Lecture 4 - More Propositional Logic

Vaishnavi Sundararajan

COL703 - Logic for Computer Science

Logical consequence

- What does it mean for a valuation τ to be a model of a formula φ?
- τ makes some atomic propositions true, and also makes φ true
- A proposition φ is called a **logical consequence** of a set Γ of propositions if any valuation that is a model for Γ is also a model for φ
- Slightly overload notation to denote this also by $\Gamma \vDash \varphi$ (even though Γ can contain non-atomic formulas)
- For an empty Γ, logical consequence is nothing but validity

More on logical consequence

Theorem(s): For any finite set $\Gamma = \{\varphi_i \mid 0 \leqslant i \leqslant n\}$ of propositions and any proposition ψ, the following are true.

$$\Gamma \vDash \psi \text{ iff } \left(\bigwedge_{0 \leqslant i \leqslant n} \varphi_i \right) \supset \psi \text{ is valid}$$

$$\Gamma \vDash \psi \text{ iff } \varphi_0 \supset (\varphi_1 \supset (\ldots (\varphi_n \supset \psi) \ldots)) \text{ is valid}$$

$$\Gamma \vDash \psi \text{ iff } \left(\bigwedge_{0 \leqslant i \leqslant n} \varphi_i \right) \wedge \neg\psi \text{ is unsatisfiable}$$

Logical consequence

Theorem: $\Gamma \vDash \psi$ iff $\left(\bigwedge_{0 \leqslant i \leqslant n} \varphi_i\right) \supset \psi$ is valid

Proof:

$\Gamma \vDash \psi$ iff any τ that is a model for Γ is also a model for ψ.

(iff) For every τ, if $[\![\varphi_i]\!]_\tau = T$ for every $0 \leqslant i \leqslant n$, then $[\![\psi]\!]_\tau = T$.

(iff) For every τ, if $[\![\bigwedge_{0 \leqslant i \leqslant n} \varphi_i]\!]_\tau = T$, then $[\![\psi]\!]_\tau = T$.

(iff) For every τ, $[\![\left(\bigwedge_{0 \leqslant i \leqslant n} \varphi_i\right) \supset \psi]\!]_\tau = T$.

(iff) $\left(\bigwedge_{0 \leqslant i \leqslant n} \varphi_i\right) \supset \psi$ is valid. ∎

Exercise: Prove the other two theorems on the previous slide.

Logical equivalence

- We say that φ *logically implies* ψ iff $\varphi \supset \psi$ is valid
- We say that φ is **logically equivalent** to ψ iff φ logically implies ψ and vice versa
- We denote this by $\varphi \Leftrightarrow \psi$
- For example, $\varphi \wedge \psi \Leftrightarrow \psi \wedge \varphi$, since $\wedge$ is commutative
- Have to show that each direction of this identity is a validity
- Can write many such identities

Propositional identities

- Negation: $\neg\neg\varphi \Leftrightarrow \varphi$
- Zero: $\varphi \wedge F \Leftrightarrow F$ and $\varphi \vee T \Leftrightarrow T$
- Identity: $\varphi \wedge T \Leftrightarrow \varphi$ and $\varphi \vee F \Leftrightarrow \varphi$
- Implication: $\varphi \supset \psi \Leftrightarrow \neg\varphi \vee \psi$
- Inversion: $\neg F \Leftrightarrow T$ and $\neg T \Leftrightarrow F$
- Simplification: $\varphi \vee \neg\varphi \Leftrightarrow T$ and $\varphi \wedge \neg\varphi \Leftrightarrow F$

For $\circ \in \{\wedge, \vee\}$, the following hold:

- Commutativity: $\varphi \circ \psi \Leftrightarrow \psi \circ \varphi$
- Idempotence: $\varphi \circ \varphi \Leftrightarrow \varphi$
- Associativity: $\varphi \circ (\psi \circ \xi) \Leftrightarrow (\varphi \circ \psi) \circ \xi$
- Distributivity: $\varphi \circ (\psi * \xi) \Leftrightarrow (\varphi \circ \psi) * (\varphi \circ \xi)$ (where $*$ is the dual of $\circ$)
- De Morgan's laws: $\neg(\varphi \circ \psi) \Leftrightarrow (\neg\varphi) * (\neg\psi)$
- Absorption: $\varphi \circ (\varphi * \psi) \Leftrightarrow \varphi$

Digression: Functional completeness

- How many functions are there on a countable set of atoms?
- Can one express each such function as an expression in some logic?
- How “big” a language do I need? How many distinct operators?
- In general, infinitely many!
- Consider $\mathbb{N}$ with addition, subtraction, multiplication, division
- Can one express exponentiation with these?
- But for the set $\{T, F\}$, Boolean identities come to the rescue!

Functional completeness

- Given any Boolean operator of any arity, it is possible to define a logically equivalent operator in propositional logic
- PL is **functionally complete** if any Boolean function can be represented as an expression in PL
- We will often instead refer to the set of operators involved in the language as being functionally complete
- In fact, we do not even need $\supset$
- **Theorem**: $\{\neg, \wedge, \vee\}$ is functionally complete

$\{\neg, \wedge, \vee\}$ is functionally complete

- n-ary Boolean function f with inputs a_1 through a_n and truth value b. $m = 2^n - 1$ rows in truth table. Denote the value of a_i in row r by a_{ri}.

Row	a_1	...	a_n	b
0	F	...	F	b_0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
m	T	...	T	b_m

Fix distinct atoms $p_1, \ldots, p_n \subseteq AP$. Define:

$$\text{pmap}(r, i) = \begin{cases} p_i, & \text{if } a_{ri} = T \\ \neg p_i, & \text{if } a_{ri} = F \end{cases}$$

Equivalent expression(s):

$$\bigvee_{0 \leqslant r \leqslant m} \left\{ \bigwedge_{1 \leqslant i \leqslant n} \text{pmap}(r, i) \;\middle|\; b_r = T \right\}$$

$$\bigwedge_{0 \leqslant r \leqslant m} \left\{ \bigvee_{1 \leqslant i \leqslant n} \neg\text{pmap}(r, i) \;\middle|\; b_r = F \right\}$$

Functional completeness

- Empty disjunction is equivalent to F
- Empty conjunction is equivalent to T
- **Exercise**: Prove that $\{\land, \neg\}$ and $\{\lor, \neg\}$ are functionally complete
- **Exercise**: Prove that $\{\land, \lor\}$ is not functionally complete

Normal Forms

- It is useful to have a notion of a “general shape” for any expression
- Think of the general expression we just created, given any operator
- Disjunction over conjunctions; each conjunct an atom or its negation
- Various such “general shapes” are possible
- A **normal form** is a “general shape” such that any expression has a logical equivalent of that particular shape

Negation Normal Form

- A **literal** is an atom (positive literal) or its negation (negative literal)
- Set $\mathcal{L}$ of literals $\quad \mathcal{L} = AP \cup \{\neg p \mid p \in AP\}$
- A formula in **negation normal form (NNF)** has the grammar

$$\varphi, \psi := \ell \in \mathcal{L} \mid \varphi \wedge \psi \mid \varphi \vee \psi$$

- An expression in NNF has negations pushed to the "innermost" level
- **Theorem**: Every expression in PL is logically equivalent to one in NNF
- Proof sketch: Consider expressions over the functionally complete set $\{\wedge, \vee, \neg\}$. Remove double negations and push negations inside using de Morgan's laws wherever possible.

Conjunctive & Disjunctive Normal Forms

- An expression in **conjunctive normal form (CNF)** is of the form

$$\delta_1 \wedge \delta_2 \wedge \ldots \wedge \delta_n$$

- Each δ_i is called a **clause**
- For CNF: each δ_i itself has the shape $\ell_{i1} \vee \ell_{i2} \vee \ldots \vee \ell_{im_i}$ (each $\ell_{ij} \in \mathcal{L}$)
- An expression in **disjunctive normal form (DNF)** is of the form

$$\delta_1 \vee \delta_2 \vee \ldots \vee \delta_n$$

  where each δ_i has the shape $\ell_{i1} \wedge \ell_{i2} \wedge \ldots \wedge \ell_{im_i}$ (each $\ell_{ij} \in \mathcal{L}$)
- **Theorem**: Every expression in PL is logically equivalent to one in CNF
- **Theorem**: Every expression in PL is logically equivalent to one in DNF
- **Exercise(s)**: Prove the above two theorems

Satisfiability/Validity Again

- Checking for satisfiability requires us to find a model
- Checking for (in)validity requires us to find a falsifying valuation
- We set up logical consequence/equivalence to simplify this process
- Easier for some normal forms than for others!
- **Falsifying CNF expressions is easy**

Falsifying CNF expressions

- A CNF expression looks like $\delta_1 \wedge \delta_2 \wedge \ldots \wedge \delta_n$
- Each δ_i of the form $\ell_{i1} \vee \ell_{i2} \vee \ldots \vee \ell_{im_i}$
- What does it mean for a CNF expression to be made false under some valuation?
- At least one clause must be made false
- Suppose $p \in AP$ and $\neg p$ both occur as literals in a clause δ_i
- Can δ_i be made false under any valuation?
- **Theorem**: $\delta_1 \wedge \delta_2 \wedge \ldots \wedge \delta_n$ can be falsified iff there is some δ_i which does not contain both a propositional atom and its negation as literals.

Satisfiability/Validity Again

- Checking for satisfiability requires us to find a model
- Checking for (in)validity requires us to find a falsifying valuation
- We set up logical consequence/equivalence to simplify this process
- Easier for some normal forms than for others!
- **Falsifying CNF expressions is easy**
- **Satisfying DNF expressions is easy**

Satisfying DNF expressions

- A DNF expression looks like $\delta_1 \lor \delta_2 \lor \ldots \lor \delta_n$
- Each δ_i of the form $\ell_{i1} \land \ell_{i2} \land \ldots \land \ell_{im_i}$
- What does it mean for a DNF expression to be made true under some valuation?
- At least one clause must be true
- **Exercise**: State and prove the corresponding theorem (dual of CNF)

Validity

- Easy to check falsification of CNF expressions
- Recall theorems about logical consequence from earlier
- First two reduce it to checking validity of an “implies” expression
- Converting that to CNF is complicated
- Use the third theorem.

$$\{\varphi_0, \ldots, \varphi_n\} \vDash \psi \text{ iff } \left(\bigwedge_{0 \leqslant i \leqslant n} \varphi_i \right) \wedge \neg\psi \text{ is unsatisfiable}$$

- Convert RHS expression to CNF as follows:
  - Convert each φ_i and $\neg\psi$ to CNF
  - Throw away unnecessary duplicates and put back together using $\wedge$s

CNF: Literals and clauses

- A CNF expression φ looks like $\delta_1 \wedge \delta_2 \wedge \ldots \wedge \delta_n$
- Think of each δ_i as a set of literals $\{\ell_{i1}, \ell_{i2}, \ldots, \ell_{im_i}\}$
- Think of φ as a set of clauses, i.e. a set of sets of literals
- The **empty set of clauses** is equivalent to T
  - $\left(\bigwedge_{1 \leqslant i \leqslant n} \delta_i\right)$ is equivalent to $\left(\bigwedge_{1 \leqslant i \leqslant n} \delta_i\right) \wedge T$ (by Identity)
  - So if $n = 0$, the conjunction is just T
- Similarly, the **empty set of literals** is equivalent to F
- If δ_i contains $p \in AP$ and $\neg p$, it is equivalent to T
- If $\delta \subseteq \delta'$ for δ and δ', then $\{\delta, \delta'\}$ is equivalent to $\{\delta\}$ (by Absorption)
- $\emptyset \subseteq \delta$ for any clause δ, so any $\{\delta_1, \ldots, \delta_n, \emptyset\}$ is equivalent to $\{\emptyset\}$

CNF: Deleting "unnecessary" clauses

- We would like to show that $\{\varphi_0, \ldots, \varphi_n\} \vDash \psi$
- Needs us to show that $(\bigwedge_{0 \leqslant i \leqslant n} \varphi_i) \wedge \neg\psi$ is unsatisfiable
- Convert $(\bigwedge_{0 \leqslant i \leqslant n} \varphi_i) \wedge \neg\psi$ into CNF
- This yields a set of clauses
- Systematically delete "unnecessary" clauses from this set of clauses
- If we are left with the empty clause at the end, the expression is unsatisfiable; therefore ψ is a logical consequence of $\{\varphi_0, \ldots, \varphi_n\}$