

syn.1 Propositional Formulas

pl:syn:fml:
sec **Formulas** of propositional logic are built up from *propositional variables*, the propositional constant \perp and the propositional constant \top using *logical connectives*.

1. A denumerable set At_0 of *propositional variables* p_0, p_1, \dots
2. The propositional constant for *falsity* \perp .
3. The propositional constant for *truth* \top .
4. The logical connectives: \neg (negation), \wedge (conjunction), \vee (disjunction), \rightarrow (*conditional*), \leftrightarrow (*biconditional*)
5. Punctuation marks: $(,)$, and the comma.

We denote this language of propositional logic by \mathcal{L}_0 .

You may be familiar with different terminology and symbols than the ones intro we use above. Logic texts (and teachers) commonly use either \sim , \neg , and $!$ for “negation”, \wedge , \cdot , and $\&$ for “conjunction”. Commonly used symbols for the “conditional” or “implication” are \rightarrow , \Rightarrow , and \supset . Symbols for “biconditional,” “bi-implication,” or “(material) equivalence” are \leftrightarrow , \Leftrightarrow , and \equiv . The \perp symbol is variously called “falsity,” “falsum,” “absurdity,” or “bottom.” The \top symbol is variously called “truth,” “verum,” or “top.”

pl:syn:fml:
defn:formulas **Definition syn.1 (Formula).** The set $\text{Frm}(\mathcal{L}_0)$ of *formulas* of propositional logic is defined inductively as follows:

1. \perp is an atomic *formula*.
2. \top is an atomic *formula*.
3. Every *propositional variable* p_i is an atomic *formula*.
4. If φ is a *formula*, then $\neg\varphi$ is a *formula*.
5. If φ and ψ are *formulas*, then $(\varphi \wedge \psi)$ is a *formula*.
6. If φ and ψ are *formulas*, then $(\varphi \vee \psi)$ is a *formula*.
7. If φ and ψ are *formulas*, then $(\varphi \rightarrow \psi)$ is a *formula*.
8. If φ and ψ are *formulas*, then $(\varphi \leftrightarrow \psi)$ is a *formula*.
9. Nothing else is a *formula*.

explanation

The definition of **formulas** is an *inductive definition*. Essentially, we construct the set of **formulas** in infinitely many stages. In the initial stage, we pronounce all atomic formulas to be formulas; this corresponds to the first few cases of the definition, i.e., the cases for \top , \perp , p_i . “Atomic **formula**” thus means any **formula** of this form.

The other cases of the definition give rules for constructing new **formulas** out of **formulas** already constructed. At the second stage, we can use them to construct **formulas** out of atomic **formulas**. At the third stage, we construct new formulas from the atomic formulas and those obtained in the second stage, and so on. A **formula** is anything that is eventually constructed at such a stage, and nothing else.

When writing a formula $(\psi * \chi)$ constructed from ψ , χ using a two-place connective $*$, we will often leave out the outermost pair of parentheses and write simply $\psi * \chi$.

Definition syn.2 (Syntactic identity). The symbol \equiv expresses syntactic identity between strings of symbols, i.e., $\varphi \equiv \psi$ iff φ and ψ are strings of symbols of the same length and which contain the same symbol in each place.

The \equiv symbol may be flanked by strings obtained by concatenation, e.g., $\varphi \equiv (\psi \vee \chi)$ means: the string of symbols φ is the same string as the one obtained by concatenating an opening parenthesis, the string ψ , the \vee symbol, the string χ , and a closing parenthesis, in this order. If this is the case, then we know that the first symbol of φ is an opening parenthesis, φ contains ψ as a substring (starting at the second symbol), that substring is followed by \vee , etc.

Photo Credits

Bibliography