22c181: Formal Methods in Software Engineering

The University of Iowa

Spring 2010

Propositional Logic

Mathematical Logic

- A discipline that studies the precise formalization of knowledge and reasoning
- Provides the foundations of Formal Methods
- In this field the word logic is also use to denote specific formal reasoning systems
- there are several logics in that sense:
  propositional, first-order, higher-order, modal, temporal, intuitionistic, linear, non-monotonic, ...

We will concentrate on propositional logic and first-order logic
(But we'll also work in some temporal logic in disguise)

Logics

A logic is a triple $\langle \mathcal{L}, \mathcal{S}, \mathcal{R} \rangle$ where

- $\mathcal{L}$, the logic's language, is a class of sentences described by a formal grammar.
- $\mathcal{S}$, the logic's semantics is a formal specification of how to assign meaning in the "real world" to the elements of $\mathcal{L}$.
- $\mathcal{R}$, the logic's inference system, is a set of formal inference rules over $\mathcal{L}$.

Propositional Logic

Each sentence (called a formula) is made of

- propositional variables, $a, b, \ldots, p, q, \ldots$
- logical constants, $\top, \bot$
- logical connectives $\wedge, \vee, \Rightarrow, \ldots$

Every propositional variable stands for a basic fact

Examples: *I'm hungry, Apples are red, Joe and Jill are married*

Propositional Logic

Ontological Commitments

Propositional Logic is about facts, statements that are either true or false, nothing else.

Semantics of Propositional Logic

Since each propositional variable stands for a fact about the world, its meaning ranges over the Boolean values $\{True, False\}$.

Same for more complex formulas.

Remark: Note the difference between $True, False$, semantical entities here, with $\top, \bot$, syntactical entities standing for them.

Formulas of Propositional Logic

- Each propositional variable ($a, b, \ldots, p, q, \ldots$) is a formula
- Each logical constant ($\top, \bot$) is a formula
- If φ and ψ are formulas, all of the following are also formulas

$$
\begin{array}{ccc}
\neg\varphi & \varphi \wedge \psi & \varphi \Rightarrow \psi \\
(\varphi) & \varphi \vee \psi & \varphi \Leftrightarrow \psi
\end{array}
$$

- Nothing else is a formula

The Language of Propositional Logic

Formally, it is the language generated by the following grammar.

- Symbols:
  - Propositional variables: $a, b, \ldots, p, q, \ldots$
  - Logical symbols:

$\top$	(true)	$\wedge$	(and)	$\Rightarrow$	(implies)	$\neg$	(not)
$\bot$	(false)	$\vee$	(or)	$\Leftrightarrow$	(equivalent)		

- Grammar Rules:

$$
\begin{array}{lcl}
Conn & ::= & \wedge \mid \vee \mid \Rightarrow \mid \Leftrightarrow \\
AtomicF & ::= & \top \mid \bot \mid a \mid b \mid \ldots \mid p \mid q \mid \ldots \\
Formula & ::= & AtomicF \mid \neg Formula \mid \\
 & & Formula\ Conn\ Formula \mid (Formula)
\end{array}
$$

Semantics of Propositional Logic

- The meaning (value) of $\top$ is always $True$. The meaning of $\perp$ is always $False$.
- The meaning of the other formulas depends on the meaning of the propositional variables.
  - Base cases: Truth Tables

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
False	*False*	*True*	*False*	*False*	*True*	*True*
False	*True*	*True*	*False*	*True*	*True*	*False*
True	*False*	*False*	*False*	*True*	*False*	*False*
True	*True*	*False*	*True*	*True*	*True*	*True*

  - Non-base Cases: Given by reduction to the base cases

    Example: the meaning of $(p \vee q) \wedge r$ is the same as the meaning of $a \wedge r$ where a has the same meaning as $p \vee q$.

The Meaning of Logical Connectives: A Warning

Disjunction

- $\phi \vee \psi$ is true when ϕ or ψ or both are true (inclusive or).

Implication

- $\phi \Rightarrow \psi$ does not require a causal connection between ϕ and ψ.

  Ex: *Sky-is-blue* $\Rightarrow$ *Snow-is-white*

- When ϕ is false, $\phi \Rightarrow \psi$ is true regardless of the value of ψ.

  Ex: $\perp \Rightarrow p$

- Beware of negations in implications.

  Ex: *Is-bird* $\Rightarrow$ *Lays-eggs*

  $\neg$*Is-bird* $\Rightarrow$ $\neg$*Lays-eggs*

Semantics of Propositional Logic

- An assignment of Boolean values to the propositional variables of a formula is an interpretation of the formula.

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	*False*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*
True	*False*	*True*	*True*	*True*
True	*True*	*True*	*False*	*True*

Interpretations:

$$\{P \mapsto \mathit{False}, H \mapsto \mathit{False}\}, \{P \mapsto \mathit{False}, H \mapsto \mathit{True}\}, \ldots$$

- The semantics of Propositional logic is compositional: the meaning of a formula is defined recursively in terms of the meaning of the formula's components.

Semantics of Propositional Logic

The meaning of a formula in general depends on its interpretation.

Some formulas, however, have always the same meaning.

P	H	$P \vee H$	$(P \vee H) \wedge \neg H$	$((P \vee H) \wedge \neg H) \Rightarrow P$
False	*False*	*False*	*False*	*True*
False	*True*	*True*	*False*	*True*
True	*False*	*True*	*True*	*True*
True	*True*	*True*	*False*	*True*

A formula is

- (un)satisfiable if it is true in some (no) interpretation,
- valid if it is true in every possible interpretation.

Entailment in Propositional Logic

Given

- a set Γ of formulas and
- a formula φ,

we write

$$\Gamma \models \varphi$$

iff every interpretation that makes all formulas in Γ true makes φ also true.

$\Gamma \models \varphi$ is read as "Γ entails φ" or "φ logically follows from Γ."

Entailment in Propositional Logic: Examples

$$
\begin{array}{rcl}
\{A, A \Rightarrow B\} & \models & B \\
\{A\} & \models & A \vee B \\
\{A, B\} & \models & A \wedge B \\
\{\} & \models & A \vee \neg A \\
\{A\} & \not\models & A \wedge B \\
\{A \vee \neg A\} & \not\models & A
\end{array}
$$

	A	B	$A \Rightarrow B$	$A \vee B$	$A \wedge B$	$A \vee \neg A$
1.	*False*	*False*	*True*	*False*	*False*	*True*
2.	*False*	*True*	*True*	*True*	*False*	*True*
3.	*True*	*False*	*False*	*True*	*False*	*True*
4.	*True*	*True*	*True*	*True*	*True*	*True*

Properties of Entailment

- $\Gamma \models \varphi$, for all $\varphi \in \Gamma$ (inclusion property of $\models$)
- if $\Gamma \models \varphi$, then $\Gamma' \models \varphi$ for all $\Gamma' \supseteq \Gamma$ (monotonicity of $\models$)
- φ is valid iff $\{\} \models \varphi$ (also written as $\models \varphi$)
- $\varphi \models \bot$ iff φ is unsatisfiable
- $\Gamma \models \varphi$ iff the set $\Gamma \cup \{\neg\varphi\}$ is unsatisfiable

Logical Equivalence

Two formulas φ_1 and φ_2 are logically equivalent, written $\varphi_1 \equiv \varphi_2$, if $\varphi_1 \models \varphi_2$ and $\varphi_2 \models \varphi_1$.

Note:

- $\varphi_1 \equiv \varphi_2$ if and only if every interpretation assigns the same Boolean value to φ_1 and φ_2.
- Implication and equivalence ($\Rightarrow, \Leftrightarrow$), which are syntactical entities, are intimately related to entailment and logical equivalence ($\models, \equiv$), which are semantical notions:

$$\varphi_1 \models \varphi_2 \quad \text{iff} \quad \models \varphi_1 \Rightarrow \varphi_2$$

$$\varphi_1 \equiv \varphi_2 \quad \text{iff} \quad \models \varphi_1 \Leftrightarrow \varphi_2$$

Example

$A \models (A \lor B)$ holds and $A \Rightarrow (A \lor B)$ is valid.

	A	B	$A \lor B$	$A \Rightarrow (A \lor B)$
1.	$False$	$False$	$False$	$True$
2.	$False$	$True$	$True$	$True$
3.	$\underline{True}$	$False$	$True$	$True$
4.	$\underline{True}$	$True$	$True$	$True$

Properties of Logical Connectives

$\wedge$ and $\vee$ are:

- commutative

$$\begin{aligned} \varphi_1 \wedge \varphi_2 &\equiv \varphi_2 \wedge \varphi_1 \\ \varphi_1 \vee \varphi_2 &\equiv \varphi_2 \vee \varphi_1 \end{aligned}$$

- associative

$$\begin{aligned} \varphi_1 \wedge (\varphi_2 \wedge \varphi_3) &\equiv (\varphi_1 \wedge \varphi_2) \wedge \varphi_3 \\ \varphi_1 \vee (\varphi_2 \vee \varphi_3) &\equiv (\varphi_1 \vee \varphi_2) \vee \varphi_3 \end{aligned}$$

- mutually distributive

$$\begin{aligned} \varphi_1 \wedge (\varphi_2 \vee \varphi_3) &\equiv (\varphi_1 \wedge \varphi_2) \vee (\varphi_1 \wedge \varphi_3) \\ \varphi_1 \vee (\varphi_2 \wedge \varphi_3) &\equiv (\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3) \end{aligned}$$

- related by $\neg$ (DeMorgan's Laws)

$$\begin{aligned} \neg(\varphi_1 \wedge \varphi_2) &\equiv \neg\varphi_1 \vee \neg\varphi_2 \\ \neg(\varphi_1 \vee \varphi_2) &\equiv \neg\varphi_1 \wedge \neg\varphi_2 \end{aligned}$$

Properties of Logical Connectives

$\wedge$, $\Rightarrow$, and $\Leftrightarrow$ are actually redundant:

$$
\begin{aligned}
\varphi_1 \wedge \varphi_2 &\equiv \neg(\neg\varphi_1 \vee \neg\varphi_2) \\
\varphi_1 \Rightarrow \varphi_2 &\equiv \neg\varphi_1 \vee \varphi_2 \\
\varphi_1 \Leftrightarrow \varphi_2 &\equiv (\varphi_1 \Rightarrow \varphi_2) \wedge (\varphi_2 \Rightarrow \varphi_1)
\end{aligned}
$$

We keep them all mainly for convenience.

Exercise. Use the truth tables to verify all the logical equivalences seen so far.

Computational Properties of Propositional Logic

- Satisfiability in PL is decidable (hence, so are entailment, validity, and equivalence).
- That is, there is a general algorithm that given any PL formula φ can always determine whether φ is satisfiable or not.
- However, satisfiability is NP-complete (and the rest are co-NP-complete).

Computational Properties of Propositional Logic

- Satisfiability in PL is decidable (hence, so are entailment, validity, and equivalence).
- That is, there is a general algorithm that given any PL formula φ can always determine whether φ is satisfiable or not.
- However, satisfiability is NP-complete (and the rest are co-NP-complete).

Note

- Many problems in formal verification can be reduced to checking the satisfiability of a propositional formula.
- Despite NO-completeness, many realistic instances can be checked very efficiently by state-of-the-art SAT solvers.