Propositional Logic: Syntax and Semantics

Supratik Chakraborty
IIT Bombay

Notation

- Variables: $p, q, r, \ldots$
  - Represent *propositions* or *declarative statements*
- Constants: $\top, \bot$
- Operators or connectives:
  - $\wedge,\ \vee,\ \neg,\ \rightarrow,\ \leftrightarrow,\ (,\ )$
    - We don't need all of them, but convenient to have them ...

Syntax of Propositional Logic

Rules for formuling formulas

- Every variable constitutes a formula
- The constants $\top$ and $\bot$ are formulae.
- If φ is a formula, so are $\neg\varphi$ and (φ)
- If φ_1 and φ_2 are formulas, so is $\varphi_1 \wedge \varphi_2$, $\varphi_1 \vee \varphi_2$, $\varphi_1 \rightarrow \varphi_2$, $\varphi_1 \leftrightarrow \varphi_2$

Propositional formulas as strings and trees

- Alphabet (over which strings are constructed):
  - Set of variable names, e.g. $\{p_1, p_2, q_1, q_2, ...\}$
  - Set of constants $\{\top, \bot\}$
  - Fixed symbols $\{\neg, \vee, \wedge, \rightarrow, \leftrightarrow\}$
- Well-formed formula: string formed according to rules on prev. slide
  - $(p_1 \vee \neg q_2) \wedge (\neg p_2 \rightarrow (q_1 \leftrightarrow \neg p_1))$
  - $p_1 \rightarrow (p_2 \rightarrow (p_3 \rightarrow p_4))$
- Well-formed formulas can be represented using trees
  - Consider the rules on prev. slide as telling how to construct a tree bottom-up
  - Parse trees: Obviate the need for '(' and ')'

Semantics of Propositional Logic

Consider a formula φ with n variables. Let 0 represent "false" and 1 represent "true"

- $[\![\varphi]\!] : \{0,1\}^n \rightarrow \{0,1\}$
- **Semantics is a function**
  - Often represented in tabular form: Truth Table
- Indicates truth value of formula, given truth values of all variables

Rules of semantics

- $[\![\neg\varphi]\!] = 1$ iff $[\![\varphi]\!] = 0$.
- $[\![\varphi_1 \wedge \varphi_2]\!] = 1$ iff $[\![\varphi_1]\!] = [\![\varphi_2]\!] = 1$.
- $[\![\varphi_1 \vee \varphi_2]\!] = 1$ iff at least one of $[\![\varphi_1]\!]$ or $[\![\varphi_2]\!]$ evaluates to 1.
- $[\![\varphi_1 \rightarrow \varphi_2]\!] = 1$ iff at least one of $[\![\varphi_1]\!] = 0$ or $[\![\varphi_2]\!] = 1$.
- $[\![\varphi_1 \leftrightarrow \varphi_2]\!] = 1$ iff both $[\![\varphi_1 \rightarrow \varphi_2]\!] = 1$ or $[\![\varphi_2 \rightarrow \varphi_1]\!] = 1$.