\neg I$ *2-4*
6	$P \vee \neg P$	$\vee I$ *5*
7	$\bot$	$\neg E$ *1, 6*
8	$P \vee \neg P$	*RAA 1-7*

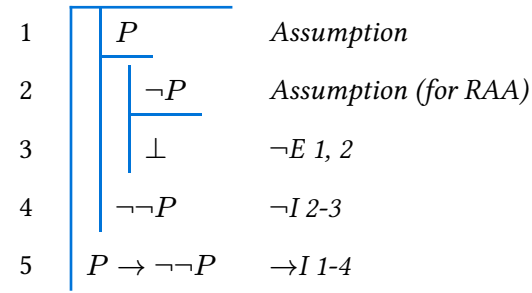
Worked Example: Proof by Contradiction (RAA) [2]

Key technique: We assumed $\neg(P \vee \neg P)$ and derived $\bot$. By RAA, we conclude $P \vee \neg P$.

This is a **classical** proof — it relies on double negation elimination and is not valid in intuitionistic logic!

Worked Example: Double Negation

Example (Double Negation Introduction): Proving $P \to \neg\neg P$:



Worked Example: Double Negation [2]

Example (Double Negation Elimination): Proving $\neg\neg P \to P$ (requires classical logic):

1	$\neg\neg P$	*Assumption*
2	$\neg P$	*Assumption (for RAA)*
3	$\bot$	*$\neg$E 1, 2*
4	P	*RAA 2-3*
5	$\neg\neg P \to P$	*$\to$I 1-4*

Worked Example: Disjunctive Syllogism

Example: Proving $(P \vee Q) \to (\neg P \to Q)$:

1	$P \vee Q$	*Assumption*
2	$\neg P$	*Assumption*
3	P	*Assumption (for case 1)*
4	$\bot$	$\neg$*E 2, 3*
5	Q	$\bot$*E 4*
6	Q	*Assumption (for case 2)*
7	Q	*R 6*
8	Q	$\vee$*E 1, 3-5, 6-7*
9	$\neg P \to Q$	$\to$*I 2-8*
10	$(P \vee Q) \to (\neg P \to Q)$	$\to$*I 1-9*

Worked Example: Disjunctive Syllogism [2]

This proof combines:

- Nested assumptions (subproofs within subproofs)
- Case analysis ($\vee$E)
- Ex falso quodlibet ($\bot$E) for the impossible case

Worked Example: Peirce's Law

Example (Peirce's Law — A Classic Challenge): Proving $((P \to Q) \to P) \to P$:

1	$(P \to Q) \to P$	*Assumption*
2	$\neg P$	*Assumption (for RAA)*
3	P	*Assumption (for RAA)*
4	$\bot$	*¬E 2, 3*
5	Q	*⊥E 4*
6	$P \to Q$	*→I 3-5*
7	P	*→E 1, 6*
8	$\bot$	*¬E 2, 7*
9	P	*RAA 2-8*
10	$((P \to Q) \to P) \to P$	*→I 1-9*

Worked Example: Peirce's Law [2]

Peirce's Law is another purely classical theorem. It's equivalent to excluded middle and cannot be proven constructively. It's named after Charles Sanders Peirce (1839–1914).

Derived Rules

Definition 26: *Derived rules* are complex inference patterns provable from basic rules, used as shortcuts.

Modus Tollens:

$$\frac{\varphi \to \psi \quad \neg\psi}{\neg\varphi}$$

Hypothetical Syllogism:

$$\frac{\varphi \to \psi \quad \psi \to \chi}{\varphi \to \chi}$$

Disjunctive Syllogism:

$$\frac{\varphi \lor \psi \quad \neg\varphi}{\psi}$$

Proof by Contradiction (RAA):

$$\frac{[\neg\varphi]\ldots\bot}{\varphi}$$

Assume negation, derive absurdity, conclude original.

Constructive Dilemma:

$$\frac{(\varphi \to \chi) \land (\psi \to \chi) \quad \varphi \lor \psi}{\chi}$$

Logical Fallacies

"The first principle is that you must not fool yourself — and you are the easiest person to fool."

— Richard Feynman

Why Study Fallacies?

Fallacies are reasoning patterns that *look* valid but *aren't*.

They appear everywhere: everyday arguments, political debates, advertising, even academic papers. The trouble is that they can be quite persuasive until you analyze them carefully.

Formal logic gives us precise tools to identify and refute fallacious reasoning.

Our approach:

1. Present the fallacy pattern
2. Give a concrete example
3. Prove formally why it fails (via countermodel)

Fallacy 1: Affirming the Consequent

Definition 27: **Affirming the Consequent** is the invalid inference:

$$\frac{P \to Q \quad Q}{P} \quad \text{✗ INVALID}$$

Example: "If it rains, the ground is wet. The ground is wet. Therefore, it rained."

Counterexample: Someone could have watered the garden!

Formal proof of invalidity:

Find an interpretation where premises are true but conclusion is false:

- Let $\nu(P) = \texttt{false}$, $\nu(Q) = \texttt{true}$
- Then $P \to Q = \texttt{false} \to \texttt{true} = \texttt{true}$ ✓

Fallacy 1: Affirming the Consequent [2]

- And $Q = \texttt{true}$ ✓
- But $P = \texttt{false}$ ✗

The premises don't entail the conclusion: $\{P \to Q, Q\} \not\models P$

Fallacy 2: Denying the Antecedent

Definition 28: **Denying the Antecedent** is the invalid inference:

$$\frac{\begin{array}{l} P \to Q \\ \neg P \end{array}}{\neg Q} \quad \text{✗ INVALID}$$

Example: "If you study hard, you will pass. You didn't study hard. Therefore, you won't pass."

Counterexample: You might be naturally talented, or the exam was easy!

Formal proof of invalidity:

- Let $\nu(P) = \texttt{false}$, $\nu(Q) = \texttt{true}$
- Then $P \to Q = \texttt{true}$ ✓
- And $\neg P = \texttt{true}$ ✓

Fallacy 2: Denying the Antecedent [2]

- But $\neg Q =$ `false` ✗

The conclusion $\neg Q$ is false while premises are true!

Compare with Valid Forms

Modus Ponens (VALID):

$$\frac{P \to Q \quad P}{Q} \quad \checkmark$$

"If P then Q; P; so Q."

Modus Tollens (VALID):

$$\frac{P \to Q \quad \neg Q}{\neg P} \quad \checkmark$$

"If P then Q; not Q; so not P."

Pattern recognition:

- **Valid:** Work with the antecedent (P) directly, or contrapose using the consequent ($\neg Q$)
- **Invalid:** Affirm the consequent (Q) or deny the antecedent ($\neg P$)

Fallacy 3: Undistributed Middle

Definition 29: **Undistributed Middle** (in syllogistic reasoning):

$$\frac{\text{All A are B} \qquad \text{All C are B}}{\text{All A are C}} \quad \text{✗ INVALID}$$

Example: "All cats are mammals. All dogs are mammals. Therefore, all cats are dogs."

Wrong! Yet the *form* looks like valid syllogistic reasoning.

Formal analysis in FOL:

Premises: $\forall x.(A(x) \rightarrow B(x))$ and $\forall x.(C(x) \rightarrow B(x))$

Fallacy 3: Undistributed Middle [2]

Countermodel: Let domain $= \{a, c\}$, with $A(a) = T$, $C(c) = T$, $B(a) = B(c) = T$, and $A(c) = C(a) = F$.

Both premises hold, but $A(a) \to C(a)$ is $T \to F = F$. ✗

Fallacy 4: False Dilemma

Definition 30: **False Dilemma** (or False Dichotomy): Presenting only two options when more exist.

$$\frac{P \vee Q \qquad \neg P}{Q}$$

The inference *is* valid (disjunctive syllogism), but the premise $P \vee Q$ may be *false*!

Example: "You're either with us or against us. You're not with us. Therefore, you're against us."

Problem: One can be neutral, undecided, or have a nuanced position.

Some fallacies aren't about *invalid inference* but about *false premises*.

The form is valid, but the argument is *unsound* because a premise doesn't hold.

Fallacy 5: Begging the Question

Definition 31: **Begging the Question** (circular reasoning): The conclusion is assumed in the premises.

Example: "This medicine works because it's effective, and we know it's effective because it works."

Formally: $P \to Q$ and $Q \to P$ don't establish either P or Q without assuming one.

Formal analysis:

Given only $P \iff Q$:

- $P = Q =$ `true`: consistent ✓
- $P = Q =$ `false`: consistent ✓

Neither follows! We have $\vdash P \iff Q$ but $\nvdash P$ and $\nvdash Q$.

Fallacy 5: Begging the Question [2]

Detection: Remove the conclusion from premises — can they still support it?

Fallacy 6: Existential Fallacy

Definition 32: **Existential Fallacy**: Concluding existence from universal statements about possibly empty classes.

$$\frac{\text{All S are P}}{\text{Some S are P}} \quad \text{✗ INVALID}$$

Example: "All unicorns are magical. Therefore, some unicorns are magical."

If there are no unicorns, the premise is vacuously true but the conclusion is false!

FOL analysis:

- Premise: $\forall x.(S(x) \to P(x))$ — true if S is empty
- Conclusion: $\exists x.(S(x) \wedge P(x))$ — false if S is empty

Countermodel: Domain $= \{a\}$, $S(a) =$ `false`, $P(a) =$ `true`.

Fallacy 6: Existential Fallacy [2]

Premise: $\forall x.(F \to T) = T$. Conclusion: $\exists x.(F \wedge T) = F$.

Proof by Contradiction: Showing Invalidity

General method to prove an inference is invalid:

1. Assume the inference $\Gamma \vDash \varphi$ is valid
2. Construct a *countermodel*: an interpretation where all premises in Γ are true, but φ is false
3. The existence of such a countermodel disproves validity

Example (Affirming the Consequent — Detailed): **Claim:** $\{P \to Q, Q\} \vDash P$ is **invalid**.

Countermodel construction:

- Choose $\nu(P) = 0, \nu(Q) = 1$
- Check premise 1: $\nu(P \to Q) = \nu(\neg P \vee Q) = \max(1, 1) = 1$ ✓
- Check premise 2: $\nu(Q) = 1$ ✓
- Check conclusion: $\nu(P) = 0$ ✗

Premises true, conclusion false → inference invalid.

Valid vs. Invalid Inference Patterns

Pattern	Valid?	Name
$P \to Q, P \therefore Q$	✓	Modus Ponens
$P \to Q, \neg Q \therefore \neg P$	✓	Modus Tollens
$P \to Q, Q \therefore P$	✗	Affirming Consequent
$P \to Q, \neg P \therefore \neg Q$	✗	Denying Antecedent
$P \lor Q, \neg P \therefore Q$	✓	Disjunctive Syllogism
$P \land Q \therefore P$	✓	Simplification

Critical thinking: Identify the logical form, then check if it matches a valid or invalid pattern.

Soundness and Completeness

"The rules of logic are to mathematics
what those of structure are to architecture."

— Bertrand Russell

The Central Question

We have two different ways to characterize "logical truth":

- **Semantic** ($\models$): True in all interpretations (truth tables)
- **Syntactic** ($\vdash$): Derivable using inference rules (proofs)

Do they coincide? This is the central question of metalogic.

If they don't match:

- Some truths are *unprovable* (bad!)
- Some "proofs" lead to *falsehoods* (worse!)
- We couldn't trust either method

If they do match:

- Proofs and truth tables give *same answers*
- We can choose whichever method is easier
- Formal reasoning is *reliable*

This question drove 20th century logic, requiring precise definitions of "proof" and "truth" — the birth of modern mathematical logic.

Soundness: Proofs Never Lie

Definition 33: A proof system is *sound* if every derivable formula is semantically valid:

$$\Gamma \vdash \varphi \quad \rightarrow \quad \Gamma \vDash \varphi$$

In other words: "You can't prove anything false."

Theorem 5 (Soundness of Natural Deduction): Natural deduction for propositional logic is sound: if $\Gamma \vdash \varphi$, then $\Gamma \vDash \varphi$.

Proof strategy: Induction on the *structure of derivations*.

We show that every inference rule *preserves validity*: if the premises of a rule are true under some interpretation, then the conclusion is also true under that interpretation.

Soundness Proof: Base Cases

Proof *(Part 1: Base Cases)*: The derivation uses only a premise or an axiom.

Case: Premise

If $\varphi \in \Gamma$ appears as a line in the derivation, then trivially $\Gamma \vDash \varphi$: any interpretation satisfying all of Γ satisfies φ in particular.

Case: Axiom (if any)

In natural deduction, we have no axioms in the usual sense (only rules). In Hilbert systems, axioms are tautologies, which are valid by definition. □

Soundness Proof: Conjunction Rules

Proof *(Part 2: Conjunction)*: **(∧I) Conjunction Introduction:**

Suppose we have derivations of φ and ψ from Γ, and by IH, $\Gamma \vDash \varphi$ and $\Gamma \vDash \psi$.

Let ν be any interpretation satisfying all formulas in Γ. Then $[\![\varphi]\!]_\nu = \texttt{true}$ and $[\![\psi]\!]_\nu = \texttt{true}$. By definition of $\wedge$: $[\![\varphi \wedge \psi]\!]_\nu = \texttt{true}$.

Therefore $\Gamma \vDash \varphi \wedge \psi$. ✓

(∧E) Conjunction Elimination:

Suppose we have a derivation of $\varphi \wedge \psi$ from Γ, and by IH, $\Gamma \vDash \varphi \wedge \psi$.

Let ν satisfy all of Γ. Then $[\![\varphi \wedge \psi]\!]_\nu = \texttt{true}$. By definition: $[\![\varphi]\!]_\nu = \texttt{true}$ and $[\![\psi]\!]_\nu = \texttt{true}$.

Therefore $\Gamma \vDash \varphi$ and $\Gamma \vDash \psi$. ✓ □

Soundness Proof: Disjunction Rules

Proof *(Part 3: Disjunction)*: **(∨I) Disjunction Introduction:**

Suppose $\Gamma \vDash \varphi$ (by IH). Let ν satisfy Γ. Then $[\![\varphi]\!]_\nu = \texttt{true}$, so $[\![\varphi \vee \psi]\!]_\nu = \texttt{true}$. Similarly for the other direction. ✓

(∨E) Disjunction Elimination:

Suppose $\Gamma \vDash \varphi \vee \psi$, and $\Gamma \cup \{\varphi\} \vDash \chi$, and $\Gamma \cup \{\psi\} \vDash \chi$ (by IH).

Let ν satisfy Γ. Then $[\![\varphi \vee \psi]\!]_\nu = \texttt{true}$.

Case 1: $[\![\varphi]\!]_\nu = \texttt{true}$. Then ν satisfies $\Gamma \cup \{\varphi\}$, so $[\![\chi]\!]_\nu = \texttt{true}$.

Case 2: $[\![\psi]\!]_\nu = \texttt{true}$. Then ν satisfies $\Gamma \cup \{\psi\}$, so $[\![\chi]\!]_\nu = \texttt{true}$.

In both cases, $[\![\chi]\!]_\nu = \texttt{true}$. Therefore $\Gamma \vDash \chi$. ✓ □

Soundness Proof: Implication Rules

Proof *(Part 4: Implication)*: **(→I) Implication Introduction:**

Suppose $\Gamma \cup \{\varphi\} \vDash \psi$ (by IH). We need to show $\Gamma \vDash \varphi \to \psi$.

Let ν satisfy Γ. We consider two cases:

Case 1: $[\![\varphi]\!]_\nu = \texttt{false}$. Then $[\![\varphi \to \psi]\!]_\nu = \texttt{true}$ (false implies anything).

Case 2: $[\![\varphi]\!]_\nu = \texttt{true}$. Then ν satisfies $\Gamma \cup \{\varphi\}$. By IH, $[\![\psi]\!]_\nu = \texttt{true}$. So $[\![\varphi \to \psi]\!]_\nu = \texttt{true}$.

In both cases, $\Gamma \vDash \varphi \to \psi$. ✓

(→E) Implication Elimination (Modus Ponens):

Suppose $\Gamma \vDash \varphi \to \psi$ and $\Gamma \vDash \varphi$ (by IH).

Let ν satisfy Γ. Then $[\![\varphi \to \psi]\!]_\nu = \texttt{true}$ and $[\![\varphi]\!]_\nu = \texttt{true}$.

If $[\![\psi]\!]_\nu = \texttt{false}$, then $[\![\varphi \to \psi]\!]_\nu = [\![\neg\varphi \lor \psi]\!]_\nu = \texttt{false}$. Contradiction.

Therefore $[\![\psi]\!]_\nu = \texttt{true}$, and $\Gamma \vDash \psi$. ✓ □

Soundness Proof: Negation Rules

Proof *(Part 5: Negation)*: **(¬I) Negation Introduction:**

Suppose $\Gamma \cup \{\varphi\} \vDash \bot$ (by IH). We show $\Gamma \vDash \neg\varphi$.

Assume for contradiction that some ν satisfies Γ but $[\![\neg\varphi]\!]_\nu = \texttt{false}$, *i.e.*, $[\![\varphi]\!]_\nu = \texttt{true}$.

Then ν satisfies $\Gamma \cup \{\varphi\}$. By IH, $[\![\bot]\!]_\nu = \texttt{true}$. But $\bot$ is always false! Contradiction.

Therefore no such ν exists, so $\Gamma \vDash \neg\varphi$. ✓

(¬E) Negation Elimination:

Suppose $\Gamma \vDash \varphi$ and $\Gamma \vDash \neg\varphi$ (by IH).

Let ν satisfy Γ. Then $[\![\varphi]\!]_\nu = \texttt{true}$ and $[\![\neg\varphi]\!]_\nu = \texttt{true}$. But $[\![\neg\varphi]\!]_\nu = \neg[\![\varphi]\!]_\nu = \texttt{false}$. Contradiction.

Therefore no ν satisfies Γ, so $\Gamma \vDash \bot$ vacuously. ✓

(⊥E) Ex Falso Quodlibet:

Suppose $\Gamma \vDash \bot$. Then no interpretation satisfies Γ. Therefore $\Gamma \vDash \varphi$ holds vacuously for any φ. ✓ □

Soundness: Summary

We have shown: Every inference rule of natural deduction preserves semantic validity.

By induction on derivation structure, if $\Gamma \vdash \varphi$, then $\Gamma \models \varphi$.

$\therefore$ **Natural deduction is sound.**

Why soundness matters:

- Proofs are *reliable* — they never lead to false conclusions
- Automated provers can be *trusted*
- If $\vdash \bot$, the system is *inconsistent* (useless)

Exam tip: You may be asked to prove soundness of a specific rule. Follow the pattern: assume premises are valid, show conclusion is valid under any interpretation.

Completeness: All Truths Are Provable

Definition 34: A proof system is *complete* if every semantically valid formula is derivable:

$$\Gamma \vDash \varphi \quad \rightarrow \quad \Gamma \vdash \varphi$$

In other words: "Everything true can be proven."

Theorem 6 (Completeness of Propositional Logic): Natural deduction for propositional logic is complete: if $\Gamma \vDash \varphi$, then $\Gamma \vdash \varphi$.

Proof strategy: We prove the *contrapositive*:

If $\Gamma \nvdash \varphi$, then $\Gamma \nvDash \varphi$.

The idea: if we *cannot* prove φ, we can construct a *countermodel* — an interpretation where all of Γ is true but φ is false.

Completeness Proof: Key Concepts

Definition 35: A set of formulas Γ is *consistent* if $\Gamma \nvdash \bot$.

Equivalently: there is no derivation of a contradiction from Γ.

Definition 36: A set of formulas Δ is *maximal consistent* if:

1. Δ is consistent
2. For every formula φ: either $\varphi \in \Delta$ or $\neg\varphi \in \Delta$ (completeness)
3. Adding any formula not in Δ makes it inconsistent (maximality)

A maximal consistent set "decides" every formula — it contains exactly one of φ or $\neg\varphi$ for each φ.

This is powerful: we can define an interpretation directly from membership in Δ!

Completeness Proof: Lindenbaum's Lemma

Theorem 7 (Lindenbaum's Lemma): Every consistent set Γ can be extended to a maximal consistent set $\Delta \supseteq \Gamma$.

Proof *(constructive)*: Let $\varphi_1, \varphi_2, \varphi_3, \ldots$ be an enumeration of all formulas.

Define a sequence of sets:

- $\Gamma_0 = \Gamma$
- $\Gamma_{n+1} = \begin{cases} \Gamma_n \cup \{\varphi_{n+1}\} & \text{if this is consistent} \\ \Gamma_n \cup \{\neg\varphi_{n+1}\} & \text{otherwise} \end{cases}$
- $\Delta = \bigcup_{n=0}^{\infty} \Gamma_n$

Claim 1: Each Γ_n is consistent (by induction).

Claim 2: Δ is consistent (a derivation uses only finitely many formulas).

Claim 3: Δ is maximal (every φ_n or $\neg\varphi_n$ was added at step n). □

Completeness Proof: Canonical Model

Definition 37: Given a maximal consistent set Δ, define the *canonical interpretation* ν_Δ:

$$\nu_\Delta(P) = \begin{cases} \texttt{true} & \text{if } P \in \Delta \\ \texttt{false} & \text{if } P \notin \Delta \end{cases}$$

for each propositional variable P.

Theorem 8 (Truth Lemma): For any formula φ and maximal consistent set Δ:

$$[\![\varphi]\!]_\nu = \texttt{true} \quad \text{iff} \quad \varphi \in \Delta$$

This is the heart of the completeness proof! It says the canonical interpretation "agrees" with membership in Δ for *all* formulas, not just atoms.

Completeness Proof: Truth Lemma

Proof: By structural induction on φ.

Base case: $\varphi = P$ (atomic). By definition of ν_Δ. ✓

Case $\varphi = \neg\psi$: $[\![\neg\psi]\!]_\nu = \texttt{true}$ iff $[\![\psi]\!]_\nu = \texttt{false}$ iff $\psi \notin \Delta$ (IH) iff $\neg\psi \in \Delta$ (maximality). ✓

Case $\varphi = \psi \wedge \chi$:

- ($\Rightarrow$) If $\psi \wedge \chi \in \Delta$, then $\psi \in \Delta$ and $\chi \in \Delta$ (by $\wedge$E being sound and Δ being maximal consistent). By IH, $[\![\psi]\!]_\nu = [\![\chi]\!]_\nu = \texttt{true}$, so $[\![\psi \wedge \chi]\!]_\nu = \texttt{true}$.
- ($\Leftarrow$) If $[\![\psi \wedge \chi]\!]_\nu = \texttt{true}$, then $[\![\psi]\!]_\nu = [\![\chi]\!]_\nu = \texttt{true}$. By IH, $\psi, \chi \in \Delta$. Since Δ is closed under derivability and $\psi, \chi \vdash \psi \wedge \chi$, we have $\psi \wedge \chi \in \Delta$. ✓

(see next page)

Completeness Proof: Truth Lemma [2]

Case $\varphi = \psi \vee \chi$:

- ($\Rightarrow$) If $\psi \vee \chi \in \Delta$, suppose for contradiction that $\psi \notin \Delta$ and $\chi \notin \Delta$. By maximality, $\neg\psi \in \Delta$ and $\neg\chi \in \Delta$. But then $\Delta \vdash \psi \vee \chi$ and $\Delta \vdash \neg\psi$ and $\Delta \vdash \neg\chi$, which by $\vee$E gives $\Delta \vdash \bot$. Contradiction with consistency!
- ($\Leftarrow$) If $[\![\psi \vee \chi]\!]_\nu = \texttt{true}$, then $[\![\psi]\!]_\nu = \texttt{true}$ or $[\![\chi]\!]_\nu = \texttt{true}$. By IH, $\psi \in \Delta$ or $\chi \in \Delta$. By $\vee$I, $\psi \vee \chi \in \Delta$.

Case $\varphi = \psi \to \chi$:

- ($\Rightarrow$) If $\psi \to \chi \in \Delta$ and $[\![\psi]\!]_\nu = \texttt{true}$, then by IH $\psi \in \Delta$. By $\to$E, $\chi \in \Delta$, so $[\![\chi]\!]_\nu = \texttt{true}$.
- ($\Leftarrow$) If $[\![\psi \to \chi]\!]_\nu = \texttt{true}$: either $[\![\psi]\!]_\nu = \texttt{false}$ or $[\![\chi]\!]_\nu = \texttt{true}$. In either case, by maximality and closure under derivability, $\psi \to \chi \in \Delta$.

All cases complete. □

Completeness Proof: Main Argument

Theorem 9 (Completeness): If $\Gamma \vDash \varphi$, then $\Gamma \vdash \varphi$.

Proof: We prove the contrapositive: if $\Gamma \nvdash \varphi$, then $\Gamma \nvDash \varphi$.

Assume $\Gamma \nvdash \varphi$. Then $\Gamma \cup \{\neg\varphi\}$ is consistent (if not, we could derive φ by RAA).

By Lindenbaum's Lemma, extend $\Gamma \cup \{\neg\varphi\}$ to a maximal consistent set Δ.

Define the canonical interpretation ν_Δ.

By the Truth Lemma:

- For each $\gamma \in \Gamma$: since $\gamma \in \Delta$, we have $[\![\gamma]\!]_\nu = \texttt{true}$
- Since $\neg\varphi \in \Delta$, we have $[\![\neg\varphi]\!]_\nu = \texttt{true}$, *i.e.*, $[\![\varphi]\!]_\nu = \texttt{false}$

Therefore ν_Δ satisfies Γ but falsifies φ.

This means $\Gamma \nvDash \varphi$. □

The Completeness Theorem: Full Statement

Theorem 10: In *propositional logic*, for any set of formulas Γ and formula φ:

$$\Gamma \vDash \varphi \quad \text{iff} \quad \Gamma \vdash \varphi$$

Practical implications:

- Automated theorem provers are *theoretically sound*
- Truth table methods and proof methods are *equivalent*
- Proof search is *as hard as SAT* (NP-complete)

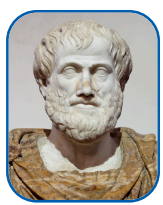
Beyond Propositional Logic: This perfect correspondence doesn't always hold:

- **First-order logic**: Complete (Gödel 1929) but undecidable (Church 1936)
- **Second-order logic**: Incomplete (no proof system captures all valid formulas)
- **Arithmetic**: Incomplete (Gödel's Incompleteness Theorems, 1931)

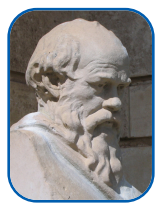
Categorical Logic

"All men are mortal. Socrates is a man. Therefore, Socrates is mortal."

— Classical syllogism



Aristotle



Socrates

George Boole

From Propositional to Categorical

Classical propositional logic treats statements as atomic units.

But human reasoning often involves *relationships between classes* of objects:

- "All birds can fly"
- "Some mammals are aquatic"
- "No reptiles are warm-blooded"

Traditional logic studies these patterns, providing a bridge to modern predicate logic.

Categorical Propositions

Definition 38: A *categorical proposition* is a statement that asserts or denies a relationship between two *categories* (classes) of objects.

Every categorical proposition has:

- *Subject term* (S): the category being described
- *Predicate term* (P): the category used in the description
- *Quantifier*: indicates how much of the subject is included
- *Quality*: affirmative or negative

Example: "All *politicians* are *corrupt*."

- Subject: politicians
- Predicate: corrupt people
- Quantifier: all (universal)
- Quality: affirmative

The Four Standard Forms

Definition 39: Traditional logic recognizes *four standard forms* of categorical propositions:

Form	Quantifier	Quality	Structure	Example
A	Universal	Affirmative	All S are P	"All cats are mammals"
E	Universal	Negative	No S are P	"No fish are mammals"
I	Particular	Affirmative	Some S are P	"Some birds are flightless"
O	Particular	Negative	Some S are not P	"Some animals are not vertebrates"

Examples of Categorical Propositions

A (Universal Affirmative):

- All students are hardworking
- Every theorem has a proof
- All prime numbers except 2 are odd

E (Universal Negative):

- No circles are squares
- No valid argument has false premises and true conclusion
- No even number greater than 2 is prime

I (Particular Affirmative):

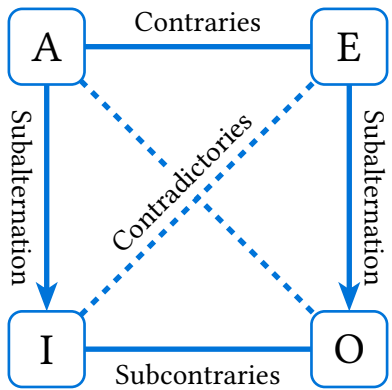
- Some politicians are honest
- Some functions are continuous
- Some equations have multiple solutions

O (Particular Negative):

- Some students are not prepared
- Some triangles are not right triangles
- Some numbers are not rational

The Square of Opposition

Definition 40: A *square of opposition* is a diagram showing the logical relationships between A, E, I, and O propositions with the same subject and predicate terms.



Logical Relationships in the Square

The square captures four relationships:

Contradictories (A–O, E–I):
- Cannot both be true
- Cannot both be false
- Exactly one must be true

Subcontraries (I–O):
- Cannot both be false
- Can both be true
- At least one is true

Contraries (A–E):
- Cannot both be true
- Can both be false
- At most one is true

Subalternation ($A \rightarrow I$, $E \rightarrow O$):
- If universal is true, particular is true
- If particular is false, universal is false

Logical Relationships in the Square [2]

Example: Given: “All roses are flowers” (A-form, `true`)

By the square of opposition:

- “No roses are flowers” (E-form) is `false` (contraries)
- “Some roses are flowers” (I-form) is `true` (subalternation)
- “Some roses are not flowers” (O-form) is `false` (contradictories)

Translation Between Traditional and Modern Logic

Categorical propositions can be translated into first-order logic:

Traditional	Modern Logic	Reading
All S are P	$\forall x(S(x) \rightarrow P(x))$	"For all x, if x is S then x is P"
No S are P	$\forall x(S(x) \rightarrow \neg P(x))$	"For all x, if x is S then x is not P"
Some S are P	$\exists x(S(x) \wedge P(x))$	"There exists x such that x is S and x is P"
Some S are not P	$\exists x(S(x) \wedge \neg P(x))$	"There exists x such that x is S and x is not P"

Example: "All students are hardworking" becomes: $\forall x(\text{Student}(x) \rightarrow \text{Hardworking}(x))$

"Some politicians are not honest" becomes: $\exists x(\text{Politician}(x) \wedge \neg\ \text{Honest}(x))$

The Existential Import Problem

Definition 41: A proposition "S is P" has *existential import* if it implies the existence of objects (at least one) in its subject class S.

The Problem:

Traditional logic (Aristotle) assumes all categorical propositions have existential import.

Modern logic questions this assumption.

Consider: "All unicorns are magical"

- Traditional: Implies unicorns exist (so the statement is false)
- Modern: True vacuously (if there are no unicorns, the implication holds trivially)

The Existential Import Problem [2]

Example (Impact on the Square): In modern logic with empty domains:

- A and E can both be true (if subject class is empty)
- I and O can both be false (if subject class is empty)
- Subalternation fails (A can be true while I is false)

The traditional square of opposition only works when we assume non-empty subject classes.

Syllogisms: Reasoning with Categories

Definition 42: *Categorical syllogism* is a form of reasoning with three categorical propositions:

- *Major premise*: contains the predicate of the conclusion
- *Minor premise*: contains the subject of the conclusion
- *Conclusion*: derived from the premises

Uses exactly three terms: major, minor, and middle.

Example (Classic syllogism):

All humans are mortal	*(Major premise)*
Socrates is human	*(Minor premise)*
Therefore, Socrates is mortal	*(Conclusion)*

Terms:

- Major term: mortal (P)
- Minor term: Socrates (S)
- Middle term: human (M)

All M are P	**(A)**
All S are M	**(A)**
All S are P	**(A)**

Syllogistic Forms and Validity

Traditional logic identified **24 valid syllogistic forms** across four figures.

Each valid form has a traditional Latin name encoding its mood (vowels = A, E, I, O):

Example (Famous Valid Forms):

- **Barbara** (AAA-1): All M are P, All S are M ∴ All S are P
- **Celarent** (EAE-1): No M are P, All S are M ∴ No S are P
- **Darii** (AII-1): All M are P, Some S are M ∴ Some S are P

Common Fallacies:

- **Undistributed Middle**: "All A are B, All C are B ∴ All A are C"
- **Four Terms**: Equivocation on word meaning
- **Existential Fallacy**: Particular conclusion from universal premises

Note: The names A, E, I, O come from Latin vowels in *affirmo* (I affirm) and *nego* (I deny).

From Categorical Logic to First-Order Logic

Traditional categorical logic has important *limitations*:

1. Only handles **simple quantification** (all, some, no)
2. **Cannot express relations**: "John is taller than Mary"
3. **Limited to two categories** per proposition
4. **No nested quantifiers**: "Every student likes some professor"
5. **Existential import** controversies

Example (What categorical logic cannot express):

- $\forall x.\exists y.R(x, y)$ — "Everyone has someone who loves them"
- $\forall x.(P(x) \to \exists y.Q(x, y))$ — "Every problem has a solution"
- Transitive closure, recursion, arithmetic

These limitations motivate **first-order logic** (predicate logic), which we study next.

First-Order Logic

"The limits of my language mean the limits of my world."

— Ludwig Wittgenstein

Limitations of Propositional Logic

Propositional logic treats statements as *atomic units* — we cannot analyze their internal structure.

Example: The syllogism below is valid, yet propositional logic cannot capture *why*:

All humans are mortal.

Socrates is human.

$\therefore$ Socrates is mortal.

In propositional logic, these are just unrelated atoms P, Q, R.

The validity depends on the *internal structure*: objects (Socrates), properties (human, mortal), and quantification (all).

What Propositional Logic Cannot Express

Objects and naming:
- "Socrates"
- "the number 7"
- "the empty set"

Properties of objects:
- "x is prime"
- "x is human"

Relations between objects:
- "x is greater than y"
- "x divides y"

Quantification:
- "all", "some", "none"
- "there exists exactly one"

Definition 43: *First-order logic* (FOL), also called *predicate logic*, extends propositional logic with variables, predicates, functions, and quantifiers.

First-Order Logic: Components

Definition 44: A *first-order language* consists of:

- **Variables**: $x, y, z, \ldots$ — range over objects in a domain
- **Constants**: $a, b, c, \ldots$ — name specific objects
- **Predicate symbols**: $P, Q, R, \ldots$ — express properties and relations
- **Function symbols**: $f, g, h, \ldots$ — map objects to objects
- **Quantifiers**: $\forall$ (universal) and $\exists$ (existential)
- **Logical connectives**: $\neg, \wedge, \vee, \rightarrow, \iff$ (as in propositional logic)

Note: *Why "first-order"?* Variables range over *objects* (first-order entities).

In second-order logic, we can also quantify over properties and relations.

Terms

Definition 45: A *term* is an expression denoting an object, defined recursively:

1. Every variable $x, y, z, \ldots$ is a term
2. Every constant $a, b, c, \ldots$ is a term
3. If f is an n-ary function symbol and $t_1, \ldots, t_n$ are terms, then $f(t_1, \ldots, t_n)$ is a term

Example: In arithmetic with constants $0, 1$, function S, and variables x, y:

- 0, 1, x, y are terms (constants and variables)
- $S(0)$, $S(x)$ are terms (function application)
- $S(S(0))$ is a term (nested application)

Note: Terms denote *objects*, not truth values. The term $S(x)$ is an object, not a statement.

Functions vs. Predicates

Functions (f, g, h):

- Map objects to *objects*
- Used inside terms
- Example: $\text{fatherOf}(x)$, $x + y$
- "The father of Alice" is an *object*.

Predicates (P, Q, R):

- Map objects to *truth values*
- Used to build formulas
- Example: $\text{IsHuman}(x)$, $x < y$
- "Alice is human" is a *statement* (T/F).

Note: A function $f(x) = y$ can always be represented as a predicate $R(x, y)$ with an additional uniqueness axiom: $\forall x.\exists! y.\, R(x, y)$.

Atomic Formulas

Definition 46: An *atomic formula* is a basic statement:

1. **Predicate application**: $P(t_1, ..., t_n)$ where P is an n-ary predicate and t_i are terms
2. **Equality**: $t_1 = t_2$ where t_1, t_2 are terms

Example:

- $\text{Human}(\text{socrates})$ — "Socrates is human"
- $\text{Less}(x, y)$ — "x is less than y"
- $\text{Prime}(S(S(0)))$ — "2 is prime"
- $\text{father}(x) = y$ — "The father of x is y"

Equality and its Properties

In FOL, equality ($=$) is a special predicate with fixed meaning: "is the same object as".

Theorem 11 (Axioms of Equality):

1. **Reflexivity**: $\forall x.\,(x = x)$
2. **Symmetry**: $\forall x, y.\,(x = y) \to (y = x)$
3. **Transitivity**: $\forall x, y, z.\,(x = y) \land (y = z) \to (x = z)$
4. **Substitution**: $\forall x, y.\,(x = y) \to (\varphi(x) \iff \varphi(y))$

Note: Substitution means if two objects are equal, they share all properties. This is *Leibniz's Law*.

First-Order Formulas

Definition 47: A *first-order formula* is defined recursively:

1. Every atomic formula is a formula
2. If φ is a formula, then $(\neg\varphi)$ is a formula
3. If φ, ψ are formulas, then $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \to \psi)$, $(\varphi \iff \psi)$ are formulas
4. If φ is a formula and x is a variable, then $(\forall x.\, \varphi)$ and $(\exists x.\, \varphi)$ are formulas

Example:

- $\forall x.\, \text{Human}(x) \to \text{Mortal}(x)$ — "All humans are mortal"
- $\exists x.\, \text{Prime}(x) \wedge (x > 100)$ — "Some prime is greater than 100"
- $\forall x. \exists y.\, (y > x)$ — "For every number, there is a greater one"

The Quantifiers

Definition 48: The *universal quantifier* $\forall x.\, \varphi(x)$ asserts that $\varphi(x)$ holds for *every* object x in the domain.

Semantically equivalent to a (possibly infinite) conjunction:

$$\forall x.\, P(x) \approx P(a_1) \wedge P(a_2) \wedge P(a_3) \wedge \ldots$$

Definition 49: The *existential quantifier* $\exists x.\, \varphi(x)$ asserts that $\varphi(x)$ holds for *at least one* object x in the domain.

Semantically equivalent to a (possibly infinite) disjunction:

$$\exists x.\, P(x) \approx P(a_1) \vee P(a_2) \vee P(a_3) \vee \ldots$$

Quantifier Examples

Example: Let domain $D = \{1, 2, 3\}$ and $\text{Even}(x)$ mean "x is even".

- $\forall x.\, \text{Even}(x)$ means $\text{Even}(1) \wedge \text{Even}(2) \wedge \text{Even}(3)$ — **False**
- $\exists x.\, \text{Even}(x)$ means $\text{Even}(1) \vee \text{Even}(2) \vee \text{Even}(3)$ — **True**

Example: In the domain of natural numbers $\mathbb{N}$:

- $\forall x. \exists y.\, (y > x)$ — "Every number has a greater one" — **True**
- $\exists y. \forall x.\, (y > x)$ — "Some number is greater than all" — **False**

The order of quantifiers matters! Swapping $\forall$ and $\exists$ changes the meaning.

Nested Quantifiers

The pattern $\forall x.\exists y.\, R(x,y)$ vs. $\exists y.\forall x.\, R(x,y)$ is fundamental.

Example: Let $L(x,y)$ mean "x loves y".

Formula	Meaning
$\forall x.\exists y.\, L(x,y)$	Everyone loves someone (possibly different)
$\exists y.\forall x.\, L(x,y)$	Someone is loved by everyone (same person)

Note: $\exists y.\forall x.\, R(x,y)$ implies $\forall x.\exists y.\, R(x,y)$, but not conversely.

Free and Bound Variables

Definition 50: An occurrence of variable x in formula φ is:

- **Bound** if it is in the scope of a quantifier $\forall x$ or $\exists x$
- **Free** if it is not bound

The *scope* of $\forall x$ (or $\exists x$) in $\forall x.\, \varphi$ is the subformula φ.

Example: In the formula $P(x) \wedge \forall x.\, Q(x)$:

- The x in $P(x)$ is **free** (not under any quantifier)
- The x in $Q(x)$ is **bound** by $\forall x$

These are *different* occurrences — the same name, different roles.

Sentences and Open Formulas

Definition 51:

- A *sentence* (closed formula) has no free variables
- An *open formula* has at least one free variable

Example:

Formula	Free vars	Type
$\forall x.\, (P(x) \to Q(x))$	none	Sentence
$P(x) \land Q(y)$	x, y	Open
$\exists x.\, R(x, y)$	y	Open
$\forall x. \exists y.\, R(x, y)$	none	Sentence

Note: Only sentences have definite truth values in a structure. Open formulas require a variable assignment.

Formalization: Vocabulary

To translate natural language into FOL, we must first define a *vocabulary* (or *signature*) that captures the relevant objects and relations.

Definition 52: A *vocabulary* Σ consists of:

- **Constant symbols**: Names for specific objects (*e.g.*, a, b, c)
- **Predicate symbols**: Properties or relations (*e.g.*, P, Q, R)
- **Function symbols**: Operations that return objects (*e.g.*, f, g, h)

Note: Each predicate and function symbol has a fixed *arity* (number of arguments).

Example: Choosing a Vocabulary

Example (Social Network):

- **Constants**: alice, bob, carol
- **Predicates**: $\text{Person}(x)$, $\text{Friends}(x, y)$, $\text{Posted}(x, p)$
- **Functions**: $\text{profileOf}(x)$ returns the profile of person x

Example (Arithmetic):

- **Constants**: $0, 1$
- **Predicates**: $x < y$, $x = y$
- **Functions**: $x + y$, $x \times y$, $S(x)$ (successor)

The Two Key Translation Patterns

The Golden Rules:

"All A are B" uses $\rightarrow$:

$$\forall x.\, A(x) \rightarrow B(x)$$

Read: "For anything, *if* it's an A, *then* it's a B."

"Some A are B" uses $\wedge$:

$$\exists x.\, A(x) \wedge B(x)$$

Read: "There's something that's *both* an A *and* a B."

Common Pitfall: $\forall x.\, A(x) \wedge B(x)$ means "Everything in the universe is both A and B."

Example: $\forall x.\, \text{Cat}(x) \wedge \text{Mammal}(x)$ would mean "Everything is a cat and a mammal" (including the moon, numbers, and this slide).

Translation Examples

Example: Let $H(x)$ = "x is human", $M(x)$ = "x is mortal", s = Socrates

English	**FOL**
All humans are mortal	$\forall x.\,[H(x) \to M(x)]$
Some humans are mortal	$\exists x.\,[H(x) \land M(x)]$
No humans are immortal	$\forall x.\,[H(x) \to M(x)]$ or $\neg\exists x.\,[H(x) \land \neg M(x)]$
Socrates is human	$H(s)$
Not all humans are mortal	$\exists x.\,[H(x) \land \neg M(x)]$

Translation: "Only" and Uniqueness

Example:

English	FOL
Only humans are rational	$\forall x.\, [R(x) \to H(x)]$
At least one student passed	$\exists x.\, [S(x) \land P(x)]$
At most one student passed	$\forall x, y.\, [(S(x) \land P(x) \land S(y) \land P(y)) \to x = y]$

Note: **"Only A are B"** = **"All B are A"** — the direction reverses!

"Only dogs bark" means "if it barks, it's a dog" (not "all dogs bark").

Definition 53: The *uniqueness quantifier* $\exists!x.\, \varphi(x)$ means "exactly one x satisfies φ":

$$\exists!x.\, \varphi(x) \quad \equiv \quad \exists x.\, (\varphi(x) \land \forall y.\, [\varphi(y) \to (y = x)])$$

Translation: Complex Examples

Example: Let $M(x, y)$ mean "x is the mother of y".

English	FOL
Everyone has a mother	$\forall x.\exists y.\, M(y, x)$
Everyone has exactly one mother	$\forall x.\exists! y.\, M(y, x)$
Some people are mothers	$\exists x.\exists y.\, M(x, y)$
Mothers are not their own mothers	$\forall x.\forall y.\, [M(x, y) \rightarrow (x \neq y)]$
Grandmother is mother's mother	$G(x, z) \iff \exists y.\, [M(x, y) \land M(y, z)]$

The Socrates Syllogism in FOL

Now we can formalize the syllogism that propositional logic could not handle:

Example:

Natural language:	**First-order logic:**
All humans are mortal.	$\forall x.\, [H(x) \to M(x)]$
Socrates is human.	$H(s)$
$\therefore$ Socrates is mortal.	$\therefore M(s)$

Derivation: Instantiate the universal with $x = s$ to get $H(s) \to M(s)$. Apply modus ponens with $H(s)$. ✓

Structures (Interpretations)

Definition 54: A *structure* $\mathcal{M} = \langle D, \mathcal{I} \rangle$ consists of:

- **Domain** D: a non-empty set of objects
- **Interpretation** $\mathcal{I}$ assigns meaning to symbols:
  - Each constant c maps to an element $\mathcal{I}(c) \in D$
  - Each n-ary predicate P maps to a relation $\mathcal{I}(P) \subseteq D^n$
  - Each n-ary function f maps to a function $\mathcal{I}(f) : D^n \to D$

Example: Structure for arithmetic: $D = \mathbb{N}$, $\mathcal{I}(0) = 0$, $\mathcal{I}(S)(n) = n + 1$, $\mathcal{I}(\text{Even}) = \{0, 2, 4, ...\}$

Variable Assignments and Satisfaction

Definition 55: A *variable assignment* $\sigma : \text{Var} \to D$ maps variables to domain elements.

Notation: $\sigma[x \mapsto d]$ is like σ but maps x to d.

Definition 56: The *satisfaction relation* $\mathcal{M}, \sigma \vDash \varphi$ is defined recursively:

- $\mathcal{M}, \sigma \vDash P(t_1, ..., t_n)$ iff $([\![t_1]\!]_\nu, ..., [\![t_n]\!]_\nu) \in \mathcal{I}(P)$
- $\mathcal{M}, \sigma \vDash \forall x.\, \varphi$ iff $\mathcal{M}, \sigma[x \mapsto d] \vDash \varphi$ for *all* $d \in D$
- $\mathcal{M}, \sigma \vDash \exists x.\, \varphi$ iff $\mathcal{M}, \sigma[x \mapsto d] \vDash \varphi$ for *some* $d \in D$

Example: Evaluating Formulas

Example: Let $D = \{1, 2, 3\}$ and $\mathcal{I}(\text{Even}) = \{2\}$.

Is $\exists x.\, \text{Even}(x)$ true?

- Try $x = 1$: $1 \in \{2\}$? ✗
- Try $x = 2$: $2 \in \{2\}$? ✓ — witness found!

Result: $\mathcal{M} \vDash \exists x.\, \text{Even}(x)$ ✓

Example: **Is $\forall x.\, \text{Even}(x)$ true?**

- Try $x = 1$: $1 \in \{2\}$? ✗ — counterexample found!

Result: $\mathcal{M} \nvDash \forall x.\, \text{Even}(x)$ ✗

Validity and Satisfiability

Definition 57:

- φ is *valid* if $\mathcal{M} \vDash \varphi$ for every structure $\mathcal{M}$
- φ is *satisfiable* if $\mathcal{M} \vDash \varphi$ for some structure $\mathcal{M}$
- φ is *unsatisfiable* if no structure satisfies it

Example (Valid FOL Formulas):

- $\forall x.\, P(x) \to P(a)$ (universal instantiation)
- $P(a) \to \exists x.\, P(x)$ (existential generalization)
- $\forall x.\, P(x) \to \exists x.\, P(x)$ (non-empty domain)

Quantifier Laws: Negation

Theorem 12 (De Morgan Laws for Quantifiers):

$$\begin{aligned} \neg\forall x.\, \varphi(x) &\equiv \exists x.\, \neg\varphi(x) \\ \neg\exists x.\, \varphi(x) &\equiv \forall x.\, \neg\varphi(x) \end{aligned}$$

Example:

- "Not everyone passed" ≡ "Someone didn't pass"
- "Nobody passed" ≡ "Everyone didn't pass"

Quantifier Laws: Distribution

Theorem 13:

$$\begin{aligned} \forall x.\,(\varphi(x) \wedge \psi(x)) &\equiv (\forall x.\,\varphi(x)) \wedge (\forall x.\,\psi(x)) \\ \exists x.\,(\varphi(x) \vee \psi(x)) &\equiv (\exists x.\,\varphi(x)) \vee (\exists x.\,\psi(x)) \end{aligned}$$

Warning: These do *not* hold in general:

$$\begin{aligned} \forall x.\,(\varphi(x) \vee \psi(x)) &\not\equiv (\forall x.\,\varphi(x)) \vee (\forall x.\,\psi(x)) \\ \exists x.\,(\varphi(x) \wedge \psi(x)) &\not\equiv (\exists x.\,\varphi(x)) \wedge (\exists x.\,\psi(x)) \end{aligned}$$

Quantifier Laws: Moving Quantifiers

If x is *not free* in ψ, we can move the quantifier:

Theorem 14:

$$
\begin{aligned}
(\forall x.\, \varphi(x)) \wedge \psi &\equiv \forall x.\, (\varphi(x) \wedge \psi) \\
(\exists x.\, \varphi(x)) \wedge \psi &\equiv \exists x.\, (\varphi(x) \wedge \psi) \\
(\forall x.\, \varphi(x)) \vee \psi &\equiv \forall x.\, (\varphi(x) \vee \psi) \\
(\exists x.\, \varphi(x)) \vee \psi &\equiv \exists x.\, (\varphi(x) \vee \psi)
\end{aligned}
$$

Note: This is the basis for converting formulas to **Prenex Normal Form**.

Prenex Normal Form

Definition 58: A formula is in *prenex normal form* (PNF) if all quantifiers appear at the front:

$$Q_1 x_1 . Q_2 x_2 . \ldots Q_n x_n . \psi$$

where each $Q_i \in \{\forall, \exists\}$ and ψ is a quantifier-free *matrix*.

Theorem 15: Every FOL formula is equivalent to one in prenex normal form.

Example:

$$\begin{aligned}
&\forall x . P(x) \to \exists y . Q(y) \\
&\equiv \neg \forall x . P(x) \lor \exists y . Q(y) \\
&\equiv \exists x . \neg P(x) \lor \exists y . Q(y) \\
&\equiv \exists x . \exists y . (\neg P(x) \lor Q(y))
\end{aligned}$$

Theories and Models

Definition 59: A *theory* T is a set of FOL sentences (axioms).

A structure $\mathcal{M}$ is a *model* of T if $\mathcal{M} \vDash \varphi$ for all $\varphi \in T$.

Example (The Theory of Groups): Vocabulary: binary function $\cdot$, constant e

Axioms:

1. $\forall x, y, z.\, (x \cdot (y \cdot z)) = ((x \cdot y) \cdot z)$ (associativity)
2. $\forall x.\, (x \cdot e = x) \wedge (e \cdot x = x)$ (identity)
3. $\forall x. \exists y.\, (x \cdot y = e) \wedge (y \cdot x = e)$ (inverses)

Models: $(\mathbb{Z}, +, 0)$, $(\mathbb{R} \setminus \{0\}, \times, 1)$, symmetry groups, ...

Example: Peano Arithmetic (PA)

Example: Vocabulary: $0, S, +, \times$

Axioms:

1. $\forall x.\, S(x) \neq 0$
2. $\forall x, y.\, (S(x) = S(y) \to x = y)$
3. $\forall x.\, x + 0 = x$
4. $\forall x, y.\, x + S(y) = S(x + y)$
5. **Induction Schema**: For any formula φ: $(\varphi(0) \land \forall x.\, (\varphi(x) \to \varphi(S(x)))) \to \forall x.\, \varphi(x)$

Note: PA is the standard theory for natural numbers. Gödel's Incompleteness Theorem shows that PA cannot prove all true statements about $\mathbb{N}$.

Natural Deduction for FOL

Natural deduction extends to FOL with rules for quantifiers:

Universal Introduction (∀I):

$$\frac{\varphi(a) \quad \textit{(a fresh)}}{\forall x.\, \varphi(x)}$$

Existential Introduction (∃I):

$$\frac{\varphi(t) \quad \textit{(witness } t\textit{)}}{\exists x.\, \varphi(x)}$$

Universal Elimination (∀E):

$$\frac{\forall x.\, \varphi(x)}{\varphi(t) \quad \textit{(any term } t\textit{)}}$$

Existential Elimination (∃E):

$$\frac{\exists x.\, \varphi(x), \;\; [\varphi(a)] \ldots \psi}{\psi \quad \textit{(a fresh)}}$$

Proof Example

Example: **Prove:** $\{\forall x.\,(H(x) \to M(x)), H(s)\} \vdash M(s)$

1.	$\forall x.\,(H(x) \to M(x))$	Premise
2.	$H(s)$	Premise
3.	$H(s) \to M(s)$	$\forall$E on 1
4.	$M(s)$	$\to$E on 2, 3

Soundness and Completeness

Theorem 16 (Soundness): If $\Gamma \vdash \varphi$, then $\Gamma \vDash \varphi$.

Theorem 17 (Completeness (Gödel, 1929)): If $\Gamma \vDash \varphi$, then $\Gamma \vdash \varphi$.

Significance: Semantic truth ($\vDash$) and syntactic proof ($\vdash$) are perfectly aligned in FOL.

Every provable statement is true, and every true statement is provable.

Compactness Theorem

Theorem 18 (Compactness): A set of sentences Γ is satisfiable if and only if every *finite* subset of Γ is satisfiable.

The Compactness Theorem implies that First-Order Logic **cannot** distinguish between "arbitrarily large finite" and "infinite".

Finite to Infinite: If a property holds for all finite graphs, it may fail for infinite ones.

Example: "Every node has exactly one successor $\Rightarrow$ there is a cycle."

- *Finite*: Always true.
- *Infinite*: False for $(\mathbb{Z}, n \mapsto n+1)$.

Infinite to Finite: If a property holds for all infinite graphs, it may fail for finite ones.

Example: "There is no maximum element."

- *Infinite*: True for $(\mathbb{Z}, <)$.
- *Finite*: Always false (every finite linear order has a max).

Key Insight: If a set of sentences has arbitrarily large finite models, it **must** have an infinite model.

Example: k-Colorability

Compactness is often used to extend properties from finite to infinite structures.

Theorem 19 (De Bruijn–Erdős): An infinite graph G is k-colorable if and only if every *finite* subgraph of G is k-colorable.

Proof:

1. Let Γ be a set of sentences describing G and assigning a color $\{1, \ldots, k\}$ to each vertex such that adjacent vertices have different colors.
2. Any *finite* subset of Γ only involves a finite number of vertices and edges.
3. This finite subgraph is k-colorable by assumption.
4. By Compactness, the entire set Γ is satisfiable.
5. Thus, the infinite graph G is k-colorable.

□

Löwenheim–Skolem Theorem

Theorem 20 (Löwenheim–Skolem): If a countable theory has an infinite model, then it has models of *every* infinite cardinality.

The Skolem Paradox: Set theory (ZFC) can be expressed in FOL and has a countable model, even though it proves the existence of uncountable sets!

Church–Turing Undecidability

Theorem 21 (Undecidability (Church, Turing, 1936)): First-order logic is *undecidable*: there is no algorithm that determines whether an arbitrary FOL sentence is valid.

This is a fundamental limit, not a matter of finding better algorithms.

However, many useful *fragments* remain decidable (propositional, monadic, EPR, …).

Decidable Fragments

Fragment	Restriction	Decidable?	Complexity
Propositional	No quantifiers	✓	NP-complete
Monadic FOL	Unary predicates only	✓	NEXPTIME
Two-variable FOL	≤ 2 variables	✓	NEXPTIME
EPR	$\exists^*\forall^*$, no functions	✓	NEXPTIME
Full FOL	Unrestricted	✗	Undecidable

Note: SAT/SMT solvers exploit decidable fragments for verification, planning, and constraint solving.

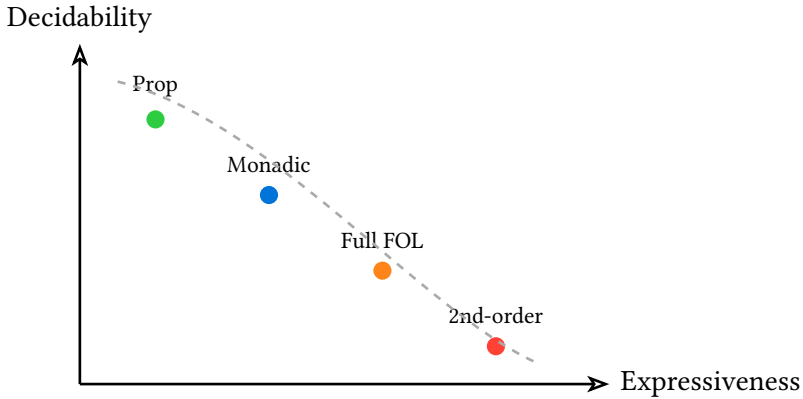
Gödel's Incompleteness (Preview)

Theorem 22 (First Incompleteness (Gödel, 1931)): Any consistent formal system capable of expressing arithmetic is *incomplete*: some true statements are unprovable.

Incompleteness ≠ Undecidability:

- Undecidability: no algorithm for validity
- Incompleteness: no proof system proves all truths about arithmetic

The Expressiveness–Decidability Trade-off



More expressive logics are harder (or impossible) to decide.

Applications of FOL

Program Verification:

- Hoare logic: $\{P\}\ C\ \{Q\}$
- Loop invariants
- Tools: Dafny, Frama-C

Databases:

- SQL WHERE $\approx$ FOL
- Relational algebra

AI and Knowledge Representation:

- Semantic web (OWL)
- Expert systems
- Planning (PDDL)

Foundations:

- Axiomatic set theory
- Model theory

FOL Summary

Logic	Expressiveness	Decidable	Complete
Propositional	Low	✓	✓
First-Order	High	✗	✓
Second-Order	Very High	✗	✗

Key points:

- FOL adds objects, predicates, functions, and quantifiers to propositional logic
- **Translation:** "All A are B" uses $\rightarrow$; "Some A are B" uses $\wedge$
- **Semantics:** Structures interpret symbols; assignments handle variables
- **Meta-theory:** FOL is complete (valid = provable) but undecidable
- **Trade-off:** More expressiveness leads to higher complexity/undecidability

Modal Logic

"A truth is necessary when its negation implies a contradiction."

— Gottfried Wilhelm Leibniz

Beyond Truth: Modes of Truth

In propositional logic, statements are simply *true* or *false*.

But when we reason in everyday life, we often care about *how* something is true:

- "2 + 2 = 4" is **necessarily** true — it couldn't be otherwise
- "It will rain tomorrow" is **possibly** true — it might happen
- "John knows the door is locked" involves **knowledge**
- "Promises ought to be kept" involves **obligation**

These are different *modes* of truth, and classical logic can't express them.

Modal logic extends classical logic with operators for these *modalities*.

The Modal Operators

Definition 60: *Modal logic* extends propositional logic with two dual operators:

- $\Box\varphi$ — "necessarily φ" (box)
- $\Diamond\varphi$ — "possibly φ" (diamond)

These are related by duality:

$$\Diamond\varphi \equiv \neg\Box\neg\varphi$$

$$\Box\varphi \equiv \neg\Diamond\neg\varphi$$

Example:

- $\Box(2 + 2 = 4)$ — "Necessarily, 2 + 2 = 4" (mathematical truth)
- $\Diamond(\text{rain tomorrow})$ — "It's possible it will rain tomorrow"
- $\Box(P \to P)$ — "Necessarily, if P then P" (logical truth)

Modal Syntax

Definition 61: The syntax of basic modal logic extends propositional logic:

- If φ is a formula, then $\Box\varphi$ and $\Diamond\varphi$ are formulas
- All propositional connectives ($\neg$, $\wedge$, $\vee$, $\to$, $\iff$) apply

Precedence: $\Box$ and $\Diamond$ bind tighter than binary connectives.

Example (Formulas in modal logic):

- $\Box P \to P$ — "If necessarily P, then P"
- $\Box(P \to Q) \to (\Box P \to \Box Q)$ — Distribution axiom (K)
- $\Box P \to \Box\Box P$ — Positive introspection (4)
- $P \to \Box\Diamond P$ — Brouwer axiom (B)

Kripke Semantics: The Key Idea

Classical semantics assigns truth values to propositions.

Kripke semantics (1959, 1963) adds *structure*:

- Multiple "possible worlds"
- An *accessibility relation* between worlds
- Truth is evaluated *at a world*

Definition 62: A *Kripke frame* is a pair $\mathcal{F} = \langle W, R \rangle$ where:

- W is a non-empty set of *possible worlds*
- $R \subseteq W \times W$ is the *accessibility relation*

We write wRv to mean "world v is accessible from world w."

Kripke Models

Definition 63: A *Kripke model* is a triple $\mathcal{M} = (W, R, V)$ where:

- $\langle W, R \rangle$ is a Kripke frame
- $V : \text{Prop} \to \mathcal{P}(W)$ is a *valuation* function assigning to each proposition the set of worlds where it is true

Definition 64: *Truth at a world* w, written $\mathcal{M}, w \vDash \varphi$, is defined:

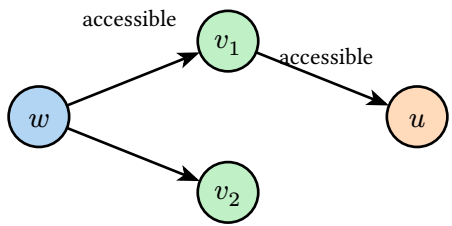
- $\mathcal{M}, w \vDash P$ iff $w \in V(P)$ (for atomic P)
- $\mathcal{M}, w \vDash \neg\varphi$ iff $\mathcal{M}, w \nvDash \varphi$
- $\mathcal{M}, w \vDash \varphi \wedge \psi$ iff $\mathcal{M}, w \vDash \varphi$ and $\mathcal{M}, w \vDash \psi$
- $\mathcal{M}, w \vDash \Box\varphi$ iff **for all** v with wRv: $\mathcal{M}, v \vDash \varphi$
- $\mathcal{M}, w \vDash \Diamond\varphi$ iff **there exists** v with wRv: $\mathcal{M}, v \vDash \varphi$

Understanding the Semantics

Intuition for $\Box$ and $\Diamond$:

- $\Box\varphi$ is true at w if φ is true in *all* worlds accessible from w
- $\Diamond\varphi$ is true at w if φ is true in *some* world accessible from w

The accessibility relation R determines "what counts as possible" from each world.



At w: $\Box P$ means "P holds at both v_1 and v_2"

Understanding the Semantics [2]

At w: $\Diamond P$ means "P holds at v_1 or v_2 (or both)"

Example: A Simple Kripke Model

Example: Consider a model with worlds $W = \{w_1, w_2, w_3\}$:

- $R = \{(w_1, w_2), (w_1, w_3), (w_2, w_3)\}$
- $V(P) = \{w_2, w_3\}$ (P is true at w_2 and w_3)
- $V(Q) = \{w_1, w_2\}$ (Q is true at w_1 and w_2)

Evaluate at w_1:

- $\mathcal{M}, w_1 \vDash P$? No, $w_1 \notin V(P)$
- $\mathcal{M}, w_1 \vDash \Box P$? Yes! From w_1, we access w_2 and w_3, both in $V(P)$
- $\mathcal{M}, w_1 \vDash \Diamond Q$? Yes! w_2 is accessible and $w_2 \in V(Q)$

Key observation: P is *false* at w_1, but $\Box P$ is *true* at w_1!

"Necessarily P" doesn't require P to hold at the current world — only at all accessible worlds.

Modal Axiom Systems

Different axiom systems capture different interpretations of necessity.

The base system **K** (after Kripke) contains:

- All propositional tautologies
- **K axiom:** $\Box(\varphi \to \psi) \to (\Box\varphi \to \Box\psi)$
- **Necessitation rule:** If $\vdash \varphi$, then $\vdash \Box\varphi$

Modal Axiom Systems [2]

Definition 65: Common additional axioms and their frame conditions:

Axiom	Name	Frame Condition
$\Box\varphi \to \varphi$	T	Reflexive: $\forall w.\ wRw$
$\Box\varphi \to \Box\Box\varphi$	4	Transitive: $wRv \land vRu \to wRu$
$\varphi \to \Box\Diamond\varphi$	B	Symmetric: $wRv \to vRw$
$\Diamond\varphi \to \Box\Diamond\varphi$	5	Euclidean: $wRv \land wRu \to vRu$

Important Modal Systems

System K:
- Base modal logic
- No conditions on R
- Weakest normal modal logic

System T (= K + T):
- Adds $\Box\varphi \to \varphi$
- Reflexive frames
- "What is necessary is actual"

System S4 (= T + 4):
- Adds transitivity
- Reflexive + transitive = preorder
- Common for knowledge logic

System S5 (= S4 + B = S4 + 5):
- Full equivalence relation
- "All worlds see all worlds"
- Strongest normal system

S5 captures the idea that possibility and necessity are "absolute" — if something is possible, it's necessarily possible.

Applications of Modal Logic

Epistemic Logic:

$\Box_a \varphi$ = "Agent a knows φ"

- $\Box_a \varphi \to \varphi$ (knowledge is true)
- $\Box_a \varphi \to \Box_a \Box_a \varphi$ (positive introspection)

Used in: AI, game theory, security protocols

Deontic Logic:

$\Box \varphi$ = "It ought to be that φ"

- Models obligations, permissions
- $\Box \varphi \to \Diamond \varphi$ (ought implies can)

Used in: Legal AI, normative systems

Temporal Logic:

- $\Box \varphi$ = "φ holds at all future times"
- $\Diamond \varphi$ = "φ holds at some future time"
- $\circ \varphi$ = "φ holds at the next time step"
- $\varphi \, \mathcal{U} \, \psi$ = "φ holds until ψ"

Applications of Modal Logic [2]

Used in: Verification of reactive systems (LTL, CTL)

Modal Logic in Computer Science

Example (Verifying a Mutex Property): Consider two processes P_1 and P_2 competing for a critical section.

Let crit_i = "process i is in critical section"

Mutual exclusion: $\Box\neg(\text{crit}_1 \wedge \text{crit}_2)$

"At all reachable states, both processes are never in the critical section simultaneously."

Model checking is the algorithmic verification of modal/temporal formulas over finite-state systems.

Tools like SPIN, NuSMV, and Alloy use these logics for specification — and many rely on SAT solvers internally!

Proofs as Programs

"The formulas-as-types notion is not just an analogy — it is an isomorphism."

— William Howard

The Curry–Howard Correspondence

A deep discovery: Proofs and programs are the *same thing* viewed differently!

- **Logic:** propositions, proofs, inference rules
- **Computation:** types, programs, evaluation rules

This is the *Curry–Howard correspondence* — also known as "propositions as types."

Historical milestones:

- 1934: Curry notices connection between combinatory logic and types
- 1969: Howard formalizes the correspondence for intuitionistic logic
- 1980s–now: Foundation of modern type theory and proof assistants (Coq, Lean, Agda)

The Dictionary

Definition 66: The Curry–Howard correspondence establishes:

Logic	Computation
Proposition φ	Type τ
Proof of φ	Program of type τ
Proposition φ is provable	Type τ is inhabited
$\varphi \wedge \psi$	Product type $\tau \times \sigma$
$\varphi \vee \psi$	Sum type $\tau + \sigma$
$\varphi \to \psi$	Function type $\tau \to \sigma$
$\bot$ (falsehood)	Empty type (no elements)
$\top$ (truth)	Unit type (one element)

Implication as Function Type

The heart of the correspondence:

A proof of $\varphi \to \psi$ is a *method* that transforms any proof of φ into a proof of ψ.

A function of type $A \to B$ is a *program* that transforms any value of type A into a value of type B.

These are the same concept viewed from different angles!

Example: The identity proof $\vdash P \to P$:

Proof:

1. Assume P *(hypothesis)*
2. We have P *(from 1)*
3. Therefore $P \to P$ *($\to$I)*

Program:

```
def identity(x: A) -> A:
    return x
```

Type: $A \to A$

Conjunction as Product Type

Example (Conjunction and Pairs): Proof of $P \wedge Q \to Q \wedge P$:

Proof:

1. Assume $P \wedge Q$
2. P by $\wedge$E from 1
3. Q by $\wedge$E from 1
4. $Q \wedge P$ by $\wedge$I from 3, 2

Program:

```
def swap(pair: (A, B)) -> (B, A):
    (a, b) = pair
    return (b, a)
```

Type: $(A \times B) \to (B \times A)$

- $\wedge$-Introduction = constructing a pair
- $\wedge$-Elimination = projecting from a pair

Disjunction as Sum Type

Example (Disjunction and Tagged Unions): Proof of $(P \to R) \to (Q \to R) \to (P \vee Q \to R)$:

Proof structure:

Given a proof of $P \to R$ and a proof of $Q \to R$, and given $P \vee Q$, do case analysis: if P, use first; if Q, use second.

Program:

```
def handle(f: A -> C,
           g: B -> C,
           x: A | B) -> C:
    match x:
        case Left(a): return f(a)
        case Right(b): return g(b)
```

Negation and Empty Types

Negation $\neg\varphi$ is defined as $\varphi \to \bot$.

Computationally: a function that takes a value of type φ and produces a value of the empty type (which is impossible — the function can never return!).

Example: A proof of $\neg(P \wedge \neg P)$ corresponds to:

```
def no_contradiction(x: (A, A -> Empty)) -> Empty:
    (a, not_a) = x
    return not_a(a)  # Calls a function that "never returns"
```

Ex falso quodlibet ($\bot \to \varphi$) corresponds to a function from the empty type to any type — which trivially exists because it's never called!

Classical vs. Intuitionistic Logic

The Curry–Howard correspondence works best with **intuitionistic logic**.

In intuitionistic logic:

- $\neg\neg\varphi \to \varphi$ is *not* provable
- $\varphi \vee \neg\varphi$ (excluded middle) is *not* provable
- Every proof is *constructive* — it provides a witness

Example (Why excluded middle is non-constructive): Consider $P \vee \neg P$. In intuitionistic logic, to prove a disjunction we must prove one of the disjuncts.

But $P \vee \neg P$ claims this *without specifying which*! It's not a program that computes anything — it's just an assertion.

Classical vs. Intuitionistic Logic [2]

Classical proofs (using RAA or LEM) correspond to programs with *control operators* like `call/cc` (call-with-current-continuation) or exceptions. The continuation captures "what happens next" — exactly like assuming a negation in RAA!

Proof Assistants and Type Theory

Modern proof assistants implement the Curry–Howard correspondence:

- Proofs are programs
- Type checking = proof verification
- Programming = proving theorems

Example (Lean 4 Syntax):

```
-- Proposition (type)
theorem and_comm : P ∧ Q → Q ∧ P := by
  intro ⟨hp, hq⟩  -- assume P ∧ Q, destruct
  exact ⟨hq, hp⟩  -- construct Q ∧ P

-- Same as:
def and_comm' : P ∧ Q → Q ∧ P :=
  fun ⟨hp, hq⟩ => ⟨hq, hp⟩
```

The Deep Correspondence

The Curry–Howard–Lambek correspondence extends to:

Logic	Type Theory	Category Theory
Propositions	Types	Objects
Proofs	Terms	Morphisms
Implication	Function space	Exponential
Conjunction	Product	Product
Disjunction	Coproduct	Coproduct
True	Unit	Terminal object
False	Empty	Initial object

The Deep Correspondence [2]

This three-way correspondence — logic, computation, algebra — is a deep unification at the heart of theoretical computer science.

Why Curry–Howard Matters

Practical applications:

1. **Verified software:** Programs proven correct *by construction*
2. **Proof automation:** Programming techniques applied to theorem proving
3. **Proof extraction:** Compile proofs into executable code
4. **Dependent types:** Types that depend on values, enabling rich specifications

Example: A function with type $\text{List}(A) \to \text{NonEmpty}(\text{List}(A)) + \text{Unit}$

The *type* guarantees: "Returns either a non-empty list or indicates the input was empty."

No runtime checks needed — it's enforced by the type system!

Looking Forward

We've traveled from truth tables to type theory, covering a lot of ground:

- **Propositional logic** gave us the basics: connectives, truth tables, proofs
- **Soundness and completeness** showed that semantic and syntactic truth coincide
- **First-order logic** let us reason about objects and their properties
- **Modal logic** added necessity, possibility, and beyond
- **Curry–Howard** revealed that proofs and programs are two views of the same thing

This is just the beginning — logic connects to almost every area of computer science and mathematics.

Topics to explore:

- Temporal logic (LTL, CTL)
- Intuitionistic logic
- Linear logic
- Description logics (OWL)

Connections:

- Computability theory
- Category theory
- Model theory
- Philosophical logic