Propositional Logic II
(syntax)

Objective: to find a better method for testing tautologies than truth table method

Solution: to generalize the problem to sets (possibly infinite) of formulas.

Satisfaction relations for sets of propositional formulas

For T a set of formulas and v a valuation (both possibly infinite), we say that v **satisfies** T, in writing $\models_p^v T$, iff for all $A \in T$ we have $\models_p^v A$.

We define $\neg T = \{\neg A \mid A \in T\}$. If not $\models_p^v \neg T$ then v **does not refute** T. This means $\models_p^v A$ for some $A \in T$

We say that S **is a propositional (tautological) consequence of** T, in writing $T \models_p S$, iff for all v such that $\models_p^v T$ we do not have $\models_p^v \neg S$, i.e. **no** v **satisfying** T **refutes** S

The special case when $T \models_p \{A\}$ is the most important relation in mathematical logic. We write $T \models_p A$ instead of $T \models_p \{A\}$ and say that A **tautologically follows from** T

Compactness theorem for propositional consequence

$T \models_p S$ iff there are finite $T' \subset T$ and $S' \subset S$ s.t. $T' \models_p S'$.

If $T' = \{A_1, \ldots, A_n\}$ and $S' = \{B_1, \ldots, B_m\}$ we have $T' \models_p S'$ iff

$$\models_p A_1 \wedge \cdots \wedge A_n \rightarrow B_1 \vee \cdots \vee B_m$$

Note that $T \models_p \emptyset$ iff T is **unsatisfiable**, i.e. for all v there is $A \in T$ s.t. $\models_p^v \neg A$.

Also, $\emptyset \models_p S$ iff S is **non-refutable**, i.e. for all v there is $A \in S$ s.t. $\models_p^v A$.
Also, not $\emptyset \models_p \emptyset$
Also, $\emptyset \models_p \{A\}$ iff A is tautology.

We will study this in more detail in Logic II.

Observations leading to better tests for tautological consequence

If T and S consist only of propositional variables then $T \models_p S$ iff $T \cap S \neq \emptyset$

If $\bot \in S$ then $T \models_p S$ iff $T \models_p S \setminus \{\bot\}$

If $\bot \in T$ then $T \models_p S$

If $(A \to B) \in S$ then

$$\begin{aligned} T \models_p S \text{ iff } \quad & T \cup \{A\} \models_p S \cup \{B\} \text{ iff} \\ & T \cup \{A\} \models_p S \setminus \{A \to B\} \cup \{B\} \end{aligned}$$

If $(A \to B) \in T$ then

$$\begin{aligned} T \models_p S \text{ iff } \quad & T \cup \{B\} \models_p S \text{ and } T \models_p S \cup \{A\} \text{ iff} \\ & T \setminus \{A \to B\} \cup \{B\} \models_p S \text{ and} \\ & T \setminus \{A \to B\} \models_p S \cup \{A\} \end{aligned}$$

Arithmetization

For finite sets T and S we can arithmetize the predicate $T \vDash_p S$ by defining in CL:

$$
\begin{aligned}
&t \vDash_p^\bullet s \leftrightarrow \forall v(\\
&\quad \forall a(a \,\varepsilon\, t \to \vDash_p^v a) \to \exists a(a \,\varepsilon\, s \to \vDash_p^v a))
\end{aligned}
$$

The properties from the previous slide can be then used to define by a **clausal definition** a fourplace predicate $\mathit{Seq}(t, v, s, w)$ taking lists of formulas t, s and lists of numbers v, w such that

$$\mathit{Seq}(t, v, s, w) \leftrightarrow t \oplus \mathit{Map}_{P^\bullet_\cdot}(v) \vDash_p^\bullet s \oplus \mathit{Map}_{P^\bullet_\cdot}(w)$$

Note that the lists v and w store the **indices** i of propositional variables $P_i^\bullet$ encountered in t and s respectively.
We then define

$$\mathit{Taut}(a) \leftarrow \mathit{Seq}(0, 0, (a, 0), 0)$$

$$
\begin{array}{ll}
Seq(0, v, 0, w) & \leftarrow v \cap w > 0 \\
Seq(0, v, (P_i^\bullet, s), w) & \leftarrow Seq(0, v, s, (i, w)) \\
Seq(0, v, (\bot^\bullet, s), w) & \leftarrow Seq(0, v, s, w) \\
Seq(0, v, (a \rightarrow^\bullet b, s), w) & \leftarrow Seq((a, 0), v, (b, s), w) \\
Seq((P_i^\bullet, t), v, s, w) & \leftarrow Seq(t, (i, v), s, w) \\
Seq((\bot^\bullet, t), v, s, w) & \\
Seq((a \rightarrow^\bullet b, t), v, s, w) & \leftarrow Seq((b, t), v, s, w) \wedge \\
 & \quad Seq(t, v, (a, s), w)
\end{array}
$$

How to derive clauses for other **connectives**? By using them on both sides of Seq and simplifying. We note that when we replace in the first four clauses the first 0 by s we have more general properties of Seq then the four clauses. For instance, for $\neg^\bullet a$ in the consequent we have: $Seq(t, v, (\neg^\bullet a, s), w)$ iff
$Seq(t, v, (a \rightarrow^\bullet \bot^\bullet, s), w)$ iff $Seq((a, t), v, (\bot^\bullet, s), w)$ iff $Seq((a, s), v, s, w)$

For $\neg^\bullet a$ in the antecedent we have:
$Seq((\neg^\bullet a, t), v, s, w)$ iff $Seq((a \rightarrow^\bullet \bot^\bullet, t), v, s, w)$ iff $Seq((\bot^\bullet, t), v, s, w)$ and $Seq(t, v, (a, s), w)$ iff $Seq(t, v, (a, s), w)$