

# Propositional Logic

Harry Sha

Fall 2026

# Overview

- Syntax.
- Semantics.
- Proof System.

# Propositional Logic - Example

Eventually, we want to study more complex logics, like first-order logic. Propositional logic is a good warm-up. Many of the concepts that will come up are extensions of the ideas used here!

# Section 1

## Syntax

# Syntax and Semantics

- Syntax has to do with the symbols we write down.
- Semantics is about how we assign “meaning” or truth to the strings of symbols we write down.

# The Language of Propositional Logic

$$\Sigma = \{P, ', \neg, \wedge, \vee, (, )\}$$

The variables are  $P, P', P'' \dots$  etc. For readability, we'll write  $P_0, P_1, P_2$ , and often use other characters too like  $P, Q, R$ .

# More Symbols

For readability, we'll also use the  $\rightarrow$  and  $\leftrightarrow$  as shorthand

- $(A \rightarrow B)$  is shorthand for  $(\neg A \vee B)$
- $(A \leftrightarrow B)$  is shorthand for  $((A \rightarrow B) \wedge (B \rightarrow A))$

# Meta vs. Object Language

Since we're about to prove theorems about propositional logic, we'll sometimes want to use logical connectives in those theorem statements. Those will usually be written in English: “and”, “or”, etc. Occasionally, we'll use  $\implies$  for “implies” and  $\iff$  for “iff”. Compare these to  $\rightarrow$  and  $\leftrightarrow$  from the object language.

This will come up later and I'll point it out.

# Propositional Formulas

The propositional formulas are defined recursively as follows.

- Every string of the form  $P$  followed by 0 or more  $'$  is a formula. These are called “atoms” or “propositional variables”.
- If  $A$  is a formula, then so is  $\neg A$ .
- If  $A$  and  $B$  are formulas, then so are  $(A \vee B)$  and  $(A \wedge B)$ .

Let  $\mathcal{F}$  be the set of propositional formulas.

**Convention:**  $P, Q, R, \dots$  are variables.  $A, B, C, \dots$  are formulas

## Exercise (later)

Show  $\mathcal{F}$  is decidable.

# Proving things about $\mathcal{F}$

To prove stuff about all formulas, we use **structural induction**. I.e. to show for all formulas  $A \in \mathcal{F}$ ,  $\mathcal{P}(A)$ , where  $\mathcal{P}$  is some predicate, it suffices to show the following.

- $\mathcal{P}(A)$  for all atoms.
- For all  $A, B \in \mathcal{F}$ ,
  - $\mathcal{P}(A) \implies \mathcal{P}(\neg A)$
  - $\mathcal{P}(A)$  and  $\mathcal{P}(B) \implies \mathcal{P}((A \wedge B))$  and  $\mathcal{P}((A \vee B))$

# Question

What does the following formula mean:

$$A \wedge B \vee C$$

# Unique Readability Theorem

The unique readability theorem says that we've added enough parentheses to make parsing a formula unambiguous.

## Theorem (Unique Readability)

*Every formula is exactly one of the following*

- *An atom*
- $\neg A$  *for a unique formula  $A$*
- $(A \wedge B)$  *for a unique pair of formulas  $(A, B)$ .*
- $(A \vee B)$  *for a unique pair of formulas  $(A, B)$ .*

# Proof

Define the following weights.

| token  | $P_i$ | $\neg$ | $($ | $\wedge, \vee$ | $)$ |
|--------|-------|--------|-----|----------------|-----|
| weight | -1    | 0      | 1   | 1              | -1  |

The weight of a string is then the sum of the weights of its tokens

**Lemma.** The weight of every formula is  $-1$ . Furthermore, every proper initial segment of the token sequence has non-negative weight.

# Proof of Lemma

|        |       |        |     |                |      |
|--------|-------|--------|-----|----------------|------|
| token  | $P_i$ | $\neg$ | $($ | $\wedge, \vee$ | $)$  |
| weight | $-1$  | $0$    | $1$ | $1$            | $-1$ |

**Lemma.** The weight of every formula is  $-1$ . Furthermore, every proper initial segment of the token sequence has non-negative weight.

# Proof of the Unique Readability Theorem

## Exercise (later)

Write a parser for propositional formulas.

# Precedence rules

Since the parentheses can be annoying to write sometimes, for readability, we can omit some parentheses and use the following precedence rules. Note these conventions are for our own readability only, the unparenthesized strings are not technically formulas.

In order of precedence:  $()$ ,  $\neg$ ,  $\wedge$ ,  $\vee$ ,  $\rightarrow$ ,  $\leftrightarrow$ .

Example:

## Precedence rules

Also,  $\rightarrow$  is right associative. I.e.  $A \rightarrow B \rightarrow C$  is  $A \rightarrow (B \rightarrow C)$  instead of  $(A \rightarrow B) \rightarrow C$

It turns out, it doesn't matter how you parenthesize  $A \circ B \circ C$  for  $\circ \in \{\wedge, \vee, \leftrightarrow\}$

## Section 2

## Semantics

# Truth

Let  $A$  be a propositional formula, for example  $(P \wedge \neg Q)$ . Is  $A$  true?

# Truth Assignments

Let  $\mathcal{A}$  be the set of atoms. I.e.  $\mathcal{A} = \{P, P', P'', \dots\}$ .

A truth assignment is a function  $v : \mathcal{A} \rightarrow \{0, 1\}$ . 0 represents False and 1 represents True.

# Truth of a Formula

A truth assignment extends recursively to formulas in the following way.

- $v(\neg A) = 1 - v(A)$
- $v((A \vee B)) = \max(v(A), v(B))$
- $v((A \wedge B)) = \min(v(A), v(B))$

Example:

# Truth

*The truth of a formula depends on the assignment of variables.*

# Satisfiable, Valid

Let  $A$  be a formula and  $S$  be a set of formulas

- A truth assignment  $v$  **satisfies**  $A$  if  $v(A) = 1$ . The assignment  $v$  satisfies  $S$  if  $v$  satisfies every formula in  $S$ .
- $A$  is **satisfiable** if some truth assignment satisfies  $A$ .
- $S$  is **satisfiable** if some truth assignment satisfies  $S$ .
- $A$  is **valid** if all truth assignments satisfy  $A$ .

Essentially, 'satisfy' means 'make true', and 'valid' means 'always true'.

**Notation.** We use  $v \models A$  to mean  $v$  satisfies  $A$ , similarly,  $v \models S$  means  $v$  satisfies  $S$ .

# Questions

- Suppose  $A$  is valid, what can we say about  $\neg A$ ?
- Suppose  $A$  is satisfiable, what can we say about  $\neg A$ ?

# Example

# Tautologically implies

Let  $S$  be a set of formulas, and  $A$  be a formula. Then  $S$  tautologically implies  $A$ , written

$$S \models A,$$

if every truth assignment satisfying  $S$  also satisfies  $A$ .

# Example

$$\{P, P \rightarrow Q\} \models Q$$

## Exercise (in class)

Show

$$S \models A \iff S \cup \{\neg A\} \text{ is unsatisfiable}$$

## Exercise (in class)

Show

If  $S \models A$  and  $S \cup \{A\} \models B$  then  $S \models B$

# Semantically equivalent

$A$  and  $B$  are semantically equivalent, written  $A \equiv B$  if

$$A \models B \text{ and } B \models A.$$

That is, for every truth assignment  $v$ ,  $v(A) = v(B)$ .

# Syntax vs. Semantics

$$P \vee Q \equiv Q \vee P.$$

Syntactically different, but semantically equivalent.

## Section 3

### Automating Logic

Is  $\models$  decidable?

Let  $S$  be a finite set, and  $A$  be a formula. Then the set  $\{\langle S, A \rangle : S \models A\}$  is decidable.

# Later...

Later, we'll see what happens when  $S$  is an infinite but decidable set.

# Summary

- We defined propositional formulas and truth assignments.
- Truth is *not* a property of formulas. Rather, the truth of a formula, depends on the truth assignment.
- By contrast, satisfiability and validity *are* properties of a formula. They mean there exists a truth assignment that makes the formula true, and all truth assignments make the formula true respectively.
- We can decide if  $S \models A$  by looking at the truth tables (when  $S$  is finite).

## Section 4

### Proofs - PK

## Showing $\models$

Earlier, we saw how to show  $S \models A$  (when  $S$  was finite) using truth tables.

Proofs are another way to show  $S \models A$ . In particular, we'll define  $S \vdash A$  to mean that there is a proof of  $A$  from the assumptions in  $S$ . We'll show that for the proof system that we study, that

$$S \models A \iff S \vdash A.$$

This holds for every set  $S$ , even infinite sets.

# Sequents

A cedent is a finite sequence of formulas. A sequent has the form

$$A_1, \dots, A_k \Rightarrow B_1, \dots, B_\ell$$

- The cedent to the left of the  $\Rightarrow$  is called the antecedent
- The cedent to the right of the  $\Rightarrow$  is called the succedent
- Cedents may be empty.
- **Convention.** We use  $\Gamma, \Delta$  for cedents,  $\varphi$  for a sequent, and  $S$  for a set of assumptions.

# Satisfies for Sequents

A truth assignment  $v$  satisfies a sequent

$$\varphi = A_1, \dots, A_k \Rightarrow B_1, \dots, B_\ell$$

if it falsifies at least one of the  $A_i$  or satisfies at least one of the  $B_i$ . We write  $v \models \varphi$ .

We write  $S \models \varphi$  if every truth assignment satisfying  $S$  satisfies  $\varphi$ .

# The Associated Formula

For a sequent

$$\varphi = A_1, \dots, A_k \Rightarrow B_1, \dots, B_\ell$$

Using the convention that the empty conjunction is True and the empty disjunction is False, the associated formula is

$$A_\varphi = (A_1 \wedge \dots \wedge A_k) \rightarrow (B_1 \vee \dots \vee B_\ell)$$

# Examples: Valid Sequents

- $A \Rightarrow A$
- $\Rightarrow A, \neg A$
- $A, \neg A \Rightarrow$
- $A, A \rightarrow B \Rightarrow B$

# Weakening, Exchange, Contraction

## Weakening

$$\text{left} \quad \frac{\Gamma \Rightarrow \Delta}{A, \Gamma \Rightarrow \Delta} \qquad \text{right} \quad \frac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, A}$$

## exchange

$$\text{left} \quad \frac{\Gamma_1, A, B, \Gamma_2 \Rightarrow \Delta}{\Gamma_1, B, A, \Gamma_2 \Rightarrow \Delta} \qquad \text{right} \quad \frac{\Gamma \Rightarrow \Delta_1, A, B, \Delta_2}{\Gamma \Rightarrow \Delta_1, B, A, \Delta_1}$$

## Contraction

$$\text{left} \quad \frac{\Gamma, A, A \Rightarrow \Delta}{\Gamma, \Delta \Rightarrow \Delta} \qquad \text{right} \quad \frac{\Gamma \rightarrow \Delta, A, A}{\Gamma \rightarrow \Delta, A}$$

# Logical Rules

## $\neg$ introduction

$$\text{left} \quad \frac{\Gamma \Rightarrow \Delta, A}{\neg A, \Gamma \Rightarrow \Delta}$$

$$\text{right} \quad \frac{A, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \neg A}$$

## $\wedge$ introduction

$$\text{left} \quad \frac{A, B, \Gamma \Rightarrow \Delta}{(A \wedge B), \Gamma \Rightarrow \Delta}$$

$$\text{right} \quad \frac{\Gamma \Rightarrow \Delta, A \quad \Gamma \Rightarrow \Delta, B}{\Gamma \Rightarrow \Delta, (A \wedge B)}$$

## $\vee$ introduction

$$\text{left} \quad \frac{A, \Gamma \Rightarrow \Delta \quad B, \Gamma \Rightarrow \Delta}{(A \vee B), \Gamma \Rightarrow \Delta}$$

$$\text{right} \quad \frac{\Gamma \Rightarrow \Delta, A, B}{\Gamma \Rightarrow \Delta, (A \vee B)}$$

# Cut

cut

$$\frac{\Gamma \Rightarrow \Delta, A \quad A, \Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta}$$

## Proof of a sequent

A PK proof of sequent  $\varphi$  from assumptions  $S$  is a finite PK proof tree whose last sequent is  $\varphi$ , and whose leaves may include sequents of the form  $\Rightarrow B$  for  $B \in S$ . The leaves can also include the axioms  $C \Rightarrow C$  for any formula  $C$ . We write

$$S \vdash_{PK} \varphi$$

If  $A$  is a formula, a PK proof of  $A$  is a proof of the sequent  $\Rightarrow A$ . We'll write  $S \vdash_{PK} A$

# Conventions

- Since we'll only be talking about PK for a while, we simply write  $S \vdash \varphi$ .
- We write  $\vdash \varphi$  to mean  $\emptyset \vdash \varphi$ .
- For a formula  $A$ ,  $S \vdash A$  abbreviates  $S \vdash (\Rightarrow A)$ .

## Example

$$\vdash \Rightarrow P \vee \neg P$$

## Example

$$\vdash \neg(P \vee Q) \Rightarrow \neg P \wedge \neg Q$$

## Section 5

### Properties of PK

# Properties of a Proof System

From a previous discussion we decided that there are several properties of a proof system that are desirable.

# Decidable?

Let  $\Sigma$  be an alphabet containing all of the symbols required to write down a PK proof.

Are the following sets decidable?

$$\{w \in \Sigma^* : w \text{ is a valid PK proof.}\}$$

$$\{\langle w, \varphi \rangle \in \Sigma^* : w \text{ is a valid PK proof of } \varphi\}$$

# Soundness and Completeness

We want to show

$$S \vdash \varphi \iff S \models \varphi.$$

The forward direction is called “soundness” and the backwards direction is called “completeness”.

# Soundness

## Theorem

*For every set of formulas  $S$ , and every sequent  $\varphi$ .*

$$S \vdash \varphi \implies S \models \varphi.$$

## Lemma: Local Soundness

**Lemma.** For every PK rule, every assignment satisfying all the premises also satisfies the conclusion.

# Soundness

## Theorem (Completeness, without assumptions)

*For every sequent  $\varphi$ ,*

$$\models \varphi \implies \vdash \varphi$$

# How can we prove this?

Look at the example before...

## Lemma: Inversion

**Lemma.** For each logical rule, if an assignment satisfies the conclusion of the rule, then it must satisfy the premise(s) of the rule.

# Proof Search

# Atomic Sequents

Eventually, our search reaches a point where all the leaves have sequents that don't have any logical connectives. These are called atomic sequents.

**Claim.** An atomic sequent is valid if and only if some atom appears on both sides.

# Notes about the Proof Search

- We never used the cut rule!
- Subformula property. Any formula at any step of the proof is a subformula of some formula in the endsequent.

# Completeness

So far, we have the following form of the completeness theorem

$$\models \varphi \implies \vdash \varphi.$$

We actually, originally wanted a stronger result which was that

$$S \models \varphi \implies S \vdash \varphi,$$

where  $S$  is some set of assumptions.

## Case $S$ is finite

**Lemma (Completeness from finite assumptions).** Let  $\Gamma \Rightarrow \Delta$  be an arbitrary sequent. Let  $S$  be a finite set of formulas. Then  $S \models \Gamma \Rightarrow \Delta$  implies  $S \vdash \Gamma \Rightarrow \Delta$ .

**Proof.** Let  $S = \{A_1, \dots, A_k\}$ . Then, the sequent  $A_1, \dots, A_k, \Gamma \Rightarrow \Delta$  is valid, so it has a proof.

∴ Proof

---

$$A_1, \dots, A_k, \Gamma \Rightarrow \Delta$$

# Compactness

So we now know how to do

$$S \models \varphi \implies S \vdash \varphi,$$

In the case that  $S$  is finite.

The last step is then to show that we can always reduce to this case!

This will be a consequence of the next theorem.

# Compactness Theorem

## Theorem

*Let  $S$  be a set of formulas. If every finite subset of  $S$  is satisfiable, then  $S$  is satisfiable.*

# Koenig's Lemma

**Lemma.** Every infinite, finitely branching rooted tree has an infinite branch.

# Proof Idea

Build an infinite, finitely branching rooted tree. The infinite branch we get from Koenig's Lemma will be our satisfying assignment.

# Tree construction

$V$  is infinite

**Claim.** For each  $n \in \mathbb{N}$ , some  $v \in \{0, 1\}^n$  is promising.

## It's connected

**Claim.** If  $x$  is promising, so is  $x[: -1]$ .

So the tree has an infinite branch. This yields a truth assignment. (How?)

Furthermore, this assignment satisfies every formula in  $S$ . (Why?)

# Consequences of Compactness

The following are equivalent

1. If every finite subset of  $S$  is satisfiable, then so is  $S$ .
2. If  $S$  is unsatisfiable, then there is a finite subset of  $S$  that is unsatisfiable
3. If  $S \models A$ , then there is some finite subset  $S_0 \subseteq S$  such that  $S_0 \models A$ .

## A similar statement for sequents

If  $S \models \varphi$ , then there is some finite subset  $S_0 \subseteq S$  such that  $S_0 \models \varphi$ .

# Putting it all together

We wanted the following form of completeness:

$$S \models \varphi \implies S \vdash \varphi.$$

We have

$$S \models \varphi \implies S_0 \models \varphi$$

$$\implies S_0 \vdash \varphi$$

$$\implies S \vdash \varphi$$

---

---

---

# Next

First order logic.