Neuro-Symbolic Artificial Intelligence
Chapter 3
Propositional Logic

Nils Holzenberger

March 5, 2024

Outline

Outline

What is logic good for?

- Represent logic and knowledge
- Represent argumentation
- Mechanize reasoning

What is logic good for?

- Computer science
- Automated theorem proving
- Proofs of programs
- AI and reasoning
  - Argumentation
  - High-level NLP
- Electronics
- Database management
- Knowledge representation & semantic Web
- Cognitive science
  - Human cognition
  - Proof – automated proof
- Contradiction
  - Anomaly detection
  - Explanation (XAI)
- Relevance
  - No continuity
  - Reason vs guess
- Basic in many curriculums

History

⚠ Logic, reasoning and argumentation are universal human abilities. In this lecture, *logic* is a formal system, which can be used to *model* human reasoning and argumentation.

- Ancient greeks
  - Stoics
  - Aristotle: syllogism and argumentation
- Medieval logic
  - William of Ockham (1288-1348)
  - de Morgan's laws
  - Ternary logic
- Traditional logic
  - Port Royal's logic
  - Antoine Arnauld & Pierre Nicole (1662)
  - Logic of propositions
- Modern Logic
  - Descartes, Leibniz
  - George Boole (1848)
  - Gottlob Frege: *Begriffschrift* (1879), quantification
  - Charles Peirce
  - Giuseppe Peano: logical axiomatization of arithmetics
  - Bertrand Russell & Alfred N. Whitehead (1925): logical axiomatization of mathematics

Outline

Symbols

Logic is about *syntax* and *semantics*

Syntax: how to *manipulate* symbols

Semantics: what *meaning* the symbols have

The use of these words is specific to logic!

Syntax

- Alphabet
  - *Propositional symbols*: p in $a \vee p$
  - *Constants*: $\top$ and $\bot$
  - *Connectors*: $\neg$ (1-place), $\wedge$ (2-place), $\vee$ (2-place)...
- Atomic formula: constants and connectors
- Propositional formula
  - Atomic formula
  - If F is a formula, then $(\neg F)$ is a formula
  - If $\bullet$ is a connector, and A and B are formulas, then $(A \bullet B)$ is a formula

The sets of atomic formulas and propositional formulas are the smallest sets having these properties.

⚠ None of those things above may be said to be "true" or "false". That pertains to the semantics.

Truth tables

A	B	A **and** B	A **or** B	A **implies** B
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

Each of these lines is a *valuation* of the logical propositions. It's a mapping of the symbols to "true" or "false".

Truth tables

A	B	A **and** B	A **or** B	A **implies** B
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

Each of these lines is a *valuation* of the logical propositions. It's a mapping of the symbols to "true" or "false".

How many 2-place connectors can I invent?

Truth tables

A	B	A **and** B	A **or** B	A **implies** B
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

Each of these lines is a *valuation* of the logical propositions. It's a mapping of the symbols to "true" or "false".

How many 2-place connectors can I invent?

6 of the 2-place connectors are trivial, which ones?

Truth tables

A	B	A **and** B	A **or** B	A **implies** B
T	T	T	T	T
T	F	F	T	F
F	T	F	T	T
F	F	F	F	T

Each of these lines is a *valuation* of the logical propositions. It's a mapping of the symbols to "true" or "false".

How many 2-place connectors can I invent?

6 of the 2-place connectors are trivial, which ones?

What about 3-place connectors?

Valuation

This is where *semantics* come into play. A *valuation* assigns "true" or "false" to propositional symbols and to propositional formulas.

$v : F \rightarrow \{\text{True}, \text{False}\}$

Valuation

This is where *semantics* come into play. A *valuation* assigns "true" or "false" to propositional symbols and to propositional formulas.
$v : F \to \{\text{True}, \text{False}\}$

A valuation v must be consistent:

$$v(\top) = \text{True}$$

$$v(\bot) = \text{False}$$

$$v(\neg F) = \text{Not } v(F)$$

$$v((A \bullet B)) = v(A) \blacksquare v(B)$$

Syntax and semantics look very similar, so we use different symbols to avoid confusion.

Valuation

This is where *semantics* come into play. A *valuation* assigns "true" or "false" to propositional symbols and to propositional formulas.

$v : F \to \{\text{True}, \text{False}\}$

A valuation v must be consistent:

$v(\top) = \text{True}$

$v(\bot) = \text{False}$

$v(\neg F) = \text{Not } v(F)$

$v((A \bullet B)) = v(A) \blacksquare v(B)$

Syntax and semantics look very similar, so we use different symbols to avoid confusion.

Syntactic connective $\bullet$	Semantic connective $\blacksquare$
$\neg$	Not
$\wedge$	And
$\vee$	Or
$\supset$	$\Rightarrow$
$\subset$	$\Leftarrow$
$\equiv$	$\Leftrightarrow$
$\uparrow$	Nand
$\downarrow$	Nor
$\not\supset$	$\not\Rightarrow$
$\not\subset$	$\not\Leftarrow$

Tautologies and satisfiability

A propositional formula X is a *tautology* if for any valuation v, $v(X) = \text{True}$

A tautology evaluates to True regarless of what its components evaluate to.

Tautologies and satisfiability

A propositional formula X is a *tautology* if for any valuation v, $v(X) = \mathrm{True}$

A tautology evaluates to True regarless of what its components evaluate to.

A set S of propositional formulas is *satisfiable* if some valuation v_0 maps every member of S to True: $\forall X \in S,\ v_0(X) = \mathrm{True}$

SAT problem: given S, find v_0.

Tautologies and satisfiability

A propositional formula X is a *tautology* if for any valuation v, $v(X) = \text{True}$

A tautology evaluates to True regarless of what its components evaluate to.

A set S of propositional formulas is *satisfiable* if some valuation v_0 maps every member of S to True: $\forall X \in S,\ v_0(X) = \text{True}$

SAT problem: given S, find v_0.

X is a tautology iff $(\neg X)$ is not satisfiable.

Tautologies and satisfiability

A propositional formula X is a *tautology* if for any valuation v, $v(X) = \text{True}$

A tautology evaluates to True regarless of what its components evaluate to.

A set S of propositional formulas is *satisfiable* if some valuation v_0 maps every member of S to True: $\forall X \in S,\ v_0(X) = \text{True}$

SAT problem: given S, find v_0.

X is a tautology iff $(\neg X)$ is not satisfiable.

Why do we need this? We will see later that proving a theorem is equivalent to proving a tautology. (Specifically, proving $S \vdash X$ is like proving that $(\neg(S \cup \{\neg X\}))$ is a tautology.)

Tautologies

- Show that X is a tautology iff $X \equiv \mathrm{T}$ is a tautology
- Show that X is a tautology iff $\mathrm{T} \supset X$ is a tautology
- Show that $(\neg(X \wedge Y)) \equiv (\neg X \vee \neg Y)$ is a tautology
- Show that $(\neg(X \vee Y)) \equiv (\neg X \wedge \neg Y)$ is a tautology
- Show that $(P \wedge (Q \vee R)) \equiv ((P \wedge Q) \vee (P \wedge R))$ is a tautology
- Show that $(P \vee (Q \wedge R)) \equiv ((P \vee Q) \wedge (P \vee R))$ is a tautology

Tautologies

- Show that X is a tautology iff $X \equiv \mathrm{T}$ is a tautology
- Show that X is a tautology iff $\mathrm{T} \supset X$ is a tautology
- Show that $(\neg(X \wedge Y)) \equiv (\neg X \vee \neg Y)$ is a tautology
- Show that $(\neg(X \vee Y)) \equiv (\neg X \wedge \neg Y)$ is a tautology
- Show that $(P \wedge (Q \vee R)) \equiv ((P \wedge Q) \vee (P \wedge R))$ is a tautology
- Show that $(P \vee (Q \wedge R)) \equiv ((P \vee Q) \wedge (P \vee R))$ is a tautology
- This is cumbersome because we need to build truth tables. In a minute we'll see how to do it without tables.

X is a tautology iff $\mathrm{T} \supset X$ is a tautology

Let's show that X is a tautology iff $\mathrm{T} \supset X$ is a tautology.

First, notice that, for any valuation v:
$v(\mathrm{T} \supset X) = v(\mathrm{T}) \Rightarrow v(X) = \mathrm{True} \Rightarrow v(X)$
Using a truth table, you can show that $\mathrm{True} \Rightarrow v(X)$ is equal to $v(X)$.
So for any valuation v: $v(\mathrm{T} \supset X) = v(x)$.

Now let's show that if X is a tautology then $\mathrm{T} \supset X$ is a tautology:
Let v be a valuation. Then $v(\mathrm{T} \supset X) = v(X) = \mathrm{True}$, the last equality being a consequence of X being a tautology.

Now let's show that if X is not a tautology then $\mathrm{T} \supset X$ is not a tautology:
X is not a tautology so there exists a valuation u such that $u(X) = \mathrm{False}$.
Consequently $u(\mathrm{T} \supset X) = u(X) = \mathrm{False}$ so $\mathrm{T} \supset X$ is not a tautology.

Logical consequence

Logical consequence: $S \models X$

If a valuation assigns True to all elements in S, then it will assign True to X.

Logical consequence

Logical consequence: $S \models X$

If a valuation assigns True to all elements in S, then it will assign True to X.

$\models X$ means that X is a tautology.

Logical consequence

Logical consequence: $S \models X$

If a valuation assigns True to all elements in S, then it will assign True to X.

$\models X$ means that X is a tautology.

This is closer to our use of logic: we're only interested in the conclusions X that we can derive from assumptions S that we know to be true.

Logical consequence

- Show that if $S \models X$ then $S \cup \{\neg X\}$ is not satisfiable. *lab session*
- Show the reciprocal.
- *Ex falso quodlibet sequitur*: Let S be a set of formulas, and A a formula such that $A \in S$ and $(\neg A) \in S$. Show that $\forall X, S \models X$
- Conversely, if $\forall X, S \models X$, show that S is not satisfiable.
- *Monotony*: show that $S \models X$ implies $S \cup \{A\} \models X$ *lab session*
- *Deduction*: show that $S \cup \{X\} \models Y$ iff $S \models (X \supset Y)$ *lab session*

Outline

Problem

The goal of automated theorem proving is to show things like $S \models X$ where X is a theorem and S are a set of assumptions.

Problem

The goal of automated theorem proving is to show things like $S \models X$ where X is a theorem and S are a set of assumptions.

We saw that this is equivalent to proving that $(\neg(S \cup \{\neg X\}))$ is a tautology.

So our goal is to prove tautologies.

Problem

The goal of automated theorem proving is to show things like $S \models X$ where X is a theorem and S are a set of assumptions.

We saw that this is equivalent to proving that $(\neg(S \cup \{\neg X\}))$ is a tautology.

So our goal is to prove tautologies.

Problem: we'll need to build gigantic truth tables to check all possible valuations.

Problem

The goal of automated theorem proving is to show things like $S \models X$ where X is a theorem and S are a set of assumptions.

We saw that this is equivalent to proving that $(\neg(S \cup \{\neg X\}))$ is a tautology.

So our goal is to prove tautologies.

Problem: we'll need to build gigantic truth tables to check all possible valuations.

Solution: valuations map syntax to semantics. Instead, stay in the space of syntax. This means modifying the syntax of the formula without modifying its semantics, until computing the valuation becomes trivial.

Replacement procedure

Define a procedure P such that if $(X \equiv Y)$ is a tautology, then $P(X) \equiv P(Y)$ is a tautology.

This is a syntactic rewriting that preserves the property of being a tautology.

Normal form for negation

- A formula is in normal form for negation if the negation symbol $\neg$ only occurs in front of symbols
  - $(\neg X \vee Y)$ is in normal form for negation
  - $(\neg(X \wedge Y))$ is not
- Use the following tautologies to rewrite negations:
  - $(\neg(\neg X)) \equiv X$
  - $(\neg(X \vee Y)) \equiv ((\neg X) \wedge (\neg Y))$
  - $(\neg(X \wedge Y)) \equiv ((\neg X) \vee (\neg Y))$

Generalized disjunction/conjunction

Define two new operators, that don't belong to the propositional language:

Generalized disjunction/conjunction

Define two new operators, that don't belong to the propositional language:

$[X_1, X_2, ..., X_n]$ is the *generalized disjunction* of $X_1, X_2, ..., X_n$

For any valuation v, $v([X_1, X_2, ..., X_n]) = \text{False}$ iff $\forall i \in [1, n]$, $v(X_i) = \text{False}$

$v([]) = \text{False}$ "X_1 or X_2 or ... or X_n"

Generalized disjunction/conjunction

Define two new operators, that don't belong to the propositional language:

$[X_1, X_2, ..., X_n]$ is the *generalized disjunction* of $X_1, X_2, ..., X_n$

For any valuation v, $v([X_1, X_2, ..., X_n]) = \text{False}$ iff $\forall i \in [1, n]$, $v(X_i) = \text{False}$

$v([]) = \text{False}$ "X_1 or X_2 or … or X_n"

$< X_1, X_2, ..., X_n >$ is the *generalized conjunction* of $X_1, X_2, ..., X_n$

For any valuation v, $v(< X_1, X_2, ..., X_n >) = \text{True}$ iff $\forall i \in [1, n]$, $v(X_i) = \text{True}$

$v(<>) = \text{True}$ "X_1 and X_2 and … and X_n"

Conjunctive normal form

Let F be a propositional formula. Its *conjunctive normal form* is a rewriting of F as

$$< C_1, C_2, ..., C_i, ..., C_n >$$

where each C_i is of the form $[X_1, X_2, ..., X_{n_i}]$. C_i is a *clause*.

The *disjunctive normal form* is the same thing *mutandem mutandis*.

Conjunctive normal form

How do we get the conjunctive normal form? Use the following tautologies:

- $\wedge$
  - $\neg(X \wedge Y) \equiv \neg X \vee \neg Y$
- $\supset$
  - $X \supset Y \equiv \neg X \vee Y$
  - $\neg X \supset Y \equiv X \wedge \neg Y$
- $\subset$
  - $X \subset Y \equiv X \vee \neg Y$
  - $\neg X \subset Y \equiv \neg X \wedge Y$
- $\uparrow$
  - $X \uparrow Y \equiv \neg X \vee \neg Y$
  - $\neg X \uparrow Y \equiv X \wedge Y$
- $\vee$
  - $\neg(X \vee Y) \equiv \neg X \wedge \neg Y$
- $\downarrow$
  - $X \downarrow Y \equiv \neg X \wedge \neg Y$
  - $\neg X \downarrow Y \equiv X \vee Y$
- $\not\supset$
  - $X \not\supset Y \equiv X \wedge \neg Y$
  - $\neg X \not\supset Y \equiv \neg X \vee Y$
- $\not\subset$
  - $X \not\subset Y \equiv \neg X \wedge Y$
  - $\neg X \not\subset Y \equiv X \vee \neg Y$

Rewriting algorithm

Rewriting a disjunction:

$$\text{Replace} < ...[...P...]... > \text{ with } < ...[...A, B...]... >$$

Rewriting a conjunction:

$$\text{Replace} < ...[...P...]... > \text{ with } < ...[...A...], [...B...]... >$$

Rewriting a negation:

$$\text{Replace} < ...[...\neg(\neg P)...]... > \text{ with } < ...[...P...]... >$$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y))$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y))$

- $< [((A \supset B) \supset (A \supset C))] >$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y))$

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y))$

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$
- $< [\neg(A \supset B), (A \supset C)] >$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$
- $< [\neg(A \supset B), (A \supset C)] >$
- $< [(A \wedge \neg B), (A \supset C)] >$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y))$

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$
- $< [\neg(A \supset B), (A \supset C)] >$
- $< [(A \wedge \neg B), (A \supset C)] >$
- $< [A, (A \supset C)], [\neg B, (A \supset C)] >$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y))$

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$
- $< [\neg(A \supset B), (A \supset C)] >$
- $< [(A \wedge \neg B), (A \supset C)] >$
- $< [A, (A \supset C)], [\neg B, (A \supset C)] >$
- $< [A, (\neg A \vee C)], [\neg B, (A \supset C)] >$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y))$

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$
- $< [\neg(A \supset B), (A \supset C)] >$
- $< [(A \wedge \neg B), (A \supset C)] >$
- $< [A, (A \supset C)], [\neg B, (A \supset C)] >$
- $< [A, (\neg A \vee C)], [\neg B, (A \supset C)] >$
- $< [A, \neg A, C], [\neg B, (A \supset C)] >$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$
- $< [\neg(A \supset B), (A \supset C)] >$
- $< [(A \wedge \neg B), (A \supset C)] >$
- $< [A, (A \supset C)], [\neg B, (A \supset C)] >$
- $< [A, (\neg A \vee C)], [\neg B, (A \supset C)] >$
- $< [A, \neg A, C], [\neg B, (A \supset C)] >$
- $< [A, \neg A, C], [\neg B, \neg A, C] >$

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$
- $< [\neg(A \supset B), (A \supset C)] >$
- $< [(A \wedge \neg B), (A \supset C)] >$
- $< [A, (A \supset C)], [\neg B, (A \supset C)] >$
- $< [A, (\neg A \vee C)], [\neg B, (A \supset C)] >$
- $< [A, \neg A, C], [\neg B, (A \supset C)] >$
- $< [A, \neg A, C], [\neg B, \neg A, C] >$

Note that the first clause will always evaluate to True. So we can rewrite the original formula as

Exercise

Conjunctive normal form of $((A \supset B) \supset (A \supset C))$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y))$

- $< [((A \supset B) \supset (A \supset C))] >$
- $< [\neg(A \supset B) \vee (A \supset C)] >$
- $< [\neg(A \supset B), (A \supset C)] >$
- $< [(A \wedge \neg B), (A \supset C)] >$
- $< [A, (A \supset C)], [\neg B, (A \supset C)] >$
- $< [A, (\neg A \vee C)], [\neg B, (A \supset C)] >$
- $< [A, \neg A, C], [\neg B, (A \supset C)] >$
- $< [A, \neg A, C], [\neg B, \neg A, C] >$

Note that the first clause will always evaluate to True. So we can rewrite the original formula as $< [\neg B, \neg A, C] >$ without changing its truth value. That's a purely *syntactic* rewriting.

Proof by resolution

- A *sequence* is a conjunction of *lines*

Proof by resolution

- A *sequence* is a conjunction of *lines*
- Each *line* is a generalized disjunction (a clause)

Proof by resolution

- A *sequence* is a conjunction of *lines*
- Each *line* is a generalized disjunction (a clause)
- *Growth* of the sequence:
  - if a clause reads as $[...(\beta_1 \vee \beta_2)...]$, insert a new line: $[...\beta_1, \beta_2...]$
  - if a clause reads as $[...(\alpha_1 \wedge \alpha_2)...]$, insert two new lines: $[...\alpha_1...]$ and $[...\alpha_2...]$
  - when adding new lines, replace $\neg\neg X$ by X, $\neg\mathrm{T}$ by $\perp$ and $\neg\perp$ by T

Proof by resolution

- A *sequence* is a conjunction of *lines*
- Each *line* is a generalized disjunction (a clause)
- *Growth* of the sequence:
  - if a clause reads as $[...(\beta_1 \vee \beta_2)...]$, insert a new line: $[...\beta_1, \beta_2...]$
  - if a clause reads as $[...(\alpha_1 \wedge \alpha_2)...]$, insert two new lines: $[...\alpha_1...]$ and $[...\alpha_2...]$
  - when adding new lines, replace $\neg\neg X$ by X, $\neg \mathrm{T}$ by $\perp$ and $\neg \perp$ by T
- *Resolution*: from lines $[A, X, B]$ and $[C, \neg X, D]$ create the line $[A, B, C, D]$, i.e. concatenate the lines leaving aside all occurrences of X and of $\neg X$

Proof by resolution

- A *sequence* is a conjunction of *lines*
- Each *line* is a generalized disjunction (a clause)
- *Growth* of the sequence:
    - if a clause reads as $[...(\beta_1 \vee \beta_2)...]$, insert a new line: $[...\beta_1, \beta_2...]$
    - if a clause reads as $[...(\alpha_1 \wedge \alpha_2)...]$, insert two new lines: $[...\alpha_1...]$ and $[...\alpha_2...]$
    - when adding new lines, replace $\neg\neg X$ by X, $\neg\top$ by $\bot$ and $\neg\bot$ by $\top$
- *Resolution*: from lines $[A, X, B]$ and $[C, \neg X, D]$ create the line $[A, B, C, D]$, i.e. concatenate the lines leaving aside all occurrences of X and of $\neg X$
- a proof of X by resolution is a sequence starting with the $[\neg X]$ line (goal) and ending with an empty clause [].
- X is a tautology if and only if X has a proof by resolution.

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

$[\neg((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)]$

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

$[\neg((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)]$

$[((A \supset B) \wedge (B \supset C))]$
$[\neg C \wedge A]$

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

$[\neg((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)]$

$[((A \supset B) \wedge (B \supset C))]$
$[\neg C \wedge A]$

$[(A \supset B)]$
$[(B \supset C)]$
$[\neg C \wedge A]$

Example

Prove $((A \supset B) \land (B \supset C)) \supset \neg(\neg C \land A)$
(knowing that $(X \supset Y) \equiv (\neg X \lor Y)$ and $(\neg(X \supset Y)) \equiv (X \land \neg Y))$

$[\neg((A \supset B) \land (B \supset C)) \supset \neg(\neg C \land A)]$

$[((A \supset B) \land (B \supset C))]$
$[\neg C \land A]$

$[(A \supset B)]$
$[(B \supset C)]$
$[\neg C \land A]$

$[\neg A, B]$
$[(B \supset C)]$
$[\neg C \land A]$

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

$[\neg((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)]$

$[((A \supset B) \wedge (B \supset C))]$
$[\neg C \wedge A]$

$[(A \supset B)]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C \wedge A]$

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

$[\neg((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)]$

$[((A \supset B) \wedge (B \supset C))]$
$[\neg C \wedge A]$

$[(A \supset B)]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C]$
$[A]$

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

$[\neg((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)]$

$[((A \supset B) \wedge (B \supset C))]$
$[\neg C \wedge A]$

$[(A \supset B)]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C]$
$[A]$

$[\neg B, C]$
$[\neg C]$
$[B]$

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

$[\neg((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)]$

$[((A \supset B) \wedge (B \supset C))]$
$[\neg C \wedge A]$

$[(A \supset B)]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C]$
$[A]$

$[\neg B, C]$
$[\neg C]$
$[B]$

$[\neg C]$
$[C]$

Example

Prove $((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)$
(knowing that $(X \supset Y) \equiv (\neg X \vee Y)$ and $(\neg(X \supset Y)) \equiv (X \wedge \neg Y)$)

$[\neg((A \supset B) \wedge (B \supset C)) \supset \neg(\neg C \wedge A)]$

$[((A \supset B) \wedge (B \supset C))]$
$[\neg C \wedge A]$

$[(A \supset B)]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[(B \supset C)]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C \wedge A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C]$
$[A]$

$[\neg B, C]$
$[\neg C]$
$[B]$

$[\neg C]$
$[C]$

$[]$

Example

Prove $((A \supset B) \land (B \supset C)) \supset \neg(\neg C \land A)$
(knowing that $(X \supset Y) \equiv (\neg X \lor Y)$ and $(\neg(X \supset Y)) \equiv (X \land \neg Y)$)

$[\neg((A \supset B) \land (B \supset C)) \supset \neg(\neg C \land A)]$

$[((A \supset B) \land (B \supset C))]$
$[\neg C \land A]$

$[(A \supset B)]$
$[(B \supset C)]$
$[\neg C \land A]$

$[\neg A, B]$
$[(B \supset C)]$
$[\neg C \land A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C \land A]$

$[\neg A, B]$
$[\neg B, C]$
$[\neg C]$
$[A]$

$[\neg B, C]$
$[\neg C]$
$[B]$

$[\neg C]$
$[C]$

$[]$

Done. We didn't need a truth table!

Example in Prolog

`A :- B` turns into $[A, \neg B]$

```
parent(marge, bart).

parent(clancy, marge).

grandparent(X,Y) :-
        parent(X,Z),
        parent(Z,Y).

?- grandparent(clancy,bart)
```

[parent(marge, marge)]

[parent(clancy, marge)]

[grandparent(X,Y), ¬ parent(X,Z), ¬ parent(Z,Y)]

[¬ grandparent(clancy,bart)]

Outline

$\vdash$ vs $\models$

Logical consequence: $S \models X$

If a valuation assigns True to all elements in S, then it will assign True to X.

$\models$ is a *semantic deduction*, typically involving truth tables.

$\vdash$ is a *syntactic deduction*, typically involving proof by resolution.

Theorem proving

- An axiomatic system is a proof system. For example the Hilbert system:
  - $(X \supset (Y \supset X))$
  - $(X \supset (Y \supset Z)) \supset ((X \supset Y) \supset (X \supset Z))$
  - $(\bot \supset X)$
  - $(X \supset \mathrm{T})$
  - $(\neg\neg X \supset X)$
  - $(X \supset (\neg X \supset Y))$
  - $((A \wedge B) \supset A)$
  - $((A \wedge B) \supset B)$
  - $((A \supset X) \supset ((B \supset X) \supset ((A \vee B) \supset X)))$
  - inference rule (modus ponens) : $\frac{X \quad (X \supset Y)}{Y}$
- This can be used to produce new theorems, through *forward chaining*
- In contrast, proof by resolution is *backward chaining*

Soundness

Axiomatic systems define $\vdash$ and $\models$. An axiomatic system is *sound* if:

Let F be a propositional formula and S a set of propositional formulas.

If there is a sequence that derives from $S \cup \{\neg F\}$ and that contains the empty clause, then $S \models F$.

In other words if $S \vdash F$ then $S \models F$.

Completeness

Axiomatic systems define $\vdash$ and $\models$. An axiomatic system is *complete* if:

Let F be a propositional formula and S a set of propositional formulas.

If $S \models F$, then there is a sequence that derives from $S \cup \{\neg F\}$ and that contains the empty clause.

Completeness is the converse of soundness: if $S \models F$ then $S \vdash F$.

Gödel's PhD thesis

Resolution is sound and complete for first-order logic